# Ryan Timbrook

## **Applied Data Science**

## **IST687 Intro to Data Science**, Spring 2019

## **Due Date:** 05/8/2019

## **Homework:** 5

### NetID: RTIMBROO

### SUID: 386792749

## #R Code - unexecuted

## ## Homework Week 5: JSON & tapply: Accident Analysis

## # ---Preprocess Steps:----------------------------------------------------------------------

## ### Clear objects from Memory

## rm(list=ls())

## ### Clear Console:

## cat("\014")

## ### Global Variable Assignments

## url.accident.analysis <- "http://opendata.maryland.gov/resource/pdvh-tf2u.json"

## ### Load Required Packages

## if(!require("RCurl")){install.packages("RCurl")}

## if(!require("curl")){install.packages("curl")}

## if(!require("RJSONIO")){install.packages("RJSONIO")}

## if(!require("jsonlite")){install.packages("jsonlite")}

## if(!require("sqldf")){install.packages("sqldf")}

## if(!require("stringr")){install.packages("stringr")}

## # ----Step 1: Load the data-----------------------------------------------------------------------------

## ## 1.1: Function to fetch the Accident JSON dataset

## fetchJSONDataSet <- function(ds.url,as.json=TRUE){

## u <- getURL(ds.url) # send URL to internet

## 

## if(as.json){

## ds <- fromJSON(u)

## }else{

## ds <- data.frame(fromJSON(u), stringsAsFactors = FALSE)

## }

## return(ds)

## }

## # Get the Accident Data from URL as JSON

## ds.accidental <- fetchJSONDataSet(url.accident.analysis, FALSE)

## #str(ds.accidental)

## #View(ds.accidental)

## # -----Step 2: Clean the data----------------------------------------------------------------------------

## ## 2.1: Function to Clean the data

## ### Remove last 8 columns, Rename the rest of the columns

## clean.accidental.ds <- function(ds, removeFirst = TRUE){

## 

## #Make all empty cells equal to NA

## ds[ds==""] <- NA

## 

## #Clean NA Columns from Dataframe

## ds <- ds[ ,!apply(ds,2,function(x) all(is.na(x)))]

## 

## #Clean empty Rows from Dataframe

## ds <- ds[!apply(ds,1,function(x) all(is.na(x))),]

## 

## ## Rename remaning columns

## new.col.names <- c(

## "ACC\_DATE",

## "ACC\_TIME",

## "ACC\_TIME\_CODE",

## "BARRACK",

## "CASE\_NUMBER",

## "CITY\_NAME",

## "COLLISION\_WITH\_1",

## "COLLISION\_WITH\_2",

## "COUNTY\_CODE",

## "COUNTY\_NAME",

## "DAY\_OF\_WEEK",

## "DIST\_DIRECTION",

## "DIST\_FROM\_INTERSECT",

## "INJURY",

## "INTERSECT\_ROAD",

## "PROP\_DEST",

## "ROAD",

## "VEHICLE\_COUNT"

## 

## )

## 

## colnames(ds) <- new.col.names

## 

## #Handle NA values

## #ds$ACC\_DATE[is.na(ds$ACC\_DATE)] <- 'NOT\_LISTED'

## #ds$ACC\_TIME[is.na(ds$ACC\_TIME)] <- 'NOT\_LISTED'

## ds$ACC\_TIME\_CODE[is.na(ds$ACC\_TIME\_CODE)] <- 'NOT\_LISTED'

## ds$BARRACK[is.na(ds$BARRACK)] <- 'NOT\_LISTED'

## ds$CASE\_NUMBER[is.na(ds$CASE\_NUMBER)] <- 'NOT\_LISTED'

## ds$CITY\_NAME[is.na(ds$CITY\_NAME)] <- 'NOT\_LISTED'

## ds$COLLISION\_WITH\_1[is.na(ds$COLLISION\_WITH\_1)] <- 'NOT\_LISTED'

## ds$COLLISION\_WITH\_2[is.na(ds$COLLISION\_WITH\_2)] <- 'NOT\_LISTED'

## ds$COUNTY\_CODE[is.na(ds$COUNTY\_CODE)] <- 'NOT\_LISTED'

## ds$COUNTY\_NAME[is.na(ds$COUNTY\_NAME)] <- 'NOT\_LISTED'

## ds$DAY\_OF\_WEEK[is.na(ds$DAY\_OF\_WEEK)] <- 'NOT\_LISTED'

## ds$DAY\_OF\_WEEK <- str\_replace\_all(ds$DAY\_OF\_WEEK," ","")

## ds$DAY\_OF\_WEEK <- str\_to\_upper(ds$DAY\_OF\_WEEK)

## ds$DAY\_OF\_WEEK <- as.factor(ds$DAY\_OF\_WEEK)

## ds$INJURY[is.na(ds$INJURY)] <- 'NO'

## ds$VEHICLE\_COUNT[is.na(ds$VEHICLE\_COUNT)] <- "1"

## ds$VEHICLE\_COUNT <- as.numeric(ds$VEHICLE\_COUNT)

## 

## ds$DIST\_FROM\_INTERSECT <- as.numeric(ds$DIST\_FROM\_INTERSECT)

## ds$INJURY <- as.factor(ds$INJURY)

## ds$DIST\_DIRECTION <- as.factor(ds$DIST\_DIRECTION)

## 

## # Remove Columns

## l <- length(ds)

## if(removeFirst){

## ## Remove first 8 columns

## ds <- ds[,-c(1:8)]

## }else{

## ## Remove last 8 columns

## ds <- ds[,-c((l-7):l)]

## }

## return(ds)

## }

## # Execute Clean DataSet Functino

## cds <- clean.accidental.ds(ds.accidental)

## # -----Step 3: Understand the data using SQL (via SQLDF)----------------------------------------------------------------------------

## # 3.1: How many accidents happen on SUNDAY - An accident can have 1 or more vehicals involved

## sql.sun.acdnt.cnt <- sqldf("select count(\*) as 'SUNDAY\_ACCT\_CNT' from cds where DAY\_OF\_WEEK = 'SUNDAY'")

## sql.sun.acdnt.cnt

## # 3.2: How many accidents had injuries

## sql.inj.acdnt.cnt <- sqldf("select count(\*) as 'INJURY\_ACCT\_CNT' from cds where INJURY = 'YES'")

## sql.inj.acdnt.cnt

## # 3.3: List the injuries by day

## sql.inj.by.day <- sqldf("select count(\*) as 'INJURY\_CNT', DAY\_OF\_WEEK from cds where INJURY = 'YES' group by DAY\_OF\_WEEK order by INJURY\_CNT")

## sql.inj.by.day

## # -----Step 4: Understand the data using tapply----------------------------------------------------------------------------

## # 4.1: How many accidents happen on Sunday

## tapp.sun.acdnt.cnt <- tapply(cds$DAY\_OF\_WEEK, cds$DAY\_OF\_WEEK=='SUNDAY', length)

## tapp.sun.acdnt.cnt["TRUE"]

## # 4.2: How many accidents had injuries

## tapp.inj.acdnt.cnt <- tapply(cds$INJURY, cds$INJURY=='YES', length)

## tapp.inj.acdnt.cnt["TRUE"]

## # 4.3: List the injuries by day

## tapp.inj.by.day <- tapply(cds$INJURY, list(cds$INJURY=='YES', cds$DAY\_OF\_WEEK), length)

## tapp.inj.by.day["TRUE",]

## #R Code – executed

> ### Global Variable Assignments

> url.accident.analysis <- "http://opendata.maryland.gov/resource/pdvh-tf2u.json"

>

> ### Load Required Packages

> if(!require("RCurl")){install.packages("RCurl")}

> if(!require("curl")){install.packages("curl")}

> if(!require("RJSONIO")){install.packages("RJSONIO")}

> if(!require("jsonlite")){install.packages("jsonlite")}

> if(!require("sqldf")){install.packages("sqldf")}

> if(!require("stringr")){install.packages("stringr")}

>

> # ----Step 1: Load the data-----------------------------------------------------------------------------

>

> ## 1.1: Function to fetch the Accident JSON dataset

> fetchJSONDataSet <- function(ds.url,as.json=TRUE){

+ u <- getURL(ds.url) # send URL to internet

+

+ if(as.json){

+ ds <- fromJSON(u)

+ }else{

+ ds <- data.frame(fromJSON(u), stringsAsFactors = FALSE)

+ }

+

+ return(ds)

+ }

>

>

> # Get the Accident Data from URL as JSON

> ds.accidental <- fetchJSONDataSet(url.accident.analysis, FALSE)

> #str(ds.accidental)

> #View(ds.accidental)

>

> # -----Step 2: Clean the data----------------------------------------------------------------------------

> ## 2.1: Function to Clean the data

> ### Remove last 8 columns, Rename the rest of the columns

> clean.accidental.ds <- function(ds, removeFirst = TRUE){

+

+ #Make all empty cells equal to NA

+ ds[ds==""] <- NA

+

+ #Clean NA Columns from Dataframe

+ ds <- ds[ ,!apply(ds,2,function(x) all(is.na(x)))]

+

+ #Clean empty Rows from Dataframe

+ ds <- ds[!apply(ds,1,function(x) all(is.na(x))),]

+

+ ## Rename remaning columns

+ new.col.names <- c(

+ "ACC\_DATE",

+ "ACC\_TIME",

+ "ACC\_TIME\_CODE",

+ "BARRACK",

+ "CASE\_NUMBER",

+ "CITY\_NAME",

+ "COLLISION\_WITH\_1",

+ "COLLISION\_WITH\_2",

+ "COUNTY\_CODE",

+ "COUNTY\_NAME",

+ "DAY\_OF\_WEEK",

+ "DIST\_DIRECTION",

+ "DIST\_FROM\_INTERSECT",

+ "INJURY",

+ "INTERSECT\_ROAD",

+ "PROP\_DEST",

+ "ROAD",

+ "VEHICLE\_COUNT"

+

+ )

+

+ colnames(ds) <- new.col.names

+

+ #Handle NA values

+ #ds$ACC\_DATE[is.na(ds$ACC\_DATE)] <- 'NOT\_LISTED'

+ #ds$ACC\_TIME[is.na(ds$ACC\_TIME)] <- 'NOT\_LISTED'

+ ds$ACC\_TIME\_CODE[is.na(ds$ACC\_TIME\_CODE)] <- 'NOT\_LISTED'

+ ds$BARRACK[is.na(ds$BARRACK)] <- 'NOT\_LISTED'

+ ds$CASE\_NUMBER[is.na(ds$CASE\_NUMBER)] <- 'NOT\_LISTED'

+ ds$CITY\_NAME[is.na(ds$CITY\_NAME)] <- 'NOT\_LISTED'

+ ds$COLLISION\_WITH\_1[is.na(ds$COLLISION\_WITH\_1)] <- 'NOT\_LISTED'

+ ds$COLLISION\_WITH\_2[is.na(ds$COLLISION\_WITH\_2)] <- 'NOT\_LISTED'

+ ds$COUNTY\_CODE[is.na(ds$COUNTY\_CODE)] <- 'NOT\_LISTED'

+ ds$COUNTY\_NAME[is.na(ds$COUNTY\_NAME)] <- 'NOT\_LISTED'

+ ds$DAY\_OF\_WEEK[is.na(ds$DAY\_OF\_WEEK)] <- 'NOT\_LISTED'

+ ds$DAY\_OF\_WEEK <- str\_replace\_all(ds$DAY\_OF\_WEEK," ","")

+ ds$DAY\_OF\_WEEK <- str\_to\_upper(ds$DAY\_OF\_WEEK)

+ ds$DAY\_OF\_WEEK <- as.factor(ds$DAY\_OF\_WEEK)

+ ds$INJURY[is.na(ds$INJURY)] <- 'NO'

+ ds$VEHICLE\_COUNT[is.na(ds$VEHICLE\_COUNT)] <- "1"

+ ds$VEHICLE\_COUNT <- as.numeric(ds$VEHICLE\_COUNT)

+

+ ds$DIST\_FROM\_INTERSECT <- as.numeric(ds$DIST\_FROM\_INTERSECT)

+ ds$INJURY <- as.factor(ds$INJURY)

+ ds$DIST\_DIRECTION <- as.factor(ds$DIST\_DIRECTION)

+

+ # Remove Columns

+ l <- length(ds)

+ if(removeFirst){

+ ## Remove first 8 columns

+ ds <- ds[,-c(1:8)]

+ }else{

+ ## Remove last 8 columns

+ ds <- ds[,-c((l-7):l)]

+ }

+

+ return(ds)

+ }

>

> # Execute Clean DataSet Functino

> cds <- clean.accidental.ds(ds.accidental)

>

>

> # -----Step 3: Understand the data using SQL (via SQLDF)----------------------------------------------------------------------------

> # 3.1: How many accidents happen on SUNDAY - An accident can have 1 or more vehicals involved

> sql.sun.acdnt.cnt <- sqldf("select count(\*) as 'SUNDAY\_ACCT\_CNT' from cds where DAY\_OF\_WEEK = 'SUNDAY'")

> sql.sun.acdnt.cnt

SUNDAY\_ACCT\_CNT

1 95

>

> # 3.2: How many accidents had injuries

> sql.inj.acdnt.cnt <- sqldf("select count(\*) as 'INJURY\_ACCT\_CNT' from cds where INJURY = 'YES'")

> sql.inj.acdnt.cnt

INJURY\_ACCT\_CNT

1 301

>

> # 3.3: List the injuries by day

> sql.inj.by.day <- sqldf("select count(\*) as 'INJURY\_CNT', DAY\_OF\_WEEK from cds where INJURY = 'YES' group by DAY\_OF\_WEEK order by INJURY\_CNT")

> sql.inj.by.day

INJURY\_CNT DAY\_OF\_WEEK

1 23 SUNDAY

2 41 MONDAY

3 42 SATURDAY

4 42 TUESDAY

5 49 FRIDAY

6 50 THURSDAY

7 54 WEDNESDAY

>

> # -----Step 4: Understand the data using tapply----------------------------------------------------------------------------

> # 4.1: How many accidents happen on Sunday

> tapp.sun.acdnt.cnt <- tapply(cds$DAY\_OF\_WEEK, cds$DAY\_OF\_WEEK=='SUNDAY', length)

> tapp.sun.acdnt.cnt["TRUE"]

TRUE

95

>

> # 4.2: How many accidents had injuries

> tapp.inj.acdnt.cnt <- tapply(cds$INJURY, cds$INJURY=='YES', length)

> tapp.inj.acdnt.cnt["TRUE"]

TRUE

301

>

> # 4.3: List the injuries by day

> tapp.inj.by.day <- tapply(cds$INJURY, list(cds$INJURY=='YES', cds$DAY\_OF\_WEEK), length)

> tapp.inj.by.day["TRUE",]

FRIDAY MONDAY SATURDAY SUNDAY THURSDAY TUESDAY WEDNESDAY

49 41 42 23 50 42 54