MSA220, Statistical Learning for Big Data. Final report

Elouan Tardivel
elouan@student.chalmers.se

1 Cats and dogs

1.1 Classification

The dataset is a fair classification problem with 99 low-resolution picture of cats and 99 of dogs.

First, I started to use the caret package to study variable importance. I took the 5 most important variable and plot a scatter-plot matrix to have some insight into the structure of the classes for these features.

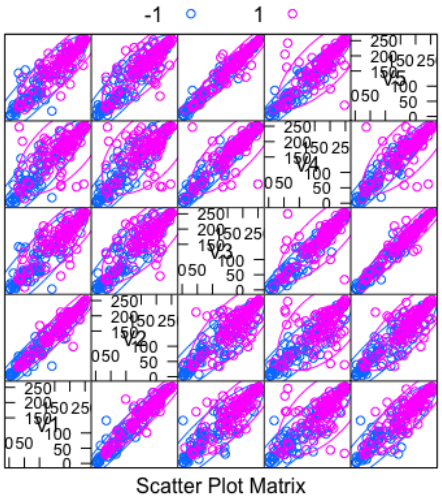


Figure 1: Feature plot from the first 5 variable given by Var IMP from the caret package

From the most important variables, the two classes are still overlapping a lot.

Both classes seem to have a common structure from those variables, only translated. That could be partially explained by the fact that the features, which are pixels, which does not make sense by there own, in image classification, the good feature for classification rely on mean and variance of line and column, gradients or more complex approaches. Therefore, transform the data in another space could be interesting.

I decided to use SVM method for training a binary classifier. I proceed to a 10 fold Cross-Validation. At every step there is therefore 90 percent of data for training and 10 percent for testing. I repeated the whole process 10 times to have 10 different junks of data to avoid outlier in the testing set. I have the following results :

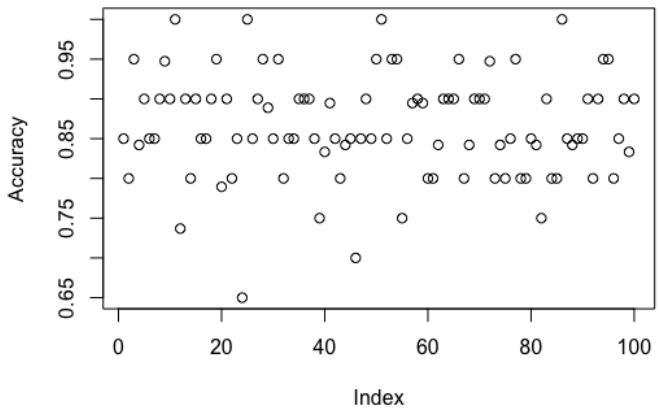


Figure 2: Variation of accuracy over the 100 iterations with radial kernel (mean 0.8655)

For a linear classifier the mean accuracy for 10 times 10 fold cross validation is

0.8403

and 0.8286 for one time 10 CV.

For a radial classifier, the mean accuracy for 10 times 10 fold cross validation is

0.8655

See in figure [2] the accuracy over the iterations. Most of the value are between 0.8 and 0.95, moreover the mean value is 0.8655, therefore, we can except having a misclassification error on new data around 14%. If we use only the most

20 important variables given by the caret package, the misclassification error increase to reach 27% on average, but with much more variance. It is not a very good error rate, however, we use only 20 of the 4096 features. Moreover, the error rate decreases only by 13%. Therefore, it is possible to only use a very small amount of features and yet having a descent misclassification error.

The figures [3] and [4] present the support vectors used in the SVM model, respectively with a radial and linear kernel. The projection on the first two principal components shows a huge overlapping between the two classes, already present in [1].

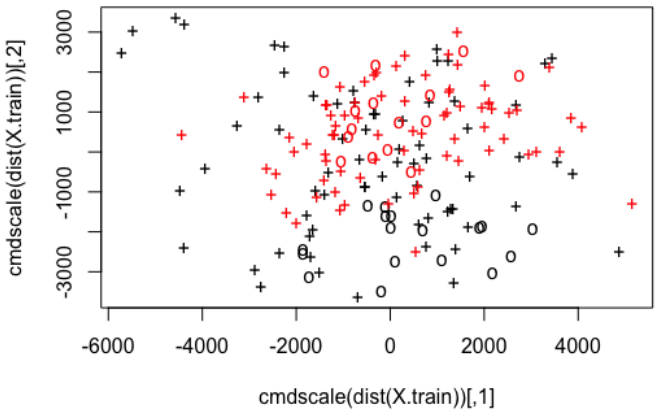


Figure 3: Scatter plot of 2 principal component from PCA (cmdscale)
Representing a radial SVM model of 145 Support Vector (cross) with cat in black and dogs in red

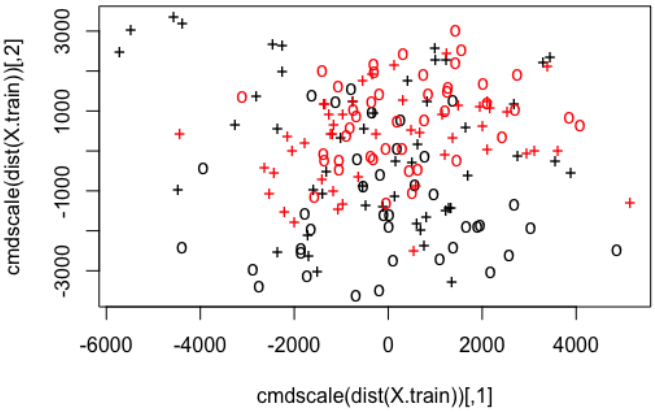
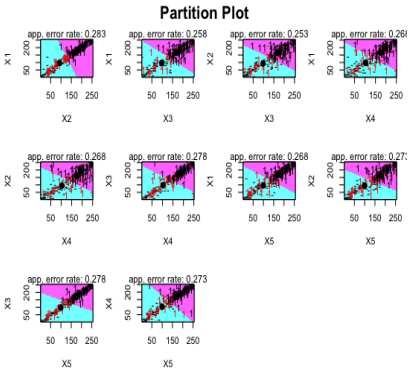
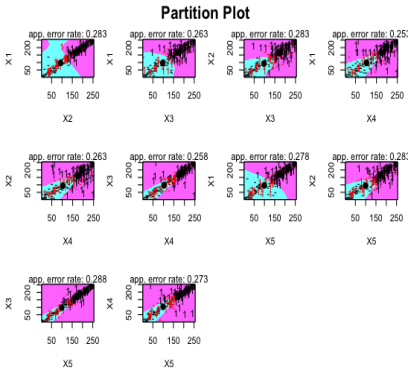


Figure 4: Scatter plot of 2 principal component from PCA (cmdscale) Representing a linear SVM model of 97 Support Vector (cross) with cat in black and dogs in red

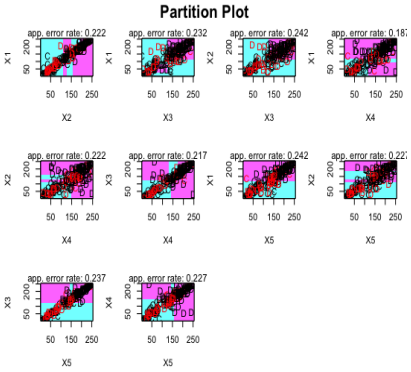
Other methods such as LDA, QDA, Rpart, and Naive Bayes do no perform better on this data set, as we can see on the different partition plots [5].



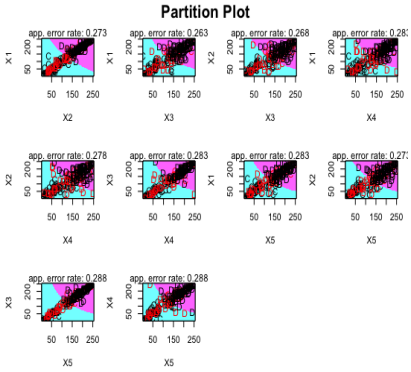
(a) lda



(b) qda



(c) rpart



(d) Naive bayes

Figure 5: Partition plot using the first 5 variables defined by variable importance from caret

The SVM results are very good, the method is fast and flexible thanks to different kernel and tuning parameters. It is also easy to understand what the method does and what can go wrong for a given dataset.

1.2 Dimension reduction

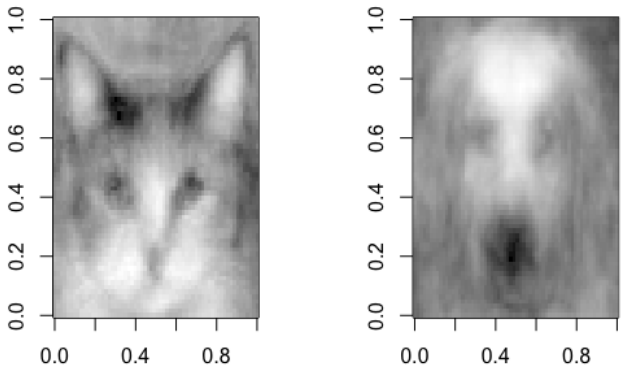


Figure 6: Picture of Cat and Dog generated by taking the average of the 99 cats and 99 dogs.

I decided to see if a typical cat or dog exits, by first plotting the image of the average over all dogs and all cats (see [6]). The picture is very blur, but it is still possible to recognize the cat and the dog. Some particularities of the cat appear, such as the shape and position of ears and the eyes or the form of the face. For the dog, the main element is the muzzle and different ears than cats.

I used SVD to operate compression of some of the pictures (see figure [7]). The result is very impressive, with only 4 SVD components the cat and the dog can be recognized. Moreover, when using only 2 components some kind of pattern appears :

- the picture of the cat become very uniform except at the position of the ears letting the background appearing.
- the black area of pixels remain where was the muzzle of the dog.

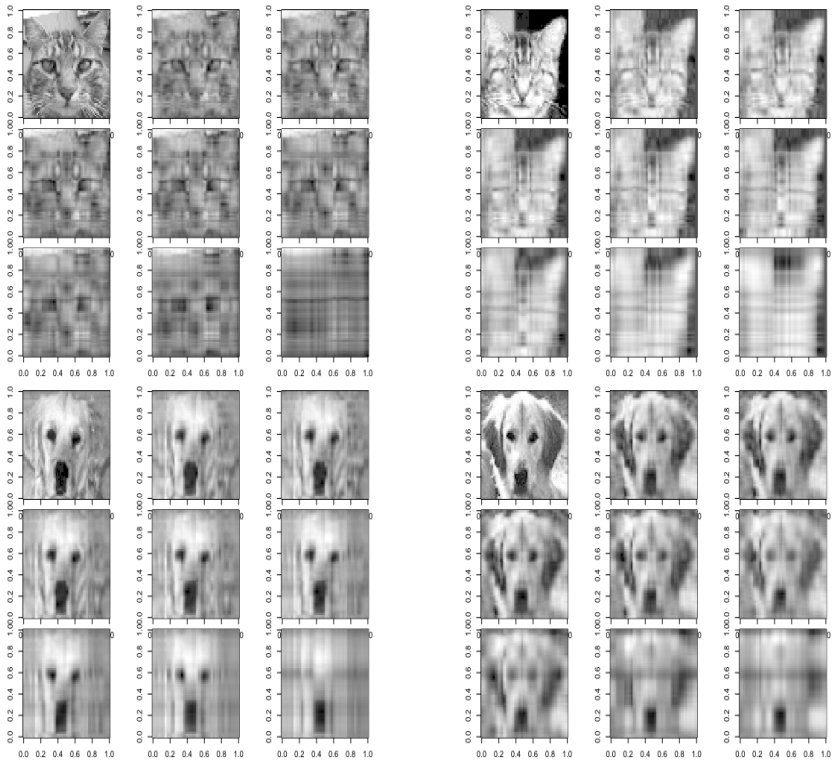


Figure 7: Picture of cat and dog. The first image is the original. The others are generated using svd from 9,8,..,2 singular values

1.3 Clustering

I used the partition around medoids (PAM) method to describe the ability to find the right number of cluster. From the silhouette width [8], $k = 2$ clusters seems to be a reasonable solution. The method is, therefore, able to learn the existence of two type of animals.

However, the quality of the cluster is very poor, the misclassification error is very high for 2 clusters.

Table 1: Prediction result from PAM clustering (0==Cat;1==Dog)

	0	1
1	46	37
2	53	62

Therefore, I decided to use the Kmeans method and test the stability of the clusters. The result shows that the clusters are very unstable using different

techniques for assessing it.

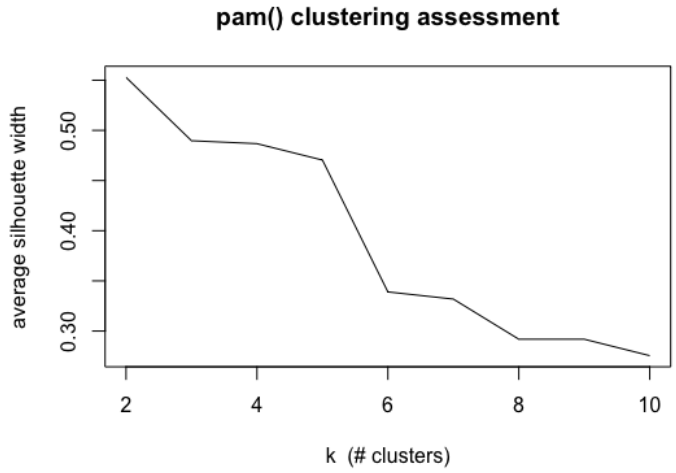


Figure 8: PAM clustering assessment

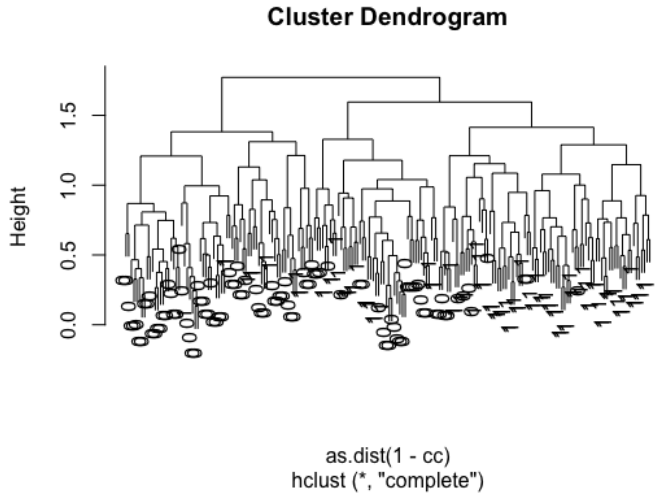


Figure 9: Hierarchical clustering. The clusters on the very left and on the very right are very pure, but in between cats and dogs are clustered together

2 Contaminated spam data

The figure [10] presents a projection over the first two component of PCA of the spam dataset. The non spam email lies on in one line, and the spam email spread out from this line to cover a huge space. As a result, we can already foresee some misclassified points.

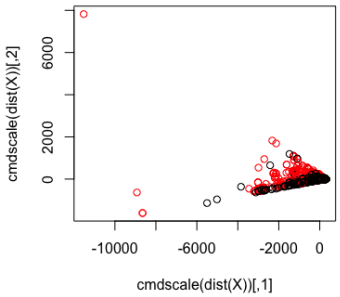


Figure 10: Projection over the first two components found by PCA (in red the spam)

I used different methods to classify the emails :

- SVM error 0.084
- Random Forest error 0.093
- RPART error 0.230
- Naive Bayes error 0.094
- KNN error 0.184

I saved all the misclassified points and then took all the common points. When using the intersection of the 2 classifiers the number of misclassified points was around 6,8%; 5,7% for 3; 4,3% for 4; and 4% for 5 classifiers. There not a huge variation over the intersection. Therefore, it is fair to argue that the mislabeled point found are among the 5% for mislabeled data.

The figure [11] presents first the misclassified points in black and other points in blue, then only the misclassified point with the spam email in red and the non-spam one in black. We saw before the non-spam email were lying on a line and the spam ones were spread out over a direction orthogonal to the latter. As a result, by looking a the second plot it would more straightforward to assign the red color to the non-spam email. I decided to use the full set of data and not sub-samples of it, as the result were already very good.

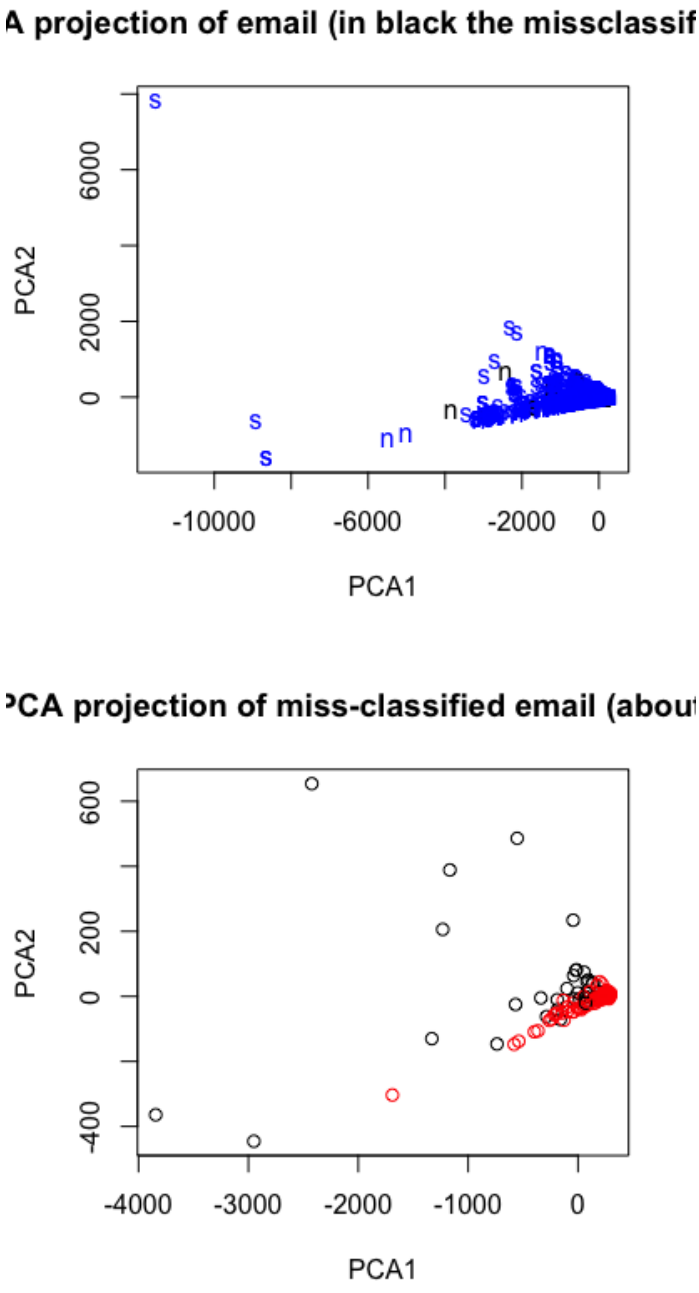


Figure 11: Misclassified email

3 Cancers classification with genes expression

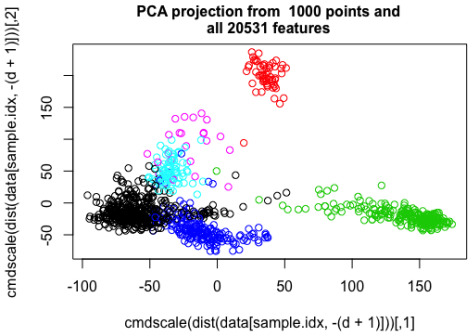


Figure 12: Projection over the first two components found by PCA using a subset of 1000 data points

The TCGA dataset presents 6 type of cancers explained by no more than 20530 genes expression and 2887 data points. Thus, it is very hard to have insights on the structure of the data as well as training a classifier due to the very high dimensionality and a huge amount of points. For instance, most of the classic methods for classification fail due the size of the matrix. However, we can use PCA to see the structure of the classifier over the 2 main components. Two out the 6 classes are very well separated in figures [12], the last 4 are grouped together with 3 distinct bulbs and one class underrepresented. The misclassification can be excepted to be mostly among these 4 classes.

Table 2: Prediction result from kknn classification with 1000 points as training set, 1887 for testing and a sample of 5000 features over 20531

	BC	GBM	KI	LU	OV	U
BC	792	0	0	1	0	0
GBM	0	109	0	0	0	0
KI	0	0	402	1	0	0
LU	4	0	0	358	1	1
OV	0	0	0	0	180	0
U	11	0	0	1	0	26

The table [3] present the classification result, as foreseen before with the PCA projection the underrepresented pink class U is very misclassified. Unexpectedly, the red class KI is also misclassified once. But in general, the result are very good with a misclassification error of 0.004.

The result is exactly the same with pairwise and one-against-all technique when

Table 3: Prediction result from svm classification with pairwise technique. Using half of the data for training and the other half for testing and a sample of 5000 features over 20530

	BC black	GBM green	KI red	LU blue	OV turquoise	U pink	class Error
BC	606	0	0	0	0	1	0
GBM	0	79	0	0	0	1	0
KI	0	0	304	0	0	0	0.003
LU	0	0	1	294	0	2	0
OV	0	0	0	0	128	2	0
U	0	0	0	0	0	26	0.187

using the svmligth function from klaR package. The result should be very different but I did not find any solution to that problem.

4 Kernel assessment for KNN

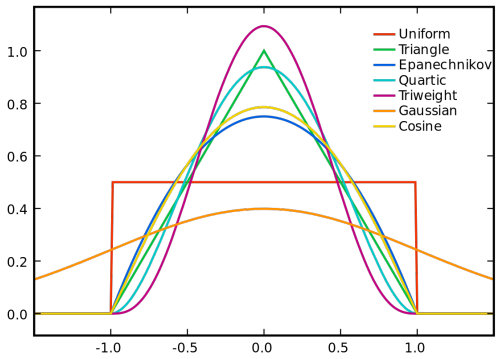


Figure 13: Plot representing various kernel from Wikipedia

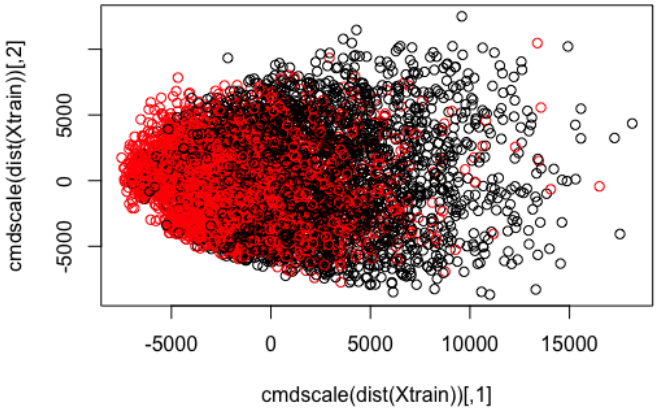


Figure 14: Scatter plot of 2 principal component from PCA (cmdscale)

I decided to compare the performance of different kernel with the KNN method. The kknn package offers an implementation of the KNN method using Minkowski distance and the classification is proceed using the maximum of summed kernel densities. In order to compare these kernel I used the Gisette dataset. It is a two classes problem classification of handwritten digits 4 and 9. The full training set, 6000 instances, has being used in the learning phase and 1000 instances have been used for testing. The dataset present 5000 features.

misclassication error : 0,039

Table 4: Classication prediction for rectangular kernel, with a misclassication error of 0,039

	-1	1	error
-1	488	12	0,024
1	27	473	0,054

Table 5: Classication prediction for inverse kernel, with a misclassication error of 0,039

	-1	1	error
-1	488	12	0,024
1	27	473	0,054

Table 6: Classication prediction for gaussian kernel, with a misclassication error of 0,039

	-1	1	error
-1	488	12	0,024
1	27	473	0,054

Table 7: Classication prediction for rank kernel, with a misclassication error of 0,041

	-1	1	error
-1	486	14	0,028
1	27	473	0,054

Table 8: Classication prediction for optimal kernel, with a misclassication error of 0,047

	-1	1	error
-1	481	19	0,038
1	28	472	0,056

Table 9: Classication prediction for Epanechnikov kernel, with a misclassication error of 0,051

	-1	1	error
-1	480	20	0,04
1	31	469	0,062

Table 10: Classication prediction for triangular kernel, with a misclassication error of 0,053

	-1	1	error
-1	478	22	0,044
1	31	469	0,062

Table 11: Classication prediction for cosines kernel, with a misclassication error of 0,053

	-1	1	error
-1	478	22	0,044
1	31	469	0,062

Table 12: Classication prediction for biweight kernel, with a misclassication error of 0,136

	-1	1	error
-1	468	32	0,064
1	36	464	0,072

Table 13: Classication prediction for triweight kernel, with a misclassication error of 0,14

	-1	1	error
-1	468	32	0,064
1	38	462	0,076

The rectangular, gaussian and inverse kernel perform very well for classifying the class -1 almost two times less than the triangular or Epanechnikov kernel and slightly worse for the class 1. On the overall the error rate stays the best for this problem. The triangular rank, optimal, Epanechnikov, and cosine kernel perform in the same range of error. The worse classifiers from far are the biweight and triweight kernel. This shows that the choice of the kernel is significant in the classification result, a model rather be built with the proper off-the-shell kernel depending on the structure of the dataset. Cross-validation and other methods can be used to select the best kernel and then improve misclassification.