hw4

Asahi (Ash) Kuroki

11/22/2021

```
### load required packages
library("caret")

## Loading required package: ggplot2

## Loading required package: lattice

library("e1071")
library("rpart")
library("rpart.plot")
```

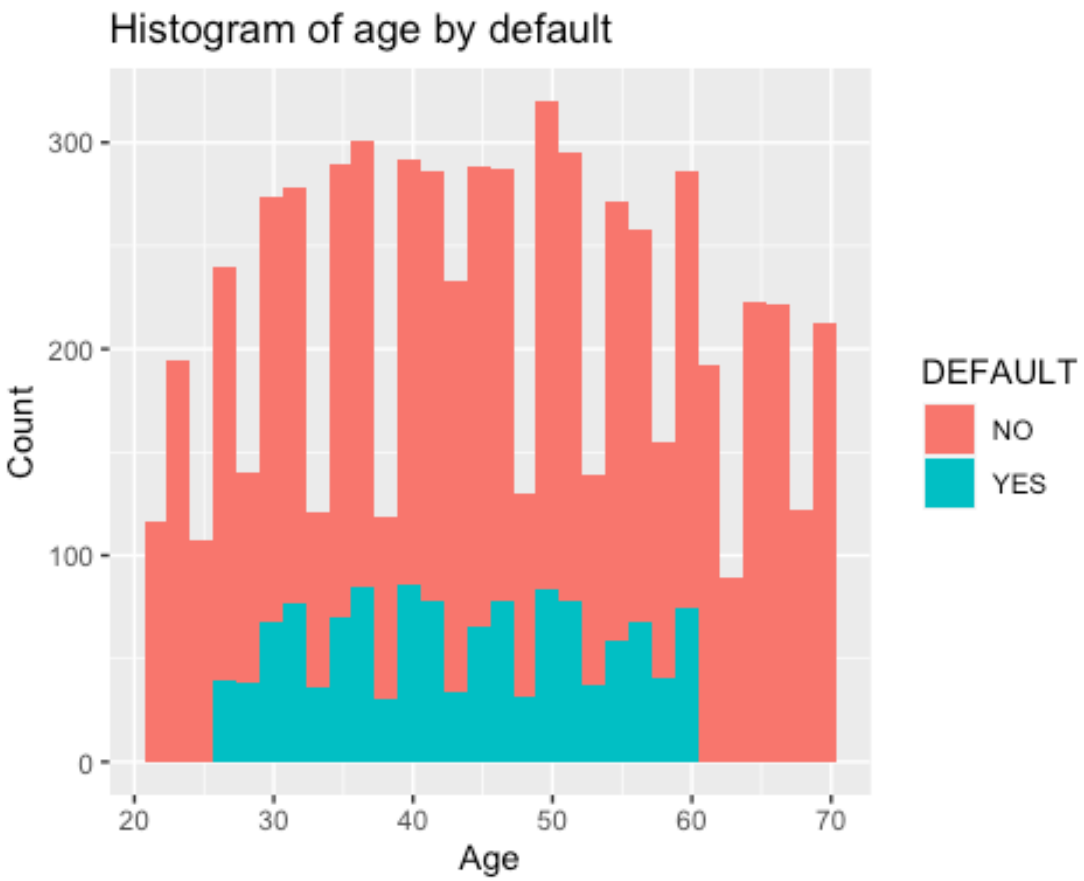
Analysis
Data Preparation / Visualization
a) load the data

```
credit_data <- read.csv('credit.csv')
credit_table <- table(credit_data$DEFAULT)
credit_table

##
##   NO  YES
## 5222 1258

prob <- sum(credit_table["YES"]) / sum(credit_table)
prob

## [1] 0.1941358
```

Probability of default = YES: 19.41%

b) Use ggplot2 to create a histogram of Age, by Default

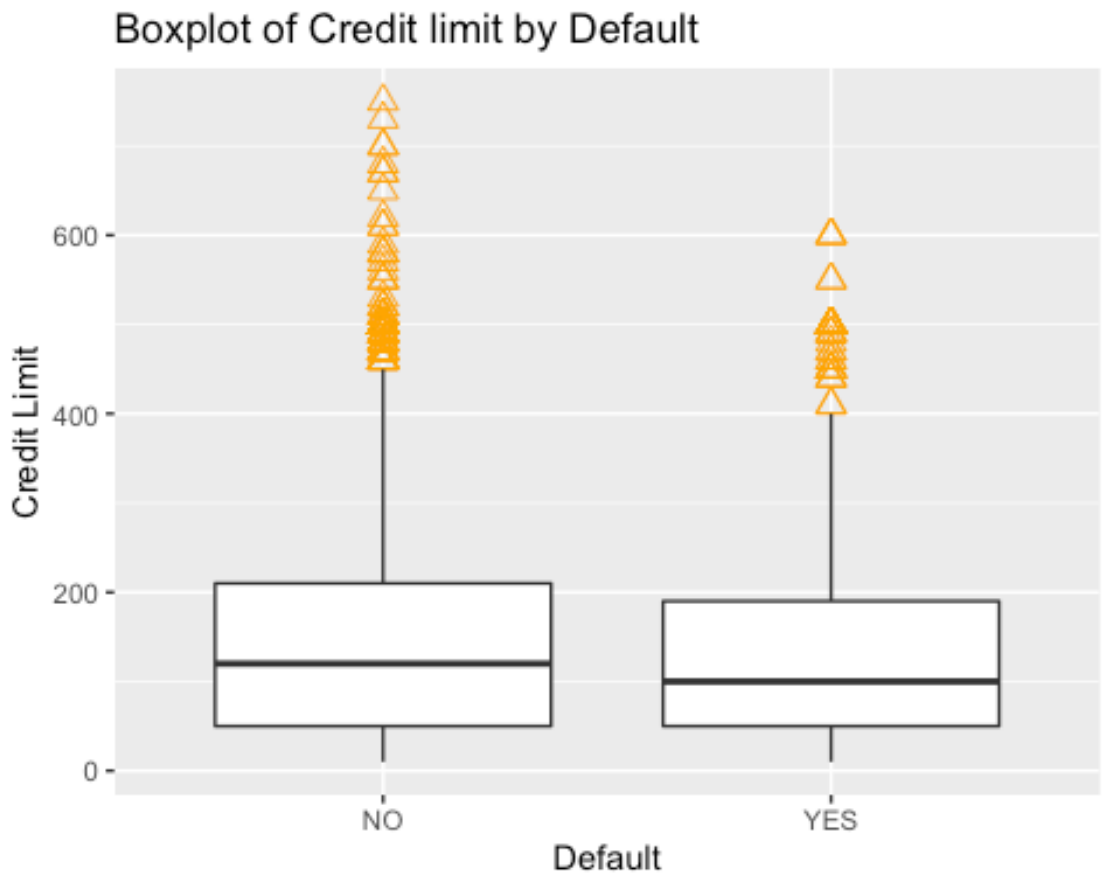
```
library(ggplot2)
ggplot(data = credit_data, aes(x = AGE, fill = DEFAULT )) +
  geom_histogram(alpha = 1) +
  ggtitle("Histogram of age by default") +
  labs(x = "Age", y = "Count")

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



People aged between about 26 - 61 have DEFAULT = YES, but people with age between 20 - 26 and 61+ do not have any default.

c) Create a box-plot of LIMIT_BAL by Default

```
ggplot(data = credit_data, aes(x = DEFAULT, y = LIMIT_BAL / 1000, group = DEF
AULT)) +
  geom_boxplot(outlier.colour="orange", outlier.shape=2, outlier.size=3) +
  ggtitle("Boxplot of Credit limit by Default") +
  labs(x = "Default", y ="Credit Limit" )
```




```
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

credit_data %>%
  group_by(DEFAULT) %>%
  summarize(mean = mean(LIMIT_BAL / 1000),
            median = median(LIMIT_BAL / 1000))

## `summarise()` ungrouping output (override with `.groups` argument)

## # A tibble: 2 x 3
##   DEFAULT  mean median
##   <chr>   <dbl>  <dbl>
```

```
## 1 NO       148.    120
## 2 YES      130.    100
```

People who did not default this month (Default = No) have higher credit limit on their credit card. However, we must take into consideration that Default = No have more outliers on the maximum side. This may happen because people who are constantly paying their credit bill on time will get a higher credit limit.

d) Split the data into 80% training and 20% test data.

```
train_rows <- createDataPartition(y = credit_data$DEFAULT, p =0.8, list = FAL
SE)
#Create the training data using the randomly picked observations from before
data_train <- credit_data[train_rows,]
#Create the test data, by taking all the remaining observations that
#were not included in data_train
data_test <- credit_data[-train_rows, ]
```

K - NN

e) Assess whether you need to standardize the data

A. Yes we need to standardize the data.

Reason 1: Different data types. While gender and marriage are numbers in between 1 - 3, others are continuous variables.

Reason 2: Numbers for age are a lot smaller than other variables such as LIMIT_BAl

We will standardize all the variables beside DEFAULT in the data frame.

```
data_train_stand <- data_train
data_test_stand <- data_test
#load the package standardize
library(standardize)

##
##  ************************************************************
##           Loading standardize package version 0.2.2
##      Call standardize.news() to see new features/changes
##  ************************************************************

#We apply the function scale to the data_train_stand and data_test_stand
data_train_stand[,1:16] <- apply(data_train_stand[,1:16], MARGIN = 2, FUN = s
cale)
data_test_stand[,1:16] <- apply(data_test_stand[,1:16], MARGIN = 2, FUN = sca
le)
```

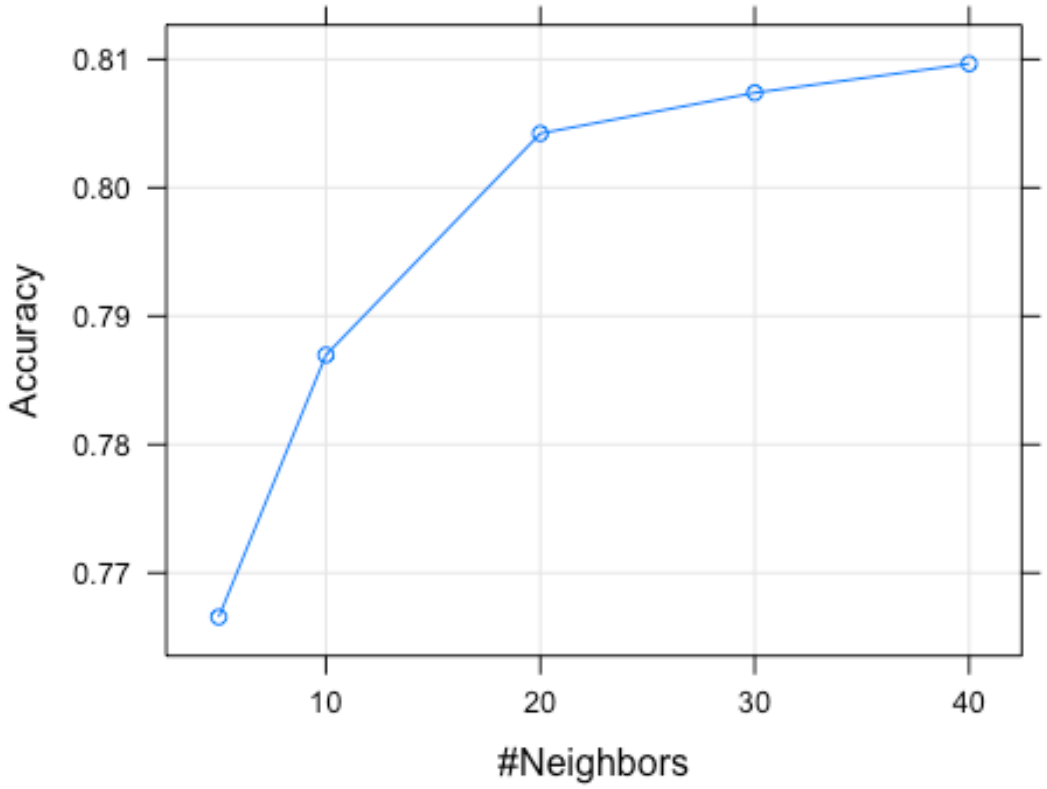
f) train a k-nn model. Use 5 different values of k

```
grid = expand.grid(k = c(5,10,20,30,40))

fitKNN <- train(data= data_train_stand, method = "knn", DEFAULT~.,
                trControl = trainControl(search="grid"), tuneGrid=grid)
fitKNN
```

```
## k-Nearest Neighbors
##
## 5185 samples
##   16 predictor
##    2 classes: 'NO', 'YES'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 5185, 5185, 5185, 5185, 5185, 5185, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.7665630  0.1997080
##   10  0.7869731  0.2092909
##   20  0.8042240  0.2238037
##   30  0.8073937  0.2082073
##   40  0.8096555  0.2044617
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 40.
```

g) Plot the accuracy of the model changes with the value of k

```
plot(fitKNN, ylab = "Accuracy")
```



Accuracy captures the proportion of the total number of predictions that were correct. The accuracy increased as the number of k increased. However, for k = 30 -> 40, the change is subtle.

h) Get the class predictions from k-nn model and produce the confusion matrix using ony the option positive = "YES

```
knn_predictions <- predict(fitKNN, data_test_stand)
confusionMatrix(knn_predictions, as.factor(data_test_stand$DEFAULT), positive
= "YES")

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   NO  YES
##        NO  1003  197
##        YES   41   54
##
##                Accuracy : 0.8162
##                  95% CI : (0.794, 0.837)
##     No Information Rate : 0.8062
##     P-Value [Acc > NIR] : 0.1902
##
##                   Kappa : 0.2302
##
```

```
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.21514
##             Specificity : 0.96073
##          Pos Pred Value : 0.56842
##          Neg Pred Value : 0.83583
##              Prevalence : 0.19382
##          Detection Rate : 0.04170
##    Detection Prevalence : 0.07336
##       Balanced Accuracy : 0.58793
##
##        'Positive' Class : YES
##
```

i) Consider the results obtained in point h).

Overall accuracy of the model is about 80.9%

DEFAULT = YES

A. Recall: 0.21116

B. Precision: 0.51961

C. F-1 score: 0.30028

DEFAULT = NO

A. Recall: 0.9531

B. Precision: 0.8340

C. F-1 score: 0.8896

Default = NO have higher recall, precision, and F-1 score compared to Default = YES. However, we have to note that this dataset is unbalanced and there are way more cases of DEFAULT = NO. Thus, F1-score is a good way to measure the model. The F-1 score for Default = NO is about 0.58 higher than F-1 score for Default = Yes. We can say that this model is better at predicting a person who will not default.

The loss in overall accuracy is caused from not being able to correctly predict a person who will default. We could see that from low recall, precision, and F-1 score of DEFAULT = YES

Decision Trees

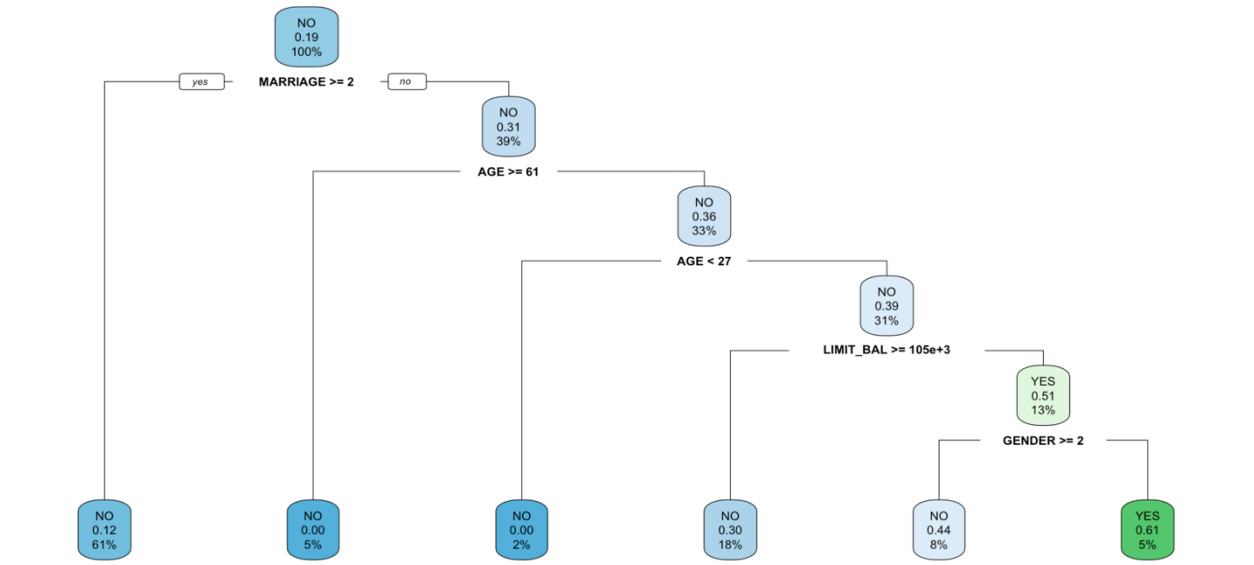
j) Assess whether to use Standardized data. Next, Train the decision tree.

A. We will use data_train because we do NOT have to standardize data in decision tree algorithm

```
fitDT <- train(data = data_train, method = "rpart", DEFAULT~.)
```

k) Plot the decision tree and attach the plot.

```
rpart.plot(fitDT$finalModel)
```



a) Does your decision tree use all the attributes in the data?

A. No, it does not. It is using marriage, Age, Limit_Bal, and Gender. It may be excluding some because those did not lead to the highest information gain.

b) How many leaf nodes does your tree have?
A. Six leaf nodes.

c) Pick two lead nodes
No: The left-most one. If marriage >= 2 (single or others) than the model will predict the person's Default to be No. 61% of the total data is in this node and the probability of Default being YES = 0.12. Since probability > 0.5, this node will classify Default to be NO.

Yes: The right-most one. IF marriage < 2 (married) && 27 <= Age < 60 && Limit_BAL < 105e+3 && Gender < 2 (male) than the person's Default will be classified as YES. There are 5% of total data in this node and the probability of Default being Yes = 0.61. Since it is greater than 0.5 this nodes will classify Default to be YES.

l) Get the class level predictions and produce confusion matrix using the option positive = "YES"

```
DT_predictions <- predict(fitDT$finalModel, newdata = data_test, type = "clas
s")
confusionMatrix(DT_predictions, as.factor(data_test$DEFAULT), positive = "YES
")

Confusion Matrix and Statistics
```

```
          Reference
Prediction   NO  YES
       NO  1012  222
       YES   32   29

               Accuracy : 0.8039
                 95% CI : (0.7812, 0.8252)
    No Information Rate : 0.8062
    P-Value [Acc > NIR] : 0.5998

                  Kappa : 0.1191

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.11554
            Specificity : 0.96935
         Pos Pred Value : 0.47541
         Neg Pred Value : 0.82010
             Prevalence : 0.19382
         Detection Rate : 0.02239
   Detection Prevalence : 0.04710
      Balanced Accuracy : 0.54244

       'Positive' Class : YES
```

m)

Overall accuracy: 80.39%. The accuracy is almost the same as the K-NN model.
DEFAULT = YES
A. Recall: 0.11554
B. Precision: 0.47541
C. F-1 score: 0.18590
DEFAULT = NO
A. Recall: 0.96935
B. Precision: 0.82010
C. F-1 score: 0.8885
The result is somewhat similar to the one we got from K-NN. Default = NO have higher recall, precision, and F-1 score compared to Default = YES. However, for the case for DT, the F-1 score for Default = NO is about 0.7 higher than F-1 score for Default = Yes. We can say that this model is better at predicting a person who will not default.
The loss in overall accuracy is caused from not being able to correctly predict a person who will default. We could see that from low recall, precision, and F-1 score of DEFAULT = YES

Evaluate Results:

n)

I will suggest my institution to use K-NN model for customers classification. K-NN have higher F1 score for Default = YES than DT. This means K-NN have a better model for classifying customers who will default. For banks it is more important to identify the customers who will default. Being able to identify those customers will lead to a decrease in loss and a increase in profit.

o)

```
DT_prob<- as.data.frame(predict(fitDT$finalModel, newdata = data_test, type =
"prob"))
DT_prob$pred_class <- ifelse(DT_prob$NO > 0.75, "NO", "YES")
DT_prob$pred_class<- as.factor(DT_prob$pred_class)
confusionMatrix(DT_prob$pred_class, as.factor(data_test$DEFAULT), positive =
"YES")
Confusion Matrix and Statistics

          Reference

Prediction  NO YES
       NO  815 104
       YES 229 147

               Accuracy : 0.7429
                 95% CI : (0.7181, 0.7665)
    No Information Rate : 0.8062
    P-Value [Acc > NIR] : 1

                  Kappa : 0.308

 Mcnemar's Test P-Value : 1.082e-11
```

```
            Sensitivity : 0.5857
            Specificity : 0.7807
         Pos Pred Value : 0.3910
         Neg Pred Value : 0.8868
             Prevalence : 0.1938
         Detection Rate : 0.1135
   Detection Prevalence : 0.2903
      Balanced Accuracy : 0.6832

       'Positive' Class : YES
```

DEFAULT = YES
A. Recall: 0.5857
B. Precision: 0.3910
C. F-1 score:0.4689

DEFAULT = NO
A. Recall: 0.7807

B. Precision: 0.8868

C. F-1 score: 0.8304

The overall accuracy of this model is 74.29%. It decreased from original accuracy of 80.39. Default = YES still have lower recall, precision, and F-1 score. However, the F-1 score for DEFAULT = YES increased from 0.18 to 0.46. Recall and precision increased too.
For Default = No, recall and F-1 score decreased, but precision increased.
The suggestion makes sense because as I said, being able to identify customers who will default is crucial for the bank. I would prefer the new DT model because it has higher F-1 score for Default = YES. Again, F-1 score is more reliable in this dataset because it is an unbalanced dataset. Also, my answer for point n changes too because this DT model has higher F-1 score for Default = Yes than K-NN model.