CMSC 691 – Introduction to Data Science

Assignment 2

Saketh Ram Vangeepuram
XV45698

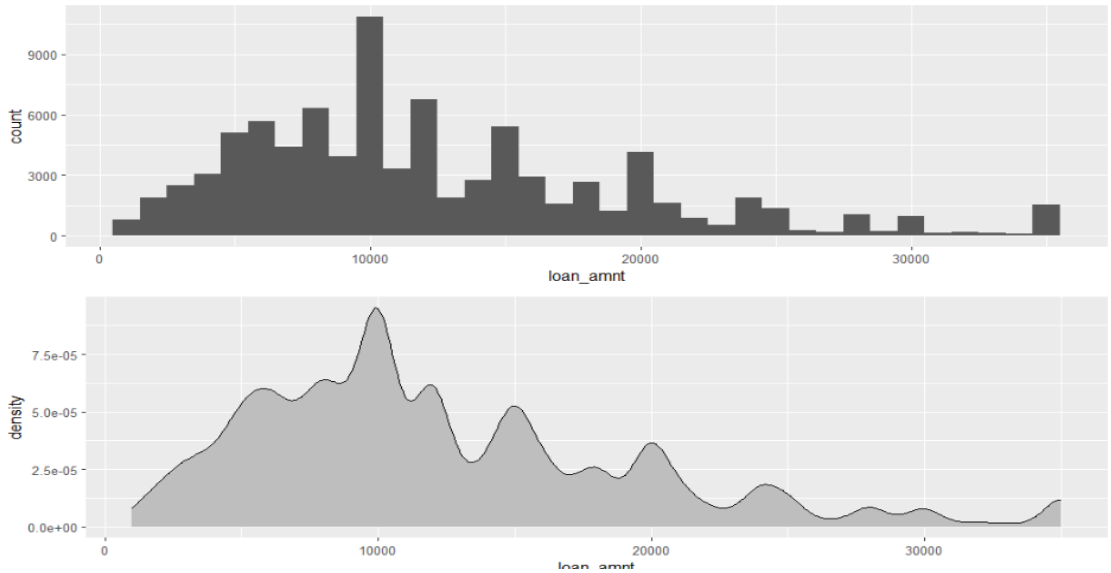
Part-1

NOTE: Please find the attached R scripts for 1. a, 1. b, and 1. C and 2.

1A. Exploratory Data Analysis

```
> glimpse(tble)
Rows: 88,451
Columns: 12
$ loan_default              <int> 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
$ loan_amnt                 <int> 15000, 8000, 14000, 4000, 18825, 10500, 6000, 19200, 10000, 16000, 6350, 12000, 650…
$ adjusted_annual_inc       <dbl> 41640, 64640, 29132, 25280, 28344, 21048, 32204, 44804, 65628, 24023, 27232, 51588,…
$ pct_loan_income           <dbl> 0.27777778, 0.10389610, 0.28000000, 0.08000000, 0.44821429, 0.31818182, 0.14668492,…
$ dti                       <dbl> 23.38, 2.76, 17.80, 29.04, 15.66, 13.20, 28.37, 23.01, 4.87, 31.71, 16.74, 12.15, 9…
$ residence_property        <chr> "Own", "Own", "Rent", "Rent", "Own", "Own", "Own", "Own", "Own", "Rent", "Own", "Ow…
$ months_since_first_credit <int> 390, 154, 86, 227, 115, 311, 181, 164, 77, 150, 156, 136, 71, 277, 105, 135, 210, 1…
$ inq_last_6mths            <int> 2, 0, 0, 0, 0, 1, 3, 1, 0, 0, 1, 1, 0, 0, 0, 3, 0, 1, 0, 1, 4, 1, 2, 1, 2, 1, 2, 0,…
$ open_acc                  <int> 18, 5, 12, 8, 9, 6, 11, 16, 8, 8, 8, 11, 6, 7, 16, 11, 11, 17, 13, 8, 24, 6, 14, 14…
$ bc_util                   <dbl> 62.1, 45.6, 52.8, 91.7, 79.4, 58.3, 98.8, 25.6, 31.1, 88.8, 54.2, 98.0, 84.8, 11.2,…
$ num_accts_ever_120_pd     <int> 0, 0, 0, 3, 1, 0, 1, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,…
$ pub_rec_bankruptcies      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,…
```

The distribution of Loan Amount using two different plots:

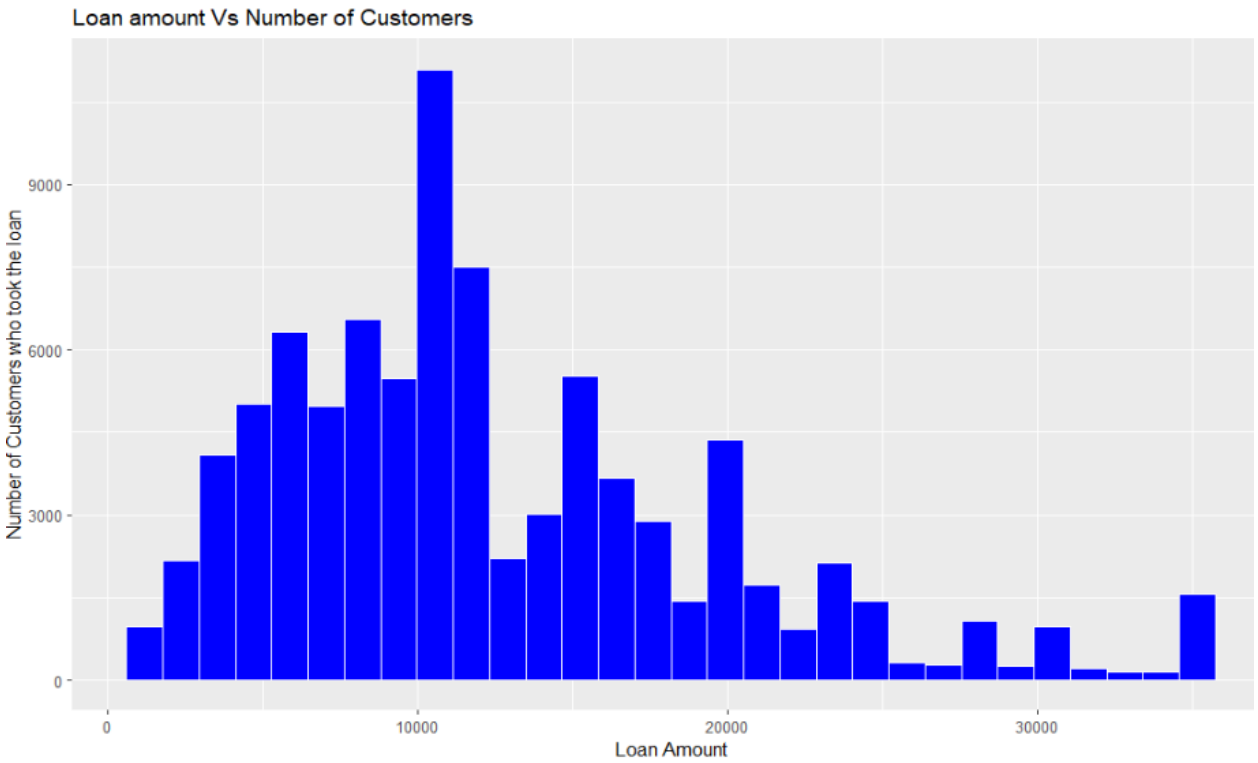


Summary of the data:

```
summary(df$loan_amnt)
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 1000    7200   10500   12435   16000   35000
```

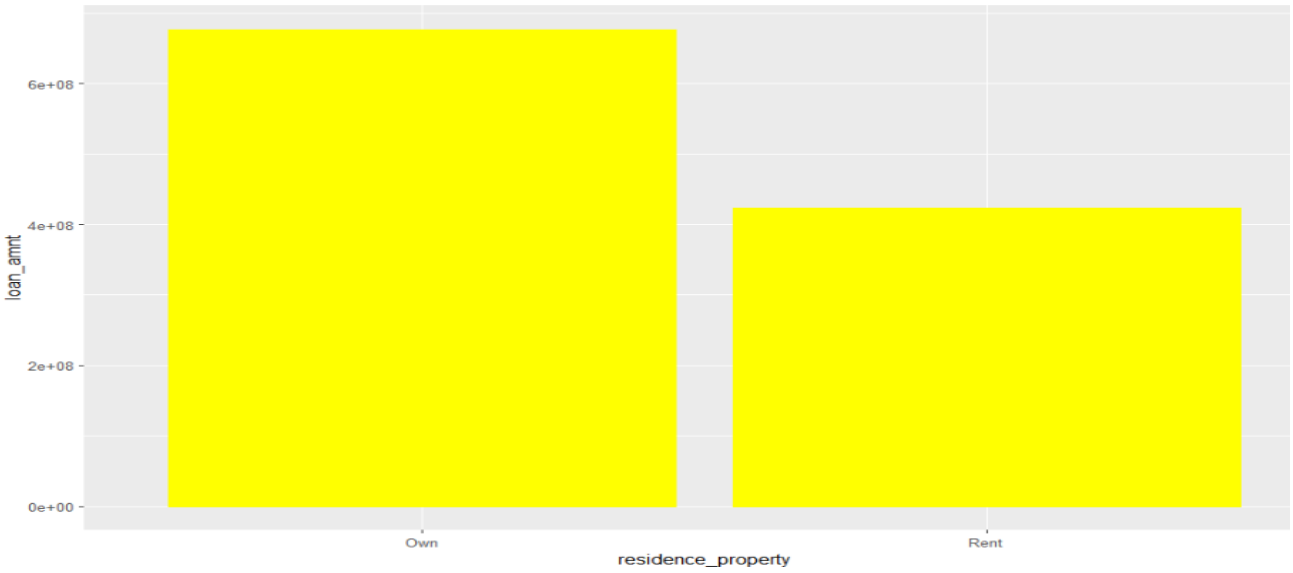
We can observe that the minimum loan amount is 1000 and the maximum amount is 35000.

Plotting a histogram to analyze the average loan amount taken by the customers:



Using the above plot, we can analyze that most customers took a loan amount between 10000 and 15000.

Plotting histogram to analyze whether or not customers who took a loan have owned property.



1B and 1C: Logistic Regression, Naive Bayes and KNN

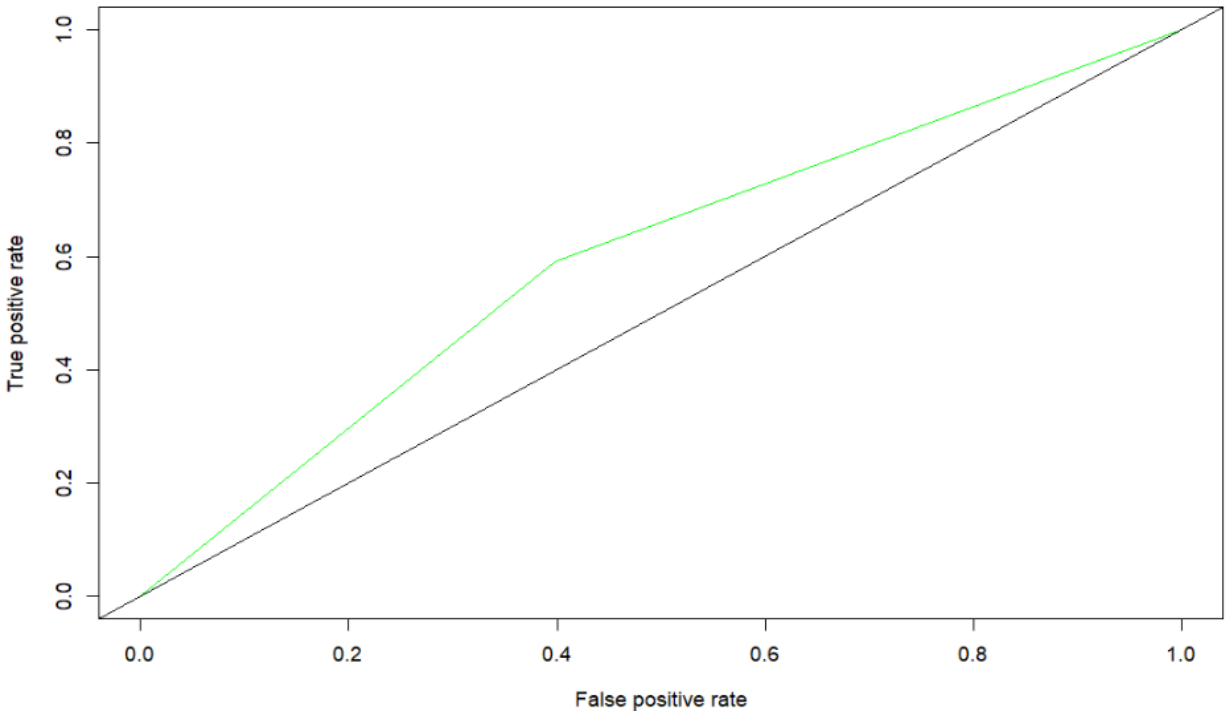
Logistic Regression :

```
> summary(logisticRegression_LendingData)

Call:
glm(formula = loan_default ~ pct_loan_income + loan_amnt + residence_property +
    inq_last_6mths + dti, family = "binomial", data = training_data_lendingData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8831  -1.1323   0.5763   1.1367   1.9341

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)        -6.232e-01  5.519e-02  -11.29   <2e-16 ***
pct_loan_income     3.093e+00  2.060e-01   15.02   <2e-16 ***
loan_amnt          -3.114e-05  2.927e-06  -10.64   <2e-16 ***
residence_property -3.255e-01  3.401e-02   -9.57   <2e-16 ***
inq_last_6mths      2.244e-01  1.612e-02   13.92   <2e-16 ***
dti                 1.964e-02  2.211e-03    8.88   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 21348  on 15398  degrees of freedom
Residual deviance: 20641  on 15393  degrees of freedom
AIC: 20653

Number of Fisher Scoring iterations: 4
```

Model output with Confusion Matrix:

```
> #It gives the information of the testing data when the response is predicted.
> testing_data_lendingData = testing_data_lendingData %>%
+   mutate(EstimatedProb = predict(logisticRegression_LendingData,
+                                  newdata = testing_data_lendingData, type = "response"))
>
> #Prediction summary of the test data.
> summary(testing_data_lendingData$EstimatedProb)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.1508  0.4255  0.4983  0.5012  0.5756  0.8592


> #printing the confusion matrix
> lendingDataset

     0    1
0 1988 1346
1 1314 1953


> confusionMatrix(lendingDataset)
Confusion Matrix and Statistics


     0    1
0 1988 1346
1 1314 1953

               Accuracy : 0.597
                 95% CI : (0.5851, 0.6089)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.1941

 Mcnemar's Test P-Value : 0.5478

            Sensitivity : 0.6021
            Specificity : 0.5920
         Pos Pred Value : 0.5963
         Neg Pred Value : 0.5978
             Prevalence : 0.5002
         Detection Rate : 0.3012
   Detection Prevalence : 0.5051
      Balanced Accuracy : 0.5970

       'Positive' Class : 0
```

ROC Curve:

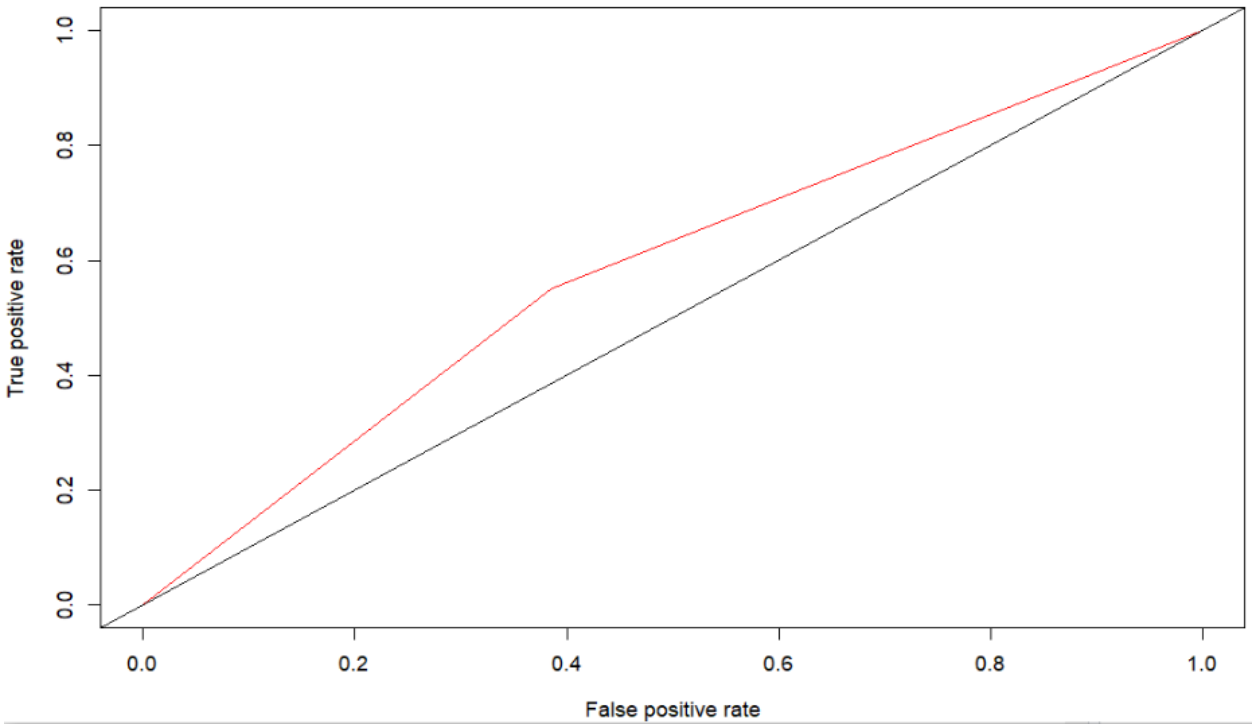


Naive Bayes Model Output with Confusion Matrix:

```
> confusionMatrix(confusion_matrix_lendingData)
Confusion Matrix and Statistics


predictions_naiveBayes    0    1
                     0 2027 1483
                     1 1275 1816

               Accuracy : 0.5822
                 95% CI : (0.5702, 0.5941)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.1643

 Mcnemar's Test P-Value : 8.094e-05

            Sensitivity : 0.6139
            Specificity : 0.5505
         Pos Pred Value : 0.5775
         Neg Pred Value : 0.5875
             Prevalence : 0.5002
         Detection Rate : 0.3071
   Detection Prevalence : 0.5317
      Balanced Accuracy : 0.5822

       'Positive' Class : 0
```

```
> confusion_matrix_lendingData

predictions_naiveBayes    0    1
                     0 2027 1483
                     1 1275 1816
```

ROC Curve:

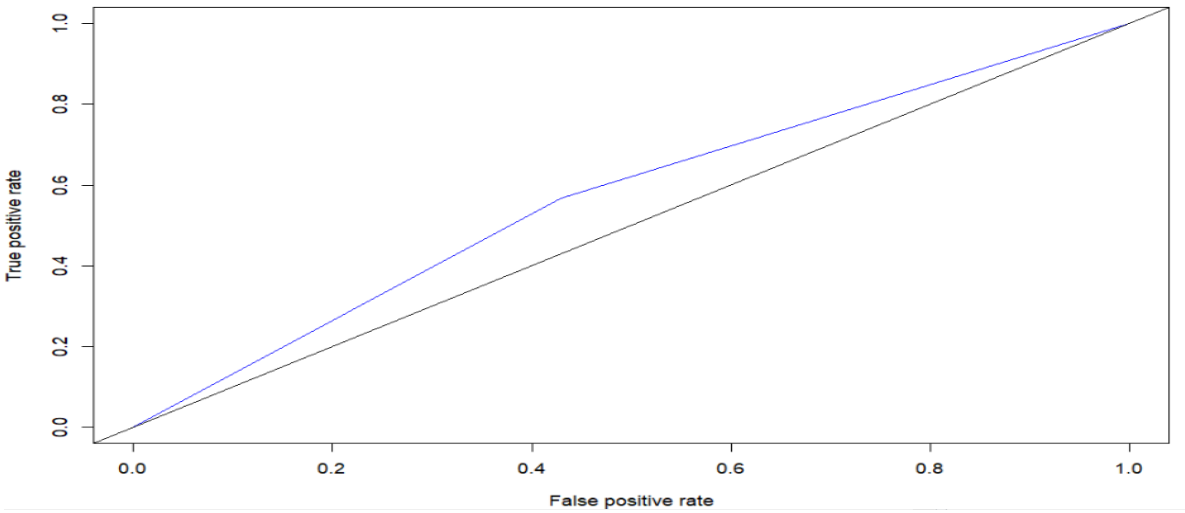


KNN Model output with Confusion Matrix:

```
> #Calculate the accuracy of created models
> acc.124 <- 100 * sum(test_labels == KNN_Pred124)/NROW(test_labels)
> acc.125 <- 100 * sum(test_labels == KNN_Pred125)/NROW(test_labels)
> acc.124
[1] 56.87017
> acc.125
[1] 56.97622
```

```
> confusion_matrix_KNN = table(KNN_Pred124 ,test_labels)
> confusionMatrix(confusion_matrix_KNN)
Confusion Matrix and Statistics

           test_labels
KNN_Pred124    0    1
          0 1886 1431
          1 1416 1868

               Accuracy : 0.5687
                 95% CI : (0.5567, 0.5807)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.1374

 Mcnemar's Test P-Value : 0.793

            Sensitivity : 0.5712
            Specificity : 0.5662
         Pos Pred Value : 0.5686
         Neg Pred Value : 0.5688
             Prevalence : 0.5002
         Detection Rate : 0.2857
   Detection Prevalence : 0.5025
      Balanced Accuracy : 0.5687

       'Positive' Class : 0
```

ROC Curve:



Observations :

- All three models, i,e KNN, Logistic Regression Model, and the Naïve Bayes Model used in the Lending dataset, have similar accuracies.
- The accuracy of the Logistic Regression model is 59.7 %. The accuracy of the Naive Bayes model is 58.22 %. And the accuracy of KNN is 56.87%.

Sensitivity:

Sensitivity is defined as the true positive rate i,e measures the proportion of actual positives which are correctly identified.

- We can see that the Sensitivity for the above models is:
  Logistic Regression: 60.21%
  Naive Bayes: 61.39%
  KNN: 57.12%

Specificity:

Specificity is defined as the true negative rate i,e it measures the proportion of the actual negatives which are correctly identified.

- We can see that the Specificity for the above models is:
  Logistic Regression: 59.20%
  Naive Bayes: 55.05%
  KNN: 56.62%

Note:

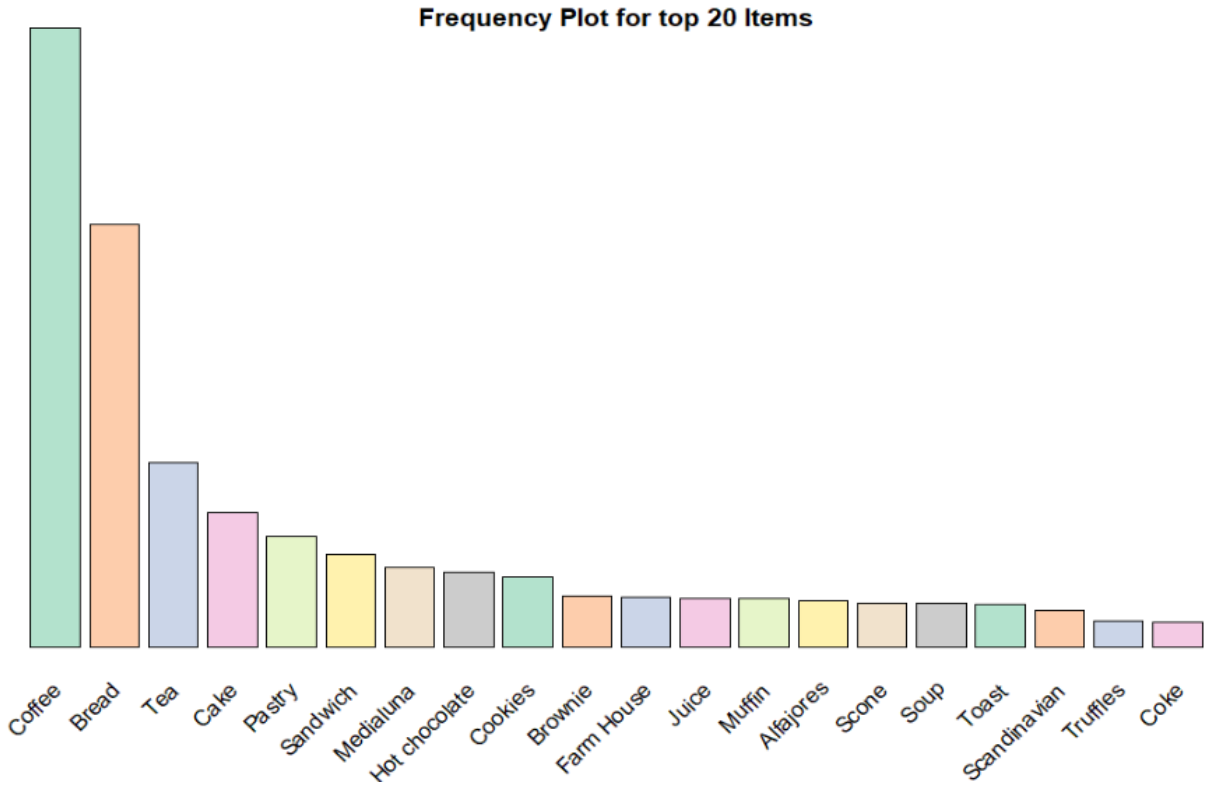
Considering all the parameters, we can say that the logistic regression model best fits the given dataset, with an accuracy of 59.7%.

2. Market Basket Analysis

Output from Market Basket Analysis:

```
> summary(read_transactions_BreadBasket)
transactions as itemMatrix in sparse format with
 9466 rows (elements/itemsets/transactions) and
 105 columns (items) and a density of 0.01898625

most frequent items:
 Coffee   Bread     Tea    Cake  Pastry (Other)
   4526    3094    1349     983     814    8105

element (itemset/transaction) length distribution:
sizes
   1    2    3    4    5    6    7    8    9   10
3955 3058 1469  662  231   64   17    4    5    1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   2.000   1.994   3.000  10.000

includes extended item information - examples:
                labels
1           Adjustment
2 Afternoon with the baker
3             Alfajores
```

Frequency Plot:



Association rules using apriori algorithm:

```
> rules_association<-apriori(read_transactions_BreadBasket,parameter=list(supp=0.001,conf=0.4,maxlen=5))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.4    0.1    1 none FALSE            TRUE       5   0.001      1      5  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 9

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[105 item(s), 9466 transaction(s)] done [0.00s].
sorting and recoding items ... [55 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [137 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

Summary of rules:

```
> summary(rules_association)
set of 137 rules

rule length distribution (lhs + rhs):sizes
  1   2   3   4
  1  30 103   3

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   3.000   3.000   2.788   3.000   4.000

summary of quality measures:
    support           confidence        coverage             lift            count
 Min.   :0.001056   Min.   :0.4000   Min.   :0.001268   Min.   : 0.8366   Min.   :  10.00
 1st Qu.:0.001373   1st Qu.:0.4800   1st Qu.:0.002324   1st Qu.: 1.0321   1st Qu.:  13.00
 Median :0.001901   Median :0.5455   Median :0.003592   Median : 1.1712   Median :  18.00
 Mean   :0.008520   Mean   :0.5637   Mean   :0.016723   Mean   : 1.9173   Mean   :  80.65
 3rd Qu.:0.003909   3rd Qu.:0.6263   3rd Qu.:0.007289   3rd Qu.: 1.3595   3rd Qu.:  37.00
 Max.   :0.478132   Max.   :0.8750   Max.   :1.000000   Max.   :43.1815   Max.   :4526.00

mining info:
                       data ntransactions support confidence
 read_transactions_BreadBasket          9466   0.001        0.4
                                                                                                call
 apriori(data = read_transactions_BreadBasket, parameter = list(supp = 0.001, conf = 0.4, maxlen = 5))
```

Inspecting rules: (This snapshot only has a few rules; please run R-script to print all the rules)

```
> inspect(rules_association)
     lhs                          rhs          support     confidence coverage    lift       count
[1]  {}                        => {Coffee}    0.478132263 0.4781323  1.000000000  1.0000000 4526
[2]  {Eggs}                    => {Bread}     0.001478977 0.5000000  0.002957955  1.5297350   14
[3]  {Granola}                 => {Coffee}    0.001795901 0.6071429  0.002957955  1.2698220   17
[4]  {Tartine}                 => {Coffee}    0.003063596 0.6304348  0.004859497  1.3185364   29
[5]  {Bakewell}                => {Coffee}    0.003063596 0.6041667  0.005070780  1.2635974   29
[6]  {Vegan mincepie}          => {Coffee}    0.003169237 0.5769231  0.005493345  1.2066182   30
[7]  {Art Tray}                => {Coffee}    0.002746672 0.6842105  0.004014367  1.4310068   26
[8]  {Extra Salami or Feta}    => {Salad}     0.001690260 0.4210526  0.004014367 40.2594365   16
[9]  {Extra Salami or Feta}    => {Coffee}    0.003274879 0.8157895  0.004014367  1.7062004   31
[10] {Keeping It Local}        => {Coffee}    0.005387703 0.8095238  0.006655398  1.6930960   51
[11] {The Nomad}               => {Coffee}    0.003274879 0.5344828  0.006127192  1.1178555   31
[12] {Frittata}                => {Coffee}    0.004542573 0.5308642  0.008556941  1.1102873   43
[13] {Smoothies}               => {Coffee}    0.004014367 0.4935065  0.008134376  1.0321548   38
[14] {Hearty & Seasonal}       => {Coffee}    0.005704627 0.5400000  0.010564124  1.1293946   54
```

Filtering rules such that the rules do not have coffee on the Right Hand Side:

```
> summary(SubRulesNoCoffee)
set of 7 rules

rule length distribution (lhs + rhs):sizes
2 3
2 5

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  2.000   2.500   3.000   2.714   3.000   3.000

summary of quality measures:
    support           confidence        coverage             lift           count
 Min.   :0.001056   Min.   :0.4167   Min.   :0.002218   Min.   : 1.275   Min.   :10.00
 1st Qu.:0.001056   1st Qu.:0.4248   1st Qu.:0.002483   1st Qu.: 1.556   1st Qu.:10.00
 Median :0.001479   Median :0.4348   Median :0.002958   Median : 3.007   Median :14.00
 Mean   :0.001404   Mean   :0.4564   Mean   :0.003094   Mean   : 9.370   Mean   :13.29
 3rd Qu.:0.001637   3rd Qu.:0.4881   3rd Qu.:0.003539   3rd Qu.: 8.968   3rd Qu.:15.50
 Max.   :0.001902   Max.   :0.5172   Max.   :0.004437   Max.   :40.259   Max.   :18.00

mining info:
                       data ntransactions support confidence
 read_transactions_BreadBasket          9466   0.001        0.4

call
 apriori(data = read_transactions_BreadBasket, parameter = list(supp = 0.001, conf = 0.4, maxlen = 5), appearanc
e = list(none = (rhs = "Coffee")))
```

```
> inspect(SubRulesNoCoffee)
    lhs                        rhs         support     confidence coverage    lift      count
[1] {Eggs}                  => {Bread}     0.001478977 0.5000000  0.002957955  1.529735 14
[2] {Extra Salami or Feta}  => {Salad}     0.001690260 0.4210526  0.004014367 40.259436 16
[3] {Cake, Jammie Dodgers}  => {Bread}     0.001584619 0.5172414  0.003063596  1.582484 15
[4] {Jammie Dodgers, Tea}   => {Bread}     0.001056412 0.4166667  0.002535390  1.274779 10
[5] {Coke, Juice}           => {Sandwich}  0.001056412 0.4761905  0.002218466  6.628852 10
[6] {Cake, Soup}            => {Tea}       0.001901542 0.4285714  0.004436932  3.007307 18
[7] {Alfajores, Cookies}    => {Juice}     0.001056412 0.4347826  0.002429749 11.306737 10
```

Top 5 rules sorted by SUPPORT:

```
> inspect(rules_association[1:5])
    lhs                        rhs         support     confidence coverage    lift      count
[1] {Eggs}                  => {Bread}     0.001478977 0.5000000  0.002957955  1.529735 14
[2] {Extra Salami or Feta}  => {Salad}     0.001690260 0.4210526  0.004014367 40.259436 16
[3] {Cake, Jammie Dodgers}  => {Bread}     0.001584619 0.5172414  0.003063596  1.582484 15
[4] {Jammie Dodgers, Tea}   => {Bread}     0.001056412 0.4166667  0.002535390  1.274779 10
[5] {Coke, Juice}           => {Sandwich}  0.001056412 0.4761905  0.002218466  6.628852 10
```

Interpreting association rules:

- Among the top 5 rules, my favorite rule is the first one, which explains:
    - When Eggs are brought, Bread Is bought with a confidence of 50%.
- Also, the second rule states that if Extra Salami or Feta is bought, Salad is purchased with a confidence of 42.10%.