Introduction to Data Science: Assignment 4a + b

Eelke Landsaat (s4056868)
Jesse Reurink (s3771385)
Darragh Spillane (s5270855)
Darren Zammit (s5284236)

Group 10

October 13, 2022

4a

4a1

a) We calculate the entropy with the following equation:

$$I(t) = -\sum_{i=0}^{c-1} p(i|t) \log_2 p(i|t)$$

In this training example set there are 9 entries, 4 of which are positive, and 5 of which are negative. Then the entropy of this set is:

$$I(t) = -\frac{4}{9}\log_2\frac{4}{9} - \frac{5}{9}\log_2\frac{5}{9} \approx 0.991$$

b) The information gain can be calculated using:

$$G = I(parent) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

For an a_1 split, we get 1 positive value and 4 negative values when $a_1 =$ False, and 3 positive values and 1 negative value when $a_1 =$ True. Then the entropies,

$$I(False) = -\frac{1}{5}\log_2\frac{1}{5} - \frac{4}{5}\log_2\frac{4}{5} \approx 0.722$$

$$I(True) = -\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4} \approx 0.811$$

Then the combined entropy:

$$I(a_1) = \frac{5}{9}\cdot 0.722 + \frac{4}{9}\cdot 0.811 \approx 0.762$$

Lastly the gain for a split in a_1,

$$G(a_1) = 0.991 - 0.762 \approx 0.229$$

Similarly for a split in a_2, we get 2 positive values and 2 negative values when $a_2 =$ False, and 2 positive values and 3 negative values when $a_2 =$ True. As such, the entropies:

$$I(False) = -\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4} = 1$$

$$I(True) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} \approx 0.971$$

Then,

$$I(a_2) = \frac{4}{9}\cdot 1 + \frac{5}{9}\cdot 0.971 \approx 0.984$$

And the gain for a split in a_2,

$$G(a_2) = 0.991 - 0.984 \approx 0.007$$

d) The error is defined as:

$$Error = 1 - max(p(i|t))$$

So,

$$Error(a_1) = 1 - (\frac{5}{9}\cdot\frac{4}{5} + \frac{4}{9}\cdot\frac{3}{4}) = \frac{2}{9}$$

$$Error(a_2) = 1 - (\frac{5}{9}\cdot\frac{3}{5} + \frac{4}{9}\cdot\frac{2}{4}) = \frac{4}{9}$$

Clearly a_1 is the better split according to the classification error rate.

e) The Gini index is given by:

$$Gini = 1 - \sum_{i=0}^{c-1} p(i|t)^2$$

So,

$$Gini(a_1) = 1 - (\frac{5}{9}(\frac{1}{5}^2 + \frac{4}{5}^2) + \frac{4}{9}(\frac{3}{4}^2 + \frac{1}{4}^2)) \approx 0.34$$

and

$$Gini(a_2) = 1 - (\frac{5}{9}(\frac{3}{5}^2 + \frac{2}{5}^2) + \frac{4}{9}(\frac{2}{4}^2 + \frac{2}{4}^2)) \approx 0.49$$

According to the Gini index, a split in a_1 is the best.

4a2

Abstract

In this report, we outline some methods used for molecular classification of Acute Myeloid Leukaemia (AML) through the Flow Cap Cytometry data set. Flow Cytometry uses

light scatter and fluorescence emission properties of a copious number of individual cells to count and sort cells and detect bio-markers within the cells. The data produced by this has proven to be effective in detecting various diseases within patients.

Introduction

The Flow Cap Cytometry data set provided contains a total of 359 subjects with the first 179 being labelled as either class 1 (Healthy) or class 2 (AML patients). Each subject consists of 31 markers with 6 quantities per marker totalling 186 potential features that may be used to classify AML patients.

The task is to build a model that correctly classifies the 20 AML cases within the 180 unlabelled subjects provided. In this report we will analyse the effects of various pre-processing strategies on the data and its impact on the modelling as well as looking into the performances of various models for predicting possible AML cases with the goal to find a model which can accurately predict AML cases given data collected via Flow Cytometry.

1 Methodology

We start with an in-depth outline of the data and its source as well as making clear the goal of this study. Next, we will discuss some of the approaches which were considered suitable for solving the problem along with the positives/negatives of each approach. We then move on to how we dealt with the pre-processing of the data, the methods we used for pre-processing and their effects on the data. Finally, we look at the methods we landed on to test the data as well as the metrics and evaluation methods we used to gauge the effectiveness of possible solutions.

1.1 Approaches Considered

As a baseline we decided to implement both K-Nearest Neighbours (KNN) and Decision Tree (DT) algorithms. KNN was chosen due to its simplicity and ability to produce non-linear decision boundaries.However KNN does have disadvantages such as its sensitivity to outliers and its assumption that all features are equally important which often is not the case [1]. The decision tree algorithm was also chosen due to simplicity and its intuitive nature although the simplicity does not suit the complex nature of the task.

Extending on from the simplicity of the baseline models we considered and experimented with many other more complex algorithms eventually landing on AdaBoost, Random forest and Largest Margin Nearest Neighbour, Logistic and support vector machines classifiers as suitable choices. AdaBoost was chosen at it is an algorithm which applies successive decision trees with misclassified instances from previous decision trees given higher weights for subsequent decision trees. The idea of having higher weights for misclassified instances seemed suitable for the task at hand due to the severe implications a misclassified cancer diagnosis can have. The Random forest algorithm was chosen as it is an extension of Decision tree's which overcomes the problem of over simplicity of the decision tree algorithm at the expense of requiring more computational power to complete. LMNN learns a metric for KNN where each instance within the training data is surrounded by at least k instances of the same class label and then the leave-one-out error is minimised. Logistic regression could also be used in this case however it's important to recall its sensitivity to outliers.

1.2 Exploratory Analysis

With 186 possible explanatory variables the main goal of the data pre-processing was to reduce dimensionality. This was primarily done through Principal component analysis (PCA) and feature selection via ANOVA. Another aim of pre-processing was to gauge the complexity of the problem this was done by embedding the data using the first components from PCA and the non-linear method tSNE.In this section we also look at the overall affect pre-processing on the data both in training and testing sets.

PCA - The aim of principal component analysis is to reduce the dimensionality of the data while losing as little of the variance of the data as possible.The graphs below shows the amount of variance in the data that can by explained by the principal components individually and cumulatively. it can be seen that after 25 principal components upwards of 99.6 percent of the variance can be explained with the amount of variance able to be explained after the 25 principal component being extremely small compared to the first 25.Figure 2 below shows boxplots of the first 5 principal components and the last 5 principal components which outlines the fact that the first principal components have far greater variance than the last.

Feature Selection (ANOVA) - Again the idea behind feature selection (via ANOVA) is to reduce the dimensionality of the data.This is done through ANOVA F-tests represents the variance between sample means/variation within samples.The higher the f score the more important the feature. See below in figure 3 a bar chart showing the importance of each feature. ANOVA was then used throughout the testing to select the best features to reduce over-fitting and noise. This is particularly important when considering the fact that some classifiers assume the features are more or less equally important.

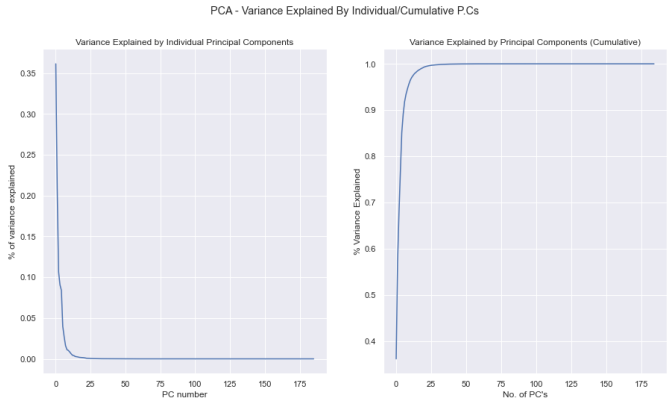

Figure 1: Variance explained vs. number of principal components for individual components and components combined (cumulative).

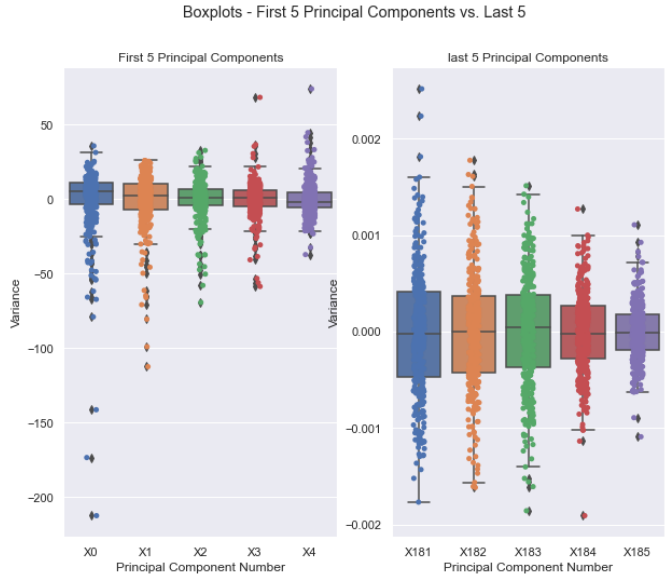

Figure 2: Boxplots showing the first 5 principal components and the last 5 principal components.Note: the scale for variance is significantly smaller for the last 5 components and thus the variance of the last 5 is significantly smaller.

Complexity Analysis - The aim of this section is to attempt to gauge the complexity of AML classification through Flow Cytometry by embedding the data first via 2 and 3 principal components and then by 2 and 3 components of t-SNE which is a non-linear method commonly used to visualize highly complex data. It can be seen in figure 4 of the 2 dimensional scatter graphs for both t-SNE and PCA that there are definite characteristics that separate AML and healthy cases however there does seem to be a handful of 'problem' cases of AML that are not easily distinguishable from healthy ones (when projected on to 2 components) although there t-SNE embedding does have less of these 'problem' cases compared to PCA. Similarly it can be seen in figure 5 when using 3 components that clear boundaries for AML and Healthy cases are present but not for all cases. In 3 dimensions it appears that the PCA embedding provides more clear separation between

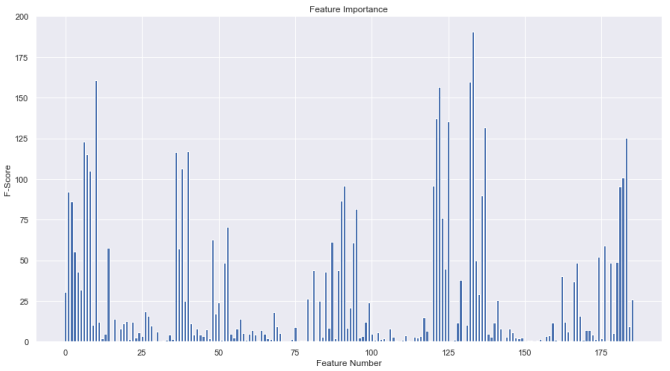

Figure 3: Bar chart showing the importance of each feature based on ANOVA F tests.

AML and Healthy cases however this is due to the fact that the depth of each point is not as apparent for the t-SNE which is non linear and relies more on depth.

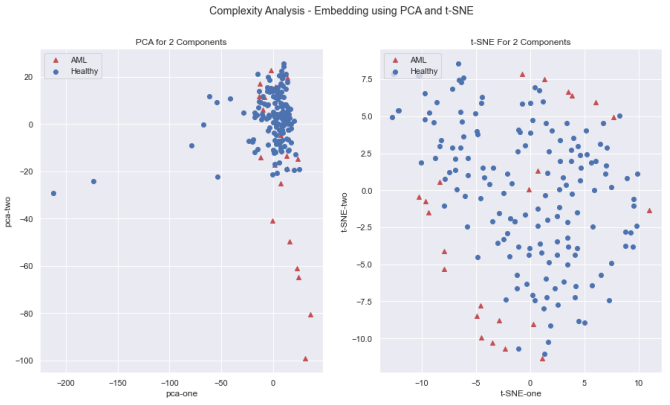

Figure 4: Scatter plots show data when projected on to 2 Principal Components and 2 t-SNE components. it is apparent in both plots that there are boundaries between AML and healthy cases.

Impact of Pre-processing - Here we will investigate the impact of pre-processing on the data by looking at the distribution of data for training and testing sets both before and after pre-processing. It can be seen below in figure 6 that the distribution of training and testing sets are extremely similar before pre-processing. In figure 7 both the training and testing data have been projected onto 3 Principal components and graphs of there distributions have been created.it is clear to see that projection the data on to principal components does not cause any discrepancies between the distribution of the test and training data. Feature selection will have now impact on the distribution of the individual features as it does not transform them in any way and z-score standardization will cause all features to have a standard normal distribution.

next we use t-SNE embedding to visualize the impact various pre-processing has on the data. 3 pre-processing methods have been used namely PCA, z-score standardization and feature selection. The scatter plots for the no pre-processing, PCA and z-score are very similar indicating that the pre-processing has little to no impact on the data with one advantage of PCA being that it reduces

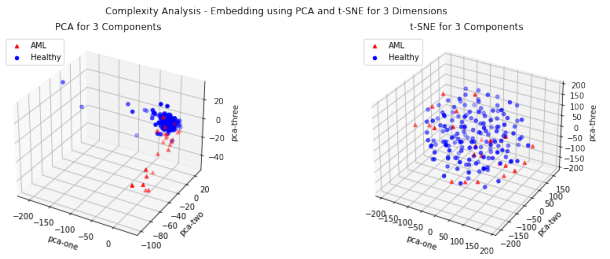

Figure 5: Scatter plots show data when projected on to 3 Principal Components and 3 t-SNE components. Again it is apparent that there are boundaries between AML and healthy cases although this is not so clear in the t-SNE case.

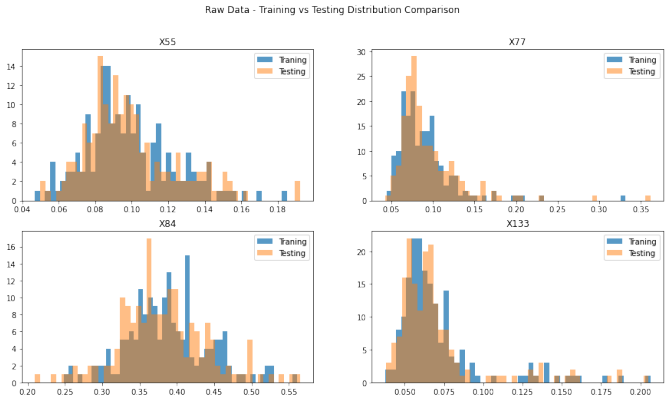

Figure 6: Graphs show the distribution of four randomly selected features before pre-processing in both training and testing sets.

the model dimensionality. In the case of feature selection classes are more separated giving it the advantage over the other methods . Due to this we have decided to use feature selection on the data due to the fact that feature selection reduces dimensionality and the likelihood of over fitting.

1.3 Evaluation Methods

Before moving on to our results we will outline the metrics and techniques used to evaluate the performance of each our models. Given the nature of the challenge we decided on recall to be the primary model evaluation metric to use. Recall penalises false negatives which is the exact reason we chose to focus on it as the implications of receiving false negatives for diagnosis of any cancer in patients can be extremely severe. Obviously other metrics such training/validation accuracy,f1,precision were used to judge the performance of the models and models optimizing these across the board were preferred to models with only one superior metric.

The main pre-processing method we used was feature selection via ANOVA as well as Smote with Tomek links. We also tried testing the classifiers using the best principal components however we got better results with feature selection which makes sense as whilst principal component analysis can help remove noise it does not fully eliminate it. Feature selection decreased over-fitting and yielded

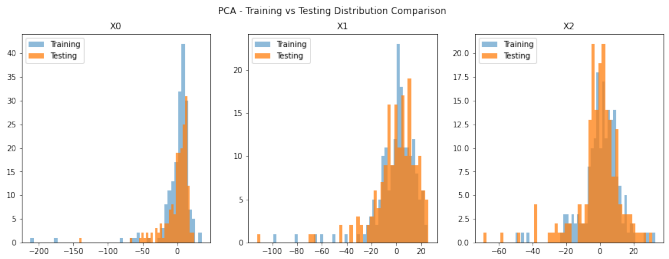

Figure 7: Graphs show the distribution of training and testing data projected on to 3 Principal Components.

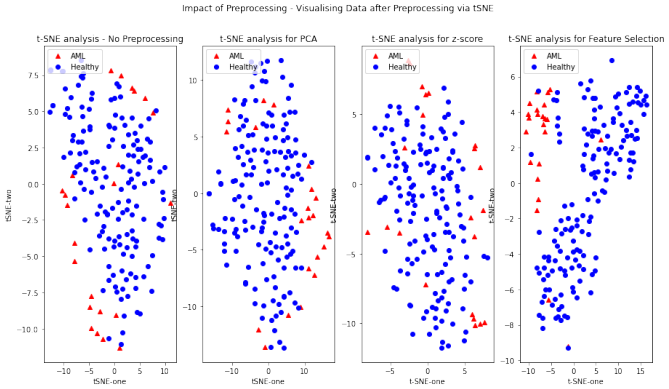

Figure 8: Graphs show the distribution of training and testing data projected on to 3 Principal Components.

better results throughout whilst Smote with Tomek links helped in some cases. We used the latter method since the data was highly imbalanced, thus oversampling was needed. The problem was that using Somek worsened over-fitting and so we used Smote and Tomek to find a middle ground. In this report we use the best 25 features although we tried many others. 25 features seemed to yield the best results and further reduction did not seem to improve results for most of the classifiers. All classifiers were cross validated with 10 folds.

2 Results

Accuracy vs k for KNN with 25 (left) and 186 (right) features

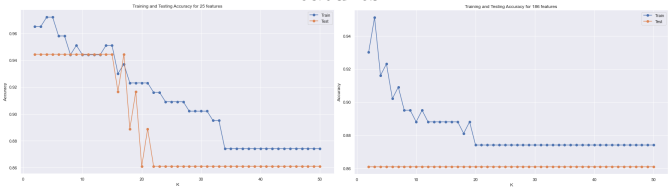

Figure 9: Graphs show the effect of removing insignificant features. One can see that the gap between the testing data (orange) and the training data (blue) decreased when removing features.

For the KNN and LMNN classifiers the value for k was selected by first plotting the values of the precision, accuracy, f1-score and recall and selecting the ones with the best results, giving more weight to the recall.

Scores vs k for KNN with SmoteTotek (left) and without (right)

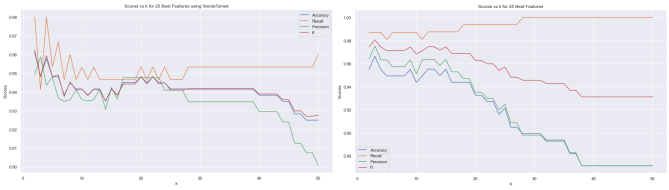

Figure 10: Graphs show the values of the scores for each test. The best values for K were selected from these.

Classifier	Parameters
SVM	'C': 100, 'degree': 1, 'kernel': 'rbf'
Logistic regression	'C': 206.9, 'max_iter': 1000, 'penalty': 'l1', 'solver': 'liblinear'
Decision tree	'criterion': 'gini', 'max_depth': 3, 'min_samples_leaf': 1, 'min_samples_split': 8, 'splitter': 'random'
Random forest	'max_depth': 20, 'min_samples_leaf': 3, 'min_samples_split': 2, 'n_estimators': 5
ADA	'learning_rate': 0.1, 'n_estimators': 20
KNN (no Smote)	'k': 4
KNN (Smote)	'k': 4
LMNN (no Smote)	'k': 4
LMNN (Smote)	'k': 5

Table 1: Optimal classifier parameters used.

The best value for k were taken to be 4 in both cases. Here we took the one with the best recall and taking into account the values for the other scores.

Scores vs k for LMNN with SmoteTomek (left) and without (right)

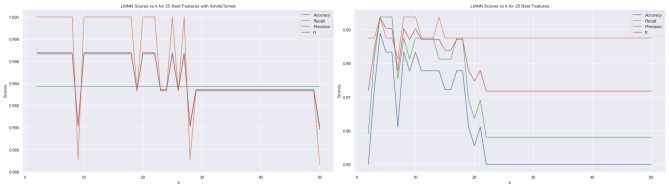

Figure 11: Graphs show the values of the accuracy (blue), precision (green), recall (orange) and f-1 score for LMNN with SmoteTomek and without.

The best values for the no smote case was 4 whilst for the other one we took 4 and 5 to be the best.

All other classifiers were optimised via hyperparameter optimisation. Below is the table for the best parameters we found for each. Smote and Tomek did not help with the decision tree, logistic regression, ADA, random forests and SVM.

We then built an ensemble using the decision tree, support vector, logistic regression and KNN using the best settings for each. The result was very similar to the Random Forests Regression which is to be expected since it is also an ensemble method which uses decision trees. Our ensemble also utilised decision trees and so the results made sense. We could not integrate the LMNN into the ensemble despite it giving the best results so we settled for these.

Classifier	Accuracy	Recall	Precision	F1-score
SVM	0.9611	0.9746	0.9816	0.9776
Logistic regression	0.9611	0.9933	0.9700	0.9749
Decision tree	0.9500	0.9871	0.9820	0.9687
Random forest	0.9778	0.9938	0.9820	0.9875
ADA	0.9667	1.000	0.9640	0.9812
KNN (no Smote)	0.9556	0.9875	0.9640	0.9752
KNN (Smote)	0.9573	0.9800	0.9383	0.9581
LMNN (no Smote)	0.9889	0.9938	0.9941	0.9937
LMNN (Smote)	0.9967	1.000	0.9938	0.9968
Ensemble	0.9556	0.9808	0.9820	0.9812

Table 2: Result metrics for different classifiers.

3 Discussion and Conclusion

Here we have tested a few classifiers to detect AML. The best five we found were from the LMNN with SmoteTotek which gave a recall of 100 percent. This was not included in the ensemble, however. Instead we used the next best five, these being the SVM, Logistic, decision tree, KNN without Smote and ADA with the most optimal parameters we found for each. The result was very similar to the random forests which is to be expected since it includes random forests itself and it is also highly optimised. In the end we decided to use the LMNN with SmoteTomek and K=5 to determine which of the patients were most likely to test positive for the disease. 21 cases were predicted.

The best five classifiers were LMNN with SmoteTomek, ADA, Ensemble, Logistic Regression

4b

4b1

The altered script can be found in myGeneticAlgorithm.m.

4b2

The script to run the visualizations can be found in featureSelectionPlots.m. Setting the number of runs of myFeatureSelectionWithGA to 100 and recording the number of features used, the accuracy obtained with feature selection and the accuracy obtained without feature

selection after each run, we obtain the plots shown in Figure 12 and Figure 13.

We can observe from Figure 12 that selecting a number of features to perform the predictions with tends to result in a much higher accuracy than using all features. Perhaps some of the features in the data set have no relation to the predicted variable at all and attempting to use them worsens the prediction. In Figure 13 we can see that the accuracy spread increases with the number of features used for the prediction, with a somewhat downward trend. This is in line with the observation that using the full feature set achieves a worse prediction than using only selected features. The results suggest that giving a low number of features a larger weight in the fitness function of the genetic algorithm may achieve even better accuracy in general (With the caveat that the best accuracy was achieved with 7 features. More experiments are needed to draw strong conclusions.).

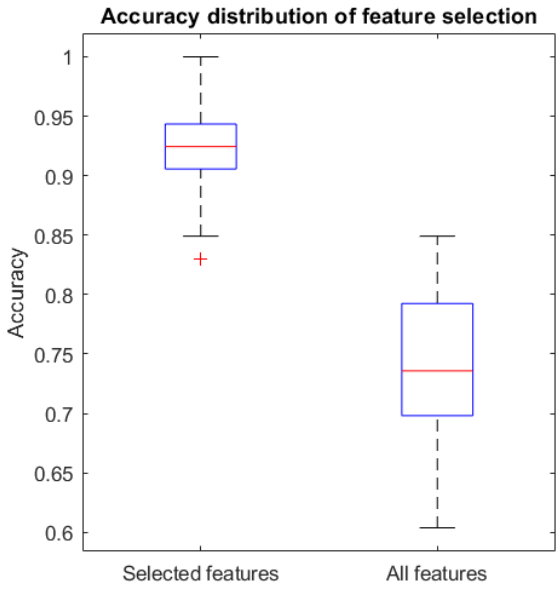

Figure 12: Results of running predictions on the wine data set with feature selection with a genetic algorithm and without feature selection 100 times. Boxplot of prediction accuracy using the selected features versus using all features.

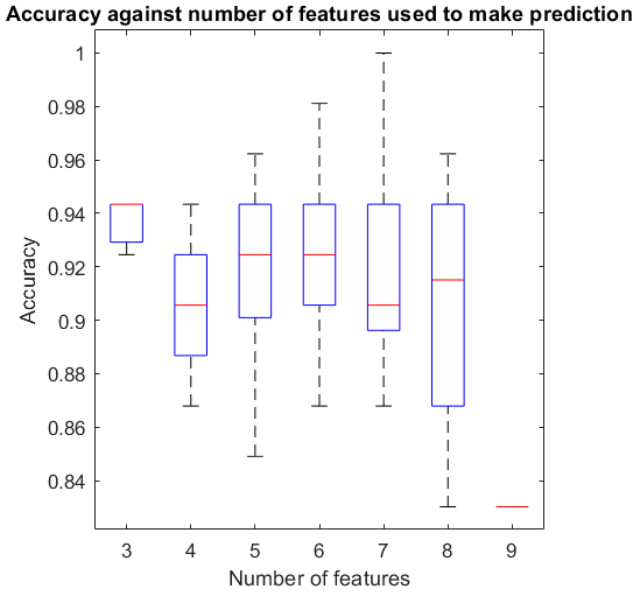

Figure 13: Results of running predictions on the wine data set with feature selection with a genetic algorithm 100 times. A comparison of the accuracies achieved using varying numbers of features.