Project Report

Course Name: Data Mining and Machine Learning I

Student ID: 2514007

Student Name: Stefan Faulkner

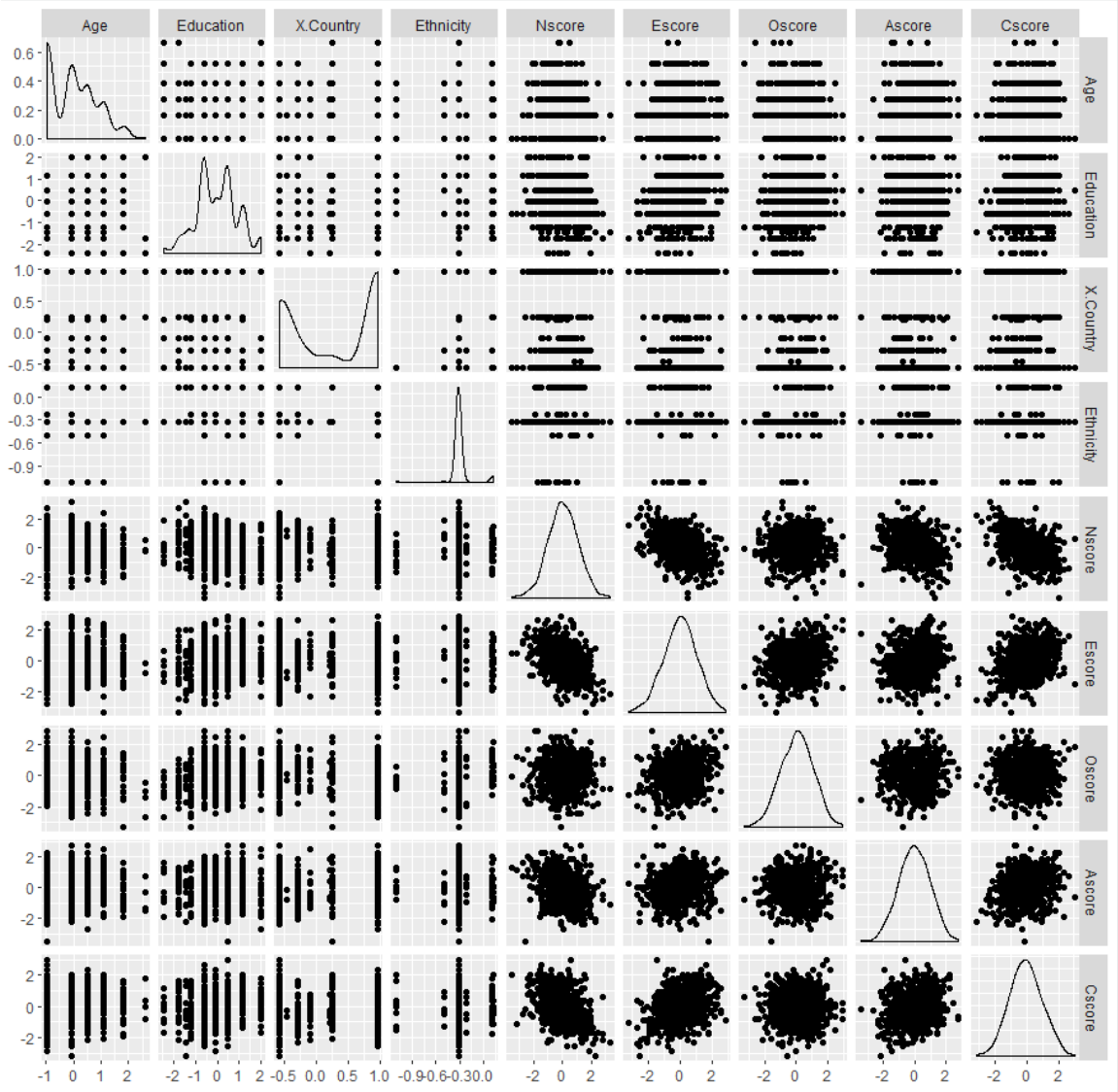
Introduction

This project aims to fit a variety of classification algorithms, which were taught to us in class. These were k-nearest neighbors (knn), support vector machines, and a wide variety of tree-based methods. Essentially a unique dataset was presented to each student and we would try to see which classification algorithm would perform the best at predicting the use of drugs by patients. We would have to apply a plethora of various methods which would be shown in the R code to help determine why this is the best classification model and then actually look on how well this would perform in the future against the test dataset. Overall, this was a fun project and help tremendously gaining a deeper understanding of the classification algorithms.

Exploratory Analysis

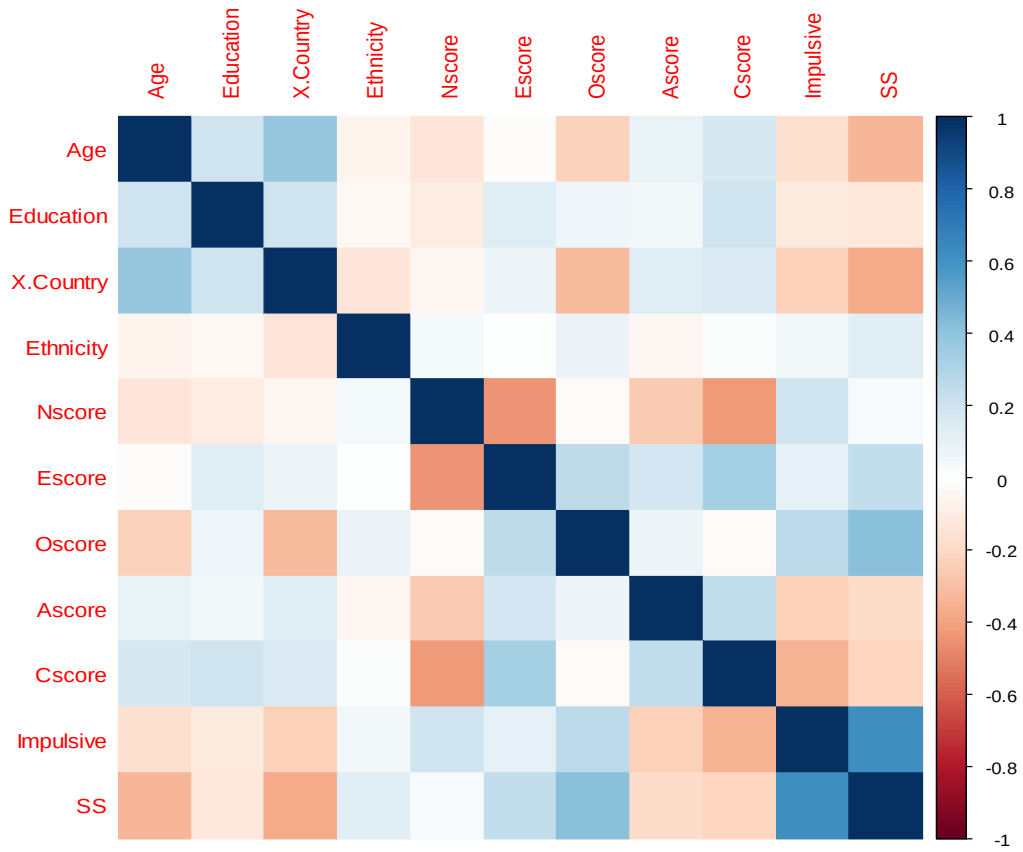
```
> str(drug_data)   #Looking on our various data types
'data.frame':   600 obs. of  12 variables:
 $ Age      : num  1.09 1.09 1.09 1.09 1.09 ...
 $ Education: num  -0.0592 0.4547 0.4547 -0.0592 -0.6111 ...
 $ X.Country: num  0.961 -0.285 0.961 0.961 -0.57 ...
 $ Ethnicity: num  -0.317 -0.317 -0.317 -0.317 -0.317 ...
 $ Nscore   : num  0.8256 -0.348 1.0212 -0.921 -0.0519 ...
 $ Escore   : num  -0.43999 -1.09207 -1.50796 -0.30033 0.00332 ...
 $ Oscore   : num  0.723 -0.178 -0.717 -1.555 0.883 ...
 $ Ascore   : num  -0.0173 -0.6063 0.2878 -0.0173 2.0397 ...
 $ Cscore   : num  0.26 0.585 -0.899 0.123 -0.143 ...
 $ Impulsive: num  -0.217 -0.711 -0.217 -1.38 -0.711 ...
 $ SS       : num  -0.2157 -0.5259 0.7654 -1.5486 0.0799 ...
 $ Class    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

The first thing, which I did in starting to explore the data, was to look at the various data types in my drug dataset. The first thing, which stood out to me, was that our dependent variable Class was of type ‘int’ initially. I know that this would cause issues later down in our analysis and model building so I corrected it and we can now see a more suitable data type ‘Factor’ above in the screenshot. In addition, since we were told explicitly that the data was transformed so I made the decision not to trouble any of the covariates above.



From the above, we used the ggpairs function to get an idea of what type a relationship is present with our predictors with histograms and scatter plots. Only a few of the predictors was chosen since the plot would get vey tedious and would be difficult to view/understand.

We can see few outliers in our top right when looking on Nscore ,Escore,Oscore,Ascore , and Cscore. In addition, if we look on the histograms we can see a huge decrease and increase in the distributions namely X.country and Ethnicity respectively.



For the plot above, I wanted to have an idea of the correlation between our explanatory variables. This is known as the correlation plot. We can see that Age is somewhat sportively correlated with Education and X.country. In addition, we can see that it almost has a 0 positive relation with Escore. We can look further and see that Education has a fair positive relation with Age and X.country . We also can see that X.country has somewhat of positive correlation with Age and Education. The Majority of the other variables have a weak correlation with the predictors, which is not bad, as we will not have much of multicollinearity. In general, the correlation coefficient has values between -1 and 1. The idea behind this is:

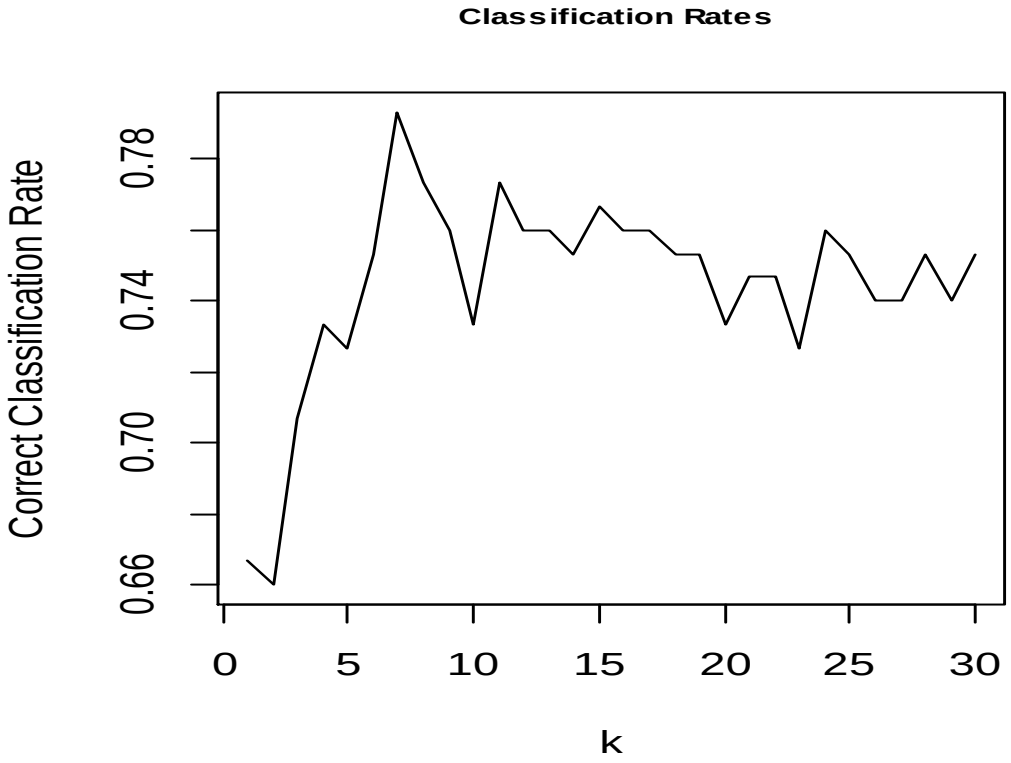
- A value closer to 0 implies weaker correlation (exact 0 implying no correlation)

- A value closer to 1 implies stronger positive correlation

- A value closer to -1 implies stronger negative correlation

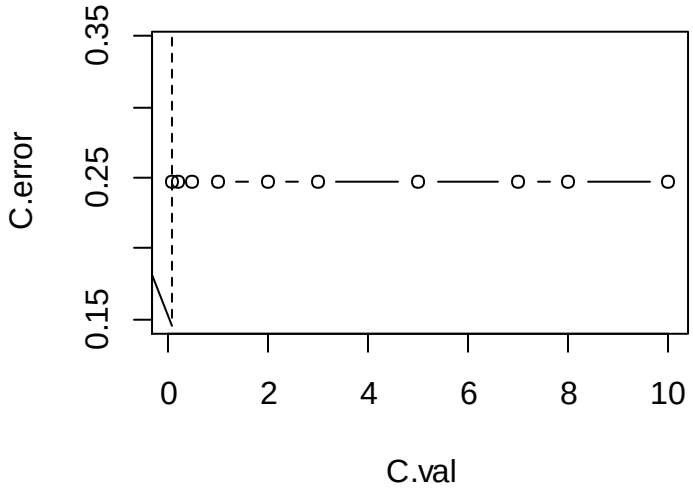
Results

The first set of results we will look at is the **K-Nearest Neighbors (knn) model**. The problem we normally have with the knn model is how we go about choosing the value 'k' when fitting our model. In general, we would not have any form of intuition about which value of k would be optimal. Therefore, the idea is to look at different values of k, and then we can examine the prediction results on the validation data. This is because we can now choose the value k, which gave us the best performance.



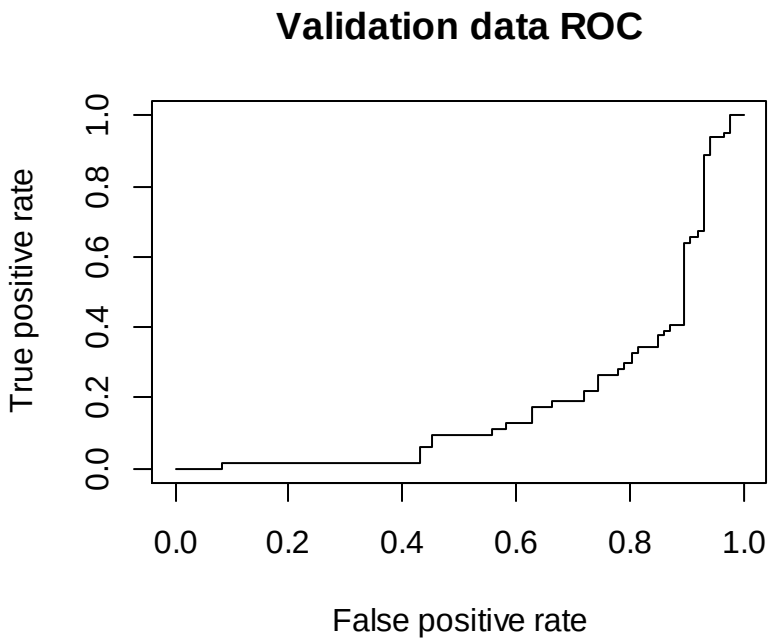
As we can see from above that k is reaching its peak around 6-8. In our R code, we see that the highest accuracy was actually k=7 (The accuracy was 79.3 %). Now knowing this we chose k=7 when we were training our data in the end as knn was chosen as the final model out of the other remaining classification algorithms.

Now looking at our next classification algorithm is **Support Vector Machines (svm)**. This process was a bit tedious as we first started out with the idea of a loop for various values of C. This was to choose C where it had the lowest error rate on the validation data. As we can see from the graph below, we had a C value of 0.1 and where we could also see the error rate being constant at around 0.25.



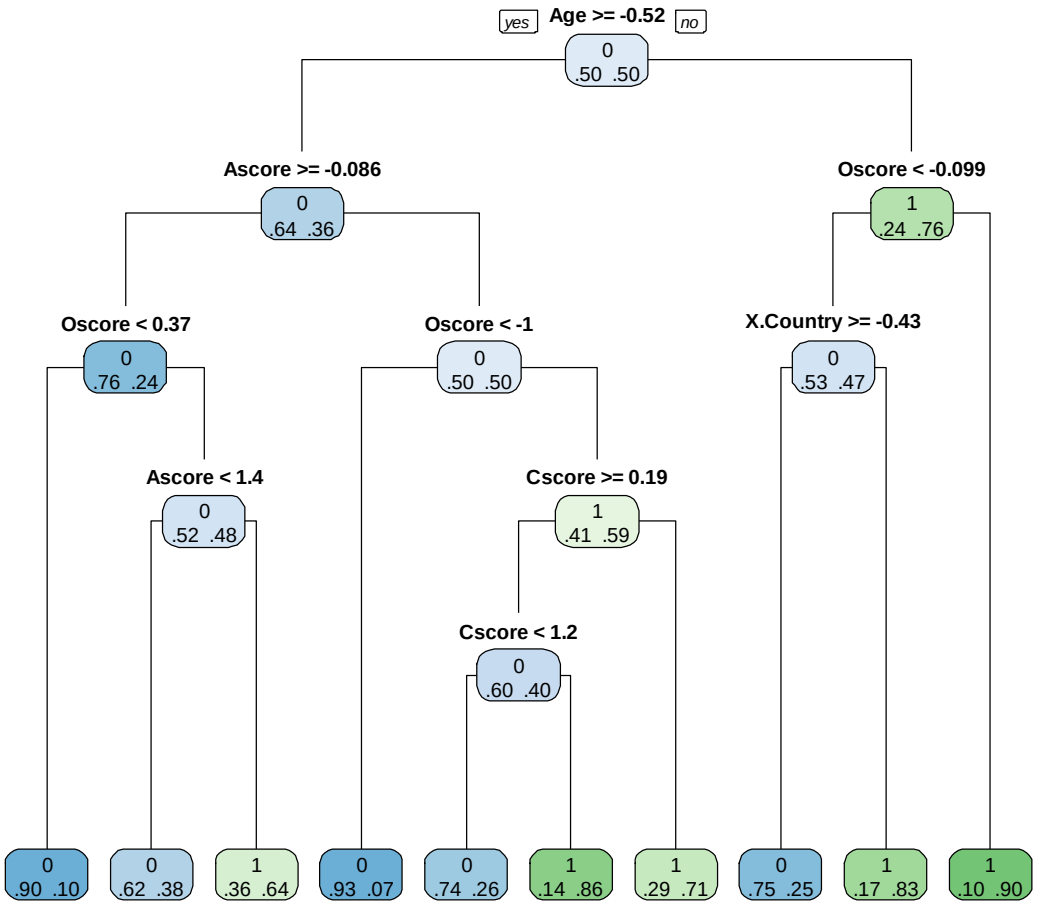
We then realize that this approach would take too long and opted for something more seamless by using the tune function and testing a variety of C values on 3 different kernels. These were Linear, Radial, and Polynomial. In doing this, we realized that our best results were the Linear Kernel , and then we used this to create our final svm model since we can easily get the cost by running the R command 'linear_tune$best.parameters'. The screenshot below is showing that the linear kernel had the highest accuracy and the lowest error rate of course.

```
> round(all_accuracies,3)
     Linear Radial Polynomial (quadr.)
[1,]  0.753  0.747                0.727
> round(all_errors,3)
     Linear Radial Polynomial (quadr.)
[1,]  0.247  0.253                0.273
>
```



Once we had done our tuning, chose the linear kernel, and built our final svm model what I looked on next was the ROC curve plot. This is used to measure the comparative performance of the model using classification error. We can see from a plot it starts off at a very low point and then increases very quickly. It's not performing extremely well but we can see it still does a fair job.

The final classification algorithm we will look at **is tree-based methods**, which covers a wide variety for example Classification Trees, Regression Trees, and so forth. What is shown below is the Classification Tree. This never had an excellent performance on our validation data. What ended up performed a bit better was our Random Forest (Bagging). We notice that the first split at the top of the tree is determined by the age. Essentially, we would follow the branches of the tree accordingly if we wanted to make a prediction on the outcome of a patient whether a drug was used at some point or never used.



Discussion

```
> print("Printing metrics for best classifcation model chosen")
[1] "Printing metrics for best classifcation model chosen"
> print(classification_rate)
[1] 0.7866667
> print(sensitivity)
[1] 0.8089888
> print(specificity)
[1] 0.7540984
>
```

We will now do a brief discussion around our final model chosen. In the end, we had chosen the k-nearest neighbors model since we saw that it performed the best on our validation data with approximately 79%. The closest that could also have been chosen was the Support Vector Machine model with approximately around 75%. With this said the idea was to use the training and validation datasets we created initially on all of our classification models built but we would use the test dataset only once when we had determined the best model. In our case as I just stated this was the knn model. Now what we needed to look on now was the metrics surrounding our final model chosen.

We saw that in terms of future performance using the test data we saw **the classification rate** was approximately 79%. This was good but honestly could have been better maybe what we could have also look into was reducing the amount of features we had. In building our model all the features were used. We noticed that we had few strong positive correlations between variables probably the filter method could have been applied and some features could have been removed. Overall, this is not a bad model in terms of accuracy. Next, we look on is **Sensitivity**. Recall that this is saying we want to see the proportion of all individuals who were correctly predicted as positives out of the number of true positives (So that mean the drug was used at some point). Lastly, we also show the percentage for **Specificity** above, which is similar but instead, we look at the proportion of individuals who were correctly predicted as negative out of the number of our true negatives (So , in other words, the drug was never used). In our context, it would have been great if our Specificity score were more higher.