Analyzing and Predicting Youth Substance Usage: Insights from Decision Tree Analysis

ABSTRACT

This report delves into the dynamics of youth alcohol use through rigorous analysis using decision tree models on data from the National Survey on Drug Use and Health. It addresses three key objectives: binary classification for alcohol use, multi-class classification for alcohol frequency in the past year, and regression to predict the age of first alcohol use. By examining demographic variables and youth experiences, the study unveils crucial insights into the factors influencing alcohol consumption among young individuals. These findings are instrumental in crafting targeted interventions and public health strategies to address and mitigate the challenges posed by alcohol use among the youth population.

INTRODUCTION

This research embarks on a comprehensive exploration of youth alcohol use, utilizing decision tree models and analytical methods on data sourced from the National Survey on Drug Use and Health (NSDUH). The primary goals of this report are to dissect the intricate dynamics of youth alcohol consumption, categorize it through binary and multi-class classification, and predict the age of first alcohol use. Through this analysis, it aims to shed light on the nuanced factors influencing alcohol use among young individuals, ranging from demographic characteristics to their experiences and interactions.

The NSDUH dataset serves as the cornerstone of this investigation, offering a robust collection of variables including demographics such as age, gender, race, and socioeconomic status, alongside detailed insights into youth experiences encompassing parental influence, peer interactions, and educational background. This rich dataset enables us to delve deep into the patterns and predictors of alcohol use among youth, providing a holistic understanding of this critical public health issue.

By setting clear objectives and employing advanced analytical techniques, this report endeavors to uncover actionable insights that can inform targeted interventions and policy initiatives aimed at promoting responsible alcohol behavior and mitigating the risks associated with excessive and early alcohol use among young individuals. Through methodical analysis and interpretation, it seek to contribute meaningfully to the discourse surrounding youth alcohol use and its societal implications.

THEORETICAL BACKGROUND

Decision Trees

Decision trees serve as versatile models capable of handling both classification and regression tasks. They operate by recursively partitioning the data into subsets based on feature conditions that optimize information gain or minimize impurity measures such as the Gini index or entropy. Noteworthy attributes of decision trees include their interpretability, ability to capture non-linear relationships and interactions between features, and key tuning parameters like maximum depth and minimum samples per leaf.

Bagging

Bagging techniques significantly enhance model stability and reduce variance by training multiple models on bootstrapped samples and then aggregating predictions through averaging or voting mechanisms. This approach is particularly effective for high-variance models like decision trees, as it mitigates overfitting and improves generalization by combining predictions from diverse models trained on different subsets of the data.

Random Forest

Random forests, an extension of bagging, introduce randomness in feature selection at each split, leading to improved generalization and reduced overfitting. This randomness enhances model diversity, especially beneficial for handling larger datasets and capturing complex relationships within the data. Key parameters such as the number of trees, the number of features considered at each split mtry, and node size play crucial roles in optimizing random forest models.

Its employees the theoretical underpinnings and tuning parameters of decision trees, bagging, and random forests is paramount for developing accurate predictive models and gaining insights into factors influencing alcohol-related behaviors among young individuals. These machine learning techniques offer valuable tools for researchers and practitioners in addressing critical issues related to youth alcohol use through data-driven approaches.

METHODOLOGY

Data cleaning the systematic methodology to analyze youth alcohol use data using machine learning techniques. The initial step involves data preparation, where it loaded the dataset, ensured its cleanliness, and handled missing values using the `na.omit()` function. Subsequently, it divided the cleaned dataset into subsets focusing on demographic information, youth experiences, and alcohol-related variables for further analysis.

For the binary classification analysis, aimed at predicting alcohol ever used 0=never, 1=ever as targeted variable, then selected relevant predictor variables from the dataset and split the data into training and testing sets. Using the decision tree algorithm `tree()` function, it constructed a decision tree model to visualize and evaluate its performance on the testing data. Additionally, it applied bagging techniques with the randomForest package to improve model stability and reduce variance. Random forests were also implemented, tuning parameters such as the number of trees `ntree` and the number of features considered at each split `mtry`.

Multi-class classification tasks related to alcohol use frequency were conducted by converting the target variable into a factor and employing decision trees, bagging, and random forests for classification. Evaluation metrics such as accuracy, MSE, and confusion matrices were utilized to assess model performance, while feature importance analysis provided insights into influential variables.

In parallel, regression analysis focused on predicting the age of first alcohol use using similar predictor variables. Decision trees, bagging, and random forests were again applied, with performance evaluation based on mean squared error (MSE). Throughout the methodology, model tuning and optimization were performed, with parameters adjusted to optimize model performance. Cross-validation techniques ensured robustness and generalizability of the models, contributing to a comprehensive analysis of youth alcohol use behaviors.

COMPUTATIONAL RESULTS

BINARY CLASSIFICATION

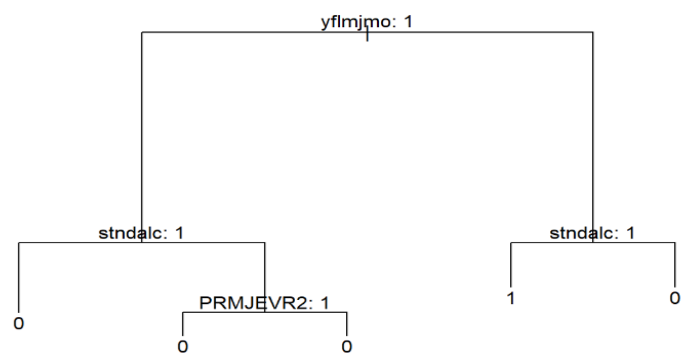
Accuracy values of different models used for binary classification are:

- Decision Tree- 0.801
- Pruned- 0.801
- Bagging- 0.815
- RandomForest-0.819

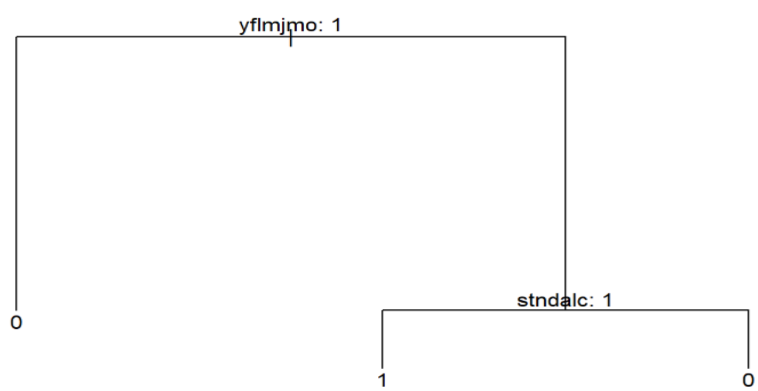
According to the accuracy of the binary classification models, the best model to predict if alcohol is ever used or not is Random Forest with an accuracy of 0.819 i.e. 81%.

Decision Trees:

Decision Tree of binary classification of alcohol use.

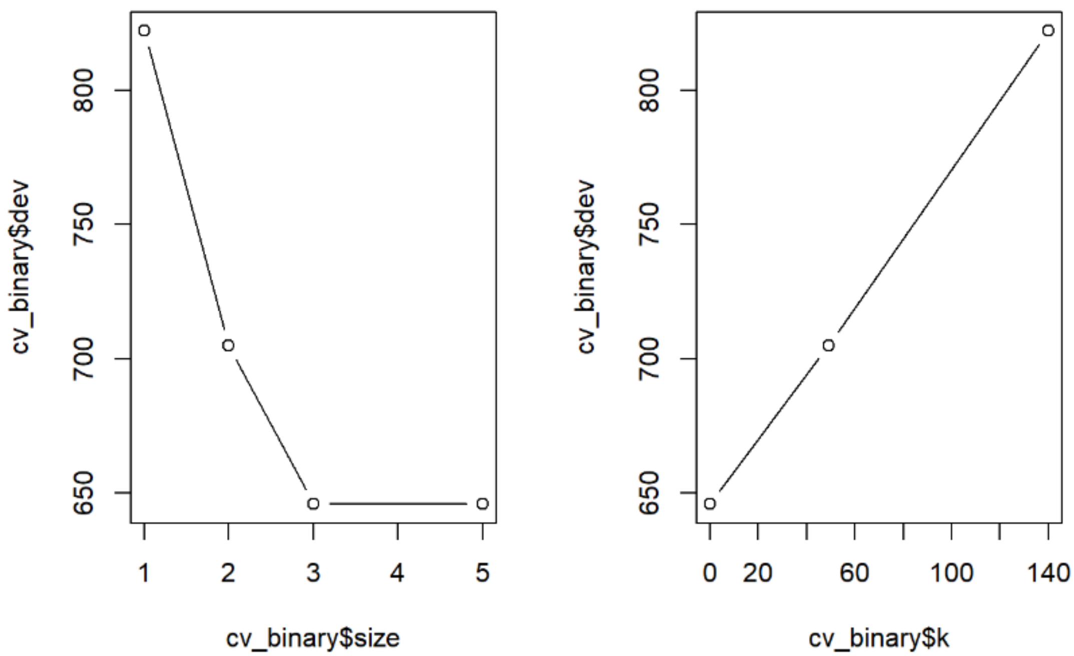


Decision Tree of binary classification of alcohol use.

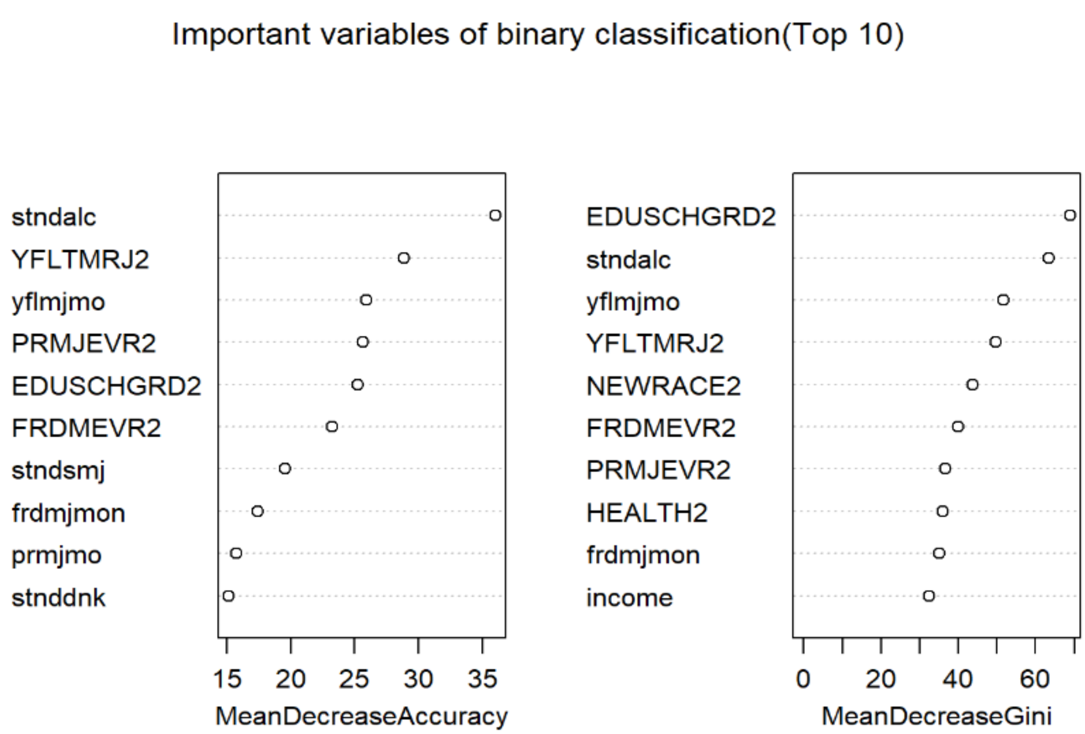


Pruned Decision Tree of binary classification of alcohol use.

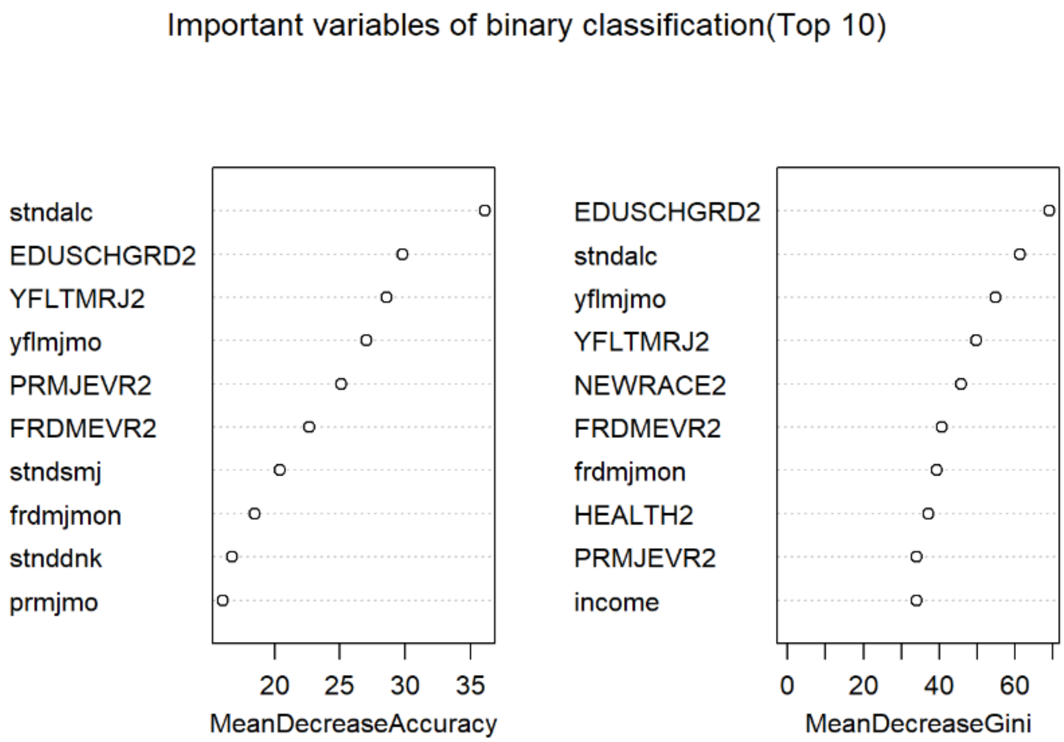
Cross validation results of binary classification:



Important variables of binary classification -Bagging



Important variables of binary classification -RandomForest



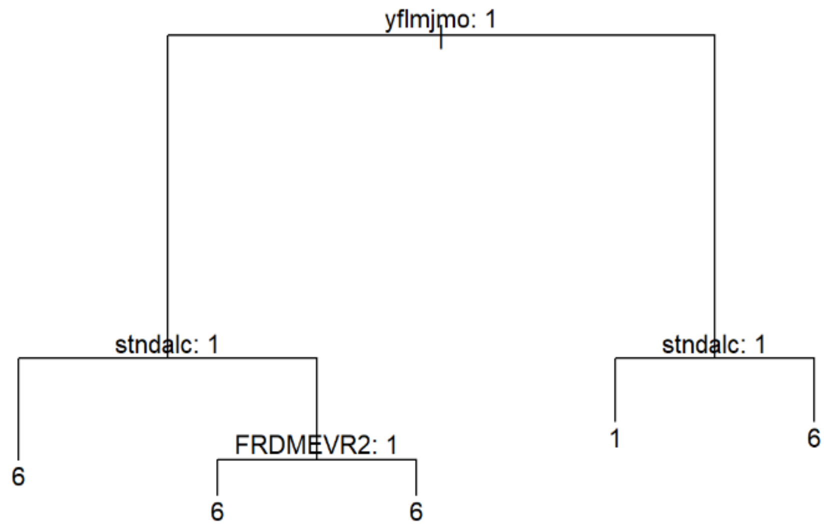
MULTI-CLASS CLASSIFICATION

Accuracy values of different models used for multi-class classification are:

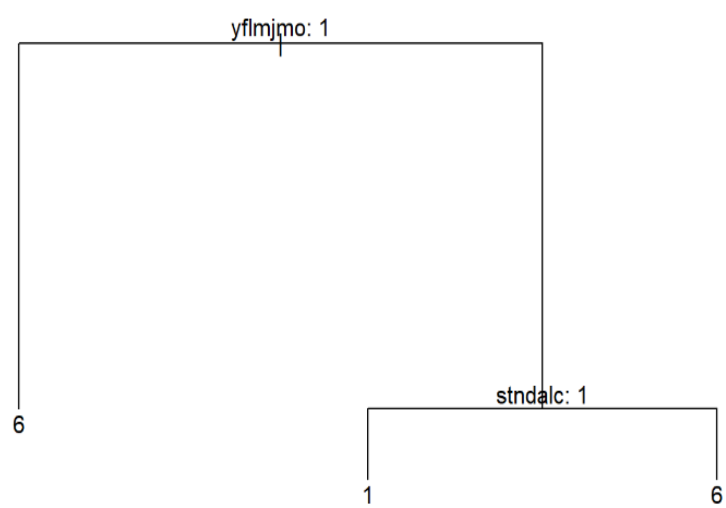
- Decision Tree- 0.797
- Pruned- 0.797
- Bagging- 0.787
- RandomForest-0.787

According to the accuracy of multi-class classification model, the best model to predict number of days alcohol was used in past year are Decision tree & pruned decision tree as both are giving same accuracy of 0.797 i.e. 79%.

Decision Trees:

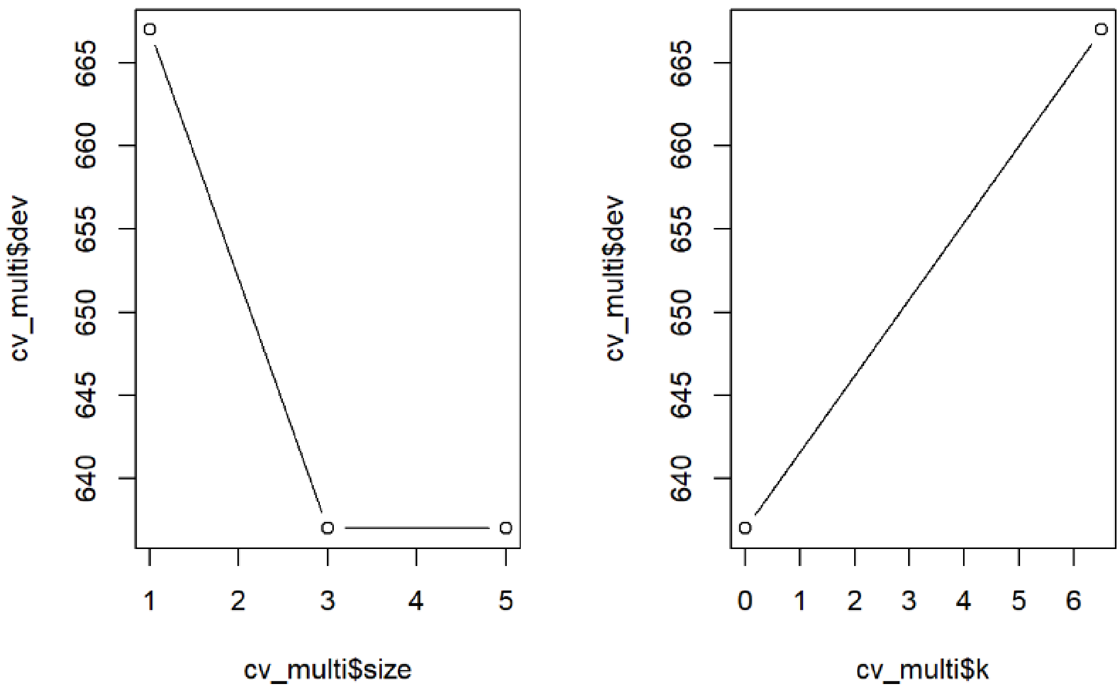


Decision Tree of multi-class classification of number of days alcohol was used in past year.

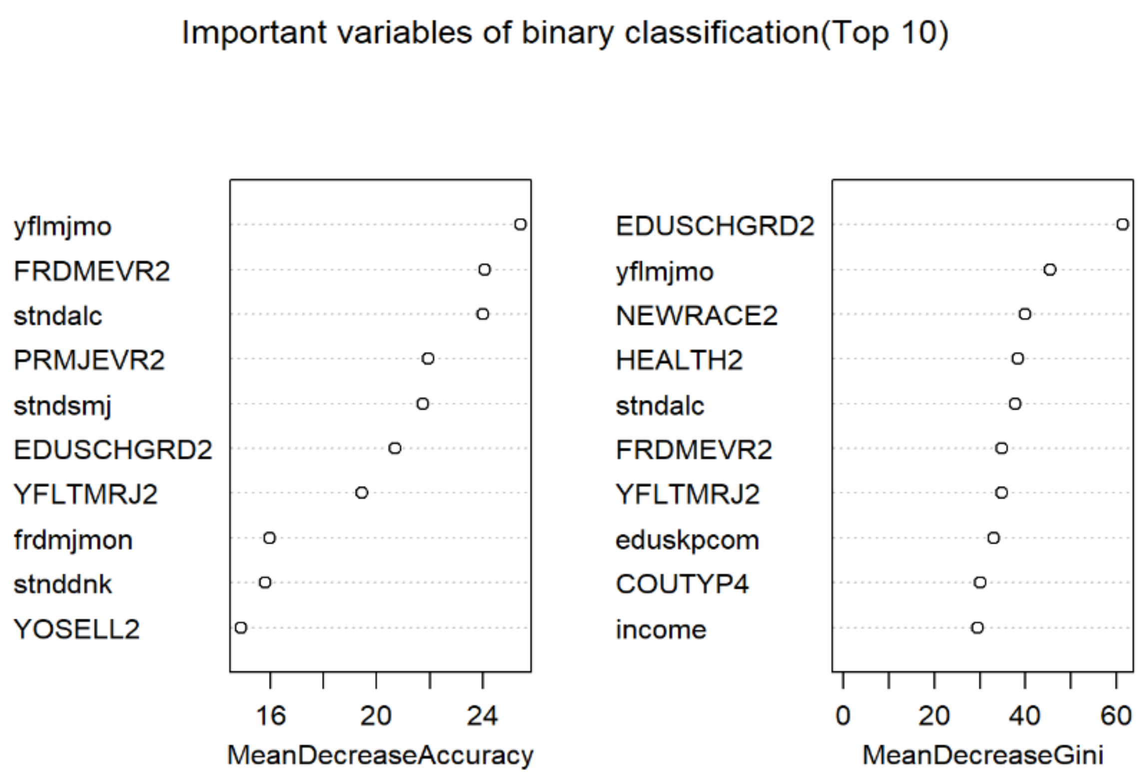


Pruned Decision Tree of multi-class classification of number of days alcohol was used in past year.

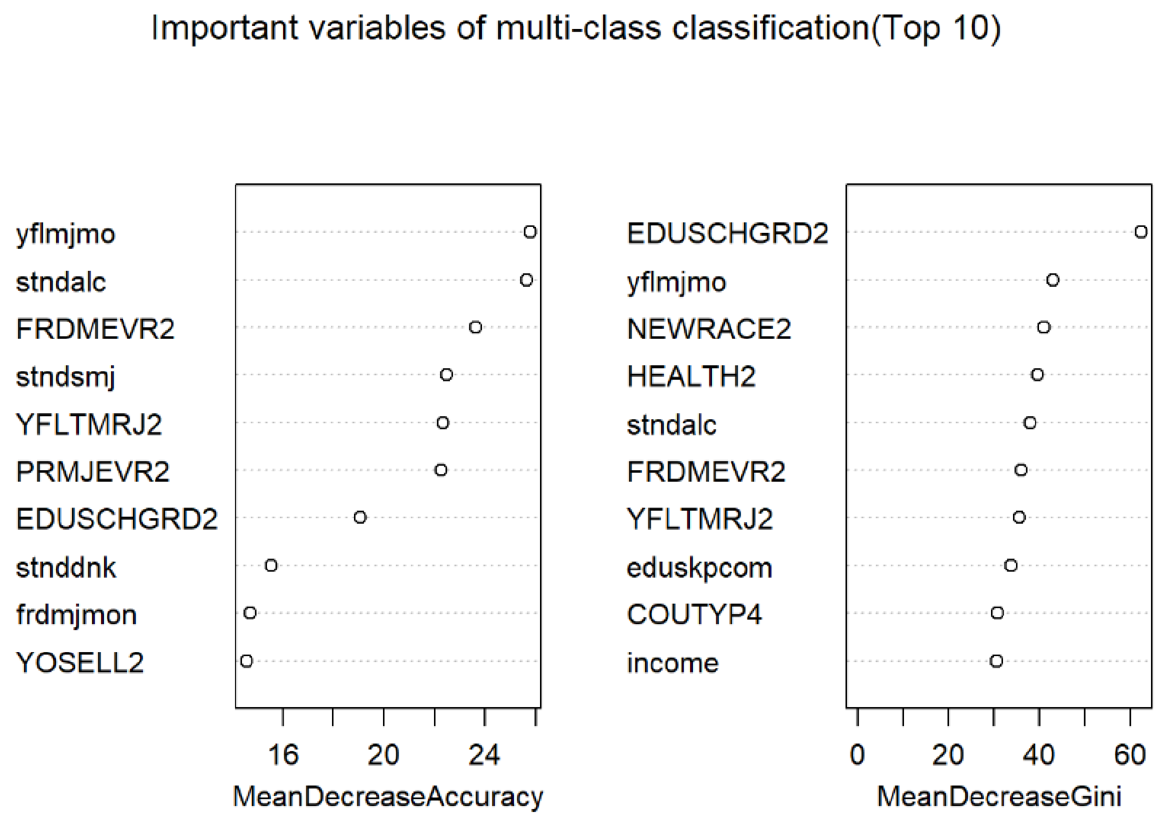
Cross validation results of binary classification:



Important variables of multi-class classification -Bagging



Important variables of multi-class classification -RandomForest



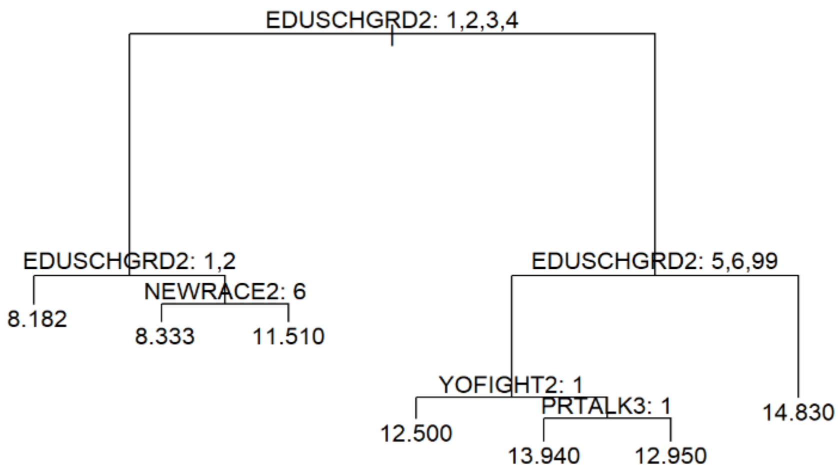
REGRESSION

Accuracy values of different models used for regression are:

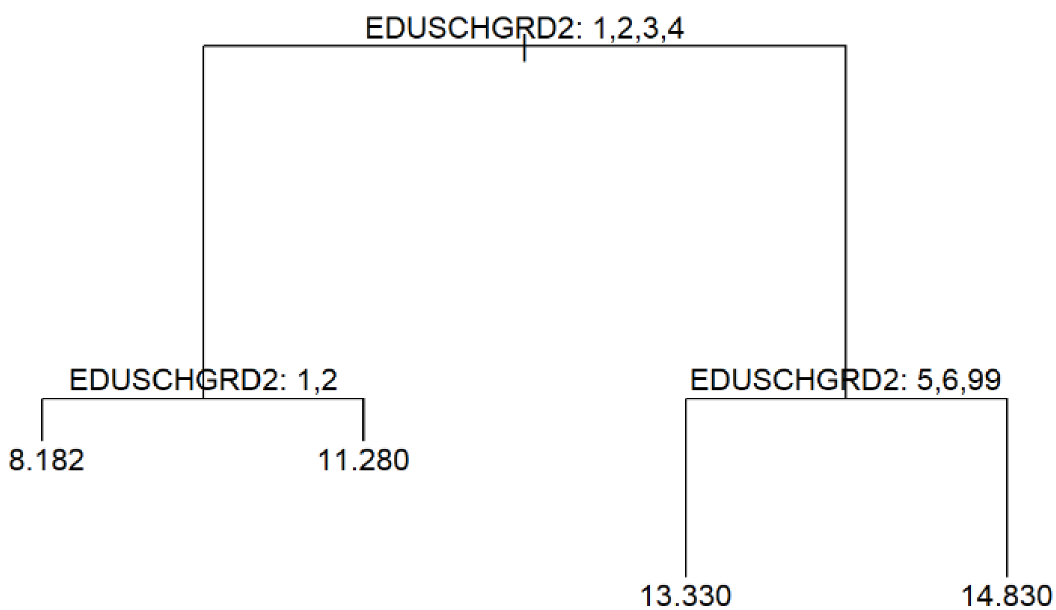
- Decision Tree- 2.909
- Pruned- 2.757
- Bagging- 2.549
- RandomForest-2.572

According to the regression mean square error of the models, the best model to predict the age that alcohol was used for the first time is bagging giving the least MSE of 2.549.

Decision Trees:

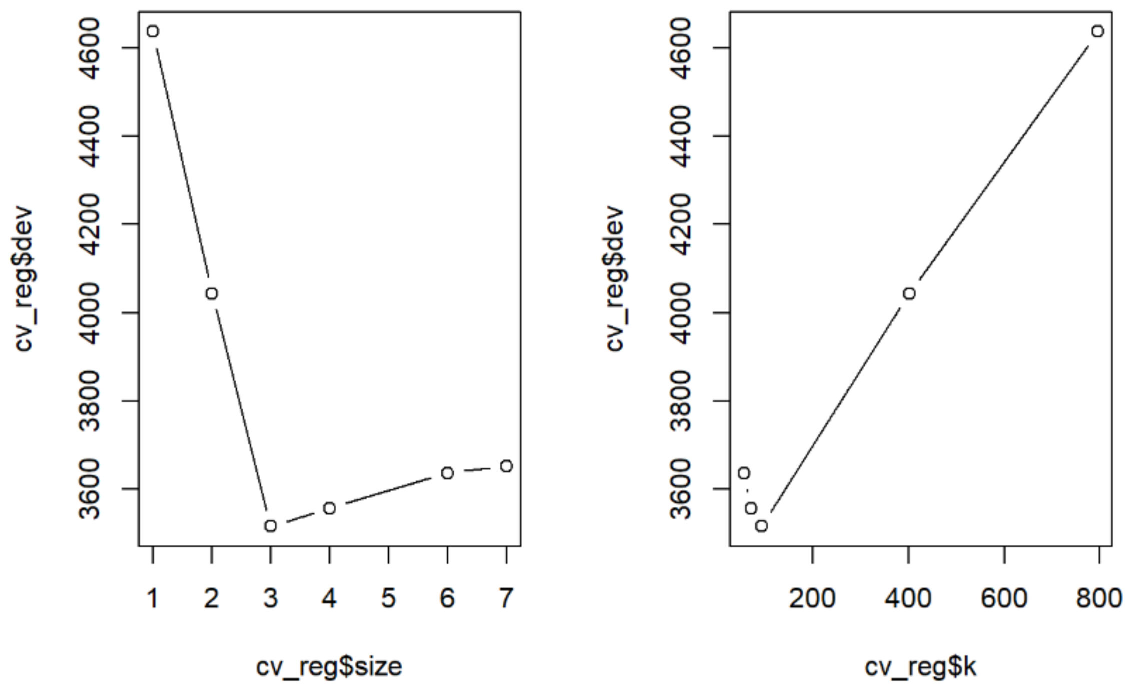


Decision Tree of regression of the age that alcohol was used for the first time.

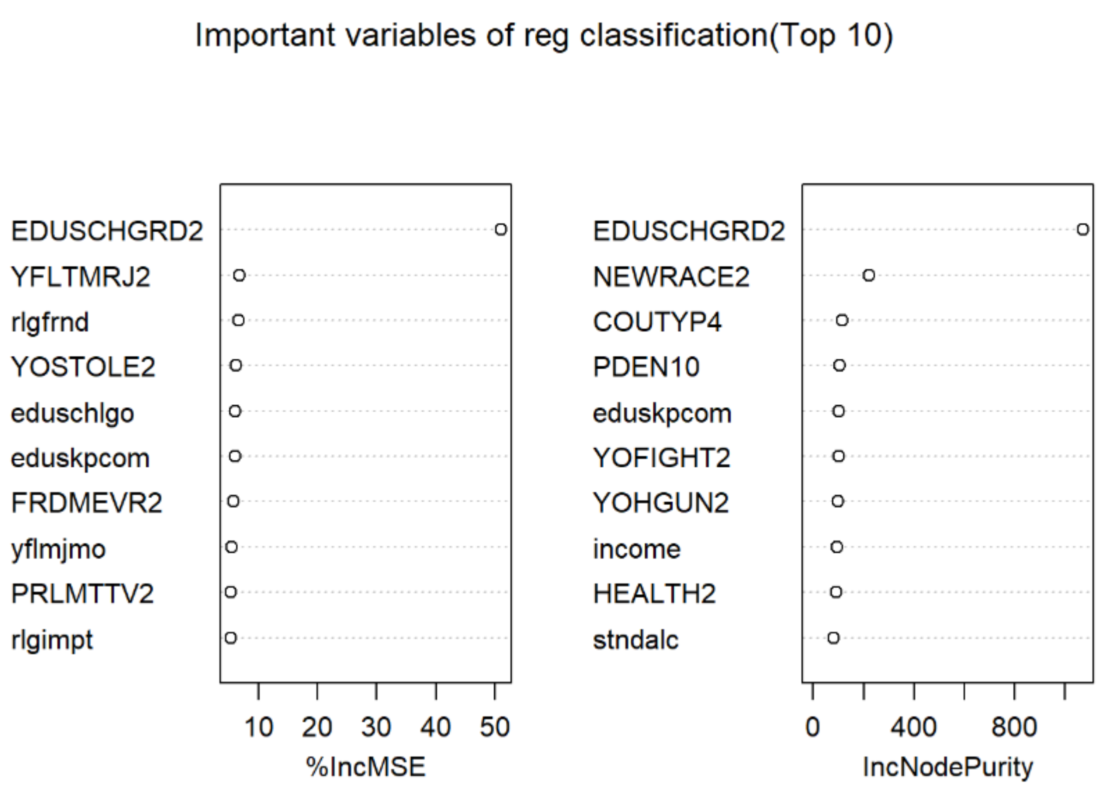


Pruned Decision Tree of regression of the age that alcohol was used for the first time.

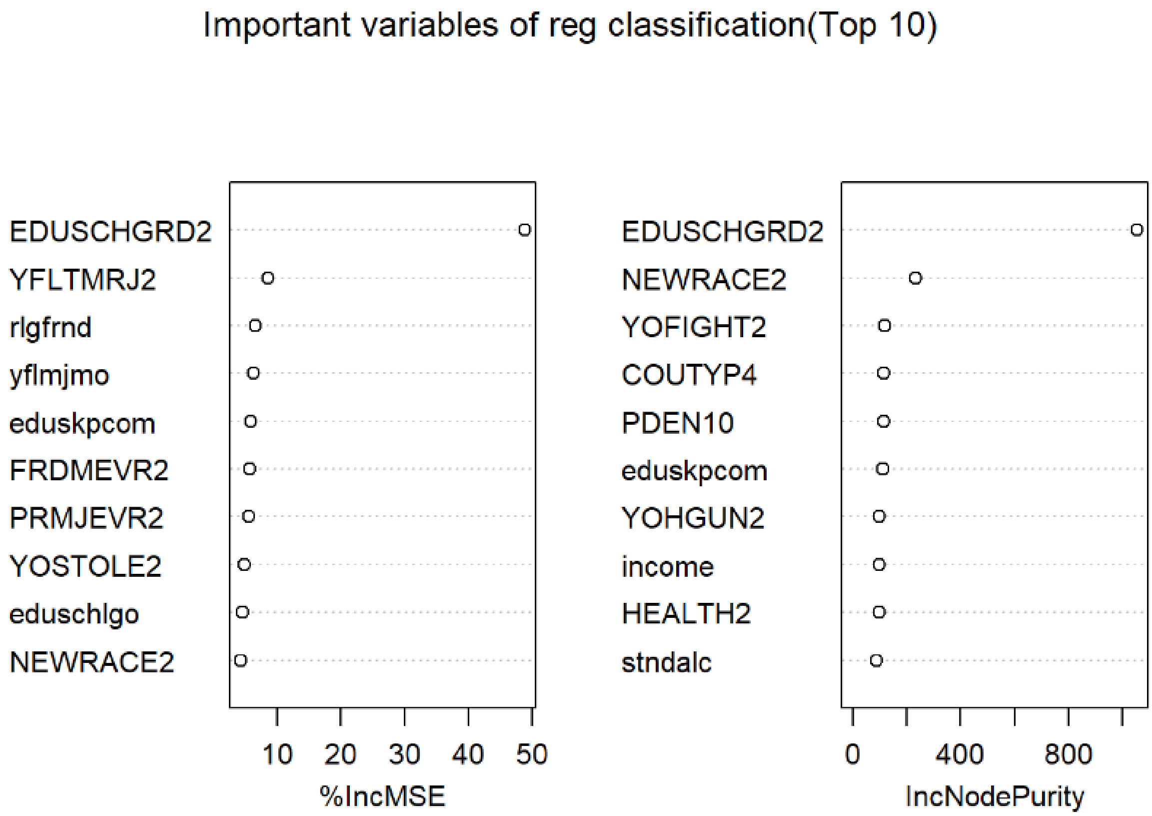
Cross validation results of Regression:



Important variables of regression -Bagging



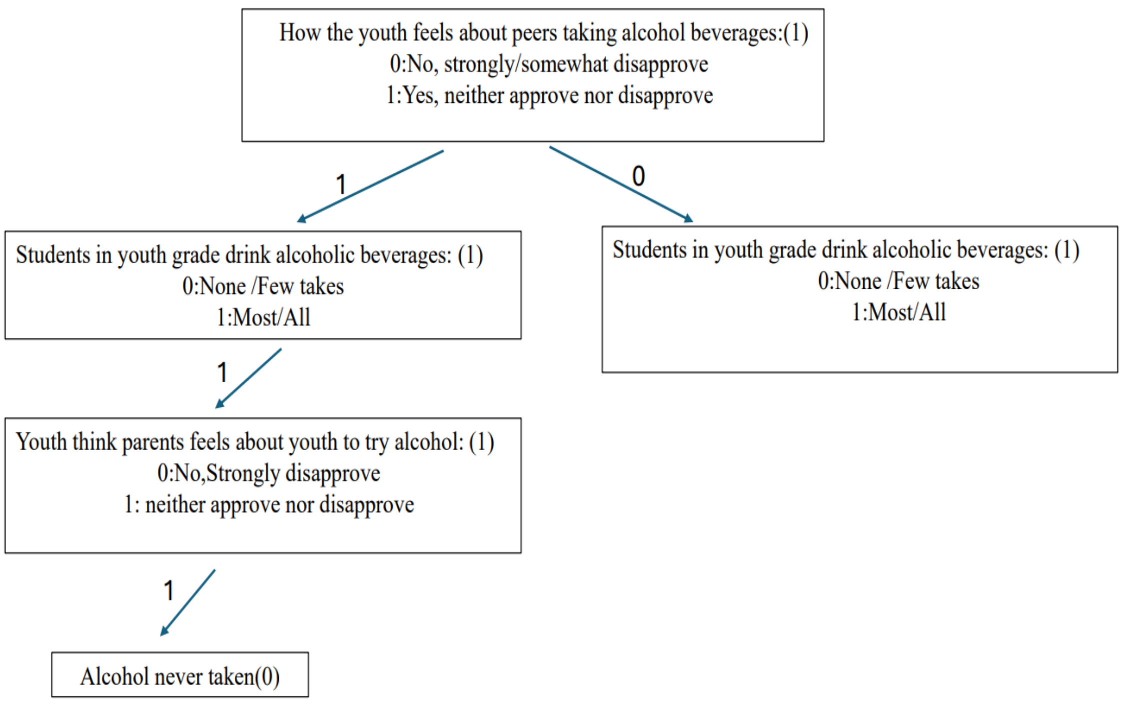
Important variables of regression -RandomForest



DISCUSSION

BINARY CLASSIFICATION

DECISION TREE



In binary classification model decision tree results whether they used alcohol or not. The first question being asked at the root node is how the youth feels about peers taking an alcohol beverage as it results in 1 it turns to left node i.e. yes, it's neither approved nor disapproved usage of alcohol. If they strongly or somewhat disapprove, the algorithm moves down the right branch, and if they neither approve nor disapprove, it moves down the left branch. At its left node.

Then the algorithm moves to the left node asking whether the student youth grade drink alcoholic beverages if it 0 then it moves to right node i.e. None/few takes and if its 1 it moves to left node where mostly all youth students drinking alcohol in this its showing 1 so its moves to left node.

Then it asks youth thinks parents feels about youth try alcohol if its 0: its moves to no, strongly dis approve if it's one it moves to left node i.e. neither some parents approve youth to take alcohol or not. So, in this case it moves to 1 i.e. 0 alcohol never taken then youth are more likely to never use alcohol.

ACCURACY:

Accuracy values of different models used for binary classification are:

- Decision Tree- 0.801
- Pruned- 0.801
- Bagging- 0.815
- RandomForest-0.819

According to the accuracy of the binary classification models, the best model to predict if alcohol is ever used or not is Random Forest with an accuracy of 0.819 i.e. 81%.

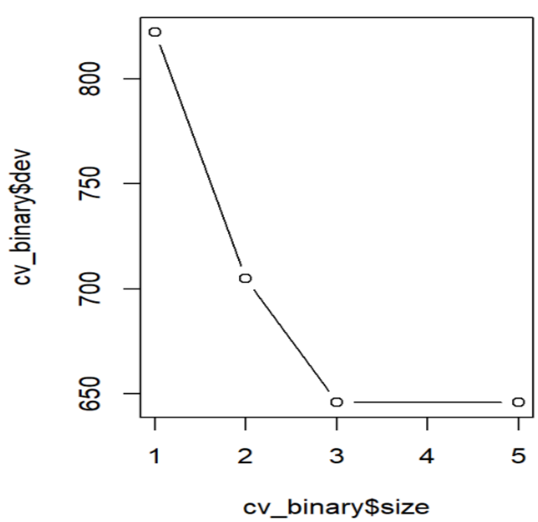
Random Forest:

Random Forest is the best for binary classification model:

```
##       0   1 class.error
## 0 1996 163  0.07549792
## 1  425 397  0.51703163
```

According to the confusion matrix, 1996 of youth who never used alcohol are classified correctly and 387 youth who uses alcohol are predicted correctly. 163 people who use alcohol are misclassified as never used and 425 people that don't use alcohol are misclassified as used.

Cross validation of binary classification:

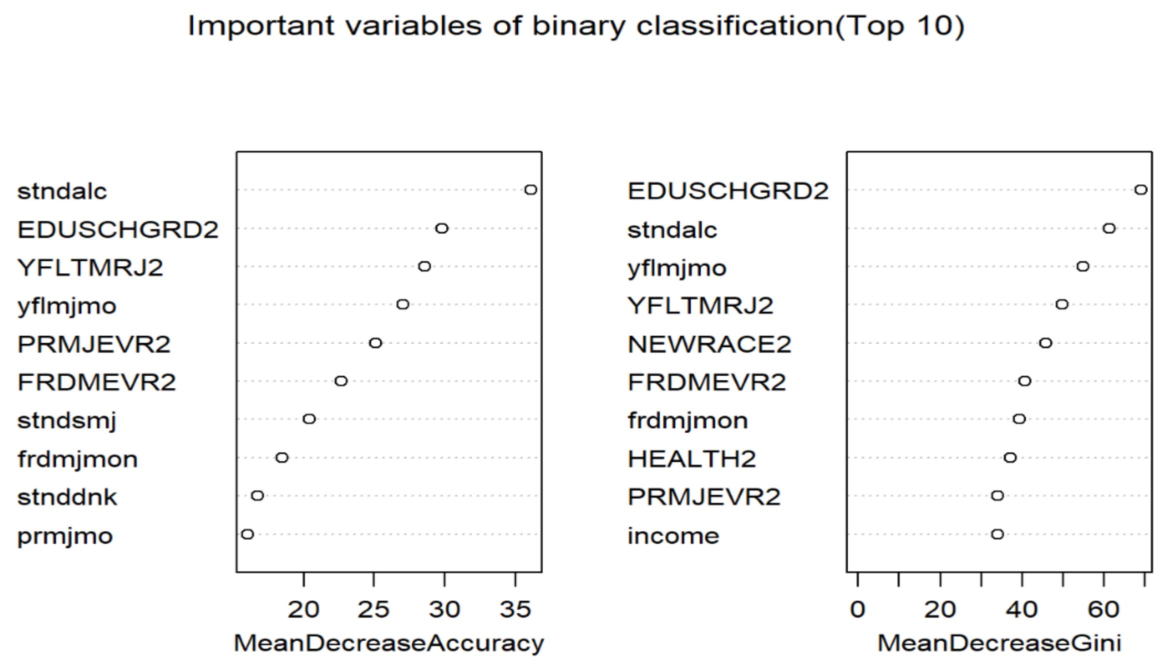


For pruned decision tree the most complex tree under consideration is selected by

cross-validation to see whether pruning the tree will improve performance i.e. best=3.

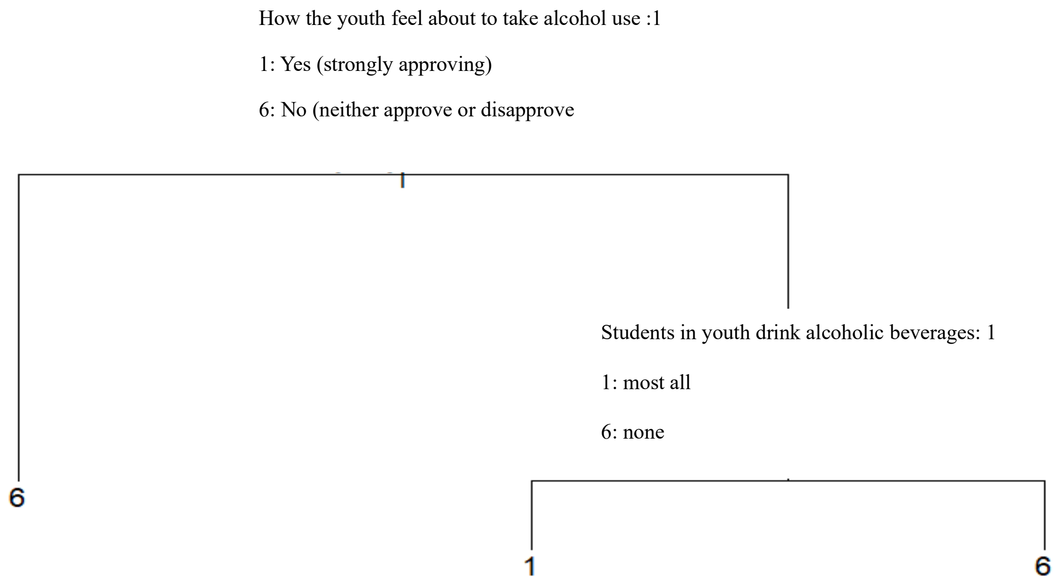
Important variables in binary classification-RandomForest

According to accuracy Random Forest is best model so interpret the important variables of binary classification of random forest.



According to the plots, the most important variables that could predict if a person ever used alcohol or not by mean decrease accuracy and mean decrease gini have both common important variable from the above plots they are top 10 important variables in the given data now in this discussing some important variables tend to be important for predicting usage of alcohol that are stndalc-students in youth grade drink alcohol beverages, EDUSCHGRD2-(educational background the youth /student can be in what grade and going to be), YFLTMRJ2(rc-how youth feels: peers try marijuana drug). Yflmjmo-(how the youth feels peers trying marijuana drug monthly), PRMJEVR2-(youth think parents feels about them to trying marijuana drug) , finding an youth weather used alcohol or not the important variables mostly on the student feeling and educational background and parental behaviors.

MULTI-CLASS CLASSIFICATION



Pruned decision tree.

Decision tree

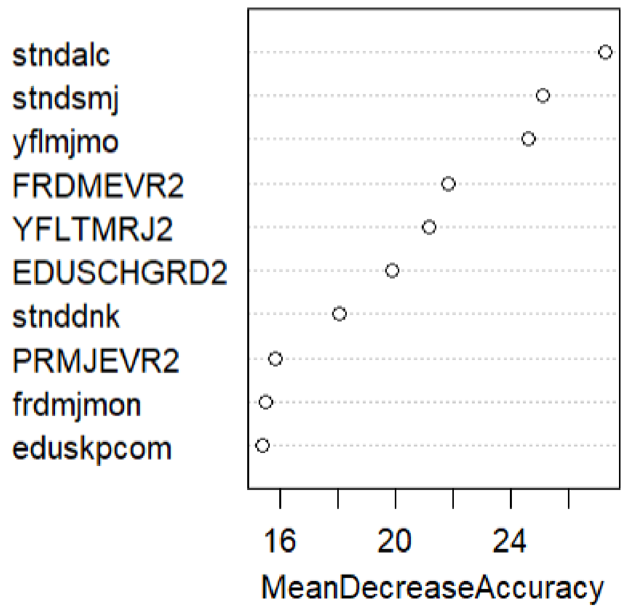
```
prune_multi_prediction    1    2    3    4    5    6
                     1   42   36   13    9    1   62
                     2    0    0    0    0    0    0
                     3    0    0    0    0    0    0
                     4    0    0    0    0    0    0
                     5    0    0    0    0    0    0
                     6  143   48   11    4    0 1282
[1] 0.8019382
```

The confusion matrix shows the number of predicted values that match or do not match the actual values in the test dataset for a multi-class classification problem. The rows of the matrix represent the predicted values, and the columns represent the actual values. In this case, the matrix is a 6x6 table, where the predicted values range from 1 to 6 classes and correspond to the frequency of alcohol use in the past year. The diagonal elements of the matrix represent the number of correct predictions, while the off-diagonal elements represent the number of incorrect predictions. Looking at the matrix, it can see that the model correctly predicted the non-user category (level 6) for 1282 cases. However, it performed poorly in predicting the other levels, particularly level 1 where it correctly predicted only 13. The model did not predict any cases in levels 2, 3,4 or 5. The overall accuracy of the model is 0.801, which means that it correctly classified 80.19% of the cases in the test data.

Bagging:

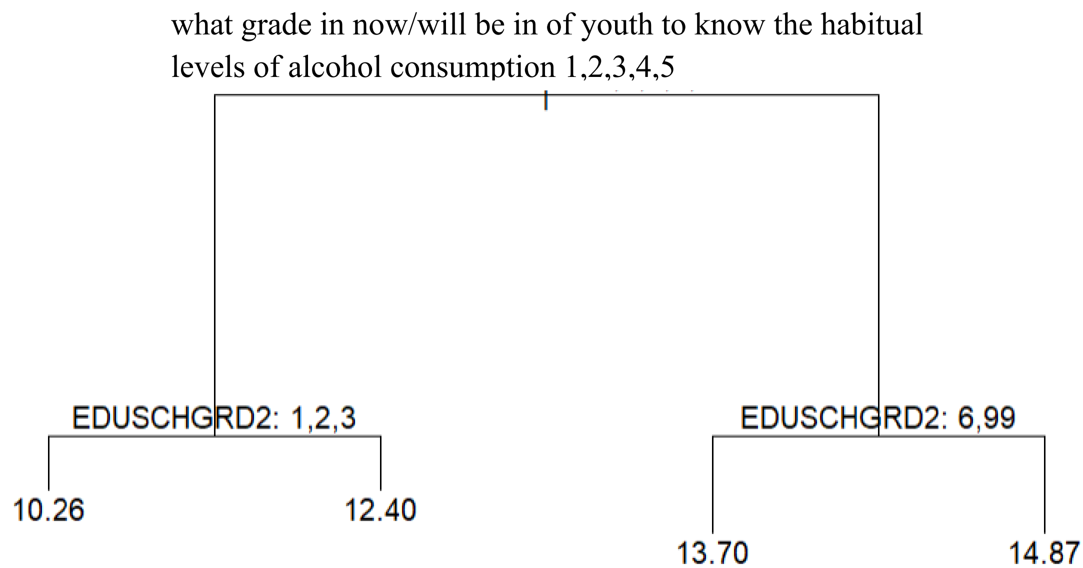
```
bagging_multi_prediction   1   2   3   4   5   6
                       1  30  26   5   6   0  24
                       2   1   1   2   0   0   1
                       3   0   0   0   0   0   0
                       4   0   0   0   0   0   0
                       5   0   0   0   0   0   0
                       6 129  48  16   5   0 961
[1] 0.7904382
```

961 cases of level 6 are classified correctly and some cases in classes 1,2 are classified correctly. So, I think I prefer bagging, With value of 0.790 i.e. 79.04%.



According to the above plot, the most important variables that can be helpful in classifying the number of days of alcohol use in past year stndalc-students in youth grade drink alcohol beverages, stndsmj": rc-students in youth grade use marijuana frdmjmon(rc-yth think: clse frnds feel abt yth use marijuana mon), YFLTMRJ2(rc-how yth feels: peers try marijuana). In terms of multiclass classification, the goal was to ascertain broader substance use patterns, including alcohol, tobacco, and marijuana use. The results indicated that besides the factors mentioned above, participation in self-esteem and problem-solving groups also emerged as crucial predictors.

REGRESSION



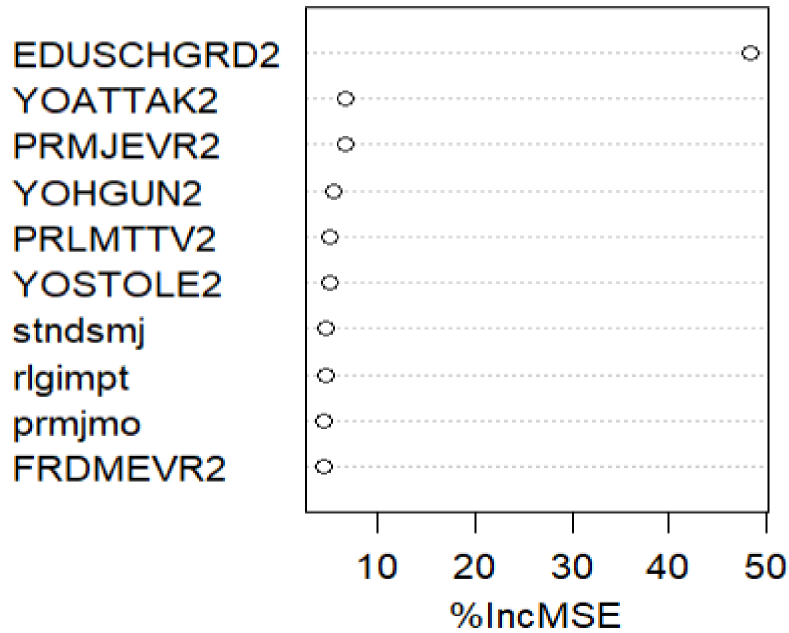
Pruned decision tree of regression.

According to the MSE of the models, the best model to predict age that alcohol was used for the first time is random forest.

These are the top 10 variables of regression classification-Random Forest where these variables helped to how many days per year a person has used alcohol.

	%IncMSE	IncNodePurity
irsex	-0.7279312	51.68982
NEWRACE2	3.3429212	183.92567
HEALTH2	0.3351978	95.61823
eduschlgo	4.3083408	43.00064
EDUSCHGRD2	36.4241722	674.36688
eduskpcom	1.1453435	84.00734
imother	1.5363406	25.08937
ifather	1.1002446	44.26671
income	2.5771080	86.27485
govtprog	2.5185810	37.73039

Random Forest



According to the above plot, the top variables that are used to predict the age of first use of alcohol is EDUSCHGRD2(what grade in now/will be in), YOATTAK2:(rc-youth attacked with intent to seriously harm), PRMJEVR2: (rc-think: parents feel about youth try marijuana). Mostly this variable EDUSCHGRD2 plays an important role in predicting how many days per year a person has used alcohol.

Influential Factors in Substance Use: In this analysis revealed that parental communication about substance use, the child's academic achievements, and their attitudes towards school play pivotal roles in influencing their likelihood of consuming alcohol. Specifically, positive school experiences and open conversations about substance risks are linked to a lower likelihood of substance use, underscoring the importance of supportive educational and familial environments.

CONCLUSION

This research is rooted in analyzing the factors contributing to youth drug use through the utilization of decision tree models and ensemble methods on data sourced from the National Survey on Drug Use and Health. The study's outcomes bear practical significance for public health interventions aimed at curbing youth drug use.

The study revealed the significance of certain demographic and youth experience variables as pivotal predictors of youth drug use, particularly alcohol consumption. Notably, the decision tree models demonstrated commendable accuracy in binary classification for alcohol use, multi-class classification for the frequency of alcohol use within a year, and regression for the initial use of alcohol. The incorporation of ensemble methods such as bagging and random forests further bolstered the models' accuracy, underscoring the value of employing diverse techniques to construct robust models for forecasting substance use among young individuals.

In terms of binary classification, the random forest model emerged with the highest accuracy of 0.819, signifying its efficacy in predicting whether an individual has ever used alcohol. Meanwhile, for multi-class classification, the decision tree and pruning models showcased the highest accuracy of 0.797, positioning them as optimal choices for predicting the number of days of alcohol use over the past year. In regression analysis, the bagging model exhibited the lowest Mean Squared Error (MSE) of 2.549, establishing its superiority in predicting the age of first alcohol use.

Through an examination of confusion matrices and feature importance plots, pivotal variables influencing alcohol use prediction were identified. For binary classification, crucial variables encompassed the youth's perception of their peers' monthly alcohol usage, their close friends' views on youth trying alcohol, and alcohol use among students in the same grade. In multi-class classification, key variables included how youth perceived parental views on alcohol use, their close friends' opinions on trying alcohol, and whether they had experimented with smoking cigarettes. Regarding regression, significant variables comprised the youth's risk perception regarding alcohol use, their weekly exercise frequency, and the frequency of feeling depressed in the past week.

In summary, models provide invaluable insights into the factors influencing alcohol use among youth, equipping policymakers and healthcare practitioners with the tools to identify and mitigate underlying risk factors effectively, thus fostering a healthier and safer environment for young individuals.

REFERENCES

- https://www.samhsa.gov/data/sites/default/files/reports/rpt42731/2022-nsduh-nnr.pdf
- https://www.datafiles.samhsa.gov/data-sources
- Referred class notes 1 and 2 regarding decision trees to work on this assignment.

Decision tree -binary classification

LAVANYA B

```
youth_data=load("C:/Users/bunad/OneDrive/Desktop/SPRING 2024/MACHINE LEARNING 2/youth_data.Rdata")
youth_data=df
```

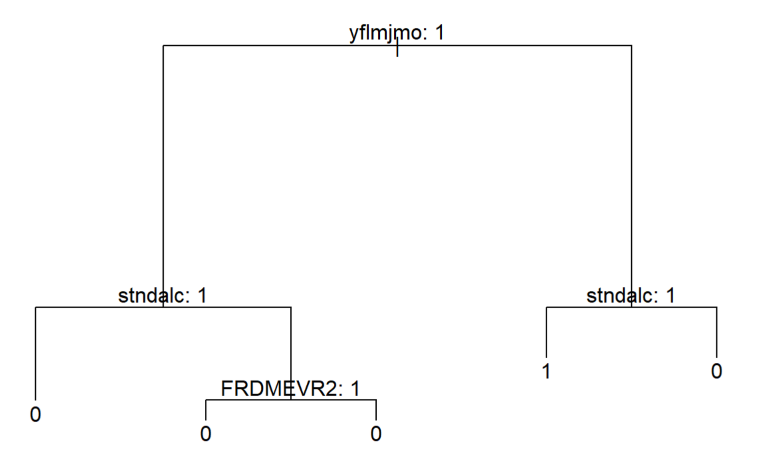
```
cleaned_youth_data=na.omit(youth_data)
```

```
cleaned_youth_data_subset=df[,c(demographic_cols,youth_experience_cols,'alcflag')]
train=sample(1:nrow(cleaned_youth_data_subset),0.7*nrow(cleaned_youth_data_subset))
training_data_binary=cleaned_youth_data_subset[train,]
testing_data_binary=cleaned_youth_data_subset[-train,
                                              ]
```

```
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.3.3
```

```
binary_tree=tree(alcflag~.,data=training_data_binary)
plot(binary_tree)
text(binary_tree,pretty = 0)
```




```
binary_tree
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 2984 3485.0 0 ( 0.72922 0.27078 )
##    2) yflmjmo: 1 2310 2101.0 0 ( 0.83074 0.16926 )
##      4) stndalc: 1 487  648.7 0 ( 0.61602 0.38398 ) *
##      5) stndalc: 2 1823 1278.0 0 ( 0.88810 0.11190 )
##       10) FRDMEVR2: 1 1724 1115.0 0 ( 0.90081 0.09919 ) *
##       11) FRDMEVR2: 2 99  126.0 0 ( 0.66667 0.33333 ) *
##    3) yflmjmo: 2 674  896.0 1 ( 0.38131 0.61869 )
##      6) stndalc: 1 328  326.6 1 ( 0.19817 0.80183 ) *
##      7) stndalc: 2 346  475.5 0 ( 0.55491 0.44509 ) *
```

```
predict_binary=predict(binary_tree,testing_data_binary,type='class')
table(predict_binary,testing_data_binary$alcflag)
```

```
##
## predict_binary    0    1
##              0 1205  287
##              1   42  117
```

```
mean(predict_binary==testing_data_binary$alcflag)
```

```
## [1] 0.8007268
```

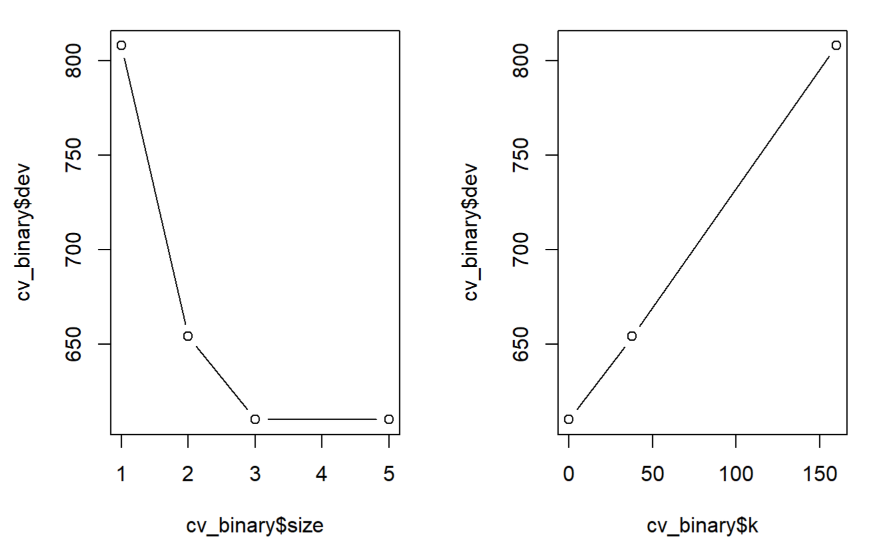
```
cv_binary=cv.tree(binary_tree,FUN=prune.misclass)
names(cv_binary)
```

```
## [1] "size"   "dev"    "k"      "method"
```

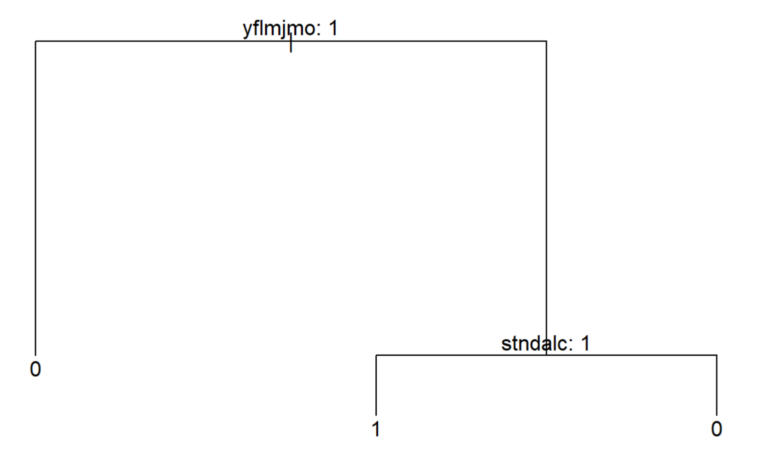
```
cv_binary
```

```
## $size
## [1] 5 3 2 1
##
## $dev
## [1] 610 610 654 808
##
## $k
## [1] -Inf    0   38  160
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
par(mfrow = c(1, 2))
plot(cv_binary$size, cv_binary$dev, type = "b")
plot(cv_binary$k, cv_binary$dev, type = "b")
```




```
prune_binary <- prune.misclass(binary_tree, best = 3)
plot(prune_binary)
text(prune_binary, pretty = 0)
```




```
prune_binary
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
## 1) root 2984 3485.0 0 ( 0.7292 0.2708 )
##   2) yflmjmo: 1 2310 2101.0 0 ( 0.8307 0.1693 ) *
##   3) yflmjmo: 2 674  896.0 1 ( 0.3813 0.6187 )
##     6) stndalc: 1 328  326.6 1 ( 0.1982 0.8018 ) *
##     7) stndalc: 2 346  475.5 0 ( 0.5549 0.4451 ) *
```

```
prune_binary_prediction = predict(prune_binary, testing_data_binary,type = "class")
table(prune_binary_prediction, testing_data_binary$alcflag)
```

```
##
## prune_binary_prediction    0    1
##                       0 1205  287
##                       1   42  117
```

```
mean(prune_binary_prediction == testing_data_binary$alcflag)
```

```
## [1] 0.8007268
```

```
missing_values=colSums(is.na(training_data_binary))
missing_values
```

```
##      irsex   NEWRACE2    HEALTH2  eduschlgo EDUSCHGRD2  eduskpcom    imother
##          0          0          1          0          0          0          0
##    ifather     income   govtprog   POVERTY3     PDEN10    COUTYP4    schfelt
##          0          0          0          0          0          0          0
##    tchgjob   avggrade   stndscig    stndsmj    stndalc    stnddnk   parchkhw
##          6        239        172        183        174        213         15
##   parhlphw   PRCHORE2   PRLMTTV2   parlmtsn   PRGDJOB2   PRPROUD2    argupar
##         32         14         29         87         20         17         61
##   YOFIGHT2   YOGRPFT2    YOHGUN2    YOSELL2   YOSTOLE2   YOATTAK2   PRPKCIG2
##         16         22         18          8          8          8         38
##   PRMJEVR2     prmjmo   PRALDLY2   YFLPKCG2   YFLTMRJ2    yflmjmo   YFLADLY2
##         37         44         36         34         35         33         32
##   FRDPCIG2   FRDMEVR2   frdmjmon   FRDADLY2   talkprob    PRTALK3   PRBSOLV2
##         60         56         61         49        108         71        122
##   PREVIOL2   PRVDRGO2   GRPCNSL2   PREGPGM2    YTHACT2   DRPRVME3   ANYEDUC3
##         58         47         44         33         17         62         53
##    rlgattd    rlgimpt    rlgdcsn    rlgfrnd    alcflag
##         93         83         93        105          0
```

Bagging

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.3
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
training_data_binary_clean=na.omit(training_data_binary)
bagging_binary <- randomForest(alcflag ~ ., data = training_data_binary_clean,
     mtry = floor(sqrt(ncol(training_data_binary_clean))), importance = TRUE)
bagging_binary
```

```
##
## Call:
##  randomForest(formula = alcflag ~ ., data = training_data_binary_clean,      mtry = floor(sqrt(ncol(training_d
ata_binary_clean))), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 18.7%
## Confusion matrix:
##      0   1 class.error
## 0 2023 153   0.0703125
## 1  405 403   0.5012376
```

```
bagging_binary_prediction = predict(bagging_binary, newdata = testing_data_binary, type = "class")
table(bagging_binary_prediction,testing_data_binary$alcflag)
```

```
##
## bagging_binary_prediction   0   1
##                         0 860 183
##                         1  78 164
```

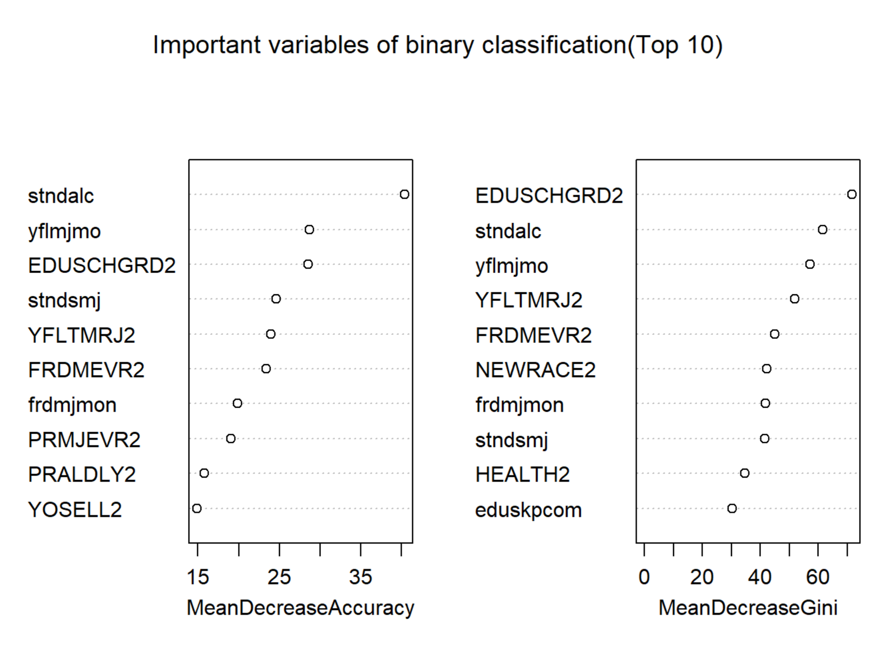
```
mean(bagging_binary_prediction== testing_data_binary$alcflag,na.rm=TRUE)
```

```
## [1] 0.7968872
```

```
importance_bagging_binary <- importance(bagging_binary)
top_10 <- head(importance_bagging_binary, 10)
top_10
```

```
##                      0           1 MeanDecreaseAccuracy MeanDecreaseGini
## irsex      -0.03416686  0.51682428            0.2957109        17.103627
## NEWRACE2    3.42997179  4.64550549            5.5041511        42.234189
## HEALTH2     0.21276282  1.46883480            0.9966403        34.720758
## eduschlgo   7.59592441 -5.57464113            5.8128129         8.667769
## EDUSCHGRD2 16.98690334 23.17315085           28.5282139        71.582460
## eduskpcom  10.69353098 -0.72759588            8.4772492        30.301028
## imother    -1.43485608  0.89257438           -0.6874139         7.722756
## ifather     3.23693123  2.47004489            4.1228180        13.098956
## income      7.04248167  0.06769337            6.5443706        28.392141
## govtprog    7.58917135 -1.34418271            6.1523087        11.296345
```

```
varImpPlot(bagging_binary, n.var = 10, sort = TRUE, main = "Important variables of binary classification(Top 1
0)")
```



#Random Forest

```
set.seed(1)
randomforest_binary <- randomForest(alcflag ~ ., data = training_data_binary_clean, mtry = sqrt(ncol(training_dat
a_binary_clean)), ntree = 500, importance = TRUE)
randomforest_binary
```

```
##
## Call:
##  randomForest(formula = alcflag ~ ., data = training_data_binary_clean,      mtry = sqrt(ncol(training_data_bi
nary_clean)), ntree = 500,      importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 8
##
##         OOB estimate of  error rate: 19.07%
## Confusion matrix:
##      0   1 class.error
## 0 2018 158  0.07261029
## 1  411 397  0.50866337
```

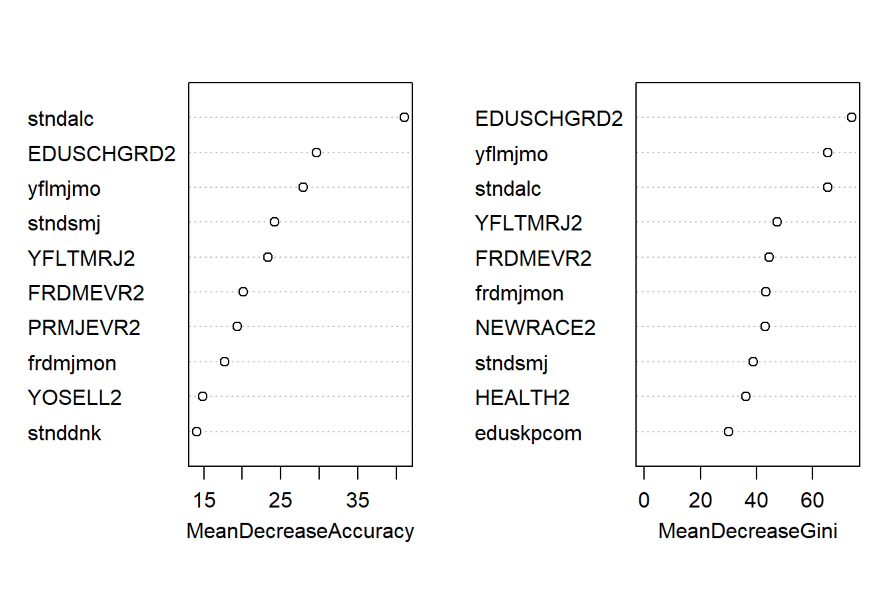
```
yhat_randomforest_binary <- predict(randomforest_binary, newdata = testing_data_binary,type='class')
mean(yhat_randomforest_binary == testing_data_binary$alcflag,na.rm=TRUE)
```

```
## [1] 0.8023346
```

```
importance_rf_binary <- importance(randomforest_binary)
top_10 <- head(importance_rf_binary, 10)
top_10
```

```
##                     0          1 MeanDecreaseAccuracy MeanDecreaseGini
## irsex      -1.2860902 -0.6245876           -1.4018574        17.369274
## NEWRACE2    5.8724984  3.7268656            6.9949289        43.128240
## HEALTH2    -0.3624786  1.4289066            0.4573118        36.229450
## eduschlgo   9.5253948 -3.9768470            7.7904301         8.906189
## EDUSCHGRD2 18.0005075 24.7830597           29.6711402        73.928664
## eduskpcom   9.6935607 -0.2438444            8.5483782        30.033559
## imother    -1.5827776 -1.5073627           -2.1208027         7.665829
## ifather     1.9839961  0.7216975            2.1070404        13.455991
## income      8.0186872 -0.2186335            7.0580636        29.113653
## govtprog    8.1416548 -1.7902315            6.2209410        10.742277
```

```
varImpPlot(randomforest_binary, n.var = 10, sort = TRUE, main = "Important variables of binary classification(Top
10)")
```



Decision tree- multi class classification

LAVANYA B

```
youth_data=load("C:/Users/bunad/OneDrive/Desktop/SPRING 2024/MACHINE LEARNING 2/youth_data.Rdata")
youth_data=df
```

```
cleaned_youth_data=na.omit(youth_data)
```

```
cleaned_youth_data_subset_multi=df[,c(demographic_cols,youth_experience_cols,'alcydays')]
cleaned_youth_data_subset_multi$alcydays <- as.factor(cleaned_youth_data_subset_multi$alcydays)
length(cleaned_youth_data_subset_multi)
```

```
## [1] 61
```

```
train=sample(1:nrow(cleaned_youth_data_subset_multi),0.7*nrow(cleaned_youth_data_subset_multi))
training_data_multi=cleaned_youth_data_subset_multi[train,]
testing_data_multi=cleaned_youth_data_subset_multi[-train,]
```

```
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.3.3
```

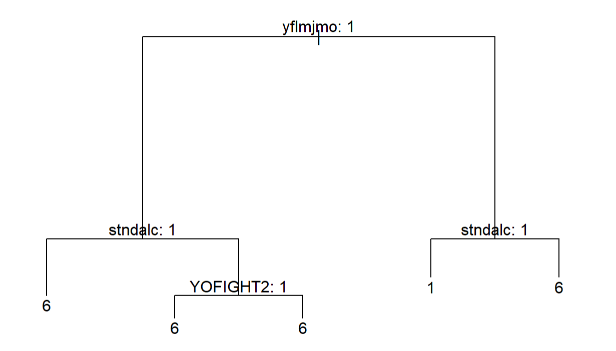
```
multi_tree <- tree(alcydays ~.,training_data_multi)
multi_tree
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 2983 4569.0 6 ( 0.1350989 0.0563191 0.0181026 0.0144150 0.0003352 0.7757291 )
##    2) yflmjmo: 1 2312 2330.0 6 ( 0.0826125 0.0289792 0.0099481 0.0051903 0.0004325 0.8728374 )
##      4) stndalc: 1 496  914.3 6 ( 0.1653226 0.0826613 0.0282258 0.0141129 0.0020161 0.7076613 ) *
##      5) stndalc: 2 1816 1274.0 6 ( 0.0600220 0.0143172 0.0049559 0.0027533 0.0000000 0.9179515 )
##       10) YOFIGHT2: 1 226  319.2 6 ( 0.1283186 0.0221239 0.0265487 0.0221239 0.0000000 0.8008850 ) *
##       11) YOFIGHT2: 2 1590  898.7 6 ( 0.0503145 0.0132075 0.0018868 0.0000000 0.0000000 0.9345912 ) *
##    3) yflmjmo: 2 671 1737.0 6 ( 0.3159463 0.1505216 0.0461997 0.0461997 0.0000000 0.4411326 )
##      6) stndalc: 1 343  982.2 1 ( 0.3323615 0.2361516 0.0816327 0.0553936 0.0000000 0.2944606 ) *
##      7) stndalc: 2 328  659.0 6 ( 0.2987805 0.0609756 0.0091463 0.0365854 0.0000000 0.5945122 ) *
```

```
summary(multi_tree)
```

```
##
## Classification tree:
## tree(formula = alcydays ~ ., data = training_data_multi)
## Variables actually used in tree construction:
## [1] "yflmjmo"  "stndalc"  "YOFIGHT2"
## Number of terminal nodes:  5
## Residual mean deviance:  1.267 = 3773 / 2978
## Misclassification error rate: 0.2199 = 656 / 2983
```

```
plot(multi_tree)
text(multi_tree, pretty = 0)
```




```
predict_multi <- predict(multi_tree, testing_data_multi, type = 'class')
table(predict_multi, testing_data_multi$alcydays)
```

```
##
## predict_multi    1    2    3    4    5    6
##             1   47   30   12    9    1   49
##             2    0    0    0    0    0    0
##             3    0    0    0    0    0    0
##             4    0    0    0    0    0    0
##             5    0    0    0    0    0    0
##             6  154   43   16    7    0 1283
```

```
mean(predict_multi == testing_data_multi$alcydays)
```

```
## [1] 0.8055724
```

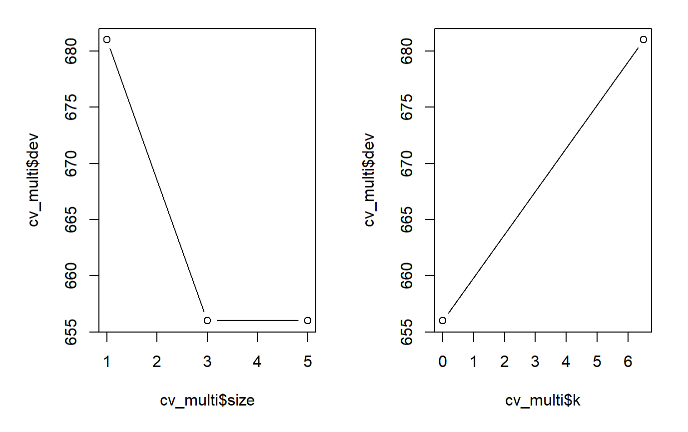
```
cv_multi=cv.tree(multi_tree,FUN=prune.misclass)
names(cv_multi)
```

```
## [1] "size"   "dev"    "k"      "method"
```

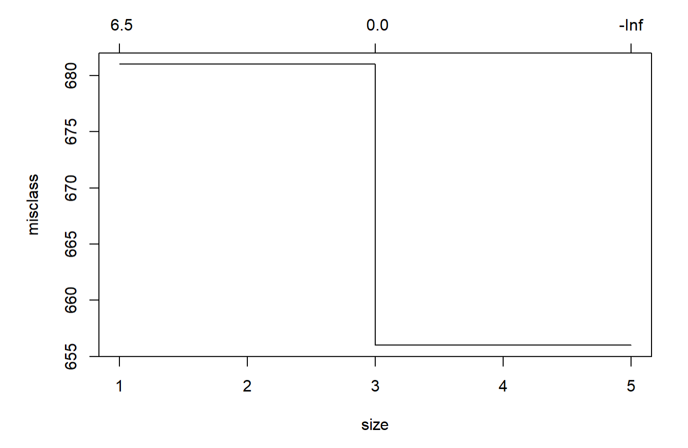
```
cv_multi
```

```
## $size
## [1] 5 3 1
##
## $dev
## [1] 656 656 681
##
## $k
## [1] -Inf  0.0  6.5
##
## $method
## [1] "misclass"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

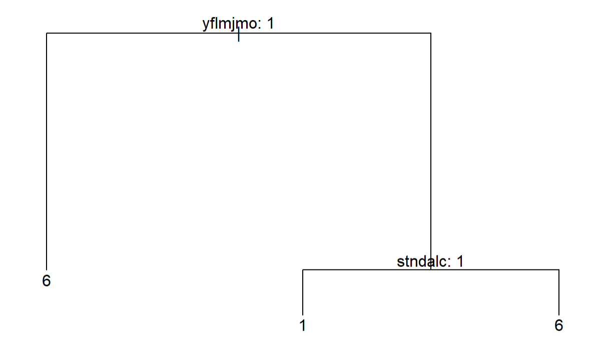
```
par(mfrow = c(1, 2))
plot(cv_multi$size, cv_multi$dev, type = "b")
plot(cv_multi$k, cv_multi$dev, type = "b")
```




```
plot(cv_multi)
```




```
prune_multi <- prune.misclass(multi_tree, best = 3)
plot(prune_multi)
text(prune_multi, pretty = 0)
```




```
prune_multi
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
## 1) root 2983 4569.0 6 ( 0.1350989 0.0563191 0.0181026 0.0144150 0.0003352 0.7757291 )
##   2) yflmjmo: 1 2312 2330.0 6 ( 0.0826125 0.0289792 0.0099481 0.0051903 0.0004325 0.8728374 ) *
##   3) yflmjmo: 2 671 1737.0 6 ( 0.3159463 0.1505216 0.0461997 0.0461997 0.0000000 0.4411326 )
##     6) stndalc: 1 343  982.2 1 ( 0.3323615 0.2361516 0.0816327 0.0553936 0.0000000 0.2944606 ) *
##     7) stndalc: 2 328  659.0 6 ( 0.2987805 0.0609756 0.0091463 0.0365854 0.0000000 0.5945122 ) *
```

```
prune_multi_prediction = predict(prune_multi, testing_data_multi,type = "class")
table(prune_multi_prediction, testing_data_multi$alcydays)
```

```
##
## prune_multi_prediction    1    2    3    4    5    6
##                      1   47   30   12    9    1   49
##                      2    0    0    0    0    0    0
##                      3    0    0    0    0    0    0
##                      4    0    0    0    0    0    0
##                      5    0    0    0    0    0    0
##                      6  154   43   16    7    0 1283
```

```
mean(prune_multi_prediction == testing_data_multi$alcydays)
```

```
## [1] 0.8055724
```

Bagging

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.3
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(tree)
training_data_multi_clean=na.omit(training_data_multi)
bagging_multi <- randomForest(alcydays ~ ., data = training_data_multi_clean,
     mtry = floor(sqrt(ncol(training_data_multi_clean))), importance = TRUE)
bagging_multi
```

```
##
## Call:
##  randomForest(formula = alcydays ~ ., data = training_data_multi_clean,      mtry = floor(sqrt(ncol(training_d
ata_multi_clean))), importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 21.69%
## Confusion matrix:
##    1 2 3 4 5    6 class.error
## 1 67 9 0 0 0  327  0.83374690
## 2 44 8 0 0 0  116  0.95238095
## 3 14 3 0 0 0   37  1.00000000
## 4 15 2 0 0 0   26  1.00000000
## 5  0 0 0 0 0    1  1.00000000
## 6 51 2 0 0 0 2261  0.02290406
```

```
bagging_multi_prediction = predict(bagging_multi, newdata = testing_data_multi, type = "class")
table(bagging_multi_prediction,testing_data_multi$alcydays)
```

```
##
## bagging_multi_prediction   1   2   3   4   5   6
##                        1  25  18   8   5   0  22
##                        2   3   1   1   0   0   0
##                        3   0   0   0   0   0   0
##                        4   0   0   0   0   0   1
##                        5   0   0   0   0   0   0
##                        6 137  47  19   8   0 991
```

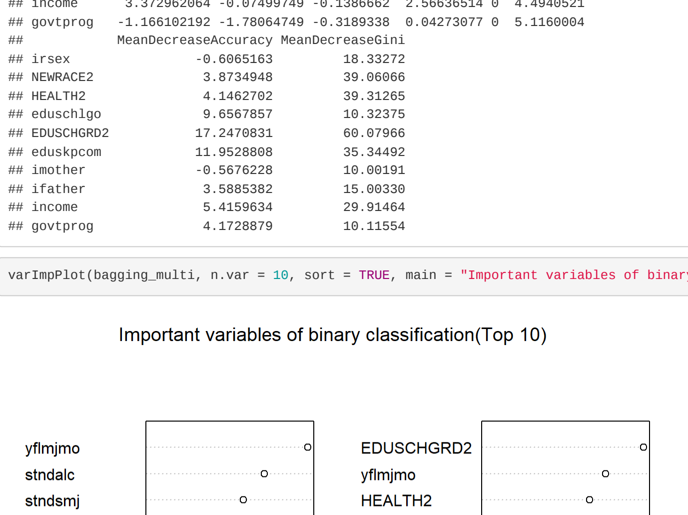
```
mean(bagging_multi_prediction== testing_data_multi$alcydays,na.rm=TRUE)
```

```
## [1] 0.7908243
```

```
importance_bagging_multi <- importance(bagging_multi)
top_10 <- head(importance_bagging_multi, 10)
top_10
```

```
##                      1           2          3           4 5          6
## irsex       0.541380104 -0.87372154  1.3001017 -0.32459138 0 -0.7902109
## NEWRACE2    0.001403991  2.11419073  0.2928847  0.64080627 0  3.9876324
## HEALTH2     1.522909789  1.22076502 -1.9943969 -1.31580183 0  4.0606129
## eduschlgo  -2.555210174  0.07973208  0.9780729  0.36707460 0 10.3884436
## EDUSCHGRD2  7.081776639  9.62197044  1.8303132  1.10279229 0 13.4905337
## eduskpcom  -3.926202091  0.32597323  2.6957031 -0.59501687 0 14.2125802
## imother     1.267046104  1.08399418  1.3906560  0.16251243 0 -1.8147255
## ifather     1.917243954  0.76406240 -1.6095800 -0.44647474 0  3.0795199
## income      3.372962064 -0.07499749 -0.1386662  2.56636514 0  4.4940521
## govtprog   -1.166102192 -1.78064749 -0.3189338  0.04273077 0  5.1160004
##            MeanDecreaseAccuracy MeanDecreaseGini
## irsex                -0.6065163         18.33272
## NEWRACE2              3.8734948         39.06066
## HEALTH2               4.1462702         39.31265
## eduschlgo             9.6567857         10.32375
## EDUSCHGRD2           17.2470831         60.07966
## eduskpcom            11.9528808         35.34492
## imother              -0.5676228         10.00191
## ifather               3.5885382         15.00330
## income                5.4159634         29.91464
## govtprog              4.1728879         10.11554
```

```
varImpPlot(bagging_multi, n.var = 10, sort = TRUE, main = "Important variables of binary classification(Top 10)")
```



#Random Forest

```
set.seed(1)
randomforest_multi <- randomForest(alcydays ~ ., data = training_data_multi_clean, mtry =sqrt(ncol(training_data_
multi_clean)), ntree = 500, importance = TRUE)
randomforest_multi
```

```
##
## Call:
##  randomForest(formula = alcydays ~ ., data = training_data_multi_clean,      mtry = sqrt(ncol(training_data_mu
lti_clean)), ntree = 500,      importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 8
##
##         OOB estimate of  error rate: 22.16%
## Confusion matrix:
##    1  2 3 4 5    6 class.error
## 1 68 13 0 0 0  322  0.83126551
## 2 46 10 0 0 0  112  0.94047619
## 3 12  3 0 0 0   39  1.00000000
## 4 13  1 0 0 0   29  1.00000000
## 5  0  0 0 0 0    1  1.00000000
## 6 67  3 0 0 0 2244  0.03025065
```

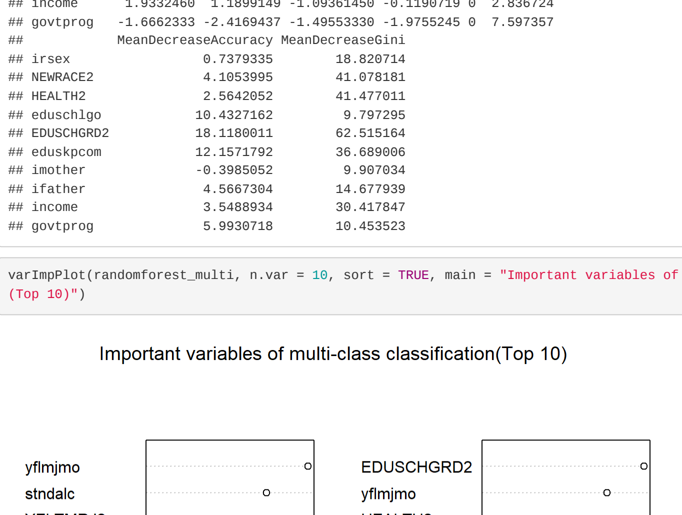
```
yhat_randomforest_multi <- predict(randomforest_multi, newdata = testing_data_multi,type='class')
mean(yhat_randomforest_multi == testing_data_multi$alcydays,na.rm=TRUE)
```

```
## [1] 0.7908243
```

```
importance_rf_multi <- importance(randomforest_multi)
top_10 <- head(importance_rf_multi, 10)
top_10
```

```
##                     1          2           3          4 5         6
## irsex       0.4436281  2.4050310 -0.15949498 -2.1520342 0  0.100984
## NEWRACE2    2.1978485  1.5628568 -0.74708757 -0.7938374 0  3.419926
## HEALTH2     0.4119498  1.1963924 -1.30236787 -2.3015968 0  2.789931
## eduschlgo  -0.5988308 -1.3705855  1.98011410  0.4704401 0 10.852503
## EDUSCHGRD2  6.4441697  9.4538598  0.59513315  0.2659495 0 14.853138
## eduskpcom  -2.7908821  1.1431355  1.78269777  0.9291195 0 13.615913
## imother     1.0451476  0.9951935 -0.08333098 -1.2446280 0 -1.292226
## ifather     3.1148435 -0.3763488 -3.26604806 -2.1854402 0  4.123788
## income      1.9332460  1.1899149 -1.09361450 -0.1190719 0  2.836724
## govtprog   -1.6662333 -2.4169437 -1.49553330 -1.9755245 0  7.597357
##            MeanDecreaseAccuracy MeanDecreaseGini
## irsex                 0.7379335        18.820714
## NEWRACE2              4.1053995        41.078181
## HEALTH2               2.5642052        41.477011
## eduschlgo            10.4327162         9.797295
## EDUSCHGRD2           18.1180011        62.515164
## eduskpcom            12.1571792        36.689006
## imother              -0.3985052         9.907034
## ifather               4.5667304        14.677939
## income                3.5488934        30.417847
## govtprog              5.9930718        10.453523
```

```
varImpPlot(randomforest_multi, n.var = 10, sort = TRUE, main = "Important variables of multi-class classification
(Top 10)")
```



DECISION_TREES_REGRESSION

LAVANYA B

```
youth_data=load("C:/Users/bunad/OneDrive/Desktop/SPRING 2024/MACHINE LEARNING 2/youth_data.Rdata")
youth_data=df
```

```
cleaned_youth_data=na.omit(youth_data)
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
cleaned_youth_data_subset_reg=df[,c(demographic_cols,youth_experience_cols,'iralcage')]
cleaned_youth_data_subset_reg <- cleaned_youth_data_subset_reg %>%
 filter(iralcage != 991)
nrow(cleaned_youth_data_subset_reg)
```

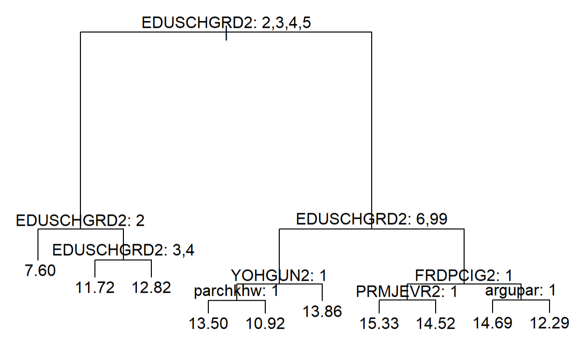
```
## [1] 1362
```

```
train=sample(1:nrow(cleaned_youth_data_subset_reg),0.7*nrow(cleaned_youth_data_subset_reg))
training_data_reg=cleaned_youth_data_subset_reg[train,]
testing_data_reg=cleaned_youth_data_subset_reg[-train,]
```

```
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.3.3
```

```
reg_tree=tree(iralcage ~.,data=training_data_reg)
plot(reg_tree)
text(reg_tree,pretty = 0)
```




```
reg_tree
```

```
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 806 3864.00 13.78
##    2) EDUSCHGRD2: 2,3,4,5 200  861.50 12.18
##      4) EDUSCHGRD2: 2 5   45.20  7.60 *
##      5) EDUSCHGRD2: 3,4,5 195  708.70 12.30
##       10) EDUSCHGRD2: 3,4 93  288.70 11.72 *
##       11) EDUSCHGRD2: 5 102  360.80 12.82 *
##    3) EDUSCHGRD2: 6,7,8,9,98,99 606 2320.00 14.31
##      6) EDUSCHGRD2: 6,99 291 1138.00 13.73
##       12) YOHGUN2: 1 26  167.50 12.31
##         24) parchkhw: 1 14   27.50 13.50 *
##         25) parchkhw: 2 12   96.92 10.92 *
##       13) YOHGUN2: 2 265  913.10 13.86 *
##      7) EDUSCHGRD2: 7,8,9,98 315  990.00 14.85
##       14) FRDPCIG2: 1 285  695.90 14.99
##         28) PRMJEVR2: 1 163  266.10 15.33 *
##         29) PRMJEVR2: 2 122  384.40 14.52 *
##       15) FRDPCIG2: 2 30  239.40 13.57
##         30) argupar: 1 16   23.44 14.69 *
##         31) argupar: 2 14  172.90 12.29 *
```

```
predict_reg=predict(reg_tree,testing_data_reg)
table(predict_reg,testing_data_reg$iralcage)
```

```
##
## predict_reg         1  3  5  6  7  8  9 10 11 12 13 14 15 16 17
##   7.6               0  1  1  1  0  0  0  1  2  2  0  0  0  0  0
##   10.9166666666667  0  0  0  0  0  0  0  0  0  0  0  1  1  0  0
##   11.7204301075269  0  1  1  2  1  2  6  2  7 10 11  2  0  0  0
##   12.2857142857143  0  0  0  0  0  0  0  0  0  1  0  0  3  0  0
##   12.3076923076923  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0
##   12.8235294117647  0  0  0  0  0  3  0  2  2  7 17 14  5  0  0
##   13.5              0  0  0  0  1  0  0  0  0  1  2  2  0  0  0
##   13.7250859106529  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0
##   13.7816377171216  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0
##   13.8641509433962  1  1  1  1  2  2  0  3  8 10 20 31 47 11  1
##   14.5245901639344  0  0  0  0  0  0  0  1  0  0  8 21 23 13  2
##   14.6875           0  0  0  0  0  0  0  0  0  0  0  0  1  1  0
##   14.8507936507937  0  0  0  0  0  0  0  0  0  0  0  2  0  0  0
##   15.3312883435583  0  0  0  0  0  0  0  0  0  2  6 12 27 21 12
```

```
mean((predict_reg-testing_data_reg$iralcage)^2)
```

```
## [1] 4.508351
```

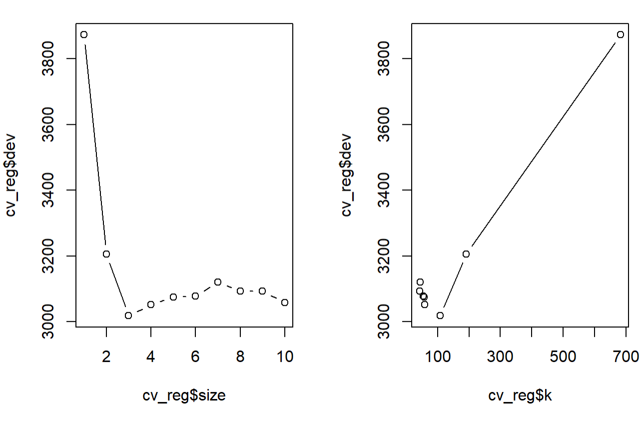
```
cv_reg=cv.tree(reg_tree,FUN=prune.tree)
names(cv_reg)
```

```
## [1] "size"   "dev"    "k"      "method"
```

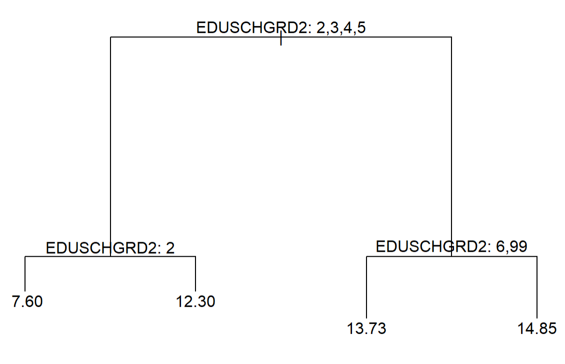
```
cv_reg
```

```
## $size
##  [1] 10  9  8  7  6  5  4  3  2  1
##
## $dev
##  [1] 3058.345 3093.169 3093.169 3120.152 3078.275 3074.757 3051.820 3018.062
##  [9] 3206.152 3873.106
##
## $k
##  [1]      -Inf  43.07202  43.12179  45.40720  54.67678  57.35898  59.19401
##  [8] 107.57128 191.68239 682.37167
##
## $method
## [1] "deviance"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
par(mfrow = c(1, 2))
plot(cv_reg$size, cv_reg$dev, type = "b")
plot(cv_reg$k, cv_reg$dev, type = "b")
```




```
prune_reg <- prune.tree(reg_tree, best = 4)
plot(prune_reg)
text(prune_reg, pretty = 0)
```




```
prune_reg
```

```
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 806 3864.0 13.78
##   2) EDUSCHGRD2: 2,3,4,5 200  861.5 12.18
##     4) EDUSCHGRD2: 2 5   45.2  7.60 *
##     5) EDUSCHGRD2: 3,4,5 195  708.7 12.30 *
##   3) EDUSCHGRD2: 6,7,8,9,98,99 606 2320.0 14.31
##     6) EDUSCHGRD2: 6,99 291 1138.0 13.73 *
##     7) EDUSCHGRD2: 7,8,9,98 315  990.0 14.85 *
```

```
prune_reg_prediction = predict(prune_reg, testing_data_reg)
table(prune_reg_prediction, testing_data_reg$iralcage)
```

```
##
## prune_reg_prediction  1  3  5  6  7  8  9 10 11 12 13 14 15 16 17
##     7.6               0  1  1  1  0  0  0  1  2  2  0  0  0  0  0
##     12.2974358974359  0  1  1  2  1  5  6  4  9 17 28 16  5  0  0
##     13.7250859106529  1  1  1  1  4  2  0  3  9 11 22 34 48 11  1
##     13.7816377171216  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0
##     14.8507936507937  0  0  0  0  0  0  0  1  0  3 14 35 54 35 14
```

```
mean((prune_reg_prediction -testing_data_reg$iralcage)^2)
```

```
## [1] 4.62419
```

```
missing_values=colSums(is.na(training_data_reg))
missing_values
```

```
##      irsex   NEWRACE2    HEALTH2  eduschlgo EDUSCHGRD2  eduskpcom    imother
##          0          0          0          0          0          0          0
##    ifather     income   govtprog   POVERTY3     PDEN10    COUTYP4    schfelt
##          0          0          0          0          0          0          0
##    tchgjob   avggrade   stndscig    stndsmj    stndalc    stnddnk   parchkhw
##          1         38         25         22         21         35          2
##   parhlphw   PRCHORE2   PRLMTTV2   parlmtsn   PRGDJOB2   PRPROUD2    argupar
##          4          1          2          8          3          5          8
##   YOFIGHT2   YOGRPFT2    YOHGUN2    YOSELL2   YOSTOLE2   YOATTAK2   PRPKCIG2
##          5          4          3          3          3          2          3
##   PRMJEVR2     prmjmo   PRALDLY2   YFLPKCG2   YFLTMRJ2    yflmjmo   YFLADLY2
##          6         10          2          5          5          5          3
##   FRDPCIG2   FRDMEVR2   frdmjmon   FRDADLY2   talkprob    PRTALK3   PRBSOLV2
##          3          7          4          5         27          6         12
##   PREVIOL2   PRVDRGO2   GRPCNSL2   PREGPGM2    YTHACT2   DRPRVME3   ANYEDUC3
##          5          7         10          4          2          9          8
##    rlgattd    rlgimpt    rlgdcsn    rlgfrnd   iralcage
##         16         14         13         19          0
```

Bagging

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.3
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
training_data_reg_clean=na.omit(training_data_reg)
bagging_reg <- randomForest(iralcage ~ ., data = training_data_reg_clean,
     mtry = floor(ncol(training_data_reg_clean)/3), importance = TRUE)
bagging_reg
```

```
##
## Call:
##  randomForest(formula = iralcage ~ ., data = training_data_reg_clean,      mtry = floor(ncol(training_data_reg
_clean)/3), importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 20
##
##           Mean of squared residuals: 3.600708
##                     % Var explained: 24.88
```

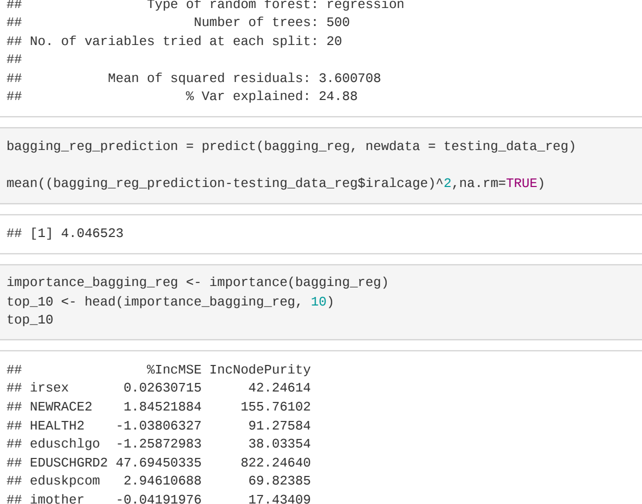
```
bagging_reg_prediction = predict(bagging_reg, newdata = testing_data_reg)

mean((bagging_reg_prediction-testing_data_reg$iralcage)^2,na.rm=TRUE)
```

```
## [1] 4.046523
```

```
importance_bagging_reg <- importance(bagging_reg)
top_10 <- head(importance_bagging_reg, 10)
top_10
```

```
##                %IncMSE IncNodePurity
## irsex       0.02630715      42.24614
## NEWRACE2    1.84521884     155.76102
## HEALTH2    -1.03806327      91.27584
## eduschlgo  -1.25872983      38.03354
## EDUSCHGRD2 47.69450335     822.24640
## eduskpcom   2.94610688      69.82385
## imother    -0.04191976      17.43409
## ifather     2.46447956      50.10733
## income      2.79504931      87.74130
## govtprog    0.50337566      32.70426
```

```
varImpPlot(bagging_reg, n.var = 10, sort = TRUE, main = "Important variables of reg classification(Top 10)")
```



#Random Forest

```
set.seed(1)
randomforest_reg <- randomForest(iralcage ~ ., data = training_data_reg_clean, mtry = floor(ncol(training_data_re
g_clean)/3), importance = TRUE)
randomforest_reg
```

```
##
## Call:
##  randomForest(formula = iralcage ~ ., data = training_data_reg_clean,      mtry = floor(ncol(training_data_reg
_clean)/3), importance = TRUE)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 20
##
##           Mean of squared residuals: 3.566915
##                     % Var explained: 25.59
```

```
yhat_randomforest_reg <- predict(randomforest_reg, newdata = testing_data_reg)
mean((yhat_randomforest_reg -testing_data_reg$iralcage)^2,na.rm=TRUE)
```

```
## [1] 4.023223
```

```
importance_rf_reg <- importance(randomforest_reg)
top_10 <- head(importance_rf_reg, 10)
top_10
```

```
##               %IncMSE IncNodePurity
## irsex       0.3102670      43.59415
## NEWRACE2    0.8901631     150.40315
## HEALTH2    -0.2260177      84.59661
## eduschlgo   1.0020716      39.74949
## EDUSCHGRD2 45.4710646     804.80331
## eduskpcom   2.5629014      66.52381
## imother    -0.2287325      17.98178
## ifather     1.7803244      48.24999
## income      0.2223839      87.64736
## govtprog    0.1920011      30.91443
```

```
varImpPlot(randomforest_reg, n.var = 10, sort = TRUE, main = "Important variables of reg classification(Top 10)")
```