Project - Exploring machine learning methods

Modern Methods of Statistical Learning SF2935

Karl Wallström
19960909
karlwall@kth.se

Ismail Jattioui
19970402
jattioui@kth.se

September 2019

Contents

1 Introduction

In this lab, we will explore and learn how and most importantly when to use LDA, QDA, and logistic regression when it comes to classification. In the first part, a study of the auto data set will be done, our goal will be to classify the fact that we are higher than the median mileage using the different models cited earlier. Then, an analysis of the LDA and QDA will be done, to understand what happen when we violate the inherent assumptions of the model. Finally the third part, is a demonstration of the consequences of the curse of dimensionality on these models and a list of motivated configurations when we should or shouldn't use one or another of these methods. The project was done in R. Notable packages used was MASS and DMwR. MASS provided us with functions lda and qda used for discriminant analysis. DMwR provided the function knn which was used for kNN modelling. Logistic regression was performed using the built in function glm.

2 Part A

2.1 a)

First, we start by loading the auto dataset, and we add a new variable, that represents the fact that we are either higher or lower than the median of mileage, we let this variable be named *HL mpg*(we will keep the same notation for the rest of the report). After this manipulation we need to remove the mileage attribute from the dataset or we would have a redundancy in the data, since the attribute HL mpg and mpg represents the same concept, therefore we shouldn't keep both. If this variable was still present in the training data any model we use will have a low error rate since the mpg encapsulate all the information we need to compute the HL mpg. Best case scenario : we don't learn anything new. Worst case scenario : we draw false conclusions about the model.

2.2 b)

We wish to know which of the features in the auto dataset that are likely to be useful for predicting how a car is classified. Therefore, we will proceed to a graphical study of the different features against the *HL mpg* feature. We have excluded the feature "name" because we won't learn anything new since each car has a different name and any correlation between the "name" and the "mileage" would only be fortuitous in this case (maybe if we had a "brand" feature it would have been more interesting). Furthermore, we have normalized the data using the mean and the standard deviation. We choose to standardize the data since for instance kNN is going to be used, we want to avoid clusters being heavily influenced by one large feature.

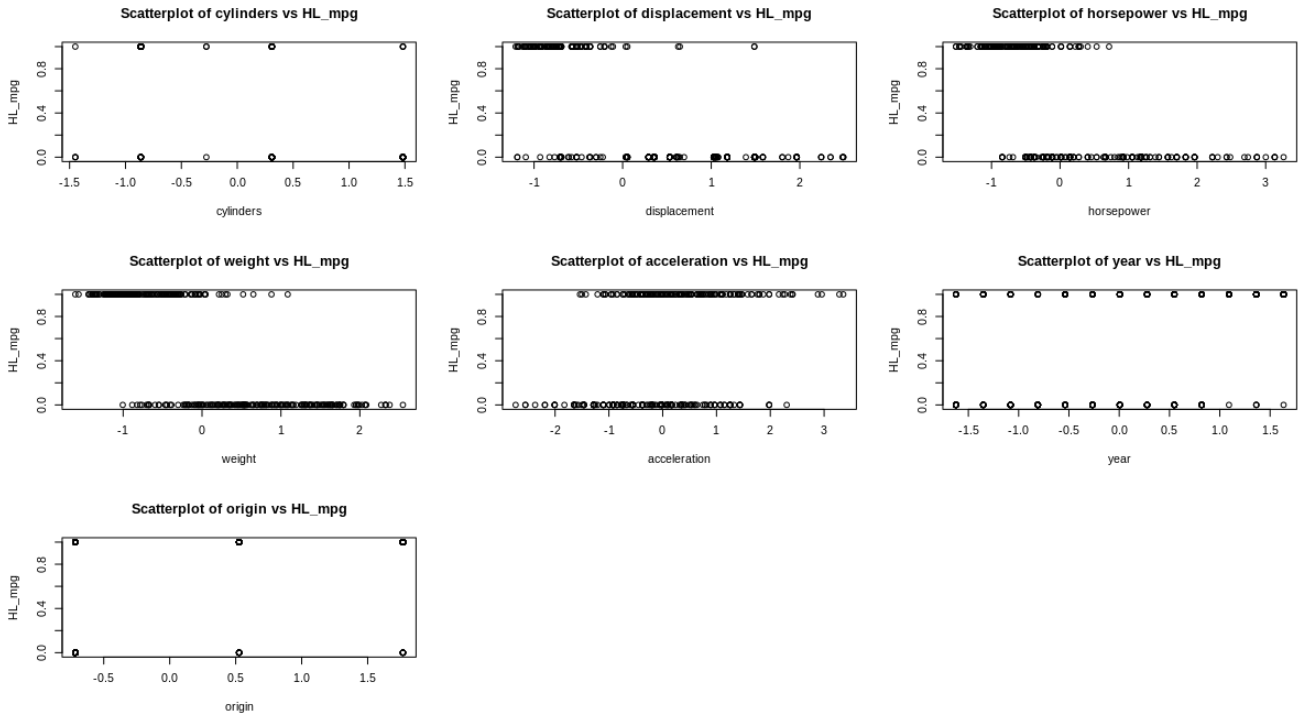

Figure 1: Possible features plotted against HL mpg

We can see that the features cylinder, origin and year doesn't give us much information about the HL mpg, because each input we have a simultaneous true and false response.

For the displacement, we can't draw conclusions in the range of [-2, 0] which is half the data, because again for each input we have a simultaneous true and false response. The same assessment can be made for the acceleration the only difference is the overlapping range which is [-1, 1] in this case.

Finally, for the horsepower and the weight features, the overlapping range isn't too "wide" and seems to follow a nice sigmoid function which could be good for predictions. Therefore, we have chosen these two attributes to train our models.

2.3 c)

From the original auto dataset that contains 392 observations, we have split it into a training set of 235 observations (so 2/3 of the observations) and the rest went to a testing set of 157 observations.

Since we are sampling the data for training, we can't simply draw conclusions about the error rate with only one simulation, so we run the simulation several times (in our case 500) and computed the mean of the error rate for each method with a fixed ratio of (2/3) for the training set. For kNN we let k=8. The errors are computed as mean square errors but multiplied by 100.

	lda error rate	qda error rate	lr error rate	knn8 error rate
Train	12,52262	13,15088	11,01834	10,19279
Test	12,75033	13,44280	11,58515	12,00821

Table 1: Mean error rate for each model

Finally for kNN we also computed training and test errors with a fixed ratio of 2/3 but where we let k vary from 1 to 200 by increments of 1. Both the training and test error increases with k overall. Letting k=1 we obviously get the lowest training error but ignoring k=1 means that both the test and training errors are minimized by k=4 in this case. We see from the graph that examining k above approximately 15 either results in no change of errors or increasing errors so the relevant range for k is quite small.

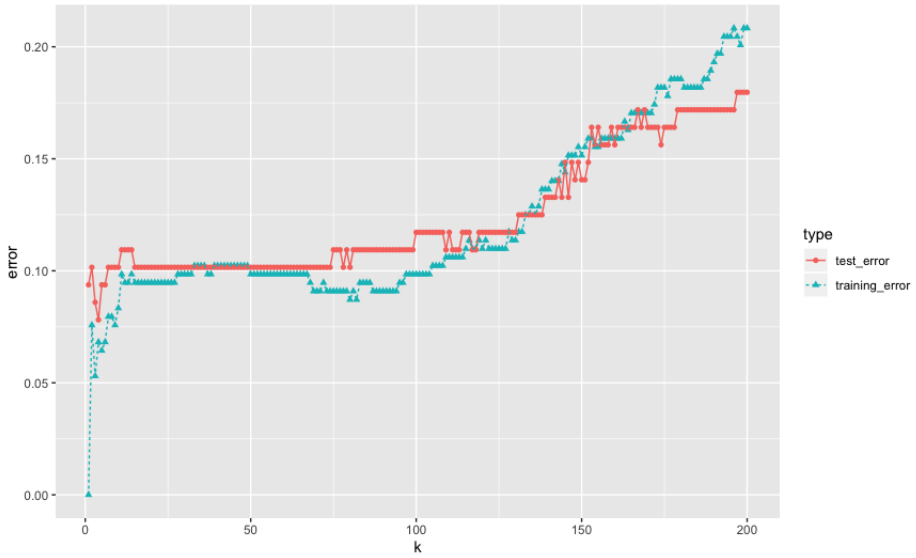

Figure 2: Train/test error for different k

2.4 d)

In this section we computed test errors for the models using different test/training splits. The errors are computed as mean square errors. The result of this can be seen in the plot below.

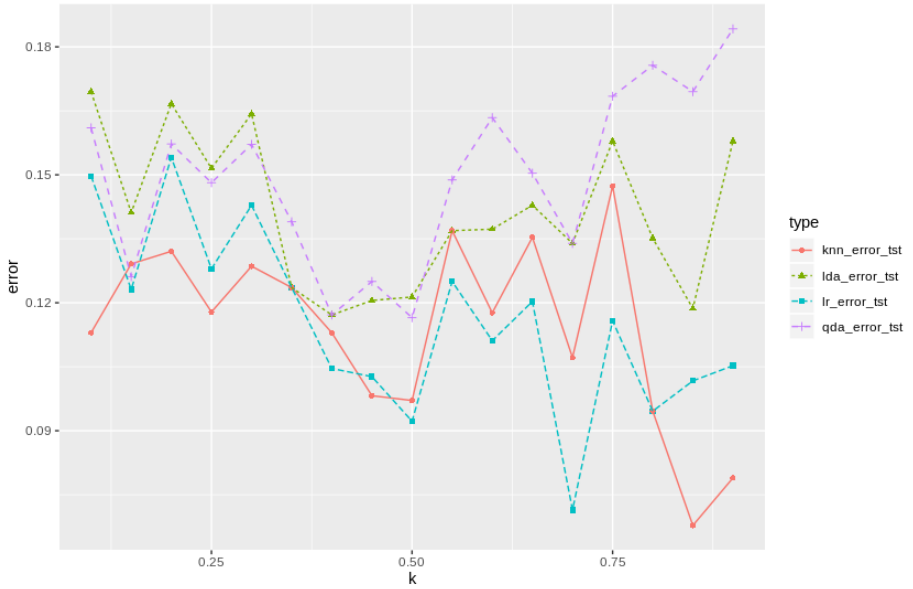

Figure 3: Test error rates when training models for different ratio of training/test set

For LDA and QDA when the ratio is roughly half training, half test, the error rate is at it minimum. In the same fashion, when the ratio is in favor of the training or the test set, the error rate is at it maximum. For high ratio of the training set, QDA performs worse than LDA. For small ratio of the training set, the LDA performs worse than QDA.

For the logisitic regression, in general we see that the error rate is decreasing when the ratio of the training set increase and when we are at more than 40% of the training set, it has better error rate than the 3 other methods.

For the knn8 , the error rate is at it minimum when the ratio is less than half the training set, after that the error rate rise progressively as the ratio increase.

3 PartB

Throughout this part we will employ an 80% training/test data ratio. All errors referred to are computed as mean square errors.

3.1 a)

First we let N=300 and M=125. Next we let s1 be the sample of size M. s1 contains points generated by a Gaussian distribution on $\mathbb{R}^2$ where the first component is sampled from an N(2,1) distribution and the second component is N(1,4). The components are sampled independently. We also let s2 be the sample of size N-M, s2 contains points generated by a Gaussian distribution on $\mathbb{R}^2$ where the first component is sampled from a N(0.25,0.25) distribution and the second from N(2,0.5), they are sampled independently. Next we compute training and test mean square error for LDA, QDA and logistic regression which we see in the table below.

	lda error rate	**qda error rate**	**lr error rate**
Train	0.06557377	0.02049180	0.05737705
Test	0.10714286	0.03571429	0.08928571

Table 2: Error rates when using s1 and s2

All of the models perform well but QDA is the best model with a test error almost a third of the other test errors of logistic regression and LDA. Examining the plot of our two samples in figure 4 we see that it is quite a clear separation between the two samples so it is not surprising our models perform well. Theoretical reasons for the performance of the models will be discussed in the analysis in 3.4.

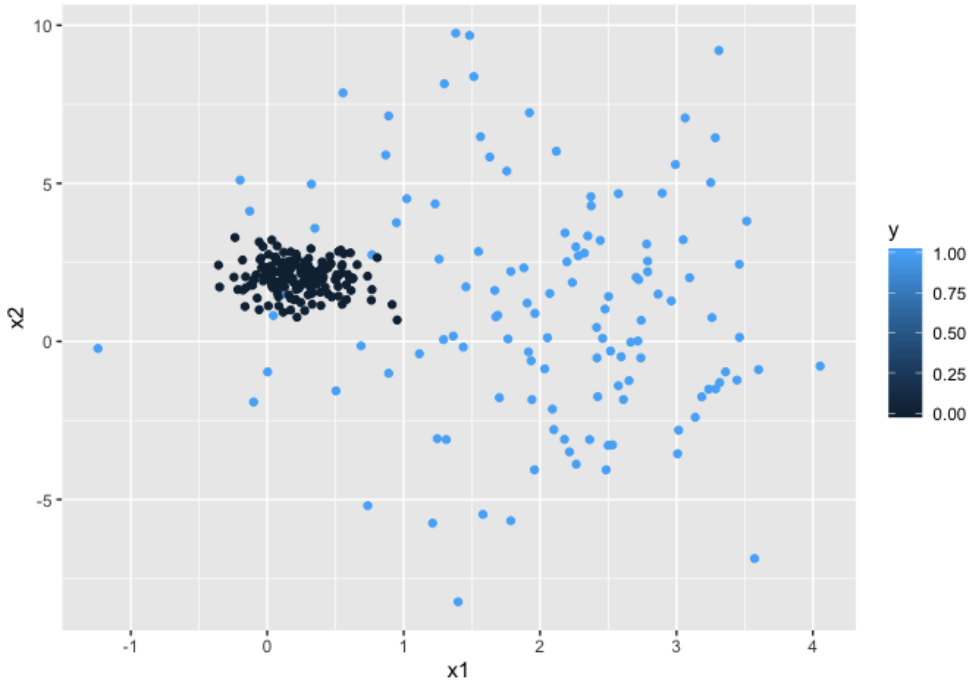

Figure 4: Black dots represent points from s2 and blue dots from s1

Next we want to find out how our choice of distribution parameters affects our results. If we want to make the problem as hard as possible for all models then we just let the parameters be the same. We want to separate the classes using the difference in the respective distributions but if they are the same this is not possible. So generally the closer the parameters of s1 are to s2 then the harder it will be for the models. To make the problem easy we can pick our parameters such that the two clusters of points are linearly separable, for instance by letting the distributions have very different means but small

variances. Another interesting choice of parameters is to let the means be similar for s1 and s2 but where s1 and s2 have very different variances, these parameters generate a small cluster located within a larger spread out cluster.

We want to examine this last case so we create a new sample t1 with components sampled from a N(2,20) distribution and a N(1,20) distribution and for the second sample we use s2 as is. The errors are presented in the table below and we see in this case that QDA vastly outperforms the other models. It seems as if both LDA and logistic regression suffers when the variances of one sample is much larger than the other sample. To summarize we see that the closer s1 is to s2 the harder it is for all models to predict but LDA and logistic regression perform worse when the variances differ a lot between the samples. We have not discussed the impact of introducing covariance between the components in s1 or s2 but it should not produce any radical new insights since covariance geometrically will just rotate our clusters.

	lda error rate	**qda error rate**	**lr error rate**
Train	0.32377049	0.00000000	0.32377049
Test	0.37500000	0.01785714	0.35714286

Table 3: Error rates when using t1 and s2

3.2 b)

Now we let s_1 consist of components where the first one is drawn from an exponential distribution with $\lambda = 15$ and the second is drawn from a normal distribution N(1,2). Next we let s_2 consist of components where the first component is drawn from an exponential distribution with $\lambda = 2$ and the second is drawn from a normal distribution N(2,0.11). Below we plot our sample and the errors after fitting our models can be seen in the table below.

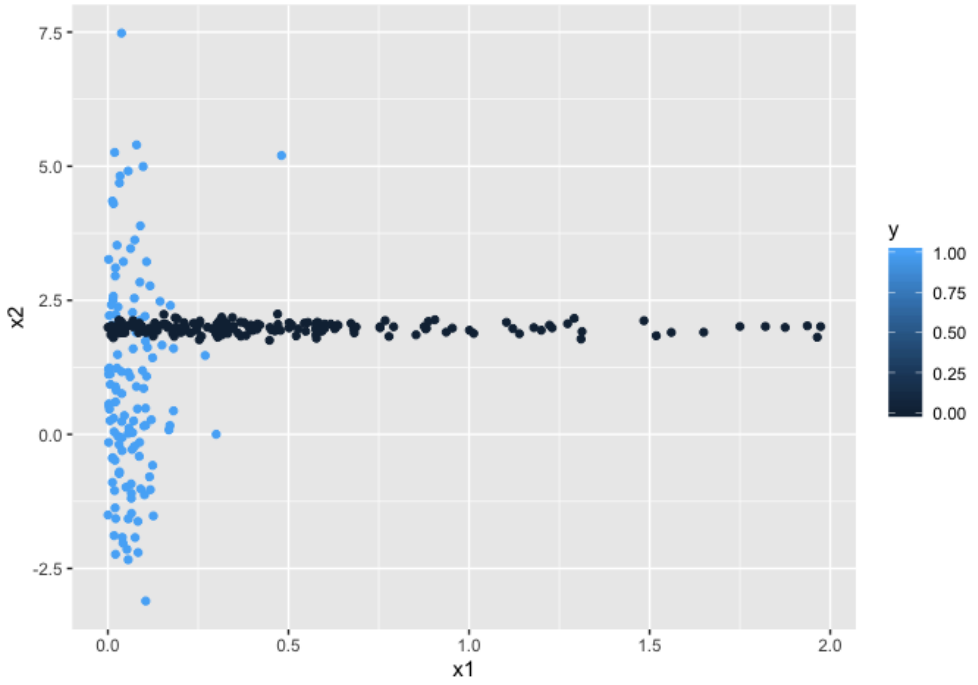

Figure 5: Black dots represent points from s2, blues dots points from s1

	lda error rate	**qda error rate**	**lr error rate**
Train	0.19672131	0.02459016	0.18032787
Test	0.23214286	0.03571429	0.28571429

Table 4: Error rates when using s1 and s2

Similarly as in a) QDA outperforms LDA and logistic regression. Logistic regression has the worst test error but it is quite close to the LDA test error.

3.3 c)

Now we let s_1 consist of components where the first component is drawn from an exponential distribution with $\lambda = 15$ and the second component is drawn from a Poisson distribution with $\lambda = 4$. Next we let s_2 consist of components where the first component is drawn from an exponential distribution with $\lambda = 2$ and the second is drawn from a Poisson distribution with $\lambda = 2.7$. Below we plot our sample and the errors after fitting our models can be seen in the table below.

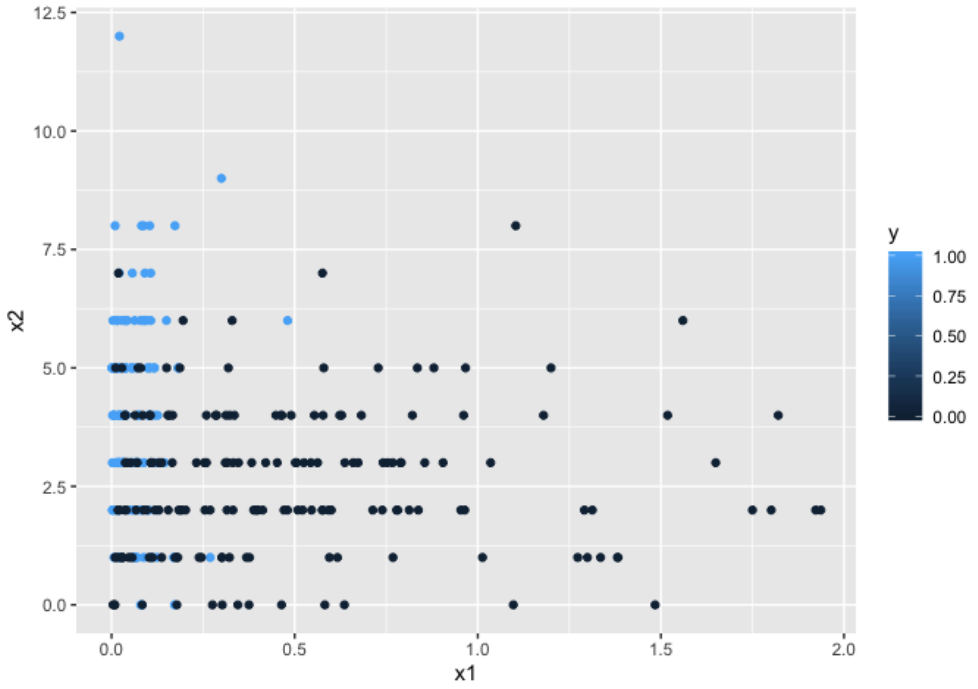

Figure 6: Black dots represent points from s2, blues dots points from s1

	lda error rate	qda error rate	lr error rate
Train	0.1844262	0.1844262	0.1680328
Test	0.2142857	0.1785714	0.2500000

Table 5: Error rates when using s1 and s2

Just as in b) QDA once again performs the best, LDA second and logistic regression last although in this case all models perform worse, all test errors are in the interval $[0.17, 0.25]$.

3.4 Analysis of the result

Since we are looking at a classification problem the parameters we choose for our distributions are very important. As explained in a) if the parameters for the two sample are the same then our samples are misleading and if the parameters are different enough the two sample clusters will be linearly separated and then we have a trivial task ahead of us. So in each case of our three cases we need to consider that the choice of parameters essentially decides how successful our models will be. Theoretically the case of b) more heavily violates the assumptions of LDA compared to in a) but we can still pick parameters in a) such that LDA performs better in b). So comparing errors for one method in the different cases can be misleading.

One common theme through all our cases is that QDA outperforms LDA and logistic regression. Both LDA and QDA assumes that the classes are drawn from normal distributions but LDA assumes a common covariance matrix between the classes while QDA allows each class to have its own matrix. What this in practice means is that QDA will employ a quadratic decision boundary while LDA will use a linear decision boundary. So essentially QDA has more options in how it wants to separate our clusters so its performance is not surprising. It should also be mentioned that if the assumptions of LDA holds then LDA should approximate the bayes decision boundary better than QDA and in turn produce a better model. The assumptions of LDA never holds in any of our cases.

Another interesting pattern in our tests is that both LDA and logistic regression produce similar results in each case. One explanation for this can be that both logistic regression and LDA result in linear decision boundaries since the log odds of both methods are linear. Furthermore, this means that the explanation above regarding why QDA performs better than LDA also applies to logistic regression in terms of flexiblity of decision boundaries. The difference between LDA and logistic regression lies in how the coefficients are estimated. To illustrate the differences and similarities between the models we plot the decision boundaries generated by each model in a) when using s1 and s2 and also for a new case with samples b1 and b2 where all components have variance 1 but b1 has mean of (1,1) and b2 has a mean of (3,3). Unfortunately we weren't able to make the plot assign different colours to the sample depending on their classification but this can at least be seen in figure 4 for the first plot, either way the plots illustrate our discussion nicely. The first set of decision boundaries shows the flexibility of QDA and the similarity between LDA and logistic regression, the last plot shows how all models can behave relatively similar.

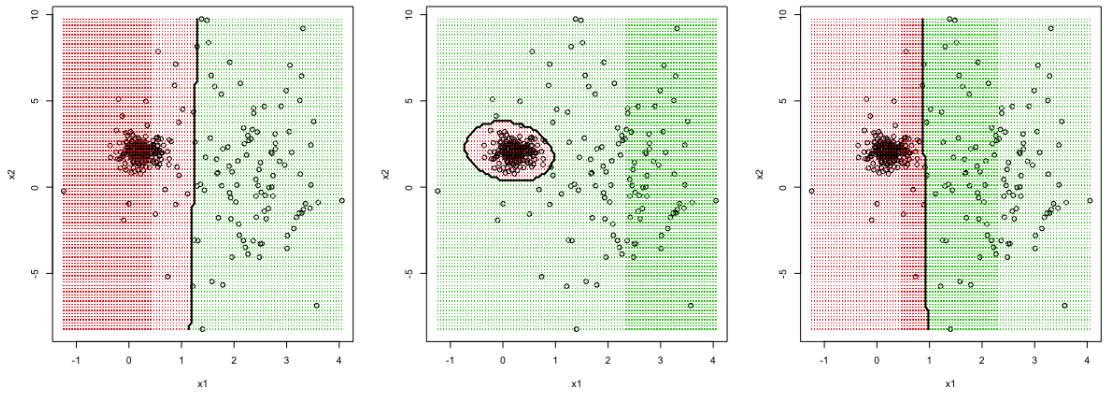

Figure 7: Decision boundaries generated by LDA, QDA and logistic regression when using s1 and s2 from a)

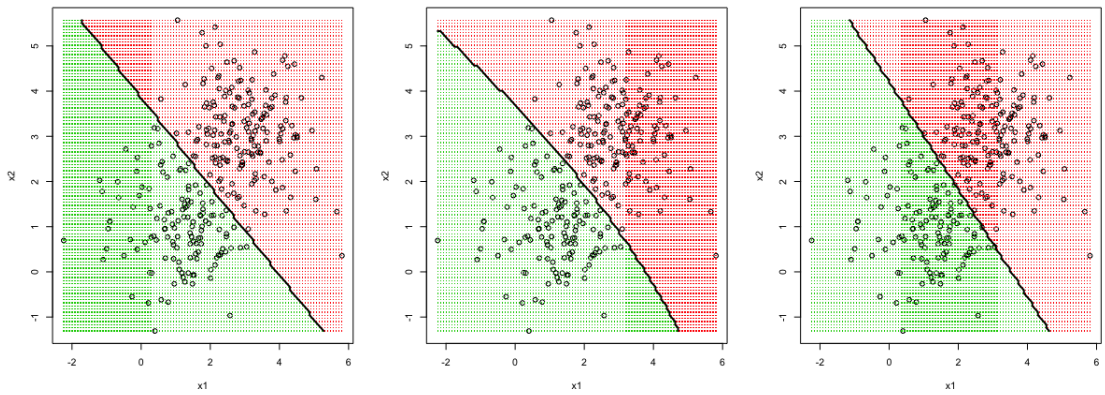

Figure 8: Decision boundaries generated by LDA, QDA and logistic regression using b1 and b2

4 Part C

4.1 a)

if $X_i \sim U(a,b)$ and [x, x+d] is a sub interval of [a,b] with fixed $d \geq 0$, then the formula is :

$$i \in \{1, .., p\} \tag{1}$$

$$P(X_i \in [x+d]) = \frac{d}{b-a} \tag{2}$$

and since X_i are supposed to be independently distributed we have :

$$P(X \in [x+d]) = (\frac{d}{b-a})^p \tag{3}$$

Therefore, for b = 1, a = 0 , d = 0.1 we get the following result:

Number of feature p	**average % sample for training**
1	10
2	0.1
100	$(0.1)^{100} = 0$

Table 6: Mean error rate for each model

The more p increases the more the fraction of the available observations will be used on average decrease exponentially, we are facing the face of the Curse of Dimensionality.

Roughly speaking knn works well with a small number of feature and when this number increase the computational power required also grow up drastically, and the value of distance measuring decreases when the number of features increase.

For LDA, since we assume the same covariance matrix for all classes that means we only need to compute one matrix of p*(p+1), for QDA K matrix of p*(p+1). In the end the number of parameters estimated is linear in p for LDA since it is linear in its input but quadratic for QDA due to the need of computing separate covariance matrices, i.e. the computational load and variance for QDA grows heavily with p. The more strict assumption of LDA, a common covariance matrix, means that it doesn't suffer as much from the increasing p as QDA in terms of variance and computational load.

4.2 b)

When the Bayes boundary decision is linear, the LDA will perform better on the test set because, the log odds of LDA is linear in X, Hence the predicted decision boundary by LDA is also linear, which is not the case for QDA. However since QDA is more flexible we will obtain a better training error rate than LDA.

On the other hand, since QDA expect a quadratic decision boundary it will outperform LDA when it comes to non linear decision boundary both in the training and testing set. (The same claim holds if we consider logistic regression instead of LDA).

When we have a small sample size, our major concern is to reduce variance, therefore it is better to use LDA in this case because of it linear decison boudary it has a high bias and a small variance. Therefore, for the training set, QDA will outperform LDA, and for the test set LDA will outperform QDA

In contrast, QDA is recommended if the training set is very large because our concern isn't the variance anymore (QDA is a high variance, small bias method) and of course if the hypothesis of the same covariance matrix doesn't stand at all. Therefore, it is more likely for QDA to outperform LDA for both training and test set.