ERASMUS UNIVERSITEIT ROTTERDAM

Erasmus School of Economics
ECONOMETRICS & MANAGEMENT SCIENCE

Master Thesis
Business Analytics & Quantitative Marketing

# Evaluating Classical, Robust and Non-parametric Binary Classification in Contaminated Datasets

## A Comparative Simulation Study

*Author:*
Gyovana Ferreira da Silva

*Student number:*
545086

*Supervisor:*
Prof. Dr. Mikhail Zhelonkin

*Second assessor:*
Prof. Dr. Jens Klooster

October 17, 2021

The content of this thesis is the author's sole responsibility. It does not reflect the view of the supervisor, second assessor, Erasmus School of Economics, or Erasmus University Rotterdam.

# Abstract

When predicting and classifying binary data, parametric methods are often used for their ease of interpretation and evaluation; however, these models are only optimal when distributional assumptions of a dataset are satisfied. When handling not normally distributed data, non-parametric machine learning methods are often turned to, even though these methods can be challenging to interpret and computationally challenging. Robust parametric methods may offer a solution. The thesis investigates the predictive performance of non-robust parametric, robust parametric, and machine learning methods. The study compares the different methods' performance based on classification accuracy, with and without feature selection, using the effect of outliers to appraise the methods. For classification accuracy, the study compares the Maximum Likelihood estimator (ML) of the non-robust parametric method logistic regression with two robust parametric estimators: Mallows Quasi-Likelihood (MQL) and Conditionally Unbiased Bounded-Influence (Cubif). The study then investigates how these parametric methods perform compared to machine-learned methods using supervised learning classifiers Naïve Bayes (NB) and Support Vector Classifiers (SVC). The forward-stepwise selection was implemented using Bayesian information criteria (BIC) and its robust version (RBIC) to appraise feature selection. The results show that the parametric methods MQL and Cubif outperform ML under all contaminated datasets. SVC reveals unreliable results in small datasets while NB proves robust. Machine learning classifiers perform slightly worse for data with low contamination levels than parametric ones; however, they perform well in datasets with high contamination. The study confirms that RBIC outperforms BIC in all contaminated datasets. In the setting of no contamination, the performance is similar or equal.

Key Words: Robust Classification, M-Estimator, Mallows Quasi-Likelihood, Cubif, Naïve Bayes, Support Vector Classifier

# Acknowledgements

First and foremost, I want to thank God. For his showers of blessings, guiding me through the way, and showing everything I needed to see to complete this research.

I want to express my profound and sincere gratitude to my supervisor Dr. Mikhail Zhelonkin, Professor at Erasmus School of Economics, Department of Econometrics, for providing invaluable guidance through this thesis, always advising when I was lost. It was a great pleasure and honor to work under his guidance.

I am incredibly grateful for my boyfriend Martin and my dearest friend Floor. Who made it possible for me to study in a calm and safe environment allowing me to thrive. I can not thank you both enough.

I thank my family, particularly my parents, for their love and sacrifices for educating and preparing me for my future. I thank my brother, grandmother, godmother, and dear friend Nadir for their caring, prayers, and love.

I also want to express my gratitude to my boyfriend's family, which provided me a friendly and loving family environment in foreigners land.

Ultimately I would like to thank my friends Camilla, Ugis, Rashed, Jędrzej, André, Esther and family, Meian, Julia, Heloisa, Mariana, William, Dhiego, Rafaela, Leandro, Abelardo, Filippe, Ana Paula, Patricia, Jussara, Ligiane, Rodolfo for all their support throughout life. I would never have arrived here without you all.

# Contents

# 1 Introduction

Supervised classification is one of the most promising areas of research implemented by machine learning. Consequently, many supervised classification techniques have been developed and studied in the past few decades (Kotsiantis et al., 2006). In the field of supervised classification, classifiers construct decision boundaries aiming to assign observation into classes. For a two-class problem in a p-dimensional feature space, these methods model the decision boundary as a hyperplane (Hastie et al., 2017).

When it comes to predicting and classifying binary data, the typical way to proceed is to use parametric modeling, which is simple to evaluate and interpret, or machine learning methods, which are not easy to interpret and can be computationally challenging. The parametric methods, such as logistic regression, will be optimal if a dataset's distributional assumptions are satisfied; however, logistic regression will fail to produce accurate predictions and can give biased estimates when those assumptions are not met (Victoria-Feser, 2002).

In the era of big data, it is rare to encounter normally distributed data, making it dangerous to rely on parametric methods solely. Practice reveals that nearly all real data hold at least one observation in their feature space that deviates dramatically from what is expected; those are known as outliers (Ritter and Gallegos, 1997). There is no precise and unified concept of an outlier to date; however, two main ideas form its conceptualization:

1. Outliers are "spurious" observations that do not follow any statistical law. In such cases, there can exist datasets consisting entirely of such outliers, which would make these data unmanageable for statistics.

2. The second idea assumes that in many circumstances, the distributional model of the data belongs to a given parametric family like the Gaussian distribution, which will be of interest in this study. This concept suggests that the observations follow some statistical law; however, in this case, they seem to appear more often than the presumed distribution allows.

This second idea can be further stratified into two different scenarios:

(A) In the first scenario, the assumption is that there is only one or very few outliers in the data relative to the total number of observations. In this scenario, these outliers might be isolated events and most likely do not concern an unsuitable distributional assumption on the data.

(B) If there are many outliers, i.e., 10%, they must be associated with a wrong distribution choice. This idea is often associated with "heavy tails." When an analysis concerning such outliers is reproduced, the outliers will resemble similarities in number and characteristics.

In cases (1) and (2A), the easiest way a statistician can proceed is by either identifying the outliers and rejecting them to repair the properties of the data or by employing robust parameter estimation methods. These methods attempt to remove or at least reduce outliers' influence, modeling the regular observations only (Ritter and Gallegos, 1997). Here we choose the latter, robust methods.

Given the presence of outliers resulting in contaminated data, two main perspectives to consider in high-dimensional space supervised classification are classification accuracy and feature selection. In this paper, we appraise the effect of outliers in both aspects. The literature often suggests that the presence of outliers can influence the performance of a classifier, but only a few studies verify such claims. This is not the case in the regression setting, where many studies showcase the problems that may arise when the data hold outliers (Acuña and Rodriguez, 2004).

The primary goal of this research is to investigate and compare the classification performance from classical logistic regression with robust logistic regression methods. Moreover, the study investigates how those robust methods perform compared to supervised learning classifiers, especially in the presence of data contaminated by outliers.

The paper aims to answer the following research questions:

- *How do robust estimators (in this study, Mallows Quasi-likelihood, and Conditionally Unbiased Bounded-Influence) perform in the classification framework in comparison with the non-robust Maximum Likelihood estimator and with the supervised learning classifiers (in this study, Support Vector Classifier and Naïve Bayes) in the presence of data contaminated by outliers?*

We investigate three main frameworks: **(A)** without feature selection, **(B1)** with feature selection using the well-studied Bayesian Information Criterion (BIC) and finally **(B2)** with feature selection using its robust version RBIC developed by Machado (1993).

To evaluate those frameworks, a Monte-Carlo simulation was constructed for 16 different scenarios, including four proportions of contaminated datasets, namely 0%, 1%, 5%, and 10%. The research investigates how the methods behave when at least 30% of the explanatory variables are not significantly different from zero.

Moreover, the study evaluates performance improvement before and after feature selection contrasting with machine learning methods that do not require dimension reduction. Later this thesis verifies its findings by examining the public binary data from Dr. Walberg of the University of Wisconsin Hospital regarding breast cancer tumors (Wolberg and Mangasarian, 1990; Wolberg et al., 1994).

The paper is organized as follows: Section 1 introduces the main problems with classification when encountering outliers in the data. Section 2 comprises a literature review that contains recent studies in classical and robust binary classification. Section 3 is dedicated to the theory and methods. We introduce the Monte-Carlo simulation design and implementation in Section 4. The hypotheses formulated to answer the research question are presented in Section 5. The real-world data is described in Section 6. Results, discussion, and comparative analysis are displayed in Section 7. Finally, Section 8 is devoted to conclusions and final remarks.

# 2 Literature

Logistic regression is one of the most traditional and applied methods in binary datasets. It uses maximum-likelihood (ML) for parameter estimation, optimal when underlying assumptions about the data hold. However, it becomes unreliable when the data fails to meet those assumptions (Victoria-Feser, 2002).

Many estimators have been proposed in an attempt to make logistic regression more robust. Huber (1981) proposed a class of estimators called M-Estimators which were a generalization of the maximum-likelihood (ML) estimator. The estimator operates similarly to the standard least-squares (LS) method or the LS estimator. Like the LS estimator, the M-estimator tries to minimize the sum of residuals. However, the M-estimators replace the squared function with a robust function, attempting to reduce the effect of outliers. The M-estimator class became popular not only due to its robustness but because it worked well with generalized linear models (GLM), a resilient generalization of the traditional linear regression presented by McCullagh and Nelder (1999). GLMs allow for the distribution of the response variable to assume many forms, rather than being restricted to the Gaussian form, such as it is in the normal linear regression context.

In 1989, Künsch et al. presented a subclass of the class of M-estimators called Conditionally Unbiased Bounded-Influence (Cubif). Their estimator was based on the work of Stefanski et al. (1986) which presented optimal bounded score functions for parameter estimation in GLMs. The Cubif estimator is defined by restricting the score function to be conditionally unbiased (conditional Fisher consistent) given the independent variables.

Cantoni and Ronchetti presented another estimator called Mallows Quasi-Likelihood (MQL) in 2001. This estimator unifies the concept of Mallows-Type with the class of M-estimators. In a sense, the estimator belongs to the M-estimator class proposed by Huber (1981), however, its influence of deviations on $y$ (dependent variable) and $x$ (independent variables) are separately bounded as in the Mallows-Type estimator proposed by Jongh et al. (1988) (Hampel, 1974; Hampel et al., 1986). Furthermore, the estimator is built on the notion of quasi-likelihood functions and robust deviances (Wedderburn, 1974; McCullagh and Nelder, 1989; Preisser and Qaqish, 1999; Heyde, 2008)

Many papers evaluate the robustness of both estimators regarding parameters estimation; however, only a few evaluate their classification performance. Ahmad et al. (2010) and Habshah and Syaiba (2012) presented a comparative simulation study contrasting ML estimator with the robust estimators Cubif by Künsch et al. (1989), Mallows-Type by Jongh et al. (1988) and Bianco and Yohai by Bianco and Yohai (1996). Their goal is predictive performance in the first paper, whereas parameter estimation in the second paper.

While many studies evaluate the estimator MQL for Poisson regression setting, few explore MQL in logistic regression. Kitromilidou and Fokianos (2015) present an analysis for robust log-linear Poisson autoregressions in the context of outliers present in the data employing the estimator MQL. Most recently, Reda Abonazel and Mohamed Saber (2020) presented a Monte Carlo simulation for Poisson regression, where they show that the MQL estimator outperforms the well-known weighted maximum likelihood in the presence of outliers.

Machine learning is recognized as a field of artificial intelligence, and its goal is to study and improve algorithms that allow computers to enhance their performances with data. The process comprises analyzing prior experiences to find reasonable and practical regularities and patterns that a human eye might neglect (Kotsiantis et al., 2006).

Two of the most popular machine learning methods in binary classification are Support Vector Classifiers (SVC) and Naïve Bayes (NB). Their popularity is likely due to their similarity to logistic regression (Hastie et al., 2017).

Golpour et al. (2020) present a comparison of SVC, NB, and logistic regression in the binary classification context, diagnosing cardiovascular diseases. They presented that all models performed the same accuracy; however, the NB model outperformed logistic regression and SVC in parsimony and simplicity. Colas and Brazdil (2006) investigate the performance of SVC to K-Nearest Neighbors (KKN) and NB on binary classification tasks. Their results show that all the classifiers achieved comparable performance on most problems; however, SVC outperforms despite good overall performance.

The first goal of this study is to evaluate predictive performance in a logistic regression setting. This research utilizes the well-studied robust estimator Cubif and the estimator MQL mainly studied in the Poisson setting. It compares their predictive performance with machine learning methods (SVC and NB) considered to be non-parametric techniques, which are generally known to outperform parametric ones when the assumptions about the data are not met (Bhattacharjee and Chaudhuri, 2020). SVC and NB were chosen for this study due to their popularity and similarities with logistic regression.

The secondary goal of this study is to investigate how the robust estimators perform in the feature selection framework by employing two penalized criteria (BIC and Robust BIC).

Deng (1999) explains that the feature selection technique selects a subset of features from the complete input features set to predict the response variable. The technique aims to obtain an accuracy performance equivalent to or higher than the complete input dataset, reducing the computational cost. The author divides the technique into three main groups; the filters methods (i.e., correlation, $\chi^2$ test), the wrapper methods (i.e., forward selection, backward elimination, stepwise selection), and the embedded methods (i.e., LASSO, elastic net, ridge regression). In particular, the wrapper methods perform a sequential model selection and use penalized criteria such as Akaike's Information Criteria (AIC) by Akaike (1998) and Bayesian Information Criteria (BIC) by Schwarz (1978). Those criteria are well-known and studied, and the latter will be of interest to this research.

AIC and BIC become biased when the underlying distributional assumption does not hold. Therefore many papers have been dedicated to investigating robust alternatives for penalized criteria (Field and Ronchetti, 1985; Hampel, 1983; Machado, 1993; Ronchetti and Staudte, 1994; Ronchetti et al., 1997; Qian and Künsch, 1998).

Machado (1993) presents a robust version of BIC, where instead of squared deviances, the criterion uses Huberized deviances in an attempt to reduce the outliers' influence. Furthermore, the criterion is mainly designed for M-Estimators, making it attractive for the entire class of M-Estimators combined with GLMs.

Ultimately, this study investigates how the parametric methods perform in contaminated datasets in the feature selection context. We contrast two main frameworks **B1** and **B2**, which employ the criteria BIC and RBIC, respectively.

# 3 Methodology

This section discusses the methods and techniques involved in this research paper. It starts by introducing logistic regression with Generalized Linear Models (GLM). We discuss the class of M-Estimators and the three parametric estimators in Section 3.2. We first introduce the well-known Maximum Likelihood (ML) estimator and then both robust estimators Mallows Quasi-Likelihood (MQL) and Conditionally Unbiased Bounded-Influence (Cubif).

We present the machine learning classifiers, Naïve Bayes (NB) and Support Vector Classifiers (SVC), in Section 3.3. An explanation of the performance measures investigated in this study is approached in Section 3.4. Finally, feature selection and the criteria Bayesian Information Criteria (BIC) and Robust Bayesian Information Criteria (RBIC) are discussed in Section 3.5.

## 3.1 Logistic Regression with Generalized Linear Models (GLM)

Generalized linear models (GLM) are a resilient generalization of conventional linear regression that enables the dependent variable to assume a discrete distributional form instead of the traditional Gaussian distribution. The linear model assumes that $E(Y \mid X)$ is equal to a linear combination $X^T\beta$.

Nelder and Wedderburn (1972) and McCullagh and Nelder (1999) showed that the GLM is determined by two major components: the distribution of the dependent variable and the link function. The first shall satisfy the requirement of being a member of the Exponential Dispersion Model family (EDMs). If that is the case, then the class of models that connects $\mu = E(Y \mid X)$ to $\eta = G(\mu) = X^T\beta$ can be treated in a unified way. This is achieved by having a function that relates the first to the latter or equivalently,

$$E(Y \mid X) = G^{-1}(X^T\beta). \tag{1}$$

In our simulation, the response variable has two possible outcomes, making it a Bernoulli response. This distribution belongs to the EDM family as demonstrated by Mueller (2004), and hence the first requirement for the GLM is fulfilled.

The second component required by the model, is the link function. Which for the Bernoulli case is, $\eta = \log\left(\frac{\mu}{1-\mu}\right)$.

The Bernoulli GLM has the form of,

$$\text{GLM(EDM; Link Function)} = \begin{cases} y_i \sim Bin(\mu_i, 1) & \text{(random component),} \\ \log\left(\frac{\mu}{1-\mu}\right) = \beta_0 + \beta_1 x_i & \text{(systematic component).} \end{cases} \tag{2}$$

Looking at the systematic component in Equation 2, we notice that the Bernoulli GLM is equivalent to the logistic regression.

The parameters $\hat{\beta}$'s are estimated by Fisher scoring, using iteratively reweighted least-squares (IRLS) algorithm presented at Algorithm 1. The method involves repeatedly fitting a weighted linear regression of a working response variable on the covariates; each regression uses a new value of the parameter estimates, which in turn give new working responses and weights, and the process is iterated (Hastie et al., 2017; Dunn and Smyth, 2018).

**Algorithm 1:** IRLS algorithm

```
Result: $\hat{\beta} = (\hat{\beta}_1, ..., \hat{\beta}_k)$
Initialization: Set all $\hat{\beta}_j^{(0)} = 0$, for$j = 0, 1, \cdots, k$. Calculate $\eta_i^{(0)} = \sum_j x_{ij}\hat{\beta}_j^{(0)}$ and $\hat{\mu}_j^{(0)} = \hat{\pi}_j^{(0)} = \frac{\exp(\eta_i^{(0)})}{1+\exp(\eta_i^{(0)})}$.
while $\left|\eta_i^{(t)} - \eta_i^{(t-1)}\right| < \epsilon_\eta \; or \; \left|\pi_i^{(t)} - \pi_i^{(t-1)}\right| < \epsilon_\pi$ do
    for $i = 1, 2, \cdots, n$ do
        Set, $W^{(i)} = \text{diag}(\pi_1(1-\pi_1), \pi_2(1-\pi_2), \cdots, \pi_n(1-\pi_n))$. Here $\pi_i = \pi_i^{(i-1)}$
        Set $z_i^{(i)} = \hat{\eta}_i^{(i-1)} + (W^{(i)})^{-1}(y_i - \hat{\pi}_i{}^{(i-1)})$
        Estimate $\hat{\beta}^{(i)} = (X^T W^{(i)} X)^{-1} X^T W^{(i)} z^{(i)}$
        Set, $\eta_i^{(i)} = \sum_j x_{ij}\hat{\beta}_j^{(i)}$, $\hat{\pi}_j^{(i)} = \frac{\exp(\eta_i^{(i)})}{1+\exp(\eta_i^{(i)})}$
Return $\hat{\beta}^{(final)}$
```

## 3.2 M-Estimator

Huber (1964) introduced the M-estimator as a generalization of the ML estimator. The estimator operates similarly to the standard Least-Squares method. Like the Least-

Squares (LS) estimator, the M-estimator tries to minimize the sum of residuals. However, the M-estimators replace the squared function with another function $\rho$, attempting to reduce the effect of outliers. The estimator $T_n = T_n(x_1, \cdots, x_n)$ is defined by the minimum problem,

$$\sum \rho(x_i, T_n) = \min!, \tag{3}$$

or equivalently (under some regularity conditions) by

$$\sum \psi(x_i, T_n) = 0. \tag{4}$$

Here $x_1, \ldots, x_n$ are independently and identically distributed observations that follows the distribution $F_\beta$ with parameter $\beta$. We denote the likelihood of $F_\beta$ by $f_\beta(x)$. The function $\rho$ is an arbitrary real function and when its partial derivative is satisfied $\psi(x, \beta) = (\partial/\partial\beta)\rho(x, \beta)$, then $T_n$ satisfies Equation 4. The function $\psi(x, \beta)$ is also known as the score function (Huber and Ronchetti, 1981).

We assume that there exist a functional form $T(F)$ defined as

$$\int \psi(x, T(F))dF(x) = 0, \tag{5}$$

for any probability distribution $F$. Furthermore, Fisher consistency is required and we say that the estimator $T(F_\beta) = \beta$ is Fisher Consistent if Equation 5 is satisfied for all $\beta$ (Huber and Ronchetti, 1981).

The influence function of $T$ in $F$ is equal to

$$IF(x; T; F) = \frac{\psi(x, T(F))}{-\int \frac{\partial}{\partial\beta}\psi(x, T(F))F(dx)}. \tag{6}$$

Therefore, $IF(x; T, F)$ describes the effect of a single outlier in $x$ on the estimator $T$ (Künsch et al., 1989). We observe in Equation 6 that the influence function of an M-estimator is proportional to its score function $\psi(x, \beta)$ (Huber and Ronchetti, 1981).

According to Huber and Ronchetti (1981) and Künsch et al. (1989), under certain conditions, M-estimators are also asymptotically normal with mean zero and covariance matrix

$$V(T, F) = E\left[IF(x; T; F)IF(x; T; F)^T\right]. \tag{7}$$

### 3.2.1 Maximum Likelihood Estimator (ML)

The Maximum Likelihood (ML) estimator is a estimator which obtains optimal results when the underlying assumptions hold; however it is susceptible to outliers (Victoria-Feser, 2002).

Dunn and Smyth (2018) derive the ML estimator's score function and the Fisher scoring function for the GLM setting which can be represented in the following matrix form

$$\begin{aligned} U &= X^T W M \frac{(y-\mu)}{\phi}, \\ I &= X^T W \frac{X}{\phi}. \end{aligned} \tag{8}$$

Here $W$ is the diagonal matrix of working weights, $M$ is the diagonal matrix of link derivatives $d\eta_i/d\mu_i$ and $\phi$ is the dispersion parameter. The score vector $U = [U_0, \cdots, U_p]^T$ for $\beta$, in Equation 8, gives the vector of derivatives of the log-likelihood with respect to the coefficient vector $\beta = [\beta_0, \cdots, \beta_p]$.

The parameters $\hat{\beta}$, as outlined in Section 3.1 are estimated by Fisher scoring, using Iteratively Reweighted Least Squares (IRLS) algorithm presented at Algorithm 1. The value of $\phi$ is not needed to obtain the estimates $\hat{\beta}$'s (Dunn and Smyth, 2018). As shown on the right hand side of Equation 9, it cancels out of the term $I()^{-1}U()$

$$\begin{aligned} \hat{\beta}^{(r+1)} &= \hat{\beta}^{(r)} + I(\hat{\beta}^{(r)})^{-1} U(\hat{\beta}^{(r)}) \\ &= \hat{\beta}^{(r)} + (X^T W X)^{-1} X^T W M (y - \hat{\mu}) \\ &= (X^T W X)^{-1} X^T W z. \end{aligned} \tag{9}$$

The superscript $(r)$ indicates the $rth$ iterate, and all quantities on the right hand side are evaluated at $\hat{\beta}^{(r)}$. The working weights response vector is indicated by $z = \hat{\eta} + M(y - \hat{\mu})$ (Dunn and Smyth, 2018).

We implement the GLM with ML employing the function *glm*() in the R package **stats**. The number of iterations in the auxiliary function *control.glm*() is increased to 1000 and the positive convergence tolerance reduced to 0.001.

### 3.2.2 Mallows Quasi-likelihood Estimator (MQL)

The MQL is described as a particular case in the M-estimators class, where the influence of deviations on $y$ and $x$ are bounded separately; additionally, the estimator uses continuous down weighting to limit the influence of outliers (Cantoni and Ronchetti, 2001b).

The estimator is defined as the solution of the estimating equations,

$$\psi(y_i, \mu_i) = \nu(y_i, \mu_i) w(x_i) \mu_i' - \alpha(\beta) = 0. \tag{10}$$

Here the constant $\alpha(\beta)$ is the correction term that ensures Fisher Consistency and $\nu(\cdot)$ is a bounded chosen function. The weight function is defined as $\omega(x)$, and finally $\mu_i = g(x_i^T \beta)^{-1}$.

To describe the estimator, we employ the notation of Cantoni and Ronchetti (2001b). The choice of a bounded function $\nu(\cdot)$ guarantees robustness by putting a bound on the influence function. Therefore a bounded function $\nu(y, \mu)$ is proposed to control deviations in the $y - space$ and leverage points are down-weighted by the weights $\omega(x)$.

For the binomial particular case the MQL estimator solves the set of estimating equations,

$$\sum_{i=1}^{n} \left[ \psi_c(r_i) \omega(x_i) \frac{1}{V^{1/2}(\mu_i)} \mu_i' - \alpha(\beta) \right] = 0. \tag{11}$$

The Fisher consistency correction is $\alpha(\beta) = \frac{1}{n} \sum E[\psi_c(r_i)] \omega(x_i) \frac{1}{V^{1/2}(\mu_i)} \mu_i'$. The score function is $\nu(y_i, \mu_i) = \psi_c(r_i) \frac{1}{V^{1/2}(\mu_i)}$, with the Pearson residuals $r_i = \frac{y_i - \mu_i}{V^{1/2}(\mu_i)}$. The Huber function $\psi_c$ is defined by

$$\psi_c(r) = \begin{cases} r & |r| \leq c, \\ c \operatorname{sign}(r) & |r| > c. \end{cases} \tag{12}$$

And its respective weight function,

$$\omega(x) = \begin{cases} x, & \text{if} \quad d(x) \leq c, \\ \frac{x}{d(x)}, & \text{if} \quad d(x) > c. \end{cases} \tag{13}$$

Here $d(x)$ denotes the robust Mahalanobis distance, $c$ is a tuning constant that determines the asymptotic efficiency.

We implement the MQL estimator employing the function *glmrob*() in the R package

**robustbase**. We set the option of *weights.on.x* to *robCov*. This function uses the weights based on the robust Mahalanobis distance of the design matrix (Maechler et al., 2021). Moreover in the auxiliary function *glmrob.control*() we increase the number of iterations to 1000 and reduce the positive convergence tolerance to 0.001.

### 3.2.3 Conditionally Unbiased Bounded-Influence Estimator (Cubif)

The Cubif, just like the MQL estimator, belong to the class of M-estimators and are defined by limiting the score function to be conditionally unbiased, given the independent variables. Cubif accommodates the abnormal data by reducing the influence of the outliers using continuous down weighting. This study uses the notation and definitions proposed by Künsch et al. (1989) and adapted by Habshah and Syaiba (2012).

Refer to Equation 4 in Section 3.2 the general form of the M-estimator was defined. Moreover, for $\hat{\beta}_n$ to be consistent, the estimating equation has to be unbiased for all $\beta$ as shown in Equation 5. Fisher consistency is a minimal requirement, however, in linear and generalized linear regressions, it can become hard to achieve. In those settings, the estimation is too weak because it involves the distribution of the predictor $x$ (Law et al., 1986). Contrarily, conditional Fisher consistency does not rely on the independent variables being random. Even if they are, it does not involve the distribution of the independent variables, as shown in

$$E(\psi(y,x,\beta) \mid x) = \int\int \psi(y,x,\beta) P_\beta(dy \mid x) = 0, \tag{14}$$

for all $\beta$ and $x$. Using such conditional property, Künsch et al. (1989) constructed a robust M-estimator with the following score function

$$\psi_{cond}(y,x,\beta,B) = W(y,x,\beta,b,B)\left\{y - g(x^T\beta) - c\left(x^T\beta, \frac{b}{h(x,B)}\right)\right\}. \tag{15}$$

Here B is a dispersion matrix, $h(x,B) = (x^T B^{-1} x)^{1/2}$ measures the leverage and $b$ is bound on the measure of infinitesimal sensitivity. The function $c = \left(x^T\beta, \frac{b}{h(x,B)}\right)$ is a corrected bias with corrected residuals defined by

$$r(y,x,\beta,b,B) = y_i - g(x^T\beta) - c\left(x^T\beta, \frac{b}{h(x,B)}\right). \tag{16}$$

The weight function is $W(y, x, \beta, b, B) = W_b\left(r(y, x, \beta)h(x, B)\right)$ with $W_b$ equal to the Huber weight function defined as $W_b(x) = \min(1, \frac{b}{|x|})$. The scalar function $c$ and the matrix $B$ are chosen so that the estimator is conditionally Fisher consistent for all $\beta$ and $x$, and such that $s(\psi_{cond}) = b$ is guaranteed. The $s(\psi)$ is the self-standardized sensitivity and is defined as

$$s(\psi)^2 = \sup_{y,x} \psi(y, x, \beta)^T W(\psi, \beta)^{-1} \psi(y, x, \beta). \tag{17}$$

The sensitivity measures the maximum influence an observation has on a linear combination of parameters, with a standardization by the asymptotic standard deviation of this linear combination (Künsch et al., 1989). Moreover, the estimator is conditionally Fisher consistent and has a bounded influence function for any choice of $B$, ensuring that the outliers have limited influence.

We implement the *Cubif* estimator employing the function *glmRob*() in the R package **robust** (Wang et al., 2020). As for the other two estimators in the auxiliary function *glmRob.control*() the number of iterations is increased to 1000 and the positive convergence tolerance reduced to 0.001.

## 3.3 Machine Learning Classifiers

### 3.3.1 Naïve Bayes Classifier (NB)

The Naïve Bayes is a classifier that employs Bayes' theorem to extract the conditional posterior probabilities assuming conditional independence among the independent variables. An NB classifier considers each independent variable to contribute individually to the conditional posterior probability, despite any possible correlations between the features; in other words, it assumes that given the class variables, the value of any feature is entirely independent of other features. This study defines the NB classifier using the notation from Majka (2020).

Using the Bayes theorem, the class-specific conditional probabilities are equal to

$$P(Y = C_k \mid X = x) = \frac{P(Y = C_k)P(X = x \mid Y = C_k)}{P(X = x)}. \tag{18}$$

The classifier decomposes the conditional posterior probabilities $P(Y = C_k \mid X = x)$ into the product of the likelihood $P(X = x \mid Y = C_k)$ times the prior probabilities $P(Y = C_k)$ scaled by the marginal likelihood of the data $P(X = x)$.

The independence assumption is generally unlikely to hold. Consequently, the independent variables are "naively" assumed to be conditionally independent, given the class label $C_k$. The classifier is named after this assumption.

From Equation 18 we obtain the following

$$\begin{aligned} P(Y = C_k \mid X = x) &= \frac{P(Y = C_k)P(X_i = x_i \mid Y = C_k)}{P(X_1 = x_1, \cdots, X_p = x_p)}, \\ P(Y = C_k \mid X = x) &\propto P(Y = C_k) \prod_{i=1}^{d} P(X_i = x_i \mid Y = C_k), \\ \log P(Y = C_k \mid X = x) &\propto \log P(Y = C_k) + \sum_{i=1}^{p} P(X_i = x_i \mid Y = C_k). \end{aligned} \tag{19}$$

The denominator $P(X_1 = x_1, \cdots, X_p = x_p)$, in the Bayesian setting, is a constant with respect to the class label $C_k$. Thus the conditional probability $P(Y = C_k \mid X = x)$ in equation 19 is proportional to the numerator. Moreover, to simplify the computations we use logarithmic transformation.

The classifier will choose the class which maximizes the log-posterior probability to be the prediction as show by

$$\hat{C} = \underset{k \in \{1, \cdots, K\}}{\arg\max} \left( \log P(Y = C_k) + \sum_{i=1}^{p} P(X_i = x_i \mid Y = C_k) \right). \tag{20}$$

### Parameter estimation

The posterior probability function shown in Equation 19 can be also shown as

$$\begin{aligned}\log \frac{P(C_k = i \mid X)}{P(C_k = J \mid X)} &= \log \frac{\pi_i g_i(X)}{\pi_J g_J(X)} \\ &= \log \frac{\pi_i \prod_{k=1}^{p} g_{ik}(X_k)}{\pi_J \prod_{k=1}^{p} g_{Jk}(X_k)} \\ &= \log \frac{\pi_i}{\pi_J} + \sum_{k=1}^{p} \log \frac{g_{ik}(X_k)}{g_{Jk}(X_k)} \\ &= \alpha_i + \sum_{k=1}^{p} f_{ik}(X_k),\end{aligned} \tag{21}$$

with $\alpha_i$ as the log prior probabilities and $f(X)$ the likelihood function. In this fashion, the function $f_{ik}(X_k)$ acts as the parameter of the model, and can be estimated in a non-parametric way, using kernel density estimation (KDE). This study opts for such a technique as it stays robust even when the underlying distributional assumptions do not hold. In Equation 21, the NB can also be seen as a generalized additive model (GAM), which is a particular case of GLM.

To estimate the $k-$th class conditional probability function, the following equation is considered:

$$\hat{f}_{hik} = \frac{1}{n_k h_{ik}} \sum_{j=1}^{n} K\left(\frac{x - x_i^{(k)}}{h_{ik}}\right) \mathrm{I}(y^{(k)} = C_k). \tag{22}$$

Here $n_k$ is the number of samples in the $k-$th class, $h_{ik}$ is a class-specific bandwidth that controls the smoothness (usually chosen to be as small as the data allows), $K(\cdot)$ is a kernel function that defines the shape of the density curve and the indicator function controls which observation belongs to a specific class (Silverman, 1999).

We use the R package **naivebayes**, employing the function *naive_bayes()* with *usekernel* option equal true (Majka, 2017).

### 3.3.2 Support Vector Classifier (SVC)

Support Vector Classifier, like logistic regression, is a method that creates a hyperplane or decision boundary in the N-dimensional feature space that separates the data points into classes. SVC uses the "kernel trick" to find the best line separator which acts as a decision boundary with the same distance from the boundary point of both classes (Hastie et al., 2017).

This study considers the classification problem with two classes $Y = \{-1, 1\}$. The training data consists of N pairs $(x_i, y_i)$ with $x_i = \mathbb{R}^p$ and $\left\{x : f(x) = x^T\beta + \beta_0 = 0\right\}$ defines a hyperplane.

SVCs are commonly defined as a quadratic problem of the form

$$\begin{gathered} \min_{\beta,\beta_0} \frac{1}{2}\left\|\beta\right\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{subject to: } \xi_i \geq 0,\ y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i\ \forall_i \end{gathered} \tag{23}$$

where $C$ is the cost hyperparameter and $\xi$ are the slack variables which are the distance of the datapoints to the margin.

SVCs use an implicit mapping $h$ of the input data into a high-dimensional feature space defined by a function that returns the inner product $\langle h(x), h(x')\rangle$ between the images of two data points $x, x'$ in the feature space, called "kernel function." The learning process takes place in the feature space, and the data points only appear inside the dot products with other points. This is called "kernel trick" (Karatzoglou and Meyer, 2006; Schölkopf et al., 2000). In other words, if a projection $h : X \to H$ is used, the dot product $\langle h(x), h(x')\rangle$ can be represented by a function $k$, $k(x, x') = \langle h(x), h(x')\rangle$ , which is mathematically simpler than directly projecting $x$ and $x'$ onto the feature space $H$. One compelling property of the SVCs is that, once a valid kernel function has been chosen, it is possible to work with any space dimensions without any notable additional computational cost, as feature mapping is never performed (Karatzoglou and Meyer, 2006).

The most popular kernel according to Schölkopf et al. (2000) and Karatzoglou and Meyer (2006) is the radial basis function (RBF). Mueller and Massaron (2016) explain that RBF can map and approximate almost any nonlinear function, provided that $\gamma$, the shape parameter, is tuned. In addition, it can detect complex classification rules that other algorithms may fail to find. The RBF kernel creates a margin around every support vector, drawing bubbles in the feature space. The $\gamma$ hyper-parameter dictates expansion or restriction on the volume of the bubbles so that they fuse and shape the classification areas. Moreover, Chung et al. (2003) show that RBF is very flexible and can adapt itself to different learning strategies. Nevertheless, this flexibility comes at the expense of a larger variance in the accuracy estimation.

Our study proceeds with the RBF kernel due to its high flexibility and data adaptability. Equation 24 shows the mathematical form of the RBF kernel,

$$K(x, x') = \langle h(x), h(x') \rangle = \exp(-\gamma \left\| x - x' \right\|^2). \tag{24}$$

When using the RBF kernel, two hyperparameters are of utmost concern to be tuned. The first is $\gamma$, and the second is $C$. The hyper-parameter, $\gamma$, defines how much influence a training example has. The penalty parameter of the regularization parameter, $C$, controls the trade-off between the simplicity of the decision surface and the misclassification of training examples.

The parameters were tuned with the function *tune.svm*() available in the R-package **e1071** to find the optimal hyperparameters for each dataset (Meyer et al., 2019). Furthermore, the values for $C$ and $\gamma$ were picked exponentially apart as shown by *Scikit-learn.org*[1]. To reduce computational cost, we choose the following grid:

$$C = \{0.01, 0.1, 1, 10, 100\},$$
$$\gamma = \{1, 0.1, 0.01, 0.001\}.$$

We chose the range of values which contain combination of hyperparameters that achieves the highest accuracy as showed by `Scikit-learn.org`.

## 3.4 Performance Measures

Supervised classification aims to build a compact model of the class labels' distribution in terms of the input features or independent variables (Kotsiantis et al., 2006). The resulting classifier assigns class labels to the testing observations where we know the dependent variables' values but not the class label's value.

The correctness of classification can be evaluated by computing four counts which

[1] Scikit-learn.org, RBF SVM parameters

constitute a confusion matrix, displayed in Table 1.

**Table 1:** Confusion matrix in binary classification

| | Predicted positive (1) | Predicted negative (0) |
|---|---|---|
| Actual positive (1) | True positive (TP) | False negative (FN) |
| Actual negative (0) | False positive (FP) | True negative (TN) |

From those four counts, we can obtain several performance measures divided into three groups (Goutte and Gaussier, 2005). The first numerical value performance measures (i.e. accuracy, precision, recall), are considered biased when learning with skewed data (Bradley, 1997; Goutte and Gaussier, 2005; Powers, 2015). The second, graphical performance measures (i.e., roc-curves and precision-recall curves) are used when uncertainty about the misclassification costs or the class distribution is not similar (imbalancement). Graphical measures can present a classifier's performance for various costs and class distributions (Goutte and Gaussier, 2005). Finally, complex numerical measures (i.e., F1-Scores, G-Mean, and Youden's Index) involve combining numerical value performance measures in one metric to solve for the imbalance problem.

To compare the classifiers' performance, this study uses the following metrics: one complex numerical metric (F1-Score), one graphical metric (ROC-AUC), and the Logarithmic Loss (LL) to evaluate prediction error, or how close is the predicted value is to the true value. The first two metrics are threshold metrics, which are sensitive to thresholds. Contrary to LL, these metrics do not consider how close the prediction value is to the true value. They focus only on whether the predicted value is above or below a threshold value (Liu et al., 2014). Furthermore, to ensure reliable results, our study opted for 5-Fold cross-validated metrics. To estimate the performance measures, we use the R packages **MLmetrics** (Yan, 2016).

### F1-Score

The F1-score is the harmonic mean of recall and precision, taking into account all observations that were wrongly classified. It is a representation of all false positives and false negatives

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \cdot \ \text{Recall}}{\text{Precision} + \text{Recall}}, \qquad \text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}.$$

The F1-score is a popular evaluation metric for imbalanced data. (Goutte and Gaussier, 2005). While not as straightforward to interpret as accuracy, F1 is more suitable than accuracy when the data has an imbalanced class distribution. The metric values range from 0 to 1. A high score indicates high precision and recall (Goutte and Gaussier, 2005).

### Area Under a Receiver Operating Characteristics Curve (AUC)

The AUC is a performance measurement for binary classifier at various threshold settings (Hanley and Mcneil, 1982). AUC measures the degree of separability in the ROC curve, a two-dimensional probability curve with false positive rates (FPR) on the x-axis against the true positive rates (TPR) on the y-axis (Bradley, 1997).

The AUC value lies within the range $[0, 1]$, and it tells how much the model can distinguish between classes. A perfect classifier has AUC near to the value 1. The other extreme, near the value 0, means it has the worst measure of separability. Moreover, when AUC is 0.5, it means the model has no class separation capacity whatsoever (Mandrekar, 2010). Furthermore, it is overall robust measure to evaluate the performance of score classifiers because its calculation relies on the complete ROC curve and thus involves all possible classification thresholds (Mandrekar, 2010).

### Logarithmic Loss (Log-loss or LL)

Logarithmic Loss indicates how close the prediction probability $\pi_i$ is to the corresponding actual value $y_i$ as defined by

$$\text{LL} = -\frac{1}{n} \sum_{i=1}^{n} \left(y_i \log \hat{\pi}_i + (1 - y_i) \log(1 - \hat{\pi}_i)\right).$$

The more the predicted probability diverges from the actual value, the higher the LL value. In other words, the LL value is for a classification setting what the mean squared error is for a regression setting (Vovk, 2015).

## 3.5 Feature Selection

When it comes to feature selection using wrapper methods, Deng (1999) explains that there are three main approaches: forward selection (FS), forward-stepwise selection (FSS), and backward-stepwise elimination (BSE).

The first approach searches through all possible feature subsets combinations, which becomes computationally expensive and likely infeasible when $p$ (number of features) is relatively large. To solve such an issue, one can opt for either backward stepwise elimination or forward stepwise selection.

Backward-stepwise elimination starts with the full model and sequentially deletes the predictor with the most negligible impact on the fit. The candidate for dropping is the variable with the smallest Z-score. It can only be used when $N > p$, while FSS can always be used (Hastie et al., 2017).

Contrarily, forward-stepwise selection starts with the intercept and sequentially adds the predictor that most improves the model's fit. The technique assesses candidates' models multiple times, in a sequence of comparisons between models, performing a sequential model selection. Hastie et al. (2017) emphasize that the method is not computationally expensive compared to other feature selection techniques, and because it is a constrained search, it has a lower variance. In addition, from a robustness point of view, FSS is preferred. The bigger the feature set, the higher the probability of outliers.

As the penalty criteria, this study investigates the performance under two alternatives: the well-studied BIC and its robust version RBIC presented by Machado (1993).

The BIC is formally defined as

$$BIC = \sum_i d_i^2 + p\log(n) \tag{25}$$

and the RBIC as

$$RBIC = \sum_i \rho_c\left\{d_i\right\} + \frac{1}{2}p\log(n). \tag{26}$$

In both cases $d$ equal the model's deviances. And for the RBIC the function $\rho$ is the Huber function defined as

$$\rho_c(d) = \begin{cases} \frac{1}{2}d^2, & |d| < c, \\ c\,|d| - \frac{1}{2}c^2, & |d| \geq c. \end{cases} \tag{27}$$

The constant $c$ regulates the amount of robustness. According to Wang et al. (2007) good choices are in the range between 1 and 2. He describes that Cantoni and Ronchetti (2001a), for example, use $c = 1.20$, whereas Street et al. (1988), use $c = 1.25$. Since we are investigating the MQL estimator proposed by Cantoni and Ronchetti (2001b) we proceed with $c = 1.20$. In both cases, the model chosen is the one with the lowest BIC or RBIC.

We perform feature selection in all models but the machine learning classifiers. SVC, as previously mentioned, does not perform feature mapping (Karatzoglou and Meyer, 2006). Its performance is independent of the dimensionality of the feature space. SCV uses regularisation to avoid over-fitting, which makes feature selection unnecessary. Therefore it requires only the tuning of the hyperparameters to guarantee performance. For the NB, there is no natural method to evaluate feature importance since the method works by determining the conditional and unconditional probabilities correlated with the given features and predicts the highest probability class.

Algorithm 2 provides the details of the FSS technique (Liu, 2015).

**Algorithm 2:** Forward-Stepwise Selection

```
Result: Optimal model $\mathbf{M}_j$
Let $\mathbf{M}_0$ denote the null model, which contain no predictors.
for $k = 1, \cdots, p-1$ : do
    (a) Consider all $p-k$ models that augment the predictors in $M_k$ with one additional predictor;
    (b) Choose the best among these $p-k$ models, and call it $\mathbf{M}_{k+1}$. Here the best is defined as having the smallest BIC or RBIC;
Select a single best model $\mathbf{M}_j \in \{\mathbf{M}_0, \cdots, \mathbf{M}_p\}$ using BIC or RBIC;
```

# 4 Monte Carlo Simulation

We designed a Monte Carlo simulation to investigate how robust the GLM with the MQL estimator and the GLM with the Cubif estimator are compared to the classical ML estimator and non-parametric machine learning classifiers SVC and NB, in the binary classification framework. For that, we replicate the simulation design from Bianco et al. (2019) to generate a clean training sample.

First, we generated a training sample $\mathbf{S}$ of i.i.d observations $(y_i, x_i)$, $1 \leq i \leq n$, $x_i \in \mathbb{R}^p$ and $y_i \mid x_i \sim Bi(1, F(\gamma_o + x_i^T \beta_0))$, where the intercept $\gamma_0 = 1$. The covariates distribution is $N_p(0, \Sigma)$, where $\Sigma$ is a $p \times p$ variance-covariance matrix, defined as $cov(x_i, x_j) = (0.6)^{|i-j|}$, a variation of a Toeplitz matrix with correlation equal to 0.6 (Gray, 2006). The uncontaminated setting was denoted $\mathbf{C0}$.

To confront our estimators with challenging scenarios, we choose $p$ and $n$ such that the ratios $p/n$ are moderately large, with $n \in \{400, 500\}$ and $p \in \{8, 12\}$. The choice of a large sample size was to better ensure the overlapping cases in each replication. As presented by Victoria-Feser (2002) and Habshah and Syaiba (2012), small datasets of 50 observations or less with no overlapping cases even without contamination can lead to unidentifiable parameter estimates.

To evaluate the gain in performance due to feature selection, we chose the true regression parameter as $\beta = (1, \cdots, 1, 0, 0, 0, 0, 0)^T \in \mathbb{R}^p$,. Five components are null, and the rest are equal to one. This way, there were two settings: one where the majority of variables were not significantly different from zero, and the second where the majority was significantly different from zero. The number of replications was $R = 300$.

To study the impact of contamination, we explored three settings by adding a proportion $\epsilon = 0.01$, 0.05, or 0.10 of outliers. Here we replicated the simulation design from Victoria-Feser (2002). We start by generating misclassified points $(\tilde{y}, \tilde{x})$. First, we generated $\tilde{y}$ by taking proportions $\epsilon$ of the response variable chosen randomly and change them from 0 to 1 or 1 to 0, which constitutes the misclassification type error. Second, we generated $\tilde{x}$ by taking proportions $\epsilon$ of the explanatory variables and substitute them with the value $l = 5$. Such replacement generates misspecification in all the explanatory variables, a phenomenon also known as leverage. Lastly, misclassification and misspecification errors are assumed to be simultaneous. To add contamination to the data we used the function *contaminate* from the R package **simFrame** (Alfons et al., 2009).

Summarizing, we considered the scenarios **C1**, **C2**, and **C3** corresponding to adding a proportion $\epsilon = 0.01$, 0.05, and 0.10 outliers to the data. The study refers to these datasets as contaminated datasets. Furthermore, we considered four different settings with two different sample sizes and the number of features, creating in total, 16 different datasets.

Algorithm 3 describes in detail the implementation of the Monte Carlo Simulation. Similarly to Bianco et al. (2019) this study uses K-Fold Cross-Validation to assess the models' performances. However, instead of 10-Fold like Bianco et al. (2019) we opted for 5-Fold to save computational time.

The sampler employed in this simulation is independent, which bootstrapped 300 distributions given the parameters configuration.

**Algorithm 3:** Monte Carlo Simulation Implementation

```
Result: $F1_j = (F1_1, \cdots, F1_{300})$, $AUC_j = (AUC_1, \cdots, AUC_{300})$,
        $LogLoss_j = (LogLoss_1, \cdots, LogLoss_{300})$, for $j = (0, 1, 2, 3)$
Initialization: Set parameters $n \in \{400, 500\}$ and $p \in \{8, 12\}$
for $i = 1, \cdots, 300$ do
    Set seed equal $i$
    1. Generate datasets C0, C1, C2 and C3
    2. Initiate 5-Fold CV using datasets C0, C1, C2 and C3
        (A) Without Feature Selection
        • Train GLM-ML, GLM-MQL, GLM-Cubif, NB and SVC in the training sample;
        • Access the models in the testing set and extract cross-validated performance
          measures;
        (B) With Feature Selection
        • For GLM-ML, GLM-MQL and GLM-Cubif perform Forward-Stepwise Selection on
          the training set;
        • Access the previously obtained models in the testing set and extract
          cross-validated performance measures;
Return: Cross-Validated F1-Score, AUC and LogLoss.
```

# 5 Hypotheses

To answer the research question presented in Section 1 we formulated the following hypotheses:

**Hypothesis 1:** MQL and Cubif will perform better than ML in scenarios **C1**, **C2**, and **C3** where there is contaminated data. The models MQL and Cubif will also perform better than ML in simulations $(n, p) = (400, 12)$ and $(n, p) = (500, 12)$, where most of the variables are significantly different from zero and consequently contain more contributing observations, compared to simulations $(n, p) = (400, 8)$ and $(n, p) = (500, 8)$, where most of the variables are not significantly different from zero. This will be indicated by higher AUC, higher F1-score, and lower LL for MQL and Cubif.

**Hypothesis 2:** MQL and Cubif will perform just as well as the machine learning methods SVC and NB in scenarios **C1**, **C2**, and **C3** where there is contaminated data. This will be indicated by higher AUC, higher F1-score, and lower LL for MQL and Cubif.

**Hypothesis 3:** Supervised learning classifiers will achieve higher accuracy measures in simulations $(n, p) = (500, 8)$ and $(n, p) = (500, 12)$, where the sample size is larger, compared to simulations $(n, p) = (400, 8)$ and $(n, p) = (400, 12)$. This will be indicated by higher AUC, higher F1-score, and lower LL for NB and SVC.

**Hypothesis 4:** The criterion RBIC will perform better than BIC under in scenarios **C1**, **C2**, and **C3** where there is contaminated data. This will be indicated by higher AUC, higher F1-score, and lower LL for RBIC.

# 6 Breast Cancer Data

The original dataset with the title *Wisconsin Breast Cancer Database (January 8, 1991)* used in this paper is publically available. It was retrieved from the Machine Learning Repository website of the University of California, Irvine (UCI) `http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/`. Olvi Mangasarian donated the data on July 15, 1992, and Dr. William H. Wolberg, a physician at the University of Wisconsin Hospital at Madison, USA, created the data as it is today.

To build the database, Dr. Walberg analyzed fluid samples obtained from patients with solid breast masses (Wolberg and Mangasarian, 1990; Wolberg et al., 1994; Borges, 1989). He employed a graphical computer program called Xcyt to investigate cytological features based on a digital scan Wolberg et al. (1994). The same source outlines that the program Xcyt employs a curve-fitting algorithm to estimate for each cell in the sample a total of ten features. The software evaluates each feature on a 1 to 10 scale. Outputs closer to 1 have a high probability of being benign, and outputs closer to 10 are more likely to be malignant.

Breast cancer treatment success relies on how early the lumps are detected (Litin and Nanda, 2018). Over the past decade, medical science has been increasingly investing in technology to develop robust and accurate algorithms that can predict based on medical information whether a particular person has malignant tumors or not (Ibnouhsein et al., 2018). This dataset aims to provide cytological information in order to allow for such investigation.

The dataset comprises nine categorical features ranging from 1 to 10, one binary dependent variable (breast lumps), and one index variable. We present in Table 2 all features and their respective description. Moreover, the dataset contains 16 missing values out of 699 observations; the class distribution is not equal with 458 (65.5%) of the observations classified as benign and 241 (34.5%) as malignant.

To analyze the results, we amended the original data. First, the dependent variable's values were adjusted. The values of a benign tumor were reassigned to range from 2 to 0, and the values of the malignant tumor were reassigned from 4 to 1. The values (1,0) are suitable for binary methods such as logistic regression. We discarded the 16 missing values and the index feature because it does not add value to the analysis—the final dataset comprises nine features, the dependent variable, and 683 observations.

To accommodate to the simulation framework, we scaled and centered the data employing the function *scale*() from the R package **base**. Next, we created scenario **C1**, **C2** and **C3** by adding 1%, 5% and 10% of contamination respectively on $y$ and $x$ as outlined Section 4.

The original data is not normally distributed, and the features seem to be highly correlated, leading to singular matrices. To mitigate this issue, we opted to increase the number of folds in the cross-validation from 5 to 10. This way, the training sample increases, reducing the probability of singular matrices. As in the simulation framework, we replicated $R = 300$ times.

**Table 2:** Features' name, descriptions, and scale.

| Attribute | Domain |
|---|---|
| 0. Sample code number | id number |
| 1. Clump Thickness | 1 - 10 |
| 2. Uniformity of Cell Size | 1 - 10 |
| 3. Uniformity of Cell Shape | 1 - 10 |
| 4. Marginal Adhesion | 1 - 10 |
| 5. Single Epithelial Cell Size | 1 - 10 |
| 6. Bare Nuclei | 1 - 10 |
| 7. Bland Chromatin | 1 - 10 |
| 8. Normal Nucleoli | 1 - 10 |
| 9. Mitoses | 1 - 10 |
| 10. Breast Lump | (0 for benign, 1 for malignant) |

# 7 Results

This section first presents the findings for all 16 scenarios of the Monte-Carlo simulation and afterward the findings for the real-world data "Breast Cancer Dataset."

## 7.1 Monte Carlo Simulation

The sample size $n$ and dimension of the covariates $p$ are $(n, p) = (400, 8)$, $(400, 12)$, $(500, 8)$ and $(500, 12)$. The Monte-Carlo simulation with those particular parameters are named **MC1**, **MC2**, **MC3** and **MC4** respectively. Table 3, 4, 5, and 6 show the 10%—trimmed means of the F1-Score, AUC, and LL under **C0**, **C1**, **C2**, and **C3** for all four $(n, p)$ pairs respectively.

The number of observations $n \in \{400, 500\}$ were chosen to verify the claim that all methods deliver better results in bigger datasets. Whereas the number of features $p \in \{8, 12\}$ were explicitly chosen to define two different scenarios where one has the majority of variables contributing to the predictions and the other with the majority of the variables not contributing to the predictions.

### A. Simulation MC1 with $(n, p) = (400, 8)$

Analyzing Table 3, we observe that the highest AUC for all scenarios belongs to SVC. The opposite occurs with the F1-Score, which in the majority is the lowest. This setting of high AUC and low F1 is typical for skewed datasets. In a skewed dataset, the AUC value is usually high due to a high volume of negative samples. At the same time, the F1 remains low since it is an overall measure combining precision and recall (lower recall arises when the dataset is skewed). SVC thrives when there is a high volume of data available. Due to the contaminated variables introduced, this dataset is skewed. As a result, SVC struggles to make reliable predictions. In this first set **MC1**, where the number of observations is moderately large, though it is relatively small compared to the other datasets **MC3** and **MC4**. As most variables do not contribute to the fit (**B1**), the classifier struggles to predict the classes, and the F1 dramatically reduces.

**Table 3:** 10%—trimmed means of the measures F1-Score (F1), AUC, and Log-Loss (LL) under $(n, p) = (400, 8)$; 'F.S'. stands for 'Feature Selection', **C0**, **C1**, **C2** and **C3** stands for datasets with 0%, 1%, 5% and 10% contaminated observations respectively; The green highlights show the best performing method, whereas the red highlights show the worst-performing method given the three frameworks: No feature selection (**A**), feature selection with BIC (**B1**), and feature selection with RBIC (**B2**)

| | | **C0** | | | **C1** | | | **C2** | | | **C3** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F.S. | Method | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL |
| **A** | ML | 0.889 | 0.800 | 0.424 | 0.866 | 0.785 | 0.502 | 0.754 | 0.690 | 0.606 | 0.692 | 0.643 | 0.640 |
| | MQL | 0.888 | 0.800 | 0.426 | 0.871 | 0.790 | 0.514 | 0.806 | 0.753 | 0.784 | 0.696 | 0.650 | 0.677 |
| | Cubif | 0.889 | 0.800 | 0.424 | 0.870 | 0.790 | 0.504 | 0.805 | 0.749 | 0.668 | 0.716 | 0.685 | 0.691 |
| | NB | 0.875 | 0.789 | 0.518 | 0.859 | 0.778 | 0.617 | 0.818 | 0.745 | 0.665 | 0.768 | 0.701 | 0.717 |
| | SVC | 0.917 | 0.760 | 0.497 | 0.904 | 0.739 | 0.534 | 0.863 | 0.685 | 0.635 | 0.811 | 0.664 | 0.782 |
| **B1** | ML | 0.891 | 0.803 | 0.419 | 0.871 | 0.788 | 0.498 | 0.764 | 0.695 | 0.624 | 0.693 | 0.625 | 0.658 |
| | MQL | 0.890 | 0.802 | 0.422 | 0.869 | 0.789 | 0.510 | 0.647 | 0.622 | 0.707 | 0.690 | 0.660 | 0.673 |
| | Cubif | 0.891 | 0.803 | 0.419 | 0.872 | 0.791 | 0.499 | 0.765 | 0.709 | 0.658 | 0.687 | 0.650 | 0.673 |
| **B2** | ML | 0.891 | 0.802 | 0.419 | 0.872 | 0.790 | 0.497 | 0.765 | 0.697 | 0.626 | 0.692 | 0.629 | 0.656 |
| | MQL | 0.890 | 0.802 | 0.423 | 0.873 | 0.791 | 0.510 | 0.810 | 0.758 | 0.774 | 0.713 | 0.684 | 0.835 |
| | Cubif | 0.891 | 0.802 | 0.419 | 0.874 | 0.793 | 0.498 | 0.806 | 0.749 | 0.670 | 0.710 | 0.673 | 0.688 |

NB underperforms for scenarios **C0**, and **C1**; yet, its performance for **C2** and **C3** (the highest contaminated scenarios) is higher than MQL and Cubif for framework **B1** and **B2**, which makes it the best performing method in this setting. NB does not perform poorly under **C0** and **C1**, holding high results slightly smaller than the parametric methods. The results show that, as expected, the performance of NB decreases due to the added contamination, yet it does so at a much slower pace than the parametric methods. This shows that the non-parametric kernel density estimation of NB used to estimate the model's parameters is more robust than parametric methods in this dataset **MC2**, where the underlying assumptions regarding the data distribution do not hold.

As expected, the non-robust estimator ML does not perform well under the highly contaminated datasets in simulation **MC2**, however, it is the best performing method for scenario **C0**, which contains the least amount of contaminated data. Its influence function is unbounded, which makes the estimator highly susceptible to any proportion of contamination. In scenario **C0**, it is the best performing method for both feature selection settings; however, its performance decreases for all percentages of contamination, and drastically for **C2** and

**C3** at 5% and 10% level respectively.

Comparing MQL and Cubif, we observe that both methods perform the same without feature selection in **C0** and **C1**. The same does not hold after feature selection with RBIC. The results of this simulation **MC2** suggest that MQL is more suitable for data with high proportions of contamination (i.e., 5% and 10%). In contrast, Cubif outperforms MQL in datasets with low proportions of contamination (i.e., 0% and 1%), with similar results to the ML estimator. It also performs well in high contamination settings. The results show that in most cases, Cubif holds the lowest LL, indicating that its predictions on average are closer to the true value than MQL. The low LL may be due to Cubif's use of conditional Fisher scoring, making it less reliant on the distribution of the independent variables. Nevertheless, MQL appears to be more robust in highly contaminated datasets. Due to its continuous down weighting using the Huber function and weights based on the robust Mahalanobis distance, the MQL estimator manages to reduce the influence of the leverages added on a larger scale.

In the feature selection setting, the results show that the RBIC outperforms BIC for all contaminated datasets (**C1**, **C2** and **C3**). For **C0** the performance is similar or equal, which reinforces that BIC can be optimal for "clean" datasets; however, BIC becomes biased in the presence of outliers. While AUC and F1-Score in all contamination scenarios for all three parametric estimators increase comparing **B2** with **A**, the same does not hold when comparing **B1** with **A**. The reason is that BIC uses square deviances while RBIC uses Huber deviances. The bounded Huber deviances in the RBIC reduce the influence of the outliers, allowing the feature selection technique to pick the variables which improve the fit.

### B. Simulation MC2 with $(n, p) = (400, 12)$

Table 4 shows the results for one of the settings where the majority of the variables are significantly different from zero.

SVC seems to perform better than the previous scenario, where most variables are not significantly different from zero. For **C0**, it is by far the best performing method, with the highest AUC and F1 and lowest LL. In **C1**, **C2**, and **C3**, the same does not happen. The method holds like before the highest AUC yet most times the lowest F1-Score, making it once more dangerous to rely solely on its results, confirming our first hypothesis that robust methods perform better than non-robust ones.

**Table 4:** 10%—trimmed means of the measures F1-Score (F1), AUC, and Log-Loss (LL) under $(n, p) = (400, 12)$; 'F.S'. stands for 'Feature Selection', **C0**, **C1**, **C2** and **C3** stands for datasets with 0%, 1%, 5% and 10% contaminated observations respectively; The green highlights show the best performing method, whereas the red highlights show the worst-performing method given the three frameworks: No feature selection (**A**), feature selection with BIC (**B1**), and feature selection with RBIC (**B2**)

| | | **C0** | | | **C1** | | | **C2** | | | **C3** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F.S. | Method | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL |
| **A** | ML | 0.951 | 0.872 | 0.292 | 0.925 | 0.848 | 0.450 | 0.768 | 0.702 | 0.610 | 0.687 | 0.636 | 0.645 |
| | MQL | 0.950 | 0.870 | 0.305 | 0.931 | 0.858 | 0.509 | 0.862 | 0.815 | 1.115 | 0.701 | 0.654 | 0.798 |
| | Cubif | 0.950 | 0.871 | 0.293 | 0.931 | 0.858 | 0.462 | 0.856 | 0.804 | 0.727 | 0.714 | 0.673 | 0.723 |
| | NB | 0.943 | 0.863 | 0.334 | 0.928 | 0.853 | 0.458 | 0.876 | 0.808 | 0.612 | 0.817 | 0.752 | 0.735 |
| | SVC | 0.989 | 0.932 | 0.205 | 0.978 | 0.834 | 0.355 | 0.925 | 0.717 | 0.546 | 0.862 | 0.647 | 0.662 |
| **B1** | ML | 0.951 | 0.873 | 0.289 | 0.923 | 0.845 | 0.456 | 0.778 | 0.708 | 0.628 | 0.689 | 0.613 | 0.661 |
| | MQL | 0.949 | 0.868 | 0.298 | 0.909 | 0.829 | 0.499 | 0.652 | 0.628 | 0.705 | 0.706 | 0.670 | 0.679 |
| | Cubif | 0.951 | 0.872 | 0.289 | 0.920 | 0.842 | 0.474 | 0.758 | 0.704 | 0.674 | 0.671 | 0.634 | 0.679 |
| **B2** | ML | 0.951 | 0.873 | 0.289 | 0.927 | 0.850 | 0.451 | 0.786 | 0.717 | 0.632 | 0.688 | 0.619 | 0.659 |
| | MQL | 0.949 | 0.869 | 0.300 | 0.929 | 0.855 | 0.500 | 0.846 | 0.794 | 0.931 | 0.725 | 0.693 | 0.963 |
| | Cubif | 0.951 | 0.873 | 0.289 | 0.930 | 0.856 | 0.462 | 0.849 | 0.796 | 0.728 | 0.718 | 0.678 | 0.722 |

With NB, the opposite occurs. In scenario **C0**, the results reveal that the method is the worst performing method. However, for **C2**, and **C3** its results are higher than all parametric methods in all feature settings (no feature selection, feature selection with BIC and RBIC). Moreover, in **C1**, it is higher than all three parametric methods before feature selection and after with BIC.

Surprisingly ML is the best performing method after feature selection with BIC for scenarios **C0**, **C1**, and **C2**. Nevertheless, these results do not imply that ML works well under contamination; rather, BIC as feature selection criteria is a poor choice for those datasets. As discussed previously in Section 3.5, the square deviations make BIC biased under contaminated datasets.

RBIC still works better than BIC in this framework, and all RBIC's performance measures are higher than BIC for the three scenarios (**C1**, **C2** and **C3**). This was not the case when compared to before feature selection. Here we have that the majority of the variables are significantly different from zero, or in other words, they contribute to the output's prediction, which seems to play a role. For ML in all four contaminated datasets, the metrics improve after feature selection with RBIC. The same does not hold for robust the methods. For MQL and Cubif, the metrics improve only in **C3**.

Just as in the previous simulation **MC1**, when comparing MQL and Cubif, the first overperforms in scenarios with highly contaminated data (**C2** and **C3**) and the latter in scenarios with lower contamination (**C0** and **C1**).

## C. Simulation MC3 with $(n, p) = (500, 8)$

This framework contains more observations than the previous one, and it is under the scenario where most variables are not significantly different from zero.

**Table 5:** 10%—trimmed means of the measures F1-Score (F1), AUC, and Log-Loss (LL) under $(n, p) = (500, 8)$; 'F.S'. stands for 'Feature Selection', **C0**, **C1**, **C2** and **C3** stands for datasets with 0%, 1%, 5% and 10% contaminated observations respectively; The green highlights show the best performing method, whereas the red highlights show the worst-performing method given the three frameworks: No feature selection (**A**), feature selection with BIC (**B1**), and feature selection with RBIC (**B2**)

| | | | **C0** | | | **C1** | | | **C2** | | | **C3** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F.S. | Method | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL |
| **A** | ML | 0.888 | 0.801 | 0.424 | 0.868 | 0.787 | 0.499 | 0.759 | 0.696 | 0.601 | 0.696 | 0.648 | 0.636 |
| | MQL | 0.887 | 0.801 | 0.426 | 0.873 | 0.794 | 0.512 | 0.814 | 0.761 | 0.778 | 0.699 | 0.652 | 0.661 |
| | Cubif | 0.888 | 0.801 | 0.424 | 0.872 | 0.793 | 0.503 | 0.809 | 0.754 | 0.661 | 0.720 | 0.689 | 0.683 |
| | NB | 0.874 | 0.788 | 0.519 | 0.862 | 0.780 | 0.592 | 0.820 | 0.747 | 0.649 | 0.769 | 0.702 | 0.704 |
| | SVC | 0.910 | 0.776 | 0.477 | 0.899 | 0.770 | 0.497 | 0.856 | 0.717 | 0.588 | 0.803 | 0.668 | 0.743 |
| **B1** | ML | 0.891 | 0.804 | 0.420 | 0.872 | 0.791 | 0.495 | 0.767 | 0.700 | 0.621 | 0.692 | 0.627 | 0.654 |
| | MQL | 0.890 | 0.802 | 0.422 | 0.872 | 0.792 | 0.507 | 0.648 | 0.624 | 0.707 | 0.695 | 0.663 | 0.670 |
| | Cubif | 0.891 | 0.804 | 0.420 | 0.874 | 0.794 | 0.497 | 0.772 | 0.717 | 0.656 | 0.688 | 0.652 | 0.670 |
| **B2** | ML | 0.890 | 0.804 | 0.420 | 0.858 | 0.780 | 0.509 | 0.759 | 0.695 | 0.624 | 0.687 | 0.624 | 0.654 |
| | MQL | 0.889 | 0.801 | 0.424 | 0.860 | 0.787 | 0.522 | 0.800 | 0.753 | 0.760 | 0.714 | 0.682 | 0.824 |
| | Cubif | 0.890 | 0.803 | 0.420 | 0.858 | 0.788 | 0.512 | 0.794 | 0.749 | 0.667 | 0.700 | 0.671 | 0.686 |

SVC and NB repeat the same behavior as it did in simulation **MC1**. SVC struggles to predict accurate results in the contaminated scenarios, whereas NB is the best performing method under high contaminated scenarios (**C1**, **C2** and **C3**). MQL under **B2** performs better than other methods for highly contaminated datasets (**C2** and **C3**) whereas Cubif for datasets with lower contamination levels (**C0** and **C1**).

The 100 added observations do not increase the performance for **A** and **B1**, however they do for **B2**. For **B2**, there is an improvement in all three metrics (AUC, F1, and LL) for all three parametric models (ML, MQL and Cubif). In **B2** most of the variables are non zero, making the percentage of contributing observations higher than in **A** and **B1**. This is likely the reason why the only performance improvement in a dataset with a higher amount of observations was under scenario **B2**.

## D. Simulation MC4 with $(n, p) = (500, 12)$

Unlike the previous framework, the performance of all metrics under all scenarios increases, which indicates that the 100 added observations contributed to the predictive power of the model.

In this simulation **MC4**, the non-parametric machine learning methods SVC and NB are by far the best performing. SVC has the highest AUC and F1 and lowest LL under **C0** and **C1** (none or low levels of contamination). This confirms that machine learning methods, as mentioned in Section 2, perform better with greater amounts of data, as they use available data to extract patterns and then predict results. In this simulation, most variables contribute to the result, which means that the majority of the observations added were relevant additions and contributed to the classifier's performance. In the same fashion, NB performs better than other methods under **C2** and **C3**.

Under **B2**, the robust MQL is the best performing method for highly contaminated datasets, whereas Cubif is for low contaminated datasets. There is a dramatic increase in performance comparing **B1** and **B2**, whereas, for **A** against **B2**, the increase in all methods is only visible for the 10% contamination scenario (**C3**).

**Table 6:** 10%—trimmed means of the measures F1-Score (F1), AUC, and Log-Loss (LL) under $(n, p) = (500, 12)$; 'F.S'. stands for 'Feature Selection', **C0**, **C1**, **C2** and **C3** stands for datasets with 0%, 1%, 5% and 10% contaminated observations respectively; The green highlights show the best performing method, whereas the red highlights show the worst-performing method given the three frameworks: No feature selection (**A**), feature selection with BIC (**B1**), and feature selection with RBIC (**B2**)

| | | **C0** | | | **C1** | | | **C2** | | | **C3** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F.S. | Method | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL |
| **A** | ML | 0.953 | 0.875 | 0.283 | 0.927 | 0.853 | 0.445 | 0.765 | 0.701 | 0.606 | 0.692 | 0.641 | 0.640 |
| | MQL | 0.952 | 0.874 | 0.292 | 0.933 | 0.861 | 0.500 | 0.861 | 0.816 | 1.114 | 0.702 | 0.655 | 0.768 |
| | Cubif | 0.953 | 0.875 | 0.285 | 0.933 | 0.861 | 0.460 | 0.855 | 0.806 | 0.722 | 0.720 | 0.681 | 0.720 |
| | NB | 0.946 | 0.865 | 0.325 | 0.930 | 0.855 | 0.435 | 0.877 | 0.808 | 0.601 | 0.819 | 0.754 | 0.713 |
| | SVC | 0.988 | 0.923 | 0.218 | 0.975 | 0.853 | 0.336 | 0.917 | 0.751 | 0.516 | 0.852 | 0.662 | 0.630 |
| **B1** | ML | 0.954 | 0.876 | 0.280 | 0.925 | 0.850 | 0.451 | 0.774 | 0.707 | 0.627 | 0.689 | 0.622 | 0.656 |
| | MQL | 0.952 | 0.873 | 0.287 | 0.909 | 0.831 | 0.499 | 0.637 | 0.615 | 0.709 | 0.712 | 0.676 | 0.674 |
| | Cubif | 0.954 | 0.875 | 0.280 | 0.922 | 0.846 | 0.470 | 0.753 | 0.698 | 0.674 | 0.674 | 0.635 | 0.674 |
| **B2** | ML | 0.954 | 0.877 | 0.279 | 0.930 | 0.856 | 0.445 | 0.782 | 0.714 | 0.629 | 0.689 | 0.626 | 0.653 |
| | MQL | 0.952 | 0.874 | 0.289 | 0.932 | 0.859 | 0.496 | 0.845 | 0.795 | 0.935 | 0.729 | 0.697 | 0.960 |
| | Cubif | 0.954 | 0.877 | 0.280 | 0.933 | 0.861 | 0.458 | 0.848 | 0.797 | 0.727 | 0.719 | 0.680 | 0.718 |

## 7.2 Breast Cancer Data

Table 7 displays the results for the Breast Cancer Dataset. SVC is the best performing method, holding the highest AUC, F1, and lowest LL. The dataset comprises 683 observations where all variables contribute to the results as outlined in Section 6. The dataset's results resembles the scenario where $(n, p) = (500, 12)$. This scenario holds the largest sample size in this study, and the majority of the features as significantly different from zero. SVC and NB have a higher performance than the parametric methods in all four contamination settings. The results bolster that machine learning methods operate better when there is a large volume of data available.

Like in the Monte Carlo simulation, ML becomes the worst performing method for all contamination levels. On the other hand, MQL appears to outperform Cubif in all three contamination settings. For **C1** and **C2** the performances are comparable, yet for **C3** Cubif under performs. Moreover, under framework **A** in contaminated datasets, MQL and Cubif, particularly the first, display comparable performance to NB, which is a non-parametric method. Our results support our first hypothesis that MQL and Cubif will perform better than ML in scenarios **C1**, **C2**, and **C3** where there is contaminated data.

In the feature selection analysis, in settings with high proportions of contamination **C2** and **C3** (5% and 10%), RBIC performs better than BIC, as in the Monte Carlo simulation. Surprisingly, for low proportions (0% and 1%) of contamination BIC deliver better results. There is no significant improvement comparing framework **A** with **B1** and **B2**. In fact, most values remain the same or increase and decrease by a tiny percentage. This is because all variables contribute to the prediction, making the feature reduction unnecessary.

**Table 7:** 10%—trimmed means of the measures F1-Score (F1), AUC, and Log-Loss (LL) for Breast Cancer Dataset; 'F.S'. stands for 'Feature Selection', **C0**, **C1**, **C2** and **C3** stands for datasets with 0%, 1%, 5% and 10% contaminated observations respectively; The green highlights show the best performing method, whereas the red highlights show the worst-performing method given the three frameworks: No feature selection (**A**), feature selection with BIC (**B1**), and feature selection with RBIC (**B2**)

| | | **C0** | | | **C1** | | | **C2** | | | **C3** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F.S. | Method | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL | AUC | F1 | LL |
| **A** | ML | 0.995 | 0.952 | 0.096 | 0.990 | 0.941 | 0.183 | 0.968 | 0.900 | 0.367 | 0.928 | 0.777 | 0.478 |
| | MQL | 0.994 | 0.950 | 0.180 | 0.988 | 0.946 | 0.314 | 0.968 | 0.930 | 0.801 | 0.944 | 0.914 | 1.367 |
| | Cubif | 0.995 | 0.952 | 0.102 | 0.990 | 0.942 | 0.195 | 0.965 | 0.879 | 0.376 | 0.865 | 0.661 | 0.523 |
| | NB | 0.993 | 0.961 | 0.287 | 0.993 | 0.957 | 0.312 | 0.990 | 0.937 | 0.313 | 0.982 | 0.917 | 0.381 |
| | SVC | 0.998 | 0.970 | 0.068 | 0.997 | 0.960 | 0.100 | 0.993 | 0.948 | 0.124 | 0.982 | 0.932 | 0.187 |
| **B1** | ML | 0.994 | 0.947 | 0.109 | 0.988 | 0.935 | 0.193 | 0.968 | 0.897 | 0.370 | 0.939 | 0.797 | 0.475 |
| | MQL | 0.994 | 0.952 | 0.114 | 0.989 | 0.948 | 0.234 | 0.966 | 0.925 | 0.677 | 0.939 | 0.901 | 0.505 |
| | Cubif | 0.993 | 0.947 | 0.112 | 0.989 | 0.940 | 0.198 | 0.968 | 0.903 | 0.380 | 0.938 | 0.821 | 0.505 |
| **B2** | ML | 0.992 | 0.940 | 0.122 | 0.988 | 0.932 | 0.197 | 0.968 | 0.898 | 0.369 | 0.942 | 0.800 | 0.475 |
| | MQL | 0.992 | 0.943 | 0.136 | 0.987 | 0.940 | 0.257 | 0.968 | 0.931 | 0.704 | 0.943 | 0.912 | 1.225 |
| | Cubif | 0.992 | 0.940 | 0.122 | 0.988 | 0.936 | 0.206 | 0.968 | 0.914 | 0.408 | 0.943 | 0.873 | 0.532 |

# 8 Concluding Remarks

## 8.1 Discussion

The simulation shows that robust estimators MQL and Cubif outperform the traditional ML, confirming our first hypothesis (H1). Both robust estimators hold higher performance under contamination than ML for all pairs $(n, p)$. The results reveal that MQL tends to perform better than Cubif for high proportions of contamination, namely the contamination conditions of 5% and 10%. Nevertheless, when analyzing the LL of both methods, the results show that in most cases, Cubif holds a lower LL to MQL, indicating that its predictions, on average, are closer to the true value than MQL.

When comparing the robust estimators with both supervised learning classifiers, the results showed that SVC holds high values of AUC; however, it also held low values of F1, implying that the model struggled to predict the classes when exposed to the contamination that skewed the data. Still, SVC is the best performing method for the Breast Cancer Data, supporting the argument that supervised classifiers thrive and tend to perform better than robust parametric methods when the data is abundant and sustaining the third hypothesis **H3**. NB also performed well in the Monte Carlo simulation and with the Breast Cancer Data. NB performed slightly worse than parametric estimators in settings with low data contamination, partially supporting our second hypothesis (H2). However, NB, like SVC, performed better than the parametric methods in the datasets with high contamination levels. It seemed that the performance of NB decreased due to the added contamination as expected, yet at a much slower pace than the parametric methods.

The study confirms our fourth hypothesis (H4) for the feature selection analysis that RBIC outperforms BIC for all contaminated datasets. In the setting of no contamination, the performance is similar or equal, demonstrating that BIC is optimal or the same for clean datasets; however, it becomes biased in the presence of outliers. For RBIC, on the other hand, the results maintained high even for large proportions of contamination. Confirming that the bounded Huber deviances reduce the influence of the outliers, allowing the feature selection technique to pick the variables which improve the fit. The non-robust criteria showed decreased performance compared to no feature selection, whereas the robust version improved.

Both non-parametric supervised learning classifiers and robust parametric methods performed well in settings with contaminated data. The study confirmed that supervised learning classifiers improve their performance for the settings with more observations, sustaining **H3** and might be a better a choice for large datasets. For smaller datasets, the parametric methods were shown to have only a slight improvement over the non-parametric methods, yet are much smaller than the machine learning methods. As stated in Section 2, both methods (NB and SVC) thrive in a massive volume of data once they work by analyzing prior experiences to extract reasonable and practical regularities and patterns to make predictions.

Ultimately, in regards to our third hypothesis, if the goal of a model is purely classification, supervised learning methods achieve higher performance if there is enough data availability. Parametric methods, particularly robust methods, are valuable for parameter interpretation. In SVC, for example, it is not possible to extract the parameter values. NB in such frameworks becomes an attractive option. Its performance is comparable to SVC in large data volume, and the method allows for parameter extraction.

## 8.2 Limitations and Recommendations

The methodological choices were constrained by computational power and package availability. For future studies, it is recommended to increase the sample sizes and the number of features. The validation technique k-Fold Cross-Validation delivers better results with more folds; therefore, using 10-Fold instead of 5-Fold might produce more accurate results.

Other variable reduction methods should also be investigated in future studies, especially embedded methods (i.e., LASSO, Elastic Net, Ridge Regression).

Regarding supervised learning, other popular methods can be further compared. In future studies, for example, random forest and neural networks are worth investigating.

In the M-estimator class, several other estimators could be examined in the same framework as this study investigated MQL and Cubif, for example, Bianco and Yohai (BY) and Weighted Bianco and Yohai (WBY) for logistic regression by Bianco and Yohai (1996).

# References

Acuña, E. and Rodriguez, C. (2004). On detection of outliers and their effect in supervised classification. *University of Puerto Rico at Mayaguez*, 15.

Ahmad, S., Ramli, N. M., and Midi, H. (2010). Robust estimators in logistic regression: A comparative simulation study. *Journal of Modern Applied Statistical Methods*, 9(2):502–511.

Akaike, H. (1998). Information theory and an extension of the maximum likelihood principle. In *Selected papers of hirotugu akaike*, pages 199–213. Springer.

Alfons, A., Templ, M., and Filzmoser, P. (2009). simframe: An object-oriented framework for statistical simulation. Technical report, Research Report CS-2009-1, Department of Statistics and Probability Theory.

Bhattacharjee, R. and Chaudhuri, K. (2020). When are non-parametric methods robust? In *International Conference on Machine Learning*, pages 832–841. PMLR.

Bianco, A. M., Boente, G., and Chebi, G. (2019). Penalized robust estimators in logistic regression with applications to sparse models. *arXiv preprint arXiv:1911.00554*.

Bianco, A. M. and Yohai, V. J. (1996). Robust estimation in the logistic regression model. In *Robust statistics, data analysis, and computer intensive methods*, pages 17–34. Springer.

Borges, L. R. (1989). Analysis of the wisconsin breast cancer dataset and machine learning for breast cancer detection. *Group*, 1(369):15–19.

Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159.

Cantoni, E. and Ronchetti, E. (2001a). Resistant selection of the smoothing parameter for smoothing splines. *Statistics and Computing*, 11(2):141–146.

Cantoni, E. and Ronchetti, E. (2001b). Robust inference for generalized linear models. *Journal of the American Statistical Association*, 96(455):1022–1030.

Chung, K.-M., Kao, W.-C., Sun, C.-L., Wang, L.-L., and Lin, C.-J. (2003). Radius margin bounds for support vector machines with the rbf kernel. *Neural computation*, 15(11):2643–2681.

Colas, F. and Brazdil, P. (2006). Comparison of svm and some older classification algorithms in text classification tasks. In *IFIP International Conference on Artificial Intelligence in Theory and Practice*, pages 169–178. Springer.

Deng, K. (1999). *Omega: On-line memory-based general purpose system classifier*. Carnegie Mellon University.

Dunn, P. K. and Smyth, G. K. (2018). Generalized linear models: Estimation. In *Generalized Linear Models With Examples in R*, pages 243–263. Springer.

Field, C. and Ronchetti, E. (1985). A tail area influence function and its application to testing. *Sequential Analysis*, 4(1-2):19–41.

Golpour, P., Ghayour-Mobarhan, M., Saki, A., Esmaily, H., Taghipour, A., Tajfard, M., Ghazizadeh, H., Moohebati, M., and Ferns, G. A. (2020). Comparison of support vector machine, naïve bayes and logistic regression for assessing the necessity for coronary angiography. *International Journal of Environmental Research and Public Health*, 17(18):6449.

Goutte, C. and Gaussier, E. (2005). A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European conference on information retrieval*, pages 345–359. Springer.

Gray, R. M. (2006). Toeplitz and circulant matrices: A review.

Habshah, M. and Syaiba, B. (2012). The performance of classical and robust logistic regression estimators in the presence of outliers. *Editorial Board*, page 313.

Hampel, F., Ronchetti, E., Rousseeuw, P., and Stahel, W. (1986). Robust statistics: the approach based on influence functions.

Hampel, F. R. (1974). The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393.

Hampel, F. R. (1983). The robustness of some nonparametric procedures. *A Festschrift for Erich L. Lehmann*, pages 209–238.

Hanley, J. and Mcneil, B. (1982). The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143:29–36.

Hastie, T., Friedman, J., and Tisbshirani, R. (2017). *The Elements of statistical learning: data mining, inference, and prediction*. Springer.

Heyde, C. C. (2008). *Quasi-likelihood and its application: a general approach to optimal parameter estimation*. Springer Science & Business Media.

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101.

Huber, P. J. (1981). *Robust statistics*. Wiley.

Huber, P. J. and Ronchetti, E. M. (1981). *Robust statistics*. Wiley, 2 edition.

Ibnouhsein, I., Jankowski, S., Neuberger, K., and Mathelin, C. (2018). The big data revolution for breast cancer patients. *European journal of breast health*, 14(2):61.

Jongh, P. J., Wet, T. D., and Welsh, A. H. (1988). Mallows-type bounded-influence-regression trimmed means. *Journal of the American Statistical Association*, 83(403):805.

Karatzoglou, A. and Meyer, D. (2006). Support vector machines in r. *Journal of Statistical Software*, 15(9).

Kitromilidou, S. and Fokianos, K. (2015). Mallows' quasi-likelihood estimation for log-linear poisson autoregressions. *Statistical Inference for Stochastic Processes*, 19(3):337–361.

Kotsiantis, S. B., Zaharakis, I. D., and Pintelas, P. E. (2006). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26(3):159–190.

Künsch, H. R., Stefanski, L. A., and Carroll, R. J. (1989). Conditionally unbiased bounded-influence estimation in general regression models, with applications to generalized linear models. *Journal of the American Statistical Association*, 84(406):460–466.

Law, J., Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J., and Stahel, W. A. (1986). Robust statistics-the approach based on influence functions. *The Statistician*, 35(5):565.

Litin, S. C. and Nanda, S. (2018). *Mayo Clinic family health book, 2018, 5th Edition.* Mayo Clinic.

Liu, A. (2015). Linear model selection and regularization. `https://web.stanford.edu/~hastie/MOOC-Slides/model_selection.pdf`. [Online; Accessed August, 17, 2021].

Liu, Y., Zhou, Y., Wen, S., and Tang, C. (2014). A strategy on selecting performance metrics for classifier evaluation. *International Journal of Mobile Computing and Multimedia Communications*, 6:20–35.

Machado, J. (1993). Robust model selection and m-estimation. *Econometric Theory*, 9(3):478–493.

Maechler, M., Rousseeuw, P., Croux, C., Todorov, V., Ruckstuhl, A., Salibian-Barrera, M., Verbeke, T., Koller, M., Conceicao, E. L., and di Palma, M. A. (2021). Package 'robustbase'. *Basic Robust Statistics.*

Majka, M. (2017). Package 'naivebayes'.

Majka, M. (2020). Introduction to naivebayes package. `https://cran.r-project.org/web/packages/naivebayes/vignettes/intro_naivebayes`. [Online; Accessed May, 15, 2021].

Mandrekar, J. N. (2010). Receiver operating characteristic curve in diagnostic test assessment. *Journal of Thoracic Oncology*, 5(9):1315–1316.

McCullagh, P. and Nelder, J. (1989). Generalized linear models ii.

McCullagh, P. and Nelder, J. A. (1999). *Monographs on Statistics and Applied Probability 37: Generalized Linear Models.* Chapman and Hall/CRC.

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F., Chang, C.-C., Lin, C.-C., and Meyer, M. D. (2019). Package 'e1071'. *The R Journal.*

Mueller, J. and Massaron, L. (2016). *17. Going a Step beyond Using Support Vector Machines*, page 297–309. J. Wiley amp; Sons.

Mueller, M. (2004). Generalized linear models. *Handbook of Computational Statistics (Volume I). Concepts and Fundamentals*, page 681–709.

Nelder, J. A. and Wedderburn, R. W. (1972). Generalized linear models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370.

Powers, D. M. (2015). What the f-measure doesn't measure: Features, flaws, fallacies and fixes. *arXiv preprint arXiv:1503.06410.*

Preisser, J. S. and Qaqish, B. F. (1999). Robust regression for clustered data with application to binary responses. *Biometrics*, 55(2):574–579.

Qian, G. and Künsch, H. R. (1998). On model selection via stochastic complexity in robust linear regression. *Journal of Statistical Planning and Inference*, 75(1):91–116.

Reda Abonazel, M. and Mohamed Saber, O. (2020). A comparative study of robust estimators for poisson regression model with outliers. *Journal of Statistics Applications amp; Probability*, 9(2):279–286.

Ritter, G. and Gallegos, M. T. (1997). Outliers in statistical pattern recognition and an application to automatic chromosome classification. *Pattern Recognition Letters*, 18(6):525–539.

Ronchetti, E., Field, C., and Blanchard, W. (1997). Robust linear model selection by cross-validation. *Journal of the American Statistical Association*, 92(439):1017–1023.

Ronchetti, E. and Staudte, R. G. (1994). A robust version of mallows's cp. *Journal of the American Statistical Association*, 89(426):550–559.

Schwarz, G. (1978). Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2):461 – 464.

Schölkopf, B., Smola, A. J., Williamson, R. C., and Bartlett, P. L. (2000). New support vector algorithms. *Neural Computation*, 12(5):1207–1245.

Silverman, B. W. (1999). *Density estimation for statistics and data analysis.* Chapman and Hall.

Stefanski, L. A., Carroll, R. J., and Ruppert, D. (1986). Optimally hounded score functions for generalized linear models with applications to logistic regression. *Biometrika*, 73(2):413–424.

Street, J. O., Carroll, R. J., and Ruppert, D. (1988). A note on computing robust regression estimates via iteratively reweighted least squares. *The American Statistician*, 42(2):152–154.

Victoria-Feser, M.-P. (2002). Robust inference with binary data. *Psychometrika*, 67(1):21–32.

Vovk, V. (2015). The fundamental nature of the log loss function. *Fields of Logic and Computation II*, page 307–318.

Walberg, W. H. (1993). Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences UCI. `http://archive.ics.uci.edu/ml`. [Online; Accessed April, 6, 2021].

Wang, J., Zamar, R., Marazzi, A., Yohai, V., Salibian-Barrera, M., Maronna, R., Zivot, E., Rocke, D., Martin, D., Maechler, M., et al. (2020). Package 'robust'.

Wang, Y.-G., Lin, X., Zhu, M., and Bai, Z. (2007). Robust estimation using the huber function with a data-dependent tuning constant. *Journal of Computational and Graphical Statistics*, 16(2):468–481.

Wedderburn, R. W. (1974). Quasi-likelihood functions, generalized linear models, and the gauss—newton method. *Biometrika*, 61(3):439–447.

Wolberg, W. H. and Mangasarian, O. L. (1990). Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the national academy of sciences*, 87(23):9193–9196.

Wolberg, W. H., Street, W., and Mangasarian, O. (1994). Machine learning techniques to diagnose breast cancer from image-processed nuclear features of fine needle aspirates. *Cancer Letters*, 77(2):163–171. Computer applications for early detection and staging of cancer.

Yan, Y. (2016). Mlmetrics: Machine learning evaluation metrics. r package version 1.1. 1.