DATA 622 Assignment 3

CUNY: Spring 2021

Philip Tanofsky

30 March 2021

```r
# Import required R libraries
library(vcd)
library(caret)
#library(MASS)
#library(ggplot2)
#library(mvtnorm)
#library(e1071)
#library(klaR)
#library(pROC)
#library(corrplot)
theme_set(theme_classic())

library(tidyverse)
library(tidymodels)
library(skimr)
library(baguette)
library(future)
library(xgboost)
library(vip)
library(rpart.plot)
```

```r
# Read in loan approval csv
data <- read.csv("https://raw.githubusercontent.com/ptanofsky/data622/main/assignment03/Loan_approval.c

data$Credit_History <- as.factor(data$Credit_History)

data$Total_Income <- data$ApplicantIncome + data$CoapplicantIncome

data$LoanAmt_Per_Month <- data$LoanAmount / data$Loan_Amount_Term

data$Income_To_LoanAmt <- data$Total_Income / data$LoanAmount

data$Income_To_LoanAmtMonth <- data$Total_Income / data$LoanAmt_Per_Month

summary(data)
```

```
##       Loan_ID         Gender     Married   Dependents         Education
##   LP001002:  1          : 13        :  3      : 15     Graduate    :480
##   LP001003:  1    Female:112    No :213    0 :345     Not Graduate:134
```

```
##  LP001005:  1   Male  :489   Yes:398   1 :102
##  LP001006:  1                           2 :101
##  LP001008:  1                           3+: 51
##  LP001011:  1
##  (Other) :608
##  Self_Employed ApplicantIncome CoapplicantIncome   LoanAmount
##     : 32       Min.   :  150   Min.   :    0     Min.   :  9.0
##  No :500       1st Qu.: 2878   1st Qu.:    0     1st Qu.:100.0
##  Yes: 82       Median : 3812   Median : 1188     Median :128.0
##                Mean   : 5403   Mean   : 1621     Mean   :146.4
##                3rd Qu.: 5795   3rd Qu.: 2297     3rd Qu.:168.0
##                Max.   :81000   Max.   :41667     Max.   :700.0
##                                                  NA's   :22
##  Loan_Amount_Term Credit_History   Property_Area Loan_Status  Total_Income
##  Min.   : 12      0    : 89       Rural    :179   N:192       Min.   : 1442
##  1st Qu.:360      1    :475       Semiurban:233   Y:422       1st Qu.: 4166
##  Median :360      NA's: 50        Urban    :202               Median : 5416
##  Mean   :342                                                  Mean   : 7025
##  3rd Qu.:360                                                  3rd Qu.: 7522
##  Max.   :480                                                  Max.   :81000
##  NA's   :14
##  LoanAmt_Per_Month Income_To_LoanAmt Income_To_LoanAmtMonth
##  Min.   :0.0250    Min.   : 12.09    Min.   :   808.5
##  1st Qu.:0.2861    1st Qu.: 35.53    1st Qu.: 12233.0
##  Median :0.3653    Median : 41.43    Median : 14469.3
##  Mean   :0.4803    Mean   : 51.23    Mean   : 17241.8
##  3rd Qu.:0.5139    3rd Qu.: 51.78    3rd Qu.: 17992.4
##  Max.   :9.2500    Max.   :396.37    Max.   :142692.0
##  NA's   :36        NA's   :22        NA's   :36
```

```r
dim(data)
```

```
## [1] 614  17
```

Dimensions: 614 observations

13 columns

All columns factor except:

ApplicationIncome: int

CoapplicantIncome: num LoanAmount: int Loan_Amount_Term: int Credit_History: int, should probably be factor

Loan_ID: Unique identifier Gender: Female|Male Married: No|Yes

```r
# Count penguins for each loan status / gender
ggplot(data, aes(x = Gender, fill = Loan_Status)) +
  geom_bar(alpha = 0.8) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4"),
                    guide = F) +
  theme_minimal() +
  facet_wrap(~Loan_Status, ncol = 1) +
  coord_flip()

mosaic(~ Gender + Loan_Status, data = data)
```

```r
# Count penguins for each loan status / married
ggplot(data, aes(x = Married, fill = Loan_Status)) +
  geom_bar(alpha = 0.8) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4"),
                    guide = F) +
  theme_minimal() +
  facet_wrap(~Loan_Status, ncol = 1) +
  coord_flip()

mosaic(~ Married + Loan_Status, data = data)
```

```r
# Count penguins for each loan status / dependents
ggplot(data, aes(x = Dependents, fill = Loan_Status)) +
  geom_bar(alpha = 0.8) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4"),
                    guide = F) +
  theme_minimal() +
  facet_wrap(~Loan_Status, ncol = 1) +
  coord_flip()

mosaic(~ Dependents + Loan_Status, data = data)
```

```r
# Count penguins for each loan status / Education
ggplot(data, aes(x = Education, fill = Loan_Status)) +
  geom_bar(alpha = 0.8) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4"),
                    guide = F) +
  theme_minimal() +
  facet_wrap(~Loan_Status, ncol = 1) +
  coord_flip()

mosaic(~ Education + Loan_Status, data = data)
```

```r
# Count penguins for each loan status / Self_Employed
ggplot(data, aes(x = Self_Employed, fill = Loan_Status)) +
  geom_bar(alpha = 0.8) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4"),
                    guide = F) +
  theme_minimal() +
  facet_wrap(~Loan_Status, ncol = 1) +
  coord_flip()

mosaic(~ Self_Employed + Loan_Status, data = data)
```

```r
# Count penguins for each loan status / Credit_History
ggplot(data, aes(x = Credit_History, fill = Loan_Status)) +
  geom_bar(alpha = 0.8) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4"),
                    guide = F) +
  theme_minimal() +
  facet_wrap(~Loan_Status, ncol = 1) +
  coord_flip()
```

```r
mosaic(~ Credit_History + Loan_Status, data = data)
```

```r
# Count penguins for each loan status / Property_Area
ggplot(data, aes(x = Property_Area, fill = Loan_Status)) +
  geom_bar(alpha = 0.8) +
  scale_fill_manual(values = c("darkorange", "purple", "cyan4"),
                    guide = F) +
  theme_minimal() +
  facet_wrap(~Loan_Status, ncol = 1) +
  coord_flip()

mosaic(~ Property_Area + Loan_Status, data = data)
```

```r
# Overlayed density plots
featurePlot(x = data[, 7:10],
            y = data$Loan_Status,
            plot = "density",
            # Pass in options to xyplot() to
            # make it prettier
            scales = list(x = list(relation="free"),
                          y = list(relation="free")),
            adjust = 1.5,
            pch = "|",
            layout = c(2, 2),
            auto.key = list(columns = 3))

# Overlayed density plots
featurePlot(x = data[, 14:17],
            y = data$Loan_Status,
            plot = "density",
            # Pass in options to xyplot() to
            # make it prettier
            scales = list(x = list(relation="free"),
                          y = list(relation="free")),
            adjust = 1.5,
            pch = "|",
            layout = c(2, 2),
            auto.key = list(columns = 3))
```

```r
# Use featurePlot
# https://topepo.github.io/caret/visualizations.html

# Scatterplot
featurePlot(x = data[, 7:10],
            y = data$Loan_Status,
            plot = "pairs",
            # Add a key at the top
            auto.key = list(columns = 3))

featurePlot(x = data[, 14:17],
            y = data$Loan_Status,
            plot = "pairs",
```

```r
            # Add a key at the top
            auto.key = list(columns = 3))
```

```r
featurePlot(x = data[, 7:10],
            y = data$Loan_Status,
            plot = "box",
            ## Pass in options to bwplot()
            scales = list(y = list(relation="free"),
                          x = list(rot = 90)),
            layout = c(2,2),
            auto.key = list(columns = 2))

featurePlot(x = data[, 14:17],
            y = data$Loan_Status,
            plot = "box",
            ## Pass in options to bwplot()
            scales = list(y = list(relation="free"),
                          x = list(rot = 90)),
            layout = c(2,2),
            auto.key = list(columns = 2))
```

Decision Tree for Loan Approval data

decision_tree() function from tidymodels

3 hyperparameters - cost_complexity - tree_depth - min_n

```r
# https://www.gmudatamining.com/lesson-13-r-tutorial.html

lap_data <- data
summary(data)
```

```
##       Loan_ID       Gender    Married   Dependents        Education
##  LP001002:  1         : 13      :  3     : 15     Graduate    :480
##  LP001003:  1   Female:112   No :213   0 :345     Not Graduate:134
##  LP001005:  1   Male  :489   Yes:398   1 :102
##  LP001006:  1                          2 :101
##  LP001008:  1                          3+: 51
##  LP001011:  1
##  (Other) :608
##  Self_Employed ApplicantIncome CoapplicantIncome   LoanAmount
##     : 32       Min.   :  150   Min.   :    0     Min.   :  9.0
##  No :500       1st Qu.: 2878   1st Qu.:    0     1st Qu.:100.0
##  Yes: 82       Median : 3812   Median : 1188     Median :128.0
##                Mean   : 5403   Mean   : 1621     Mean   :146.4
##                3rd Qu.: 5795   3rd Qu.: 2297     3rd Qu.:168.0
##                Max.   :81000   Max.   :41667     Max.   :700.0
##                                                  NA's   :22
##  Loan_Amount_Term Credit_History   Property_Area Loan_Status  Total_Income
##  Min.   : 12      0    : 89       Rural    :179  N:192       Min.   : 1442
##  1st Qu.:360      1    :475       Semiurban:233  Y:422       1st Qu.: 4166
##  Median :360      NA's: 50        Urban    :202              Median : 5416
```

```
##  Mean   :342                                                 Mean   : 7025
##  3rd Qu.:360                                                 3rd Qu.: 7522
##  Max.   :480                                                 Max.   :81000
##  NA's   :14
##  LoanAmt_Per_Month Income_To_LoanAmt Income_To_LoanAmtMonth
##  Min.   :0.0250    Min.   : 12.09    Min.   :   808.5
##  1st Qu.:0.2861    1st Qu.: 35.53    1st Qu.: 12233.0
##  Median :0.3653    Median : 41.43    Median : 14469.3
##  Mean   :0.4803    Mean   : 51.23    Mean   : 17241.8
##  3rd Qu.:0.5139    3rd Qu.: 51.78    3rd Qu.: 17992.4
##  Max.   :9.2500    Max.   :396.37    Max.   :142692.0
##  NA's   :36        NA's   :22        NA's   :36
```

```
# Data splitting
set.seed(1234)

lap_data_split <- initial_split(lap_data, prop=0.75,
                              strata = Loan_Status)

lap_training <- lap_data_split %>% training()

lap_test <- lap_data_split %>% testing()

set.seed(1234)
lap_folds <- vfold_cv(lap_training, v=3)
```

```
# Data exploration
# https://bcullen.rbind.io/post/2020-06-02-tidymodels-decision-tree-learning-in-r/

# Need to fix
lap_data %>%
  select(-contains("ID")) %>%
#  modify_if(is.character, as.factor) %>%
  skim() %>%
  select()
```

```
## # A tibble: 16 x 0
```

```
# Feature Engineering

lap_recipe <- recipe(Loan_Status ~ ., data = lap_training) %>%
  step_YeoJohnson(all_numeric(), -all_outcomes()) %>%
  step_normalize(all_numeric(), -all_outcomes()) %>%
  step_dummy(all_nominal(), -all_outcomes())

lap_recipe %>%
  prep() %>%
  bake(new_data = lap_training)
```

```
## # A tibble: 461 x 636
##    ApplicantIncome CoapplicantIncome LoanAmount Loan_Amount_Term Total_Income
##              <dbl>             <dbl>      <dbl>            <dbl>        <dbl>
##  1           0.639             -1.08      NA               0.174        0.173
```

```
##  2          0.249              0.771     0.0185            0.174       0.253
##  3         -0.503             -1.08     -1.28              0.174      -1.44
##  4         -0.793              0.905    -0.110             0.174      -0.180
##  5          0.678             -1.08      0.212             0.174       0.223
##  6          0.519              1.08      1.52              0.174       1.04
##  7         -0.999              0.772    -0.571             0.174      -0.766
##  8          0.0207             0.774     0.566             0.174       0.0596
##  9          1.71               1.39      2.08              0.174       2.13
## 10         -0.382              0.546    -1.16              0.174      -0.733
## # ... with 451 more rows, and 631 more variables: LoanAmt_Per_Month <dbl>,
## #   Income_To_LoanAmt <dbl>, Income_To_LoanAmtMonth <dbl>, Loan_Status <fct>,
## #   Loan_ID_LP001003 <dbl>, Loan_ID_LP001005 <dbl>, Loan_ID_LP001006 <dbl>,
## #   Loan_ID_LP001008 <dbl>, Loan_ID_LP001011 <dbl>, Loan_ID_LP001013 <dbl>,
## #   Loan_ID_LP001014 <dbl>, Loan_ID_LP001018 <dbl>, Loan_ID_LP001020 <dbl>,
## #   Loan_ID_LP001024 <dbl>, Loan_ID_LP001027 <dbl>, Loan_ID_LP001028 <dbl>,
## #   Loan_ID_LP001029 <dbl>, Loan_ID_LP001030 <dbl>, Loan_ID_LP001032 <dbl>,
## #   Loan_ID_LP001034 <dbl>, Loan_ID_LP001036 <dbl>, Loan_ID_LP001038 <dbl>,
## #   Loan_ID_LP001041 <dbl>, Loan_ID_LP001043 <dbl>, Loan_ID_LP001046 <dbl>,
## #   Loan_ID_LP001047 <dbl>, Loan_ID_LP001050 <dbl>, Loan_ID_LP001052 <dbl>,
## #   Loan_ID_LP001066 <dbl>, Loan_ID_LP001068 <dbl>, Loan_ID_LP001073 <dbl>,
## #   Loan_ID_LP001086 <dbl>, Loan_ID_LP001087 <dbl>, Loan_ID_LP001091 <dbl>,
## #   Loan_ID_LP001095 <dbl>, Loan_ID_LP001097 <dbl>, Loan_ID_LP001098 <dbl>,
## #   Loan_ID_LP001100 <dbl>, Loan_ID_LP001106 <dbl>, Loan_ID_LP001109 <dbl>,
## #   Loan_ID_LP001112 <dbl>, Loan_ID_LP001114 <dbl>, Loan_ID_LP001116 <dbl>,
## #   Loan_ID_LP001119 <dbl>, Loan_ID_LP001120 <dbl>, Loan_ID_LP001123 <dbl>,
## #   Loan_ID_LP001131 <dbl>, Loan_ID_LP001136 <dbl>, Loan_ID_LP001137 <dbl>,
## #   Loan_ID_LP001138 <dbl>, Loan_ID_LP001144 <dbl>, Loan_ID_LP001146 <dbl>,
## #   Loan_ID_LP001151 <dbl>, Loan_ID_LP001155 <dbl>, Loan_ID_LP001157 <dbl>,
## #   Loan_ID_LP001164 <dbl>, Loan_ID_LP001179 <dbl>, Loan_ID_LP001186 <dbl>,
## #   Loan_ID_LP001194 <dbl>, Loan_ID_LP001195 <dbl>, Loan_ID_LP001197 <dbl>,
## #   Loan_ID_LP001198 <dbl>, Loan_ID_LP001199 <dbl>, Loan_ID_LP001205 <dbl>,
## #   Loan_ID_LP001206 <dbl>, Loan_ID_LP001207 <dbl>, Loan_ID_LP001213 <dbl>,
## #   Loan_ID_LP001222 <dbl>, Loan_ID_LP001225 <dbl>, Loan_ID_LP001228 <dbl>,
## #   Loan_ID_LP001233 <dbl>, Loan_ID_LP001238 <dbl>, Loan_ID_LP001241 <dbl>,
## #   Loan_ID_LP001243 <dbl>, Loan_ID_LP001245 <dbl>, Loan_ID_LP001248 <dbl>,
## #   Loan_ID_LP001250 <dbl>, Loan_ID_LP001253 <dbl>, Loan_ID_LP001255 <dbl>,
## #   Loan_ID_LP001256 <dbl>, Loan_ID_LP001259 <dbl>, Loan_ID_LP001263 <dbl>,
## #   Loan_ID_LP001264 <dbl>, Loan_ID_LP001265 <dbl>, Loan_ID_LP001266 <dbl>,
## #   Loan_ID_LP001267 <dbl>, Loan_ID_LP001273 <dbl>, Loan_ID_LP001275 <dbl>,
## #   Loan_ID_LP001279 <dbl>, Loan_ID_LP001280 <dbl>, Loan_ID_LP001282 <dbl>,
## #   Loan_ID_LP001289 <dbl>, Loan_ID_LP001310 <dbl>, Loan_ID_LP001316 <dbl>,
## #   Loan_ID_LP001318 <dbl>, Loan_ID_LP001319 <dbl>, Loan_ID_LP001322 <dbl>,
## #   Loan_ID_LP001325 <dbl>, Loan_ID_LP001326 <dbl>, Loan_ID_LP001327 <dbl>, ...
```

```r
# Define model
tree_model <- decision_tree(cost_complexity = tune(),
                            tree_depth = tune(),
                            min_n = tune()) %>%
  set_engine('rpart') %>%
  set_mode('classification')
```

```r
# Define workflow
tree_workflow <- workflow() %>%
  add_model(tree_model) %>%
```

```r
  add_recipe(lap_recipe)
```

```r
# Create a grid of hyperparemeter values to test
tree_grid <- grid_regular(cost_complexity(),
                          tree_depth(),
                          min_n(),
                          levels = 2)
# view grid
tree_grid
```

```
## # A tibble: 8 x 3
##   cost_complexity tree_depth min_n
##             <dbl>      <int> <int>
## 1    0.0000000001          1     2
## 2    0.1                   1     2
## 3    0.0000000001         15     2
## 4    0.1                  15     2
## 5    0.0000000001          1    40
## 6    0.1                   1    40
## 7    0.0000000001         15    40
## 8    0.1                  15    40
```

```r
# Tune decision tree workflow
set.seed(1234)

tree_tuning <- tree_workflow %>%
  tune_grid(resamples = lap_folds,
            grid = tree_grid)

tree_tuning %>% show_best('roc_auc')
```

```
## # A tibble: 5 x 9
##   cost_complexity tree_depth min_n .metric .estimator  mean     n std_err
##             <dbl>      <int> <int> <chr>   <chr>      <dbl> <int>   <dbl>
## 1    0.0000000001         15    40 roc_auc binary     0.720     3  0.0164
## 2    0.0000000001          1     2 roc_auc binary     0.704     3  0.0140
## 3    0.1                   1     2 roc_auc binary     0.704     3  0.0140
## 4    0.1                  15     2 roc_auc binary     0.704     3  0.0140
## 5    0.0000000001          1    40 roc_auc binary     0.704     3  0.0140
## # ... with 1 more variable: .config <fct>
```

```r
# Select best model based on roc_auc
best_tree <- tree_tuning %>%
  select_best(metric = 'roc_auc')

# view the best tree parameters
best_tree
```

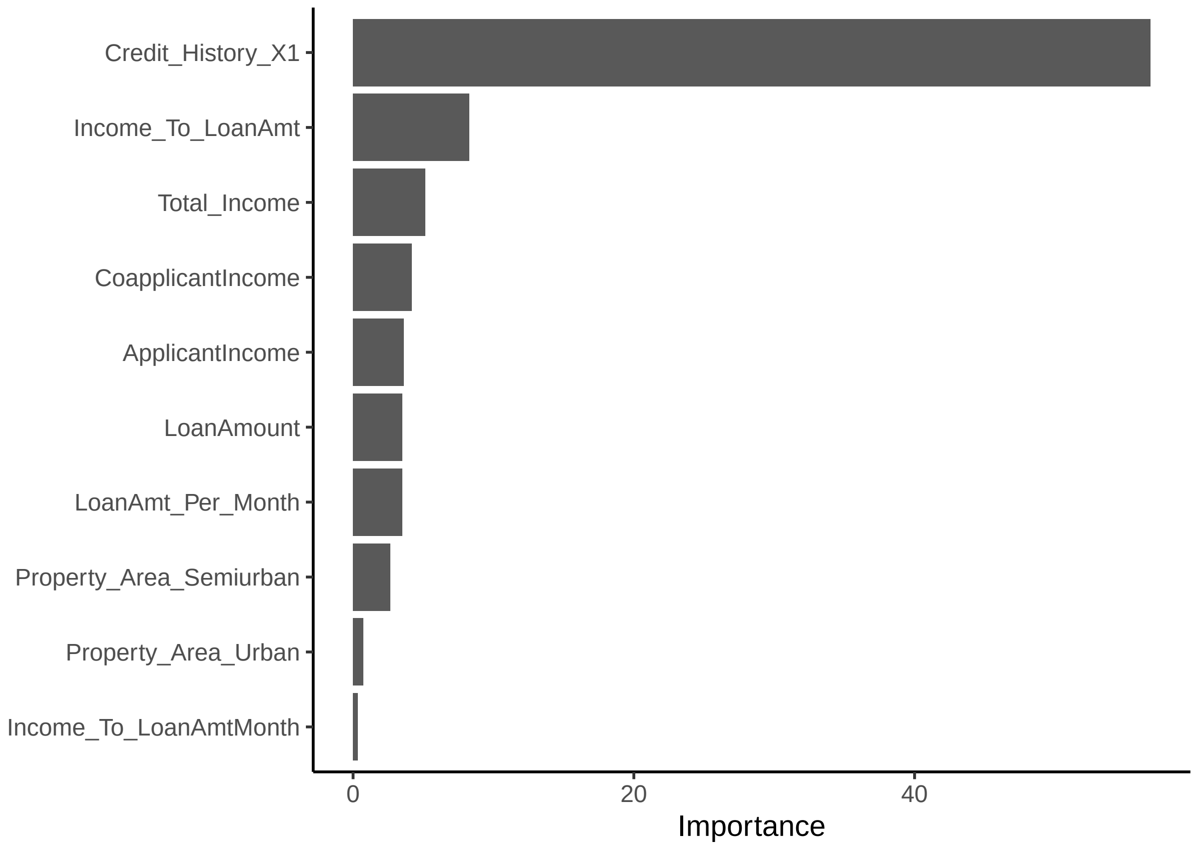
```
## # A tibble: 1 x 4
##   cost_complexity tree_depth min_n .config
##             <dbl>      <int> <int> <fct>
## 1    0.0000000001         15    40 Preprocessor1_Model7
```

```r
# finalize workflow
final_tree_workflow <- tree_workflow %>%
  finalize_workflow(best_tree)
```

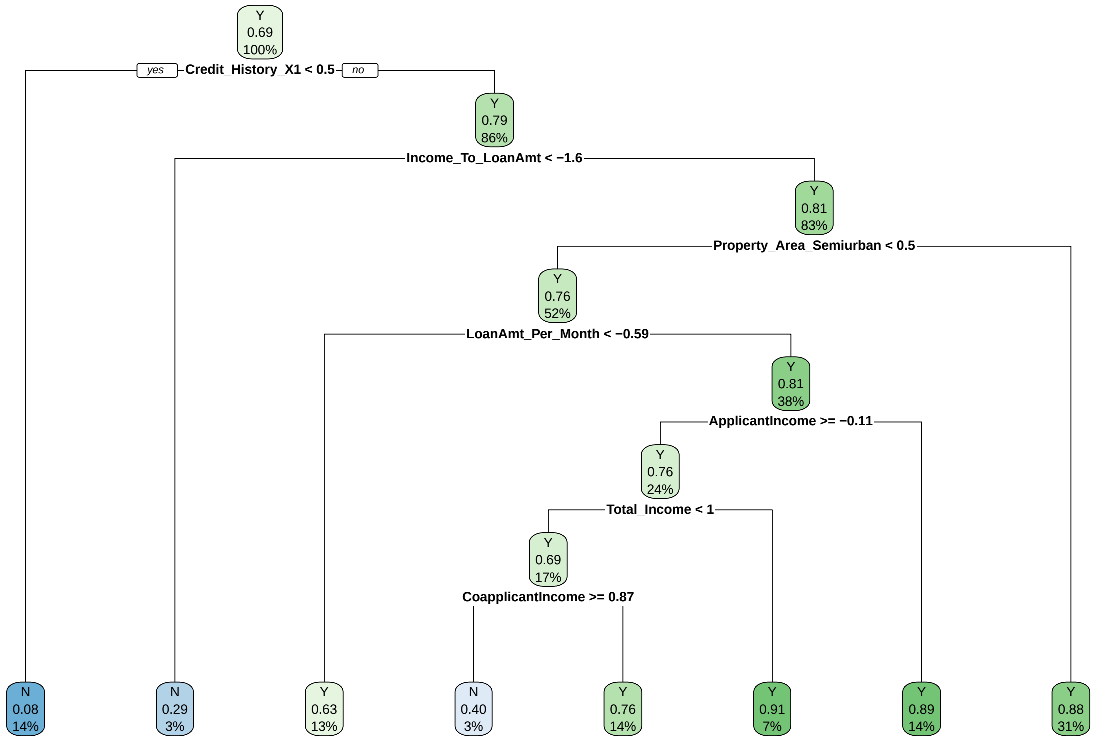
```r
# fit the model
tree_wf_fit <- final_tree_workflow %>%
  fit(data = lap_training)
```

```r
tree_fit <- tree_wf_fit %>%
  pull_workflow_fit()

vip(tree_fit)
```




```r
rpart.plot(tree_fit$fit, roundint=FALSE)
```

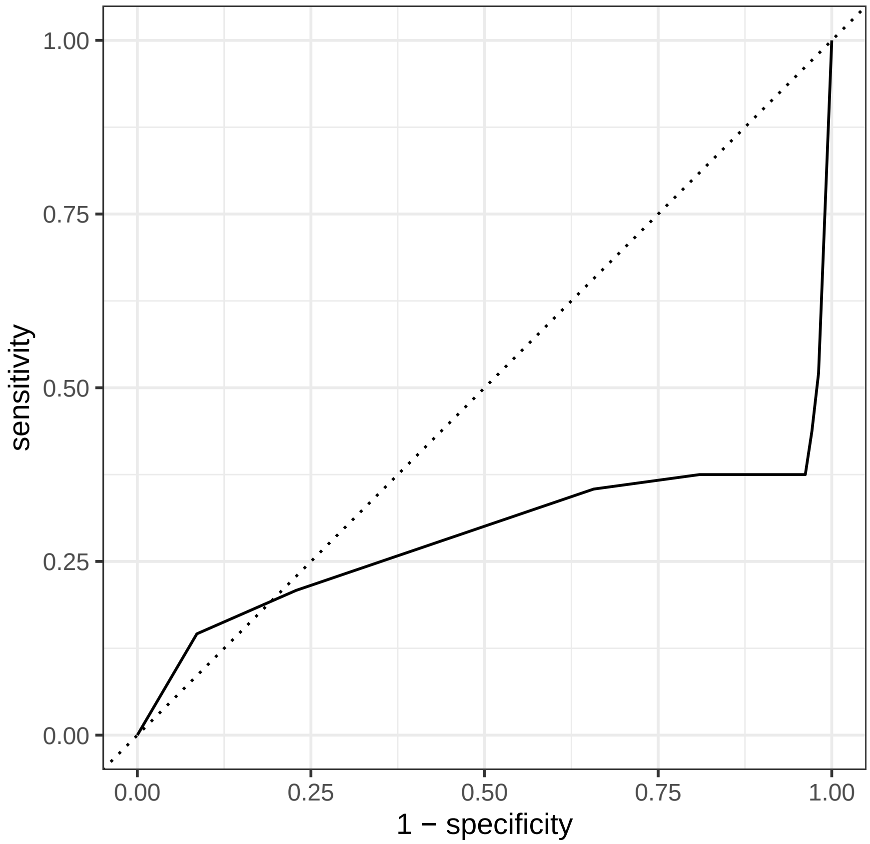



```r
# train and evaluate
tree_last_fit <- final_tree_workflow %>%
  last_fit(lap_data_split)

tree_last_fit %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <fct>
## 1 accuracy binary         0.856 Preprocessor1_Model1
## 2 roc_auc  binary         0.712 Preprocessor1_Model1
```

```r
tree_last_fit %>% collect_predictions() %>%
  roc_curve(truth = Loan_Status, estimate = .pred_Y) %>%
  autoplot()
```




```
tree_predictions <- tree_last_fit %>% collect_predictions()

conf_mat(tree_predictions, truth = Loan_Status, estimate = .pred_class)
```

```
##           Truth
## Prediction   N   Y
##          N  30   4
##          Y  18 101
```

Gradient Boosting for Loan Approval data

```
# https://bcullen.rbind.io/post/2020-06-02-tidymodels-decision-tree-learning-in-r/
# Section Boosted Trees

# Specify the model
mod_boost <- boost_tree() %>%
  set_engine("xgboost", nthreads = parallel::detectCores()) %>%
  set_mode("classification")
```

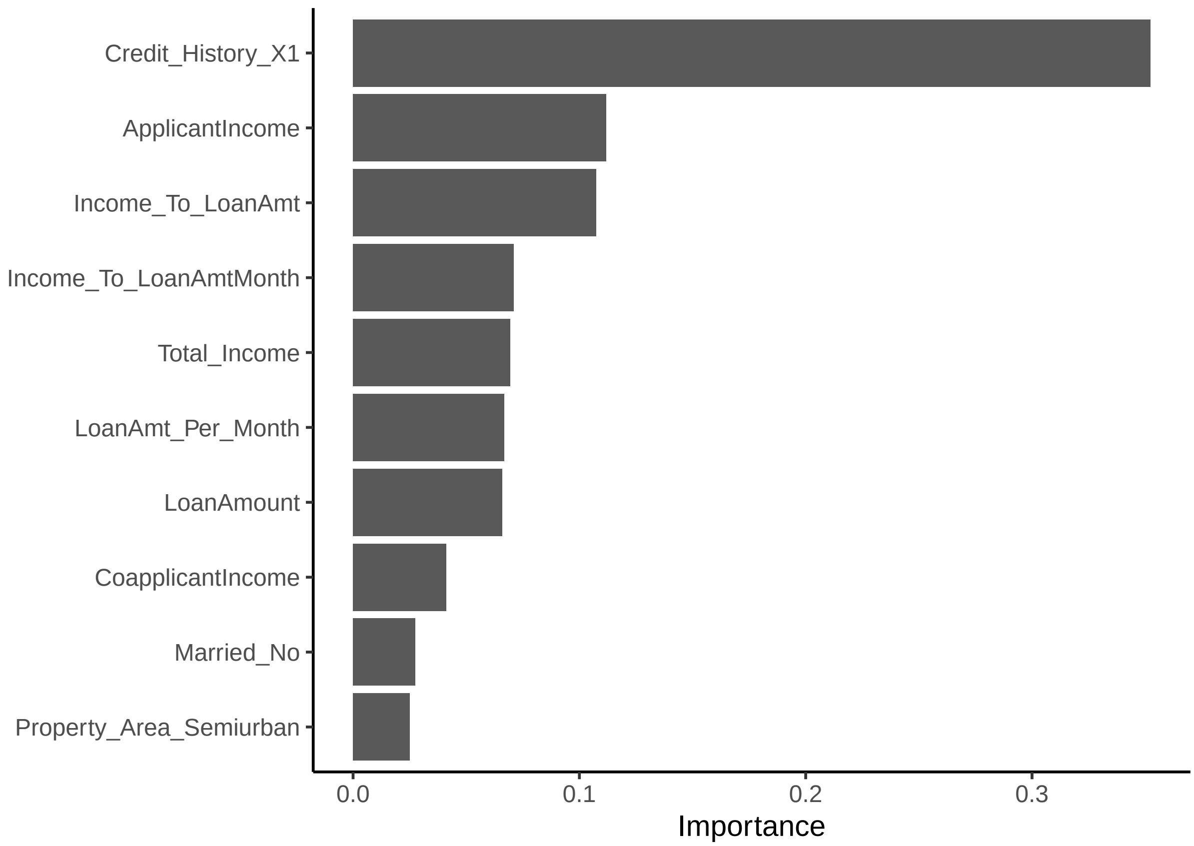
```
# Create workflow
boost_workflow <- workflow() %>%
  add_recipe(lap_recipe) %>%
  add_model(mod_boost)
```

```r
# fit the model
boost_wf_fit <- boost_workflow %>%
  fit(data = lap_training)
```

```
## [14:30:33] WARNING: amalgamation/../src/learner.cc:541:
## Parameters: { nthreads } might not be used.
##
##   This may not be accurate due to some parameters are only used in language bindings but
##   passed down to XGBoost core.  Or some parameters are not used but slip through this
##   verification. Please open an issue if you find above cases.
##
##
## [14:30:33] WARNING: amalgamation/../src/learner.cc:1061: Starting in XGBoost 1.3.0, the default evalu
```

```r
boost_fit <- boost_wf_fit %>%
  pull_workflow_fit()

vip(boost_fit)
```

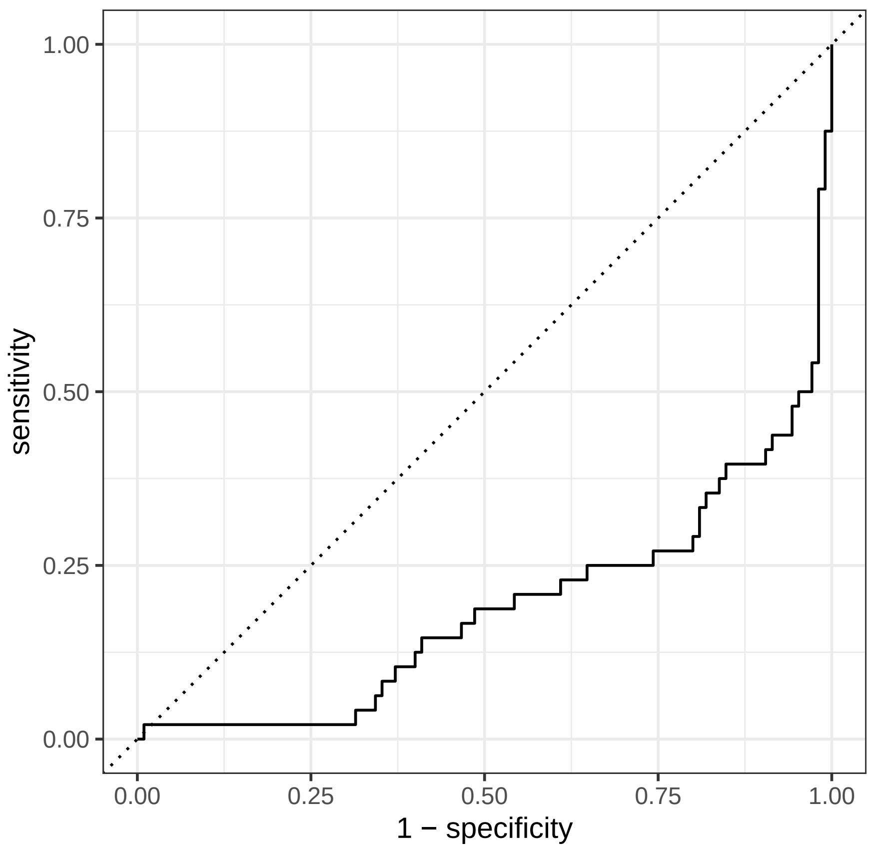



```r
# train and evaluate
boost_last_fit <- boost_workflow %>%
  last_fit(lap_data_split)

boost_last_fit %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <fct>
## 1 accuracy binary         0.817 Preprocessor1_Model1
## 2 roc_auc  binary         0.812 Preprocessor1_Model1
```

```r
boost_last_fit %>% collect_predictions() %>%
  roc_curve(truth = Loan_Status, estimate = .pred_Y) %>%
  autoplot()
```




```r
boost_predictions <- boost_last_fit %>% collect_predictions()

boost_predictions
```

```
## # A tibble: 153 x 7
##    id               .pred_N .pred_Y  .row .pred_class Loan_Status .config
##    <chr>              <dbl>   <dbl> <int> <fct>       <fct>       <fct>
##  1 train/test sp~  0.926   0.0742     8 N           N           Preprocessor1_M~
##  2 train/test sp~  0.0563  0.944     13 Y           Y           Preprocessor1_M~
##  3 train/test sp~  0.124   0.876     14 Y           N           Preprocessor1_M~
##  4 train/test sp~  0.0183  0.982     27 Y           Y           Preprocessor1_M~
##  5 train/test sp~  0.0890  0.911     30 Y           Y           Preprocessor1_M~
##  6 train/test sp~  0.307   0.693     33 Y           N           Preprocessor1_M~
##  7 train/test sp~  0.0480  0.952     38 Y           Y           Preprocessor1_M~
##  8 train/test sp~  0.444   0.556     43 Y           Y           Preprocessor1_M~
```

```
##  9 train/test sp~  0.464   0.536     45 Y           Y           Preprocessor1_M~
## 10 train/test sp~  0.191   0.809     46 Y           Y           Preprocessor1_M~
## # ... with 143 more rows
```

```r
conf_mat(boost_predictions, truth = Loan_Status, estimate = .pred_class)
```

```
##           Truth
## Prediction  N  Y
##          N 27  7
##          Y 21 98
```

```r
tree_last_fit <- last_fit(
  tree_workflow,
  split = lap_data_split
)

boost_last_fit <- last_fit(
  boost_workflow,
  split = lap_data_split
)

tree_last_fit %>% collect_metrics()
```

```
## NULL
```

```r
boost_last_fit %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <fct>
## 1 accuracy binary         0.817 Preprocessor1_Model1
## 2 roc_auc  binary         0.812 Preprocessor1_Model1
```