Thadryan Sweeney

Assignment 1

DA 5030 - Intro to Data Mining and Machine Learning

Fall 2017

Load data into R

(5 pts) Locate the data set and load the data into R.

In [205]:

```
# read our data in
data = read.csv('customertxndata.csv')

# take a look using head()
head(data)

# look at the structure of data
str(data)
```

Visits	Transactions	OS	Gender	Revenue
7	0	Android	Male	0.0000
20	1	iOS	NA	576.8668
22	1	iOS	Female	850.0000
24	2	iOS	Female	1050.0000
1	0	Android	Male	0.0000
13	1	Android	Male	460.0000

```
'data.frame': 22800 obs. of  5 variables:
 $ Visits      : int  7 20 22 24 1 13 23 14 11 24 ...
 $ Transactions: int  0 1 1 2 0 1 2 1 1 2 ...
 $ OS          : Factor w/ 2 levels "Android","iOS": 1 2 2 2 1 1 2 1 1 2 ..
.
 $ Gender      : Factor w/ 2 levels "Female","Male": 2 NA 1 1 2 2 2 2 2 2 .
..
 $ Revenue     : num  0 577 850 1050 0 ...
```

Summative Statistics

(10 pts) Calculate the following summative statistics: total number of cases, mean number of visits, median revenue, maximum and minimum number of transactions, most commonly used operating system. Exclude any cases where there is a missing value.

In [200]:

```
str(data)
str(complete_data)
```

```
'data.frame': 22800 obs. of  5 variables:
 $ Visits      : int  7 20 22 24 1 13 23 14 11 24 ...
 $ Transactions: int  0 1 1 2 0 1 2 1 1 2 ...
 $ OS          : Factor w/ 2 levels "Android","iOS": 1 2 2 2 1 1 2 1 1 2 ..
.
 $ Gender      : Factor w/ 2 levels "Female","Male": 2 NA 1 1 2 2 2 2 2 2 .
..
 $ Revenue     : num  0 577 850 1050 0 ...
'data.frame': 15600 obs. of  5 variables:
 $ Visits      : int  7 22 24 1 13 23 14 11 24 17 ...
 $ Transactions: int  0 1 2 0 1 2 1 1 2 1 ...
 $ OS          : Factor w/ 2 levels "Android","iOS": 1 2 2 1 1 2 1 1 2 1 ..
.
 $ Gender      : Factor w/ 2 levels "Female","Male": 2 1 1 2 2 2 2 2 2 2 ..
.
 $ Revenue     : num  0 850 1050 0 460 1850 480 110 1950 225 ...
 - attr(*, "na.action")=Class 'omit'  Named int [1:7200] 2 12 14 17 18 24 2
5 26 27 28 ...
  .. ..- attr(*, "names")= chr [1:7200] "2" "12" "14" "17" ...
```

In [101]:

```
# total number of entries is the number of rows
total_entries <- nrow(data)
total_entries

print("Entries with all data")
# na.omit removes all rows containing NA
complete_data <- na.omit(data)
nrow(complete_data)

print("Mean visits:")
# mean() is a built in function
mean_visits <- mean(complete_data$Visits)
mean_visits

print("Median revenue:")
# median() is a built in fuction
median_revenue <- median(complete_data$Revenue)
median_revenue

print("Max transactions")
max(complete_data$Transactions)
print("Min transactions")
min(complete_data$Transactions)

print("Most Common OS:")
# which will create subsets based specified criteria
android_users <- which(complete_data$OS == "Android")
iOS_users <- which(complete_data$OS == "iOS")
print("Android more common than IOs")
length(android_users) > length(iOS_users)
```

22800

```
[1] "Entries with all data"
```

15600

```
[1] "Mean visits:"
```

12.5926282051282

```
[1] "Median revenue:"
```

360

```
[1] "Most Common OS:"
[1] "Android more common than IOs"
```

TRUE

```
[1] "Max transactions"
```

2

```
[1] "Min transactions"
```

0

Which columns have missing data?

(10 pts) Which columns have missing data? How did you recognize them? How would you impute missing values?

I recognized them by finding rows where there is more than 0 NA. I will explain imputation as I go.

In [199]:

```
# make a frame of rows where the NA data is greater than one and inspect it
rows_missing_data <- data[rowSums(is.na(data)) > 0,]
head(rows_missing_data)
```

	Visits	Transactions	OS	Gender	Revenue
2	20	1	iOS	NA	576.8668
12	14	1	Android	NA	344.6516
14	8	1	Android	NA	344.6516
17	18	2	Android	NA	990.3062
18	16	NA	Android	Male	405.2441
24	3	NA	Android	Male	121.7745

NOTE:

I am not doing all the steps in order from here on out because I want to do the imputations before I make my graphs but all steps in the rubric will be explicitly completed

Imputation on gender data

Imputation of gender data

(15 pts) Impute missing transaction and gender values.

We will look at our subset of daa with no missing values to see if we can draw any useful conclusions for imputation of missing data

In [104]:

```
male_users <- which(complete_data $Gender == "Male")
length(male_users)

female_users <- which(complete_data $Gender == "Female")
length(female_users)
```

12930

2670

The data is so male skewed that we will replace uknown with male

For the record I think this is probably not the most nuanced idea and in my python prototype I simply wrote a function that imputed male/female to missing data at the rate at which they occur in the subset of the data with no missing parts. However, I am a n00b at R and probably won't be able to cook something like that up in the course of this assignment.

In [134]:

```
# execute the imputation and see if it worked
# where data in column 'Gender' is NA, set to male
data$Gender[is.na(data$Gender)] = 'Male'
head(data)
```

Visits	Transactions	OS	Gender	Revenue
7	0	Android	Male	0.0000
20	1	iOS	Male	576.8668
22	1	iOS	Female	850.0000
24	2	iOS	Female	1050.0000
1	0	Android	Male	0.0000
13	1	Android	Male	460.0000

Imputation of transaction data

There are no outliers in this column, so we can use a simple substitution of median/mean, etc. It's worth noting that this has to be a whole number, so we are unlikely to be able to use the mean. Let's try median instead.

In [106]:

```
print("Median transactions")
median_transactions <- median(complete_data$Transactions)
median_transactions
```

```
[1] "Median transactions"
```

1

We've found that the median in our complete data is 1, so we will go ahead and insert it where we have NA.

In [107]:

```
# execute the imputation and see if it worked
# "where data colums "Transactioins" is NA, set 1
data $Transactions[is.na(data$Transaction)] = 1

# there are no NAs is the first few dozen so we can verify another way
transactions_with_NA <- which(is.na(data$Transactions))
length(transactions_with_NA)
```

0

Splitting the data into two sets

(20 pts) 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.

In [108]:

```
# shortcut from prof S. to split the data
# create a sequence using seq() that is odd
odds <- seq(from = 1, to = nrow(data), by = 2)

# R will create new dataFrame based on indexes of number in sequence...
training_data = data[odds,]

# ...and what are evens but negative odds? Whoa, man.
validation_data = data[-odds,]

# take a peek
head(training_data)
head(validation_data)

# check size, logical for QC
nrow(training_data)
nrow(validation_data)

nrow(training_data) == nrow(validation_data)
```

	Visits	Transactions	OS	Gender	Revenue
1	7	0	Android	Male	0
3	22	1	iOS	Female	850
5	1	0	Android	Male	0
7	23	2	iOS	Male	1850
9	11	1	Android	Male	110

	Visits	Transactions	OS	Gender	Revenue
9	11	1	Android	Male	110
11	17	1	Android	Male	225

	Visits	Transactions	OS	Gender	Revenue
2	20	1	iOS	Male	576.8668
4	24	2	iOS	Female	1050.0000
6	13	1	Android	Male	460.0000
8	14	1	Android	Male	480.0000
10	24	2	iOS	Male	1950.0000
12	14	1	Android	Male	344.6516

11400

11400

TRUE

Calculate the mean revenue of the sets

(10 pts) Calculate the mean revenue for each data set and compare them. Comment on the difference.

In [109]:

```
mean_training_revenue <- mean(training_data$Revenue)
mean_validation_revenue <- mean(validation_data$Revenue)

mean_training_revenue
mean_validation_revenue
```

449.610487796957

460.260014297842

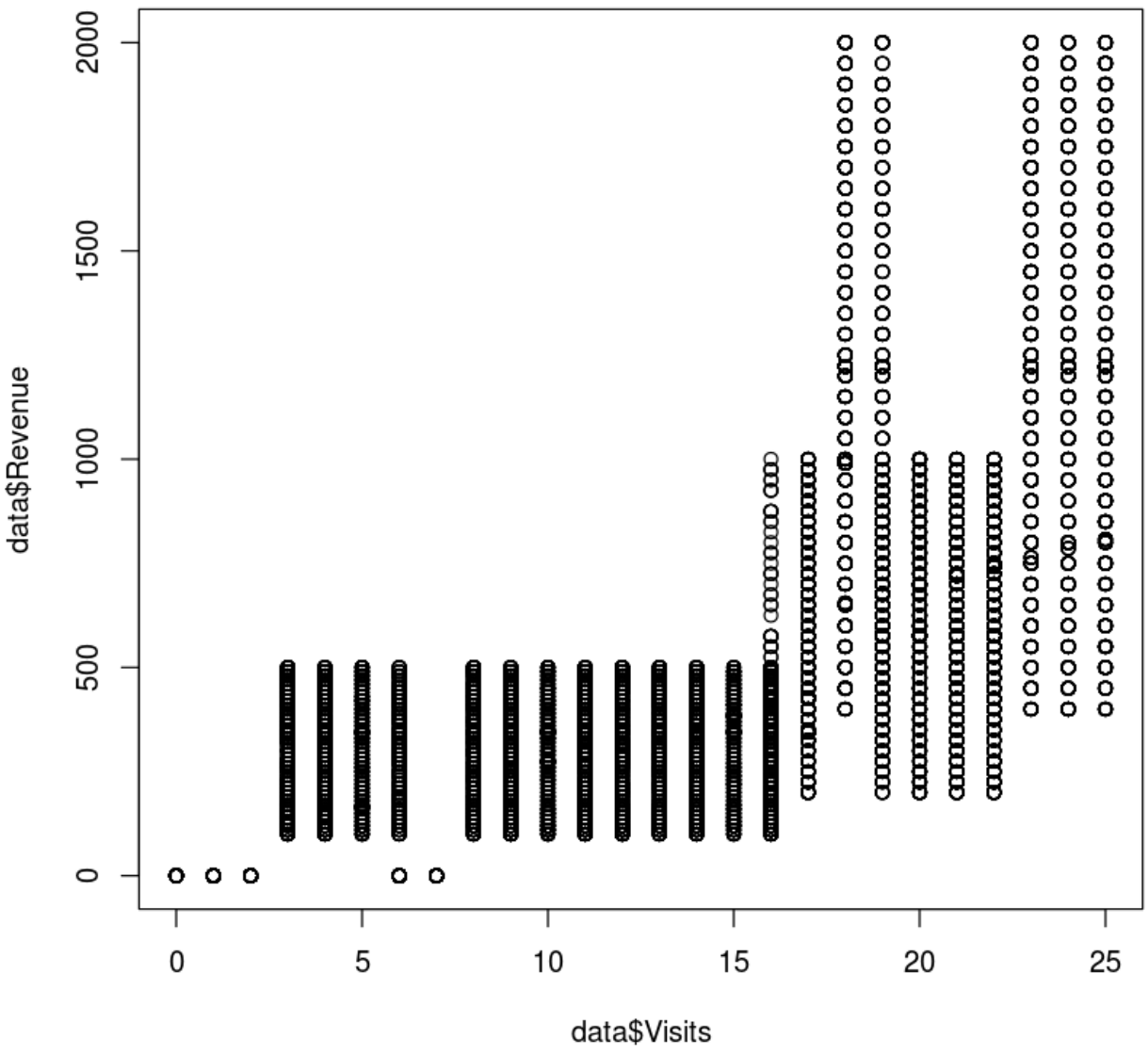
The difference is pretty small in these. I assume this is good, as it shows we aren't traning to an outlier or anything like that. It also suggests our dataset contains a fairly well randomized entries (ie, we didnt see that one revenue is much larger than the other and find out the entries were sorted by revenue). However, I suspect that this will not be able to inform us about overfitting, given that the model will be operating on something so close to it's original set.

Create a scatterplot

(15 pts) Create a scatterplot of number of visits (x-axis) versus revenue (y-axis). Comment on the correlation between the two variables.

In [110]:

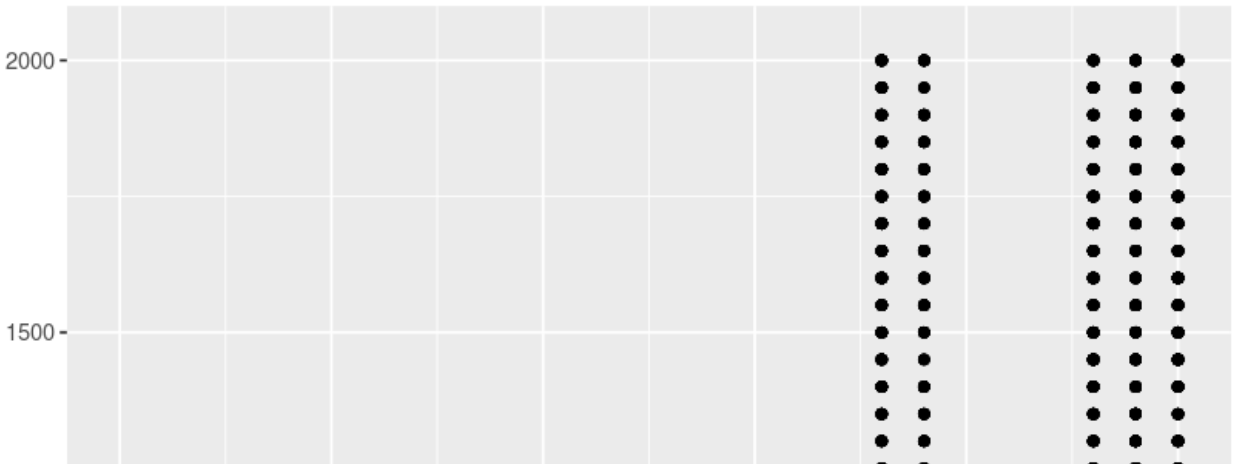
```
# super handy plot function
plot(data$Visits, data$Revenue)
```

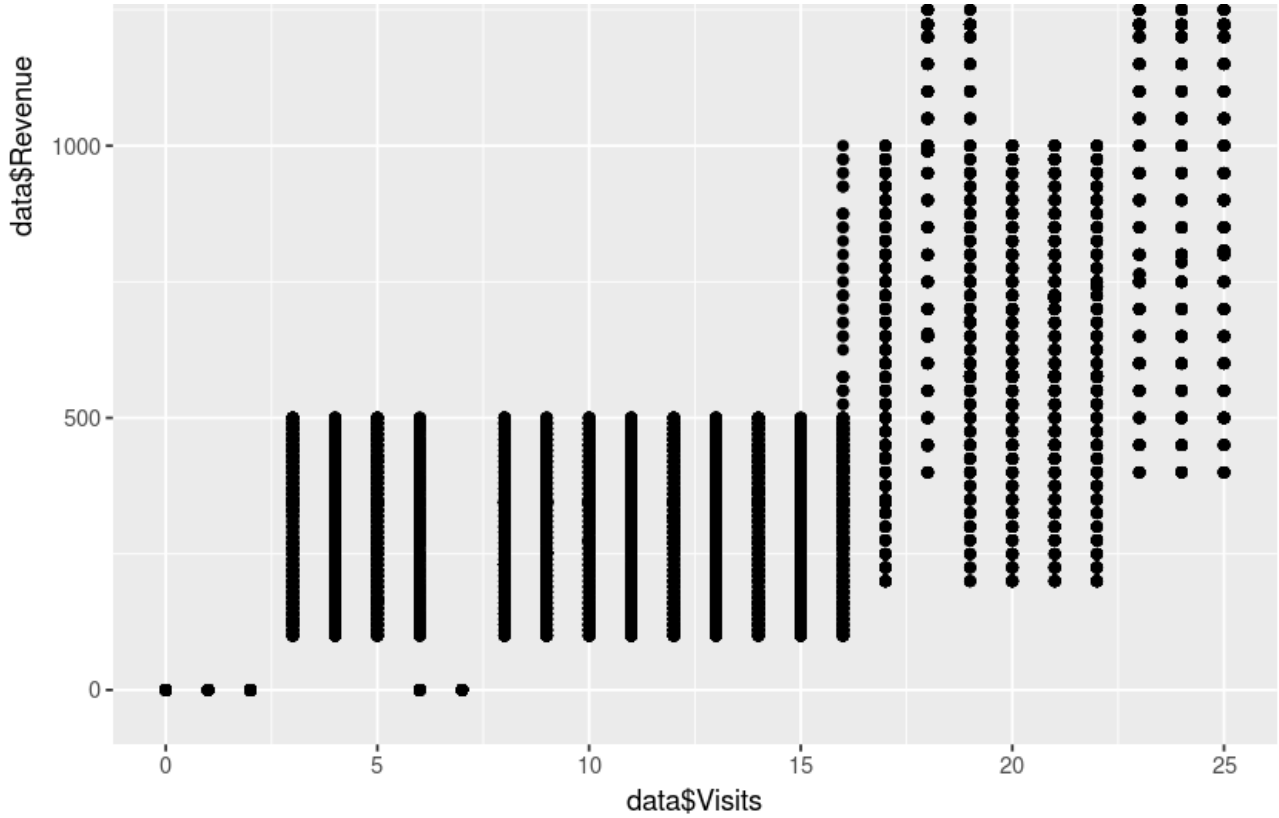


Unsuprisingly, it appears that the more visits we recieve from somone, the greater amount of money we get

In [203]:

```
# same thing done in qplot() for the lulz
qplot(data$Visits, data$Revenue)
```





R package for training/validating

(15 pts) For many data mining and machine learning tasks, there are packages in R. Find at least one package that has functions for creating training and validation data subsets and show how to use them.

The caret package is specifically for parsing data to use for training and validation. It has a function called createDataPartion() wich takes part of our data frame, a percent value (as a decimal) and some formatiing arguments. It selects which entries at random, which can help prevent unnoticed biases

In [135]:

```
# import the package
library(caret)

# the y argument takes part of a dataframe, the p takes the percent of data
to use,
# and we can designate we do or dont want to use a list
trainingInput <- createDataPartition(y = data$Gender, p = .50, list = FALSE
)

# training is data with the partition applied
training <- data[ trainingInput,]

# validation is the remainder or opposite
validation <- data[ -trainingInput,]
```

Inspect our work:

In [137]:

```
#confirm it worked
```

```
#confirm it worked
print("Training")
nrow(training)
str(training)

print("Validation")
nrow(validation)
str(validation)
```

```
[1] "Training"
```

11400

```
'data.frame': 11400 obs. of  5 variables:
 $ Visits      : int  7 20 22 24 23 11 8 24 18 25 ...
 $ Transactions: int  0 1 1 2 2 1 1 2 2 2 ...
 $ OS          : Factor w/ 2 levels "Android","iOS": 1 2 2 2 2 1 1 2 1 2 ..
.
 $ Gender      : Factor w/ 2 levels "Female","Male": 2 2 1 1 2 2 2 2 2 2 ..
.
 $ Revenue     : num  0 577 850 1050 1850 ...
[1] "Validation"
```

11400

```
'data.frame': 11400 obs. of  5 variables:
 $ Visits      : int  1 13 14 24 17 14 2 23 16 17 ...
 $ Transactions: int  0 1 1 2 1 1 0 2 NA 1 ...
 $ OS          : Factor w/ 2 levels "Android","iOS": 1 1 1 2 1 1 1 2 1 1 ..
.
 $ Gender      : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 1 2 2 ..
.
 $ Revenue     : num  0 460 480 1950 225 ...
```

We can se the sets are of the same size