TECHNISCHE UNIVERSITÄT WIEN
Vienna | Austria

DIPLOMARBEIT

Outlier-robust Logistic Regression for Imbalanced Data

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Statistik und Wirtschaftsmathematik

unter der Anleitung von

Univ.-Prof. Dipl.-Ing. Dr.techn. Peter Filzmoser

durch

Sanja Priselac
Matrikelnummer: 01428180
ausgeführt am Institut für Stochastik und Wirtschaftsmathematik
der Fakultät für Mathematik und Geoinformation
an der Technischen Universität Wien

Wien, am 14.09.2021

(Unterschrift Verfasser) (Unterschrift Betreuer)

Kurzfassung

Die logistische Regression ist eine weithin gebräuchliche Klassifizierungsmethode, die zur Modellierung einer binären abhängigen Variable verwendet wird. Viele beispielhafte Fälle solch einer binären logistischen Regression verwenden Datensätze, die eine unausgewogene Verteilung der abhängigen Variable aufweisen und zudem oftmals sogenannte Ausreißer inkludieren – dies sind atypische Beobachtungen in den zugehörigen Daten. Sowohl die Ausreißer als auch die unausgewogene Verteilung der abhängigen Variable können allerdings die Prognosequalität des Models stark verringern. Daher verlangen derartige Datenstrukturen eine robuste Methode, die für das Problem eines unausgewogenen statistischen Lernprozesses geeignet ist.

Diese Arbeit schlägt daher eine logistische Regression für unausgewogene Datensätze basierend auf dem Bianco-Yohai-Schätzer vor, welcher als höchst robuste Methode für logistische Regressionen angesehen werden kann. Das Problem einer unausgewogenen Verteilung der abhängigen Variable wird dabei angegangen, indem kostensensitive Eigenschaften in die Zielfunktion zur Parameterbestimmung integriert werden. Daher inkludiert die Umsetzung die Adaption eines iterativen Algorithmus zur Berechnung des Bianco-Yohai Schätzers. Die Arbeit stellt zudem auch eine zusätzliche Methode zur Erkennung von sogenannten Hebelpunkten vor, welche für die gewichtete Version des Algorithmus vonnöten und dabei auch von immenser Bedeutung für die Anwendung des Bianco-Yohai-Schätzers sind, da auf diese Weise dessen Anwendbarkeit sichtlich erweitert wird.

Die erhaltenen kostensensitiven Formen des Bianco-Yohai-Schätzers, sowohl in der gewichteten als auch in der Originalfassung, werden anschließend mit den jeweiligen nicht-robusten und nicht-kostensensitiven Formen verglichen. Die Ergebnisse der Simulation sowie die Anwendungsbeispiele mit einem unausgewogenen Datensatz aus dem Bereich der Kreditwürdigkeitsprüfung zeigen folgende Charakteristiken: Für unausgewogene Datensätze erhöht die Berücksichtigung der Kosten die Qualität des Bianco-Yohai-Schätzers sowohl in den originalen als auch den gewichteten Versionen signifikant. Zudem bietet diese Methode auch eine weitaus bessere Prognosequalität im Vergleich zur logistischen Regression, wenn die Daten schlechte Hebelpunkte enthalten. Somit bietet die kosten-sensitive Form des Bianco-Yohai-Schätzers sowohl in seiner ursprünglichen als auch in seiner gewichteten Version eine statistisch zuverlässige Klassifikationsmethode für die Modellierung unausgewogener Daten mit gegebenen Ausreißern.

Abstract

Logistic regression represents a widely used classification method for modeling a binary response variable. Many exemplary cases of binary logistic regression employ data sets with an imbalanced distribution of the output variable and often include outliers – atypical observations in the data. Both outliers and an imbalanced class distribution can greatly reduce the predictive power of the classifier. Therefore, such data structures require a robust method suitable for imbalanced learning problems.

This thesis proposes a robust logistic regression for imbalanced data sets based on the Bianco-Yohai estimator, a highly robust method for logistic regression. The imbalance learning problem is addressed by including the cost-sensitive features in the objective function for parameter estimation. Thus, the implementation involves adapting the iterative algorithm for computing the Bianco-Yohai estimator. The paper also proposes an additional method for detecting leverage points required for the weighted version of the estimator, which significantly expands the data domain in which the Bianco-Yohai estimator is applicable.

The obtained cost-sensitive forms of the Bianco-Yohai estimator, in the weighted and original versions, are compared with the corresponding non-robust and non-cost-sensitive forms. The results of the simulation experiments and the use case with the imbalanced data set employed for credit scoring indicate the following. For imbalanced data sets, the inclusion of cost significantly improves the performance of the Bianco-Yohai estimator in both the original and weighted versions. Moreover, the methods provide better performance compared to logistic regression when the data contain bad leverage points. Thus, the cost-sensitive form of the Bianco-Yohai estimator, in both its original and weighted versions, provides a statistically reliable classifier for modeling imbalanced data containing outliers.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Wien, am 14.09.2021

Sanja Priselac

Contents

1. Introduction

With the rapid pace of technological change, data collection techniques have improved significantly and the utilization of data processing methods has likewise increased. Statistical learning is an example of such data processing techniques and refers to methods that use statistical inference to derive conclusions based on data at hand. In the case that the statistical learning task involves the prediction of a measurement belonging to one of several possible categories, the task can be characterized as classification. One of the firmly established and common statistical learning methods for classification is logistic regression. The advantages of logistic regression include high model interpretability and the determination of the class membership probabilities instead of a mere class prediction. Moreover, due to the adequate formal model definition, many statistical inference methods also allow for further model analysis. Therefore, logistic regression is often preferred over state-of-the-art machine learning models in a diverse range of fields.

A frequent challenge in logistic regression, but also in other binary classification methods where the output measurement only features two possible categories, arises when one class is observed more frequently than another. Typical applications include modeling credit scores in finance or predicting the risk of diseases in healthcare. Such classification tasks should be modeled using imbalance learning methods, given that the resulting classifier could otherwise entail a poor predictive performance, especially for the minority class. The two most common approaches in imbalance learning include sampling methods and cost-sensitive learning, whereby the latter approach incorporates cost features into the classification paradigms. In the context of imbalance learning, cost-sensitive methods typically introduce different costs for majority and minority class observations.

Another obstacle regarding the application of learning techniques to various data structures concerns the fact that all statistical methods require a set of assumptions, which are rarely satisfied in real-world modeling problems. These assumptions usually include requirements on data distribution. Applying a statistical learning model to data that does not meet the distributional requirements often results in unacceptably low statistical efficiency of the resulting estimates. Classical statistical methods refer to modeling techniques assuming that all observations conform to the desired distribution. Although theoretically and computationally convenient, classical statistics often do not provide a suitable tool for the statistical application in data analysis. In most cases, the majority of data points offer the assumed distributional characteristics, yet however, a minority of observations frequently follow a different pattern or no pattern at all. Such data points are called outliers. Thus, the objective of robust statistics is is the development of meth-

ods that that provide reliable statistical estimates when the data contain a fraction of outliers. In practice, almost all data sets encompass outliers of some sort.

A data set with an imbalanced distribution of an output variable and a proportion of outliers should be modeled by considering both imbalance learning methods and a robust approach. Cost-sensitive features or robust methods were included in the logistic regression model since a long time, yet many statistical modeling tools do not provide cost-sensitive modeling of the robust logistic regression. Hence, this paper focuses on the implementation of a robust logistic regression model suitable for imbalance learning problems. The method proposed within this work incorporates the observation costs into the Bianco-Yohai estimator, a robust logistic regression model. The method implementation is based on the algorithm for the Bianco-Yohai estimator proposed by Croux and Haesbroeck in [1]. The paper also proposes another outlier detection method integrated into the algorithm, which renders the robust estimator more suitable for real-world settings. Thus, it hopes to contribute to a better parameter estimation in the case of imbalanced data sets containing outliers.

This thesis is organized as follows. Chapter 2 defines the logistic regression and shows the algorithm for estimating the parameters. Next, Chapter 3 addresses cost-sensitive learning for data sets with an unbalanced distribution of the output variable. The cost-sensitive form of logistic regression is defined, followed by a proposal for model evaluation metrics for imbalanced data sets. In Chapter 4, the main concepts of robustness are explained and the Bianco-Yohai estimator for logistic regression is introduced. Chapter 5 presents the existing iterative algorithm for the Bianco-Yohai estimator, followed by the algorithm modification to account for the imbalance learning costs. Chapter 6 reports empirical evaluation, whereby in Section 6.1 the parameters of the estimator are evaluated using a simulation example, and in Section 6.2 the estimation is analyzed using credit scoring data. Finally, Chapter 7 summarizes conclusions based on the evaluation results.

2. Logistic Regression Model for Classification

Statistical learning refers to methods that *learn* from data by drawing new conclusions based on the data at hand, using statistical inference. If the objective of a statistical learning task is is the prediction of an outcome measurement by using one or more input (or explanatory) variables, then the task can be characterized as supervised learning. Moreover, based on the type of the outcome variable, the supervised learning methods can be divided into regression and classification problems. In a regression task, the outcome variable is numeric (or quantitative), whereas in a classification task, the outcome variable is categorical. Logistic regression is an example of a classification problem where the output variable falls into one of the K classes. This work focuses on the logistic regression with a binary output variable, which has only two classes. [2]

2.1. Definition of the Logistic Regression Model

Logistic regression models a binary random variable Y that takes on the value one if an observed event has happened and zero in the absence of the event. For a p-dimensional input variable $\mathcal{X}$, the probability of an event is denoted as

$$\pi := \mathbb{P}(Y = 1 \mid \mathcal{X} = \boldsymbol{x}),$$

and the outcome variable Y follows the Bernoulli distribution with the probability function

$$\mathbb{P}(Y = y \mid \pi) = \pi^y (1 - \pi)^{1-y}, \quad y \in \{0, 1\}.$$

The logit function is a transformation of the posterior probability π and it is defined as follows:

$$logit(\boldsymbol{x}) = \log\left(\frac{\pi}{1-\pi}\right) = \log\frac{\mathbb{P}(y = 1 \mid \mathcal{X} = \boldsymbol{x})}{\mathbb{P}(y = 0 \mid \mathcal{X} = \boldsymbol{x})}. \tag{2.1}$$

Logistic regression models the logit transformation with the linear function of the inputs

$$\begin{aligned} logit(\boldsymbol{x}) &= \beta_0 + \beta_{1,1}x_1 + \beta_{1,2}x_2 + \cdots + \beta_{1,p}x_p \\ &= \beta_0 + \boldsymbol{\beta}_1^T \boldsymbol{x}, \end{aligned} \tag{2.2}$$

with $\boldsymbol{x} = (x_1, \cdots, x_p)^T$, $\beta_0 \in \mathbb{R}$ and $\boldsymbol{\beta}_1 = (\beta_{1,1}, \cdots, \beta_{1,p})^T \in \mathbb{R}^p$.

The posterior probability terms arise from Equation (2.2) together with the condition $\mathbb{P}(Y = 0 \mid \mathcal{X} = \boldsymbol{x}) + \mathbb{P}(Y = 1 \mid \mathcal{X} = \boldsymbol{x}) = 1$:

$$\mathbb{P}(Y = 1 \mid \mathcal{X} = \boldsymbol{x}) = \frac{\exp(\, \beta_0 + \boldsymbol{\beta}_1^T \boldsymbol{x} \,)}{1 + \exp(\, \beta_0 + \boldsymbol{\beta}_1^T \boldsymbol{x} \,)} \tag{2.3}$$

$$\mathbb{P}(Y = 0 \mid \mathcal{X} = \boldsymbol{x}) = \frac{1}{1 + \exp(\, \beta_0 + \boldsymbol{\beta}_1^T \boldsymbol{x} \,)} \tag{2.4}$$

2.2. Fitting Logistic Regression Models

Logistic regression parameters are mainly estimated with the maximum likelihood method. Maximum likelihood is appropriate for parameter estimation of the non-normal models, such as the logistic regression with a Bernoulli distributed outcome variable. [3]

Sampled data can be expressed by a model matrix $\boldsymbol{X} \in \mathbb{R}^{n \times (p+1)}$, where each row i represents a p-dimensional multivariate observation $\boldsymbol{x}_i$ with an intercept term included, and each column $j \neq 1$ represents an n-dimensional sample of the input variable. The first column of the model matrix $\boldsymbol{X}$ is $\boldsymbol{X}_{\cdot,1} = (1, \cdots, 1)^T \in \mathbb{R}^n$ and represents the intercept term. Each observation $\boldsymbol{x}_i$ has an outcome $y_i \in \{0, 1\}$. The likelihood function $L(\boldsymbol{\beta})$ represents the joint probability of $y_1, \cdots, y_n$, written as the function of the parameter $\boldsymbol{\beta}$,

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} \pi_{y_i}(\boldsymbol{x}_i, \boldsymbol{\beta})$$

with $\pi_{y_i}(\boldsymbol{x}_i, \boldsymbol{\beta}) = \mathbb{P}(Y = y_i \mid \mathcal{X} = \boldsymbol{x}_i)$ and

$$\boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \boldsymbol{\beta}_1 \end{pmatrix} \in \mathbb{R}^{(p+1)}. \tag{2.5}$$

A maximum likelihood estimator maximizes the likelihood function. In practice, the logarithm of the likelihood is more convenient and results in the log-likelihood function denoted as $l(\boldsymbol{\beta})$. Maximizing the log-likelihood is equivalent to maximizing the likelihood function itself, and therefore, the maximum likelihood estimator can be defined as:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{(p+1)}}{\arg\max}\ l(\boldsymbol{\beta}).$$

The log-likelihood function can be depicted in the following way:

$$\begin{aligned} l(\boldsymbol{\beta}) &= \sum_{i=1}^{n} \log \pi_{y_i}(\boldsymbol{x}_i, \boldsymbol{\beta}) \\ &= \sum_{i=1}^{n} \left\{ y_i \log \pi\left(\boldsymbol{x}_i, \boldsymbol{\beta}\right) + (1 - y_i) \log \left[1 - \pi\left(\boldsymbol{x}_i, \boldsymbol{\beta}\right)\right] \right\}. \end{aligned} \tag{2.6}$$

The common way to formally describe the maximum likelihood estimator for logistic regression is as a minimizer of the sum of deviances d_i:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{(p+1)}}{\arg\min} \sum_{i=1}^{n} d_i(\boldsymbol{x}_i^T \boldsymbol{\beta}; y_i) \tag{2.7}$$

with

$$d_i(\boldsymbol{x}_i^T \boldsymbol{\beta}; y_i) = -y_i \log \pi\left(\boldsymbol{x}_i, \boldsymbol{\beta}\right) - (1 - y_i) \log \left[1 - \pi\left(\boldsymbol{x}_i, \boldsymbol{\beta}\right)\right]. \tag{2.8}$$

Once the logistic regression parameters are known, the posterior probabilities are computed by applying formulas in Equation (2.3) and Equation (2.4). Afterwards, the estimated probabilities $\hat{\pi}$ are compared with a cut-off value c, where the output value of a new observation is predicted as one if $\hat{\pi} > c$, and zero if otherwise. The common cut-off value is 0.5, but different cut-off values can be used for specific settings.

The parameter $\boldsymbol{\beta}$ is usually estimated with the Newton-Raphson algorithm. Combining the log-likelihood function as shown in Equation (2.6) with the posterior probabilities from Equation (2.3) and Equation (2.4), the objective can be further rewritten as

$$\begin{aligned} l(\boldsymbol{\beta}) &= \sum_{i=1}^{n} \left\{ y_i \log \left[\frac{\exp\left(\boldsymbol{\beta}^\top \boldsymbol{x}_i\right)}{1 + \exp\left(\boldsymbol{\beta}^\top \boldsymbol{x}_i\right)} \right] + (1 - y_i) \log \left[\frac{1}{1 + \exp\left(\boldsymbol{\beta}^\top \boldsymbol{x}_i\right)} \right] \right\} \\ &= \sum_{i=1}^{n} \left\{ y_i \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i \log \left[1 + \exp\left(\boldsymbol{\beta}^\top \boldsymbol{x}_i\right)\right] - (1 - y_i) \log \left[1 + \exp\left(\boldsymbol{\beta}^\top \boldsymbol{x}_i\right)\right] \right\} \\ &= \sum_{i=1}^{n} \left\{ y_i \boldsymbol{\beta}^\top \boldsymbol{x}_i - \log \left[1 + \exp\left(\boldsymbol{\beta}^\top \boldsymbol{x}_i\right)\right] \right\}. \end{aligned}$$

Since the maximum likelihood method involves maximizing the objective log-likelihood, the first derivative is set to zero. The resulting first-order equations are also called *score* equations:

$$\begin{aligned}\frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= \sum_{i=1}^{n} \left\{ y_i \boldsymbol{x}_i - \frac{1}{1+\exp\left(\boldsymbol{\beta}^\top \boldsymbol{x}_i\right)} \exp\left(\boldsymbol{\beta}^\top \boldsymbol{x}_i\right) \boldsymbol{x}_i \right\} \\ &= \sum_{i=1}^{n} \left[y_i - \pi\left(\boldsymbol{x}_i, \boldsymbol{\beta}\right) \right] \boldsymbol{x}_i = 0.\end{aligned} \tag{2.9}$$

The score equations are nonlinear in $\boldsymbol{\beta}$ and are therefore solved in an iterative manner, by reweighting the ordinal least-square. The second derivative or Hessian matrix of the log-likelihood is likewise required for the method:

$$\frac{\partial^2 l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} = -\sum_{i=1}^{n} \pi\left(\boldsymbol{x}_i; \boldsymbol{\beta}\right) \left[1 - \pi\left(\boldsymbol{x}_i; \boldsymbol{\beta}\right)\right] \boldsymbol{x}_i \boldsymbol{x}_i^\top. \tag{2.10}$$

Starting from $\boldsymbol{\beta}_{\text{old}}$, the updated value $\boldsymbol{\beta}_{\text{new}}$ corresponds to

$$\boldsymbol{\beta}_{\text{new}} = \boldsymbol{\beta}_{\text{old}} - \left(\frac{\partial^2 l}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} (\boldsymbol{\beta}_{\text{old}}) \right)^{-1} \frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}. \tag{2.11}$$

The Newton-Raphson algorithm obtains a simpler formulation in a matrix representation. The multivariate form of the required variables is listed below:

- $\boldsymbol{y} \in \mathbb{R}^{n \times 1}$ vector of the y_i,
- $\boldsymbol{X} \in \mathbb{R}^{n \times (p+1)}$ data matrix containing observations $\boldsymbol{x}_i$,
- $\boldsymbol{\pi} \in [0,1]^{n \times 1}$ vector of estimated probabilities $\pi\left(\boldsymbol{x}_i, \boldsymbol{\beta}_{\text{old}}\right)$,
- $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ diagonal matrix with weights $\pi\left(\boldsymbol{x}_i, \boldsymbol{\beta}_{\text{old}}\right)\left(1 - \pi\left(\boldsymbol{x}_i, \boldsymbol{\beta}_{\text{old}}\right)\right)$ in the diagonal.

Accordingly, the first and second derivatives of the log-likelihood function given by Equation (2.9) and Equation (2.10), respectively, can be expressed in matrix notation as

$$\begin{aligned}\frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= \boldsymbol{X}^\top (\boldsymbol{y} - \boldsymbol{\pi}), \\ \frac{\partial^2 l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} &= -\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X}.\end{aligned}$$

Thus, the update step of the Newton-Raphson algorithm described in Equation (2.11) leads to the following expression in matrix form:

$$\begin{aligned}\boldsymbol{\beta}_{\text{new}} &= \boldsymbol{\beta}_{\text{old}} + \left(\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top (\boldsymbol{y} - \boldsymbol{\pi}) \\ &= \left(\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{W} \left(\boldsymbol{X}\boldsymbol{\beta}_{old} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{\pi})\right) \\ &= \underbrace{\left(\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{W}}_{\text{weighted LS}} \boldsymbol{z},\end{aligned}$$

with the adjusted response

$$\boldsymbol{z} = \boldsymbol{X}\boldsymbol{\beta}_{\text{old}} + \underbrace{\boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{\pi})}_{\text{adjustment}}). \tag{2.12}$$

The Newton-Raphson algorithm is also referred to as iteratively reweighted least-square (IRLS) algorithm because each iteration solves the weighted least-squares problem,

$$\boldsymbol{\beta}_{\text{new}} = \underset{\boldsymbol{\beta}}{\text{argmin}} (\boldsymbol{z} - \boldsymbol{X}\boldsymbol{\beta})^\top \boldsymbol{W} (\boldsymbol{z} - \boldsymbol{X}\boldsymbol{\beta}).$$

3. Cost-sensitive Learning for the Class Imbalance Problem

Given the augmentation of new data amounts and the simultaneous increase of the respective data utilization, new challenges have emerged in the data-driven area of statistical learning. As a result, various statistical methods evolved to solve commonly occurring issues, such as for instance imbalanced learning. Generally, imbalanced learning is defined as the learning process for data representation and information extraction with severe data distribution skews to develop effective decision boundaries to support the decision-making process [4]. Given a binary classification, an imbalanced problem refers to highly imbalanced class distributions of an output variable, introducing the terms of positive and negative classes for the minority and majority class, respectively [5]. If the class imbalance problem is not considered in advance or during the implementation of a statistical learning method, the resulting classifier could result in poor predictive performance, particularly for the minority class. The common approaches in imbalanced learning involve sampling methods and cost-sensitive learning, but diverse learner-specific methods also include kernel-based learning, active learning, one-class learning, and ensemble methods [4].

Sampling methods tackle the class imbalance problem before applying a statistical learning model in a straightforward manner and thereby dominate the imbalance learning approaches available [4]. In general, sampling methods refer to the modification of an imbalanced data set by a given mechanism in order to provide a balanced distribution, and thereby include random oversampling, random undersampling, synthetic sampling with data generation, cluster-based sampling methods, and integration of sampling and boosting [4]. In comparison, cost-sensitive learning methods target the problem of imbalanced learning by using different cost matrices that describe the costs for misclassifying any particular data example [4]. There are three broad approaches to implement cost-sensitive learning for imbalanced data. The first approach applies misclassification costs to the data set as a form of data-space weighting – these techniques are essentially cost-sensitive bootstrap sampling approaches where misclassification costs are used to select the best training distribution. In the second approach, cost-minimizing techniques are applied for the combination schemes of ensemble methods. Finally, the third approach incorporates cost-sensitive features directly into classification paradigms to fit the cost-sensitive framework into the classifiers [4]. This work focuses on the latter approach by implementing the costs into the logistic regression model and simply refers to it as "cost-sensitive learning".

3.1. Cost-sensitive Logistic Regression

In a common cost-sensitive learning method, each observation i acquires a corresponding cost c_i. When a cost-sensitive model is used for the imbalance data problem in a binary classification, there are usually two different cost values for the output variable classes. The cost matrix is employed to express the different classification error costs, as shown in Table 3.1. The value c_{FN} represents the weights that classify a positive class as a negative class, whereas the weights that categorize a negative class as a positive class correspond to the value c_{FP}.

	True Positive $y_i = 1$	True Negative $y_i = 0$
Predicted Positive $\hat{y}_i = 1$	0	c_{FP}
Predicted Negative $\hat{y}_i = 0$	c_{FN}	0

Table 3.1.: The cost matrix for the binary classification task

The misclassification costs are further incorporated into the logistic regression model and the log-likelihood function in Equation (2.6) is therefore modified, resulting in the cost-sensitive loss function for the logistic regression:

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} c_i \log \pi_{y_i}(\boldsymbol{x}_i, \boldsymbol{\beta}) \tag{3.1}$$

with

$$c_i = \begin{cases} c_{FP}, & \text{for } y_i = 0 \\ c_{FN}, & \text{for } y_i = 1 \end{cases}$$

Accordingly, the maximum likelihood estimator for the cost-sensitive logistic regression is defined as:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{(p+1)}}{\arg\min} \sum_{i=1}^{n} c_i \cdot d_i(\boldsymbol{x}_i^T \boldsymbol{\beta}; y_i) \tag{3.2}$$

where d_i are deviances defined in Equation (2.8).

Comparable to the non-cost-sensitive setting, the maximum likelihood estimator $\hat{\boldsymbol{\beta}}$ for cost-sensitive logistic regression with the log-likelihood given in Equation (3.1) can be estimated using the IRLS algorithm. For a vector $\boldsymbol{c} \in \mathbb{R}^n$ containing the cost c_i in the i-th entry, the first and second derivatives of the log-likelihood can be expressed in a matrix notation as

$$\frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \boldsymbol{X}^\top \operatorname{Diag}(\boldsymbol{c})\,(\boldsymbol{y} - \boldsymbol{\pi}),$$
$$\frac{\partial^2 l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^\top} = -\boldsymbol{X}^\top \operatorname{Diag}(\boldsymbol{c}) \boldsymbol{W} \boldsymbol{X},$$

with $\operatorname{Diag}(\boldsymbol{c})$ defined as the square diagonal matrix with the elements of the vector $\boldsymbol{c}$ on the main diagonal. Therefore, the update step of the IRLS algorithm converts to

$$\begin{aligned}
\boldsymbol{\beta}_{\text{new}} &= \boldsymbol{\beta}_{\text{old}} + \left(\boldsymbol{X}^\top \operatorname{Diag}(\boldsymbol{c}) \boldsymbol{W} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \operatorname{Diag}(\boldsymbol{c})(\boldsymbol{y} - \boldsymbol{\pi}) \\
&= \left(\boldsymbol{X}^\top \operatorname{Diag}(\boldsymbol{c}) \boldsymbol{W} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \operatorname{Diag}(\boldsymbol{c}) \boldsymbol{W} \left(\boldsymbol{X}\boldsymbol{\beta}_{old} + \boldsymbol{W}^{-1}(\boldsymbol{y} - \boldsymbol{\pi})\right) \\
&= \left(\boldsymbol{X}^\top \boldsymbol{W}_c \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{W}_c \boldsymbol{z},
\end{aligned}$$

with $\boldsymbol{W}_c := \operatorname{Diag}(\boldsymbol{c})\boldsymbol{W}$ and the adjusted response $\boldsymbol{z}$ as in Equation (2.12). Thus, each iteration of the IRLS algorithm for the cost-sensitive logistic regression solves the weighted least squares problem

$$\boldsymbol{\beta}_{\text{new}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} (\boldsymbol{z} - \boldsymbol{X}\boldsymbol{\beta})^\top \boldsymbol{W}_c (\boldsymbol{z} - \boldsymbol{X}\boldsymbol{\beta}).$$

3.2. Model Evaluation for Imbalanced Data Sets

Analyzing the performance of a classifier by means of evaluation metrics plays a central role in the quality assessment of the given statistical learning model. In imbalanced learning, the choice of the evaluation metrics is crucial, as some evaluation metrics could deliver misleading results. In general, the performance of binary classifiers is initially examined with a confusion matrix shown in Table 3.2. The values in a confusion matrix represent the following measures:

- TP is the number of positive observations correctly classified as positive (True Positives),
- FP is the number of negative observations incorrectly classified as positive (False Positive),
- FN is the number of positive observations incorrectly classified as negative (False Negatives),
- TN is the number of negative observations correctly classified as negatives (True Negatives).

	True Positive	True Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

Table 3.2.: Confusion matrix for the binary classification task. The columns represent the actual class, and the rows show the class as predicted by the model.

Many standard evaluation metrics can be derived from the confusion matrix, and some of the most common are defined below:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}, \tag{3.3}$$

$$True\ Positive\ Rate\ (TPR)\ \text{or}\ Sensitivity = \frac{TP}{TP + FN}, \tag{3.4}$$

$$False\ Positive\ Rate\ (FPR) = \frac{FP}{TN + FP}. \tag{3.5}$$

Accuracy is the evaluation measure used most frequently for model quality assessment, yet it is an inappropriate measure for imbalanced data since dummy classifiers predicting the majority class achieve high accuracy with poor predictive ability [6].

The true positive rate is also labeled specificity, and the true negative rate, given as $1 - FPR$, is usually denominated sensitivity. The latter measures (specificity and sensitivity), given by Equation (3.4) and Equation (3.5), are used to construct the receiver operating characteristic (ROC) curve, a standard technique for evaluating classifiers on data sets that exhibit a class imbalance [6]. Examples of diverse ROC curves are shown in Figure 3.1. The ROC space opposes the FPR (x-coordinate) and the TPR (y-coordinate). An ideal classifier would correspond to the $(0, 1)$ point in the ROC space, with all positive instances correctly classified, and no misclassified negative instances. Moreover, the line $y = x$ corresponds to a classifier that applies a random prediction to each instance and as such, provides a lower bound of the ROC space. Each point of an ROC curve is generated by moving the decision boundary for classification, whereby the points nearer to the left in the ROC space are the result of requiring a higher threshold for classifying an instance as positive. Therefore, ROC curves are also used to determine the decision threshold that gives the best TPR for an acceptable FPR (Neyman–Pearson method) [6].

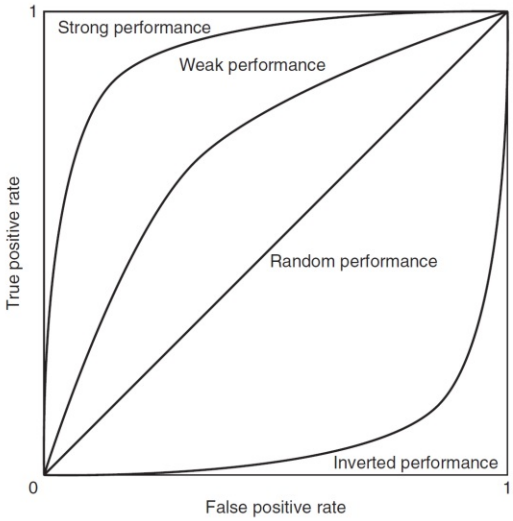


Figure 3.1.: Examples of ROC curves. The x-axis represents the FPR, with the TPR on the y-axis. Each curve represents the performance of a different classifier on a data set [6].

While ROC curves represent a visual method for assessing the effectiveness of a classifier, the area under the ROC curve (AUC) presents the corresponding metric used for evaluating classifiers under imbalance. AUC is useful because it is independent of the selected threshold and prior probabilities, but also offers a single value to compare classifiers. This work considers the normalization of the AUC measure, the Gini index:

$$gini = 2 \cdot AUC - 1 \tag{3.6}$$

Gini terminology originally stems from the area of finance and is frequently used for credit scoring models. The Gini index was introduced in [7] and was initially not derived from the AUC measure but rather connected via Equation (3.6) ([8] gives some discussion on the relationship of the Gini index with the AUC). This work will merely consider the Gini index as a normalization of the AUC measure, with the value range in $[0, 1]$. Since the Gini index proportionally depends on the AUC measure, a higher value of the Gini index points to a better predictive ability.

4. Robustness

All statistical methods explicitly or implicitly require a set of assumptions. These assumptions aim at formally defining the modeling problem at hand and thereby rendering theoretically and computationally manageable. However, those assumptions are rarely satisfactory in real-world modeling problems, and the formal models are only simplifications of reality trying to provide the best approximate solution.

Modeling assumptions mostly contain requirements on the data distribution, which determine desired statistical properties of the estimators. Such methods are called classical statistical methods and rely on the assumption that the data distribution holds entirely. Classical statistics are theoretically and computationally convenient but do not always deliver an adequate tool for the statistics application in data analysis. In practice, the data distribution model usually holds approximately and describes most of the data points, yet a minority of observations often follows another pattern or no pattern at all. Such atypical data points, which are separated from the majority of the data, are called outliers, and they can have an immense influence on classical statistics models. For instance, if the data are assumed to be normally distributed but the actual distribution has heavy tails, then the estimates based on classical statistics methods can result in unacceptably low statistical efficiency [9].

In contrast to classical statistics, robust statistics aim at deriving methods that produce reliable statistical estimates. This is not only the case when the data completely follow a given distribution but also when data contains a fraction of outliers. Robust methods fit the majority of the data - if data does not contain any outliers, the robust method approximately gives the same results as the classical method. However, if a proportion of outliers is present, the robust method approximately delivers the same results as the classical method applied to the "typical" data. This section thus provides the main concepts and types of robust estimators required for the implementation of robust logistic regression [9].

4.1. Robust Location and Covariance

In the multivariate location and covariance setting, the sampled data are represented by a model matrix $\boldsymbol{X} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_p) = (\boldsymbol{x}_{1.}, \cdots, \boldsymbol{x}_{n.})^T \in \mathbb{R}^{n \times p}$, where each row i represents a p-dimensional multivariate observation $\boldsymbol{x}_{i.}$ and each column j an n-dimensional sample of the j-th input variable $\boldsymbol{x}_j$. The observations are assumed to be sampled from an elliptically symmetric unimodal distribution with two unknown parameters, a p-dimensional vector $\boldsymbol{\mu}$ and a positive definite $p \times p$ matrix $\boldsymbol{\Sigma}$ [10]. A multivariate distribution is called elliptically symmetric and unimodal if a strictly decreasing real function g exists, so that the density function can be written in the form

$$f(\boldsymbol{x}) = \frac{1}{\sqrt{|\boldsymbol{\Sigma}|}}\, g\left(d^2(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})\right), \tag{4.1}$$

with the statistical distances $d(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ defined as

$$d(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) := \sqrt{(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})} \tag{4.2}$$

The parameter $\boldsymbol{\mu}$ represents the mean of the distribution, and the parameter $\boldsymbol{\Sigma}$ the variance-covariance matrix. Estimation of the latter two measures is of crucial importance as the parameters of location and covariance represent the initial step in the data analysis and are also required for nearly all statistical methods.

Classical statistics estimates of location and covariance are the well-known arithmetic mean and the empirical variance-covariance matrix. The arithmetic mean of the variable $\boldsymbol{x}_i$ is given as

$$\overline{x}_i = \frac{1}{n} \sum_{k=1}^{n} x_{ki}.$$

The multivariate estimate of the location parameter $\boldsymbol{\mu}$ is a vector of the arithmetic means of the p input variables

$$\overline{\boldsymbol{x}} = (\overline{x}_1, \cdots, \overline{x}_p)^T. \tag{4.3}$$

The empirical covariance between two variables $\boldsymbol{x}_j$ and $\boldsymbol{x}_k$ equals to

$$s_{jk} = \frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k). \tag{4.4}$$

Furthermore, the empirical variance of a variable $\boldsymbol{x}_j$ is obtained from Equation (4.4) for $j = k$. The classical statistics multivariate estimate of the variance-covariance matrix $\boldsymbol{\Sigma}$ has empirical variances and covariances as entries:

$$\boldsymbol{S}_{jk} = s_{jk}. \tag{4.5}$$

There are various robust approaches for estimation of location and covariance, but many estimates do not have the same algebraic properties as the parameters they represent. Robust location and covariance estimates should respond in a mathematically convenient form to specific transformations of the data. For a non-singular $p \times p$ matrix $\boldsymbol{A}$ and a vector $\boldsymbol{b}$ of length p, the linear transformation of the a p-dimensional observation $\boldsymbol{x}_{i.}$ is given as $\boldsymbol{A}\boldsymbol{x}_{i.} + \boldsymbol{b}$. When a transformation is applied on all the observations $\boldsymbol{x}_{1.}, \cdots, \boldsymbol{x}_{n.}$, it can be written in a matrix form $\boldsymbol{X}\boldsymbol{A}^T + \mathbf{1}_n \boldsymbol{b}^T$, where $\mathbf{1}_n = (1, \cdots, 1)^T \in \mathbb{R}^n$. The robust estimate of location $\hat{\boldsymbol{\mu}}_R$ should therefore fulfill:

$$\hat{\boldsymbol{\mu}}_R \left(\boldsymbol{X}\boldsymbol{A}^T + \mathbf{1}_n \boldsymbol{b}^T \right) = \hat{\boldsymbol{\mu}}_R(\boldsymbol{X})\boldsymbol{A}^T + \boldsymbol{b}, \tag{4.6}$$

and the robust estimate of the variance-covariance matrix $\hat{\boldsymbol{\Sigma}}_R$ should satisfy the criterion

$$\hat{\boldsymbol{\Sigma}}_R \left(\boldsymbol{X}\boldsymbol{A}^T + \mathbf{1}_n \boldsymbol{b}^T \right) = \boldsymbol{A}\hat{\boldsymbol{\Sigma}}_R(\boldsymbol{X})\boldsymbol{A}^T. \tag{4.7}$$

The robust estimators that meet the requirements given by Equation (4.6) and Equation (4.7) are called affine equivariant estimators. These estimators transform orderly considering changes of the origin, the scale, or under rotations.

The difference between the classical and robust estimates of location and covariance is illustrated in a two-dimensional setting. The corresponding bivariate data contains 110 points, where 100 points are sampled from a standard normal distribution, and ten outliers are added apart from the data cloud. Figure 4.1 shows the scatter plot of the obtained data.

Furthermore, Figure 4.1 also displays two ellipses. The classical tolerance ellipse is defined as a set of p-dimensional points $\boldsymbol{x}$ whose Mahalanobis distance

$$\mathrm{MD}(\boldsymbol{x}) = d(\boldsymbol{x}, \overline{\boldsymbol{x}}, \boldsymbol{S}) = \sqrt{(\boldsymbol{x} - \overline{\boldsymbol{x}})^T \boldsymbol{S}^{-1} (\boldsymbol{x} - \overline{\boldsymbol{x}})} \tag{4.8}$$

equals $\sqrt{\chi^2_{p,0.975}}$. The Mahalanobis distance showcases the distance of an observation from the center of the data cloud relative to its size and shape. In contrast, the robust tolerance ellipse is based on the robust distances

$$\mathrm{RD}(\boldsymbol{x}) = d\left(\boldsymbol{x}, \hat{\boldsymbol{\mu}}_R, \hat{\boldsymbol{\Sigma}}_R\right) \tag{4.9}$$

where $\hat{\boldsymbol{\mu}}_R$ is the robust estimate of location and $\hat{\boldsymbol{\Sigma}}_R$ is the robust covariance estimate.

The classical tolerance ellipse (red) attempts to encompass all observations, and the outliers severely influence the covariance structure. On the contrary, the robust tolerance ellipse (blue) encapsulates the non-outlining data points, it is thus more compact and reflects the structure of the majority of the respective data. Based on Figure 4.1, the robust tolerance ellipse seems to better describe the data formation, hence it is statistically more informative.

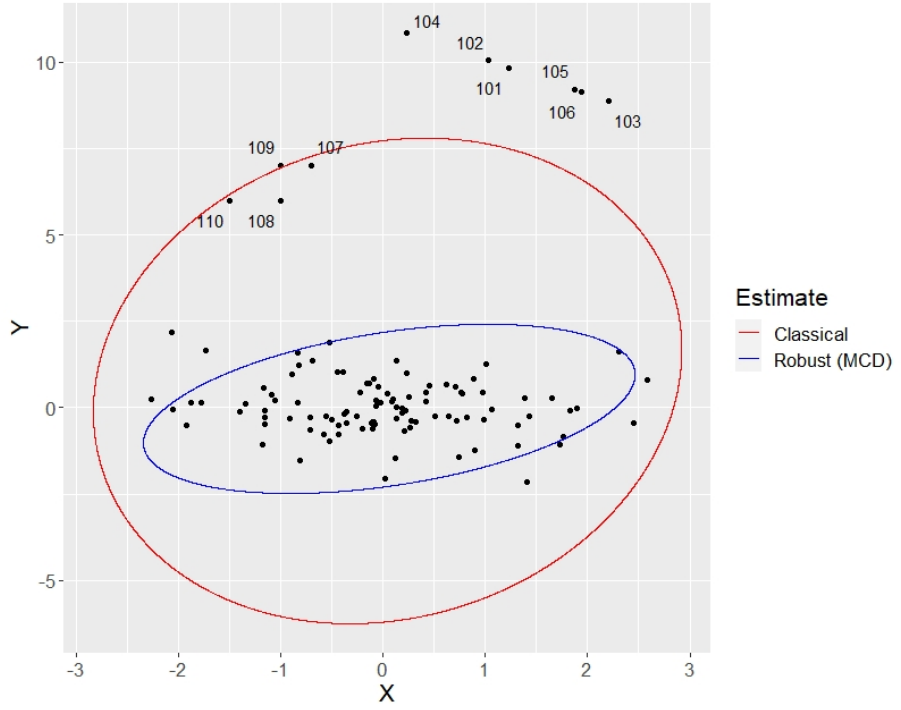


Figure 4.1.: A Scatter plot of simulated bivariate data with the indices of outliers printed next to the outlying points; the red ellipse shows the non-robust tolerance ellipse, and the blue ellipse represents the corresponding robust tolerance ellipse.

4.1.1. Minimum Covariance Determinant (MCD) Estimator

The Minimum Covariance Determinant (MCD) estimator is one of the first affine equivariant and highly robust estimators of multivariate location and covariance [10].

The raw Minimum Covariance Determinant estimator $(\hat{\boldsymbol{\mu}}_0, \hat{\boldsymbol{\Sigma}}_0)$ of location and covariance with a tuning constant $h \in [\frac{n}{2}, n]$ fulfills the following:

1. the set of h observations that generate the minimum determinant of the empirical variance-covariance matrix is identified,
2. the location estimate $\hat{\boldsymbol{\mu}}_0$ is given as the arithmetic mean of the identified h points
3. the covariance estimate $\hat{\boldsymbol{\Sigma}}_0$ is the empirical variance-covariance matrix of the identified h points, multiplied by a consistency factor c_0.

The factor c_0 is generated to obtain the consistency of the normal distribution and equals to $\alpha / F_{\chi^2_{p+2}}(q_\alpha)$ with $\alpha = \lim_{n\to\infty} h(n)/n$, and q_α the α-quantile of the the χ^2_p distribution. Moreover, a finite-sample correction factor can be incorporated as well.

The MCD estimator is the most robust for a constant $h = [(n + p + 1)/2]$, where $[a]$ is the largest integer with $[a] \leq a$. At the population level, this corresponds to $\alpha = 0.5$, yet such values of α result in a very low efficiency of the MCD estimator. The common value of α is 0.75, as it expresses a compromise between efficiency and robustness.

As a way of increasing efficiency, whilst also retaining high robustness, the weighting step is applied on the raw MCD estimator, yielding the MCD estimates for location and covariance

$$\hat{\boldsymbol{\mu}}_{MCD} = \frac{\sum_{i=1}^{n} W\left(d_i^2\right) \boldsymbol{x}_i}{\sum_{i=1}^{n} W\left(d_i^2\right)} \tag{4.10}$$

$$\hat{\boldsymbol{\Sigma}}_{MCD} = c_1 \frac{1}{n} \sum_{i=1}^{n} W\left(d_i^2\right) \left(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_{MCD}\right) \left(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_{MCD}\right)^T \tag{4.11}$$

with $d_i = d(\boldsymbol{x}, \hat{\boldsymbol{\mu}}_0, \hat{\boldsymbol{\Sigma}}_0)$, W as appropriate weighting function, and c_1 a consistency factor.

A simple and effective choice for the weight function W is

$$W(d^2) = \begin{cases} 1, & \text{if } d^2 \leqslant \chi^2_{p,0.975} \\ 0, & \text{otherwise.} \end{cases}$$

S-estimator

Compared to the MCD estimator, the S-estimator offers a different approach to address the problem of robust location and covariance estimation. The essence of the S-estimator lies in another robust estimator, namely the M-estimator of scale [9].

In the following, the univariate observations x_i, $i \in \{1, \cdots, n\}$ are assumed to satisfy the multiplicative model

$$x_i = \sigma u_i, \tag{4.12}$$

with u_i as independent and identically distributed (i.i.d) random variables with density function f_0 and a positive, unknown scale parameter σ. Density distribution functions of the random variables $\boldsymbol{x}_i$ as defined in Equation (4.12) establish a scale family with the density

$$\frac{1}{\sigma} f_0(\frac{x_i}{\sigma}).$$

Various distributions form a scale family, such as the exponential family with $f_0(x) = exp(-x)\mathbb{1}(x > 0)$ or the normal family $N(0, \sigma^2)$.

The ML estimator of the parameter σ satisfying Equation (4.12) is therefore

$$\hat{\sigma} = \arg\max_{\sigma} \frac{1}{\sigma^n} \prod_{i=1}^{n} f_0\left(\frac{x_i}{\sigma}\right). \tag{4.13}$$

Differentiating the log-likelihood function obtained from Equation (4.13) with respect to σ results in

$$\frac{1}{n}\sum_{i=1}^{n}\rho_s\left(\frac{x_i}{\widehat{\sigma}}\right)=1,$$

with

$$\rho_s(t):=-t\,\frac{f_0'(t)}{f_0(t)}.$$

The normal scale family where f_0 is the density function of the standard normal distribution $N(0,1)$ yields $\rho_s(t)=t^2$ and $\hat{\sigma}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}$, the eminent root-mean-square estimate (RMS).

In general, an estimate satisfying the criterion

$$\frac{1}{n}\sum_{i=1}^{n}\rho_s\left(\frac{x_i}{\widehat{\sigma}}\right)=\delta \tag{4.14}$$

with a positive constant δ is called M-estimator of scale. The necessary condition for the existence of a solution in Equation (4.14) is $0<\delta<\rho_s(\infty)$. Thus, for a bounded function ρ_s the assumption

$$\rho_s(\infty)=1,\quad \delta\in(0,1)$$

holds without loss of generality. Moreover, the derivative of the function ρ_s, $\psi_s:=\rho_s'$ is typically used instead of the function ρ_s itself, since the root of the derivative ψ_s corresponds to the minimum of the function ρ_s and often provides a clearer mathematical representation [11].

A common choice of the function ρ_s is the Tukey's biweight function, with derivative ψ_s given below:

$$\psi_s(y)=\begin{cases} y\left(1-(y/c_0)^2\right)^2, & \text{if } |y|<c_0\\ 0, & \text{if } |y|\geq c_0.\end{cases} \tag{4.15}$$

The M-estimator of scale is usually computed as the weighted RMS estimate. Namely, for the weights $W(x)$ defined as

$$W(x)=\begin{cases}\rho_s(x)/x^2 & \text{if} \quad x\neq 0\\ \rho_s''(0) & \text{if} \quad x=0\end{cases}$$

Equation (4.14) converts to

$$\widehat{\sigma}^2=\frac{1}{n\delta}\sum_{i=1}^{n}W\left(\frac{x_i}{\widehat{\sigma}}\right)x_i^2,$$

which is the solution of the weighted RMS.

M-estimators of scale do not fulfill Equation (4.7) and are therefore not affine equivariant, yet they satisfy the homogeneity of degree one

$$\hat{\sigma}(c\,x) = c\,\hat{\sigma}(x).$$

Finally, the S-estimator aims at minimizing the statistical distances using M-estimators. For the estimators of location and covariance $\boldsymbol{t}$ and $\boldsymbol{C}$, respectively, the squared statistical distances of the multivariate observations

$$d^2(\boldsymbol{x}_i, \boldsymbol{t}, \boldsymbol{C}) = (\boldsymbol{x}_i - \boldsymbol{t})^T \boldsymbol{C}^{-1} (\boldsymbol{x}_i - \boldsymbol{t}), \quad i \in \{1, \cdots, n\}$$

should be minimized. Small statistical distances are achieved with the M-estimator of scale

$$\hat{\sigma}\left(d^2(\boldsymbol{x}_1, \boldsymbol{t}, \boldsymbol{C}), \cdots, d^2(\boldsymbol{x}_n, \boldsymbol{t}, \boldsymbol{C})\right)$$

under the restriction on the determinant of $\boldsymbol{C}$, $\det(\boldsymbol{C}) = 1$.

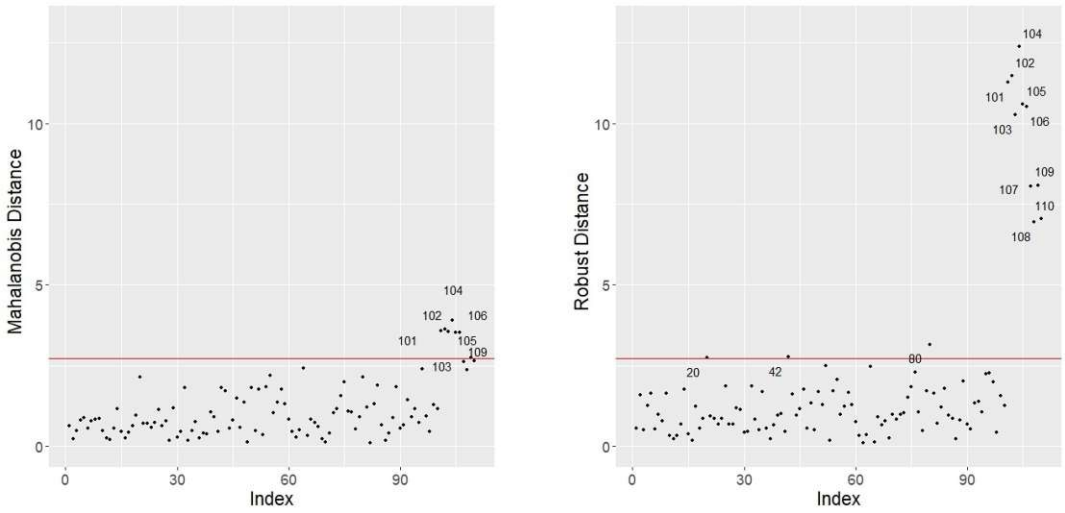
4.2. Outlier Detection

This section introduces two different methods for outlier detection using robust estimates of location and covariance. Section 4.2.1 presents the usual procedure for detecting outliers using any location and covariance estimator, focusing on the difference in outlier detection when applying classical and robust estimates, in the latter case using the MCD estimator. In contrast, Section 4.2.2 provides the *PCDist* algorithm proposed by Shieh and Hung [11], which first performs data preprocessing before computing the robust estimates of scale and covariance.

4.2.1. Outlier Detection using MCD Estimator

An important application of the MCD estimator is not only to provide robust versions of the location and covariance estimates, but also to detect outliers in multivariate data.

In order to illustrate the difference between the Mahalanobis and robust distances used to identify the outlying points in data, Figure 4.2 shows both types of distances obtained from the data presented in Figure 4.1. Figure 4.2a shows the Mahalanobis distances, whereas Figure 4.2b depicts the robust distances of the data. The red line displayed on both subfigures is at a height of $\sqrt{\chi^2_{p,0.975}}$, and the data points above the line are considered outliers. According to the non-robust Mahalanobis distances, seven out of ten outliers are detected, in contrast to the robust distances, which correctly identified all ten outliers. Moreover, the Mahalanobis distances of the outlying points are very close to the red border line. This illustrates the so-called masking effect, which emerges when classical estimates are strongly affected by contamination, so that diagnostic tools, such as the Mahalanobis distances, are incapable to detect outliers.



(a) Mahalanobis distances (b) Robust Distances

Figure 4.2.: Two different types of distances derived from classical (a) and robust (b) statistics; the x-axis represents the index of the observation, and the y-axis the corresponding distance; the red line has a height of $\sqrt{\chi^2_{p,0.975}}$; the indices of outliers are displayed next to the outlying points

Since the robust distances are not sensitive to the masking effect, the outlier detection using the MCD estimator can provide more reliable outlier diagnostics compared to classical variance-covariance estimates [10].

4.2.2. The PCDist Algorithm for Outlier Detection

Most data sets originating from real-world settings contain various explanatory variables, rendering the use of direct statistical methods computationally impractical or even impossible. For instance, the inverse of a covariance matrix required for the statistical distances in Equation (4.2) often does not exist due to the singularity of the matrix. The singularity of the covariance matrix mainly stems from the multicollinearity between inputs, which is not directly related to the number of variables, yet most high-dimensional data sets cannot provide the inverse of the covariance matrix. Moreover, as shown in Section 4.2.1, typical methods for detecting outliers rely on computing a distance function for each observation. However, due to the data sparsity in high dimensions, these distances are practically meaningless [11]. Therefore, a dimension reduction performed before applying the robust methods can enable and improve the outlier detection. The *PCDist* algorithm performs a principal component analysis (PCA) as a first step towards outlier detection.

For a data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ with n observations and p variables, classical PCA projects the data on the n principal components, which are a linear combination of the input variables

$$\boldsymbol{w}_i = \boldsymbol{X}\boldsymbol{v}_i$$

that maximize the variance

$$\boldsymbol{v}_i = \underset{\boldsymbol{v}^T\boldsymbol{v}=1}{\arg\max}(\boldsymbol{X}\boldsymbol{v})$$

subject to the constraint of orthogonality

$$\mathrm{Cov}(\boldsymbol{w}_i, \boldsymbol{w}_j) = 0,\ \forall j < i,\ i, j \in \{1, \cdots, n\}.$$

The principal components are organized by the amount of variance in the data that they explain. Essentially, PCA transforms the data space into an orthogonal space where higher principal components explain more variance in the data. Although PCA generates n principal components, typically only $m < n$ principal components are employed, which explain an adequate amount of the variance, thereby reducing the dimensionality of the data. There are several methods for adjusting the required number of principal components, yet an automatic selection method based on the scree plot from [12] is used.

In addition, the PCDist algorithm also allows for data grouping. When the data are arranged into distinct classes, the outlier detection can be performed for each class separately, which improves the outlier detection in case of different distributions between the groups. The PCDist algorithm uses the S-estimator and can be summarized as follows [13]:

1. Dimension reduction. The first step consists of performing PCA on the entire data, ignoring the class structure. An adequate number of principal components m is automatically attained, and the subsequent algorithm steps are performed in the reduced PCA space.
2. Each class j in the selected low-dimensional space is subjected to outlier detection:
   - Robust multivariate location and covariance estimates $(\boldsymbol{t}, \boldsymbol{C})$ are computed using the S-estimator.
   - The robust distances $RD_i = d(\boldsymbol{x}_i, \boldsymbol{t}, \boldsymbol{C})$ are calculated according to Equation (4.2).
   - Robust distances RD_i are compared to a threshold $\sqrt{\chi^2_{p,0.975}}$. The outliers in the class j are observations with $RD_i > \sqrt{\chi^2_{p,0.975}}$.
3. The final outlier set is defined as the union of the outliers from each group j.

4.3. Robust Logistic Regression

The outlier detection in regression imposes new challenges compared to simply identifying outliers in a data set. First and foremost, the regression context enables the categorization of outliers, as described in Section 4.3.1. The outlier types affect the regression estimates differently and are therefore managed in another manner. This section defines the Bianco-Yohai estimator, a highly robust M-type estimator for logistic regression. The first M-type estimator for regression, defined for the linear regression and used as a basis for other regression types, is defined in Section 4.3.2. Finally, the Bianco-Yohai estimator for logistic regression is presented in Section 4.3.3.

4.3.1. Outliers in Regression

In a supervised statistical learning setting with an input matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and an output vector $\boldsymbol{y} \in \mathbb{R}^n$, the outliers can be divided into three groups. An outlying observation $(\boldsymbol{x}_i., y_i) \in \mathbb{R}^p \times \mathbb{R}$ can therefore be specified as follows:

1. Vertical outlier. Observation $\boldsymbol{x}_i$ is in the usual data range, but the corresponding output variable y_i does not fit the model,
2. Good leverage point. Observation $\boldsymbol{x}_i$ is an outlier, thus unusual in the x-space of the explanatory variables, but the corresponding output variable y_i fits the model,
3. Bad leverage point. Observation $\boldsymbol{x}_i$ is an outlier, thus unusual in the x-space of explanatory variables, and the corresponding output variable y_i does not fit the model

The three types of outliers in regression are illustrated in the case of simple linear regression. In general, the linear regression model is defined as

$$y_i = \boldsymbol{x}_i^T \boldsymbol{\beta} + r_i, \quad r_i \sim f_0(0, \sigma^2). \tag{4.16}$$

Simple linear regression refers to the linear regression model with only one input variable. An example of a data set for simple linear regression containing all three types of outliers is shown in Figure 4.3.

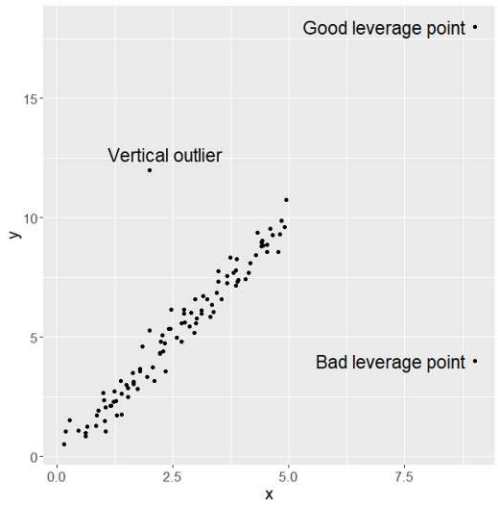


Figure 4.3.: Scatter plot of a data set suited for simple linear regression with one explanatory variable x on the x-axis and the output variable y on the y-axis. The types of outliers are denoted in the plot.

4.3.2. *M*-estimator for Linear Regression

The M-estimator for linear regression is based on a similar concept as the M-estimator of scale introduced in Section 4.1.1. For the linear regression model as defined in Equation (4.16), the variables y_i have the density function

$$\frac{1}{\sigma} f_0 \left(\frac{y_i - \boldsymbol{x}_i^T \boldsymbol{\beta}}{\sigma} \right).$$

Differentiating the log-likelihood function results in the definition of the M-estimator for regression:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \frac{1}{n} \sum_{i=1}^{n} \rho_0 \left(\frac{r_i(\boldsymbol{\beta})}{\hat{\sigma}} \right), \tag{4.17}$$

with function ρ_0 defined as $\rho_0 := -\log f_0$. An analogue to the normal equations emerges by setting the derivative of the term in Equation (4.17) to zero

$$\sum_{i=1}^{n} \psi_0 \left(\frac{r_i(\widehat{\boldsymbol{\beta}})}{\hat{\sigma}} \right) \boldsymbol{x}_i = 0, \tag{4.18}$$

where $\psi_0 := \rho_0'$. For $W(r) := \psi_0(r)/r$ and $w_i = W\left(r_i(\boldsymbol{\beta})/\hat{\sigma}\right)$, Equation (4.18) converts to

$$\sum_{i=1}^{n} w_i \left(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta} \right) \boldsymbol{x}_i = 0. \tag{4.19}$$

The M-estimator for regression can thus be characterized as weighted least-square and is therefore computed using the iterative reweighted least-squares algorithm. Let $\hat{\boldsymbol{\beta}}_0$ be the initial solution and $\hat{\boldsymbol{\beta}}_m$ be the approximation at iteration m, then the residuals $r_i = r_i(\hat{\boldsymbol{\beta}}_m)$ in iteration m provide the weights $w_i = W\left(r_i/\hat{\sigma}\right)$. The estimate for the next iteration $\hat{\boldsymbol{\beta}}_{m+1}$ is the solution of Equation (4.19). It is important to start with the robust estimator $\hat{\boldsymbol{\beta}}_0$, as the algorithm could converge to a non-robust solution otherwise.

4.3.3. Bianco-Yohai Estimator for Logistic Regression

As mentioned in Section 2.2, the maximum likelihood estimator for logistic regression is a minimizer of the sum of deviances:

$$\hat{\boldsymbol{\beta}}_{ML} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{(p+1)}}{\arg\min} \sum_{i=1}^{n} d_i(\boldsymbol{x}_i^T\boldsymbol{\beta}; y_i), \tag{4.20}$$

with deviances d_i given by Equation (2.8).

Although the ML method implies the most efficient statistical estimators, the efficiency does not persist in presence of outliers. The robust alternative for logistic regression is achieved by replacing the deviance function with another one, resulting in

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{(p+1)}}{\arg\min} \sum_{i=1}^{n} \varphi(\boldsymbol{x}_i^T\boldsymbol{\beta}; y_i). \tag{4.21}$$

In the cost-sensitive setting, in accordance to Equation (3.2), the robust estimator of interest is defined as

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{(p+1)}}{\arg\min} \sum_{i=1}^{n} c_i \cdot \varphi(\boldsymbol{x}_i^T\boldsymbol{\beta}; y_i), \tag{4.22}$$

with the misclassification costs c_i.

φ is a positive and almost everywhere differentiable function. Moreover, it needs to satisfy the condition

$$\varphi(s, 0) = \varphi(-s, 1) \tag{4.23}$$

for any score s, where the score is a dot product of the observation and the parameter vector $\boldsymbol{\beta}$, $s_i = \boldsymbol{x}_i^T\boldsymbol{\beta}$. Due to the condition in Equation (4.23), later calculations use the univariate function

$$\phi(s) := \varphi(s, 0)$$

instead of a bivariate function φ for easier computation. A term $\phi(s)$, which corresponds to an observation with $y = 0$, provides the impact of a score s for the value of the objective function in Equation (4.21). The function ϕ should be non-decreasing, as the large values of the score s should not relate to negative class observations and therefore should receive a greater weight in the objective minimisation function. Another requirement of the

function ϕ is $\lim_{s\to-\infty}\phi(s) = 0$, suggesting that the large negative values of the score s do not have an impact on the objective function.

An example of the ϕ function satisfying the latter demands is delineated in Figure 4.4, compared to the non-robust deviances. The classical deviances are unbounded and reach high values even for relatively small positive scores, while the robust version of the deviances remains stable and bounded for all values of scores, preventing significant impact of outliers on the objective function in Equation (4.21).

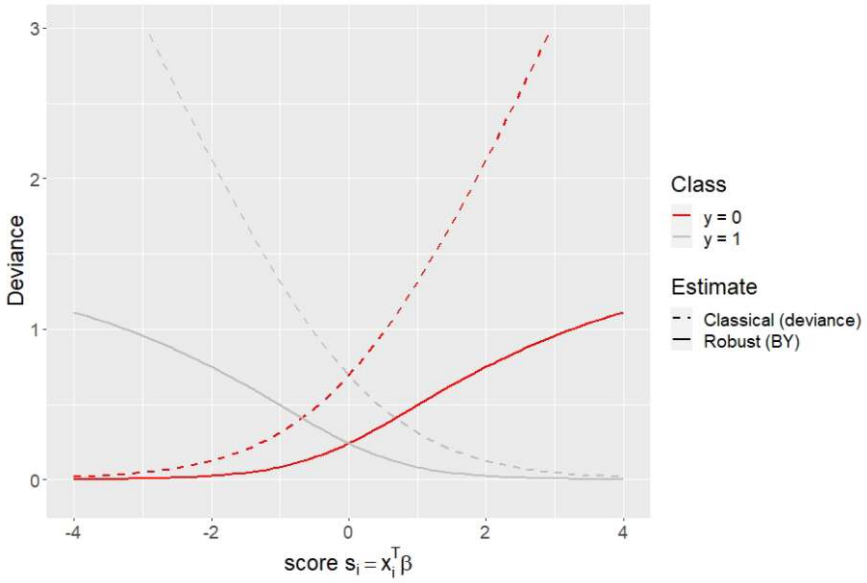


Figure 4.4.: Classical and robust version of deviances. The x-axis represents the score values; the y-axis captures the values of deviances. The color represents the output variable class of the scores, and the line type distinguishes the classical (dashed line) vs. robust (full line) estimate.

The robust estimator of interest given by Equation (4.21) belongs to the class of M-type estimators and follows the same principles as the M-estimator for linear regression described in Section 4.3.2, thus differentiating Equation (4.21) with respect to $\boldsymbol{\beta}$ yields the first-order condition

$$\frac{1}{n}\sum_{i=1}^{n}\Psi\left(\boldsymbol{x}_i^T\boldsymbol{\beta}; y_i\right)\boldsymbol{x}_i = 0,$$

where $\Psi(s;0) = \partial\varphi(s;0)/\partial s$ and $\Psi(s;1) = -\Psi(-s;0)$. Due to the latter property of Ψ, the function $\psi(s) := \Psi(s;0) = \phi'(s)$ is used instead of the bivariate notation. The ML estimator for logistic regression is an example of the M-estimator with $\phi_{\mathrm{ML}}(s) = -\ln(1-\pi(s))$.

Bianco and Yohai [14] proposed a highly robust version of the M-estimator for logistic regression, defined as

$$\hat{\boldsymbol{\beta}} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} \left\{ \rho\left(d\left(\boldsymbol{x}_i^T \boldsymbol{\beta}; y_i\right)\right) + C\left(\boldsymbol{x}_i^T \beta\right) \right\},$$

where $C(\boldsymbol{x}_i^T \boldsymbol{\beta})$ is a bias correction term given by

$$C(s) = G(\pi(s)) + G(1 - \pi(s)) - G(1),$$

with

$$G(t) = \int_0^t \rho'(-\ln u) \mathrm{d}u.$$

In particular, the Bianco-Yohai (BY) estimator corresponds to the following univariate function ϕ:

$$\phi_{\mathrm{BY}}(s) = \rho(-\ln(1 - \pi(s))) + G(\pi(s)) + G(1 - \pi(s)) - G(1). \tag{4.24}$$

The function ϕ_{BY} should satisfy the requirement $\lim_{s \to -\infty} \phi(s) = 0$. Obtained from the formation of the Bianco-Yohai estimator, it is evident that the resulting function ϕ_{BY} only depends on the choice of the function ρ.

One of the crucial characteristics of any statistical estimator is the set of conditions required for its existence. As for the ML method, the estimator exists once there is an overlap between positive and negative observations. Formally, it implies that for $I^0 = \{i \in \{1, \cdots, n\} \mid y_i = 0\}$ and $I^1 = \{i \in \{1, \cdots, n\} \mid y_i = 1\}$, there is no $\boldsymbol{\beta} \in \mathbb{R}^p$, such that

$$\boldsymbol{x}_i^T \boldsymbol{\beta} \geqslant 0 \quad \forall i \in I^1 \quad \text{and} \quad \boldsymbol{x}_i^t \boldsymbol{\beta} \leqslant 0 \quad \forall i \in I^0.$$

The overlap in data points is likewise required for the existence of M-estimators, yet the function $\psi = \phi'$ should meet some additional criteria.

Proposition 1. *[1] Let $\varphi : \mathbb{R}^2 \to \mathbb{R}$ be a positive function and put $\phi(s) = \varphi(s; 0) = \varphi(-s; 1)$. Assume that ϕ is a nondecreasing, continuous function with continuous derivative ψ such that $\lim_{s \to -\infty} \phi(s) = 0$. Let $\hat{\boldsymbol{\beta}}$ be the estimator for the parameters of a logistic regression model with intercept defined in Equation (4.21). If the following conditions:*

1. *There is an overlap in the sample.*
2. *There exists L_0 such that ψ is increasing on $(-\infty, L_0]$ and either decreasing or increasing on $[L_0, \infty)$.*
3. $\lim_{s \to \infty} \psi(st)/\psi(-s) = \infty \quad \forall t > 0,$

hold true, then the estimator $\hat{\boldsymbol{\beta}}$ exists and is finite in norm.

The first condition in Proposition 1 is identical with the existence criterion for the ML method. The second condition captures two different forms of the ψ function - it is either increasing, as for the ML estimator, or redescending. The third condition is trivially fulfilled for the increasing ψ, yet for the redescending form, it states that the function ψ should redescend to zero more quickly on the side of correctly classified observations ($s < 0$) than on the side of misclassified data points ($s > 0$).

The BY estimator given by Equation (4.24) requires the function ρ, which should be suitable with the conditions on $\psi = \phi'$. A function ρ with a derivative presented below satisfies the existence conditions from Proposition 1:

$$\rho'(t) = \begin{cases} \mathrm{e}^{-\sqrt{d}} & \text{if } t \leqslant d \\ \mathrm{e}^{-\sqrt{t}} & \text{otherwise,} \end{cases}$$

for a given constant d. The constant d is determined to attain the compromise between the efficiency and robustness – higher values of d result in the more efficient, but less robust estimator and vice versa. The typical value of d is 0.5. For the proposed derivative ρ', analytical forms of the corresponding functions ρ and G are given as

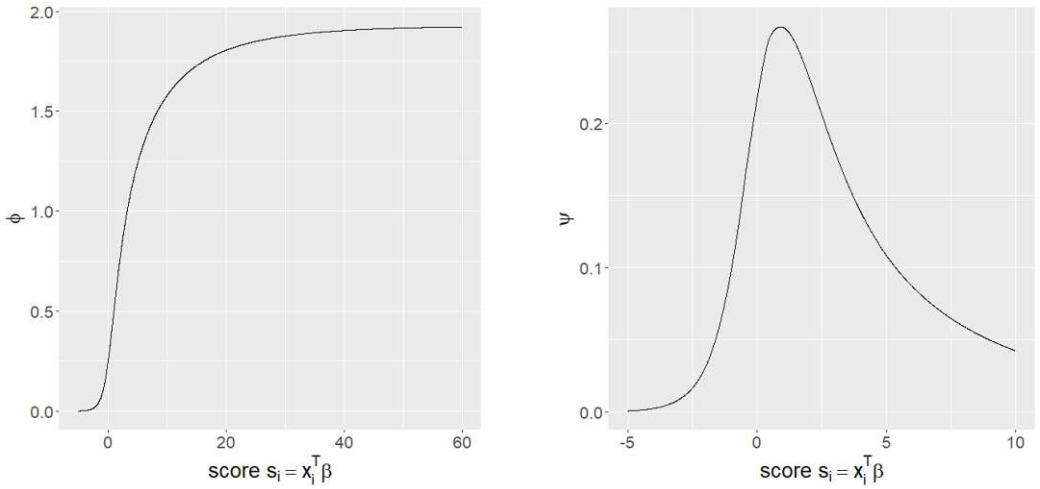
$$\rho(t) = \begin{cases} t\mathrm{e}^{-\sqrt{d}} & \text{if } t \leqslant d, \\ -2\mathrm{e}^{-\sqrt{t}}(1+\sqrt{t}) + \mathrm{e}^{-\sqrt{d}}(2(1+\sqrt{d})+d) & \text{otherwise} \end{cases}$$

and

$$G(t) = \begin{cases} t\mathrm{e}^{-\sqrt{-\ln t}} + \mathrm{e}^{1/4}\sqrt{\pi}\Phi\left(\sqrt{2}\left(\frac{1}{2} + \sqrt{-\ln t}\right)\right) - \mathrm{e}^{-1/4}\sqrt{\pi} & \text{if } t \leqslant \mathrm{e}^{-d} \\ \mathrm{e}^{-\sqrt{d}}t - \mathrm{e}^{-1/4}\sqrt{\pi} + \mathrm{e}^{1/4}\sqrt{\pi}\Phi\left(\sqrt{2}\left(\frac{1}{2} + \sqrt{d}\right)\right) & \text{otherwise} \end{cases}$$

where Φ is the normal cumulative distribution function. Attained functions ϕ and ψ of the BY estimator are shown in Figure 4.5.

An additional method aiming to produce a more robust BY estimator proposes a weighting step to downweight the leverage points. Leverage points can be identified by calculating statistical distances for a given location and covariate estimate. The classical location and covariance estimates are very sensitive to outliers and prone to the masking effect, thus the robust location and covariance versions are used instead. The proposed method uses the minimum covariance determinant estimator for location and covariance and identifies the leverage points using robust distances given by Equation (4.9).



(a) Function ϕ (b) Function ψ

Figure 4.5.: Functions ϕ (a) and ψ (b) of the Bianco-Yohai estimator

The weighted version of the Bianco Yohai estimator (WBY) is therefore defined as:

$$\hat{\boldsymbol{\beta}}_n = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \omega_i \varphi_{\mathrm{BY}} \left(z_i^t \boldsymbol{\beta}; y_i \right) \tag{4.25}$$

where the function φ_{BY} represents the φ function used in the minimization task for the BY estimator. The weight ω_i of the observation $\boldsymbol{x}_i$ equals to

$$\omega_i = \begin{cases} 1, & \text{if } RD(\boldsymbol{x}_i) \leq \sqrt{\chi^2_{p,0.975}} \\ 0, & \text{otherwise.} \end{cases} \tag{4.26}$$

5. Implementation

This section presents the algorithm for implementing the cost-sensitive Bianco-Yohai estimator for logistic regression used in imbalanced learning. It is an adaptation of the non-cost-sensitive algorithm for robust logistic regression using the Bianco-Yohai estimator introduced by Croux and Haesbroeck [1] and is presented in Section 5.1.

5.1. Algorithm for the Bianco-Yohai estimator

Similar to other M-type estimators, the Bianco-Yohai estimator is computed using an iterative algorithm. An important advantage of this method is the ability to detect the so-called *explosion* of the estimator. Namely, the criteria for the existence of the BY estimator stated in Proposition 1 require data that contain an overlap between classes, but the explosion is possible even in the presence of the class overlap, which makes it difficult to identify in advance.

The parameter of interest $\boldsymbol{\beta}$ is thus written as

$$\boldsymbol{\beta} = \frac{\boldsymbol{\xi}}{\sigma}, \tag{5.1}$$

with $\|\boldsymbol{\xi}\| = 1$ and $\sigma = \frac{1}{\|\beta\|} \geq 0$. The parameter $\boldsymbol{\xi}$ lies in the unit sphere of $\mathbb{R}^p$ denoted as S^{p-1}.

The optimization problem at hand, written in terms of the variables $\boldsymbol{\xi}$ and σ, corresponds to

$$(\hat{\sigma}, \hat{\boldsymbol{\xi}}) = \underset{(\sigma,\boldsymbol{\xi})\in\mathbb{R}^+\times S^{p-1}}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n} \varphi\left(\boldsymbol{x}_i^T \frac{\boldsymbol{\xi}}{\sigma}, y_i\right). \tag{5.2}$$

The objective function is minimized by altering between minimizing $\boldsymbol{\xi}$ and σ separately:

- Given the parameter $\boldsymbol{\xi}$, Equation (5.2) converts to the one-dimensional optimization problem in σ. Univariate nonlinear optimization problems are well studied and can be easily solved using established routines. The corresponding solution is denoted by $\hat{\sigma}_1$.

The parameter σ represents the inverse of $\|\boldsymbol{\beta}\|$ and therefore can detect the explosion of the parameter $\boldsymbol{\beta}$. In the case of an explosion, the estimate $\hat{\sigma}_1$ is numerically indistinguishable from zero, thus the algorithm stops and reports the explosion of the parameter.

- If the parameter σ is known, the optimization task in Equation (5.2) transforms into the minimization problem under constraint

$$\min f(\boldsymbol{\xi}) := \frac{1}{n}\sum_{i=1}^{n} \varphi\left(\boldsymbol{x}_i^T \frac{\boldsymbol{\xi}}{\sigma}, y_i\right) \text{ under } g(\boldsymbol{\xi}) := \boldsymbol{\xi}^T\boldsymbol{\xi} - 1 = 0. \tag{5.3}$$

An initial solution of Equation (5.3) is denoted by $\hat{\boldsymbol{\xi}}_0$. In the surrounding of $\hat{\boldsymbol{\xi}}_0$, the function f can be estimated with

$$f(\hat{\boldsymbol{\xi}}_0 + h) \approx f(\hat{\boldsymbol{\xi}}_0) + \operatorname{grad} f(\hat{\boldsymbol{\xi}}_0)^T h,$$

pointing to the largest decrease in the opposite direction of the gradient $\operatorname{grad} f(\hat{\boldsymbol{\xi}}_0)$. The gradient algorithm without constraints takes a step ϵh, with $h = -\operatorname{grad} f(\hat{\boldsymbol{\xi}}_0)$ and a small value of the scalar ϵ. However, the new value $\hat{\boldsymbol{\xi}}_0 + \epsilon h$ should satisfy the constraint to provide the valid solution to the optimization problem in Equation (5.3). Therefore, the surface $S := \{\boldsymbol{\xi} \in \mathbb{R}^p \mid g(\boldsymbol{\xi}) = 0\}$ is approximated by the tangent hyperplane at $\hat{\boldsymbol{\xi}}_0$, given by

$$S(\hat{\boldsymbol{\xi}}_0) = \{t \in \mathbb{R}^p \mid t = \hat{\boldsymbol{\xi}}_0 + v \text{ with } v^T \operatorname{grad} g(\hat{\boldsymbol{\xi}}_0) = 0\}.$$

In order to find the solution of the optimization task in Equation (5.3), the step size h should be determined such that $\hat{\boldsymbol{\xi}}_0 + \epsilon h$ approximately satisfies the constraint as an element of $S(\hat{\boldsymbol{\xi}}_0)$, while reaching the smallest value for $\epsilon \operatorname{grad} f(\hat{\boldsymbol{\xi}}_0)^T h$. Such step size is obtained by projecting $-\operatorname{grad} f(\hat{\boldsymbol{\xi}}_0)$ onto $S(\hat{\boldsymbol{\xi}}_0)$, resulting in

$$h = -\operatorname{grad} f(\hat{\boldsymbol{\xi}}_0) + \frac{\left[\operatorname{grad} g(\hat{\boldsymbol{\xi}}_0)^T \operatorname{grad} f(\hat{\boldsymbol{\xi}}_0)\right] \operatorname{grad} g(\hat{\boldsymbol{\xi}}_0)}{\left\|\operatorname{grad} g(\hat{\boldsymbol{\xi}}_0)\right\|^2}.$$

Since $\operatorname{grad} g(\hat{\boldsymbol{\xi}}_0) = 2\hat{\boldsymbol{\xi}}_0$, the step size h turns into

$$h = -\operatorname{grad} f(\hat{\boldsymbol{\xi}}_0) + \left[\hat{\boldsymbol{\xi}}_0^T \operatorname{grad} f(\hat{\boldsymbol{\xi}}_0)\right] \hat{\boldsymbol{\xi}}_0. \tag{5.4}$$

The updated value of the estimate $\hat{\boldsymbol{\xi}}$ is thus given as $\hat{\boldsymbol{\xi}}_1 = \hat{\boldsymbol{\xi}}_0 + \epsilon h / \|h\|$, with h as depicted in Equation (5.4). For a sufficiently small value of ϵ, the decrease of the objective function can always be found unless $\hat{\boldsymbol{\xi}}_0$ yields the local minimum. The value of ϵ is determined by a step-halving procedure. Starting with $\epsilon = 1$, the value of $\hat{\boldsymbol{\xi}}_1$ is preserved if $f(\hat{\boldsymbol{\xi}}_1) < f(\hat{\boldsymbol{\xi}}_0)$. If the objective function has not decreased,

the function value is calculated for

$$\hat{\boldsymbol{\xi}}_1 = \hat{\boldsymbol{\xi}}_0 + \left(\frac{1}{2}\right)^t \frac{h}{\|h\|}, \quad \text{with } t \in \{1, 2, \ldots, maxhalf\}.$$

Once a decrease occurs, the corresponding value of $\hat{\boldsymbol{\xi}}_1$ is maintained. If no decrease is reached after a given number of halving steps, the procedure reports the local minimum at $(\sigma, \hat{\boldsymbol{\xi}}_1)$.

Thus, the algorithm iterates as follows. Starting from the initial solution $\hat{\boldsymbol{\beta}}_0$, the values of $\hat{\boldsymbol{\xi}}_0$ and $\hat{\sigma}_0$ are determined, and the global solution is obtained by switching between the latter two minimization subproblems. It is crucial that the initial parameter $\hat{\boldsymbol{\beta}}_0$ is robust to ensure the robustness of the final solution (in case of convergence).

5.2. Cost-sensitive Bianco-Yohai estimator

The cost-sensitive version of the Bianco-Yohai estimator, as defined in Equation (4.22), must include the costs c_i in the algorithm presented in Section 5.1. For a parameter $\boldsymbol{\beta}$ defined as in Equation (5.1), the cost-sensitive optimization problem that adopts the minimization task in Equation (5.2) transforms to

$$(\hat{\sigma}, \hat{\boldsymbol{\xi}}) = \underset{(\sigma, \boldsymbol{\xi}) \in \mathbb{R}^+ \times S^{p-1}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} c_i \cdot \varphi\left(\boldsymbol{x}_i^T \frac{\boldsymbol{\xi}}{\sigma}, y_i\right), \tag{5.5}$$

where c_i denotes the cost for the imbalance learning of the observation i.

As shown in Section 5.1, the algorithm alternates between minimizing $\boldsymbol{\xi}$ and σ. The changes in the minimization task for $\boldsymbol{\xi}$ and σ include the following:

- Given the parameter $\boldsymbol{\xi}$, the univariate parameter σ is the solution of the adjusted optimization problem from Equation (5.5)
- If the parameter σ is known, the minimization problem under constraint from Equation (5.3) converts to

$$\min \widetilde{f}(\boldsymbol{\xi}) := \frac{1}{n} \sum_{i=1}^{n} c_i \cdot \varphi\left(\boldsymbol{x}_i^T \frac{\boldsymbol{\xi}}{\sigma}, y_i\right) \text{ under } g(\boldsymbol{\xi}) = \boldsymbol{\xi}^T \boldsymbol{\xi} - 1 = 0. \tag{5.6}$$

The gradient algorithm under constraint is performed likewise to the non-cost-sensitive case, with

$$\operatorname{grad} \widetilde{f}(\hat{\boldsymbol{\xi}}_0) = \operatorname{Diag}(\boldsymbol{c}) \operatorname{grad} f(\hat{\boldsymbol{\xi}}_0),$$

and vector $\boldsymbol{c}$ containing the cost c_i in the i-th entry.

As suggested in [1], the weighted ML estimator is used as the initial solution. The weighted ML estimator is the minimizer of the objective function

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} w_i \log \pi_{y_i}(\boldsymbol{x}_i, \boldsymbol{\beta}), \quad \text{with } w_i \in \{0, 1\}. \tag{5.7}$$

The weights w_i are assigned to the observations depending on their position in the input space. The algorithm allows for two different calculations of the leverage points:

1. using the MCD estimator from Section 4.1.1 and identifying the outliers as in Section 4.2.1
2. using the PCDist algorithm presented in Section 4.2.2, relying on the S-estimator presented in Section 4.1.1.

Both methods compute robust distances $RD_i = RD(\boldsymbol{x}_i)$ which are then compared to a cut-off value $\sqrt{\chi^2_{p,0.975}}$. The weight of the observation $\boldsymbol{x}_i$ is thus given as

$$w_i = \begin{cases} 1, & \text{if } RD(\boldsymbol{x}_i) \leq \sqrt{\chi^2_{p,0.975}} \\ 0, & \text{otherwise.} \end{cases} \tag{5.8}$$

The values in Equation (5.8) also serve as weights for the weighted Bianco-Yohai estimator defined in Equation (4.26).

The algorithm is implemented in the programming language `R` [15]. It is a modification of the existing functions `BYlogreg`, `glmrobBY`, and `glmrob` from the `R` package `robustbase` [16]. The relevant parts of the code described in this section can be found in Appendix A.

6. Evaluation

The performance of the implemented cost-sensitive Bianco-Yohai estimator in original and weighted form is analyzed in comparison to non-cost-sensitive and non-robust estimators. The comparison includes a total of six estimators, the logistic regression (LR), the Bianco-Yohai estimator and the weighted Bianco-Yohai (WBY) estimator, both in non-cost-sensitive and cost-sensitive form, respectively. The cost-sensitive form of the estimators is determined using the class proportions. Namely, for a data set with n observations, where n_0 denotes the number of observations with the output variable $y_i = 0$, and n_1 denotes the number of observations with $y_i = 1$, the cost of an observation i is defined as

$$c_i := \begin{cases} \frac{n_0}{n}, & \text{if } y_i = 1 \\ \frac{n_1}{n}, & \text{if } y_i = 0. \end{cases} \tag{6.1}$$

By defining the cost in this way, the majority class is simultaneously downweighted and the minority class is upweighted, depending on the imbalance proportion of the classes.

Section 6.1 provides the evaluation based on the estimated parameters using a simulation example with artificial data, while Section 6.2 analyzes the performance of the algorithms with the imbalanced data set used for credit scoring [17].

6.1. Simulation with the Artificial Data

The performance of the implemented cost-sensitive forms of the BY and the WBY estimators is determined based on parameter estimates in a simulation experiment. The simulation includes data configurations with different settings depending on the number of explanatory variables, the type of outliers and the imbalance proportion. The number of explanatory variables is set to $p = 2$ and $p = 10$, resulting in a low-dimensional and a higher-dimensional data set. The true values of the parameter $\boldsymbol{\beta}$ are initially determined and equal to $\boldsymbol{\beta} = (0, 2, 2)^T$ for $p = 2$ and $\boldsymbol{\beta} = (0, 1, \cdots, 1)^T \in \mathbb{R}^{11}$ for $p = 10$. Four different types of data sets are constructed to compare the estimators based on the outlier type:

I For $n = 5000$ observations, the explanatory variables are distributed according to a standard normal distribution $N(0, 1)$. The dependent variable y_i is generated

according to the following model equations:

$$y_i = \begin{cases} 0, & \text{if } \boldsymbol{x}_i^T \boldsymbol{\beta} + \epsilon_i \leq c \\ 1, & \text{if } \boldsymbol{x}_i^T \boldsymbol{\beta} + \epsilon_i > c, \end{cases} \tag{6.2}$$

with a positive constant c and error terms ϵ_i following a logistic distribution $logis(0, 1)$ with scale parameter $s = 1$. This data set serves as a baseline, and the other three configurations build on the corresponding data points.

II For a data set as described in configuration I, 10% of the observations are converted to bad leverage points. The observations are sampled from the majority class and their output value is not altered, but their coordinates in the input space are transformed as follows. The first $p-1$ entries of a sampled observation $\boldsymbol{x}_i$ are not altered and are therefore distributed according to a standard normal distribution, while the p-th entry is modified to satisfy the equation of the hyperplane

$$\boldsymbol{x}^T \boldsymbol{\beta} = c + 5\sqrt{p},$$

where c is a positive constant from Equation (6.2). Thus, the bad leverage points are added in parallel to the decision hyperplane.

III 10% of the data points in configuration I are mislabeled. The observations are sampled from the majority class with the output variable $y_i = 0$ and converted to the minority class by setting $y_i = 1$ to create vertical outliers.

IV This configuration represents the combination of the configurations II and III. 10% of the data points in configuration I are modified, of which 5% of the data points are converted to bad leverage points as in configuration II, and the other 5% are mislabeled as in configuration III. All modified observations are sampled from the majority class.

The constant c from Equation (6.2) is set to obtain different imbalance proportions, yielding approximately 20%, 10%, 5%, or 1% of the observations from the minority class in the data configuration I, also referred to as positives. The resulting data configurations are shown in Figure 6.1 for $p = 2$ explanatory variables and 20% of minority class observations.

The algorithm allows for two different leverage-point detection methods - the method based on the MCD estimator and the method using the PCDist algorithm. Leverage-point detection is required for the weighting process in the initial solution and the weights in the WBY estimator. The method for identifying leverage points does not affect the original BY estimator because it is only used for the initial solution to obtain a robust estimator, but further iterations of the algorithm converge to the final solution regardless of the initial parameter. Therefore, the simulation includes the WBY estimator computed using both leverage-point detection methods, whilst the BY estimator is computed employing only the MCD estimator.

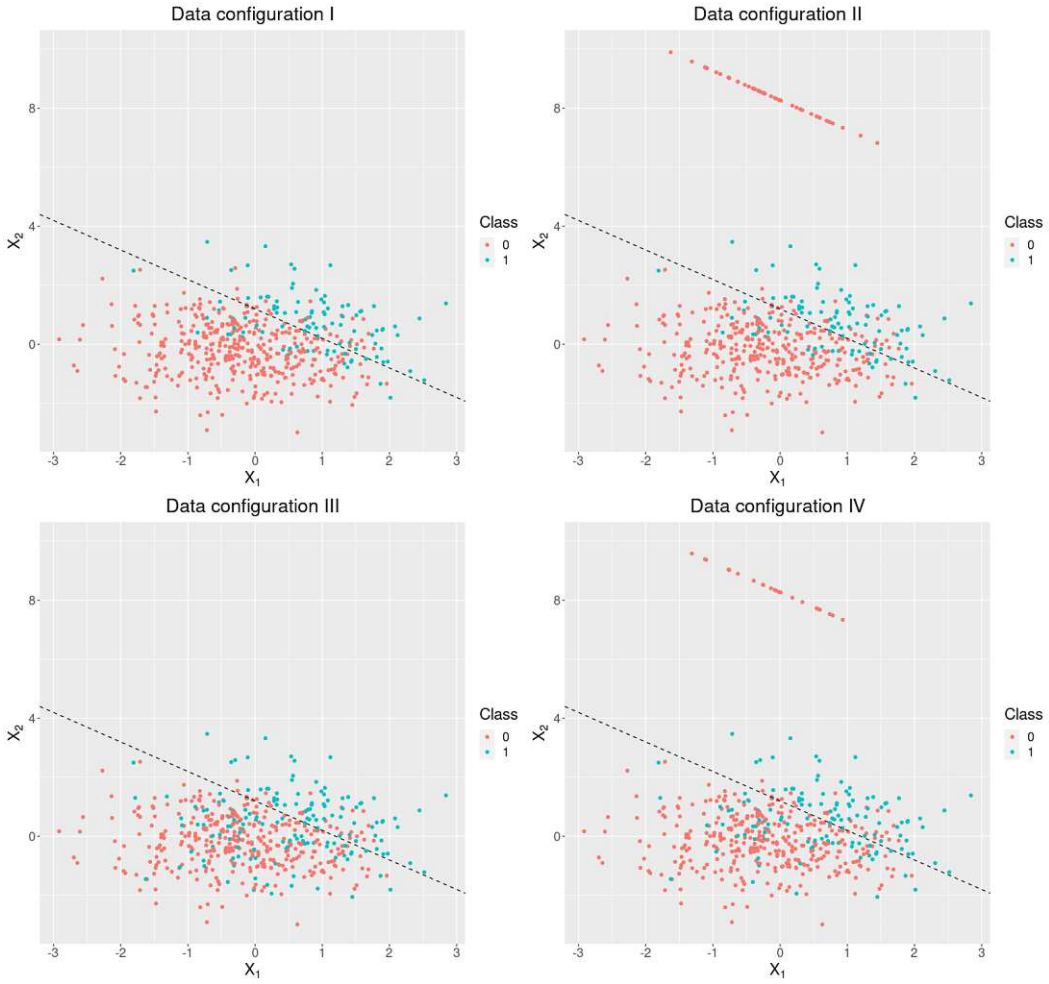


Figure 6.1.: An example of four data configurations. The color of the points indicates the class of the output variable, and the dashed line represents the decision boundary.

The simulation consists of $m = 500$ runs, with new data points randomly generated in each run. The data configurations I-IV are used to train the classifiers, and the classifier performance is evaluated based on bias and mean square error, comparing the estimated coefficients with the true parameter. Given a parameter estimate $\hat{\boldsymbol{\beta}}_i$ of the i-th simulation run, the bias and the mean square error (MSE) are calculated as

$$\text{Bias} = \left\| \frac{1}{m} \sum_{i=1}^{m} \hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta} \right\| \quad \text{and} \quad \text{MSE} = \frac{1}{m} \sum_{i=1}^{m} \left\| \hat{\boldsymbol{\beta}}_i - \boldsymbol{\beta} \right\|^2,$$

where $\|.\|$ denotes the Euclidean norm.

The resulting values of bias and MSE for the total of eight estimators and various imbalance proportions are shown in Table 6.1 and Table 6.2 for $p = 2$ and $p = 10$, respectively. All simulation settings – regardless of the number of explanatory variables, the data configuration, or the imbalance proportion – show the remarkable decrease in bias and MSE for all cost-sensitive estimators compared to their non-cost-sensitive forms. Thus, further analysis focuses on the performance of the cost-sensitive forms of the estimators. Unsurprisingly, all classifiers yield approximately the same values for bias and MSE for data configuration I. Adding leverage-points in data configuration II resulted in quite a different behaviour of the BY and the WBY estimator. The BY estimator does not lead to better performance compared to the logistic regression. In contrast, the WBY estimator shows better performance at 20% and 10% positives and worse performance at 5% and 1% positives. The values of bias and MSE for the data configuration III show no significant difference in classifier performance. In data configuration IV, both robust methods, the BY estimator and the WBY estimator, outperform the logistic regression model in all imbalance simulation settings. The WBY estimator yields the lowest values for bias and MSE.

In summary, based on the simulation results, the following can be stated. In general, the cost-sensitive algorithms significantly improve parameter estimation in an imbalance learning problem. The number of explanatory variables did not affect the behaviour of the robust estimators, except in the case of the WBY estimator, where the leverage detection methods perform differently in low and higher dimensional spaces. In a low-dimensional space, the WBY estimator computed by using the weights of the MCD estimator notably outperforms the WBY estimator using the weights of the PCDist algorithm. However, in a higher dimensional space, both leverage detection methods provide equivalent results. Moreover, the utilization of leverage detection methods on data without outliers degrades the performance of the model. In the case of pure vertical outliers, both robust estimators provide similar estimation compared to the logistic regression. However, in the presence of leverage points, the BY estimator provides more accurate or at least similar estimates compared to logistic regression, depending on data contamination. The WBY estimator yields notably better results, with an exception in settings with extremely imbalanced data, where the performance of the classifier significantly decreases. One possible reason for the poorer performance of the WBY estimator is the potential exclusion of positives after the leverage detection, which severely affects the methods in case of a very small number of positive observations.

Configuration	I		II		III		IV	
	Bias	MSE	Bias	MSE	Bias	MSE	Bias	MSE
20 % positives								
LR	2.808	7.684	2.779	7.508	1.913	3.562	2.548	6.313
LR. c.s.	1.425	1.995	2.104	4.306	1.707	2.834	2.175	4.600
BY	2.807	7.677	2.773	7.480	1.858	3.361	2.548	6.314
BY. c.s.	1.425	1.998	2.171	4.591	1.661	2.687	2.154	4.512
WBY *(MCD)*	2.807	7.679	2.675	6.976	1.860	3.369	1.969	3.777
WBY. c.s. *(MCD)*	1.425	1.999	1.292	1.649	1.654	2.664	1.317	1.693
WBY *(PCDist)*	2.913	8.277	2.757	7.418	1.813	3.204	2.049	4.098
WBY. c.s. *(PCDist)*	1.543	2.346	1.410	1.964	1.503	2.205	1.133	1.265
10 % positives								
LR	4.294	17.960	3.557	12.308	2.493	6.045	3.010	8.812
LR. c.s.	2.106	4.357	2.138	4.450	2.059	4.124	2.298	5.135
BY	4.295	17.975	3.530	12.124	2.454	5.859	3.017	8.852
BY. c.s.	2.106	4.360	2.140	4.494	2.051	4.093	2.324	5.257
WBY *(MCD)*	4.296	17.988	4.176	16.995	2.468	5.926	2.752	7.372
WBY. c.s. *(MCD)*	2.107	4.364	1.989	3.895	2.058	4.122	1.758	3.011
WBY *(PCDist)*	4.492	19.670	4.308	18.100	2.442	5.800	2.840	7.854
WBY. c.s. *(PCDist)*	2.329	5.350	2.208	4.819	1.999	3.888	1.606	2.522
5 % positives								
LR	5.542	29.934	4.373	18.609	2.883	8.083	3.368	11.028
LR. c.s.	2.624	6.803	2.214	4.786	2.314	5.209	2.444	5.810
BY	5.542	29.945	4.316	18.131	2.833	7.804	3.373	11.060
BY. c.s.	2.625	6.811	2.157	4.702	2.320	5.236	2.470	5.935
WBY *(MCD)*	5.548	30.018	5.435	28.811	2.875	8.038	3.169	9.773
WBY. c.s. *(MCD)*	2.628	6.828	2.516	6.271	2.344	5.344	2.101	4.298
WBY *(PCDist)*	5.891	33.856	5.631	30.939	2.820	7.733	3.196	9.940
WBY. c.s. *(PCDist)*	2.971	8.777	2.853	8.109	2.304	5.165	2.047	4.084
1 % positives								
LR	8.499	70.912	6.679	43.583	3.421	11.379	3.920	14.942
LR. c.s.	3.891	16.288	2.564	6.709	2.692	7.050	2.778	7.507
BY	8.501	71.117	6.556	42.023	3.409	11.303	3.910	14.868
BY. c.s.	3.992	17.512	2.611	8.347	2.693	7.056	2.762	7.420
WBY *(MCD)*	8.654	74.183	8.469	70.792	3.471	11.715	3.821	14.208
WBY. c.s. *(MCD)*	4.052	18.405	3.932	17.343	2.737	7.289	2.648	6.823
WBY *(PCDist)*	11.201	126.469	9.600	91.763	3.442	11.526	3.784	13.943
WBY. c.s. *(PCDist)*	5.881	47.092	5.685	43.795	2.720	7.197	2.624	6.705

Table 6.1.: The values of Bias and MSE obtained from 500 simulation runs for $\boldsymbol{p}=\boldsymbol{2}$. The columns represent different data configurations and the rows represent the three types of estimators, both cost-sensitive (c.s.) and non-cost-sensitive, using a different imbalance proportion. Leverage detection for the initial solution in the logistic regression and the BY estimator was performed using the MCD estimator, and the WBY estimator was calculated using both methods, indicated in parentheses.

Configuration	I		II		III		IV	
	Bias	MSE	Bias	MSE	Bias	MSE	Bias	MSE
20 % positives								
LR	3.057	9.204	2.801	7.721	2.151	4.547	2.332	5.347
LR c.s.	1.676	2.804	1.645	2.688	1.994	3.909	1.919	3.625
BY	3.063	9.244	2.785	7.638	2.069	4.208	2.330	5.338
BY c.s.	1.680	2.821	1.633	2.652	1.937	3.690	1.858	3.401
WBY *(MCD)*	3.064	9.251	2.933	8.480	2.070	4.215	2.138	4.503
WBY c.s. *(MCD)*	1.682	2.827	1.549	2.411	1.935	3.682	1.554	2.386
WBY *(PCDist)*	3.094	9.438	2.963	8.656	2.052	4.143	2.148	4.548
WBY c.s. *(PCDist)*	1.712	2.933	1.582	2.514	1.904	3.567	1.526	2.307
10 % positives								
LR	4.681	21.553	4.165	17.058	2.736	7.351	2.975	8.693
LR c.s.	2.501	6.238	2.167	4.676	2.368	5.510	2.238	4.931
BY	4.691	21.665	4.135	16.820	2.678	7.045	3.023	8.981
BY c.s.	2.511	6.295	2.153	4.624	2.358	5.466	2.205	4.787
WBY *(MCD)*	4.692	21.679	4.575	20.611	2.684	7.076	2.948	8.541
WBY c.s. *(MCD)*	2.513	6.311	2.397	5.757	2.360	5.476	2.047	4.131
WBY *(PCDist)*	4.743	22.159	4.613	20.966	2.671	7.008	2.963	8.634
WBY c.s. *(PCDist)*	2.564	6.578	2.443	5.982	2.341	5.386	2.020	4.028
5 % positives								
LR	6.016	35.613	5.357	28.235	3.124	9.584	3.392	11.298
LR c.s.	3.146	9.939	2.659	7.093	2.628	6.784	2.497	6.136
BY	6.047	36.015	5.317	27.829	3.059	9.189	3.428	11.545
BY c.s.	3.173	10.136	2.661	7.119	2.634	6.817	2.489	6.095
WBY *(MCD)*	6.050	36.064	5.938	34.752	3.071	9.263	3.366	11.136
WBY c.s. *(MCD)*	3.179	10.182	3.074	9.547	2.640	6.851	2.397	5.660
WBY *(PCDist)*	6.132	37.060	5.995	35.424	3.053	9.156	3.378	11.215
WBY c.s. *(PCDist)*	3.258	10.716	3.141	9.978	2.625	6.771	2.376	5.564
1 % positives								
LR	8.891	78.301	8.028	63.740	3.635	12.971	3.984	15.591
LR c.s.	5.153	29.301	4.189	19.089	2.983	8.741	2.923	8.409
BY	9.135	83.016	8.079	64.782	3.613	12.816	3.951	15.339
BY c.s.	6.030	46.699	4.729	28.966	2.985	8.754	2.927	8.430
WBY *(MCD)*	9.205	84.456	9.081	82.186	3.637	12.983	3.913	15.050
WBY c.s. *(MCD)*	6.203	50.584	6.198	52.685	2.999	8.834	2.886	8.202
WBY *(PCDist)*	9.451	89.169	9.226	84.924	3.617	12.846	3.899	14.945
WBY c.s. *(PCDist)*	6.551	56.143	6.639	65.142	2.987	8.766	2.872	8.125

Table 6.2.: The values of Bias and MSE obtained from 500 simulation runs for $\boldsymbol{p=10}$. The columns represent different data configurations and the rows represent the three types of estimators, both cost-sensitive (c.s.) and non-cost-sensitive, using a different imbalance proportion. Leverage detection for the initial solution in the logistic regression and the BY estimator was performed using the MCD estimator, and the WBY estimator was calculated using both methods, indicated in parentheses.

6.2. Example with the Credit Score Data

The *"Give me some credit"* data contains demographic and financial information of 150,000 borrowers used in *"Give Me Some Credit" Kaggle Competition* [17], [18]. The characteristics of the individuals are represented by ten explanatory variables. The output variable *SeriousDlqin2yrs* indicates whether a client will experience financial distress in the next two years. The distribution of the dependent variable *SeriousDlqin2yrs* in terms of absolute and relative frequencies is shown in Table 6.3, implying high class imbalance.

SeriousDlqin2yrs	Absolute Frequency	Relative Frequency (%)
0	111,912	93.051
1	8,357	6.949

Table 6.3.: Frequency of the output variable *SeriousDlqin2yrs*

The data variables are listed and described in Table 6.4, and the univariate distribution of explanatory variables for each output variable class is presented in Figure 6.2. The univariate distribution of most input variables exhibits strong skewness – the asymmetric shape of the density plots, which deviates from the symmetric bell curve. Therefore, modelling was performed with two forms of the data set, the original data and the log-transformed data. The logarithm transformation was performed on the variables with skewed distribution – it includes all input variables except for variables *age, NumberOfOpenCreditLinesAndLoans*, and *NumberOfDependents*. The resulting distribution of transformed explanatory variables for each output class is shown in Figure 6.3. The logarithm transformation rendered the distribution of the explanatory variables more symmetric and provided better separation between output classes in the transformed variable distribution.

Variable	Type	Description
SeriousDlqin2yrs	Binary	Person experienced 90 days past due delinquency or worse
MonthlyIncome	Numeric	Monthly income
DebtRatio	Numeric	Monthly debt payments, alimony, living costs divided by monthly gross income
Age	Numeric	Age of borrower in years
NumberOfDependents	Numeric	Number of dependents in family excluding themselves (spouse, children, etc.)
NumberOfOpenCreditLinesAndLoans	Numeric	Number of open loans (installment like car loan or mortgage) and lines of credit (e.g. credit cards)
NumberRealEstateLoansOrLines	Numeric	Number of mortgage and real estate loans including home equity lines of credit
RevolvingUtilizationOfUnsecuredLines	Numeric	Total balance on credit cards and personal lines of credit except real estate and no installment debt like car loans divided by the sum of credit limits
NumberOfTime30-59DaysPastDueNotWorse	Numeric	Number of times borrower has been 30-59 days past due but no worse in the last 2 years
NumberOfTime60-9DaysPastDueNotWorse	Numeric	Number of times borrower has been 60-89 days past due but no worse in the last 2 years
NumberOfTimes90DaysLate	Numeric	Number of times borrower has been 90 days or more past due

Table 6.4.: Description of the variables from the data set *"Give me some credit"* [18]

The observations used for modelling include only the instances without missing values in the input variables, resulting in a total number of 120,269 observations. Both forms of the data set, the original and the log-transformed, were divided into a training set and a test set with 75% and 25% of observations, respectively. The training set was used to model the six classifiers – the logistic regression, the BY estimator and the WBY estimator, both in cost-sensitive and non-cost-sensitive forms, respectively. For the cost-sensitive form, the costs defined in Equation (6.1) are considered. The leverage detection method used for the BY and the WBY estimator was performed using the PCDist algorithm, since the MCD estimator could not be computed. The performance of the classifiers was evaluated using the Gini index for the predictions of the test set.

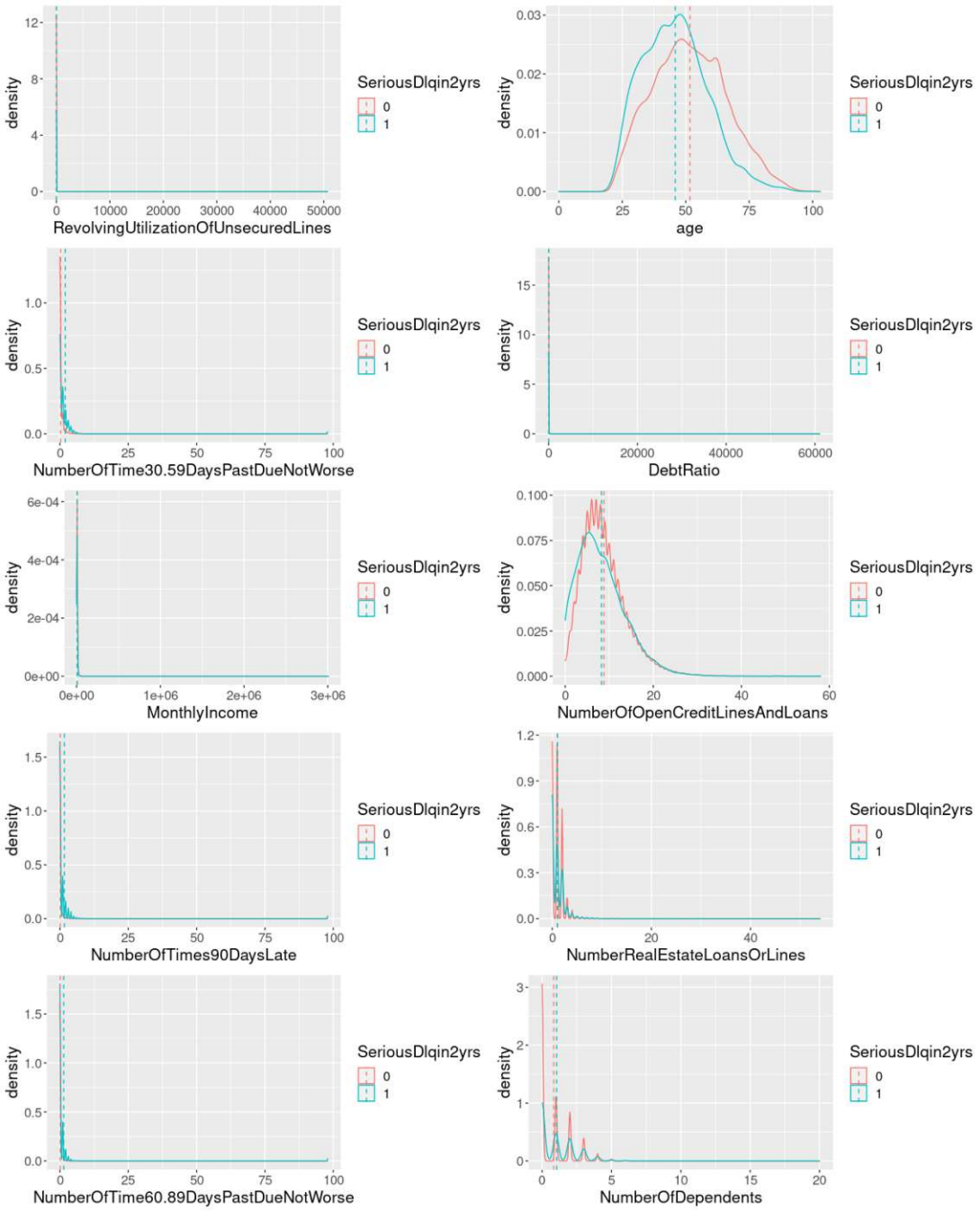


Figure 6.2.: Distribution of variables from the data set *"Give me some credit"* depending on the output variable class. The red color represents the majority class and the blue color represents the minority class. The mean values of the variables for each class are displayed with the dashed line.

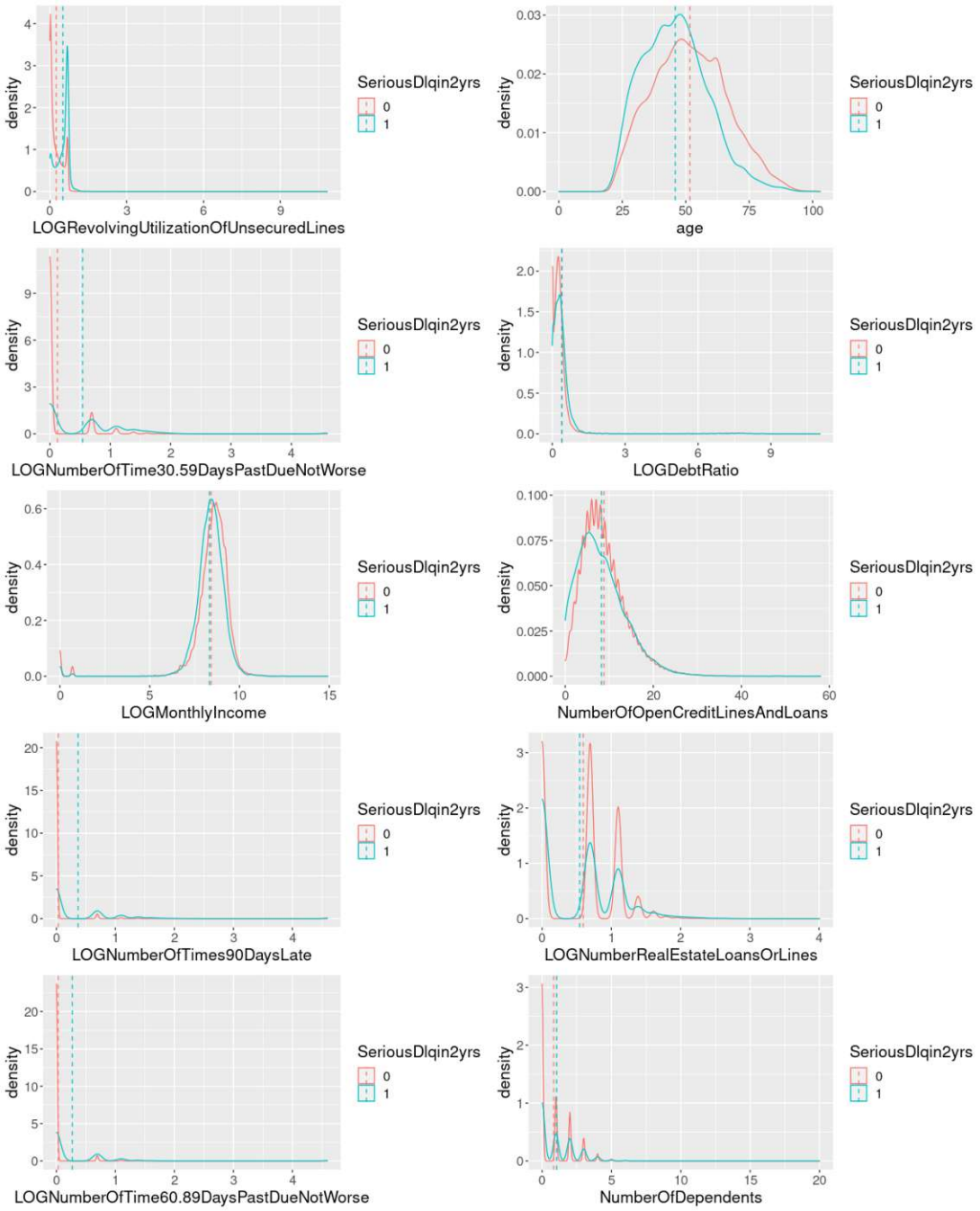


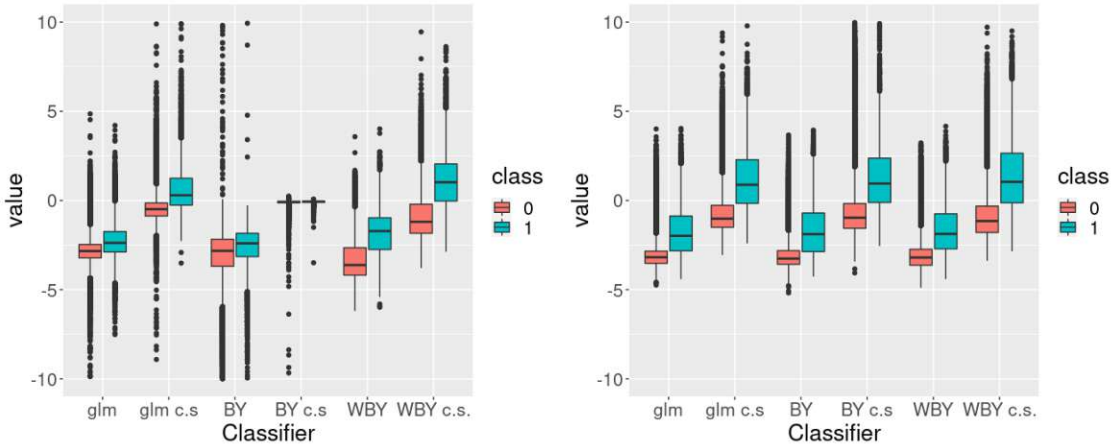
Figure 6.3.: Distribution of log-transformed variables from the data set *"Give me some credit"* depending on the output variable class. The red color represents the majority class and the blue color represents the minority class. The mean values of the variables for each class are displayed with the dashed line.

Data set	Original	Log-transformed
LR	0.399	0.661
LR c.s.	0.595	**0.695**
BY	0.237	0.638
BY c.s.	0.160	0.666
WBY	0.571	0.629
WBY c.s.	**0.639**	0.666

Table 6.5.: The values of the Gini index for the original and the log-transformed data set. Leverage detection for the initial solution and the WBY estimator is performed using the PCDist algorithm.

Table 6.5 shows the values of the Gini index for the original data and the log-transformed data. For the original data, the BY estimator has the worst performance, whilst the WBY offers the best results. Introducing imbalanced learning costs in the case of logistic regression and the WBY estimator notably improves the values of the Gini index. The large difference between the values of the Gini index for the BY and the WBY estimator indicates a considerable number of bad leverage points in the explanatory variable space. In contrast, the values of the Gini index for the log-transformed data are very similar for all six classifiers, with the cost-sensitive methods providing slightly better results. The classifier with the best performance is the cost-sensitive logistic regression. Similar values of the Gini index for the BY and the WBY estimator suggest the small number of bad leverage points in the space of explanatory variables for the log-transformed data. The possible absence of outliers explains the similar performance of all six estimators.

For a better understanding of the resulting models, the distribution of score values for each model is presented in the form of box-plots in Figure 6.4. The score values are grouped according to the class of the output variable, resulting in two box-plots per classifier, which is shown in Figure 6.4a for the original data and in Figure 6.4b for the log-transformed data. In general, a good classifier should result in a good distributional separation between the scores of the two output variable classes, ideally providing mostly negative scores for the majority class and mostly positive scores for the minority class, since such scores provide small values of deviances, as illustrated in Figure 4.4. In case of the original data, the box-plots of the scores obtained from the cost-sensitive logistic regression and the cost-sensitive WBY show favourable behaviour, while both the non-cost-sensitive and cost-sensitive forms of the Bianco-Yohai estimator fail to separate the distribution between the scores of the two classes. Concerning the log-transformed data, all classifiers lead to good distributional separation between classes, which explains the similar values of the Gini index. However, only the cost-sensitive methods achieve the desired property of obtaining mostly positive scores for the minority class and mostly negative scores for the majority class. Therefore, a good classification of the non-cost-sensitive classifiers would require the additional verification of an appropriate threshold for predicting the scores, different from a default value of zero.



(a) Original Data

(b) Log-transformed Data

Figure 6.4.: Box-plots of the score values $s_i = \boldsymbol{x}_i^T \hat{\boldsymbol{\beta}}$ resulting from the models with the original data (a) and the log-transformed data (b). The score values are grouped according to the class of the output variable.

The evaluation using the Gini index shows an application of the implemented classifiers in a real-world setting. Given a data set with an unbalanced distribution of the output variable and a similar distribution of explanatory variables across two output classes, the use of imbalance learning costs notably improves the value of the Gini index. In the additional presence of bad leverage points, a cost-sensitive robust classifier such as WBY can provide best performance, as shown in the case of the *"Give me some credit"* data set. When the distribution of explanatory variables provides separation between output classes, the cost-sensitive methods do not necessarily provide a large performance improvement in terms of the Gini index value, but they do lead to more accurate classifiers. In absence of outliers, the robust methods such as the BY or the WBY estimator provide performance results similar to the non-robust methods, as shown in the case of the *"Give me some credit"* data set with log-transformed explanatory variables.

7. Conclusions

With the immense increase regarding the utilization of statistical learning techniques in numerous disciplines, statistical methods need to be modified to ensure good statistical efficiency for various data structures. Logistic regression represents one of the first methods to model a binary response variable and hence it is widely used in many domains, mainly because the model interpretability and the additional class probabilities make human reasoning of the model straightforward. Many application cases of binary logistic regression employ data sets with an imbalance between the two classes of the output variable. Moreover, the data used for modeling often contain atypical observations that are separated from the majority of the data points and can greatly reduce the predictive power of the resulting classifier. Therefore, such data structures require a robust method suitable for imbalance learning problems.

This thesis proposes a cost-sensitive robust logistic regression model for imbalanced data sets based on the Bianco-Yohai estimator. The Bianco-Yohai estimator aims to replace the deviances of logistic regression with the bounded function of scores in the minimization objective for parameter estimation. In order to address the problem of imbalanced learning, the observation costs are included in the objective function. The implementation involves adapting the iterative algorithm introduced by Croux and Haesbroeck [1], which starts from a robust solution and converges to the Bianco-Yohai estimator. As a simple initial solution, the authors propose a weighted logistic regression, where the weights of the leverage points, computed based on the MCD estimator, are set to zero. The MCD estimator is affine equivariant and highly robust, but cannot be computed in practice quite often. Therefore, the implementation includes an additional method for detecting leverage points based on the S-estimator, the PCDist algorithm. The weights based on the leverage points are also used to obtain the weighted Bianco-Yohai estimator. Introducing the PCDist algorithm for leverage detection significantly increases the data domain in which the Bianco-Yohai estimator is applicable.

The obtained cost-sensitive forms of the Bianco-Yohai estimator, in the weighted and original versions, are compared with their non-cost-sensitive forms and the logistic regression as the corresponding non-robust method. The evaluation was based on simulation experiments and an imbalanced data set used for credit scoring. In the simulation example, the parameter estimates are compared with the true value of the parameter at different settings for the number of explanatory variables, outlier types and imbalance proportions. The simulation results show that including imbalanced costs remarkably improves the performance of the Bianco-Yohai estimator in both the original and weighted

versions. Moreover, compared to logistic regression, the Bianco-Yohai estimator generally provides a better parameter estimate when the data contain bad leverage points, but does not improve the estimate in the case of vertical outliers. In the example of the credit score data set, the performance of the estimators is analyzed using the Gini index. Based on the values of the Gini index, the cost-sensitive robust regression provides the best estimate for imbalanced data with outliers compared to the non-robust and non-cost-sensitive methods. Thus, the cost-sensitive form of the Bianco-Yohai estimator, in both its original and weighted versions, provides a statistically reliable classifier for imbalanced data that maintains its performance in the presence of outliers.

A. The code for the Algorithm Implementation

The full implementation code can be found here: `https://github.com/sanjapriselac/Cost-sensitive-Robust-Logistic-Regression-in-R`

The following code snippet presents the crucial implementation steps described in Chapter 5.

```
## x0 - the data matrix X with the values of the input variables
## y - the output variable
## initwml (TRUE or FALSE) - the initial solution as the weighted ML
## weights - the costs for the imbalance learning

## Computation of the initial value of the optimization process
gstart <-
  if(initwml) {
    if (outmethod == "mcd") {
      mcd <- covMcd(x0, alpha=0.75, tolSolve = 1e-20)
      D <- mahalanobis(mcd$X, mcd$center, mcd$cov)
      vc  <- qchisq(0.975, p-1)
      wrd <- D <= vc

      if (method == "WBY") {
        wby <- as.numeric(wrd)
      }
    } else {
      outpcd <- OutlierPCDist(x0, grouping = as.factor(y))
      wrd <- as.logical(outpcd@flag)

      if (method == "WBY") {
        wby <- outpcd@flag
      }
    }
    glm.fit(x[wrd,], y[wrd], weights = weights[wrd], family=family)$coef
  } else {
    if (method == "WBY") {
      if (outmethod == "mcd") {
        mcd <- covMcd(x0, alpha=0.75, tolSolve = 1e-40) #SP commented
        D <-  mahalanobis(mcd$X, mcd$center, mcd$cov)
        vc  <- qchisq(0.975, p-1)
        wby <- as.numeric(D <= vc)
      } else {
        outpcd <- OutlierPCDist(x0, grouping = as.factor(y))
        wby <- outpcd@flag
      }

    }
    glm.fit(x, y, weights = weights, family=family)$coef
  }

if (method == "WBY") {
  weights <- weights * wby
}

sigma1 <- 1/sqrt(sum(gstart^2))
xistart <- gstart*sigma1
stscores <- x %*% xistart

## Initial value for the objective function
oldobj <- mean(phiBY3(stscores/sigma1, y, const) * weights)


```

```
  converged <- FALSE
  kstep <- 1L

  while(kstep < kmax && !converged)
  {
    unisig <- function(sigma) mean(phiBY3(stscores/sigma, y, const) * weights)
    optimsig <- optimize(unisig, interval = c(0, 10^10))
    if(trace.lev) cat(sprintf("k=%2d, s1=%12.8g: => new s1= %12.8g",
                              kstep, sigma1, optimsig$minimum))
    sigma1 <- optimsig$minimum

    if(sigma1 < sigma.min) {
      if(trace.lev) cat("\n")
      warning(gettextf("Implosion: sigma1=%g became too small", sigma1))
      kstep <- kmax #-> *no* convergence
    } else {
      scores <- stscores/sigma1
      newobj <- mean(phiBY3(scores, y,const) * weights)
      oldobj <- newobj
      grad.BY <- colMeans(((derphiBY3(scores,y,const)*weights) %*% matrix(1,ncol=p))*x)
      h <- -grad.BY + as.numeric(grad.BY %*% xistart) *xistart
      finalstep <- h/sqrt(sum(h^2))

      if(trace.lev) {
        if(trace.lev >= 2) cat(sprintf(", obj=%12.9g: ", oldobj))
        cat("\n")
      }

      xi1 <- xistart+finalstep
      xi1 <- xi1/sum(xi1^2)
      scores1 <- (x %*% xi1)/sigma1
      newobj <- mean(phiBY3(scores1,y,const) * weights)

      ## If 'newobj' is not better, try taking a smaller step size:
      hstep <- 1.
      jhalf <- 1L
      while(jhalf <= maxhalf & newobj > oldobj)
      {
        hstep <- hstep/2
        xi1 <- xistart+finalstep*hstep
        xi1 <- xi1/sqrt(sum(xi1^2))
        scores1 <- x %*% xi1/sigma1
        newobj <- mean(phiBY3(scores1,y,const) * weights)
        if(trace.lev >= 2)
          cat(sprintf("  jh=%2d, hstep=%13.8g => new obj=%13.9g\n",
                      jhalf, hstep, newobj))
        jhalf <- jhalf+1L
      }

      converged <-
        not.improved <- (jhalf > maxhalf && newobj > oldobj)
      if(not.improved) {
        ## newobj is "worse" and step halving did not improve
        message("Convergence Achieved")
      } else {
        jhalf <- 1L
        xistart <- xi1
        oldobj <- newobj
        stscores <- x %*% xi1
        kstep <- kstep+1L
      }
    }
  } ## while( kstep )

  if(kstep == kmax) {
    warning("No convergence in ", kstep, " steps.")
  }
  gammaest <- xistart/sigma1  # SP the estimator
  V <- vcovBY3(x, y, const, estim=gammaest, addIntercept=FALSE)
  list(convergence=TRUE, objective=oldobj, coefficients=gammaest,
        cov = V, sterror = sqrt(diag(V)),
        iter = kstep)
```

List of Figures

List of Tables

Bibliography

[1] C. Croux and G. Haesbroeck. "Implementing the Bianco and Yohai estimator for logistic regression". In: *Computational Statistics & Data Analysis* 44.1 (2003). Special Issue in Honour of Stan Azen: a Birthday Celebration, pp. 273–295. ISSN: 0167-9473. DOI: 10.1016/S0167-9473(03)00042-2.

[2] T. Hastie, R. Tibshirani, and J. H. Friedman. "Linear Methods for Classification". In: *The elements of statistical learning: data mining, inference, and prediction*. 2nd ed. Springer series in statistics. New York, NY: Springer, 2009, pp. 101–138. ISBN: 978-0-387-84858-7. DOI: 10.1007/978-0-387-84858-7.

[3] P. K. Dunn and G. K. Smyth. "Chapter 4: Beyond Linear Regression: The Method of Maximum Likelihood". In: *Generalized Linear Models With Examples in R*. New York, NY: Springer New York, 2018, pp. 165–209. ISBN: 9781441901187. DOI: 10.1007/978-1-4419-0118-7_4.

[4] H. He and Y. Ma. "Introduction". In: *Imbalanced Learning: Foundations, Algorithms, and Applications*. New Jersey, NJ: Wiley, July 2013, pp. 1–12. ISBN: 978-1-118-07462-6. DOI: 10.1002/9781118646106.ch1.

[5] C. Sammut and G. I. Webb, eds. *Encyclopedia of machine learning and data mining*. Second edition. Springer reference. New York, NY: Springer, 2017. ISBN: 9781489976857. DOI: 10.1007/978-1-4899-7687-1.

[6] H. He and Y. Ma. "Imbalanced Datasets: From Sampling to Classifiers". In: *Imbalanced Learning: Foundations, Algorithms, and Applications*. New Jersey, NJ: Wiley, July 2013, pp. 43–59. ISBN: 978-1-118-07462-6. DOI: 10.1002/9781118646106.ch3.

[7] C. Gini. "On the measurement of concentration and variability of characters". In: *Metron - International Journal of Statistics* LXIII.1 (2005), pp. 1–38.

[8] E. Schechtman and G. Schechtman. "The relationship between Gini terminology and the ROC curve". In: *Metron - International Journal of Statistics* 77.3 (Dec. 2019), pp. 171–178. DOI: 10.1007/s40300-019-00160-.

[9] R. A. Maronna, R. D. Martin, and V. J. Yohai. *Robust Statistics: Theory and Methods*. en. 1st ed. Wiley Series in Probability and Statistics. New Jersey, NJ: Wiley, Mar. 2006. ISBN: 9780470010945. DOI: 10.1002/0470010940.

[10] M. Hubert, M. Debruyne, and P. J. Rousseeuw. "Minimum covariance determinant and extensions". In: *WIREs Computational Statistics* 10.3 (Dec. 2017). ISSN: 1939-0068. DOI: 10.1002/wics.1421.

[11] A. D. Shieh and Y. S. Hung. "Detecting Outlier Samples in Microarray Data". In: *Statistical Applications in Genetics and Molecular Biology* 8.1 (Jan. 2009), pp. 1–24. ISSN: 1544-6115. DOI: 10.2202/1544-6115.1426.

[12] M. Zhu and A. Ghodsi. "Automatic dimensionality selection from the scree plot via the use of profile likelihood". In: *Computational Statistics & Data Analysis* 51 (Feb. 2006), pp. 918–930. DOI: 10.1016/j.csda.2005.09.010.

[13] P. Filzmoser and V. Todorov. "Robust tools for the imperfect world". In: *Information Sciences* 245 (2013). Statistics with Imperfect Data, pp. 4–20. ISSN: 0020-0255. DOI: 10.1016/j.ins.2012.10.017.

[14] A. M. Bianco and V. J. Yohai. "Robust Estimation in the Logistic Regression Model". In: *Robust Statistics, Data Analysis, and Computer Intensive Methods*. Vol. 109. New York, NY: Springer New York, 1996, pp. 17–34. ISBN: 9781461223801. DOI: 10.1007/978-1-4612-2380-1_2.

[15] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2020. URL: https://www.R-project.org/.

[16] M. Maechler, P. Rousseeuw, C. Croux, V. Todorov, A. Ruckstuhl, M. Salibian-Barrera, T. Verbeke, M. Koller, E. L. T. Conceicao, and M. Anna di Palma. *robustbase: Basic Robust Statistics*. R package version 0.93-8. 2021. URL: http://robustbase.r-forge.r-project.org/.

[17] *Give Me Some Credit Data Set*. https://www.kaggle.com/c/GiveMeSomeCredit/data. Accessed: 2021-09-14.

[18] L. Zhang, H. Ray, J. Priestley, and S. Tan. "A descriptive study of variable discretization and cost-sensitive logistic regression on imbalanced credit data". In: *Journal of Applied Statistics* 47.3 (July 2019), pp. 568–581. ISSN: 1360-0532. DOI: 10.1080/02664763.2019.1643829.