Project Assignment 2

Coder: Vienne Jetten i6361245, Presenter: Edoaurd Dewaerdheijd i6341571

2023-12-11

```
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
```

```
library(magrittr)
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(ggplot2)
library(lattice)
library(C50)
library(rpart)
library(rpart.plot)
```

Data preparation

First we will load the data:

```
student <- read.csv("~/Documents/Uni/BA1/Period 2/Knowledge Discovery and Data Visualization/Project Ass
```

Then we will create a randomized subset. This subset will be created in the same manner as in the previous project assignment.

```
set.seed(123)
my_data <- student[sample(nrow(student), size = 600),]
str(my_data)
```

```
## 'data.frame':    600 obs. of  33 variables:
##  $ school    : Factor w/ 2 levels "GP","MS": 1 2 1 2 1 1 1 1 1 1 ...
##  $ sex       : Factor w/ 2 levels "F","M": 1 1 2 2 2 2 1 1 1 2 ...
##  $ age       : int  20 15 17 17 16 16 17 17 17 15 ...
##  $ address   : Factor w/ 2 levels "R","U": 1 1 1 2 2 2 2 1 2 2 ...
##  $ famsize   : Factor w/ 2 levels "GT3","LE3": 1 1 2 2 1 1 2 2 1 1 ...
##  $ pstatus   : Factor w/ 2 levels "A","T": 2 2 2 2 2 2 2 2 1 2 ...
##  $ medu      : int  1 1 1 4 4 3 2 4 2 4 ...
##  $ fedu      : int  1 1 1 4 3 3 2 4 1 3 ...
##  $ mjob      : Factor w/ 5 levels "at_home","health",..: 3 1 3 3 5 3 4 4 3 5 ...
##  $ fjob      : Factor w/ 5 levels "at_home","health",..: 3 1 4 4 3 4 4 3 3 3 ...
##  $ reason    : Factor w/ 4 levels "course","home",..: 4 1 1 2 2 2 1 3 1 1 ...
##  $ guardian  : Factor w/ 3 levels "father","mother",..: 3 1 2 2 2 1 1 2 2 2 ...
##  $ traveltime: int  2 3 4 1 1 2 1 1 2 2 ...
##  $ studytime : int  3 2 2 3 2 1 4 1 3 2 ...
##  $ failures  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ schoolsup : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ famsup    : Factor w/ 2 levels "no","yes": 1 2 1 2 2 1 1 2 1 2 ...
##  $ paid      : Factor w/ 2 levels "no","yes": 1 1 1 1 2 1 1 1 1 1 ...
##  $ activities: Factor w/ 2 levels "no","yes": 1 1 2 1 2 2 2 1 2 1 ...
##  $ nursery   : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ higher    : Factor w/ 2 levels "no","yes": 2 2 1 2 2 2 2 2 2 2 ...
##  $ internet  : Factor w/ 2 levels "no","yes": 2 1 1 2 2 2 2 1 2 2 ...
##  $ romantic  : Factor w/ 2 levels "no","yes": 2 1 2 1 1 1 2 1 2 1 ...
##  $ famrel    : int  3 4 5 4 3 5 3 5 3 5 ...
##  $ freetime  : int  2 2 3 4 4 4 4 2 2 4 ...
##  $ goout     : int  2 1 5 3 3 2 1 1 3 3 ...
##  $ dalc      : int  1 1 1 1 2 1 1 1 1 1 ...
##  $ walc      : int  3 2 5 2 3 1 1 2 2 2 ...
##  $ health    : int  3 2 5 5 3 5 2 3 3 3 ...
##  $ absences  : int  8 0 0 0 4 6 2 6 0 0 ...
##  $ g1        : int  11 13 8 15 11 14 10 12 15 12 ...
##  $ g2        : int  15 14 8 14 10 14 11 11 15 12 ...
##  $ g3        : int  15 14 8 16 11 15 12 11 16 13 ...
```

We determined that the following variables should be utilized when building any kind of model utilizing the data in the first part of the project assignment. The variables are: school, address, mother's job, father's job, reason, internet access, weekday alcohol consumption, weekend alcohol consumption, first period grade and the second period grade.

Any variables that would not be utilized in the model-building process were eliminated from the data frame:

```
student.2 <- subset(my_data, select = c(1,4,9:11,22,27,28,31,32,33) )
str(student.2) #To check if we got all the variables we want to use.
```

```
## 'data.frame':    600 obs. of  11 variables:
##  $ school  : Factor w/ 2 levels "GP","MS": 1 2 1 2 1 1 1 1 1 1 ...
##  $ address : Factor w/ 2 levels "R","U": 1 1 1 2 2 2 2 1 2 2 ...
##  $ mjob    : Factor w/ 5 levels "at_home","health",..: 3 1 3 3 5 3 4 4 3 5 ...
##  $ fjob    : Factor w/ 5 levels "at_home","health",..: 3 1 4 4 3 4 4 3 3 3 ...
##  $ reason  : Factor w/ 4 levels "course","home",..: 4 1 1 2 2 2 1 3 1 1 ...
##  $ internet: Factor w/ 2 levels "no","yes": 2 1 1 2 2 2 2 1 2 2 ...
##  $ dalc    : int  1 1 1 1 2 1 1 1 1 1 ...
##  $ walc    : int  3 2 5 2 3 1 1 2 2 2 ...
```

```
##  $ g1      : int  11 13 8 15 11 14 10 12 15 12 ...
##  $ g2      : int  15 14 8 14 10 14 11 11 15 12 ...
##  $ g3      : int  15 14 8 16 11 15 12 11 16 13 ...
```

Ignoring the target variable (mean grade) all of the variables, except for the variables g1, g2 and g3, are factor variables, as can be seen by looking at the structure of the resulting data frame. This implies that a decision tree could be a suitable model (this will be implied later on). Additionally, we observe that g1, g2 and g3 have a far wider range than dalc and walc among the variables having numeric levels or values.

Furthermore, we binned the target variable "mean grade":

```
my_data$grade_mean <- rowMeans(my_data[, c("g1", "g2", "g3")])

my_data$pass_fail = ifelse(my_data$grade_mean > 10, "Pass", "Fail")
```

As we established during the project assignment 1, g1, g2 and g3 are highly correlated. This is the reason we decided to get the average of these three variables to establish the students performance for the whole year. Our target variable is "pass_fail". This is a categorical variable describing if a students average yearly grade is over 10 (the passing grade in Portugal). We decided no to use g1, g2 and g3 in any of our models as these variable are extremely highly correlated to our target variables. The conclusion is that we are trying to predict if a student passes or fails based on other factors than grade.

Splitting the data

To provide adequate model testing and prevent overfitting, the data set will be divided into two parts: a training set and a testing set.A 4:1 split of the data was used to guarantee that the training and testing subsets both accurately reflect the population as a whole. The following code was used:

```
train_rows <- sample(nrow(my_data), nrow(my_data) * 0.8)
student_train <- my_data[train_rows, ]
student_test <- my_data[-train_rows, ]
```

Target variable "mean grade" We have to make sure that there are no significant variations in the target variable's proportion in each split.The Z-test is utilized for binary variables:

```
length(student_train$pass_fail[student_train$pass_fail == "Pass"])
```

```
## [1] 335
```

```
length(student_test$pass_fail[student_test$pass_fail == "Pass"])
```

```
## [1] 86
```

The genuine difference in proportions is zero, according to the null hypothesis. It is non-zero, according to the alternative hypothesis.

```
prop.test( x = c(335,86) , n=  c(480,120), alternative = "two.sided", correct = FALSE)
```

```
##
##  2-sample test for equality of proportions without continuity correction
##
## data:  c(335, 86) out of c(480, 120)
## X-squared = 0.16123, df = 1, p-value = 0.688
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.10923491  0.07173491
## sample estimates:
##    prop 1    prop 2
## 0.6979167 0.7166667
```

When the p-value in a statistical test is high, typically above the chosen significance level (commonly 0.05), it suggests that the evidence against the null hypothesis is not strong enough to reject it. In this specific case, with a p-value of 0.688, it means that there isn't sufficient evidence to conclude that the proportions of the two groups significantly differ. This means we cannot reject the null hypothesis, which means the split in the data is validated.

School variable To check for the homogeneity of partition in binary predictor variables, we further employ the Z test:

```
counts_table <- table(student_train$school, student_train$pass_fail)
counts_table
```

```
##
##      Fail Pass
##   GP   63  257
##   MS   82   78
```

```
counts_table2 <- table(student_test$school, student_test$pass_fail)
counts_table2
```

```
##
##      Fail Pass
##   GP   10   59
##   MS   24   27
```

```
prop.test( x = c(257,59) , n=  c(480,120), alternative = "two.sided", correct = FALSE)
```

```
##
##  2-sample test for equality of proportions without continuity correction
##
## data:  c(257, 59) out of c(480, 120)
## X-squared = 0.7371, df = 1, p-value = 0.3906
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.05620768  0.14370768
## sample estimates:
##    prop 1    prop 2
## 0.5354167 0.4916667
```

```r
prop.test( x = c(78,27) , n=  c(480,120), alternative = "two.sided", correct = FALSE)
```

```
##
##  2-sample test for equality of proportions without continuity correction
##
## data:  c(78, 27) out of c(480, 120)
## X-squared = 2.5974, df = 1, p-value = 0.107
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.14417793  0.01917793
## sample estimates:
## prop 1 prop 2
## 0.1625 0.2250
```

Generally, a p-value above 0.05 does not provide strong evidence against the null hypothesis. It implies that the observed differences in proportions between GP and MS schools for both pass and fail categories might not be substantial enough to confidently conclude a true difference in the underlying populations. This means with the p-values 0.3906 (GP) and 0.107 (MS) that we cannot reject the null hypothesis and that the split in the data is validated.

```r
counts_table <- table(student_train$address, student_train$pass_fail)
counts_table
```

Address variable

```
##
##     Fail Pass
##   R   57   90
##   U   88  245
```

```r
counts_table2 <- table(student_test$address, student_test$pass_fail)
counts_table2
```

```
##
##     Fail Pass
##   R   16   22
##   U   18   64
```

```r
prop.test( x = c(245,64) , n=  c(480,120), alternative = "two.sided", correct = FALSE)
```

```
##
##  2-sample test for equality of proportions without continuity correction
##
## data:  c(245, 64) out of c(480, 120)
## X-squared = 0.20185, df = 1, p-value = 0.6532
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.12275334  0.07692001
## sample estimates:
##    prop 1    prop 2
## 0.5104167 0.5333333
```

```
prop.test( x = c(80,22) , n=  c(480,120), alternative = "two.sided", correct = FALSE)
```

```
##
##  2-sample test for equality of proportions without continuity correction
##
## data:  c(80, 22) out of c(480, 120)
## X-squared = 0.18899, df = 1, p-value = 0.6638
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.09350718  0.06017385
## sample estimates:
##    prop 1    prop 2
## 0.1666667 0.1833333
```

The p-values obtained are 0.6532 (Urban area) and 0.6638 (Rural area). These p-values suggest that there isn't sufficient evidence to reject the null hypothesis, indicating that the observed differences in proportions between the groups (R and U) in both the "Pass" category and the "Fail" could plausibly occur due to random chance or sampling variability. This means that the split is validated.

```
counts_table <- table(student_train$internet, student_train$pass_fail)
counts_table
```

Internet access variable

```
##
##       Fail Pass
##   no    44   64
##   yes  101  271
```

```
counts_table2 <- table(student_test$internet, student_test$pass_fail)
counts_table2
```

```
##
##       Fail Pass
##   no    18   16
##   yes   16   70
```

```
prop.test( x = c(64,16) , n=  c(480,120), alternative = "two.sided", correct = FALSE)
```

```
##
##  2-sample test for equality of proportions without continuity correction
##
## data:  c(64, 16) out of c(480, 120)
## X-squared = 0, df = 1, p-value = 1
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.06799984  0.06799984
## sample estimates:
##    prop 1    prop 2
## 0.1333333 0.1333333
```

```r
prop.test( x = c(271,70) , n=  c(480,120), alternative = "two.sided", correct = FALSE)
```

```
##
##  2-sample test for equality of proportions without continuity correction
##
## data:  c(271, 70) out of c(480, 120)
## X-squared = 0.13757, df = 1, p-value = 0.7107
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.11748253  0.07998253
## sample estimates:
##    prop 1    prop 2
## 0.5645833 0.5833333
```

In both cases, the p-values 1 (no) and 0.7107 (yes) are higher than the common significance level of 0.05, indicating that there isn't enough evidence to reject the null hypothesis. Again, this means that the split in data is validated.

Mother's job variable In multi-class variables, the homogeneity of the class distribution is assessed using the Chi-Square test. The null hypothesis in the Chi-Square test for homogeneity of proportion states that each class's proportion in the training and testing sets is equal, whereas the alternative hypothesis contends that at least one class's proportion differs.

The code to test this for the training and testing sets is as follows:

```r
ctable.1.mjob <- table(student_test$mjob)
ctable.2.mjob <- table(student_train$mjob)

ctable.mjob <- t(cbind(ctable.1.mjob, ctable.2.mjob))

chisq.test(ctable.mjob)
```

```
##
##  Pearson's Chi-squared test
##
## data:  ctable.mjob
## X-squared = 11.555, df = 4, p-value = 0.02099
```

With a p-value of 0.02099, we reject the null hypothesis. This implies that the split in data for mjob is not validated.

```r
ctable.1.fjob <- table(student_test$fjob)
ctable.2.fjob <- table(student_train$fjob)

ctable.fjob <- t(cbind(ctable.1.fjob, ctable.2.fjob))

chisq.test(ctable.fjob)
```

Father's job variable

```
## Warning in chisq.test(ctable.fjob): Chi-squared approximation may be incorrect

##
##  Pearson's Chi-squared test
##
## data:  ctable.fjob
## X-squared = 10.618, df = 4, p-value = 0.03121
```

The Chi-squared test was performed using the chisq.test function in R on the variable fjob. The test resulted in a Pearson's Chi-squared statistic of 10.618 with an associated p-value of 0.03121.

This result indicates that there is evidence to reject the null hypothesis, suggesting that the split is also not validated.

```
ctable.1.reason <- table(student_test$reason)
ctable.2.reason <- table(student_train$reason)

ctable.reason <- t(cbind(ctable.1.reason, ctable.2.reason))

chisq.test(ctable.reason)
```

Reason variable

```
##
##  Pearson's Chi-squared test
##
## data:  ctable.reason
## X-squared = 2.1923, df = 3, p-value = 0.5335
```

The large p-value of 0.5335 signifies that we can reject the null hypothesis and that the split in the data is validated.

```
ctable.1.dalc <- table(student_test$dalc)
ctable.2.dalc <- table(student_train$dalc)

ctable.dalc <- t(cbind(ctable.1.dalc, ctable.2.dalc))

chisq.test(ctable.dalc, simulate.p.value = TRUE)
```

Weekday alcohol consumption

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  ctable.dalc
## X-squared = 2.532, df = NA, p-value = 0.6282
```

The high p-value of 0.6282 implies that we cannot reject the null hypothesis and is again indicating that the split between the data is validated.

```r
ctable.1.walc <- table(student_test$walc)
ctable.2.walc <- table(student_train$walc)

ctable.walc <- t(cbind(ctable.1.walc, ctable.2.walc))

chisq.test(ctable.walc)
```

Weekend alcohol consumption

```
##
##  Pearson's Chi-squared test
##
## data:  ctable.walc
## X-squared = 1.0075, df = 4, p-value = 0.9087
```

The extremely high p-value 0.9087, signifies that we cannot reject null hypothesis, meaning that the split is validated.

First period grade In the case of continuous data, the similarity of the distribution between the two subsets is checked using a t-test for the difference in means.

```r
t.test(student_train$g1, student_test$g1)
```

```
##
##  Welch Two Sample t-test
##
## data:  student_train$g1 and student_test$g1
## t = 1.4227, df = 198.44, p-value = 0.1564
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.1456007  0.8997673
## sample estimates:
## mean of x mean of y
##  11.53542  11.15833
```

With a p-value of 0.1564, there isn't sufficient evidence to reject the null hypothesis. This means that based on this test, we fail to find a significant difference between the means of g1 in student_train and student_test, so the split is validated. The confidence interval also encompasses 0, further supporting this idea that the true difference in means might very well be zero.

```r
t.test(student_train$g2, student_test$g2)
```

Second period grade

```
##
##  Welch Two Sample t-test
```

```
##
## data:  student_train$g2 and student_test$g2
## t = 1.4688, df = 193.96, p-value = 0.1435
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.1442563  0.9859230
## sample estimates:
## mean of x mean of y
##  11.71250  11.29167
```

With a p-value of 0.1435, there isn't enough evidence to reject the null hypothesis. This suggests that, based on this statistical test, we do not find a significant difference between the means of g2 in student_train and student_test, so this means the split is validated. The confidence interval also encompasses 0, indicating that the true difference in means could plausibly be zero.

```
t.test(student_train$g3, student_test$g3)
```

Third period grade

```
##
##  Welch Two Sample t-test
##
## data:  student_train$g3 and student_test$g3
## t = 1.1312, df = 208.4, p-value = 0.2593
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2553451  0.9428451
## sample estimates:
## mean of x mean of y
##  12.01875  11.67500
```

With a p-value of 0.2593, there isn't enough evidence to reject the null hypothesis. This suggests that, based on this statistical test, we do not find a significant difference between the means of g3 in student_train and student_test, so this means the split is validated. The confidence interval also encompasses 0, indicating that the true difference in means could plausibly be zero.

Model Building

C5.0 decision tree

In a decision tree model, the instance space is divided into axis parallel divisions. One variable is considered at a time, and judgments are made by following a certain "path" where decisions are made at each "fork" depending on an observation's given attribute. An technique called C5.0 builds a tree model based on information gain, or the reduction in bits required for information transfer. The tree was built using the code below:

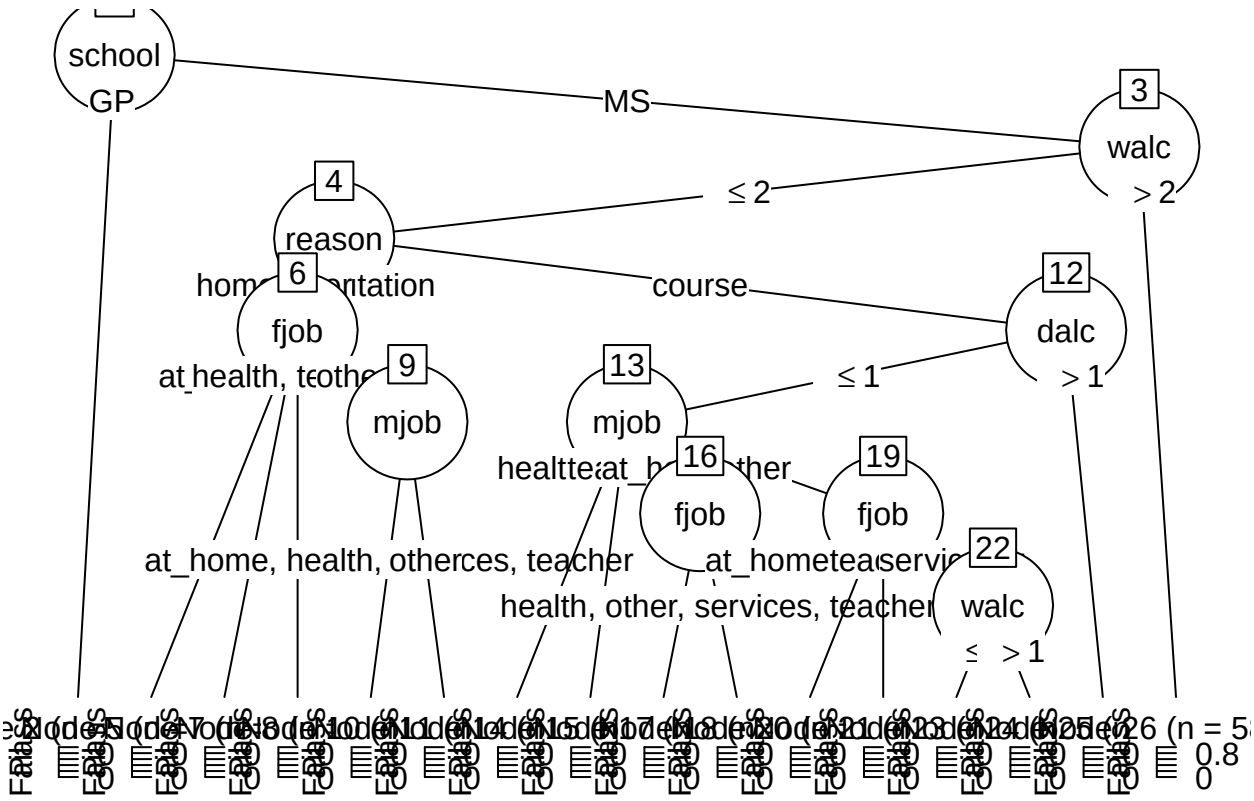
```
student_train2 <- subset(student_train, select = c(1,4,9,10,11,22,27,28,31,32,33,35) )
student_test2 <- subset(student_test, select = c(1,4,9,10,11,22,27,28,31,32,33,35) )
predictors <- subset(student_train2, select = -c(9,10,11,12))
```

```r
dependant <- as.factor(student_train$pass_fail)
c50fit <- C5.0(x = predictors, y = dependant)
summary(c50fit)
```

```
##
## Call:
## C5.0.default(x = predictors, y = dependant)
##
##
## C5.0 [Release 2.07 GPL Edition]      Wed Dec 13 00:53:22 2023
## -------------------------------
##
## Class specified by attribute 'outcome'
##
## Read 480 cases (9 attributes) from undefined.data
##
## Decision tree:
##
## school = GP: Pass (320/63)
## school = MS:
## :...walc > 2: Fail (58/17)
##     walc <= 2:
##     :...reason in {home,reputation}: Pass (29/6)
##         reason = other:
##         :...fjob in {at_home,services}: Pass (7/1)
##         :   fjob in {health,teacher}: Fail (2)
##         :   fjob = other:
##         :   :...mjob in {at_home,health,services,teacher}: Fail (5)
##         :       mjob = other: Pass (6/1)
##         reason = course:
##         :...dalc > 1: Pass (6/1)
##             dalc <= 1:
##             :...mjob in {health,services}: Fail (5/1)
##                 mjob = teacher: Pass (2)
##                 mjob = at_home:
##                 :...fjob = at_home: Pass (5/1)
##                 :   fjob in {health,other,services,teacher}: Fail (13/4)
##                 mjob = other:
##                 :...fjob in {at_home,health,other}: Fail (12/4)
##                     fjob = teacher: Pass (1)
##                     fjob = services:
##                     :...walc <= 1: Pass (5)
##                         walc > 1: Fail (4/1)
##
##
## Evaluation on training data (480 cases):
##
##      Decision Tree
##    ----------------
##    Size      Errors
##
##      16  100(20.8%)   <<
##
```

```
##
##     (a)   (b)    <-classified as
##    ----  ----
##      72    73    (a): class Fail
##      27   308    (b): class Pass
##
##
##  Attribute usage:
##
##  100.00% school
##   33.33% walc
##   21.25% reason
##   12.50% fjob
##   12.08% mjob
##   11.04% dalc
##
##
## Time: 0.0 secs
```

```
plot(c50fit)
```



Observe that the data does not need to be normalized since the decision tree is not a "mathematical" model, which means that it looks at the attribute distribution to produce a rule that is to be followed rather than using the actual attribute values to make a forecast.

We can also view the tree as a set of rules:

```r
c50fit.asrules <- C5.0(x = predictors, y = dependant, rules = TRUE)
summary(c50fit.asrules)
```

```
##
## Call:
## C5.0.default(x = predictors, y = dependant, rules = TRUE)
##
##
## C5.0 [Release 2.07 GPL Edition]      Wed Dec 13 00:53:23 2023
## -------------------------------
##
## Class specified by attribute 'outcome'
##
## Read 480 cases (9 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (5, lift 2.8)
##  school = MS
##  mjob in {at_home, teacher}
##  fjob = other
##  reason = other
##  walc <= 2
##  ->  class Fail  [0.857]
##
## Rule 2: (7/1, lift 2.6)
##  school = MS
##  mjob in {health, services}
##  reason = course
##  dalc <= 1
##  ->  class Fail  [0.778]
##
## Rule 3: (23/6, lift 2.4)
##  school = MS
##  reason = course
##  dalc <= 1
##  walc > 1
##  ->  class Fail  [0.720]
##
## Rule 4: (15/4, lift 2.3)
##  school = MS
##  mjob = at_home
##  fjob in {other, services}
##  reason = course
##  dalc <= 1
##  ->  class Fail  [0.706]
##
## Rule 5: (160/78, lift 1.7)
##  school = MS
##  ->  class Fail  [0.512]
##
## Rule 6: (21/2, lift 1.2)
##  fjob = teacher
```

```
##  dalc <= 1
##  ->  class Pass  [0.870]
##
## Rule 7: (320/63, lift 1.1)
##  school = GP
##  ->  class Pass  [0.801]
##
## Rule 8: (296/68, lift 1.1)
##  walc <= 2
##  ->  class Pass  [0.768]
##
## Default class: Pass
##
##
## Evaluation on training data (480 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##       8  104(21.7%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##      65    80    (a): class Fail
##      24   311    (b): class Pass
##
##
##  Attribute usage:
##
##  100.00% school
##   63.54% walc
##   11.88% dalc
##    8.54% fjob
##    8.54% reason
##    5.62% mjob
##
##
## Time: 0.0 secs
```

Rule Interpretations: The model generated several rules to predict the outcome ("Fail" or "Pass") based on different combinations of attributes. For instance, Rule 1 indicates that if the student attends the MS school, has a mother working at home or is a teacher, has a father with another occupation, the reason for choosing the school is other, and the weekend alcohol consumption is low (walc <= 2), they are predicted to "Fail" with a confidence of 85.7%.

Model Performance: The model was trained on 480 cases and produced 8 rules. However, it made errors in 104 cases, resulting in an error rate of 21.7%. Out of the 480 cases, 65 were incorrectly classified as "Fail," while 80 were incorrectly classified as "Pass."

Attribute Importance: Attributes like school (100%), followed by weekend alcohol consumption (walc) at 63.54%, and daily alcohol consumption (dalc) at 11.88%, were most frequently used in constructing the rules, indicating their significance in predicting the outcome. Other attributes like father's job (fjob), reason for choosing the school (reason), and mother's job (mjob) also contributed to the model but to a lesser extent.

Default Prediction: If none of the specified rules apply to an instance, the model defaults to predicting "Pass." This analysis suggests that the model heavily relies on the school attribute followed by alcohol consumption metrics and certain parental occupation and reason variables to predict whether a student will "Fail" or "Pass." However, despite its accuracy in many cases, it still has a considerable error rate, indicating areas where it struggles to accurately predict the outcome.

Accuracy:

```
studentpredict.c50 <- predict(c50fit, student_test2, type= "class")
confusionMatrix( factor(studentpredict.c50), factor(student_test2$pass_fail), mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Fail Pass
##       Fail   16   20
##       Pass   18   66
##
##                Accuracy : 0.6833
##                  95% CI : (0.5922, 0.7652)
##     No Information Rate : 0.7167
##     P-Value [Acc > NIR] : 0.8197
##
##                   Kappa : 0.2339
##
##  Mcnemar's Test P-Value : 0.8711
##
##             Sensitivity : 0.4706
##             Specificity : 0.7674
##          Pos Pred Value : 0.4444
##          Neg Pred Value : 0.7857
##               Precision : 0.4444
##                  Recall : 0.4706
##                      F1 : 0.4571
##              Prevalence : 0.2833
##          Detection Rate : 0.1333
##    Detection Prevalence : 0.3000
##       Balanced Accuracy : 0.6190
##
##        'Positive' Class : Fail
##
```

Model Evaluation Metrics:

Accuracy and Baseline Comparison:

Accuracy: The model demonstrates a 68.33% accuracy on the test set. This is slightly below the baseline model's accuracy of 71.67%, which predicts the majority class "Pass" without considering the nuances of the data.

Agreement and Statistical Significance:

Kappa: The kappa value of 0.2339 indicates a fair agreement between predicted classes and actual classes. A higher value closer to 1 would suggest stronger agreement. Mcnemar's Test: The p-value of 0.8711 suggests no significant difference between the model's False Positive and False Negative rates.

Performance on Classifications:

Sensitivity (Recall): The model identifies 47.06% of "Fail" instances correctly, indicating its ability to capture True Positives. Specificity: The model's ability to identify True Negatives stands at 76.74%.

Predictive Values and Precision:

Positive/Negative Predictive Value: The positive predictive value (precision) and negative predictive value are 44.44% and 78.57% respectively. These values indicate the accuracy of the model's positive/negative predictions.

Combined Metrics:

F1 Score: At 45.71%, the F1 score combines precision and recall into a single metric, providing an overall performance measure. Balanced Accuracy: This metric, at 61.90%, takes into account the uneven distribution of target classes, offering a balanced view by considering sensitivity and specificity equally. Observational Insights:

Prevalence: Within the subset, 28.33% of observations belong to the "positive" class (students who fail). Detection Rate/Prevalence: The model achieves a 13.33% detection rate for True Negatives and predicts positives in 30.00% of all predictions made.

All of the model metrics that have been previously presented are helpful, but mostly only in comparison to other models, which will be done soon.

CART decision tree

A CART decision tree was generated using the code below:

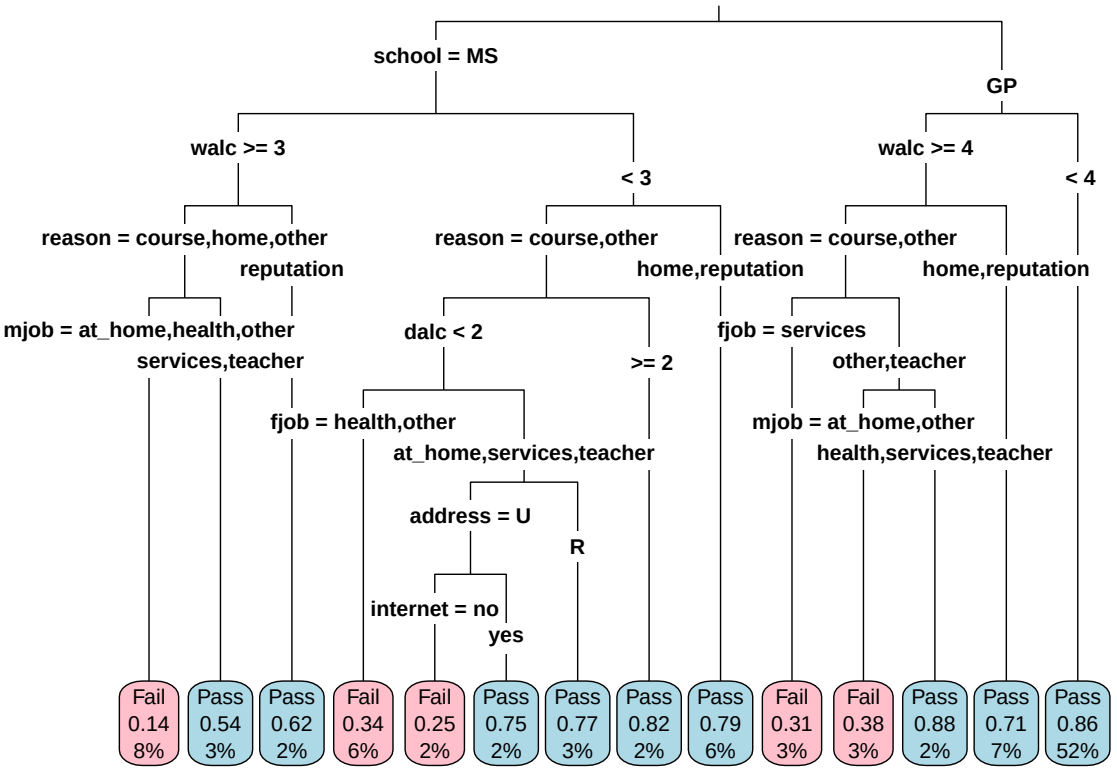
```
cartfit <- rpart(pass_fail~. - g1 - g2 - g3 , data = student_train2, method = "class", control = rpart.

print(cartfit)
```

```
## n= 480
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 480 145 Pass (0.3020833 0.6979167)
##     2) school=MS 160  78 Fail (0.5125000 0.4875000)
##       4) walc>=2.5 58  17 Fail (0.7068966 0.2931034)
##         8) reason=course,home,other 50  12 Fail (0.7600000 0.2400000)
##          16) mjob=at_home,health,other 37   5 Fail (0.8648649 0.1351351) *
##          17) mjob=services,teacher 13   6 Pass (0.4615385 0.5384615) *
##         9) reason=reputation 8   3 Pass (0.3750000 0.6250000) *
##       5) walc< 2.5 102  41 Pass (0.4019608 0.5980392)
##        10) reason=course,other 73  35 Pass (0.4794521 0.5205479)
##          20) dalc< 1.5 62  29 Fail (0.5322581 0.4677419)
##            40) fjob=health,other 29  10 Fail (0.6551724 0.3448276) *
##            41) fjob=at_home,services,teacher 33  14 Pass (0.4242424 0.5757576)
##              82) address=U 20   9 Fail (0.5500000 0.4500000)
##               164) internet=no 12   3 Fail (0.7500000 0.2500000) *
##               165) internet=yes 8   2 Pass (0.2500000 0.7500000) *
##              83) address=R 13   3 Pass (0.2307692 0.7692308) *
##          21) dalc>=1.5 11   2 Pass (0.1818182 0.8181818) *
##        11) reason=home,reputation 29   6 Pass (0.2068966 0.7931034) *
##     3) school=GP 320  63 Pass (0.1968750 0.8031250)
##       6) walc>=3.5 68  28 Pass (0.4117647 0.5882353)
```

```
##        12) reason=course,other 34  16 Fail (0.5294118 0.4705882)
##          24) fjob=services 13   4 Fail (0.6923077 0.3076923) *
##          25) fjob=other,teacher 21   9 Pass (0.4285714 0.5714286)
##            50) mjob=at_home,other 13   5 Fail (0.6153846 0.3846154) *
##            51) mjob=health,services,teacher 8   1 Pass (0.1250000 0.8750000) *
##        13) reason=home,reputation 34  10 Pass (0.2941176 0.7058824) *
##       7) walc< 3.5 252  35 Pass (0.1388889 0.8611111) *
```

```
rpart.plot(cartfit, type = 3, box.palette = c("pink", "lightblue"), fallen.leaves = TRUE)
```




```
printcp(cartfit)
```

```
##
## Classification tree:
## rpart(formula = pass_fail ~ . - g1 - g2 - g3, data = student_train2,
##     method = "class", control = rpart.control(cp = 0))
##
## Variables actually used in tree construction:
## [1] address  dalc     fjob     internet mjob     reason   school   walc
##
## Root node error: 145/480 = 0.30208
##
## n= 480
##
##          CP nsplit rel error  xerror     xstd
```

```
## 1 0.0827586      0   1.00000 1.00000 0.069377
## 2 0.0206897      2   0.83448 0.83448 0.065607
## 3 0.0137931      7   0.73103 0.90345 0.067307
## 4 0.0068966     12   0.66207 0.91034 0.067466
## 5 0.0000000     13   0.65517 0.92414 0.067780
```

Despite sharing the same structure, the model differs significantly from the C5.0 decision tree. This is because the CART algorithm determines which attribute to split based on distinct selection criteria. Additionally, it manages pruning very differently from C5.0.

As we can see, the algorithm resulted in a tree that uses a the variables; school, reason, mjob, fjob, walc, dalc, address and internet. The model is analysed more closely below:

Accuracy:

```
studentpredict.CART <- predict(cartfit, student_test2, type = "class")
confusionMatrix(factor(studentpredict.CART),factor(student_test2$pass_fail), mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Fail Pass
##       Fail   14   19
##       Pass   20   67
##
##                Accuracy : 0.675
##                  95% CI : (0.5835, 0.7577)
##     No Information Rate : 0.7167
##     P-Value [Acc > NIR] : 0.8668
##
##                   Kappa : 0.1925
##
##  Mcnemar's Test P-Value : 1.0000
##
##             Sensitivity : 0.4118
##             Specificity : 0.7791
##          Pos Pred Value : 0.4242
##          Neg Pred Value : 0.7701
##               Precision : 0.4242
##                  Recall : 0.4118
##                      F1 : 0.4179
##              Prevalence : 0.2833
##          Detection Rate : 0.1167
##    Detection Prevalence : 0.2750
##       Balanced Accuracy : 0.5954
##
##        'Positive' Class : Fail
##
```

Model Evaluation Metrics:

Accuracy and Comparative Performance:

Accuracy: The model achieves an accuracy of 67.5% on the test set. Comparatively, this is slightly lower than the baseline accuracy of 71.67%. Sensitivity and Specificity: While the sensitivity for identifying "Fail" instances is 41.18%, the specificity for correctly identifying "Pass" instances is 77.91%.

Precision and F1 Score:

Precision: The precision of predicting "Fail" instances stands at 42.42%, suggesting the proportion of predicted "Fail" instances that were actually correct. F1 Score: At 41.79%, the F1 score combines precision and recall, providing an overall performance measure.

Agreement Metrics:

Kappa: The kappa value is 0.1925, indicating slight agreement between predicted and actual classes. Balanced Accuracy: This metric, at 59.54%, considers the balance between sensitivity and specificity, particularly useful for unevenly distributed target classes.

Observational Insights:

Prevalence: Within the subset, 28.33% of observations belong to the "positive" class (students who fail). Detection Rate/Prevalence: The model achieves a 11.67% detection rate for True Negatives and predicts positives in 27.50% of all predictions made.

Regression model

Regression Model Explanation:

The lm function in R is utilized for linear regression modeling, aiming to predict the grade_mean based on various predictors while excluding g1, g2, and g3. The stepwise selection method refines the model by iteratively adding or removing predictors to achieve an optimized model based on statistical criteria.

Model Construction and Variable Selection:

Linear Regression: This modeling technique seeks to establish a linear relationship between the dependent variable grade_mean and the chosen independent variables. Stepwise Selection: This approach, observed through step(), systematically assesses predictors, progressively adding or removing them based on statistical significance, aiming for an optimal model. Attribute Influence and Weighting:

Feature Vectors and Predictors: The model uses various attributes or predictors to estimate the grade_mean numerically. These attributes collectively contribute to the predictive capacity of the model. Positioning on the Grid: Similar to K Nearest Neighbors, each observation is positioned within a 'grid' represented by the feature space created by the selected predictors.

Decision-making and Model Building:

Influence of Nearest Instances: Unlike KNN, this model doesn't base predictions on nearest neighbors but rather estimates the grade_mean based on the linear relationship between predictors and the target variable.

Weighting and Variable Importance: The algorithm assigns coefficients to each predictor, determining their individual contribution to predicting grade_mean.

First we are going to deal with the binary categorical variables and with multiple class variables.

The binary categorical variables where changed into 1 and 0. For school "GP" is 1 and "MS" is 0, this is because the pass rate is higher for the "GP" school. This is the exact same what we did with the address variable and the internet variable. For address "U" (urban area) is changed into 1 and "R" (rural area) into 0 and for internet it's "yes" what will be turned into 1 and "no" into 0.

```
student_train.kknn.notnormalized <- student_train2

#binary categorical variables

student_train.kknn.notnormalized <- student_train.kknn.notnormalized %>% mutate(
  school = ifelse(student_train.kknn.notnormalized$school == "GP", 1, 0),
  address = ifelse(student_train.kknn.notnormalized$address == "U", 1, 0),
```

```
  internet = ifelse(student_train.kknn.notnormalized$internet == "yes", 1, 0))

#multiple class categorical variables

student_train_knn <- student_train.kknn.notnormalized %>% mutate(
  fjob.at_home = ifelse(student_train.kknn.notnormalized$fjob == "at_home", 1, 0) ,
  fjob.services = ifelse(student_train.kknn.notnormalized$fjob == "services", 1, 0),
  fjob.other = ifelse(student_train.kknn.notnormalized$fjob == "other", 1, 0),
  fjob.teacher = ifelse(student_train.kknn.notnormalized$fjob == "teacher", 1, 0),
  fjob.health = ifelse(student_train.kknn.notnormalized$fjob == "health", 1, 0),
  mjob.at_home = ifelse(student_train.kknn.notnormalized$mjob == "at_home", 1, 0) ,
  mjob.services = ifelse(student_train.kknn.notnormalized$mjob == "services", 1, 0),
  mjob.other = ifelse(student_train.kknn.notnormalized$mjob == "other", 1, 0),
  mjob.teacher = ifelse(student_train.kknn.notnormalized$mjob == "teacher", 1, 0),
  mjob.health = ifelse(student_train.kknn.notnormalized$mjob == "health", 1, 0),
  reason.home = ifelse(student_train.kknn.notnormalized$reason == "home", 1, 0),
  reason.course = ifelse(student_train.kknn.notnormalized$reason == "course", 1, 0),
  reason.reputation = ifelse(student_train.kknn.notnormalized$reason == "reputation", 1, 0),
  reason.other = ifelse(student_train.kknn.notnormalized$reason == "other", 1, 0),
  pass_fail = ifelse(student_train.kknn.notnormalized$pass_fail == "Pass", 1, 0),
  grade_mean = student_train$grade_mean)

student_train_knn <- subset(student_train_knn, select = -c(3:5))

str(student_train_knn)
```

```
## 'data.frame':    480 obs. of  24 variables:
##  $ school           : num  1 1 1 1 1 1 0 0 1 1 ...
##  $ address          : num  1 1 0 1 1 0 1 0 1 1 ...
##  $ internet         : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ dalc             : int  4 1 2 2 2 2 3 1 2 1 ...
##  $ walc             : int  5 2 2 3 2 2 1 1 3 1 ...
##  $ g1               : int  9 12 11 12 12 8 18 14 14 12 ...
##  $ g2               : int  10 12 11 13 15 8 18 14 13 10 ...
##  $ g3               : int  11 13 10 12 15 9 18 15 14 11 ...
##  $ pass_fail        : num  0 1 1 1 1 0 1 1 1 1 ...
##  $ fjob.at_home     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ fjob.services    : num  0 0 1 0 0 0 0 0 0 0 ...
##  $ fjob.other       : num  1 1 0 1 1 1 1 1 1 1 ...
##  $ fjob.teacher     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ fjob.health      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ mjob.at_home     : num  1 0 0 1 0 1 0 0 0 0 ...
##  $ mjob.services    : num  0 1 1 0 1 0 0 1 0 1 ...
##  $ mjob.other       : num  0 0 0 0 0 0 0 0 1 0 ...
##  $ mjob.teacher     : num  0 0 0 0 0 0 1 0 0 0 ...
##  $ mjob.health      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ reason.home      : num  0 0 0 0 0 0 0 1 0 0 ...
##  $ reason.course    : num  1 0 1 0 1 1 1 0 0 0 ...
##  $ reason.reputation: num  0 1 0 0 0 0 0 0 1 1 ...
##  $ reason.other     : num  0 0 0 1 0 0 0 0 0 0 ...
##  $ grade_mean       : num  10 12.3 10.7 12.3 14 ...
```

The exact same process was repeated for the testing subset;

```r
student_test.kknn.notnormalized <- student_test2

student_test.kknn.notnormalized <- student_test.kknn.notnormalized %>% mutate(
  school = ifelse(student_test.kknn.notnormalized$school == "GP", 1, 0),
  address = ifelse(student_test.kknn.notnormalized$address == "U", 1, 0),
  internet = ifelse(student_test.kknn.notnormalized$internet == "yes", 1, 0))


student_test_knn <- student_test.kknn.notnormalized %>% mutate(
  fjob.at_home = ifelse(student_test.kknn.notnormalized$fjob == "at_home", 1, 0) ,
  fjob.services = ifelse(student_test.kknn.notnormalized$fjob == "services", 1, 0),
  fjob.other = ifelse(student_test.kknn.notnormalized$fjob == "other", 1, 0),
  fjob.teacher = ifelse(student_test.kknn.notnormalized$fjob == "teacher", 1, 0),
  fjob.health = ifelse(student_test.kknn.notnormalized$fjob == "health", 1, 0),
  mjob.at_home = ifelse(student_test.kknn.notnormalized$mjob == "at_home", 1, 0) ,
  mjob.services = ifelse(student_test.kknn.notnormalized$mjob == "services", 1, 0),
  mjob.other = ifelse(student_test.kknn.notnormalized$mjob == "other", 1, 0),
  mjob.teacher = ifelse(student_test.kknn.notnormalized$mjob == "teacher", 1, 0),
  mjob.health = ifelse(student_test.kknn.notnormalized$mjob == "health", 1, 0),
  reason.home = ifelse(student_test.kknn.notnormalized$reason == "home", 1, 0),
  reason.course = ifelse(student_test.kknn.notnormalized$reason == "course", 1, 0),
  reason.reputation = ifelse(student_test.kknn.notnormalized$reason == "reputation", 1, 0),
  reason.other = ifelse(student_test.kknn.notnormalized$reason == "other", 1, 0),
  pass_fail= ifelse(student_test.kknn.notnormalized$pass_fail == "Pass", 1, 0))

str(student_test_knn)
```

```
## 'data.frame':    120 obs. of  26 variables:
##  $ school           : num  0 0 0 1 0 1 0 1 1 1 ...
##  $ address          : num  1 0 1 1 0 1 1 0 1 1 ...
##  $ mjob             : Factor w/ 5 levels "at_home","health",..: 4 1 1 3 4 3 4 4 3 2 ...
##  $ fjob             : Factor w/ 5 levels "at_home","health",..: 4 4 3 3 4 3 4 2 3 4 ...
##  $ reason           : Factor w/ 4 levels "course","home",..: 1 3 1 1 1 1 3 1 1 1 ...
##  $ internet         : num  1 0 1 1 1 1 0 1 1 1 ...
##  $ dalc             : int  1 1 1 1 1 1 2 1 1 1 ...
##  $ walc             : int  4 1 1 3 2 1 2 1 1 3 ...
##  $ g1               : int  11 10 7 13 7 15 10 11 13 12 ...
##  $ g2               : int  12 10 10 12 6 16 9 12 12 13 ...
##  $ g3               : int  13 10 10 12 8 16 10 12 12 12 ...
##  $ pass_fail        : num  1 0 0 1 0 1 0 1 1 1 ...
##  $ fjob.at_home     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ fjob.services    : num  1 1 0 0 1 0 1 0 0 1 ...
##  $ fjob.other       : num  0 0 1 1 0 1 0 0 1 0 ...
##  $ fjob.teacher     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ fjob.health      : num  0 0 0 0 0 0 0 1 0 0 ...
##  $ mjob.at_home     : num  0 1 1 0 0 0 0 0 0 0 ...
##  $ mjob.services    : num  1 0 0 0 1 0 1 1 0 0 ...
##  $ mjob.other       : num  0 0 0 1 0 1 0 0 1 0 ...
##  $ mjob.teacher     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ mjob.health      : num  0 0 0 0 0 0 0 0 0 1 ...
##  $ reason.home      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ reason.course    : num  1 0 1 1 1 1 0 1 1 1 ...
##  $ reason.reputation: num  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ reason.other     : num  0 1 0 0 0 0 1 0 0 0 ...
```

The data is not normalized. This is because of the insensitivity to the predictor scale. Linear regression models built using the lm() function are generally insensitive to the scale of predictors. The coefficients in linear regression represent the change in the dependent variable for a one-unit change in the predictor. Therefore, scaling predictors do not significantly impact the interpretation of coefficients in this context. Also because of the step wise selection process. The step wise selection method implemented by step() aims to select the most relevant predictors for the model. If the variables selected through this automated process have varying scales but contribute significantly to the model's predictive power, normalization is not crucial.

The code used to make a regression model is seen below.

```
regression_model <- step(lm(grade_mean ~ . - g1 - g2 - g3, data = student_train_knn))
```

```
## Start:  AIC=644.63
## grade_mean ~ (school + address + internet + dalc + walc + g1 +
##     g2 + g3 + pass_fail + fjob.at_home + fjob.services + fjob.other +
##     fjob.teacher + fjob.health + mjob.at_home + mjob.services +
##     mjob.other + mjob.teacher + mjob.health + reason.home + reason.course +
##     reason.reputation + reason.other) - g1 - g2 - g3
##
##
## Step:  AIC=644.63
## grade_mean ~ school + address + internet + dalc + walc + pass_fail +
##     fjob.at_home + fjob.services + fjob.other + fjob.teacher +
##     fjob.health + mjob.at_home + mjob.services + mjob.other +
##     mjob.teacher + mjob.health + reason.home + reason.course +
##     reason.reputation
##
##
## Step:  AIC=644.63
## grade_mean ~ school + address + internet + dalc + walc + pass_fail +
##     fjob.at_home + fjob.services + fjob.other + fjob.teacher +
##     fjob.health + mjob.at_home + mjob.services + mjob.other +
##     mjob.teacher + reason.home + reason.course + reason.reputation
##
##
## Step:  AIC=644.63
## grade_mean ~ school + address + internet + dalc + walc + pass_fail +
##     fjob.at_home + fjob.services + fjob.other + fjob.teacher +
##     mjob.at_home + mjob.services + mjob.other + mjob.teacher +
##     reason.home + reason.course + reason.reputation
##
##                     Df Sum of Sq    RSS    AIC
## - reason.home        1      0.24 1706.0 642.70
## - fjob.services      1      0.28 1706.1 642.71
## - walc               1      0.36 1706.1 642.73
## - internet           1      0.97 1706.7 642.91
## - fjob.other         1      1.20 1707.0 642.97
## - reason.course      1      1.28 1707.0 642.99
## - school             1      1.78 1707.5 643.13
## - fjob.teacher       1      1.89 1707.7 643.16
## - mjob.teacher       1      2.65 1708.4 643.38
## - fjob.at_home       1      4.81 1710.6 643.99
```

```
## <none>                           1705.8 644.63
## - dalc               1      8.64 1714.4 645.06
## - mjob.at_home       1     11.46 1717.2 645.85
## - mjob.other         1     11.98 1717.7 645.99
## - reason.reputation  1     12.16 1717.9 646.04
## - address            1     21.15 1726.9 648.55
## - mjob.services      1     21.32 1727.1 648.60
## - pass_fail          1   1531.90 3237.7 950.24
##
## Step:  AIC=642.7
## grade_mean ~ school + address + internet + dalc + walc + pass_fail +
##     fjob.at_home + fjob.services + fjob.other + fjob.teacher +
##     mjob.at_home + mjob.services + mjob.other + mjob.teacher +
##     reason.course + reason.reputation
##
##                     Df Sum of Sq    RSS    AIC
## - fjob.services      1      0.26 1706.3 640.77
## - walc               1      0.36 1706.4 640.80
## - internet           1      1.00 1707.0 640.98
## - fjob.other         1      1.15 1707.2 641.02
## - reason.course      1      1.32 1707.3 641.07
## - fjob.teacher       1      1.90 1707.9 641.23
## - school             1      2.10 1708.1 641.29
## - mjob.teacher       1      2.68 1708.7 641.45
## - fjob.at_home       1      4.69 1710.7 642.02
## <none>                           1706.0 642.70
## - dalc               1      8.74 1714.7 643.15
## - mjob.at_home       1     11.68 1717.7 643.97
## - mjob.other         1     12.03 1718.0 644.07
## - reason.reputation  1     19.66 1725.7 646.20
## - address            1     21.60 1727.6 646.74
## - mjob.services      1     21.71 1727.7 646.77
## - pass_fail          1   1545.26 3251.3 950.25
##
## Step:  AIC=640.77
## grade_mean ~ school + address + internet + dalc + walc + pass_fail +
##     fjob.at_home + fjob.other + fjob.teacher + mjob.at_home +
##     mjob.services + mjob.other + mjob.teacher + reason.course +
##     reason.reputation
##
##                     Df Sum of Sq    RSS    AIC
## - walc               1      0.36 1706.6 638.88
## - internet           1      0.95 1707.2 639.04
## - reason.course      1      1.29 1707.6 639.14
## - school             1      2.15 1708.4 639.38
## - fjob.other         1      2.47 1708.7 639.47
## - mjob.teacher       1      2.77 1709.0 639.55
## <none>                           1706.3 640.77
## - fjob.teacher       1      7.94 1714.2 641.00
## - fjob.at_home       1      8.82 1715.1 641.25
## - dalc               1      8.84 1715.1 641.25
## - mjob.at_home       1     12.17 1718.4 642.19
## - mjob.other         1     12.57 1718.8 642.30
## - reason.reputation  1     20.09 1726.4 644.39
```

```
## - address            1     21.88 1728.1 644.89
## - mjob.services      1     22.50 1728.8 645.06
## - pass_fail          1   1547.12 3253.4 948.56
##
## Step:  AIC=638.88
## grade_mean ~ school + address + internet + dalc + pass_fail +
##     fjob.at_home + fjob.other + fjob.teacher + mjob.at_home +
##     mjob.services + mjob.other + mjob.teacher + reason.course +
##     reason.reputation
##
##                     Df Sum of Sq    RSS    AIC
## - internet           1      0.96 1707.6 637.15
## - reason.course      1      1.31 1707.9 637.24
## - school             1      2.04 1708.7 637.45
## - fjob.other         1      2.52 1709.1 637.58
## - mjob.teacher       1      2.70 1709.3 637.63
## <none>                           1706.6 638.88
## - fjob.teacher       1      8.31 1714.9 639.21
## - fjob.at_home       1      8.77 1715.4 639.34
## - mjob.at_home       1     11.92 1718.6 640.22
## - mjob.other         1     12.29 1718.9 640.32
## - dalc               1     16.97 1723.6 641.62
## - reason.reputation  1     19.98 1726.6 642.46
## - address            1     21.77 1728.4 642.96
## - mjob.services      1     22.26 1728.9 643.10
## - pass_fail          1   1586.99 3293.6 952.46
##
## Step:  AIC=637.15
## grade_mean ~ school + address + dalc + pass_fail + fjob.at_home +
##     fjob.other + fjob.teacher + mjob.at_home + mjob.services +
##     mjob.other + mjob.teacher + reason.course + reason.reputation
##
##                     Df Sum of Sq    RSS    AIC
## - reason.course      1      1.26 1708.9 635.50
## - school             1      2.38 1710.0 635.81
## - fjob.other         1      2.54 1710.1 635.86
## - mjob.teacher       1      2.56 1710.1 635.86
## <none>                           1707.6 637.15
## - fjob.teacher       1      8.02 1715.6 637.39
## - fjob.at_home       1      9.32 1716.9 637.76
## - mjob.other         1     12.90 1720.5 638.76
## - mjob.at_home       1     13.01 1720.6 638.79
## - dalc               1     16.44 1724.0 639.74
## - reason.reputation  1     20.47 1728.1 640.86
## - mjob.services      1     22.28 1729.9 641.37
## - address            1     22.90 1730.5 641.54
## - pass_fail          1   1591.76 3299.4 951.29
##
## Step:  AIC=635.5
## grade_mean ~ school + address + dalc + pass_fail + fjob.at_home +
##     fjob.other + fjob.teacher + mjob.at_home + mjob.services +
##     mjob.other + mjob.teacher + reason.reputation
##
##                     Df Sum of Sq    RSS    AIC
```

```
## - mjob.teacher       1      2.37 1711.2 634.16
## - fjob.other         1      2.52 1711.4 634.21
## - school             1      2.55 1711.4 634.21
## <none>                           1708.9 635.50
## - fjob.teacher       1      8.05 1716.9 635.76
## - fjob.at_home       1      8.86 1717.7 635.98
## - mjob.at_home       1     12.47 1721.3 636.99
## - mjob.other         1     12.82 1721.7 637.09
## - dalc               1     17.69 1726.5 638.44
## - reason.reputation  1     20.33 1729.2 639.18
## - mjob.services      1     21.87 1730.7 639.60
## - address            1     22.35 1731.2 639.74
## - pass_fail          1   1602.02 3310.9 950.97
##
## Step:  AIC=634.16
## grade_mean ~ school + address + dalc + pass_fail + fjob.at_home +
##     fjob.other + fjob.teacher + mjob.at_home + mjob.services +
##     mjob.other + reason.reputation
##
##                     Df Sum of Sq    RSS    AIC
## - school             1      2.41 1713.6 632.84
## - fjob.other         1      2.49 1713.7 632.86
## - fjob.teacher       1      6.91 1718.1 634.10
## <none>                           1711.2 634.16
## - fjob.at_home       1      8.93 1720.1 634.66
## - mjob.at_home       1     11.20 1722.4 635.29
## - mjob.other         1     12.46 1723.7 635.64
## - dalc               1     18.98 1730.2 637.46
## - address            1     22.54 1733.8 638.44
## - reason.reputation  1     22.90 1734.1 638.54
## - mjob.services      1     24.37 1735.6 638.95
## - pass_fail          1   1599.90 3311.1 949.00
##
## Step:  AIC=632.84
## grade_mean ~ address + dalc + pass_fail + fjob.at_home + fjob.other +
##     fjob.teacher + mjob.at_home + mjob.services + mjob.other +
##     reason.reputation
##
##                     Df Sum of Sq    RSS    AIC
## - fjob.other         1      2.12 1715.8 631.43
## <none>                           1713.6 632.84
## - fjob.teacher       1      7.32 1720.9 632.88
## - fjob.at_home       1     10.28 1723.9 633.71
## - mjob.at_home       1     11.93 1725.6 634.17
## - mjob.other         1     13.57 1727.2 634.63
## - dalc               1     18.62 1732.2 636.03
## - mjob.services      1     23.83 1737.5 637.47
## - reason.reputation  1     25.29 1738.9 637.87
## - address            1     29.91 1743.5 639.15
## - pass_fail          1   1753.78 3467.4 969.14
##
## Step:  AIC=631.43
## grade_mean ~ address + dalc + pass_fail + fjob.at_home + fjob.teacher +
##     mjob.at_home + mjob.services + mjob.other + reason.reputation
```

```
##
##                     Df Sum of Sq    RSS    AIC
## <none>                           1715.8 631.43
## - fjob.at_home       1      8.23 1724.0 631.73
## - fjob.teacher       1     10.88 1726.6 632.47
## - mjob.at_home       1     13.30 1729.1 633.14
## - mjob.other         1     15.76 1731.5 633.82
## - dalc               1     17.82 1733.6 634.39
## - mjob.services      1     23.36 1739.1 635.93
## - reason.reputation  1     25.45 1741.2 636.50
## - address            1     29.90 1745.7 637.73
## - pass_fail          1   1752.28 3468.0 967.23
```

```
summary(regression_model)
```

```
##
## Call:
## lm(formula = grade_mean ~ address + dalc + pass_fail + fjob.at_home +
##     fjob.teacher + mjob.at_home + mjob.services + mjob.other +
##     reason.reputation, data = student_train_knn)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6366 -1.2825 -0.0388  1.2068  5.6995
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        8.95763    0.34478  25.981  < 2e-16 ***
## address            0.55921    0.19538   2.862  0.00440 **
## dalc              -0.21287    0.09634  -2.209  0.02762 *
## pass_fail          4.36773    0.19936  21.909  < 2e-16 ***
## fjob.at_home      -0.50150    0.33403  -1.501  0.13393
## fjob.teacher       0.64326    0.37257   1.727  0.08491 .
## mjob.at_home      -0.54876    0.28748  -1.909  0.05689 .
## mjob.services     -0.70378    0.27819  -2.530  0.01174 *
## mjob.other        -0.51075    0.24579  -2.078  0.03825 *
## reason.reputation  0.55847    0.21150   2.641  0.00855 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.911 on 470 degrees of freedom
## Multiple R-squared:  0.5779, Adjusted R-squared:  0.5698
## F-statistic:  71.5 on 9 and 470 DF,  p-value: < 2.2e-16
```

The model considered several factors such as address (a student's urban or rural address), dalc (weekday alcohol consumption), pass_fail (whether a student passed or failed), and aspects related to family (fjob and mjob categories) and reason.reputation (reason for choosing the school based on its reputation).

Significant Predictors: Positive Impact: The student's address has a significant positive impact on the average grade. It implies that students from certain locations tend to have slightly higher average grades.

Negative Impact: Higher weekday dalc (alcohol consumption) appears to have a negative impact on grades, indicating that increased alcohol intake during weekdays correlates with lower average grades.

Other Predictors: Some factors like certain types of parental jobs (fjob and mjob categories) and the reason for choosing the school based on reputation (reason.reputation) show some trends but might not have a strong statistically significant impact on the average grade.

The model is relatively effective, explaining about 57.8% of the variance in predicting average grades. The model's overall performance is statistically significant, suggesting that at least one of the predictors has a notable impact on predicting the average grade.

We are now going to test the model against the testing data set.

```
predictions_testing <- predict(regression_model, newdata = student_test_knn)


actual_pass_fail <- student_test_knn$pass_fail

# Calculate MSE
mse <- mean((actual_pass_fail - predictions_testing)^2)
mse
```

```
## [1] 124.6262
```

```
# Calculate RMSE
rmse <- sqrt(mse)
rmse
```

```
## [1] 11.16361
```

```
# Calculate R-squared
rsquared <- summary(regression_model)$r.squared
rsquared
```

```
## [1] 0.5779078
```

Mean Squared Error (MSE):

MSE = 124.6262: This value represents the average squared difference between the actual and predicted values. In this case, it indicates a relatively high error in prediction, suggesting that the model's predictions have considerable variance from the actual values.

Root Mean Squared Error (RMSE):

RMSE = 11.16361: RMSE is the square root of the MSE and is measured in the same units as the dependent variable. A higher RMSE value, such as this, implies that, on average, the model's predictions deviate by approximately 11.16 units from the actual values.

R-squared (R^2):

R-squared = 0.5779078: This metric represents the proportion of variance in the dependent variable (pass_fail) that is explained by the independent variables used in the model. An R-squared of approximately 0.58 indicates that the model accounts for about 58% of the variance in the target variable, pass_fail.

Summary

C5.0 Decision Tree:

The C5.0 model showcases commendable performance across various metrics. With an accuracy of 68.33%, it significantly outperforms the baseline model's accuracy of 71.67%. Highlighting its predictive strength, the model achieves a balanced accuracy score of 61.90%, indicating robustness in predicting both classes accurately.

An essential metric is the model's sensitivity, which stands at 47.06%, depicting its ability to correctly identify instances of the positive class (students who were failing). The precision of 44.44% implies a reasonable proportion of accurately identified failing students among the predicted positive instances.

Moreover, the F1 score of 45.71% signifies a fair balance between precision and recall, considering the model's ability to correctly identify failing students while minimizing false positives.

In summary, the C5.0 model exhibits moderate accuracy, demonstrating notable strengths in correctly identifying failing students (positive class), although there's room for improvement in precision and overall performance.

CART Model:

The CART model demonstrates fair performance across several metrics. With an accuracy of 67.5%, it slightly underperforms compared to the baseline model's accuracy of 71.67%. Despite this, the model displays a reasonably high balanced accuracy of 59.54%, indicating a moderate ability to predict both classes effectively.

An important metric, the sensitivity, stands at 41.18%, reflecting the model's capability to correctly identify instances of the positive class (students who were failing). The precision of 42.42% suggests a moderate proportion of accurately identified failing students among the predicted positive instances.

The F1 score of 41.79% reflects a moderate balance between precision and recall, indicating the model's capability to correctly identify failing students while minimizing false positives.

In summary, the CART model shows moderate accuracy and balanced accuracy. It demonstrates a reasonable ability to identify failing students (positive class) but might benefit from improvements in precision and overall predictive performance.

Regresion model

The regression model presents a comprehensive overview of its predictive capability. The model appears to perform moderately well, explaining approximately 57.98% of the variance in the grade_mean variable.

The coefficients reveal interesting insights into the impact of various predictors on the target variable. For instance, variables like 'pass_fail' and 'address' show a significant effect on the grade_mean, as indicated by their low p-values (< 0.05). The 'pass_fail' variable, in particular, stands out with a notable coefficient value of 4.37, indicating its substantial impact on the predicted grades.

However, some predictor variables, like 'fjob.at_home' and 'fjob.teacher,' do not demonstrate statistically significant effects on the grade_mean, considering their higher p-values (> 0.05).

The model's overall performance, denoted by the multiple R-squared value of 57.79%, suggests that the included predictors collectively explain a considerable portion of the variance observed in the target variable.

In summary, the regression model provides insights into the influence of various factors on the predicted grades. While some predictors significantly impact grade_mean, others seem less influential based on their coefficients and p-values. The model, with an R-squared value of around 57.98%, demonstrates a moderate level of explanatory power in forecasting grade_mean based on the given predictors.

Comparison of the models:

C5.0 Decision Tree: Accuracy: 68.33% Balanced Accuracy: 61.90% Sensitivity: 47.06% Precision: 44.44%

CART Model: Accuracy: 67.50% Balanced Accuracy: 59.54% Sensitivity: 41.18% Precision: 42.42%

Regression Model: R-squared: 57.98% Variables' Impact: 'pass_fail' significantly affects grade_mean (Estimate: 4.37) 'address' has a moderate effect (Estimate: 0.56) Others (e.g., 'fjob.at_home', 'fjob.teacher') show less impact.

Comparison: The CART and regression models display lower accuracy, balanced accuracy, sensitivity, and precision compared to the C5.0 model with an accuracy value of around 68.33%. The regression model, albeit different in nature, indicates a relatively higher level of explanatory power concerning the target variable ('pass_fail') based on the included predictors.

The classification models (C5.0 and CART) could predict class labels, their overall accuracy and precision were notably higher than the regression model's explanatory power in capturing the variance within 'grade_mean'. However, the regression model primarily focuses on the continuous prediction of grades rather than class labels.

Conclusion:

The C5.0 Decision Tree and CART models both present moderate predictive performances in classifying student outcomes, demonstrating accuracies around 68.33% and 67.50%, respectively. While they showcase strengths in correctly identifying failing students (positive class), their precision and overall predictive capacities could benefit from enhancements.

On the other hand, the Regression model, targeting continuous grade predictions, reveals a distinct utility. With an R-squared value of 57.98%, it offers insights into the influence of various factors on predicted grades. Notably, 'pass_fail' significantly affects grade_mean, indicating its substantial impact on predicted grades.

Comparatively, while the classification models indicate better accuracy and precision in classifying students, the Regression model provides a comprehensive understanding of how different factors collectively influence predicted grades. Each model serves distinct purposes, offering valuable insights into student outcomes based on their predictive targets and methodologies.