Lab02 name: Yixuan Li; UPI:Yil845

06/08/2021

#The purpose of this lab is to get experience accessing data from the web in different formats.

Acquiring data from website as format of csv, html and json

1, Download csv file from State highway traffic monitoring site and save it as ‘traffic-monitoring-sites.csv’.

```
download.file("https://opendata.arcgis.com/api/v3/datasets/b90f8908910f44a493c6501c3565ed2d_0/do
wnloads/data?format=csv&spatialRefId=2193","traffic-monitoring-sites.csv")
```

2, Download html page from TMS daily traffic counts API page and Save the file as traffic-daily-counts.html.

```
library(httr)
library(xml2)
install.packages("rvest")
library(rvest)

download.file("https://services.arcgis.com/CXBb7LAjgIIdcsPt/arcgis/rest/services/TMS_Telemetry_S
ites/FeatureServer/0/query?outFields=*&where=1%3D1","traffic-monitoring-sites.html")
```

3, Download JSON page from TMS daily traffic counts API page and Save the file as traffic-daily-counts.json.

```
download.file("https://services.arcgis.com/CXBb7LAjgIIdcsPt/arcgis/rest/services/TMS_Telemetry_S
ites/FeatureServer/0/query?where=1%3D1&objectIds=&time=&resultType=none&outFields=*&returnIdsOnl
y=false&returnUniqueIdsOnly=false&returnCountOnly=false&returnDistinctValues=false&cacheHint=fal
se&orderByFields=&groupByFieldsForStatistics=&outStatistics=&having=&resultOffset=&resultRecordC
ount=&sqlFormat=none&f=pjson&token=", "traffic-daily-counts.json")
```

Import data into R dataframe

4, read csv file into a R dataframe

```
#read.csv("traffic-monitoring-sites.csv")
head(read.csv("traffic-monitoring-sites.csv"))
```

```
##      ï..X       Y OBJECTID SH RS    RP  siteRef lane           type
## 1 1687583 6092785        1 10  0  7.38 01000007 Both Non-Continuous
## 2 1686763 6095528        2 10  7  3.01 01000011 Both Non-Continuous
## 3 1683980 6099669        3 10  7  8.23 01000015 Both Non-Continuous
## 4 1680411 6110299        4 10 17 11.14 01000029 Both     Continuous
## 5 1658993 6124062        5 10 48 11.93 01000060 Both Non-Continuous
## 6 1643685 6127916        6 10 79  2.20 01000076 Both Non-Continuous
##   percentHeavy equipmentCurrent
## 1          7.4        Dual Loop
## 2          6.6        Dual Loop
## 3          6.9        Dual Loop
## 4          8.4        Dual Loop
## 5         10.2        Dual Loop
## 6          6.4        Dual Loop
##                                                        description
## 1                                        Sth of Puketona Rd (SH11)
## 2                    Nth of Wakelin Rd (about 3.5km north of SH11)
## 3                      Springbank Road 1km south of Waimate Nth Rd
## 4                                  About 1km south of Takou Bay Rd
## 5                               About 1.1 km north of Salvation Rd
## 6 By Doubtless Bay Croquet Club 50-100m W of Stratford Dr Cable Bay
##           region acceptedDays AADT5yearsAgo AADT4yearsAgo AADT3yearsAgo
## 1 01 - Northland           28          4566          4988          5401
## 2 01 - Northland           35          8603          9415          9737
## 3 01 - Northland           42          6691          7275          7201
## 4 01 - Northland          126          4127          4466          4704
## 5 01 - Northland           74          2112          2402          2458
## 6 01 - Northland           42          4025          4412          4568
##   AADT2yearsAgo AADT1yearAgo                siteType
## 1          5501         4954 Regional Non-Continuous
## 2          9952         9001 Regional Non-Continuous
## 3          7918         7419 Regional Non-Continuous
## 4          4934         4853     Regional Continuous
## 5          2621         2913 Regional Non-Continuous
## 6          4668         4668 Regional Non-Continuous
```

5, Below R codes extract data from html file and save it as a R dataframe.

```
html <- read_html("traffic-monitoring-sites.html")

extractVar <- function(xpath) {
  span <- xml_find_all(html, xpath)
  text <- xml_text(span)
  gsub("^ +| +$", "", text)
}

site<- extractVar("/html/body/div/table/tr[5]/td[2]")
class <- extractVar("/html/body/div/table/tr[6]/td[2]")
count <- extractVar("/html/body/div/table/tr[10]/td[2]")
trafficcount<-data.frame(site,class,count)

head(trafficcount)
dim(trafficcount)
```

6, Underlying codes Read the data from the JSON file into an R data frame

```
#install.packages(jsonlite)
library(jsonlite)
```

```
## Warning: package 'jsonlite' was built under R version 4.0.5
```

```
geoJson <- fromJSON(readLines("traffic-daily-counts.json"))
```

```
## Warning in readLines("traffic-daily-counts.json"): incomplete final line found
## on 'traffic-daily-counts.json'
```

```
OBJECTID<-geoJson[[6]]$attributes$OBJECTID
startDate <-geoJson[[6]]$attributes$startDate
siteID<-geoJson[[6]]$attributes$siteID
regionName<-geoJson[[6]]$attributes$regionName
SiteRef <- geoJson[[6]]$attributes$SiteRef
classWeight <-geoJson[[6]]$attributes$classWeight
siteDescription<-geoJson[[6]]$attributes$siteDescription
laneNumber<-geoJson[[6]]$attributes$laneNumber
flowDirection<-geoJson[[6]]$attributes$flowDirection
trafficCount<-geoJson[[6]]$attributes$trafficCount
geoJson <-data.frame(OBJECTID, startDate, siteID,regionName, SiteRef, classWeight,siteDescriptio
n,laneNumber,flowDirection,trafficCount)
head(geoJson)
```

```
##   OBJECTID    startDate siteID      regionName  SiteRef classWeight
## 1        1 1.514765e+12    916 11 - Canterbury 07700006       Light
## 2        2 1.514765e+12    916 11 - Canterbury 07700006       Light
## 3        3 1.514765e+12   2595  14 - Southland 00621171       Light
## 4        4 1.514765e+12   2595  14 - Southland 00621171       Light
## 5        5 1.514765e+12     57 06 - Hawkes Bay 05100015       Light
## 6        6 1.514765e+12     57 06 - Hawkes Bay 05100015       Light
##                    siteDescription laneNumber flowDirection trafficCount
## 1 Ashburton - Nth of Racecourse Rd          2             2          504
## 2 Ashburton - Nth of Racecourse Rd          1             1          572
## 3        Btwn Vogel & Durham - Dec          3             2         2189
## 4        Btwn Vogel & Durham - Dec          4             2         2746
## 5   SH 51 Junction with Farndon Rd          1             1         6083
## 6   SH 51 Junction with Farndon Rd          2             2         5427
```

```
dim(geoJson)
```

```
## [1] 2000   10
```

```
class(geoJson)
```

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

7, Explore data from json dataframe

How many different days are in the data set? two

```
unique(geoJson$startDate)
```

```
## [1] 1.514765e+12 1.546301e+12
```

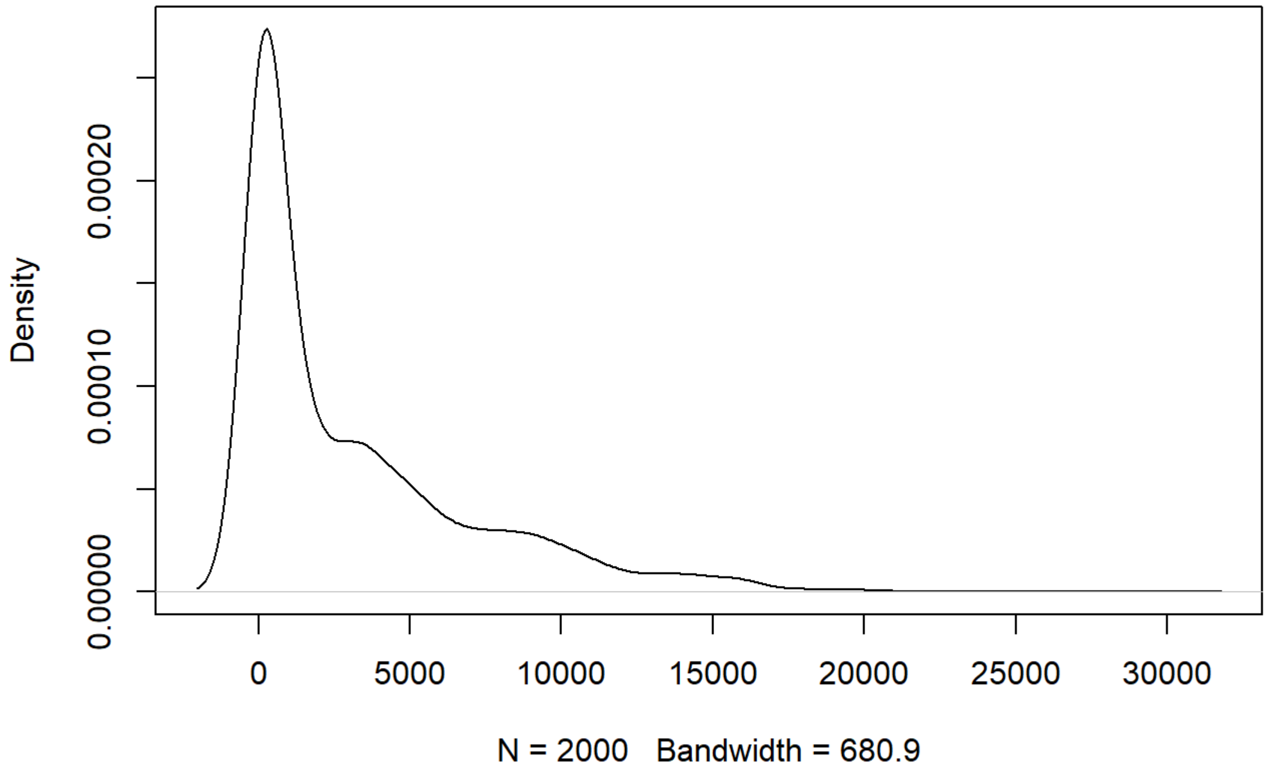
How many different sites?

```
length(unique(geoJson$siteID))
```

```
## [1] 509
```

How are the counts distributed? It is a positive skewed distribution.

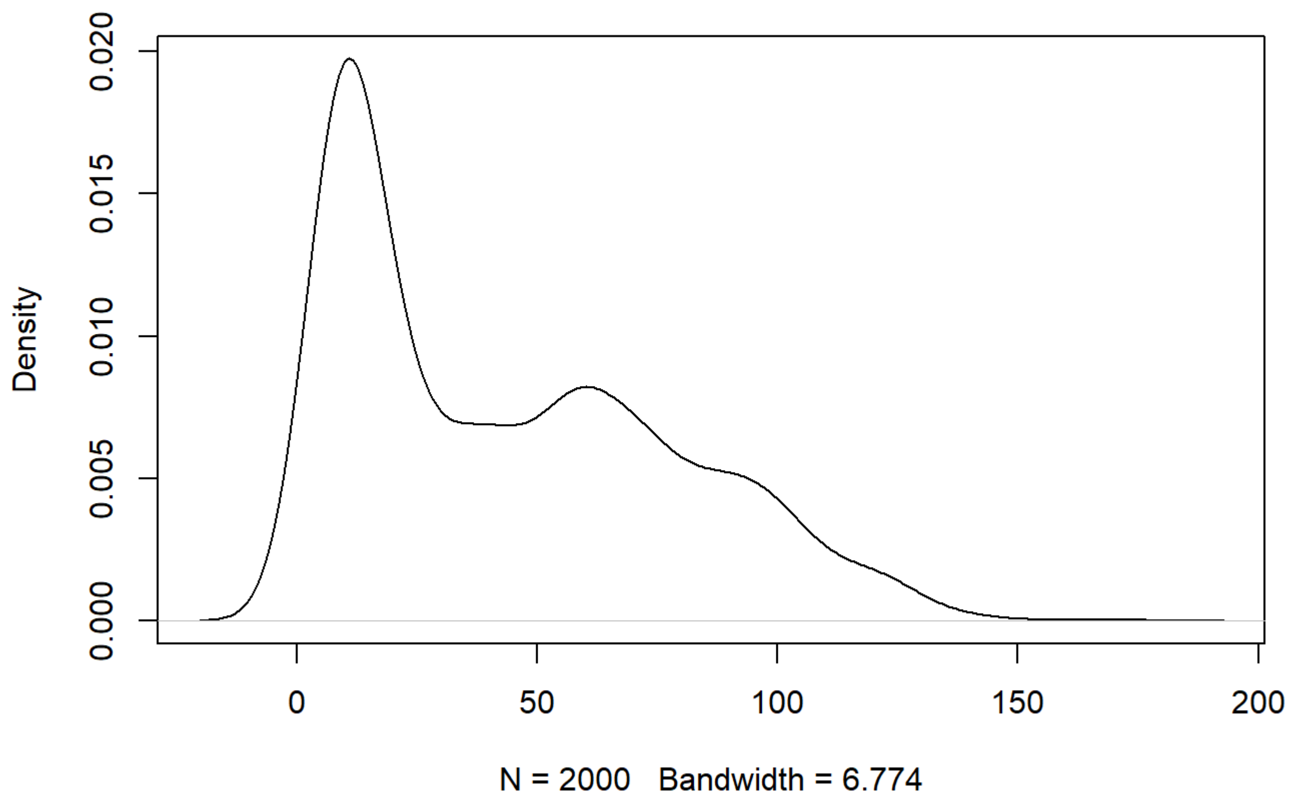
```
plot(density(geoJson$trafficCount), main="")
```



###

create a new column of scount which is the square-root of the counts. The distribution of scount shows several fluctuations and plateau, with a peak at ~10.

```
geoJson$scount<-sqrt(geoJson$trafficCount)
plot(density(geoJson$scount), main="")
```



###

subset data into light and heavy vehicles respectively

```
geoJsonH<-subset(geoJson,geoJson$classWeight == "Heavy")
geoJsonL<-subset(geoJson,geoJson$classWeight == "Light")
```

Transform

8,Create a new variable **scount** which contains the square-root of the counts.Because large variation of siteID, we also create a new variable for **ssiteID**

```
geoJson$scount<-sqrt(geoJson$trafficCount)
geoJson$ssiteID<-sqrt(geoJson$siteID)
```

Model

9, split data into training (90%) and test (10%) sets

```
index <- sample(rep(1:10, length.out=nrow(geoJson)))
train <- geoJson[index > 1, ]
test <- geoJson[index == 1, ]
```

fit two models using the training data; one that predicts scount from the overall mean of scount and one that predicts scount from class.

```
obs <- test$scount
predMean <- mean(train$scount)
lmfit <- lm(scount ~ classWeight, train)
predLM <- predict(lmfit, test)
```

Calculate the RMSE for both models using the test data

```
RMSE <- function(obs, pred) {
    sqrt(mean((obs - pred)^2))
}
RMSE(obs,predMean)
```

```
## [1] 32.2264
```

```
RMSE(obs,predLM)
```

```
## [1] 19.3417
```

fit a model including siteID as a predictor

```
lmfit_site <- lm(scount ~ classWeight+ssiteID, train)
predLM_site <- predict(lmfit_site, test)

RMSE(obs,predMean)
```

```
## [1] 32.2264
```

```
RMSE(obs,predLM_site)
```

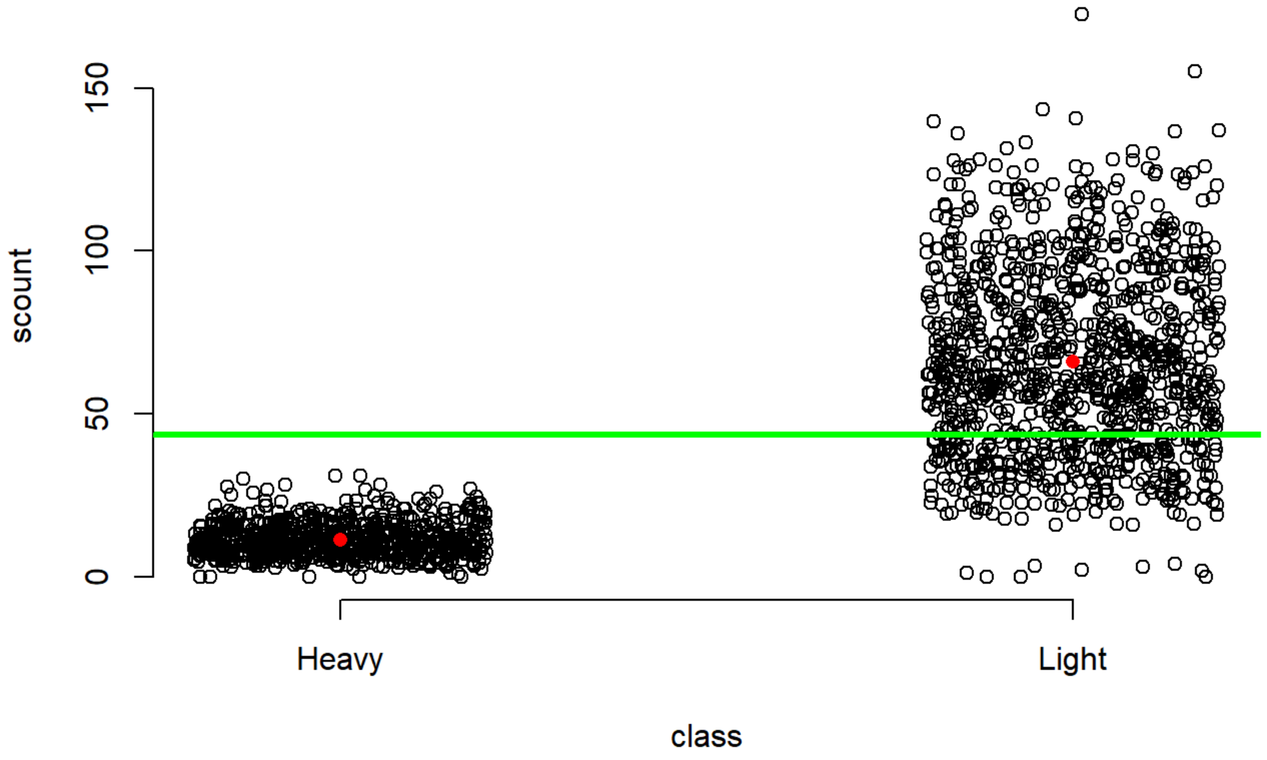
```
## [1] 19.3654
```

Visualise

10, Create a plot that shows the predictions of both models against the data.r

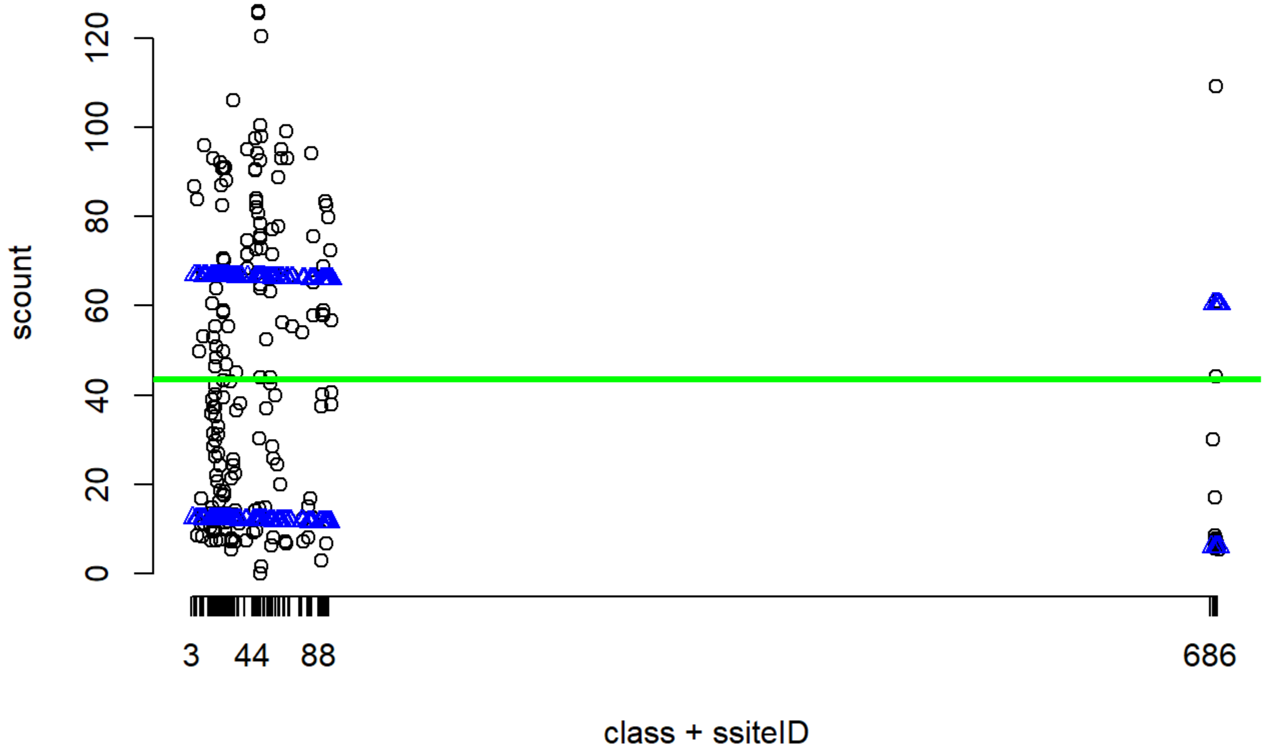
####(1) this plot shows scount vs. class, where blue line shows overall mean in train data, read dots show the predicted scount based on class type in test data. The model did not predict very well

```
plot(scount ~ jitter(as.numeric(factor(classWeight))), train,
     xlab="class", axes=FALSE)
axis(2)
axis(1, at=as.numeric(unique(factor(test$classWeight))),
     label=unique(factor(test$classWeight)))
abline(h=predMean, col="green",lwd=3)
points(as.numeric(unique(factor(test$classWeight))),
       predict(lmfit, data.frame(classWeight=unique(factor(test$classWeight)))),
       pch=16, col="red")
```



####(2) this plot shows scount vs. class + siteID, where blue line shows overall mean of scount in train data, red line shows the predicted regression line based on scount vs. class + siteID. This model did not work as well. We can see fitted data into 4 groups according to ssiteID and probably class as well.

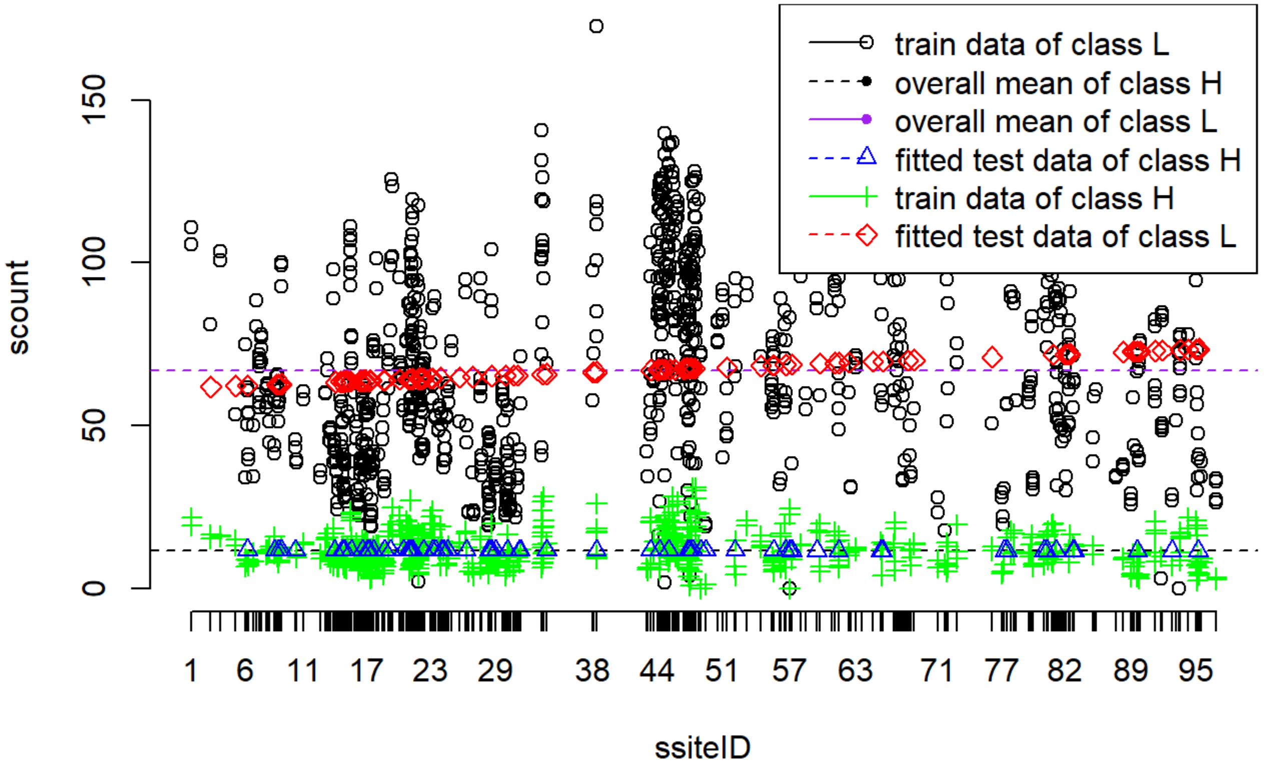
```
plot(scount ~ jitter(as.numeric(factor(classWeight))+ssiteID), test,
     xlab="class + ssiteID", ylab="scount",axes=FALSE)
axis(2)
axis(1, at=test$ssiteID,
     label=round(test$ssiteID))
abline(h=predMean, col="green",lwd=3)
points((jitter(as.numeric(factor(classWeight))+test$ssiteID)),
       predict(lmfit_site, data.frame(ssiteID=unique(jitter(as.numeric(factor(classWeight))+test
$ssiteID)))),
       pch=2, col="blue")
```



####(3) the underlying plot shows big variation of class H & L in terms of scount and siteID distribution. In order to understand data better, we choose ssiteID < 100 data set and plot scount according to vehicle class.

```
trainsplit<-split(train,train$classWeight)
newtrainH<- subset(trainsplit[[1]],trainsplit[[1]]$ssiteID<100)
newtrainL<- subset(trainsplit[[2]],trainsplit[[2]]$ssiteID<100)
newmeanH<- mean(newtrainH$scount)
newmeanL<- mean(newtrainL$scount)
testsplit<- split(test,test$classWeight)
NewtestH<- subset(testsplit[[1]],testsplit[[1]]$ssiteID<100)
newtestL<- subset(testsplit[[2]],testsplit[[2]]$ssiteID<100)
lmfit_H <- lm(scount ~ ssiteID, newtrainH)
predLM_H<- predict(lmfit_H, newtrainH)
lmfit_L <- lm(scount ~ ssiteID, newtrainL)
predLM_L<- predict(lmfit_L, newtrainL)

plot(newtrainL$ssiteID, newtrainL$scount,xlab="ssiteID", ylab="scount", axes=FALSE)
axis(1, at=newtrainL$ssiteID,
     label=round(newtrainL$ssiteID))
axis(2)
abline(h=newmeanH,col="black",lty=2)
abline(h=newmeanL,col="purple",lty=2)
points(newtrainH$ssiteID, newtrainH$scount,pch=3,col="green")
points(NewtestH$ssiteID,
       predict(lmfit_H, data.frame(ssiteID= NewtestH$ssiteID)),
       pch=2, col="blue")
points(newtestL$ssiteID,
       predict(lmfit_L, data.frame(ssiteID= newtestL$ssiteID)),
       pch=5, col="red")
legend(x="topright",legend=c("train data of class L","overall mean of class H","overall mean of
 class L","fitted test data of class H","train data of class H","fitted test data of class L"),
        col=c("black","black","purple","blue","green","red"), lwd=1, lty=c(1,2),
        pch=c(1,20,20,2,3,5), merge=FALSE)
```



Summary

In this lab we have learn how to download data from internet with various file format, such as csv, html and json files. Because each file has distinctive structure, we need to write specific R codes to extract data and save them in R dataframes. Furthermore, we used json file to do a test on linear and mean fittings. Firstly we split the data into training (90%) and test(10%) datasets. Secondly we did both overall mean and linear fitting based on square root of count (scount) and vehicle class (classWeight). Both root mean squared errors (RMSEs) were calculated as 35.7 (test vs. overall mean) and 22.0 (test vs. linear fitting of training data). Thirdly we added siteID into class for a multiple linear regression fitting, the RMSEs were almost no difference. The graph 1 and 2 show both models did not predict well because different range of scount in terms of class. To prove it, we seperated train data into heavy and light data and plot them against the ordered siteID, it showed large variation between two class.

In the end, we limited our data within the range of ssiteID < 100. In these data range we can see more clearly the regression fitting. It shows model predicts well for different vehicle types.

P.s. I did not use siteRef in this study because I would lose 1/3 of data if I force it into numeric values.