## Contents



# Apptopia

DATA SCIENCE CHALLENGE


Sai Dheeraj Illendula

# 1. Executive Summary

- The following pages will demonstrate the steps taken while building the model and also R code and results. Appendix section contains different plots and other analysis.
- Demonstrated two approaches in building the model.
    1) Traditional - (Data Set partition, Model training, Model Execution)
    2) Using Cross Validation – (K-fold = **4**)
- Best Features for the model:
    1) Top 4 features - **Country, total_estimated_installs, operating_system, app_subcategory**
    2) Top 4 features – **Country, total_estimated_installs, operating_system, app_subcategory**
- Metric used for the model is RMSE (Root Mean Square Error).
- Model is validated using k fold cross-validation (Approach 2)
- Missing values (NA) are removed as it is less than **2%** of dataset. These values are predicted using the model built. Code is included in Appendix

Best model is obviously the one built using cross-validation (avoids over-fitting). First approach is included just to demonstrate the traditional method.

Algorithm used in building this predictive model is ‘Random Forest’. It is a robust model which can handle both categorical and continuous variables. Variance caused if using decision trees can be avoided using ‘Random Forest’ (Ensemble of Decision trees), which addresses variance/bias tradeoff perfectly.

## 2. R Packages

Different packages are used for building this model. These are used for data exploration and manipulation, visualization, algorithms etc.

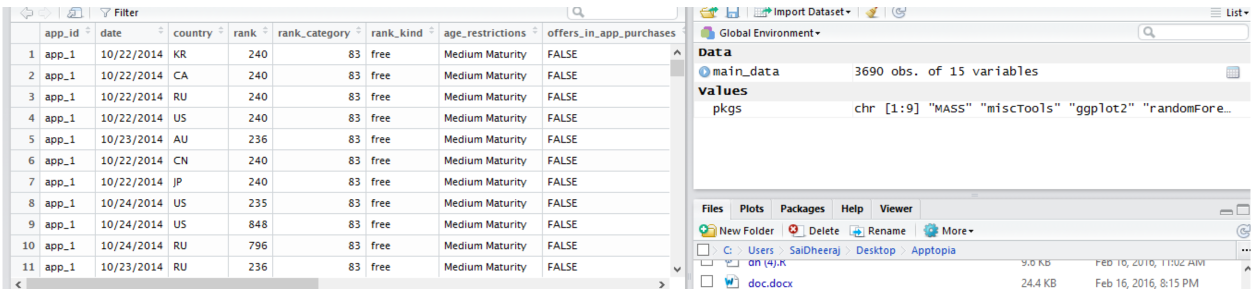
Loading Required Packages

```
> pkgs = c("MASS","miscTools","ggplot2","randomForest","dplyr","caret","stats
","plyr","gridExtra")
> lapply(pkgs, require, character.only = TRUE)
[[1]]
[1] TRUE

[[2]]
[1] TRUE

[[3]]
[1] TRUE

[[4]]
[1] TRUE

[[5]]
[1] TRUE

[[6]]
[1] TRUE

[[7]]
[1] TRUE

[[8]]
[1] TRUE

[[9]]
[1] TRUE
```

# 3. Data Import

Importing Data set to R

```
> main_data = read.csv("train (data sci challenge).csv")
```




# 4. Data Exploration and Manipulation

## 4.1 Dimension and Structure of Data

```
> dim(main_data)
[1] 3690   15
```

```
> str(main_data)
'data.frame':   3690 obs. of 15 variables:
 $ app_id                 : Factor w/ 77 levels "app_1","app_10",..: 1 1 1 1
1 1 1 1 1 1 ...
 $ date                   : Factor w/ 3 levels "10/22/2014","10/23/2014",..:
1 1 1 1 2 1 1 3 3 3 ...
 $ country                : Factor w/ 11 levels "AU","CA","CN",..: 9 2 10 11
1 3 8 11 11 10 ...
 $ rank                   : int  240 240 240 240 236 240 240 235 848 796 ...
 $ rank_category          : int  83 83 83 83 83 83 83 83 83 83 ...
 $ rank_kind              : Factor w/ 5 levels "free","grossing",..: 1 1 1 1
1 1 1 1 1 1 ...
 $ age_restrictions       : Factor w/ 4 levels "Everyone","High Maturity",..
: 4 4 4 4 4 4 4 4 4 4 ...
 $ offers_in_app_purchases : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ paid                   : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ us_price               : int  0 0 0 0 0 0 0 0 0 0 ...
 $ app_category           : int  38 38 38 38 38 38 38 38 38 38 ...
 $ app_subcategory        : int  83 83 83 83 83 83 83 83 83 83 ...
 $ total_estimated_installs: Factor w/ 12 levels "1,000 - 5,000",..: 9 9 9 9
9 9 9 9 9 9 ...
 $ operating_system       : Factor w/ 12 levels "1.1 and up","1.5 and up",..
: 6 6 6 6 6 6 6 6 6 6 ...
 $ downloads              : int  1 6 3 95 6 0 0 85 85 1 ...
```

Separating 'NA' values:

```
> row.has.na <- apply(main_data, 1, function(x){any(is.na(x))})
> sum(row.has.na)
[1] 68
> data.un <- main_data[row.has.na,]
> dim(data.un)
[1] 68 15
> main_data <- main_data[!row.has.na,]
> dim(main_data)
[1] 3622   15
```

## 4.2 Data Type Coversion

```
> main_data$rank_category = factor(main_data$rank_category)
> class(main_data$rank_category)
[1] "factor"
```

```
> main_data$app_category = factor(main_data$app_category)
> class(main_data$app_category)
[1] "factor"
```

```
> main_data$app_subcategory = factor(main_data$app_subcategory)
> class(main_data$app_subcategory)
[1] "factor"
```

Rationale:

rank_category, app_category and app_subcategory are categorical variables, the file contains these as numeric variables. It is required to convert these variables to categorical.

Extracting day from date while converting it to 'date' format

```
> main_data$day <- format(as.Date(main_data$date,format="%m/%d/%y"), "%d")
> class(main_data$day)
[1] "character"
> main_data$day <- as.integer(main_data$day)
> class(main_data$day)
[1] "integer"
```

Eliminating columns 'date' and 'app_id'

```
> main_subset = subset(main_data, select =  -c(app_id,date))
```

Rationale:

Date – As the whole data is only from three dates (10/22/2014, 10/23/2014, 10/24/2014), it is logical to extract date (unique value) and remove 'Date' column. Also model can understand 'day' well than 'date'

App_id – As the test data contains new app_id's (that are not in train data). Building the model using app_id will not be helpful. Hence removed 'app_id' column

## 4.3 Feature Engineering

a. 'Rank'

- Normalizing 'rank' column

Current range of values for 'rank' column

```
> summary(main_data$rank)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    4.0   197.0   359.0   399.4   547.0  1237.0 


> main_subset$ranks <- (main_subset$rank - min(main_subset$rank)) / (max(main
_subset$rank) - min(main_subset$rank))
> summary(main_subset$ranks)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.1565  0.2879  0.3207  0.4404  1.0000 
```

- Rationale: As the range of values is high, it is better to normalize rank column.

b. Rank_Category

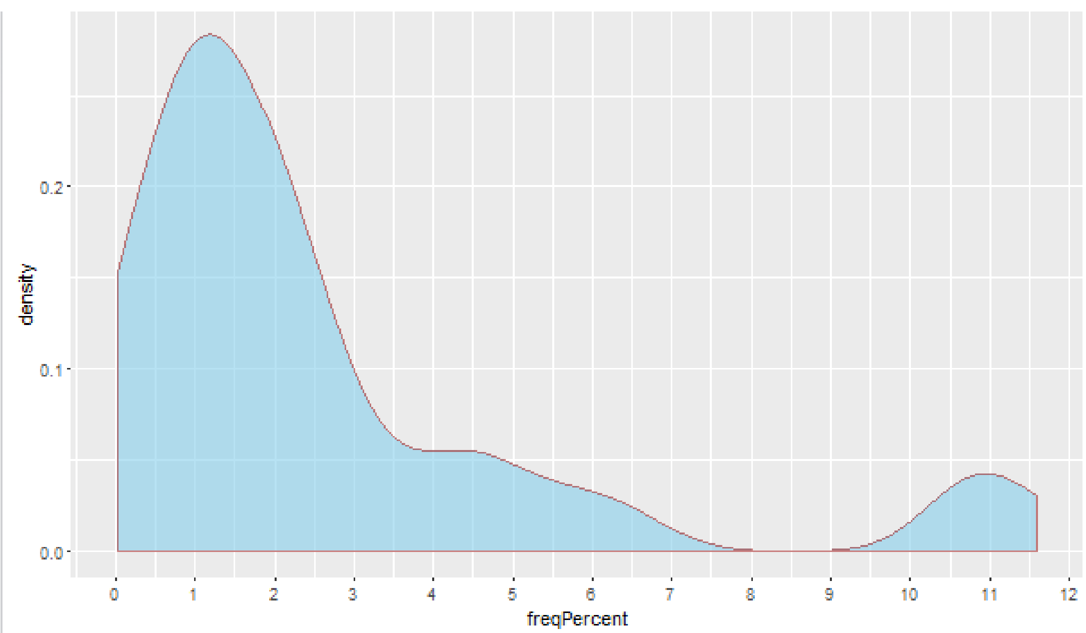
Binning rank_category

```
> freqDistRankCat = count(main_subset, vars = "rank_category")
> freqDistRankCat$freqPercent = round((freqDistRankCat$freq / sum(freqDistRan
kCat$freq)*100),digits = 2)
> freqDistRankCat
         rank_category freq freqPercent
      1              4   60        1.66
      2              5  396       10.93
      3              6   70        1.93
      4              8   34        0.94
      5              9  130        3.59
      6             10   81        2.24
      7             11    2        0.06
      8             12   74        2.04
      9             13    9        0.25
      10            14   44        1.21
      11            15    4        0.11
      12            16   72        1.99
      13            17    2        0.06
      14            18   46        1.27
      15            20   73        2.02
      16            21   96        2.65
      17            23   68        1.88
      18            24  157        4.33
      19            25   36        0.99
```

```
20                26   48         1.33
21                27  163         4.50
22                28  114         3.15
23                29  398        10.99
24                30   32         0.88
25                38   36         0.99
26                39    1         0.03
27                83  209         5.77
28                84   52         1.44
29                85   41         1.13
30                86   28         0.77
31                88  237         6.54
32                91  109         3.01
33                92   87         2.40
34                95   94         2.60
35                96   16         0.44
36                97  428        11.82
37                98   37         1.02
38                99   38         1.05
> summary(freqDistRankCat$freqPercent)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.030   0.990   1.770   2.632   2.920  11.820
```

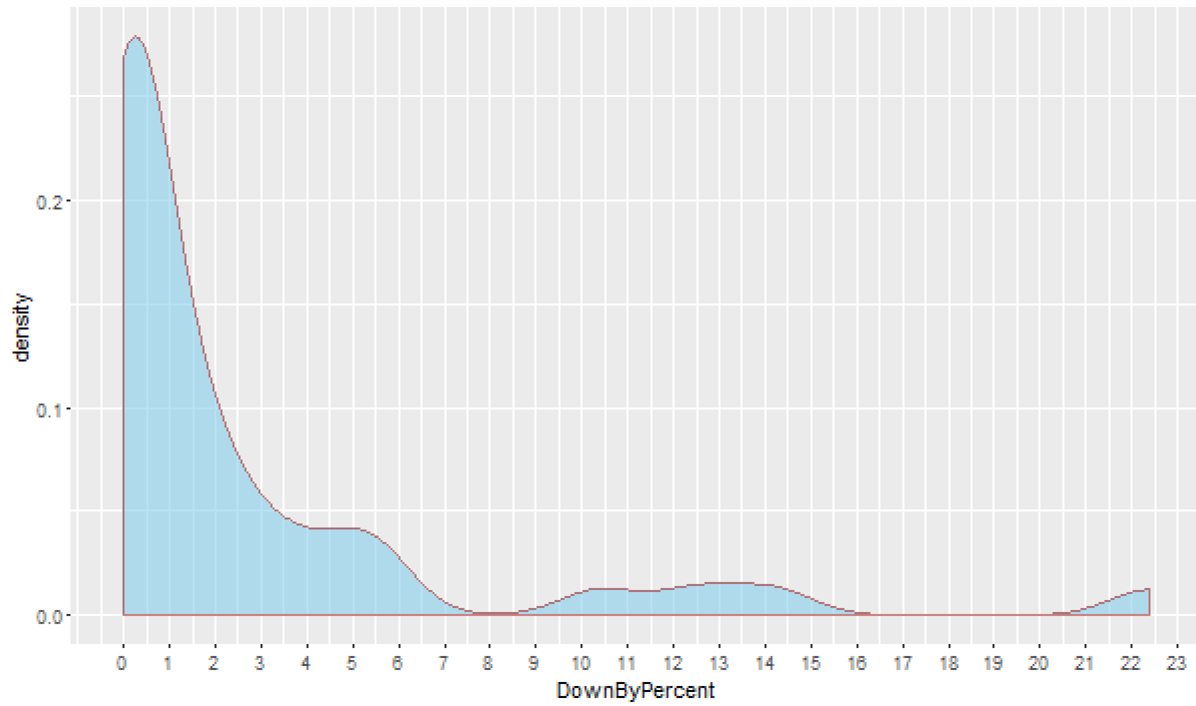
Density plot for rank_category by count

```
> densityFreqDistRankCat = ggplot(data = freqDistRankCat, aes(x = freqPercent
))
> densityFreqDistRankCat = densityFreqDistRankCat + geom_density(fill = "skyb
lue",alpha = 0.6, color = "brown")
> densityFreqDistRankCat = densityFreqDistRankCat + scale_x_continuous(breaks
= 0:13)
> densityFreqDistRankCat
```



Frequency distribution for rank_category by downloads

```
> downloadsByRankCat
   Rank_Category SumOfDownloads DownByPercent
1              4             42          0.11
2              5           8426         22.40
3              6            230          0.61
4              8             42          0.11
5              9            866          2.30
6             10            256          0.68
7             11              0          0.00
8             12             11          0.03
9             13              0          0.00
10            14           1071          2.85
11            15              0          0.00
12            16            719          1.91
13            17              0          0.00
14            18            526          1.40
15            20            111          0.30
16            21            261          0.69
17            23           5330         14.17
18            24           4699         12.49
19            25             41          0.11
20            26              9          0.02
21            27           3908         10.39
22            28            664          1.77
23            29           1849          4.92
24            30             45          0.12
25            38             24          0.06
26            39              2          0.01
27            83           1526          4.06
28            84             43          0.11
29            85           2049          5.45
30            86             16          0.04
31            88           2118          5.63
32            91             36          0.10
33            92             60          0.16
34            95            723          1.92
35            96              3          0.01
36            97           1271          3.38
37            98            349          0.93
38            99            289          0.77
> summary(downloadsByRankCat$DownByPercent)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.000   0.070   0.645   2.632   2.713  22.400 
> densityDownsRankCat = ggplot(data = downloadsByRankCat, aes(x = DownByPerce
nt))
> densityDownsRankCat =  densityDownsRankCat + geom_density(fill = "skyblue",
color = "brown", alpha = 0.6)
> densityDownsRankCat = densityDownsRankCat + scale_x_continuous(breaks = 0:3
7)
> densityDownsRankCat
```




```
> for( i in 1:nrow(downloadsByRankCat)){
+   tempValue = downloadsByRankCat$DownByPercent[i]
+   if(tempValue >= 0 && tempValue <= 1){
+     downloadsByRankCat$Bin_RankCat[i] = "Bin_RankCat_1"
+   }else if(tempValue > 1 && tempValue <= 4){
+     downloadsByRankCat$Bin_RankCat[i] = "Bin_RankCat_2"
+   }else if(tempValue > 4 && tempValue <= 8){
+     downloadsByRankCat$Bin_RankCat[i] = "Bin_RankCat_3"
+   }else if(tempValue > 8 && tempValue <= 16){
+     downloadsByRankCat$Bin_RankCat[i] = "Bin_RankCat_4"
+   }else if(tempValue >= 20){
+     downloadsByRankCat$Bin_RankCat[i] = "Bin_RankCat_5"
+   }
+ }
```

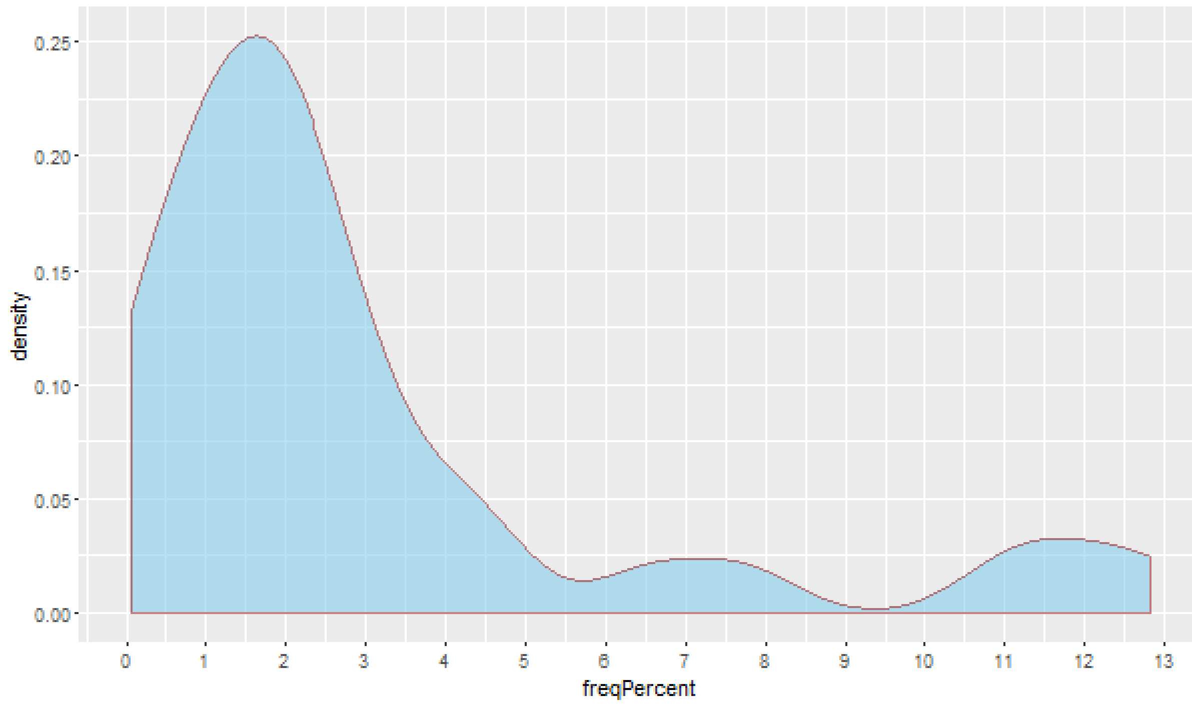
Rationale:

- Levels of rank_category are wide spread and also do not levels in a specific range. When this type of situation occurs, it will be better to bin the variable by observing its behavior (pattern) with response variable.
- Frequency of each level is obtained with respective to its count and also with downloads.
- Density plot for these frequency distribution is plotted.
- Though both density plots have similar pattern, bins are created by observing the density plot of rank_category variable with downloads.
- Bins are split at percentages (break points) at 4, 8, 16, 20 resulting in 5 bins by observing the pattern in the plot.

c. App_subcategory

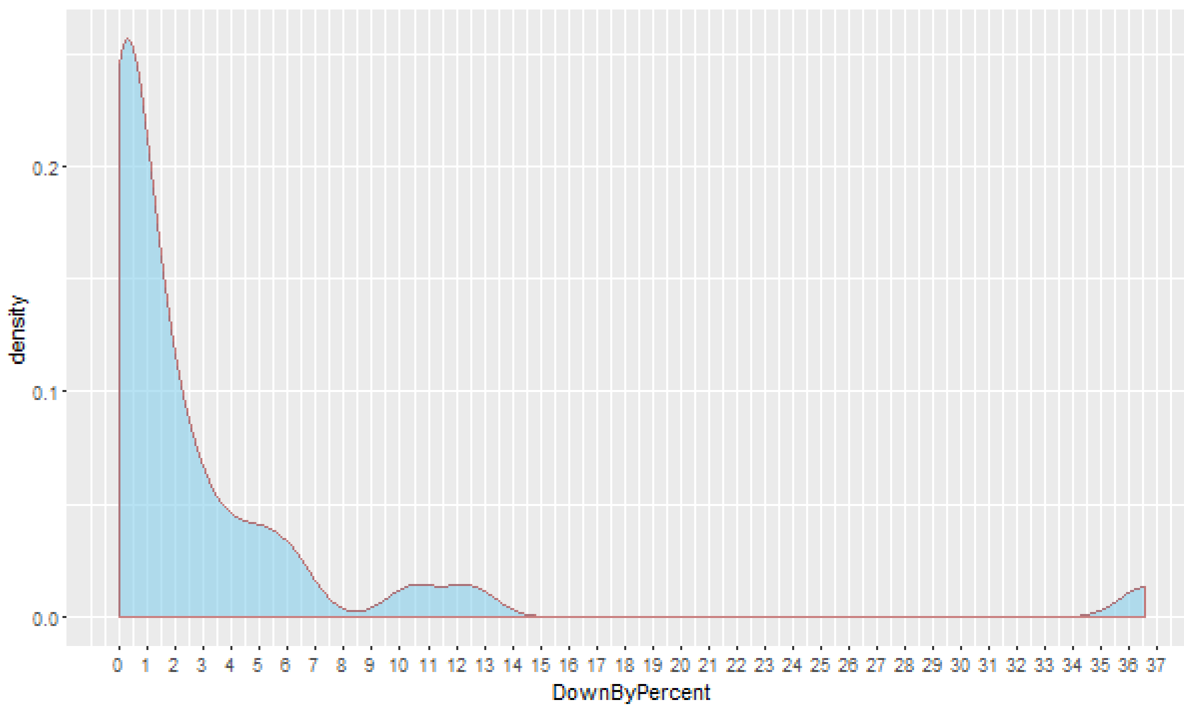
Binning app_subcategory

```
> freqDistAppSubCat = count(main_subset, vars = "app_subcategory")
> freqDistAppSubCat$freqPercent = round((freqDistAppSubCat$freq / sum(freqDis
tAppSubCat$freq)*100),digits = 2)
> freqDistAppSubCat
   app_subcategory freq freqPercent
1                4   60        1.66
2                6   70        1.93
3                9  130        3.59
4               10   81        2.24
5               11    2        0.06
6               12   74        2.04
7               13    9        0.25
8               14   45        1.24
9               15    4        0.11
10              16   72        1.99
11              17    2        0.06
12              18   46        1.27
13              20   73        2.02
14              21   96        2.65
15              23  464       12.81
16              24  157        4.33
17              25   70        1.93
18              26   48        1.33
19              27  163        4.50
20              28  114        3.15
21              29  398       10.99
22              30   32        0.88
23              83  281        7.76
24              84   52        1.44
25              85   42        1.16
26              86   28        0.77
27              88  237        6.54
28              91  109        3.01
29              92   87        2.40
30              95   94        2.60
31              96   16        0.44
32              97  428       11.82
33              99   38        1.05
> summary(freqDistAppSubCat$freqPercent)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.060   1.160   1.990   3.031   3.150  12.810 


> densityFreqDistAppSubCat = ggplot(data = freqDistAppSubCat, aes(x = freqPer
cent))
> densityFreqDistAppSubCat = densityFreqDistAppSubCat + geom_density(fill = "
skyblue",alpha = 0.6, color = "brown")
> densityFreqDistAppSubCat = densityFreqDistAppSubCat + scale_x_continuous(br
eaks = 0:13)
> densityFreqDistAppSubCat
```




```
> downloadsByAppSubCat = aggregate(main_subset$downloads, by = list(App_SubCa
tegory = main_subset$app_subcategory), FUN = sum)
> names(downloadsByAppSubCat)[2] = "SumOfDownloads"
> downloadsByAppSubCat$DownByPercent = round((downloadsByAppSubCat$SumOfDownl
oads /sum(downloadsByAppSubCat$SumOfDownloads)*100),digits = 2)
> downloadsByAppSubCat
   App_SubCategory SumOfDownloads DownByPercent
1                4             42          0.11
2                6            230          0.61
3                9            866          2.30
4               10            256          0.68
5               11              0          0.00
6               12             11          0.03
7               13              0          0.00
8               14           1073          2.85
9               15              0          0.00
10              16            719          1.91
11              17              0          0.00
12              18            526          1.40
13              20            111          0.30
14              21            261          0.69
15              23          13756         36.57
16              24           4699         12.49
17              25             83          0.22
18              26              9          0.02
19              27           3908         10.39
20              28            664          1.77
21              29           1849          4.92
22              30             45          0.12
23              83           1574          4.18
24              84             43          0.11
25              85           2374          6.31
26              86             16          0.04
```

```
27                88           2118           5.63
28                91             36           0.10
29                92             60           0.16
30                95            723           1.92
31                96              3           0.01
32                97           1271           3.38
33                99            289           0.77
> summary(downloadsByAppSubCat$DownByPercent)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.00    0.10    0.68    3.03    2.85   36.57


> densityDownsAppSubCat = ggplot(data = downloadsByAppSubCat, aes(x = DownByP
ercent))
> densityDownsAppSubCat =  densityDownsAppSubCat + geom_density(fill = "skybl
ue", color = "brown", alpha = 0.6)
> densityDownsAppSubCat = densityDownsAppSubCat + scale_x_continuous(breaks =
0:37)
> densityDownsAppSubCat
```




```
for( i in 1:nrow(downloadsByAppSubCat)){
  tempValue = downloadsByAppSubCat$DownByPercent[i]
  if(tempValue >= 0 && tempValue <= 1){
    downloadsByAppSubCat$Bin_AppSubCat[i] = "Bin_AppSubCat_1"
  }else if(tempValue > 1 && tempValue <= 4){
    downloadsByAppSubCat$Bin_AppSubCat[i] = "Bin_AppSubCat_2"
  }else if(tempValue > 4 && tempValue <= 8){
    downloadsByAppSubCat$Bin_AppSubCat[i] = "Bin_AppSubCat_3"
  }else if(tempValue > 8 && tempValue <= 15){
    downloadsByAppSubCat$Bin_AppSubCat[i] = "Bin_AppSubCat_4"
  }else if(tempValue >= 30){
    downloadsByAppSubCat$Bin_AppSubCat[i] = "Bin_AppSubCat_5"
  }}
```

Rationale:

- Similar approach which is followed for rank_category is implemented for app_subcategory.
- There is high correlation between these two variables which can be easily observed with plots.

Assigning rows to bins created

```
> for( i in 1:nrow(main_subset)){
+   # Get the category
+   app_subcategory_temp = main_subset$app_subcategory[i]
+   rank_category_temp = main_subset$rank_category[i]
+   # Get the bins for these categories from downloadsByAppSubCat and downloa
dsByRankCat tables
+   main_subset$Bin_AppSubCat[i] = (filter(downloadsByAppSubCat, App_SubCateg
ory == app_subcategory_temp))$Bin_AppSubCat
+   main_subset$Bin_RankCat[i] = (filter(downloadsByRankCat, Rank_Category ==
rank_category_temp))$Bin_RankCat
+ }
```

Conversion to 'Factor'

```
> main_subset$Bin_AppSubCat = as.factor(main_subset$Bin_AppSubCat)
> main_subset$Bin_RankCat = as.factor(main_subset$Bin_RankCat)
```

Remove rank,rank_category, app_subcategory

```
> main_subset = subset(main_subset, select = -c(rank,rank_category,app_subcat
egory))
> str(main_subset)
'data.frame':   3622 obs. of  14 variables:
 $ country                : Factor w/ 11 levels "AU","CA","CN",..: 9 2 10 11
1 3 8 11 11 10 ...
 $ rank_kind              : Factor w/ 5 levels "free","grossing",..: 1 1 1 1
1 1 1 1 1 1 ...
 $ age_restrictions       : Factor w/ 4 levels "Everyone","High Maturity",..
: 4 4 4 4 4 4 4 4 4 4 ...
 $ offers_in_app_purchases : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ paid                   : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ us_price               : int  0 0 0 0 0 0 0 0 0 0 ...
 $ app_category           : Factor w/ 2 levels "38","39": 1 1 1 1 1 1 1 1 1
1 ...
 $ total_estimated_installs: Factor w/ 12 levels "1,000 - 5,000",..: 9 9 9 9
9 9 9 9 9 9 ...
 $ operating_system       : Factor w/ 12 levels "1.1 and up","1.5 and up",..
: 6 6 6 6 6 6 6 6 6 6 ...
 $ downloads              : int  1 6 3 95 6 0 0 85 85 1 ...
 $ day                    : int  22 22 22 22 23 22 22 24 24 24 ...
 $ ranks                  : num  0.191 0.191 0.191 0.191 0.188 ...
 $ Bin_AppSubCat          : Factor w/ 5 levels "Bin_AppSubCat_1",..: 3 3 3 3
3 3 3 3 3 3 ...
 $ Bin_RankCat            : Factor w/ 5 levels "Bin_RankCat_1",..: 3 3 3 3 3
3 3 3 3 3 ...
```

Converting logical variables to numeric (0 and 1)

```
> cols <- sapply(main_subset, is.logical)
> main_subset[,cols] <- lapply(main_subset[,cols], as.integer)
> head(main_subset)
  country rank_kind age_restrictions offers_in_app_purchases paid us_price
1      KR      free  Medium Maturity                       0    0        0
2      CA      free  Medium Maturity                       0    0        0
3      RU      free  Medium Maturity                       0    0        0
4      US      free  Medium Maturity                       0    0        0
5      AU      free  Medium Maturity                       0    0        0
6      CN      free  Medium Maturity                       0    0        0
  app_category total_estimated_installs operating_system downloads day     ra
nks
1           38          50,000 - 100,000       2.2 and up         1  22 0.1914
031
2           38          50,000 - 100,000       2.2 and up         6  22 0.1914
031
3           38          50,000 - 100,000       2.2 and up         3  22 0.1914
031
4           38          50,000 - 100,000       2.2 and up        95  22 0.1914
031
5           38          50,000 - 100,000       2.2 and up         6  23 0.1881
590
6           38          50,000 - 100,000       2.2 and up         0  22 0.1914
031
    Bin_AppSubCat   Bin_RankCat
1 Bin_AppSubCat_3 Bin_RankCat_3
2 Bin_AppSubCat_3 Bin_RankCat_3
3 Bin_AppSubCat_3 Bin_RankCat_3
4 Bin_AppSubCat_3 Bin_RankCat_3
5 Bin_AppSubCat_3 Bin_RankCat_3
6 Bin_AppSubCat_3 Bin_RankCat_3
```

# 5. Traditional method – Approach 1

## 5.1 Data Partition

```
> set.seed(1990)>
> trainPercentage = 0.70 # Percentage of rows in training data set>
> noTrainingRows = floor(trainPercentage * nrow(main_subset))>
> trainingIndex = sample(seq_len(nrow(main_subset)), size = noTrainingRows)>
> trainingData = main_subset[trainingIndex,]
> testingData = main_subset[-trainingIndex,]
```

## 5.2 User Defined Functions

Function to calculate mean absolute error - takes two arguments: actual value, predicted value

```
> funcMAE = function(actual,predicted){
+   mean(abs(actual-predicted))
+ }
```

Function to create a scatter plot between predicted and actual values, with a smoothing curve

```
> createPredVsAcSP = function(actualValues, predictedValues){
+
+   p = ggplot(aes(x = actual, y = predicted), data = data.frame(actual = act
ualValues,predicted = predictedValues))
+
+   p = p + geom_point(color = "steelblue",size = 2) + geom_smooth(color = "r
ed",method = "lm")
+
+   p
+ }
```

Function to create a residual plot

```
> createResidualPlot = function(actualValues, predictedValues){
+
+   res = actualValues - predictedValues
+
+   p = ggplot(aes(x = actual, y = residuals)
+                  ,data = data.frame(actual = actualValues
+                                   ,residuals = res))
+   p = p + geom_point(color = "steelblue", size = 2)
+
+   p = p + geom_hline(yintercept = 0, color = "red")
+
+   p
+
+ }
```

## 5.3 Model Training

Random Forest

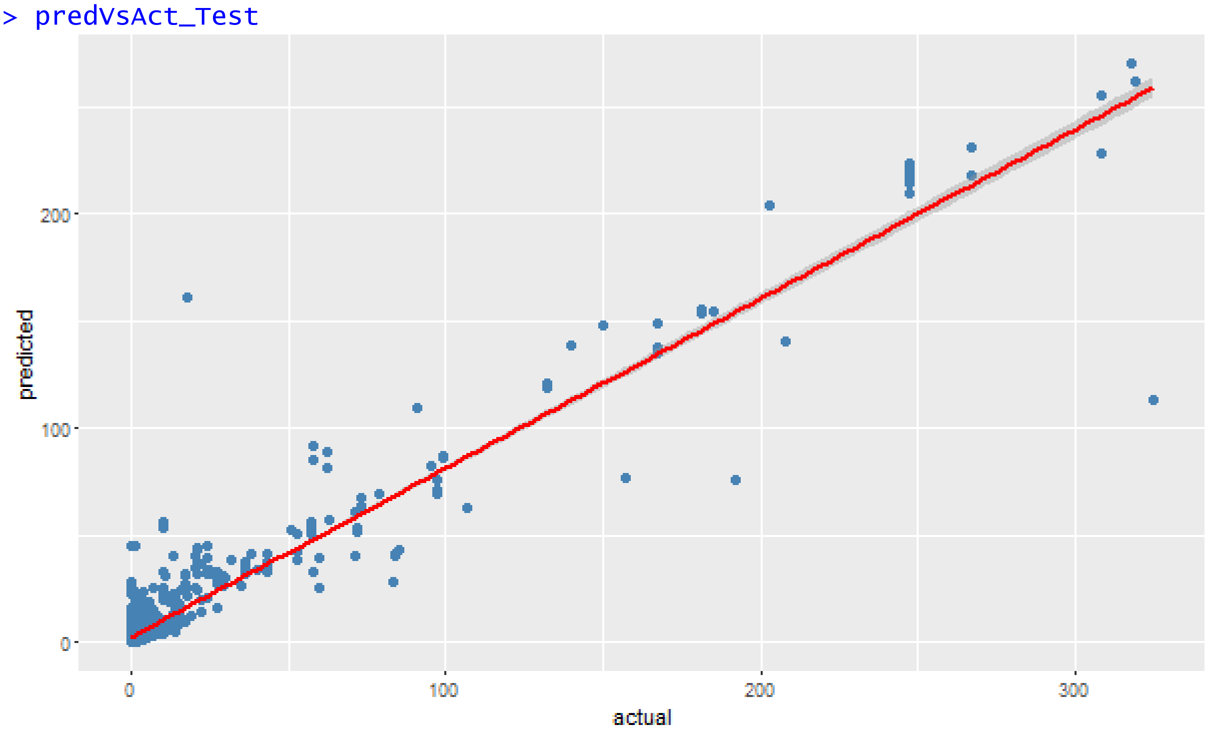
```
> randForModel <- randomForest(downloads ~ ., data = trainingData, ntree = 10
0)
> summary(randForModel)
                Length Class  Mode     
call               4   -none- call     
type               1   -none- character
predicted       2535   -none- numeric  
mse              100   -none- numeric  
rsq              100   -none- numeric  
oob.times       2535   -none- numeric  
importance        13   -none- numeric  
importanceSD       0   -none- NULL     
localImportance    0   -none- NULL     
proximity          0   -none- NULL     
ntree              1   -none- numeric  
mtry               1   -none- numeric  
forest            11   -none- list     
coefs              0   -none- NULL     
y               2535   -none- numeric  
test               0   -none- NULL     
inbag              0   -none- NULL     
terms              3   terms  call     
> r_Train <- cor(trainingData$downloads, predict(randForModel, trainingData))
> r_Train
[1] 0.9806383
> r2_Train <- rSquared(trainingData$downloads, (trainingData$downloads - pred
ict(randForModel, trainingData)))
> r2_Train
          [,1]
[1,] 0.9448768
> mse_Train <- mean((trainingData$downloads - predict(randForModel, trainingD
ata))^2)
> mse_Train
[1] 71.16239
> rmse_Train <- sqrt(mse_Train)
> rmse_Train
[1] 8.43578
> mae_Train = funcMAE(trainingData$downloads,predict(randForModel,trainingDat
a))
> mae_Train
[1] 2.670076
```

## 5.4 Model Testing

```
> testPred = predict(randForModel, testingData)
> r_Test = cor(testingData$downloads,testPred)
> r_Test
[1] 0.9528586
> r2_Test = rSquared(testingData$downloads, (testingData$downloads-testPred))
> mse_Test = mean((testingData$downloads-testPred)^2)
> r2_Test
         [,1]
[1,] 0.892964
> mse_Test
[1] 155.2705
> rmse_Test = sqrt(mse_Test)
> rmse_Test
[1] 12.46076
> mae_Test = funcMAE(testingData$downloads,testPred)
> mae_Test
[1] 3.923012
```

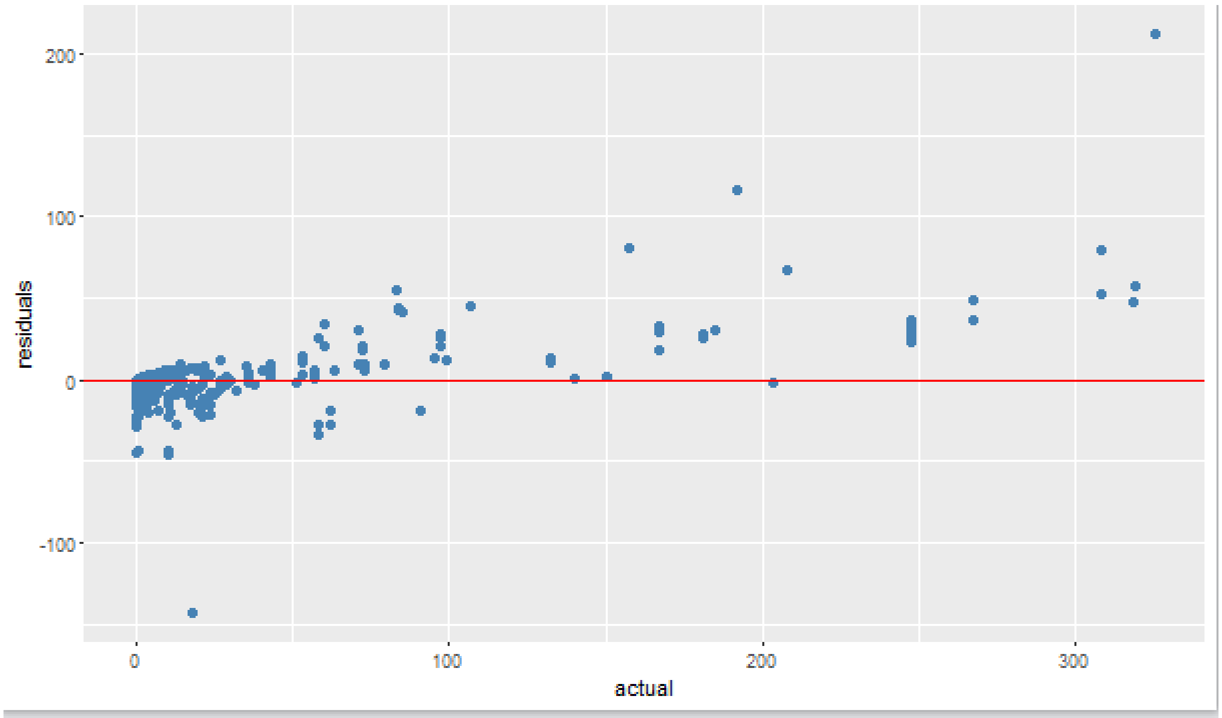
Scatterplot

```
> predVsAct_Test = createPredVsAcSP(actualValues = testingData$downloads
+                                   ,predictedValues = testPred)
> predVsAct_Test
```



Residual Plot

```
> residualPlotTest = createResidualPlot(actualValues = testingData$downloads,
predictedValues = testPred)
> residualPlotTest
```




## 5.5 Model Execution

```
> mainTest <- read.csv("test (data sci challenge).csv")
> testOriginal = mainTest
```

Before executing the model, it is helpful to check if there are any new levels in test data set and make the decision accordingly.

```
> for(colName in names(mainTest)){
+   if(is.factor(mainTest[[colName]])){
+     levels(mainTest[[colName]]) = levels(main_subset[[colName]])
+   }
+ }
```

Applying model:

```
> mainPred <- predict(randForModel, mainTest)
```

Saving the predicted values in new column

```
> PredictedTestingData <- testOriginal
> PredictedTestingData$PredictedDownloads <- round(mainPred)
```

## 6. Best Method – Approach 2

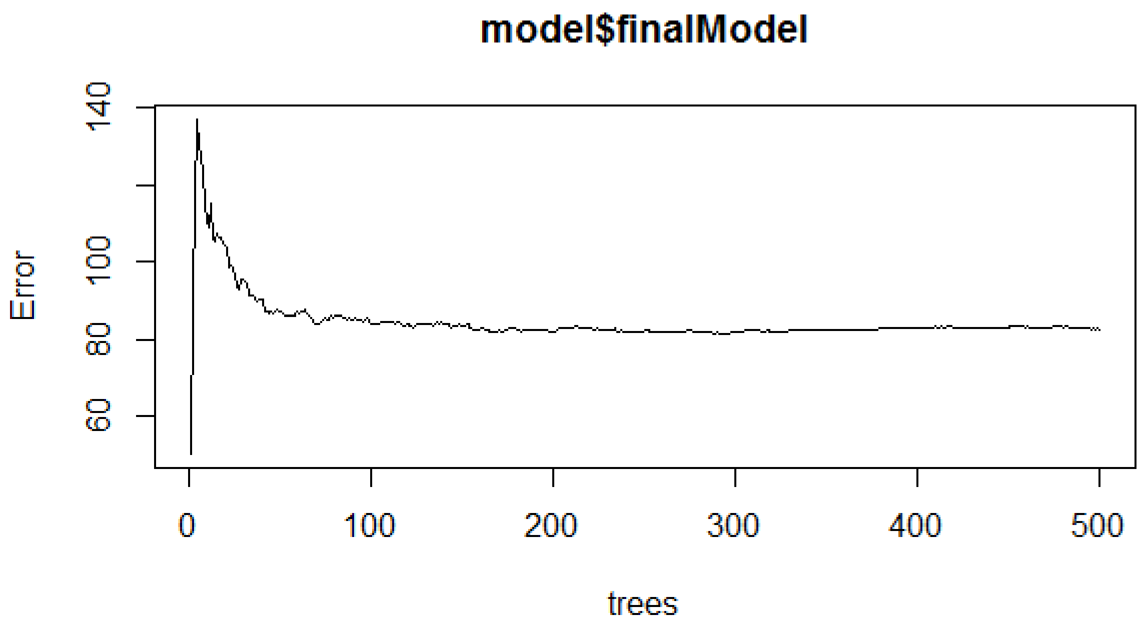
Predictive model using cross-validation:

### 6.1 Model Training

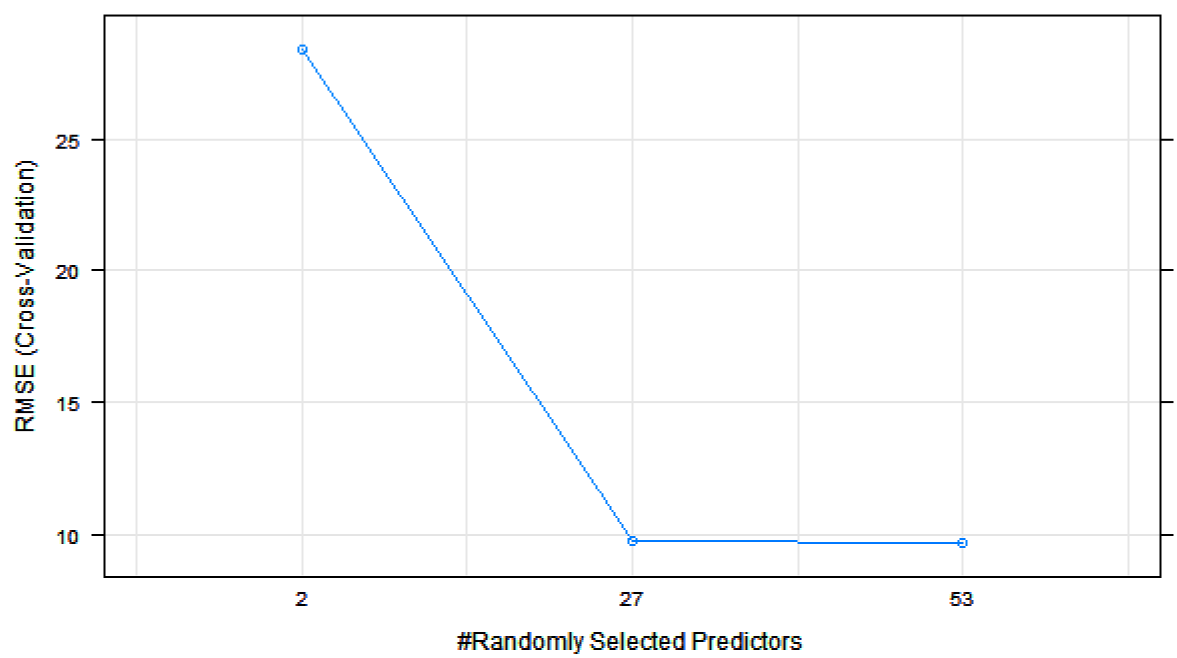
```
> train_control <- trainControl(method = 'cv', number = 4)
> model <- train(downloads~., data = main_subset, trControl = train_control,
method = 'rf', metric = 'RMSE', importance = TRUE)
> model
Random Forest

3622 samples
  13 predictor

No pre-processing
Resampling: Cross-Validated (4 fold)
Summary of sample sizes: 2716, 2718, 2716, 2716
Resampling results across tuning parameters:

  mtry  RMSE       Rsquared   RMSE SD   Rsquared SD
   2    28.396052  0.5093472  3.640578  0.04270059
  27     9.757302  0.9284604  3.445572  0.03902746
  53     9.667543  0.9258877  4.177185  0.05078209

RMSE was used to select the optimal model using  the smallest value.
The final value used for the model was mtry = 53.

> summary(model)
                Length Class      Mode
call               4   -none-     call
type               1   -none-     character
predicted       3622   -none-     numeric
mse              500   -none-     numeric
rsq              500   -none-     numeric
oob.times       3622   -none-     numeric
importance        53   -none-     numeric
importanceSD       0   -none-     NULL
localImportance    0   -none-     NULL
proximity          0   -none-     NULL
ntree              1   -none-     numeric
mtry               1   -none-     numeric
forest            11   -none-     list
coefs              0   -none-     NULL
y               3622   -none-     numeric
test               0   -none-     NULL
inbag              0   -none-     NULL
xNames            53   -none-     character
problemType        1   -none-     character
tuneValue          1   data.frame list
obsLevels          1   -none-     logical
```

```
> varImp(model)
rf variable importance

  only 20 most important variables shown (out of 53)

                                               Overall
countryUS                                       100.00
countryRU                                        66.21
countryGB                                        61.99
countryDE                                        51.39
countryKR                                        49.07
total_estimated_installs100,000 - 500,000        47.13
total_estimated_installs1,000,000 - 5,000,000    46.15
countryFR                                        45.40
age_restrictionsHigh Maturity                    39.60
countryCN                                        28.87
countryES                                        25.40
total_estimated_installs50,000 - 100,000         25.07
operating_system2.2 and up                       23.44
operating_system2.1 and up                       22.90
total_estimated_installs5,000,000 - 10,000,000   22.86
Bin_AppSubCatBin_AppSubCat_5                     21.88
total_estimated_installs500,000 - 1,000,000      20.90
countryCA                                        20.38
operating_system2.3 and up                       17.43
operating_system4.0.3 and up                     17.12
```

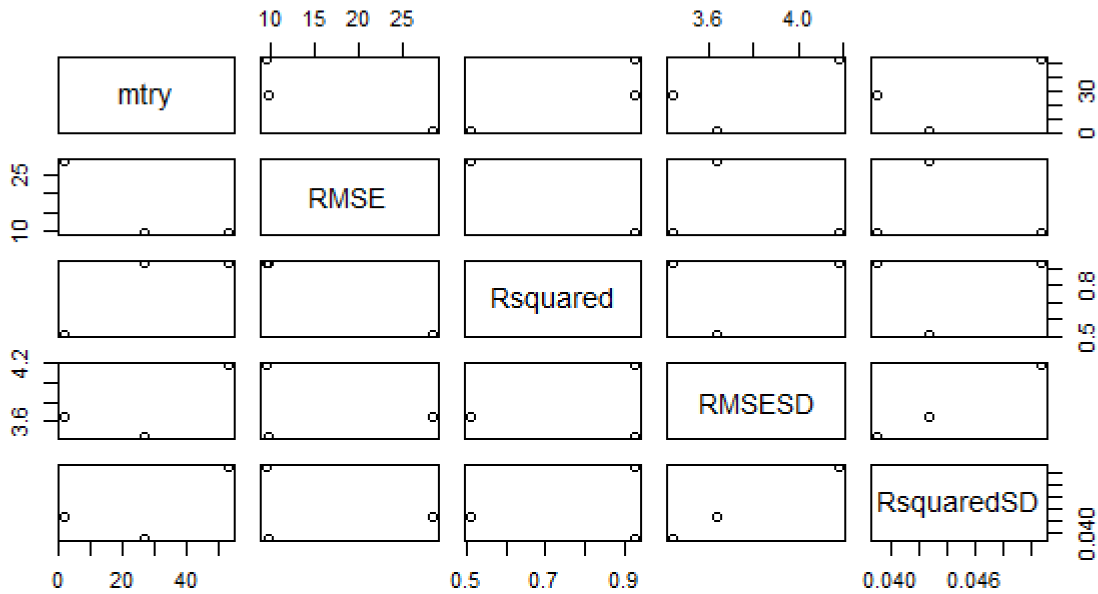
```
> plot(model$finalModel)
```




```
> plot(model, plotType = 'line')
```



```
> plot(model$results)
```




```
> (model$bestTune)
  mtry
3   53
```

### 6.2 Model Execution

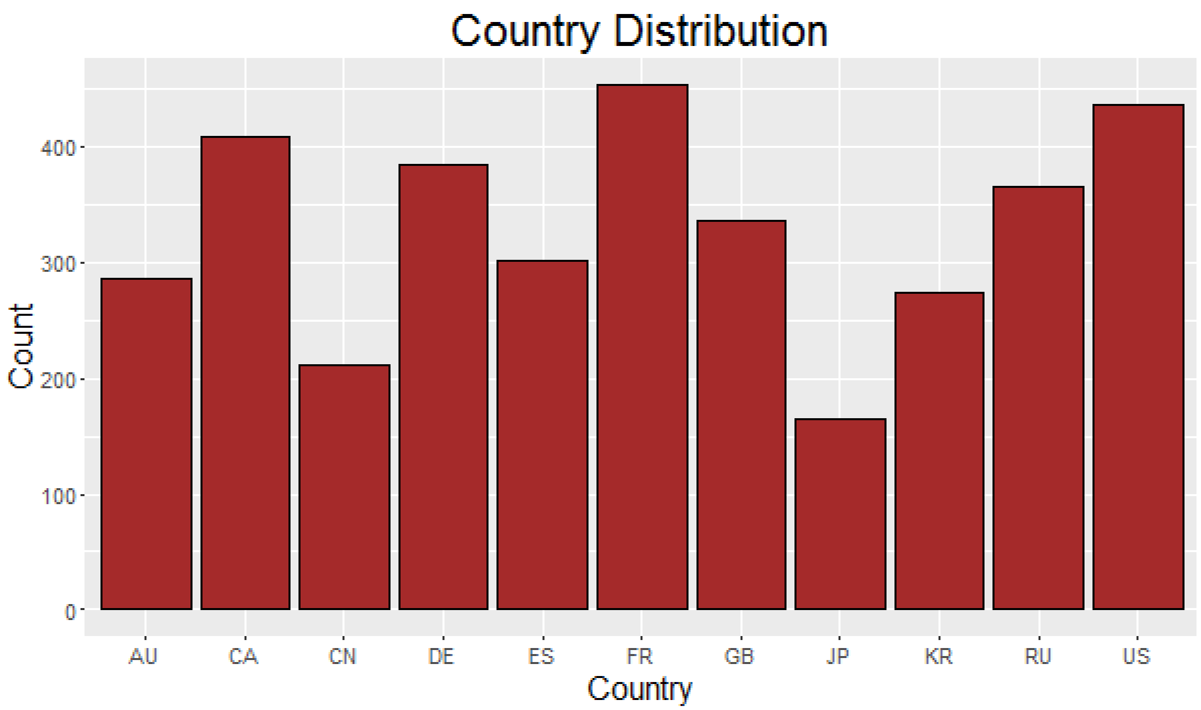
```
> pred <- predict(model, mainTest)
> mainTest$downloads_pred <- round(pred)
```

# 7. Appendix

a. Distribution Plots (for each variable)

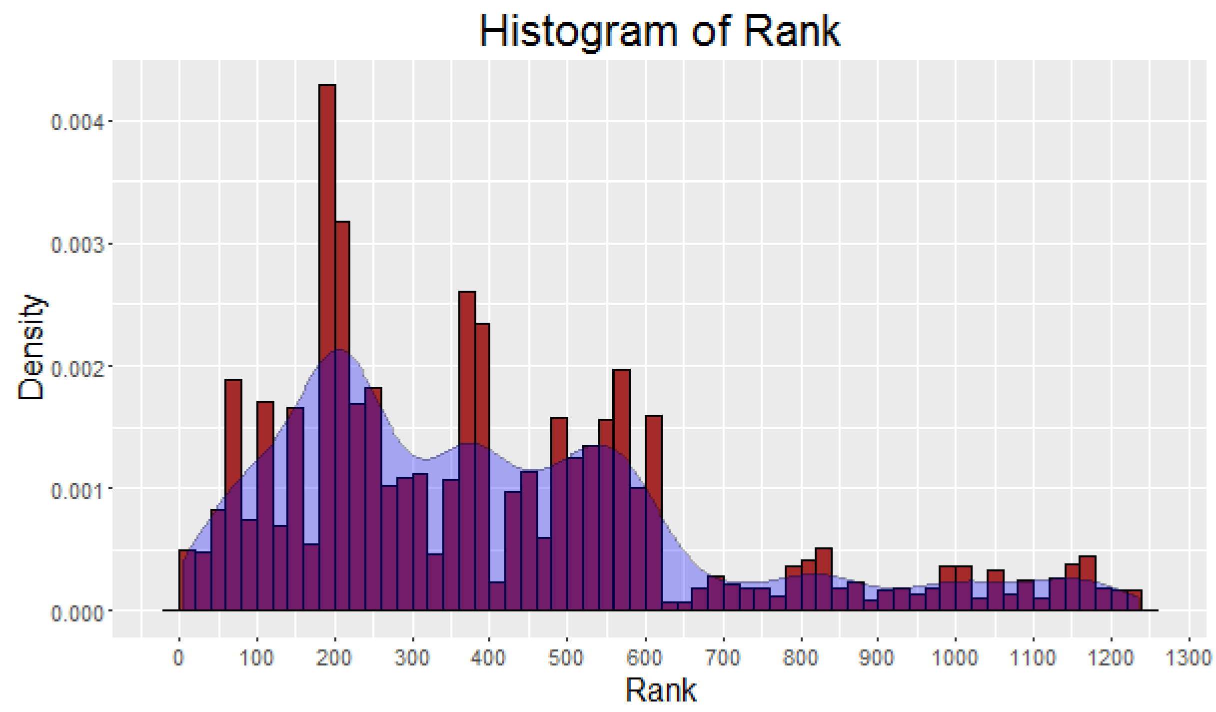
**Country**

```
> bpcountry = ggplot(data = main_subset, aes(x = country))
> bpcountry = bpcountry + geom_bar(fill = "brown", color = "black")
> bpcountry = bpcountry + labs(title = "Country Distribution", x= "Country",
y = "Count")
> bpcountry = bpcountry + theme(axis.title = element_text(size = 16), axis.te
xt = element_text(size = 11), title = element_text(size = 18))
> bpcountry
```




**Rank**

```
> histRank <- ggplot(main_subset, aes( x = rank))
> histRank <- histRank + geom_histogram(binwidth = 20
+                                       ,aes (y = ..density..)
+                                       ,fill = I("Brown")
+                                       ,colour = I("Black"))
> histRank <- histRank + geom_density(fill = "blue", alpha = 0.3)
> histRank <- histRank + labs(title = "Histogram of Rank", x = "Rank", y = "D
ensity")
> histRank <- histRank + scale_x_continuous(breaks = 100*(0:15))
> histRank = histRank + theme(axis.title = element_text(size = 16), axis.text
= element_text(size = 11), title = element_text(size = 18))
> histRank
```




**Rank_category**

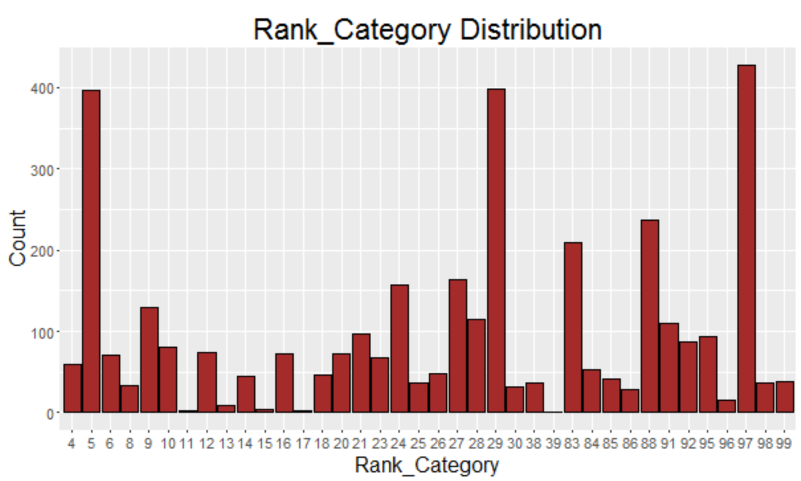
bpRankCat = ggplot(data = main_subset, aes(x = rank_category))

bpRankCat = bpRankCat + geom_bar(fill = "brown", color = "black")

bpRankCat = bpRankCat + labs(title = "Rank_Category Distribution", x= "Rank_Category", y = "Count")

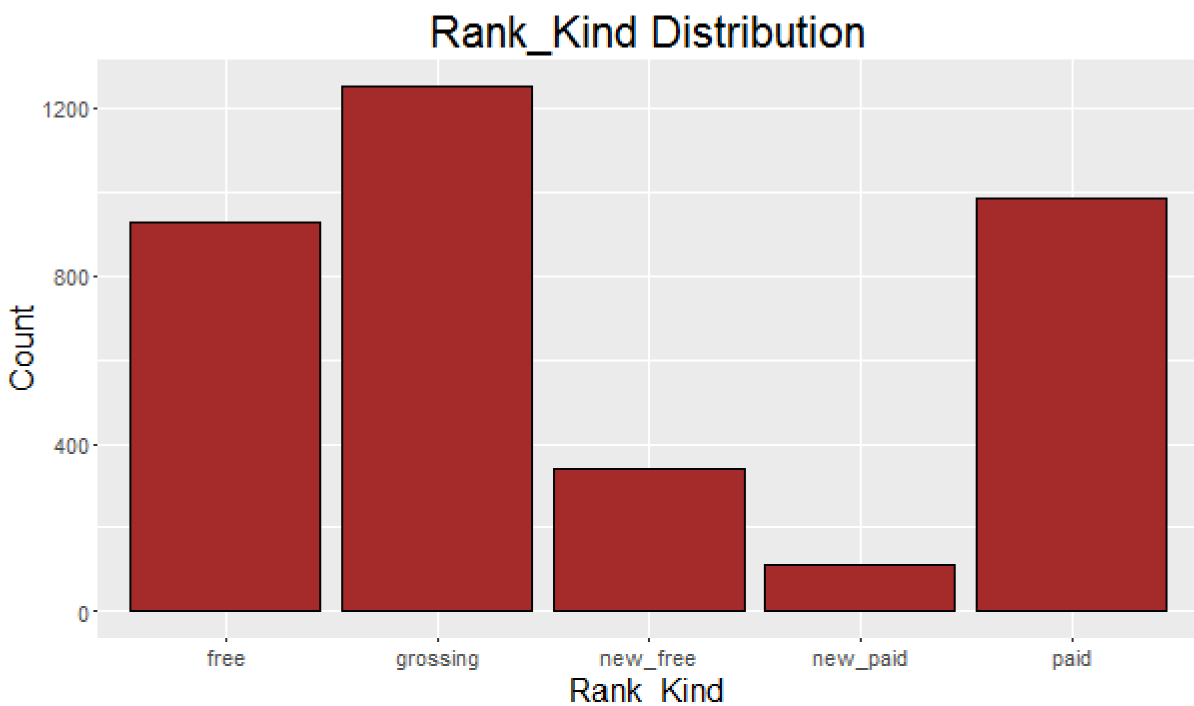
bpRankCat = bpRankCat + theme(axis.title = element_text(size = 16), axis.text = element_text(size = 11), title = element_text(size = 18))

bpRankCat



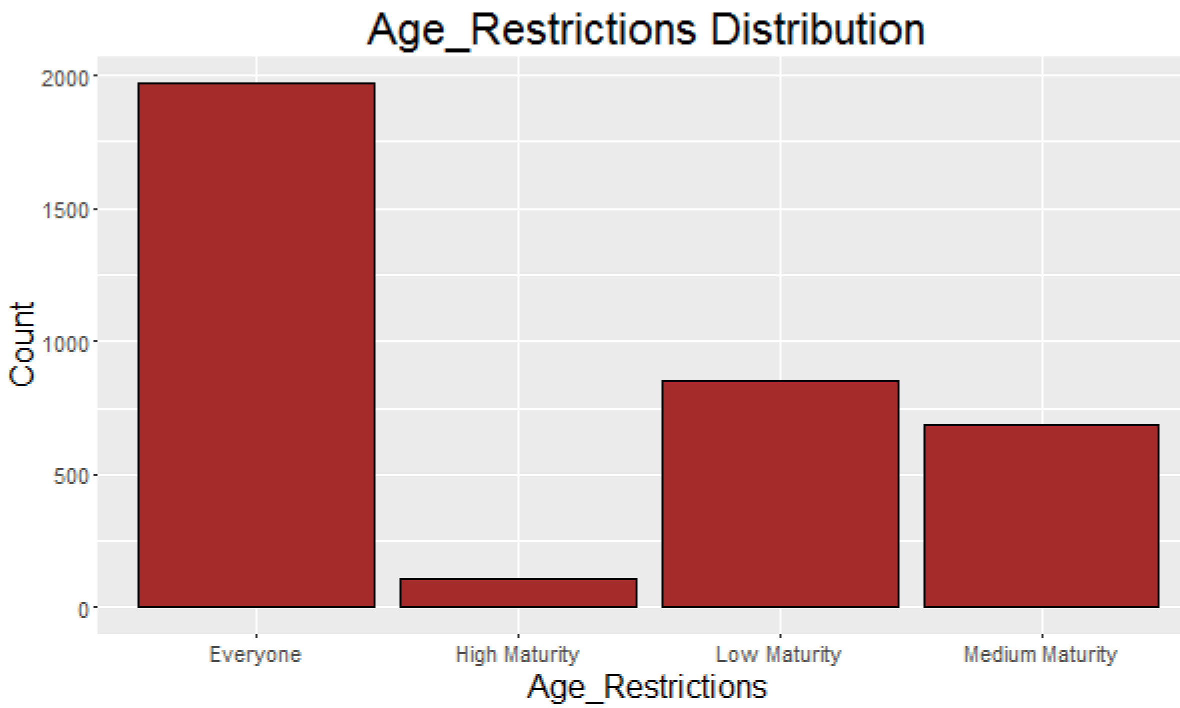
**Rank_Kind**

```
> bpRankkind = ggplot(data = main_subset, aes(x = rank_kind))
> bpRankkind = bpRankkind + geom_bar(fill = "brown", color = "black")
> bpRankkind = bpRankkind + labs(title = "Rank_Kind Distribution", x= "Rank_K
ind", y = "Count")
> bpRankkind = bpRankkind + theme(axis.title = element_text(size = 16), axis.
text = element_text(size = 11), title = element_text(size = 18))
> bpRankkind
```



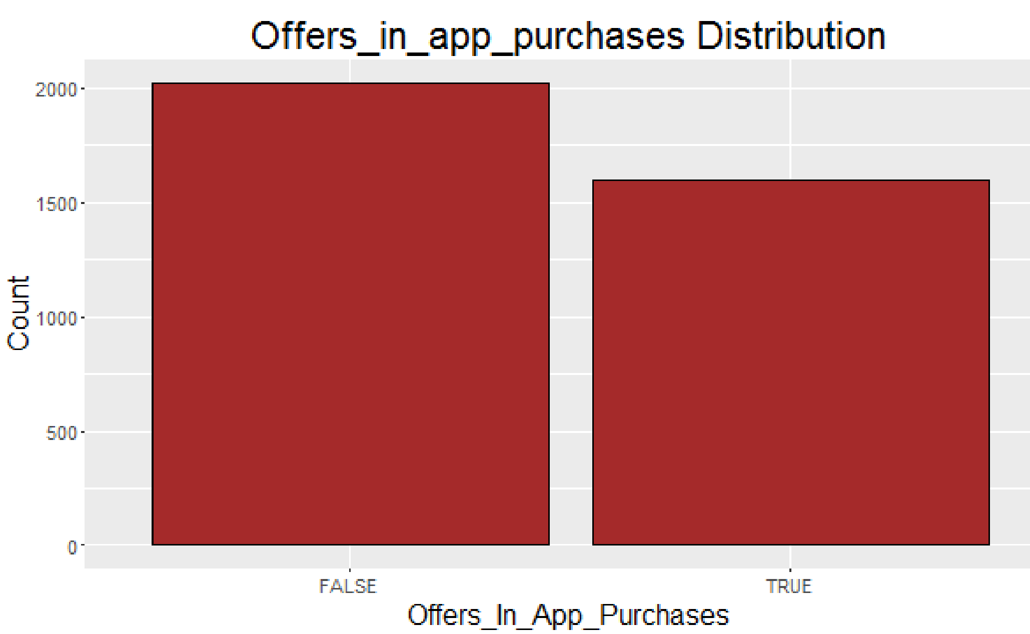

**Age_Restrictions**

```
> bpAgeRest = ggplot(data = main_subset, aes(x = age_restrictions))
> bpAgeRest = bpAgeRest + geom_bar(fill = "brown", color = "black")
> bpAgeRest = bpAgeRest + labs(title = "Age_Restrictions Distribution", x= "A
ge_Restrictions", y = "Count")
> bpAgeRest = bpAgeRest + theme(axis.title = element_text(size = 16), axis.te
xt = element_text(size = 11), title = element_text(size = 18))
> bpAgeRest
```



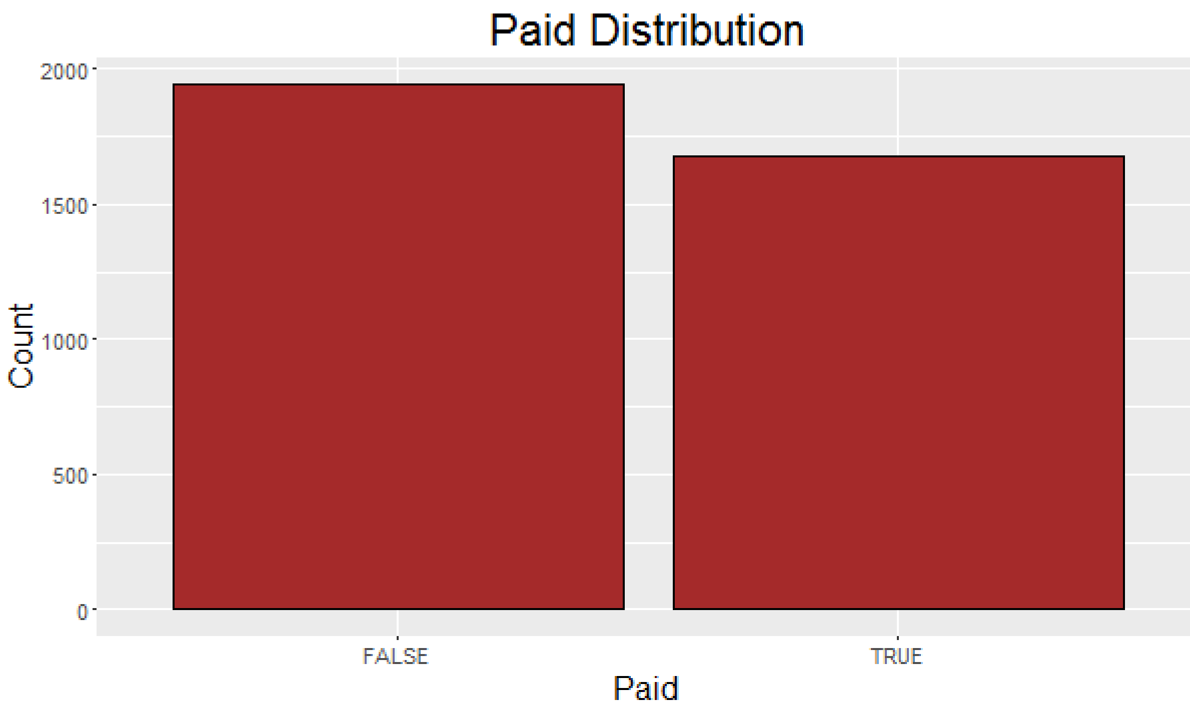

**Offers_in_app_purchases**

```
> bpOffer = ggplot(data = main_subset, aes(x = offers_in_app_purchases))
> bpOffer = bpOffer + geom_bar(fill = "brown", color = "black")
> bpOffer = bpOffer + labs(title = "Offers_in_app_purchases Distribution", x=
"Offers_In_App_Purchases", y = "Count")
> bpOffer = bpOffer + theme(axis.title = element_text(size = 16), axis.text =
element_text(size = 11), title = element_text(size = 18))
> bpOffer
```



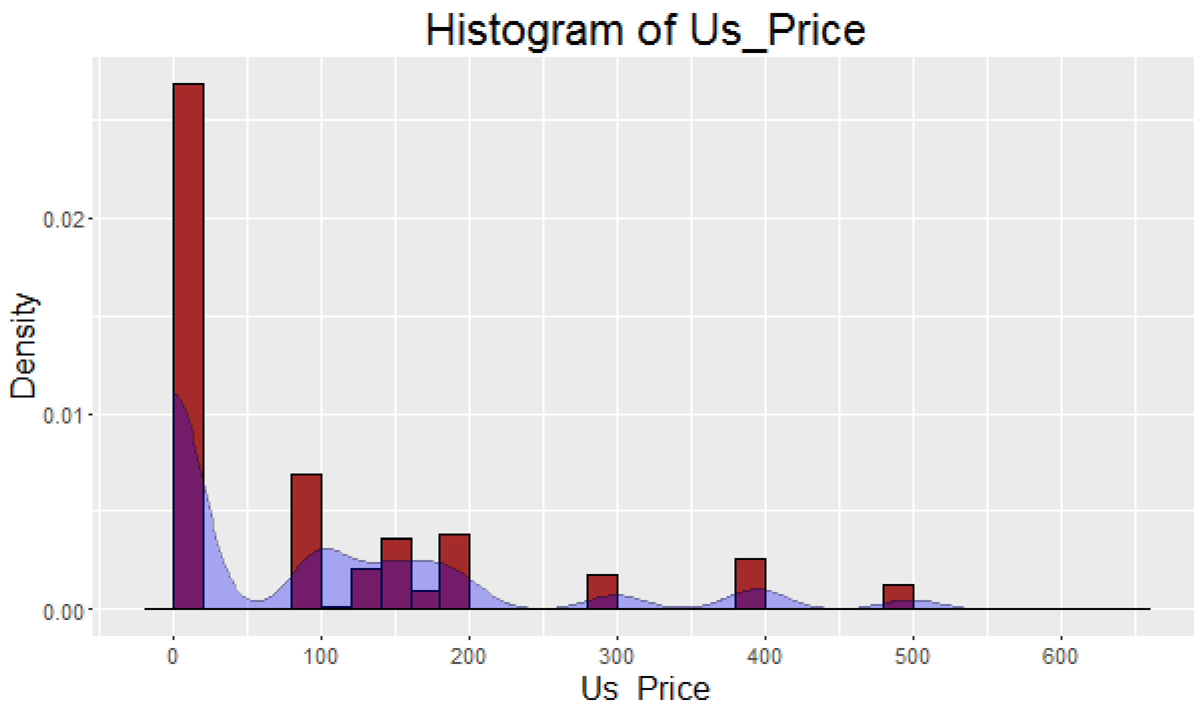
**Paid**

```
> bpPaid = ggplot(data = main_subset, aes(x = paid))
> bpPaid = bpPaid + geom_bar(fill = "brown", color = "black")
> bpPaid = bpPaid + labs(title = "Paid Distribution", x= "Paid", y = "Count")
> bpPaid = bpPaid + theme(axis.title = element_text(size = 16), axis.text = e
lement_text(size = 11), title = element_text(size = 18))
> bpPaid
```



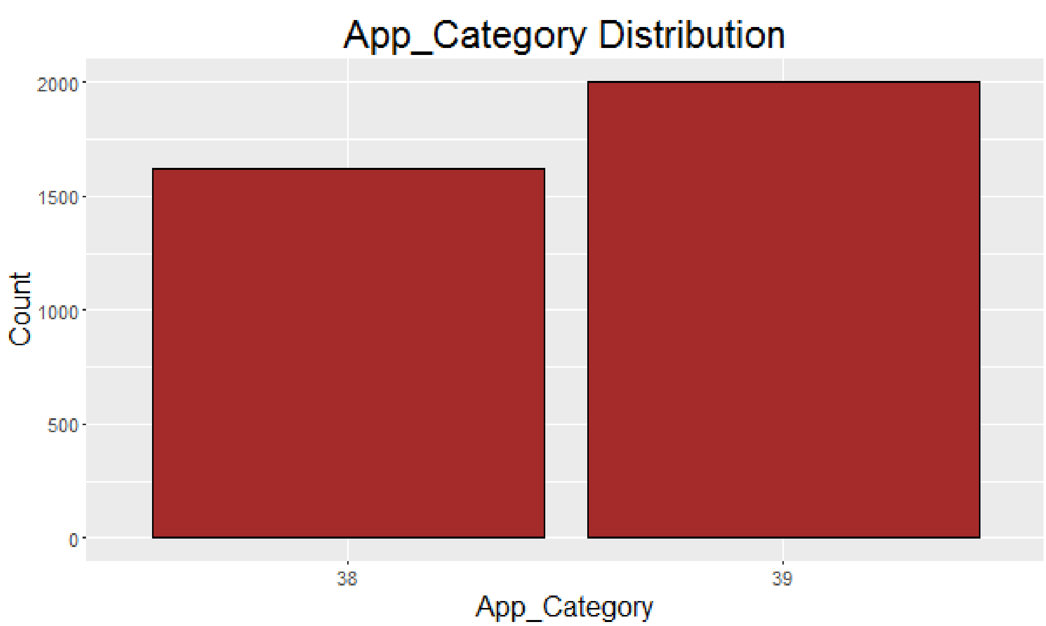

**Us_Price**

```
> histPrice <- ggplot(main_subset, aes( x = us_price))
> histPrice <- histPrice + geom_histogram(binwidth = 20
+                                         ,aes (y = ..density..)
+                                         ,fill = I("Brown")
+                                         ,colour = I("Black"))
> histPrice <- histPrice + geom_density(fill = "blue", alpha = 0.3)
> histPrice <- histPrice + labs(title = "Histogram of Us_Price", x = "Us_Pric
e", y = "Density")
> histPrice <- histPrice + scale_x_continuous(breaks = 100*(0:15))
> histPrice <- histPrice + theme(axis.title = element_text(size = 16), axis.t
ext = element_text(size = 11), title = element_text(size = 18))
> histPrice
```



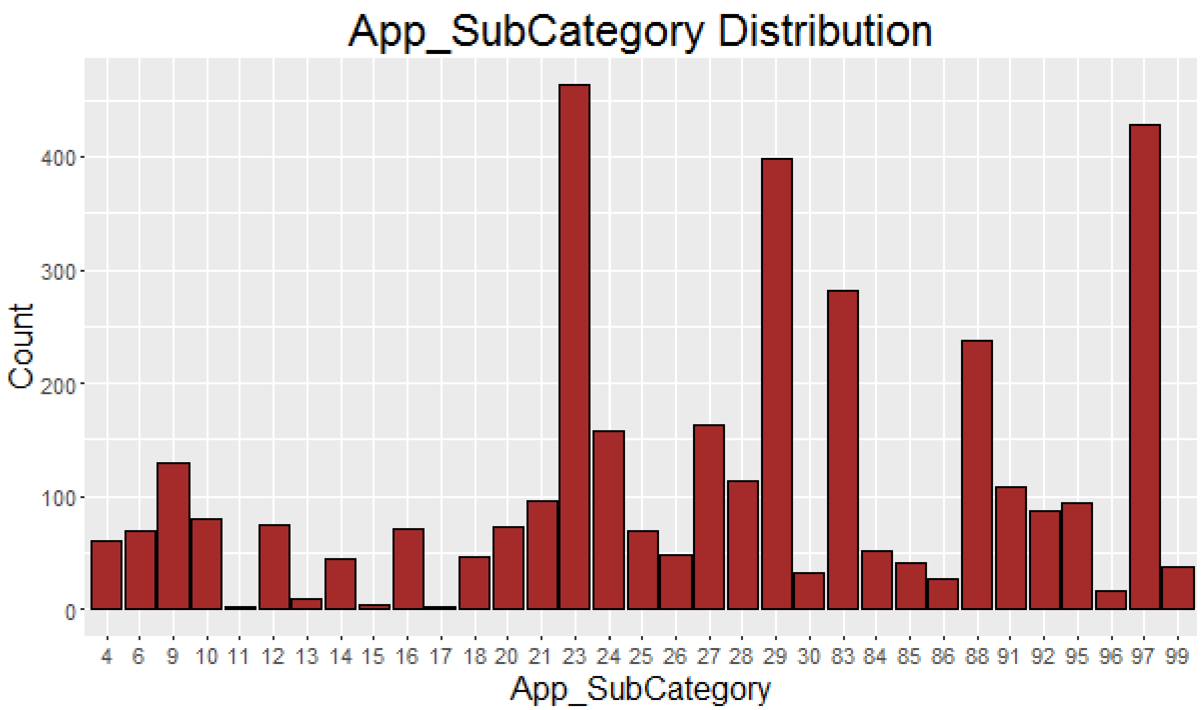

**App_Category**

```
> bpAppCat = ggplot(data = main_subset, aes(x = app_category))
> bpAppCat = bpAppCat + geom_bar(fill = "brown", color = "black")
> bpAppCat = bpAppCat + labs(title = "App_Category Distribution", x= "App_Cat
egory", y = "Count")
> bpAppCat = bpAppCat + theme(axis.title = element_text(size = 16), axis.text
= element_text(size = 11), title = element_text(size = 18))
> bpAppCat
```



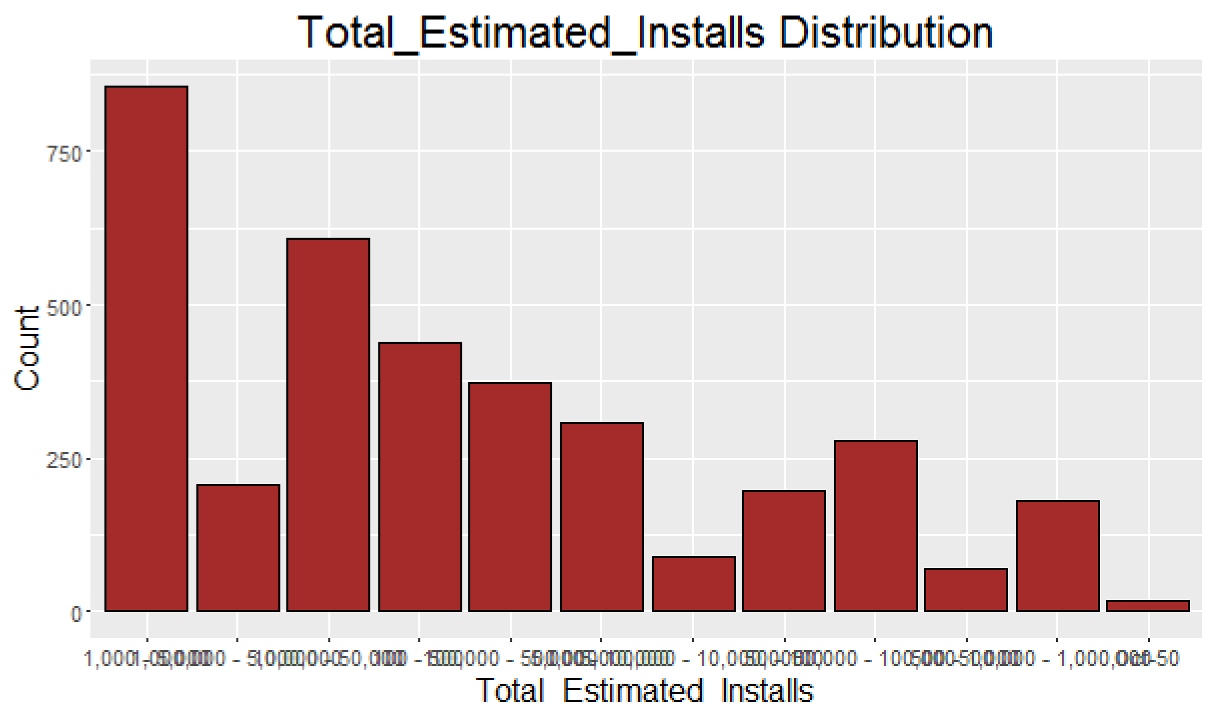
**App_SubCategory**

```
> bpAppSubCat = ggplot(data = main_subset, aes(x = app_subcategory))
> bpAppSubCat = bpAppSubCat + geom_bar(fill = "brown", color = "black")
> bpAppSubCat = bpAppSubCat + labs(title = "App_SubCategory Distribution", x=
"App_SubCategory", y = "Count")
> bpAppSubCat = bpAppSubCat + theme(axis.title = element_text(size = 16), axi
s.text = element_text(size = 11), title = element_text(size = 18))
> bpAppSubCat
```



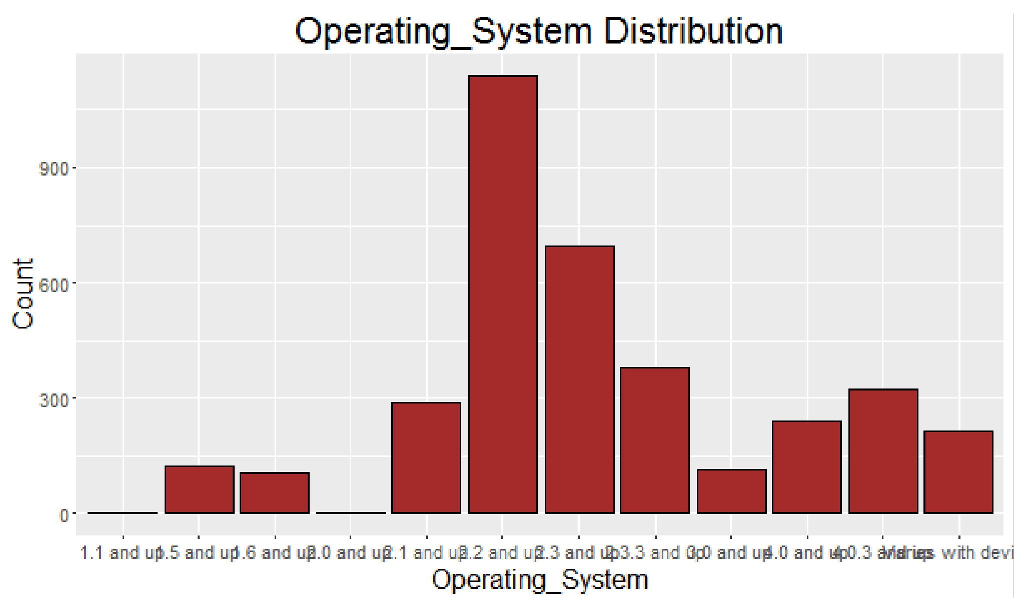

**Total_estimated_installs**

```
> bpTotIns = ggplot(data = main_subset, aes(x = total_estimated_installs))
> bpTotIns = bpTotIns + geom_bar(fill = "brown", color = "black")
> bpTotIns = bpTotIns + labs(title = "Total_Estimated_Installs Distribution",
x= "Total_Estimated_Installs", y = "Count")
> bpTotIns = bpTotIns + theme(axis.title = element_text(size = 16), axis.text
= element_text(size = 11), title = element_text(size = 18))
> bpTotIns
```



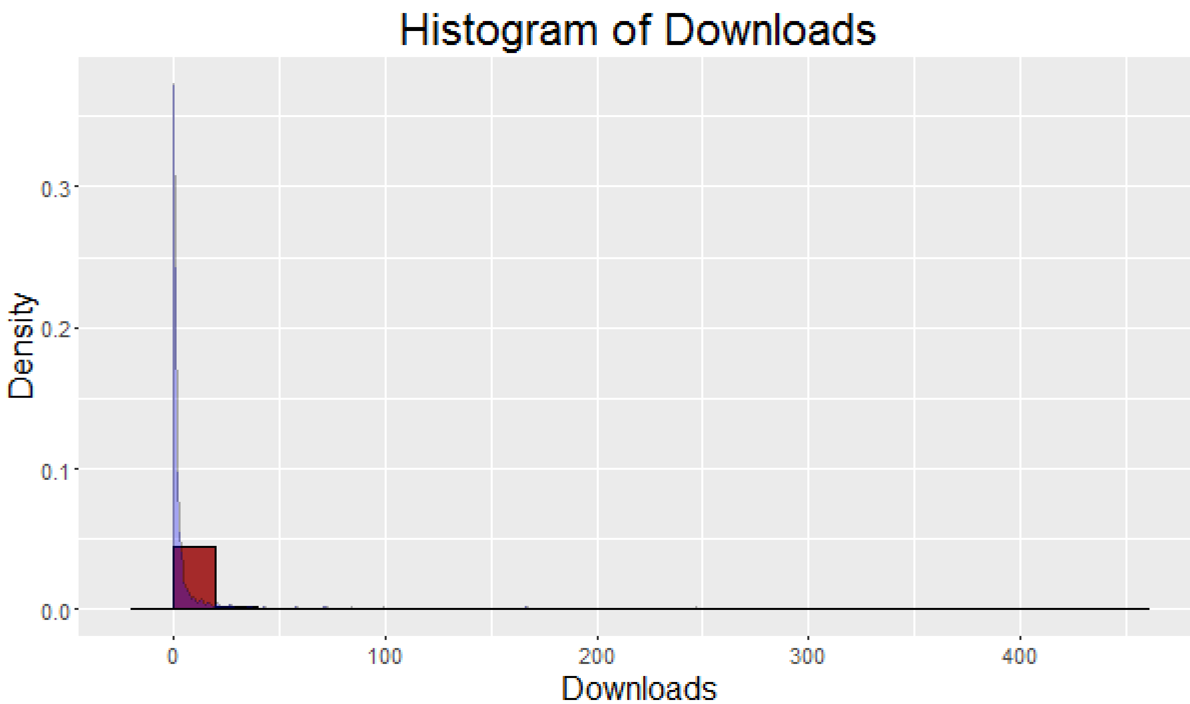

**Operating_System**

```
> bpOs = ggplot(data = main_subset, aes(x = operating_system))
> bpOs = bpOs + geom_bar(fill = "brown", color = "black")
> bpOs = bpOs + labs(title = "Operating_System Distribution", x= "Operating_S
ystem", y = "Count")
> bpOs = bpOs + theme(axis.title = element_text(size = 16), axis.text = eleme
nt_text(size = 11), title = element_text(size = 18))
> bpOs
```

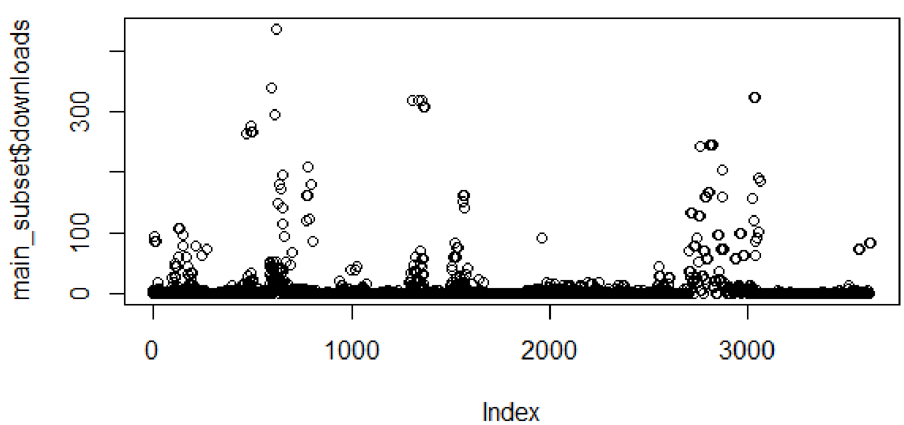


**Downloads**

```
> histDown <- ggplot(main_subset, aes( x = downloads))
> histDown <- histDown + geom_histogram(binwidth = 20
+                                         ,aes (y = ..density..)
+                                         ,fill = I("Brown")
+                                         ,colour = I("Black"))
> histDown <- histDown + geom_density(fill = "blue", alpha = 0.3)
> histDown <- histDown + labs(title = "Histogram of Downloads", x = "Download
s", y = "Density")
> histDown <- histDown + scale_x_continuous(breaks = 100*(0:15))
> histDown <- histDown + theme(axis.title = element_text(size = 16), axis.tex
t = element_text(size = 11), title = element_text(size = 18))
> histDown
```



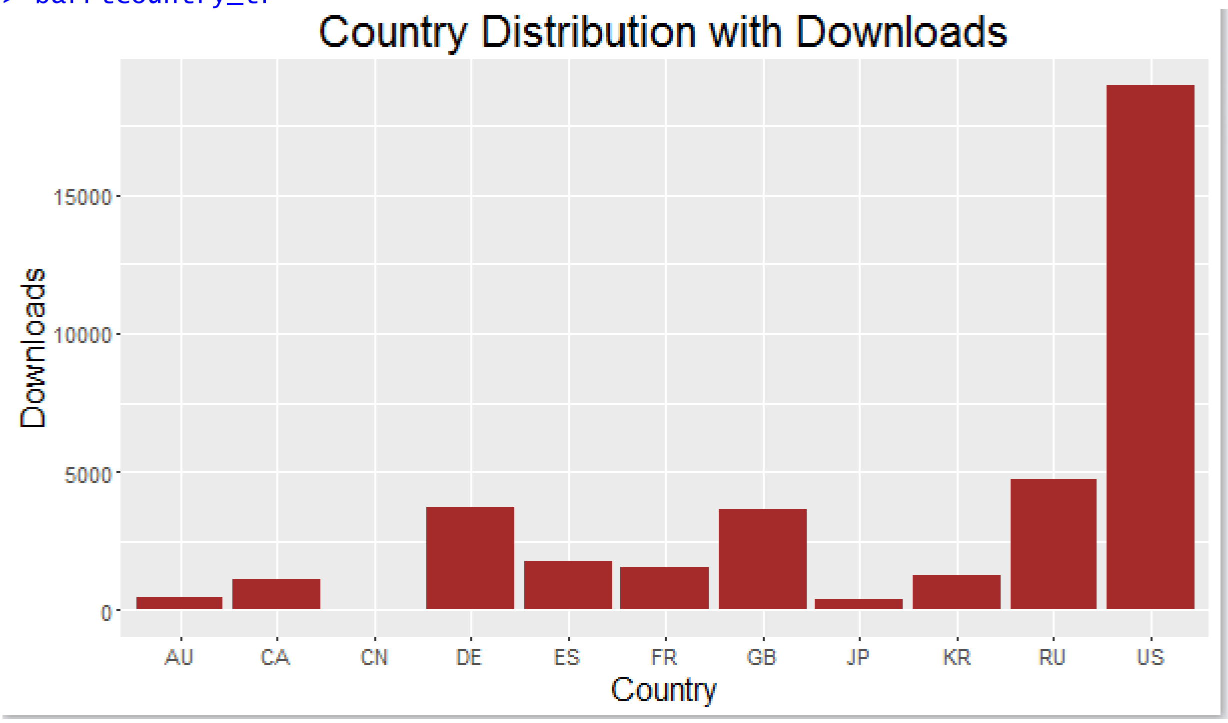

```
> plot(main_subset$downloads)
```



b. Plot to visualize relation between each variable with downloads

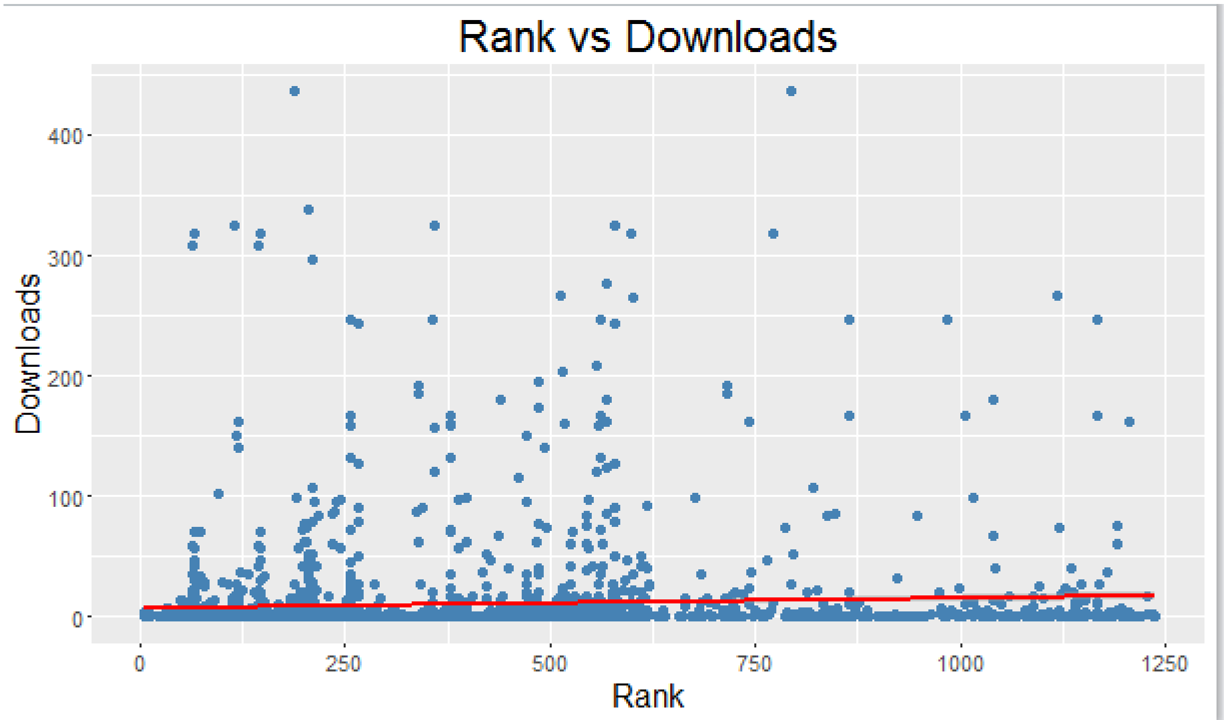
**Country vs Downloads**

```
> barPtcountry_tr = ggplot(data = main_subset, aes(x = country, y = downloads
))
> barPtcountry_tr = barPtcountry_tr + geom_bar(stat = "identity", fill = "bro
wn")
> barPtcountry_tr = barPtcountry_tr + labs(title = "Country Distribution with
Downloads", x= "Country", y = "Downloads")
> barPtcountry_tr = barPtcountry_tr + theme(axis.title = element_text(size =
16), axis.text = element_text(size = 11), title = element_text(size = 18))
> barPtcountry_tr
```



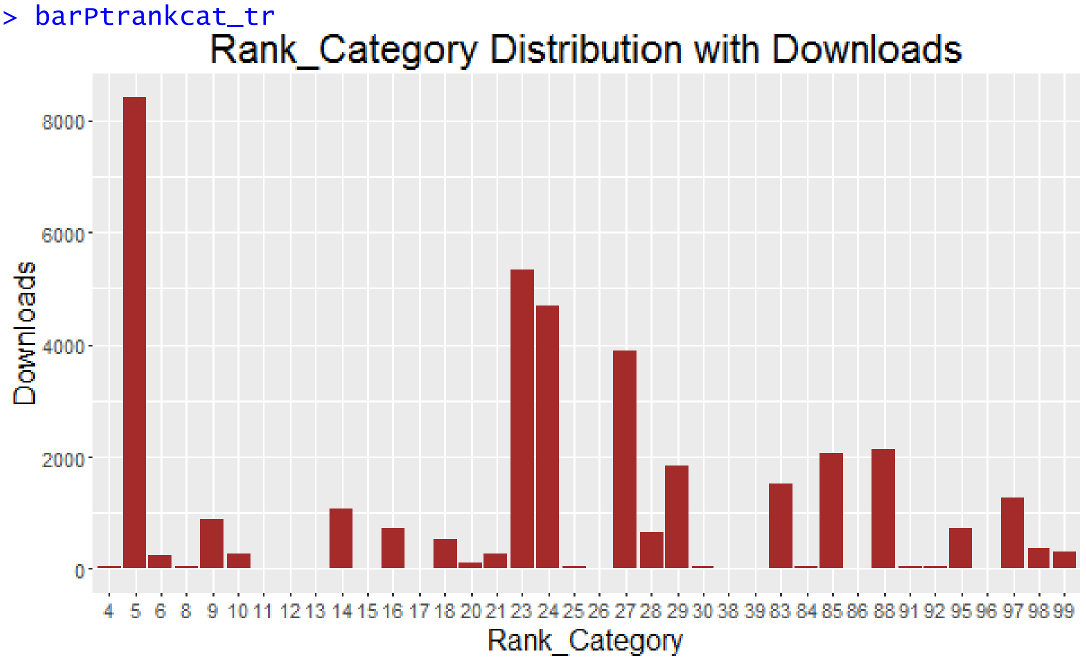

**Rank vs Downloads**

```
> spRankDown <- ggplot(main_subset, aes(x = rank, y = downloads))
> spRankDown <- spLogRankDown + geom_point(color = "steelblue", size = 2) + g
eom_smooth(method = "lm", color = "red")
> spRankDown <- spLogRankDown + labs(title = "Rank vs Downloads", x = "Rank",
y = "Downloads")
> spRankDown = spLogRankDown + theme(axis.title = element_text(size = 16), ax
is.text = element_text(size = 11), title = element_text(size = 18))
> spRankDown
```



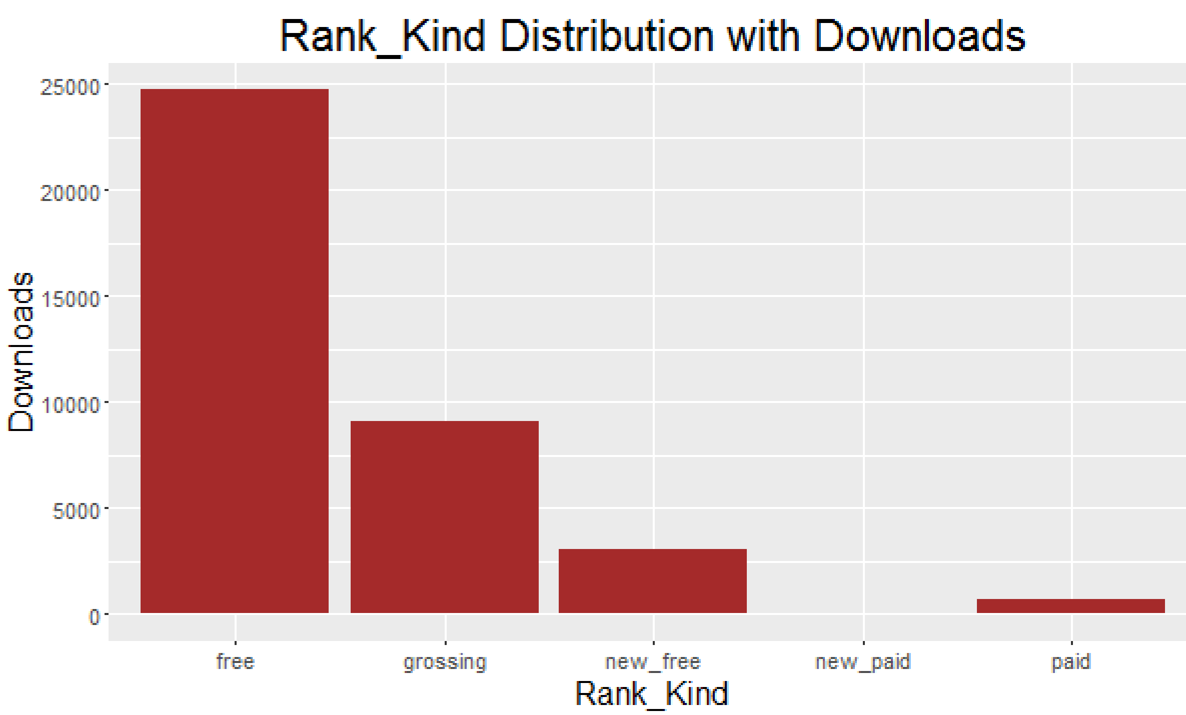

**Rank_Catgeory vs Downloads**

```
> barPtrankcat_tr = ggplot(data = main_subset, aes(x = rank_category, y = dow
nloads))
> barPtrankcat_tr = barPtrankcat_tr + geom_bar(stat = "identity", fill = "bro
wn")
> barPtrankcat_tr = barPtrankcat_tr + labs(title = "Rank_Category Distributio
n with Downloads", x= "Rank_Category", y = "Downloads")
> barPtrankcat_tr = barPtrankcat_tr + theme(axis.title = element_text(size =
16), axis.text = element_text(size = 11), title = element_text(size = 18))
> barPtrankcat_tr
```



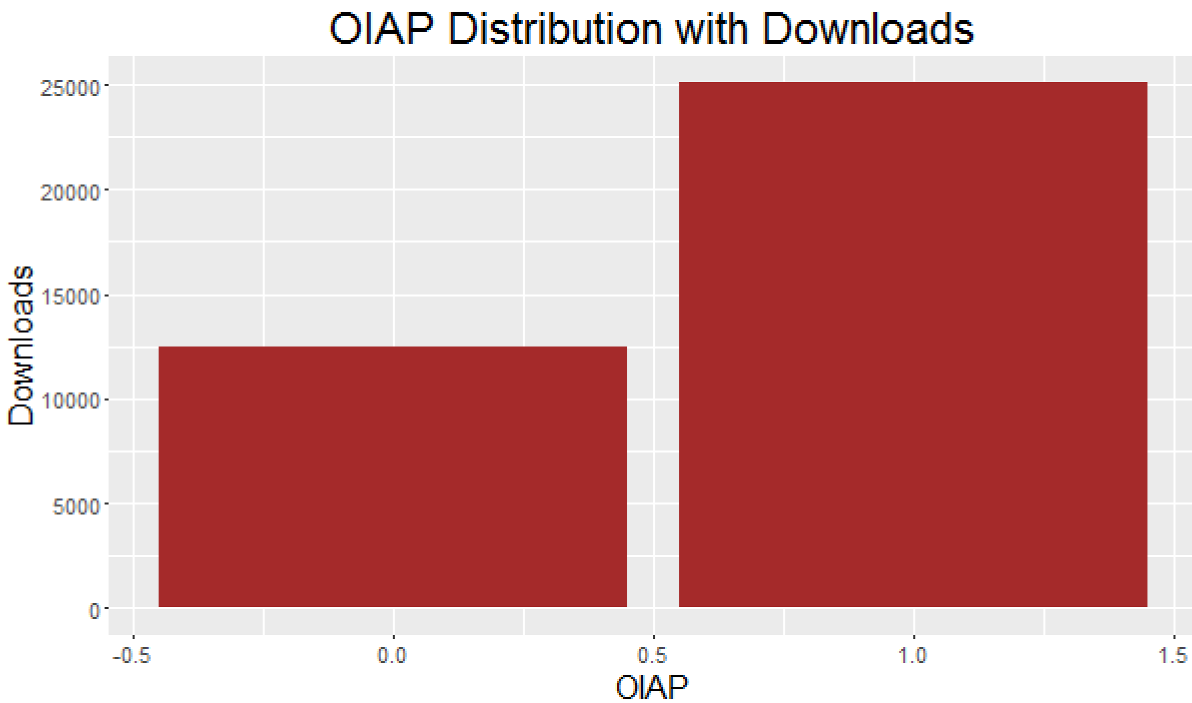
**Rank_Kind vs Downloads**

```
> barPtrankind_tr = ggplot(data = main_subset, aes(x = rank_kind, y = downloa
ds))
> barPtrankind_tr = barPtrankind_tr + geom_bar(stat = "identity", fill = "bro
wn")
> barPtrankind_tr = barPtrankind_tr + labs(title = "Rank_Kind Distribution wi
th Downloads", x= "Rank_Kind", y = "Downloads")
> barPtrankind_tr = barPtrankind_tr + theme(axis.title = element_text(size =
16), axis.text = element_text(size = 11), title = element_text(size = 18))
> barPtrankind_tr
```



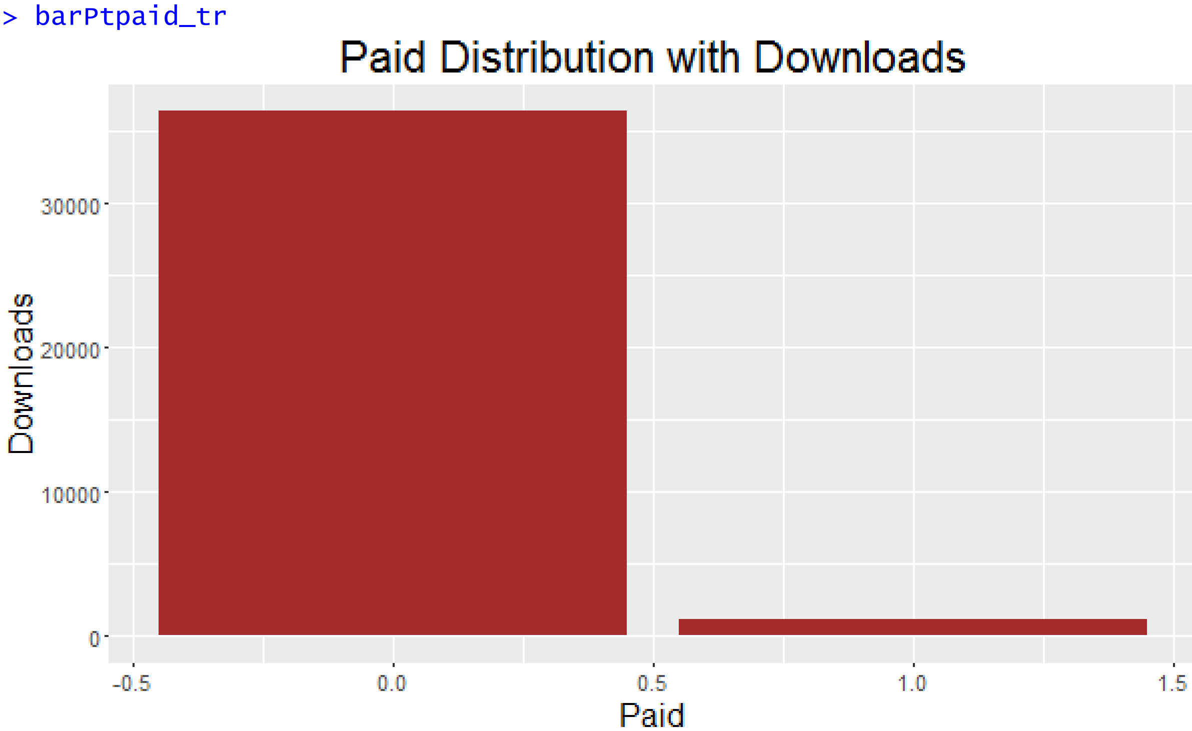

**Offers_In_App_Purchases vs Downloads**

```
> barPtoiap_tr = ggplot(data = main_subset, aes(x = offers_in_app_purchases,
y = downloads))
> barPtoiap_tr = barPtoiap_tr + geom_bar(stat = "identity", fill = "brown")
> barPtoiap_tr = barPtoiap_tr + labs(title = "OIAP Distribution with Download
s", x= "OIAP", y = "Downloads")
> barPtoiap_tr = barPtoiap_tr + theme(axis.title = element_text(size = 16), a
xis.text = element_text(size = 11), title = element_text(size = 18))
> barPtoiap_tr
```



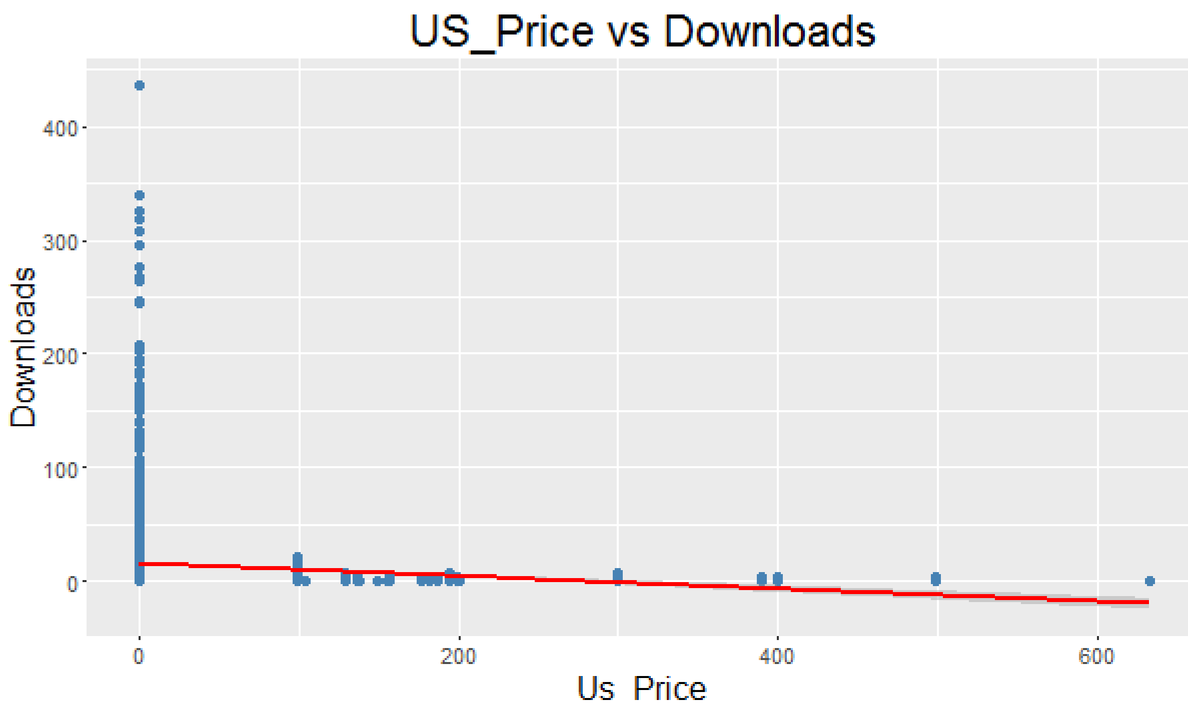

**Paid vs Downloads**

```
> barPtpaid_tr = ggplot(data = main_subset, aes(x = paid, y = downloads))
> barPtpaid_tr = barPtpaid_tr + geom_bar(stat = "identity", fill = "brown")
> barPtpaid_tr = barPtpaid_tr + labs(title = "Paid Distribution with Download
s", x= "Paid", y = "Downloads")
> barPtpaid_tr = barPtpaid_tr + theme(axis.title = element_text(size = 16), a
xis.text = element_text(size = 11), title = element_text(size = 18))
> barPtpaid_tr
```



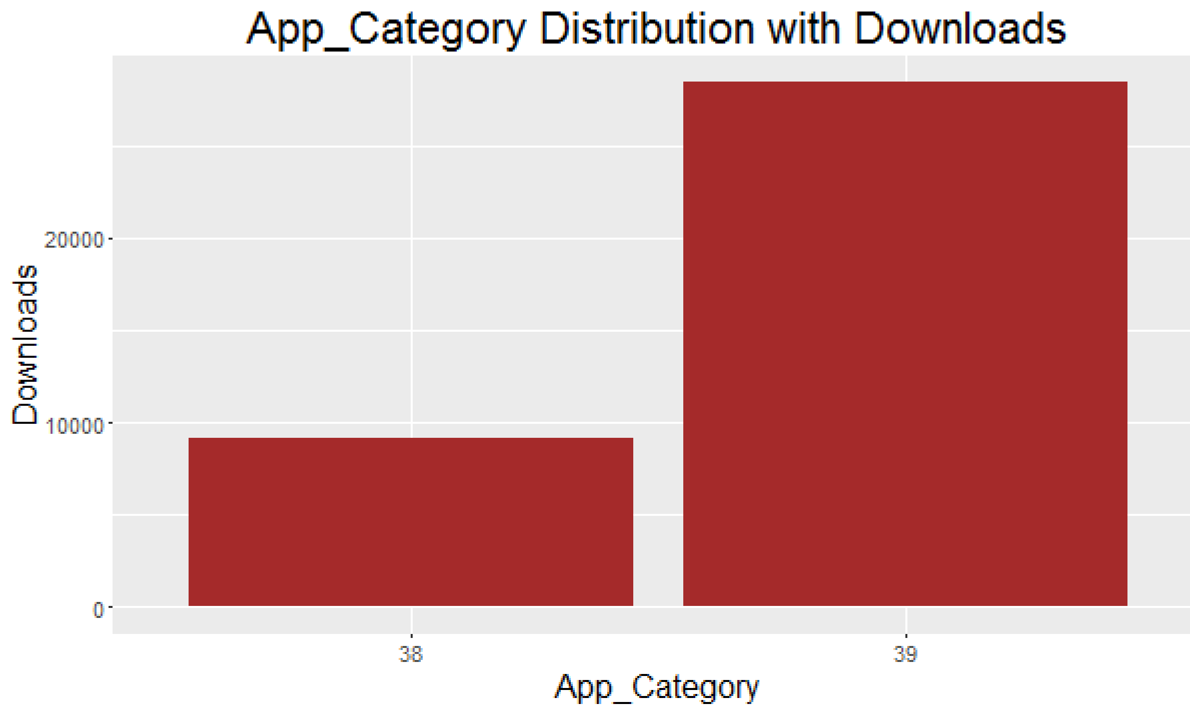
**Us_Price vs Downloads**

```
> spusprice <- ggplot(main_subset, aes(x = us_price, y = downloads))
> spusprice <- spusprice + geom_point(color = "steelblue", size = 2) + geom_s
mooth(method = "lm", color = "red")
> spusprice <- spusprice + labs(title = "US_Price vs Downloads", x = "Us_Pric
e", y = "Downloads")
> spusprice = spusprice + theme(axis.title = element_text(size = 16), axis.te
xt = element_text(size = 11), title = element_text(size = 18))
> spusprice
```



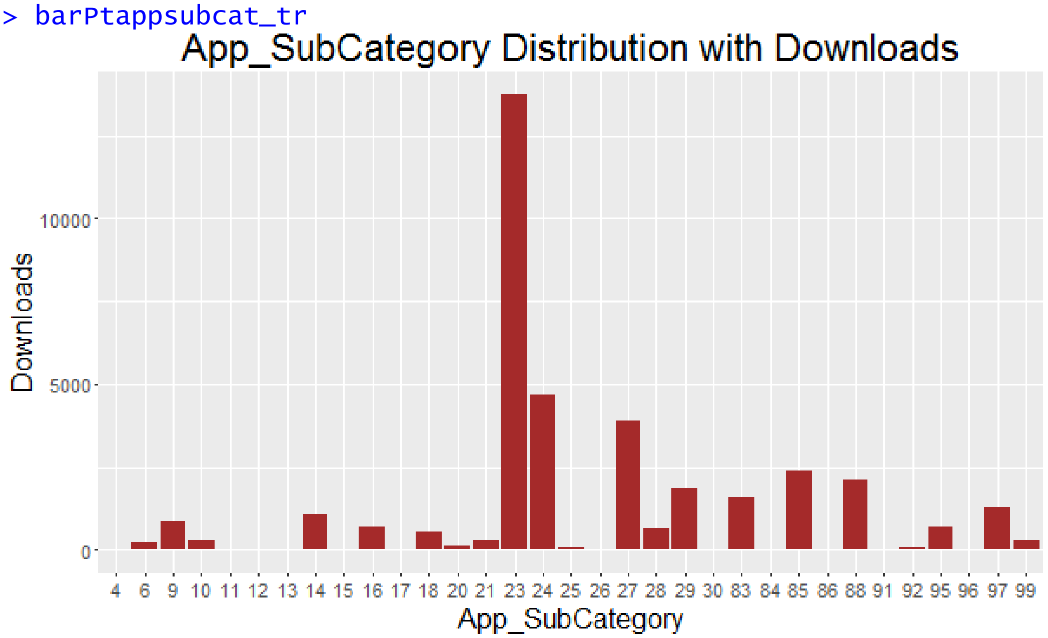

**App_category vs Downloads**

```
> barPtappcat_tr = ggplot(data = main_subset, aes(x = app_category, y = downl
oads))
> barPtappcat_tr = barPtappcat_tr + geom_bar(stat = "identity", fill = "brown
")
> barPtappcat_tr = barPtappcat_tr + labs(title = "App_Category Distribution w
ith Downloads", x= "App_Category", y = "Downloads")
> barPtappcat_tr = barPtappcat_tr + theme(axis.title = element_text(size = 16
), axis.text = element_text(size = 11), title = element_text(size = 18))
> barPtappcat_tr
```



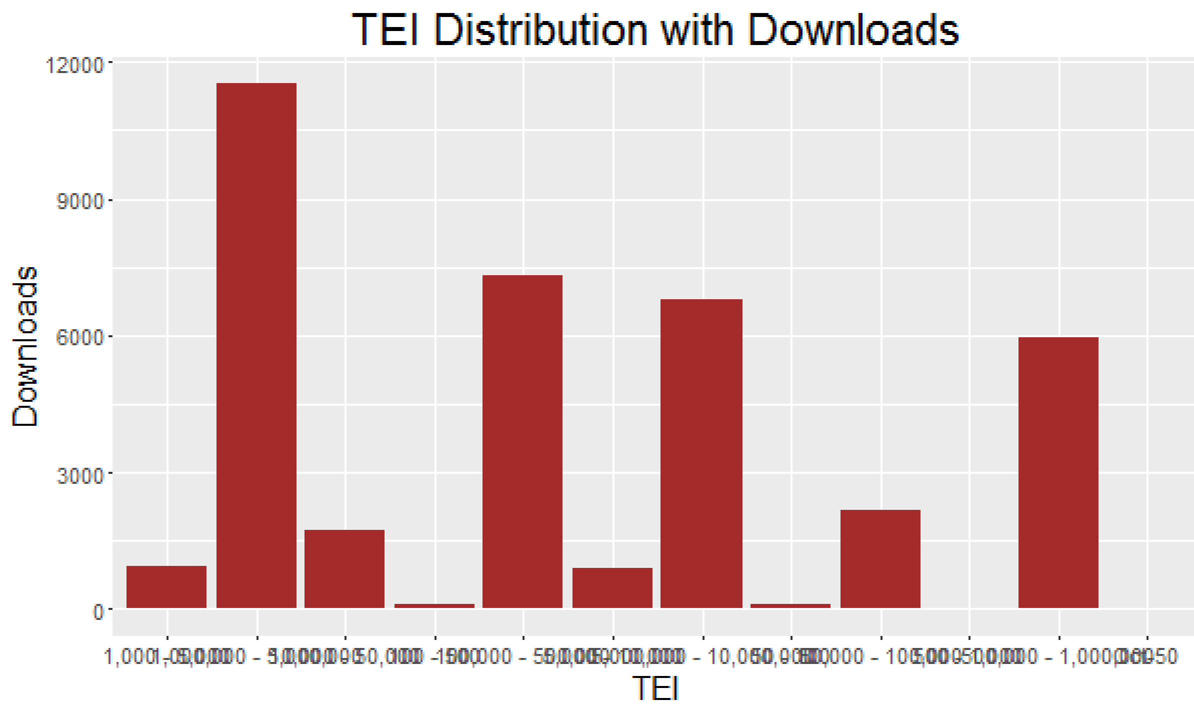

**App_SubCategory vs Downloads**

```
> barPtappsubcat_tr = ggplot(data = main_subset, aes(x = app_subcategory, y =
downloads))
> barPtappsubcat_tr = barPtappsubcat_tr + geom_bar(stat = "identity", fill =
"brown")
> barPtappsubcat_tr = barPtappsubcat_tr + labs(title = "App_SubCategory Distr
ibution with Downloads", x= "App_SubCategory", y = "Downloads")
> barPtappsubcat_tr = barPtappsubcat_tr + theme(axis.title = element_text(siz
e = 16), axis.text = element_text(size = 11), title = element_text(size = 18)
)
> barPtappsubcat_tr
```



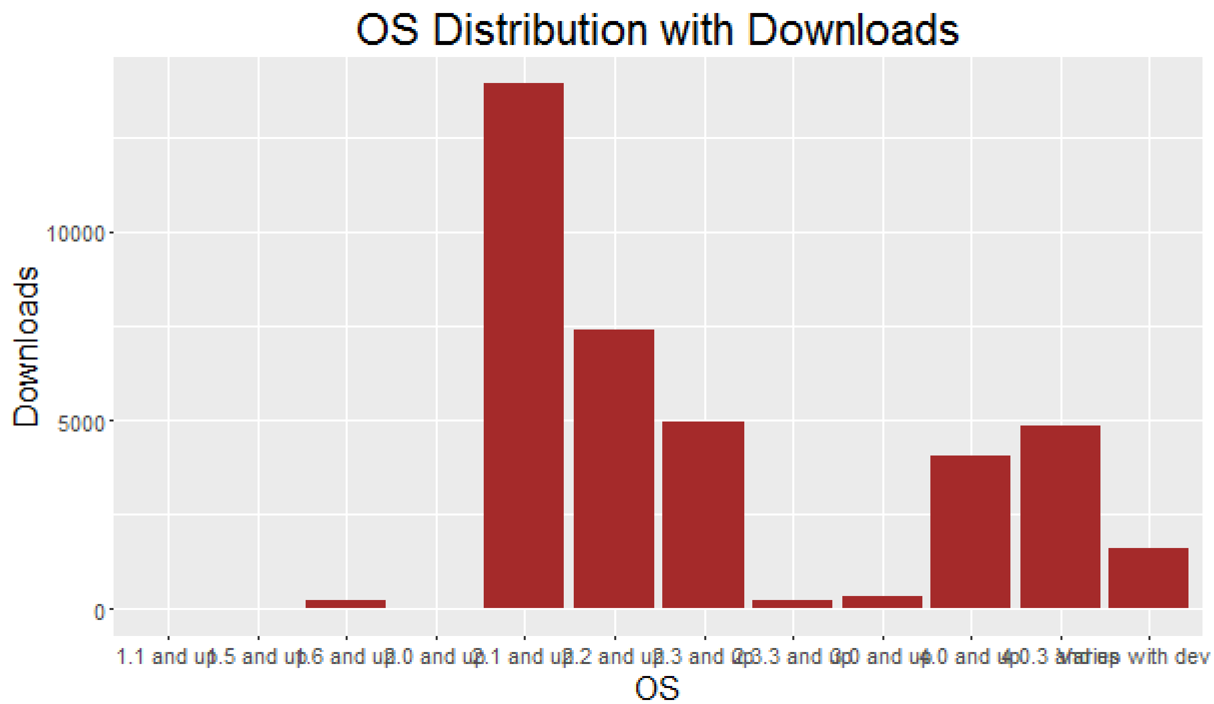
**Total_estimated_installs (TEI) vs Downloads**

```
> barPttei_tr = ggplot(data = main_subset, aes(x = total_estimated_installs,
y = downloads))
> barPttei_tr = barPttei_tr + geom_bar(stat = "identity", fill = "brown")
> barPttei_tr = barPttei_tr + labs(title = "TEI Distribution with Downloads",
x= "TEI", y = "Downloads")
> barPttei_tr = barPttei_tr + theme(axis.title = element_text(size = 16), axi
s.text = element_text(size = 11), title = element_text(size = 18))
> barPttei_tr
```



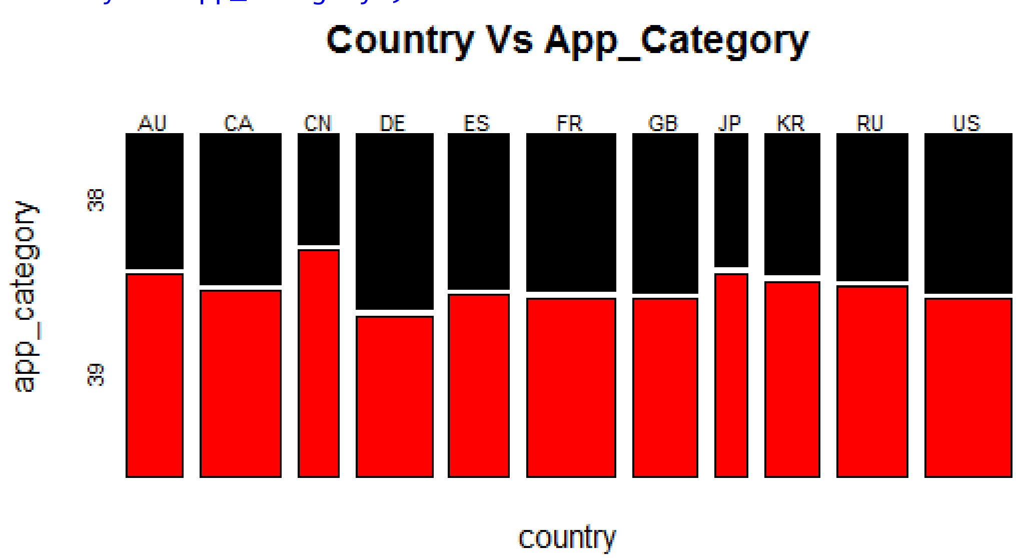

**Operating_sytem vs Downloads**

```
> barPtos_tr = ggplot(data = main_subset, aes(x = operating_system, y = downl
oads))
> barPtos_tr = barPtos_tr + geom_bar(stat = "identity", fill = "brown")
> barPtos_tr = barPtos_tr + labs(title = "OS Distribution with Downloads", x=
"OS", y = "Downloads")
> barPtos_tr = barPtos_tr + theme(axis.title = element_text(size = 16), axis.
text = element_text(size = 11), title = element_text(size = 18))
> barPtos_tr
```

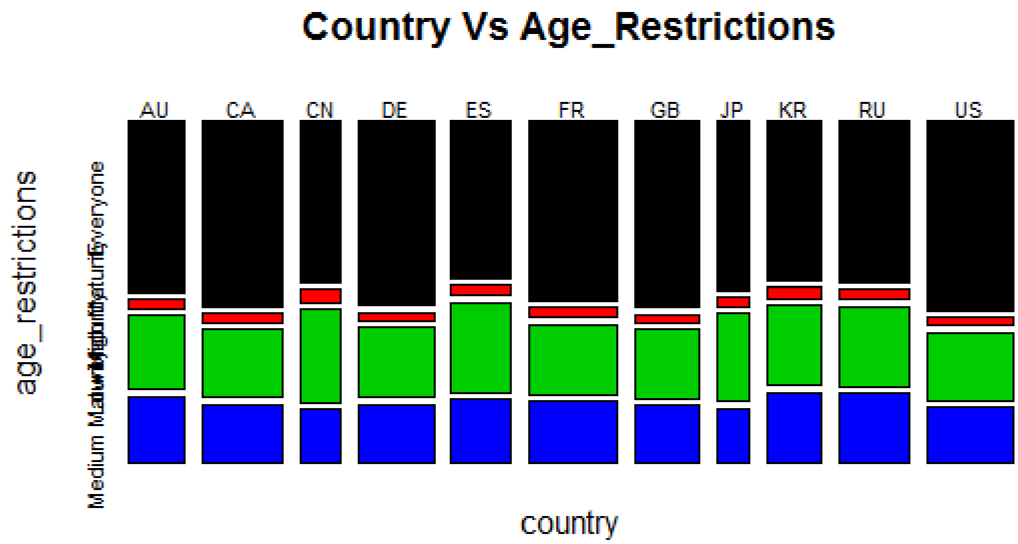



c. Plots between two categorical variables (Few)

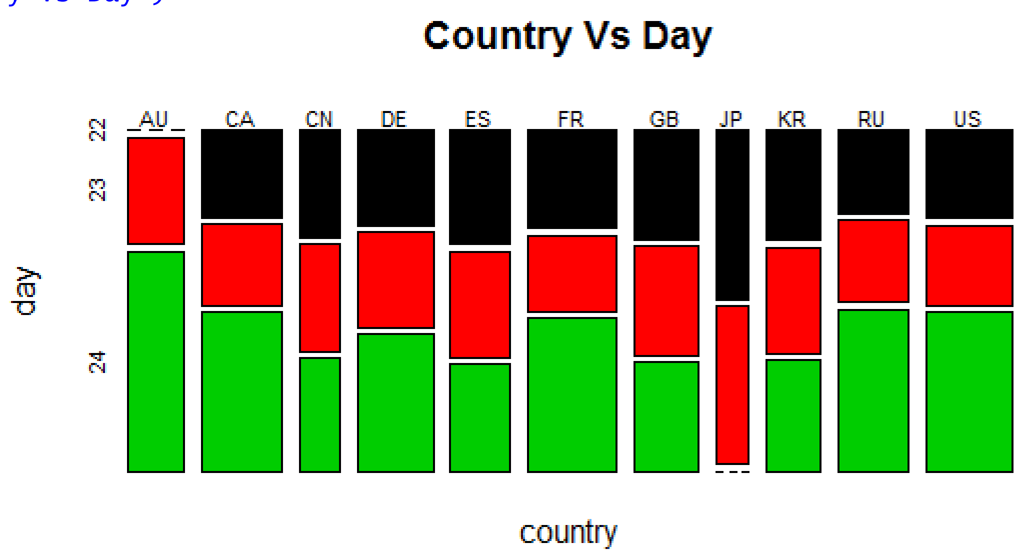
```
> mosaicplot(~ country + app_category, data = main_subset, color = 1:20, main
= "Country Vs App_Category")
```



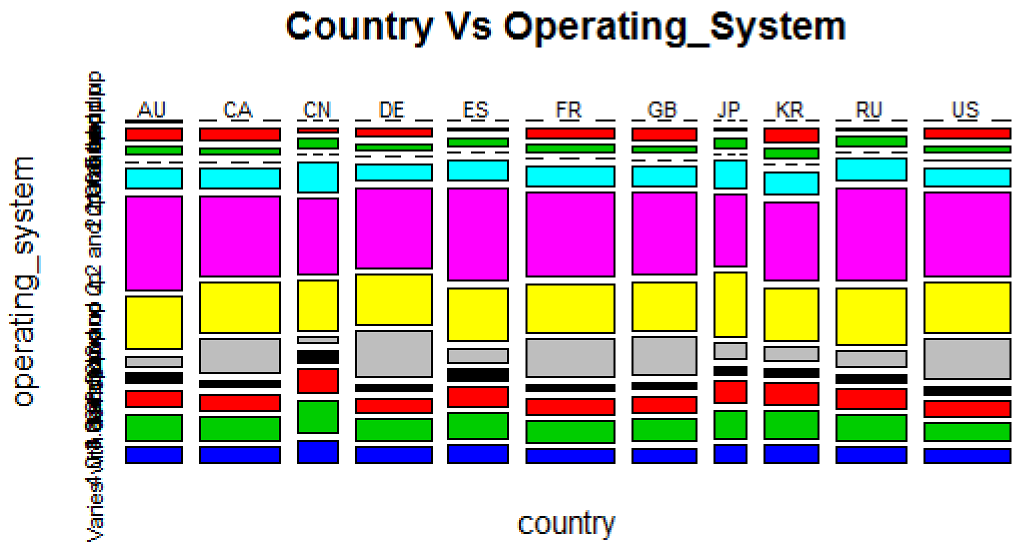
```
> mosaicplot(~ country + age_restrictions, data = main_subset, color = 1:20,
main = "Country Vs Age_Restrictions")
```




```
> mosaicplot(~ country + day, data = main_subset, color = 1:20, main = "Count
ry Vs Day")
```



```
> mosaicplot(~ country + operating_system, data = main_subset, color = 1:20,
main = "Country Vs Operating_System")
```




**Testing Dataset Preparation**

```
> mainTest = read.csv("test (data sci challenge).csv")
```

```
> str(mainTest)
'data.frame':   507 obs. of  14 variables:
 $ app_id                   : Factor w/ 13 levels "app_78","app_79",..: 1 1 1
1 1 1 1 1 1 1 ...
 $ date                     : Factor w/ 3 levels "10/22/2014","10/23/2014",..:
1 1 1 1 1 1 1 1 1 1 ...
 $ country                  : Factor w/ 11 levels "AU","CA","CN",..: 2 2 4 4 5
5 6 6 6 7 ...
 $ rank                     : int  292 324 323 291 291 323 238 884 266 323 ...
 $ rank_category            : int  14 14 14 14 14 14 14 14 14 14 ...
 $ rank_kind                : Factor w/ 3 levels "free","grossing",..: 3 2 2 3
3 2 3 2 2 2 ...
 $ age_restrictions         : Factor w/ 3 levels "Everyone","Low Maturity",..:
3 3 3 3 3 3 3 3 3 3 ...
 $ offers_in_app_purchases : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ paid                     : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
 $ us_price                 : int  499 499 499 499 499 499 499 499 499 499 ...
 $ app_category             : int  39 39 39 39 39 39 39 39 39 39 ...
 $ app_subcategory          : int  14 14 14 14 14 14 14 14 14 14 ...
 $ total_estimated_installs: Factor w/ 8 levels "1,000 - 5,000",..: 1 1 1 1 1
1 1 1 1 1 ...
 $ operating_system         : Factor w/ 8 levels "1.1 and up","1.6 and up",..:
4 4 4 4 4 4 4 4 4 4 ...
```

```
> mainTest$rank_category = factor(mainTest$rank_category)
> mainTest$app_category = factor(mainTest$app_category)
```

```
> mainTest$app_subcategory = factor(mainTest$app_subcategory)


> Day = format(as.Date(mainTest$date,format="%m/%d/%y"), "%d")
> mainTest$day <- as.integer(Day)


> mainTest = subset(mainTest, select =  -c(app_id,date))

> mainTest$ranks <- (mainTest$rank - min(mainTest$rank)) / (max(mainTest$rank
) - min(mainTest$rank))
> summary(main_subset$ranks)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.1573  0.2928  0.3232  0.4410  1.0000 



> for( i in 1 : nrow(mainTest)){
+   # Get the category
+   app_subcategory_temp = as.character(mainTest$app_subcategory[i])
+   rank_category_temp = as.character(mainTest$rank_category[i])
+   # Get the bins for these categories from downloadsByAppSubCat and downloa
dsByRankCat tables
+   app_subcatbin_temp = (filter(downloadsByAppSubCat, App_SubCategory == app
_subcategory_temp))$Bin_AppSubCat
+   rank_bin_temp = (filter(downloadsByRankCat, Rank_Category == rank_categor
y_temp))$Bin_RankCat
+   
+   # The length of the variable will be zero when it doesn't find the level 
in reference table, i.e., for 31.
+   # In that case, we will get the bin of category 30 and assign it to the r
ow
+   if(length(rank_bin_temp) == 0 || length(app_subcatbin_temp) == 0 ){
+     mainTest$Bin_AppSubCat[i] = (filter(downloadsByAppSubCat, App_SubCatego
ry == 30))$Bin_AppSubCat
+     mainTest$Bin_RankCat[i] = (filter(downloadsByRankCat, Rank_Category == 
30))$Bin_RankCat
+   }else{
+     mainTest$Bin_AppSubCat[i] = rank_bin_temp
+     mainTest$Bin_RankCat[i] = app_subcatbin_temp
+   }
+ }
> mainTest = subset(mainTest, select = -c(rank,rank_category,app_subcategory)
)
> str(mainTest)
'data.frame':	507 obs. of  13 variables:
 $ country                : Factor w/ 11 levels "AU","CA","CN",..: 2 2 4 4 5 
5 6 6 6 7 ...
 $ rank                   : int  292 324 323 291 291 323 238 884 266 323 ...
 $ rank_kind              : Factor w/ 3 levels "free","grossing",..: 3 2 2 3 
3 2 3 2 2 2 ...
 $ age_restrictions       : Factor w/ 3 levels "Everyone","Low Maturity",..: 
3 3 3 3 3 3 3 3 3 3 ...
 $ offers_in_app_purchases : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ paid                   : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
 $ us_price               : int  499 499 499 499 499 499 499 499 499 499 ...
```

```
 $ app_category           : Factor w/ 2 levels "38","39": 2 2 2 2 2 2 2 2 2 
2 ...
 $ total_estimated_installs: Factor w/ 8 levels "1,000 - 5,000",..: 1 1 1 1 1 
1 1 1 1 1 ...
 $ operating_system       : Factor w/ 8 levels "1.1 and up","1.6 and up",..: 
4 4 4 4 4 4 4 4 4 4 ...
 $ day                    : int  22 22 22 22 22 22 22 22 22 22 ...
 $ Bin_AppSubCat          : chr  "Bin_RankCat_2" "Bin_RankCat_2" "Bin_RankCa
t_2" "Bin_RankCat_2" ...
 $ Bin_RankCat            : chr  "Bin_AppSubCat_2" "Bin_AppSubCat_2" "Bin_Ap
pSubCat_2" "Bin_AppSubCat_2" ...
> mainTest$Bin_AppSubCat = as.factor(mainTest$Bin_AppSubCat)
> mainTest$Bin_RankCat = as.factor(mainTest$Bin_RankCat)


> cols <- sapply(mainTest, is.logical)
> mainTest[,cols] <- lapply(mainTest[,cols], as.integer)
> head(mainTest)
  country rank_kind age_restrictions offers_in_app_purchases paid us_price
1      CA      paid  Medium Maturity                       0    1      499
2      CA  grossing  Medium Maturity                       0    1      499
3      DE  grossing  Medium Maturity                       0    1      499
4      DE      paid  Medium Maturity                       0    1      499
5      ES      paid  Medium Maturity                       0    1      499
6      ES  grossing  Medium Maturity                       0    1      499
  app_category total_estimated_installs operating_system day     ranks Bin_Ap
pSubCat
1           39            1,000 - 5,000       2.2 and up  22 0.2229787 Bin_Ra
nkCat_2
2           39            1,000 - 5,000       2.2 and up  22 0.2502128 Bin_Ra
nkCat_2
3           39            1,000 - 5,000       2.2 and up  22 0.2493617 Bin_Ra
nkCat_2
4           39            1,000 - 5,000       2.2 and up  22 0.2221277 Bin_Ra
nkCat_2
5           39            1,000 - 5,000       2.2 and up  22 0.2221277 Bin_Ra
nkCat_2
6           39            1,000 - 5,000       2.2 and up  22 0.2493617 Bin_Ra
nkCat_2
      Bin_RankCat
1 Bin_AppSubCat_2
2 Bin_AppSubCat_2
3 Bin_AppSubCat_2
4 Bin_AppSubCat_2
5 Bin_AppSubCat_2
6 Bin_AppSubCat_2
```

**Predicting 'NA' values**

```
> names(data.un)
 [1] "app_id"                   "date"                    "country"
 [4] "rank"                     "rank_category"           "rank_kind"
 [7] "age_restrictions"         "offers_in_app_purchases" "paid"
[10] "us_price"                 "app_category"            "app_subcategory"
[13] "total_estimated_installs" "operating_system"        "downloads"
```

```
> #******************************** Scoring NA data #*************************
************#
> names(data.un)
 [1] "app_id"                  "date"                     "country"
 [4] "rank"                    "rank_category"            "rank_kind"
 [7] "age_restrictions"        "offers_in_app_purchases"  "paid"
[10] "us_price"                "app_category"             "app_subcategory"
[13] "total_estimated_installs" "operating_system"         "downloads"
> data.un <- data.un[,-15]
> # Find data types of columns
> str(data.un)
'data.frame':	68 obs. of  14 variables:
 $ app_id                  : Factor w/ 77 levels "app_1","app_10",..: 22 22 2
2 22 22 22 22 22 22 22 ...
 $ date                    : Factor w/ 3 levels "10/22/2014","10/23/2014",..:
3 3 3 3 2 2 2 2 2 2 ...
 $ country                 : Factor w/ 11 levels "AU","CA","CN",..: 1 1 1 1 1
1 11 7 7 8 8 ...
 $ rank                    : int  105 105 94 94 94 94 94 94 105 105 ...
 $ rank_category           : int  28 28 28 28 28 28 28 28 28 28 ...
 $ rank_kind               : Factor w/ 5 levels "free","grossing",..: 2 2 1 1
1 1 1 1 2 2 ...
 $ age_restrictions        : Factor w/ 4 levels "Everyone","High Maturity",..
: 3 3 3 3 3 3 3 3 3 3 ...
 $ offers_in_app_purchases : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
 $ paid                    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ us_price                : int  0 0 0 0 0 0 0 0 0 0 ...
 $ app_category            : int  38 38 38 38 38 38 38 38 38 38 ...
 $ app_subcategory         : int  28 28 28 28 28 28 28 28 28 28 ...
 $ total_estimated_installs: Factor w/ 12 levels "1,000 - 5,000",..: 2 2 2 2
2 2 2 2 2 2 ...
 $ operating_system        : Factor w/ 12 levels "1.1 and up","1.5 and up",..
: 7 10 7 10 7 10 7 10 7 10 ...
>
> ############ Converting datatypes
> data.un$rank_category = factor(data.un$rank_category)
> data.un$app_category = factor(data.un$app_category)
> data.un$app_subcategory = factor(data.un$app_subcategory)
>
> # Generate new columns
> Day = format(as.Date(data.un$date,format="%m/%d/%y"), "%d")
>
> # Add columns to dataframe
> data.un$day <- as.integer(Day)
>
> # Eliminating unnecessary columns
> data.un = subset(data.un, select =  -c(app_id,date))
>
> # Normalizing 'Rank' column
> summary(data.un$rank)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   74.0    93.0    94.0   231.6   105.0   717.0
> data.un$ranks <- (data.un$rank - min(data.un$rank)) / (max(data.un$rank) -
min(data.un$rank))
> summary(data.un$ranks)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.00000 0.02955 0.03110 0.24500 0.04821 1.00000
```

```
>
> # Loop through the test data and assign bins to rank_category and app_subca
tegory
> # For the additional level 31, assign the same to which category 30 is assi
gned.
> for( i in 1 : nrow(data.un)){
+   # Get the category
+   app_subcategory_temp = as.character(data.un$app_subcategory[i])
+   rank_category_temp = as.character(data.un$rank_category[i])
+   # Get the bins for these categories from downloadsByAppSubCat and downloa
dsByRankCat tables
+   app_subcatbin_temp = (filter(downloadsByAppSubCat, App_SubCategory == app
_subcategory_temp))$Bin_AppSubCat
+   rank_bin_temp = (filter(downloadsByRankCat, Rank_Category == rank_categor
y_temp))$Bin_RankCat
+
+   # The length of the variable will be zero when it doesn't find the level
in reference table, i.e., for 31.
+   # In that case, we will get the bin of category 30 and assign it to the r
ow
+   if(length(rank_bin_temp) == 0 || length(app_subcatbin_temp) == 0 ){
+     data.un$Bin_AppSubCat[i] = (filter(downloadsByAppSubCat, App_SubCategor
y == 30))$Bin_AppSubCat
+     data.un$Bin_RankCat[i] = (filter(downloadsByRankCat, Rank_Category == 3
0))$Bin_RankCat
+   }else{
+     data.un$Bin_AppSubCat[i] = rank_bin_temp
+     data.un$Bin_RankCat[i] = app_subcatbin_temp
+   }
+ }
>
> # Remove rank_category, app_subcategory
> data.un = subset(data.un, select = -c(rank,rank_category,app_subcategory))
>
> str(data.un)
'data.frame':   68 obs. of  13 variables:
 $ country                 : Factor w/ 11 levels "AU","CA","CN",..: 1 1 1 1 1
1 11 7 7 8 8 ...
 $ rank_kind               : Factor w/ 5 levels "free","grossing",..: 2 2 1 1
1 1 1 1 2 2 ...
 $ age_restrictions        : Factor w/ 4 levels "Everyone","High Maturity",..
: 3 3 3 3 3 3 3 3 3 3 ...
 $ offers_in_app_purchases : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
 $ paid                    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ us_price                : int  0 0 0 0 0 0 0 0 0 0 ...
 $ app_category            : Factor w/ 1 level "38": 1 1 1 1 1 1 1 1 1 1 ...
 $ total_estimated_installs: Factor w/ 12 levels "1,000 - 5,000",..: 2 2 2 2
2 2 2 2 2 2 ...
 $ operating_system        : Factor w/ 12 levels "1.1 and up","1.5 and up",..
: 7 10 7 10 7 10 7 10 7 10 ...
 $ day                     : int  24 24 24 24 23 23 23 23 23 23 ...
 $ ranks                   : num  0.0482 0.0482 0.0311 0.0311 0.0311 ...
 $ Bin_AppSubCat           : chr  "Bin_RankCat_2" "Bin_RankCat_2" "Bin_RankCa
t_2" "Bin_RankCat_2" ...
 $ Bin_RankCat             : chr  "Bin_AppSubCat_2" "Bin_AppSubCat_2" "Bin_Ap
pSubCat_2" "Bin_AppSubCat_2" ...
>
```

```
> # Convert the character type columns to Factor type
> data.un$Bin_AppSubCat = as.factor(data.un$Bin_AppSubCat)
> data.un$Bin_RankCat = as.factor(data.un$Bin_RankCat)
>
> # Converting logical to binary
> cols <- sapply(data.un, is.logical)
> data.un[,cols] <- lapply(data.un[,cols], as.integer)
> head(data.un)
     country rank_kind age_restrictions offers_in_app_purchases paid us_price
1049      AU  grossing     Low Maturity                       1    0        0
1050      AU  grossing     Low Maturity                       1    0        0
1051      AU      free     Low Maturity                       1    0        0
1052      AU      free     Low Maturity                       1    0        0
1053      US      free     Low Maturity                       1    0        0
1054      US      free     Low Maturity                       1    0        0
     app_category total_estimated_installs operating_system day      ranks
1049           38    1,000,000 - 5,000,000       2.3 and up  24 0.04821151
1050           38    1,000,000 - 5,000,000       4.0 and up  24 0.04821151
1051           38    1,000,000 - 5,000,000       2.3 and up  24 0.03110420
1052           38    1,000,000 - 5,000,000       4.0 and up  24 0.03110420
1053           38    1,000,000 - 5,000,000       2.3 and up  23 0.03110420
1054           38    1,000,000 - 5,000,000       4.0 and up  23 0.03110420
     Bin_AppSubCat     Bin_RankCat
1049 Bin_RankCat_2 Bin_AppSubCat_2
1050 Bin_RankCat_2 Bin_AppSubCat_2
1051 Bin_RankCat_2 Bin_AppSubCat_2
1052 Bin_RankCat_2 Bin_AppSubCat_2
1053 Bin_RankCat_2 Bin_AppSubCat_2
1054 Bin_RankCat_2 Bin_AppSubCat_2
>
> # Match training and testing factor levels
> for(colName in names(data.un)){
+   if(is.factor(data.un[[colName]])){
+     levels(data.un[[colName]]) = levels(main_subset[[colName]])
+   }
+ }
>
> data.un$naPred = predict(randForModel, data.un) # New factor levels not pre
sent in the training data
>
> round(data.un$naPred)
 [1] 18 18 19 18 90 81 19 18 15 15 79 74  8  7 23 22 24 24 10  7 20 19 11  9
20 20 10
[28]  8 16 15 22 21 18 16 16 16 79 78 83 78 19 18  9  7 10  8 16 16 17 17 20
20 78 74
[55] 17 17 16 15 20 18 15 15 19 18 95 89 14 15
```

**Correlations (Few)**

```
> attach(main_data)
The following objects are masked from main_data (pos = 4):

    age_restrictions, app_category, app_id, app_subcategory, country, date,
    day, downloads, offers_in_app_purchases, operating_system, paid, rank,
    rank_category, rank_kind, total_estimated_installs, us_price

> chisq.test(table(country,as.factor(rank_category))) # Variables are indepen
dent, because p is > 0.05 and hence we fail to reject null hypothesis

        Pearson's Chi-squared test

data:  table(country, as.factor(rank_category))
X-squared = 329.96, df = 370, p-value = 0.9337


> chisq.test(table(rank_category,as.factor(app_subcategory))) # There is an a
ssociation because p is < 0.05 and hence we reject null hypothesis

        Pearson's Chi-squared test

data:  table(rank_category, as.factor(app_subcategory))
X-squared = 115810, df = 1184, p-value < 2.2e-16


> chisq.test(table(country, rank_kind))

        Pearson's Chi-squared test

data:  table(country, rank_kind)
X-squared = 118.07, df = 40, p-value = 1.235e-09


> chisq.test(table(country, age_restrictions))

        Pearson's Chi-squared test

data:  table(country, age_restrictions)
X-squared = 26.57, df = 30, p-value = 0.6458

> chisq.test(table(app_category, app_subcategory))

        Pearson's Chi-squared test

data:  table(app_category, app_subcategory)
X-squared = 3622, df = 32, p-value < 2.2e-16


> chisq.test(table(total_estimated_installs, operating_system))

        Pearson's Chi-squared test

> chisq.test(table(country, total_estimated_installs))

        Pearson's Chi-squared test
```

```
data:  table(country, total_estimated_installs)
X-squared = 265.12, df = 110, p-value = 7.943e-15

> chisq.test(table(country, operating_system))

	Pearson's Chi-squared test

data:  table(country, operating_system)
X-squared = 180.84, df = 110, p-value = 2.431e-05


> chisq.test(table(offers_in_app_purchases, paid))

	Pearson's Chi-squared test with Yates' continuity correction

data:  table(offers_in_app_purchases, paid)
X-squared = 1059.2, df = 1, p-value < 2.2e-16


> detach(main_data)


> counry_aov <- aov(downloads~country, data = main_data)
> summary(country)
 AU  CA  CN  DE  ES  FR  GB  JP  KR  RU  US 
285 409 211 385 301 454 336 165 274 365 437 
> tei_aov <- aov(downloads ~ total_estimated_installs, data = main_data)
> summary(tei_aov)
                           Df  Sum Sq Mean Sq F value Pr(>F)    
total_estimated_installs   11 1135544  103231   100.3 <2e-16 ***
Residuals                3610 3714203    1029                   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> os_aov <- aov(downloads ~ operating_system, data = main_data)
> summary(os_aov)
                   Df  Sum Sq Mean Sq F value Pr(>F)    
operating_system   11  524716   47701   39.81 <2e-16 ***
Residuals        3610 4325032    1198                   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> kind_aov <- aov(downloads ~ rank_kind, data = main_data)
> summary(kind_aov)
              Df  Sum Sq Mean Sq F value Pr(>F)    
rank_kind      4  362385   90596   73.02 <2e-16 ***
Residuals   3617 4487362    1241                   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> rankcat_aov <- aov(downloads ~ rank_category, data = main_data)
> summary(rankcat_aov)
                Df  Sum Sq Mean Sq F value Pr(>F)    
rank_category   37  648306   17522   14.95 <2e-16 ***
Residuals     3584 4201442    1172                   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> appcat_aov <- aov(downloads ~ app_category, data = main_data)
> summary(appcat_aov)
```

```
                Df  Sum Sq Mean Sq F value   Pr(>F)    
app_category     1   65358   65358   49.45 2.42e-12 ***
Residuals     3620 4784389    1322                     
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> appsubcat_aov <- aov(downloads ~ app_subcategory, data = main_data)
> summary(appsubcat_aov)
                  Df  Sum Sq Mean Sq F value Pr(>F)    
app_subcategory   32  484725   15148   12.46 <2e-16 ***
Residuals       3589 4365023    1216                   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```