Data 622 Assignment 1

Keith DeNivo

Read in Data

```
file_url <- "https://raw.githubusercontent.com/division-zero/Data-622/refs/heads/main/assignm
# Download the file to a temporary location
temp_file <- tempfile(fileext = ".csv")
download.file(file_url, destfile = temp_file, mode = "wb")
# Read the csv file
fulldata <- read.delim(file_url, sep = ";", header = TRUE, stringsAsFactors = FALSE)
# View the data
head(fulldata)
```

```
  age       job marital   education default housing loan   contact month
1  56 housemaid married    basic.4y      no      no   no telephone   may
2  57  services married high.school unknown      no   no telephone   may
3  37  services married high.school      no     yes   no telephone   may
4  40    admin. married    basic.6y      no      no   no telephone   may
5  56  services married high.school      no      no  yes telephone   may
6  45  services married    basic.9y unknown      no   no telephone   may
  day_of_week duration campaign pdays previous    poutcome emp.var.rate
1         mon      261        1   999        0 nonexistent          1.1
2         mon      149        1   999        0 nonexistent          1.1
3         mon      226        1   999        0 nonexistent          1.1
4         mon      151        1   999        0 nonexistent          1.1
5         mon      307        1   999        0 nonexistent          1.1
6         mon      198        1   999        0 nonexistent          1.1
  cons.price.idx cons.conf.idx euribor3m nr.employed  y
1         93.994         -36.4     4.857        5191 no
2         93.994         -36.4     4.857        5191 no
```

```
3         93.994         -36.4     4.857        5191 no
4         93.994         -36.4     4.857        5191 no
5         93.994         -36.4     4.857        5191 no
6         93.994         -36.4     4.857        5191 no
```

```r
# Clean up the temporary file
unlink(temp_file)
```

```r
#smaller data set for training or testing
file_url <- "https://raw.githubusercontent.com/division-zero/Data-622/refs/heads/main/assignm
# Download the file to a temporary location
temp_file <- tempfile(fileext = ".csv")
download.file(file_url, destfile = temp_file, mode = "wb")
# Read the csv file
partdata <- read.delim(file_url, sep = ";", header = TRUE, stringsAsFactors = FALSE)
# View the data
head(partdata)
```

```
  age         job marital         education default housing    loan   contact
1  30 blue-collar married          basic.9y      no     yes      no  cellular
2  39    services  single       high.school      no      no      no telephone
3  25    services married       high.school      no     yes      no telephone
4  38    services married          basic.9y      no unknown unknown telephone
5  47      admin. married university.degree      no     yes      no  cellular
6  32    services  single university.degree      no      no      no  cellular
  month day_of_week duration campaign pdays previous    poutcome emp.var.rate
1   may         fri      487        2   999        0 nonexistent         -1.8
2   may         fri      346        4   999        0 nonexistent          1.1
3   jun         wed      227        1   999        0 nonexistent          1.4
4   jun         fri       17        3   999        0 nonexistent          1.4
5   nov         mon       58        1   999        0 nonexistent         -0.1
6   sep         thu      128        3   999        2     failure         -1.1
  cons.price.idx cons.conf.idx euribor3m nr.employed  y
1         92.893         -46.2     1.313      5099.1 no
2         93.994         -36.4     4.855      5191.0 no
3         94.465         -41.8     4.962      5228.1 no
4         94.465         -41.8     4.959      5228.1 no
5         93.200         -42.0     4.191      5195.8 no
6         94.199         -37.5     0.884      4963.6 no
```

```
# Clean up the temporary file
unlink(temp_file)
```

Basic data info

```
summary(fulldata)
```

```
      age            job              marital           education
 Min.   :17.00   Length:41188       Length:41188       Length:41188
 1st Qu.:32.00   Class :character   Class :character   Class :character
 Median :38.00   Mode  :character   Mode  :character   Mode  :character
 Mean   :40.02
 3rd Qu.:47.00
 Max.   :98.00
   default            housing              loan             contact
 Length:41188       Length:41188       Length:41188       Length:41188
 Class :character   Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character


    month           day_of_week           duration         campaign
 Length:41188       Length:41188       Min.   :   0.0   Min.   : 1.000
 Class :character   Class :character   1st Qu.: 102.0   1st Qu.: 1.000
 Mode  :character   Mode  :character   Median : 180.0   Median : 2.000
                                       Mean   : 258.3   Mean   : 2.568
                                       3rd Qu.: 319.0   3rd Qu.: 3.000
                                       Max.   :4918.0   Max.   :56.000
     pdays          previous       poutcome          emp.var.rate
 Min.   :  0.0   Min.   :0.000   Length:41188       Min.   :-3.40000
 1st Qu.:999.0   1st Qu.:0.000   Class :character   1st Qu.:-1.80000
 Median :999.0   Median :0.000   Mode  :character   Median : 1.10000
 Mean   :962.5   Mean   :0.173                      Mean   : 0.08189
 3rd Qu.:999.0   3rd Qu.:0.000                      3rd Qu.: 1.40000
 Max.   :999.0   Max.   :7.000                      Max.   : 1.40000
 cons.price.idx  cons.conf.idx     euribor3m      nr.employed
 Min.   :92.20   Min.   :-50.8   Min.   :0.634   Min.   :4964
 1st Qu.:93.08   1st Qu.:-42.7   1st Qu.:1.344   1st Qu.:5099
 Median :93.75   Median :-41.8   Median :4.857   Median :5191
 Mean   :93.58   Mean   :-40.5   Mean   :3.621   Mean   :5167
```

```
 3rd Qu.:93.99   3rd Qu.:-36.4   3rd Qu.:4.961   3rd Qu.:5228
 Max.   :94.77   Max.   :-26.9   Max.   :5.045   Max.   :5228
      y
 Length:41188
 Class :character
 Mode  :character
```

#converting yes to 1 and no to 0 for binary classification

```
partdata$term_deposit <- ifelse(partdata$y == "yes", 1, 0)
partdata$term_deposit_factor <- factor(partdata$term_deposit, levels = c(0, 1))
```

Base models

Build a tree model based on full data set

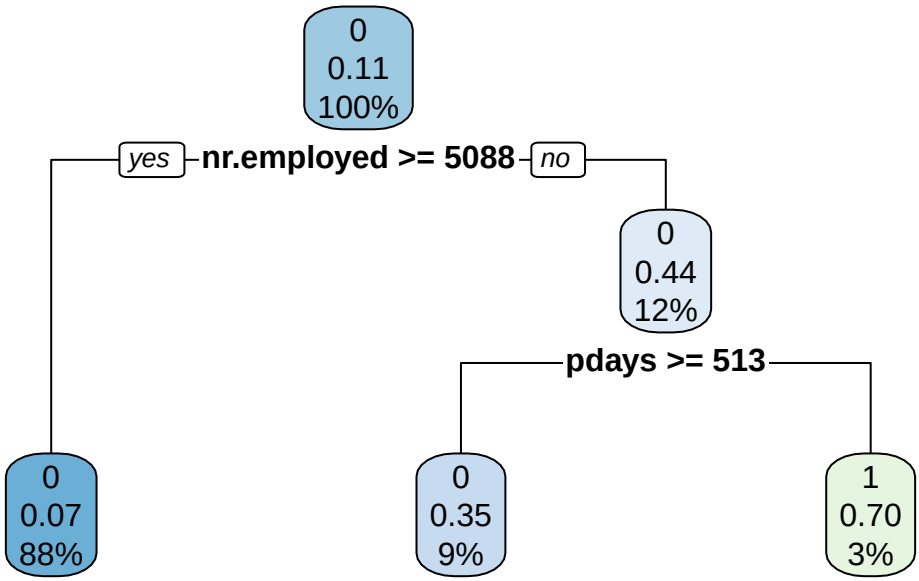
cp complexity parameter default is 0.01

```
fulldata$term_deposit <- ifelse(fulldata$y == "yes", 1, 0)
fulldata$term_deposit_factor <- factor(fulldata$term_deposit, levels = c(0, 1))
set.seed(42)
#removing columns/features that are directly related to target variable
modeldata <- fulldata |> dplyr::select(-y, -term_deposit, -duration)
#taking the full data set and splitting 0.7:0.3 train:test split.
train_index <- sample(seq_len(nrow(modeldata)), size = 0.7 * nrow(modeldata))
train_data <- modeldata[train_index, ]
test_data  <- modeldata[-train_index, ]

#train on full data set without missing features
tree_model <- rpart(
  term_deposit_factor ~ .,
  data = train_data,
  method = "class",
  control = rpart.control(
    cp = 0.01,
    minsplit = 20,
    maxdepth = 30
```

```
  )
)

tree_model
```

```
n= 28831

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 28831 3236 0 (0.88775970 0.11224030)
  2) nr.employed>=5087.65 25369 1724 0 (0.93204304 0.06795696) *
  3) nr.employed< 5087.65 3462 1512 0 (0.56325823 0.43674177)
    6) pdays>=513 2599  909 0 (0.65025010 0.34974990) *
    7) pdays< 513 863  260 1 (0.30127462 0.69872538) *
```

```
rpart.plot(tree_model)
```




```
pred_class <- predict(tree_model, test_data, type = "class")
```

```r
table(Predicted = pred_class, Actual = test_data$term_deposit_factor)
```

```
         Actual
Predicted     0     1
        0 10841  1134
        1   112   270
```

```r
mean(pred_class == test_data$term_deposit_factor)
```

```
[1] 0.8991665
```

```r
confusionMatrix(pred_class, test_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 10841  1134
         1   112   270

               Accuracy : 0.8992
                 95% CI : (0.8937, 0.9044)
    No Information Rate : 0.8864
    P-Value [Acc > NIR] : 2.847e-06

                  Kappa : 0.2667

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9898
            Specificity : 0.1923
         Pos Pred Value : 0.9053
         Neg Pred Value : 0.7068
             Prevalence : 0.8864
         Detection Rate : 0.8773
   Detection Prevalence : 0.9691
      Balanced Accuracy : 0.5910

       'Positive' Class : 0
```

```
Accuracy : 0.8992
```

sensitivity 0.9898

specificity 0.1923

kappa 0.2667

Random Forest

```
#random forest model
set.seed(42)
rf_model <- randomForest(
  term_deposit_factor ~ .,
  data = train_data,
  ntree = 500,
  importance = TRUE
)

forrest_pred_class <- predict(rf_model, test_data, type = "class")


rf_model
```

```
Call:
 randomForest(formula = term_deposit_factor ~ ., data = train_data,      ntree = 500, importa
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 4

        OOB estimate of  error rate: 10.29%
Confusion matrix:
      0   1 class.error
0 25029 566  0.02211369
1  2402 834  0.74227441
```

```
#confusion matrix of model
table(Predicted = forrest_pred_class, Actual = test_data$term_deposit_factor)
```

```
         Actual
Predicted     0     1
        0 10699  1013
        1   254   391
```

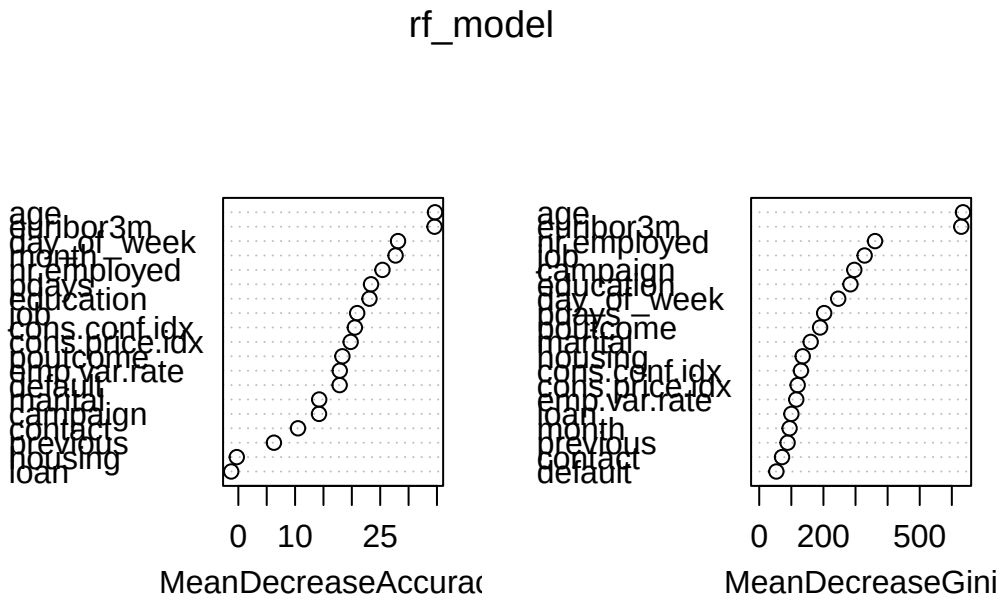
```
mean(forrest_pred_class == test_data$term_deposit_factor)
```

```
[1] 0.897467
```

```
confusionMatrix(forrest_pred_class, test_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 10699  1013
         1   254   391

               Accuracy : 0.8975
                 95% CI : (0.892, 0.9028)
    No Information Rate : 0.8864
    P-Value [Acc > NIR] : 4.377e-05

                  Kappa : 0.334

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9768
            Specificity : 0.2785
         Pos Pred Value : 0.9135
         Neg Pred Value : 0.6062
             Prevalence : 0.8864
         Detection Rate : 0.8658
   Detection Prevalence : 0.9478
      Balanced Accuracy : 0.6277

       'Positive' Class : 0
```

```
varImpPlot(rf_model)
```



Adaboost

```
#adaboost
ada_partdata <-partdata |>  dplyr::select(-y, -term_deposit, -default)
ada_partdata[] <- lapply(ada_partdata, function(x) {
  if (is.character(x)) factor(x) else x
})



set.seed(42)

idx <- createDataPartition(ada_partdata$term_deposit_factor, p = 0.7, list = FALSE)
part_train_data <- ada_partdata[idx, ]
part_test_data  <- ada_partdata[-idx, ]


ada_model <- boosting(
```

```
  term_deposit_factor ~ .,
  data = part_train_data,
  mfinal = 100,     # number of boosting iterations
  boos = TRUE
)

#ada_model

ada_pred <- predict(ada_model, part_test_data)

ada_pred$class <- factor(
  ada_pred$class,
  levels = levels(part_test_data$term_deposit_factor)
)

confusionMatrix(ada_pred$class, part_test_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1053   58
         1   47   77

               Accuracy : 0.915
                 95% CI : (0.898, 0.9299)
    No Information Rate : 0.8907
    P-Value [Acc > NIR] : 0.002777

                  Kappa : 0.5472

 Mcnemar's Test P-Value : 0.329114

            Sensitivity : 0.9573
            Specificity : 0.5704
         Pos Pred Value : 0.9478
         Neg Pred Value : 0.6210
             Prevalence : 0.8907
         Detection Rate : 0.8526
   Detection Prevalence : 0.8996
      Balanced Accuracy : 0.7638
```

```
       'Positive' Class : 0
```

```
ada_model$importance
```

```
                age            campaign      cons.conf.idx     cons.price.idx
         10.8225638           4.0182370           1.4636420           1.4424047
                contact         day_of_week            duration           education
          0.8964083           5.7047086          26.4282450           7.1719461
       emp.var.rate           euribor3m             housing                 job
          1.4904542          10.6128940           2.1332916          10.5441830
                loan             marital               month         nr.employed
          0.7079387           2.0421481           9.8731241           1.3662911
               pdays            poutcome            previous
          1.6024729           1.0359175           0.6431296
```

Remove features for all models

remove duration, default, loan, contact, education, employment variation rate. make the number of yes equal to the number of nos for term deposit subscription. Randomly selected term deposit row containing yes to equal the number of term deposit rows containting nos.

```
set.seed(42)
mod_data <-  modeldata |> dplyr::select( -default, -loan, -contact, -education, -emp.var.rate
#removing highly correlated features

term_deposit_no <- mod_data %>% filter(term_deposit_factor == 0)
term_deposit_yes  <- mod_data %>% filter(term_deposit_factor == 1)
#separating the positives and negatives
num_yes <- nrow(term_deposit_yes)


no_sample <- term_deposit_no %>% sample_n(num_yes) #sample the same number of negatives as po

equal_data <- bind_rows(term_deposit_yes, no_sample) #a dataset with equal number of positive
```

```
#create the training dataset and test data set based off the equal yes and no data above
eqtrain_index <- sample(seq_len(nrow(equal_data)), size = 0.7 * nrow(equal_data))
eqtrain_data <- equal_data[eqtrain_index, ]
eqtest_data  <- equal_data[-eqtrain_index, ]
```

rebuild models

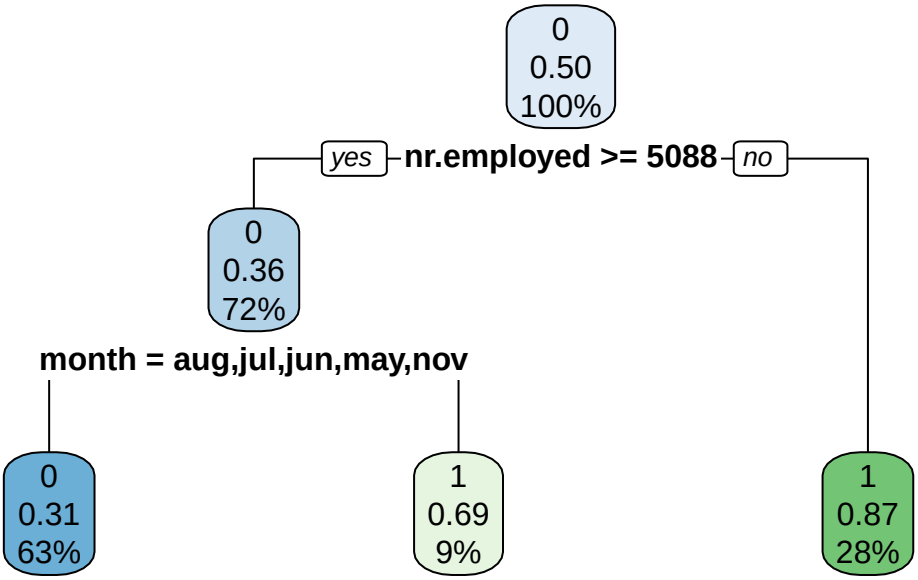
Rebuild the models

Decision Trees

```
set.seed(42)
#decision tree model

tree_model <- rpart(
  term_deposit_factor ~ .,
  data = eqtrain_data,
  method = "class",
  control = rpart.control(
    cp = 0.01,
    minsplit = 30,
    maxdepth = 20
  )
)

tree_model
```

```
n= 6496

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 6496 3246 0 (0.5003079 0.4996921)
  2) nr.employed>=5087.65 4699 1678 0 (0.6429027 0.3570973)
    4) month=aug,jul,jun,may,nov 4088 1255 0 (0.6930039 0.3069961) *
    5) month=apr,mar,oct 611  188 1 (0.3076923 0.6923077) *
  3) nr.employed< 5087.65 1797  229 1 (0.1274346 0.8725654) *
```

```
rpart.plot(tree_model)
```




```
pred_class <- predict(tree_model, eqtest_data, type = "class")
#pred_prob <- predict(tree_model, eqtest_data, type = "prob")



table(Predicted = pred_class, Actual = eqtest_data$term_deposit_factor)
```

```
         Actual
Predicted    0    1
        0 1215  566
        1  175  828
```

```
mean(pred_class == eqtest_data$term_deposit_factor)
```

```
[1] 0.7338362
```

```
#confusionMatrix(pred_class$class, actual, positive = "1")
confusionMatrix(pred_class, eqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics
```

```
          Reference
Prediction    0    1
         0 1215  566
         1  175  828

               Accuracy : 0.7338
                 95% CI : (0.717, 0.7502)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4679

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8741
            Specificity : 0.5940
         Pos Pred Value : 0.6822
         Neg Pred Value : 0.8255
             Prevalence : 0.4993
         Detection Rate : 0.4364
   Detection Prevalence : 0.6397
      Balanced Accuracy : 0.7340

       'Positive' Class : 0
```

0.7338 correct however the data is more balanced.

sensitivity: 0.8741 actual positives , specificity: 0.5940 actual negatives

kappa 0.4679

experiment 1:

cp = 0.001 overfit then prune

```
set.seed(42)
tree_model <- rpart(
  term_deposit_factor ~ .,
  data = eqtrain_data,
  method = "class",
  control = rpart.control(
    cp = 0.001,
```

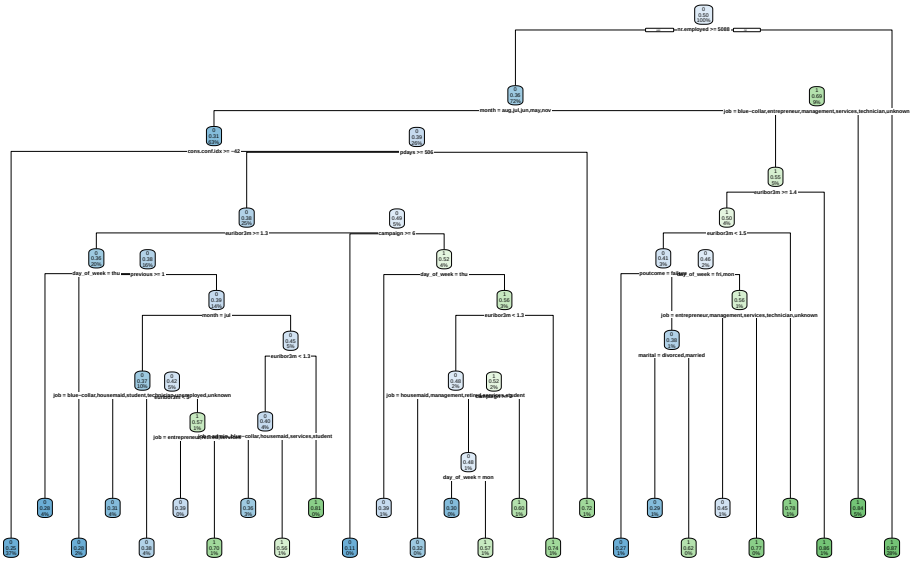
```
    minsplit = 30,
    maxdepth = 20
  )
)

tree_model
```

```
n= 6496

node), split, n, loss, yval, (yprob)
      * denotes terminal node

   1) root 6496 3246 0 (0.5003079 0.4996921)
     2) nr.employed>=5087.65 4699 1678 0 (0.6429027 0.3570973)
       4) month=aug,jul,jun,may,nov 4088 1255 0 (0.6930039 0.3069961)
         8) cons.conf.idx>=-42.35 2430  602 0 (0.7522634 0.2477366) *
         9) cons.conf.idx< -42.35 1658  653 0 (0.6061520 0.3938480)
          18) pdays>=505.5 1601  612 0 (0.6177389 0.3822611)
            36) euribor3m>=1.2755 1307  468 0 (0.6419281 0.3580719)
              72) day_of_week=thu 247   68 0 (0.7246964 0.2753036) *
              73) day_of_week=fri,mon,tue,wed 1060  400 0 (0.6226415 0.3773585)
               146) previous>=0.5 134   37 0 (0.7238806 0.2761194) *
               147) previous< 0.5 926  363 0 (0.6079914 0.3920086)
                 294) month=jul 632  232 0 (0.6329114 0.3670886)
                   588) job=blue-collar,housemaid,student,technician,unemployed,unknown 281
                   589) job=admin.,entrepreneur,management,retired,self-employed,services 35
                    1178) euribor3m< 4.9625 283  107 0 (0.6219081 0.3780919) *
                    1179) euribor3m>=4.9625 68   29 1 (0.4264706 0.5735294)
                      2358) job=entrepreneur,retired,services 28   11 0 (0.6071429 0.3928571)
                      2359) job=admin.,management,self-employed 40   12 1 (0.3000000 0.70000
                 295) month=may 294  131 0 (0.5544218 0.4455782)
                   590) euribor3m< 1.349 263  106 0 (0.5969582 0.4030418)
                    1180) job=admin.,blue-collar,housemaid,services,student 204   73 0 (0.642
                    1181) job=entrepreneur,management,retired,self-employed,technician,unempl
                   591) euribor3m>=1.349 31    6 1 (0.1935484 0.8064516) *
            37) euribor3m< 1.2755 294  144 0 (0.5102041 0.4897959)
              74) campaign>=5.5 19    2 0 (0.8947368 0.1052632) *
              75) campaign< 5.5 275  133 1 (0.4836364 0.5163636)
               150) day_of_week=thu 74   29 0 (0.6081081 0.3918919) *
               151) day_of_week=fri,mon,tue,wed 201   88 1 (0.4378109 0.5621891)
                 302) euribor3m< 1.2545 136   65 0 (0.5220588 0.4779412)
                   604) job=housemaid,management,retired,services,student 31   10 0 (0.67741
```

```
                  605) job=admin.,blue-collar,entrepreneur,self-employed,technician,unemploy
                   1210) campaign>=1.5 65   31 0 (0.5230769 0.4769231)
                     2420) day_of_week=mon 23    7 0 (0.6956522 0.3043478) *
                     2421) day_of_week=fri 42   18 1 (0.4285714 0.5714286) *
                   1211) campaign< 1.5 40   16 1 (0.4000000 0.6000000) *
                 303) euribor3m>=1.2545 65   17 1 (0.2615385 0.7384615) *
           19) pdays< 505.5 57   16 1 (0.2807018 0.7192982) *
         5) month=apr,mar,oct 611  188 1 (0.3076923 0.6923077)
          10) job=blue-collar,entrepreneur,management,services,technician,unknown 311  139 1 (
            20) euribor3m>=1.3685 267  133 1 (0.4981273 0.5018727)
              40) euribor3m< 1.4905 200   82 0 (0.5900000 0.4100000)
                80) poutcome=failure 55   15 0 (0.7272727 0.2727273) *
                81) poutcome=nonexistent,success 145   67 0 (0.5379310 0.4620690)
                 162) day_of_week=fri,mon 79   30 0 (0.6202532 0.3797468)
                   324) marital=divorced,married 58   17 0 (0.7068966 0.2931034) *
                   325) marital=single 21    8 1 (0.3809524 0.6190476) *
                 163) day_of_week=thu,tue,wed 66   29 1 (0.4393939 0.5606061)
                   326) job=entrepreneur,management,services,technician,unknown 44   20 0 (0.54
                   327) job=blue-collar 22    5 1 (0.2272727 0.7727273) *
              41) euribor3m>=1.4905 67   15 1 (0.2238806 0.7761194) *
            21) euribor3m< 1.3685 44    6 1 (0.1363636 0.8636364) *
          11) job=admin.,housemaid,retired,self-employed,student,unemployed 300   49 1 (0.1633
       3) nr.employed< 5087.65 1797  229 1 (0.1274346 0.8725654) *
```

```
rpart.plot(tree_model)
```

```
Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



```
pred_class <- predict(tree_model, eqtest_data, type = "class")
#pred_prob <- predict(tree_model, eqtest_data, type = "prob")



table(Predicted = pred_class, Actual = eqtest_data$term_deposit_factor)
```

```
         Actual
Predicted    0    1
        0 1204  518
        1  186  876
```

```
mean(pred_class == eqtest_data$term_deposit_factor)
```

```
[1] 0.7471264
```

```
#confusionMatrix(pred_class$class, actual, positive = "1")
confusionMatrix(pred_class, eqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics
```

```
          Reference
Prediction    0    1
         0 1204  518
         1  186  876

               Accuracy : 0.7471
                 95% CI : (0.7305, 0.7632)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4944

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8662
            Specificity : 0.6284
         Pos Pred Value : 0.6992
         Neg Pred Value : 0.8249
             Prevalence : 0.4993
         Detection Rate : 0.4325
   Detection Prevalence : 0.6185
      Balanced Accuracy : 0.7473

       'Positive' Class : 0
```

performs slightly better 0.7471 accuracy

sensitivity: 0.8662

specificity 0.6284

kappa 0.4944

remove additonal features from data: month, day

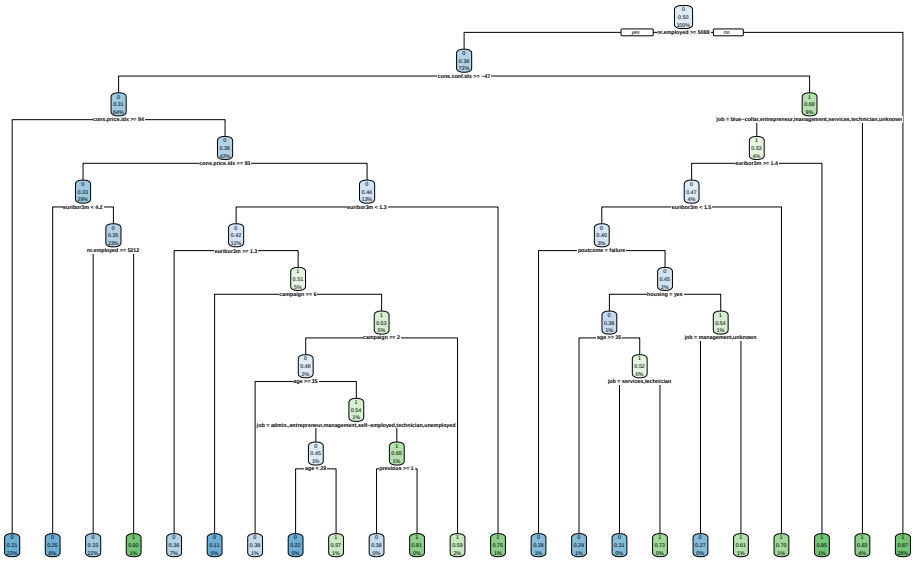
```
set.seed(42)
feqtrain_data <- eqtrain_data |> dplyr::select(-day_of_week,-month)
feqtest_data <- eqtest_data |> dplyr::select(-day_of_week,-month)

tree_model <- rpart(
  term_deposit_factor ~ .,
  data = feqtrain_data,
```

```r
  method = "class",
  control = rpart.control(
    cp = 0.001,
    minsplit = 30,
    maxdepth = 20
  )
)

tree_model
```

```
n= 6496

node), split, n, loss, yval, (yprob)
      * denotes terminal node

   1) root 6496 3246 0 (0.50030788 0.49969212)
     2) nr.employed>=5087.65 4699 1678 0 (0.64290275 0.35709725)
       4) cons.conf.idx>=-46.65 4125 1290 0 (0.68727273 0.31272727)
         8) cons.price.idx>=93.956 1400  297 0 (0.78785714 0.21214286) *
         9) cons.price.idx< 93.956 2725  993 0 (0.63559633 0.36440367)
          18) cons.price.idx>=93.0465 1885  622 0 (0.67002653 0.32997347)
            36) euribor3m< 4.207 380   94 0 (0.75263158 0.24736842) *
            37) euribor3m>=4.207 1505  528 0 (0.64916944 0.35083056)
              74) nr.employed>=5211.95 1446  474 0 (0.67219917 0.32780083) *
              75) nr.employed< 5211.95 59    5 1 (0.08474576 0.91525424) *
          19) cons.price.idx< 93.0465 840  371 0 (0.55833333 0.44166667)
            38) euribor3m< 1.349 787  331 0 (0.57941550 0.42058450)
              76) euribor3m>=1.2755 469  170 0 (0.63752665 0.36247335) *
              77) euribor3m< 1.2755 318  157 1 (0.49371069 0.50628931)
               154) campaign>=5.5 19    2 0 (0.89473684 0.10526316) *
               155) campaign< 5.5 299  140 1 (0.46822742 0.53177258)
                 310) campaign>=1.5 161   77 0 (0.52173913 0.47826087)
                   620) age>=34.5 65   25 0 (0.61538462 0.38461538) *
                   621) age< 34.5 96   44 1 (0.45833333 0.54166667)
                    1242) job=admin.,entrepreneur,management,self-employed,technician,unemplo
                      2484) age< 28.5 18    4 0 (0.77777778 0.22222222) *
                      2485) age>=28.5 35   15 1 (0.42857143 0.57142857) *
                    1243) job=blue-collar,services,student 43   15 1 (0.34883721 0.65116279)
                      2486) previous>=0.5 16    6 0 (0.62500000 0.37500000) *
                      2487) previous< 0.5 27    5 1 (0.18518519 0.81481481) *
                 311) campaign< 1.5 138   56 1 (0.40579710 0.59420290) *
            39) euribor3m>=1.349 53   13 1 (0.24528302 0.75471698) *
```

```
   5) cons.conf.idx< -46.65 574  186 1 (0.32404181 0.67595819)
    10) job=blue-collar,entrepreneur,management,services,technician,unknown 290  137 1 (0
      20) euribor3m>=1.396 243  113 0 (0.53497942 0.46502058)
        40) euribor3m< 1.481 190   76 0 (0.60000000 0.40000000)
          80) poutcome=failure 54   15 0 (0.72222222 0.27777778) *
          81) poutcome=nonexistent,success 136   61 0 (0.55147059 0.44852941)
           162) housing=yes 79   30 0 (0.62025316 0.37974684)
             324) age>=34.5 48   14 0 (0.70833333 0.29166667) *
             325) age< 34.5 31   15 1 (0.48387097 0.51612903)
               650) job=services,technician 16    5 0 (0.68750000 0.31250000) *
               651) job=blue-collar,entrepreneur,management,unknown 15    4 1 (0.2666666
           163) housing=no,unknown 57   26 1 (0.45614035 0.54385965)
             326) job=management,unknown 11    3 0 (0.72727273 0.27272727) *
             327) job=blue-collar,entrepreneur,services,technician 46   18 1 (0.39130435
        41) euribor3m>=1.481 53   16 1 (0.30188679 0.69811321) *
      21) euribor3m< 1.396 47    7 1 (0.14893617 0.85106383) *
    11) job=admin.,housemaid,retired,self-employed,student,unemployed 284   49 1 (0.1725
  3) nr.employed< 5087.65 1797  229 1 (0.12743461 0.87256539) *
```

```
rpart.plot(tree_model)
```



```r
pred_class <- predict(tree_model, feqtest_data, type = "class")
#pred_prob <- predict(tree_model, eqtest_data, type = "prob")



table(Predicted = pred_class, Actual = feqtest_data$term_deposit_factor)
```

```
         Actual
Predicted    0    1
        0 1212  523
        1  178  871
```

```r
mean(pred_class == feqtest_data$term_deposit_factor)
```

```
[1] 0.748204
```

```r
#confusionMatrix(pred_class$class, actual, positive = "1")
confusionMatrix(pred_class, feqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1212  523
         1  178  871

               Accuracy : 0.7482
                 95% CI : (0.7316, 0.7642)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4966

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8719
            Specificity : 0.6248
         Pos Pred Value : 0.6986
         Neg Pred Value : 0.8303
             Prevalence : 0.4993
```

```
         Detection Rate : 0.4353
   Detection Prevalence : 0.6232
      Balanced Accuracy : 0.7484

       'Positive' Class : 0
```

slightly better performance

accuracy 0.7482

sensitivity : 0.8719

specificity: 0.6248 slightly worse

kappa .4966

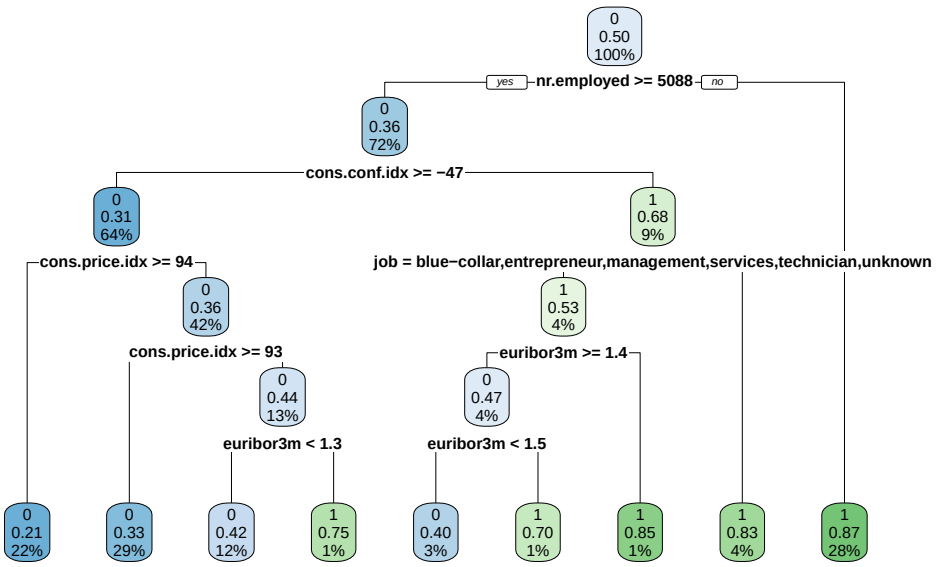
pruning the tree 5

```
set.seed(42)
tree_model <- rpart(
  term_deposit_factor ~ .,
  data = feqtrain_data,
  method = "class",
  control = rpart.control(
    cp = 0.001,
    minsplit = 30,
    maxdepth = 5
  )
)

tree_model
```

```
n= 6496

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 6496 3246 0 (0.5003079 0.4996921)
   2) nr.employed>=5087.65 4699 1678 0 (0.6429027 0.3570973)
     4) cons.conf.idx>=-46.65 4125 1290 0 (0.6872727 0.3127273)
       8) cons.price.idx>=93.956 1400  297 0 (0.7878571 0.2121429) *
       9) cons.price.idx< 93.956 2725  993 0 (0.6355963 0.3644037)
        18) cons.price.idx>=93.0465 1885  622 0 (0.6700265 0.3299735) *
```

```
      19) cons.price.idx< 93.0465 840  371 0 (0.5583333 0.4416667)
        38) euribor3m< 1.349 787  331 0 (0.5794155 0.4205845) *
        39) euribor3m>=1.349 53   13 1 (0.2452830 0.7547170) *
    5) cons.conf.idx< -46.65 574  186 1 (0.3240418 0.6759582)
     10) job=blue-collar,entrepreneur,management,services,technician,unknown 290  137 1 (0.4
       20) euribor3m>=1.396 243  113 0 (0.5349794 0.4650206)
         40) euribor3m< 1.481 190   76 0 (0.6000000 0.4000000) *
         41) euribor3m>=1.481 53   16 1 (0.3018868 0.6981132) *
       21) euribor3m< 1.396 47    7 1 (0.1489362 0.8510638) *
     11) job=admin.,housemaid,retired,self-employed,student,unemployed 284   49 1 (0.172535
  3) nr.employed< 5087.65 1797  229 1 (0.1274346 0.8725654) *
```

```
rpart.plot(tree_model)
```




```
pred_class <- predict(tree_model, feqtest_data, type = "class")
#pred_prob <- predict(tree_model, eqtest_data, type = "prob")



table(Predicted = pred_class, Actual = feqtest_data$term_deposit_factor)
```

```
         Actual
```

```
Predicted    0    1
        0 1247  588
        1  143  806
```

```r
mean(pred_class == feqtest_data$term_deposit_factor)
```

```
[1] 0.7374282
```

```r
#confusionMatrix(pred_class$class, actual, positive = "1")
confusionMatrix(pred_class, feqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1247  588
         1  143  806

               Accuracy : 0.7374
                 95% CI : (0.7207, 0.7537)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4751

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8971
            Specificity : 0.5782
         Pos Pred Value : 0.6796
         Neg Pred Value : 0.8493
             Prevalence : 0.4993
         Detection Rate : 0.4479
   Detection Prevalence : 0.6591
      Balanced Accuracy : 0.7377

       'Positive' Class : 0
```

sensitivity improves slightly and everything else is worse.

retraining the Random Forest model with the equally split data

```r
set.seed(42)
#random forest model

rf_model <- randomForest(
  term_deposit_factor ~ .,
  data = eqtrain_data,
  #method = "class"
  ntree = 500,
  importance = TRUE
)

forrest_pred_class <- predict(rf_model, eqtest_data, type = "class")
forrest_pred_prob <- predict(rf_model, eqtest_data, type = "prob")

rf_model
```

```
Call:
 randomForest(formula = term_deposit_factor ~ ., data = eqtrain_data,      ntree = 500, impo
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 25.03%
Confusion matrix:
     0    1 class.error
0 2830  420   0.1292308
1 1206 2040   0.3715342
```

```r
table(Predicted = forrest_pred_class, Actual = eqtest_data$term_deposit_factor)
```

```
         Actual
Predicted    0    1
        0 1223  544
        1  167  850
```

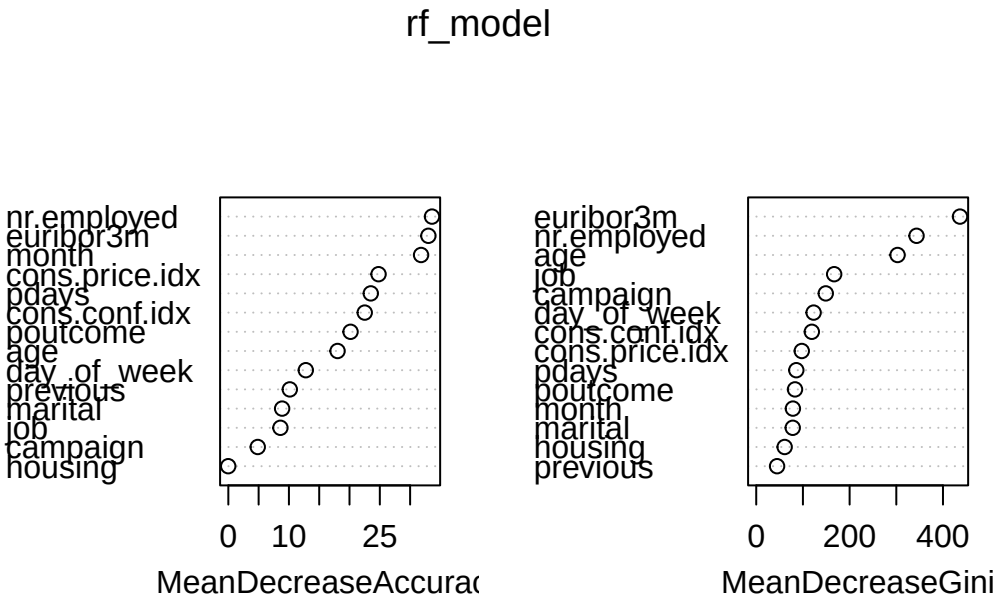
```r
mean(forrest_pred_class == eqtest_data$term_deposit_factor)
```

```
[1] 0.7446121
```

```r
confusionMatrix(forrest_pred_class, eqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1223  544
         1  167  850

               Accuracy : 0.7446
                 95% CI : (0.728, 0.7607)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4894

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8799
            Specificity : 0.6098
         Pos Pred Value : 0.6921
         Neg Pred Value : 0.8358
             Prevalence : 0.4993
         Detection Rate : 0.4393
   Detection Prevalence : 0.6347
      Balanced Accuracy : 0.7448

       'Positive' Class : 0
```

```r
varImpPlot(rf_model)
```



surprisingly model performance is roughly equal to just one decision tree

kappa 0.4894, sensitivity 0.8791, specificity 0.6105 accuracy 0.7446

increase the number of trees from 500 to 1000 (2x)

```
set.seed(42)
rf_model <- randomForest(
  term_deposit_factor ~ .,
  data = eqtrain_data,
  #method = "class"
  ntree = 1000,
  importance = TRUE
)

forrest_pred_class <- predict(rf_model, eqtest_data, type = "class")
forrest_pred_prob <- predict(rf_model, eqtest_data, type = "prob")

rf_model
```

```
Call:
 randomForest(formula = term_deposit_factor ~ ., data = eqtrain_data,      ntree = 1000, impo
               Type of random forest: classification
```

```
                    Number of trees: 1000
No. of variables tried at each split: 3

        OOB estimate of  error rate: 25.18%
Confusion matrix:
     0    1 class.error
0 2825  425   0.1307692
1 1211 2035   0.3730746
```

```r
table(Predicted = forrest_pred_class, Actual = eqtest_data$term_deposit_factor)
```

```
         Actual
Predicted    0    1
        0 1226  542
        1  164  852
```

```r
mean(forrest_pred_class == eqtest_data$term_deposit_factor)
```

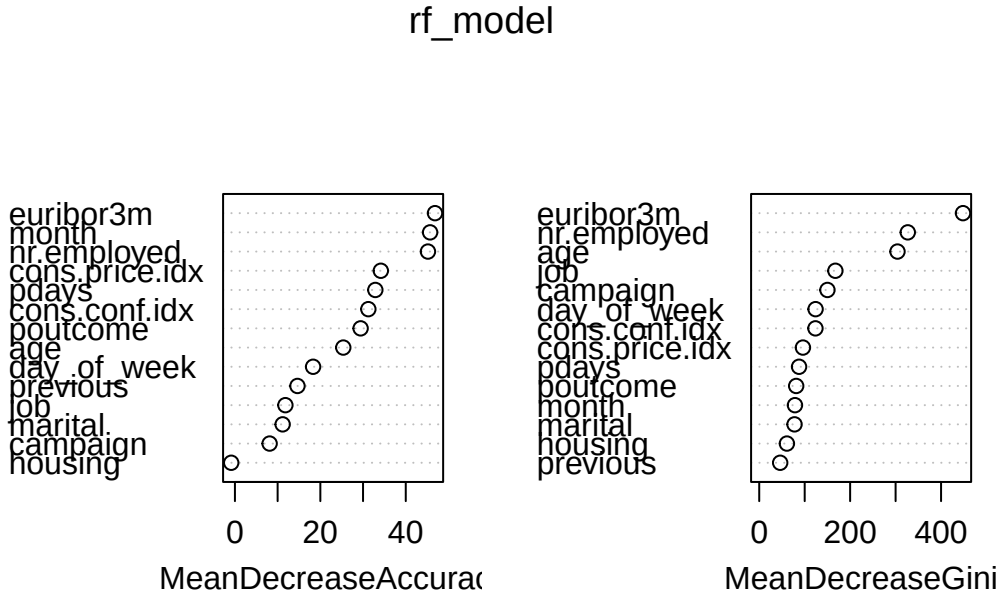
```
[1] 0.746408
```

```r
confusionMatrix(forrest_pred_class, eqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1226  542
         1  164  852

               Accuracy : 0.7464
                 95% CI : (0.7298, 0.7625)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.493

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8820
            Specificity : 0.6112
```

```
          Pos Pred Value : 0.6934
          Neg Pred Value : 0.8386
              Prevalence : 0.4993
          Detection Rate : 0.4404
    Detection Prevalence : 0.6351
       Balanced Accuracy : 0.7466

        'Positive' Class : 0
```

```
varImpPlot(rf_model)
```



only a slight improvement:

0.745 accuracy,

kappa 0.4901, sensitivity 0.8799, specificity 0.6105.

lets remove features that have both a low mean decrease accuracy and mean decrease gini

day_of_week, marital, housing, previous

```r
set.seed(42)
rfeqtrain_data <- eqtrain_data |> dplyr::select(-day_of_week,-marital, -housing, -previous)
rfeqtest_data <- eqtest_data |> dplyr::select(-day_of_week,-marital, -housing, -previous)

rf_model <- randomForest(
  term_deposit_factor ~ .,
  data = rfeqtrain_data,
  #method = "class"
  ntree = 1000,
  importance = TRUE
)

forrest_pred_class <- predict(rf_model, rfeqtest_data, type = "class")
forrest_pred_prob <- predict(rf_model, rfeqtest_data, type = "prob")

rf_model
```

```
Call:
 randomForest(formula = term_deposit_factor ~ ., data = rfeqtrain_data,      ntree = 1000, i
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 3

        OOB estimate of  error rate: 24.82%
Confusion matrix:
     0    1 class.error
0 2846  404   0.1243077
1 1208 2038   0.3721503
```

```r
table(Predicted = forrest_pred_class, Actual = rfeqtest_data$term_deposit_factor)
```

```
         Actual
Predicted    0    1
        0 1236  556
        1  154  838
```

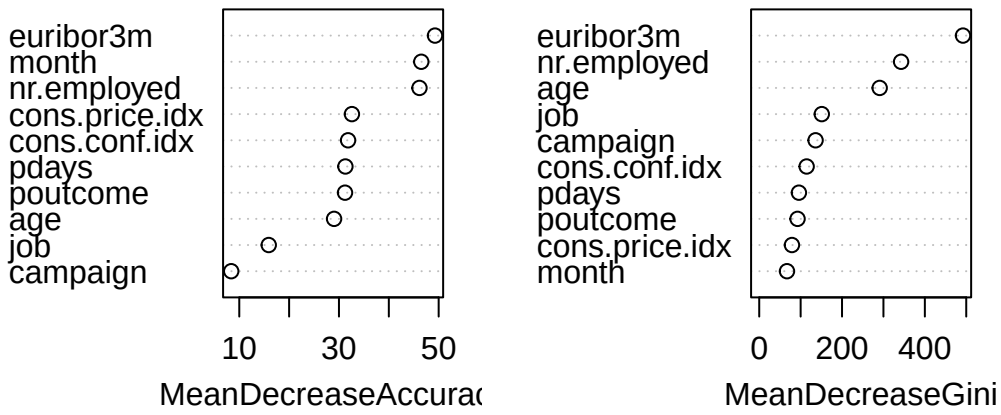
```r
mean(forrest_pred_class == rfeqtest_data$term_deposit_factor)
```

```
[1] 0.7449713
```

```r
confusionMatrix(forrest_pred_class, rfeqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1236  556
         1  154  838

               Accuracy : 0.745
                 95% CI : (0.7283, 0.7611)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4902

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8892
            Specificity : 0.6011
         Pos Pred Value : 0.6897
         Neg Pred Value : 0.8448
             Prevalence : 0.4993
         Detection Rate : 0.4440
   Detection Prevalence : 0.6437
      Balanced Accuracy : 0.7452

       'Positive' Class : 0
```

```r
varImpPlot(rf_model)
```



Accuracy : 0.7453, Kappa : 0.4909, Sensitivity : 0.8827, Specificity : 0.6083
specificity decreased

tuned tree

```
set.seed(42)
rfeqtrain_data$term_deposit_factor <- factor(
  ifelse(rfeqtrain_data$term_deposit_factor == 1, "yes", "no"),
  levels = c("yes", "no")   # convert back to yes and no
)

rfeqtest_data$term_deposit_factor <- factor(
  ifelse(rfeqtest_data$term_deposit_factor == 1, "yes", "no"),
  levels = c("yes", "no")   # convert back to yes and no
)

tuned_rf <- train(
  term_deposit_factor ~ .,
  data = rfeqtrain_data,
  method = "rf",
  trControl = trainControl(
```

```
    method = "cv",
    number = 5,
    classProbs = TRUE,
    summaryFunction = twoClassSummary
  ),
  metric = "ROC",
  tuneLength = 5
)

tuned_rf
```

```
Random Forest

6496 samples
  10 predictor
   2 classes: 'yes', 'no'

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 5197, 5197, 5196, 5197, 5197
Resampling results across tuning parameters:

  mtry  ROC        Sens       Spec
   2    0.7885651  0.6207633  0.8747692
   8    0.7951654  0.6512595  0.8403077
  15    0.7866247  0.6885348  0.7652308
  22    0.7802309  0.6931563  0.7446154
  29    0.7780476  0.6956193  0.7390769

ROC was used to select the optimal model using the largest value.
The final value used for the model was mtry = 8.
```

```
varImp(tuned_rf)
```

```
rf variable importance

  only 20 most important variables shown (out of 29)

                    Overall
euribor3m           100.000
age                  66.876
```

```
nr.employed          63.408
campaign             32.624
cons.conf.idx        17.360
pdays                15.097
cons.price.idx       14.416
poutcomesuccess       8.745
poutcomenonexistent   8.294
jobblue-collar        7.667
jobtechnician         7.347
jobservices           6.258
monthmay              6.098
jobmanagement         5.600
monthoct              4.873
jobself-employed      3.849
jobentrepreneur       3.818
jobretired            3.075
jobunemployed         2.878
jobstudent            2.839
```

```
forrest_pred_class <- predict(tuned_rf, rfeqtest_data, type = "raw")

table(Predicted = forrest_pred_class, Actual = rfeqtest_data$term_deposit_factor)
```

```
         Actual
Predicted  yes   no
      yes  893  197
      no   501 1193
```

```
mean(forrest_pred_class == rfeqtest_data$term_deposit_factor)
```

```
[1] 0.7492816
```

```
confusionMatrix(forrest_pred_class, rfeqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  yes   no
       yes  893  197
       no   501 1193
```

```
               Accuracy : 0.7493
                 95% CI : (0.7327, 0.7653)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4987

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6406
            Specificity : 0.8583
         Pos Pred Value : 0.8193
         Neg Pred Value : 0.7043
             Prevalence : 0.5007
         Detection Rate : 0.3208
   Detection Prevalence : 0.3915
      Balanced Accuracy : 0.7494

       'Positive' Class : yes
```

```
head(tuned_rf$resample)
```

```
        ROC      Sens      Spec Resample
1 0.7882103 0.6456086 0.8246154    Fold1
2 0.7893777 0.6379045 0.8261538    Fold5
3 0.7977160 0.6563945 0.8446154    Fold2
4 0.8112722 0.6630769 0.8661538    Fold3
5 0.7892509 0.6533128 0.8400000    Fold4
```

```
sd(tuned_rf$resample$ROC)
```

```
[1] 0.009782462
```

```
sd(tuned_rf$resample$Kappa)
```

```
[1] NA
```

```r
sd(tuned_rf$resample$accuracy)
```

```
[1] NA
```

```
 Accuracy : 0.7478
 Kappa : 0.4958
 Sensitivity : 0.6413                          Specificity : 0.8547
  specificity improved and sensitivity decreased.
```

```r
#adaptive boosting model


#ada_partdata <-partdata |>  dplyr::select(-y, -term_deposit, -default)
#ada_partdata[] <- lapply(ada_partdata, function(x) {
#  if (is.character(x)) factor(x) else x
#})

##ada_partdata$default <- factor(ada_partdata$default, levels = c("no", "yes", "unknown"))

set.seed(42)

#idx <- createDataPartition(ada_partdata$term_deposit_factor, p = 0.7, list = FALSE)
#part_train_data <- ada_partdata[idx, ]
#part_test_data  <- ada_partdata[-idx, ]


ada_model <- boosting(
  term_deposit_factor ~ .,
  data = eqtrain_data,
  mfinal = 100,     # number of boosting iterations
  boos = TRUE
)

#ada_model

ada_pred <- predict(ada_model, eqtest_data)

ada_pred$class <- factor(
  ada_pred$class,
  levels = levels(eqtest_data$term_deposit_factor)
)
```

```r
confusionMatrix(ada_pred$class, eqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1195  503
         1  195  891

               Accuracy : 0.7493
                 95% CI : (0.7327, 0.7653)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4987

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8597
            Specificity : 0.6392
         Pos Pred Value : 0.7038
         Neg Pred Value : 0.8204
             Prevalence : 0.4993
         Detection Rate : 0.4292
   Detection Prevalence : 0.6099
      Balanced Accuracy : 0.7494

       'Positive' Class : 0
```

```r
sort(ada_model$importance, decreasing = TRUE)
```

```
   nr.employed          pdays      euribor3m          month       poutcome
    43.1271088     16.3860165     15.7885071     12.5864874      4.8673345
            job cons.price.idx            age       campaign    day_of_week
     2.1103654      2.0172030      1.2179434      0.5869548      0.5002386
       previous  cons.conf.idx        housing        marital
     0.3902263      0.1722818      0.1329486      0.1163839
```

one of the best models so far

```
Accuracy : 0.7493

Kappa : 0.4987

Sensitivity : 0.8597                    Specificity : 0.6392
```

increase mfinal

```
set.seed(42)
adaeqtrain_data <- eqtrain_data
adaeqtest_data <- eqtest_data

adaeqtrain_data$term_deposit_factor <- factor(
  ifelse(eqtrain_data$term_deposit_factor == 1, "yes", "no"),
  levels = c("yes", "no")
)
adaeqtest_data$term_deposit_factor <- factor(
  ifelse(eqtest_data$term_deposit_factor == 1, "yes", "no"),
  levels = c("yes", "no")
)

ada_model <- boosting(
  term_deposit_factor ~ .,
  data = adaeqtrain_data,
  mfinal = 200,     # number of boosting iterations
  boos = TRUE
)

#ada_model

ada_pred <- predict(ada_model, adaeqtest_data)

ada_pred$class <- factor(
  ada_pred$class,
  levels = levels(adaeqtest_data$term_deposit_factor)
)

confusionMatrix(ada_pred$class, adaeqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics
```

```
          Reference
Prediction  yes   no
       yes  880  190
       no   514 1200

               Accuracy : 0.7471
                 95% CI : (0.7305, 0.7632)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4944

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6313
            Specificity : 0.8633
         Pos Pred Value : 0.8224
         Neg Pred Value : 0.7001
             Prevalence : 0.5007
         Detection Rate : 0.3161
   Detection Prevalence : 0.3843
      Balanced Accuracy : 0.7473

       'Positive' Class : yes
```

```r
sort(ada_model$importance, decreasing = TRUE)
```

```
   nr.employed      euribor3m          pdays          month       poutcome
    41.0024894     20.3760138     14.0619534     10.8658134      6.0795533
           job            age cons.price.idx    day_of_week       campaign
     2.6244126      1.2643830      1.2474492      0.7700321      0.7401658
      previous  cons.conf.idx        housing        marital
     0.4339900      0.2841913      0.1260173      0.1235353
```

```
Accuracy : 0.7443 Kappa : 0.4876 Sensitivity : 0.6329
            Specificity : 0.8539
```

lets remove features that have both a low importance

day_of_week, marital, housing, previous

```r
set.seed(42)

ada_model <- boosting(
  term_deposit_factor ~ .,
  data = rfeqtrain_data,
  mfinal = 100,     # number of boosting iterations
  boos = TRUE
)

#ada_model

ada_pred <- predict(ada_model, rfeqtest_data)

ada_pred$class <- factor(
  ada_pred$class,
  levels = levels(rfeqtest_data$term_deposit_factor)
)

confusionMatrix(ada_pred$class, rfeqtest_data$term_deposit_factor)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  yes   no
       yes  875  192
       no   519 1198

               Accuracy : 0.7446
                 95% CI : (0.728, 0.7607)
    No Information Rate : 0.5007
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4894

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6277
            Specificity : 0.8619
         Pos Pred Value : 0.8201
         Neg Pred Value : 0.6977
             Prevalence : 0.5007
         Detection Rate : 0.3143
```

```
   Detection Prevalence : 0.3833
      Balanced Accuracy : 0.7448

       'Positive' Class : yes
```

```r
sort(ada_model$importance, decreasing = TRUE)
```

```
   nr.employed      euribor3m          pdays          month       poutcome
    31.8924797     31.8622077     16.6109313      8.4684527      4.8096019
cons.price.idx            job            age       campaign  cons.conf.idx
     1.8227384      1.7970666      1.5591451      0.8168511      0.3605257
```

```
Accuracy : 0.7446, Sensitivity : 0.6277,                    Specificity : 0.8619
```

```
Kappa : 0.4894
```

Model

Changes to data/experiments

Accuracy

Kappa

specificity

sensitivity

note

Decision Tree

Fulldata minus duration

0.8992

0.2667

0.9898

0.1923

Cp = 0.01

Maxdepth = 30

Decision Tree

Remove highly correlated features, made the number of yes equal to the number of no for the term deposits

0.7338

0.4679

0.8741

0.5940

Decision Tree

Same data set as above smaller complexity parameter

0.7471

0.4944

0.8662

0.6284

Cp=0.001

Decision Tree

Removed additional features day_of_week and month

0.7482

0.4966

0.8719

0.6248

Cp = 0.001

Decision Tree

The max depth or number of splits is reduced effectively pruning the tree

0.7374

0.4751

0.8971

0.5782

Cp=0.001 maxdepth = 5

Model

Changes to data

Accuracy

Kappa

Specificity

Sensitivity

note

Random Forest

Fulldata minus duration

0.8975

0.334

0.9768

0.2785

Ntree = 500

Random Forest

Remove highly correlated features, made the number of yes equal to the number of no for the term deposits

0.7446

0.4894

0.8799

0.6098

Random Forest

Increased the number of trees in the random forest

0.7464

0.493

0.8820

0.6112

Ntree = 1000

Random Forest

Removed more features based on mean decrease accuracy and mean decrease gini

0.745

0.4902

0.8892

0.6011

Ntree = 1000

Random Forest

5 fold cross validation maximize Area Under the ROC Curve

0.7493

0.4987

0.8583

0.6406

Model

Changes to data

Accuracy

Kappa

specificity

sensitivity

note

adaboost

Started with smaller provided data set due to computation time

0.915

0.5472

0.9573

0.5704

Mfinal = 100

adaboost

Remove highly correlated features, made the number of yes equal to the number of no for the term deposits

0.7493

0.4987

0.8597

0.6392

adaboost

Increase number of boosting iterations

0.7471

0.4944

0.8633

0.6313

Mfinal = 200

Adaboost

Removed features that have a low importance: day_of_week, marital, housing, previous

0.7446

0.4894

0.8619

0.6277

The dataset provided contained customer information and campaign information for term deposit subscriptions. Three models were compared. Initially, variance was low when the data was mostly unmodified due to there being a high bias towards none subscribers. The random forest and ensemble methods reduce variance by averaging many trees, while also lowering bias compared to a single tree. AdaBoost reduces bias by focusing on misclassified cases That was most prominent when comparing the models on a data set that lacked feature engineering.

For the experiments explored in this study sampling of the data and the features that the models were trained on were varied. Also, hyperparameters of the models were adjusted to see the impact on the results. Initially models were biased towards nonsubscribers, so adjustments were made to the models and data to increase accuracy of predicting subscribers.

Initially models were trained on the full data set with 70% of the dataset to be used for training and 30% was used for testing minus duration which was dependent on the target value. All models showed high accuracy and a low kappa. The low kappa signifies that the models were not making predictions much better than statistically random chance. The specificity was high due to the models being biased towards the term deposit not being subscribed to by customers. The number of customers not subscribed heavily outweighed the number of customers that did and the models reflected that. Roughly 10% of the customers subscribed, so if the models said that every customer did not subscribe then the model would be 90% accurate. So, the models showed that they were indeed 90% accurate when the data is 90% not subscribed. The kappa was highest for adaboost and lowest for the decision tree. It should be noted that the adaboost was not originally trained on the full 70% of the dataset due to the computation time. Sensitivity was also in the order of adaboost> random forests> decision tree. This indicated that the decision tree was the worst at predicting subscribers.

To improve the model's sensitivity towards correctly identifying subscribers a new data frame was constructed. The data frame contained all the subscribers and an equal number of randomly sampled nonsubscribers was constructed. The new data frame also had a few highly correlated features removed, such as: default, loan, contact, education, and employment variation rate.

70% of this new 50/50 subcriber/nonsubscriber dataset was used to train the models. They were then tested on the remaining 30% of the dataset. Now an accuracy above 50% would indicate the model may be better than random guessing. When the models were trained on the newly sampled data set the bias shifted towards subscribers more, but it did make the model better at differentiating between the classes since they were equal. The kappa increased for the decision tree and the random forest, indicating that they were not getting their accuracy purely by random chance. Sensitivity, the proportion of true positives correctly identified, also increased for all the models. The overall accuracy and specificity decreased due to the data consisting of significantly less nonsubscribers than the original dataset. The accuracy, kappa, and specificity decreased for adaboost when trained under the new dataset, this may indicate that adaboost works better with complexity or biased data.

For the next set of experiments the models were changed individually. Focusing on the decision tree, it appeared that it may have been underfitted due to only using a couple of features. The complexity parameter was lowered 10-fold to see if a more fitted tree could would have more accurate predictions. Accuracy, kappa, and sensitivity all increased. The sensitivity only decreased slightly. The tree had grown significantly more complex by changing this parameter. In order to reduce complexity a couple of more features were removed: day_of_week and month. Which had a minimal effect on the model. To avoid any overfitting and reduce complexity the tree was pruned to have a max depth of 5 sets of splits. This resulted in a model that had a lower sensitivity or true positive outcomes. The most complex model had the best results when it came to differentiating subscribers from nonsubscribers.

For the random forest the number of trees in the model increased 2-fold to see if the added complexity would improve the model performance. The model did have a slight increase

in all metrics, however with significantly increased model building time. The mean decrease gini and mean decrease accuracy charts were compared. Features that were both low in decrease in accuracy and decrease in gini (class purity due to split) were then removed from the model: day_of_week, marital, housing, and previous. Model performance mostly remained the same after removal of these features, indicating they were not necessary. Finally the random forest was tuned using a 5 fold cross validation. From this there was a decent increase in finding true positives with not much change in accuracy and kappa. Specificity decreased significantly though.

The first unique adaboost experiment was to increase the number of boosting iterations 2-fold which also slightly decreased the sensitivity. Looking at the ranked importance of the features day_of_week, marital, housing, and previous were removed. The removal of these features decreased the performance of the model slightly.

It appears that decision trees, adaboost and random forests are very similar in performance especially if the random forest is tuned. It appears that in order to improve the performance of the models additional feature engineering may make a dramatic difference. Since they were trained on a similar and even the same data set the performance is not as dramatic as I would have thought. The decision tree started off as more biased towards nonsubscribers when compared to random forests and adaboost, but its performance improved by making the class sizes equivalent. There are models with very similar metrics between all three. Additional tuning the models would be needed to truly strike a balance between detecting positives and negatives. I would recommend using the random forest due to it achieving the highest sensitivity after tuning. Adaboost is not far behind and may potentially be better with better hyperparameter selection. It appears that the dataset is complex and it models it the unmodified data the best. A complex decision tree performed remarkably similarly to the other two models. With an updating dataset adaboost would be should be better to handle the changes without much revision.

reference:

Input variables: # bank client data: 1 - **age** (numeric) 2 - **job : type of job** (categorical: “admin.”,“blue-collar”,“entrepreneur”,“housemaid”,“management”,“retired”,“self-employed”,“services”,“student”,“technician”,“unemployed”,“unknown”) 3 - **marital : marital status** (categorical: “divorced”,“married”,“single”,“unknown”; note: “divorced” means divorced or widowed) 4 - **education** (categorical: “basic.4y”,“basic.6y”,“basic.9y”,“high.school”,“illiterate”,“profes
5 - **default: has credit in default**? (categorical: “no”,“yes”,“unknown”) 6 - **housing: has housing loan**? (categorical: “no”,“yes”,“unknown”) 7 - **loan: has personal loan**? (categorical: “no”,“yes”,“unknown”) # related with the last contact of the current campaign: 8 - **contact: contact communication type** (categorical: “cellular”,“telephone”) 9 - **month: last contact month of year** (categorical: “jan”, “feb”, “mar”, …, “nov”, “dec”) 10 - **day_of_week: last contact day of the week** (categorical: “mon”,“tue”,“wed”,“thu”,“fri”) **11 - duration: last contact duration, in seconds (numeric). Important note: this attribute highly affects the output target (e.g., if duration=0 then y=“no”). Yet, the duration is not known before a call is performed. Also, after the end**

of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model. # other attributes: 12 - **campaign: number of contacts performed during this campaign and for this client** (numeric, includes last contact) 13 - **pdays: number of days that passed by after the client was last contacted from a previous campaign** (numeric; **999 means client was not previously contacted**) 14 - **previous: number of contacts performed before this campaign and for this client (numeric)** 15 - **poutcome: outcome of the previous marketing campaign** (categorical: "failure","nonexistent","success") # social and economic context attributes 16 - **emp.var.rate: employment variation rate** - quarterly indicator (numeric) 17 - **cons.price.idx: consumer price index** - monthly indicator (numeric)
18 - **cons.conf.idx: consumer confidence index** - monthly indicator (numeric)
19 - **euribor3m: euribor 3 month rate** - daily indicator (numeric) 20 - **nr.employed: number of employees** - quarterly indicator (numeric)

Output variable (desired target): 21 - **y** - has the client subscribed a term deposit? (binary: "yes","no")