title: "SE project Edmonds Data exploration and question" author: "Darius McDaniel" date: "November 20, 2015" output: word\_document ---

## This document will aim to answer different questions from the edmonds data. There will be a question answered for most of the main topics covered in this semesters class.

All programs will be added and committed to github and found here : <https://github.com/djmcdan/SE_Project.git>

Below are all the packages used in this project and also the data used.

##   
Read 0.0% of 771385 rows  
Read 10.4% of 771385 rows  
Read 22.0% of 771385 rows  
Read 31.1% of 771385 rows  
Read 37.6% of 771385 rows  
Read 48.0% of 771385 rows  
Read 53.2% of 771385 rows  
Read 60.9% of 771385 rows  
Read 71.3% of 771385 rows  
Read 73.9% of 771385 rows  
Read 84.3% of 771385 rows  
Read 94.6% of 771385 rows  
Read 98.5% of 771385 rows  
Read 771385 rows and 203 (of 203) columns from 0.575 GB file in 00:00:24

##   
Read 22.8% of 2416174 rows  
Read 56.3% of 2416174 rows  
Read 90.2% of 2416174 rows  
Read 2416174 rows and 4 (of 4) columns from 0.101 GB file in 00:00:05

##   
Read 0.0% of 2445924 rows  
Read 10.2% of 2445924 rows  
Read 20.0% of 2445924 rows  
Read 31.1% of 2445924 rows  
Read 42.1% of 2445924 rows  
Read 53.1% of 2445924 rows  
Read 64.2% of 2445924 rows  
Read 75.6% of 2445924 rows  
Read 86.7% of 2445924 rows  
Read 98.1% of 2445924 rows  
Read 2445924 rows and 24 (of 24) columns from 0.414 GB file in 00:00:15

##   
Read 0.0% of 3780756 rows  
Read 9.8% of 3780756 rows  
Read 18.3% of 3780756 rows  
Read 28.3% of 3780756 rows  
Read 37.8% of 3780756 rows  
Read 47.3% of 3780756 rows  
Read 56.9% of 3780756 rows  
Read 66.4% of 3780756 rows  
Read 75.9% of 3780756 rows  
Read 84.6% of 3780756 rows  
Read 92.6% of 3780756 rows  
Read 3780756 rows and 8 (of 8) columns from 0.470 GB file in 00:00:20

Now we start with questions for exploration in 5 topics discussed this semester around using Dyplyr, Visualizations, sqldf, objects and JSON/XML to answer different questions with the Edmonds car data.

Questions

1. Dplyr.   
   
Based on the visitor data how long did the visits last and did the longer visit times lead to purchases or vis versa?

To first answer this question we must do some data cleaning.

Answers to posed questions

#a. What was the average,min and max time spent on the site for all visits  
mean\_med\_visitor\_time

## Source: local data table [1 x 2]  
##   
## mean\_hours\_visited median\_hours\_visited  
## 1 21.73256 0.6347222

#b. Now for those that purchased new or used car what was the avg time visited?  
  
avgs\_visitor\_time\_purchaseNw

## Source: local data table [1 x 4]  
##   
## min\_hours\_visited mean\_hours\_visited med\_hours\_visited max\_hours\_visited  
## 1 -1.155556 25.77313 0.7894444 23259.6

avgs\_visitor\_time\_purchaseUs

## Source: local data table [1 x 4]  
##   
## min\_hours\_visited mean\_hours\_visited med\_hours\_visited max\_hours\_visited  
## 1 0 28.05843 0.8994444 23259.6

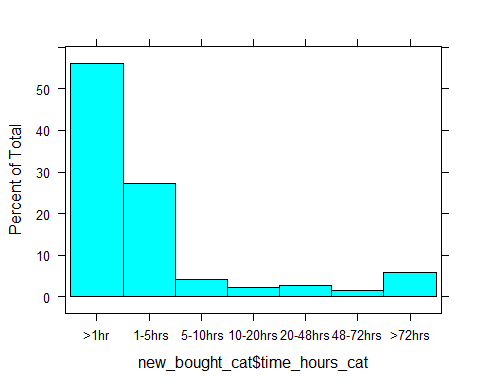
#c. How many purchases were >1hr, 1-5, 5-10,10-20, gt20 hrs  
  
  
table(new\_bought\_cat$time\_hours\_cat )

##   
## >1hr 1-5hrs 5-10hrs 10-20hrs 20-48hrs 48-72hrs >72hrs   
## 43491 21068 3258 1713 2187 1133 4551

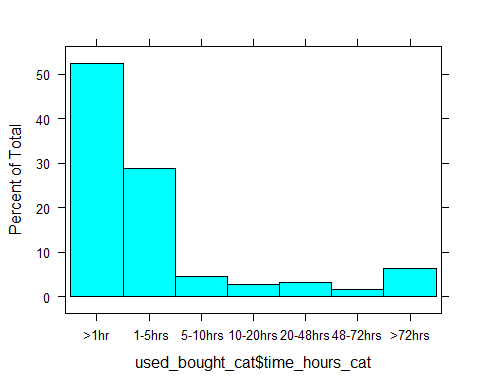
table(used\_bought\_cat$time\_hours\_cat )

##   
## >1hr 1-5hrs 5-10hrs 10-20hrs 20-48hrs 48-72hrs >72hrs   
## 7104 3897 622 375 425 226 865

histogram(new\_bought\_cat$time\_hours\_cat)



histogram(used\_bought\_cat$time\_hours\_cat)



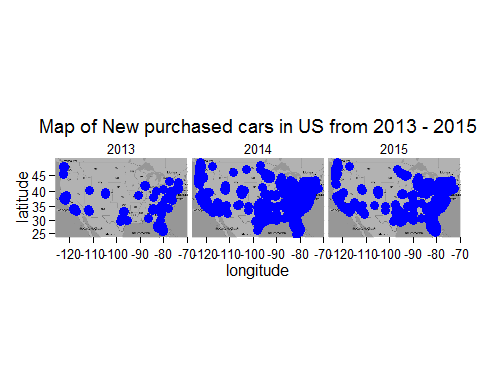
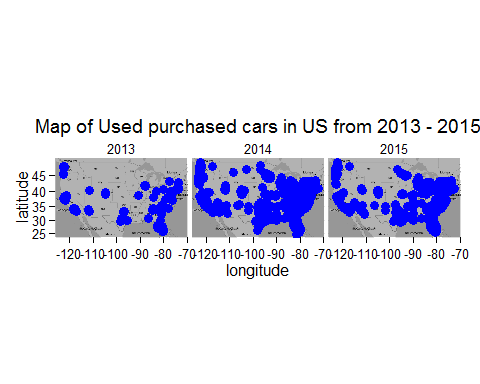
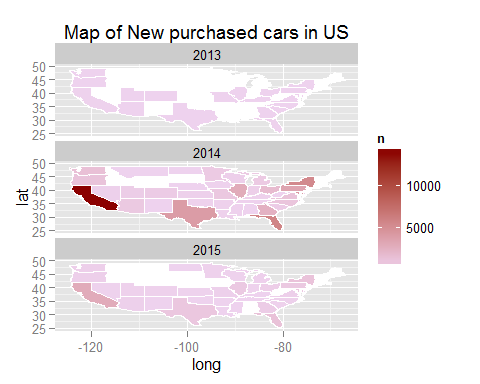
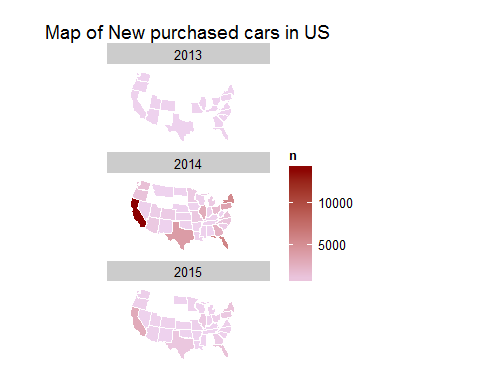
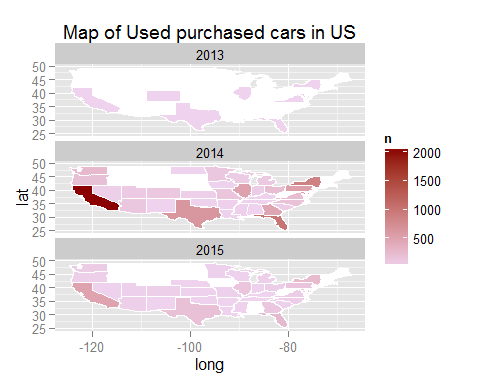
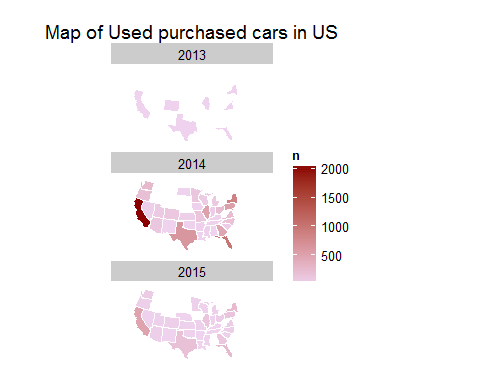
#Observation Over 50percent of purchases that happened within an hour of visiting the site for new and used cars  
  
#d. How many new cars had less than 100 miles on them and greater than 100 miles?  
  
#New  
 table(milage\_bought\_new$mileage\_cat100 )

##   
## 1 2   
## 64891 10687

#Used   
 table(milage\_bought\_used$mileage\_cat100 )

##   
## 1 2   
## 247 13045

2. Visualizations ( googlevis, ggplot2)  
   
 Using zip codes provided from transactions data map the purchases using googlevis? Than look the transactions separated by the year the car was purchased using ggplot2 .

1. SQL (sqldf / Rsqlite)

* Create a SQLite database and figure out : How many new cars where bought by manufactuer year ?, How many different body types were purchased new and used displaying only the top 10 ?, What are the top five makers that sold the most new and used cars?,  
  What are the average, min and max mileages bought along with the average, min and max price bought?

## [1] TRUE

## [1] TRUE

## [1] TRUE

## [1] TRUE

## [1] TRUE

## [1] "Configuration\_" "Leads\_" "Shopping\_" "Transactions\_"   
## [5] "Visitor\_"

## [1] "visitor\_key" "session\_count"   
## [3] "first\_session\_start\_datetime" "last\_session\_start\_datetime"   
## [5] "new\_flag" "used\_flag"   
## [7] "cpo\_flag" "preprod\_flag"   
## [9] "new\_page\_views" "used\_page\_views"   
## [11] "cpo\_page\_views" "preprod\_page\_views"   
## [13] "new\_dwell\_time" "used\_dwell\_time"   
## [15] "cpo\_dwell\_time" "preprod\_dwell\_time"   
## [17] "consideration\_count" "configuration\_count"   
## [19] "nci\_count" "compact\_config\_count"   
## [21] "compact\_consid\_count" "compact\_dealerengage\_count"   
## [23] "compact\_inventory\_count" "compact\_leads\_count"   
## [25] "engage\_count" "models\_excl\_count"   
## [27] "mydp\_count" "new\_leads\_count"   
## [29] "option\_count" "price\_count"   
## [31] "research\_count" "used\_consideration\_count"   
## [33] "paid\_agg\_search\_flag" "free\_agg\_search\_flag"   
## [35] "book\_agg\_search\_flag" "tot\_dwell\_time"   
## [37] "page\_views" "leasing\_art"   
## [39] "sell\_adv\_art" "saft\_adv\_art"   
## [41] "fuel\_eco\_art" "drv\_tip"   
## [43] "mnth\_loan\_pay\_calc" "auto\_lease\_calc"   
## [45] "afford\_calc" "inc\_reb\_calc"   
## [47] "gas\_guzz\_calc" "inc\_car"   
## [49] "inc\_index\_car" "dlr\_rev\_index\_inv"   
## [51] "ddp\_sale\_rev\_inv" "buy\_guide\_carrev"   
## [53] "buy\_guide\_know\_carrev" "buy\_guide\_index\_carrev"   
## [55] "feat\_art\_carrev" "comparator"   
## [57] "maint\_calls\_maint" "maint\_howto\_maint"   
## [59] "new\_finder" "features\_nmydp"   
## [61] "safety\_nmydp" "mpg\_nmydp"   
## [63] "lt\_rd\_test\_nmydp" "comp\_testrev\_nmydp"   
## [65] "used\_cars\_tmv\_appraiser" "app13\_umydp"   
## [67] "app4\_umydp" "pc\_adv"   
## [69] "pc\_calc" "pc\_car\_inc"   
## [71] "pc\_car\_rev" "pc\_car\_inv"   
## [73] "pc\_comparator" "pc\_maint"   
## [75] "pc\_new\_finder" "pc\_new\_mydp"   
## [77] "pc\_appr" "pc\_used\_mydp"   
## [79] "drr\_sales\_submit\_count" "drr\_service\_submit\_count"   
## [81] "forums\_submit\_count" "campaign\_keyword"   
## [83] "campaign\_category" "imp\_rubicon"   
## [85] "imp\_gm" "imp\_edmunds\_com\_house\_ad"   
## [87] "imp\_toyota" "imp\_targeted\_incentives"   
## [89] "imp\_chrysler\_llc" "imp\_honda"   
## [91] "imp\_adp\_dms" "imp\_state\_farm"   
## [93] "imp\_mazda" "imp\_nissan"   
## [95] "imp\_ford" "imp\_hyundai"   
## [97] "imp\_lexus" "imp\_subaru"   
## [99] "imp\_1x1\_tracking" "imp\_chevrolet"   
## [101] "imp\_acura" "imp\_mercedes"   
## [103] "imp\_volkswagen" "imp\_kia\_ma"   
## [105] "imp\_mitsubishi\_msa" "imp\_jeep"   
## [107] "imp\_gmc" "imp\_house"   
## [109] "imp\_lincoln" "imp\_tier\_iii\_adv"   
## [111] "imp\_cadillac" "imp\_volvo"   
## [113] "imp\_porsche" "imp\_experian"   
## [115] "imp\_buick" "imp\_infiniti"   
## [117] "imp\_kia" "imp\_scion"   
## [119] "imp\_land\_rover" "imp\_mini"   
## [121] "imp\_answerfinancial" "imp\_law\_test"   
## [123] "imp\_floodlight" "imp\_carchex"   
## [125] "imp\_suite\_66" "imp\_fiat"   
## [127] "imp\_ferrari" "imp\_adops\_test\_adv"   
## [129] "imp\_smart" "imp\_sway\_program"   
## [131] "imp\_cpo" "clk\_total"   
## [133] "clk\_goog\_ad\_ex" "clk\_rubicon"   
## [135] "clk\_gm" "clk\_edmunds\_com\_house\_ad"   
## [137] "clk\_toyota" "clk\_targeted\_incentives"   
## [139] "clk\_chrysler\_llc" "clk\_honda"   
## [141] "clk\_adp\_dms" "clk\_state\_farm"   
## [143] "clk\_mazda" "clk\_nissan"   
## [145] "clk\_ford" "clk\_hyundai"   
## [147] "clk\_lexus" "clk\_subaru"   
## [149] "clk\_1x1\_tracking" "clk\_chevrolet"   
## [151] "clk\_acura" "clk\_mercedes"   
## [153] "clk\_volkswagen" "clk\_kia\_ma"   
## [155] "clk\_mitsubishi\_msa" "clk\_jeep"   
## [157] "clk\_gmc" "clk\_house"   
## [159] "clk\_lincoln" "clk\_tier\_iii\_adv"   
## [161] "clk\_cadillac" "clk\_volvo"   
## [163] "clk\_porsche" "clk\_experian"   
## [165] "clk\_buick" "clk\_infiniti"   
## [167] "clk\_kia" "clk\_scion"   
## [169] "clk\_land\_rover" "clk\_mini"   
## [171] "clk\_answerfinancial" "clk\_law\_test"   
## [173] "clk\_floodlight" "clk\_carchex"   
## [175] "clk\_suite\_66" "clk\_fiat"   
## [177] "clk\_ferrari" "clk\_adops\_test\_adv"   
## [179] "clk\_smart" "clk\_sway\_program"   
## [181] "clk\_cpo" "first\_agg\_search\_type"   
## [183] "second\_agg\_search\_type" "third\_agg\_search\_type"   
## [185] "fourth\_agg\_search\_type" "fifth\_agg\_search\_type"   
## [187] "last\_agg\_search\_type" "first\_platform\_type"   
## [189] "last\_platform\_type" "first\_device\_model"   
## [191] "last\_device\_model" "first\_referring\_url"   
## [193] "last\_referring\_url" "make\_count"   
## [195] "model\_count" "submodel\_count"   
## [197] "modelyear\_count" "style\_count"   
## [199] "zip" "dma\_name"   
## [201] "credit\_active" "credit\_worthiness"   
## [203] "age\_range"

## [1] "visitor\_key" "dealer\_location\_id" "date\_sold"   
## [4] "price\_bought" "dma\_bought" "zip\_bought"   
## [7] "state\_bought" "model\_year\_bought" "make\_bought"   
## [10] "model\_bought" "trim\_bought" "bodytype\_bought"   
## [13] "transmission\_bought" "color\_bought" "mileage\_bought"   
## [16] "new\_or\_used\_bought" "msrp\_bought"

Queries for Questions in the database

#Lets see how many new cars where bought by manufactuer year  
  
dbGetQuery(db,"SELECT model\_year\_bought as Model\_year\_purchased\_New, count(\*) as modelYR\_freq from Transactions\_ WHERE new\_or\_used\_bought = 'N' GROUP BY model\_year\_bought")

## Model\_year\_purchased\_New modelYR\_freq  
## 1 2012 32  
## 2 2013 2696  
## 3 2014 46777  
## 4 2015 31684  
## 5 2016 81

dbGetQuery(db,"SELECT model\_year\_bought as Model\_year\_purchased\_Used, count(\*) as modelYR\_freq from Transactions\_ WHERE new\_or\_used\_bought = 'U' GROUP BY model\_year\_bought")

## Model\_year\_purchased\_Used modelYR\_freq  
## 1 1980 1  
## 2 1983 1  
## 3 1990 2  
## 4 1991 1  
## 5 1993 1  
## 6 1994 2  
## 7 1995 3  
## 8 1996 6  
## 9 1997 7  
## 10 1998 16  
## 11 1999 17  
## 12 2000 33  
## 13 2001 49  
## 14 2002 65  
## 15 2003 101  
## 16 2004 129  
## 17 2005 189  
## 18 2006 195  
## 19 2007 363  
## 20 2008 474  
## 21 2009 355  
## 22 2010 821  
## 23 2011 2438  
## 24 2012 3196  
## 25 2013 3188  
## 26 2014 2141  
## 27 2015 313

#How many different body types were purchased new and used displaying only the top 10  
  
dbGetQuery(db,"select bodytype\_bought, count(\*) as freq from Transactions\_ WHERE new\_or\_used\_bought = 'N' group by bodytype\_bought order by freq desc LIMIT 10")

## bodytype\_bought freq  
## 1 4 Door SUV 32038  
## 2 4 Door Sedan 28087  
## 3 4 Door Hatchback 5372  
## 4 4 Door Passenger Van 3774  
## 5 4 Door Crew Cab Short Bed Truck 2533  
## 6 2 Door Coupe 2177  
## 7 4 Door Wagon 2107  
## 8 5 Door Hatchback 1779  
## 9 2 Door Convertible 782  
## 10 4 Door Extended Cab Truck 412

dbGetQuery(db,"select bodytype\_bought, count(\*) as freq from Transactions\_ WHERE new\_or\_used\_bought = 'U' group by bodytype\_bought order by freq desc LIMIT 10")

## bodytype\_bought freq  
## 1 4 Door Sedan 5463  
## 2 4 Door SUV 4878  
## 3 4 Door Hatchback 678  
## 4 2 Door Coupe 571  
## 5 4 Door Passenger Van 487  
## 6 4 Door Crew Cab Short Bed Truck 434  
## 7 2 Door Convertible 424  
## 8 4 Door Wagon 319  
## 9 5 Door Hatchback 205  
## 10 2 Door Hatchback 120

#What are the top five makers that sold the most new and used cars   
  
dbGetQuery(db,"select make\_bought, count(\*) as freq from Transactions\_ WHERE new\_or\_used\_bought = 'N' GROUP BY make\_bought order by freq desc LIMIT 5")

## make\_bought freq  
## 1 Honda 17039  
## 2 Toyota 12174  
## 3 Subaru 5298  
## 4 Nissan 4671  
## 5 Hyundai 4564

dbGetQuery(db,"select make\_bought, count(\*) as freq from Transactions\_ WHERE new\_or\_used\_bought = 'U' GROUP BY make\_bought order by freq desc LIMIT 5")

## make\_bought freq  
## 1 Honda 1832  
## 2 Toyota 1634  
## 3 Ford 939  
## 4 Mercedes-Benz 898  
## 5 BMW 823

#Honda and Toyota both have the most purchases for new and used  
  
  
#What was the average, min and max mileages bought along with the average, min and max price bought for the top ten  
  
#New  
dbGetQuery(db,"select make\_bought, count(\*) as freq, min(price\_bought), avg(price\_bought), max(price\_bought),min(mileage\_bought), avg(mileage\_bought), max(mileage\_bought)   
 from Transactions\_ WHERE new\_or\_used\_bought = 'N' GROUP BY make\_bought order by freq desc LIMIT 10")

## make\_bought freq min(price\_bought) avg(price\_bought)  
## 1 Honda 17039 0 25083.83  
## 2 Toyota 12174 13436.33 27685.70  
## 3 Subaru 5298 0 26249.49  
## 4 Nissan 4671 10030 26286.60  
## 5 Hyundai 4564 0 25882.26  
## 6 Ford 4121 0 33134.95  
## 7 Mazda 3525 12758 24368.85  
## 8 Lexus 3241 29142.75 41824.23  
## 9 Mercedes-Benz 2968 0 52808.22  
## 10 BMW 2816 0 48274.03  
## max(price\_bought) min(mileage\_bought) avg(mileage\_bought)  
## 1 NA 0 63.15729  
## 2 NA 0 95.71456  
## 3 NA 0 65.22122  
## 4 NA 0 90.10340  
## 5 NA 0 160.12708  
## 6 NA 0 154.69376  
## 7 NA 0 72.40738  
## 8 NA 0 70.50849  
## 9 NA 0 241.48416  
## 10 NA 0 202.16513  
## max(mileage\_bought)  
## 1 NA  
## 2 NA  
## 3 NA  
## 4 NA  
## 5 NA  
## 6 NA  
## 7 NA  
## 8 NA  
## 9 NA  
## 10 NA

#Used  
dbGetQuery(db,"select make\_bought, count(\*) as freq, min(price\_bought), avg(price\_bought), max(price\_bought),min(mileage\_bought), avg(mileage\_bought), max(mileage\_bought)   
 from Transactions\_ WHERE new\_or\_used\_bought = 'U' GROUP BY make\_bought order by freq desc LIMIT 10")

## make\_bought freq min(price\_bought) avg(price\_bought)  
## 1 Honda 1832 10000 18261.78  
## 2 Toyota 1634 0 19221.86  
## 3 Ford 939 0 21199.82  
## 4 Mercedes-Benz 898 0 38165.51  
## 5 BMW 823 0 35034.83  
## 6 Nissan 770 0 17803.40  
## 7 Chevrolet 765 1000 22424.13  
## 8 Hyundai 754 10301 17041.56  
## 9 Volkswagen 662 0 18027.69  
## 10 Lexus 657 0 29717.14  
## max(price\_bought) min(mileage\_bought) avg(mileage\_bought)  
## 1 NA 0 36502.40  
## 2 NA 0 39599.64  
## 3 NA 0 36284.88  
## 4 NA 10005 22354.30  
## 5 NA 1 23394.97  
## 6 NA 0 33506.52  
## 7 NA 10 35383.07  
## 8 NA 0 30128.75  
## 9 NA 0 30922.27  
## 10 NA 10 30433.96  
## max(mileage\_bought)  
## 1 NA  
## 2 NA  
## 3 NA  
## 4 NA  
## 5 NA  
## 6 NA  
## 7 NA  
## 8 NA  
## 9 NA  
## 10 NA

#Honda and Toyota both have the most purchases for new and used

1. Objects

* Questions
* a.Create printout of the data created to look at purchases of new cars bought with over 100 miles and used cars with under 100 miles.
* b.Summarize the total purchase amounts .
* c.Plot using ggmap to see where purchases of new cars bought with over 100 miles and used cars with under 100 miles were.

NewMilesOver100 <- transactions %>% filter(mileage\_bought >100 & new\_or\_used\_bought == "N")  
UsedMilesUnder100 <- transactions %>% filter(mileage\_bought < 100 & new\_or\_used\_bought == "U")  
  
#Create Object  
  
cardata <- function(m,x,y,z) {  
 Edcars <- list(title = m, id=x, source=y, data=z)  
 class(Edcars) <- "cars"  
 return(Edcars)  
}  
  
PurchasedNewOver100 <- cardata("New Cars Purchased with Miles Over 100 in","US","http://www.edmunds.com",NewMilesOver100)  
PurchasedUsedUnder100 <- cardata("Used Cars Purchased with Miles Under 100 in","US","http://www.edmunds.com",UsedMilesUnder100)  
  
#a.  
  
print.cars <- function(object) {  
 header <- paste("Data:",object$title,"Country:",object$id,"SOURCE:",object$source,sep=" ")  
 cat(header,"\n\n")  
 str(object$data,0)  
}  
  
  
  
#Want to see all all the people who bought cars over or under 100 miles new and used  
  
#b.  
  
plot.cars <- function(object,...) {  
 df <- object$data  
  
   
 zipcode\_tran <- df %>% mutate(zip = zip\_bought) %>% select(zip) %>% distinct()  
 library(zipcode)  
 data(zipcode)  
   
 dfout <- df %>% mutate(zip = as.factor(zip\_bought),year\_purchased = as.numeric( format( as.Date( date\_sold, origin='1900-1-1'), '%Y')))  
   
 dfout\_latlon = merge(dfout,zipcode, by.x='zip', by.y='zip')   
  
   
 p <- qmplot(longitude, latitude, data = dfout\_latlon, colour = I("red"),size = I(2), darken = .1) + facet\_wrap(~ year\_purchased, ncol = 1) + expand\_limits() + theme\_minimal()  
  
   
 plot(p)  
 }  
  
#c.  
summary.cars <- function(object) {  
   
 df <- object$data  
 sum <- df %>% summarise(mean\_price\_purchased = mean(price\_bought , na.rm = TRUE),lowest\_price\_purchased = min(price\_bought , na.rm = TRUE),highest\_price\_purchased = max(price\_bought , na.rm = TRUE) )  
   
 print(sum)  
}

Answers

#a.Create printout of the data created to look at purchases of new cars bought with over 100 miles and used cars with under 100 miles.  
  
 print(PurchasedNewOver100)

## Data: New Cars Purchased with Miles Over 100 in Country: US SOURCE: http://www.edmunds.com   
##   
## Classes 'data.table' and 'data.frame': 11182 obs. of 17 variables:  
## - attr(\*, ".internal.selfref")=<externalptr>

print(PurchasedUsedUnder100)

## Data: Used Cars Purchased with Miles Under 100 in Country: US SOURCE: http://www.edmunds.com   
##   
## Classes 'data.table' and 'data.frame': 269 obs. of 17 variables:  
## - attr(\*, ".internal.selfref")=<externalptr>

#b.Summarize the total purchase amounts .  
   
 summary(PurchasedNewOver100)

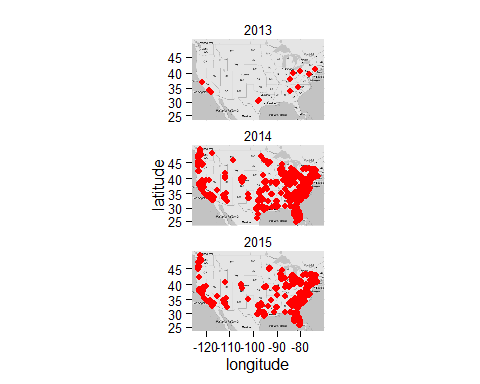
## mean\_price\_purchased lowest\_price\_purchased highest\_price\_purchased  
## 1: 33658.02 0 190000

summary(PurchasedUsedUnder100)

## mean\_price\_purchased lowest\_price\_purchased highest\_price\_purchased  
## 1: 32647.36 0 117690

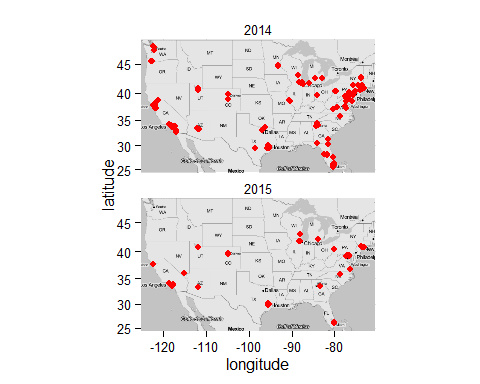
#c.Plot using ggmap to see where purchases of new cars bought with over 100 miles and used cars with under 100 miles were.  
   
 plot(PurchasedNewOver100)

## Using zoom = 4...



plot(PurchasedUsedUnder100)

## Using zoom = 4...



JSON/XML  
  
 e. Create a data frame based on top models bought in each year and pull important consummer rating data?