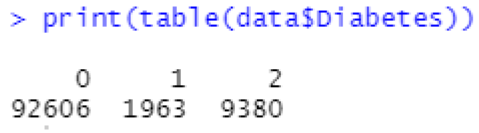
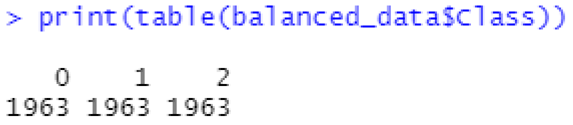
Data preparation efforts post Milestone 2

As shown below our original dataset was highly unbalanced over the Diabetes class. The number of survey participants without diabetes was far higher than those that were pre-diabetic or diabetic



```
> print(table(data$Diabetes))

    0     1     2 
92606  1963  9380 
```

To balance our dataset we used the downSample function in R and undersampled the majority class (Diabetes ==0 and Diabetes ==2) to match the size of the minority class (Diabetes == 1). The resultant dataset had the following distribution



```
> print(table(balanced_data$Class))

   0    1    2 
1963 1963 1963 
```

Having this balanced dataset ensures that our model does not make biased predictions towards the majority class leading to better generalization and performance on minority classes.

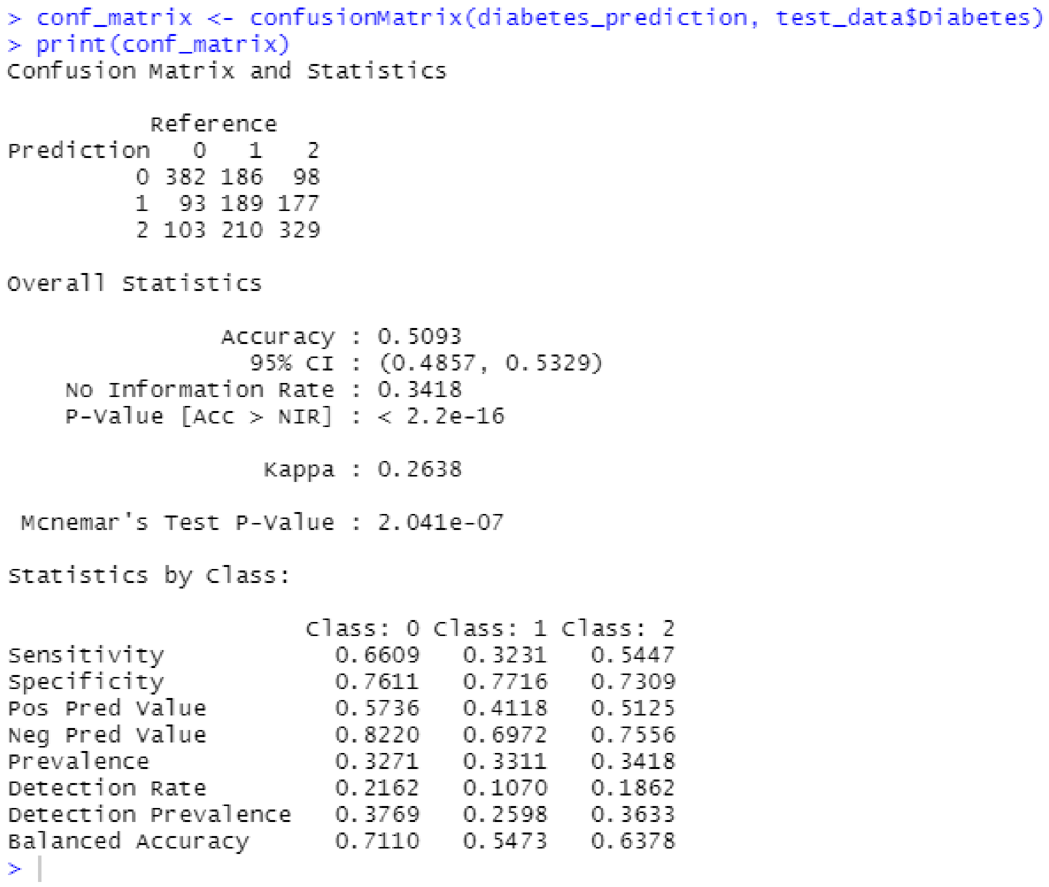
Question 1. SVMs

In this question you are expected to explore how to apply SVMs to answer your research question(s).

a. Train and test a linear SVM model

We used the ‘ksvm’ function from the ‘kernlab’ package in R to train a linear SVM classifier. Our response variable has 3 classes and the ksvm function by default uses the One-vs-One approach for this multi class classification problem. In this approach a binary classifier is created for each possible pair of classes (Class 0 vs Class 1, Class 0 vs Class 2 , Class 1 vs Class 2)

Following table shows the confusion matrix and the results we obtained



```
> conf_matrix <- confusionMatrix(diabetes_prediction, test_data$Diabetes)
> print(conf_matrix)
Confusion Matrix and Statistics

          Reference
Prediction   0   1   2
         0 382 186  98
         1  93 189 177
         2 103 210 329

Overall Statistics

               Accuracy : 0.5093
                 95% CI : (0.4857, 0.5329)
    No Information Rate : 0.3418
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.2638

 Mcnemar's Test P-Value : 2.041e-07

Statistics by Class:

                     Class: 0 Class: 1 Class: 2
Sensitivity            0.6609   0.3231   0.5447
Specificity            0.7611   0.7716   0.7309
Pos Pred Value         0.5736   0.4118   0.5125
Neg Pred Value         0.8220   0.6972   0.7556
Prevalence             0.3271   0.3311   0.3418
Detection Rate         0.2162   0.1070   0.1862
Detection Prevalence   0.3769   0.2598   0.3633
Balanced Accuracy      0.7110   0.5473   0.6378
> 
```

For this model the statistics are:

	Class 0 (Non Diabetic)	Class 1 (Pre-Diabetic)	Class 2 (Diabetic)
Precision	0.57	0.41	0.51
Recall	0.66	0.32	0.54
Specificity	0.76	0.77	0.73
Accuracy	0.71	0.55	0.64
F1 Score	0.61	0.36	0.53

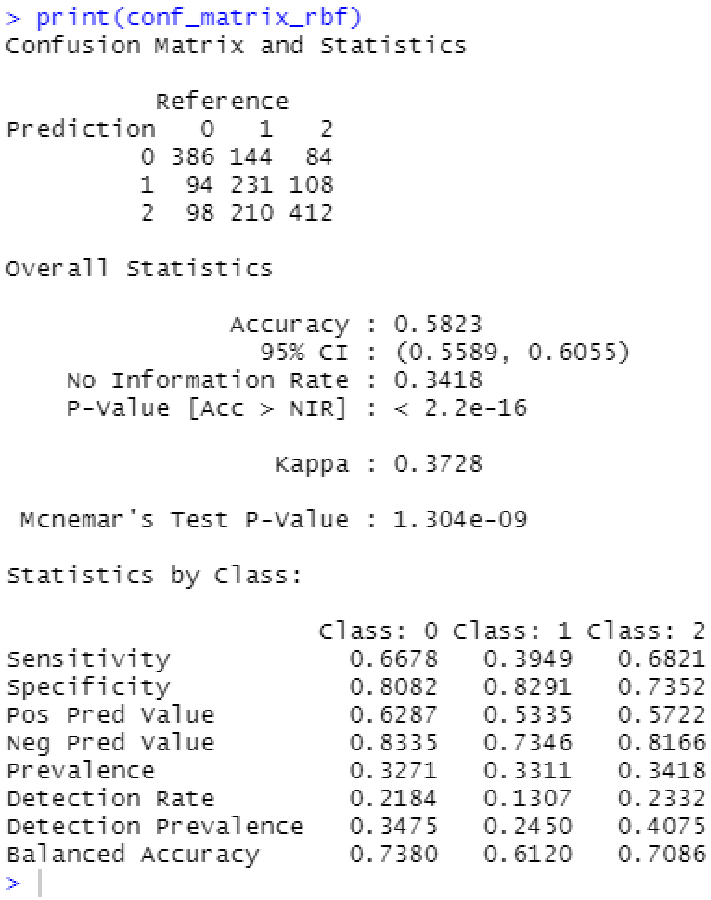
The confusion matrix and sensitivity values indicate that the linear classifier performs well in correctly predicting non-diabetic (class 0) and diabetic (class 2) participants. However, a substantial number of pre-diabetic participants are misclassified, suggesting a challenge in accurately identifying this specific class.

b. Kernels

Replicate analyses in (a) for at least another type of non-linear kernel.

For both (a) and (b), report either the confusion matrix, the precision, recall, specificity, accuracy and the F1 measure for each class in your setting, or the RMSE for regression settings.

Using the radial basis function (rbf) kernel, we got the following results



```
> print(conf_matrix_rbf)
Confusion Matrix and Statistics

          Reference
Prediction   0   1   2
         0 386 144  84
         1  94 231 108
         2  98 210 412

Overall Statistics
                                          
               Accuracy : 0.5823          
                 95% CI : (0.5589, 0.6055)
    No Information Rate : 0.3418          
    P-Value [Acc > NIR] : < 2.2e-16       
                                          
                  Kappa : 0.3728          
                                          
 Mcnemar's Test P-Value : 1.304e-09       

Statistics by Class:

                     Class: 0 Class: 1 Class: 2
Sensitivity            0.6678   0.3949   0.6821
Specificity            0.8082   0.8291   0.7352
Pos Pred Value         0.6287   0.5335   0.5722
Neg Pred Value         0.8335   0.7346   0.8166
Prevalence             0.3271   0.3311   0.3418
Detection Rate         0.2184   0.1307   0.2332
Detection Prevalence   0.3475   0.2450   0.4075
Balanced Accuracy      0.7380   0.6120   0.7086
> 
```

	Class 0 (Non Diabetic)	Class 1 (Pre-Diabetic)	Class 2 (Diabetic)
Precision	0.63	0.53	0.57
Recall	0.67	0.39	0.68
Specificity	0.80	0.83	0.73
F1 Score	0.65	0.45	0.62
Accuracy	0.74	0.61	0.70

Similar to the linear classifier the rbf classifier is also using a One vs One approach.

The RBF kernel generally shows improved results compared to the linear kernel in terms of accuracy, sensitivity, and specificity for each class. The overall accuracy of the model increased from 50% to 58%. The model has also shown a marked increase in the classification accuracy for the pre-diabetes class.

We can see that the RBF kernel is particularly effective when dealing with non-linear relationships in the data, and it has captured more complex patterns than the linear kernel.

Question 2. Neural Networks

In this question, you will explore the applicability of Neural Networks to your research question(s). Remember to normalize the input variables to [0-1] ranges.

You are expected to do an exhaustive evaluation of the NN prediction results for different hidden layers and report: (1) performance results, (2) activation function and (3) number of neurons per layer for each case. Also, plot the best neural network model.

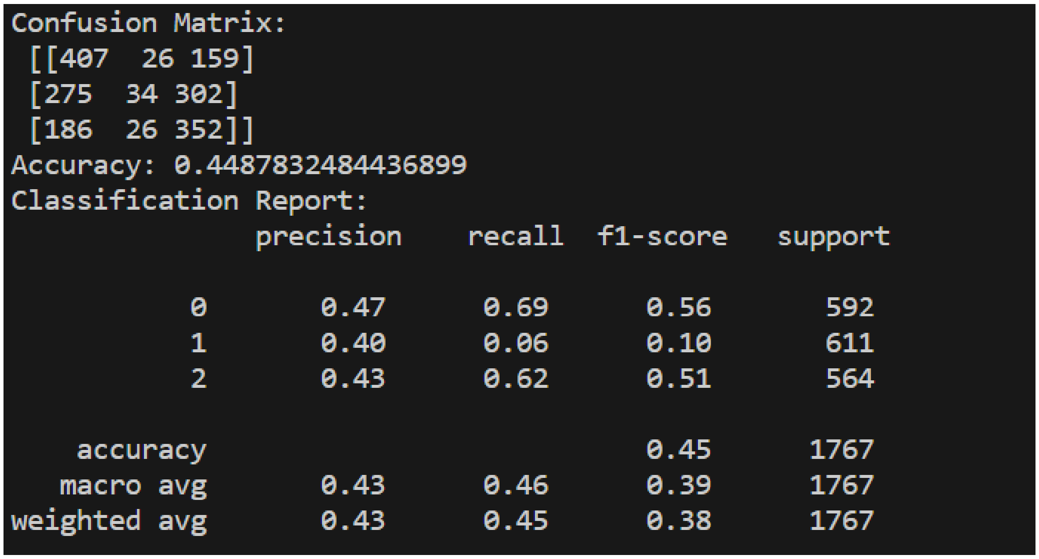
For this particular analysis, we chose not to include categorical predictor variables with multiple levels (e.g Income_category which has 11 classes). The decision stems from the complexity associated with normalizing these variables, which would entail one-hot encoding. This process would result in a substantial increase in the number of input variables for each class, posing challenges in terms of both model execution and interpretability.

We normalized our continuous variables “Age”, “BMI”, “Average_Drink”, “Alcohol_consumed”

For the variables that are categorical and binary, we left them as is since their values are already either 0 or 1, requiring no normalization.

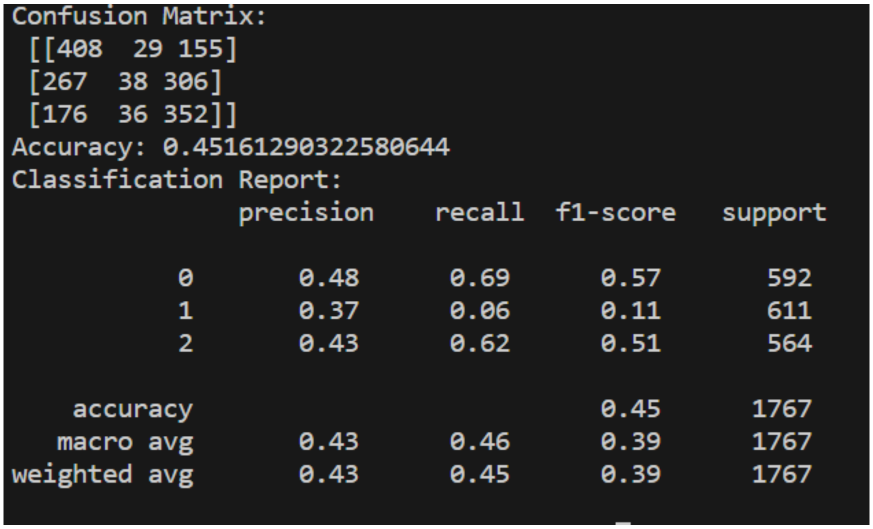
The neural network was designed in Python using scikit since the neuralnet package in R was taking a lot of time to run the models. We experimented with various setups, adjusting the number of hidden layers, hidden neurons, and activation functions. Below are some instances of the trials conducted, along with their corresponding performance outcomes.

Activation function: Logit
Two hidden layers with 5 and 3 neurons



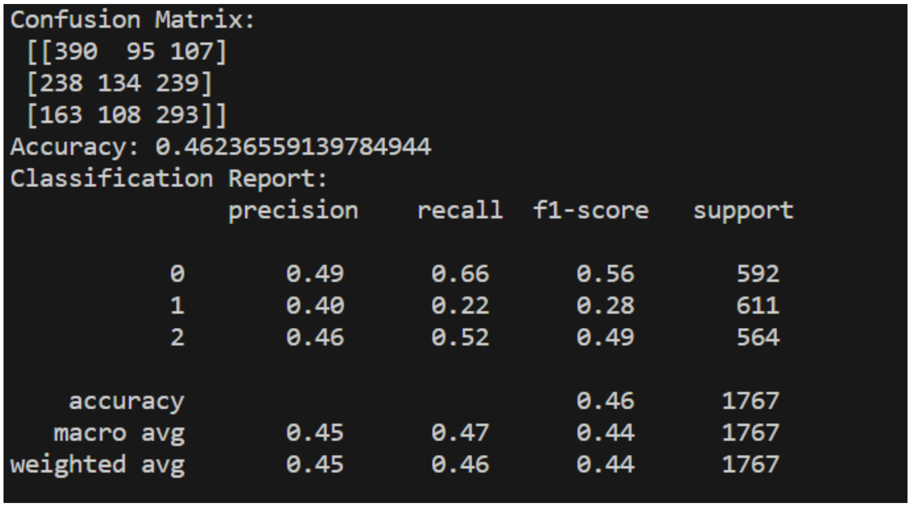
```
Confusion Matrix:
 [[407  26 159]
 [275  34 302]
 [186  26 352]]
Accuracy: 0.4487832484436899
Classification Report:
               precision    recall  f1-score   support

           0       0.47      0.69      0.56       592
           1       0.40      0.06      0.10       611
           2       0.43      0.62      0.51       564

    accuracy                           0.45      1767
   macro avg       0.43      0.46      0.39      1767
weighted avg       0.43      0.45      0.38      1767
```

Activation function: ReLU
Three hidden layers with 6,4 and 2 neurons



```
Confusion Matrix:
 [[408  29 155]
 [267  38 306]
 [176  36 352]]
Accuracy: 0.45161290322580644
Classification Report:
              precision    recall  f1-score   support

           0       0.48      0.69      0.57       592
           1       0.37      0.06      0.11       611
           2       0.43      0.62      0.51       564

    accuracy                           0.45      1767
   macro avg       0.43      0.46      0.39      1767
weighted avg       0.43      0.45      0.39      1767
```

Activation function: tanh
Three hidden layers with 6,4 and 2 neurons



```
Confusion Matrix:
 [[390  95 107]
 [238 134 239]
 [163 108 293]]
Accuracy: 0.46236559139784944
Classification Report:
              precision    recall  f1-score   support

           0       0.49      0.66      0.56       592
           1       0.40      0.22      0.28       611
           2       0.46      0.52      0.49       564

    accuracy                           0.46      1767
   macro avg       0.45      0.47      0.44      1767
weighted avg       0.45      0.46      0.44      1767
```

In the end we found that keeping the tanh activation function along with a 5,3,3 hidden neuron architecture gave us the best results with an accuracy value of close to 50%

Confusion Matrix:

 [[391 109 92]

 [237 189 185]

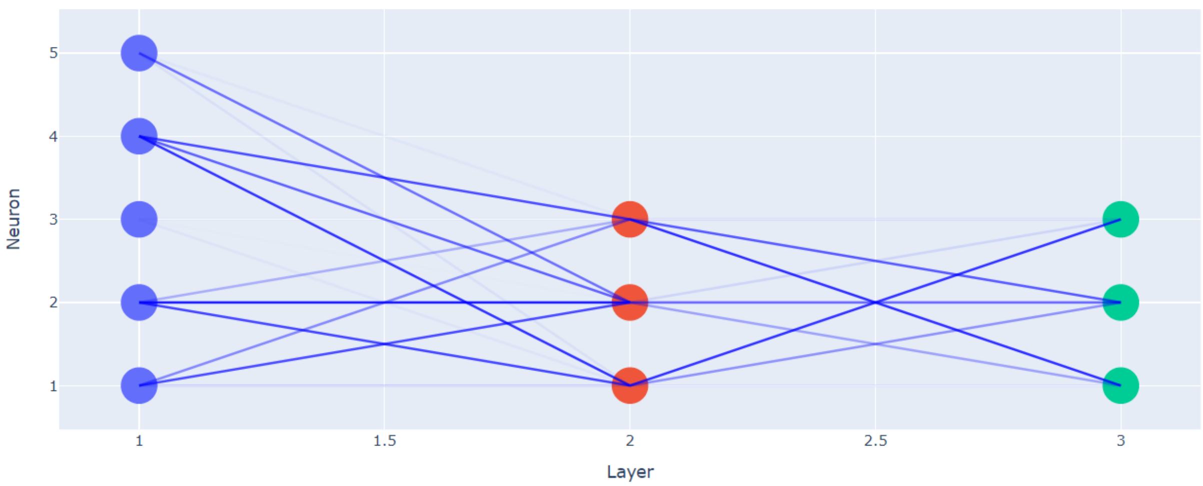
 [162 135 267]]

Accuracy: 0.479343520090549

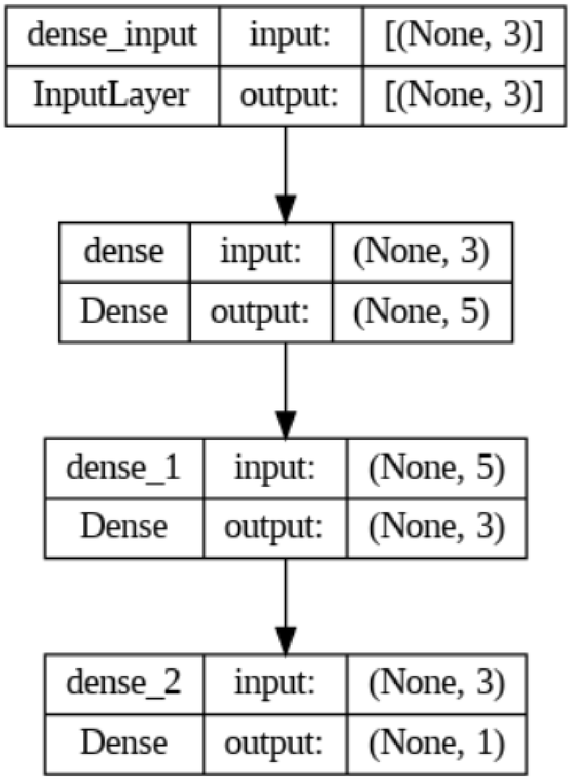
Classification Report:

```
             precision    recall  f1-score   support

           0       0.49      0.66      0.57       592
           1       0.44      0.31      0.36       611
           2       0.49      0.47      0.48       564

    accuracy                           0.48      1767
   macro avg       0.47      0.48      0.47      1767
weighted avg       0.47      0.48      0.47      1767
```

Following are the plots of this neural network (obtained using matplotlib)



Plot 1



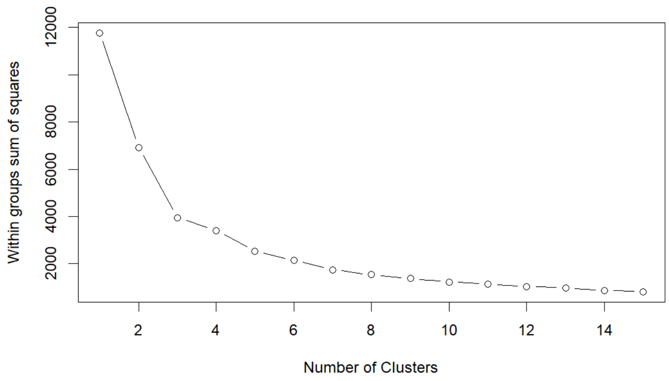
Plot 2

Question 3. Clustering

We decided to go ahead with approach 1, where we have only considered the numerical variables in our dataset (BMI and Alcohol_consumed)

K Means Clustering:

1. The best number of clusters: 3



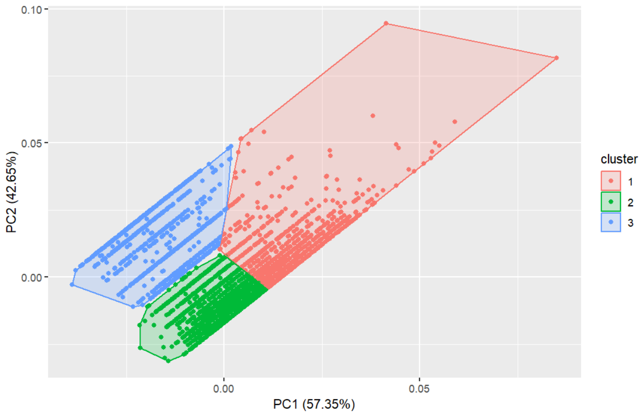
2. Number of elements per cluster:

   Cluster1- 1453

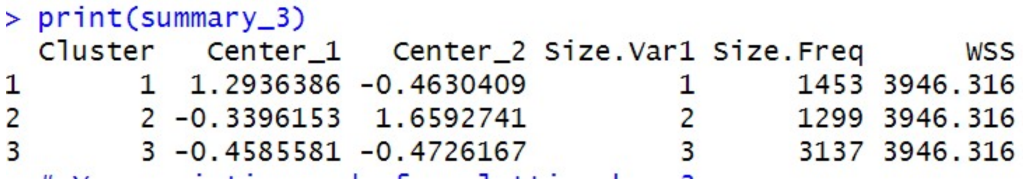
   Cluster2- 1299

   Cluster3- 3137

3. Diagram:



4. Interpretation of the results:



```
> print(summary_3)
  Cluster    Center_1   Center_2 Size.Var1 Size.Freq      WSS
1       1  1.2936386 -0.4630409         1      1453 3946.316
2       2 -0.3396153  1.6592741         2      1299 3946.316
3       3 -0.4585581 -0.4726167         3      3137 3946.316
```

Cluster Centers:

Cluster 1 Center: Approximately (1.29, -0.46)

Cluster 2 Center: Approximately (-0.34, 1.66)

Cluster 3 Center: Approximately (-0.46, -0.47)

The cluster centers represent prototypical points in the feature space. They are the means of the feature values for the data points within each cluster. These coordinates give a sense of the central tendency of each cluster with respect to the scaled age and number of drinks consumed.

3. Cluster Sizes:

The size of each cluster indicates the number of data points it contains. In this context, Cluster 3 is the largest, followed by Cluster 1 and then Cluster 2. The sizes provide information about the prevalence or abundance of certain patterns within the dataset.

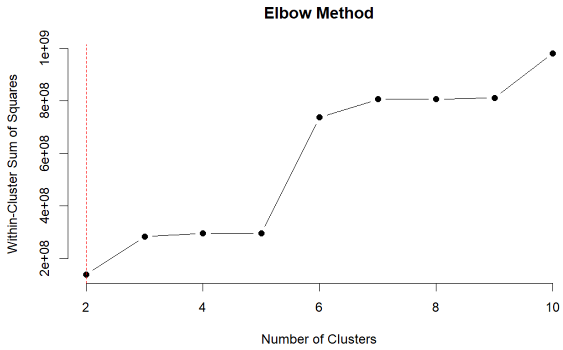
4. Within-Cluster Sum of Squares (WSS):

WSS for Each Cluster: 3946.316

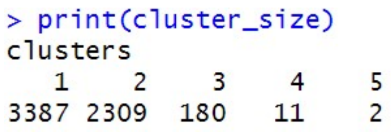
The within-cluster sum of squares (WSS) is a measure of how tightly the data points within each cluster are grouped around their respective centers. It is a metric that quantifies the compactness of clusters. In this case, the WSS is the same for all three clusters, indicating that each cluster exhibits a similar level of internal cohesion or tightness around its center.

Hierarchical Clustering

1. The best number of clusters: 5

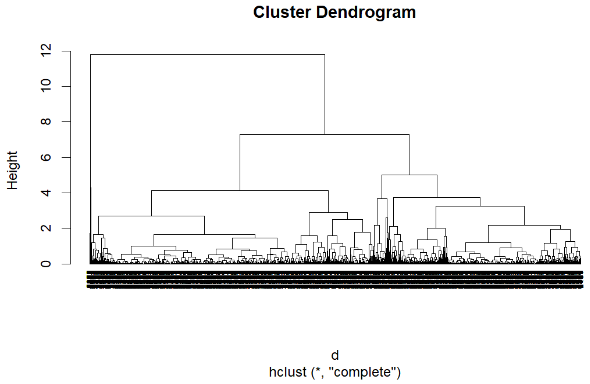


2. Number of elements per cluster:



```
> print(cluster_size)
clusters
   1    2    3    4    5 
3387 2309  180   11    2 
```

3. Diagram:



4. Interpretation of the results:

Cluster Size: Indicates the number of observations (data points) in each cluster.

Cluster 1 has the largest size (3387).

Cluster 2 is the second-largest (2309).

Cluster 3 is moderately sized (180).

Clusters 4 and 5 are smaller, with 11 and 2 observations, respectively.

Cluster Height: Represents the height at which clusters are merged in the hierarchical clustering dendrogram. Lower heights typically indicate that the clusters are more similar or were merged earlier in the hierarchical clustering process.

Cluster 2 has the lowest height (0.04449921), suggesting it was merged early.

Cluster 5 has the second-lowest height (0.09196503).

Clusters 1, 3, and 4 have higher heights, indicating later merging.

3. Overall Observations:

Hierarchy of Merging: The clusters form a hierarchy in terms of merging, with Cluster 2 being the earliest to merge.

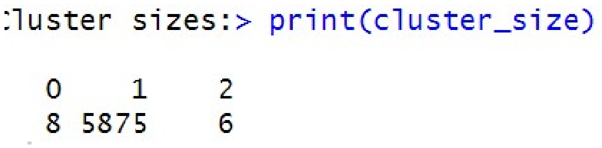
Cluster Size Variation: There is significant variation in the sizes of the clusters, with Cluster 1 being the largest and Clusters 4 and 5 being the smallest.

Interpretation Complexity: The interpretation of hierarchical clustering can be more complex due to the hierarchical nature of the analysis.

Density based clustering

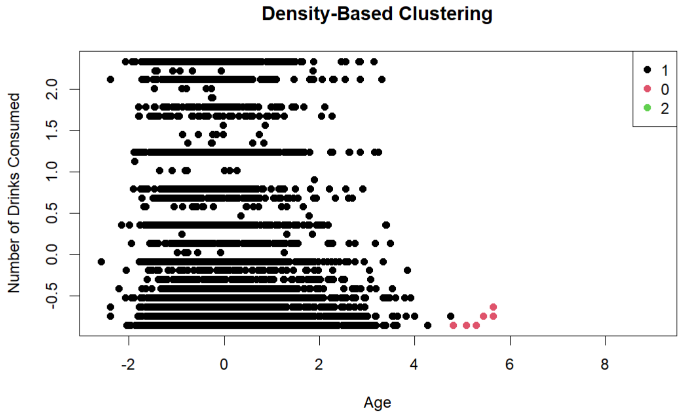
Density based clustering such as dbscan does not require specifying the number of clusters beforehand like k means does. We have performed some analysis and visualization to understand the clustering structure and decide on an appropriate set of parameters(eg., ‘eps’ and ‘minPts’).

1. Number of clusters: 3
2. Cluster Size:



```
:luster sizes:> print(cluster_size)

   0    1    2
   8 5875    6
```

3. Diagram



4. Interpretation of th results:

Noise Points (Cluster 0):

These are data points that do not belong to any of the identified clusters. In DBSCAN, points that don't meet the density criteria to form a cluster are considered as noise. In this case, there are 8 such points.

Cluster 1:

This is the largest cluster with 5875 points. These points are part of dense regions in the feature space according to the density criteria defined by the eps and minPts parameters.

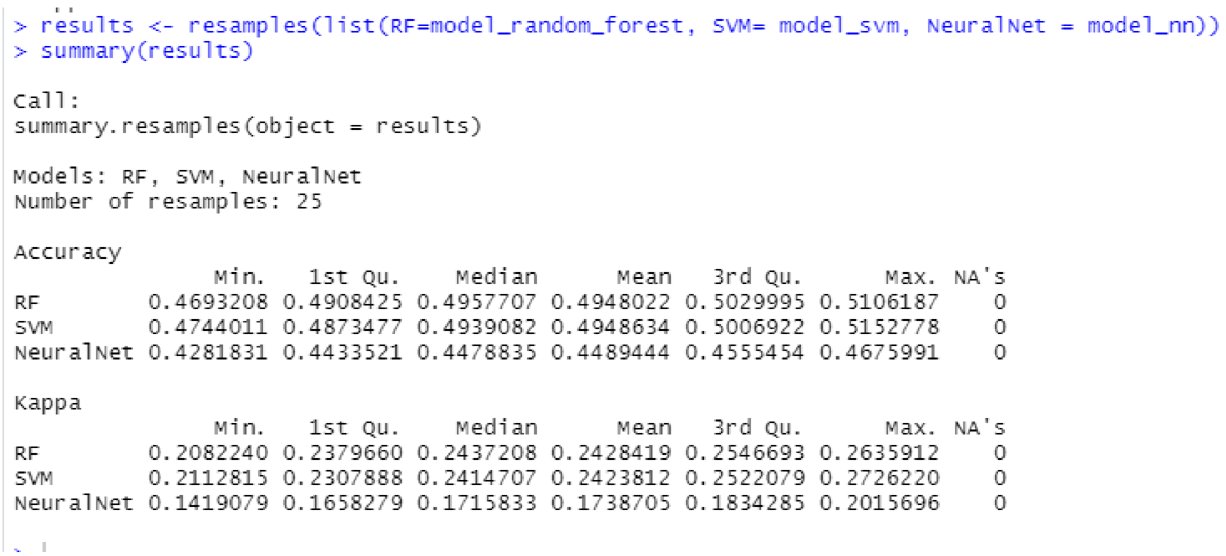
Cluster 2:

This cluster contains 6 points. It's a smaller cluster in terms of size.

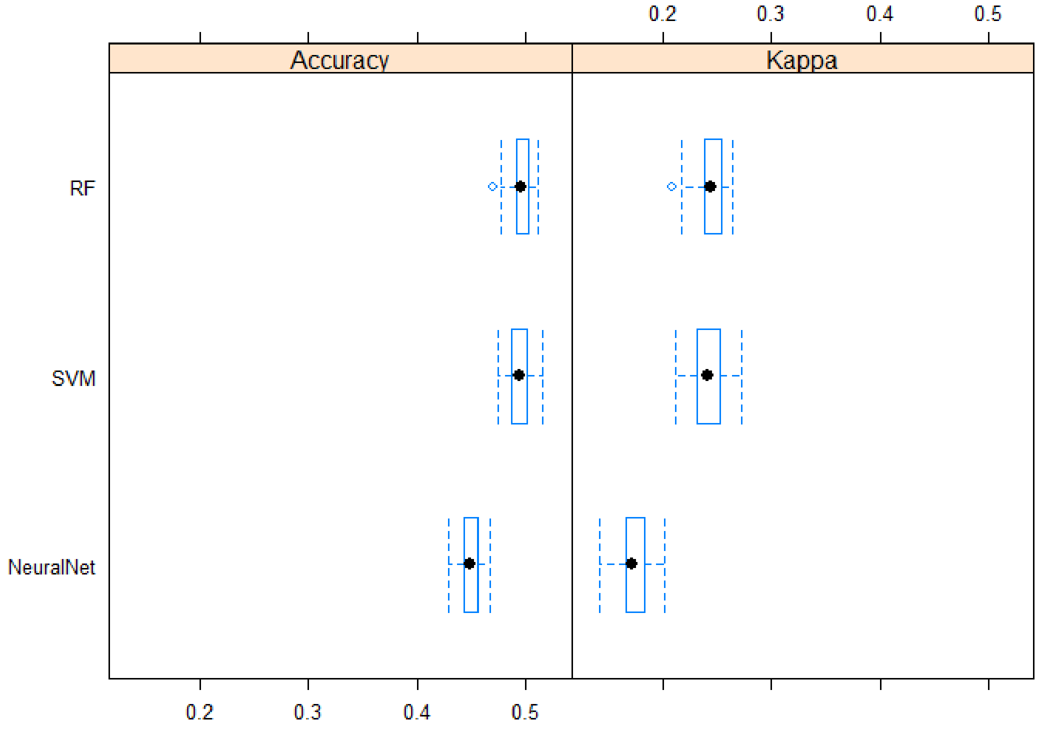
Question 4. Comparative Analysis

The summary of the comparative analysis results shows performance metrics (Accuracy and Kappa) for three different models Random Forest, Support Vector Machine (Radial), and Neural Network

- Using k- fold cross validation with k=10 and number of repeats=3



```
> results <- resamples(list(RF=model_random_forest, SVM= model_svm, NeuralNet = model_nn))
> summary(results)

Call:
summary.resamples(object = results)

Models: RF, SVM, NeuralNet
Number of resamples: 25

Accuracy
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
RF        0.4693208 0.4908425 0.4957707 0.4948022 0.5029995 0.5106187    0
SVM       0.4744011 0.4873477 0.4939082 0.4948634 0.5006922 0.5152778    0
NeuralNet 0.4281831 0.4433521 0.4478835 0.4489444 0.4555454 0.4675991    0

Kappa
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
RF        0.2082240 0.2379660 0.2437208 0.2428419 0.2546693 0.2635912    0
SVM       0.2112815 0.2307888 0.2414707 0.2423812 0.2522079 0.2726220    0
NeuralNet 0.1419079 0.1658279 0.1715833 0.1738705 0.1834285 0.2015696    0
```



For the Random Forest (RF) model, accuracy ranges from approximately 0.469 to 0.51, with a mean around 0.495.

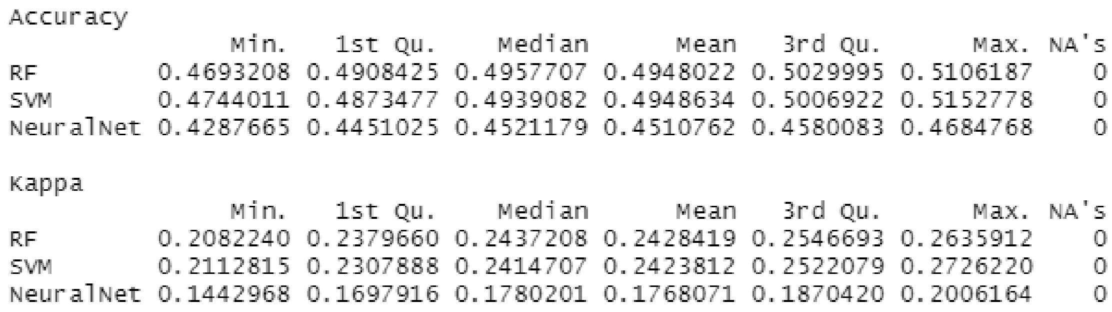
For the Support Vector Machine (SVM) model, accuracy ranges from approximately 0.474 to 0.515, with a mean around 0.495.

For the Neural Network (NeuralNet) model, accuracy ranges from approximately 0.428 to 0.467, with a mean around 0.448

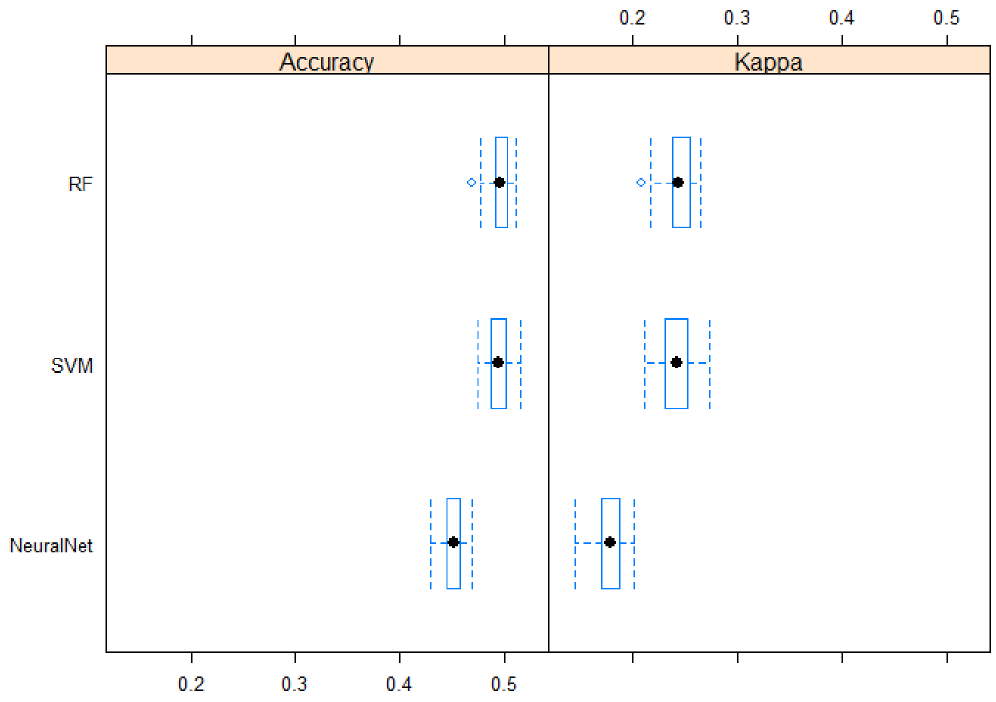
We can see that both Random Forest and SVM have similar accuracy scores with neural networks having a lower score.This difference might be attributed to a potential factor during the neural network modeling process. Specifically, the neural network model may have been constructed without considering all features, particularly excluding certain categorical features. This omission might have influenced the model's performance, contributing to the observed differences in accuracy

Kappa scores account for the possibility of predictions occurring by chance. It adjusts accuracy for imbalances in the data. From the boxplot we can observe that the plots of the kappa scores and the accuracy scores are very similar, indicating that our data is well balanced.

-Using random subsampling cross validation with 50 subsamples drawn each time

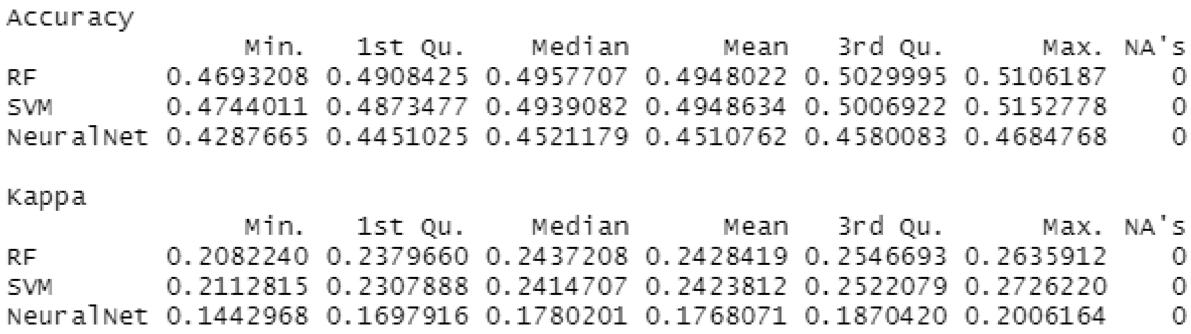


```
Accuracy 
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
RF        0.4693208 0.4908425 0.4957707 0.4948022 0.5029995 0.5106187    0
SVM       0.4744011 0.4873477 0.4939082 0.4948634 0.5006922 0.5152778    0
NeuralNet 0.4287665 0.4451025 0.4521179 0.4510762 0.4580083 0.4684768    0

Kappa 
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
RF        0.2082240 0.2379660 0.2437208 0.2428419 0.2546693 0.2635912    0
SVM       0.2112815 0.2307888 0.2414707 0.2423812 0.2522079 0.2726220    0
NeuralNet 0.1442968 0.1697916 0.1780201 0.1768071 0.1870420 0.2006164    0
```

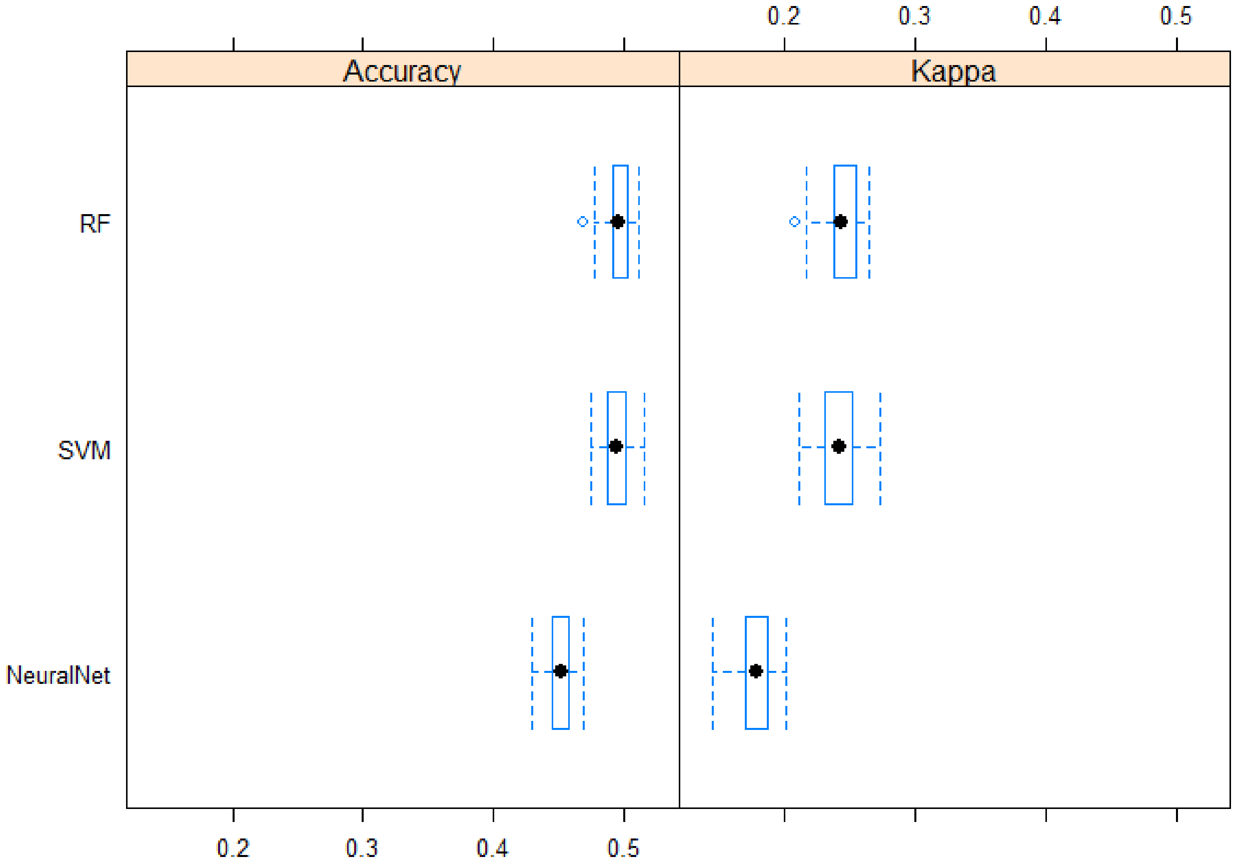


The accuracy values for Random forest and SVM are similar to that of k-fold cross validation, however for neural networks we can notice a slight increase in the mean accuracy scores (from 0.448 to 0.451).

-Using data split cross validation (80% training, 20% test data)



```
Accuracy 
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
RF        0.4693208 0.4908425 0.4957707 0.4948022 0.5029995 0.5106187    0
SVM       0.4744011 0.4873477 0.4939082 0.4948634 0.5006922 0.5152778    0
NeuralNet 0.4287665 0.4451025 0.4521179 0.4510762 0.4580083 0.4684768    0

Kappa 
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
RF        0.2082240 0.2379660 0.2437208 0.2428419 0.2546693 0.2635912    0
SVM       0.2112815 0.2307888 0.2414707 0.2423812 0.2522079 0.2726220    0
NeuralNet 0.1442968 0.1697916 0.1780201 0.1768071 0.1870420 0.2006164    0
```



The accuracy and Kappa values for all 3 models seems to be similar to that of the data sampling cross validation technique.

Question 5. Feature Selection [5 pts]

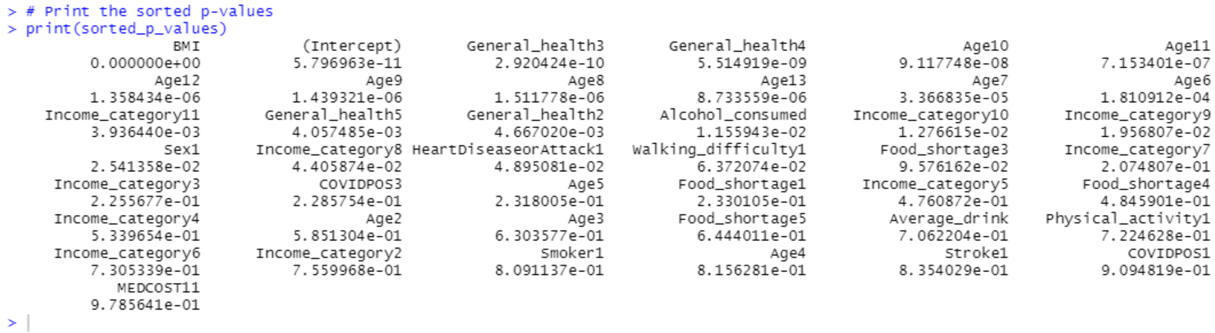
Select three models from Milestones 2 or 3 (linear/logistic regression, Naïve Bayes, Decision Trees, SVM or Neural Networks) and apply three different types of feature selection: filter, wrapper or embedding, one per model. Report results focusing on improvements in performance (or lack of) due to feature selection.

1) Feature selection on the Multinomial Logistic regression model using the selection method

   The initial model was trained using all of our 15 available features, but due to the predominance of multicategorical variables, the feature set expanded to include a substantial number of variables—42 in total.

   We obtained an accuracy of 52.12% with this original model.

   To conduct feature selection, we initially calculated the p-values for all the variables using Wald's test. Subsequently, we arranged these p-values in ascending order and selected the top 5 features with the most significance. Following were the results-



```
> # Print the sorted p-values
> print(sorted_p_values)
                  BMI           (Intercept)       General_health3       General_health4                 Age10                 Age11 
         0.000000e+00          5.796963e-11          2.920424e-10          5.514919e-09          9.117748e-08          7.153401e-07 
                Age12                  Age9                  Age8                 Age13                  Age7                  Age6 
         1.358434e-06          1.439321e-06          1.511778e-06          8.733559e-06          3.366835e-05          1.810912e-04 
    Income_category11       General_health5       General_health2      Alcohol_consumed     Income_category10      Income_category9 
         3.936440e-03          4.057485e-03          4.667020e-03          1.155943e-02          1.276615e-02          1.956807e-02 
                 Sex1      Income_category8 HeartDiseaseorAttack1   Walking_difficulty1        Food_shortage3      Income_category7 
         2.541358e-02          4.405874e-02          4.895081e-02          6.372074e-02          9.576162e-02          2.074807e-01 
     Income_category3             COVIDPOS3                  Age5        Food_shortage1      Income_category5        Food_shortage4 
         2.255677e-01          2.285754e-01          2.318005e-01          2.330105e-01          4.760872e-01          4.845901e-01 
     Income_category4                  Age2                  Age3        Food_shortage5         Average_drink    Physical_activity1 
         5.339654e-01          5.851304e-01          6.303577e-01          6.444011e-01          7.062204e-01          7.224628e-01 
     Income_category6      Income_category2               Smoker1                  Age4               Stroke1             COVIDPOS1 
         7.305339e-01          7.559968e-01          8.091137e-01          8.156281e-01          8.354029e-01          9.094819e-01 
            MEDCOST11 
         9.785641e-01 
> 
```

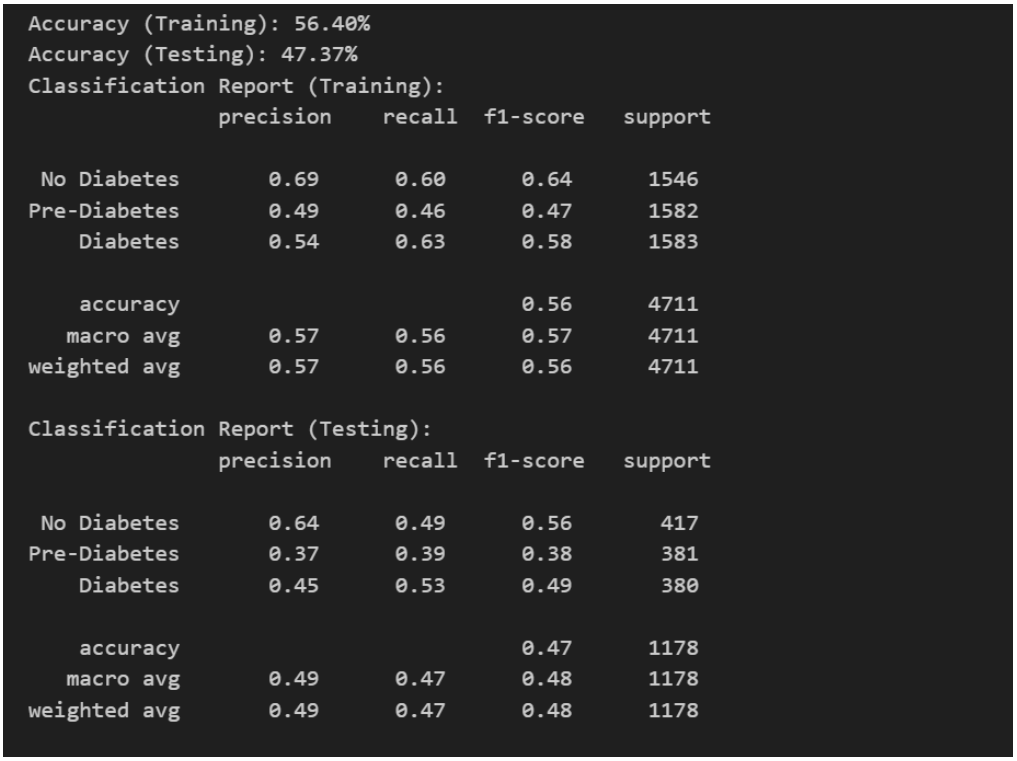
   The top 5 features were- BMI, Age, General_health, Income_category and Alcohol_consumed

   We created a logistic regression model considering these five variables and we obtained an accuracy of 51.18%. This is only marginally less than our original accuracy of 52% indicating that we can limit the number of predictors to 5,thereby simplifying the model and enhancing interpretability without significantly compromising accuracy.

2) Implementing feature selection on the Decision tree model using a tree-based embedding method

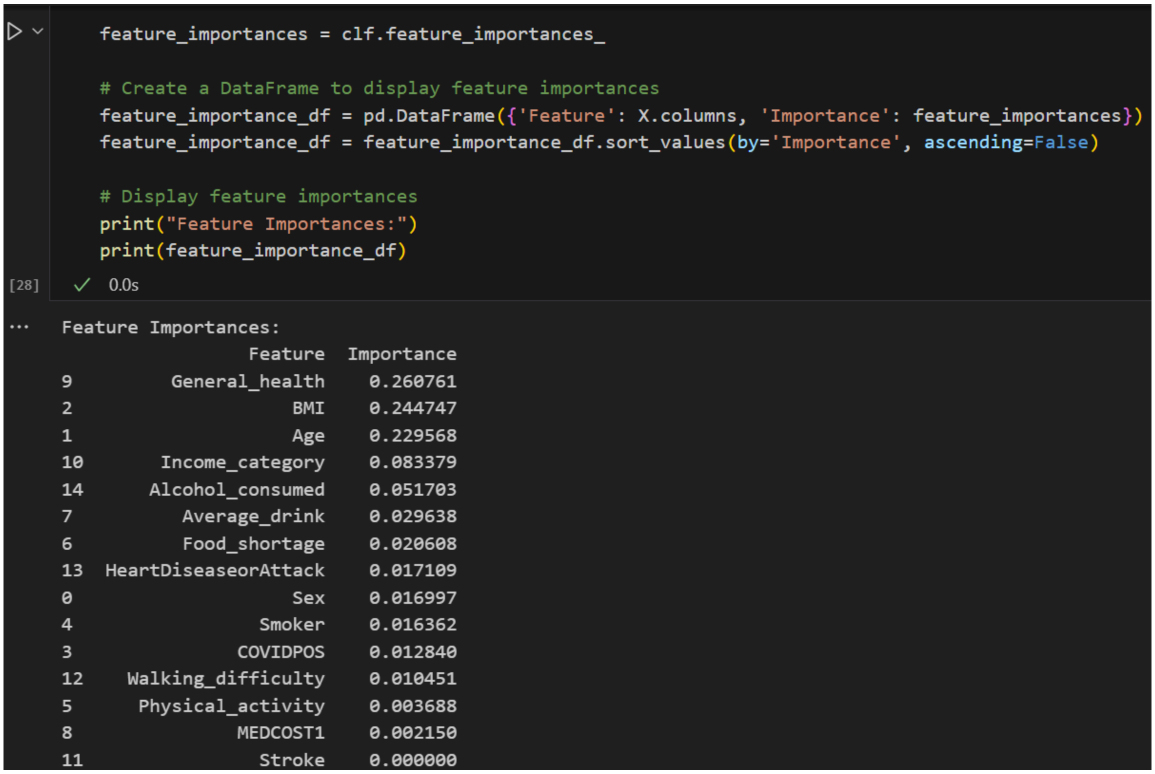
   This was implemented in Python since our decision tree was also implemented in

Python. With our original implementation of the decision tree we had obtained the following results:



```
Accuracy (Training): 56.40%
Accuracy (Testing): 47.37%
Classification Report (Training):
               precision    recall  f1-score   support

 No Diabetes       0.69      0.60      0.64      1546
Pre-Diabetes       0.49      0.46      0.47      1582
    Diabetes       0.54      0.63      0.58      1583

    accuracy                           0.56      4711
   macro avg       0.57      0.56      0.57      4711
weighted avg       0.57      0.56      0.56      4711

Classification Report (Testing):
               precision    recall  f1-score   support

 No Diabetes       0.64      0.49      0.56       417
Pre-Diabetes       0.37      0.39      0.38       381
    Diabetes       0.45      0.53      0.49       380

    accuracy                           0.47      1178
   macro avg       0.49      0.47      0.48      1178
weighted avg       0.49      0.47      0.48      1178
```

In this method the feature selection is done during the training of the model itself. We derive the feature importance values from the trained model and order them as shown below-

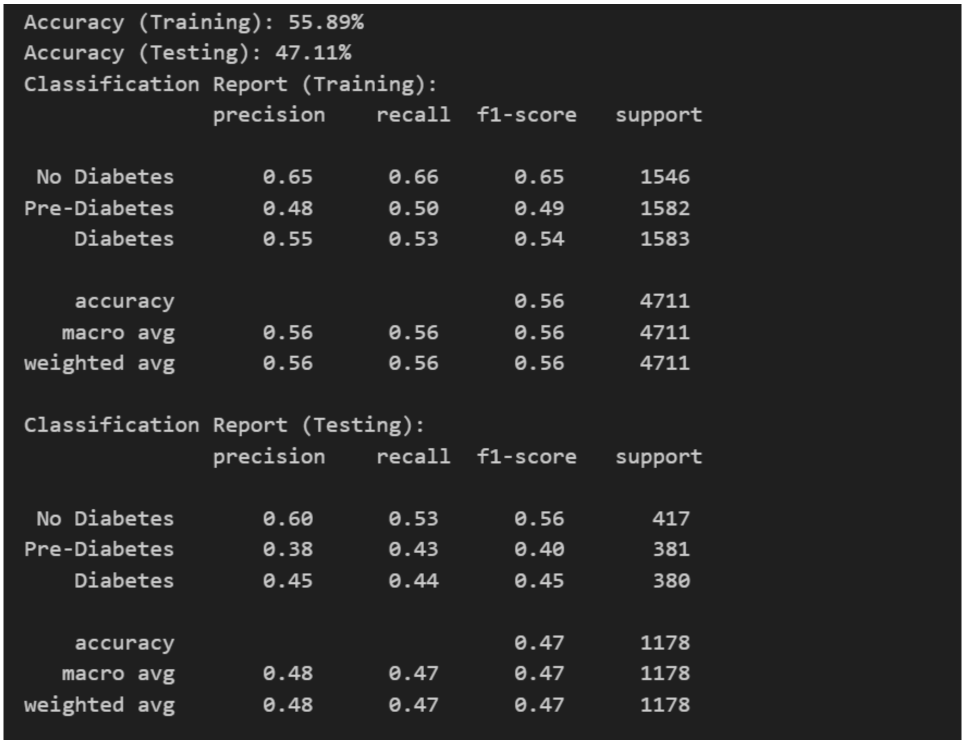


```python
feature_importances = clf.feature_importances_

# Create a DataFrame to display feature importances
feature_importance_df = pd.DataFrame({'Feature': X.columns, 'Importance': feature_importances})
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=False)

# Display feature importances
print("Feature Importances:")
print(feature_importance_df)
```

```
Feature Importances:
                 Feature  Importance
9         General_health    0.260761
2                    BMI    0.244747
1                    Age    0.229568
10       Income_category    0.083379
14      Alcohol_consumed    0.051703
7          Average_drink    0.029638
6          Food_shortage    0.020608
13  HeartDiseaseorAttack    0.017109
0                    Sex    0.016997
4                 Smoker    0.016362
3               COVIDPOS    0.012840
12    Walking_difficulty    0.010451
5      Physical_activity    0.003688
8              MEDCOST1     0.002150
11                Stroke    0.000000
```

From this we select the top 5 features which are - General_health,BMI, Age,Income_category and Alcohol_consumed

Running the model with these as our predictor variables we obtain the following results:



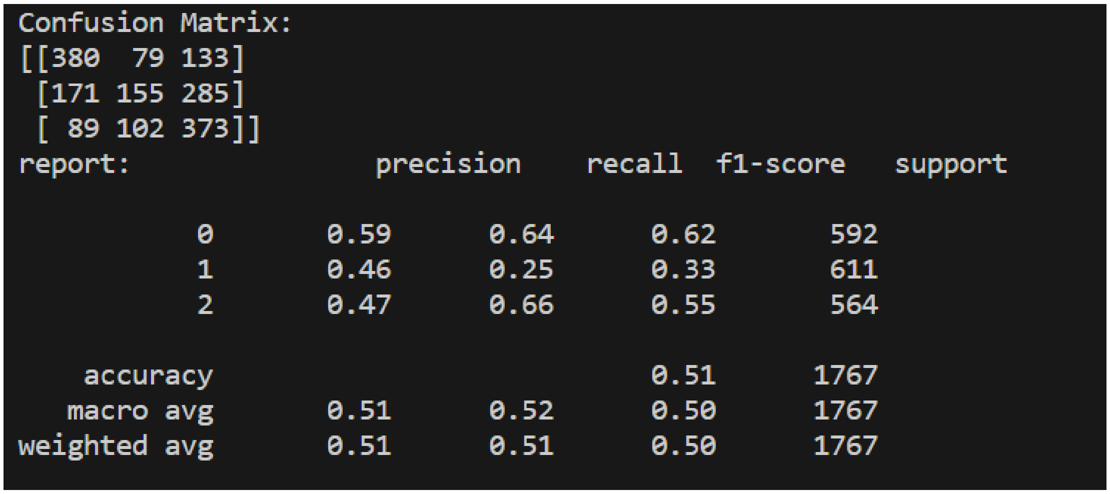
```
Accuracy (Training): 55.89%
Accuracy (Testing): 47.11%
Classification Report (Training):
               precision    recall  f1-score   support

 No Diabetes       0.65      0.66      0.65      1546
Pre-Diabetes       0.48      0.50      0.49      1582
    Diabetes       0.55      0.53      0.54      1583

    accuracy                           0.56      4711
   macro avg       0.56      0.56      0.56      4711
weighted avg       0.56      0.56      0.56      4711

Classification Report (Testing):
               precision    recall  f1-score   support

 No Diabetes       0.60      0.53      0.56       417
Pre-Diabetes       0.38      0.43      0.40       381
    Diabetes       0.45      0.44      0.45       380

    accuracy                           0.47      1178
   macro avg       0.48      0.47      0.47      1178
weighted avg       0.48      0.47      0.47      1178
```

We can notice that the testing accuracy dropped by a negligible amount (from 47.37% to 47.11%). This indicates that we can use our new model with five predictor variables and still get similar performance metrics as our original model (which had fifteen variables)

3) Implementing wrapper feature selection in the SVM model using sequential forward feature selection method

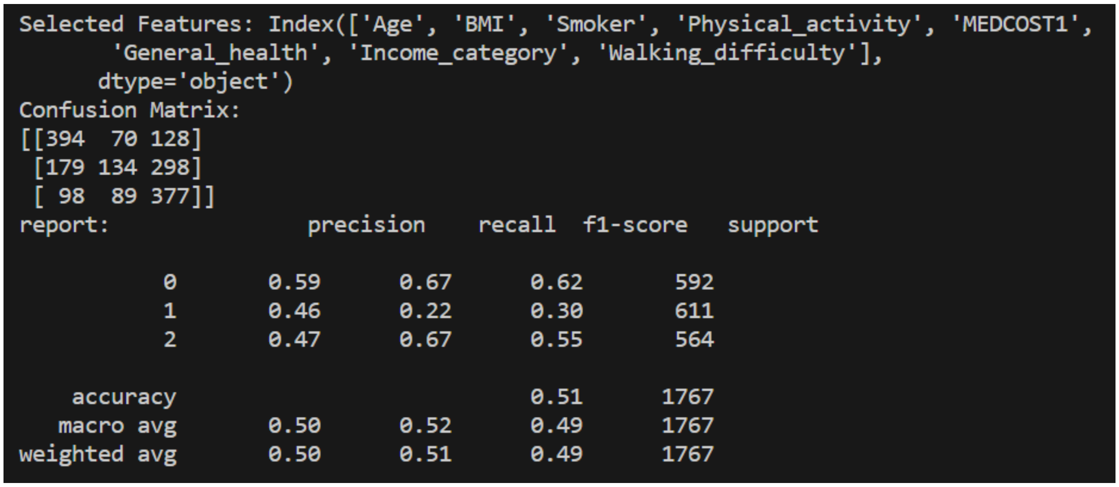
   As discussed in class,the step function used to perform SFS in R is typically used for linear regression models and is not directly applicable to SVM models with categorical dependent variables. Therefore we rewrote the original radial SVM model in Python and implemented SFS using the ‘SequentialFeatureSelector’ class from the mlxtend library.

   Following are the results of the original radial SVM (with all the features). We notice an accuracy of 51%



```
Confusion Matrix:
[[380  79 133]
 [171 155 285]
 [ 89 102 373]]
report:               precision    recall  f1-score   support

           0       0.59      0.64      0.62       592
           1       0.46      0.25      0.33       611
           2       0.47      0.66      0.55       564

    accuracy                           0.51      1767
   macro avg       0.51      0.52      0.50      1767
weighted avg       0.51      0.51      0.50      1767
```

Following are the results and the features chosen after running SFS-



```
Selected Features: Index(['Age', 'BMI', 'Smoker', 'Physical_activity', 'MEDCOST1',
       'General_health', 'Income_category', 'Walking_difficulty'],
      dtype='object')
Confusion Matrix:
[[394  70 128]
 [179 134 298]
 [ 98  89 377]]
report:               precision    recall  f1-score   support

           0       0.59      0.67      0.62       592
           1       0.46      0.22      0.30       611
           2       0.47      0.67      0.55       564

    accuracy                           0.51      1767
   macro avg       0.50      0.52      0.49      1767
weighted avg       0.50      0.51      0.49      1767
```

We can see that not only has the accuracy remained the same (51%) , but the recall scores for Class 0 and Class 2 of the new model have actually increased showing that feature selection has potentially improved the model.

Across all the feature selection techniques employed, it emerged that 'Age,' 'BMI,' 'General_health,' and 'Income_category' stood out as the most consistently predictive features.

Question 6. Extra Credit [10 pts]

Discuss potential ethical issues in the research questions you have proposed for your project. As covered in Class 10 (Ethical Data Science part), delve into concerns around data collection, variable manipulation and model evaluation.

Ethical Data Science:

How was the data collected?

- Centres for Disease Control and Prevention
- The BRFSS conducts over 506,000 interviews, utilizing a structured questionnaire that includes core questions, optional modules, and state-added questions

How were the variables computed?

- 2022 BRFSS Survey Data
- A key feature in recent years has been the inclusion of cellular telephone use in data collection, enhancing representativeness and coverage. The weighting methodology, known as 'raking' or iterative proportional fitting, replaced the previous post-stratification weighting. This method allows for the inclusion of additional demographic variables like education, marital status, property ownership, and telephone ownership. This approach is designed to minimize selection bias and improve representation across diverse demographics

Ethical Data Collection:

Permission Requested and Informed consent:

- While specific details about informed consent are not directly available in the provided resources, the BRFSS's adherence to scientific and ethical standards likely involves obtaining consent, especially considering the personal nature of health-related data.

Privacy:

- The specific details regarding the use of anonymization and pseudonymization techniques in the CDC's Behavioral Risk Factor Surveillance System (BRFSS) data are not readily available from the provided resources. Typically, large-scale health data collections like the BRFSS employ various data privacy and security measures to protect participant confidentiality, but the exact nature of these measures for the BRFSS data isn't specified in the available documentation.

Data that fairly represents the population modelled:

- The BRFSS's methodology, especially the incorporation of new statistical methods like small area estimation and efforts to reach hard-to-reach respondents, indicates a commitment to fairly representing the population being modeled. The CDC continuously works to improve data collection methods to enhance the quality and representativeness of the data, including piloting new data collection modes and expanding survey coverage to address emerging health issues

- As for the fair representation of the data and whether any specific data was omitted, the BRFSS aims to represent the population accurately by using a methodology that includes various demographic variables. However, the omission of specific data types or categories is not explicitly detailed in the public-facing documentation. This kind of information is often managed internally and may not be publicly disclosed due to privacy and ethical considerations.

Ethical Variable Manipulation

Variables that embed ideology:

- AGE- Indicates the reported age in 5 year categories

- Smoker- Indicates whether or not the respondent had smoked at least a 100 cigarettes in their lifetime. There were a lot of features regarding smoking. We felt that this one captured the smoking frequency and intensity appropriately. Similar to 'SEXVAR'

The value 'No' was modified from 2 to 0.

- Food shortage (How often did the food that you bought not last, and you didn't have money to get more?)

1	Always
2	Usually
3	Sometimes
4	Rarely
5	Never
7	Don't know/Not Sure
9	Refused

- Income category:

Value	Value Label
1	Less than $15,000 Notes: INCOME3=1,2
2	$15,000 to < $25,000 Notes: INCOME3=3,4
3	$25,000 to < $35,000 Notes: INCOME3=5
4	$35,000 to < $50,000 Notes: INCOME3=6
5	$50,000 to < $100,000 Notes: INCOME3=7,8
6	$100,000 to < $200,000 Notes: INCOME3=9,10
7	$200,000 or more Notes: INCOME3=11
9	Don’t know/Not sure/Missing Notes: INCOME3=77, 99, or missing

- General Health

Value	Value Label
1	Excellent
2	Very good
3	Good
4	Fair
5	Poor

7	Don’t know/Not Sure
9	Refused

Fairness in models you create

Implications of the models you create:

Dataset Representativeness: The dataset you used is the Behavioral Risk Factor Surveillance System (BRFSS) survey, which is a telephone-based survey of adults in the U.S. This could mean that certain demographic groups, especially those without regular telephone access or who are less likely to participate in such surveys, may be underrepresented. This can limit the generalizability of your model to these groups.

Cultural and Lifestyle Differences: The dataset primarily reflects the lifestyle, dietary habits, and health behaviours of the U.S. population. Therefore, the model might not accurately predict diabetes risk for individuals from different cultural or geographical backgrounds where lifestyle habits significantly differ.

Lack of Hereditary and Family History Data: An important limitation of your model could be the absence of data on hereditary factors and family history of diabetes. Genetics play a crucial role in the risk of developing diabetes, especially Type 1 diabetes. Without incorporating family medical history and genetic predispositions, the model might miss a key predictive factor, potentially affecting its accuracy in identifying individuals at high risk for diabetes. This aspect is particularly important as it could significantly change risk assessments, especially for individuals with a strong family history of diabetes, regardless of their personal lifestyle and behavioral factors.

Age and Racial Diversity: The BRFSS dataset categorizes age in broad ranges and may not fully capture the nuances of age-specific risk factors for diabetes. Additionally, if the dataset does not sufficiently represent all racial and ethnic groups, the model may not accurately predict diabetes risk across different races and ethnicities.

Diabetes Classification: The diabetes variable in your dataset is categorized into non-diabetic, prediabetic, and diabetic. This classification might oversimplify the complex nature of diabetes progression and risk

CONTRIBUTION

Code, Report, Presentation	Dhiraj	Sakshi	Shashank
Q1			100%
Q2	100%		
Q3		100%	
Q4	33.33%	33.33%	33.33%
Q5	33.33%	33.33%	33.33%
Q6	33.33%	33.33%	33.33%