Exercise Sheet 6

```
library(tidyverse)
library(forcats)
library(stringr)
library(purrr)
library(caret)

# Function that calculates the Gini Index of a partitioning of x w.r.t. y
myGini <- function(x,y) {
  ti <- tibble(x, y) # generates a table from one attribute x (e.g. sex), and alco_prob
  rat <- prop.table(table(ti$x)) # calculates the percentage amount of females and males
  ti <- ti %>%
    split(.$x) %>% # number of males and females w.r.t alco_prob (alc and no_alc)
    map(~prop.table(table(.$y))) %>% # applies function to calculate the percentage
    #amount of alc_prop and and n_alc_prop with females ind males
    map(~ 1 - sum(.^2)) %>%
    unlist()
  return(sum(ti*rat))
}
```

What factors explain excessive alcohol consumption among students? The record for the task sheet comes from a survey of students who attended mathematics and Portuguese courses and contains many interesting details about their sociodemographics, life circumstances and learning success. The ordinal scaled variables `Dalc` and `Walc` give information about the alcohol consumption of the students on weekdays and weekends. Create a binary target variable `alc_prob` as follows:

```
library(tidyverse)
library(caret)
# (adapt path)
student <- read_csv("student_alc.csv")
student <- student %>%
    map_if(is.character, as.factor) %>%
  bind_cols()
student <- student %>%
  mutate(alc_prob = ifelse(Dalc + Walc >= 6, "alc_p", "no_alc_p"))
```

1. Calculate the Gini index for the target variable `alc_prob` and the *Gini index* for each variable with respect to `alc_prob`. Determine the 5 variables with the highest *Gini Gain*.

```
# attributes(student) <- NULL

#395 observations, 30 variables
str(student)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    395 obs. of  32 variables:
##  $ sex       : Factor w/ 2 levels "F","M": 1 1 1 1 1 2 2 1 2 2 ...
##  $ age       : num  18 17 15 15 16 16 16 17 15 15 ...
##  $ famsize   : Factor w/ 2 levels "GT3","LE3": 1 1 2 1 1 2 2 1 2 1 ...
```

```
##  $ Pstatus   : Factor w/ 2 levels "A","T": 1 2 2 2 2 2 2 1 1 2 ...
##  $ Medu      : num  4 1 1 4 3 4 2 4 3 3 ...
##  $ Fedu      : num  4 1 1 2 3 3 2 4 2 4 ...
##  $ Mjob      : Factor w/ 5 levels "at_home","health",..: 1 1 1 2 3 4 3 3 4 3 ...
##  $ Fjob      : Factor w/ 5 levels "at_home","health",..: 5 3 3 4 3 3 3 5 3 3 ...
##  $ reason    : Factor w/ 4 levels "course","home",..: 1 1 3 2 2 4 2 2 2 2 ...
##  $ guardian  : Factor w/ 3 levels "father","mother",..: 2 1 2 2 1 2 2 2 2 2 ...
##  $ traveltime: num  2 1 1 1 1 1 1 2 1 1 ...
##  $ studytime : num  2 2 2 3 2 2 2 2 2 2 ...
##  $ failures  : num  0 0 3 0 0 0 0 0 0 0 ...
##  $ schoolsup : Factor w/ 2 levels "no","yes": 2 1 2 1 1 1 1 2 1 1 ...
##  $ famsup    : Factor w/ 2 levels "no","yes": 1 2 1 2 2 2 1 2 2 2 ...
##  $ paid      : Factor w/ 2 levels "no","yes": 1 1 2 2 2 2 1 1 2 2 ...
##  $ activities: Factor w/ 2 levels "no","yes": 1 1 1 2 1 2 1 1 1 2 ...
##  $ nursery   : Factor w/ 2 levels "no","yes": 2 1 2 2 2 2 2 2 2 2 ...
##  $ higher    : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ internet  : Factor w/ 2 levels "no","yes": 1 2 2 2 1 2 2 1 2 2 ...
##  $ romantic  : Factor w/ 2 levels "no","yes": 1 1 1 2 1 1 1 1 1 1 ...
##  $ famrel    : num  4 5 4 3 4 5 4 4 4 5 ...
##  $ freetime  : num  3 3 3 2 3 4 4 1 2 5 ...
##  $ goout     : num  4 3 2 2 2 2 4 4 2 1 ...
##  $ Dalc      : num  1 1 2 1 1 1 1 1 1 1 ...
##  $ Walc      : num  1 1 3 1 2 2 1 1 1 1 ...
##  $ health    : num  3 3 3 5 5 5 3 1 1 5 ...
##  $ absences  : num  6 4 10 2 4 10 0 6 0 0 ...
##  $ G1        : num  5 5 7 15 6 15 12 6 16 14 ...
##  $ G2        : num  6 5 8 14 10 15 12 5 18 15 ...
##  $ G3        : num  6 6 10 15 10 15 11 6 19 15 ...
##  $ alc_prob  : chr  "no_alc_p" "no_alc_p" "no_alc_p" "no_alc_p" ...
```

```r
# class distribution
table(student$alc_prob)
```

```
## 
##    alc_p no_alc_p 
##       74      321
```

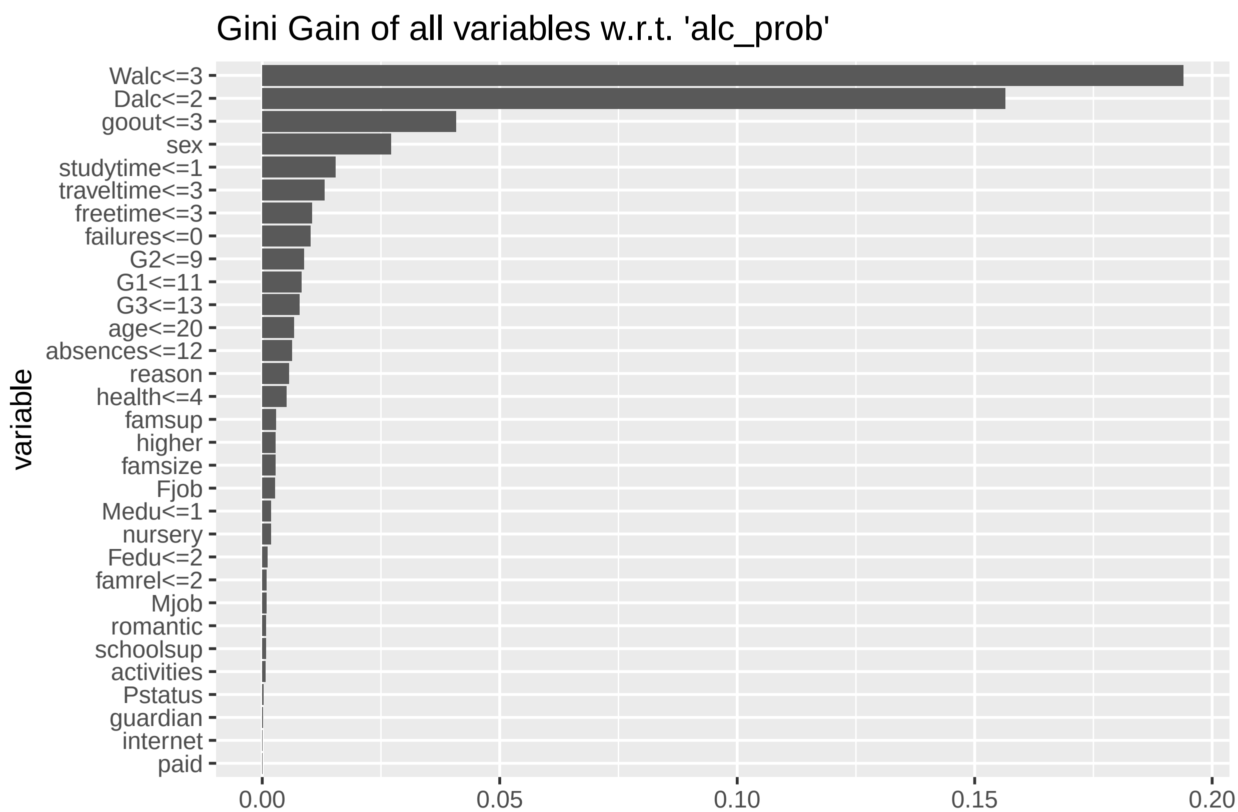
```r
# Gini-Index of target variable
gini_class <- 1 - sum(prop.table(table(student$alc_prob))^2)
gini_class
```

```
## [1] 0.3044897
```

```r
# Gini-Index of each variable w.r.t. 'alc_prob'
li_gini <- vector("list", length = ncol(student))

for(var in 1:ncol(student)) {
  if(is.factor(student[[var]])) {
    df_gini <- tibble(
      variable = names(student)[[var]],
```

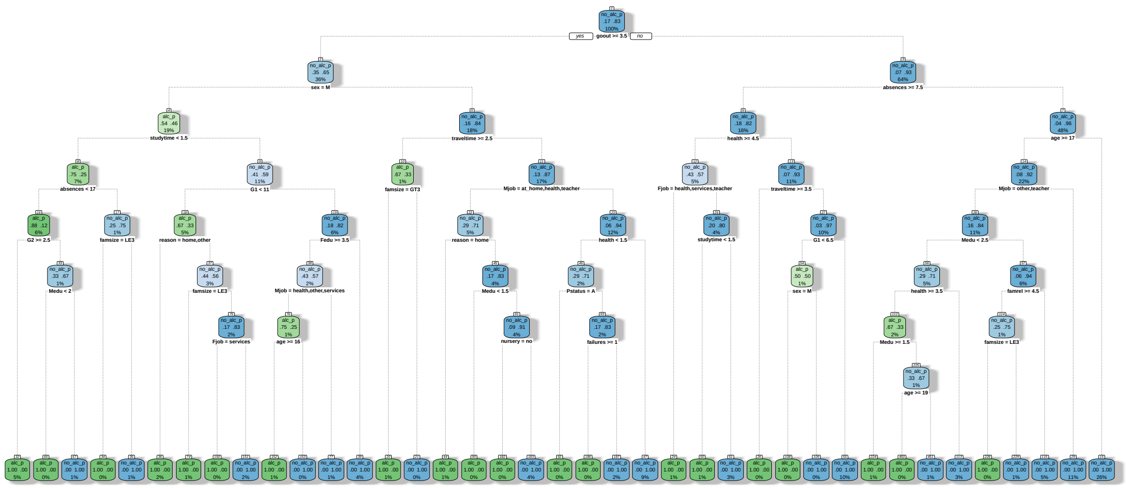
```r
      gini = NA
    )
    df_gini$gini[1] <- myGini(student[[var]], student$alc_prob)
    li_gini[[var]] <- df_gini
  }

  # For numeric variables calculate Gini index for all possible split points
  if(is.numeric(student[[var]])) {
    split_points <- sort(unique(student[[var]]))
    df_gini <- tibble(
      variable = str_c(names(student)[[var]], "<=", split_points),
      gini = NA
    )

    for(sp in 1:length(split_points)) {
      temp_var <- cut(student[[var]], breaks = c(-Inf, split_points[sp], Inf))
      df_gini$gini[sp] <- myGini(temp_var, student$alc_prob)
    }
    # Choose best split, i.e. split with lowest Gini Index
    li_gini[[var]] <- df_gini %>% filter(!is.nan(gini)) %>% arrange(gini) %>% slice(1)
  }
}
student_gini <- do.call("rbind", li_gini)
```

```r
student_gini %>%
  filter(!variable == "alc_prob") %>%
  mutate(gini_gain = myGini(1, student$alc_prob) - gini) %>%
  mutate(variable = forcats::fct_reorder(variable, gini_gain)) %>%
  ggplot(aes(x = variable, y = gini_gain)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(title = "Gini Gain of all variables w.r.t. 'alc_prob'", y = "")
```



2. Learn 2 different decision trees with `alc_prob` as target variable. For the first tree, nodes should be further partitioned until the class distribution of all resulting leaf nodes is pure. For the second tree, nodes with a cardinality of less than 20 instances should not be further partitioned. Determine the quality of the trees by calculating sensitivity (*True Positive Rate*) and specificity (*True Negative Rate*) for a 70%:30% split in training and test sets. Display the decision trees graphically and discuss the differences in quality measures.

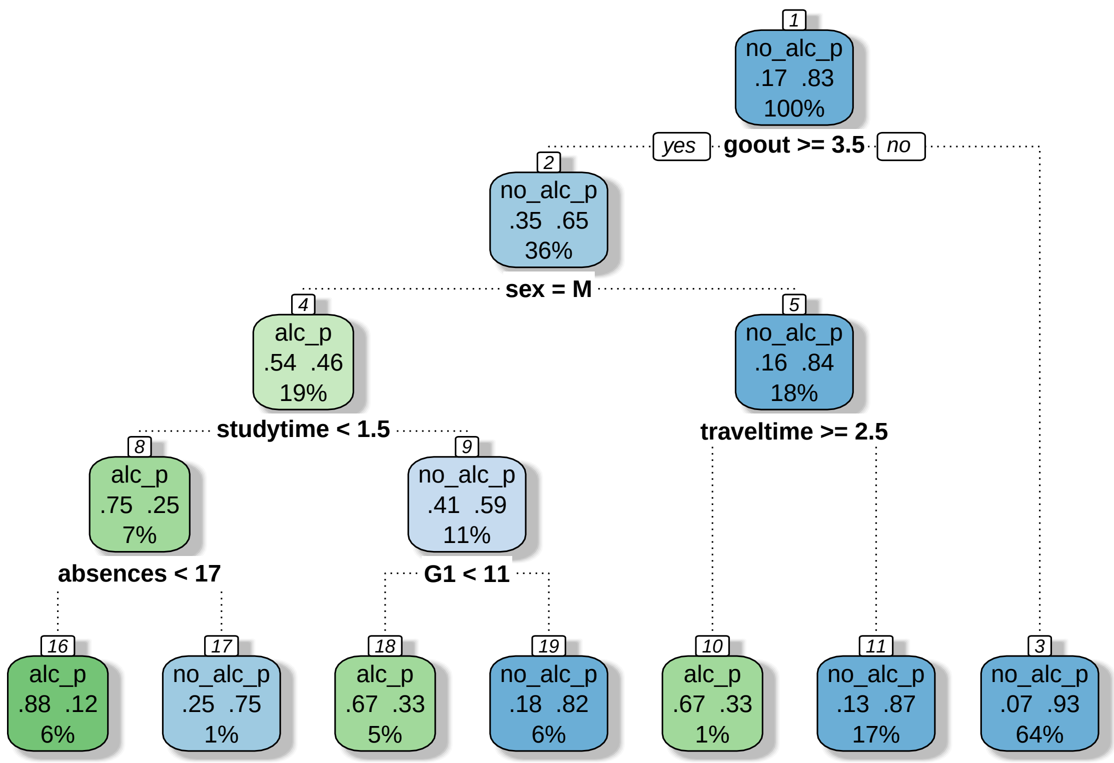
```
set.seed(123)
inTrain <- sample(c(FALSE, TRUE), size = nrow(student), replace = TRUE, prob = c(.3, .7))

student <- map_df(student, ~if(is.character(.)){factor(.)}else{.})

student_train <- student %>% select(-Walc, -Dalc) %>% filter(inTrain)
student_test <- student %>% select(-Walc, -Dalc) %>% filter(!inTrain)


library(rpart)
library(rattle)
fit <- rpart(alc_prob ~ ., data = student_train, control = rpart.control(minsplit = 1, minbucke
fancyRpartPlot(fit, sub = "")
```



```r
p <- predict(fit, student_test %>% select(-alc_prob) , type = "class")

cm <- confusionMatrix(student_test$alc_prob, p, dnn = c("True Label", "Predicted Label"))
cm
```

```
## Confusion Matrix and Statistics
##
##             Predicted Label
## True Label alc_p no_alc_p
##   alc_p       10       15
##   no_alc_p    18       71
##
##                Accuracy : 0.7105
##                  95% CI : (0.6181, 0.7916)
##     No Information Rate : 0.7544
##     P-Value [Acc > NIR] : 0.8830
##
##                   Kappa : 0.1896
##
##  Mcnemar's Test P-Value : 0.7277
##
##             Sensitivity : 0.35714
##             Specificity : 0.82558
##          Pos Pred Value : 0.40000
##          Neg Pred Value : 0.79775
##              Prevalence : 0.24561
##          Detection Rate : 0.08772
##    Detection Prevalence : 0.21930
##       Balanced Accuracy : 0.59136
##
##        'Positive' Class : alc_p
##
```

```r
fit <- rpart(alc_prob ~ ., data = student_train, control = rpart.control(minsplit = 20, minbuc
fancyRpartPlot(fit, sub = "")
```




```r
p <- predict(fit, student_test %>% select(-alc_prob) , type = "class")

cm <- confusionMatrix(student_test$alc_prob, p, dnn = c("True Label", "Predicted Label"))
cm

## Confusion Matrix and Statistics
##
##             Predicted Label
## True Label alc_p no_alc_p
##   alc_p       11       14
##   no_alc_p     3       86
##
##                Accuracy : 0.8509
##                  95% CI : (0.772, 0.9107)
##     No Information Rate : 0.8772
##     P-Value [Acc > NIR] : 0.84152
##
##                   Kappa : 0.4826
##
##  Mcnemar's Test P-Value : 0.01529
##
```

```
##             Sensitivity : 0.78571
##             Specificity : 0.86000
##          Pos Pred Value : 0.44000
##          Neg Pred Value : 0.96629
##              Prevalence : 0.12281
##          Detection Rate : 0.09649
##    Detection Prevalence : 0.21930
##       Balanced Accuracy : 0.82286
##
##        'Positive' Class : alc_p
##
```

3. Use `randomForest::randomForest()` to create a random forest with 200 trees. As candidates for a split within a tree a random sample of 5 variables should be drawn. Calculate Accuracy, Sensitivity and Specificity for the Out-of-the-Bag instances and show the most important variables (`?importance`).

```
library(randomForest)
set.seed(123)
rf <- randomForest(alc_prob ~ ., data = student %>% select(-Dalc, -Walc), ntree = 200, mtry = 5

cm <- rf$confusion[1:2,1:2]
acc <- sum(diag(cm))/sum(sum(cm))
acc
```

```
## [1] 0.8202532
```

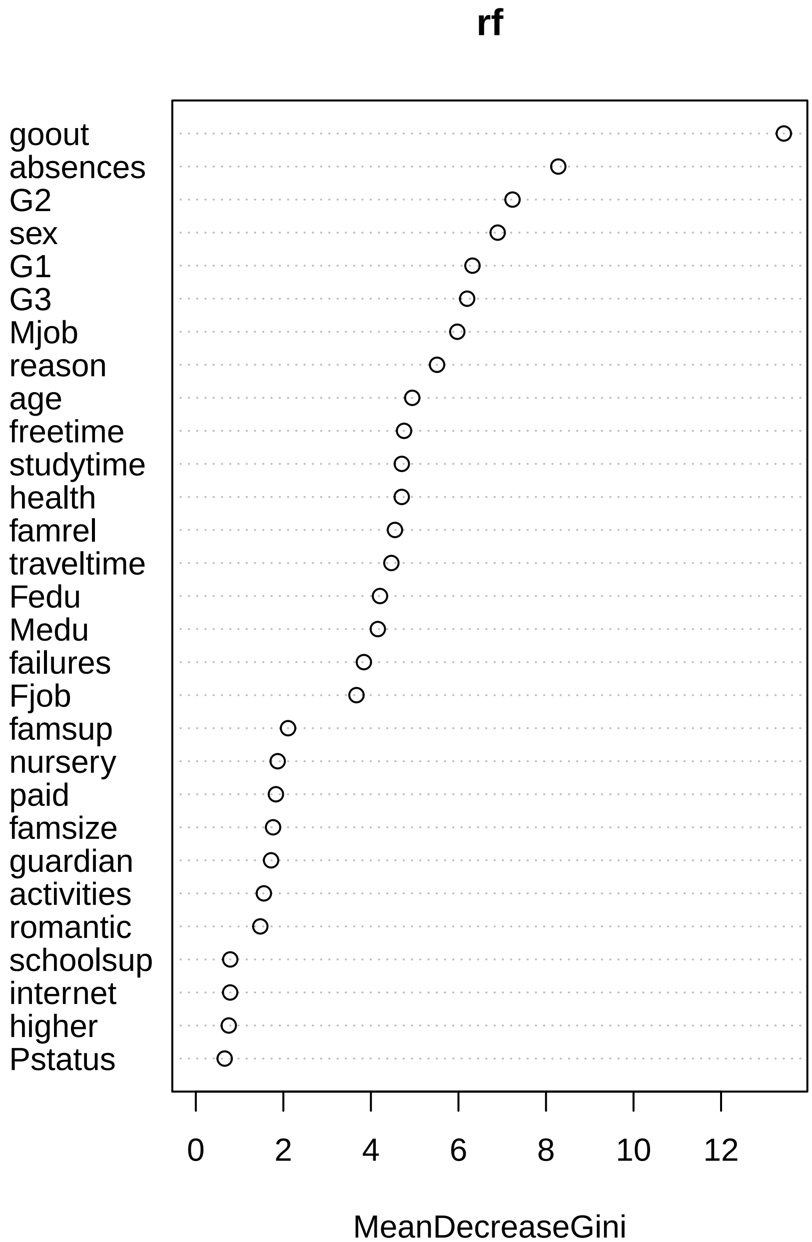
```
sens <- cm[1,1]/sum(cm[1,])
sens
```

```
## [1] 0.1621622
```

```
spec <- cm[2,2]/sum(cm[2,])
spec
```

```
## [1] 0.9719626
```

```
# ...from ?importance
# Here are the definitions of the variable importance measures. The first measure is computed f
#
# The second measure is the total decrease in node impurities from splitting on the variable, a

varImpPlot(rf, type = 2)
```



Dataset: http://isgwww.cs.uni-magdeburg.de/cv/lehre/VisAnalytics/material/exercise/datasets/student_alc.csv
(Source: https://www.kaggle.com/uciml/student-alcohol-consumption)