task\_2.R

kaushikshamantha

2025-04-14

original\_bank\_data = read.csv("Bank Churn Data CMM703.csv", header = TRUE)  
  
# number of columns/variables  
ncol(original\_bank\_data)

## [1] 13

#columns names  
colnames(original\_bank\_data)

## [1] "CustomerId" "Surname" "CreditScore" "Geography"   
## [5] "Gender" "Age" "Tenure" "Balance"   
## [9] "NumOfProducts" "HasCrCard" "IsActiveMember" "EstimatedSalary"  
## [13] "Exited"

# number of row  
nrow(original\_bank\_data)

## [1] 10000

# quick overview of the dataset  
str(original\_bank\_data)

## 'data.frame': 10000 obs. of 13 variables:  
## $ CustomerId : int 15634602 15647311 15619304 15701354 15737888 15574012 15592531 15656148 15792365 15592389 ...  
## $ Surname : chr "Hargrave" "Hill" "Onio" "Boni" ...  
## $ CreditScore : int 619 608 502 699 850 645 822 376 501 684 ...  
## $ Geography : chr "France" "Spain" "France" "France" ...  
## $ Gender : chr "Female" "Female" "Female" "Female" ...  
## $ Age : int 42 41 42 39 43 44 50 29 44 27 ...  
## $ Tenure : int 2 1 8 1 2 8 7 4 4 2 ...  
## $ Balance : num 0 83808 159661 0 125511 ...  
## $ NumOfProducts : int 1 1 3 2 1 2 2 4 2 1 ...  
## $ HasCrCard : int 1 0 1 0 1 1 1 1 0 1 ...  
## $ IsActiveMember : int 1 1 0 0 1 0 1 0 1 1 ...  
## $ EstimatedSalary: num 101349 112543 113932 93827 79084 ...  
## $ Exited : int 1 0 1 0 0 1 0 1 0 0 ...

# by looking at the dataset description its seems that HasCrCard, IsActiveMember, and Exited are categorical variables  
# lets convert them into categorical variables using factors  
  
# before that, customer id and surname is not required for the analysis, so remove it from the dataset  
original\_bank\_data$CustomerId = NULL  
original\_bank\_data$Surname = NULL  
  
# lets look at null values in the dataset  
sapply(  
 original\_bank\_data,  
 FUN = function(x) sum(is.na(x))  
)

## CreditScore Geography Gender Age Tenure   
## 0 0 0 0 0   
## Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary   
## 0 0 0 0 0   
## Exited   
## 0

# remove the null values if exists, since there are no null values  
bank\_data = na.omit(original\_bank\_data)  
  
# variable delete because not used from here(had to remove because need memory for modeling)  
original\_bank\_data = NULL  
  
change\_to\_factor = function(param\_bank\_data, param\_feature) {  
 return(  
 factor(  
 param\_bank\_data[[param\_feature]],  
 levels = c(0, 1),  
 labels = c("No", "Yes")  
 )  
 )  
}  
  
bank\_data$HasCrCard = change\_to\_factor(bank\_data, 'HasCrCard')  
bank\_data$IsActiveMember = change\_to\_factor(bank\_data, 'IsActiveMember')  
bank\_data$Exited = change\_to\_factor(bank\_data, 'Exited')  
  
# lets do a quick summary statistics  
# i install "vtable" package because it seems more eye pleasing  
# install.packages('vtable')  
library(vtable)

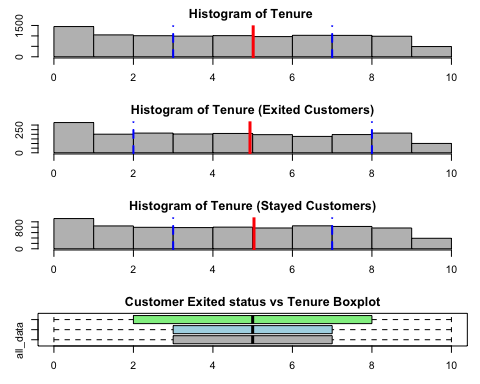
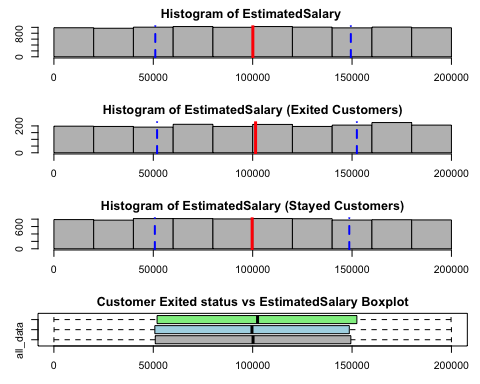
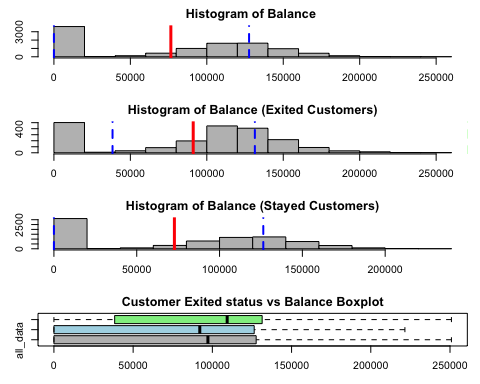
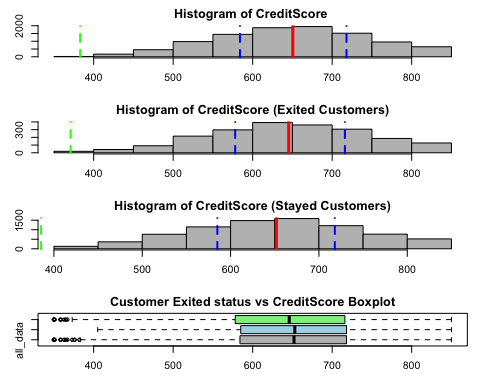
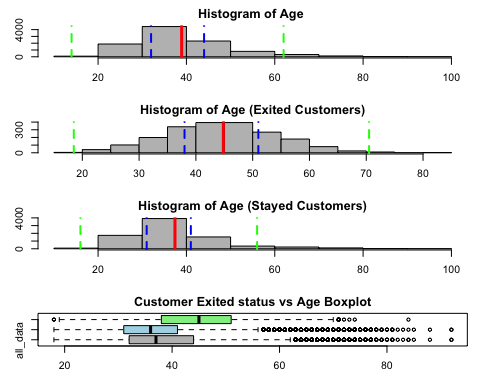
## Loading required package: kableExtra

sumtable(bank\_data)

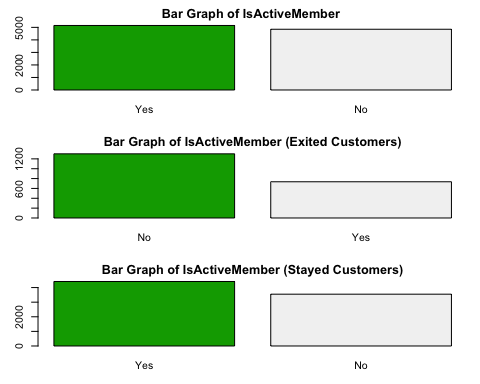
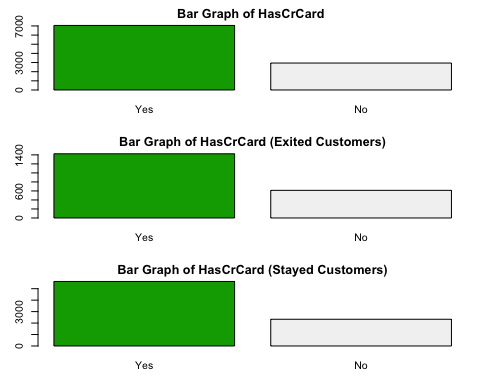
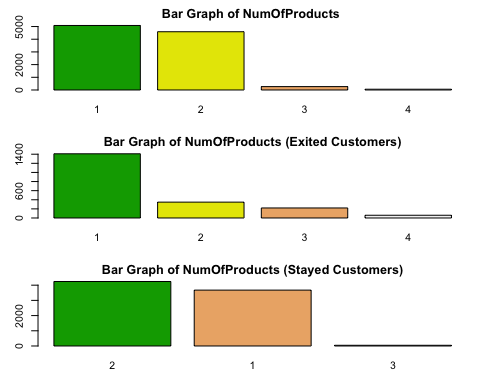
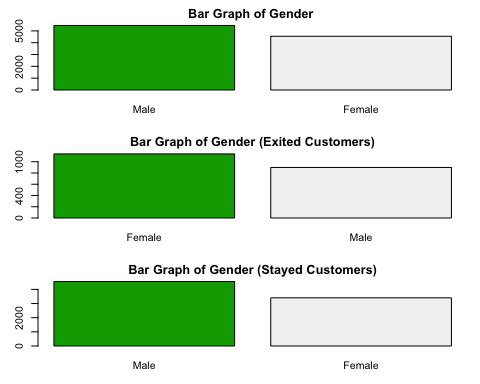
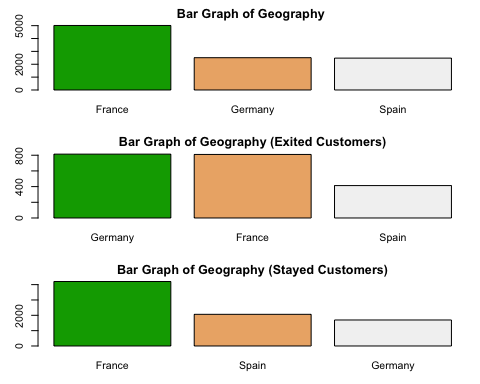
Summary Statistics

| Variable | N | Mean | Std. Dev. | Min | Pctl. 25 | Pctl. 75 | Max |
| --- | --- | --- | --- | --- | --- | --- | --- |
| CreditScore | 10000 | 651 | 97 | 350 | 584 | 718 | 850 |
| Geography | 10000 |  |  |  |  |  |  |
| … France | 5014 | 50% |  |  |  |  |  |
| … Germany | 2509 | 25% |  |  |  |  |  |
| … Spain | 2477 | 25% |  |  |  |  |  |
| Gender | 10000 |  |  |  |  |  |  |
| … Female | 4543 | 45% |  |  |  |  |  |
| … Male | 5457 | 55% |  |  |  |  |  |
| Age | 10000 | 39 | 10 | 18 | 32 | 44 | 92 |
| Tenure | 10000 | 5 | 2.9 | 0 | 3 | 7 | 10 |
| Balance | 10000 | 76486 | 62397 | 0 | 0 | 127644 | 250898 |
| NumOfProducts | 10000 | 1.5 | 0.58 | 1 | 1 | 2 | 4 |
| HasCrCard | 10000 |  |  |  |  |  |  |
| … No | 2945 | 29% |  |  |  |  |  |
| … Yes | 7055 | 71% |  |  |  |  |  |
| IsActiveMember | 10000 |  |  |  |  |  |  |
| … No | 4849 | 48% |  |  |  |  |  |
| … Yes | 5151 | 52% |  |  |  |  |  |
| EstimatedSalary | 10000 | 100090 | 57510 | 12 | 51002 | 149388 | 199992 |
| Exited | 10000 |  |  |  |  |  |  |
| … No | 7963 | 80% |  |  |  |  |  |
| … Yes | 2037 | 20% |  |  |  |  |  |

# quantitative variables  
featureset = c("Age", "CreditScore", "Balance", "EstimatedSalary", "Tenure")  
  
exited\_customers = subset(bank\_data, Exited == "Yes")  
stayed\_customers = subset(bank\_data, Exited == "No")  
  
plot\_histogram = function (param\_feature, param\_bank\_data, param\_title) {  
  
 column\_data = param\_bank\_data[[param\_feature]]  
  
 hist(  
 column\_data,  
 main = paste("Histogram of", param\_feature, param\_title),  
 col = "gray",  
 breaks = 10  
 )  
  
 abline(v = mean(column\_data), col='red', lwd = 3)  
  
 q1 = quantile(column\_data, 0.25)  
 q3 = quantile(column\_data, 0.75)  
 iqr\_value = q3 - q1  
  
 lower\_bound = q1 - 1.5 \* iqr\_value  
 upper\_bound = q3 + 1.5 \* iqr\_value  
  
 # add vertical lines for q1, q3, and whisker bounds  
 abline(v = q1, col = "blue", lwd = 2, lty = 2)  
 abline(v = q3, col = "blue", lwd = 2, lty = 2)  
 abline(v = lower\_bound, col = "green", lwd = 2, lty = 2) # lower Bound  
 abline(v = upper\_bound, col = "green", lwd = 2, lty = 2) # upper Bound  
}  
  
plot\_boxplot = function(param\_feature, param\_bank\_data) {  
  
 # extract numeric column  
 original\_data = param\_bank\_data[[param\_feature]]  
  
 # split the data by 'Exited' and convert it to a dataframe  
 grouped\_data = split(original\_data, param\_bank\_data[['Exited']])  
  
 # create a list including all data + grouped data  
 final\_data = c(list(all\_data = original\_data), grouped\_data)  
  
 boxplot(  
 final\_data,   
 col = c("gray", "lightblue", "lightgreen"),   
 main = paste("Customer Exited status vs", param\_feature, "Boxplot"),  
 xlab = "Exited status",  
 ylab = param\_feature,  
 horizontal = TRUE  
 )  
}  
  
  
for(feature in featureset) {  
  
 layout(matrix(1:4, ncol = 1, byrow = TRUE))  
 par(mar = c(3, 3, 2, 1)) # reduce margins  
  
 plot\_histogram(feature, bank\_data, "")  
 plot\_histogram(feature, exited\_customers, "(Exited Customers)")  
 plot\_histogram(feature, stayed\_customers, "(Stayed Customers)")  
   
 plot\_boxplot(feature, bank\_data)  
}



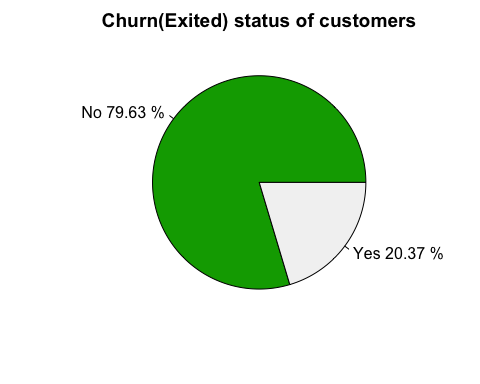
# qualitative variables  
  
featureset = c("Geography", "Gender", "NumOfProducts", "HasCrCard", "IsActiveMember")  
  
plot\_bargraph = function (param\_feature, param\_bank\_data, param\_title) {  
  
 feature\_data = param\_bank\_data[[param\_feature]]  
 feature\_count = sort(table(feature\_data), decreasing = TRUE)  
  
 bar\_colors = terrain.colors(length(names(feature\_count)))  
  
 barplot(  
 feature\_count,  
 col = bar\_colors,  
 main = paste("Bar Graph of", param\_feature, param\_title),  
 ylim = c(0, max(feature\_count) + 10)  
 )  
}  
  
for(feature in featureset) {  
  
 layout(matrix(1:3, ncol = 1, byrow = TRUE))  
 par(mar = c(3, 3, 2, 1)) # reduce margins  
  
 plot\_bargraph(feature, bank\_data, "")  
 plot\_bargraph(feature, exited\_customers, "(Exited Customers)")  
 plot\_bargraph(feature, stayed\_customers, "(Stayed Customers)")  
}



exited\_customers = NULL  
stayed\_customers = NULL  
  
# we have to predict the churn status of the customer?  
# y == "churn status", YES or NO, Qualitative Variable and Binary  
# x == (other fields)  
# model should be binary logistic regression model  
  
  
counts = table(bank\_data$Exited)  
print(counts)

##   
## No Yes   
## 7963 2037

# you can see the percentages are not equal meaning dataset is not balance  
  
percentages = round(counts / sum(counts) \* 100, 2)  
labels = paste(names(counts), percentages, "%")  
colors = terrain.colors(2)  
par(mfrow = c(1, 1))  
pie(counts, labels = labels, col = colors, main = "Churn(Exited) status of customers")



# lets start create a model to check customer churn status  
set.seed(2425499)  
  
number\_of\_rows = nrow(bank\_data)  
train\_percentage = 0.8  
train\_ids = sample(1:number\_of\_rows, number\_of\_rows \* train\_percentage, replace = FALSE)  
  
train\_dataset = bank\_data[train\_ids, ]  
test\_dataset = bank\_data[-train\_ids, ]  
  
names(bank\_data)

## [1] "CreditScore" "Geography" "Gender" "Age"   
## [5] "Tenure" "Balance" "NumOfProducts" "HasCrCard"   
## [9] "IsActiveMember" "EstimatedSalary" "Exited"

bank\_model = glm(  
 formula = Exited ~ .,  
 data = train\_dataset,  
 family = binomial  
)  
  
summary(bank\_model)

##   
## Call:  
## glm(formula = Exited ~ ., family = binomial, data = train\_dataset)  
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.387e+00 2.734e-01 -12.390 < 2e-16 \*\*\*  
## CreditScore -6.337e-04 3.128e-04 -2.026 0.0428 \*   
## GeographyGermany 7.307e-01 7.640e-02 9.564 < 2e-16 \*\*\*  
## GeographySpain 1.658e-02 7.812e-02 0.212 0.8319   
## GenderMale -5.285e-01 6.089e-02 -8.680 < 2e-16 \*\*\*  
## Age 7.190e-02 2.860e-03 25.135 < 2e-16 \*\*\*  
## Tenure -2.038e-02 1.048e-02 -1.945 0.0518 .   
## Balance 2.374e-06 5.722e-07 4.149 3.35e-05 \*\*\*  
## NumOfProducts -7.413e-02 5.264e-02 -1.408 0.1591   
## HasCrCardYes -4.866e-02 6.646e-02 -0.732 0.4640   
## IsActiveMemberYes -1.077e+00 6.445e-02 -16.711 < 2e-16 \*\*\*  
## EstimatedSalary 7.846e-07 5.307e-07 1.479 0.1393   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 8058.8 on 7999 degrees of freedom  
## Residual deviance: 6867.0 on 7988 degrees of freedom  
## AIC: 6891  
##   
## Number of Fisher Scoring iterations: 5

refined\_bank\_model = glm(  
 formula = Exited ~ CreditScore + Geography + Gender + Age + Balance + IsActiveMember,  
 data = train\_dataset,  
 family = binomial,  
)  
  
# validate the model  
column\_index = which(names(test\_dataset) == "Exited")  
  
predictions = predict.glm(  
 refined\_bank\_model,  
 newdata = test\_dataset[, -column\_index],  
 type = "response"   
)  
  
test\_dataset$predicted\_class = factor(ifelse(predictions > 0.5, "Yes", "No"))  
  
table(  
 test\_dataset$predicted\_class,   
 test\_dataset$Exited,  
 dnn = c("Predicted", "Actual")  
)

## Actual  
## Predicted No Yes  
## No 1531 324  
## Yes 51 94

table(test\_dataset$Exited)

##   
## No Yes   
## 1582 418

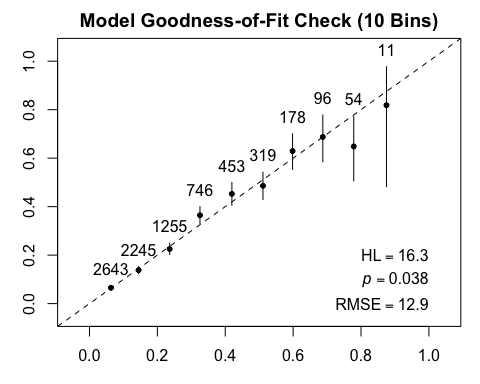
mean(test\_dataset$predicted\_class == test\_dataset$Exited)

## [1] 0.8125

# accuracy is 81.25% (0.8125)  
  
# goodness of fit of the model   
# hosmer and lemeshow test   
# h0 : model is adequate  
# h1 : model is not adequate   
  
# install.packages("modEvA")  
library(modEvA)  
  
hoslem\_results <- HLfit(  
 model = refined\_bank\_model,  
 bin.method = "n.bins",  
 n.bins = 10,  
 main = "Model Goodness-of-Fit Check (10 Bins)"  
)

## Arguments min.bin.size and min.prob.interval are ignored by this bin.method.

## Warning in getBins(obs = obs, pred = pred, bin.method = bin.method, n.bins =  
## n.bins, : There is at least one bin with less than 15 values, for which  
## comparisons may not be meaningful; consider using a bin.method that allows  
## defining a minimum bin size



# Print the test results  
par(mfrow = c(1, 1))  
print(hoslem\_results)

## $bins.table  
## BinCenter NBin BinObs BinPred BinObsCIlower  
## (0.0118,0.104] 0.06323023 2643 0.06507756 0.06360322 0.05597051  
## (0.104,0.196] 0.14474537 2245 0.13808463 0.14631650 0.12407053  
## (0.196,0.287] 0.23627184 1255 0.22549801 0.23746048 0.20264319  
## (0.287,0.378] 0.32547471 746 0.36461126 0.32879822 0.32999554  
## (0.378,0.47] 0.41952456 453 0.45253863 0.41933783 0.40603970  
## (0.47,0.561] 0.51111659 319 0.48589342 0.51293455 0.42983322  
## (0.561,0.653] 0.59835489 178 0.62921348 0.60267986 0.55376354  
## (0.653,0.744] 0.68745992 96 0.68750000 0.69068894 0.58481840  
## (0.744,0.836] 0.77853027 54 0.64814815 0.78267347 0.50623786  
## (0.836,0.928] 0.87502005 11 0.81818182 0.88055476 0.48224415  
## BinObsCIupper  
## (0.0118,0.104] 0.07516176  
## (0.104,0.196] 0.15305046  
## (0.196,0.287] 0.24964613  
## (0.287,0.378] 0.40030220  
## (0.378,0.47] 0.49966116  
## (0.47,0.561] 0.54221794  
## (0.561,0.653] 0.70028198  
## (0.653,0.744] 0.77824736  
## (0.744,0.836] 0.77318812  
## (0.836,0.928] 0.97716880  
##   
## $chi.sq  
## [1] 16.30613  
##   
## $DF  
## [1] 8  
##   
## $p.value  
## [1] 0.03820228  
##   
## $RMSE  
## [1] 12.93108

# p-value is 0.038 < 0.05 meaning we reject h0.  
# the conclusion of the statement is model is the model is not adequate.  
  
# model has 81.25% accuracy, but the Hosmer-Lemeshow (H-L) test shows p-value is 0.038  
# that means a potential lack of fit.  
  
# what is the next step then,  
# \* we can change the weights meaning 'glm' function support 'weights' as a parameter.  
# \* we can give priority to the 'Yes' and redo the modeling.