CSP 571 Project - Random Forest

```
library(plyr)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(ggplot2)
library(gridExtra)
library(ggthemes)
library(caret)
```

```
## Loading required package: lattice
```

```
## Warning in system("timedatectl", intern = TRUE): running command 'timedatectl'
## had status 1
```

```
library(lattice)
library(MASS)
library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(party)
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
##
## Attaching package: 'modeltools'
```

```
## The following object is masked from 'package:plyr':
##
##     empty
```

```
## Loading required package: strucchange
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich
```

```
library(sandwich)
library(rpart)
library(rattle)
```

```
## Loading required package: tibble

## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

##
## Attaching package: 'rattle'

## The following object is masked from 'package:randomForest':
##
##     importance
```

```
library(GoodmanKruskal)
library(e1071)
library(rpart.plot)
library(caTools)
library(SciViews)
library(class)
```

```
churn_data <- read.csv('data/BankChurners.csv')
```

```
sapply(churn_data, function(x) sum(is.na(x))) # No Nans Awesome
```

```
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
```

```
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
##
## Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_
##
## Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_
##
```

```
# REMOVE IRRELEVANT COLUMNS


drop <- c("Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Edu
churn_data = churn_data[,!(names(churn_data) %in% drop)]
churn_data[sapply(churn_data, is.character)]<- lapply(churn_data[sapply(churn_data, is.character)], as.
```

```
summary(churn_data)
```

```
##            Attrition_Flag  Customer_Age   Gender   Dependent_count
##  Attrited Customer:1627   Min.   :26.00   F:5358   Min.   :0.000
##  Existing Customer:8500   1st Qu.:41.00   M:4769   1st Qu.:1.000
##                           Median :46.00            Median :2.000
##                           Mean   :46.33            Mean   :2.346
##                           3rd Qu.:52.00            3rd Qu.:3.000
##                           Max.   :73.00            Max.   :5.000
##
##       Education_Level  Marital_Status       Income_Category  Card_Category
##  College      :1013   Divorced: 748   $120K +       : 727   Blue    :9436
##  Doctorate    : 451   Married :4687   $40K - $60K   :1790   Gold    : 116
##  Graduate     :3128   Single  :3943   $60K - $80K   :1402   Platinum:  20
##  High School  :2013   Unknown : 749   $80K - $120K  :1535   Silver  : 555
##  Post-Graduate: 516                   Less than $40K:3561
##  Uneducated   :1487                   Unknown       :1112
##  Unknown      :1519
##  Months_on_book  Total_Relationship_Count Months_Inactive_12_mon
##  Min.   :13.00   Min.   :1.000           Min.   :0.000
##  1st Qu.:31.00   1st Qu.:3.000           1st Qu.:2.000
```

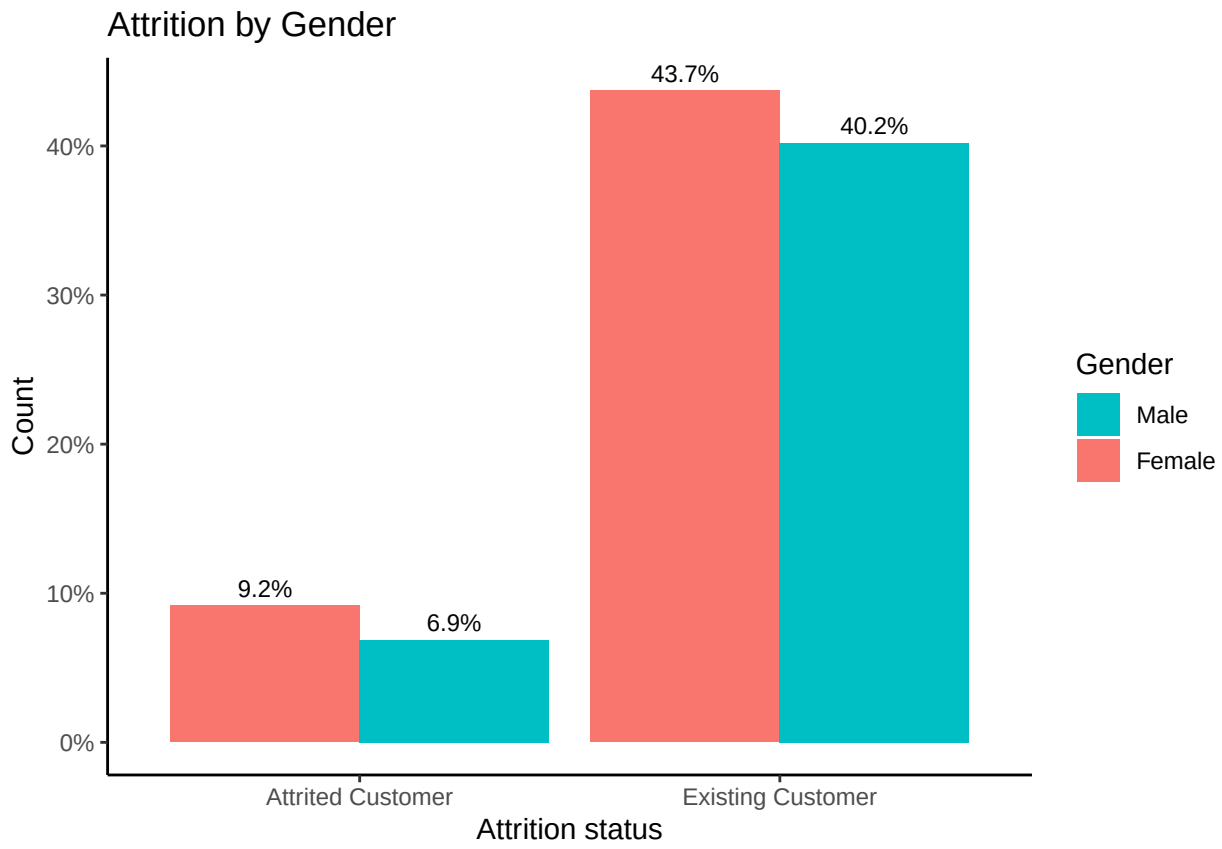
```
##  Median :36.00   Median :4.000            Median :2.000
##  Mean   :35.93   Mean   :3.813            Mean   :2.341
##  3rd Qu.:40.00   3rd Qu.:5.000            3rd Qu.:3.000
##  Max.   :56.00   Max.   :6.000            Max.   :6.000
##
##  Contacts_Count_12_mon  Credit_Limit   Total_Revolving_Bal Avg_Open_To_Buy
##  Min.   :0.000         Min.   : 1438   Min.   :   0        Min.   :    3
##  1st Qu.:2.000         1st Qu.: 2555   1st Qu.: 359        1st Qu.: 1324
##  Median :2.000         Median : 4549   Median :1276        Median : 3474
##  Mean   :2.455         Mean   : 8632   Mean   :1163        Mean   : 7469
##  3rd Qu.:3.000         3rd Qu.:11068   3rd Qu.:1784        3rd Qu.: 9859
##  Max.   :6.000         Max.   :34516   Max.   :2517        Max.   :34516
##
##  Total_Amt_Chng_Q4_Q1 Total_Trans_Amt Total_Trans_Ct   Total_Ct_Chng_Q4_Q1
##  Min.   :0.0000       Min.   :  510   Min.   : 10.00   Min.   :0.0000
##  1st Qu.:0.6310       1st Qu.: 2156   1st Qu.: 45.00   1st Qu.:0.5820
##  Median :0.7360       Median : 3899   Median : 67.00   Median :0.7020
##  Mean   :0.7599       Mean   : 4404   Mean   : 64.86   Mean   :0.7122
##  3rd Qu.:0.8590       3rd Qu.: 4741   3rd Qu.: 81.00   3rd Qu.:0.8180
##  Max.   :3.3970       Max.   :18484   Max.   :139.00   Max.   :3.7140
##
##  Avg_Utilization_Ratio
##  Min.   :0.0000
##  1st Qu.:0.0230
##  Median :0.1760
##  Mean   :0.2749
##  3rd Qu.:0.5030
##  Max.   :0.9990
##
```

Let's see the

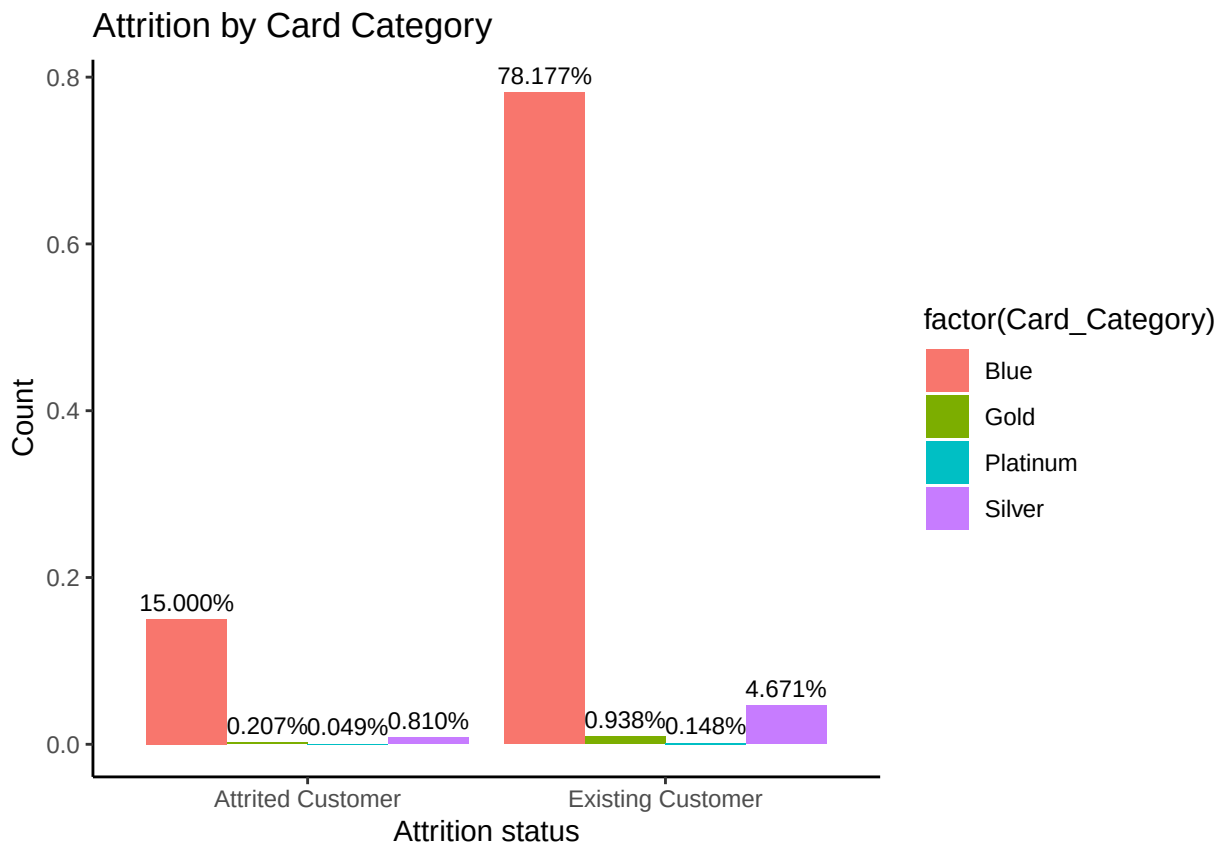
```
ggplot(churn_data, aes(x=Attrition_Flag,
                  y= prop.table(stat(count)),
                  fill= factor(Gender),
                  label= scales::percent(prop.table(stat(count))))) +
  geom_bar(position = position_dodge())+
  geom_text(stat="count",
            position = position_dodge(.9),
            vjust= -0.5, size=3)+
  scale_y_continuous(labels = scales::percent)+
  labs(title = "Attrition by Gender",
       x= "Attrition status",
       y="Count")+
  theme_classic()+
  scale_fill_discrete(
    name="Gender",
    breaks=c("M", "F"),
    labels=c("Male", "Female" )
  )
```

```
## Warning: `stat(count)` was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(count)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
```

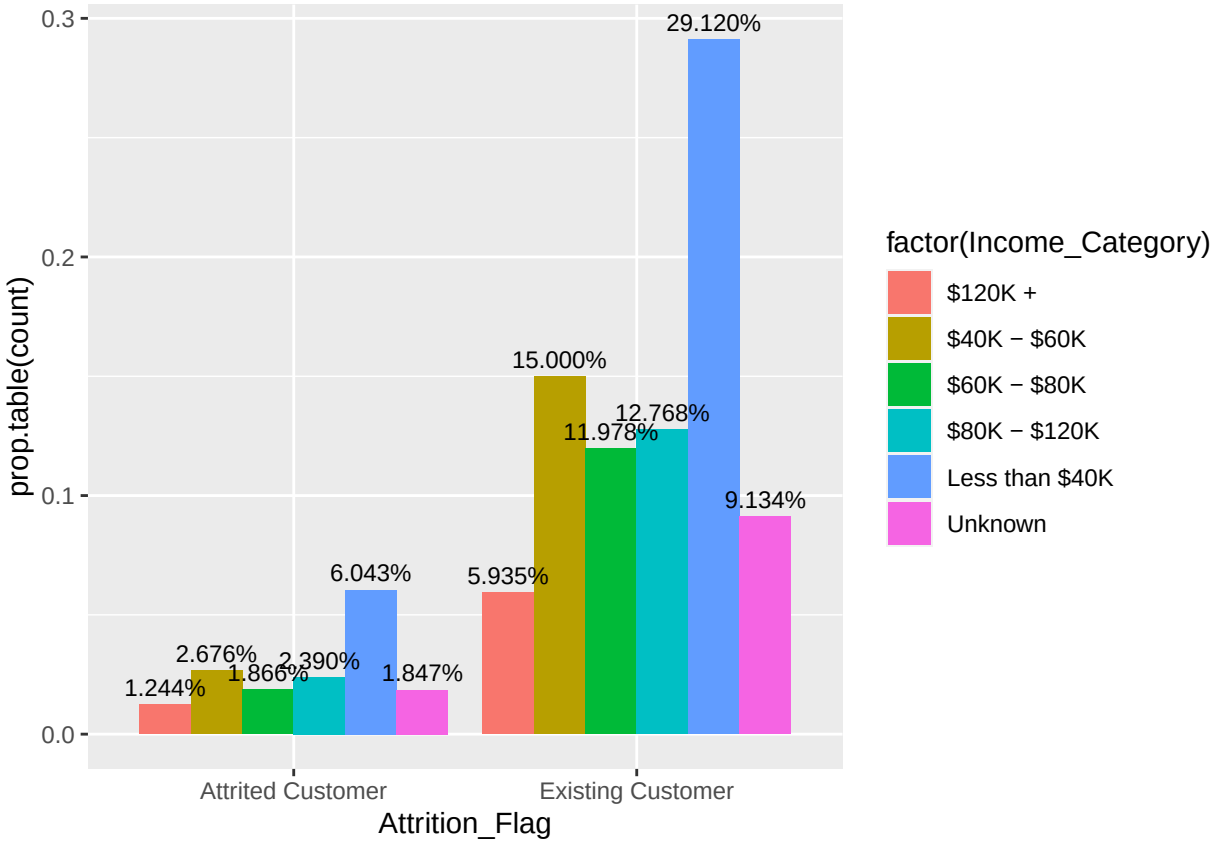
generated.




```
ggplot(churn_data, aes(x=Attrition_Flag,
                       y= prop.table(stat(count)),
                       fill= factor(Card_Category),
                       label= scales::percent(prop.table(stat(count))))) +
  geom_bar(position = position_dodge())+
  geom_text(stat="count",
            position = position_dodge(.9),
            vjust= -0.5, size=3)+
  labs(title = "Attrition by Card Category",
       x= "Attrition status",
       y="Count")+
  theme_classic()
```



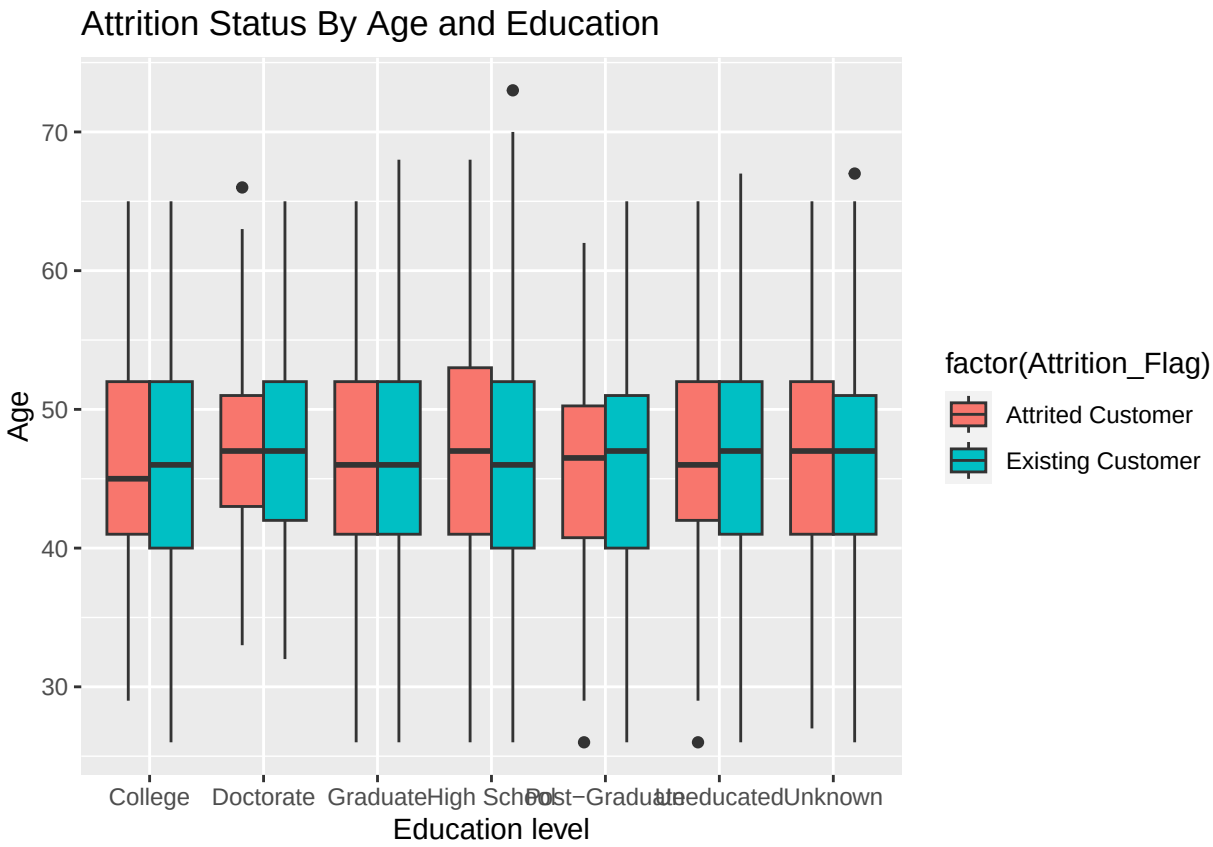

```
ggplot(churn_data, aes(x=Attrition_Flag,
                       y= prop.table(stat(count)),
                       fill= factor(Income_Category),
                       label= scales::percent(prop.table(stat(count))))) +
  geom_bar(position = position_dodge())+
  geom_text(stat="count",
            position = position_dodge(.9),
            vjust= -0.5, size=3)
```



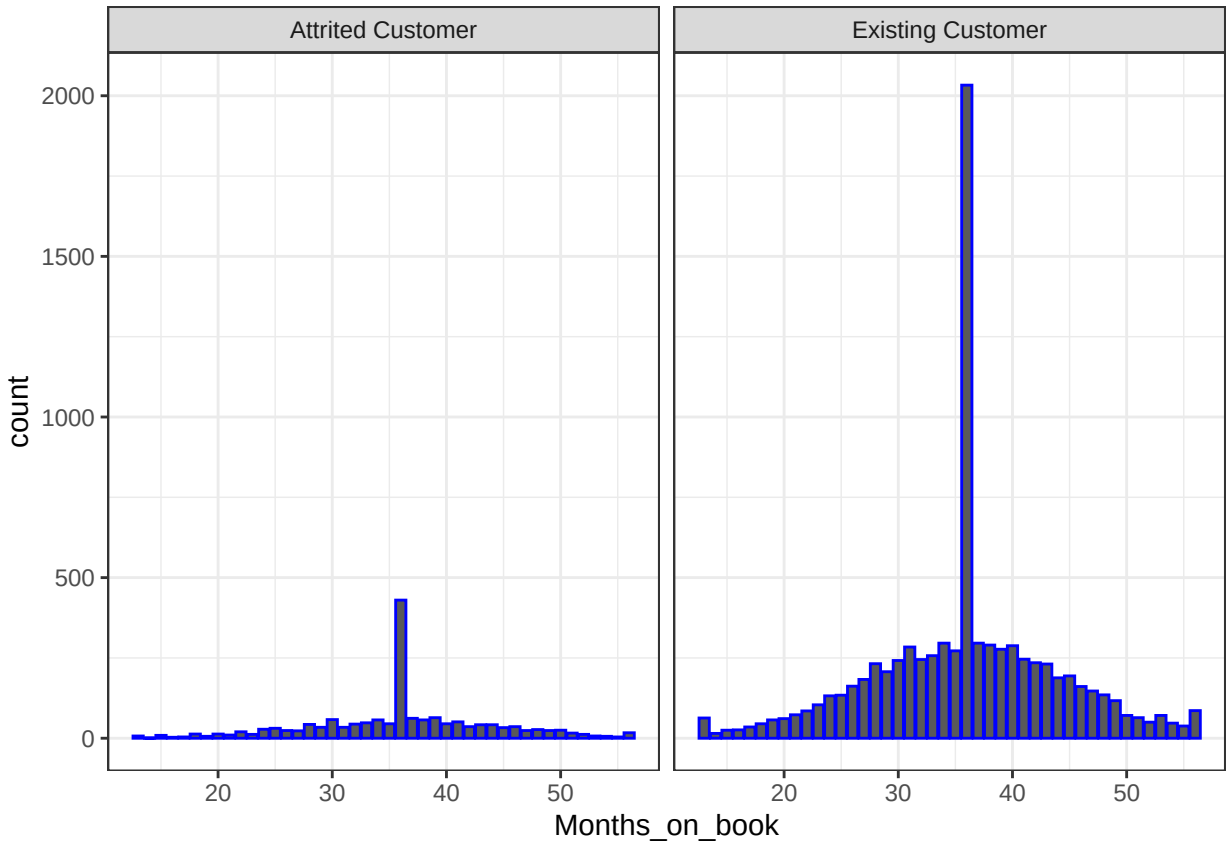

```
  labs(title = "Attrition by Income Category",
       x= "Attrition status",
       y="Count")+
  theme_classic()
```

```
## NULL
```

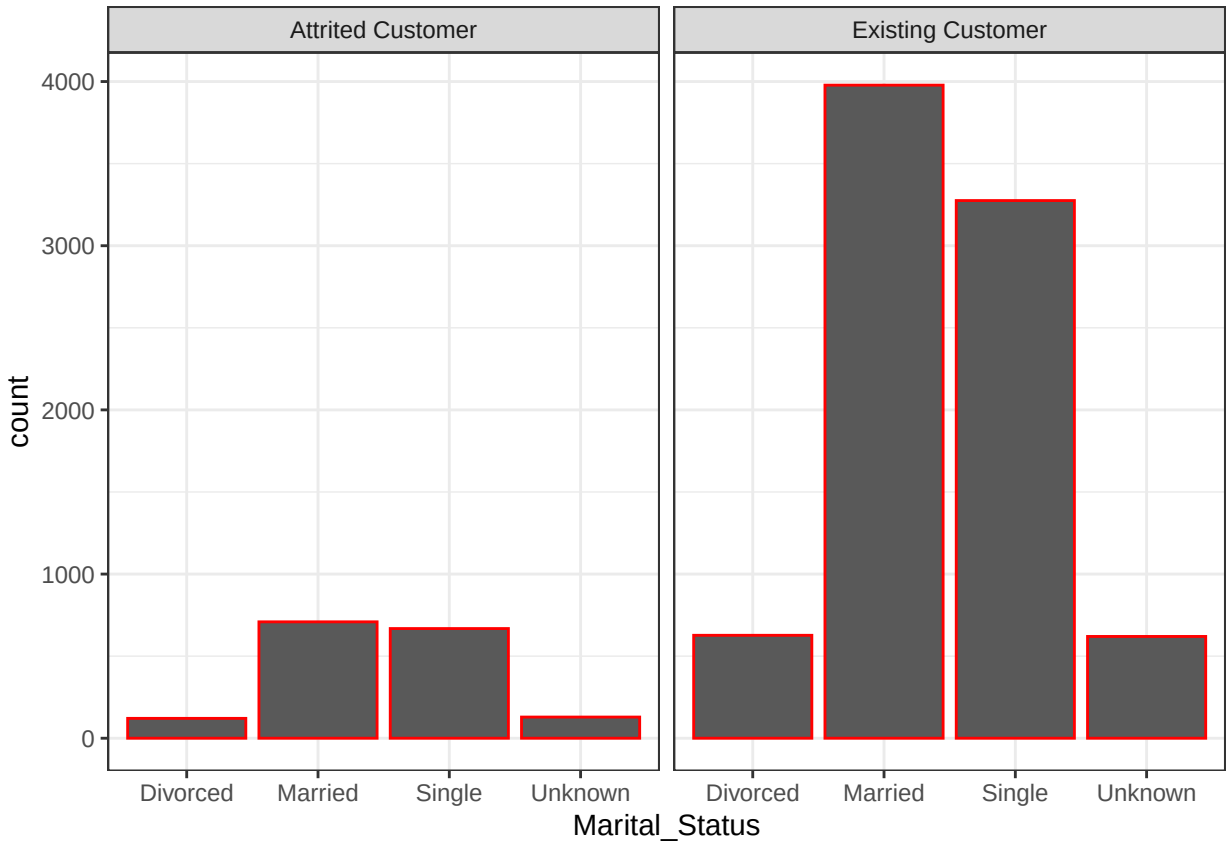
```
ggplot(churn_data, aes(y=Customer_Age,
                   x= Education_Level,
                   fill= factor(Attrition_Flag))) +
  geom_boxplot(position = position_dodge())+
  labs(title = "Attrition Status By Age and Education",
       x= "Education level",
       y="Age")
```



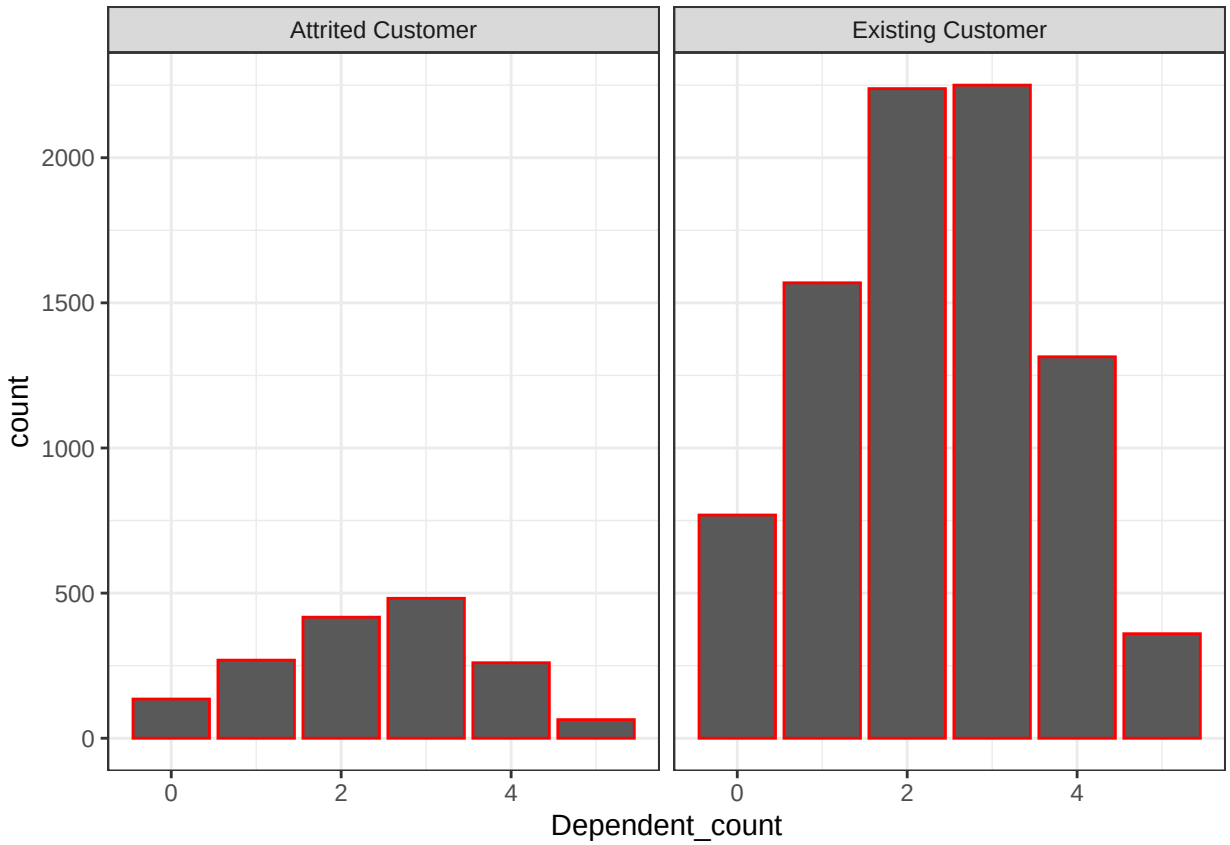

```
ggplot(churn_data, aes(Months_on_book))+
  geom_bar(col="blue")+ facet_wrap(~Attrition_Flag)+theme_bw()
```




```
ggplot(churn_data, aes(Marital_Status))+
  geom_bar(col="red")+ facet_wrap(~Attrition_Flag)+theme_bw()
```




```
ggplot(churn_data, aes(Dependent_count))+
  geom_bar(col="red")+ facet_wrap(~Attrition_Flag)+theme_bw()
```

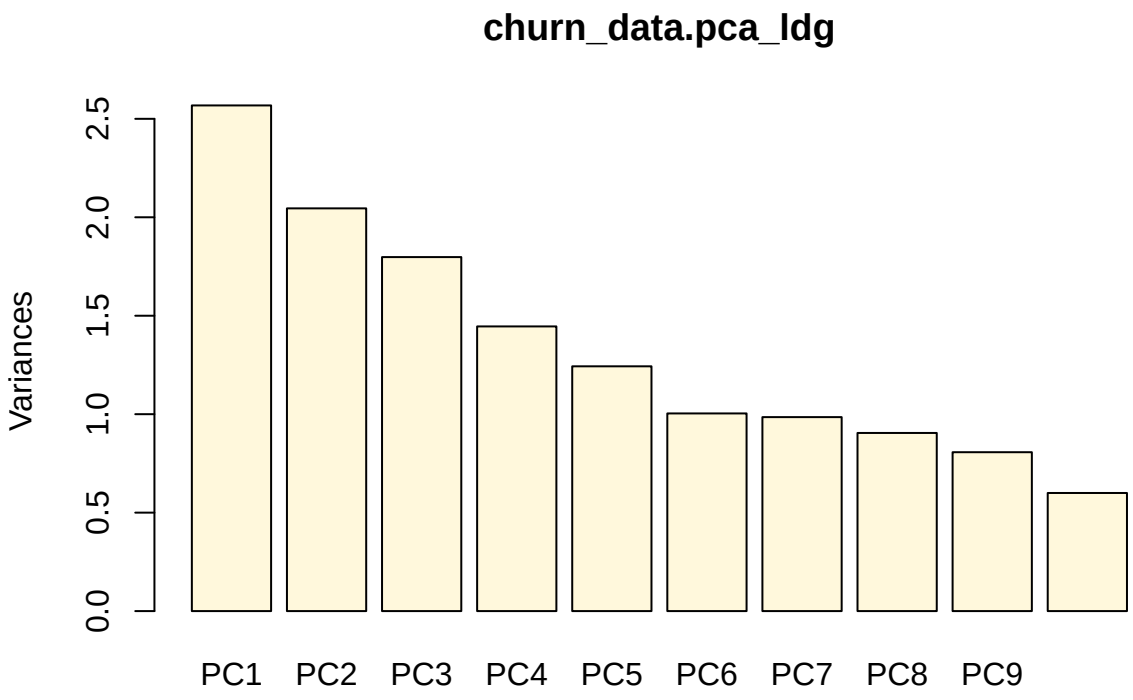



```
#PCA on numerical columns ## maybe even do MCA for the categorical ones
churn_data.pca_ldg <- pcomp(scale(churn_data[,c(2,4,9:20)]), center = TRUE)
churn_data.pca <- prcomp(scale(churn_data[,c(2,4,9:20)]), center = TRUE)

summary(churn_data.pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5    PC6     PC7
## Standard deviation     1.6025 1.4301 1.3408 1.2024 1.11491 1.0019 0.99250
## Proportion of Variance 0.1834 0.1461 0.1284 0.1033 0.08879 0.0717 0.07036
## Cumulative Proportion  0.1834 0.3295 0.4579 0.5612 0.64998 0.7217 0.79203
##                           PC8     PC9    PC10    PC11    PC12    PC13
## Standard deviation     0.95112 0.89829 0.77448 0.47086 0.45909 0.40948
## Proportion of Variance 0.06462 0.05764 0.04284 0.01584 0.01505 0.01198
## Cumulative Proportion  0.85665 0.91429 0.95713 0.97297 0.98802 1.00000
##                           PC14
## Standard deviation     4.245e-16
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00
```

```
screeplot(churn_data.pca_ldg)
```

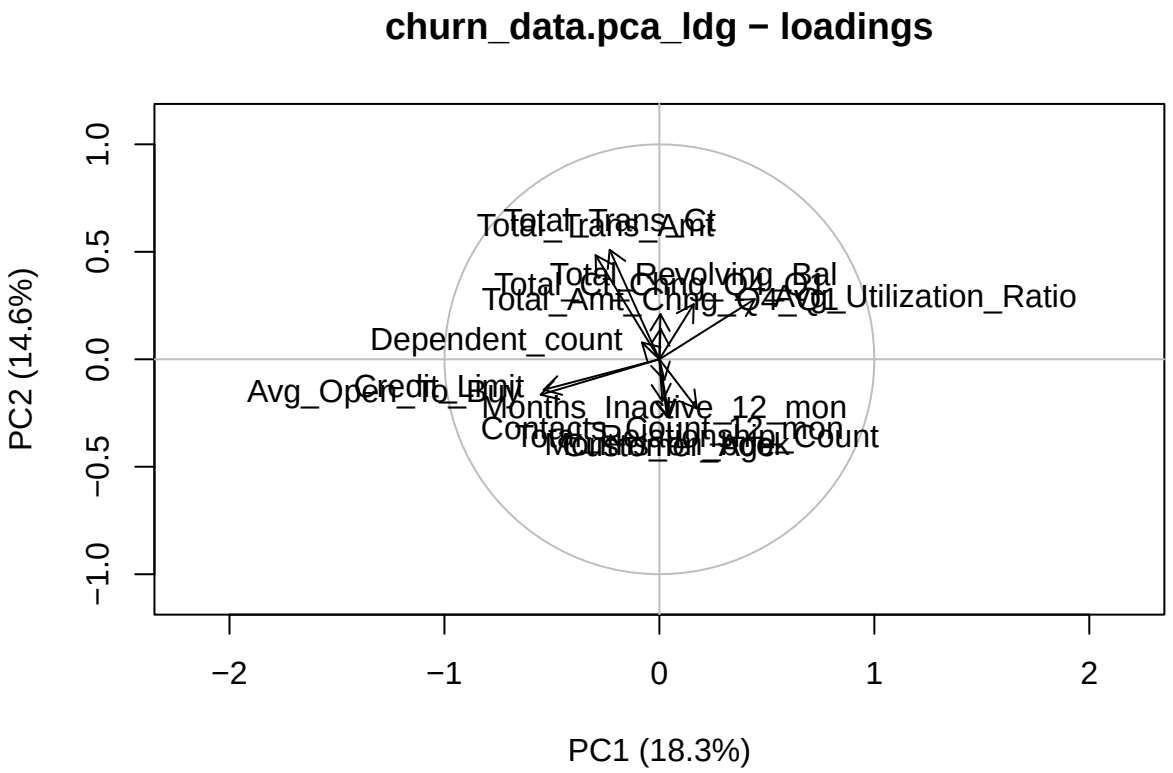



```
(churn_data.ldg <- loadings(churn_data.pca_ldg))
```

```
##
## Loadings:
##                          PC1    PC2    PC3    PC4    PC5    PC6    PC7
## Customer_Age                    -0.271  0.630
## Dependent_count                        -0.149        -0.133 -0.693 -0.313
## Months_on_book                  -0.269  0.630
## Total_Relationship_Count  0.174 -0.229 -0.171 -0.281
## Months_Inactive_12_mon                        0.102         0.194 -0.935
## Contacts_Count_12_mon           -0.194 -0.120                0.641
## Credit_Limit             -0.538 -0.142        -0.269 -0.278
## Total_Revolving_Bal       0.160  0.256  0.141 -0.369 -0.622
## Avg_Open_To_Buy          -0.552 -0.165        -0.236 -0.222
## Total_Amt_Chng_Q4_Q1              0.142        -0.535  0.411  0.116
## Total_Trans_Amt          -0.297  0.484  0.232  0.166         0.142
## Total_Trans_Ct           -0.231  0.509  0.217  0.176
## Total_Ct_Chng_Q4_Q1               0.210        -0.519  0.392
## Avg_Utilization_Ratio     0.444  0.280  0.107 -0.152 -0.354
##                          PC8    PC9    PC10   PC11   PC12   PC13   PC14
## Customer_Age             -0.106                       -0.694 -0.101
## Dependent_count          -0.588  0.104
## Months_on_book           -0.124                        0.693
## Total_Relationship_Count  0.177  0.865 -0.107
## Months_Inactive_12_mon    0.242
## Contacts_Count_12_mon    -0.708  0.139
```

```
## Credit_Limit                                0.184                0.706
## Total_Revolving_Bal                        -0.540        -0.237
## Avg_Open_To_Buy                             0.232               -0.706
## Total_Amt_Chng_Q4_Q1     -0.118 -0.189 -0.671
## Total_Trans_Amt                  0.209      -0.259 -0.136  0.658
## Total_Trans_Ct                   0.363       0.259  0.113 -0.620
## Total_Ct_Chng_Q4_Q1      -0.105         0.712
## Avg_Utilization_Ratio                       0.682         0.302
##
##                  PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10
## SS loadings    1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
## Proportion Var 0.071 0.071 0.071 0.071 0.071 0.071 0.071 0.071 0.071 0.071
## Cumulative Var 0.071 0.143 0.214 0.286 0.357 0.429 0.500 0.571 0.643 0.714
##                 PC11  PC12  PC13  PC14
## SS loadings    1.000 1.000 1.000 1.000
## Proportion Var 0.071 0.071 0.071 0.071
## Cumulative Var 0.786 0.857 0.929 1.000
```

```
plot(churn_data.pca_ldg, which = "loadings",)
```




```
pc_data <- churn_data.pca$x[,1:10]
cat_data <- churn_data[,c(1,3,5:8)]
churn_pca <-data.frame(cat_data, pc_data)
churn_pca[sapply(churn_pca, is.character)]<- lapply(churn_pca[sapply(churn_pca, is.character)], as.facto
```

```
# Splitting the PCA DATA
```

```r
train_indices <- createDataPartition(churn_pca$Attrition_Flag, p = 0.80, list = FALSE)

# Select the rows for training and testing based on the partition created above
training_pca <- churn_pca[train_indices,]
testing_pca <- churn_pca[-train_indices,]

# Print the dimensions of the training and testing datasets to ensure they are split correctly
cat("Training data dimensions:", dim(training_pca), "\n")
```

```
## Training data dimensions: 8102 16
```

```r
cat("Testing data dimensions:", dim(testing_pca), "\n")
```

```
## Testing data dimensions: 2025 16
```

```r
# Print summary statistics of the training and testing datasets if desired
# summary(training_pca)
# summary(testing_pca)
```

```r
# Splitting the regular dataset
# Create a data partition with 80% of the data for training and 20% for testing
train_indices <- createDataPartition(churn_data$Attrition_Flag, p = 0.80, list = FALSE)

# Select the rows for training and testing based on the partition created above
train_data <- churn_data[train_indices,]
test_data <- churn_data[-train_indices,]

# Print the dimensions of the training and testing datasets to ensure they are split correctly
cat("Training data dimensions:", dim(train_data), "\n")
```

```
## Training data dimensions: 8102 20
```

```r
cat("Testing data dimensions:", dim(test_data), "\n")
```

```
## Testing data dimensions: 2025 20
```

```r
# Print summary statistics of the training and testing datasets if desired
# summary(train_data)
# summary(test_data)
```

```r
# Random Forest for PCA data
# Train a random forest model with 500 trees using the PCA training dataset
rf_pca <- randomForest(Attrition_Flag ~ ., ntree = 500, family = "binomial", data = training_pca)

# Print summary statistics of the random forest model
print(summary(rf_pca))
```

```
##                 Length Class  Mode
## call               5   -none- call
## type               1   -none- character
## predicted       8102   factor numeric
## err.rate        1500   -none- numeric
## confusion          6   -none- numeric
## votes          16204   matrix numeric
## oob.times       8102   -none- numeric
## classes            2   -none- character
## importance        15   -none- numeric
```

```
## importanceSD        0  -none- NULL
## localImportance     0  -none- NULL
## proximity           0  -none- NULL
## ntree               1  -none- numeric
## mtry                1  -none- numeric
## forest             14  -none- list
## y                8102  factor numeric
## test                0  -none- NULL
## inbag               0  -none- NULL
## terms               3  terms  call
```

```
# Print the random forest model to inspect the model structure and parameters
rf_pca
```

```
##
## Call:
##  randomForest(formula = Attrition_Flag ~ ., data = training_pca,      ntree = 500, family = "binomial
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 8.73%
## Confusion matrix:
##                   Attrited Customer Existing Customer class.error
## Attrited Customer               701               601  0.46159754
## Existing Customer               106              6694  0.01558824
```

```
# Predict on test
rf_pca_pred <- predict(rf_pca, testing_pca)

caret::confusionMatrix(rf_pca_pred, test_data$Attrition_Flag)
```

```
## Confusion Matrix and Statistics
##
##                    Reference
## Prediction          Attrited Customer Existing Customer
##   Attrited Customer                25               144
##   Existing Customer               300              1556
##
##                Accuracy : 0.7807
##                  95% CI : (0.7621, 0.7986)
##     No Information Rate : 0.8395
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0097
##
##  Mcnemar's Test P-Value : 1.895e-13
##
##             Sensitivity : 0.07692
##             Specificity : 0.91529
##          Pos Pred Value : 0.14793
##          Neg Pred Value : 0.83836
##              Prevalence : 0.16049
##          Detection Rate : 0.01235
##    Detection Prevalence : 0.08346
##       Balanced Accuracy : 0.49611
```

```
##
##        'Positive' Class : Attrited Customer
##
```

```
# Random Forest for regular data
# Train a random forest model with 500 trees using the regular training dataset
rf_reg <- randomForest(Attrition_Flag ~ ., ntree = 500, family = "binomial", data = train_data)

# Print summary statistics of the random forest model
print(summary(rf_reg))
```

```
##                 Length Class  Mode
## call               5   -none- call
## type               1   -none- character
## predicted       8102   factor numeric
## err.rate        1500   -none- numeric
## confusion          6   -none- numeric
## votes          16204   matrix numeric
## oob.times       8102   -none- numeric
## classes            2   -none- character
## importance        19   -none- numeric
## importanceSD       0   -none- NULL
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            14   -none- list
## y               8102   factor numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
## terms              3   terms  call
```

```
# Print the random forest model to inspect the model structure and parameters
rf_reg
```

```
##
## Call:
##  randomForest(formula = Attrition_Flag ~ ., data = train_data,      ntree = 500, family = "binomial"
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 3.76%
## Confusion matrix:
##                    Attrited Customer Existing Customer class.error
## Attrited Customer              1080               222  0.17050691
## Existing Customer                83              6717  0.01220588
```

```
# Predict the outcomes for the testing dataset using the trained model
rf_reg_pred <- predict(rf_reg, test_data)

# Print confusion matrix to evaluate the performance of the model
caret::confusionMatrix(rf_reg_pred, test_data$Attrition_Flag)
```

```
## Confusion Matrix and Statistics
##
```

```
##                    Reference
## Prediction          Attrited Customer Existing Customer
##   Attrited Customer               262                19
##   Existing Customer                63              1681
##
##                Accuracy : 0.9595
##                  95% CI : (0.95, 0.9677)
##     No Information Rate : 0.8395
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.841
##
##  Mcnemar's Test P-Value : 2.049e-06
##
##             Sensitivity : 0.8062
##             Specificity : 0.9888
##          Pos Pred Value : 0.9324
##          Neg Pred Value : 0.9639
##              Prevalence : 0.1605
##          Detection Rate : 0.1294
##    Detection Prevalence : 0.1388
##       Balanced Accuracy : 0.8975
##
##        'Positive' Class : Attrited Customer
##
```

```
#SVM on Regular data

# Create a formula for the model
formula <- Attrition_Flag ~ Customer_Age + Gender + Dependent_count + Education_Level +
  Marital_Status + Income_Category + Card_Category + Months_on_book +
  Total_Relationship_Count + Months_Inactive_12_mon + Contacts_Count_12_mon +
  Credit_Limit + Total_Revolving_Bal + Avg_Open_To_Buy + Total_Amt_Chng_Q4_Q1 +
  Total_Trans_Amt + Total_Trans_Ct + Total_Ct_Chng_Q4_Q1 + Avg_Utilization_Ratio

# Train the SVM model
svm_model <- svm(formula, data=train_data, kernel="radial", cost=1, scale=TRUE)

predictions <- predict(svm_model, newdata=test_data)

# Calculate the accuracy of the model
accuracy <- sum(predictions == test_data$Attrition_Flag) / length(predictions)
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.929876543209877"
```

```
# Display the confusion matrix
confusion_matrix <- table(Predicted=predictions, Actual=test_data$Attrition_Flag)
print(confusion_matrix)
```

```
##                    Actual
## Predicted           Attrited Customer Existing Customer
##   Attrited Customer               218                35
##   Existing Customer               107              1665
```

```r
#SVM on PCA data

# Create a formula for the model
formula <- Attrition_Flag ~ .

# Train the SVM model
svm_model <- svm(formula, data=training_pca, kernel="radial", cost=1, scale=TRUE)

predictions <- predict(svm_model, newdata=testing_pca)

# Calculate the accuracy of the model
accuracy <- sum(predictions == test_data$Attrition_Flag) / length(predictions)
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.768395061728395"
```

```r
# Display the confusion matrix
confusion_matrix <- table(Predicted=predictions, Actual=testing_pca$Attrition_Flag)
print(confusion_matrix)
```

```
##                    Actual
## Predicted           Attrited Customer Existing Customer
##   Attrited Customer               159                27
##   Existing Customer               166              1673
```

```r
#Naive Bayes on PCA

naive_bayes<- naiveBayes(Attrition_Flag ~ ., data= training_pca)
naive_bayes
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
## Attrited Customer Existing Customer
##         0.1607011         0.8392989
##
## Conditional probabilities:
##                    Gender
## Y                           F         M
##   Attrited Customer 0.5783410 0.4216590
##   Existing Customer 0.5252941 0.4747059
##
##                    Education_Level
## Y                       College  Doctorate   Graduate High School Post-Graduate
##   Attrited Customer 0.09062980 0.06144393 0.29569892  0.19662058    0.05453149
##   Existing Customer 0.10044118 0.04308824 0.30985294  0.20044118    0.04911765
##                    Education_Level
## Y                    Uneducated    Unknown
##   Attrited Customer 0.14516129 0.15591398
##   Existing Customer 0.14691176 0.15014706
```

```
##
##                    Marital_Status
## Y                     Divorced    Married     Single    Unknown
##   Attrited Customer 0.07834101 0.44086022 0.40168971 0.07910906
##   Existing Customer 0.07411765 0.47044118 0.38279412 0.07264706
##
##                    Income_Category
## Y                      $120K + $40K - $60K $60K - $80K $80K - $120K
##   Attrited Customer 0.07680492  0.17204301  0.10983103   0.14516129
##   Existing Customer 0.07044118  0.17558824  0.14294118   0.14852941
##                    Income_Category
## Y                   Less than $40K    Unknown
##   Attrited Customer     0.38709677 0.10906298
##   Existing Customer     0.35235294 0.11014706
##
##                    Card_Category
## Y                          Blue        Gold    Platinum      Silver
##   Attrited Customer 0.937788018 0.011520737 0.003840246 0.046850998
##   Existing Customer 0.928970588 0.011764706 0.001911765 0.057352941
##
##                    PC1
## Y                           [,1]     [,2]
##   Attrited Customer  0.009847941 1.477074
##   Existing Customer -0.011869104 1.626362
##
##                    PC2
## Y                          [,1]     [,2]
##   Attrited Customer -1.1456016 1.163076
##   Existing Customer  0.2221814 1.370563
##
##                    PC3
## Y                          [,1]     [,2]
##   Attrited Customer -0.34592980 1.244377
##   Existing Customer  0.07217633 1.351322
##
##                    PC4
## Y                          [,1]     [,2]
##   Attrited Customer  0.7719836 1.104188
##   Existing Customer -0.1373948 1.166909
##
##                    PC5
## Y                           [,1]     [,2]
##   Attrited Customer  0.055185261 1.261397
##   Existing Customer -0.007779051 1.085183
##
##                    PC6
## Y                          [,1]      [,2]
##   Attrited Customer  0.1616715 0.9794838
##   Existing Customer -0.0352820 1.0036383
##
##                    PC7
## Y                          [,1]      [,2]
##   Attrited Customer -0.25796523 0.8935708
##   Existing Customer  0.05723346 1.0031491
```

```
##
##                     PC8
## Y                           [,1]      [,2]
##   Attrited Customer -0.23673532 0.9188637
##   Existing Customer  0.04322711 0.9510744
##
##                     PC9
## Y                           [,1]      [,2]
##   Attrited Customer -0.49669523 0.9616842
##   Existing Customer  0.09321176 0.8517633
##
##                     PC10
## Y                           [,1]      [,2]
##   Attrited Customer -0.11706259 0.7957774
##   Existing Customer  0.01776927 0.7670369
```

```
nb_pred<- predict(naive_bayes, testing_pca)
caret::confusionMatrix(nb_pred, testing_pca$Attrition_Flag)
```

```
## Confusion Matrix and Statistics
##
##                     Reference
## Prediction           Attrited Customer Existing Customer
##   Attrited Customer               123                26
##   Existing Customer               202              1674
##
##                Accuracy : 0.8874
##                  95% CI : (0.8728, 0.9009)
##     No Information Rate : 0.8395
##     P-Value [Acc > NIR] : 5.09e-10
##
##                   Kappa : 0.465
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.37846
##             Specificity : 0.98471
##          Pos Pred Value : 0.82550
##          Neg Pred Value : 0.89232
##              Prevalence : 0.16049
##          Detection Rate : 0.06074
##    Detection Prevalence : 0.07358
##       Balanced Accuracy : 0.68158
##
##        'Positive' Class : Attrited Customer
##
```

```
#Naive Bayes for regular data

naive_bayes<- naiveBayes(Attrition_Flag ~ ., data= train_data)
naive_bayes
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
```

```
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
## Attrited Customer Existing Customer
##         0.1607011         0.8392989
##
## Conditional probabilities:
##                     Customer_Age
## Y                        [,1]     [,2]
##   Attrited Customer 46.81644 7.698245
##   Existing Customer 46.28000 8.076747
##
##                     Gender
## Y                            F         M
##   Attrited Customer 0.5645161 0.4354839
##   Existing Customer 0.5192647 0.4807353
##
##                     Dependent_count
## Y                        [,1]     [,2]
##   Attrited Customer 2.381720 1.278716
##   Existing Customer 2.331176 1.304173
##
##                     Education_Level
## Y                       College  Doctorate   Graduate High School Post-Graduate
##   Attrited Customer 0.08525346 0.05376344 0.30721966  0.18817204    0.05760369
##   Existing Customer 0.10088235 0.04029412 0.31397059  0.20000000    0.05264706
##                     Education_Level
## Y                     Uneducated    Unknown
##   Attrited Customer 0.15284178 0.15514593
##   Existing Customer 0.14029412 0.15191176
##
##                     Marital_Status
## Y                        Divorced    Married     Single    Unknown
##   Attrited Customer 0.07680492 0.43010753 0.41321045 0.07987711
##   Existing Customer 0.07500000 0.46691176 0.38661765 0.07147059
##
##                     Income_Category
## Y                       $120K + $40K - $60K $60K - $80K $80K - $120K
##   Attrited Customer 0.08218126  0.16897081  0.11751152   0.14592934
##   Existing Customer 0.07044118  0.17955882  0.14088235   0.15455882
##                     Income_Category
## Y                     Less than $40K    Unknown
##   Attrited Customer     0.38095238 0.10445469
##   Existing Customer     0.34470588 0.10985294
##
##                     Card_Category
## Y                            Blue        Gold    Platinum      Silver
##   Attrited Customer 0.937019969 0.012288786 0.002304147 0.048387097
##   Existing Customer 0.932352941 0.011323529 0.001617647 0.054705882
##
##                     Months_on_book
## Y                        [,1]     [,2]
```

```
##   Attrited Customer 36.30568 7.750646
##   Existing Customer 35.88471 8.052497
##
##                     Total_Relationship_Count
## Y                        [,1]     [,2]
##   Attrited Customer 3.274194 1.561723
##   Existing Customer 3.908382 1.528853
##
##                     Months_Inactive_12_mon
## Y                        [,1]     [,2]
##   Attrited Customer 2.693548 0.890088
##   Existing Customer 2.268971 1.013590
##
##                     Contacts_Count_12_mon
## Y                        [,1]     [,2]
##   Attrited Customer 2.983871 1.116111
##   Existing Customer 2.350441 1.082961
##
##                     Credit_Limit
## Y                        [,1]     [,2]
##   Attrited Customer 8042.462 9082.686
##   Existing Customer 8739.516 9122.628
##
##                     Total_Revolving_Bal
## Y                         [,1]     [,2]
##   Attrited Customer  667.0945 913.1183
##   Existing Customer 1253.7831 758.4376
##
##                     Avg_Open_To_Buy
## Y                        [,1]     [,2]
##   Attrited Customer 7375.367 9085.890
##   Existing Customer 7485.733 9119.683
##
##                     Total_Amt_Chng_Q4_Q1
## Y                         [,1]      [,2]
##   Attrited Customer 0.6953740 0.2153437
##   Existing Customer 0.7709824 0.2156971
##
##                     Total_Trans_Amt
## Y                        [,1]     [,2]
##   Attrited Customer 3084.426 2291.824
##   Existing Customer 4634.583 3497.879
##
##                     Total_Trans_Ct
## Y                        [,1]     [,2]
##   Attrited Customer 44.83180 14.53153
##   Existing Customer 68.52338 22.90380
##
##                     Total_Ct_Chng_Q4_Q1
## Y                         [,1]      [,2]
##   Attrited Customer 0.5559316 0.2219095
##   Existing Customer 0.7411901 0.2270544
##
##                     Avg_Utilization_Ratio
```

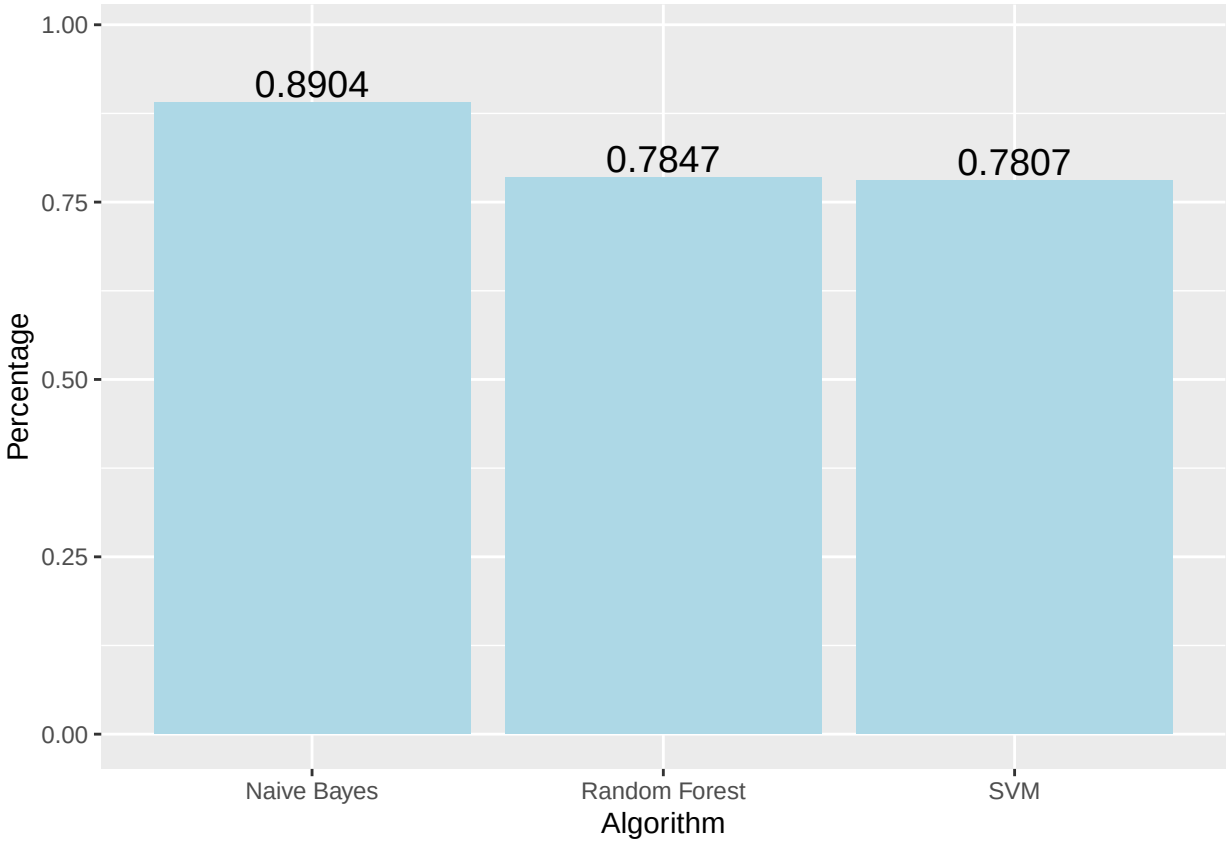
```
## Y                        [,1]      [,2]
##   Attrited Customer 0.1616728 0.2629379
##   Existing Customer 0.2956069 0.2723920
```

```
nb_pred<- predict(naive_bayes, test_data)
caret::confusionMatrix(nb_pred, testing_pca$Attrition_Flag)
```

```
## Confusion Matrix and Statistics
##
##                    Reference
## Prediction          Attrited Customer Existing Customer
##   Attrited Customer                41                279
##   Existing Customer               284               1421
##
##                Accuracy : 0.722
##                  95% CI : (0.7019, 0.7414)
##     No Information Rate : 0.8395
##     P-Value [Acc > NIR] : 1.0000
##
##                   Kappa : -0.0382
##
##  Mcnemar's Test P-Value : 0.8661
##
##             Sensitivity : 0.12615
##             Specificity : 0.83588
##          Pos Pred Value : 0.12813
##          Neg Pred Value : 0.83343
##              Prevalence : 0.16049
##          Detection Rate : 0.02025
##    Detection Prevalence : 0.15802
##       Balanced Accuracy : 0.48102
##
##        'Positive' Class : Attrited Customer
##
```

```
# Comparision of different models on PCA Data

H = c(0.7847,0.7807,0.8904)
names1 = c("Random Forest","SVM" , "Naive Bayes")
experiment <- data.frame(Algorithm = names1,
                         Percentage = H)
ggplot(data = experiment, mapping = aes(x=Algorithm, y=Percentage)) +
  geom_bar(stat="identity", position = "dodge",fill="lightblue") + scale_fill_brewer(palette = "Pastel2
  geom_text(aes(label = Percentage), vjust = -0.2, size = 5,
            position = position_dodge(0.9)) +
  ylim(0, max(experiment$Percentage)*1.1)
```


1.00
0.75
0.50
0.25
0.00
Percentage
0.8904
0.7847
0.7807
Naive Bayes
Random Forest
SVM
Algorithm