CSC3060 AIDA – Assignment 2

Thomas Pickup

40145342

11 January 2019

Table of Contents

[Introduction 3](#_Toc534921498)

[Section 1 3](#_Toc534921499)

[Section 1.1 3](#_Toc534921500)

[Section 1.2 4](#_Toc534921501)

[Section 1.3 4](#_Toc534921502)

[Section 1.4 5](#_Toc534921503)

[Section 1.5 5](#_Toc534921504)

[Section 2 5](#_Toc534921505)

[Section 2.1 5](#_Toc534921506)

[Section 2.2 6](#_Toc534921507)

[Section 2.3 6](#_Toc534921508)

[Section 3 6](#_Toc534921509)

[Section 3.1 6](#_Toc534921510)

[Section 3.2 7](#_Toc534921511)

[Section 4 7](#_Toc534921512)

[Conclusions 8](#_Toc534921513)

# Introduction

This report follows on from Assignment 1 for the Artificial Intelligence and Data Analytics module. In Section 1, I will build upon the features gathered from the handwritten images in Assignment 1 by using a Logistic Regression model to attempt to predict whether a character is a digit or a letter.

For the next three sections, I will use a dataset that was provided to me. I will explore three different classification tests to predict what a character is. The first of these, in section 2, will be the K Nearest Neighbours algorithm. I will evaluate the KNN Algorithm both with and without cross validation. The second algorithm is done by building a decision tree, this will be done in section 3 along with the random forest classification algorithm. Finally, in section 4, I will build a model that predicts the symbol for an unknown test item. This set of test items will be provided without the labels.

I will be using R for all of the code in this Assignment. After I have explored each of these sections, I will write up my conclusions in this report.

# Section 1

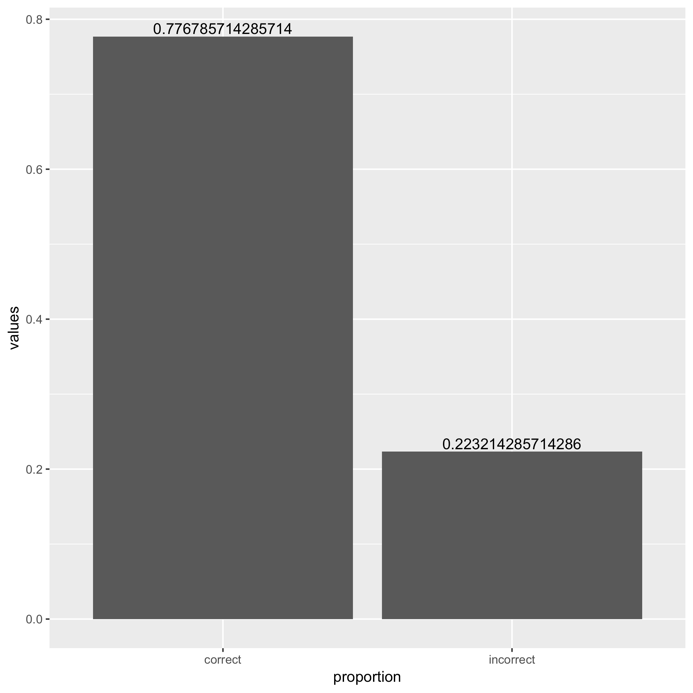
## Section 1.1

For Section 1.1, I was tasked with producing a Logistic Regression Classifier from the data collected in Assignment 1. I was then asked to report back on the accuracy of this model.

As I did not find a feature with the most significant difference in Assignment 1, I will be using the feature Number of Eyes, this corresponds to the column V18. I believe that this feature will provide the most usefulness when discriminating between a letter and a digit. This is because there are far more letters in the dataset with ‘eyes’ (e.g. a, b, d, e, g) than digits (e.g. 4, 6).

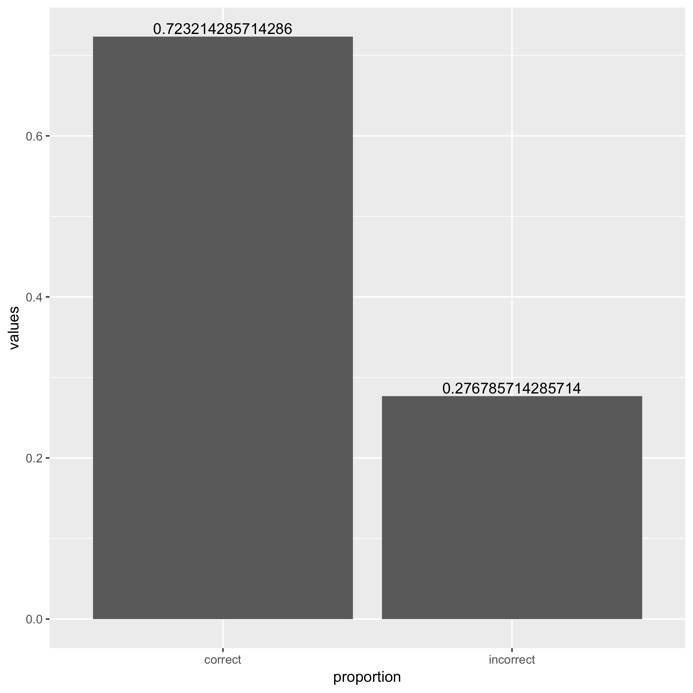
To produce the model, I first had to create an extra column in the dataset that would identify whether a row was a digit or not. To do this, I created a subset of letters and digits then created a column called digit which was filled with zeroes, the program then goes through and checks to see if the label of a digit is less than 20, if it is then it is a digit. This provides both a way to check the accuracy of the model, as well as the predicator used to build the model.

Once the model was created, I used the predict function to test the accuracy of the model against the original data provided. In the diagram below you can see that the Logistic Regression Classifier was correct roughly 77.7% of the time.



## Section 1.2

In Section 1.2, I was tasked with creating a Logistic Regression Model from the data gathered in Assignment 1. However in this instance I was to use the first 8 features to fit my model.   
The result of this was that the accuracy of the model seemed to have dropped slightly to an accuracy of 72.3%.



## Section 1.3

In Section 1.3, I was tasked with repeating the above tests, but with Cross Validation. This meant repeating the same process as in Section 1.1 and Section 1.2, but with cross validation to calculate the accuracy of the model. Cross Validation requires us to split the training dataset into *k* different folds, for this section we used 7 as the value of *k*. The results of correct predictions vs incorrect predictions are shown below:

|  |  |
| --- | --- |
|  |  |
| Part 1 Accuracy Graph | Part 1 Accuracy Graph with Cross Validation |
|  |  |
|  |  |
| Part 2 Accuracy Graph | Part 2 Accuracy Graph with Cross Validation |

As you can see in the Graph’s above, cross validation gave a more ‘real life’ accuracy given the fact that we used the training dataset as our test dataset. Cross Validation allows us to identify overfitting. Overfitting is when a model is too complex, as our training set was also our test set, this meant that there was a higher accuracy rating.

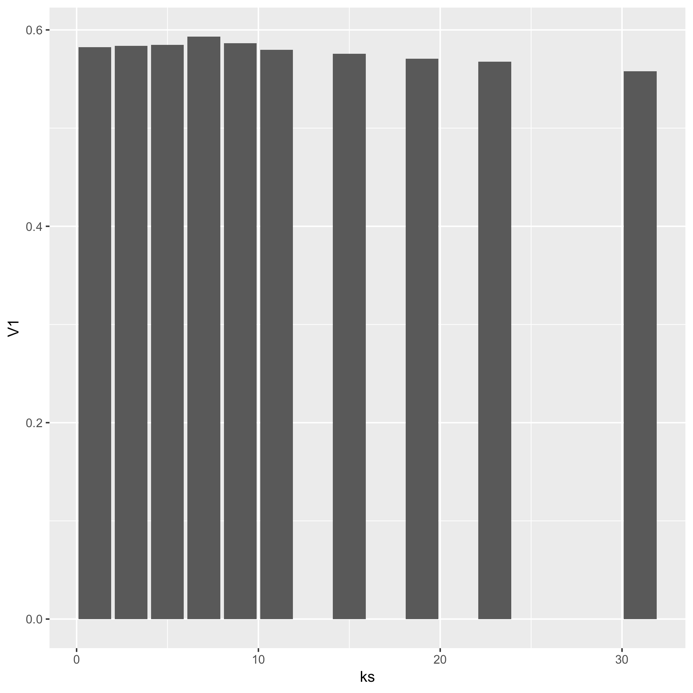
## Section 1.4

## Section 1.5

# Section 2

## Section 2.1

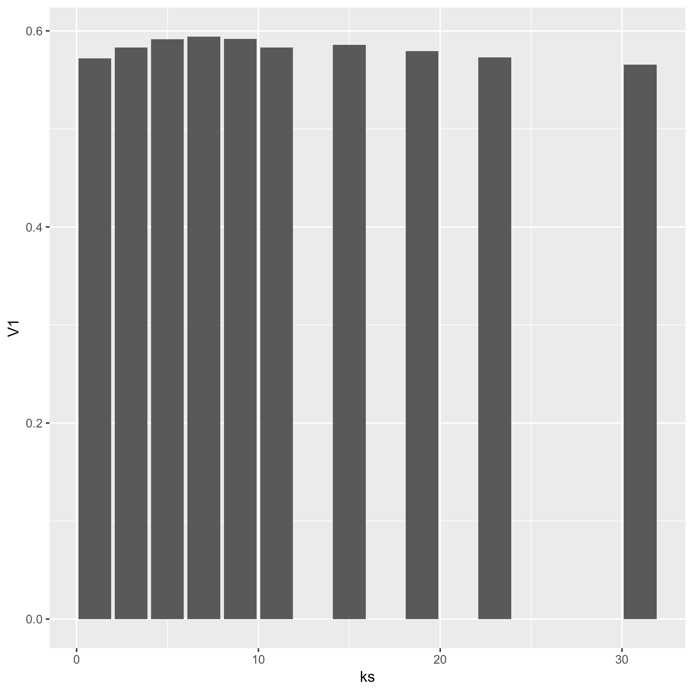
As we can see in the section below, the value of K that gave the highest accuracy is 7, this gave an accuracy of 59.3%



|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **K Value** | 1 | 3 | 5 | 7 | 9 | 11 | 15 | 19 | 23 | 31 |
| **Accuracy** | 0.5822785 | 0.5837854 | 0.5846896 | 0.5931284 | 0.5864979 | 0.5795660 | 0.5756480 | 0.5705244 | 0.5675105 | 0.5578662 |

## Section 2.2

As we can see below, including cross validation the value of k that gave the highest accuracy stayed at 7, this gave an accuracy of 59.4%.



|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **K Value** | 1 | 3 | 5 | 7 | 9 | 11 | 15 | 19 | 23 | 31 |
| **Accuracy** | 0.5719955 | 0.5829932 | 0.5916100 | 0.5941043 | 0.5918367 | 0.5831066 | 0.5857143 | 0.5792517 | 0.5730159 | 0.5654195 |

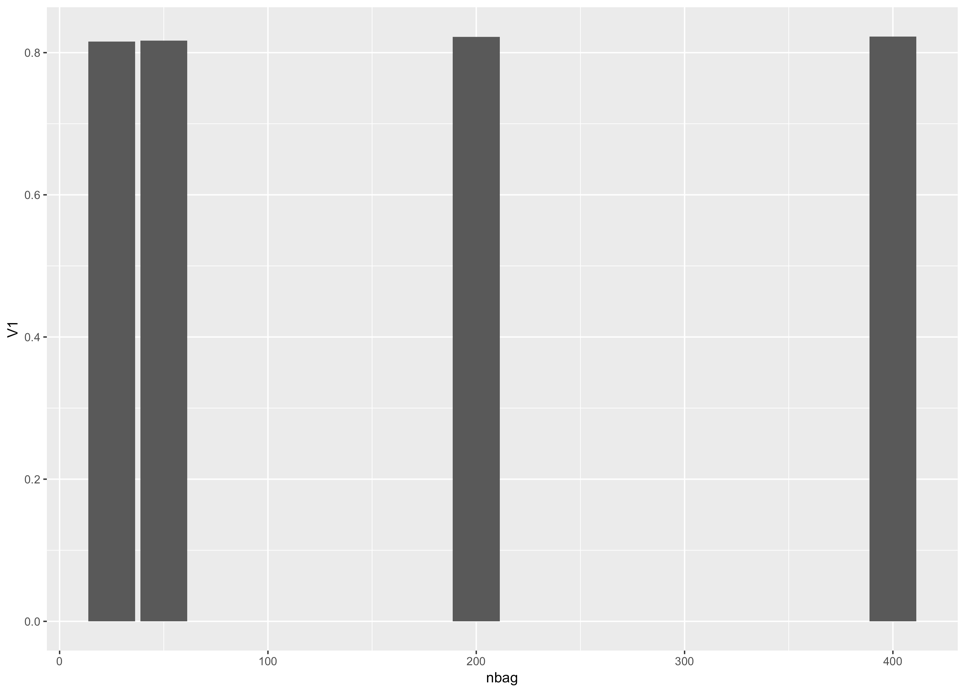
## Section 2.3

# Section 3

## Section 3.1

In Section 3.1, I built a Decision Tree Classifier, this classifier used Bagging. To achieve this, I used the *IPRED* library that includes a function called bagging, this automated the process and allowed me to change the number of bags as well as to calculate the out-of-bag estimation.

The results of this code are shown below:



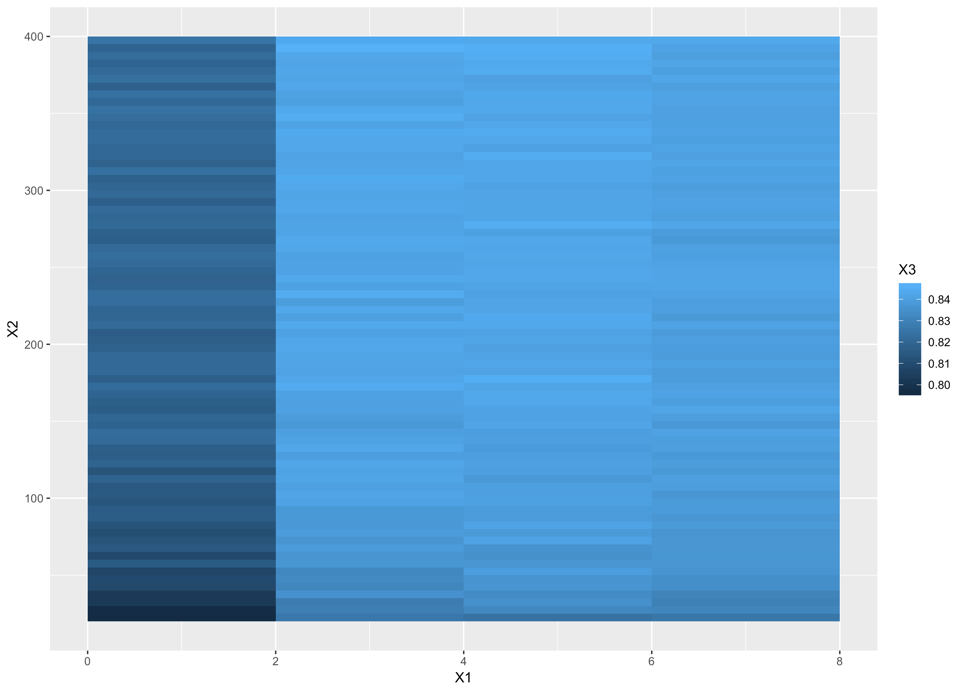
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Number Of Bags** | 25 | 50 | 200 | 400 |
| **Accuracy** | 0.815532879818594 | 0.816780045351474 | 0.822222222222222 | 0.822789115646258 |

The results show us that the more bags are used, roughly speaking, the higher the accuracy. However there is a slight gain for a lot more processing power needed, on my Core i5 processor it took around 20 minutes to computer the bagging process when at 400 bags.

## Section 3.2

In Section 3.2, I was tasked with implementing the Random Forest Algorithm with 5-Fold Cross Validation. To do this I used the RandomForest Library. I also varied both the Number of Trees the algorithm was to create as well as the number of predicators to consider at each node.

Once I ran the algorithm multiple times, I got the following Matrix:



We can see that as the number of Trees used and number of predicators considered, the accuracy increases.

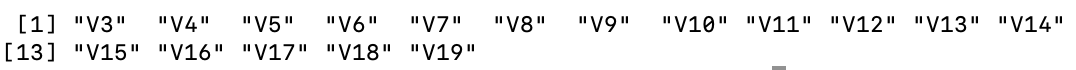
I also sorted by accuracy and found the optimal number of Tree’s and Predicator’s, based on accuracy, was 4 Predicators Considered and 395 Trees made, this gave an accuracy of 84.6%.

# Section 4

For Section 4, I decided to use the KNN algorithm. However based on the accuracy previously, I needed to choose a better set of classifiers. To achieve this, I used the Boruta library.

The Boruta library is a specialist algorithm that is used for feature selection on datasets. It is a wrapper method that derives from the Random Forest algorithm, demonstrated in Section 3.2, the algorithm uses a subset of features to train a model. Based on the inferences that is drawn from the model, it decides to add or remove features from the subset until it has a list of selected features and rejected features.

The Boruta algorithm selected the following features:



I used these features to construct a KNN classifier, passing through the test dataset and the training dataset and labels. The KNN algorithm gave me a list of predictions that I outputed to the CSV file in the root of the project folder.

I chose the KNN algorithm for the final model as it is not binary in it’s approach, unlike the logitstic regression algorithm I can classify by the label rather than whether something is or isn’t. It also allowed me to choose the amount of neighbours to consider, in this case I chose 7 neighbours due to the fact that this gave the best results in section 2.

# Conclusions

To conclude, I have explored various machine learning algorithms, from Logistic Regressions to Random Forests. I have also produced my own optimized model using an external library to give me the best features for a KNN classifier.