ERSITAT DE
CELONA

2021

PhD in Economics | Jessica Estefania Pesantez Narvaez

UNIVERSITAT DE
BARCELONA

PhD in Economics

Risk Analytics in Econometrics

Jessica Estefania Pesantez Narvaez

LIBERTAS PERFVNDET
OMNIA LVCE

UNIVE
BARC

PhD in Economics

Thesis title:

Risk Analytics in Econometrics

PhD student:

Jessica Estefania Pesantez Narvaez

Advisors:

Montserrat Guillen

Manuela Alcañiz

Date:

March 2021

UNIVERSITAT DE BARCELONA

Dedication

To my parents
Joyce and Freddy

Acknowledgements

Undertaking this PhD has been one of the most important life-changing experiences and it would not have been possible to do without the support and guidance that I received from many people.

I would like to express my appreciation to Prof. Montserrat Guillén, the best supervisor worldwide. This journey together actually started when I was in the undergrad, and then you were assigned as my tutor, then you became my master thesis supervisor, and now my PhD supervisor. The technical and emotional support I have received from you during all these years is priceless. Thanks for all your patience, guidance, constant feedback, time, valuable pieces of advice, and encouragement to pursue my thesis. Thank you so much for having trusted me and let me work along with you. You are definitely one of my best scientific role models, and I think that every time we worked together until late at night, I got really impressed by your brilliant mind and got more inspired to bring out the best of me in this thesis. There is no single person that you could have done this better than you.

I also wish to show my gratitude to Prof. Manuela Alcañiz. Your words of encouragement always accompanied my days in the faculty, thank you very much for the goodwill and work to run projects with Montse and me. I would also like to thank special members of the RiskCenter, Professors: Luis Ortiz, Mercedes Ayuso, Catalina Bolancé, Helena Chuliá, Miguel Santolino, Ana María Perez, Ramón Alemany and David Moriña. It has been a pleasure to have shared some moment with you. Your pieces of advices, support, and enthusiasm really turned my days happier and more engaged to my research.

The acknowledgments would not be complete without my research colleagues. Viviana Z., Liliana C., Giorgios T., Akin C., Michel L., Ivan H., Bong-Ha S., Marianna M., Florencia S. and my super former prof. Ester M. We have shared a lot of amazing experiences inside or outside the university. Thanks for every moment, visit, word of motivation, piece of advice or laugh. My days got more colorful with

your company and friendship.

I would like to thank my wonderful friends who gave their friendship and support, and made me feel at home despite the distance: Alexandra C., Maria R., Oscar J., Karina B., Edison C., Diana G., and Melissa G.

The assistance provided by Jordi Roca and Eduard Eneriz is greatly appreciated. You have always demonstrated a lot of commitment, goodwill to solve the student's issues and kindness. Such management is impressive.

The work would not materialize without the financial support of Spanish Ministry of Economy, FEDER grant ECO2016-76203-C2-2-P.

Last but not least, the most important and special acknowledgements go to my family. I will start with my parents, my biggest drivers, the ones who have been always by my side giving me all their love, unconditional support in my best but also in worst days. Their encouragement in every step I made have been crucial in my achievements. I have no words to describe how grateful I am with you. I feel so lucky and blessed to have you in my life, and this PhD thesis is for you. Another super special thanks is for my grandparents, you have been always giving me heartening, cheering, inspirational words and unconditional support during all my life. This has been also my strength and motivation to go ahead. Another special thanks goes to my uncles and aunts who have always been aware of me and express their afecction in every form.

This accomplishment does not just belong to me, I wouldn't be up here if it weren't for my *family*.

Contents

List of Figures

List of Tables

Chapter 1: Introduction

The concept of risk has been a fundamental notion during all human history, however, risk assessment as science has been evolving 3-4 decades ago (Aven, 2016). The overarching models review the presence of extreme events in specific situations and perform generic risk scenarios to evaluate impacts. Extreme observations or events are found in the tails of heavy-tailed distributions, and deviate from the mean by a certain number of standard deviations. As a consequence, their predictability is more difficult than the behaviour around the mean.

Extremal values turn into event responses when a latent random variable is trigged after the probability of event's occurrence is high enough, giving as a result a binary response composed by events (usually coded as one) and non-events (usually coded as zero). This thesis considers two important approaches of extremal events: rare events and class-imbalanced data. The latter group of observations correspond to situations where the difference of proportion between events and non-events is less reported than rare events. In other words, the degree of imbalance is more extreme in rare events than in class imbalanced data, so that the number of ones is hundreds to thousands smaller than the number of zeros such as King and Zeng (2001) defined them.

The first postulations of extremes events were linked to subjectivity and chance due to the so-called inability to discover all previous and influential factors that affect extreme events (Clemen and Winkler, 1999). Recently, risk analysis has evolved by controlling extremal values from a basic estimation of probabilities to complex risk simulations and modelling.

Traditionally, probabilistic modelling has been the quantitative modelling approach used by econometricians to predict binary rare event phenomena, also known as risk probabilistic analysis. And, it has been extensively examined and used (Clemen and Winkler, 1999; Bedford et al., 2001; Buizza, 2008; Hansson and Aven, 2014; Mohsin et al., 2017). Probabilistic models incorporate a set of covariates and

probability distributions into models to forecast a set of two possible outcomes of the rare event, so that they predict a phenomenon based on some knowledge of preceding conditions. However, the prediction of rare events remains difficult since their probability of occurrence is quite low.

Nowadays, statistical learning plays a new role in the field of predictive modelling (Ahmed et al., 2017; Zhu et al., 2018; Henriques et al., 2020; Guikema, 2020). A key concept underlying this approach is representing input data through advanced analytics techniques, and then generalizing the learnt patterns to predict future events. This flexibility allows complex or unstructured data sets, obtained for example by web scrapping and device collection, to be better adjusted than if they would have been done by a probabilistic model.

In the framework of risk analysis, rare events can be perceived as potential hazards or unusual events that might bring together disruptive effects or relevant consequences (Aven, 2018). So statistical learning can improve the anticipation of those rare events, and therefore, the estimation of the likelihood that they will materialize. As a result, statistical learning-based techniques known as advanced analytics are steadily demanded for this purpose in order to derive precise conclusions and save time, money, and reputation.

In fact, new advances have witnessed the combination of risk analysis with intensive computing techniques, novel types of big data, and advanced analytics in an upcoming trend called "Risk Analytics". The aforementioned trend would be able to help answer more accurately new research questions involving rare events that evidently appear quite often in econometric modelling. For instance, the occurrence of natural disasters, real state bubbles, school dropout, workplace accidents, among others.

Unlike probabilistic modelling, risk analytics are not self-interpretable and, in many cases, considered black boxes. For instance, econometric models deliver parameter estimates which are the changes in the dependent variable associated with one-unit change of the covariate while other predictors are being held constant. Nevertheless, risk analytics deliver as default a prediction value for each observation only, so analysts have to use available additional tools of balanced data phenomena to understand the rare event phenomena.

My thesis adds to the resolution of the following two research problems: i) How econometricians can improve the comprehension of environments with the occur-

rence of rare phenomena and imbalanced data? And also, ii) How econometricians improve the rare events and imbalanced data prediction accuracy? Currently, econometricians have some methodological limitations with probabilistic models that impede to capture more complex realities. With the realization of this dissertation, I aim to enhance the robustness of econometric modelling with rare events and class-imbalanced data, as well as the understanding of their causality in order to derive more accurate conclusions.

This PhD dissertation considers within all its chapters a supervised statistical learning setting, where X_{ip} is the data matrix where i corresponds to the observations (or instances) and p corresponds to the independent variables (attributes or features), with $i = 1, ..., n$ and $p = 1, ..., P$. There are n observations and P independent variables. And Y_i is a binary response variable for observation i, that is imbalanced or rare event. It is also known as dependent, endogenous or response variable. Y_i will be used to denote indistintively the aformentioned variable or the observed value of the dependent variable. We always pursue to predict Y_i taking into consideration the covariates X_p.

I provide a unified framework of how the aforementioned research problems can be solved through three methodological parts. The **first section** has a probabilistic modelling approach and consists of Chapters 2 and 3 which focus on improvements in prediction accuracy of a econometric model for binary response. The **second section** presents Chapter 4 where a list of trials of how some famous boosting-based algorithms and classical models might be combined for optimum results. Finally, the **third section** has a risk analytics approach and contains Chapters 5 and 6 with a focus on boosting-based algorithms for binary response.

This thesis is organized as follows. **Chapter 2** is entitled as "Improving Prediction Accuracy in extreme observations". My contribution consists of proposing a new logistic regression model combined with a weighting estimation procedure that incorporates a tuning parameter. Here I analyse some predictive performance indicators to analyse the extreme points behaviour. I show that the parameter defining the weights can be used to improve predictive accuracy, at least when the original predictive value is distant from the response average. A publicly available data set is used only to illustrate the new method. Hereby, I finally discuss the potential benefits of this methodology in imbalanced binary decision problems.

Chapter 3 is entitled as "Predictive Modelling of rare events with Complex Designed Survey Data". I study the logistic regression as a modelling technique

for rare binary dependent variables with much fewer events (ones) than non-events (zeros), and how it tends to underestimate their probability of occurrence. After exploiting the vast literature devoted to the prediction of binary rare data, I discovered several ways to improve the predictive performance through modifications in the likelihood estimation. Therefore, my contribution consists of proposing two weighting mechanisms that are incorporated in a pseudo-likelihood estimation that improve the predictive capacity of rare binary responses in data collected by complex surveys. I combine sampling weights with specific correctors that lead to lower root mean square errors for event observations in almost all deciles. A case study is discussed where this method is implemented to predict the probability of suffering a workplace accident in a logistic regression model that is estimated with data from a survey in Ecuador.

Chapter 4 is entitled as "Review of trials of classical and new boosting-based algorithms with Telematics Data". I examine rigorously several boosting-based algoritms that have been considered successful predictive techniques in the reviewed literature. I also present some new adapted versions of some original boosting algorithms that were motivated to improve the prediction of rare events by decreasing the root mean square error in the highest deciles of prediction of prediction (more probability of rare event occurrence) and lowest deciles of prediction (more probability of rare event non-occurrence). The main objective of this chapter is to gain statistical intuition of how conventional and proposed algorithms might react when predicting rare events.

Chapter 5 is entitled as "A Synthetic Penalized Logitboost to model Mortgage Lending with Imbalanced data". I examine that most classical econometric methods and tree boosting based algorithms tend to increase the prediction error with binary imbalanced data. Hence, my contribution consists of proposing a boosting-based algorithm called Synthetic Penalized Logitboost based on weighting corrections. The procedure (i) improves the prediction performance under the phenomenon in question, (ii) allows interpretability since coefficients can get stabilized in the recursive procedure, and (iii) reduces the risk of overfitting. A mortgage lending case study with publicly available data is used to illustrate the proposed method. I could obtain results whose errors are smaller in many extreme prediction scores, outperforming a number of existing methods. Additionally, the interpretations are consistent with results obtained using a classic econometric model.

Chapter 6 is entitled as "RiskLogitboot regression for rare events in binary response: An econometric approach". My contribution is to develop a boosting-based

machine learning algorithm called RiskLogitboost regression is presented for rare events in binary response. It pursues to (i) reduce the prediction error of the rare class, and (ii) approximate to an econometric model with coefficient estimation that allows the interpretability of the model. Hereby, I propose a weighting mechanism that oversamples and under samples observations according to their misclassification likeliness. Moreover, I also incorporate a generalized least squares bias correction strategy in the boosting procedure in order to reduce the prediction error. The RiskLogitboost is tested in a real insurance data set as an illustrative example. Results show that RiskLogitboost regression improves the rate of detection of rare events compared to some boosting-based and tree-based algorithms.

The various chapters of this dissertation can be found in:

1. Pesantez-Narvaez J., Guillen M. (2020). Weighted Logistic Regression to Improve Predictive Performance in Insurance. *Advances in Intelligent Systems and Computing*, 894, 22-34.

2. Pesantez-Narvaez, J., & Guillen, M. (2020). Penalized logistic regression to improve predictive capacity of rare events in surveys. *Journal of Intelligent & Fuzzy Systems*, 38(5), 5497-5507.

3. Pesantez-Narvaez, J., Guillen, M., & Alcañiz, M. (2019). Predicting motor insurance claims using telematics data—XGBoost versus logistic regression. *Risks*, 7(2), 70.

4. Guillen, M., & Pesantez-Narvaez, J. (2018). Machine Learning and Predictive Modeling for Automobile Insurance Pricing. *Anales del Instituto de Actuarios Españoles*, (24), 123-147.

5. Pesantez-Narvaez, J., Guillen, M., & Alcañiz, M. (2021). A Synthetic Penalized Logitboost to Model Mortgage Lending with Imbalanced Data. *Computational Economics*, 57, 281–309.

6. Pesantez-Narvaez, J., Guillen, M., & Alcañiz, M. (2021) RiskLogitboost Regression for Rare Events in Binary Response: An Econometric Approach. *Mathematics* 9(5), 579.

I also contributed to the following original publications which are aligned to the framework of this thesis:

1. Pesantez-Narvaez, J., Arroyo-Cañada, F.J., Argila-Irurita, A.M., Solé-Moro, M.L., & Guillen, M., forthcoming, Monitoring web-based evaluation of online reputation in Barcelona. *Advances in Intelligent Systems and Computing*, accepted.

2. Pesantez-Narvaez, J., Guillen, M., & Alcañiz, A., forthcoming, Modelling Subjective Happiness with a Survey Poisson Model and XGBoost using an Economic Security Approach. *Advances in Intelligent Systems and Computing*, accepted.

Chapter 2: Improving prediction accuracy in extreme observations

2.1 Introduction

Predictive modelling relates especially to knowledge discovery – methods for automatically detecting and constructing data patterns that have some significant predictive value Kuhn et al. (2013). Statisticians and econometricians translate business and economic problems into mathematical formulations to apply statistical modelling techniques and support decision making process. In fact, the scope of predictive modelling is even broader in areas like actuarial economics operations, targeted-marketing, credit industry, business development, public health, behavioural economics and other research and innovation fields. Within the insurance sector, accurate predictions are highly demanded (Frees et al., 2014). For instance, for the computation of policy holder's likeliness of accident, mortality, claim rates, life and non-life insurance policy lapses, insurance premiums pricing, forecast of future liabilities, fraud detection, loss estimation, and many other applications.

One popular method for binary dependent variables is the logistic regression model. This is implemented in a sample of n individuals who might correspond to policyholders, firms, business units or agents. The variable of interest is a binary response and every observation has a set of P covariates or characteristics that influence the dependent variable. The logistic regression final result allows predicting the expected value of the response, so the probability of occurrence.

Initially, all observations have the same weight. This means that the observations

This chapter is co-authored with Prof. Montserrat Guillén, and is an adapted version of Pesantez-Narvaez J., Guillen M. (2020) "Weighted Logistic Regression to Improve Predictive Performance in Insurance". *Advances in Intelligent Systems and Computing*, 894, 22-34.

have exactly the same relevance for the inference, or it is assumed that the observations are generated by a simple random sample. So, each observation has the same degree of representativeness of the population from which it has been extracted. The idea to introduce weights in the likelihood estimation is considered to improve predictive accuracy. In fact, this approach was firstly introduced in the missing data literature by (Robins et al., 1994). In geography studies, where the effect of each observation in the estimation result might strongly depend on the location (Agterberg et al., 1993), weights are used to correct this effect.

The sampling weights used in sample surveys are instruments to infer about a population using only specificic observations of it. In statistics, in the univariate setting a weighted average is calculated by multiplying the weighting mechanism *Wi* by the value of each observation (Winship and Radbill, 1994; Bethlehem and Keller, 1987). The simple summation of the weighted observations equal to sample size, in other words, the sum of weights is equal to sample size. Sampling weights are interpreted as a the relative size that each observation has with respect to the total number of population units that this observation represents.

Weighting can be aimed at differentiating the contribution of observations, so an individual, denoted by subscript i, with a small weight, it will have little influence on the results compared to other observations with a larger weight.

The objective of this paper is to study the influence of changing the weights of the initial sample in order to improve the accuracy of a simple predictive model. In particular, we focus on the logistic regression model and we study extreme observations with respect to the covariates, the ones that are farthest from the corresponding mean. As these data points might be considered unusual, their probability might be low and the predictive model can be inaccurate to anticipate their response. Our aim is to improve the prediction for this particular part of the sample.

This chapter proposes a specific weighting procedure to be incorporated in the likelihood estimation of a logistic regression as a particular case of the weighted likelihood method. The definition of the weighting mechanism depends on a parameter refered henceforth as tuning parameter and denoted as Φ. A discussion on the tuning parameter adjustment outcomes are presented. Additionally, I address how to find an optimal value for the tuning parameter. We use a real sample of insurance customers as an illustrative data set. It contains customers' information and their decision to buy a full coverage insurance versus a basic insurance product (Guillen, 2014). All analyses are performed in R language.

This chapter is divided in the following four parts. Section 2.2 presents the methodology description where theoretical basis of logistic regression and presented, as well as the statistical explanantion of the tuning parameter in the likelihood estimation. Section 2.3 describes an illustrative data set and some descriptive statistics. Section 2.4 shows the results of the performance measures which are used to evaluate the sensitivity analysis in predictive capacity of the weighted logistic regression model. Section 2.5 contains final conclusions.

2.2 Methodology

Logistic Regression Definition

A logistic regression model is a commonly used regression method to predict a binary discrete choice endogenous variable explained by one or more nominal, ordinal or ratio-level exogenous variables (Greene, 2003). Additionally, it is a particular predictive modelling technique because it aims at finding the probability of occurrence of an event and the result is bounded between 0 and 1.

The logistic regression model is a particular case of the generalized linear model (McCullagh and Nelder, 1983). The logit function is the canonical link and is given as:

$$\begin{aligned} g(\pi_i) &= log\left(\frac{\pi_i}{1-\pi_i}\right) \\ &= \beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p \end{aligned} \tag{2.1}$$

for $i = 1, \ldots, n$ observations, $p = 1, \ldots, P$ denoting the covariates where β_0, β_1,..., β_p are the model parameters, and π_i is the probability of the observed event in response Y_i. Thus, in the logistic regression model, the logit transform, *i.e.* the log-odds of the probability of the event, equals the linear predictor:

$$\pi(Y_i = 1) = E(Y_i) = \frac{e^{\beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p}}{1 + e^{\beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p}}. \tag{2.2}$$

A logistic regression can be estimated by maximum likelihood (for further details

see McCullagh and Nelder (1983)).

The Tuning Parameter in the Weighted Likelihood

To formally define the concept of weighted logistic regression, we first address the notion of weighted likelihood estimation in general.

Let $\tilde{Y} \cong (Y_1, Y_2, ..., Y_n)'$ be a simple random sample of a binary random variable with a probability density function [1]:

$$P\left(Y_i = y_i | X_i, \beta\right) = \frac{e^{\left(\beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p\right)}}{1 + e^{\left(\beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p\right)}} \tag{2.3}$$

with the vector of parameters $\beta = (\beta_0, \beta_1, ..., \beta_P)'$ where $\beta \subseteq \Re^{P+1}$.

Let Θ be the sampling space, in other words, all possible values of $\tilde{Y}$ Then the likelihood function is defined for $Y \cong (Y_1, Y_2, ..., Y_n)' \in \Theta$ as:

$$\mathbf{L}\left(\cdot|\tilde{Y}\right) : \beta \rightarrow L\left(\beta|\tilde{Y}\right) = P\left(\tilde{Y}|X, \beta\right) = \prod_{i=1}^{n} P\left(Y_i = yi|X_i, |\beta\right). \tag{2.4}$$

We take the logarithm of the likelihood because it is a strictly increasing function and the extreme points are the same for the logarithm of the likelihood function and for the likelihood function itself. Consequently, using the properties of the logarithm, we define the log-likelihood function as:

$$lnL\left(\beta|\tilde{Y}\right) = \sum_{i=1}^{n} lnP\left(Y_i = y_i|X_i, \beta\right). \tag{2.5}$$

For each $\tilde{Y} \in \theta$ the maximum likelihood estimator of β is denoted as $\hat{\beta}$ and it corresponds to the value of β that maximizes the likelihood $\mathrm{L}(\cdot|\tilde{Y})$ since the purpose is to find the parameters for which the probability of the observed data is the greatest possible value:

$$\mathbf{L}\left(\hat{\beta}|\tilde{Y}\right) = max_{\beta} ln \prod_{i=1}^{n} f(Y_i|\beta). \tag{2.6}$$

where $f(Y_i|\beta)$ is $P(Y_i|X_i, \beta)^{Y_i}$ $(1 - P(Y_i|X_i, \beta)^{Y_i})^{1-Y_i}$ The weighted logistic regression method is based on a weighted log-likelihood estimation, denoted by $\mathbf{l}\left(\beta\right)$, which is defined as:

[1] We use ' to denote the transpose of a vector

$$\mathbf{l}(\beta) = \sum_{i=1}^{n} W_i \times f(Y_i|\beta), \tag{2.7}$$

where $W_i = (W_1, W_2, \ldots, W_n)'$ is the vector of weights.

This modification can be a consequence, for instance, of the existence of common sampling weighting $Wi = \frac{\Upsilon_i}{\varrho_i}$ where Υ_i is the fraction of the decision-making population, and ϱ_i is the analogous fraction of the decision-making sample that are represented by observation i (Manski and Lerman, 1977).

In this chapter, we propose a vector of weights which is constructed as follows:

$$\tilde{W}_i = \left|\hat{Y} - \bar{Y}\right|^{\Phi}, i = 1, ...n \tag{2.8}$$

where $\hat{Y}_i$ is defined as the estimated probability obtained by the standard logistic regression model for observation i (as in (2.2) from Section 2.2). Let us consider $\bar{Y}$ as the mean value of the endogenous variable. The weights' definition depends on Φ, which is a real number and can be called the tuning parameter. This parameter is calibrated later [2], however it should be noted that when $\Phi = 0$, then all weights are equal to one.

A change of the vector of weights determines a change in the estimated model coefficients, as well as their level of significance, which can even reverse the impact from positive to negative or the other way round, the weighted estimation procedure can modify the magnitude of influence on the outcome binary variable of each covariate, which, in turn, directly influences the results and so, the confusion matrix.

The main idea for defining weights as (2.8) is to find the best tuning value. These weights depend on the distance between the initial predictive value and the mean of the observed outcome. For a positive tuning parameter the weight is larger in the most extremal observations which lay far from the mean, whereas it is smaller more than those that are close to the mean. The possible scenarios for selecting the tuning parameter are:

In this paper, the concept of weight is not defined as in other cases in the literature on weighted regression. Other approaches such as Adaboost and similar machine learning algorithms (see (Friedman et al., 2000)) have a totally different approach to weighting. In that case, more weight is given to wring predictions and less weight is given to correct predictions.

[2]The tuning parameter is chosen so that the best value provides the best predictive performance obtained by the goodness of fit tests.

Tuning parameter values	Description
$\Phi = 0$	The maximum likelihood estimation remain the same as the unweighted model
$\Phi > 0$	The weighting gives more importance to the observations whose original predictive value is far from the mean,
$\Phi < 0$	The weighting gives more importance to the observations whose original predictive value is close to the mean.

Table 2.1: Tuning Parameter Scenarios

This proposal does not look at the similarity between the predicted and the observed response. Observations that are distant from the average predicted response should be given more importance than to those that are closer to the average.

The estimation procedure is not corrected directly in order to improve accuracy in one step. The main idea is to look at the distance between the predicted value and the observed value for each observation and then to re-estimate. This difference is substantial and it is the reason why our contribution differs, up to our knowledge, to existing approaches.

The notion of Real Adaboost, coined by (Friedman et al., 2000), suggests that the weight should be a transformation of the class probability estimate. These authors show the statistical equivalence of the weighted estimating procedures to the minimization of a loss functions. The proposal is an additional approach that is suitable for distant observations, where distance is defined by a norm in the space of the covariates.

2.3 Illustrative Data

Data have been taken from a Spanish insurance company. A sample of 4,000 policy holders of motor insurance has been analyzed [3].

The motor insurance data set has seven covariates. Policy holder's age (*Age*),

[3]The data set can be found in the following web of R resources for quantitative analysis at the University of Barcelona: www.ub.edu/rfa/R

number of years in the company (*Seniority*), insured's gender(*Men*), type of zone (*Urban*), vehicle use (*Private*), insured' marital status (*Single, Married, Others*); and, finally, the choice to buy a full coverage policy versus a simple one is captured by the dependent variable Y_i.

Additionally, Table 2.2 shows some brief descriptive statistics of the data. Firstly, the percentage of women who purchase a full coverage insurance vs a basic coverage is quite similar (almost a half) whereas 70.27% men decided to buy a basic coverage product. Furthermore, a big percentage of the married insurance holders seem to prefer the basic coverage with reference to the single and other insurance holders. Most people who drive in rural areas have purchased a basic coverage while people in urban areas have almost a similar tendency between basic and full coverage. Moreover, the average age of insurance holders who choose a basic coverage is older than the one of full coverage. And finally, people who have more seniority in the company purchase more often full coverage insurance than newer customers.

Covariates		**Basic Coverage (Y=0)**	**Full Coverage (Y=1)**	**Total**
Age (years)		48.27	43.09	46.47
Seniority		9.93	12.66	10.88
Sex	Woman	498 (50.30%)	492 (49.70%)	990
	Man	2115 (70.27%)	895 (29.73%)	3010
Driving Area	Rural	1906 (72.83%)	711 (27.17%)	2617
	Urban	707 (51.12%)	676 (48.88%)	1383
Vehicle Use	Commercial	33 (84.62%)	6 (15.38%)	39
	Private	2580 (65.14%)	1381 (34.86%)	3691
Marital Status	Single	467 (54.24%)	394 (45.76%)	861
	Married	2047 (65.85%)	926 (31.15%)	2973
	Other	99 (59.64%)	67 (40.36%)	166

Continuous variables are expressed in the mean. The number of observations is 4,000. The percentage of individuals who choose full coverage is 34.68%.

Table 2.2: Motor Insurance Data Set

2.4 Results

In this section, tuning parameter behavior is studied with the proposed weighting procedure through some statistical measures. The idea is to evaluate the decision to purchase a full coverage insurance (coded as 1) versus a basic coverage (coded as 0) determined by some exogenous variables through a logistic regression as base model:

$$\pi\left(Y_i = 1\right) = \frac{e^{ZZ_i}}{1 + e^{ZZ_i}} \tag{2.9}$$

$$\pi\left(Y_i = 0\right) = 1 - \frac{e^{ZZ_i}}{1 + e^{ZZ_i}} \tag{2.10}$$

$$\begin{aligned} ZZ_i = \beta_0 + \beta_1 * Age_i + \beta_2 * Seniority_i + \beta_3 * Men + \beta_4 * Urban_i \\ + \beta_5 * Private_i + \beta_6 * Married_i + \beta_6 * Others_i, \end{aligned} \tag{2.11}$$

where Y_i is the response variable, in this case the binary full coverage purchase variable, β_0 is the constant coefficient, and β_p are the coefficients related to the independent variables in Table 2.2. Similar approaches with discussions on the classifiers have been used by other authors (Guelman and Guillen, 2014; Guelman et al., 2014, 2015).

Weighted Log-Likelihood Performance

The log-likelihood function summarizes information on the parameter that is given by the sample. Since the original likelihood estimation is now being adjusted by the proposed weights, it is necessary to ensure that the new function is still concave. So, a global maximum likelihood estimate can be found numerically after a some iterations.

Figure 2.1 shows the maximum log likelihood values with $\Phi \in \{0, 30\}$ where a maximum of all can be detected when $\Phi \in \{7, 10\}$.

Norm of the Estimated Parameters

A norm, denoted by $\|.\|$, finds a strictly positive length of a vector $\tilde{V}$ in a vector space $(\tilde{V}, \|.\|)$. The norm metric on $\tilde{V}$ is generally defined by $\|\mathbf{ll} - \mathbf{cc}\|$ with **ll** and **cc** vectors (Deza and Deza, 2009).

The intuitive idea of measuring the distance between the estimated coefficients

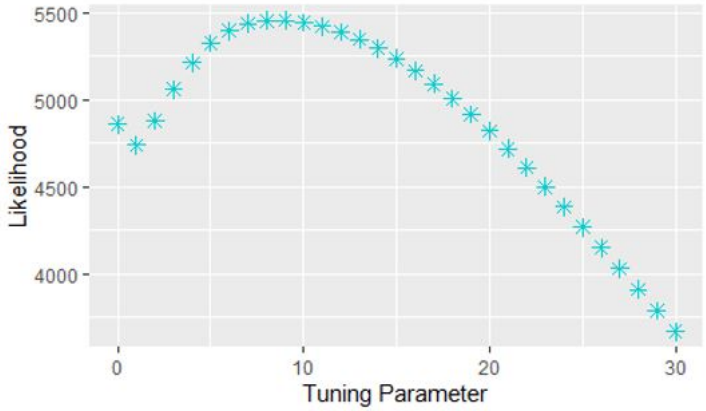

Figure 2.1: Maximum log likelihood values of the estimated models versus the tuning parameter from 0 to 30

$\hat{\beta}$ from the base model (2.2) and the estimated coefficients $\hat{\beta}_{\Phi}$ from the proposed weighted logistic model with the weighting procedure of (2.8) is defined as the Euclidean distance between vectors, namely, $\left\|\hat{\beta} - \hat{\beta}_{\Phi}\right\|^{0.5}$ with $\Phi \in \{0, 100\}$.

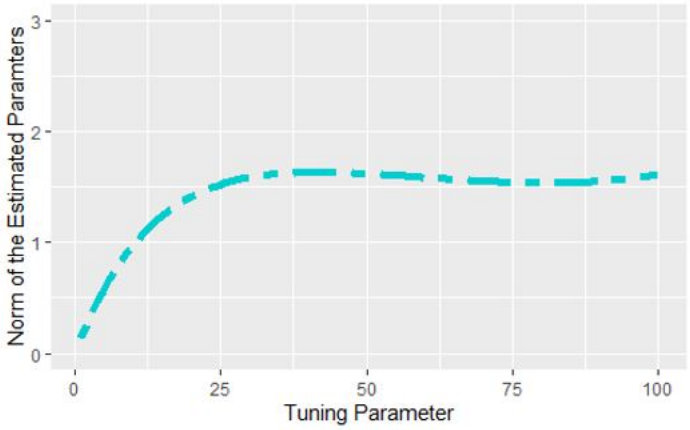

Figure 2.2: Maximum log likelihood values of the estimated models versus the tuning parameter from 0 to 100

Figure 2.2 shows that the norm when $\Phi \in \{22, 30\}$ is approximately the largest which means that the tuning parameter taken in this interval already shows that the parameter estimates for the weighted maximum likelihood are distant from the unweighted model.

Performance Metrics

Performance metrics are used to evaluate the corrected of models under certain

criteria. In particular, confusion matrix or classification matrix is one of the most intuitive metrics to measure the classification performance of classifiers with respect to some test data in studies of artificial intelligence, information retrieval and data mining (Jiang and Liu, 2013). Thus, this predictive method is used to evaluate the accuracy of the results of the model under a given classifier (Ting, 2017).

		Predicted	
		Basic Coverage ($\hat{Y}$=0)	Full Coverage ($\hat{Y}$=1)
Observed	Basic Coverage (Y=0)	True Negative (TN)	False Positive (FP)
	Full Coverage (Y=1)	False Negative (FN)	True Positive(TP)

Table 2.3: Confusion Matrix Definition

A confusion matrix is a two-dimensional matrix where observed data is compared with the predicted values under the given classification algorithm. Table 2.3 shows the four alternative classification outcomes when placing the models results into a confusion matrix.

The three measures of classification performance that we are going to analyze are:

- Sensitivity, which measures the proportion of policy holders that were classified in the full coverage insurance among those who effectively purchased full coverage insurance. TP / (TP+FN).
- Specificity, which measures the rate between the policy holders who were classified in the basic coverage insurance among those who purchased basic coverage insurance. TN / (TN+FP).
- Accuracy, which measures the rate of policy holders who are correctly classified. (TP+TN)/(TP+TN+FP+FN).

Consequently, the confusion matrix is used to measure the predicting performance of a model with Φ varying from to 0 to 10. The purpose is to find the value of Φ that guarantees the highest levels of sensitivity, specificity and accuracy.

In Figure 2.3 the tuning parameter value of each model is written above each plotted point. Sensitivity and specificity are evaluated at a threshold equal to 0.3.

The purpose of Figure 2.3 is to find the model that is geometrically closest to the point (0,1). This rule is considered as the optimal criterion to find the best predictive model with high sensitivity and high specificity.

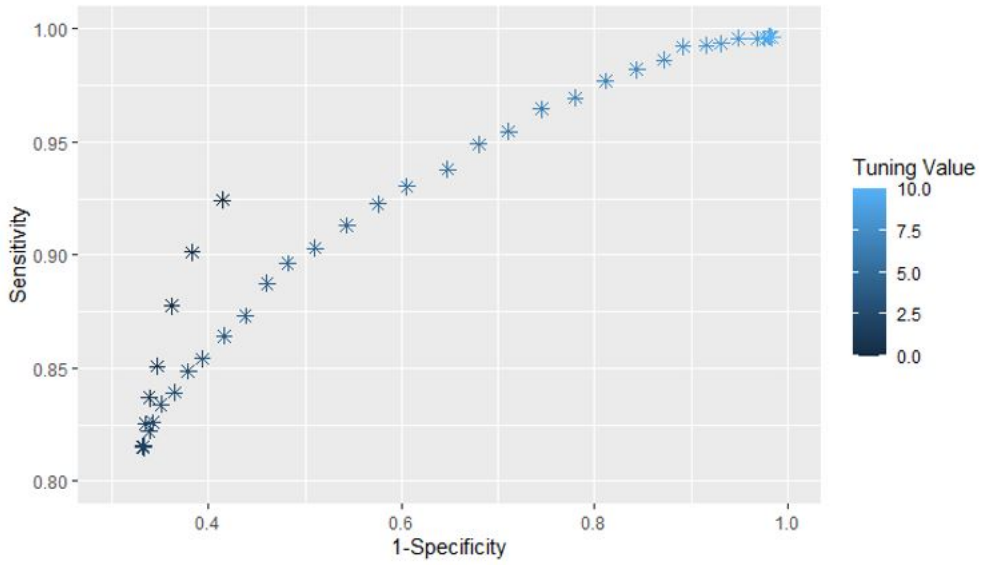

Figure 2.3: Classification performance (sensibility and 1-specificity) of the estimated weighted logistic regressions

The optimal tuning parameter Φ is equal to 1 and, in this case the sensitivity is 0.84, the specificity is 0.66 and the accuracy is 0.72.

Table 2.4 shows the results of the confusion matrix of model (2.11) estimated with an unmodified version of the logistic regression model, and the confusion matrix of model (2.11) estimated with a weighted logistic regression and its optimal tuning parameter value $\Phi = 1$. The weighted model shows a better true negative rate than the unweighted model, however, this weighted model has a lower true positive rate. This result is not surprising since the proposed weighting mechanism is focused for extreme observations, and Table 2.4 shows the metric in aggregated terms.

Extreme Points Analysis

Data points which are far from the mean predictors can be considered extreme. Thus, the first and the last decile of the predictions (associated to policy who are the least likely to purchase a full coverage insurance and the policy holders who are the most likely to purchase a full coverage insurance respectively) are analyzed.

The root mean square error (RMSE) is calculated for the first and the last decile of predictions from the weighted ($\Phi = 1$) and the unweighted logistic regression models ($\Phi = 0$). The RMSE is used to measure the distance between the predicted values by the model are from the observed ones. Then the smaller RMSE value the model has, the better predictive performance it has.

Unweighted Logistic Model			
		Predicted	
		($\hat{Y}$=0)	($\hat{Y}$=1)
Observed	(Y=0)	1708	905
	(Y=1)	171	1216
Weighted Logistic Model with $\Phi = 1$			
		Predicted	
		($\hat{Y}$=0)	($\hat{Y}$=1)
Observed	(Y=0)	1729	226
	(Y=1)	884	1161

Table 2.4: Confusion matrix for the unweighted and weighted logistic model

The RMSE is defined after a table has been constructed to separate the observations in deciles according to the predictive value. Then,

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(Y_i - \hat{Y}_i)^2}{N}}, \tag{2.12}$$

where N is the number of observations in each decile, in this case 400.

Table 2.5 shows that the weighted logistic regression model has a lower RMSE in the highest decile of predictions. This model has a predictive accuracy that is better than an unweighted model for those policy holders with a large predicted probability of purchasing full coverage insurance. The RMSE is also small in the smallest decile.

	Smallest Decile	**Highest Decile**
Unweighted Logistic regression	0.492	3.161
Weighted Logistic regression	0.385	1.511

Table 2.5: Confusion Matrix Definition

Parameter Estimates

Table 2.6 presents the model estimates for the classical logistic regression (unweighted) and the weighted logistic regression with a tuning parameter equal to 1.

Parameter Estimates of the Unweighted model		
Variable	**Parameter Estimate**	**Pvalue**
Intercept	-0.257 (0.486)	0.596
Men	-0.961 (0.009)	0.001
Urban	1.173 (0.008)	0.001
Private	1.065 (0.469)	0.023
Marital ($married$)	-0.083 (0.096)	0.384
Marital ($others$)	0.161 (0.096)	0.421
Age	-0.058 (0.004)	0.001
Seniority	0.133 (0.007)	0.001
Parameter Estimates of the Weighted model		
Variable	**Parameter Estimate**	**Pvalue**
Intercept	-0.479 (0.527)	0.363
Men	-0.782 (0.109)	0.001
Urban	0.908 (0.001)	0.001
Private	1.059 (0.495)	0.033
Marital ($married$)	-0.020 (0.124)	0.869
Marital ($others$)	0.236 (0.253)	0.351
Age	-0.048 (0.005)	0.001
Seniority	0.099 (0.008)	0.001

The standard errors are expressed in brackets.

Table 2.6: Parameter Estimates for the Weighted and Unweighted Model

2.5 Conclusions

Based on the first exploratory analysis from Section 2.4, the maximum log-likelihood values among all the estimated weighted logistic regression models correspond to a tuning parameter between 7 and 10. The intuitive idea is that, in the weighting mechanism, a tuning parameter can improve the likelihood of the estimated model.

The results show that a tuning parameter between 2 and 4 has the largest norm of the difference between the vector of estimated parameters in the weighted model and the vector of estimated parameters in the unweighted model.

The best tuning parameter that accomplishes the highest specificity and sensitivity rates is equal to 1. This choice is based on the optimal criterion presented in Section 2.4. For this case, the estimated weighted model has less root mean squared

error in the extreme deciles than the unweighted model. The proposed weighting mechanism obtains more correct predictions than the classical logistic regression for the policy holders that are most likely to purchase full coverage insurance. Thus, future retention managerial strategies for this group of insurance policy holders can be based on the proposed weighted estimation procedure with $\Phi = 1$.

The conclusion is that weighted logistic regression offers an array of opportunities to improve classifiers and we aim at pursuing further research in the analysis of subsamples of the population that correspond to the extremes, rather than looking at the global performance.

Chapter 3: Predictive modelling of rare events with complex designed survey data

3.1 Introduction

Ample experimental and observational research in economic science involves binary label problems that deals with much fewer events (ones) than non-events (zeros). We address the statistical problem of modelling survey data as in (King and Zeng, 2001), who propose a method to correct the likelihood estimate in logistic regression that seeks to predict rare events.

Examples of phenomena that do not occur very often can be found in all areas, where the percentage of cases of interest falls below 5 or even 1%. In socioeconomic surveys, model rare phenomena could include the estimation of the proportion of workers who changed their job in the week prior to the interview. In health surveys, responses to the use of certain drugs or diseases can also be quite infrequent.

Economic research and policymaking are undergoing profound transformations, thanks to the availability of larger multipurpose social survey data sets to address rare event-research questions, uncover more of its causal effects and contribute strongly to their evidence-base (Connelly et al., 2016; Langedijk et al., 2019). For instance, these type of data can be found in Spanish Survey of Living Conditions, European Survey of Enterprises on New and Emerging Risks – Managing safety

This chapter is co-authored with Prof. Montserrat Guillen, and is an adapted version of Pesantez-Narvaez, J., & Guillen, M. (2020). Penalized logistic regression to improve predictive capacity of rare events in surveys. *Journal of Intelligent & Fuzzy Systems*, 38(5), 5497-5507.

and health at work, Ecuadorian National Survey of Employment, Unemployment and Underemployment, UK Millennium Cohort Study among many others.

Observational data such as surveys collect information from designed, structured and representative sample of respondents from well-defined finite population so that researchers can make inferences and generalizations about the entire population through statistical inference. Particularly, simple random sampling (SRS) is an unbiased surveying technique that allows every respondent involved to have the same probability of being chosen. However, because every individual has to be listed before the randomization procedure, this technique might be cumbersome for large-scale population studies.

As a result, due to economic and time costs, surveys are usually conducted using complex sampling designs (e.g. stratified, cluster or two-stage sampling) rather than SRS. Sampling weights are defined to make the sample representative of the population and to avoid selection bias, even if the observations in some survey designs are dependent.

The design effect, which measures the ratio between the variance estimation under a specific sample design and that of an SRS, varies from one survey to another and even varies for each estimator within a given survey. Deviations from SRS are expected to produce a loss of efficiency, but this loss should be kept as low as possible. Sampling errors should be carefully estimated, and inference in general must consider the data collection mechanism.

Even when sampling weights are considered in the modelling process, randomness is still influenced by the sampling procedure. Lumley et al. (2004) demonstrates that the modelling process must take into account sampling weights as well as the random part of the model to obtain the precision of the estimates, and to assess modelling performance.

Apart from the complexity in the way survey samples are obtained, the presence of rare events i.e. binary dependent variables that have few non-zero cases, is quite common in practice. This can represent a challenge for the performance of predictive models, which seek to determine the factors affecting the probability of the rare event. The reason for this is that the small number of observed cases leads to quite unstable model results. King and Zeng (2001) prove that binary dependent models, in particular logistic regression, tend to underestimate the event probability for this type of rare event data, and they propose a correction procedure in the usual

logistic regression maximum likelihood estimation to manage bias. However, they leave aside rare binary dependent variable modelling prediction as a design-based analysis with sampling weights. Yet, ignoring sampling weights might affect the meaning and precision of the coefficients.

Modifications of the maximum likelihood estimation through weights are not new in the vast literature devoted to generalized linear models. For instance, Wedderburn (1974) introduces the quasi-likelihood function, Manski and Lerman (1977) modify the weighted exogenous sampling likelihood function estimator by weighting each observation's contribution to the likelihood. Manski and Lerman (1977) and Field and Smith (1994) incorporate weighting mechanisms in the maximum likelihood estimation method.

The objective is to improve the predictive capacity of models for rare phenomena with data collected in a complex sample design. The proposal consists on a new method and we also present a case study, where we analyse survey data to model the occurrence of workplace accidents.

This chapter proposes a statistical procedure that incorporates both approaches. We consider rare events in samples that deviate from SRS and we modify the maximum likelihood estimation to improve the predictive accuracy of the model. Hence, we aim to contribute to the existing literature by proposing a weighting mechanism that can be incorporated in the likelihood estimation, which then naturally becomes a pseudo-likelihood estimation, of a penalized logistic regression model. This mechanism is capable of performing two joint tasks: first, it controls the randomness of a sampling procedure by considering the sampling weighting, stratification or clustering that originates from a complex survey design; and second, it provides the model with greater sensitivity, in order to obtain more accurate predictions of rare events than if only a weighted design-based logistic regression model had been used.

The motivation for proposing a weighting mechanism is that it allows us to differentiate between the relevance of observations in the sample. In this way, we can avoid the under-representation or over-representation of observations when it comes to estimating choice probabilities from choice-based samples as introduced by (Manski and Lerman, 1977). But the mechanism extends this idea further, so that the importance of the observations varies depending on the proximity to the mean value of the response. An adjustment parameter calibrates the impact of the weighting mechanism on the model estimation. In addition, a threshold value is

chosen to provide the best predictive performance.

Following on from this introduction, this paper is divided in four parts. Section 3.2 outlines the methodology and the two weighting mechanisms are presented and justified in detail. Three criteria are proposed to find the best predictive model among all possible models by choosing an optimal weight adjustment and a classifying threshold. Section 3.3 describes the data used herein as an illustrative example. Specifically, we are interested in modelling the occurrence of workplace accidents. Section 3.4 presents the results and the predictive performance obtained in the case study. Section 3.5 concludes.

3.2 Methodology

Let X_{ip} be the data matrix where i corresponds to observations (or instances) and p corresponds to the independent variables (attributes or features), with $i = 1, \ldots, n$ and $p = 1, \ldots, P$. There are n observations and P independent variables. And let Y_i be the binary outcome for observation i.

The purpose is to classify observations between the binary outcome Y_i taking into consideration the covariates X_p.

3.2.1 Penalized logistic regression and pseudo-likelihood estimation

One supervised method of machine learning is the logistic regression model. Greene (2003) and McCullagh and Nelder (1983) define logistic regression as a predictive method used for binary classification problems which, unlike a linear regression model, provides estimates about the probability of an outcome.

To formally define the penalized logistic regression model, we first introduce the pseudo-likelihood estimation (weighted maximum likelihood) with survey data.

For every instance X_i (row vector of X_{ip}), the outcome response is either $Y_i = 1$ if the observations belong to a positive class (event) or $Y_i = 0$ if they belong to a negative class (non-event).

Binary variable Y_i is a Bernoulli trial:

$$Y_i \sim Bernoulli\left(Y_i|\pi_i\right),$$

where π_i is the probability that Y_i equals 1 and is specified as:

$$\pi_i = \pi\left(Y_i = 1|X_{i1}, ..., X_{iP}\right) = \frac{e^{\left(\beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p\right)}}{1 + e^{\left(\beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p\right)}} \tag{3.1}$$

Conversely, the probability that Y_i equals 0 is $1 - p_i$. Unlike linear regression, logistic regression uses a logit function as the linear predictor, which is the log odds of the positive response, defined as:

$$\begin{aligned} g\left(\pi_i\right) &= log\left(\frac{\pi_i}{1-\pi_i}\right) \\ &= \beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p. \end{aligned} \tag{3.2}$$

Then, the classical likelihood function is the joint Bernoulli probability distribution of observed values of Y_i as follows:

$$L\left(\beta; X_i\right) = \prod_{i=1}^{n} \pi_i^{Y_i}\left(1-\pi_i\right)^{1-Y_i}. \tag{3.3}$$

Parameter estimates of the classical logistic regression can be found by maximizing the likelihood or log-likelihood function [1]. Due to computational convenience, we use the log-likelihood function, which we denote by $\mathbf{L}$ for simplicity:

$$\mathbf{L} = \sum_{i=1}^{n} ln\pi_i^{Y_i} + ln\left(1-\pi_i\right)^{1-Y_i}. \tag{3.4}$$

Furthermore, if weights are incorporated in the log-likelihood function (3.4) then a pseudo-likelihood is obtained, which is also known as a weighted log-likelihood:

$$\mathbf{l}\left(\beta\right) = \sum_{i=1}^{n} W_i\left(ln\pi_i^{Y_i} + ln\left(1-\pi_i\right)^{1-Y_i}\right). \tag{3.5}$$

where W_i represents the weight of the i-th observation. Therefore, estimating the

[1] Maximizing a log-likelihood function is equivalent to maximizing a likelihood function.

parameter vector becomes a maximization problem whose objective function is the pseudo-likelihood function defined in (3.5).

Maximization in (3.5) can be computed with the *survey*() package in R: Partial derivate equations are solved by an iteratively reweighted least squares algorithm, which is a Fisher scoring algorithm (further details can be found in (Green, 1984)). The *survey*() package created by (Lumley et al., 2004) not only allows the weighting procedure to be incorporated, but it also adapts the penalized logistic regression to complex survey designs in order to provide design-based standard errors. So, if survey data include a stratified and/or a clustered design, the maximization includes the corresponding formulas to find correct sample-based standard errors.

Winship and Radbill (1994) note the importance of weighting the observations from complex samples in order to derive unbiased estimates of population features. Weighting can be used to both guarantee sample representativeness in a modelling process (as noted by (Manski and Lerman, 1977)) and to control the relevance of observations. Thus, our approach proposes weighting observations not only to correct a survey sample design but also to improve its predictions. This is of particular interest for low frequency events, which are more difficult to predict than high frequency occurrences. Our corrections are introduced in a penalized logistic regression model with a pseudo-likelihood estimation method.

Sample correction and weighting aimed at improving predictive capacity have both been widely discussed in the literature but, to the best of our knowledge, in these discussions they have typically been addressed separately. We aim to study these weighting procedures jointly and define W_i in (3.5) according to these objectives.

Weighting Mechanisms

Let SW_i be a vector of sampling weights, and PW_i a vector of predictive weights. These two weighting mechanisms are introduced in (3.5) where W_i is the result of the product between SW_i and PW_i.

The basis for the sampling weights lies in the probability of choosing a respondent. This means that each observation in the sample is given a weight to account for the probability of that observation being selected from the population. For this reason, sampling weights incorporate an expansion factor that is equal to the number of population units represented by each observation in the sample. Sampling

weights are defined as follows:

$$SW_i = \frac{F_{exp_i} \times n}{\sum_{i=1}^{n} F_{exp_i}} \tag{3.6}$$

where F_{exp_i} is the vector of expansion factors defined as the inverse of the probability of choosing each observation in the sample, n is the total number of the sample.

For the predictive weighting, PW_i, we propose two alternatives, which we call PWa_i and PWb_i:

a)
$$PWa_i = \left|\hat{Y}_i - \bar{Y}\right|^{\epsilon}$$

b)
$$PWb_i = \left|\hat{Y}_i - \bar{Y}^{\epsilon}\right|$$

where $\hat{Y}_i$ is the vector of estimated probabilities of a simple initial weighted, design-based logistic regression (accounting for SW_i only, where other sample-design features such as stratification and/or clustering would only affect standard errors) and $\hat{Y}$ is the estimated weighted mean response of the dependent variable. Let ϵ be the adjustment parameter that calibrates the distance between $\hat{Y}_i$ and $\bar{Y}$ in both alternatives PWa_i and PWb_i.

Note that the estimated probabilities $\hat{Y}_i$ lie between 0 and 1.

- PWa_i differentiates the weight of observations that are located far from the mean. The possible scenarios for selecting the adjustment parameter are:
  - $\epsilon = 0$:The maximum pseudo-likelihood estimation remains the same as the weighted design-based model.
  - $\epsilon > 0$: The weighting attaches greater importance to the observations whose original predictive value is located far from the mean response.
  - $\epsilon < 0$: The weighting gives greater importance to the observations whose original predictive value is located near the mean response.
- PWb_i isolates the estimated probabilities from the mean. The choice of the threshold is usually located near the mean response. Observations whose predicted probability is located near the mean are more likely to be influenced by the choice of the threshold, than those that have a predictive probability that is located far from the mean. This weighting mechanism allows three possible scenarios for selecting the adjustment parameter:

- $\epsilon = 0$: Then $\bar{Y}^{\epsilon}$=1 and the predictive weights equal the estimated probability of the non-event, (1-$\hat{Y}_i$). More weight to the observations which are much greater than the mean and less weight to the observations which are much smaller than the mean.
- $\epsilon < 0$: Less weight to the observations which are located far from the mean and more weight to the observations which are located near the mean.

So far, PW_i and SW_imay have a different scale. While the sampling weights in (3.6) sum up to n, this is not necessarily true of the predictive weights. Therefore, we propose rescaling them and obtaining the new PWa_i* and PWb_i* as follows:

$$PWa_i* = \frac{PWa_i \times n}{\sum_{i=1}^{n} PWa_i} \tag{3.7}$$

$$PWb_i* = \frac{PWb_i \times n}{\sum_{i=1}^{n} PWb_i} \tag{3.8}$$

Then, the two final weights PWa_i* and PWb_i* combining the sampling and predictive weights can be defined as:

$$PSWa_i* = SW_i \times PWa_i* \tag{3.9}$$

$$PSWb_i* = SW_i \times PWb_i* \tag{3.10}$$

3.2.2 Choosing the adjustment parameter

Three criteria are established for choosing the adjustment parameter to test the predictive performance of each model.

1. **Receiving operating characteristic (ROC) optimal criterion**

   Hanley and McNeil (1982) propose the ROC curve as a graphical plot that seeks to determine the relationship between sensitivity – i.e. the percentage of true positive values (on the y-axis) – and 1-specificity – i.e. the percentage of false positive values (on the x-axis). Sensitivity and specificity are measures of the performance of a binary classification method. Sensitivity is a measure of the proportion of actual positives (events) that are correctly identified as such, while specificity is a measure of the proportion of actual negatives (non-events). The ROC curve illustrates the capacity of the logistic

regression model, as a particular case of a binary classifier method given a threshold Ψ.

The threshold is a fixed value in [0,1], which determines when an estimated probability is large enough for the binary prediction to take the value of 1. The desired model should have a high true positive rate as well as a small false negative rate Therefore, the best prediction model would yield a point on the ROC curve that is as close as possible to the coordinate (0,1).

The ROC optimal criterion is based on setting all possible adjustment parameters ϵ in the domain of the penalized logistic regression, considering that for each ϵ, there is a choice of possible thresholds $[0.01, 0.02, \ldots, 0.99]$. The best model coordinates in the ROC plot are those with the shortest distance to the point (0,1). All ROC distances to the coordinate (0,1) are computed. Therefore, the ROC optimal criterion is a minimization problem where ϵ and Ψ have to be found.

2. **Constrained received operating characteristic (C-ROC) criterion**

The C-ROC criterion is motivated by a discussion of desirable statistical performance measures of a good predictive model. A good predictive model would be expected to accomplish maximum levels of sensitivity, minimum type I and type II errors or, at least, a minimum type II error.

First, a predictive model with maximum sensitivity is especially important for identifying the true positive rate ($Y_i = 1$), which is the main point of interest for our study. However, finding such a model might imply very low levels of specificity, which might be a disadvantage. Second, a good predictive model can also be expected to have the smallest possible false positive and false negative rates. However, it is far from straightforward to minimize both false positive and false negative rates, because when one is low the other is high.

Thus, finding a suitable cut-off threshold for deciding the best predictive model in line with this criterion requires making a compromise. Third, reducing type II errors might be considered dangerous in prediction implementations because, in some cases, the reason for predicting rare events is to

prevent them.

Thus, so far, it would seem that the three requirements are all necessary, but that they are not all feasible at the same time. For this reason, taking as our base criterion the ROC analysis described above, we propose using the C-ROC, which comprises the following two steps:

2.1. Find the first adjustment parameters based on the ROC optimal criterion [2]. In order words, this requires ranking the models from best to worst in terms of how well they meet the ROC criterion and selecting the first m ones.

2.2. Maintaining the subset based on this previous order and finding the adjustment parameter whose corresponding model has the highest sensitivity value. If values are equal then, once fixed, select the one with the highest specificity. The goal is to retain the model with the highest levels of sensitivity, reducing a minimum specificity. This is feasible if the adjustment parameters of each predictive model are first sorted according to the ROC criterion.

3. **Assessing performance with the root mean square error (RMSE)**

This is a statistical measure that rates the difference between observed and predictive values: the smaller the RMSE, the better the model's predictive performance. The RMSE is calculated as follows:

$$RMSE = \sqrt{\frac{\left[\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2\right]}{n}} \tag{3.11}$$

where $\hat{Y}_i$ is the predicted values from the estimated model. In our application, we have used this criterion only for the subsample of events and (3.11) was used to analyse predictive performance rather than as a criterion to select the adjustment parameters.

[2] Let m be an integer positive number. The intuitive idea of m is just how many better models in terms of the ROC criterion, the analyst is willing to sacrifice in order to decide for a model with a higher sensitivity among those m models. However, m should be small enough so as to maintain models ROC distance as small as possible. It is advisable to select m between 2 and 10, nevertheless the choice depends on the quantity and characteristics of sample data. In the application shown in this article, m is fixed equal to 6.

3.3 Illustrative Data

We use a workplace accident data set taken from the Ecuadorian National Survey of Employment, Underemployment and Unemployment (ENEMDU) conducted in December 2017 by the *Instituto Nacional de Estadísticas y Censos* (INEC). The data were collected in personal interviews to gather information about the labour market in Ecuador. The survey employs a two-stage sampling design: the first step involves the stratification of 2,586 primary sample units (PSUs) represented as sectors, and the second step involves choosing 12 secondary sample units (SSUs) per every PSU represented as dwellings by a simple probabilistic sampling. The final observation unit is the household.

In the ENEMDU, all members of a dwelling are interviewed and so all the members of a dwelling form a cluster. This means a potentially positive correlation in their answers to the questionnaire. This would imply greater standard errors in the estimated coefficients than if the clustered sampling design was not taken into consideration.

The data set comprises 110,283 individuals and 313 variables. Only the subset of individuals that were employed at the time of the survey was selected. This is a subsample of 31,057 workers. The data set contained the following information about each worker: The employee's age *(Age)*, workers' seniority in the current job measured in number of years *(Seniority)*, the employee's gender *(Men)*, the employee's type of workplace zone *(Urban)*, the employee's marital status *(Single, Married, Other)*, whether the employee has a workplace safety training or not *workplace safety training*, the employee's number of working hours per week *(Working hours)*.

Table 3.1 shows the descriptive statistics of this data set. Overall, employees who declared that they had suffered a workplace accident represent 3.11% of the total, which means the occurrence of such events is quite rare. The mean age of workers who had suffered a workplace accident is 3 years more than that of those who had not suffered an accident. Among male employees, 4.09% had suffered a workplace accident, while only 1.80% of women had. Rural workers present a slightly higher rate of work-place accidents (3.28%) than urban workers (2.98%). Married employees had a higher workplace accident rate with respect to single workers and those of other marital status. Finally, the number of weekly working hours under Ecuadorian law is fixed at 40 (Art. 47 of the Ecuadorian labor code). Workers who exceed this limit by 2 hours are more likely to suffer a workplace accident than workers whose

average weekly working hours are 38.

Covariates		**No Workplace Accident (Y=0)**	**Workplace Accident (Y=1)**	**Total**
Age (years)		36.78	39.57	36.87
Seniority		8.08	9.23	8.11
Sex	Woman	13,145 (98.20%)	241(1.80%)	13,361
	Man	17,021 (95.91%)	726 (4.09%)	17,696
Area	Rural	12,252 (96.72%)	416 (3.28%)	12,634
	Urban	17,914 (97.02%)	551 (2.98%)	18,423
Marital Status	Single	9,617 (97.91%)	205 (2.09%)	9,801
	Married	17,761 (96.35%)	672 (3.65%)	18,389
	Other	2,788 (96.87%)	90 (3.13%)	2,867
Working hours		38.17	42.02	38.29
Workplace Safety Training	Yes	6,696 (94.79%)	368 (5.21%)	7,064
	No	23,396 (97.51%)	598 (2.49%)	23,993
Total		30,091 (96.89%)	966 (3.11%)	31,057

Table 3.1: Descriptive statistics of the workplace accident data set

Additionally, employees who had received workplace safety training presented a higher rate of accidents (5.21%) than employees who had not received such training (2.49%). This result may be due to the fact that workers in dangerous work places tend to receive more workplace safety training than others. Finally, the mean number of years of seniority is higher among workers who had suffered workplace accidents than those who had not.

3.4 Results

This section presents the results of the logistic regression with sampling weights and two estimated penalized logistic regression models based on weighting mechanisms $PSWa_i$ and $PSWb_i$ for each of the criterion proposed in Section 3.3.

Table 3.2 shows the predictive performance measures of three types of model: the first is a simple weighted design-based logistic regression model where only the SW_i, sampling weight mechanism is used, as well the sampling design. The

second is the model estimated using $PSWa_i$, and the third is the model estimated using $PSWb_i$. For the second and third model types, we present the first six models that best meet the ROC optimal criterion.

The results in Table 3.2 for the ROC criterion show that the adjustment parameter with the lowest ROC distance is ϵ = 0.05, a threshold Ψ = 0.03 and a sensitivity that equals 59.731%, when the weighting mechanism $PSWa_i$ is used in the predictive modelling. The lowest ROC distance when $PSWb_i$ is used is obtained for the adjustment parameter ϵ = 0.6, a threshold Ψ = 0.03 and a sensitivity that equals 59.834%.

Statistical predictive performance of the weighted design-based model						
	Sensitivity (%)	**Specificity (%)**	**Accuracy (%)**	**ROC criterion (distance)**		Ψ
	56.522	66.458	66.114	0.549		0.03
Statistical predictive performance measures obtained using $PSWa$						
Order	**Sensitivity (%)**	**Specificity (%)**	**Accuracy (%)**	**ROC criterion (distance)**	ϵ	Ψ
1º	59.731	63.743	63.619	0.542	0.05	0.03
2º	59.524	63.966	63.828	0.542	0.00	0.03
3º	**60.870**	**62.354**	**62.308**	**0.543**	**0.40**	**0.03**
4º	60.663	62.471	62.414	0.544	0.35	0.03
5º	59.110	64.145	63.989	0.544	-0.10	0.03
6º	59.938	63.215	63.113	0.544	0.15	0.03
Statistical predictive performance measures obtained using $PSWb$						
Order	**Sensitivity (%)**	**Specificity (%)**	**Accuracy (%)**	**ROC criterion (distance)**	ϵ	Ψ
1º	**59.834**	**63.923**	**63.796**	**0.540**	**0.60**	**0.03**
2º	59.524	63.999	63.860	0.542	-0.30	0.03
3º	59.524	63.999	63.860	0.542	-0.25	0.03
4º	59.524	63.993	63.854	0.542	-0.75	0.03
5º	59.524	63.993	63.854	0.542	-0.70	0.03
6º	59.524	63.993	63.854	0.542	-0.65	0.03

Models that meet the C-ROC criterion are bold character when only the first six models are considered.

Table 3.2: Statistical Predictive Performance Measures

Figures 3.1 and 3.2 show the ROC representation of all possible models based on weighting alternatives a and b respectively; thus, every dot represents a model. When $PSWa_i$ is used under the C-ROC criterion the best adjustment parameter is ϵ = 0.4 and a threshold $\Psi = 0.03$, being among the six best models according to the ROC criterion. In this case, the highest sensitivity value is 60.87%. Note we ignore the first two models with a better ranking under the ROC criterion because of their lower sensitivity values (59.731% and 59.524%, respectively). When $PSWb_i$ is used, the best sensitivity of the six models corresponds to an adjustment parameter $\epsilon = 0.6$ and a threshold $\Psi = 0.03$. Here the ROC criterion leads to a highest sensitivity value of 59.834%.

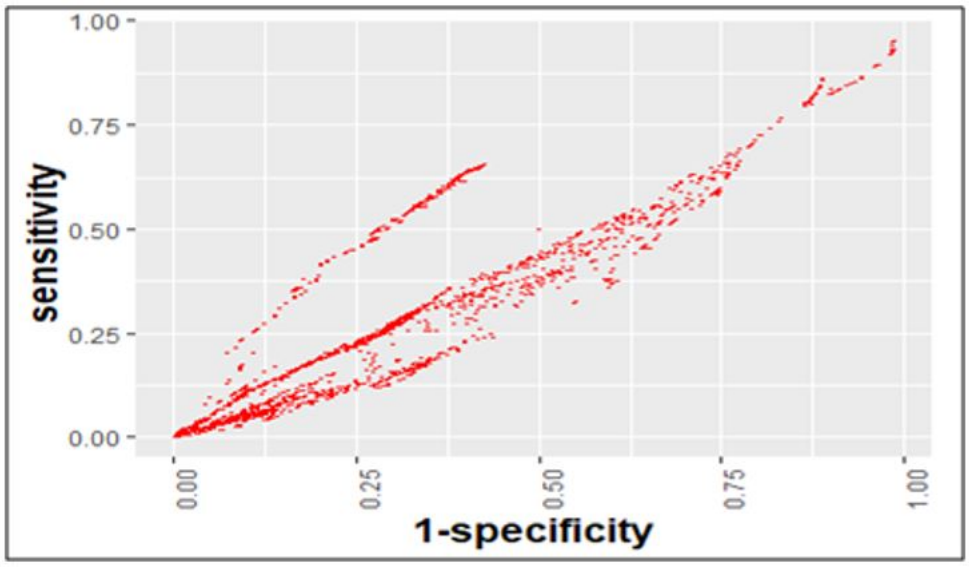

Figure 3.1: The classification performance (sensibility and 1-specificity) of the estimated weighted model with $PSWa_i$ when ϵ varies.

Note that the adjustment parameter ϵ is jointly chosen with Ψ (among all the possible values for Ψ). All the optimal combinations have a threshold Ψ = 0.03 in the subset of models obtained when using $PSWb_i$ and $PSWa_i$, even when all other possibilities were considered. In the weighted design-based logistic regression model (first row of Table 3.3), a threshold Ψ = 0.03 was set because this value is the mean of the dependent variable.

Thus, having selected the best adjustment parameters and thresholds that fulfil the proposed C-ROC criterion when using $PSWa_i$ and $PSWb_i$, we can conclude that the $PSWa$ with $\epsilon = 0.1$ and $\Psi = 0.03$ has the highest sensitivity and, thus, gives the best predictive performance in terms of the ROC criterion.

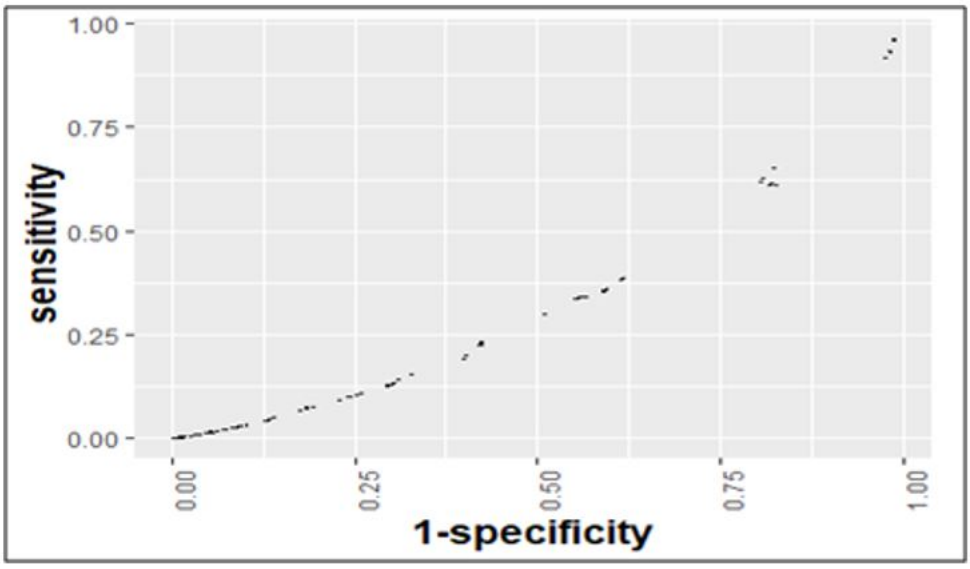

Figure 3.2: The classification performance (sensibility and 1-specificity) of the estimated weighted model with $PSWb_i$ when ϵ varies.

In Table 3.3, the RMSE was calculated for the lowest (RMSE1) to the highest (RMSE10) decile of predictions based on the best adjustment parameters under the C-ROC criterion solely for employees that had suffered a workplace accident ($Y_i = 1$).

	Intervals	**Weighted design-based model**	**PSWa** ($\epsilon = 0.4$ and $\Psi = 0.03$)	**PSWb** ($\epsilon = 0.6$ and $\Psi = 0.03$)
RMSE 1	[0.005;0.012]	0.99039	0.99008	0.99046
RMSE 2	(0.012;0.015]	0.98674	0.98643	0.98674
RMSE 3	(0.015;0.018]	0.98372	0.98341	0.98370
RMSE 4	(0.018;0.021]	0.98080	0.98006	0.98099
RMSE 5	(0.021;0.025]	0.97698	0.97386	0.97707
RMSE 6	(0.025;0.029]	0.97255	0.97152	0.97269
RMSE 7	(0.029;0.034]	0.96890	0.96908	0.96909
RMSE 8	(0.034;0.041]	0.96260	0.95959	0.96226
RMSE 9	(0.041;0.057]	0.95190	0.94667	0.95105
RMSE 10	(0.057;0.163]	0.92421	0.91959	0.92285

Table 3.3: RMSE results of the estimated models when $Y_i = 1$

Under RMSE criterion, the model estimated using $PSWa_i$, has smaller RMSE

values than those of the other two models in Table 3.3. Although the improvement appears quite small, it is important to note that in this example only 3.11% of employees suffered an accident, which means this event is extremely rare. When we improve the sensitivity by only a few percentage points we obtain a significant impact on the global prediction performance, as events classed as workplace accidents might be hard to predict.

Taking all the results from the previous criteria, the weighting mechanism $PSWa_i$ is the best in terms of improving a model's predictive performance. This does not mean that $PSWb_i$ is not a suitable weighting mechanism; but, due to the type of exogenous variables in the model and the frequency of the dependent variable, $PSWa_i$ is more effective.

Figures 3.3, 3.4, and 3.5 show the predictions of the workplace accident and no workplace accident observations for each model (weighted design-based model, alternative a and alternative b with their optimal ϵ and Ψ). The proposed weighting mechanisms improve the predictive performance without producing abrupt or incoherent results. This outcome is also supported in Appendix (Table 3.4), where the model parameter estimates are presented. In fact, all three figures seem to have a similar density distribution.

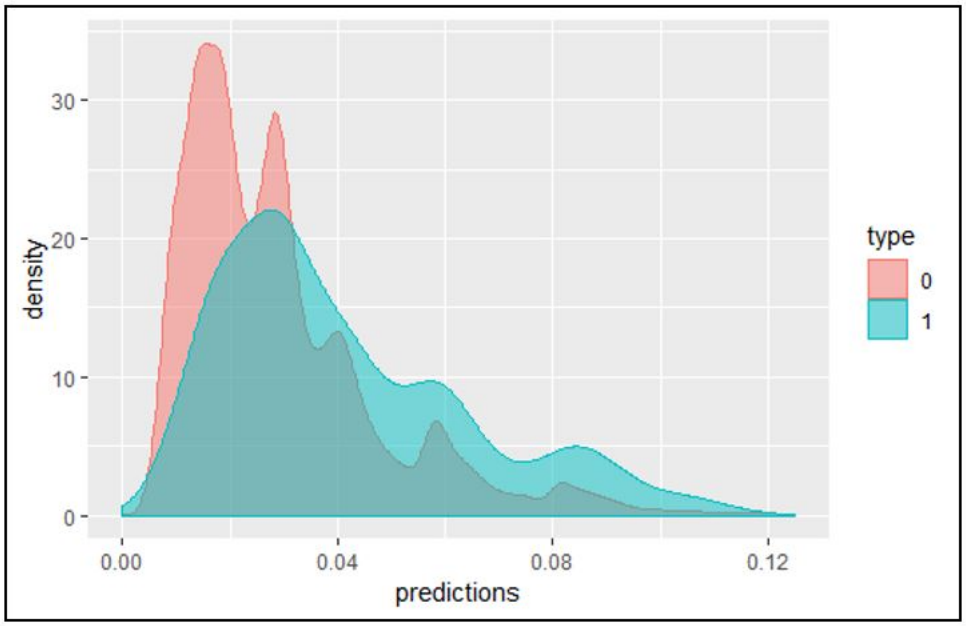

Figure 3.3: Predictions obtained by the weighted model colored by $Y_i = 1$ and $Y_i = 0$.

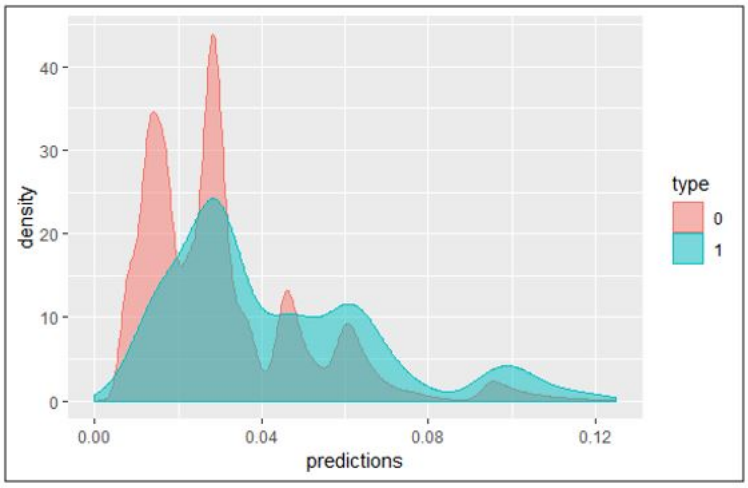

Figure 3.4: Predictions obtained by the weighted model with $PSWa$ ($\epsilon = 0.4$) colored by $Y_i = 1$ and $Y_i = 0$.

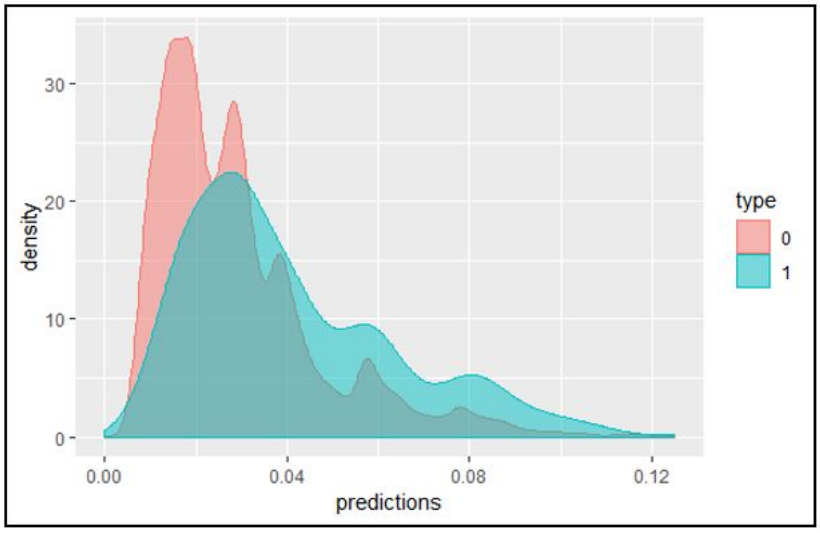

Figure 3.5: Predictions obtained by the weighted model with $PSWb$ ($\epsilon = 0.6$) colored by $Y_i = 1$ and $Y_i = 0$.

Figure 3.6 presents the histogram of predictions for observations that are equal to 1 for all three methods. Alternatives a (pink) and b (green) are located to the right of the histogram of predictions for the weighted design-based model (blue). It seems that alternatives a and b have a greater frequency of predictions equal to 1 for the observations that lie closer to the mean (0.031) and to the right of the figure. This seems to indicate that the predictive performance is improved, in the sense that it is more likely to detect cases $Y_i = 1$ under alternative a than it is in the other cases.

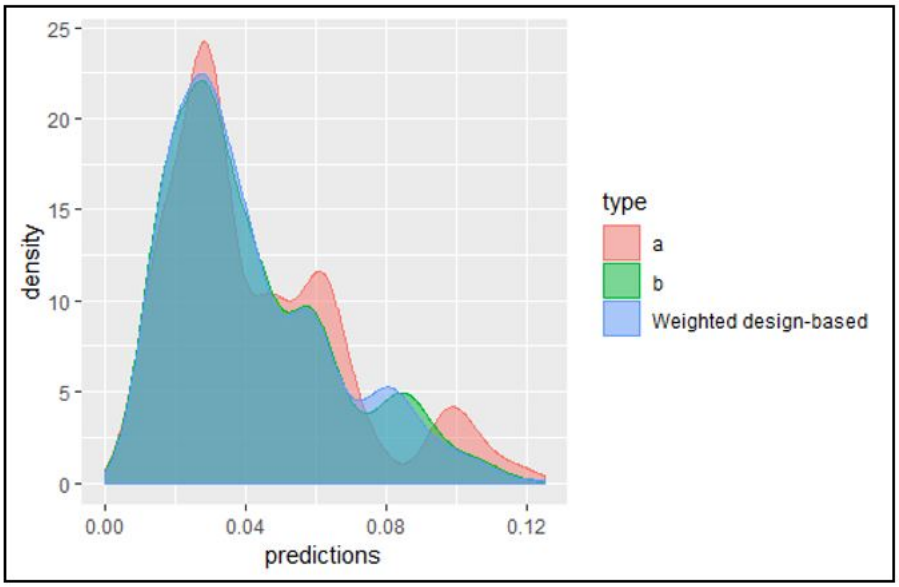

Figure 3.6: Predictions for the observations that are equal to 1 of the unweighted model, alternatives a and b.

3.5 Conclusions

The main conclusion is that the methods proposed can improve the predictive performance of logistic regression classifiers in survey data and that this is specially so for most deciles of the predictive distribution. This chapter has compared two weighted procedures with the baseline model and shown that the choice of a specific weighting parameter, together with that of the threshold, leads to better accuracy than that obtained with the weighted design-based logistic regression model.

Moreover, it has been proposed the ROC optimal criterion and the C-ROC optimal criterion as alternatives for measuring the predictive performance of a weighted estimation. Their standard procedures can be replicated in similar cases that seek to predict rare binary events.

This chapter has found evidence that predicting the outcome response for respondents of a survey asked whether or not they had suffered a workplace accident can be improved for these individuals in all deciles of the prediction. This means that $PSWa$ is able to predict individuals whose characteristics lie farther from the mean values. This result shows that the discrimination capacity can be improved by underweighting or overweighting observations, even if they already carry a sample weight.

Our analysis has a number of limitations. First, we might have implemented a cross-validation exercise by leaving part of the sample out of the estimation process. In this way, we could then have tested the model performance on a test sample; however, the proportion of ones in the dependent variable is so small that the test sample presents a serious lack of events (employees with accidents). Second, we deal here with a phenomenon that has a very low frequency because only a small fraction of the respondents suffered a workplace accident. We won-der if the results might differ when analyzing phenomena that are more frequent. However, the method described shows that the score (probability of a response equal to 1) obtained under alternative a or b provides an index of risk which gives more accurate predictions for workers and that it can serve as a measure of workplace safety. In short, our method can be used to identify those workers at greatest risk of suffering an accident in the workplace.

Further research needs to be dedicated to the definition of combined weights. Here, we have proposed multiplying sampling weights with predictive weights with a previous rescaling. Other alternatives, such as standardization or geometrical averaging, could also be explored.

Appendix A

Table 3.4 shows the results of the parameter and standard error estimates from the three logistic regression models (weighted design-based standard errors, weighted with $PSWa$ and weighted with $PSWb$). The results show that the coefficients of the weighted a and b models only change slightly with respect to the base weighted model. Standard errors, which are all design-based, are also similar.

Only the conclusion regarding the significant influence of the number of working days would differ if the PSWa weight were implemented.
In this case, we would conclude, therefore, that working hours do not have a significant effect on the probability of suffering a workplace accident.

Variables	Weighted		PSWa		PSWb	
Intercept	-3.422	***	-2.88	***	-3.409	***
	(0.291)		(0.39)		(0.282)	
Urban	-0.338	***	-0.496	***	-0.371	***
	(0.095)		(0.123)		(0.1)	
Man	0.678	**	0.772	***	0.696	***
	(0.203)		(0.168)		(0.193)	
Marital (married)	0.428	*	0.525	***	0.457	**
	(0.165)		(0.137)		(0.153)	
Marital (others)	0.256		0.31		0.311	
	(0.251)		(0.278)		(0.29)	
Working hours	0.014	**	0.002		0.014	***
	(0.005)		(0.003)		(0.004)	
Workplace safety training	-0.741	***	-0.78	***	-0.748	***
	(0.114)		(0.163)		(0.115)	
Seniority	0.008		0.01		0.007	
	(0.006)		(0.007)		(0.005)	

The standard errors are shown in parentheses, and the significance of coefficients is given as follows: . ,, **, *** correspond respectively, to the 0.05, 0.01, 0.001, 0 levels of significance.*

Table 3.4: Final results of the estimates from the unweighted model, the model weighted with PSWa (ϵ = 0.4 and ψ =0.03) and the model weighted with PSWb (ϵ = -0.25 and ψ =0.03)

Chapter 4: Review of trials of boosting-based algorithms with telematics data

4.1 Introduction

Predicting the occurrence of accident claims in insurance economics lies at the heart of premium calculation, but with the development of new artificial intelligence methods, the question of choosing a suitable model has yet to be completely solved. In this chapter, the recently machine learning methods and classical models are considered and compared regarding their predictive performance in a sample of policy holders, along with their telematic information.

The advantages and disadvantages of the various methods are discussed, and this study showed that a slightly improved predictive power is obtained with some modern boosting-based or tree-based algorithms, but this has complicated the interpretation of the impact of covariates on the expected response. In the case of automobile insurance, where the premium calculation is regulated and has to be fully specified, the weight of each risk factor in the final price needs to be disclosed and the connection between the observed covariate value and the estimated probability of a claim needs to be shown. If these conditions are not met, the regulating authority may deny the insurance company the right to commercialize that product.

This chapter is an extended version of Pesantez-Narvaez, J., Guillen, M., & Alcañiz, M. (2019). Predicting motor insurance claims using telematics data—XGBoost versus logistic regression. *Risks*, 7(2), 70. Additionally, some extracts were taken from Guillen, M., & Pesantez-Narvaez, J. (2018). Machine Learning and Predictive Modeling for Automobile Insurance Pricing. *Anales del Instituto de Actuarios Españoles*, (24), 123-147.
There is some overlapping in the description of the logistic regression with the previous chapters, but I present it to keep self-contributed.

This study discussed, nevertheless, why the use of some famous boosting-based algorithms remain interesting for actuaries and how methods both old and new might be combined for optimum results. Additionally, this study presented a detailed list of trials of modified machine learning methods, in particular, boosting-based algorithms.

To compare the various competing methods or algorithms, a real dataset comprising of motor insurance policy holders and their telematics measurements were used, that is, real-time driving information collected and stored via telecommunication devices. More specifically, GPS-based technology captures an insured's driving behavior patterns, including distance travelled, driving schedules, and driving speed, among many others. Here, pay-as-you-drive (PAYD) insurance schemes represent an alternative method for pricing premiums based on personal mileage travelled and driving behaviors. Guillen et al. (2019); Roel et al. (2018) and Perez-Marin and Guillen (2019) showed the potential benefits of analyzing telematics information when calculating insurance premiums. Gao and Wuthrich (2019) analyzed high-frequency GPS location data (second per second) of individual car drivers and trips. Gao and Wuthrich (2018) and Gao et al. (2019) investigated the predictive power of covariates extracted from telematics car driving data using the speed-acceleration heatmaps proposed by (Wüthrich, 2017). Further, Hultkrantz et al. (2012) highlighted the importance of PAYD insurance plans insofar as they allow insurance companies to personalize premium calculation and, so, charge fairer rates.

The rest of this chapter is organized as follows. First, a literature review is presented. Second, the notation is introduced and various methods are outlined. Third, the dataset is described and some descriptive statistics are provided. Fourth, the results of our comparisons in both a training and a testing sample are reported. Finally, following the conclusion, some practical suggestions are offered about the feasibility of applying new machine learning methods to the field of insurance economics.

4.2 Literature Review

There is a vast literature devoted to addressing new theoretical and experimental modeling problems driven by the recent explosion of big data obtained by emerging technologies. The increasing number of big data problems placed statistical learning theory as a very demanding field in many scientific and business disci-

plines. Statistical learning aims to build a predictive function based on data rather than incorporating probability distributions into the model of a phenomenon such like classical probabilistic models do. As a result, statistical learning techniques are more efficient dealing with complex data sets.

In some way, statistical learning is the result of the historical evolution of statistical models. The data-generating processes began with the appearance of linear regression models for astronomical data officially published by Legendre in 1805, but co-credited to Gauss in 1795 according to (Seal, 1967). In 1963, Tikhonov (1963) proposed the Rigde regression used as a regularization method. Later, generalized linear models were proposed by (Nelder and Wedderburn, 1972). Both of them pursued to fit linear relationships between the covariates and the target variable (also known as response variable). In 1986, Hastie and Tibshirani (1986) proposed the generalized additive models in order to fit the non-linearities with non-parametric methods. In 1996, Tibshirani (2011) proposed the Lasso (Least absolute shrinkage and selection operator) regression that performs both variable selection and regularization in order to improve the predictive capacity of the statistical model it produces.

Statistical learning theory is also a framework for machine learning that combines fields of statistics and functional analysis to solve prediction problems based on data. The so-called machine learning is a recent technology, application of artificial intelligence. Guillen and Pesantez-Narvaez (2018) highlighted the Turing Test, created by Alan Turing in 1950, to chronologically place the beginning of the era of artificial intelligence, since it is the moment when a process can contrast if a human interacts with another human or a computer. Ever since, computing has been developed at a great speed, fact that impedes to provide a detailed report of the various milestones up to today (Turing, 2009). Some of the principal advances include, in 1957 the invention of the first artificial neural network by Frank Rosenblatt, and in 1967, the proposal of the nearest neighbour algorithm as a way to interpolate or to complete missing information with a specific metric in a multidimensional space (Weinberger and Saul, 2009). In the following decades, bunch of efforts were focus on robotics and natural language processing development. However, it is from the 90's when programs began be created by computers themselves to analyse large amounts of data and to draw conclusions automatically, or put it in other words, "learn" from data to reveal results.

As a result, a parallel language is created aside statistics and traditional econometrics that not only persists, but it is also gaining awareness. It even establishes

that the classical statistical models constitute the first vestiges of machine learning, since it is said, ample, that the parameter estimation of a linear model is actually an automated computation that allows finding patterns in the data. In 2006, Geoffrey Hinton coined the term "deep learning" to explain new algorithms for object and image recognition. In the following decade, although most of the advances occurred in this same area, platforms appeared that allow the implementation of artificial intelligence methods in distributed machines, in order to facilitate the treatment of large databases by segmenting the problem of having excessive volume of data for a single computer at sub-stream resolution on several different computers.

The machine learning methods are classified into three groups: supervised learning, unsupervised learning, and reinforcement learning. Alpaydin (2004) wrote one of the first books of this field unifying in a single common framework the problems and solutions of machine learning. In particular, the supervised learning methods such as decision trees, artificial neural networks or the support vector machine, learning is done through analysis of the past, trying to reproduce the response or how to anticipate what has happened. Kotsiantis et al. (2007) made one of the first reviews of supervised classification methods.

In the case of insurance economics, analysts who use supervised methods are requested to choose a training sample where the basis for prediction formulation are established. This sample may consist, for example, of observing the accident rate that the insured have suffered in previous years. By comparing the response predicted by some supervised method with the one observed, it can be decided which of the methods produces less error. Later, a testing sample is used to assess whether the method is also the best with some data points that were not used in the first phase. As in predictive models, one starts from a set of policies with some known characteristics, for example, whether a policy holder have had an accident claim.

With the supervised learning system, a way of predicting whether or not someone has had an accident is created, as it would be done for example with a classic logistic regression model, and then one compares the observed values with the predicted values of the model. It is concluded that the algorithm that achieves the most hits, is better. In this sense, it should be noted that errors can be weighted differently, if they go one way or the other. That is, when the model predicts that there is a claim and none has been declared, or, on the contrary, when the model predicts that there is no claim and there has been.

The unsupervised learning methods pursue to find structures with similar patterns

among the observations analysed such as the premiums. Once the subsets are found, they can be treated in the same way, for example, pricing the same premium to a group of policy holders. Regarding the reinforcement learning, Sutton et al. (1998) highlighted the use of these methods to carry out specific actions while optimizing a final benefit. To do this, past experience is analysed, and the best capture of reality is taken to make decisions, for example, by customers' backlash when receiving a discount in order to apply it later in similar policy renewal operations.

4.3 Methodology

In a data set of n individuals and P covariates, there is a binary response variable Y_i, $i = 1, \ldots, n$ taking values $\{0, 1\}$; and a set of covariates denoted as X_{ip}, $p = 1, \ldots, P$. The conditional probability density function of $Y_i = t$ $(t = 0, 1)$ given X_i $(X_{i1}, \ldots, X_{iP})$, is denoted as $h_t\,(X_i)$. Equivalently, it can be said that $\pi\,(Y_i = t) = h_t$, and that $E\,(Y_i) = \pi\,(Y_i = 1) = h_1\,(X_i)$.

4.3.1 Machine learning algorithms

Logistic Regression: This is a technique borrowed by machine learning from the field of statistics since several decades and is the basis for many applied areas such as drug testing, credit scoring, fraud analysis and any classification problem at hand. Nowadays, it is the benchmark method for binary classification problems (problems with two class values). Logistic regression operates in a similar fashion to linear regression by finding the values for the coefficients that weight each input variable. Unlike linear regression, the prediction for the output is transformed using a non-linear function so that the results can be interpreted as a score or a predicted probability. Classification predictions, confusion matrices and the ROC curve help to interpret the implementation of the results. Unfortunately, little has been said about outliers in this context and the role of extreme observations.

The logistic regression uses the logit function as a canonical link function, in other words, the log ratio of the probability functions $h_t\,(X_i)$ is a linear function of X; that is:

$$
\begin{aligned}
ln\frac{h_1\left(X_i\right)}{h_0\left(X_i\right)} &= ln\left(\frac{\pi_i}{1-\pi_i}\right) \\
&= \beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p
\end{aligned}
\tag{4.1}
$$

where β_0, β_1,..., β_p are the model coefficients, and $\pi\left(Y_i = 1\right)$ is the probability of observing the event in the response (response equal to 1), and $\pi\left(Y_i = 0\right)$ is the probability of not observing the event in the response (response equal to 0).

The link function provides the relationship between the linear predictor $g\left(\pi\right)$=β_0 +$\sum_{p=1}^{P} X_{ip}\beta_p$ and the mean of the response given certain covariates. In a logistic regression model, the expected response is:

$$
\begin{aligned}
&E\left(Y_i\right) = \pi\left(Y_i = 1\right) \\
&= \frac{e^{\left(\beta_0+\sum_{p=1}^{P} X_{ip}\beta_p\right)}}{1+e^{\left(\beta_0+\sum_{p=1}^{P} X_{ip}\beta_p\right)}}
\end{aligned}
\tag{4.2}
$$

A logistic regression can be estimated by the maximum likelihood (for further details see, for example, (Greene, 2003)). Therefore, the idea underlying a logistic regression model is that there must be a linear combination of risk factors that is related to the probability of observing an event. The data analyst's task is to find the fitted coefficients that best estimate the linear combination in (4.2) and to interpret the relationship between the covariates and the expected response. In a logistic regression model, a positive estimated coefficient indicates a positive association. Thus, when the corresponding covariate increases, the probability of the event response also increases. If the estimated coefficient is negative, then the association is negative and, therefore, the probability of the event decreases when the observed value of the corresponding covariate increases. Odds-ratios can be calculated as the exponential values of the fitted coefficients and they can also be directly interpreted as the change in odds when the corresponding factor increases by one unit.

Apart from their interpretability, the popularity of logistic regression models is based on two characteristics: (i) The maximum likelihood estimates are easily found; and (ii) the analytical form of the link function in (2) always provides predictions between 0 and 1 that can be directly interpreted as the event probability estimate. For these motives, logistic regression has become one of the most popular classifiers, their results providing a straightforward method for predicting scores or

propensity values which, in turn, allow new observations to be classified to one of the two classes in the response. For R users, the *glm* function is the most widely used procedure for obtaining coefficient estimates and their standard errors, but alternatively, a simple optimization routine can easily be implemented.

Support Vector Machine (SVM): Support Vector Machines are machine learning algorithms that find a hyperplane that splits the input variable space as to separate the best the points in the input variable space, using their class, either $Y_i = 1$ or $Y_i = -1$. Then a separating hyperplane has the property that:

$$\xi_0 + \xi_1 X_1 + \xi_2 X_2 + ... + \xi_P X_P < 0 \tag{4.3}$$

and,

$$\xi_0 + \xi_1 X_1 + \xi_2 X_2 + ... + \xi_P X_P > 0 \tag{4.4}$$

Equivalently,

$$Y_i \left(\xi_0 + \xi_1 X_1 + \xi_2 X_2 + ... + \xi_P X_P\right) > 0 \tag{4.5}$$

for parameters $\xi_0, \xi_1, ..., \xi_P$. Note that if (4.5) would have been an hyperplane of two-dimensional space $Y_i \left(\xi_0 + \xi_1 X_1 + \xi_2 X_2\right)$=0, it would be simply the equation of a line, that separates one region from another, as shown in the Figure 4.1 below:

The SVM learning algorithm finds the coefficients that result in the best separation of the classes by the hyperplane. The distance between the hyperplane and the closest data points is referred to as the *margin* M. The best or optimal hyperplane that can separate the two classes is the line that has the largest margin. Only these points are relevant in defining the hyperplane and in the construction of the classifier. These points are called the support vectors. They support or define the hyperplane. In practice, an optimization algorithm is used to find the values for the coefficients that optimize the margin.

$$\begin{aligned} \min_{\xi_0, \xi_1, .., \xi_P, M} \quad & M \\ \text{s.t.} \quad & \sum_{p=1}^{P} \xi^2 = 1 \\ & Y_i \left(\xi_0 + \xi_1 X_1 + \xi_2 X_2 + ... + \xi_P X_P\right) \leq M \end{aligned} \tag{4.6}$$

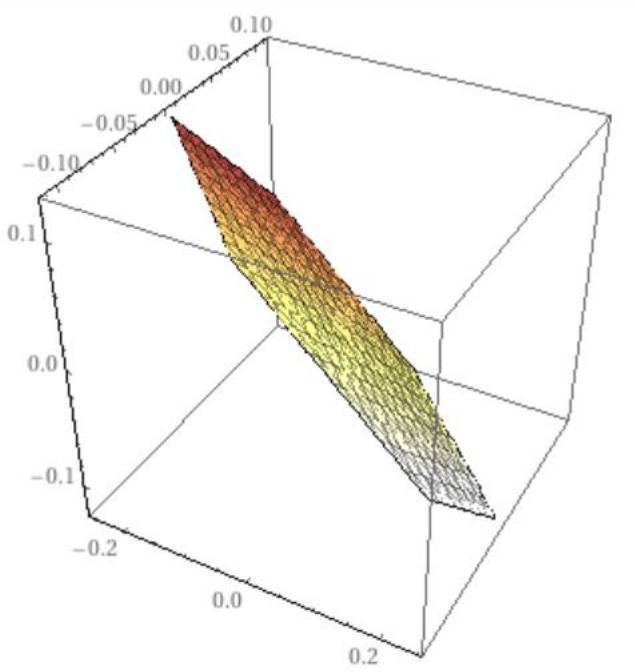

Figure 4.1: Illustrative representation of Y_i ($\xi_0 + \xi_1 X_1 + \xi_2 X_2$)=0 in three dimensional space.

Tree-Based Models: As classifiers, trees are an important type of algorithm for predictive modeling machine learning. The reason is that these techniques admit a graphical representation of the tree model as a binary tree or categorical tree, that show how data are split step by step, why they are so and which the sequential order of the organized sorting is. Each node represents a single input variable and a split point on that variable when the variable is quantitative. The leaf nodes of the tree contain an output variable which is used to make the prediction. Large values and extremes cause an enormous instability on the construction of trees, something that can be assessed via bootstrapping and that has given rise to some of the more sophisticated methods described below.

Tree-based models where the target variable is categorical are called classification trees, and when target variable takes numeric values are called regression trees.

Each tree-based model has:

- Internal decision node (non-leaf): It is labeled with an input feature, in other words, each node is associated to one of the covariates, and it may have two or more branches that come out of it. Each node represents all possible values

or categories that these covariate may take.

- Leaves: Each leaf represents a value of the target variable given the values of the input variables represented by the path from the root to the leaf.

The most primitive tree-based models were decision trees. They stratify a space of covariates into some simple regions. Actually, trees get branched from top to bottom, beginning with the decision nodes associated to the most influential features (close to the root), until the least influential ones. For example if there are two input variables: gender (female/male), and satisfied (Yes/No), where gender is the most important one on the final prediction of the variable: insurance plan purchase (Yes/No). The tree will start to split with the decision node "gender", and then will continue with the decision node "satisfied".

On the other hand, classification trees have leaves that represent class labels and branches represent conjunctions of features or covariates that lead to those class labels. In the top-down splitting procedure, the most influential or important features are considered the first. There are three proposed metrics to measure the quality of the split:

- Information Gain
- Gini Impurity
- Chi Square

Information Gain: It measures the gain of each split in each step of the branching. It orders decision nodes associated to each feature, then the decision node with the highest information gain will split the first, and so on until the one with fewest information gain. The information gain is the capacity of mapping each observation into the correct leaf (final prediction).

In order to measure the information gain, we first introduce the definition of entropy by (Shannon, 1948). The information entropy measures how much information there is in an event. Let's take the following two scenarios:

- In a sample completely homogeneous (where all cases are classified equally), we have minimum uncertainty about the classification of the observations. One may choose any observation, and will know a priori the result of the classification. In this case the entropy is zero.
- In a sample equally distributed (where each classification has the same number of observations), one have a maximum uncertainty in the sense that it is worst scenario to detect a priori where an observation will be classified. The entropy is one.

Thus, the entropy (S) measures the uncertainty of a system. Shannon (1948) expressed it mathematically as:

$$E(S) = \sum_{i=1}^{C} P_i log P_i, \tag{4.7}$$

where S is the sample of observations (analyzed system), C is the number of class labels of the feature, and P_i is the proportion of the class label corresponding to each observation i. There are 4 steps to obtain the information gain of a decision node:

1. Calculate the entropy $E(S)$ taking the target variable.
2. Calculate the entropy of each branch, and then sum proportionally the branches to calculate the total entropy $E(T, X)$.

$$E(Y, X) = \sum_{c \in X}^{C} P(c) E(S_c), \tag{4.8}$$

3. Obtain the information gain by subtracting (4.8) of (4.7).

$$Gain(Y, X) = E(Y) - E(Y, X), \tag{4.9}$$

4. Select the attribute with more gain as decision node for that split.

Gini Impurity: It measures the likelihood of incorrect classification of a new observation given a random variable, if that new instance was randomly classified according to the distribution of class labels from the data set.

Gini impurity G(S) is lower bounded by 0, and it occurs when the data set contains only one class.

The formula for calculating the Gini impurity of a data set or feature is as follows:

$$G(S) = \sum_{i=1}^{C} \pi_i * (1 - \pi_i), \tag{4.10}$$

The branching process is done similarly to how information gain was calculated for entropy, instead of taking a weighted sum of the entropies of each branch of a decision tree, one take a weighted sum of the Gini impurity.

Chi square: It finds out the statistical significance among the sub-nodes according to their corresponding parent nodes. It is the sum of squares of standardised differences between observed and expected frequencies of target variable. The Chi

square method works with categorical target variables, and can build two or more splits. The higher the value of Chi-Square higher the statistical significance of differences between sub-node and parent node.

Chi-Square of each node is calculated using the following formula:

$$Chi - square = \frac{\frac{(ActualValue - ExpectedValue)^2}{ExpectedValue}}{2} \tag{4.11}$$

On the other hand, *regression trees* are used when a target variable is continuous. The metric used to measure the quality of the split for a continuous target variable is the reduction of variance. It is an algorithm that uses the standard formula of variance to choose the best split. The split with lower variance is selected as the criteria to split all data points.

Naïve Bayes: Naïve Bayes works as a straightforward predictive modeling procedure. The model is comprised of two types of probabilities that can be calculated directly from training data: i) The probability of each class and ii) the conditional probability for each class given a predictor value. Once calculated, the probability model can be used to make predictions for new data using Bayes' Theorem:

$$\pi\left(C_p|X\right) = \frac{\pi\left(C_p\right)\pi\left(X|C_p\right)}{\pi\left(X\right)} \tag{4.12}$$

for each p possible outcomes or classes C_p.

Assuming that each input variable is independent and normally distributed is a strong assumption and it is quite unfeasible for real data, nevertheless, the technique is very effective on a large range of complex problems.

K-Nearest Neighbors Algorithm: Predictions on the *KNN* method are made for a new data point by searching through the entire training set for the K most similar instances (the neighbors) and summarizing the output variable for those K instances. For regression problems, this might be the mean output variable, for classification problems this might be the mode (or most common) class value. The cornerstone of this approach is how to determine the distance value between the data instances. The simplest approach is to find the Euclidean distance, but this can require a lot of memory or space to store all of the data similarities and must be updated as a new instance comes in. The idea of distance or closeness is challenged by very high dimensions (lots of input variables) which can negatively affect

the performance of the algorithm. This is called the curse of dimensionality, which suggests to use those input variables that are most relevant to predicting the output variable. Naturally, the role of extremes is also relevant in this technique.

Random Forest: Random Forest is a powerful machine learning algorithm that relies on Bootstrap Aggregation or bagging. The bootstrap takes samples of the data, calculates the value of interest, and then averages all of the values to give a better estimation of the true value.

AdaBoost: Boosting is an ensemble technique that attempts to create a strong classifier from a number of weak classifiers. The way it works is by building a model from the training data, then creating a second model that attempts to correct the errors from the first model. Models are added until the training set is predicted perfectly or a maximum number of models are added. AdaBoost was the first really successful boosting algorithm developed for binary classification.

The AdaBoost was first developed by (Schapire and Freund, 2012). The AdaBoost trains weak classifiers or predictors $\hat{Y}_i$ on D weighted versions of the training sample giving higher weight to cases that are currently misclassified. This is done for a sequence of weighted samples. The final predictor $\hat{Y}_i^D$ is defined to be a linear combination of the classifiers from each stage.

1. Set initial weights $w_i = \frac{1}{n}, i = 1, ..., n$.

2. Repeat for $d = 1, ..., D$:

   - Fit the classifier $\hat{Y}_i$ using weights on the training data X_{ip}.
   - Computer the error (err) in each d iteration:

$$err^d = E_w \left[\mathbf{1}_{Y_i \neq \hat{Y}_i} \right], \tag{4.13}$$

   - Compute a constant $\hat{C}$ that adjusts the weighting in each d iteration:

$$\hat{C}^d = log \frac{1 - err^d}{err^d}, \tag{4.14}$$

   - Set $w_i \leftarrow w_i exp[\hat{C}^d \mathbf{1}_{Y_i \neq \hat{Y}_i}]$, and renormalize so that $\sum_i w_i = 1$.

3. Output the final classifier $\hat{Y_i}^D$:

$$\hat{Y_i}^D = sign\left[\sum_{d=1}^{D} \hat{C}^{(d)} \hat{Y}_i^d\right]. \tag{4.15}$$

XGBoost:

Chen and Guestrin (2016) proposed the XGBoost as an alternative method for predicting a response variable given certain covariates. The main idea underpinning this algorithm is that it builds D classification and regression trees (or CARTs) one by one, so that each subsequent model (tree) is trained using the residuals of the previous tree. In other words, the new model corrects the errors made by the previously trained tree and then predicts the outcome.
In the XGBoost, each ensemble model uses the sum of D functions to predict the output:

$$\hat{Y_i} = \Gamma\left(X_i\right) = \sum_{d=1}^{D} f_d\left(X_i\right), \qquad f_d \in \Gamma, i = 1, ...n. \tag{4.16}$$

where Γ is the function space of the CART models, and each f_d corresponds to an independent CART structure which we denote as q. In other words, q is the set of rules of an independent CART that classifies each individual i into one leaf. The training phase involves classifying n observations so that, given the covariates X, each leaf has a score that corresponds to the proportion of cases which are classified into the response event for that combination of X_i. We denote this score as w_q.

Thus, we can write q as a function $q : \mathfrak{R}^P \rightarrow T$, where T is the total number of leafs of a tree and j is later used to denote a particular leaf, $j = 1, \ldots, T$. To calculate the final prediction for each individual, we sum the score of the leafs as in (4.16), where $\Gamma = f(X) = w_q(X)$, with $q : \mathfrak{R} \rightarrow T$, and $w \in \mathfrak{R}^T$.

The XGBoost method minimizes a regularized objective function, i.e. the loss function plus the regularization term:

$$\varphi = \sum_{i=1}^{n} \ell(Y_i, \hat{Y_i}) + \sum_{d=1}^{D} \iota(f_d), \tag{4.17}$$

where ℓ is a convex loss function that measures the difference between the observed response Y_i and predicted response $\hat{Y_i}$ and $\iota = \mu T + \frac{1}{2}\lambda\|w\|_2^2$, ι is the regularization term also known as the shrinkage penalty which penalizes the complexity of the model and avoids the problem of overfitting.

The intuition underpinning the regularization proposed in (4.17) involves reducing the magnitude of w, so that the procedure can avoid the problem of overfitting. The larger the ι, the smaller the variability of the scores (Goodfellow et al., 2016).

The objective function at the $d-th$ iteration is :

$$\varphi^d = \sum_{i=1}^{n} \ell(Y_i, \hat{Y_i}^{d-1} + f_d(X_i)) + \iota(f_d), \tag{4.18}$$

where $\hat{Y_i}^{d-1}$ is the prediction of the $i-th$ observation at the $(d-1)-th$ iteration.

Due to the non-linearities in the objective function to be minimized, the XGBoost is an algorithm that uses a second-order Taylor approximation of the objective function φ in (4.18)

$$\varphi^d \cong \sum_{i=1}^{n} \left[\left(Y_i, \hat{Y_i}^{d-1}\right) + g_i f_d\left(X_i\right) + \frac{1}{2} h_i f_d^2\left(X_i\right) \right] + \iota\left(f_d\right), \tag{4.19}$$

where $g_i = \partial_{\hat{Y_i}^{d-1}} \ell\left(Y_i, \hat{Y_i}^{d-1}\right)$, and $h_i = \partial^2_{\hat{Y_i}^{d-1}} \ell(Y_i, \hat{Y_i}^{d-1})$ denote the first and second derivatives of the loss function ℓ with respect to the component corresponding to the predicted classifier. Since we minimize (4.19) with respect to f_d, we can simplify this expression by removing constant terms as follows:

$$\varphi^d = \sum_{i=1}^{n} \left[g_i f_d(X_i) + \frac{1}{2} h_i f_d^2(X_i) \right] + \iota(f_d), \tag{4.20}$$

Substituting the shrinkage penalty of (4.18) in (4.20), we obtain:

$$\varphi^d = \sum_{i=1}^{n} \left[g_i f_d(X_i) + \frac{1}{2} h_i f_d^2(X_i) \right] + \mu T + \frac{1}{2} \lambda \|w\|_2^2. \tag{4.21}$$

The $l2$-norm shown in (4.21) is equivalent to the sum of the squared weights of all T leafs. Therefore (4.21) is expressed as:

$$\varphi^d = \sum_{i=1}^{n} \left[g_i f_d(X_i) + \frac{1}{2} h_i f_d^2(X_i) \right] + \mu T + \frac{1}{2} \lambda w_j^2. \tag{4.22}$$

Now, let us define $I_j = i|q(X_i)$, I_j is the set of observations that are classified into one leaf j, $j = 1, \ldots, T$. Each I_j receives the same leaf weight w_j. So L^d in (4.22) can also be seen as an objective function that corresponds to each set I_j. In this sense, the $f_d\left(X_i\right)$, which is assigned to the observations, corresponds to the weight w_j that is assigned to each set I_j. Therefore (4.22) is expressed as:

$$\varphi^d = \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \mu T. \tag{4.23}$$

In order to find the optimal leaf weight w_j^*, we derive (4.23) with respect to w_j, let the new equation be equal to zero, and clear the value of w_j^*. Then we obtain:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}. \tag{4.24}$$

The (4.23) was updated by replacing the new w_j^*. The next boosting iteration will minimize the following objective function:

$$\begin{aligned} \hat{\varphi}^d &= \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) \left(-\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} (h_i + \lambda)} \right) + \frac{1}{2} \left(\sum_{i \in I_j} (h_i + \lambda) \right) \left(-\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} (h_i + \lambda)} \right)^2 \right] + \mu T \\ &= -\frac{1}{2} \sum_{i=1}^{n} \frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} + \mu T. \end{aligned} \tag{4.25}$$

Once the best objective function has been defined and the optimal leaf weights assigned to I_j, we next consider what the best split procedure will be. Because (4.25) is derived for a wide range of functions, we are not able to identify all possible tree structures q in each boosting iteration. This algorithm starts by building a single leaf and continues by adding new branches. Consider the following example:
Here, I_J and I_R are the sets of observations that are in the left and right parts of a node following a split. Therefore, I=I_L+I_R.

$$\hat{\varphi}^d = \left[-\sum_{i=1}^{n} \frac{\left(\sum_{i \in I_j} g_i \right)^2}{\sum_{i \in I_j} (h_i + \lambda)} + \sum_{i=1}^{n} \frac{\left(\sum_{i \in I_L} g_i \right)^2}{\sum_{i \in I_L} (h_i + \lambda)} + \sum_{i=1}^{n} \frac{\left(\sum_{i \in I_R} g_i \right)^2}{\sum_{i \in I_R} (h_i + \lambda)} \right] - \mu, \tag{4.26}$$

$\hat{\varphi}^d$ of (4.26) is the node impurity measure, which is calculated for the P covariates. The split is determined by the maximum value of (4.26). For example, in the case of CART algorithms, the impurity measure for categorical target variables can be information gain, Gini impurity or chi-square, while for continuous target variables it can be the Gini impurity.

Once the tree f_d is completely built (i.e., its branches and leaf weights are established), observations are mapped on the tree (from the root to one corresponding

leaf). Thus, the algorithms will update from (4.18) to (4.26) as many times as D boosting iterations are established and the final classification is the sum of the D obtained functions which are shown in (4.16). Consequently, the XGBoost corrects the mistaken predictions in each iteration, as far as this is possible, and tends to overfit the data. Thus, to prevent overfitting, the regularization parameter value in the objective function is highly recommended.

4.4 Illustrative Data

The case-study database comprised of 2767 drivers under 30 years of age who underwrote a pay-as-you-drive (PAYD) policy with a Spanish insurance company. Their driving activity was recorded using a telematics system. This information was collected from 1 January through 31 December 2011. The data set contained the following information about each driver: The insured's age *(age)*, the age of the vehicle *(ageveh)* in years; the insured's gender *(male)*; the driving experience *(drivexp)* in years; the percentage of total kilometers travelled in urban areas *(pkmurb)*; the percentage of total kilometers travelled at night—that is, between midnight and 6 am *(pkmnig)*; the percentage of kilometers above the mandatory speed limits *(pkmexc)*; the total kilometers *(kmtotal)*; and, finally, the presence of an accident claim with fault *(Y)* which was coded as 1 when, at least, one claim where the fault occurred in the observational period and was reported to the insurance company, and 0 otherwise. This study is interested in predicting Y using the aforementioned covariates. This data set has been extensively studied in (Ayuso et al., 2014, 2016b,a) and (Boucher et al., 2017).

Table 4.1 shows the descriptive statistics for the accident claims data set. This highlighted that a substantial part of the sample did not suffer an accident in 2011, with just 7.05% of drivers reporting at least one accident claim. The insureds with no accident claim seemed to have travelled fewer kilometers than those presenting a claim. The non-occurrence of accident claims was also linked to a lower percentage of driving in urban areas and a lower percentage of kilometers driven above mandatory speed limits. In this dataset, 7.29% of men and 6.79% of women had an accident during the observation year.

The data set was divided randomly into a training data set of 1937 observations (75% of the total sample) and a testing data set of 830 observations (25% of the total sample). The function CreateDataPartion of R was used to maintain the same proportion of events (coded as 1) of the total sample in both the training and testing

data sets.

Variables		**Non-Occurrence of Accident Claims (Y = 0)**	**Occurrence of Accident Claims (Y = 1)**	**Total**
Age (years)		25.10	24.55	25.06
Gender	Female	1263 (93.21%)	92 (6.79%)	1355
	Male	1309 (92.71%)	103 (7.29%)	1412
Driving experience (years)		4.98	4.46	4.94
Age of vehicle (years)		6.37	6.17	6.35
Total kilometers travelled		7094.63	7634.97	7132.71
Percentage of total kilometers travelled in urban areas		24.6	26.34	24.72
Percentage of total kilometers above the mandatory speed limit		6.72	7.24	6.75
Percentage of total kilometers travelled at night		6.88	6.66	6.86
Total number of cases		2572 (92.95%)	195 (7.05%)	2767

The mean of the variables according to the occurrence and non-occurrence of accident claims. The absolute frequency and row percentage is shown for the variable gender.

Table 4.1: The description of the variables in the accident claims data set

4.5 Results

This section provides the results focused on two approaches, the first one (Section 4.5.1) consists of presenting a comparison of methods through some prediction met-

rics in a unique data set (described in Section 4.3), with the aim of detecting their general patterns of behaviour, potential advantages as well as their potential drawbacks. The second approach (Section 4.5.2, 4.5.3, 4.5.4) details minutely the results of a very classical method *"Logistic Regression"* versus a very modern method *"XGBoost"*; in terms of interpretability, predictive capacity and overfitting. The purpose is to understand under which conditions modern methods outperform the basic ones.

4.5.1 Comparison of Methods

Tables 4.2 and 4.3 show the results of the comparison of the described methods for the training sample and for the testing sample. Firstly, the exploration suggests that the predictive measures that similarly perform for both data sets, do not have the overfitting problem. For instance, the random forest perfectly classify observations in the training data sets, but fails almost their half of capacity when predictive the testing data set.

The gradient tree boost seems to have the best sensitivity and specificity level, followed by the XGBoost (without hyperparameter optimization) and the logistic regression, performing similarly in both samples.

All gradient boosting techniques showed a very high level of sensitivity, but poor of specificity. This result does not mean that all these methods provide exactly the same results, actually, their estimated probabilities are more accumulated in the highest deciles of prediction, and in fact should be similar among methods.

Methods	$Y_i = 0, \hat{Y}_i = 0$	$Y_i = 1, \hat{Y}_i = 0$	$Y_i = 0, \hat{Y}_i = 1$	$Y_i = 1, \hat{Y}_i = 1$	**Sensitivity**	**Specificity**	**Accuracy**	**RMSE**
Logistic Regression	1016	54	783	84	0.609	0.5648	0.5679	0.2560
Decision Tree	1063	31	736	107	0.7764	0.5909	0.604	0.2495
Support Vector Machine	957	137	842	1	0.008	0.5319	0.4946	0.262
Naïve Bayes	1799	138	0	0	0	1	0.9288	0.267
K-Nearest Neighbors	1866	120	8	25	0.1724	0.9957	0.9372	0.2505
Random Forest	1799	0	0	138	1	1	0.9551	0.097
AdaBoost	1799	138	0	0	0	1	0.1564	0.2564
XGBoost	1030	55	775	77	0.5833	0.5706	0.5715	0.2508
Gradient Boost (Logistic)	0	0	1799	138	1	0	0.071	0.9632
Gradient Boost (Savage)	0	0	1799	138	1	0	0.0712	0.9635
Gradient Boost (Tangent)	0	0	1799	138	1	0	0.0712	0.9637
Gradient Boost (Logcosh)	0	0	1799	138	1	0	0.0712	0.5350
Gradient Boost (Mean Square Logarithmic)	0	0	1799	138	1	0	0.0712	0.4813
Gradient Tree Boost	1114	37	685	101	0.7318	0.6192	0.6275	0.2495
Two-step Adapted Gradient Boost Logistic	0	0	1799	138	1	0	0.0712	0.9637
Adapted Robust Gradient Boost	0	0	1799	138	1	0	0.0712	0.5317
Adapted M Boosted Regression	0	0	1799	138	1	0	0.0712	0.5308
W2 Weighted Linear Logitboost	1798	138	1	0	0	0.9994	0.9282	0.2649

The mean of the dependent variable is the threshold compute the predictive measures.

Table 4.2: Comparison of Methods in the training sample

Methods	$Y_i = 0$, $\hat{Y}_i = 0$	$Y_i = 1$, $\hat{Y}_i = 0$	$Y_i = 0$, $\hat{Y}_i = 1$	$Y_i = 1$, $\hat{Y}_i = 1$	**Sensitivity**	**Specificity**	**Accuracy**	**RMSE**
Logistic Regression	406	25	367	32	0.5614	0.5252	0.527711	0.2532
Decision Tree	466	32	307	25	0.4386	0.6928	0.5916	0.2576
Support Vector Machine	445	36	328	21	0.3684	0.5756	0.5615	0.2521
Naïve Bayes	773	57	0	0	0	1	0.9313	0.262
K-Nearest Neighbors	667	50	11	0	0	0.9838	0.9162	0.2894
Random Forest	428	27	345	30	0.5263	0.5537	0.5518	0.2589
AdaBoost	773	57	0	0	0	1	0.1638	0.9144
XGBoost	524	38	243	25	0.3968	0.6831	0.6614	0.2651
Gradient Boost (Logistic)	0	0	773	57	1	0	0.0687	0.9634
Gradient Boost (Savage)	0	0	773	57	1	0	0.0687	0.9634
Gradient Boost (Tangent)	0	0	773	57	1	0	0.0687	0.9634
Gradient Boost (Logcosh)	0	0	773	57	1	0	0.0687	0.5339
Gradient Boost (Mean Square Logarithmic)	0	0	773	57	1	0	0.0687	0.4818
Gradient Tree Boost	465	30	308	27	0.4737	0.6016	0.5927	0.254
Two-step Adapted Gradient Boost Logistic	0	0	773	57	1	0	0.0686	0.9645
Adapted Robust Gradient Boost	0	0	773	57	1	0	0.0686	0.5305
Adapted M Boosted Regression	0	0	773	57	1	0	0.0687	0.5299
W2 Weighted Linear Logitboost	771	57	2	0	0	0.9974	0.9289	0.2605

The mean of the dependent variable is the threshold compute the predictive measures.

Table 4.3: Comparison of Methods in the testing sample

The W2 Weighted linear Logitboost performs similarly in both samples and provides a strong specificity to the model, and its overall root mean square error is considerable lower than the other gradient boost methods.

To conclude, there is not a straight way to improve radically the prediction performance, but weighting mechanisms can control the importance of observations in the estimation, the loss functions do not seem to have a relevant impact on the final prediction, so more attention should be paid to the base learner instead. XGboost must be optimally trained to provide better results.

4.5.2 Coefficient Estimates

Table 4.4 presents the estimates obtained using the two methods. Note, however, that the values are not comparable in magnitude as they correspond to different specifications. The logistic regression uses its classical standard method to compute the coefficients of the variables and their standard errors. However, the boosting process of the XGBoost builds D models in reweighted versions and, so, we obtain a historical record of the D times $P + 1$ coefficient estimates. XGBoost can only obtain a magnitude of those coefficients if the base learner allows it, and this is not the case when f_d are CART models.

The signs obtained by the logistic regression point estimate and the mean of the XGBoost coefficients are the same. Inspection of the results in Table 4.4 shows that, in general, older insureds are less likely to suffer a motor accident than younger policy holders. In addition, individuals who travel more kilometers in urban areas are more likely to have an accident than those that travel fewer kilometers in urban areas. We are not able to interpret the coefficients of the XGBoost, but by inspecting the maximum and minimum values of the linear booster case, we obtain an idea of how the estimates fluctuate until iteration D.

Only the coefficients of age and percentage of kilometers travelled in urban areas are significantly different from zero in the logistic regression model, but we have preferred to keep all the coefficients of the covariates in the estimation results so as to show the general effect of the telematics covariates on the occurrence of accident at-fault claims in this dataset, and to evaluate the performance of the different methods in this situation.

Figure 4.2 shows the magnitude of all the estimates of the XGBoost in 200 it-

Parameter Estimates	Logistic Regression			XGBoost (linear booster)		
	Lower Bound	Estimate	Upper Bound	Minimum	Mean	Maximum
Constant	-2.8891	-0.5442	1.8583	-2.676	-2.6690	-1.7270
*age	-0.2059	-0.0994	0.0011	-0.2573	-0.2416	-0.0757
drivexp	-0.1285	-0.0210	0.0906	-0.0523	-0.0517	-0.0069
ageveh	-0.0786	-0.0249	0.0257	-0.0897	-0.0885	-0.022
male	-0.3672	0.0039	0.3751	0.0019	0.0020	0.0070
kmtotal	-0.0203	0.0266	0.0707	0.0137	0.1164	0.1176
pkmnig	-0.0354	-0.0046	0.0239	-0.0292	-0.0290	-0.0061
pkmexc	-0.0122	0.0144	0.0385	0.0180	0.1007	0.1016
*pkmurb	0.0002	0.0146	0.0286	0.0436	0.2008	0.2023

*In the logistic regression columns, the point estimates are presented with the lower and upper bound of a 95% confidence interval. In the XGBoost columns, the means of the coefficient estimates with a linear boosting of the D iterations are presented. Similarly, bounds are presented with the minimum and maximum values in the iterations. There are no regularization parameter values. * indicates that the coefficient is significant at the 90% confidence level in the logistic regression estimation.*

Table 4.4: The parameter estimates of the logistic regression and XGBoost with linear booster

erations. From approximately the tenth iteration, the coefficient estimates tend to become stabilized. Thus, no extreme changes are present during the boosting.

4.5.3 Prediction Performance

The performance of the two methods is evaluated using the confusion matrix, which compares the number of observed events and non-events with their corresponding predictions. Usually, the larger the number of correctly classified responses, the better the model. However, out-of-sample performance is even more important than in-sample results. This means that the classifier must be able to predict the observed events and non-events in the testing sample and not just in the training sample.

The predictive measures used to compare the predictions of the models are sensitivity, specificity, accuracy and the root mean square error (RMSE). Sensitivity measures the proportion of actual positives that are classified correctly as such, i.e. True positive/(True positive + False negative). Specificity measures the proportion of actual negatives that are classified correctly as such, i.e. True negative/(True negative + False positive). Accuracy measures the proportion of total cases clas-

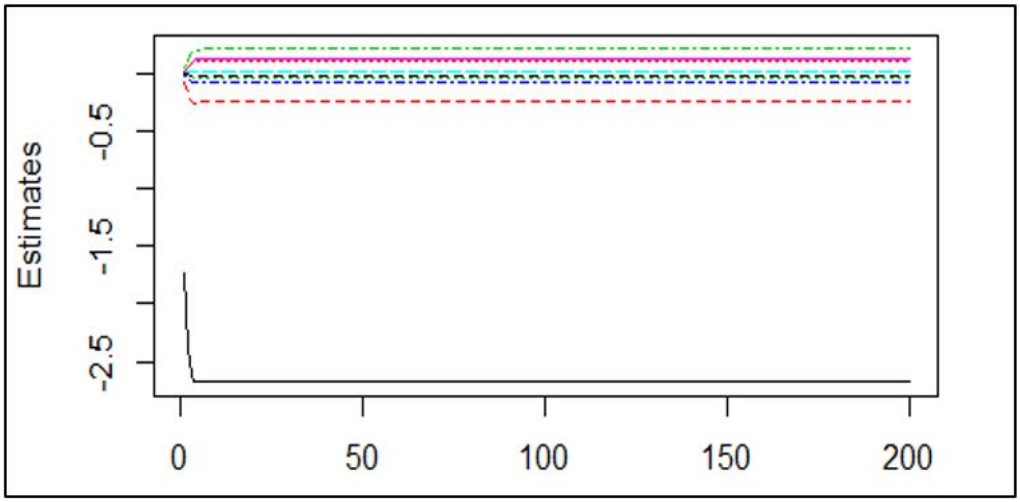

Figure 4.2: The magnitude of all the estimates in the D=200 iterations. Different colors indicate each of the coefficients in the XGBoost iteration.

sified correctly (True positive + True negative)/Total cases. RMSE measures the distance between the observed and predicted values of the response. It is calculated as follows:

$$\sqrt{\sum_{i=1}^{n} \frac{(Y_i - \hat{Y}_i)^2}{n}} \tag{4.27}$$

The higher the sensitivity, the specificity and the accuracy, the better the models predict the outcome variable. The lower the value of RMSE, the better the predictive performance of the model.

Table 4.5 presents the confusion matrix and the predictive measures of the methods (the logistic regression, XGBoost with a tree booster and XGBoost with a linear booster) for the training and testing samples. The results in Table 4.5 indicate that the performance of the XGBoost with the linear booster (last column) is similar to that of the logistic regression both in the training and testing samples. XGBoost using the tree approach provides good accuracy and a good RMSE value in the training sample, but it does not perform as well as the other methods in the case of the testing sample. More importantly, XGBoost fails to provide good sensitivity. In fact, the XGBoost with the tree booster clearly overfits the data, because while it performs very well in the training sample, it fails to do so in the testing sample. For instance, sensitivity is equal to 100% in the training sample for the XGBoost tree booster methods, but it is equal to only 7.9% in the testing sample.

Testing Data Set			
Predictive Measures	**Logistic Regression**	**XGBoost (tree booster)**	**XGBoost (linear booster)**
$Y_i = 0, \hat{Y}_i = 0$	524	692	516
$Y_i = 1, \hat{Y}_i = 0$	38	58	38
$Y_i = 0, \hat{Y}_i = 1$	243	75	251
$Y_i = 1, \hat{Y}_i = 1$	25	5	25
Sensitivity	0.3968	0.0790	0.3968
Specificity	0.6831	0.9022	0.6728
Accuracy	0.6614	0.8397	0.6518
RMSE	0.2651	0.2825	0.2651
Training Data Set			
Predictive Measures	**Logistic regression**	**XGBoost (tree booster)**	**XGBoost (linear booster)**
$Y_i = 0, \hat{Y}_i = 0$	1030	1794	1030
$Y_i = 1, \hat{Y}_i = 0$	55	0	55
$Y_i = 0, \hat{Y}_i = 1$	775	11	775
$Y_i = 1, \hat{Y}_i = 1$	77	132	77
Sensitivity	0.5833	1.0000	0.5833
Specificity	0.5706	0.9939	0.5706
Accuracy	0.5715	0.9943	0.5715
RMSE	0.2508	0.0373	0.2508

The threshold used to convert the continuous response into a binary response is the mean of the outcome variable. The authors performed the calculations.

Table 4.5: Confusion matrix and predictive measures of the logistic regression, XGBoost with a tree booster and XGBoost with a linear booster for the testing and training data sets.

It cannot be concluded from the foregoing, however, that XGBoost has a poor relative predictive capacity. Model-tuning procedures have not been incorporated in Table 4.5; yet, tuning offers the possibility of improving the predictive capac-

ity by modifying some specific parameter estimates. The following are some of the possible tuning actions that could be taken: fixing a maximum for the number of branches of the tree (maximum depth), establishing a limited number of iterations of the boosting, or fixing a number of subsamples in the training sample. The xgboost package in R denotes these tuning options as general parameters, booster parameters, learning task parameters, and command line parameters, all of which can be adjusted to obtain different results in the prediction.

Figure 4.3 shows the ROC curve obtained using the three methods on the training and testing samples. We confirm that the logistic regression and XGBoost (linear) have a similar predictive performance. The XGBoost (tree) presents an outstanding AUC in the case of the training sample, and the same value as the logistic regression in the testing sample; however, as discussed in Table 4.5, it fails to maintain this degree of sensitivity when this algorithm is used with new samples.

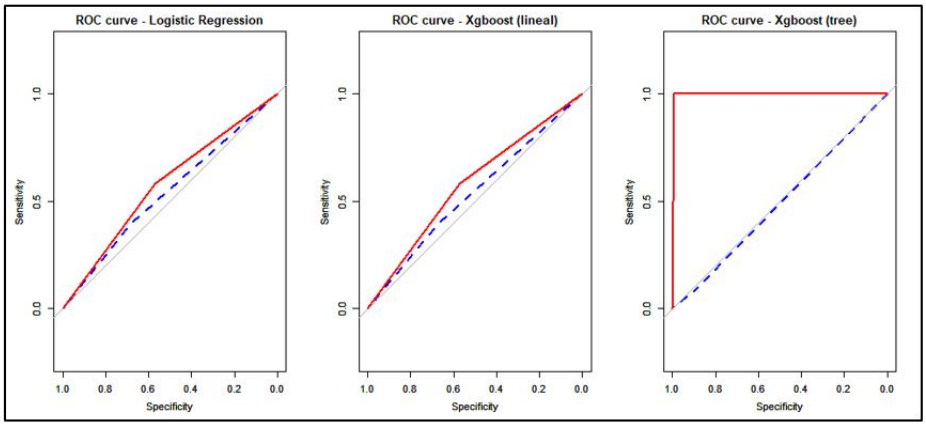

Figure 4.3: The Receiver Operating Characteristics (ROC) curve obtained using the three methods on the training and testing samples. The red solid line represents the ROC curve obtained by each method in the training sample, and the blue dotted line represents the ROC curve obtained by each method in the testing sample. The area under the curve (AUC) is 0.58 for the training sample (T.S) and 0.49 for the testing sample (Te.S) when logistic regression is used; 0.58 for the T.S and 0.53 for the Te.S when XGBoost (linear booster) is used; and, 0.997 for the T.S and 0.49 for the Te.S when the XGBoost (tree booster) is used.

4.5.4 Overfitting

One of the most frequently employed techniques for addressing the overfitting problem is regularization. This method shrinks the magnitude of the coefficients of the

covariates in the modelling as the value of the regularization parameter increases.

In order to determine whether the XGBoost (tree booster) can perform better than the logistic regression model, we propose a simple sensitivity analysis of the regularization parameters. In so doing, we evaluate the evolution of the following confusion matrix measures: accuracy, sensitivity and specificity – according to some given regularization parameter values for the training and the testing sample – and, finally, choose the regularization parameter that gives the highest predictive measures in the training and testing samples.

We consider two regularization methods. First, we consider the L2 (Ridge), which is Chen and Guestrin (2016)'s original proposal and takes the l2-norm of the leaf weights. It has a parameter λ that multiplies that l2-norm. Second, we consider the L1 (Lasso) method, which is an additional implementation possibility of the xgboost package in R that takes the l1-norm of the leaf weights. It has a parameter α that multiplies that l1-norm. Consequently, λ and α calibrate the regularization term in (4.17). For simplicity, no tree pruning was implemented, so $\mu = 0$ in (4.17).

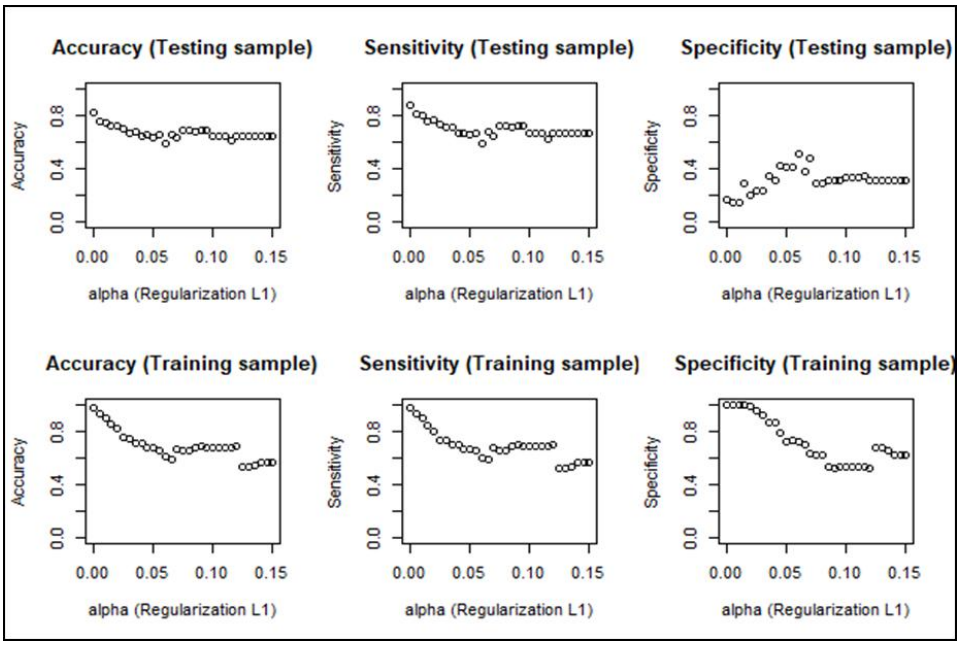

Figure 4.4: The predictive measures according to α. L1 method applied to the training and testing samples

The values of α and λ should be as small as possible, because they add bias to the estimates, and the models tend to become underfitted as the values of the regularization parameters become larger. For this reason, we evaluate their changes in a

small interval. Figure 4.4 shows the predictive measures for the testing and training samples according to the values of α when the L1 regularization method is implemented.

When $\alpha = 0$, we obtain exactly the same predictive measure values as in Table 4.5 (column 3) because the objective function has not been regularized. As the value of α increases, the models' accuracy and sensitivity values fall sharply – to at least $\alpha = 0.06$ in the training sample. In the testing sample, the fall in these values is not as pronounced; however, when α is lower than 0.06 the specificity performance is the lowest of the three measures. Moreover, selecting an α value lower than 0.05 results in higher accuracy and sensitivity measures, but lower specificity. In contrast, when α equals 0.06 in the testing sample, we obtain the highest specificity level of 0.5079, with corresponding accuracy and sensitivity values of 0.5892 and 0.5988, respectively. In the training sample, when $\alpha = 0.06$ the specificity, accuracy and sensitivity are: 0.7227, 0.6086, and 0.6000, respectively. As a result when α is fixed at 0.06, the model performs similarly in both the testing and training samples.

Thus, with the L1 regularization method ($\alpha = 0.06$), the new model recovers specificity, but loses some sensitivity when compared with the performance of the first model in Table 4.5, for which no regularization was undertaken. Thus, we conclude that $\alpha = 0.06$ can be considered as providing the best trade-off between correcting for overfitting while only slightly reducing the predictive capacity.

Figure 4.5 shows the predictive measures for the testing and training samples according to the values of λ when the L2 regularization method is implemented. From $\lambda = 0$ to $\lambda = 0.30$ all predictive measures are around 100% in the training sample; however, very different results are recorded in the testing sample. Specifically, accuracy and sensitivity fall slowly, but specificity is low – there being no single λ that makes this parameter exceed at least 20%. As such, no λ can help improve specificity in the testing sample. The L2 regularization method does not seem to be an effective solution to correct the problem of overfitting in our case study data set.

The difference in outcomes recorded between the L1 and L2 regularization approaches might also be influenced by the characteristics of each regularization method. Goodfellow et al. (2016) explain that L1 penalizes the sum of the absolute value of the weights, and that it seems to be robust to outliers, has feature selection, provides a sparse solution, and is able to give simpler but interpretable models. In contrast, L2 penalizes the sum of the square weights, has no feature selection, is not robust to outliers, is more able to provide better predictions when the response variable is

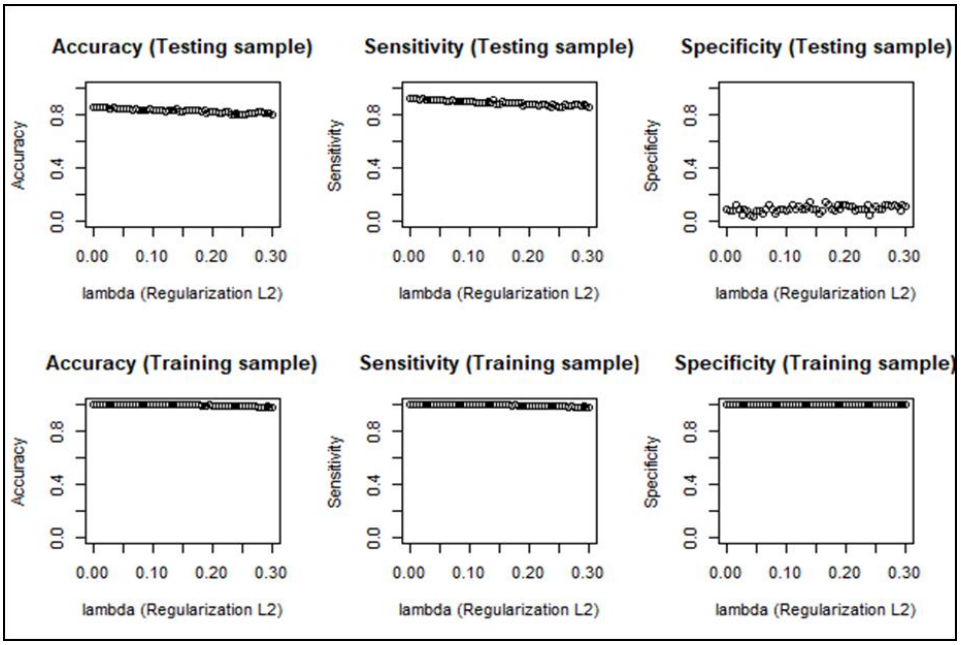

Figure 4.5: The predictive measures according to λ. L2 method applied to the training and testing samples

a function of all input variables, and is better able to learn more complex models than L1.

4.6 Conclusions

XGBoost, and other boosting models, are dominant methods today among machine-learning algorithms and are widely used because of their reputation for providing accurate predictions. This novel algorithm is capable of building an ensemble model characterized by an efficient learning method that seems to outperform other boosting-based predictive algorithms. Unlike the majority of machine learning methods, XGBoost is able to compute coefficient estimates under certain circumstances and, so, the magnitude of the effects can be studied. The method allows the analyst to measure not only the final prediction, but also the effect of the covariates on a target variable at each iteration of the boosting process, which is something that traditional econometric models (e.g. generalized linear models) do in one single estimation step.

When a logistic regression and XGBoost compete to predict the occurrence of accident claims without model-tuning procedures, the predictive performance of the XGBoost (tree booster) is much higher than that of the logistic regression in the

training sample, but considerably poorer in the testing sample. Thus, a simple regularization analysis has been proposed here to correct this problem of overfitting. However, the improvement in predictive performance of the XGBoost following this regularization is similar to that obtained by the logistic regression. This means additional efforts have to be taken to tune the XGBoost model so as to obtain a higher predictive performance without overfitting the data. This might be considered as the trade-off between obtaining a better performance, and the simplicity it provides for interpreting the effect of the covariates.

Based on our results, the classical logistic regression model can predict accident claims using telematics data and provide a straightforward interpretation of the coefficient estimates. Moreover, the method offers a relatively high predictive performance considering that only two coefficients are significant at the 90% confidence level. These results are not bettered by the XGBoost method.

When the boosting framework of XGBoost is not based on a linear booster, interpretability becomes difficult, as a model's coefficient estimates cannot be calculated. In this case, variable importance can be used to evaluate the weight of the individual covariates in the final prediction. Here, we obtained different conclusions for the two methods employed. Thus, given that the predictive performance of XGBoost was not much better than that of the logistic regression, even after careful regularization, we conclude that the new methodology needs to be adopted carefully, especially in a context where the number of event responses (accident) is low compared to the opposite response (no accident). Indeed, this phenomenon of unbalanced response is attracting more and more attention in the field of machine learning.

Appendix A: Alternative boosting-based algorithms

The boosting principle is the inspiration for the methodological strategy that we promote in this project. In boosting methods, training data that is hard to predict is given more weight, whereas easy to predict instances are given less weight. Models are created sequentially one after the other, each updating the weights on the training instances that affect the learning performed by the next tree in the sequence. After all the trees are built, predictions are made for new data, and the performance of each tree is weighted by how accurate it was on training data. This is the idea behind what we do, to give more weight to the extreme cases, rather than to clean data with outliers removed.

Gradient Boost:
The main idea behind the Gradient Boost proposed by (Friedman, 2001) is to compute a sum of optimized functions through an iterative procedure. The gradient boosting procedure starts with an initial guess of prediction $\hat{Y}^0$. It consists on minimizing a loss function through an argmin between the observed Y_i and an arbitrary constant ρ.

$$\hat{Y_i}^0 = argmin_\rho \sum_{i=1}^{n} \varphi(Y_i, \rho) \tag{4.28}$$

For $d = 1$ to D do:
Let $\widetilde{r_i}$ be the pseudo-residuals which is negative gradient of $\varphi(Y_i, \hat{Y_i}^d)$.

$$\widetilde{r_i}^d = -\frac{\partial \varphi(Y_i, \hat{Y_i}^d)}{\partial \hat{Y_i}^d}\Big|_{\hat{Y_i}^d = \hat{Y_i}^{d-1}}. \tag{4.29}$$

Then the squared error between the pseudo-residual and $F(X, u)$ is minimized. It delivers an updated u^d.

$$u^d = argmin_{u,\varpi} \sum_{i=1}^{n} \left[\widetilde{r_i}^d - \varpi F(x; u)\right]^2. \tag{4.30}$$

Let γ be the result of a minimized loss function between the observed Y_i and $\hat{Y_i}^d$ + $\gamma F(X; u^d)$. Note that $\hat{Y_i}^d$ is the prediction by the given covariates X_{ip} and the updated parameters u^d.

$$\gamma^d = argmin_\gamma \sum_{i=1}^{n} \varphi\left(Y_i, \hat{Y_i}^d + \gamma^d F(X; u^d)\right). \tag{4.31}$$

The final prediction at M iteration is the sum of the previous prediction $\hat{Y_i}^{m-1}$ and $\gamma \hat{Y_i}^d$.

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \gamma^d F(X; u^d) \tag{4.32}$$

endfor
End Algorithm

Gradient Boost (Logistic Loss Function):

- Alternative initial guess for $\hat{Y_i}^0$
- Loss Function: Logistic

- Base learner: Linear regression

Begin Algorithm

$$\hat{Y_i}^0 = \frac{1}{2} log \frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.33}$$

For $d = 1$ to D do:

$$\widetilde{r_i}^d = \frac{2Y_i}{1 + exp(2Y_i \hat{Y}^{d-1})}. \tag{4.34}$$

$$(\gamma^d, u^d) = argmin_{u,\gamma} \sum_{i=1}^{n} log \left[(1 + \exp(-2Y_i(\hat{Y}^{d-1} + \gamma^d F(X; u)) \right]. \tag{4.35}$$

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \gamma^d F(X; u^d). \tag{4.36}$$

endfor
End Algorithm

Gradient Boost (Savage Loss Function):

- Initial guess for $\hat{Y_i}^0$
- Loss Function: Savage $\frac{1}{1+exp(2Y_i \hat{Y_i})^2}$. It is a quasi-convex function that let converge machine learning algorithms in fewer iterations (see further in (Masnadi-Shirazi and Vasconcelos, 2009)).
- Base learner: Linear regression Begin Algorithm

$$\hat{Y_i}^0 = \frac{1}{2} log \frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.37}$$

For $d = 1$ to D do:

$$\widetilde{r_i}^d = -\frac{-4Y_i \exp 2Y_i \hat{Y_i}^{d-1}}{1 + exp(2Y_i \hat{Y_i}^{d-1})^3}. \tag{4.38}$$

$$u^d = argmin_{u,\varpi} \sum_{i=1}^{n} \left[\widetilde{r_i}^d - \varpi F(X; u^d) \right]^2. \tag{4.39}$$

$$(\gamma^d, u^d) = argmin_{u,\gamma} \sum_{i=1}^{n} log \left[\frac{1}{1 + \exp(-2Y_i(\hat{Y}^{d-1} + \gamma^d F(X; u^d))} \right] \tag{4.40}$$

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \gamma^d F(X; u^d) \tag{4.41}$$

endfor
End Algorithm

Gradient Boost (Tangent Loss Function):

- Alternative initial guess for $\hat{Y_i}^0$
- Loss Function: Tangent
  $2\arctan(Y_i\hat{Y}_i - 1)^2$. This loss function has some bounds that penalize perfect classified observations in order to prevent overfitting (see further in (Schulter et al., 2013)).
- Base learner: Linear regression

Begin Algorithm

$$\hat{Y_i}^0 = \frac{1}{2} log \frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.42}$$

For $d = 1$ to D do:

$$\widetilde{r_i}^d = -\frac{4Y_i(2\arctan(Y_i\hat{Y_i})) - 1}{Y_i^2\hat{Y_i}^2 + 1}. \tag{4.43}$$

$$u^d = argmin_{u,\varpi} \sum_{i=1}^{n} \left[\widetilde{r_i}^d - \varpi F(X;u)\right]^2. \tag{4.44}$$

$$(\gamma^d, u^d) = argmin_{u,\gamma} \sum_{i=1}^{n} \left[2\arctan(Y_i(\hat{Y}^{d-1} + \gamma^d F(X;u)) - 1\right]^2 \tag{4.45}$$

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \gamma^d F(X;u^d) \tag{4.46}$$

endfor
End Algorithm

Gradient Boost (Logcosh Loss Function)

- Alternative initial guess for $\hat{Y_i}^0$
- Loss Function: Logcosh
  $\frac{1}{n}\sum_{i=1}^{n} log\left(cosh(Y_i - \hat{Y_i})\right)$. It is the logarithm of the hyperbolic cosine of the prediction error.
- Base learner: Linear regression Begin Algorithm

$$\hat{Y_i}^0 = \frac{1}{2} log \frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.47}$$

For d = 1 to D do:

$$\widetilde{r_i}^d = -\frac{1}{n} \sum_{i=1}^{n} \left[\frac{\sinh(\hat{Y_i} - Y_i)}{\cosh(\hat{Y_i} - Y_i)} \right]. \tag{4.48}$$

$$u^d = argmin_{u,\varpi} \sum_{i=1}^{n} \left[\widetilde{r_i}^d - \varpi F(X; u^d) \right]^2. \tag{4.49}$$

$$(\gamma^d, u^d) = argmin_{u,\gamma} \frac{1}{n} \sum_{i=1}^{n} \left[log(cosh(Y_i - \hat{Y_i}^{d-1} - \gamma^d F(X; u))) \right] \tag{4.50}$$

$$\hat{Y_i}^m = \hat{Y_i}^{d-1} + \gamma^d F(X; u^d) \tag{4.51}$$

endfor
End Algorithm

Gradient Boost (Mean Square Logarithmic Error Loss Function)

- Alternative initial guess for $\hat{Y_i}^0$
- Loss Function: Mean square logarithmic error $\frac{1}{n} \sum_{i=1}^{n} \left[log\,(Y_i + 1) - log\left(\hat{Y_i} + 1\right) \right]$. It can be interpreted as a measure of the ratio between the Y_i and $\hat{Y_i}$. Also if it is used in regression models, large and small errors receive a similar penalization.
- Base learner: Linear regression Begin Algorithm

$$\hat{Y_i}^0 = \frac{1}{2} log \frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.52}$$

For d = 1 to D do:

$$\widetilde{r_i}^d = -\frac{1}{n} \sum_{i=1}^{n} \left[\frac{2\log(\hat{Y_i} + 1) - \log(Y_i + 1)}{\hat{Y_i} + 1} \right]. \tag{4.53}$$

$$u^d = argmin_{u,\varpi} \sum_{i=1}^{n} \left[\widetilde{r_i}^d - \varpi F(X; u) \right]^2. \tag{4.54}$$

$$(\gamma^d, u^d) = argmin_{u,\gamma} \frac{1}{n} \sum_{i=1}^{n} \left[\log(Y_i + 1) - \log(\hat{Y_i}^{d-1} + \gamma^d F(X; u) + 1) \right]^2 \tag{4.55}$$

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \gamma^d F(X; u^d) \tag{4.56}$$

endfor
End Algorithm

Gradient Boost (Cross Entropy Loss Function):

- Alternative initial guess for $\hat{Y}_i^0$
- Loss Function: Cross Entropy
- Base learner: Linear regression

Begin Algorithm

$$\hat{Y}_i^0 = \frac{1}{2} log \frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.57}$$

For $d = 1$ to D do:

$$\widetilde{r}_i^d = \frac{1}{n} \sum_{i=1}^{n} \frac{\hat{Y}_i - Y_i}{(\hat{Y}_i - 1)\hat{Y}_i}. \tag{4.58}$$

$$u^d = argmin_{u,\varpi} \sum_{i=1}^{n} \left[\widetilde{r}_i^d - \varpi F(X;u)\right]^2. \tag{4.59}$$

$$(\gamma^d, u^d) = argmin_{u,\gamma} \sum_{i=1}^{n} \left[Y_i \log\left(\hat{Y}_i^{d-1} + \gamma^d F(X;u)\right) + (1-Y_i)\log\left(1-\hat{Y}_i^{d-1} + \gamma^d F(X;u)\right)\right]. \tag{4.60}$$

$$\hat{Y}_i^d = \hat{Y}_i^{d-1} + \gamma^d F(X;u^d) \tag{4.61}$$

endfor
End Algorithm

Gradient Tree Boost:

- Initial guess for $\hat{Y}_i^0$ proposed by (Friedman, 2001)
- Loss Function: Logistic
  Also known as negative binomial log-likelihood Loss: $log(1 + e^{-2Y_i\hat{Y}_i})$. It is used in the two-class logistic regression and classification (L2_TreeBoost) introduced by (Friedman, 2001). Its convexity and linear growth make less sensitive to outliers.
- Base learner: CART

Begin Algorithm

$$\hat{Y}_i^0 = \frac{1}{2} log \frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.62}$$

For $d = 1$ to D do:

$$\widetilde{r_i}^d = \frac{2Y_i}{1 + exp(2Y_i\hat{Y}^{d-1})}. \tag{4.63}$$

This base learner $F(X_i; u, R)$ equals to $\sum_{j=1}^{J} u_p(1 \in R_j)$ with J terminal nodes mostly known as leaves, R_j classification rules of j leaves. u corresponds to the score of each which is the proportion of cases which are classified into Y_i given covariates X_{ip}.

All observations are mapped in the tree, considering $\widetilde{r_i}$ as target variable and covariates X_{ip}.

$$R_{jd}{}_1^J = j - leafscores(\widetilde{r_i}, X_1{}^n). \tag{4.64}$$

γ_{jd} is obtained for each leaf by minimizing a logistic loss function between the observed Y_i, and the $Y_i^{\hat{d}-1} + \gamma^d$.

$$\gamma_{jd} = argmin_\gamma \sum_{X_i \in R_{jd}}^{n} log\left[(1 + \exp(-2Y_i(\hat{Y}^{d-1} + \gamma^d)\right]. \tag{4.65}$$

However, since there is no closed form for (4.65). A new version is obtained with the Newton-Raphson method.

$$\gamma_{jd} = \frac{\sum_{X_i \in R_{jd}} \widetilde{r_i}}{\sum_{X_i \in R_{jd}} |\widetilde{r_i}\,(2 - |\widetilde{r_i}|)|} \tag{4.66}$$

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \sum_{j=1}^{J} \gamma_{jd} 1(x \in R_{jd}) \tag{4.67}$$

endFor
End Algorithm

Two-step Adapted Gradient Boost Logistic

This proposal was motivated as an extended version of (Pesantez-Narvaez and Guillen, 2020b) and might be considered as a particular case of cost-sensitive learning by establishing varying costs of different misclassification types. The two-step adapted gradient boost incorporated a weighting procedure that is used to differentiate the contribution of observations, and correct problems of over-representation and under-representation in order to improve the predictive performance of the model.

These two-steps procedure consists on: firstly, measuring weights once a confusion matrix is obtained with fitted $\hat{Y}_i$ by a logistic regression. And secondly, weights are introduced in the gradient boosting with a logistic loss function.

First step:

- Obtain $\hat{Y}_i$ (continuous scores) from a logistic regression.
- Transform the obtained $\hat{Y}_i$ into binary by choosing an arbitrary threshold ψ that will transform $\hat{Y}_i$ into binary.
- Compute a confusion matrix with the observed Y_i and the predicted $\hat{Y}_i$. And let w_i be the vector of weights that is created as follows:
  - Observations will be assigned a weight equal to 0.9 if the misclassification comes from $\hat{Y}_i = 1$ and $Y_i = 0$. In this case, predicted observations are over-estimated, and they will be forced to have less weight on the final prediction.
  - Observations are assigned a weight equal to 1.1 if the misclassification comes from $\hat{Y}_i = 0$ and $Y_i = 1$. In this case, predicted observations are under-estimated, they might be forced to have more weight on the final prediction.
  - Correctly predicted observations are assigned a weight equal to 1. In this case, any modification of weights is necessary.

Second Step:

- A gradient boosting with a logistic loss function is fitted.
- The base learner is a weighted linear regression with a vector of weights w_i defined in the first step.

There are other types of cost-sensitive learning such as: tree-building to minimize the misclassification costs in order to to choose the best split of a tree (explained further in (Riddle et al., 1994)), or to determine where the branch of the should be pruned (explained further in (Bradford et al., 1998)). Moreover, Sun et al. (2007) affirms that each observation might be assigned with the lowest risk through the Bayes risk theory.

Adapted Robust Gradient Boost

- Alternative initial guess for $\hat{Y}_i^0$

- Loss Function: Logistic
- Base learner: Weighted robust linear regression

Begin Algorithm

$$\hat{Y_i}^0 = \frac{1}{2} log \frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.68}$$

For $d = 1$ to D do:

$$\widetilde{r_i}^d = \frac{2Y_i}{1 + exp(2Y_i \hat{Y}^{d-1})}. \tag{4.69}$$

$$u^d = argmin_{u,\varpi} \sum_{i=1}^{n} \left[\widetilde{r_i}^d - \varpi F(X;u) \right]^2. \tag{4.70}$$

$$(\gamma^d, u^d) = argmin_{u,\gamma} \sum_{i=1}^{n} log \left[(1 + \exp(-2Y_i(\hat{Y}^{d-1} + \gamma^d F(X;u)) \right]. \tag{4.71}$$

The tuning constant for Huber's psi-function equals to 1 (ϕ =1).

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \gamma^d F(X;u^d) \tag{4.72}$$

endfor
End Algorithm

Adapted *M* Boosted Regression:

Recalling the original *M* Boosted Regression, it considers a Huber loss function (Huber, 1964) which is robust long-tailed error distributions and outliers.

$$\varphi(Y_i, \hat{Y_i}) = \begin{cases} \frac{1}{2}\left(Y_i - \hat{Y_i}\right) & \left|Y_i - \hat{Y_i}\right| \leq \theta \\ \theta \left|Y_i - \hat{Y_i}\right| - \theta/2 & \left|Y_i - \hat{Y_i}\right| \geq \theta. \end{cases} \tag{4.73}$$

And a $\widetilde{r_i}$ as:

$$\widetilde{r_i} = \begin{cases} Y_i^d - \hat{Y}^d & \left|Y_i^d - \hat{Y}^d\right| \leq \delta \\ \theta sign\left(Y_i^d - \hat{Y}^d\right) & \left|Y_i^d - \hat{Y}^d\right| > \delta \end{cases} \tag{4.74}$$

The new Adapted M Boosted Regression will perform in the following way:

- Alternative initial guess for $\hat{Y_i}^0$

- Loss Function: Logistic
- M Boosted Regression pseudo-residuals (modified)
- Base learner: Weighted robust linear regression

Begin Algorithm

$$\hat{Y^0} = \delta log \frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.75}$$

For $d = 1$ to D do:

$$\widetilde{r_i}^d = \frac{2Y_i}{1 + exp(2Y_i Y^{d-1}(X))}. \tag{4.76}$$

Let's rebuild a new pseudo-residuals such that r_i is the actual difference between Y_i^d and $\hat{Y_i}^d$. However, this difference was measured by the negative gradient of the logistic loss function in the previous step. So we can express a transformed pseudo-residuals $\widetilde{rt}_i$.

$$\widetilde{rt}_i = \begin{cases} \widetilde{r}^{d-1} & \left|r^{d-1}\right| \leq \delta \\ \delta sign\left(r^{d-1}\right) & \left|r^{d-1}\right| > \delta \end{cases} \tag{4.77}$$

Then optimize γ:

$$\gamma^d = argmin_\gamma \sum_{i=1}^{n} \varphi\left(Y_i, \hat{Y}_{m-1} + \gamma F(X_i; u)\right) \tag{4.78}$$

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \gamma^d F(X; u^d) \tag{4.79}$$

endfor
End Algorithm

Adapted Weighted Delta Linear Boost

This proposal consists of two steps: firstly, calibrating weights based on the results of a logistic regression, and secondly, training a Delta Boosting Machine as proposed by (Lee and Lin, 2018) with weights established in the first step.

First step:

- Let's train a logistic regression and obtain $\hat{Y_i}$.
- Let's transform the obtained $\hat{Y_i}$ into binary by choosing an arbitrary threshold ψ that will transform $\hat{Y_i}$ into binary.

- Let's compute a confusion matrix with the observed Y_i and the predicted $\hat{Y}_i$. And let w_i be the vector of weights that is created as follows:
  - Observations will be assigned a weight equal to 0.9 if the misclassification comes from $\hat{Y}_i = 1$ and $Y_i = 0$. In this case, predicted observations are over-estimated, and they will be forced to have less weight on the final prediction.
  - Observations are assigned a weight equal to 1.1 if the misclassification comes from $\hat{Y}_i = 0$ and $Y_i = 1$. In this case, predicted observations are under-estimated, they might be forced to have more weight on the final prediction.
  - Correctly predicted observations are assigned a weight equal to 1. In this case, any modification of weights is necessary.

Second Step: Let's train a Delta Boosting Machine with the following specifications:

- Alternative $\hat{Y^0}$.
- Loss Function: Logistic
- Base learner: Weighted linear regression with weights w_i defined in the first step.

Begin Algorithm

$$\hat{Y^0} = 0.5log\frac{1+\bar{Y}}{1-\bar{Y}} \tag{4.80}$$

For $d = 1$ to D do: Let's compute the individual loss minimizer as the working response:

$$\widetilde{\delta_i} = \frac{2Y_i}{1 + exp(2Y_i\hat{Y_i^{d-1}})}, \tag{4.81}$$

Let's obtain u:

$$\gamma^d = argmin_\gamma \sum_{i=1}^{n} \varphi\left(Y_i, \hat{Y}_{m-1} + \gamma F(X_i; u)\right) \tag{4.82}$$

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \gamma^d F(X; u^d) \tag{4.83}$$

W2 Weighted Linear Logitboost

The W2 Weighted Linear Logitboost is an adapted version of the original Logitboost proposed by (Friedman et al., 2000). Herein, the pseudo-residuals $\tilde{r}_i$ are transformed into a working response z_i through a quadratic approximation of the log-likelihood known as χ^2.

- Set the initial w_i^0 equal to $1/n$, so that all observations have the same weight in the first iteration.
- $\hat{Y^0} = 0$
- Let $\pi(X_i)$ be the probability estimates. $\pi^0(X_i) = \frac{1}{2}$.
- Alternative w_i from the second iteration onwards.

For $d = 1, ..., D$
Compute the working response:

$$z_i = \frac{Y_i - \pi(X_i)}{\pi\left(X_i\right)\left(1 - \pi\left(X_i\right)\right)}, \tag{4.84}$$

Compute the vector of weights:

$$w_i = \frac{p(X_i)(1 - p(X_i))}{|Y_i - p(X_i)|} \tag{4.85}$$

Normalize weights by dividing each of them by the sum of all the weights w_i.

$$w_i = \frac{w_i}{\sum_{i=1}^{n} w_i} \tag{4.86}$$

Fit z_i to the covariates X_{ip} by a weighted least-squares regression with weights w_i. In other words, $F(X_i; u)$ is a weighted least-squares regression.

Update the final prediction;

$$\hat{Y_i}^d = \hat{Y_i}^{d-1} + \frac{1}{2}F(X_i, u). \tag{4.87}$$

Compute the probabilities:

$$\pi(X_i)^d = min\left\{\frac{1}{1 + exp(-2\hat{Y_i^{d-1}})}, 1\right\} \tag{4.88}$$

end for
End Algorithm

Chapter 5: A synthetic penalized logitboost to model mortgage lending with imbalanced data

5.1 Introduction

Predicting binary decision problems is important in empirical economics. For instance, identifying whether an applicant will default in future or be turned down under the Home Mortgage Disclosure Act[1] (HMDA) contributes to the study of financial inclusion policy. In fact, the notion of having events versus non-events (a binary response) can be the result of a latent and unobserved random variable that triggers an event when it is high enough, so that extreme values then turn into event responses.

Class-imbalanced data are relevant primarily in the context of supervised machine learning involving two (dichotomous) or more classes. Imbalanced means that the number of observations is not the same for each class of a categorical variable, in other words, one class is represented by a large number of observations while the other is represented by only a few (Japkowicz and Stephen, 2002).

In the context of mortgage lending, for example, Munnell et al. (1996) have dealt with an imbalanced class problem. They found that black and Hispanic applicants

This chapter can be found in Pesantez-Narvaez, J., Guillen, M., & Alcañiz, M. (2021). A Synthetic Penalized Logitboost to Model Mortgage Lending with Imbalanced Data. *Computational Economics*, 57, 281–309.

[1]HMDA is a disclosure law that provides publicly available information on the US mortgage market where applicants' characteristics are registered in order to identify possible patterns of discriminatory lending. The 94th United States Congress found that some financial institutions tend to decline qualified applicants without sufficient rationale.

were more likely than whites to be denied mortgage loans. Thus, the class corresponding to the applicants who were denied was much smaller than the applicants who were approved. The minority class (denied mortgage lending) could be coded as one, while the majority class (approved for mortgage lending) could be coded as zero.

There is evidence that the prediction accuracy of this type of events seems to remain problematic. King and Zeng (2001) note that classical econometric methods can underestimate the probability of occurrence in the minority class, while Krawczyk (2016) finds that machine-learning methods tend to exhibit a bias towards the majority class.

There is a vast literature devoted to proposing techniques to handle the class imbalance problem. Barandela et al. (2003); Kotsiantis et al. (2006); Longadge et al. (2013) and Lin et al. (2017) summarize four types of techniques:

(i) data preprocessing (balancing the data by oversampling, which increases the number of observations in the minority class, or by undersampling, which reduces observations in the majority class) using an algorithm approach (creating or modifying algorithms with the threshold and one-class learning methods),

(ii) cost-sensitive solutions (minimizing the costs of misclassification),

(iii) feature selection (finding the optimal combination of covariates that gives the best classification), and

(iv) resampling techniques incorporated in classifier ensembles such as boosting or bagging, which have given risen to proposals such as Synthetic Minority Oversampling (SMOTE) (Chawla et al., 2002), RUSBoost (Seiffert et al., 2009), UnderBagging (UB) (Barandela et al., 2003), and OverBagging (Wang and Yao, 2009).

In our view, however, the modelling and interpretability of imbalanced class phenomena in a joint process without overfitting data remains a subject beyond the scope of machine learning. We propose a Synthetic Penalized Logitboost that aims to decrease the mean square error in the highest and lowest prediction scores of the probability of minority class occurrence, by introducing a weighting mechanism that recalibrates a Logitboost to reduce the risk of overfitting. The Synthetic Penalized Logitboost improves the detection of extremes in the data if the purpose is to

look for unusual patterns rather than for average cases. For this purpose, we borrow the specification of the model put forward by (Munnell et al., 1996) to predict mortgage loan denial with a logistic regression.

The chapter is divided into five sections after the introduction. Section 5.2 describes the theoretical framework that motivates the paper. Section 5.3 describes the methodology in detail, specifically logistic regression (econometric model for binary prediction), Logitboost, Gradient Tree Boost (boosting-based machine learning for binary prediction) and the proposed algorithm. Section 5.4 describes the data set used in an illustrative example. Section 5.5 sets out the results and predictive performance measured by the root-mean-square error and includes the model's interpretation. Finally, Section 5.6 contains the conclusions.

5.2 Theoretical Framework

Considering a supervised statistical learning framework, let us start from a data set of n observations with a quantitative target variable (dependent variable) $Y_j, j = 1, \cdots, n$ that has some relationship with a set of P predictor variables denoted as $X_{ip}, p = 1, \cdots, P$ (also known as covariates). This can be written as:

$$Y_i = F(X_{ip}) + \varepsilon_i, \tag{5.1}$$

where F is a deterministic function of the X_{ip}, and ε_i is the error or disturbance term that captures the influence of omitted factors, is independent of X_{ip} and has zero mean.

In econometrics, parametric models, such as linear or generalized linear models, and non-parametric models, such as spline regressions or generalized additive models, adopt their corresponding regression form. So, in simple models, instead of estimating the corresponding $P-$dimensional function $F\left(X_{ip}\right)$, it is necessary only to obtain the $P + 1$ coefficient estimates β_p of the linear predictor $\beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip}$.

Machine learning also uses alternative F in the form of classification and decision trees (Breiman et al., 1984), radial basis functions (Gomez-Verdejo et al., 2005), and random Markov fields (Dietterich et al., 2008), among others. The function F is known as a *base learner* in the machine learning literature.

Function F can be used to make inferences or predictions, or both. Even though

econometric models are aimed at explanatory or predictive modelling, or both, non-econometric models are mainly used for prediction purposes (classification or regression problems)[2], because their F functions are not able to provide coefficient estimates that are directly interpretable as marginal effects.

When F is used for prediction purposes, given that (5.1) has an error term that averages zero, a predicted target variable $\hat{Y}_i$, for $\hat{F}$ that estimates the observed F, can be written as follows:

$$\hat{Y}_i = \hat{F}(X_{ip}). \tag{5.2}$$

In this setting, James et al. (2013) identify two types of errors: reducible and irreducible. When the expected value or average of the squared difference between the observed Y_i and predicted $\hat{Y}_i$ is taken, we obtain:

$$E\left(Y_i - \hat{Y}_i\right)^2 = E\left[F(X_{ip}) + \varepsilon_i - \hat{F}(X_{ip})\right]^2, \tag{5.3}$$

which gives as a result:

$$E\left(Y_i - \hat{Y}_i\right)^2 = \left[F(X_{ip}) - \hat{F}(X_{ip})\right]^2 + Var(\varepsilon_i), \tag{5.4}$$

where the reducible error is $\left[F(X_{ip}) - \hat{F}(X_{ip})\right]^2$, and the irreducible error is $Var\left(\varepsilon_i\right)$ (variance of the error term). In fact, machine learning with non-econometric models aims to minimize the reducible error, which is equivalent to minimizing the distance between Y_i and $\hat{Y}_i$. This distance is known as the loss function, and will be denoted as $\varphi\left(Y_i, \hat{Y}_i\right)$.

Note that $Var\left(\varepsilon_i\right)$ cannot be reduced because these models only have a deterministic part that excessively learns from a given data set, in other words, they remove the only stochastic term. Consequently, highly accurate predictive machine learning algorithms such as certain tree-based or boosting-based techniques may result in overfitting, which means that the fitted models do not perform well on other databases. This is known as non-reproducibility. This result has also been verified by (Pesantez-Narvaez et al., 2019).

Many loss functions have been proposed to develop machine learning algorithms

[2] If Y_i is qualitative, we have with a classification problem, whereas if Y_i is quantitative, we have a regression problem. The latter must not be confused with a linear regression model. The machine learning and econometrics literatures have some discrepancies in terminology.

with greater predictive accuracy. They must be convex and differentiable. This paper will focus on the exponential loss function that is used in a Logitboost:

$$\varphi\left(Y_i, \hat{Y}_i\right) = e^{Y_i \hat{Y}_i}. \tag{5.5}$$

In order to increase the predictive capacity, therefore, it makes sense to consider a simple econometric method like a base learner in a boosting-based algorithm. Firstly, the irreducible error may be effectively reduced by readjusting the base learner to improve the model fit. Secondly, the reducible error can also be computed. The statistical intuition behind choosing a primitive econometric model is that the newest iterations of boosting-based algorithms correct the prediction error by considering the previous iterations. This can be done more efficiently if the base learner is a weak[3] one, because there is more variability to learn in weak base learners than in strong ones that already have good predictive performance and no or almost no variability.

5.3 Description of Methodology

Three groups of boosting-based algorithms are considered: the classical econometric model, gradient boosting for classification and Logitboost-based algorithms. The first group consists of logistic regression. The second group consists of the original gradient boosting algorithm and gradient boosting tree. The third group consists of the original Logitboost and the proposed Synthetic Penalized Logitboost.

Note that $F\left(X_{ip}; u\right)$ is the base learner mentioned earlier. It is a function of covariates X_{ip} and the parameters[4] represented by u.

In the data set that will be used in the following section, there are n individuals and P covariates. The target variable Y_i is now an observed binary response variable that takes two values coded as 1 for the minority class (denied mortgage loan) and 0 for the majority class (approved for mortgage loan). Let D be the number of iterations of the boosting procedure, with $d = 1, ..., D$.

[3] Schapire and Freund (2012) define a weak learner as a particular case of base learner whose predictive performance is slightly better than chance, and typically far from zero.

[4] For example, if $F\left(X_{ip} : u\right)$ is a regression model, u represents the coefficient estimates β, whereas if $F\left(X_{ip}; u\right)$ is a classification and regression tree (CART), then u represents branches of the tree (splitting rules).

Logistic Regression

Let us assume that in the data set of n individuals and P covariates, the target variable is now an observed binary response variable that takes two values coded as 1 for the rare class and 0 for the majority class. A logistic regression is a classical econometric tool that is used to model and predict binary dependent variables explained by quantitative or qualitative covariates. It is a specific case of a generalized linear model when the link is the logit function and is given as:

$$\log\left(\frac{\pi(Y_i=1)}{1-\pi(Y_i=1)}\right)=\beta_0+\sum_{p=1}^{P}X_{ip}\beta_p, \tag{5.6}$$

where $\beta_0,\beta_1,\cdots,\beta_p$ are the model parameters, and $\pi(Y_i=1)$ is the probability that Y_i equals to 1 conditional on the covariates. By a simple algebraic manipulation, $\pi(Y_i=1)$ is:

$$\pi(Y_i=1)=E(Y_i)=\frac{e^{\beta_0+\sum_{p=1}^{P}X_{ip}\beta_p}}{1+e^{\beta_0+\sum_{p=1}^{P}X_{ip}\beta_p}}. \tag{5.7}$$

A logistic regression can be estimated by maximum likelihood method (for further details, see for example (McCullagh and Nelder, 1983)).

Gradient Boosting

The idea behind the Gradient Boosting proposed by (Friedman, 2001) is to compute a sum of optimized functions through an iterative process. The optimized functions are the result of a minimization of a loss function φ.

Let us assume that in the data set of n individuals and P covariates, the target variable Y_i is now continuous. The gradient boosting procedure starts with an initial guess of prediction $\hat{Y}_i^0$. It then consists of minimizing a loss function through an *argmin* between the observed $\hat{Y}_i$ and and an arbitrary constant ρ.

$$\hat{Y}_i^0=argmin_\rho\sum_{i=1}^{n}\varphi\left(Y_i,\rho\right). \tag{5.8}$$

Begin Algorithm:

For $d=1$ to D do:

Let $\tilde{r}_i^d$ be the vector of the pseudo-residual which is the negative gradient of $\varphi\left(Y_i,\hat{Y}_i^d\right)$

at iteration d.

$$\tilde{r}_i^d = - \left. \frac{\partial \varphi \left(Y_i, \hat{Y}_i^d \right)}{\partial \hat{Y}_i^d} \right|_{Y_i = \hat{Y}_i^{d-1}} . \tag{5.9}$$

Then the squared error between the pseudo-residual and $F(X, u)$ is minimized. This results in an updated u^d:

$$u^d = argmin_{u,\beta} \sum_{i=1}^{n} \left[\tilde{r}_i^d - \beta F \left(X_{ip}; u^d \right) \right]^2 . \tag{5.10}$$

Let γ be the result of a minimized loss function between the observed Y_i and $\hat{Y}_i^d + \gamma F \left(X_{ip}; u^d \right)$. Note that $\hat{Y}_i^d$ is the prediction from the given covariates X_{ip} and the updated parameters u at iteration d.

$$\gamma^d = argmin_{\gamma} \sum_{i=1}^{n} \varphi \left[Y_i, \hat{Y}_i^d + \gamma F \left(X_{ip}; u^d \right) \right] . \tag{5.11}$$

The final prediction at iteration D is the sum of the previous prediction $\hat{Y}_i^{d-1}$ and $\hat{Y}_i^d$.

$$\hat{Y}_i^d = \hat{Y}_i^{d-1} + \gamma F \left(X_{ip}; u^d \right) . \tag{5.12}$$

End For

End Algorithm

Gradient $L2$ TreeBoost (Two-Class Logistic Boost

Let us assume that in the data set of n individuals and P covariates, the target variable Y_i is now an observed binary response variable that takes two values coded as 1 for the rare class and 0 for the majority class. The $L2$ TreeBoost proposed by (Friedman, 2001) differs from the Original Gradient Boost in:

1. Initial prediction $\hat{Y}_i^0$
2. Loss function: Logistic loss function
3. Base learner: Decision tree

The first estimation is calculated as follows:

$$\hat{Y}_i^0 = \frac{1}{2} \log \left(\frac{1 + \bar{Y}}{1 - \bar{Y}} \right) , \tag{5.13}$$

where $\bar{Y}$ is the mean of the dependent variable.

Begin Algorithm:

For $d = 1$ to D do:

$$\tilde{r}_i^d = \frac{2Y_i}{1 + e^{2Y_i \hat{Y}_i^{d-1}}}. \tag{5.14}$$

The base learner $F\left(X_{ip}; u, R\right)$ equals $\sum_{j=1}^{J} u_j \, 1\left(X_{ip} \varepsilon R\right)$ with J with J terminal nodes known as leaves, and R_j regions or classification rules, $j = 1, \cdots, J$. Parameters u correspond to the score of each leaf, which is the proportion of cases classified into Y_i given covariates X_{ip}. The tree-based algorithms are theoretically more efficient than linear or generalized linear methods in capturing non-linearities. The idea is that tree-based algorithms use information gain (measured by Gini impurity or entropy) to split a node. This helps to order the decision nodes associated with each covariate X_{ip}, so that the decision node with the highest information gain will split first, and so on until the one with lowest information gain. The information gain builds the R_j classification rules that map each observation i onto the correct leaf j by minimizing the entropy or Gini impurity of each node, so that the observations contained in the node are the most homogeneous (see further details in (Hastie et al., 2009)).

Now R_{jd} is computed by mapping all observations onto leaf j of tree ($j = 1, \cdots, J$) at iteration d, considering $\tilde{r}_i$ as the target variable and covariate X_{ip} as follows:

$$R_{jd} = j - \, leaf \text{ scores } (\tilde{r}_i, X_1^n). \tag{5.15}$$

Therefore γ_j^d is calculated for each leaf by minimizing a logistic loss function between the observed Y_i and $\hat{Y}_i^{d-1} + \gamma^d$.

$$\gamma_j^d = argmin_\gamma \sum_{X_i \in R_{jd}}^{n} \log\left[1 + e^{-2Y_i\left(\hat{Y}_i^{d-1} + \gamma^d\right)}\right]. \tag{5.16}$$

However, since there is no closed form for the previous equation, an approximation of γ_i^d is obtained through the Newton-Raphson method as follows:

$$\gamma_j^d = argmin_\gamma \frac{\sum_{X_i \in R_{jd}} \tilde{r}_i}{\sum_{X_i \in R_{jd}} |\tilde{r}_i \left(2 - |\tilde{r}_i|\right)|}. \tag{5.17}$$

And the final prediction $\hat{Y}_i^d$ is computed as:

$$\hat{Y}_i^d = Y_i^{d-1} + \sum_{j=1}^{J} \gamma_j^d 1(X_i \varepsilon R_{jd}). \tag{5.18}$$

End For
End Algorithm

Since tree-based algorithms generally overfit, decision tree pruning is considered in order to build a smaller tree with fewer J terminal nodes that lead to smaller variance by retaining the most relevant information and removing the least relevant (see further details in (Hastie et al., 2009)). For simplicity, Gradient $L2$ TreeBoost will be referred to as Gradient Tree Boost from here on.

Logitboost

The previous gradient boosting algorithms require the minimization of a loss function $\varphi\left(Y_i, \hat{Y}_i\right)$. However, Friedman (2001) have managed to approximate a logistic function as an additive logistic regression known as "Logitboost".

Let us assume that in the data set of n individuals and P covariates, the target variable Y_i is now an observed binary response variable that takes two values coded as 1 for the rare class and 0 for the majority class.

The Logitboost has some initial conditions:

1. Initial prediction $\hat{Y}_i^0 = 0$.
2. Let $\pi\left(X_i\right)$ be the probability estimates $p^0\left(X_i\right) = \frac{1}{2}$.

Begin Algorithm:

For $d = 1$ to D do:

This algorithm initializes by computing the working response z_i:

$$z_i^d = \frac{Y_i^{d-1} - \pi(X_i)^{d-1}}{\pi(X_i)^{d-1}\left(1 - p(X_i)^{d-1}\right)}. \tag{5.19}$$

In this case the χ^2 is a quadratic approximation of the log-likelihood with which a logistic regression can be estimated. According to (Friedman et al., 2000), the χ^2

can be a gentle alternative when the exponential loss function is used. Therefore, the working response z_i is an analogous expression to the pseudo-residuals $\tilde{r}_i$.

Again, the exponential loss function written in (5.5) is:

$$\begin{aligned} \varphi\left(Y_i, \hat{Y}_i\right) &= e^{Y_i \hat{Y}_i}, \\ e^{Y_i \hat{Y}_i} &= \frac{|Y_i - p(X_i)|}{\sqrt{p(X_i)\left(1 - p(X_i)\right)}}, \end{aligned} \tag{5.20}$$

where $\hat{Y}_i$ is obtained as follows:

$$\hat{Y}_i^d = \frac{1}{2} \log\left(\frac{\pi(X_i)^{d-1}}{\left(1 - \pi(X_i)^{d-1}\right)}\right). \tag{5.21}$$

A vector of weights w_i is computed as follows:

$$w_i^d = \pi(X_i)^{d-1}\left(1 - \pi(X_i)^{d-1}\right). \tag{5.22}$$

A base learner $F\left(X_i; u\right)$ must be trained by fitting a weighted least squares regression with a vector of weights w_i and a target variable z_i. Note that even though a binary target variable is set for this boosting, this F admits continuous target variables. The reason is that the working response z_i transforms the binary variable Y_i into a continuous one, so that two classes are still found in the first iteration. However, from the second iteration onwards, observations of z_i start to change during the boosting, so that at the end several values of z_i are found.

$$\beta^d = argmin_\beta \sum_{i=1}^{n} w_i \left[z_i^d - \left(\beta_0 + \sum_{p=1}^{P} X_{ip}\beta_p\right)\right]^2. \tag{5.23}$$

$\hat{Y}_i^d$ has to be updated as follows:

$$\hat{Y}_i^d = \hat{Y}_i^{d-1} + \frac{1}{2} F\left(X_{ip}; u^d\right). \tag{5.24}$$

Parameters u^d are the coefficient estimates β obtained in the linear regression.

Then the probabilities have to be updated:

$$\pi(X_i)^d = \frac{e^{\hat{Y}_i^d}}{e^{\hat{Y}_i^d} + e^{-\hat{Y}_i^d}}. \tag{5.25}$$

End For

End Algorithm

Synthetic Penalized Logitboost

The proposed Synthetic Penalized Logitboost incorporates slight changes to the original Logitboost and introduces a new alternative weighting mechanism w_i. This methodological proposal was particularly motivated by (Pesantez-Narvaez and Guillen, 2020a,b). They managed to propose weighting corrections in parametric models to improve their predictive performance for binary dependent variables.

We keep the two initial conditions for $\hat{Y}_i^0$ and $\pi^0(X_i)$:

1. Initial prediction: $\hat{Y}_i^0 = 0$.
2. Let $\pi(X_i)$ be the probability estimates $\pi^0(X_i) = \frac{1}{2}$.

Begin Algorithm:

For $d = 1$ to D do:

This algorithm initializes by computing the working response z_i:

$$z_i^d = \frac{Y_i^{d-1} - \pi(X_i)^{d-1}}{\pi(X_i)^{d-1}\left(1 - \pi(X_i)^{d-1}\right) + \delta}, \tag{5.26}$$

where δ is a very small number (close to zero), e.g. 0.0001, so we avoid division by zero.

$\hat{Y}_i$ is obtained as follows:

$$Y_i^d = \frac{1}{2}\log\left(\frac{\pi(X_i)^{d-1}}{1 - \pi(X_i)^{d-1}}\right). \tag{5.27}$$

A vector of weights w_i is computed as follows:

$$w_i^d = \begin{cases} \pi(X_i)^{d-1}\left(1 - \pi(X_i)^{d-1}\right) + \bar{Y}\left|Y_i - \pi(X_i)^{d-1}\right|, & \text{if} \quad |Y_i - \pi(X_i)| < \bar{Y}, \\ \pi(X_i)^{d-1}\left(1 - \pi(X_i)^{d-1}\right), & \text{if} \quad |Y_i - \pi(X_i)| \geq \bar{Y}. \end{cases} \tag{5.28}$$

This weighting mechanism aims to penalize by giving less weight to observations whose distance between the observed Y_i and the probability estimates $\pi(X_i)$ is

greater than the mean of the dependent variable. In other words, we penalize observations which are more likely to be misclassified. This weighting mechanism leads to stabilization after very few iterations of the boosting procedure.

Weights must be normalized by dividing by the sum of the vector of weights:

$$w_i^d = \frac{w_i^d}{\sum_{i=1}^n w_i^d}. \tag{5.29}$$

$F(X_{ip}; u)$ has to be trained as weighted least squares with weights w_i:

$$\beta^d = argmin_\beta \sum_{i=1}^{n} w_i \left[z_i^d - \left(\beta_0 + \sum_{p=0}^{P} X_{ip}\beta_p \right) \right]^2. \tag{5.30}$$

$\hat{Y}_i^d$ has to be computed as follows:

$$\hat{Y}_i^d = \hat{Y}_{i-1}^d + \frac{1}{2} F(X_{ip}; u^d). \tag{5.31}$$

And we must update the probabilities:

$$\pi(X_i)^d = \min \left\{ \frac{1}{1 + e^{-2\hat{Y}_i^{d-1}}} + \delta, 1 \right\}. \tag{5.32}$$

The final $\pi(X_i)$ is related to the log-odds through (5.31)

$$\begin{aligned} \pi^d(Y_i = 1|X) &= \frac{1}{1 + e^{-2\hat{Y}_i^{d-1}}}, \\ \pi^d(Y_i = 0|X) &= \frac{1}{1 + e^{2\hat{Y}_i^{d-1}}}. \end{aligned} \tag{5.33}$$

End For
End Algorithm

5.4 Illustrative Data and Descriptive Statistics

In order to illustrate the proposed methodology, we use a publicly available Home Mortgage Disclosure Act (HDMA) cross-section data set, which was collected by the U.S. Government through a survey designed to gather additional information on minority group applicants. The intention was to uncover whether discrimination based on the applicants' race occurs in mortgage lending. The sample has 2381

applicants who were chosen by a simple random sample in Boston, Massachusetts (United States) in 1997-1998.[5]There is an equal number of denials among white and minority applicants in order to provide sufficient power to validate any discrimination.

Table 5.1 describes the variables in the Home Mortgage Disclosure Act (HDMA) cross-section data set.

Variables	**Description**
Dir	debt payment to total income ratio.
Hir	housing expenses to income ratio.
Lvr	ratio of size of loan to assessed value of property.
Css	consumer credit score from 1, as the best score, to 6 as the lowest score.
Mcs	mortgage credit score from 1, as the best score, to 4 as the lowest score.
Uria	1989 Massachusetts unemployment rate in the applicant's industry.
Pbcr	whether the applicant has a public bad credit record.
Dmi	whether the applicant was denied mortgage insurance.
Self	whether the applicant is self-employed.
Single	whether the applicant is single.
Condominium	whether the applicant lives in a condominium.
Black	whether the applicant is black.
Y	which was coded as 1 when the mortgage application was denied, and 0 otherwise.

Table 5.1: Description of the Home Mortgage Disclosure Act (HDMA) cross-section data set.

[5]Even though these data are old, we believe that they are useful to show the implementation and testing of the newly proposed model since the data set contains the required variables to replicate the model proposed by (Munnell et al., 1996)

Variables		**Denied Mortgage Application** ($Y_i = 1$)	**Approved Mortgage Application** ($Y_i = 0$)	**Total**
Dir		0.389	0.323	0.331
Hir		0.29	0.251	0.255
Lvr		0.816	0.727	0.738
Css		3.302	1.955	2.116
Mcs		1.881	1.699	1.721
Uria		4.014	3.742	3.774
Pbcr	No	209 (9.48%)	1996 (90.52%)	2,205
	Yes	76 (43.43%)	99.0 (56.57%)	175
Dmi	No	241 (10.33%)	2091 (89.67%)	2,332
	Yes	44 (91.67%)	4 (8.33%)	48
Self	No	239 (11.36%)	1864 (88.64%)	2,103
	Yes	46 (16.61%)	231 (83.39%)	277
Single	No	144 (9.97%)	1300 (90.03%)	1,444
	Yes	141 (15.06%)	795 (84.94%)	936
Condominium	No	189 (11.16%)	1505 (88.84%)	1,694
	Yes	96 (13.99%)	590 (86.01%)	686
Black	No	189 (9.26%)	1852 (90.74%)	2,041
	Yes	96 (28.32%)	243 (71.68%)	339
Total		285 (11.97%)	2095 (88.03%)	2,380

Mean of continuous covariates in the denied group, in the approved group and in the total. Counts and row proportions are shown for dichotomous covariates.

Table 5.2: Descriptive statistics for the HDMA data set (1997-1998).

From Table 5.2, variables refer to the debt payment to total income ratio (Dir); housing expenses to income ratio (Hir); ratio of size of loan to assessed value of property (Lvr); consumer credit score from 1, as the best score, to 6 as the lowest score (Css); mortgage credit score from 1, as the best score, to 4 as the lowest score (Mcs); whether the applicant has a public bad credit record (Pbcr); whether

the applicant was denied mortgage insurance (Dmi); whether the applicant is self-employed (Self); whether the applicant is single (Single); 1989 Massachusetts unemployment rate in the applicant's industry (Uria); whether the applicant lives in a condominium (Condominium); whether the applicant is black (Black); and finally, the mortgage application (Y), which was coded as 1 when the mortgage application was denied, and 0 otherwise

Table 5.2 above shows the descriptive statistics for the HDMA data set. The last row reveals that a substantial part of the sample has an approved mortgage application (88.03%). The mean ratios corresponding to the debt to total income and housing expenses to income are slightly higher for applicants whose mortgage application was denied, which means that their debt is higher than it is for the other applicants. Additionally, the mean ratio of the size of loan to assessed value of property is almost 9% higher for people with a denied mortgage application. The credit score and mortgage score of approved mortgage applicants are, respectively, 0.6 times and 0.88 better than the scores of denied applicants. Whereas 56.57% of applicants with a bad public credit record were approved, 43.43% were denied. Moreover, 8.33% of applicants who were denied mortgage insurance had an approved mortgage application, while 91.67% were also denied their mortgage application. While 83.39% of self-employed applicants were approved, 88.65% of applicants who were not self-employed were approved. Also, 84.94% of single applicants were approved, while 15.06% were not. There is a slight percentage difference between applicants who live in a condominium and had an approved mortgage application and applicants who live in a condominium and had a denied mortgage application. Lastly, 71.68% of black applicants were approved, while 90.74% of non-black applicants were approved.

5.5 Results and Discussion

This section contains two parts. The first part presents the results of the prediction performance of the Synthetic Penalized Logitboost in comparison to the algorithms described in Section 5.3. The results are shown below based on three calculations. The second part presents a proposal to recover the interpretability of the Synthetic Penalized Logitboost model.

5.5.1 Prediction Performance

Table 5.3 presents the root-mean-square error [6] (RMSE) of Logistic regression, Logitboost, Gradient Tree Boost and the Synthetic Logitboost, tested for three scenarios: the entire sample (all observations), the observations that correspond to $Y_i = 1$, and the observations that correspond to $Y_i = 0$. The RMSE is suitable to measure the distance between the observed Y_i and the predicted $\hat{Y}_i$, so the predictive performance will not depend for example on the precision of the threshold picked to build a confusion matrix.

The Gradient Boost (tree) is built with the model developer's default hyperparameters from the gbm package in R, which correspond to the number of trees (100), the maximum depth of variable interactions (1), the minimum number of observations in the terminal nodes of the trees (10), and shrinkage (0.1). The Gradient Boost (tree) GS-CV is built with 10-fold cross validation and optimized hyperparameters through grid search, which correspond to the number of trees (150), the maximum depth of variable interactions (2), the minimum number of observations in the terminal nodes of the trees (10), and shrinkage (0.1) with the caret package in R. Logistic, Logitboost, and Synthetic Penalized Logitboost are built according to the definitions in Section 5.3, and they do not have hyperparameters.

In the first calculation, Logistic regression and Logitboost perform almost the same, confirming numerically what was noted theoretically. Synthetic Penalized Logitboost has a smaller RMSE in some of the lowest and highest accumulated predictions, even when it is compared with the Gradient Tree Boosting models (with and without optimized hyperparameters).

When analysing the observations that correspond to denied applications ($Y_i = 1$), both Gradient Tree Boost models perform worse than Logistic and Logitboost for some high score predictions. This confirms the fact that optimized Gradient Tree Boost methods risk failing to predict the minority class ($Y_i = 1$) even when their performance is better with the complete data set. However, the Synthetic Penalized Logitboost performs better than Logistic and Logitboost in the lowest accumulated predictions, and better than the Gradient Tree Boost GS-CV in the 1% and 5% highest accumulated predictions.

[6] The root-mean-square error is calculated as follows: $\sqrt{\sum_{i=1}^{n} \frac{(Y_i - \hat{Y}_i)^2}{n}}$.

RMSE (All Observed)												
Methods	**Lower Extreme**						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
Logistic	0.0067	0.2025	0.1696	0.1635	0.1614	0.1605	0.2739	0.4143	0.4510	0.4236	0.3884	0.3661
Logitboost	0.0064	0.2026	0.1697	0.1635	0.1614	0.1575	0.2739	0.4143	0.4511	0.4236	0.3884	0.3649
Gradient Tree Boost	0.0266	0.0931	0.1566	0.1438	0.1439	0.1687	0.1975	0.4020	0.4595	0.4351	0.3905	0.3583
Gradient Boost (tree) GS - CV	0.0182	0.0195	0.1432	0.1362	0.1236	0.1468	0.1918	0.3391	0.4235	0.4083	0.3698	0.3426
Synthetic Penalized Logitboost	0.0094	0.1568	0.1568	0.1631	0.1610	0.1540	0.2711	0.4138	0.4569	0.4297	0.3890	0.3678
RMSE (When $Y_i = 1$)												
Methods	**Lower Extreme**						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
Logistic	0.9875	0.9802	0.9693	0.9527	0.9368	0.9173	0.0051	0.0110	0.0402	0.1475	0.2673	0.3625
Logitboost	0.9879	0.9808	0.9701	0.9536	0.9379	0.9185	0.0051	0.0108	0.0404	0.1474	0.2673	0.3626
Gradient Tree Boost	0.9706	0.9663	0.9616	0.9503	0.9304	0.9063	0.0150	0.0413	0.0747	0.1661	0.2771	0.3605
Gradient Boost (tree) GS - CV	0.9779	0.9732	0.9671	0.9517	0.9273	0.8930	0.0112	0.0321	0.0498	0.0898	0.1613	0.2441
Synthetic Penalized Logitboost	0.9836	0.9775	0.9684	0.954	0.9393	0.9210	0.0061	0.0147	0.0502	0.1526	0.2777	0.3770

Table 5.3: Root-Mean-Square Error of Logistic regression, Logitboost, Gradient Tree Boost and the Synthetic Penalized Logitboost, tested for the entire sample, when $Y_i = 1$, and when $Y_i = 0$.

Table 5.3 (Continued)

RMSE (When $Y_i = 0$)

Methods	Lower Extreme						Upper Extreme					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
Logistic	0.0064	0.0113	0.0151	0.0205	0.0251	0.0296	0.7233	0.4759	0.3694	0.2797	0.2352	0.2069
Logitboost	0.0061	0.0109	0.0146	0.0198	0.0244	0.0289	0.1210	0.0889	0.0739	0.0584	0.049	0.0426
Gradient Tree Boost	0.0266	0.0270	0.0285	0.0310	0.0331	0.0353	0.6737	0.4504	0.3521	0.2665	0.2240	0.1968
Gradient Boost (tree) GS - CV	0.0180	0.0194	0.0207	0.0231	0.0251	0.0272	0.7044	0.4596	0.3545	0.2651	0.2209	0.1935
Synthetic Penalized Logitboost	0.0092	0.0136	0.0168	0.0214	0.0252	0.0290	0.7055	0.4731	0.3676	0.2777	0.2334	0.2051

Three calculations are presented above. The first set of results is displayed in the top part of Table 5.3 (All Observed Y), where RMSE is calculated for all observations in the sample. The second set of results is displayed in the middle of Table 5.3 (When $Y = 1$) only for observations that correspond to denied applications, whereas the third set of results is displayed in the bottom part of Table 5.3 (When $Y = 0$) only for observations that correspond to approved applications. All results are analysed by groups of scores. So, each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the lowest accumulated prediction scores is shown on the left-hand side of the table under "Lower Extreme", and each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the highest accumulated prediction scores is shown on the right-hand side of the table under "Upper Extreme".

RMSE for Testing Data Set												
Methods	**Lower Extreme**						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
Logistic	0.0059	0.0091	0.0113	0.2582	0.2730	0.2579	0.1650	0.4374	0.4276	0.4239	0.3874	0.3645
Logitboost	0.0069	0.1812	0.1274	0.1279	0.1051	0.1275	0.0000	0.4006	0.4449	0.4051	0.396	0.3602
Gradient Boost (tree)	0.4880	0.2312	0.1648	0.1647	0.1647	0.2150	0.042	0.4361	0.4618	0.4563	0.4151	0.3702
Gradient Boost (tree) GS - CV	0.0114	0.0135	0.1656	0.1841	0.1781	0.1838	0.0477	0.3956	0.4519	0.4179	0.3966	0.3659
Synthetic Penalized Logitboost	0.0064	0.0078	0.1272	0.0906	0.1041	0.1270	0.0001	0.4354	0.4369	0.4149	0.4079	0.3750
RMSE for Training Data Set												
Methods	**Lower Extreme**						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
Logistic	0.0054	0.1982	0.1981	0.1975	0.1620	0.1412	0.3856	0.4017	0.4472	0.4136	0.3658	0.3444
Logitboost	0.007	0.2033	0.2190	0.1943	0.1910	0.1851	0.2541	0.3964	0.4499	0.4245	0.3892	0.3645
Gradient Boost (tree)	0.2825	0.1798	0.1565	0.1680	0.1565	0.1790	0.0512	0.3711	0.4449	0.4274	0.3847	0.3698
Gradient Boost (tree) GS - CV	0.01	0.0124	0.0146	0.0183	0.0783	0.0953	0.0516	0.2964	0.4148	0.4082	0.3738	0.3451
Synthetic Penalized Logitboost	0.0055	0.0084	0.0110	0.1213	0.1587	0.1618	0.0205	0.4362	0.4594	0.4364	0.3991	0.3792

The HMDA database was randomly split into training data (70%) and testing data (30%). Each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the lowest accumulated prediction scores is shown on the left-hand side of the table under "Lower Extreme", and each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the highest accumulated prediction scores is shown on the right-hand side of the table under "Upper Extreme".

Table 5.4: Root-Mean-Square Error of Logistic regression, Logitboost, Gradient Tree Boost and the Synthetic Penalized Logitboost for the training and testing HMDA data sets.

Testing Data Set				
Predictive Measures	**Logistic Regression**	**Logitboost**	**Gradient Boost (Tree) GS - CV**	**Synthetic Penalized Logitboost**
Recall	0.7000	0.7333	0.7333	0.6556
Specificity	0.7837	0.8269	0.8381	0.8446
Accuracy	0.7731	0.8151	0.8249	0.8207
Precision	0.3182	0.3793	0.3952	0.3782
F1 Score	0.4375	0.5	0.5136	0.4797
RMSE	0.2774	0.2654	0.2669	0.3327
Training Data Set				
Predictive Measures	**Logistic Regression**	**Logitboost**	**Gradient Boost (Tree) GS - CV**	**Synthetic Penalized Logitboost**
Recall	0.7026	0.7026	0.7282	0.6872
Specificity	0.8090	0.8110	0.8525	0.7967
Accuracy	0.7965	0.7983	0.8379	0.7839
Precision	0.3278	0.3301	0.3955	0.3095
F1 Score	0.4470	0.4492	0.5126	0.4268
RMSE	0.2757	0.2757	0.2590	0.2780

The HMDA database was randomly split into training data (70%) and testing data (30%). The threshold used to convert the continuous response into a binary response is the mean of the outcome variable. Recall measures the ratio of applicants who were classified in the denied mortgage application group to those who were effectively denied. Specificity measures the ratio of applicants who were classified in the denied group to those who were not denied. Accuracy measures the proportion of applicants who are correctly classified. Precision is the ratio of correctly predicted denied applicants to the total predicted denied applicants. The F1 Score is the weighted average of Precision and Recall.

Table 5.5: Models that meet the C-ROC criterion are bold character when only the first six models are considered.

When analysing the observations that correspond to accepted applications ($Y_i = 0$), Logitboost differs considerably from Logistic in the highest predictions, where it performs much better, while in the lowest scores, the results are very similar for both models. Now, Gradient Tree Boost GS-CV performs better than the two classical methods, while Synthetic Penalized Logitboost also generally performs better

than the classical methods.

It can be concluded that Synthetic Penalized Logitboost makes slightly more accurate predictions than the other algorithms in most observations for the scores in the upper and lower extremes.

The second calculation in Table 5.4 shows the RMSE of the previously discussed methods split into testing and training HMDA data sets. The Synthetic Penalized Logitboost performs quite similarly in the training and testing data sets. This result might be explained by the fact that the algorithm is built with an error term that allows for random variation in covariates when modelling the target variable; and consequently, it avoids overfitting. A similar behaviour is obtained with logistic regression, which is a parametric model. Gradient Tree Boost requires hyperparameter optimization and cross-validation procedures to correct overfitting.

While correction methods to avoid overfitting are widely accepted in the machine learning literature, it is risky in terms of interpretation to tune shrinkage parameters. As their values increase, they deliberately shrink or disappear variables (nodes) with smaller entropy or Gini impurity. However, empirical econometric analysis demands the measurement of the coefficient estimates even when they are not significant in the model; otherwise the analyst may lose control of their natural effect on the dependent variable.

The third calculation in Table 5.5 presents the predictive measures of the discussed methods. The Synthetic Penalized Logitboost has more accuracy than Logistic and Logitboost and more specificity than Gradient Boost (Tree) GS-CV in the testing data sets. In aggregate terms, the Synthetic Penalized Logitboost has larger RMSE than alternative methods. Note that the error correction through penalization is focused on observations which are far from the average values, so the proposed method tends not to affect the predictive improvement of mean observations.

We observe quite similar patterns of performance when the Synthetic Penalized Logitboost is applied to data sets that have low frequencies, for example, in HDMA 2012 and 2017. The results obtained for the testing and training data sets are very close to each other and do not differ significantly. Moreover, the Synthetic Penalized Logitboost has lower RMSE than the alternative methods in the 1% and/or 5% lower and upper extremes. Further details and discussion of results obtained with HDMA 2012 and 2017 are presented in the Appendix.

Figure 5.1 shows the evolution of the RMSE within 100 iterations of the Synthetic Penalized Logitboost. This algorithm gets the RMSE stable after many iterations. While there is no theoretical guarantee that the proposed method will stabilize after some iterations, we obtained similar behaviour when applying the Synthetic Penalized Logitboost to the HDMA 2012 and 2017 data sets. We propose trying alternative initial values if this does not happen.

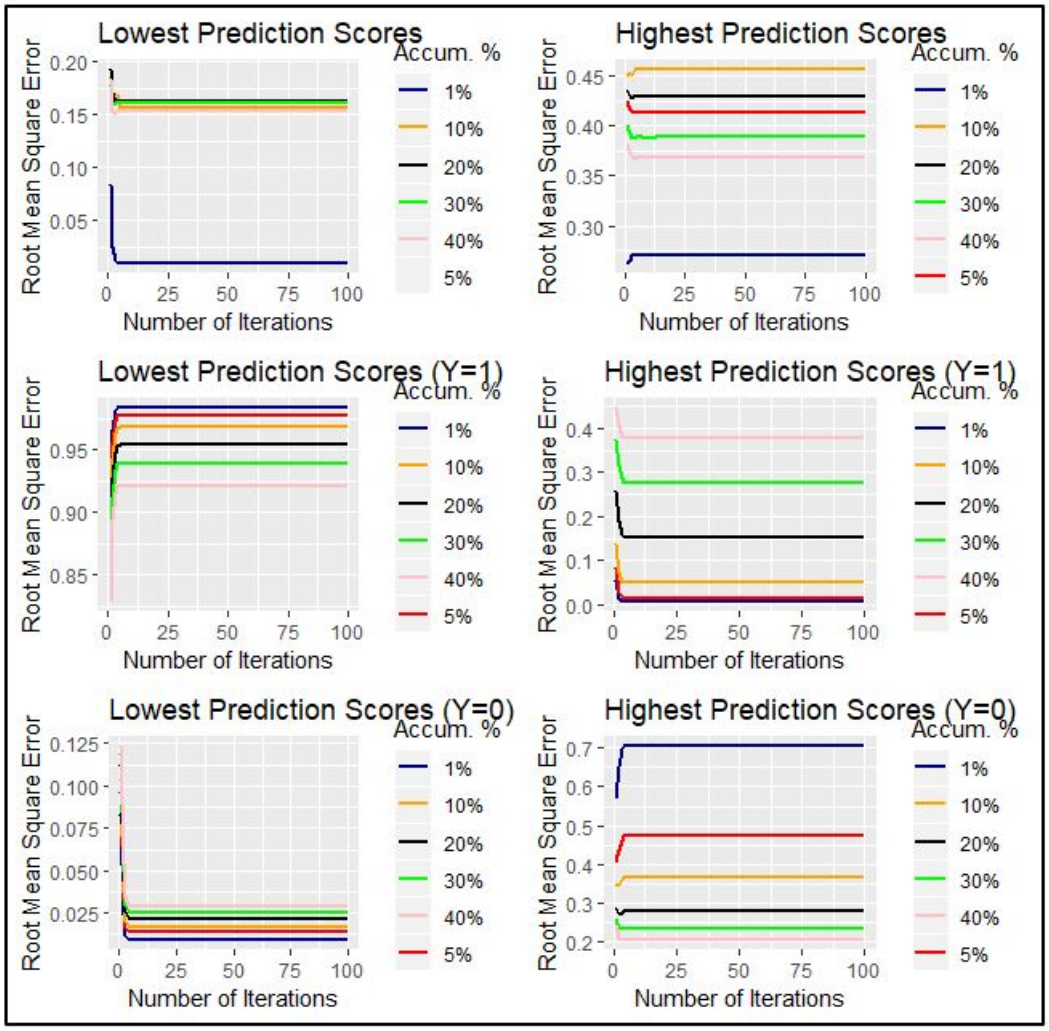

Figure 5.1: RMSE data set across 100 iterations of the Synthetic Penalized Logitboost for the HDMA data set.

The RMSE is smaller and more homogeneous for observations in the minority group ($Y_i = 1$) in the lowest predictions, while the RMSE is larger and more heterogeneous for observations in the majority group ($Y_i = 0$) in the highest predictions. In aggregate terms, the lowest 1% and the highest 1% of predicted scores (extreme

values) have a much more accurate performance than the other accumulated percentages of predictions.

5.5.2 Recovering the interpretability of the model

Machine learning algorithms are sometimes considered black boxes since their interpretability is not straightforward. In contrast, the Synthetic Penalized Logitboost can be seen as a method that recalibrates a least square regression in reweighted versions and penalizes incorrect predictions, so its interpretability can be recovered. Let us note again in Figure 5.1 that when the RMSE achieves stabilization in the boosting procedure (minimum variance), so too do the coefficient estimates of the model. Therefore, if the coefficients are averaged, one might gain some intuition about the sign and magnitude of the covariate effect on the response.

Table 5.6 shows the coefficient estimates obtained by a logistic regression and the Synthetic Penalized Logitboost. The results obtained by the logistic regression are consistent with the conclusions obtained by (Munnell et al., 1996). Moreover, the sign of the mean of the coefficient estimates of the Synthetic Penalized Logitboost within iterations is almost the same before and after the stabilization. The signs and the magnitude of the coefficients are consistent with the ones obtained by logistic regression. Nonetheless, the magnitude seems to be expressed on another scale, which was expected since the target variable used in the two methods is not the same.

Regarding the economic interpretation, Table 5.6 provides interesting results. Both the applicants with a high debt payment to income ratio and the applicants with a high ratio of size of loan to assessed value of property are more likely to receive a denied mortgage application. Moreover, the applicants with the lowest consumer and mortgage credit scores are more likely to be denied. Single applicants are more likely than non-single applicants to have a denied mortgage application. A higher unemployment rate in the applicant's industry is also more likely to result in denial. Last but not least, black applicants are more likely than others to have a denied mortgage application, even when controlling for all the ratios and factors included in the model. The Synthetic Penalized Logitboost provides similar interpretations, as the mean coefficients have almost the same sign[7] as the logistic

[7]Note that the mean of the coefficient estimates of Pbcr and Condominium differ from the logistic regression. However, the effect of Pbcr is similar to the findings of (Munnell et al., 1996), and the effect of Condominium should be analysed in depth since more types of living spaces (more and less expensive) must be controlled for to verify payment guarantee.

regression coefficients, even though they are not directly comparable in size.

Coefficient Estimates	Logistic Regression			Synthetic Penalized Logitboost			
	Lower Bound	Estimate	Upper Bound	Min	Mean	Max	Mean (After Stabilization)
Intercept*	-8.2496	-7.1289	-6.061	-3.0488	-0.0641	0.0027	-0.0037
Dir *	2.7441	4.7742	6.8259	-0.0196	0.043	1.8492	0.0016
Hir	-2.8311	-0.4221	2.0323	-0.4991	-0.0166	0.0036	-0.0043
Lvr *	0.8324	1.7980	2.7881	-0.0086	0.0122	0.4326	0.0009
Css *	0.2168	0.2948	0.3726	0.0000	0.0031	0.1243	0.0003
Mcs *	3.538	4.5154	5.7623	-0.0008	0.0027	0.0751	0.0005
Pbcr	-0.0334	0.2464	0.5243	-0.0153	0.0122	0.807	0
Dmi *	0.8239	1.2281	1.6259	-0.0066	0.0449	2.8376	0.0013
Self *	0.1972	0.6224	1.0305	-0.0003	0.0066	0.2217	0.0007
Single *	0.1015	0.4078	0.7141	-0.0009	0.0055	0.1524	0.0011
Uria *	0.0002	0.0687	0.1336	-0.0001	0.0007	0.0232	0.0001
Condominium	-0.3677	-0.0320	0.2970	-0.0142	-0.0002	0.0053	0.0001
Black *	0.3707	0.7266	1.0753	-0.0002	0.0081	0.3526	0.0006

*The logistic regression columns show the point estimates of the lower and upper bounds of a 95% confidence interval. The XGBoost columns show the means of the coefficient estimates with a linear boosting of the D iterations. Similarly, the bounds are presented with the minimum and maximum values in the iterations. The stabilization starts from the fourth iteration onwards. * indicates that the coefficient is significant at the 90% confidence level in the logistic regression estimation. The calculations were performed in R and scripts are available from the authors.*

Table 5.6: Coefficient Estimates for the Logistic Regression and the Synthetic Penalized Logitboost in the HDMA data set.

5.6 Conclusions

We borrowed the mortgage lending model specification put forward by (Munnell et al., 1996) to provide a real-life application in empirical economics using the proposed algorithm. We conclude that weighting corrections in machine learning algorithms with an econometric base learner can improve the predictive performance by decreasing the RMSE in several segments of the predictions. The Synthetic Penalized Logitboost preserves a stochastic term and trains a weighted linear regression

as base learner in order to prevent overfitting. Hence, the algorithm can be used to reproduce alternative data sets without losing power.

Although the improvement in predictive performance is not excessively high, we provide evidence that it can lead to smaller RMSE than the Gradient Tree Boost (recognized for smartly capturing non-linearities) for observations that belong to the minority class in imbalanced data problems that tend to be underestimated by econometric methods and machine learning algorithms in general.

Beyond that, empirical sciences face challenges with machine learning architecture when their purpose is not only to make predictions using imbalanced data, but also to explain their causes in detail. On one hand, economists have used econometrics thus far to analyse the determinants of a specific phenomenon, but some models tend to be simplified due to the rigidity of linear specifications in most classical models. On the other hand, machine learning handles more large-scale complex data accurately but cannot provide direct coefficient estimates to link the corresponding effects of exogenous variables on the response outcome. The Synthetic Penalized Logitboost has started to combine these two approaches by providing some statistical intuition of its coefficient estimates since the base learner is a weighted least squares regression. As a result, the model always stabilizes its coefficients, while also being able to deal with complex structures and imbalanced phenomena.

Since the Synthetic Penalized Logitboost strongly penalizes observations whose probability estimates deviate considerably from the observed target variable, we wonder whether the predictive performance could be further improved in more imbalanced data sets or more complex models than the one presented here. While the model specification in (Munnell et al., 1996) works with tailor-made survey data, our proposed model can also work with extensive data obtained through web scraping or with device-collected data.

Appendix A

This section provides the results of the prediction performance of the Synthetic Penalized Logitboost in comparison to the algorithms described in Section 5.3 for the HDMA 2012 and HDMA 2017 datasets.

Table 5.7 shows the RMSE of Logistic regression, Logitboost, Gradient Tree Boost and Synthetic Penalized Logitboost for the training and testing HDMA 2012 data sets. The Synthetic Penalized Logitboost has a lower RMSE than the other methods, especially in the 1% and 5% of lower and upper extremes. The second-best prediction performance for the lower extremes corresponds to the results obtained by the Logistic and Logitboost with a prediction error equal to zero, while second-best for the upper extremes corresponds to the Gradient Boost (tree) GS-CV. Moreover, the Synthetic Penalized Logitboost has a similar performance in the testing and training data sets.

Table 5.8 presents additional predictive measures of Logistic regression, Logitboost, Gradient Tree Boost and the Synthetic Penalized Logitboost for the testing and training HDMA 2012 data sets. The Synthetic Penalized Logitboost has the highest recall with similar rates in the training and testing data sets. The second highest recall corresponds to the Gradient Boost (tree) GS – CV.

Figure 5.2 shows RMSE across 100 iterations of the Synthetic Penalized Logitboost for the HDMA 2012. Approximately the first 5 to 10 iterations have brusque changes, however after iteration 30 approximately the RMSE gets stable.

Table 5.9 shows the RMSE of Logistic regression, Logitboost, Gradient Tree Boost and Synthetic Penalized Logitboost for the training and testing HDMA 2017 data sets. All methods have a prediction error equal to zero in the lower extreme, while the Penalized Logitboost and Logistic regression have the smallest RMSE in the upper extremes. Additionally, Table 5.10 presents alternative predictive measures for the mentioned methods. The Synthetic Penalized Logitboost has again the highest recall, even when RMSE in aggregated terms is higher than others. Finally, Figure 5.3 shows that the RMSE gests stable after iteration 40 approximately.

Considering the results examined for HMDA, HMDA 2012 and HDMA 2017, the Synthetic Penalized Logitboost increases the true positive rate when predicting a model, in particular in the most extreme observations. And it can reach convergence after some iterations in the boosting procedure.

RMSE for Testing Data Set												
Methods	**Lower Extreme**						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
Logistic	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4940	0.4890	0.4760	0.4630	0.4520	0.4410
Logitboost	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4950	0.4900	0.4770	0.4630	0.4520	0.4410
Gradient Boost (tree)	0.0004	0.0010	0.0030	0.0030	0.0040	0.0040	0.5080	0.4880	0.4770	0.4610	0.4490	0.4390
Gradient Boost (tree) GS - CV	0.0460	0.0240	0.0170	0.0160	0.0130	0.0110	0.4790	0.4770	0.4710	0.4600	0.4510	0.4410
Synthetic Penalized Logitboost	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4730	0.4620	0.4640	0.4550	0.4480	0.4430
RMSE for Training Data Set												
Methods	**Lower Extreme**						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
Logistic	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4950	0.4890	0.4770	0.4640	0.4520	0.4400
Logitboost	0.0150	0.0070	0.0070	0.0060	0.0050	0.0040	0.4960	0.4890	0.4780	0.4640	0.4520	0.4410
Gradient Boost (tree)	0.0010	0.0010	0.0030	0.0040	0.0040	0.0050	0.5040	0.4890	0.4770	0.4630	0.4510	0.4400
Gradient Boost (tree) GS - CV	0.0380	0.0220	0.0170	0.01600	0.0130	0.0120	0.4780	0.4770	0.4720	0.4600	0.4510	0.4400
Synthetic Penalized Logitboost	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5000	0.4850	0.4760	0.4630	0.4510	0.4410

The HMDA database was randomly split into training data (70%) and testing data (30%). Each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the lowest accumulated prediction scores is shown on the left-hand side of the table under "Lower Extreme", and each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the highest accumulated prediction scores is shown on the right-hand side of the table under "Upper Extreme". The Gradient Boost (tree) GS – CV is built with 10-fold cross validation and optimized hyperparameters through grid search.

Table 5.7: Root-Mean-Square Error of Logistic regression, Logitboost, Gradient Tree Boost and the Synthetic Penalized Logitboost for the training and testing HMDA 2012 data sets.

Testing Data Set				
Predictive Measures	**Logistic Regression**	**Logitboost**	**Gradient Boost (Tree) GS - CV**	**Synthetic Penalized Logitboost**
Recall	0.9946	0.9954	0.9978	0.9981
Specificity	0.6552	0.6548	0.6532	0.6511
Accuracy	0.6939	0.6935	0.6923	0.6905
Precision	0.2710	0.2698	0.2693	0.2682
F1 Score	0.4259	0.4245	0.4241	0.4228
RMSE	0.2851	0.2845	0.2842	0.2877
Training Data Set				
Predictive Measures	**Logistic Regression**	**Logitboost**	**Gradient Boost (Tree) GS - CV**	**Synthetic Penalized Logitboost**
Recall	0.9955	0.9963	0.998	0.9983
Specificity	0.6548	0.6541	0.653	0.6510
Accuracy	0.6935	0.6928	0.692	0.6902
Precision	0.2694	0.2685	0.2682	0.2671
F1 Score	0.4240	0.4230	0.4228	0.4214
RMSE	0.2844	0.2841	0.2837	0.2875

The HMDA database was randomly split into training data (70%) and testing data (30%). The threshold used to convert the continuous response into a binary response is the mean of the outcome variable.

Table 5.8: Predictive measures of Logistic regression, Logitboost, Gradient Tree Boost and the Synthetic Penalized Logitboost for the testing and training HMDA 2012 data sets.

RMSE for Testing Data Set												
Methods	**Lower Extreme**						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
Logistic	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4266	0.4216	0.4061	0.3957	0.391	0.3769
Logitboost	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4282	0.4226	0.4108	0.3991	0.3922	0.3767
Gradient Boost (tree)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4514	0.4157	0.4074	0.3969	0.3854	0.3738
Gradient Boost (tree) GS - CV	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4240	0.4193	0.4098	0.3989	0.3885	0.3768
Synthetic Penalized Logitboost	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4266	0.4216	0.4061	0.3957	0.391	0.3769
RMSE for Training Data Set												
Methods	**Lower Extreme**						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
Logistic	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4435	0.4234	0.4115	0.3995	0.3925	0.3768
Logitboost	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4427	0.423	0.4115	0.3994	0.3926	0.3768
Gradient Boost (tree)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4797	0.4478	0.4234	0.4110	0.397	0.3762
Gradient Boost (tree) GS - CV	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4406	0.4227	0.4048	0.3934	0.3856	0.3770
Synthetic Penalized Logitboost	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4435	0.4234	0.4115	0.3995	0.3925	0.3768

The HMDA database was randomly split into training data (70%) and testing data (30%). Each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the lowest accumulated prediction scores is shown on the left-hand side of the table under "Lower Extreme", and each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the highest accumulated prediction scores is shown on the right-hand side of the table under "Upper Extreme". The Gradient Boost (tree) GS – CV is built with 10-fold cross validation and optimized hyperparameters through grid search.

Table 5.9: Root-Mean-Square Error of Logistic regression, Logitboost, Gradient Tree Boost and the Synthetic Penalized Logitboost for the training and testing HMDA 2017 data sets.

Testing Data Set				
Predictive Measures	**Logistic Regression**	**Logitboost**	**Gradient Boost (Tree) GS - CV**	**Synthetic Penalized Logitboost**
Recall	0.9983	0.9983	0.9983	0.9981
Specificity	0.6604	0.6604	0.6604	0.6605
Accuracy	0.6839	0.6839	0.6839	0.6840
Precision	0.1805	0.1805	0.1805	0.1805
F1 Score	0.3057	0.3057	0.3057	0.3057
RMSE	0.2385	0.2385	0.2378	0.2387
Training Data Set				
Predictive Measures	**Logistic Regression**	**Logitboost**	**Gradient Boost (Tree) GS - CV**	**Synthetic Penalized Logitboost**
Recall	0.9983	0.9983	0.9983	0.9984
Specificity	0.6637	0.6637	0.6637	0.6637
Accuracy	0.6870	0.6870	0.6870	0.6870
Precision	0.1817	0.1817	0.1817	0.1817
F1 Score	0.9983	0.3074	0.3074	0.3074
RMSE	0.2382	0.2382	0.2374	0.2383

The HMDA database was randomly split into training data (70%) and testing data (30%). The threshold used to convert the continuous response into a binary response is the mean of the outcome variable.

Table 5.10: Predictive measures of Logistic regression, Logitboost, Gradient Tree Boost and the Synthetic Penalized Logitboost for the testing and training HMDA 2017 data sets.

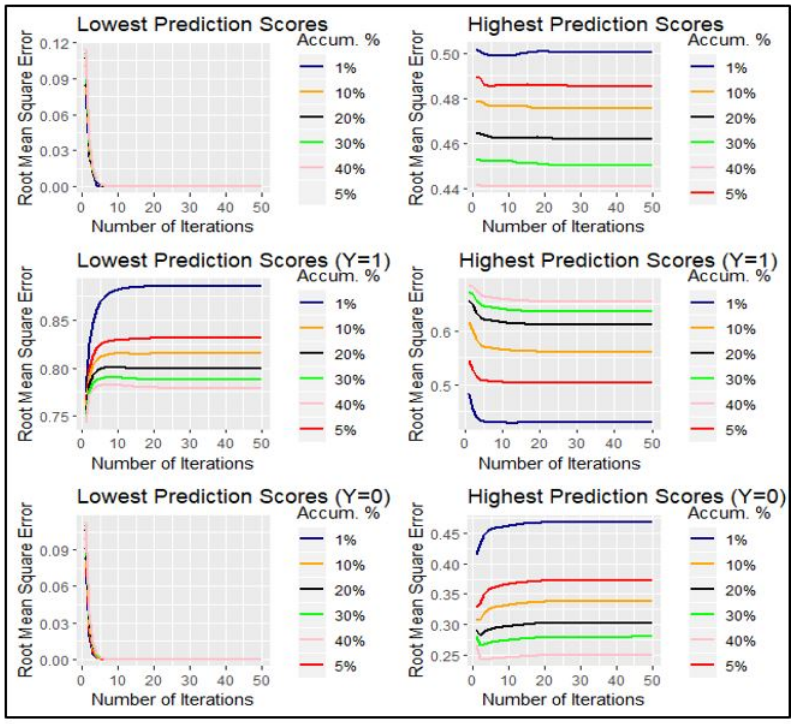

Figure 5.2: RMSE across iterations of the Synthetic Penalized Logitboost for the HDMA 2012.

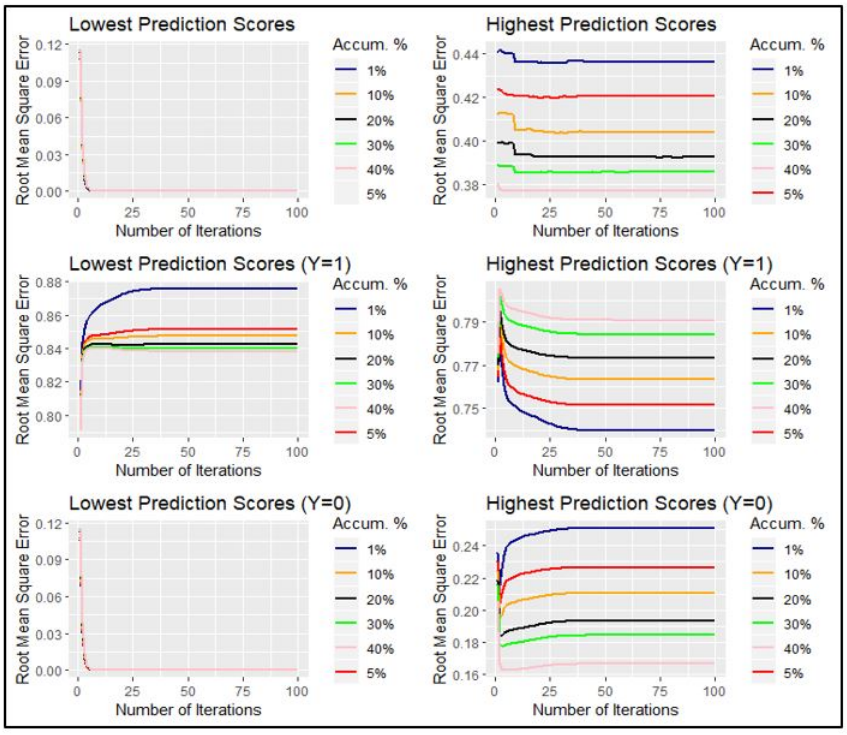

Figure 5.3: RMSE across iterations of the Synthetic Penalized Logitboost for the HDMA 2017.

Chapter 6: RiskLogitboost regression for rare events in binary response: An econometric approach

6.1 Introduction

Research on rare events is steadily increasing in real-world applications of risk management. Examples include fraud detection (Wei et al., 2013), credit default prediction (Jiang et al., 2018), bankruptcy prediction (Barboza et al., 2017), emerging markets anomalies (Zaremba and Czapkiewicz, 2017), customer churn predictions (Verbeke et al., 2014) and accident occurrence for insurance studies (Ayuso et al., 2014). We address the rare event modeling problem with a purposefully designed method to identify rare potential hazards in advance and facilitate an understanding of their causes.

Rare events are extremely uncommon patterns whose atypical behavior is difficult to predict and detect. A broad consensus King and Zeng (2001); Maalouf and Trafalis (2011); Pesantez-Narvaez and Guillen (2020a) favors the definition of rare events data as binary variables with much fewer events (ones) than non-events (zeros). In other words, the degree of imbalance is more extreme in rare events than it is in class imbalanced data, such that rare events are characterized by the number of ones being hundreds to thousands of times smaller than the number of zeros.

Developing algorithms that can handle rare events powered by the latest machine learning advances faces two important challenges:

This chapter can be found in Pesantez-Narvaez, J., Guillen, M., & Alcañiz, M. (2021). RiskLogitboost Regression for Rare Events in Binary Response: An Econometric Approach. *Mathematics* 9(5), 579.

(i) Some models exhibit bias towards the majority class or underestimate the minority class. Some classifiers are suitable for balanced data (Loyola-González et al., 2016; Krawczyk, 2016) or treat the minority class as noise (Beyan and Fisher, 2015). Moreover, some popular tree-based and boosting-based algorithms have been shown to have a high predictive performance measured only with evaluation metrics that consider all observations equally important (Pesantez-Narvaez et al., 2019).

(ii) Unlike econometric methods, several machine learning methods are considered black boxes in terms of interpretation. They are frequently interpreted using single metrics such as the classification accuracy as unique descriptions of complex tasks (Doshi-Velez and Kim, 2017), and they are not able to provide robust explanations for high-risk environments.

In this paper we address these two challenges in an attempt to predict and explain rare events, which will be referred to as dependent or target variables. We propose a RiskLogitboost regression, which is a Logitboost-based algorithm that leads to the convergence of coefficient estimates after some iterations, as occurs when using Iteratively Re-Weighted procedures. Moreover, bias and weighting corrections are incorporated to improve the predictive capacity of the events (ones).

More specifically, our prediction strategy consists of: (i) increasing the accuracy of minority class prediction, and (ii) building an interpretable model similar to classical econometric models. After the introduction this paper is organized as follows. Section 6.2 presents the background of the three main approaches used in this research: boosting methods for imbalanced data sets, penalized regression models, and interpretable machine learning. Section 6.3 describes in detail the proposed RiskLogitboost regression in the rare event problem framework. Section 6.4 shows the illustrative data used to prove the RiskLogitboost regression. Section 6.5 discusses the results obtained in terms of predictive capacity and interpretability. And, finally Section 6.6 presents the conclusions of the paper.

6.2 Background

Supervised machine learning methods are used to predict a response variable denoted as Y_i, $i = 1, \cdots, n$. The data consist of a sample of n observations of the response, and the prediction is established by a set of covariates denoted as Y_{ip}, $p = 1, \cdots P$, with P predictor variables. The model is trained by a base learner $F\left(X_{ip}; u\right)$, which is a function of covariates Y_{ip} and the parameters represented by

u. The predicted response is denoted as $\hat{Y}_i$.

The purpose of supervised machine learning is to minimize the learning error measured by a loss function φ using an optimization strategy like gradient descent. The loss function is the distance between the observed Y_i and the predicted response $\hat{Y}_i$ which is denoted as $\varphi\left(Y_i, \hat{Y}_i\right)$.

6.2.1 Boosting methods

Boosting methods for additive functions are developed within an iterative process through a numerical optimization technique called gradient descent. Each function minimizes a specified loss function φ. Friedman (2001) applied the boosting strategy to some loss criteria for classification and regression problems[1] such as: least-squares $\left(Y_i - \hat{Y}_i\right)^2$ for the least-squares regression; least absolute-deviation $\left|Y_i - \hat{Y}_i\right|$ for the least-absolute-deviation regression; Huber for M-Regression $0.5\left(Y_i - \hat{Y}_i\right)^2$ if $\left|Y_i - \hat{Y}_i\right| \leq \delta$ or $\delta\left|Y_i - \hat{Y}_i\right| - \delta/2$ otherwise; and the Logistic binomial log-likelihood $\log\left(e^{-2Y_i\hat{Y}_i}\right)$ for two-class Logistic classification.

The Gradient Boosting Machine shown in Algorithm 1 is the base proposal of Friedman (2001). The algorithm initializes with a prediction guess of $\hat{Y}_i^0$. Then a boosting process of D iterations is carried out in four stages: The first transforms the new response denoted as $\tilde{r}_i^d$ computed as the negative gradient of $\varphi\left(Y_i, \hat{Y}_i^d\right)$ at iteration d. The second stage fits a least squares regression with the recently computed $\tilde{r}_i^d$ as the response. The third stage minimizes the loss function between the observed Y_i and $\hat{Y}_i^d + \gamma F\left(X_{ip}; u^d\right)$ and the result is delivered in γ. Finally, the last stage updates the prediction $\hat{Y}_i^d$ by summing $\hat{Y}_i^{d-1}$ and $\gamma F\left(X_{ip}; u^d\right)$.

Adaboost was one of the first boosting-based prediction algorithms (Freund et al., 1996; Freund and Schapire, 1997). It trains the base learner in reweighted version by allocating more weight to misclassified observations. Many other boosting techniques have since been derived, such as RealBoost (Friedman et al., 2000), which allows a probability estimate instead of a binary outcome. Logitboost (Friedman et al., 2000), can be used for two-class prediction problems by optimizing an exponential criterion. Gentle Adaboost (Friedman et al., 2000), builds on Real Adaboost

[1] We use the term "classification problem" if Y_i is qualitative, whereas if Y_i is quantitative, we use the term "regression problem". The latter does not refer to regression models studied in econometrics; it refers to a predictive model.

and uses probability estimates to update functions. Madaboost (Domingo et al., 2000) modifies the weighting system of Adaboost. Brownboost (Freund, 2001) is based on finding solutions to Brownian differential equations. Delta Boosting (Lee and Lin, 2018) uses a delta basis instead of the negative gradient as transformed response.

Algorithm 1 Gradient Boosting Machine

Initial values: $\hat{Y}_i^0 = argmin_\rho \sum_{i=1}^n \varphi(Y_i, \rho)$.
For $d = 1$ to D do:

2.1 Transformation: $\tilde{r}_i^d = - \left. \frac{\partial \varphi\left(Y_i, \hat{Y}_i^d\right)}{\partial \hat{Y}_i^d} \right|_{Y_i = \hat{Y}_i^{d-1}}$.

2.2 Fitting: $u^d = argmin_{u,\varpi} \sum_{i=1}^n \left[\tilde{r}_i^d - \varpi F\left(X_{ip}; u\right)\right]^2$.

2.3 Minimizing: $\gamma^d = argmin_\gamma \sum_{i=1}^n \varphi\left[Y_i, \hat{Y}_i^d + \gamma F\left(X_{ip}; u^d\right)\right]$.

2.4 Updating: $\hat{Y}_i^d = \hat{Y}_i^{d-1} + \gamma^d F\left(X_{ip}; u^d\right)$.

End for

In the context of rare event and imbalanced prediction problems, various boosting-based methods have been proposed in the literature, including but not limited to RareBoost (Joshi et al., 2001), which calibrates the weights depending on the accuracy of each iteration. Asymmetric Adaboost (Viola and Jones, 2001) is a variant of Adaboost and incorporates a cascade classifier. SMOTEBoost (Chawla et al., 2002) incorporates SMOTE (synthetic minority over-sampling techniques) in a boosting procedure. DataBoost-IM (Guo and Viktor, 2004) treats outliers and extreme observations in a separate procedure to generate synthetic examples of majority and minority classes. RUSBoost (Seiffert et al., 2009) trains using skewed data. MSMOTEBoost (Hu et al., 2009) rebalances the minority class and eliminates noise observations. Additional cost-sensitive methods (Fan et al., 1999; Ting, 2000; Wang et al., 2010; Sun et al., 2006, 2007; Masnadi-Shirazi and Vasconcelos, 2010) have been developed by introducing cost items in the boosting procedure.

Other boosting extensions include the tree boosting-based methods, which have been considered a great success due to their predictive capacity in the machine learning community. The tree gradient boost (Friedman, 2001) varies from the original gradient boost in the initial value of the first prediction $\hat{Y}_i^0$, and the use

of a Logistic loss function and a tree base learner.

A tree gradient boost as shown in Algorithm 2 consists of six stages. The first one states the values for the initial prediction, $\hat{Y}_i^0$. The second stage obtains the new transformed response with the negative gradient of a Logistic loss function. The third maps the observations onto J leaves of the tree at iteration d. The tree learner is $\sum_{j=1}^{J} u_j 1\left(X_{ip} \varepsilon R_j\right)$ with J terminal nodes known as leaves, and R_j classification rules (regions), $j = 1, \cdots, J$. Parameters u correspond to the score of each leaf, which is the proportion of cases classified as events given covariates X_{ip}. Gini and entropy are two metrics for choosing how to split a tree. Gini is a measurement of the likelihood of an incorrect classification of a new observation if it were randomly classified according to the distribution of class labels of the covariates. Entropy measures how much information there is in a node.

The fourth stage requires minimizing a Logistic loss function: $argmin_\gamma \sum_{X_j \in R_{id}} \log\left[1 + e^{-2Y_i\left(\hat{Y}_i^{d-1} + \gamma^d\right)}\right]$ delivered in γ_j^d. However, since there is no closed form for γ_j^d, a Newton-Raphson approximation is computed. And finally, the sixth stage updates the final $\hat{Y}_i^d$.

Tree gradient boosting techniques tend to overfit especially when data are complex or highly imbalanced (Pesantez-Narvaez et al., 2019). Regularization is a popular strategy to penalize the complexity of the tree and allow out-of-sample reproducibility. This involves adding a shrinkage penalty or regularization term to the loss function $\phi\left(Y_i, \hat{Y}_i\right)$ so that the leaf scores shrink: $\sum_{i=1}^{n} \varphi\left(Y_i, \hat{Y}_i\right) + \sum_{d=1}^{D} \eta'\left(\hat{Y}_i^d\right)$[2]. Moreover, Breiman et al. (1984) introduced the cost-complexity pruning that penalizes the number of terminal nodes J according to the following expression: $\sum_{i=1}^{n} \varphi\left(Y_i, \hat{Y}_i\right) + \sum_{d=1}^{D} \lambda J$. As a consequence, these strategies seem quite risky for analysts who want to keep the effect of the covariates even when this effect is small or not significant, because after applying regularization or pruning the score of the leaf is arbitrarily shrunk and the correspondingly less important characteristics disappear.

6.2.2 Penalized regression methods

In the econometric setting, regression models have commonly been used to describe the relationship between a response Y_i and a set of covariates X_{ip}. Regression mod-

[2] $\eta' = \lambda \|\dot{u}\|$, where λ is a regularization parameter associated with $L1$-norm or $L2$-norm of the scores vector.

Algorithm 2 Tree Gradient Boost

Initial values: $\hat{Y}_i^0 = \frac{1}{2}\log\left(\frac{1+\bar{Y}}{1-\bar{Y}}\right)$, where $\bar{Y}$ is the mean of Y_i.
For $d = 1$ to D do:

2.1 Transformation: $\tilde{r}_i^d = \frac{2Y_i}{1+e^{2Y_i\hat{Y}_i^{d-1}}}$.

2.2 Mapping: $R_{jd} = j -$ leaf scores $\left(\tilde{r}_i, X_1^n\right)$.

2.3 Minimizing: $\gamma^d = argmin_\gamma \frac{\sum_{X_i \epsilon R_{jd}} \tilde{r}_i}{\sum_{X_i \epsilon R_{jd}} |\tilde{r}_i(2-|\tilde{r}_i|)|}$.

2.4 Updating: $\hat{Y}_i^d = \hat{Y}_i^{d-1} + \sum_{j=1}^J \gamma_j^d 1\left(X_i \varepsilon R_{jd}\right)$.

End for

els are used to predict a target variable $\hat{Y}_i$, and allow interpretability of the coefficients by measuring the effect of the covariates on the expected response.

Logistic regression models are used to model the binary variable Y_i. In fact, Y_i follows a Bernoulli distribution, where π_i is the probability that Y_i equals 1, expressed as follows:

$$\pi_i = \frac{e^{X_i\tilde{\beta}}}{1 + e^{X_i\tilde{\beta}}}. \tag{6.1}$$

Note that $X_i\tilde{\beta}$ is the matrix notation of $\beta_0 + \sum_{p=1}^P X_{ip}\beta_p$, where $\tilde{\beta}$ is the parameter vector. And $1 - \pi_i$ is the probability that Y_i equals 0:

$$\pi_i = \frac{1}{1 + e^{X_i\beta}}. \tag{6.2}$$

The Logistic regression uses a logit function as the linear predictor defined as:

$$\eta_i = \log\left(\frac{\pi_i}{1-\pi}\right) = \beta_0 + \sum_{p=1}^{P} X_{ip}\tilde{\beta}_p. \tag{6.3}$$

Then, the classical likelihood function is the joint Bernoulli probability distribution of observed values of Y_i as follows:

$$\ell\left(\beta_0, \cdots, \beta_p; X_i\right) = \prod_{i=1}^{n}\left[\pi^{Y_i}(1-\pi_i)^{1-Y_i}\right]. \tag{6.4}$$

Taking logarithms of (6.4), and replacing with expressions (6.1) and (6.2) we obtain:

$$\ell\left(\beta_0, \cdots, \beta_p; X_i\right) = \sum_{i=1}^{n} \left[Y_i(X_i\tilde{\beta}) - \log\left(1 + e^{X_i\tilde{\beta}}\right)\right]. \tag{6.5}$$

Then Logistic regression estimates can be found by maximizing the log likelihood from (6.5) or minimizing the negative log likelihood function, which can be seen as a loss function to be minimized. Maximization is achieved by derivating $l\left(\beta_0, \cdots, \beta_p; X_i\right)$ by all the $P + 1$ parameters, obtaining a vector of $P + 1$ partial derivate equations known as the score and denoted as $\bar{\bar{l}}\left(\beta_0, \cdots, \beta_p; X_i\right)$. [3]

$$\bar{\bar{l}}\left(\beta_0, \cdots, \beta_p; X_i\right) = \left[\frac{\partial 1}{\partial \beta_0}, \cdots \frac{\partial 1}{\partial \beta_P}\right]'. \tag{6.6}$$

However, when fitting a simple model like a Logistic regression, it is sometimes the case that many variables are not strongly associated with the response Y_i which lowers the classification accuracy of the model. James et al. (2013) recognized that this problem can be improved with alternative fitting procedures such as constraining or shrinking (also known as regularization) before considering non-linear models. The idea is that complex models are sometimes built with irrelevant variables, but by shrinking coefficient estimates we manage to reduce variance and thus the prediction error.

However, when complex models arise, the machine learning literature suggests imposing some degree of penalty on the Logistic regression so that the variables that contribute less are shrunk through a regularization procedure.

Ridge Logistic regression, shown in Algorithm 3, follows the dynamics of the Logistic regression, but the term $\lambda \left[\sum_{p=1}^{P} \beta_p\right]^2$ known as the regularization penalty is added to the negative likelihood function as in (6.4). Thus covariates with a minor contribution are forced to be close to zero.

On the other hand, Lasso Logistic regression, shown in Algorithm 4, follows the dynamics of the Logistic regression, but a regularization penalty $\lambda \left|\sum_{p=1}^{P} \beta_p\right|$ is added to the negative likelihood function. In this case, less contributive covariates are forced to be exactly zero. In both cases, λ 1 is a shrinkage parameter, so the larger it is, the smaller the magnitude of the coefficient estimates (James et al., 2013).

[3] We denote $'$ to transpose vectors and matrices.

Algorithm 3 Ridge Logistic Regression

Minimizing the negative likelihood function: $L = -\prod_{i=1}^{n} \pi^{Y_i}(1-\pi)^{1-Y_i}$.

Penalizing: $L^* = -\prod_{i=1}^{n} \pi^{Y_i}(1-\pi)^{1-Y_i} + \lambda \left[\sum_{p=1}^{P} \beta_p\right]^2$.

Algorithm 4 Lasso Logistic Regression

Minimizing the negative likelihood function: $L = -\prod_{i=1}^{n} \pi^{Y_i}(1-\pi)^{1-Y_i}$.

Penalizing: $L^* = -\prod_{i=1}^{n} \pi^{Y_i}(1-\pi)^{1-Y_i} + \lambda \left|\sum_{p=1}^{P} \beta_p\right|$.

6.2.3 Interpretable machine learning

Unlike statistical models in econometrics, machine learning algorithms are generally not self-explanatory. For example, generalized linear models provide coefficient estimates and their standard errors give information about the effect of covariates, whereas machine learning requires alternative methods to make the models understandable. Two popular approaches are described below.

Variable importance (VI), as proposed by (Breiman et al., 1984), measures the influence of inputs on the variation of $\hat{Y}_i$. We obtain the importance in a decision tree by summing the improvements in the loss function over all splits on a specific covariate X_p, in other words, variable importance is calculated by the node impurity weighted by the node probability[4]. For ensemble techniques, the (VI) of all the trees that composed the ensemble is averaged.

Partial Dependence Plots (PDP) proposed by (Friedman, 2001) show the marginal effect of a covariate X_p on the prediction. The predicted function $\hat{Y}$ is evaluated in certain values of the specific covariate X_p while averaging over a range of values of all the other covariates.

6.3 The rare event problem with RiskLogitboost regression

The RiskLogitboost regression is an extension of Logitboost (Friedman et al., 2000) that modifies the weighting procedure to improve the classification of rare events. It also adapts a bias correction from McCullagh and Nelder (1983) in the boosting procedure, which is also applied to regression models such as those in (King and

[4]The node probability is calculated by the number of observations contained in that node of the tree divided by total number of observations.

Zeng, 2001; Maalouf and Trafalis, 2011).

To formally define the RiskLogitboost regression, it is described briefly the Logitboost shown in Algorithm 5. It first initializes with $\hat{Y}_i^0 = 0$ and $\pi^0(X_i) = 0.5$. Then the boosting procedure continues with four stages. The first one transforms the response. Logitboost uses the exponential loss function $e^{Y_i \hat{Y}_i}$ which is a quadratic approximation of χ^2 and z_i (transformed response) as well (see further details in Appendix A). The second stage involves calculating the weights by computing the variance of the transformed response $Var\left[z_i|X\right]$ (see further details in Appendix B). The third stage fits a least squares regression with response z_i. Finally, the fourth stage updates the prediction $\hat{Y}_i^d$ and $\pi(X_i)$ by computing $F\left(X_{ip}; u^d\right)$ and $X_i\tilde{\beta}$ for this particular case.

Algorithm 5 Logitboost

Initial values: $\hat{Y}_i^0 = 0$, $\pi^0(X_i) = 0.5$, where $\pi(X)$ are the probability estimates.
For $d = 1$ to D do:

2.1 Transformation: $z_i^d = \frac{Y_i^{d-1} - \pi(X_i)^{d-1}}{\pi(X_i)^{d-1}\left(1-\pi(X_i)^{d-1}\right)}$.

2.2 Weighting: $w_i^d = \pi(X_i)^{d-1}\left(1 - \pi(X_i)^{d-1}\right)$.

2.3 Minimizing: $\beta^d = argmin_\beta \sum_{i=1}^n w_i^d \left[z_i^d - \left(\beta_0 + \sum_{p=1}^P \beta_p\right)\right]^2$.

2.4 Updating: $\hat{Y}_i^d = \hat{Y}_i^{d-1} + \frac{1}{2}F\left(X_{ip}; u^d\right)$, and

$$\pi(X_i)^d = \frac{e^{\hat{Y}_i^{d-1}}}{e^{\hat{Y}_i^{d-1}} + e^{-\hat{Y}_i^{d-1}}}.$$

End for

6.3.1 RiskLogitboost regression weighting mechanism to improve rare-class learning

The proposed weighting mechanism might be considered as a mixed case of oversampling and undersampling. The main idea is to overweight observations whose estimated probability $\pi(X_i)$ is farther from the observed value Y_i, in other words, observations that are more likely to be misclassified. The new majority class observations are interpolated through a threshold that determines the calibration of weights. The proposed weighting mechanism takes the following form:

$$w_i^* = \begin{cases} [\pi(X_i)\,(1-\pi(X_i))]\,(1+|Y_i-\pi(X_i)|)\,, & \text{if} \quad |Y_i-\pi(X_i)| > \bar{Y}, \\ [\pi(X_i)\,(1-\pi(X_i))]\,(1-|Y_i-\pi(X_i)|)\,, & \text{if} \quad |Y_i-\pi(X_i)| \leq \bar{Y}. \end{cases}$$

The original weights w_i of the Logitboost are now multiplied by a factor $1 \pm |Y_i - \pi(X_i)|$ that is related to the distance between Y_i and $\pi(X_i)$.

Figure 6.1 shows the relationship between weights according to the estimated probabilities of the Logitboost and the RiskLogitboost regression. Logitboost overweights observations whose estimated probability is around 0.5 and then decreases gradually and symmetrically on either side. The result of the weighting mechanism in the RiskLogitboost regression shows that low estimated probabilities are overweighted when $Y_i = 1$ while high estimated probabilities are underweighted when $Y_i = 0$. In Figure 6.1 we show that once the weighting mechanism is transformed, we maintain the u – inverted shape for $Y = 1$ and $Y = 0$.

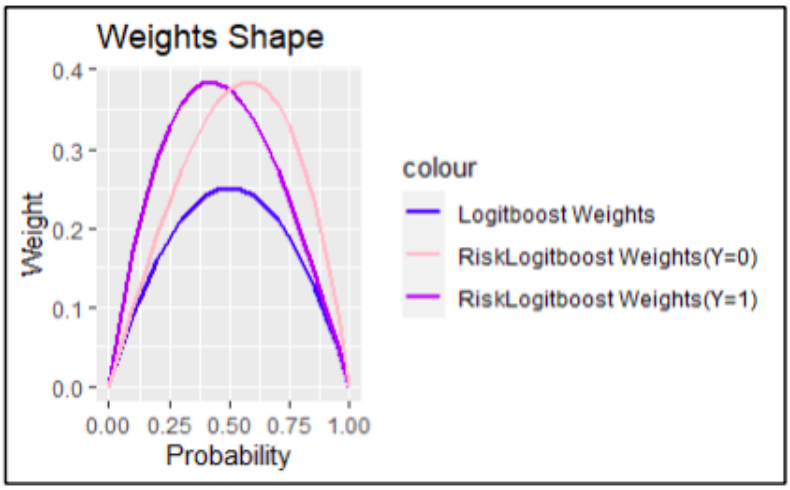

Figure 6.1: Plot of weights versus estimated probabilities of the Logitboost and the RiskLogitboost regression.

Pesantez-Narvaez and Guillen (2020a,b); Pesantez-Narvaez et al. (2021); Mease et al. (2007); Liska et al. (2019) proposed weighting mechanisms for parametric and non-parametric models to improve the predictive performance of imbalanced and rare data.

6.3.2 Bias correction with weights

Bias correction will lead to a lower root mean square error. McCullagh and Nelder (1983) showed that the bias of the coefficient estimators for any generalized model

can be computed as $(X'WX)^{-1}X'W\aleph$, where W is the diagonal matrix of w_i. However, we propose replacing w_i by w_i^* since the behavior, and therefore the bias, for the RiskLogitboost is computed as $(X'W^*X)^{-1}X'W^*\aleph$.

The factor $\aleph$ equals $Q_{ii}\left(\pi^D(X_i) - 0.5\right)$, where Q_{ii} is the diagonal elements of the Fisher information matrix denoted as Q. The matrix Q measures the amount of information that matrix X carries about the parameters, in other words, it is the variance of the gradient of the log-likelihood function with respect to the parameter vector known as the score.

Q_{rk} is the Fisher information matrix for two arbitrary generic parameters: β_k and β_r.

$$Q_{rk} = -E\left(\frac{\partial^2 \ln \ell\left(\beta_0, \cdots, \beta_k, \cdots \beta_r, \cdots, \beta_p; X_i\right)}{\partial \beta_r \beta_k}\right). \tag{6.7}$$

Now let's take the partial derivative of $\ell\left(\beta_0, \cdots, \beta_p; X_i\right)$ in (6.5) with respect to β_k

$$\frac{\partial \ell}{\partial \beta_k} = \sum_{i=1}^{n} Y_i \frac{\partial \ell}{\partial \beta_k}(X\tilde{\beta}) - \frac{\partial \ell}{\partial \beta_k} \log\left(1 + e^{X\tilde{\beta}}\right), \tag{6.8}$$

where

$$\frac{\partial \ell}{\partial \beta_k}(X_i\tilde{\beta}) = X_{ik}, \tag{6.9}$$

and

$$\frac{\partial \ell}{\partial \beta_k} \log\left(1 + e^{X_i\tilde{\beta}}\right) = \frac{e^{X\tilde{\beta}}}{1 + e^{X\tilde{\beta}}} \frac{\partial \ell}{\partial \beta_k}(X_i\tilde{\beta}) = \pi_i X_{ik}. \tag{6.10}$$

Considering (6.9) and (6.10), we obtain:

$$\frac{\partial \ell}{\partial \beta_k} = \sum_{i=1}^{n} Y_i X_{ik} - \pi_i X_{ik}. \tag{6.11}$$

Now, let's compute the second derivative of (6.8) with respect to β_r.

$$\frac{\partial^2 \ell}{\partial \beta_k \beta_r} = \frac{\partial}{\partial \beta_r} \frac{\partial \ell}{\partial \beta_k} = \sum_{i=1}^{n} X_{ik}\left(Y_i - \frac{\partial}{\partial \beta_r}(\pi_i)\right). \tag{6.12}$$

And,

$$\frac{\partial}{\partial \beta_r}(\pi_i) = \frac{e^{X_i\tilde{\beta}}\frac{\partial}{\partial \beta_r}(X_i\beta)(1+e^{X_i\tilde{\beta}}) - e^{X_i\tilde{\beta}}e^{X_i\tilde{\beta}}\frac{\partial}{\partial \beta_r}(X_i\tilde{\beta})}{(1+e^{X_i\tilde{\beta}})^2} = \pi_i X_{ir}(1-\pi_i). \tag{6.13}$$

Plugging (6.13) into (6.12):

$$\frac{\partial^2 \ell}{\partial \beta_k \beta_r} = -\sum_{i=1}^{n} X_{ik}X_{ir}\pi_i(1-\pi_i). \tag{6.14}$$

Recall that $Var(Y_i) = \pi_i(1-\pi_i)$ since Y_i follows a Bernoulli distribution and coincides with vector w_i (second stage of Algorithm 5). However, the new RiskLogitboost replaces w_i with w_i^* again in equation (6.14). If we generalize expression (6.14) for all P parameters, we obtain:

$$\frac{\partial^P \ell}{\partial \beta_1 \cdots \beta_P} = X'W^*X, \tag{6.15}$$

where W^* is the diagonal matrix of w_i^*. Equation (6.15) is also the variance-covariance matrix. Thus, Q is expressed as an $n_{times}n$ symmetric matrix:

$$Q = X(X'W^*X)^{-1}X'. \tag{6.16}$$

Finally, each transformed parameter is computed as:

$$\beta_{\text{RiskLogitboost}} = \beta^d - (X'WX)^{-1}X'W\aleph \tag{6.17}$$

6.3.3 RiskLogitboost Regression

The RiskLogitboost regression (Algorithm 6) modifies the original version of Logitboost to improve the classification of the rare events (ones). This algorithm comprises 11 stages. The first states the initial values of the prediction $\hat{Y}_i$ and probability $\pi(X_i)$.

The second obtains the transformed answer as explained in Algorithm 5. In the third stage we compute $\hat{Y}_i^d = \frac{1}{2}\log\left(\frac{\pi(X_i)^{d-1}}{1-\pi(X_i)^{d-1}}\right)$, which is the value that minimizes a negative binomial log-likelihood loss function: $\log\left(1+e^{-2Y_i\hat{Y}_i}\right)$ used for two-class classification and regression problems. However, $\hat{Y}_i$ also minimize the exponential loss function $e^{-Y_i\hat{Y}_i}$ used in Logitboost (Friedman et al., 2000). Therefore, the exponential loss function approximates the log-likelihood denoted as transformed

answer z_i, as explained in Algorithm 5. The fourth stage computes the weights that were explained in detail in Section 6.3.1. The fifth stage normalizes the weights of the previous stage so as to convert them into a distribution that must add up to 1.

The fifth stage consists of fitting a weighted linear regression to z_i^d and obtaining the $P+1$ parameters β. As proposed in the original Logitboost, the sixth stage updates the final prediction $\hat{Y}_i^d$ to fit the model by maximum likelihood using Newton steps as follows: we update the prediction $\hat{Y}_i + F\left(X_{ip}; u^d\right)$, where u corresponds to parameters β. The outcome of $F\left(X_{ip}; u^d\right)$ would be $X_i\tilde{\beta}$ in a logistic regression with π_i expressed in (6.1), which is $e^{2F(X_{ip};u^d)}$, as follows:

$$\pi = \frac{e^{2F(X_{ip};u^d)}}{1+e^{2F(X_{ip};u^d)}} = \frac{e^{2X_i\tilde{\beta}}}{1+e^{2X_i\tilde{\beta}}}. \tag{6.18}$$

Recalling $\ell\left(\beta_0, \cdots, \beta_p; X_i\right)$ from (6.5), we compute the expected log-likelihood of $\hat{Y}_i + F\left(X_{ip}; u^d\right)$:

$$E\left[\ell\left(\hat{Y}_i + F\left(X_{ip}; u^d\right)\right)\right] = \sum_{i=1}^{n} 2Y_i\left(\hat{Y}_i + F\left(X_{ip}; u^d\right)\right) - \log\left(1+2e^{\hat{Y}_i+F(X_{ip};u^d)}\right). \tag{6.19}$$

The Newton method for minimizing a strictly convex function requires the first and second derivatives. Let $\bar{g}$ be the first derivative and $\bar{H}$ be the second derivative, also known as the Hessian matrix.

$$\bar{g} = \frac{\partial E\left[\ell\left(\hat{Y}_i + F\left(X_{ip}; u^d\right)\right)\right]}{\partial F\left(X_{ip}; u^d\right)},$$
$$\bar{g} = 2E\left[Y_i - \pi_i\right]. \tag{6.20}$$
$$\bar{H} = \frac{\partial^2 E\left[\ell\left(\hat{Y}_i + F\left(X_{ip}; u^d\right)\right)\right]}{\partial F\left(X_{ip}; u^d\right)^2},$$
$$\bar{H} = -4E\left[\pi_i(1-\pi_i)\right]. \tag{6.21}$$

Hence,

$$\hat{Y}_i = \hat{Y}_i - \bar{H}^{-1}\bar{g},$$
$$\hat{Y}_i = \hat{Y}_i + \frac{1}{2}E\left(\frac{Y_i - \pi_i}{\pi_i(1-\pi_i)}\right). \tag{6.22}$$

This result is a very close approximation of the iteratively reweighted least squares method (Appendix A, equation (6.24)) to the likelihood shown in (6.5). The key difference is the factor $1/2$ that multiplies the expected value. The seventh stage

consists of checking that probabilities are bounded between 0 and 1, since adding a δ might lead to a number larger than 1.

The eighth stage consists of inverting $\frac{1}{2}\log\left(\frac{\pi(X_i)^{d-1}}{1-\pi(X_i)^{d-1}}\right)$ (explained in third stage), which yields the probability estimates. Once the iterative process is finished, we obtain the coefficient estimates of iteration D in stage nine through the expression suggested by (Liska et al., 2019; De Menezes et al., 2017). And last but not least, we obtain β^* by subtracting β^D- bias.

Algorithm 6 RiskLogitboost regression

Initial values: $\hat{Y}_i^0 = 0$,
$\pi^0(X_i) = 0.5$, where $\pi(X - I)$ are the probability estimates.
For $d = 1$ to D do:

2.1 Transformation: $z_i^d = \frac{Y_i^{d-1} - \pi(X_i)^{d-1}}{\pi(X_i)^{d-1}\left(1-\pi(X_i)^{d-1}\right)} + \delta$, where $\delta = 0.0001$.

2.2 Population Minimizer: $\hat{Y}_i^d = \frac{1}{2}\log\left(\frac{\pi(X_i)^{d-1}}{1-\pi(X_i)^{d-1}}\right)$.

2.3 Weighting:

$$w_i^* = \begin{cases} \left[\pi(X_i)\left(1-\pi(X_i)\right)\right]\left(1+|Y_i-\pi(X_i)|\right), & \text{if} \quad |Y_i-\pi(X_i)| > \bar{Y}, \\ \left[\pi(X_i)\left(1-\pi(X_i)\right)\right]\left(1-|Y_i-\pi(X_i)|\right), & \text{if} \quad |Y_i-\pi(X_i)| \leq \bar{Y}. \end{cases}$$

2.4 Normalizing: $w_i^d = \frac{w_i^{*d}}{\sum_{i=1}^n w_i^{*d}}$.

2.5 Minimizing: $\beta^d = argmin_\beta \sum_{i=1}^n w_i^d \left[z_i^d - \left(\beta_0 + \sum_{p=1}^P \beta_p\right)\right]^2$

2.6 Updating prediction: $\hat{Y}_i^d = \hat{Y}_i^{d-1} + \frac{1}{2}F\left(X_{ip}; u^d\right)$.

2.7 Checking probabilities: $\pi(X_i)^d = \min\left\{\frac{1}{1+e^{-2\hat{Y}_i^{d-1}}} + \delta, 1\right\}$.

End for Converting:

$$\pi^d(Y_i = 1|X) = \frac{1}{1+e^{-2\hat{Y}_i^{d-1}}},$$

$$\pi^d(Y_i = 0|X) = \frac{1}{1+e^{2\hat{Y}_i^{d-1}}}.$$

Obtaining the P Parameters: $\beta_p^D = \frac{\sum_{i=1}^n (X_p^D z_i^D)}{\sum_{i=1}^n (X_p^D)^2}$, $\forall p = 1, \cdots, P$.
Correcting Bias: $\beta^* = \beta^D - (X_i' w_i X_i)^{-1} X_i' w_i \aleph_i$.

6.4 Illustrative data

The illustrative data set used for testing classical and alternative machine learning algorithms is a French third-part liability motor insurance data set available from (Charpentier, 2014) through his publicly available data sets in the library CASdatasets in R. It contains 413,169 observations that were recorded mostly in one year about risk factors of third-party liability motor policies.

This data set contains the following information about vehicle characteristics: The power of the car ordered by category (Power); the car brand divided into seven categories (Brand); the fuel type, either diesel or regular (Gas). This data set also includes information about the policy holder's characteristics such as: the policy region in France based on the 1970–2015 classification (Region); the number of inhabitants per km2 in the city in which the driver resides (Density). Finally, more information about policy holders' characteristics: the car age measured in years (Car age); and the driver's age (Driver Age). And finally, the occurrence of accident claims Y_i is coded as 1 if the policy holder had suffered at least one accident, and otherwise, coded as 0. A total of 3.75% of the policy holders had reported at least one accident.

6.5 Discussion of results

This section firstly presents the predictive performance of some machine learning algorithms jointly with the RiskLogitboost regression when $Y = 1$ in the extreme observations; and secondly shows the interpretation of the model through the coefficient estimates.

6.5.1 Predictive performance of extremes

Table 6.1 shows the Root Mean Square Error (RMSE) for observations when $Y = 1$ and $Y = 0$. Even though the Boosting Tree has optimized hyperparameters, it produced a larger error than all other methods when $Y = 1$[5]. This means that the riskiest observations (with misclassifications costs) are poorly detected, and observations whose probability is not high enough are more likely to be misclassified.

[5]The Boosting Tree is built with 10-fold cross validation and has optimized hyperparameters through grid search which correspond to the number of trees (50), the maximum depth of variable interactions (1), the minimum number of observations in the terminal nodes of the trees (10), and shrinkage (0.1). And the Lasso and Ridge Logistic models had the lowest deviance among several trials with shrinkage values.

The RiskLogitboost regression had the lowest error for observations whose estimated probability was in the lower extremes. This is an important result since the proportion of cases for this set of observations usually tends to be underestimated by traditional predictive modeling techniques. Moreover, the RiskLogitboost regression perfectly predicted observations whose estimated probability was in the highest extremes, suggesting that observations that are more likely to belong to the rare event ($Y = 1$) will never be misclassified. From a risk analysis perspective, this is a valuable achievement since it reduces misclassification costs for this group.

Observations classified with SMOTEBoost and RUSBoost outperform Logitboost, Ridge Logistic, Lasso Logistic, and Boosting Tree, however, their predictive performance is still below the RiskLogitboost regression. Even though the SMOTEBoost and RUBoost are designed to handle imbalance data sets, RiskLogitboost seems to be more efficient detecting rare events .

In contrast, when $Y = 0$ the Boosting Tree, Ridge Logistic regression and Lasso Logistic had a lower RMSE than the RiskLogitboost regression. These three methods classify the non-events ($Y = 0$) accurately whereas the RiskLogitboost regression tends to underestimate their occurrence.

The results when $Y = 1$ also showed that Logitboost was superior, in predictive capacity terms, to the Ridge Logistic regression, Lasso Logistic regression and Boosting Tree in the testing data set. In this particular case, the Ridge Logistic regression and Lasso Logistic performed similarly in the training data set.

Figure 6.2 shows the highest and lowest prediction scores for all observed response Y. The RiskLogitboost regression started with higher levels of RMSE in the first iterations, after which they decreased until becoming stable. The RMSE did not vary from the fortieth iteration onwards. As a result, we were able to maintain the convergence process since the proposed transformation for the weighting procedure (Section 6.3.1) achieved identical stability to that of the original Logitboost.

Training Data Set (RMSE Y=1)												
	Lower Extreme						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
RiskLogitboost regression	0.2454	0.1825	0.1496	0.1132	0.0927	0.0803	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Ridge Logistic	0.9629	0.9629	0.9629	0.9629	0.9628	0.9628	0.9627	0.9627	0.9627	0.9627	0.9627	0.9627
Lasso Logistic	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628
Boosting Tree	0.9787	0.9747	0.9727	0.97	0.9679	0.9665	0.9162	0.9293	0.9417	0.9495	0.9522	0.9539
Logitboost	0.9829	0.9799	0.9781	0.9736	0.9707	0.9688	0.9416	0.9479	0.9505	0.9530	0.9545	0.9557
SMOTEBoost	0.6963	0.6901	0.6852	0.6800	0.6761	0.6725	0.6046	0.6090	0.6117	0.6178	0.6222	0.6264
RUSBoost	0.5811	0.5742	0.562	0.5517	0.5447	0.5391	0.4466	0.4727	0.4853	0.4931	0.4970	0.5001
Testing Data Set (RMSE Y=1)												
	Lower Extreme						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
RiskLogitboost regression	0.4690	0.3725	0.3133	0.2421	0.1991	0.1724	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Ridge Logistic	0.9629	0.9629	0.9629	0.9629	0.9628	0.9628	0.9627	0.9627	0.9627	0.9627	0.9627	0.9627
Lasso Logistic	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628	0.9628
Boosting Tree	0.9788	0.975	0.9731	0.9705	0.9683	0.9669	0.9156	0.9297	0.9424	0.9498	0.9525	0.9542
Logitboost	0.8745	0.8723	0.871	0.8688	0.8674	0.8665	0.8558	0.8577	0.8586	0.8595	0.8601	0.8606
SMOTEBoost	0.6959	0.6901	0.6854	0.6801	0.6762	0.6727	0.6042	0.6088	0.6116	0.6180	0.6226	0.6270
RUSBoost	0.5781	0.5600	0.5515	0.5425	0.5358	0.5312	0.4434	0.4539	0.4727	0.4858	0.4913	0.4948

Table 6.1: Root Mean Square Error (RMSE) for observations with $Y = 1$ and $Y = 0$.

Table 6.1 (Continued)

Training Data Set (RMSE Y=0)

	Lower Extreme						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
RiskLogitboost regression	0.7510	0.8220	0.8610	0.9060	0.9350	0.9510	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Ridge Logistic	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370
Lasso Logistic	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370
Boosting Tree	0.0200	0.0220	0.0250	0.0270	0.0290	0.0310	0.0770	0.0580	0.0510	0.0470	0.0450	0.0440
Logitboost	0.0160	0.0190	0.0200	0.0230	0.0260	0.0290	0.0570	0.0510	0.0490	0.0460	0.0450	0.0430
SMOTEBoost	0.2978	0.3070	0.3116	0.3171	0.3206	0.3240	0.3958	0.3909	0.3865	0.3797	0.3752	0.3704
RUBSbost	0.4219	0.4403	0.4488	0.4579	0.4646	0.4692	0.5566	0.5463	0.5281	0.5149	0.5094	0.5058

Testing Data Set (RMSE Y=0)

	Lower Extreme						**Upper Extreme**					
	0.01	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**	**0.01**	**0.05**	**0.1**	**0.2**	**0.3**	**0.4**
RiskLogitboost regression	0.5450	0.6490	0.7130	0.7990	0.8600	0.8960	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Ridge Logistic	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370
Lasso Logistic	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370	0.0370
Boosting Tree	0.0200	0.0220	0.0250	0.02700	0.0290	0.031	0.0770	0.0580	0.0510	0.0470	0.0450	0.0440
Logitboost	0.1250	0.1270	0.1280	0.1300	0.1310	0.1320	0.1440	0.1420	0.1410	0.1400	0.1400	0.1390
SMOTEBoost	0.2976	0.3069	0.3116	0.3171	0.3206	0.3240	0.3959	0.3909	0.3865	0.3790	0.3750	0.3705
RUBSbost	0.4189	0.4259	0.4383	0.4487	0.4558	0.4614	0.5534	0.5280	0.5154	0.5074	0.5034	0.5003

The results are presented for observations that correspond to policy holders who suffered an accident (Y=1) and those who did not (Y=0). All results were analyzed by groups of scores. So, each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the lowest accumulated prediction scores is shown on the left-hand side of the table under "Lower Extreme", and each RMSE for 1%, 5%, 10%, 20%, 30% and 40% of the highest accumulated prediction scores is shown on the right-hand side of the table under "Upper Extreme".

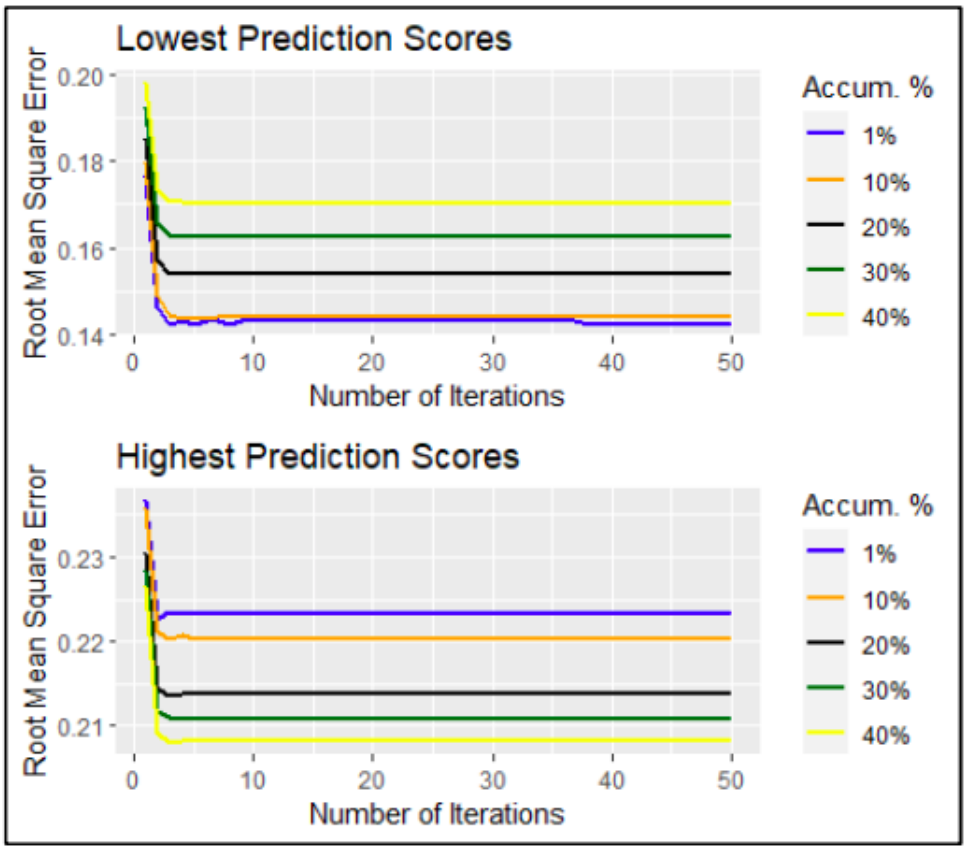

Figure 6.2: The highest and lowest prediction scores for all observed response Y within 50 iterations ($D = 50$) obtained with the RiskLogitboost regression.

6.5.2 Interpretable RiskLogitboost regression

Table 6.2 presents the coefficient estimates, standard errors and confidence intervals obtained by the RiskLogitboost regression. Due to the design and the way of fitting the RiskLogitboost regression similar to generalized linear models (i.e logistic regression) as fully explained in Section 6.3, we may obtain either the odds ratio by exponentiating the estimated coefficient estimates.

The results provided by the RiskLogitboost regression suggest that the likelihood of a policy holder having an accident increased if they had e, k, l, m, n, o type Power vehicle; in particular, drivers with o – type Power were the most likely to have an accident among all types of Power.

The policy holder was more likely to have an accident if they drove in the Regions of Haute-Normandie and Limousin), whereas driving in the Regions of Bretagne,

Centre, Haute Normandie, Ile de France, Pays de la Loire, Basse Normandie, Nord Pas de Calais and Poitou Charentes did not influence the likelihood of a person having an accident.

Policy holders driving Renault, Nissan or Citroen cars were less likely to have an accident than those driving the other brands of car.

As expected, the Lasso Logistic regression shrunk to zero all coefficients except the one corresponding to the intercept; in this sense, this method is not informative and is actually disadvantageous for analysing the effects. The Ridge Logistic Regression provided a very small magnitude of the coefficient estimates, and overall, the covariates in the Ridge Logistic regression seemed to have a small effect on the final prediction, which makes sense because 96.25% of the cases had not reported an accident. However, this model risks underestimating the probability of having an accident.

All in all, the coefficients obtained by the RiskLogitboost regression are much bigger than those obtained by the other regressions since this type of algorithm tends to overestimate the probability of occurrence of the target variable to avoid classifying risky observations as $\hat{Y}_i = 1$ instead of $\hat{Y}_i = 0$.

Table 6.3 shows the variable importance of the six most relevant covariates according to RiskLogitboost, Boosting Tree, Ridge Logistic and Logitboost regressions. The results show no consensus between the methods; however, the Boosting Tree and Ridge Logistic regression have certain categories of Brand and Region as the most important covariates, while certain categories of Power and Region seem to be the most relevant according to Logitboost and RiskLogitboost.

As a consequence, it seems that there is no consensus in the results provided by the variable importance technique, which is risky in terms of interpretation. Analysts should consider that the results of a Boosting Tree, Ridge Logistic or Lasso Logistic regression can generate misleading inferences because they underestimate the occurrence of rare events; the covariates that appear to be most contributive will be those with more effect on non-events ($Y = 0$). By contrast, the variable importance technique suggests that RiskLogitboost better identifies the covariates that are the most influential in the occurrence of rare events ($Y = 1$).

Variables	Categories	RiskLogitboost regression	RiskLogitboost regression (Standard Error)	RiskLogitboost regression (Confidence Intervals)
Power	*Intercept	20.8740	7.4130	[6.3445 ; 35.4035]
	e	-0.6527	3.5641	[-7.6383 ; 6.3329]
	f	-1.3379	3.4769	[-8.1526 ; 5.4768]
	g	-0.8003	3.4506	[-7.5635 ; 5.9629]
	h	4.9061	4.9344	[-4.7653 ; 14.5780]
	i	7.8770	5.4611	[-2.8268 ; 18.5808]
	j	8.0675	5.5682	[-2.8462 ; 18.9812]
	* *k*	18.188	7.1178	[4.2371 ; 32.1389]
	* *l*	45.3320	1.0540	[43.2662 ; 47.3978]
	* *m*	99.6840	1.5136	[96.7173 ; 102.6507]
	* *n*	144.1900	1.7590	[140.7424 ; 147.6376]
	* *o*	145.8000	17.6033	[111.2975 ; 180.3025]
Brand	Japanese (except Nissan) or Korean	-7.6774	5.7732	[-18.9929 ; 3.6381]
	Mercedes, Chrysler or BMW	-2.0130	6.7667	[-15.2757 ; 11.2497]
	Opel, General Motors or Ford	-6.5298	5.7170	[-17.7351 ; 4.6755]
	other	8.2048	7.9329	[-7.3437 ; 23.7533]
	* Renault, Nissan or Citroen	-10.3760	4.9954	[-20.1669 ; -0.5850]
	Volkswagen, Audi, Skoda or Seat	-5.5055	5.8621	[-16.9952 ; 5.9842]

Table 6.2: Coefficient Estimates, Standard Error and Confidence Intervals provided by the RiskLogitboost regression.

Table 6.2 (Continued)

Variables	Categories	RiskLogitboost regression	RiskLogitboost regression (Standard Error)	RiskLogitboost regression (Confidence Intervals)
Region	Basse-Normandie	10.279	7.1850	[-3.8036;24.3616]
	Bretagne	-3.4953	4.9434	[-13.1844 ; 6.1938]
	Centre	-6.5749	4.2924	[-14.9880 ; 1.8382]
	* Haute-Normandie	27.606	9.3055	[9.3672 ; 45.8448]
	Ile-de-France	-4.1033	5.12264	[-14.1437 ; 5.9371]
	* Limousin	34.552	10.0028	[14.9465 ; 54.1575]
	Nord-Pas-de-Calais	0.0897	5.7443	[-11.1691 ; 11.3485]
	Pays-de-la-Loire	-2.731	5.0910	[-12.7094 ; 7.2474]
	Poitou-Charentes	2.4523	5.9926	[-9.2932 ; 14.1978]
Density		0.0003	0.00025	[-0.0003 ; 0.0009]
Gas Regular		0.0187	2.1895	[-4.2727 ; 4.3101]
Car Age		0.1053	0.1969	[-0.2806 ; 0.4912]
Driver Age		0.0217	0.0712	[-0.1179 ; 0.1613]

*The base category is other for the covariates Power, Brand and Region, and diesel for the covariate Gas. * Indicates that the coefficient is significant at the 95% confidence level. The standard error (se) root square of the diagonal of the variance-covariance matrix was computed as* $(X_i' w_i^D X_i)^{-1}$. *We built a 95% confidence interval for* β *as* $[\beta - 1.96\,se, \beta + 1.96\,se]$.

Order	RiskLogitboost	Boosting Tree	Ridge Logistic	Logitboost
First	Power o	Driver Age	Brand Japanese (except Nissan) or Korean	Region Limousin
Second	Power n	Brand Japanese (except Nissan) or Korean	Region Haute-Normandie	Power m
Third	Power m	Car Age	Region Brand Opel, General Motors or Ford	Power l
Fourth	Power l	Density	Brand Volkswagen, Audi, Skoda or Seat	Power n
Fifth	Region Limousin	Brand Opel, General Motors or Ford	Region Nord-Pas-de-Calais	Region Haute-Normandie
Sixth	Region Haute-Normandie	Region Haute-Normandie	Brand Mercedes, Chrysler or BMW	Power k

The Lasso Logistic regression has no significant coefficient estimates with which to compute the variable importance technique.

Table 6.3: Variable importance of the six most relevant covariates according to RiskLogitboost, Boosting Tree, Ridge Logistic regression and Logitboost.

Figure 6.3 shows the partial dependence plot (PDP) obtained from a Boosting Tree. Each plot shows an average model prediction for each value of the covariate of interest. The intuitive interpretation of this plot is that the magnitude on the y axis shows more or less likelihood of the occurrence of the event ($Y = 1$). In this particular case, drivers with m-type Power were more likely to have an accident than drivers with d-type Power. Newer vehicles were less likely to be involved in an accident than older ones. Drivers aged between approximately 30 and 80 were less likely to have an accident than very old or very young drivers. Moreover, policy holders who drove in the region of Limousin were the least likely to have an accident in comparison with other regions of France. And last but not least, it seems that Japanese (except Nissan) or Korean vehicles were more likely to be involved in an accident than the other brands.

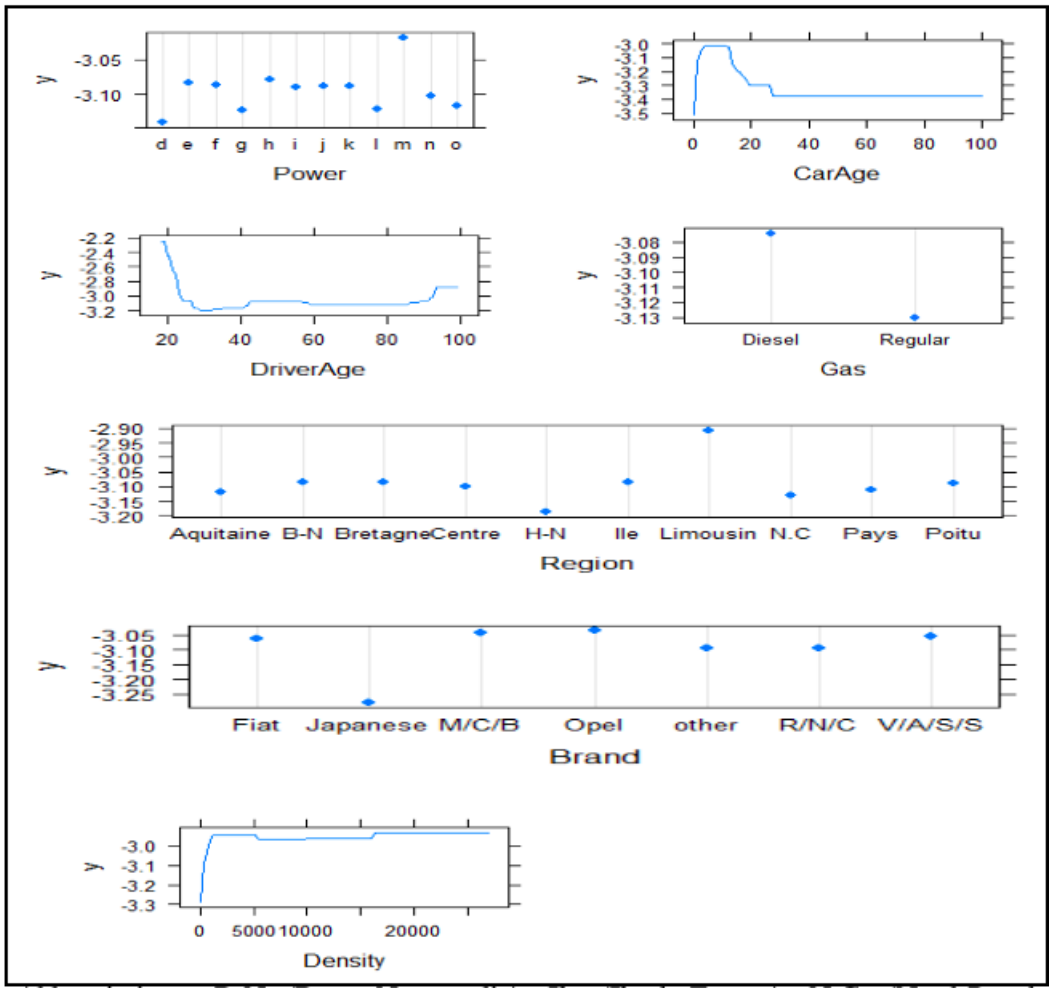

Figure 6.3: Partial dependence plots from the Boosting Tree. Abbreviations: B-N (Basse-Normandie), Ile (Ile-de-France), N.C. (Nord-Pas-de-Calais), Pays (Pays-de-la-Loire), Poitu (Poitou-Charentes), Japanese [Japanese (except Nissan) or Korean], M/C/B (Mercedes, Chrysler or BMW), V/A/S/S (Volkswagen, Audi, Skoda or Seat), Opel (Opel, General Motors or Ford)

6.6 Conclusions

On balance, RiskLogitboost brings a key advantage in the prediction of rare events, principally when the detection of the minority class is fundamental or extremely important in the case study, and the impact of false negatives is irrelevant or barely important. The treatment and the interpretation of the rare events is more accurate when using the RiskLogitboost, and it may contribute to the prevention of events whose occurrence would be disastrous, and whose cost, policy holders are not willing to accept or able to afford .

The RiskLogitboost regression is a boosting-based machine learning algorithm shown to improve the prediction of rare events compared to certain well-known tree-based and boosting-based algorithms. It will be of most value where the failure to predict the occurrence of the rare event when it will occur is high. RiskLogitboost regression implements a weighting mechanism and a bias correction that lower prediction error to better predict such rare events by overestimating their probabilities. The results presented here show that the lowest RMSE is presented in the upper and lower extremes when $Y = 1$. This comes at a cost. The RiskLogitboost regression RMSE tended to increase when $Y = 0$ in the extreme observations due to the fact that the algorithm adjusts misclassified observations, which, in the context of rare events with a binary response are coded as $Y = 1$. This cost is low, when the cost of false negatives is low much smaller than the cost of false positives.

While regularization procedures can be incorporated in econometric methods such as logistic regression, they have two main drawbacks. First, the resulting models may not be adequately interpretable because the shrinkage from such procedures depends on the penalty term, causing loss of the real effect of the covariates on the final prediction. Second, such procedures cannot classify rare events efficiently.

The Tree Boosting regression had the lowest RMSE in the majority class observations ($Y = 0$) but showed poor performance in the minority class observations. It is also more in the nature of a black box in terms of interpretability, requiring more reliance on the variable importance method and PDP. The PDP from the Tree Boosting regression is relatively informative, but all covariates are treated as significant or relevant for the final prediction, which is sometimes inconsistent with an econometric model like a regression. Moreover, while a PDP is easy to implement when there are only a few variables, with more variables' interpretation is more difficult. It is often desirable to achieve both high predictive performance for rare events and interpretability. Tree-based and boosting-based methods may be unsuitable in such situations because they underestimate the probability that the rare event will occur while also underestimating the effect of the covariates that are most important to predicting the rare event rather than the majority class. RiskLogitboost delivers high predictive performance while also facilitating interpretation by identifying the covariates most important to prediction of the rare event .

The RiskLogitboost has still limitations when decreasing the false negative rate since it focuses on reducing efficiently the error of observations $Y_i = 1$. However, for those case studies whose cost of false negative rate tends to be high, the pro-

posed method would be redesigned so as to improve the detection of observations $Y_i = 0$. And this would be the proposal for further research.

Appendix A: Computation of z_i as transformed response

A Taylor transformation is applied in (6.2) so that η_i is expanded around π_i. Let η_i be expressed as $\Gamma(Y_i)$.

$$\begin{aligned}\Gamma(Y_i) \cong & \Gamma\left(\pi(X_i)\right) + \left(Y_i - \pi(X_i)\right)\Gamma'(\pi(X_i)) \\ \cong & \log\left(\frac{\pi_i}{1-\pi_i}\right) + \left(Y_i - \pi(X_i)\right)\left(-\frac{1}{(\pi(X_i)-1)\pi(X_i)}\right) \\ \cong & \eta_i + \frac{Y_i - \pi(X_i)}{\left(1-\pi(X_i)\right)\pi(X_i)}.\end{aligned} \tag{6.23}$$

We denote $\Gamma(Y_i)$ as the transformed response z_i shown in Algorithm 5.

$$z_i \cong \eta_i + \frac{Y_i - \pi(X_i)}{\left(1-\pi(X_i)\right)\pi(X_i)}. \tag{6.24}$$

Appendix B: Computation of weights

The weights of the Logitboost are obtained by computing the variance of the transformed response $Var[z_i|X]$.

$$\begin{aligned}Var[z_i|X] = & Var\left[\Gamma(\pi(X_i))|\,X\right] + Var\left[\left(Y_i - \pi(X_i)\right)\Gamma(\pi(X_i))|\,X\right] \\ = & 0 + \Gamma'(\pi(X_i))^2 Var\left[Y_i\right] + \pi(X_i)^2 Var\left[\Gamma(\pi(X_i))\right] \\ = & \Gamma'(\pi(X_i))^2 Var\left[Y_i\right] \\ = & \left(-\frac{1}{\pi(X_i)(\pi(X_i)-1)}\right)^2 \left[\pi(X_i)(1-\pi(X_i))\right] \\ = & \pi(X_i)(1-\pi(X_i)).\end{aligned} \tag{6.25}$$

Chapter 7: Conclusions

Through the realization of each chapter of this thesis, I have contributed to the resolution of two important empirical economic research problems: i) How econometricians can improve the comprehension of environments with the occurrence of rare phenomena and imbalanced data? And also, ii) How econometricians improve the rare events and imbalanced data prediction accuracy?. This dissertation has provided methodological proposals that enhance the robustness of econometric modelling with rare event and class imbalanced data, to do so I developed strong theoretical basis with which various algorithms were built, and tested with illustrative examples to detect changes in accuracy prediction and interpretation with respect to other algorithms or methods, and to derive more accurate conclusions from the modelling procedure.

In chapter 2, I developed a weighting mechanism which is incorporated in the likelihood estimation of a classical logistic regression model that combined with an optimal tuning parameter is able to improve substantially the predictive performance. Additionally, I proposed a decision rule to choose the optimal tuning parameter that will calibrate the weight according to the distance between the observed and predicted values. I could examine the weighted log-likelihood performance. Firstly, I could reduce the root mean square error in the lowest and highest deciles of prediction, which means that the observations which are farthest from the mean, are now more likely to be classified correctly. Secondly, I detected a strictly positive value of the norm of the difference between the vector of estimated parameters in the weighted model (new proposal) and the vector of estimated parameters in the unweighted model (classical), which means that the new proposal did only improve the predictive capacity, but also changed the magnitude of the coefficient estimates so that one can interpret the model in a more realist way.

In chapter 3, I proposed a methodological strategy for improving the predictive performance of rare events prediction when they come from complex survey designs, in other words, when data are not independent. I developed a weighting

mechanism that borrows the core idea of Chapter 2, but it adapted specific correctors that incorporate complex designed data and lead to lower root mean square errors for event observations. Additionally, I proposed a C-ROC criterion to evaluate the best model tuned by the weighting mechanism considering the sensitivity as a priority. Finally, some other alternatives for weighting are mentioned for further research.

In chapter 4, I proposed a deep exploration of classical and modern boosting-based methods in order to gain statistical intuition of why some methods or algorithms overcome others, and under which circumstances. I theoretically and empirically concluded that many advanced boosting-based and tree-based algorithms are formulated through deterministic strategies that excessively or even optimally learn from data, so that their predictive capacity is very strong in the training sample, but weak in the testing sample. Regularization techniques are usually proposed to solve the so called "overfitting problem", nevertheless, the shrinkage of the coefficient estimates cause the lose of control over the real interpretation of effects. I proposed a strategy to correct the overfitting problem of a XGBoost model, making it the least complex as possible. However, its predictive performance almost equaled a Logistic model after the correction. Furthermore, many sophisticated advanced analytics algorithms classified efficiently mean observations but poorly the extreme or imbalanced data points. Finally, I concluded that new risk analytics algorithms are highly demanded, so that they can be used as econometric methods, and efficient rare events classifiers.

In chapter 5, I developed a boosting-based algorithm called "Synthetic Penalized Logitboost". It addresses the methodological problems for binary imbalanced data mentioned in the previous chapter. I borrowed the mortgage lending model specification put forward by (Munnell et al., 1996) to provide a real-life application of identification whether an applicant will default in future or be turned down under the Home Mortgage Disclosure Act (HMDA). The Synthetic Penalized Logitboost allows the interpretability of the effects so that the conclusions obtained might contribute to the study of financial inclusion policy. Additionally, the proposed algorithm proved to improve the prediction performance and reduce the risk of overfitting. Last but not least, the proposed Synthetic Penalized Logitboost was tested in real publicly HDMA data sets of 1997, 2012 and 2017, concluding the same prediction performance for all of them.

In chapter 6, the last one of this dissertation, I proposed the "RiskLogitboost regression" which is a boosting-based algorithm. It succeeded to approximate an

econometric model that allows interpretability and reduce the prediction error of the rare class, where the degree of imbalance is more extreme. The RiskLogitboost in a quite adapted version of the original Logitboost, it has a weighting mechanism purposefully designed to overweight or underweight data points depending on the closeness of the real values with their predicted probabilities. I potentiated the prediction error reduction through a bias correction strategy used for generalized linear models. The RiskLogitboost was tested in a third party motor insurance data set and compared with some other optimized tree-based methods and boosting-based algorithms. Results showed the smallest errors for event observations, and lower errors for non-event observations compared to what various methods obtained for minority class predictions. The RiskLogitboost improved the comprehension of the rare phenomena usually underestimated by most modern advanced analytics algorithms.

To conclude, the methodological contribution of this dissertation aimed to improve the comprehension of rare phenomena and binary imbalanced data whose applications in the real world are steadily increasing especially in actuarial economics, empirical economics and financial economics. This dissertation has focused on binary dependent variables, which lies into a more difficult problem than discrete dependent variables, where the number of zeros exceed the number of levels of the count variable. In the binary setting, one only have the option to compute a predicted probability of occurrence, decide for a suitable threshold to trigger the event from non-event. However, in a count data setting, one have a distribution to adjust, which is more informative than a binary choice. So the changes from 2 to 3 or 0 to 1 are not very abrupt as in binary case which are only from 0 to 1.

Final Remarks

Besides the contribution of each chapter, I intend to build a publicly available a CRAN package in the statistical software *R*. This package will be the compilation of all the proposed algorithms within the chapters so that users can employ them in their own cases study.

One of the most importat limitations is the availability of publicly available insurance data. The data sets shown in the previous chapters have been provided thanks to agreements with Spanish insurers under confidentiality clauses, or representative subsets of the original databases like in (Charpentier, 2014). Whereas the data set is more extensive, the analysis and the results will become broader and deeper.

More strategic alliances between universities and corporations that favor more

real and updated research, should be encouraged. This fact will impact directly on the innovation of the productive sector.

This PhD disseration has focused efforts on cross-sectional data to study deeply the treatment of imbalanced data and rare events. However, this research might be extended to panel data and time series data, where phenomena is not captured in a specific period of time.

Apart from this, this PhD thesis considers rare events and imbalanced data in the binary response of a model. However, some additional research might be extended to count data, where the occurrence of few or non-events are considered rare or extreme.

Bibliography

Agterberg, F., Bonham-Carter, G., Cheng, Q., and Wright, D. (1993). Weights of evidence modeling and weighted logistic regression for mineral potential mapping. *Computers in Geology*, 25:13–32.

Ahmed, E., Yaqoob, I., Hashem, I. A. T., Khan, I., Ahmed, A. I. A., Imran, M., and Vasilakos, A. V. (2017). The role of big data analytics in internet of things. *Computer Networks*, 129:459–471.

Alpaydin, E. (2004). *Introduction to machine learning*. MIT press, Massachussetts.

Aven, T. (2016). Risk assessment and risk management: Review of recent advances on their foundation. *European Journal of Operational Research*, 253(1):1–13.

Aven, T. (2018). An emerging new risk analysis science: Foundations and implications. *Risk Analysis*, 38(5):876–888.

Ayuso, M., Guillen, M., and Marín, A. M. P. (2016a). Using gps data to analyse the distance travelled to the first accident at fault in pay-as-you-drive insurance. *Transportation Research part C: Emerging Technologies*, 68:160–167.

Ayuso, M., Guillen, M., and Pérez-Marín, A. (2016b). Telematics and gender discrimination: some usage-based evidence on whether men's risk of accidents differs from women's. *Risks*, 4(2):10.

Ayuso, M., Guillen, M., and Pérez-Marín, A. M. (2014). Time and distance to first accident and driving patterns of young drivers with pay-as-you-drive insurance. *Accident Analysis & Prevention*, 73:125–131.

Barandela, R., Valdovinos, R. M., and Sanchez, J. S. (2003). New applications of ensembles of classifiers. *Pattern Analysis & Applications*, 6(3):245–256.

Barboza, F., Kimura, H., and Altman, E. (2017). Machine learning models and bankruptcy prediction. *Expert Systems with Applications*, 83:405–417.

Bedford, T., Cooke, R., et al. (2001). *Probabilistic risk analysis: foundations and methods*. Cambridge University Press, Cambridge.

Bethlehem, J. G. and Keller, W. J. (1987). Linear weighting of sample survey data. *Journal of Official Statistics*, 3(2):141–153.

Beyan, C. and Fisher, R. (2015). Classifying imbalanced data sets using similarity based hierarchical decomposition. *Pattern Recognition*, 48(5):1653–1672.

Boucher, J.-P., Côté, S., and Guillen, M. (2017). Exposure as duration and distance in telematics motor insurance using generalized additive models. *Risks*, 5(4):54.

Bradford, J. P., Kunz, C., Kohavi, R., Brunk, C., and Brodley, C. E. (1998). Pruning decision trees with misclassification costs. In *European Conference on Machine Learning*, pages 131–136. Springer.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). Classification and regression trees. *Cole Statistics/Probability Series*.

Buizza, R. (2008). The value of probabilistic prediction. *Atmospheric Science Letters*, 9(2):36–42.

Charpentier, A. (2014). *Computational actuarial science with R*. CRC press.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM.

Clemen, R. T. and Winkler, R. L. (1999). Combining probability distributions from experts in risk analysis. *Risk Analysis*, 19(2):187–203.

Connelly, R., Playford, C. J., Gayle, V., and Dibben, C. (2016). The role of administrative data in the big data revolution in social science research. *Social Science Research*, 59:1–12.

De Menezes, F. S., Liska, G. R., Cirillo, M. A., and Vivanco, M. J. (2017). Data classification with binary response through the boosting algorithm and logistic regression. *Expert Systems with Applications*, 69:62–73.

Deza, M. M. and Deza, E. (2009). Encyclopedia of distances. In *Encyclopedia of distances*, pages 1–583. Springer.

Dietterich, T. G., Domingos, P., Getoor, L., Muggleton, S., and Tadepalli, P. (2008). Structured machine learning: the next ten years. *Machine Learning*, 73(1):3.

Domingo, C., Watanabe, O., et al. (2000). Madaboost: A modification of Adaboost. In *In Proceedings of the Thirteenth Annual Conference on Computational Learning Theory (COLT)*, pages 180–189, Graz, Austria.

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv:1702.08608*.

Fan, W., Stolfo, S. J., Zhang, J., and Chan, P. K. (1999). Adacost: misclassification cost-sensitive boosting. In *Icml*, volume 99, pages 97–105.

Field, C. and Smith, B. (1994). Robust estimation: A weighted maximum likelihood approach. *International Statistical Review*, 62(3):405–424.

Frees, E. W., Derrig, R. A., and Meyers, G. (2014). *Predictive modeling applications in actuarial science*, volume 1. Cambridge University Press, Cambridge.

Freund, Y. (2001). An adaptive version of the boost by majority algorithm. *Machine Learning*, 43(3):293–318.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *In Machine Learning: Proceedings of the Thirteenth International Conference*, volume 96, pages 148–156, San Francisco.

Friedman, J., Hastie, T., Tibshirani, R., et al. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *Annals of Statistics*, 28(2):337–407.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232.

Gao, G., Meng, S., and Wuthrich, M. V. (2019). Claims frequency modeling using telematics car driving data. *Scandinavian Actuarial Journal*, 2019(2):143–162.

Gao, G. and Wuthrich, M. V. (2018). Feature extraction from telematics car driving heatmaps. *European Actuarial Journal*, 8(2):383–406.

Gao, G. and Wuthrich, M. V. (2019). Convolutional neural network classification of telematics car driving data. *Risks*, 7(1):6.

Gomez-Verdejo, V., Arenas-Garcia, J., Ortega-Moral, M., and Figueiras-Vidal, A. R. (2005). Designing RBF classifiers for weighted boosting. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 1057–1062. IEEE.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge, Cambridge.

Green, P. J. (1984). Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives. *Journal of the Royal Statistical Society: Series B (Methodological)*, 46(2):149–170.

Greene, W. H. (2003). *Econometric analysis*. Pearson Education India.

Guelman, L. and Guillen, M. (2014). A causal inference approach to measure price elasticity in automobile insurance. *Expert Systems with Applications*, 41(2):387–396.

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2014). A survey of personalized treatment models for pricing strategies in insurance. *Insurance: Mathematics and Economics*, 58:68–76.

Guelman, L., Guillen, M., and Perez-Marin, A. M. (2015). Uplift random forests. *Cybernetics and Systems*, 46(3-4):230–248.

Guikema, S. (2020). Artificial intelligence for natural hazards risk analysis: Potential, challenges, and research needs. *Risk Analysis*, 40(6):1117–1123.

Guillen, M. (2014). Regression with categorical dependent variables. *Predictive Modeling Applications in Actuarial Science*, 1:65–86.

Guillen, M., Nielsen, J. P., Ayuso, M., and Pérez-Marín, A. M. (2019). The use of telematics devices to improve automobile insurance rates. *Risk Analysis*, 39(3):662–672.

Guillen, M. and Pesantez-Narvaez, J. (2018). Machine learning and predictive modeling for automobile insurance pricing. *Anales del Instituto de Actuarios Españoles*, 4(24):123–147.

Guo, H. and Viktor, H. L. (2004). Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *ACM Sigkdd Explorations Newsletter*, 6(1):30–39.

Hanley, J. A. and McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36.

Hansson, S. O. and Aven, T. (2014). Is risk analysis scientific? *Risk Analysis*, 34(7):1173–1183.

Hastie, T. and Tibshirani, R. (1986). Generalized additive models. *Statistical Science*, 1(3):297–318.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Henriques, A. C. V., de Souza Meirelles, F., and da Cunha, M. A. V. C. (2020). Big data analytics: achievements, challenges, and research trends. *Independent Journal of Management & Production*, 11(4):1201–1222.

Hu, S., Liang, Y., Ma, L., and He, Y. (2009). MSMOTE: Improving classification performance when training data is imbalanced. In *2009 Second International Workshop on Computer Science and Engineering*, volume 2, pages 13–17. IEEE.

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 1(35):73–101.

Hultkrantz, L., Nilsson, J.-E., and Arvidsson, S. (2012). Voluntary internalization of speeding externalities with vehicle insurance. *Transportation Research part A: Policy and Practice*, 46(6):926–937.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer, New Yorkk.

Japkowicz, N. and Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis*, 6(5):429–449.

Jiang, C., Wang, Z., Wang, R., and Ding, Y. (2018). Loan default prediction by combining soft information extracted from descriptive text in online peer-to-peer lending. *Annals of Operations Research*, 266(1-2):511–529.

Jiang, N. and Liu, H. (2013). Understand system's relative effectiveness using adapted confusion matrix. In *International Conference of Design, User Experience, and Usability*, pages 294–302. Springer.

Joshi, M. V., Kumar, V., and Agarwal, R. C. (2001). Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 257–264. IEEE.

King, G. and Zeng, L. (2001). Logistic regression in rare events data. *Political Analysis*, 9(2):137–163.

Kotsiantis, S., Kanellopoulos, D., Pintelas, P., et al. (2006). Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science and Engineering*, 30(1):25–36.

Kotsiantis, S. B., Zaharakis, I., and Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. *Emerging Artificial Intelligence Applications in Computer Engineering*, 160(1):3–24.

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232.

Kuhn, M., Johnson, K., et al. (2013). *Applied predictive modeling*, volume 26. Springer.

Langedijk, S., Vollbracht, I., and Paruolo, P. (2019). The potential of administrative microdata for better policy-making in Europe. *Data-Driven Policy Impact Evaluation*, page 333.

Lee, S. C. and Lin, S. (2018). Delta boosting machine with application to general insurance. *North American Actuarial Journal*, 22(3):405–425.

Lin, W.-C., Tsai, C.-F., Hu, Y.-H., and Jhang, J.-S. (2017). Clustering-based undersampling in class-imbalanced data. *Information Sciences*, 409:17–26.

Liska, G. R., Cirillo, M. Â., de Menezes, F. S., and Bueno Filho, J. S. d. S. (2019). Machine learning based on extended generalized linear model applied in mixture experiments. *Communications in Statistics-Simulation and Computation*, pages 1–15.

Longadge, R., Dongre, S., and Malika, L. (2013). Class imbalance problem in data mining: Review. *International Journal of Computer Science and Network*, 2(1):83–87.

Loyola-González, O., Martínez-Trinidad, J. F., Carrasco-Ochoa, J. A., and García-Borroto, M. (2016). Study of the impact of resampling methods for contrast pattern based classifiers in imbalanced databases. *Neurocomputing*, 175:935–947.

Lumley, T. et al. (2004). Analysis of complex survey samples. *Journal of Statistical Software*, 9(1):1–19.

Maalouf, M. and Trafalis, T. B. (2011). Robust weighted kernel logistic regression in imbalanced and rare events data. *Computational Statistics & Data Analysis*, 55(1):168–183.

Manski, C. F. and Lerman, S. R. (1977). The estimation of choice probabilities from choice based samples. *Econometrica*, (8):1977–1988.

Masnadi-Shirazi, H. and Vasconcelos, N. (2009). On the design of loss functions for classification: theory, robustness to outliers, and savageboost. In *Advances in neural information processing systems*, pages 1049–1056.

Masnadi-Shirazi, H. and Vasconcelos, N. (2010). Cost-sensitive boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):294–309.

McCullagh, P. and Nelder, J. A. (1983). *Generalized linear models. Monographs on statistics and applied probability*. Chapman and Hall: London.

Mease, D., Wyner, A. J., and Buja, A. (2007). Boosted classification trees and class probability/quantile estimation. *Journal of Machine Learning Research*, 8(3):409–439.

Mohsin, M., Sardar, M. U., Hasan, O., and Anwar, Z. (2017). IoTRiskAnalyzer: A probabilistic model checking based framework for formal risk analytics of the internet of things. *IEEE*, 5:5494–5505.

Munnell, A. H., Tootell, G. M., Browne, L. E., and McEneaney, J. (1996). Mortgage lending in boston: Interpreting HMDA data. *The American Economic Review*, (1):25–53.

Nelder, J. A. and Wedderburn, R. W. (1972). Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384.

Perez-Marin, A. M. and Guillen, M. (2019). Semi-autonomous vehicles: Usage-based data evidences of what could be expected from eliminating speed limit violations. *Accident Analysis & Prevention*, 123:99–106.

Pesantez-Narvaez, J. and Guillen, M. (2020a). Penalized logistic regression to improve predictive capacity of rare events in surveys. *Journal of Intelligent & Fuzzy Systems*, (5):1–11.

Pesantez-Narvaez, J. and Guillen, M. (2020b). Weighted logistic regression to improve predictive performance in insurance. *Advances in Intelligent Systems and Computing*, (894):22–34.

Pesantez-Narvaez, J., Guillen, M., and Alcañiz, M. (2019). Predicting motor insurance claims using telematics data—XGBoost versus logistic regression. *Risks*, 7(2):70.

Pesantez-Narvaez, J., Guillen, M., and Alcañiz, M. (2021). A synthetic penalized logitboost to model mortgage lending with imbalanced data. *Computational Economics*, pages 281—-309.

Riddle, P., Segal, R., and Etzioni, O. (1994). Representation design and brute-force induction in a Boeing manufacturing domain. *Applied Artificial Intelligence an International Journal*, 8(1):125–147.

Robins, J. M., Rotnitzky, A., and Zhao, L. P. (1994). Estimation of regression coefficients when some regressors are not always observed. *Journal of the American statistical Association*, 89(427):846–866.

Roel, V., Antonio, K., and Claeskens, G. (2018). Unraveling the predictive power of telematics data in car insurance pricing. *Journal of Royal Statistics Society: Series C(Applied Statistics)*, 67:1275–304.

Schapire, R. E. and Freund, Y. (2012). *Boosting: Foundations and algorithms*. MIT press.

Schulter, S., Wohlhart, P., Leistner, C., Saffari, A., Roth, P. M., and Bischof, H. (2013). Alternating decision forests. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 508–515.

Seal, H. L. (1967). Studies in the history of probability and statistics. XV The historical development of the Gauss linear model. *Biometrika*, 54(1-2):1–24.

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., and Napolitano, A. (2009). Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 40(1):185–197.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

Sun, Y., Kamel, M. S., and Wang, Y. (2006). Boosting for learning multiple classes with imbalanced class distribution. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 592–602. IEEE.

Sun, Y., Kamel, M. S., Wong, A. K., and Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378.

Sutton, R. S., Barto, A. G., et al. (1998). *Introduction to Reinforcement Learning*, volume 135. MIT Press Cambridge.

Tibshirani, R. (2011). Regression shrinkage and selection via the lasso: a retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(3):273–282.

Tikhonov, A. N. (1963). On the solution of ill-posed problems and the method of regularization. *Soviet Mathematics*, (5):1035.

Ting, K. M. (2000). A comparative study of cost-sensitive boosting algorithms. In *In Proceedings of the 17th International Conference on Machine Learning*.

Ting, K. M. (2017). Confusion matrix. *Encyclopedia of Machine Learning and Data Mining*, 260.

Turing, A. M. (2009). Computing machinery and intelligence. In *Parsing the Turing test*, pages 23–65. Springer.

Verbeke, W., Martens, D., and Baesens, B. (2014). Social network analysis for customer churn prediction. *Applied Soft Computing*, 14:431–446.

Viola, P. and Jones, M. (2001). Fast and robust classification using asymmetric adaboost and a detector cascade. *Advances in Neural Information Processing Systems*, 14:1311–1318.

Wang, S., Chen, H., and Yao, X. (2010). Negative correlation learning for classification ensembles. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Wang, S. and Yao, X. (2009). Diversity analysis on imbalanced data sets by using ensemble models. In *2009 IEEE Symposium on Computational Intelligence and Data Mining*, pages 324–331. IEEE.

Wedderburn, R. W. (1974). Quasi-likelihood functions, generalized linear models, and the Gauss—Newton method. *Biometrika*, 61(3):439–447.

Wei, W., Li, J., Cao, L., Ou, Y., and Chen, J. (2013). Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web*, 16(4):449–475.

Weinberger, K. Q. and Saul, L. K. (2009). Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(2).

Winship, C. and Radbill, L. (1994). Sampling weights and regression analysis. *Sociological Methods & Research*, 23(2):230–257.

Wüthrich, M. V. (2017). Covariate selection from telematics car driving data. *European Actuarial Journal*, 7(1):89–108.

Zaremba, A. and Czapkiewicz, A. (2017). Digesting anomalies in emerging european markets: A comparison of factor pricing models. *Emerging Markets Review*, 31:1–15.

Zhu, L., Yu, F. R., Wang, Y., Ning, B., and Tang, T. (2018). Big data analytics in intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 20(1):383–398.