CLASSIFICATION MODELS FOR PREDICTING DRUG USE

ILSIN SU SAHIN

Introduction to problem

This project's objective is to examine a dataset containing data on drug use and various patient characteristics. The dataset consists of 600 patients, 300 each described by 11 features including age, education, country of origin, ethnicity, and scores related to personality traits such as neuroticism, extraversion, openness, agreeableness, and conscientiousness. The main goal is to create a classification model that can accurately predict whether a patient will use drugs.

Process & Methodology

First, we will pre-process the dataset by analysing its structure, and encoding variables with categories. The dataset will then be divided into training and test sets, with the test set containing 30% of the data and remaining untouched during the modelling process. The training dataset will be tested to the application of four classification techniques: K-Nearest Neighbours (KNN), Classification Trees (CTs), Support Vector Machines (SVM), and Model-Based Classification.

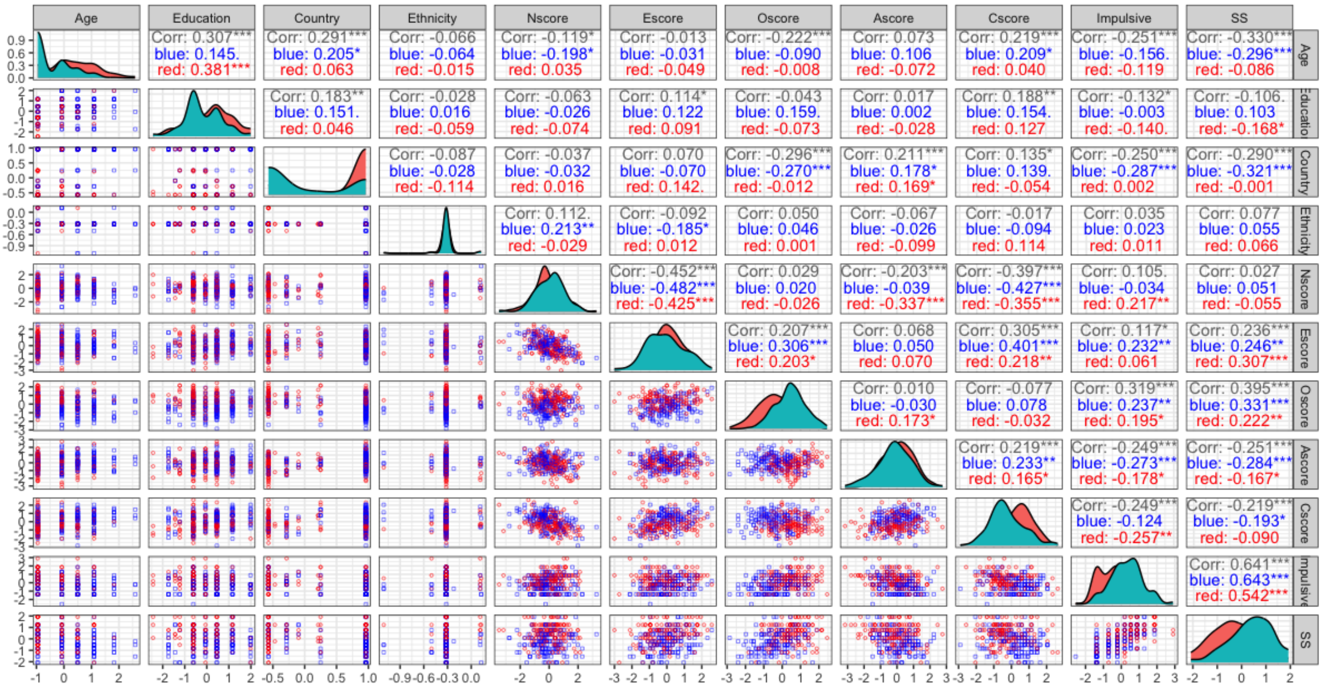
Each model will be trained using the respective algorithm, and their performance will be evaluated using metrics such as accuracy, sensitivity, specificity, area under the ROC curve, and kappa statistics. We will compare and choose the model with the best performance based on these metrics, taking its interpretability into account as well. The test set will be used to apply the selected model to predict the outcomes of drug use, and the model's accuracy will be evaluated by contrasting the predicted categories with the actual observations.

Table below provides a summary of the total observations in the dataset, along with the distribution of individuals who have "never used" drugs and those who have "used at some point," both in the test and train data subsets.

	Total Observations	Total – "Never Used"	Total – "Used at some point"
Test Data	180	88	92
Train Data	420	212	208
Total	600	300	300

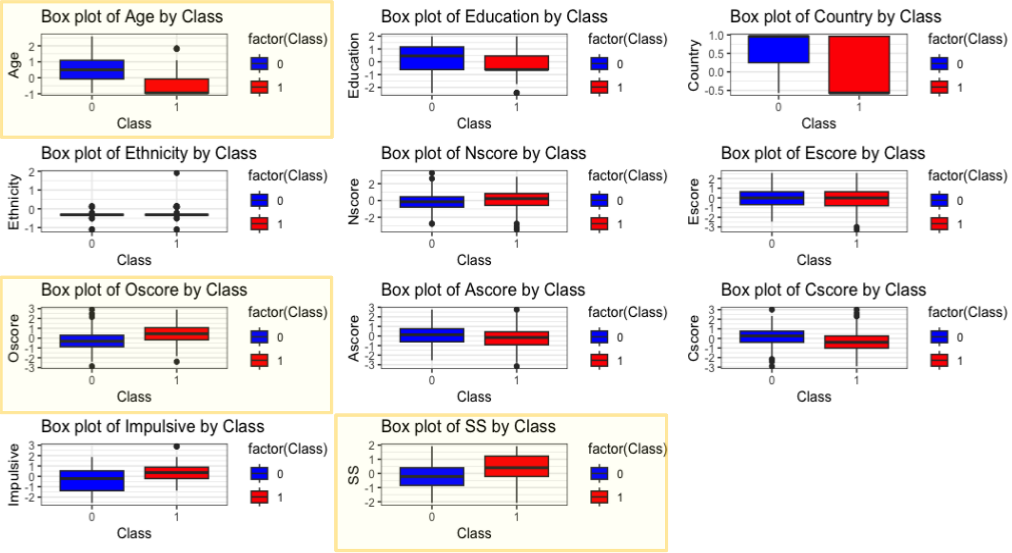
Exploratory Data Analysis

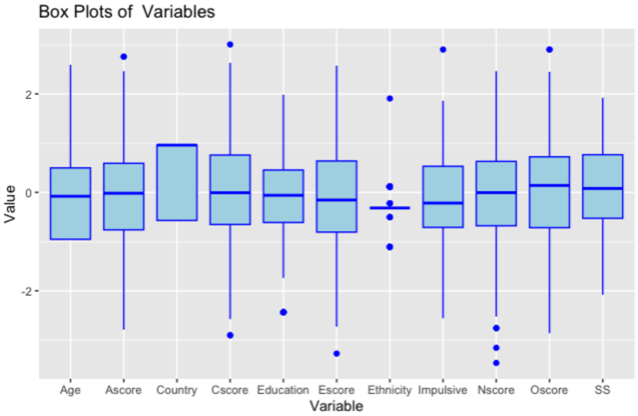
Pairs Plot



Upon examining the pairs plot, which displays the relationships between the 11 pairs of variables, categorized by patients who never used drugs (blue) and those who used drugs at some point (red), it becomes evident that there are no distinct boundaries between the categories, and no significant correlations are observed among the variables. However, a few examples from the pairs graph support this assertion.

The boxplots clearly show that the variables "Age," "Oscore," and "SS" differ noticeably between the two categories. Particularly, those who have never used drugs tend to be considerably older than those who have done so at some point in the past. These results emphasise the significance of taking particular factors into account when attempting to identify potential relationships between drug use and patient characteristics.





When examining the boxplots, the medians, and lower and upper quantiles of the vast majority of the variables are similar. The variable "Ethnicity," however, stands out because it exhibits pronounced variations in the quartiles and median values between the categories. In addition, a few outliers can be found in the data, especially in the "Nscore" variable. The variable "Country" has a higher median than the other variables, which is also noteworthy. According to these results, ethnicity and country of origin may help to distinguish between the different groups of drug users, even though the majority of variables do not show any discernible differences between them.

Classification Techniques

For each of the models built, we will test their accuracy through use of a cross-classification table. We will use these to specify the predicted versus actual class. See example below.

	Pred: "Never Used"	***Pred: "Used at some point"***
True: "Never Used"	True Positive	False Negative
True: "Used at some point"	False Positive	True Negative

Additionally, we will assess each model's performance by its classification and misclassification ability. The classification, or accuracy, rate is the total true predictions divided by the total number of observations. Simply put, this is the proportion of correct predictions, with 1 being every observation correctly predicted, and 0 no observations correctly predicted.

$$Accuracy\ Rate\ = \frac{True\ Positive\ + True\ Negative}{n}$$

K-Nearest Neighbours (KNN)

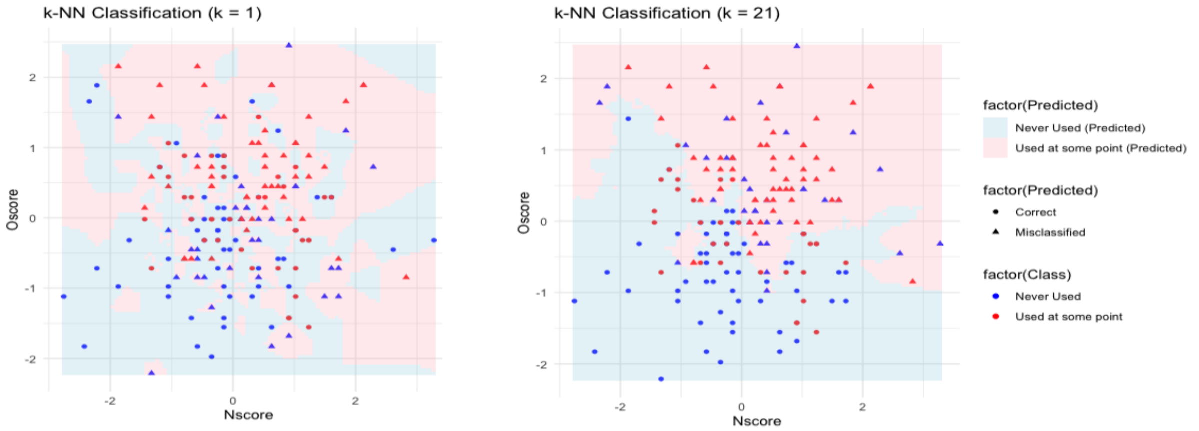
We will now discuss KNN and its application to our training data. KNN is a simple technique, which clusters observations together based on their proximity to one another. Observations, which are closer together, are grouped and classified. The distance is specified through Euclidian distance, which measures the straight-line distance between points.

$$distanceE\ (p, q)\ =\ \sqrt{(p1 - q1)^2 + \ldots + (pn - qn)^2}$$

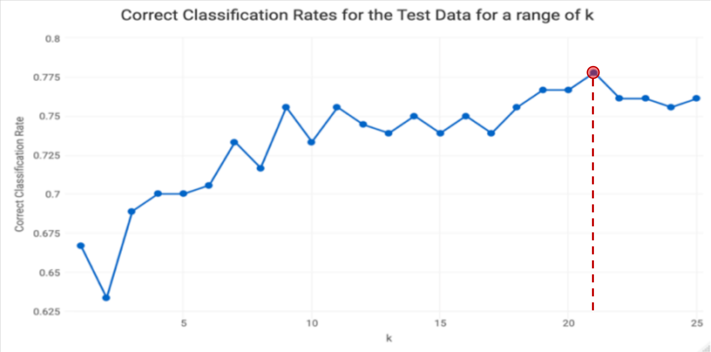
When implementing this technique, we must choose a value for K. This value specifies the number of closes neighbours the algorithm should compare against to classify new observations. For example, if K is five, then five closest neighbours will be assessed and the category with the majority out of these five observations will be used to classify the new observation. A small number for K will set smaller 'neighbourhoods' and the classifier is able to discover very subtle patterns. A larger value for K may ignore some of the noise in the data in an attempt to impose a wider and more general pattern for classification. There is no set way for choosing K, and so we will test a couple of values for K, these will be R's default setting of K=1, the number of categories in our data set K=10, and the square root of the number of observations in our training data set K=21.

Value of K	Accuracy Rate
K=1	0.67
K=10	0.75
K=21	0.79

Based on the accuracy rates you provided for different values of K, it appears that higher values of K (specifically K=10 and K=21) performed better compared to K=1. This outcome suggests that a larger neighbourhood size was more effective in classifying the new observations in your dataset. The classification results of the k-NN algorithm for different values of K are shown in the plots below. Based on their actual class labels and predicted class labels, the data points are coloured and shaped. Also shown are the decision boundaries that divide the predicted classes. It is noted that the K=1 case exhibits greater adaptability to the data than the K=21 case.



The resulting plot displays the correct classification rates for the test data as the value of k increases from 1 to 25. It helps visualize the relationship between k and the accuracy of the k-NN algorithm. The plot indicates that choosing a k value of 21 (which is close to the square root of the number of test data observations) yields a relatively high correct classification rate. This confirms that for this dataset and problem, a larger k value is more effective in achieving better classification accuracy.

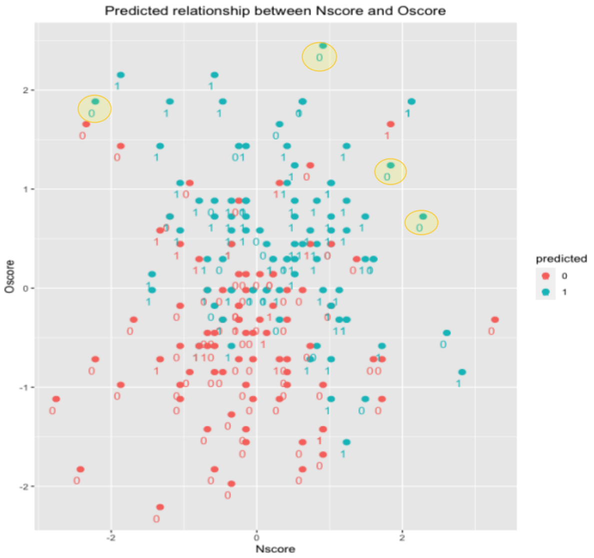


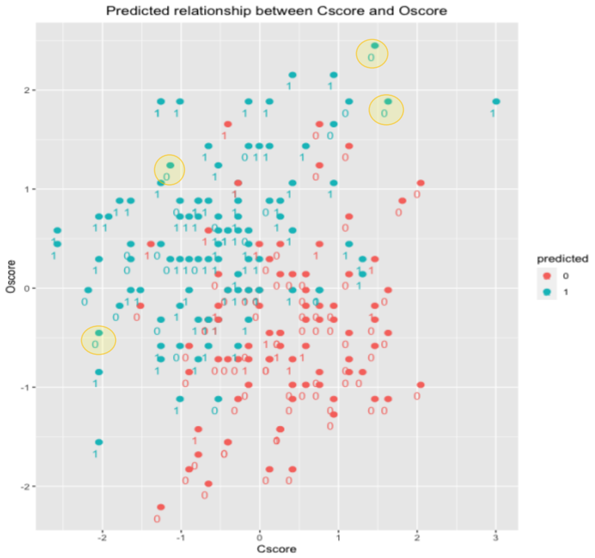
Therefore, based on the performance we can see here, we would select K=21 as our model of choice for the KNN technique. The cross- classification matrix for K=21 is as follows:

	Pred: "Never Used"	**Pred: "Used at some point"**
True: "Never Used"	157	34
True: "Used at some point"	53	176

The model's specificity (84%) is higher than its sensitivity (75%), indicating that it is more accurate at classifying patients who have never used drugs (class 0) than those who have used drugs at some point in the past (class 1). The correct classification rate is also higher for the positive class (class 1) (84%) than for the negative class (class 0) (75%). With a 79% accuracy rate, the model performs ok overall, but there is room for improvement, particularly in achieving a higher sensitivity rate.

The plots show the k-NN model's predicted relationship between particular variables and class labels. They demonstrate that the model is more likely to correctly predict instances of class 1 than class 0 instances. They reveal that the model tends to correctly predict class 1 instances more accurately than class 0 instances as mentioned previously.

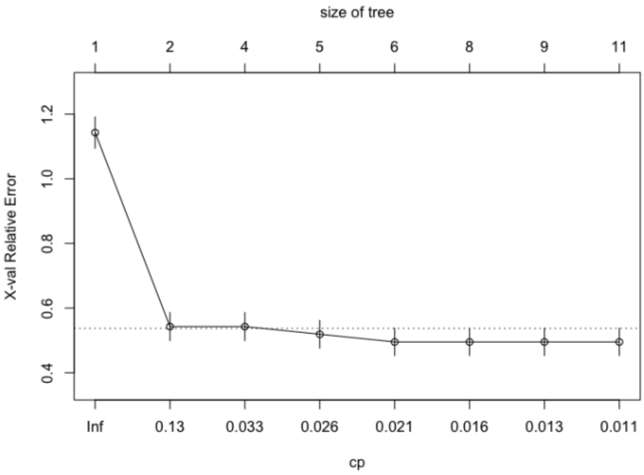




Tree-Based Clustering

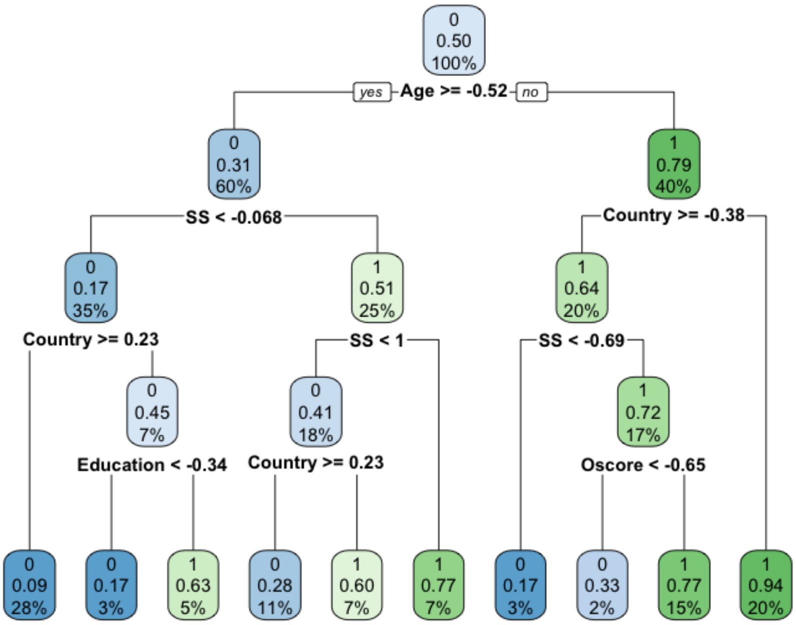
Using the knowledge gained from the training data, CTs are a type of supervised learning algorithm that analyses the training data and generates an inferred model that can be used for mapping, predicting, and classifying new observations. The interpretation of tree-based models is straightforward and intuitive because they divide the data into groups according to predicted classification.

The plot shows, as the complexity parameter (CP) decreases, indicating a more complex tree, the model achieves a lower cross-validated error. This suggests that a smaller CP allows the model to capture more intricate patterns in the data, resulting in improved predictive accuracy. Additionally, the relative error and cross-validated error decrease as the number of splits (nsplit) increases, indicating that more splits lead to better model performance.



In order to direct the tree-building process, the CT was built with a set of specific parameters. The conditions set, such as the minimum split requirement of 10 observations and a minimum bucket size of 2, ensure that nodes with insufficient data are not further partitioned. Additionally, the complexity of 0, maximum depth of 4 also restricts the tree's levels, improving interpretability and avoiding overfitting.

The classification tree generated can be seen below:



The resulting CT was analysed by examining the split conditions at each node. The initial split was based on the "Age" variable, where values greater than or equal to -0.515255 were classified as 0, while values less than this threshold were classified as 1. Within each group, further splits were performed based on other variables, such as "SS" and "Country," to refine the classification. This hierarchical structure allows for the identification of key variables that contribute to the prediction process.

The cross-classification table for the decision tree is presented as follows:

	Pred: "Never Used"	**Pred: "Used at some point"**
True: "Never Used"	165	31
True: "Used at some point"	45	179

The model's accuracy has been estimated to be 0.7611, meaning that in roughly 76.11% of cases, it predicts the class labels correctly. According to the model's sensitivity and specificity values, 73.33% of instances in the "Never Used" class are correctly identified (sensitivity), and 78.89% of instances in the "Used at Some Point" class are correctly classified (specificity). The model's positive and negative predictive values (PPV and NPV, respectively) are 0.7765 and 0.7474. These values indicate the proportions of true positive and true negative predictions compared to all positive and negative predictions, respectively. Overall, with a correct classification rate of 0.79 for the "Never Used" class and 0.85 for the "Used at some point" class, the model performs reasonably well overall.

Support Vector Machines

SVM work best with problems with two classes, which applies to our situation. Additionally, they are a preferred method when the data set contains more variables than observations, which is not the case in our situation. SVM aim to identify the best separating line (vector) between our categories. In SVM this decision is based on a quantity known as the 'margin', and we want the margin to be as big as possible to separate our patients never used the drug and patients have used the drug at some point of their life.

We have performed model fitting using linear, radial, and polynomial Support Vector Machines (SVM) to determine the best fit for our model. The table below presents the error rates associated with each method:

Error Rates		
Linear	**Radial**	**Polynomial**
0.2205	0.2143	0.2237

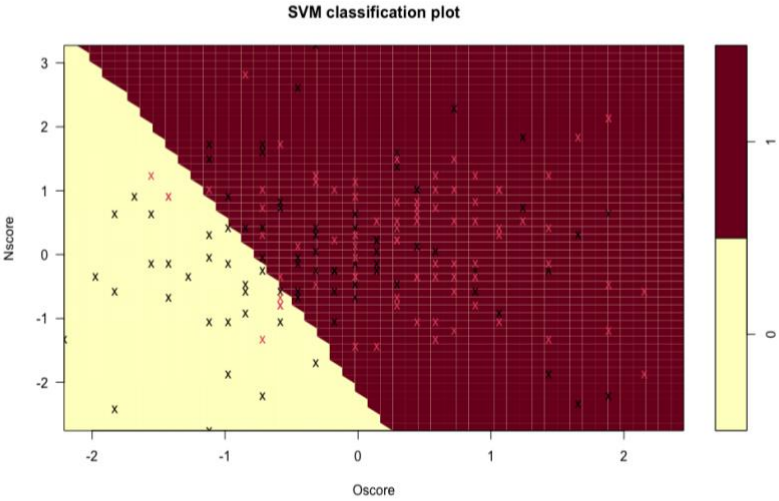
The radial SVM model has the lowest error rate out of the three when the error rates are compared. This suggests that when compared to the linear and polynomial SVM models, the radial SVM model predicts the target variable with higher accuracy and precision. As a result, it makes sense to select the radial SVM model as the best fit for the provided data given the lower error rate.

We concentrated mainly on two important parameters: cost and gamma, in order to construct an efficient SVM model. The cost parameter controls the trade-off between the number of misclassifications and the permitted margin error. A lower cost value strives for fewer misclassifications at the expense of a narrower margin while a higher cost value allows for more misclassifications. For the best decision-making, cost and margin functionality must be balanced. On the other hand, the curvature of the decision boundary is controlled by the kernel parameter gamma. While a lower gamma value produces a smoother decision boundary, a higher gamma value results in a more complex decision boundary with increased sensitivity to minute differences in the data. It's crucial to choose a gamma value that improves accuracy without overfitting the model and jeopardising its ability to generalise to new observations.

To determine the best value for gamma and cost, we simulate various combinations of these parameters, with cost ranging between 0.01 and 10 in increments of 0.05, and gamma between 0 ad 1 in increments of 0.005. This test finds that a cost of 0.01 and a gamma of 0.045 fits best to our data.

The radial Support Vector Machine (SVM) model's predictions for the test data are displayed in the plot that is provided. The points on the plot are color-coded according to the expected class labels, and it shows the relationship between the "Nscore" and "Oscore" variables in the test data.

The plot shows that the model predicts Class 0 data more accurately than Class 1 data. A greater percentage of the points classified as Class 0 are correctly predicted and fall within a particular area, demonstrating a higher level of accuracy for Class 0 predictions. For Class 1, the distribution is more dispersed, which suggests a lower level of accuracy and more misclassifications.



The cross-classification table for the SVM is presented as follows:

	Pred: "Never Used"	**Pred: "Used at some point"**
True: "Never Used"	162	42
True: "Used at some point"	48	168

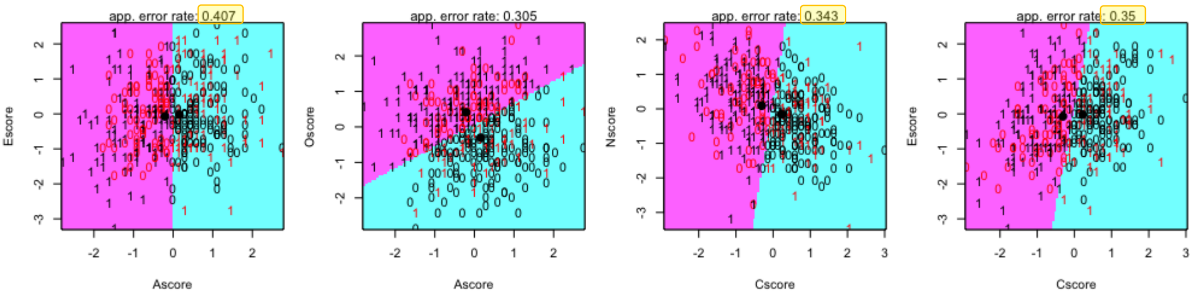
The SVM model achieved an overall accuracy of 78.57%, indicating that approximately 78.57% of the test instances were correctly classified. The sensitivity for Class 0 was 77.14%, demonstrating the model's ability to correctly identify instances belonging to Class 0. The specificity for Class 1 was 80.00%, indicating the model's ability to correctly identify instances belonging to Class 1. The positive predictive value (PPV) for Class 0 was 79.41%, meaning that approximately 79.41% of instances predicted as Class 0 were true positives. The negative predictive value (NPV) for Class 1 was 77.78%, indicating that approximately 77.78% of instances predicted as Class 1 were true negatives.

Model Based Classification

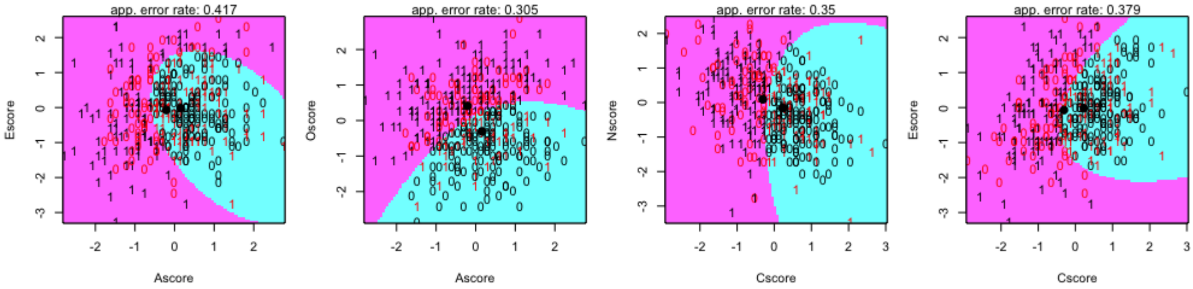
Model-Based classification methods based on statistical models, such as linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA), are effective tools for this task. LDA assumes equal class covariances and constructs linear decision boundaries, making it suitable for linearly separable classes and providing interpretable results.

We have performed model fitting using LDA, and QDA to determine the best fit for our model. To assess and compare the performance of these models, we utilized various evaluation metrics, including error rates.

Partition Plot- LDA Model



Partition Plot- QDA Model



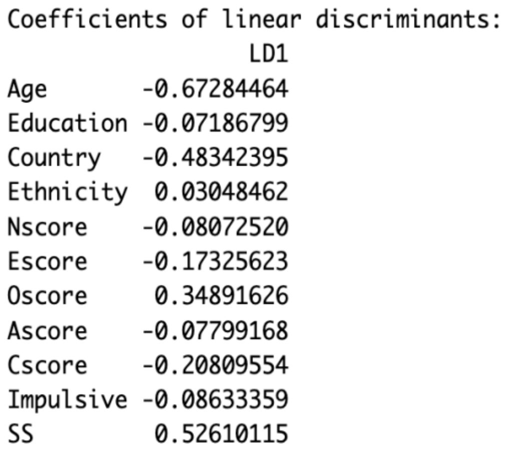
We found that LDA consistently outperformed QDA during our analysis, showing a slightly lower error rate. This suggests that in our dataset, LDA is better able to accurately classify instances than QDA. To assess the models further each model’s error rate on train data is calculated. The table below presents the error rates associated with each method:

Error Rates	
LDA	**QDA**
0.1905	0.2005

Following an evaluation of the training data and the performance of Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA), we discovered that LDA had a lower error rate than QDA, with a 0.1905 error rate compared to 0.2005 for QDA. As a result, LDA's advantage over QDA in terms of classification performance in our dataset is further supported. It shows that LDA is marginally more accurate in classifying instances in the training data. Therefore, we will pursue with LDA model for our analysis.

By analysing these coefficients, we can determine which variables are better at discriminating between the classes. From the given coefficients, it can be observed that variables such as Oscore (0.3489), SS (0.5261), Age (-0.6728), Country (-0.4834), and Cscore (0.2081) have relatively larger coefficients. These variables contribute significantly to the linear combination that separates the two classes. Higher values of these variables are associated with Class 1.

Based on the coefficients, it can be concluded that Oscore and SS are the variables that demonstrate the strongest discriminatory power in the LDA model.. Additionally, Age, Country, and Cscore also contribute significantly to the separation between the classes.



```
Coefficients of linear discriminants:
                    LD1
Age         -0.67284464
Education   -0.07186799
Country     -0.48342395
Ethnicity    0.03048462
Nscore      -0.08072520
Escore      -0.17325623
Oscore       0.34891626
Ascore      -0.07799168
Cscore      -0.20809554
Impulsive   -0.08633359
SS           0.52610115
```

The cross-classification table for the LDA is presented as follows:

	Pred: “Never Used”	**Pred: “Used at some point”**
True: “Never Used”	167	37
True: “Used at some point”	43	173

Out of 204 instances belonging to Class 0, 167 were correctly classified, and out of 216 instances belonging to Class 1, 173 were correctly classified, according to the cross-classification table for the Linear Discriminant Analysis (LDA) model. The overall accuracy of the model was 80.95%. It demonstrated an 82.38% specificity for Class 1 and a 79.52% sensitivity for Class 0. Class 0 had a positive predictive value of 81.86%, while Class 1 had a negative predictive value of 80.09%. These findings show that the LDA model did a good job of correctly classifying instances into the two classes.

Comparison

We can make comparisons between the four classification models using the provided accuracy, sensitivity, specificity, and kappa figures: Linear Discriminant Analysis (LDA), Support Vector Machine with Radial Basis Function kernel (SVM), K-Nearest Neighbours (KNN), Decision Tree (CT). The results are as follows:

	CTs	KNN	SVM	LDA
Accuracy	0.69	0.77	0.79	0.74
Sensitivity	0.69	0.73	0.74	0.71
Specificity	0.68	0.81	0.84	0.78
Kappa	0.38	0.54	0.59	0.49

The overall accuracy of the model's predictions is measured by accuracy. SVM RADIAL performs best in this comparison with an accuracy of 0.79, closely followed by KNN with 0.77. LDA has the highest accuracy score of 0.74, while Decision Tree has the lowest accuracy score of 0.69. As a result, when compared to LDA and Decision Tree, SVM L and KNN are more effective at making precise predictions.

Sensitivity, also referred to as the true positive rate, measures how well a model can distinguish positive examples from the true positive class. Strong sensitivity values of 0.74 and 0.73 for SVM RADIAL and KNN respectively show their ability to capture positive instances. The sensitivity scores for LDA and Decision Tree are slightly lower, at 0.71 and 0.69, respectively.

The model's ability to correctly distinguish negative instances from the actual negative class is measured by specificity, also known as the true negative rate. With a specificity of 0.84, SVM outperforms all other models, demonstrating its ability to precisely identify negative instances. While LDA and Decision Tree only manage specificity scores of 0.78 and 0.68, respectively, KNN also displays promising specificity with a score of 0.81.

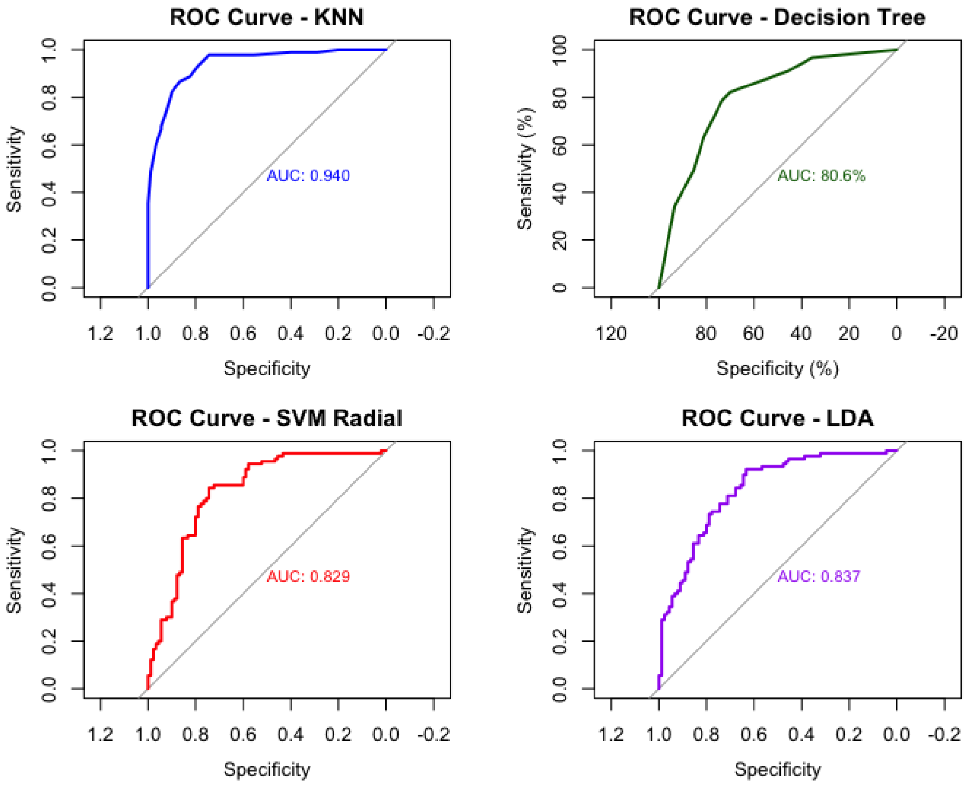
The performance of the models is evaluated using kappa, a statistic that considers the degree of agreement between the model's predictions and the actual class labels while taking the possibility of random chance into account. The highest kappa value is obtained by SVM (0.59), closely followed by KNN (0.54). LDA has a kappa score of 0.49, while Decision Tree has the lowest score at 0.38.

SVM consistently outperformed the other models when these evaluation metrics were considered, displaying higher accuracy, sensitivity, specificity, and kappa values. As a result, among CTs, KNN, SVM, and LDA for the given dataset, the SVM model can be said to be the best-performing model based on this analysis.

Overall, the results show that SVM, which consistently achieves high accuracy, sensitivity, specificity, and kappa values, is the best-performing model out of the four. KNN, the second-best model, exhibits competitive performance as well, excelling particularly in accuracy and sensitivity. LDA performs moderately well but trails SVM and KNN in the majority of metrics. With lower scores across all evaluation metrics, Decision Tree performs the least favourably, indicating that it may not be the best option for this specific classification task.

ROC Curves

A binary classification model's effectiveness at various classification thresholds can be evaluated graphically using a ROC (Receiver Operating Characteristic) curve. The trade-off between the true positive rate (sensitivity or recall) and the false positive rate (1 - specificity) as the classification threshold changes is visually represented by the ROC curve. The Area Under the ROC Curve (AUC) is a single scalar value that quantifies the overall performance of the model. By comparing the ROC curves of different models we can assess their performance. A model with a curve closer to the top-left corner indicates better performance.



KNN has the highest ROC score, which shows that it can effectively distinguish between positive and negative instances. It also performs well on the dataset overall, as evidenced by its high accuracy, sensitivity, specificity, and kappa score. The model seems to generalise well, but it's important to remember that because it must determine distances to every training point, prediction on large datasets may incur increased computational costs.

In comparison to KNN, the Decision Tree model has a lower ROC score, which suggests that it may not be as effective at differentiating between positive and negative instances. The model may not generalise as well because it also has relatively lower accuracy, sensitivity, specificity, and kappa scores. Although the results indicate that the Decision Tree may be prone to overfitting on the dataset, it is still simple to interpret.

The ROC of SVM with Radial Basis Function Kernel is moderate. The model performs ok and is particularly good at detecting non-linear relationships in the data. Additionally, it exhibits comparatively high specificity, indicating its ability to accurately recognise negative instances. Even though SVM requires a lot of computation, it performs well on the dataset.

The ROC score of LDA is comparable to that of SVM, and it exhibits a fair amount of accuracy, sensitivity, specificity, and kappa score. Although LDA is a helpful dimensionality reduction method, it might not be as good at capturing complex decision boundaries or managing non-linear relationships between features and classes as SVM or KNN.

Conclusion & Recommendation for Improvements

With this project, we aimed to create a classification model that could forecast drug use based on various patient traits. K-Nearest Neighbours (KNN), Decision Trees (CTs), Support Vector Machines (SVM), and Linear Discriminant Analysis (LDA) were the four classification methods we investigated. Using a variety of performance metrics, including accuracy, sensitivity, specificity, kappa, and ROC curves, we assessed these models. Based on the analysis, KNN was the second best-performing model, closely followed by SVM with a Radial Basis Function kernel. The Decision Tree model performed the least favourably, while LDA displayed a moderate level of performance.

Recommendations for Improvements:

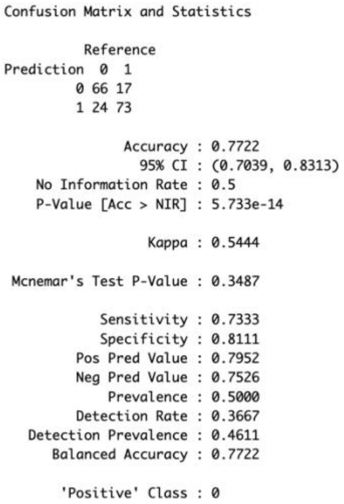
1. Feature Selection: The dataset has eleven features, but it's possible that not all of them are equally useful for predicting drug use. To find and keep only the most pertinent and instructive features, we can consider using feature selection techniques like correlation analysis, feature importance ranking, or stepwise regression.

2. Feature Engineering: We can consider engineering new features that could capture significant relationships between the existing variables in addition to the feature selection process. The performance of the model might be enhanced by, for instance, combining specific variables or developing interaction terms.

3. Ensemble Methods: By combining the features of various models like ensemble techniques like Random Forest or Gradient Boosting we can improve predictive accuracy.

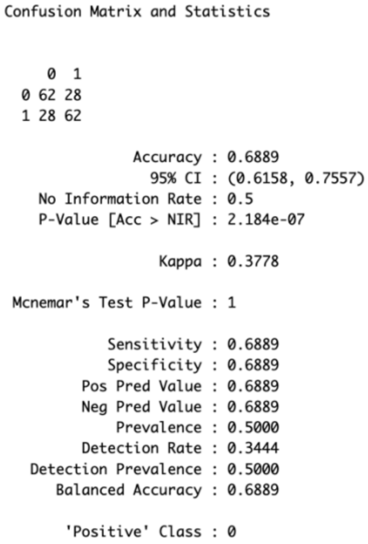
Appendix

Table 1: KNN - Confusion Matrix



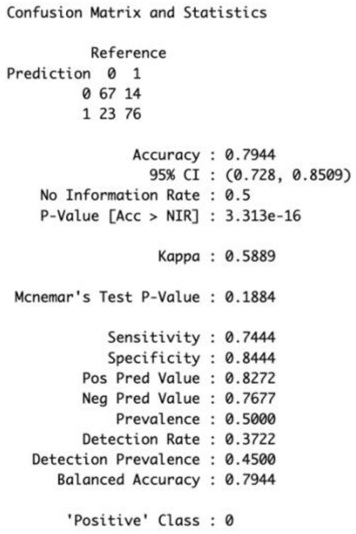
```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 66 17
         1 24 73

               Accuracy : 0.7722
                 95% CI : (0.7039, 0.8313)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : 5.733e-14

                  Kappa : 0.5444

 Mcnemar's Test P-Value : 0.3487

            Sensitivity : 0.7333
            Specificity : 0.8111
         Pos Pred Value : 0.7952
         Neg Pred Value : 0.7526
             Prevalence : 0.5000
         Detection Rate : 0.3667
   Detection Prevalence : 0.4611
      Balanced Accuracy : 0.7722

       'Positive' Class : 0
```

Table 2: Decision Tree - Confusion Matrix



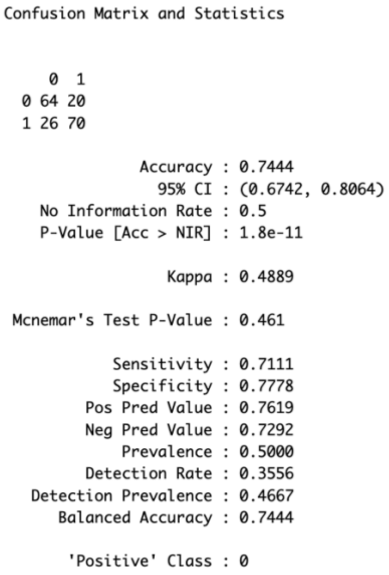
```
Confusion Matrix and Statistics


     0  1
  0 62 28
  1 28 62

               Accuracy : 0.6889
                 95% CI : (0.6158, 0.7557)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : 2.184e-07

                  Kappa : 0.3778

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.6889
            Specificity : 0.6889
         Pos Pred Value : 0.6889
         Neg Pred Value : 0.6889
             Prevalence : 0.5000
         Detection Rate : 0.3444
   Detection Prevalence : 0.5000
      Balanced Accuracy : 0.6889

       'Positive' Class : 0
```

Table 3: SVM - Confusion Matrix



```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 67 14
         1 23 76

               Accuracy : 0.7944
                 95% CI : (0.728, 0.8509)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : 3.313e-16

                  Kappa : 0.5889

 Mcnemar's Test P-Value : 0.1884

            Sensitivity : 0.7444
            Specificity : 0.8444
         Pos Pred Value : 0.8272
         Neg Pred Value : 0.7677
             Prevalence : 0.5000
         Detection Rate : 0.3722
   Detection Prevalence : 0.4500
      Balanced Accuracy : 0.7944

       'Positive' Class : 0
```

Table 4: LDA – Confusion Matrix



```
Confusion Matrix and Statistics


    0  1
 0 64 20
 1 26 70

               Accuracy : 0.7444
                 95% CI : (0.6742, 0.8064)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : 1.8e-11

                  Kappa : 0.4889

 Mcnemar's Test P-Value : 0.461

            Sensitivity : 0.7111
            Specificity : 0.7778
         Pos Pred Value : 0.7619
         Neg Pred Value : 0.7292
             Prevalence : 0.5000
         Detection Rate : 0.3556
   Detection Prevalence : 0.4667
      Balanced Accuracy : 0.7444

       'Positive' Class : 0
```