Topics in Advanced Statistics

The performance of MCD Estimators from FastMCD algorithm in Linear Discriminant Analysis

Tatev Aslanyan [429514]

February 2019

Erasmus School of Economics

1 Introduction

Linear Discriminant Analysis (LDA) is one of the most widely-used classification methods for predicting qualitative response variables. The classical LDA, which relies on the sample means and covariance matrices, obtains a rule that separates observations among different groups (classes) and also classifies new observations into one of those earlier obtained groups. However, this method is highly sensitive to the outliers and it becomes inappropriate once the data is contaminated. Therefore, in order to improve the robustness of the LDA, it's necessary to consider robust estimators for means and covaraince matrices. The robustified LDA employs robust estimators for location and scatter to limit the impact of outliers in the data.
There are many methods for estimating the multivariate location and scatter for contaminated data. For instance, the minimum volume ellipsoid (MVE) method of Rousseeuw (1984) which looks for the ellipsoid that covers h observations with smallest volume, provides estimators for location and scatter with positive breakdown value. Rousseeuw and Leroy (1987) proposed algorithm called MINVOL to approximate the MVE. Cook, Hawkins, and Weisberg (1992) and Agullo (1996) introduced some methods to determine the exact MVE. Another method which provides highly robust estimators for location and scatter is the minimum covariance determinant method(MCD) introduced by Rousseeuw (1984) which finds a set of h observations with the smallest determinant of covariance matrix.
CD is one of the widely-known high-breakdown methods[1] which can deal with a substantial amount of outliers in the data. Eventhough, the breakdown points of MVE and MCD are the same, MCD has several advantages compared to MVE. Namely, because MCD is asymptotically normal (Butler et al.,1993), it is statistically more efficient. Moreover, the robust distances obtained by MCD are more suited for exposing the multivariate outliers than the ones of MVE, since MCD-based distances are more precise. However, MCD has limitations with regard of its computation time. Therefore, Rousseeuw and Van Driessen (1999) introduced the algorithm of FastMCD which finds the exact MCD for small data sets and it provides accurate results for large data sets within reasonable amount of time. We will use FastMCD algorithm to compute robust estimators for location and scatter for robustified LDA. In this assignment we aim to answer following research questions:

- Does the FastMCD provide better estimates for the Linear Discriminant Analysis in the presence of outliers?
- Does Robustified LDA based on FastMCD estimates perform equally good in the presence of predictor outliers and class outliers.
- What are the advantages and disadvantages of classical LDA and robustified LDA?

The remainder of this report is structured as follows: In section 2, the methodology behind MCD and algorithm FastMCD will be discussed. We will also describe the methodology behind classical - and robustified LDA. Section 3 will present the simulation study with the data generating process and the corresponding results. In section 4, we will describe

[1]Breakdown point:the exact amount of outliers that can distort the outcome of an estimator, in the worst case scenario

the empirical results of classical and robustified LDA based on Swiss bank note data. Section 5 will summarize the findings of this report with answers to the research questions.

2 Methodology

2.1 MCD estimation and FastMCD Algorithm

As it was mentioned in previous section, the idea behind MCD is to find subset of h points which has the smallest determinant for the covaraince matrix, resulting into the most concentrated subset. Suppose that $\boldsymbol{X}$ = (X_1,...,X_p) is the p-variate data matrix with n observations. Moreover, suppose H is a subset with h elements where h can be considered the minimum number of points which must not be outlying. For the MCD its highest possible breakdown point is at h = $\lfloor \frac{(n+p+1)}{2} \rfloor$. We then calculate the following measures:

$$T_H = \frac{1}{h}\sum_{i \in H} x_i \qquad S_H = \frac{1}{h}\sum_{i \in H}(x_i - T_H)(x_i - T_H)'$$
$$d_i^2 = (x_i - T_H)' S_H^{-1}(x_i - T_H) \qquad i = 1, ..., n \tag{1}$$

Where T_H is the sample mean, S_H is the sample covariance and d_i represents the Mahalanobis distance of i^{th} observation. Consequently, suppose H^* is constructed from h observations corresponding to h smallest d_i's from ordered distances $d_{1:n} \leq d_{2:n} \leq ... \leq d_{n:n}$ with mean T^* and covariance S^* following the same approach as in equation (1). Then the basic theorem on which the MCD is based on states that the determinant of S^* is smaller(or equal) then the one of S. This theorem is the corner stone of the C-steps, concentration steps.
The MCD algorithm takes as an input some H-subset with size h and in each C-step computes the corresponding sample mean, covariance and the Mahalanobis distances for all n observation like in equation (1). Then it constructs new h-subset corresponding the h smallest distances. This procedure with C-steps continues as long as the there is no difference in determinants of covariance matrices of new and previous C-steps or if the determinant of new covariance matrix becomes 0. One of the challenges of MCD is the way of finding the initial subset H. So, all $\binom{n}{h}$ subsets with h elements can be considered as initial subsets. However, as mentioned in the previous section, this method is computationally infeasible and time consuming. In order to approximate the MCD estimator we will therefore use the FastMCD algorithm to obtain robust estimators for location and scatter.
In FastMCD algorithm instead of looking at all possible h-subsets, we find the best l initial subsets based on twice repeated C-steps and we only use these subsets for main convergence C-steps which saves substantial computation time. To obtain this best l initial h-subsets we randomly draw a subset with p+1 elements from the data (H_0) and use this sample to construct an initial h-subset(H_1), which we use to perform C-steps for twice(to get H_2 and H_3). We repeat this for m times to get m different H_3 subsets and we pick l of those which corresponds to the lowest l determinants of corresponding covariance matrices. The reasoning behind the usage of only two C-steps for computing

initial subsets is that good and bad initial subsets can be differentiated even after few C-steps (Rousseeuw and Van Driessen, 1999). After finding these top l initial h-subsets, we perform C-steps on each of these subsets until their convergence. Finally, we choose the best H subset (H$_{final}$) which corresponds to the lowest determinant of their respective covariance matrix. The sample mean and sample covariance obtained from this subset are the MCD estimates of location and scatter, respectively.
Under a normal distribution this MCD estimator of location (T$_{MCD}$) is Fisher consistent for the mean μ. However, the MCD estimator of scatter(S$_{MCD}$) is not Fisher consistent for the covariance matrix Σ, therefore there is a need of correction. Using Fisher consistency factor c$_\alpha$ = $\dfrac{\alpha}{F_{\Gamma(p*0.5+1,1)}}$ where the F_Γ is the cumulative distribution of gamma density. Finally, we use the small sample correction c$_{np}$ (Pison et al.,2002). We then obtain the following Fisher Consistent MCD estimators for mean and covariance matrix of a normal distribution with mean μ and covariance matrix Σ, also called Raw MCD estimators:

$$\begin{aligned}\hat{\mu}_{Raw.MCD} &= T_{MCD} = \frac{1}{h}\sum_{i\in H_{final}} x_i \\ \hat{\Sigma}_{Raw.MCD} &= c_\alpha c_{np} S_{MCD} = c_\alpha c_{np}\frac{1}{h}\sum_{i\in H_{final}} (x_i - T_{MCD})(x_i - T_{MCD})'\end{aligned} \tag{2}$$

Reweighted MCD
One of the disadvantages of not using all n data points and using only h of them can lead to loss of information. If the difference between h and n is very large there is a need for trade-off between the conservative choice of h and the efficiency of the model. Therefore, we can gain some efficiency by detecting outliers with Raw MCD estimates and exclude only those from the estimation procedure. Suppose that x$_i$ $\sim$ N(μ, Σ) then d$^2(x_i, \mu, \Sigma)$ is Chi-squared distributed with p degrees of freedom where is the squared of Mahalanobis distance is: d$^2(x_i, \mu, \Sigma) = (x_i - \mu)'\Sigma^{-1}(x_i - \mu)$. d$^2(x_i, \mu, \Sigma)$. Moreover, this holds also for large sample sizes if consistent estimators for μ and Σ are used. Using this property we assign binary weights to all data points with the following weight function:

$$w_i = \begin{cases} 1 & \text{if } \mathrm{d}^2(x_i, \hat{\mu}_{Raw.MCD}, \hat{\Sigma}_{Raw.MCD}) \ \leq \ \chi^2_{1-\delta} \\ 0 & \text{if } \mathrm{d}^2(x_i, \hat{\mu}_{Raw.MCD}, \hat{\Sigma}_{Raw.MCD}) \ > \ \chi^2_{1-\delta} \end{cases} \tag{3}$$

The outliers get a weight 0 and the δ is chosen in such a way that the ellipsoid based on μ and Σ will contain (1-delta)*100 percent of all data points.
To make this estimates Fisher consistent for again use the Fisher consistency factor mentioned earlier only now with (1-δ) instead of α. Moreover, we also use a small sample correction to obtain the following robust reweighted MCD estimates for the mran and covaraince matrix: Then the reweighted MCD estimators for location and covariance matrix are determined as follows:

$$\begin{aligned}\hat{\mu}_{rwgt} &= T_{rwgt} = \frac{1}{\sum_{i=1}^{n} w_i x_i} \\ \hat{\Sigma}_{rwgt} &= c_{1-\delta} c_{np} S_{rwgt} = c_{1-\delta} c_{np}\frac{1}{\sum_{i=1}^{n} w_i}\sum_{i=1}^{n} (x_i - T_{rwgt})(x_i - T_{rwgt})'\end{aligned} \tag{4}$$

2.2 Linear Discriminant Analysis

As it was mentioned in section 1 the LDA is one of the most commonly used supervised learning techniques for data classification. Here, we model the distribution of the predictors X separately in each of the classes (i.e.Y), and then we use Bayes' theorem to flip these around into estimates for $Pr(Y = k|X = x)$. More specifically, a p-dimensional variable X following multivariate Gaussian distribution $X \sim N(\mu, \Sigma)$ with mean μ and covariance matrix Σ. For K > 1 where K represents the number of classes. LDA classifier assumes that the observations in kth class are drawn from a multivariate Gaussian distribution $N(\mu_\kappa, \Sigma)$, where μ_κ is a class-specific mean, and Σ is the covariance matrix which is common for all K classes. The derivation of the LDA decision rule can be found in the Appendix where we donote by δ_κ(x) the discriminant function becomes:

$$\begin{aligned}\delta_\kappa(x) &= -\frac{1}{2}(x-\mu_\kappa)^\tau\Sigma^{-1}(x-\mu_\kappa) + \log(\pi_\kappa) \\ &= -\frac{1}{2}(x^\tau\Sigma^{-1}x - x^\tau\Sigma^{-1}\mu_\kappa - \mu_\kappa^\tau\Sigma^{-1}x + \mu_\kappa^\tau\Sigma^{-1}\mu_\kappa) + \log(\pi_\kappa) \\ &= x^\tau\Sigma^{-1}\mu_\kappa - \frac{1}{2}\mu_\kappa^\tau\Sigma^{-1}\mu_\kappa + \log\pi_\kappa\end{aligned} \tag{5}$$

The third line of equation (9) is due to the fact that x$^\tau\Sigma^{-1}x$ does not depend on k. Finally, from the final results of equation (9) we observe that this decision of LDA depends linearly on x which is also the reason of the name of the method Linear Discriminant Analysis. In the remainder of our analysis we assume that K = 2. Then the classifier assigns an observation X = x to the class with the following way:

$$\hat{y} = \begin{cases} 0 & \text{if } \mathrm{d}_0(x) > d_1(x) \\ 1 & \text{if } \mathrm{d}_0(x) < d_1(x) \end{cases} \tag{6}$$

3 Simulation Study

In this section we perform a simulation study to evaluate the performance of classical LDA and robustified LDA, where the classical LDA is based on the standard mean and covariance matrices, and the robustified LDA is based on the reweighted location estimator T$_{rwgt}$ and reweighted scatter estimator S$_{rwgt}$ determined by FastMCD algorithm. We apply classical and robustified LDA in two different contamination settings: contamination in the predictors and contamination in class variable.

Contamination in the predictors

In the first contamination setting we apply both classical and robustified LDA to the same setting for comparison purposes. We use the following settings employed by Croux and Dehon (2001), Todorov and Pires (2007) and Yahaya et al. (2016):

- A: $n_1 = n_2 = 60, \quad \mu_1 = \mathbf{0}_p, \quad \mu_2 = \mathbf{1}_p, \quad \Sigma = I_p$
- B: $n_1 = n_2 = 100, \quad \mu_1 = \mathbf{0}_p, \quad \mu_2 = \mathbf{1}_p, \quad \Sigma = I_p$

- C: $n_1 = 140, n_2 = 60, \quad \mu_1 = \mathbf{0}_p, \quad \mu_2 = \mathbf{1}_p, \quad \Sigma = I_p$
- D: $n_1 = n_2 = 200, \quad \mu_1 = \mathbf{0}_p, \quad \mu_2 = \mathbf{1}_p, \quad \Sigma = I_p$

We use the above setting for different values of sample sizes, predictors and contamination values for both classical and robustified LDA to express any possible advantages and disadvantages of tho methods in different settings. For each method, we draw two random samples from the multivariate normal distribution with different means but with the same covariance matrix such that $\Sigma_1 = \Sigma_2 = \Sigma$. Consequenctly, we add to each of the random samples randlomly generated contamination drawn from multivariate normal distribution with mean $\widetilde{\mu_1} = [5, 5, .., 5]'_p$, $\widetilde{\mu_2} = [-4, -4, ..., -4]'_p$ respectively and covariance matrices $\widetilde{\Sigma_1} = \widetilde{\Sigma_2} = \widetilde{\Sigma} = 0.25^2 I_p$. We can summarize this for contamination level ϵ as follows:

$$\pi_1 : (1-\varepsilon)N_p(\mu_1, \Sigma_1) + \varepsilon N_p(\widetilde{\mu_1}, \widetilde{\Sigma_1})$$
$$\pi_2 : (1-\varepsilon)N_p(\mu_2, \Sigma_2) + \varepsilon N_p(\widetilde{\mu_2}, \widetilde{\Sigma}_2)$$

We then randomly divide this generated sample into training and test set with 60/40 division respectively. We also create a class variable consisting of n_1 zeros and n_2 ones. We use the generated training set for fitting the data with created LDA function. Then we use this fitted model to predict the classes of the testing data. Given the specified labels corresponding to this testing data, we are able to calculate the misclassification rate. For each iteration of our simulation we calculate this error rates for both methods and we take the average of all misclassification rates for each method over all iterations. So, for each contamination level we obtain two average misclassification rates for classical and robustified LDA respectively.

Contamination in the class variable

In the second contamination setting we once again for each iteration of the simulation randomly sample from the multivariate normal distribution with the samples sizes n_1 and n_2 and corresponding μ_1 and μ_2 but different Σ following the same specifications A,B,C and D as mentioned earlier. This time, instead of replacing the observations in these samples consisting of predictor values, we add contamination to the created class vectors $\mathrm{y}_1 = 0_{n_1}$ and $\mathrm{y}_2 = 1_{n_2}$. We do this by means of replacing a part y_1 with 1_ε (a vector of ones) and replacing y_2 for 0_ε (a vector of zeros) for the chosen contamination level ε. Furthermore, we repeat the procedure of dividing the data into training and test sets again with 60/40 division rule. Finally, we use the training set to fit the model and the testing set for the prediction to obtain the average misclassification error rate for both classical and robustified LDA. The detailed results for this simulation can be found in Table 4.

3.1 Error Rate and Classification efficiency

One of the simplest measures which evaluates the performance of a classification model is the misclassification rate which is determined as follows:

$$\text{Misclassification rate} = \frac{\text{Number of observations incorrectly classified}}{\text{Total number of observations}} \tag{7}$$

Another misclassification measure has been proposed by Croux et al.(2008) for obtaining theoretical results concerning the classification efficiency of classical and robustified LDA which measures the difference between the optimal error rate(for the uncontaminated data) and the error rate of an estimated discriminant rule(for the contaminated data). According to this study, the Error Rate for subset H$_m$, the misclassification probability when we work with an estimated discriminant rule can be obtained as follows:

$$ER(H_m) = \pi_1 \Phi(\frac{\theta}{\Delta} - \frac{\Delta}{2}) + \pi_2 \Phi(-\frac{\theta}{\Delta} - \frac{\Delta}{2}) \tag{8}$$

Where the π_1 and π_2 are the prior probabilities corresponding to 2 groups, $\theta = \log(\pi_2/\pi_1)$ and $\Delta = \sqrt{(\mu_1 - \mu_2)'\Sigma^{-1}(\mu_1 - \mu_2)}$. Moreover, the Φ represents the cumulative distribution function of univariate standard normal. The ER corresponding to non-contaminated data we define as optimal Error Rate (ER$_{opt}$) since for any other contaminated data sample the ER will be always larger than this value. We then calculate for each ε level and for corresponding simulation we can calculate the Error Rate (ER$_\varepsilon$). Consequently, the expected Loss in error rate, loss of efficiency is defined as follows:

$$\overline{Loss_\varepsilon} = \frac{1}{m}\sum_{i=1}^{m} ER_\varepsilon^i - ER_{opt} = \overline{ER_\varepsilon} - ER_{opt} \tag{9}$$

Finally, we define the finite sample relative classification efficiency of the robustified LDA with respect to classical LDA which can be used to compute the relative efficiency of both methods with the following way:

$$RCE_\varepsilon(Robust, Classical) = \frac{Loss_\varepsilon(Classical)}{Loss_\varepsilon(Robust)} \tag{10}$$

3.2 Simulation Results

Contamination in the predictors
Table 3 and present the 100 simulation results for different settings described in section 3 for the approach of contamination in the predictors. From the first column corresponding to the $\varepsilon = 0$, we observe that the misclassification rate of the classical LDA is smaller compared to the robustified LDA. This holds for all three values of predictors (p=3,6,10). In case there is 5 or 10 percent contamination in the data there is large difference between the misclassification rates of two methods for all four settings.
Namely, the error rate of robusified LDA is almost twice as small as the one of the classical LDA for the settings B and C at the contamination level 0.05. Moreover, this is also the case for $\varepsilon = 0.10$ for the settings A, B and D. However, when the contamination level increases to 20% then the difference between the error rates of classical and robustified LDA becomes smaller, even though it still holds that robustfied LDA outperforms the classical LDA in all four settings. When there is very large amount of contamination in the data then the robustified LDA still performs better or equally well as classical LDA for the settings A, B and D for all three diffrent predictor amounts. However, this doesn't hold for the setting C at 40% contamination levels. Moreover, we observe that as ϵ increases, the error rates of both methods increases, but for high contamination levels the robust error rates become very similar to the classical error rates. From these simulation results we can conclude that when there is no contamination in the data the classical LDA outperforms robustified LDA independent of the number of observations in two classes

and number of predictors. When there significant amount of contamination present in the predictors than the robustified LDA outperforms classical LDA for small and also for large samples. However, when contamination level is very large then robustified LDA does slightly better then the classical LDA and even worse when the sizes of two groups differ(setting C).
In Table 3 it is observed that for non contamination level ($\varepsilon < 0$) or ($\varepsilon = 0$) the standard LDA method performs better than the robust one. When the contamination level increases, the differences between them become smaller and in this case robust LDA performs better than standar LDA.
Contamination in the class variable
Table 2 represents the simulation results where the contamination has been added to the class variable. We observe again that the classical LDA outperforms robustified LDA when there is no contamination in the data. For all three different amounts of predictors (p = 3, 6, 10) the misclassification rates of the robustified LDA, in majority of cases, is larger than the ones of classical LDA which means that the robustified LDA doesn't perform better than classical LDA even if there is contamination in the class variable. What we also observe is that for the setting A,B where the total sample size is small then the error rates is large and these error rates gets smaller as the sample size increases, as it is the case for setting C and D. The reason for this results can be explained by the fact that the robustified LDA in our analysis is based on the location and scatter estimates determined by FastMCD algorithm which intends to detect and take care for the outliers present in the data consisting of predictors. Therefore, in this second approach where contamination has been added to the class variable, the robustified LDA based on MCD estimators doesn't perform better than the classical LDA. Thus, we can conclude that the robustified LDA outperforms classical LDA when there is contamination in the predictor data but it underperforms when there is contamination in the class variable. Classical LDA also outperforms robustified LDA when there is not contamination in the data.

4 Swiss Bank Data

The dataset consists of measurements from 200 old Swiss bank notes: 100 genuine and 100 counterfeit. The goal is to classify Swiss 1000 franc bank notes as counterfeit or genuine based on different measurements of the bank note. These measurements are the six predictors which are the length of the note, the left edge width, the right edge width, the bottom margin width, the top margin width, and the length of diagonal. All measurements are in millimetres.

4.1 Empirical Results

In this part of our analysis we apply the classical and robustified LDA on the Swiss bank note data. Table 4 and Table 5 presents the pooled covariance matrices of the classical and robustified LDA respectively. Moreover, they also present the group means corresponding to both methods. When comparing these tables we observe that both group means and pooled covariance matrices differ for two methods. As it was mentioned in section 2.1, reweighted MCD estimates are based on the weights assigned to the variables using the

Mahalanobis distances and the threshold of $\chi^2_{1-\delta}p$. In FastMCD algorithm, weights 0 are assigned to the outliers and 1 to non-outliers.
Therefore, when applying the algorithm on Swiss Bank data with $\delta = 0.025$, m = 500, l = 10 and $\alpha = 0.6$ we observe in total of 25 outliers. Figure 1 shows the Mahalanobis distances for 200 bank notes, 100 genuine notes are represented with blue color while 100 counterfeit notes are represented by red color. The horizontal line corresponds to the threshold value $\chi^2_{1-\delta} = 14.45$ for 6 degree of freedom. As we can see from the figure there are outliers present in this data set. Moreover, we observe that there are more outliers from "counterfeits" class than from the "genuine" class. The presence of these outliers explain the reason why the earlier results of classical LDA and robustified LDA were different because if there are no outliers in the data, ideally both LDA results should not differ significantly which was the case as we mentioned earlier.
In order to evaluate the performance of prediction, we split the data into a training set and a test set with different size and compare the misclassification rate of classical and robust LDA results. Repetition of this process and taking the average are necessary to reduce variability from the random split. Table 6 presents the result of this cross validation. When the size of the training set decreases, misclassification rates of both LDA become smaller as expected because the training set contain more information and variation to better predict the test set. It's also noticeable that the robust LDA rates are higher than the standard LDA rates at size 30% and 40% while robust LDA rates are smaller at the size 50% and higher. This is due to the fact that with smaller size there are lower chances that the training set contains the outliers since there percentage of outliers is 0.125 over all sample.

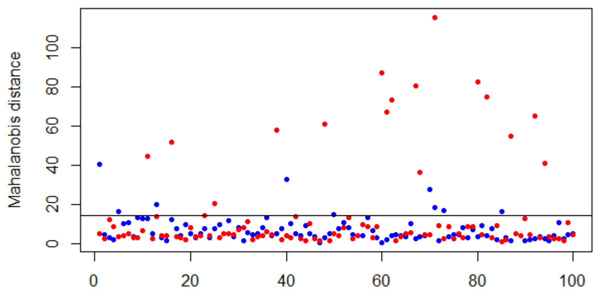

Figure 1: Mahalanobis distances of Swissbank data

As it was mentioned earlier there is always a trade off between robustness and efficiency of the method. In order to calculate the relative efficiency of two classification methods, the classical and the robustified LDA we propose to to use the Loss function from section 3.1. For this we use the LDA classification results for both methods and we calculate the corresponding ER rates. As optimal ER rate we propose to use the ER rate of Table 3 corresponding to p = 6 and setting A for the$\varepsilon = 0$ for both methods. So, the ER$_{opts}$ = 0.0878 and the ER$_{optr}$ = 0.0965 . Then we use the LDA results of the standard and robustified methods from Table 4 to obtain the ER$_s$ = 0.1068and the ER$_r$ = 0.1010. The RCE is then obtained as follows:

$$RCE_{\varepsilon}(Robust, Classical) = \frac{ER_s - ER_{opts}}{ER_r - ER_{optr}} = \frac{0.1068 - 0.0878}{0.1010 - 0.0965} = 0.95 \qquad (11)$$

What we observe is that the loss of efficiency in robustified LDA is larger than the loss in classical LDA since the RCE value is 0.95 because denominator is larger than the nominator. Therefore, we can conclude in the presence of the contamination level there is always as efficiency loss in the robustified LDA.
For loss function, AdaBoost is an iterative process in which each step selects a weak classifier h_t, which minimizes:

$$Z_t = \sum D_i \exp(-y_i h_t(x)) \tag{12}$$

where regarding to Singer and Shapire D_i is the weight on example i at step t, x_i is the example, h_t is a confidence rated binary classifier and $y_i \epsilon -1, 1$. One disadvantage of AdaBoost is that it minimizes a quantity related to classification error but actually does not minimize the number of false negatives. We can formalize this approach of an asymmetric loss function as follows:

$$ALoss(i) = \begin{cases} \sqrt(k) & \text{if } y_i = 1 \text{ and } C(x_i) = -1 \\ \frac{1}{\sqrt(k)} & \text{if } y_i = -1 \text{ and } C(x_i) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where C_{x_i} is the class assigned by the boosted classifier and the false negatives "cost" (k times) more than the false positives.The bound on the asymmetric loss is:

$$\exp(-y_i \sum h_t(x_i)) \exp(y_i \sqrt(k)) \geq ALoss(i) \tag{14}$$

Finally, the minimization of this bound could be achieved using the AdaBoost method by pre-weighting each time.

5 Conclusion

According to the above analysis in simulation, we can conclude that FastMCD algorithm provides better estimates for the LDA in the presence of outliers. However, the robustified LDA which uses these estimates from FastMCD only perform well when the outliers are in predictor variables and not when the outliers are in the class variable. Therefore, the advantage of robustified LDA is the better performance in case of contamination in the explanatory variables and the disadvantage is the poorer performance in case of contamination in the class variable as compared to classical LDA. However, by the empirical results on Swiss bank notes data, there is also a trade-off between efficiency and robustness, that is the robust LDA has a higher loss in efficiency than classical LDA.

6 References

Cook, R. D., Hawkins, D. M., Weisberg, S. (1992). Comparison of model misspecification diagnostics using residuals from least mean of squares and least median of squares fits. Journal of the American Statistical Association, 87(418), 419-424.

Croux C., Filzmoser P. and Joossens K. (2008). Classification Efficiencies for Robust Linear Discriminant Analysis, 18(2), 581-599.

Leroy, A. M., Rousseeuw, P. J. (1987). Robust regression and outlier detection. Wiley Series in Probability and Mathematical Statistics, New York: Wiley.

Pison G.,Van Aelst S., and Willems G.(2002). Small sample corrections for LTS and MCD, 55(1) 111–123
Rousseeuw, P. J. (1984). Least median of squares regression. Journal of the American statistical association, 79(388), 871-880.

Rousseeuw, P. J., Driessen, K. V. (1999). A fast algorithm for the minimum covariance determinant estimator. Technometrics, 41(3), 212-223.

Todorov V., Pires A.M.(2007). Comparative Performance of Several Robust Linear Discriminant Analysis Methods, 5(1), 63-83.

Yahaya S.S.S, Yai-Fung L., Hazlina A. and Zurni O.,(2016).Robust Linear Discriminant Analysis, 12(4), 312-316.

7 Appendix

7.1 Derivation of the decision rule

The multivariate Gaussian distribution:

$$f(x \mid Y = \kappa)) = \frac{1}{(2\pi)^{p/2} \mid \Sigma \mid^{1/2}} \exp(-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)) \tag{15}$$

Where the covariance matrix Σ can be estimated by pooled covariance matrix :

$$\hat{\Sigma} = \frac{(n_0 - 1)\hat{\Sigma_0} + (n_1 - 1)\hat{\Sigma_1}}{n_0 + n_1 - 2} \tag{16}$$

and the priors are estimated as follows:

$$\hat{\pi}_1 = \frac{\hat{n}_0}{n}, \hat{\pi}_2 = \frac{\hat{n}_1}{n} \tag{17}$$

We plug in the $f_\kappa(X = x)$ for kth class into the Bayes' Theorem:

$$\begin{aligned} P_\kappa(x) = \frac{\pi_\kappa f_\kappa(x)}{\sum_{l=1}^{\kappa} \pi_l f_l(x)} &= \frac{\pi_\kappa \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp(-\frac{1}{2}(x-\mu_\kappa)^\tau \Sigma^{-1}(x-\mu_\kappa))}{\sum_{l=1}^{\kappa} \pi_l \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp(-\frac{1}{2}(x-\mu_l)^\tau \Sigma^{-1}(x-\mu_l))} \\ = \frac{\pi_\kappa \exp(-\frac{1}{2}(x-\mu_\kappa)^\tau \Sigma^{-1}(x-\mu_\kappa))}{\sum_{l=1}^{\kappa} \pi_l \exp(-\frac{1}{2}(x-\mu_l)^\tau \Sigma^{-1}(x-\mu_l))} \end{aligned} \tag{18}$$

$$\begin{aligned} \log(P_\kappa(x)) &= \log(\frac{\pi_\kappa \exp(-\frac{1}{2}(x-\mu_\kappa)^\tau \Sigma^{-1}(x-\mu_\kappa))}{\sum_{l=1}^{\kappa} \pi_l \exp(-\frac{1}{2}(x-\mu_l)^\tau \Sigma^{-1}(x-\mu_l))}) \\ &= \log(\pi_\kappa) - \frac{1}{2}(x-\mu_\kappa)^\tau \Sigma^{-1}(x-\mu_\kappa) - \log(\sum_{l=1}^{\kappa} \pi_l) - \log(\sum_{l=1}^{\kappa} \exp(-\frac{1}{2}(x-\mu_l)^\tau \Sigma^{-1}(x-\mu_l))) \end{aligned} \tag{19}$$

Where the log(P$_\kappa$) is the δ_κ(x), discriminant function of LDA. The last two terms are constant as they do not depend on k class,they could be eliminated from the formula due to the fact that they are the same for every class k.

7.2 Tables

p=3						
Configuration		$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.4$
A	standard	0.1950	0.3625	0.5500	0.4750	0.3750
	robust	0.2079	0.2625	0.2771	0.3688	0.3000
B	standard	0.2009	0.2994	0.5038	0.5498	0.4144
	robust	0.2055	0.2238	0.2700	0.3400	0.3419
C	standard	0.1738	0.3212	0.3238	0.2800	0.2250
	robust	0.1801	0.2088	0.2638	0.3463	0.2950
D	standard	0.1944	0.2188	0.5188	0.5938	0.4188
	robust	0.1967	0.2125	0.2625	0.3625	0.3375
p=6						
Configuration		$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.4$
A	standard	0.1339	0.2083	0.2417	0.3208	0.3750
	robust	0.1583	0.1250	0.1917	0.2500	0.3333
B	standard	0.1200	0.3628	0.4170	0.4545	0.5107
	robust	0.1306	0.1651	0.2172	0.3247	0.4208
C	standard	0.1211	0.3147	0.3237	0.3920	0.3617
	robust	0.1215	0.1542	0.1917	0.3345	0.3230
D	standard	0.1138	0.3276	0.5236	0.5158	0.4124
	robust	0.1166	0.1594	0.2034	0.4310	0.3392
p=10						
Configuration		$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.4$
A	standard	0.0742	0.4167	0.4375	0.4792	0.4375
	robust	0.0981	0.1667	0.2083	0.5000	0.3958
B	standard	0.0710	0.3250	0.6375	0.5500	0.4000
	robust	0.0860	0.1125	0.1875	0.4625	0.3000
C	standard	0.0631	0.3500	0.4000	0.3625	0.1875
	robust	0.0680	0.1625	0.2250	0.4625	0.2375
D	standard	0.0609	0.31875	0.53125	0.37500	0.35625
	robust	0.0634	0.10000	0.18125	0.38125	0.30000

Table 1: Misclassification rate of the simulation results with predictor contamination

p=3						
Configuration		k = 0	k = 0.05	k = 0.1	k = 0.2	k = 0.4
A	standard	0.2065	0.2406	0.2644	0.3550	0.4971
	robust	0.2102	0.2492	0.2763	0.3619	0.4869
B	standard	0.1991	0.2295	0.2663	0.3365	0.4693
	robust	0.2026	0.2394	0.2709	0.3436	0.4733
C	standard	0.1733	0.2288	0.2593	0.3338	0.4697
	robust	0.1781	0.2310	0.2638	0.3343	0.4753
D	standard	0.1938	0.2318	0.2599	0.3246	0.4688
	robust	0.1946	0.2347	0.2612	0.3264	0.4633

p=6						
Configuration		k = 0	k = 0.05	k = 0.1	k = 0.2	k = 0.4
A	standard	0.1296	0.1792	0.2171	0.3191	0.4944
	robust	0.1518	0.1469	0.1898	0.2673	0.3750
B	standard	0.1198	0.1584	0.2065	0.3029	0.4803
	robust	0.1293	0.1710	0.2133	0.3071	0.4823
C	standard	0.1078	0.1536	0.1939	0.2836	0.4815
	robust	0.1183	0.1624	0.1998	0.3006	0.4815
D	standard	0.1197	0.1537	0.1970	0.2771	0.4519
	robust	0.1223	0.1557	0.1981	0.2782	0.4589

p=10						
Configuration		k = 0	k = 0.05	k = 0.1	k = 0.2	k = 0.4
A	standard	0.0744	0.1319	0.1898	0.2971	0.4827
	robust	0.0992	0.1656	0.2121	0.3215	0.4900
B	standard	0.0645	0.1209	0.1748	0.2763	0.4756
	robust	0.0769	0.1321	0.1851	0.2863	0.4728
C	standard	0.0598	0.1149	0.1649	0.2743	0.4711
	robust	0.0696	0.1190	0.1824	0.2863	0.4740
D	standard	0.0659	0.1084	0.1575	0.2536	0.4472
	robust	0.0677	0.1087	0.1576	0.2572	0.4532

Table 2: Misclassification rate of the simulation results with class contamination

p=3						
Configuration		$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.4$
A	ERs	0.1828	0.3781	0.4349	0.3763	0.2765
	ERr	0.1820	0.1970	0.2023	0.2023	0.1974
B	ERs	0.1860	0.4083	0.4594	0.3845	0.4680
	ERr	0.1818	0.2006	0.1948	0.2003	0.1976
C	ERs	0.1266	0.2896	0.2831	0.2843	0.2304
	ERr	0.1358	0.1599	0.1697	0.1731	0.1957
D	ERs	0.1932	0.4154	0.4747	0.4004	0.2706
	ERr	0.1913	0.1915	0.1934	0.1997	0.1994
p=6						
Configuration		$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.4$
A	ERs	0.0965	0.3537	0.3885	0.3456	0.2327
	ERr	0.0990	0.1070	0.1119	0.2504	0.2653
B	ERs	0.0878	0.2525	0.3103	0.3815	0.4212
	ERr	0.0965	0.1055	0.1118	0.2563	0.5872
C	ERs	0.0930	0.2871	0.2815	0.2975	0.2088
	ERr	0.0924	0.0940	0.0949	0.2098	0.4670
D	ERs	0.1040	0.2978	0.2613	0.3215	0.4405
	ERr	0.1050	0.1082	0.1131	0.3076	0.4404
p=10						
Configuration		$\varepsilon = 0$	$\varepsilon = 0.05$	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.4$
A	ERs	0.0409	0.2961	0.3588	0.2962	0.2054
	ERr	0.0466	0.0978	0.1149	0.2506	0.0895
B	ERs	0.0479	0.3464	0.4422	0.3490	0.2288
	ERr	0.0489	0.0472	0.0561	0.2864	0.0609
C	ERs	0.0514	0.5191	0.6298	0.5515	0.3460
	ERr	0.0540	0.0437	0.1731	0.4109	0.1095
D	ERs	0.0521	0.3802	0.4120	0.3561	0.2558
	ERr	0.0548	0.0507	0.0717	0.3252	0.1196

Table 3: Error rates of the simulation results with predictor contamination

PooledCov	Length	Left	Right	Bottom	Top	Diagonal
Length	0.1371	0.0447	0.0406	-0.0217	0.0169	0.0085
Left	0.0447	0.0988	0.0663	0.0163	0.0185	-0.0240
Right	0.0406	0.0663	0.1076	0.0198	0.0153	0.0052
Bottom	-0.0217	0.0163	0.0198	0.8472	-0.3768	0.1191
Top	0.0169	0.0186	0.0154	-0.3768	0.4128	-0.0487
Diagonal	0.0085	-0.0241	0.0052	0.1191	-0.0486	0.2555
Means	Length	Left	Right	Bottom	Top	Diagonal
Group1	214.823	130.300	130.193	10.530	11.133	139.450
Group2	214.969	129.943	129.720	8.305	10.168	141.517

Table 4: Classical LDA results on Swiss bank notes data

PooledCov	Length	Left	Right	Bottom	Top	Diagonal
Length	0.117725	0.049012	0.043662	0.006851	0.029609	0.016807
Left	0.049013	0.095276	0.068131	0.054168	0.009702	0.001083
Right	0.043662	0.068131	0.103868	0.026235	0.017966	0.017941
Bottom	0.006851	0.054168	0.026234	0.631860	-0.390560	-0.048828
Top	0.029609	0.0097020	0.017966	-0.390560	0.434101	-0.012981
Diagonal	0.016807	0.001083	0.017941	-0.048828	-0.012981	0.153605
Means	Length	Left	Right	Bottom	Top	Diagonal
Group1	214.9934	129.9264	129.7022	8.276923	10.17253	141.5527
Group2	214.7744	130.2622	130.1780	10.879268	11.10122	139.6171

Table 5: Robustified LDA results on Swiss bank notes data

Training size	30%	40%	50%	60%	70%	80%
Standard	0.0087	0.0069	0.0060	0.0057	0.0067	0.0050
Robust	0.0186	0.01167	0.0053	0.0045	0.0067	0.0000

Table 6: Misclassification rate of empirical LDA results on Swiss bank notes data