Module 1: Introduction and Essentials

Preparing Data for Modeling

Swathi Dhanasekaran

04/03/2020

An organization has collected data on customer visits, transactions, operating system, and gender and desires to build a model to predict revenue. For the moment, the goal is to prepare the data for modeling. Analyze the data set in the following manner:

```
library(data.table)
library(ggplot2)
library(dplyr)
```

1) Download this data set and then upload the data. Each row represents a customer's interactions with the organization's web store. The first column is the number of visits of a customer, the second the number of transactions of that customer, the third column is the customer's operating system, and the fourth column is the customer's reported gender, while the last column is revenue, i.e., the total amount spent by that customer.

```
mydat <- fread('https://da5030.weebly.com/uploads/8/6/5/9/8659576/cu
stomertxndata.csv')
str(mydat)
```

```
## Classes 'data.table' and 'data.frame':   22800 obs. of  5 variabl
es:
##  $ V1: int  7 20 22 24 1 13 23 14 11 24 ...
##  $ V2: int  0 1 1 2 0 1 2 1 1 2 ...
##  $ V3: chr  "Android" "iOS" "iOS" "iOS" ...
##  $ V4: chr  "Male" NA "Female" "Female" ...
##  $ V5: num  0 577 850 1050 0 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
colnames(mydat) <- c("nVisits","nTrans","OS","Gender","Revenue")
```

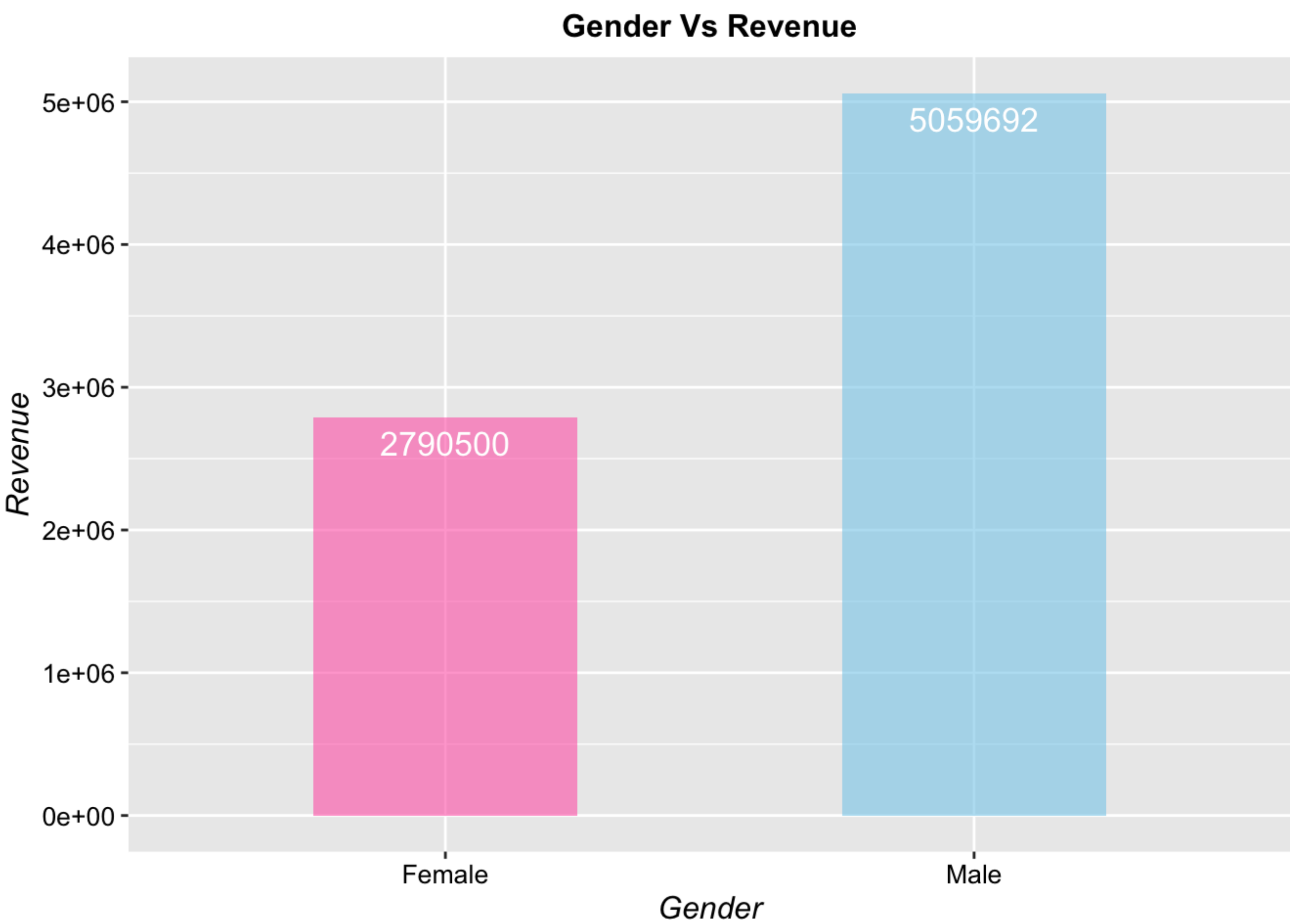
2) Calculate the following summative statistics: total transaction amount (revenue), mean number of visits, median revenue, standard deviation of revenue, most common gender. Exclude any cases where there is a missing value.

```
cat("Total Transacation :", sum(mydat$Revenue), "\nMean number of vi
sits :", round(mean(mydat$nVisits)), "\nMedian revenue :",median(myd
at$Revenue), "\nSD of revenue :",sd(mydat$Revenue), "\nMost Common G
ender :", names(sort(table(mydat$Gender), decreasing = TRUE)[1]))
```

```
## Total Transacation : 10372524
## Mean number of visits : 12
## Median revenue : 344.6516
## SD of revenue : 425.9884
## Most Common Gender : Male
```

3) Create a bar/column chart of gender (x-axis) versus revenue (y-axis). Omit missing values, i.e., where gender is NA

```
gen_rev <- aggregate(Revenue ~ Gender, mydat[!is.na(mydat$Gender),],
sum)
ggplot(data = as.data.frame(gen_rev), aes(x=Gender, y=Revenue)) +
  geom_bar(stat = "identity", width=0.5, fill = c("hotpink","skyblu
e"), alpha = 0.6) +
  geom_text(aes(label=round(Revenue)), vjust=1.6, color="white", siz
e=4.5) +
  ggtitle("Gender Vs Revenue") +
  theme(plot.title = element_text(color="black", size=12, face="bol
d",hjust = 0.5),
        axis.title.x = element_text(color="black", size=12, face="it
alic"),
        axis.title.y = element_text(color="black", size=12, face="it
alic"),
        axis.text.x = element_text(color="black", size=10, angle=0),
        axis.text.y = element_text(color="black", size=10, angle=0))
```



4) What is the Pearson Moment of Correlation between number of visits and revenue? Comment on the correlation.

```
cat("Pearson Correlation Coffecient between visit and revenue :", round(cor(mydat$nVisits, mydat$Revenue, method = "pearson"), 2))
```

```
## Pearson Correlation Coffecient between visit and revenue : 0.74
```

There is a fairly strong positively-correlated linear relationship between the number of visits and the revenue generated, i.e, they are directly proportional.

5) Which columns have missing data? How did you recognize them? How would you impute missing values?

```
#Function to return a logical value of TRUE if missing values in column
is_missing <- function(x) {
  n <- sum(is.na(x))
  ret <- n!=0
  return(ret)
}
cat("\nColumn(s) with missing values :", paste(shQuote(names(which(apply(mydat, 2, is_missing)==TRUE)), type="sh"), collapse=",", sep=","))
```

```
##
## Column(s) with missing values : 'nTrans','Gender'
```

```
#Mean of number of transactions after removing NA
imputeVal_nTrans <- round(mean(mydat$nTrans[!is.na(mydat$nTrans)]))
cat("\nImputed value for number of transactions :",imputeVal_nTrans)
```

```
##
## Imputed value for number of transactions : 1
```

```
#Function to calculate mode
Mode <- function(x) {
      uniq_x <- sort(unique(x))
      uniq_x[which.max(tabulate(match(x, uniq_x)))]
}

#Mode of the char vector after removing NA
imputeVal_Gender <- Mode(mydat$Gender[!is.na(mydat$Gender)])
cat("\nImputed value for Gender :", imputeVal_Gender)
```

```
##
## Imputed value for Gender : Male
```

6) Impute missing transaction and gender values. Use the mean for transaction (rounded to the nearest whole number) and the mode for gender.

```
imputed_mydat <- mydat %>%
  mutate(Gender = replace(Gender, is.na(Gender), imputeVal_Gender),
         nTrans = replace(nTrans, is.na(nTrans), imputeVal_nTrans))

cat("Number of columns with missing NA in imputed dataframe :",sum(a
pply(imputed_mydat, 2, is_missing)))
```

```
## Number of columns with missing NA in imputed dataframe : 0
```

7) Split the data set into two equally sized data sets where one can be used for training a model and the other for validation. Take every odd numbered case and add them to the training data set and every even numbered case and add them to the validation data set, i.e., row 1, 3, 5, 7, etc. are training data while rows 2, 4, 6, etc. are validation data.

```
#Odd rows makeup training dataset
training <- imputed_mydat[seq(1,nrow(imputed_mydat),2),]

#Even rows makeup validation dataset
validation <- imputed_mydat[seq(2,nrow(imputed_mydat),2),]
```

8) Calculate the mean revenue for the training and the validation data sets and compare them. Comment on the difference.

```
cat("***Mean Revenue***\nTraining Dataset :",mean(training$Revenue),
"\nValidation Dataset :", mean(validation$Revenue))
```

```
## ***Mean Revenue***
## Training Dataset : 449.6105
## Validation Dataset : 460.26
```

```
diff <- (mean(validation$Revenue)-mean(training$Revenue))/mean(valid
ation$Revenue)
cat("\nMean revenue of validation dataset is",round(diff*100,2),"% h
igher than that in training dataset.")
```

```
##
## Mean revenue of validation dataset is 2.31 % higher than that in
training dataset.
```

9) For many data mining and machine learning tasks, there are packages in R. Use the sample() function to split the data set, so that 60% is used for training and 20% is used for testing, and another 20% is used for validation. To ensure that your code is reproducible and that everyone gets the same result, use the number 77654 as your seed for the random number generator.

```r
set.seed(77654)
spec = c(train = .6, test = .2, validate = .2)

#So we cut the rows at breaks defined by the spec proportions, and l
abel them as a group vector
#Sample() by default has replace=TRUE and simply reorders the labels
in the vector
groups = sample(cut(
  x=seq(nrow(imputed_mydat)),
  breaks=nrow(imputed_mydat)*cumsum(c(0,spec)), #defines breaks base
d on cumulative sums of c(0,spec)
  labels = names(spec)
))

#We split the data based on the defined groups label vector into a l
ist of 3 dfs
datasets = split(imputed_mydat, groups)

#We can confirm the proportion of split as follows:
sapply(datasets, nrow)/nrow(imputed_mydat)
```

```
##    train     test validate
##      0.6      0.2      0.2
```