# Customer_Experience_in_R

## R Programming: Customer Experience in R

### Example

```
# Importing the data.table
# ---
#
library("data.table")
library(stats)
library(psych)
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```
# Reading our dataset
# ---
#
hospitality_dt <- fread('http://bit.ly/HospitalityDataset')
View(hospitality_dt)
attach(hospitality_dt)
```

```
# What is the structure of the data?
# ---
#
head(hospitality_dt)
```

```
##    user_id gender timestamp survey_completion score amount          branch
## 1:  621602      M   11:58.1          TIMEDOUT     -   1320   Nairobi South
## 2:  242833      F   45:20.0          FINISHED     5   1460 Nairobi Central
## 3:  621602      M   00:36.0          TIMEDOUT     -   1270   Nairobi South
## 4:  621602      M   10:15.0          TIMEDOUT     -    700   Nairobi North
## 5: 6345755      M   54:58.1          TIMEDOUT     -    680   Nairobi North
## 6:  751525      M   35:52.7          TIMEDOUT     -    460    Nairobi West
```

```
# How many variables and observations are there?
#
ncol(hospitality_dt)
```

```
## [1] 7
```

```
nrow(hospitality_dt)
```

```
## [1] 296852
```

```
#learn more about the dataset
help(hospitality_dt)
```

```
## No documentation for 'hospitality_dt' in specified packages and libraries:
```

```
## you could try '??hospitality_dt'
```

```
??hospitality_dt
```

```
## starting httpd help server ... done
```

```
str(hospitality_dt)
```

```
## Classes 'data.table' and 'data.frame':   296852 obs. of  7 variables:
##  $ user_id          : int  621602 242833 621602 621602 6345755 751525 6591998 401557 17887026 1697459
##  $ gender           : chr  "M" "F" "M" "M" ...
##  $ timestamp        : chr  "11:58.1" "45:20.0" "00:36.0" "10:15.0" ...
##  $ survey_completion: chr  "TIMEDOUT" "FINISHED" "TIMEDOUT" "TIMEDOUT" ...
##  $ score            : chr  "-" "5" "-" "-" ...
##  $ amount           : int  1320 1460 1270 700 680 460 570 1820 260 690 ...
##  $ branch           : chr  "Nairobi South" "Nairobi Central" "Nairobi South" "Nairobi North" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
class(hospitality_dt)
```

```
## [1] "data.table" "data.frame"
```

```
typeof(hospitality_dt)
```

```
## [1] "list"
```

```
length(hospitality_dt)
```

```
## [1] 7
```

```
names(hospitality_dt) #display variable names
```

```
## [1] "user_id"           "gender"            "timestamp"
## [4] "survey_completion" "score"             "amount"
## [7] "branch"
```

```
#attributes(hospitality_dt) #names(hospitality_dt), class(hospitality_dt), row.names(hospitality_dt)
```

```
# What is the missing data?
#
sum(is.na(hospitality_dt))
```

```
## [1] 0
```

```
# NB: Let's deal with "-" in our scores variable
# Assumption is that those customers did not fill in the survey
#
hospitality_dt$score[hospitality_dt$score == "-"] <- NA

head(hospitality_dt)
```

```
##    user_id gender timestamp survey_completion score amount          branch
## 1:  621602      M   11:58.1          TIMEDOUT  <NA>   1320   Nairobi South
## 2:  242833      F   45:20.0          FINISHED     5   1460 Nairobi Central
## 3:  621602      M   00:36.0          TIMEDOUT  <NA>   1270   Nairobi South
## 4:  621602      M   10:15.0          TIMEDOUT  <NA>    700   Nairobi North
## 5: 6345755      M   54:58.1          TIMEDOUT  <NA>    680   Nairobi North
## 6:  751525      M   35:52.7          TIMEDOUT  <NA>    460    Nairobi West
```

```
# Getting rid of missing data, check size and preview
# Size of original dataset was 296852
```

```
#
hospitality_dt1 <- na.omit(hospitality_dt)
nrow(hospitality_dt1)
```

```
## [1] 36402
```

```
head(hospitality_dt1)
```

```
##     user_id gender timestamp survey_completion score amount          branch
## 1:   242833      F   45:20.0          FINISHED     5   1460 Nairobi Central
## 2:  1697459      M   39:01.6          TIMEDOUT     9    690    Nairobi East
## 3: 17144551      F   55:19.5          TIMEDOUT     0   1380 Nairobi Central
## 4: 17887216      F   00:38.1          TIMEDOUT     9    990   Nairobi South
## 5:   630299      F   03:49.9          TIMEDOUT     9    840    Nairobi West
## 6:   607011      M   20:46.1          TIMEDOUT    10    460   Nairobi South
```

```
View(hospitality_dt1)
attach(hospitality_dt1)
```

```
## The following objects are masked from hospitality_dt:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id
```

```
# What is the overall proportion of repeat customers?
#duplicated() function uses logical values to determine duplicated values.

#duplicated(hospitality_dt1$user_id)

sum(duplicated(hospitality_dt1$user_id))
```

```
## [1] 6749
```

```
dim(hospitality_dt1[duplicated(hospitality_dt1$user_id),])[1] #gives you number of duplicates
```

```
## [1] 6749
```

```
table(duplicated(hospitality_dt1$user_id))
```

```
##
## FALSE  TRUE
## 29653  6749
```

```
mean(duplicated(hospitality_dt1$user_id))
```

```
## [1] 0.1854019
```

```
sum(duplicated(hospitality_dt1$user_id)) / nrow(hospitality_dt1)
```

```
## [1] 0.1854019
```

```
# How many times do customers come back on average?


#unique() function uses numeric indicators to determine unique values.

library(plyr)

#unique(hospitality_dt1$user_id)
```

```
#count(unique(hospitality_dt1$user_id))

#table(unique(hospitality_dt1$user_id))

dim(hospitality_dt1[unique(hospitality_dt1$user_id),])[1] #gives you number of uniques
```

```
## [1] 29653
```

```
# How many customers are repeat customers per branch?
#
sum(duplicated(hospitality_dt1[,c('user_id','branch')]))
```

```
## [1] 4574
```

```
# What is the NPS?
#

# Importing our NPS library
#
library(NPS)

# Converting score column to numeric
#
hospitality_dt1$score <- as.numeric(as.character(hospitality_dt1$score))

# Computing our NPS
nps(hospitality_dt1$score)
```

```
## [1] 0.6367782
```

```
# Here are the proportions of respondents giving each Likelihood to
# recommend response
#
prop.table(table(hospitality_dt1$score))
```

```
##
##           0           1           2           3           4           5
## 0.031893852 0.009147849 0.009834624 0.009422559 0.010109335 0.023872315
##           6           7           8           9          10
## 0.018900060 0.041069172 0.095791440 0.098538542 0.651420252
```

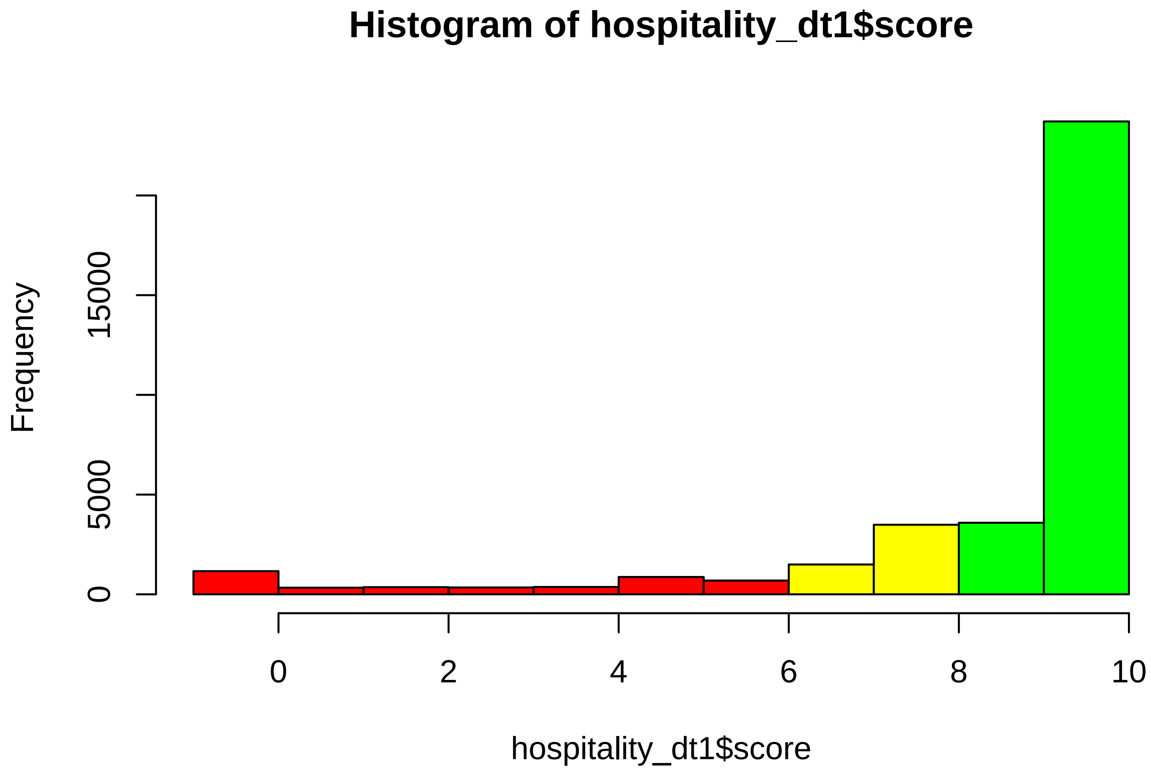
```
# Plotting a histrogram of the scores
#

# Lets first import tidyverse
#
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------------ tidyverse 1
```

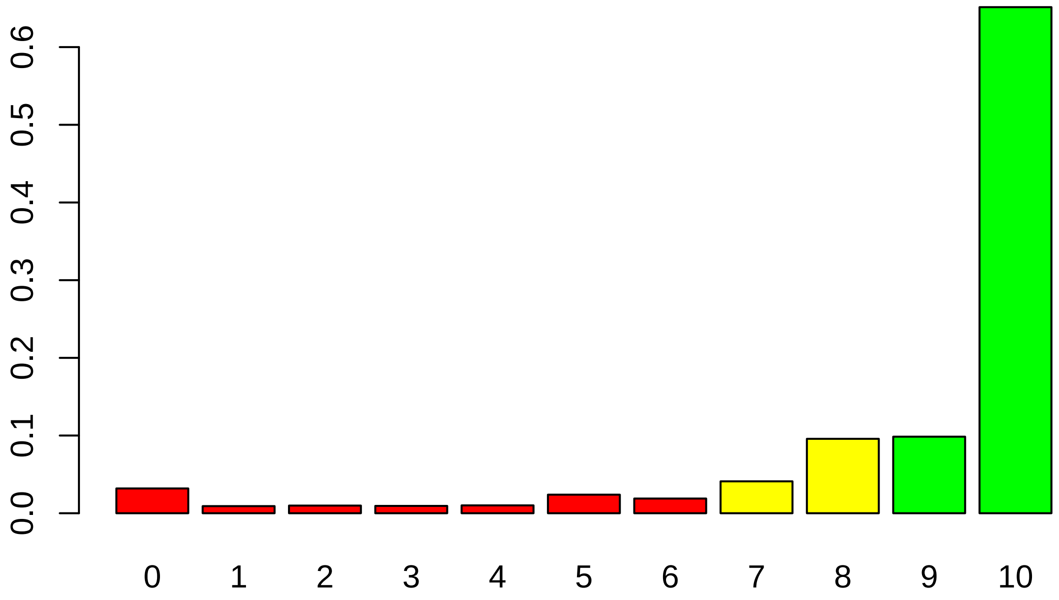
```
## <U+2713> tibble  2.1.3     <U+2713> dplyr   0.8.3
## <U+2713> tidyr   1.0.0     <U+2713> stringr 1.4.0
## <U+2713> readr   1.3.1     <U+2713> forcats 0.4.0
## <U+2713> purrr   0.3.3
```

```
## -- Conflicts ------------------------------------------------------------------------ tidyverse_conflic
## x ggplot2::%+%()     masks psych::%+%()
## x ggplot2::alpha()   masks psych::alpha()
## x dplyr::arrange()   masks plyr::arrange()
## x dplyr::between()   masks data.table::between()
## x purrr::compact()   masks plyr::compact()
## x dplyr::count()     masks plyr::count()
## x dplyr::failwith()  masks plyr::failwith()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::id()        masks plyr::id()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x dplyr::mutate()    masks plyr::mutate()
## x dplyr::rename()    masks plyr::rename()
## x dplyr::summarise() masks plyr::summarise()
## x dplyr::summarize() masks plyr::summarize()
## x purrr::transpose() masks data.table::transpose()
```

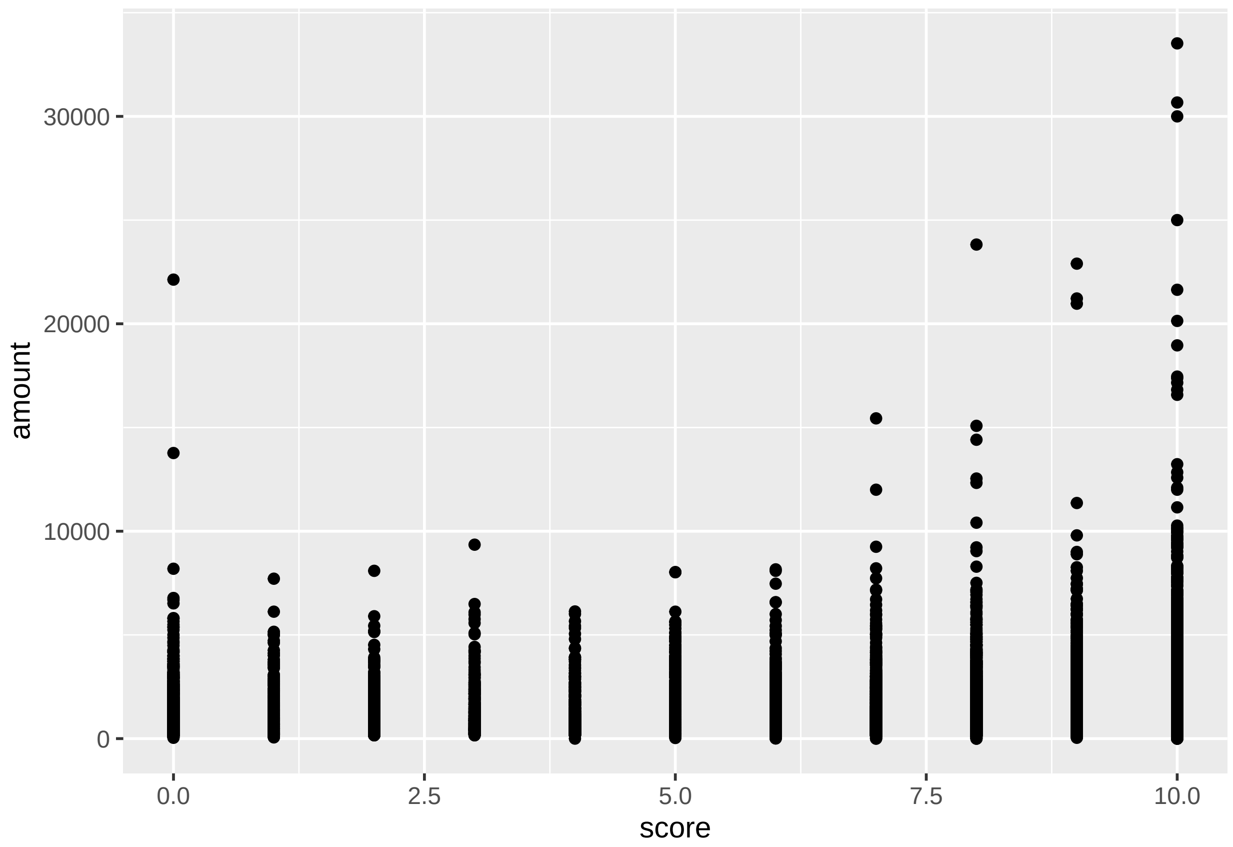
```
hist(
  hospitality_dt1$score, breaks = -1:10,
  col = c(rep("red", 7), rep("yellow", 2), rep("green", 2))
)
```



```
# Here's a barplot. It's very similar, though for categorical responses
# it's often slightly easier to interpret
#
barplot(
 prop.table(table(hospitality_dt1$score)),
 col = c(rep("red", 7), rep("yellow", 2), rep("green", 2))
)
```



```
# Is there a relationship between NPS segment and amount spent?
#
ggplot(hospitality_dt1, aes(x=score, y=amount)) + geom_point()
```



### Exercise

```
#Build a data model with unique id only


hospitality_dt1[!duplicated(hospitality_dt1$user_id),] #gives you unique rows
```

```
##         user_id gender timestamp survey_completion score amount          branch
##     1:   242833      F   45:20.0          FINISHED     5   1460 Nairobi Central
##     2:  1697459      M   39:01.6          TIMEDOUT     9    690    Nairobi East
##     3: 17144551      F   55:19.5          TIMEDOUT     0   1380 Nairobi Central
##     4: 17887216      F   00:38.1          TIMEDOUT     9    990   Nairobi South
##     5:   630299      F   03:49.9          TIMEDOUT     9    840    Nairobi West
##    ---
## 29649:   423355      M   00:28.5          FINISHED    10   1040       Satellite
## 29650:  1235116      M   04:42.4          TIMEDOUT     8    580    Nairobi West
## 29651: 18205871      M   40:54.7          FINISHED     3   1600   Nairobi South
## 29652:   677307      F   25:32.0          FINISHED    10    570    Nairobi West
## 29653:    97324      F   54:03.2          FINISHED    10    530 Nairobi Central
```

```
#Data with unique id only
hospitality_dt2u <- hospitality_dt1[!duplicated(hospitality_dt1$user_id),]
View(hospitality_dt2u)
attach(hospitality_dt2u)
```

```
## The following objects are masked from hospitality_dt1:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id
```

```
## The following objects are masked from hospitality_dt:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id
```

```r
nrow(hospitality_dt2u)
```

```
## [1] 29653
```

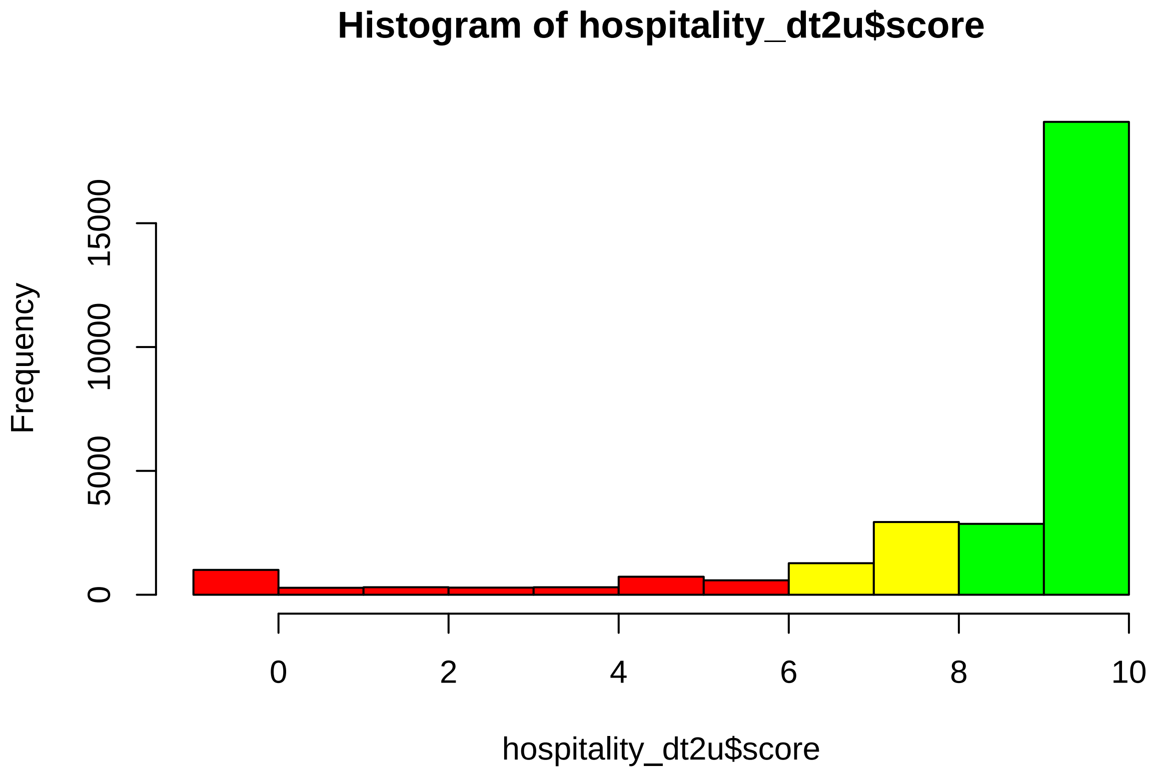
```r
# Converting score column to numeric
hospitality_dt2u$score <- as.numeric(as.character(hospitality_dt2u$score))

# Computing our NPS
nps(hospitality_dt2u$score)
```

```
## [1] 0.6227026
```

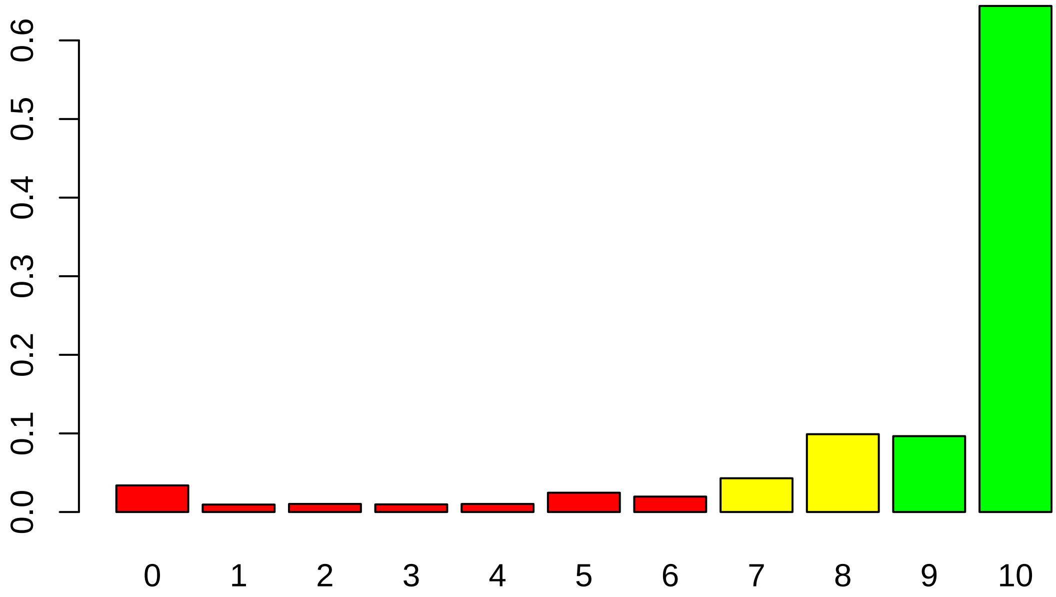
```r
# proportions of respondents giving each Likelihood to

prop.table(table(hospitality_dt2u$score))
```

```
##
##           0           1           2           3           4           5
## 0.033824571 0.009476276 0.010251914 0.009712339 0.010218190 0.024550636
##           6           7           8           9          10
## 0.019627019 0.042963612 0.099011904 0.096516373 0.643847166
```

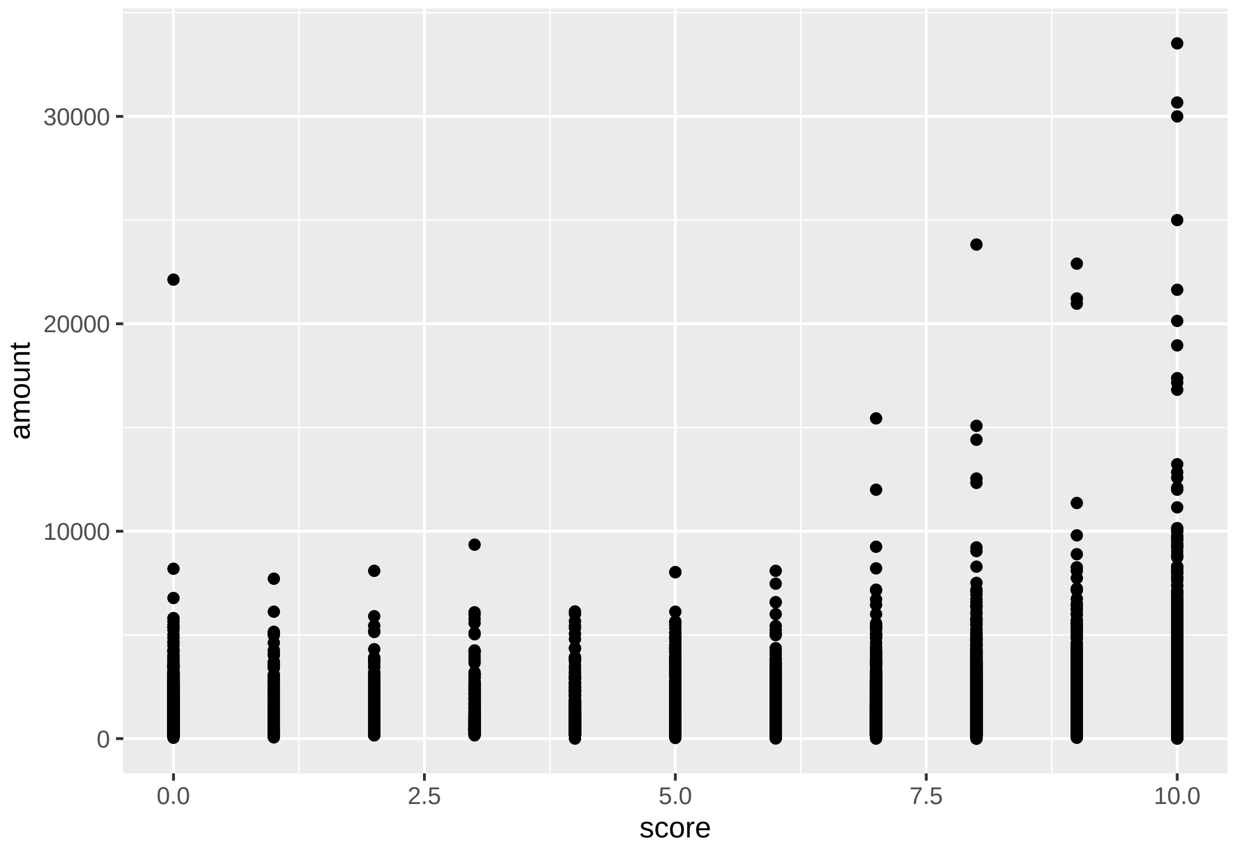
```r
#Histogram

hist(
  hospitality_dt2u$score, breaks = -1:10,
  col = c(rep("red", 7), rep("yellow", 2), rep("green", 2))
)
```



```
#Barplot

barplot(
 prop.table(table(hospitality_dt2u$score)),
 col = c(rep("red", 7), rep("yellow", 2), rep("green", 2))
)
```



```r
ggplot(hospitality_dt2u, aes(x=score, y=amount)) + geom_point()
```



```
#For the unique userID data: separate the genders, find the average amount spent, find average NPS
hospitality_dt2uF <- hospitality_dt2u[hospitality_dt2u$gender == "F"]
View(hospitality_dt2uF)
attach(hospitality_dt2uF)
```

```
## The following objects are masked from hospitality_dt2u:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt1:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id
```

```
nrow(hospitality_dt2uF)
```

```
## [1] 14966
```

```
mean(hospitality_dt2uF$amount)
```

```
## [1] 1149.317
```

```r
# Converting score column to numeric
#
hospitality_dt2uF$score <- as.numeric(as.character(hospitality_dt2uF$score))

# Computing our NPS
nps(hospitality_dt2uF$score)
```

```
## [1] 0.6015635
```

```r
prop.table(table(hospitality_dt2uF$score))
```

```
##
##          0          1          2          3          4          5          6
## 0.03821997 0.01109181 0.01209408 0.01082454 0.01175999 0.02712816 0.02011225
##          7          8          9         10
## 0.04102633 0.09494855 0.09615128 0.63664306
```

```r
hospitality_dt2uM <- hospitality_dt2u[hospitality_dt2u$gender == "M"]
View(hospitality_dt2uM)
attach(hospitality_dt2uM)
```

```
## The following objects are masked from hospitality_dt2uF:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt2u:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt1:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id
```

```r
nrow(hospitality_dt2uM)
```

```
## [1] 14687
```

```r
mean(hospitality_dt2uM$amount)
```

```
## [1] 1128.545
```

```r
# Converting score column to numeric
#
hospitality_dt2uM$score <- as.numeric(as.character(hospitality_dt2uM$score))

# Computing our NPS
nps(hospitality_dt2uM$score)
```

```
## [1] 0.6442432
```

```r
prop.table(table(hospitality_dt2uM$score))
```

```
##
##           0           1           2           3           4           5
## 0.029345680 0.007830054 0.008374753 0.008579015 0.008647103 0.021924151
##           6           7           8           9          10
## 0.019132566 0.044937700 0.103152448 0.096888405 0.651188126
```

```r
#Add a column with the word 'repeat' for repeated user ID and 'non-repeat' for unique user ID

#Data with repeated id only

hospitality_dt1[duplicated(hospitality_dt1$user_id),] #gives you duplicate rows
```

```
##        user_id gender timestamp survey_completion score amount          branch
##    1: 17430789      F   28:02.5          FINISHED     9    570 Nairobi Central
##    2:   328437      F   17:03.2          FINISHED    10   1600   Nairobi South
##    3:   668285      M   36:33.7          TIMEDOUT     9    170   Nairobi South
##    4:   206998      F   32:55.0          FINISHED    10    950   Nairobi North
##    5:   323566      M   08:43.0          TIMEDOUT     9    500 Nairobi Central
##   ---
## 6745:   444277      F   01:03.8          FINISHED    10    200   Nairobi North
## 6746: 17158635      M   30:29.0          FINISHED    10    680    Nairobi West
## 6747:  2246544      F   37:53.3          FINISHED    10    580    Nairobi East
## 6748:  1147687      M   58:04.0          FINISHED     9    300   Nairobi South
## 6749:   314116      M   58:53.3          FINISHED     9    200   Nairobi North
```

```r
hospitality_dt2r <- hospitality_dt1[duplicated(hospitality_dt1$user_id),]
View(hospitality_dt2r)
attach(hospitality_dt2r)
```

```
## The following objects are masked from hospitality_dt2uM:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt2uF:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt2u:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt1:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id
```

```r
nrow(hospitality_dt2r)
```

```
## [1] 6749
```

```r
#Whatever is on the left of the <- sign "gets" whatever is on the right

hospitality_dt2r$repeat_customer<-"repeat"
hospitality_dt2u$repeat_customer<-"non-repeat"


#To join two data frames (datasets) vertically
hospitality_dt1new <- rbind(hospitality_dt2r, hospitality_dt2u)
View(hospitality_dt1new)
attach(hospitality_dt1new)
```

```
## The following objects are masked from hospitality_dt2r:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt2uM:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt2uF:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt2u:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt1:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id

## The following objects are masked from hospitality_dt:
##
##     amount, branch, gender, score, survey_completion, timestamp,
##     user_id
```

```r
nrow(hospitality_dt1new)
```

```
## [1] 36402
```

```r
# Can we build a logistic regression model to predict
# whether a customer will be a repeat customer or not?
#

hospitality_dt1new$repeat_customer <- factor(hospitality_dt1new$repeat_customer,
                                             levels = c("repeat","non-repeat"),
                                      labels = c(0,1))

# Converting repeat_customer column to numeric

hospitality_dt1new$repeat_customer <- as.numeric(as.character(hospitality_dt1new$repeat_customer))
```

```
hospnew.glm = glm(formula=repeat_customer ~ amount + score + gender , data = hospitality_dt1new,
                  family=binomial)
hospnew.glm
```

```
##
## Call:  glm(formula = repeat_customer ~ amount + score + gender, family = binomial,
##     data = hospitality_dt1new)
##
## Coefficients:
## (Intercept)       amount        score      genderM
##   1.8627815    0.0000707   -0.0443532   -0.1300867
##
## Degrees of Freedom: 36401 Total (i.e. Null);  36398 Residual
## Null Deviance:       34910
## Residual Deviance: 34800     AIC: 34800
```

```
summary(hospnew.glm)
```

```
##
## Call:
## glm(formula = repeat_customer ~ amount + score + gender, family = binomial,
##     data = hospitality_dt1new)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3541   0.5695   0.6366   0.6600   0.6976
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.863e+00  6.072e-02  30.679  < 2e-16 ***
## amount       7.070e-05  1.339e-05   5.281 1.29e-07 ***
## score       -4.435e-02  6.224e-03  -7.126 1.03e-12 ***
## genderM     -1.301e-01  2.706e-02  -4.807 1.53e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 34909  on 36401  degrees of freedom
## Residual deviance: 34795  on 36398  degrees of freedom
## AIC: 34803
##
## Number of Fisher Scoring iterations: 4
```

```
#amount spent, score and male gender are significant.

#The logistic regression coefficients give the change in the log odds of the outcome for
#a one unit increase in the predictor variable.

#For a one unit increase in amount spent, the log odds of being a repeat customer increases
#by 0.0000707

#For every one unit change in score, the log odds of repeat (versus non-repeat) decreases
#by -0.0443532
```

```r
#Visiting Stony Hill coffee house being male versus being female changes the log odds of
#being a repeat customer by -0.1300867.



#confidence intervals for the coefficient estimates
## CIs using profiled log-likelihood
confint(hospnew.glm)
```

```
## Waiting for profiling to be done...
```

```
##                     2.5 %        97.5 %
## (Intercept)  1.744776e+00  1.982818e+00
## amount       4.481482e-05  9.727911e-05
## score       -5.665899e-02 -3.225732e-02
## genderM     -1.831450e-01 -7.706445e-02
```

```r
## CIs using standard errors
confint.default(hospnew.glm)
```

```
##                     2.5 %        97.5 %
## (Intercept)  1.743777e+00  1.981786e+00
## amount       4.446038e-05  9.694222e-05
## score       -5.655231e-02 -3.215412e-02
## genderM     -1.831252e-01 -7.704813e-02
```

```r
#We can test for an overall effect of gender using the wald.test function of the aod library.

#The order in which the coefficients are given in the table of coefficients is the same
#as the order of the terms in the model.

#This is important because the wald.test function refers to the coefficients by their order
#in the model. We use the wald.test function. b supplies the coefficients, while Sigma supplies
#the variance covariance matrix of the error terms, finally Terms tells R which terms in the
#model are to be tested, in this case, terms 4.

library(aod)

wald.test(b = coef(hospnew.glm), Sigma = vcov(hospnew.glm), Terms = 4)
```

```
## Wald test:
## ----------
##
## Chi-squared test:
## X2 = 23.1, df = 1, P(> X2) = 1.5e-06
```

```r
#The chi-squared test statistic of 23.1, with one degree of freedom is associated with
#a p-value of 0.0000015 indicating that the overall effect of rank is statistically significant.

#exponentiate the coefficients and interpret them as odds-ratios

## odds ratios only
exp(coef(hospnew.glm))
```

```
## (Intercept)      amount       score     genderM
##   6.4416292   1.0000707   0.9566160   0.8780193
```

```
#To put it all in one table, we use cbind to bind the coefficients and confidence intervals #column-wis

## odds ratios and 95% CI
exp(cbind(OR = coef(hospnew.glm), confint(hospnew.glm)))
```

```
## Waiting for profiling to be done...

##                    OR     2.5 %    97.5 %
## (Intercept) 6.4416292 5.7246174 7.2631852
## amount      1.0000707 1.0000448 1.0000973
## score       0.9566160 0.9449162 0.9682574
## genderM     0.8780193 0.8326474 0.9258302
```

```
#For every one unit increase in amount spent, the odds of being a repeat customer
#(versus non-repeat) increases by a factor of 1.0000707
```