DA5030_PeerReview_1_Minxin Cheng_09/19

00 Load packages

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

0 Read in data file

```
# read in data file
txnData <- read.csv("customertxndata.csv")

# rename columns
colnames(txnData) <- c("numVisit",
                       "trans",
                       "opSystem",
                       "gender",
                       "revenue")
```

1. Calculate: a. total transaction amount, mean number of visit, median revenue, standard deviation of revenue, most common gender (exclude any cases where there is a missing value)

```
# calculate total transaction, mean number of visit, median revenue, and standard deviation of revenue
sumTxnData <- txnData %>%
  summarise(totalTransaction = sum(revenue, na.rm = TRUE),
            meanNumVisit = mean(numVisit, na.rm = TRUE),
            medianRevenue = median(revenue, na.rm = TRUE),
            sdRevenue = sd(revenue, na.rm = TRUE))
sumTxnData
```

```
##   totalTransaction meanNumVisit medianRevenue sdRevenue
## 1         10372524     12.48673      344.6516  425.9871
```

```r
# check the number of each gender
sumTxnDataGender <- txnData %>%
  count(gender)
sumTxnDataGender
```

```
##   gender     n
## 1 Female  2670
## 2   Male 14729
## 3   <NA>  5400
```

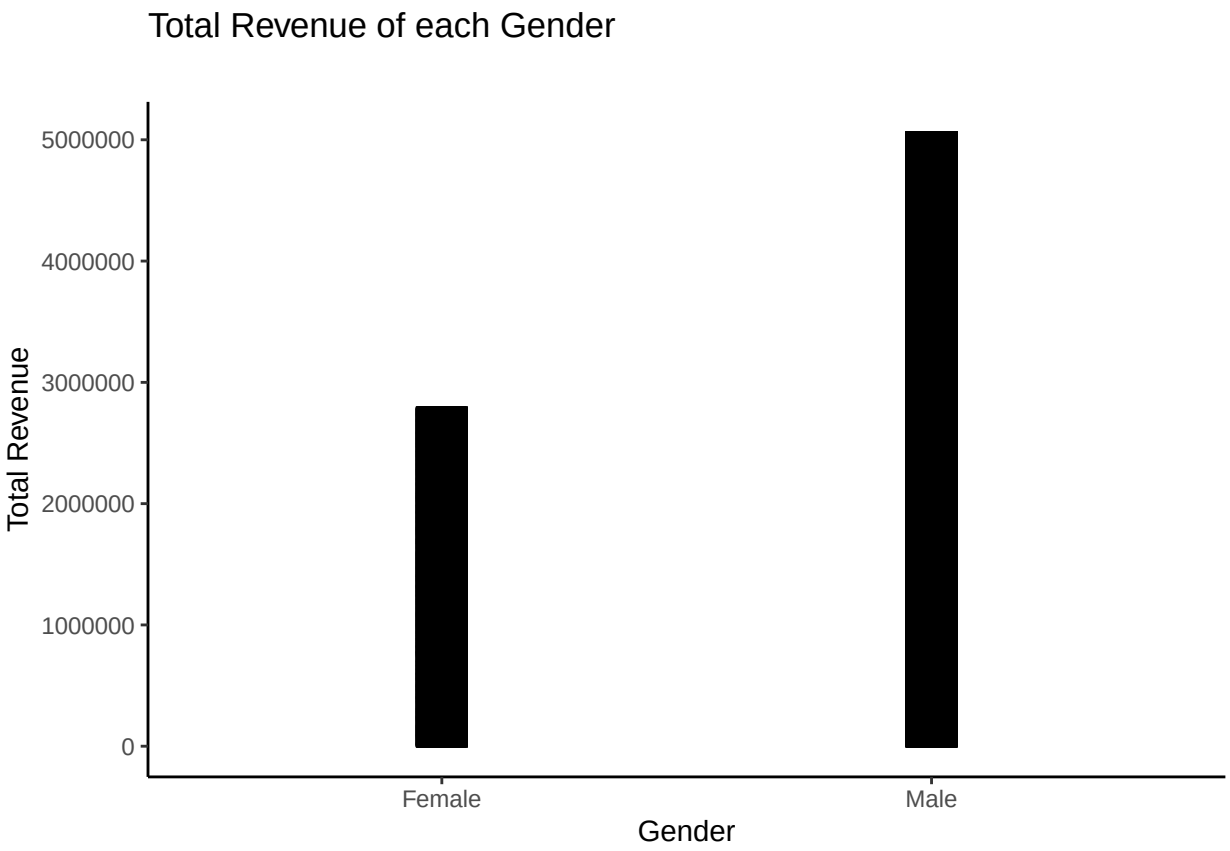
As the summary table above, male is the most common gender

2. Create a bar/column chart of gender (x-axis) versus revenue (y-axis). Omit missing values

```r
# remove missing values in gender and revenue columns
genderRevenue <- txnData %>%
  filter(!is.na(gender),
         !is.na(revenue))

# plot the bar graph
genderRevenueBar <- ggplot(genderRevenue, aes(gender, revenue)) +
  geom_bar(stat = "identity", width = 0.1, color = "black")

# add title, labels, and format the y axis
genderRevenueBar +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE)) +
  labs(title = "Total Revenue of each Gender",
       subtitle = "",
       x = "Gender",
       y = "Total Revenue") +
  theme_classic()
```



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

```
# run the Peason correlation test
pearsonTxnData <- cor.test(txnData$numVisit,
                           txnData$revenue,
                           method = "pearson")
pearsonTxnData
```

```
##
##  Pearson's product-moment correlation
##
## data:  txnData$numVisit and txnData$revenue
## t = 165.54, df = 22797, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.7328857 0.7446763
## sample estimates:
##       cor
## 0.7388375
```

As the test result, p value is lest than 0.05 meaning the number of visits and revenue are significantly correlated.

4.Which columns have missing data? how did you recognize them? how would you impute missing values?

```r
# generate a table of number of missing values in each column if any
checkNA <- txnData %>%
  summarise(across(where(~any(is.na(.))),
                   ~sum(is.na(.))))
checkNA
```

```
##   trans gender
## 1  1800   5400
```

As the summary table, gender and revenue columns have missing value (1800 and 5400 each). the missing values can be imputed 1) by calculating the mean/median/mode of the exist values, 2) from new individuals that were not included in the sample, 3) choose value from other individuals that have similar values on other variables, 4) regressing the missing variable on other variables

5. Impute missing transaction and gender values, use the mean for transaction (rounded to the nearest whole number) and the mode for gender

```r
# impute missing transaction values using mean
txnData <- txnData %>%
  mutate(numVisit = ifelse(is.na(numVisit),
                           ave(numVisit,
                               FUN = function(x) mean(x, na.rm = TRUE)),
                           numVisit))

# create a function to get the mode of gender
getMode <- function(v) {
  uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}
genderImpute <- getMode(txnData$gender)

# impute missing gender using mode
txnData <- txnData %>%
  mutate(gender = ifelse(is.na(gender),
                         genderImpute,
                         gender))
```

6. 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

```r
# select odd rows as training data
trainingTxnData <- txnData %>%
  filter(row_number() %% 2 == 1)

# select even rows as validation data
validTxnData <- txnData %>%
  filter(row_number() %% 2 == 0)
```

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

```
trainingMeanRevenue <- mean(trainingTxnData$revenue)
validMeanRevenue <- mean(validTxnData$revenue)

trainingMeanRevenue
```

```
## [1] 460.26
```

```
validMeanRevenue
```

```
## [1] 449.6499
```

8. 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 smae result, use the number 77654 as your seed for the random number generator. Use the code fragment below for reference

```
# set the seed
set.seed(77654)

# select the row numbers of training data, testing data, and validation data
trainingSample <- sample.int(n = nrow(txnData),
                             size = floor(0.6 * nrow(txnData)),
                             replace = FALSE)
testingSample <- sample.int(n = nrow(txnData),
                            size = floor(0.2 * nrow(txnData)),
                            replace = FALSE)
validSample <- sample.int(n = nrow(txnData),
                          size = floor(0.2 * nrow(txnData)),
                          replace = FALSE)

# split the dataset using the row numbers
trainingData <- txnData[trainingSample, ]
testingData  <- txnData[testingSample, ]
validData <- txnData[validSample, ]

# calculate mean revenue
mean(trainingData$revenue)
```

```
## [1] 455.575
```

```
mean(testingData$revenue)
```

```
## [1] 451.8467
```