Project Report

Kota Revanth

2017-08-03

DA-5020: Collect, Store, Retrieve Data

CRN 51974.

Full Summer

```r
library(tidyverse)
library(rvest)
library(stringr)
library(RSQLite)
```

Problem Description:

Unlike New cars, Used cars does not have a Standard Price at which they are sold. The Prices of used cars are highly Unpredictable and depend on various factors like the number of miles they travelled before being listed on a website for sale, Their initial year of release, the type of make, location etc.,

As, we all want a best choice of car at an optimal price, In this project I Scraped used cars from www.cargurus.com and performed analysis on how the car prices are varying on various factors like miles the car travelled before being listed for sale, make of car, the initial release year of a car

Some of the Questions I would like to answer from this Project are:

1. What are the most frequently listed car makes?

2. In General, How the Price of Cars are varying with car mileage? (The distance travelled by a car before being listed for sale)

3. In General, How the car prices are varying along the years?

4. Comparing Average Prices of Different Makes:

I have written two functions too to compare different make of cars for evaluating them.

Methodology:

1. Built a Web Scraper using "rvest" package to collect all the details of cars listed on "www.cargurus.com" website and cleaned the data using "stringr" Package

Cargurus is one of the most popular websites with a number of used cars listed based on location. Prospective car buyers can find insights about used cars before making a choice.

Variables of interest I, scraped from the website are

1. Car_years : The year the car was actually launched by car makers
2. Car_make : The Company which Produced the car.
3. car_model : The name of the car
4. Car_price : Price at which the car is listed in the website
5. Car_mileage : total number of miles the car travelled before being listed in the website by the owner for sale

2. Stored the Scraped and cleaned data in a relational “SQLite”" database using “RSQLite” Package and queried the database using SQL to extract the information necessary for analysis

3. Used “ggplots” to answer the questions mentioned above

Data Scraping and Cleaning :

Scraping information from a single page of a website.

```
get_one_page_information <- function(page){
  cars_url<- "https://www.cargurus.com/Cars/spt_cheap_cars-Boston_L15690?page=%s"
  cars_url <- sprintf(cars_url, page)
  cars <- read_html(cars_url)

  v1 <- tibble(
    car_years = cars %>%
      html_nodes(".cg-dealFinder-result-model") %>%
      html_text() %>%
      str_replace_all("\n","") %>%
      str_extract("(.*),") %>%
      str_replace_all("Used\\sCars(.*)","") %>%
      str_replace_all("^\\s+","") %>%
      str_replace_all("\\s+$","") %>%
      str_replace_all("\\$","") %>%
      str_extract_all("^[0-9]+\\s") %>%
      str_replace_all("\\s$",""),

    car_make = cars %>%
      html_nodes(".cg-dealFinder-result-model") %>%
      html_text() %>%
      str_replace_all("\n","") %>%
      str_extract("(.*),") %>%
      str_replace_all("Used\\sCars(.*)","") %>%
      str_replace_all("^\\s+","") %>%
      str_replace_all("\\s+$","") %>%
      str_replace_all("\\$","") %>%
      str_extract_all("^[0-9]+\\s[A-Za-z-]+\\s") %>%
      str_replace_all("^[0-9]+\\s","") %>%
      str_replace_all("\\s$",""),

    car_model = cars %>%
      html_nodes(".cg-dealFinder-result-model") %>%
      html_text() %>%
      str_replace_all("\n","") %>%
      str_extract("(.*),") %>%
      str_replace_all("Used\\sCars(.*)","") %>%
      str_replace_all("^\\s+","") %>%
      str_replace_all("\\s+$","") %>%
      str_replace_all("\\$","") %>%
      str_extract_all("^[0-9]+\\s[A-Za-z-]+\\s(.*)") %>%
      str_replace_all("^[0-9]+\\s[A-Za-z-]+","") %>%
      str_replace_all("\\s$",""),

    car_price = cars %>%
      html_nodes(".cg-dealFinder-result-stats") %>%
      html_text() %>%
      str_replace_all("\n","") %>%
      str_extract_all("\\$[0-9,]+") %>%
      str_replace_all("\\$","") %>%
      str_replace_all(",","") %>%
      as.double(),

    car_mileage = cars %>%
      html_nodes(".cg-dealFinder-result-stats") %>%
      html_text() %>%
      str_replace_all("\n","") %>%
      str_extract_all("[(M|m)ileage:]+\\s[0-9,]+") %>%
      str_replace_all("[(M|m)ileage:]+\\s","") %>%
      str_replace_all(",","") %>%
      as.double()

  )

}
```

The above function does both, scraping and cleaning of data from a selected page of the website according to your choice.
The above function to scrape and clean the data reduces effort in getting clean data as it performs both scraping and cleaning of data altogether. The Page argument in the function takes a positive non zero integer value to scrape from a page specified by the user
Here, I added `as.double()` funtion for car_mileage and car_price columns to parse them into numeric formats as it would help me in data visualization

Steps Performed while scraping information from a single page:

1. *Using "rvest" package and selector gadget, could access the necessary information from webpage and scrape the required part of information on the website.*

Data Cleaning:

1. *Data cleaning involved seperating car_years, car_make, car_model from the text exctracted from the web page into three different columns and then replacing unnecessary information.*
2. *Conversion of car_price and car_years from character to numeric forms*

3. Remove unnecessary information from the scraped column and extracted the necessary part

4. the scraped columns are bound together by collecting them as a tibble.

Scraping information from multiple pages of website:

```
get_all_car_results <- function(page,pages,start){
  multiple_page <- data.frame()
  for ( page in start:pages){
    single_page <- get_one_page_information(page)
    multiple_page = rbind(single_page,multiple_page)
    Sys.sleep(0.5)
  }

  multiple_page
}
```

Here, by running a for loop we could append all the information scraped from a single page for different values of pages resulted from "get_one_page_information" funtion, there by producing a dataframe that contains information scraped from multiple pages. Here, the function takes "page","pages", "start" as arguments to specify from which page we like to collect the data and the number of pages we would like to collect the data.

Data Collection:

```
cars_scraped <- get_all_car_results(page = 1,pages = 50, start = 1)
```

Here, the data which is scraped is stored by the name car_scraped which is a tibble with the following variables:
1. Car_years : The year the car was actually launched by car makers
2. Car_make : The Company which Produced the car.
3. car_model : The name of the car
4. Car_price : Price at which the car is listed in the website
5. Car_mileage : total number of miles the car travelled before being listed in the website by the owner for sale
Here the funtion"get_all_car_results" scraps information of all cars listed on website for 50 pages

Data Storage:

```
used_cars <- dbConnect(SQLite(), dbname = "Used_cars")
```

```
dbRemoveTable(used_cars,"scraped_used_cars")
```

```
dbWriteTable(conn = used_cars, name = "Scraped_used_cars", value = cars_scraped, row.names = FALSE)
```

Here, I used A relational data base to store the collected data, as the data in all columns are related to each other. And also to maintain a uniformity in the structure of data.
That is all the rows and columns in the table are atomic and consistent along the table.

Excerpt of the data stored and retrieved from SQL Data base:

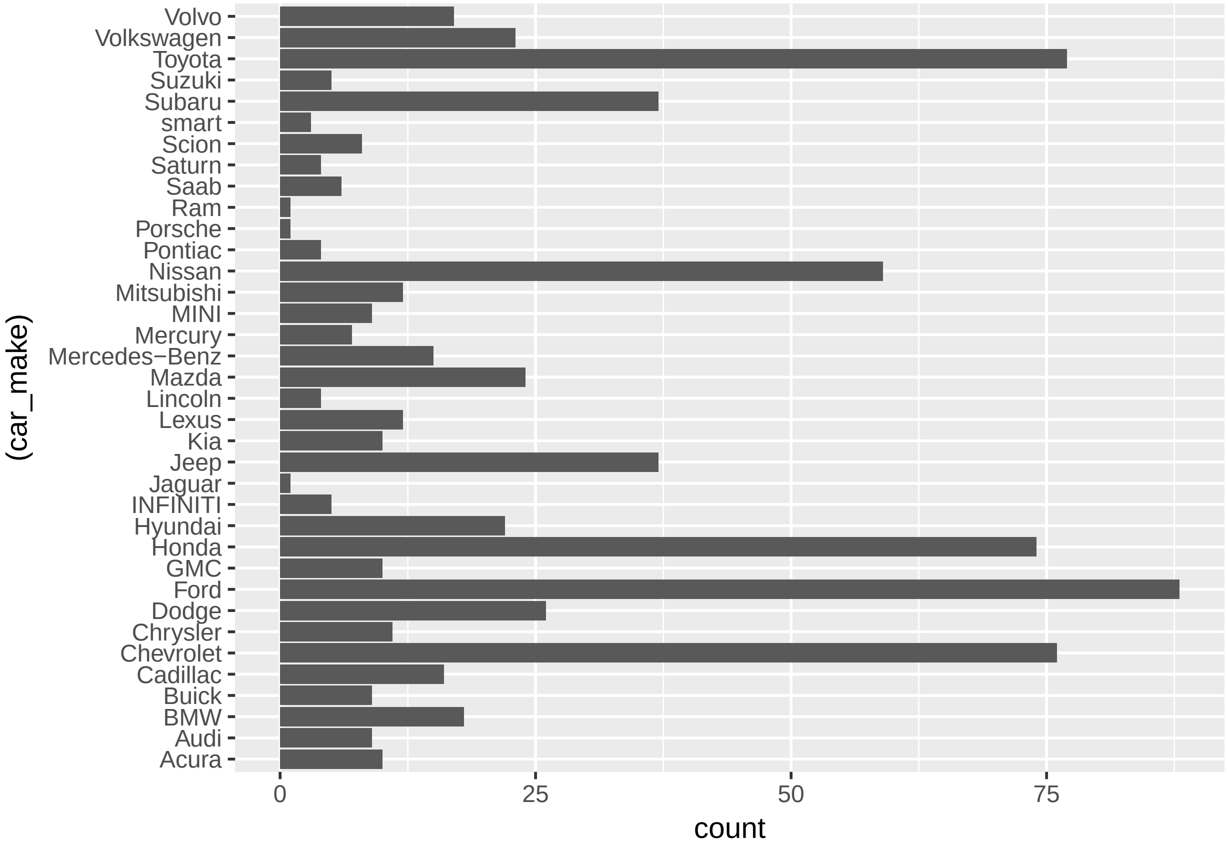
```r
head(dbReadTable(used_cars,"scraped_used_cars"), 10)
```

```
##    car_years  car_make                  car_model car_price car_mileage
## 1       2009      Ford                   Focus SE      5300       83394
## 2       2010    Nissan     Versa 1.8 SL Hatchback      5385       91218
## 3       2008      Jeep  Grand Cherokee Laredo 4WD      4995      167975
## 4       2007     Honda            Accord Coupe EX      5300      119000
## 5       2005     Honda                CR-V SE AWD      5499      157000
## 6       2010      Ford                   Focus SE      4750       93000
## 7       2008     Dodge            Charger SXT AWD      5495      150151
## 8       2005 Chevrolet         TrailBlazer LT 4WD      2999      166997
## 9       2005   Hyundai         XG350 4 Dr L Sedan      3995       73000
## 10      2009      MINI                Cooper Base      5997      124242
```

Data retrieval by querying DataBase and Data Visualization

1. What are the most frequently listed car makes?

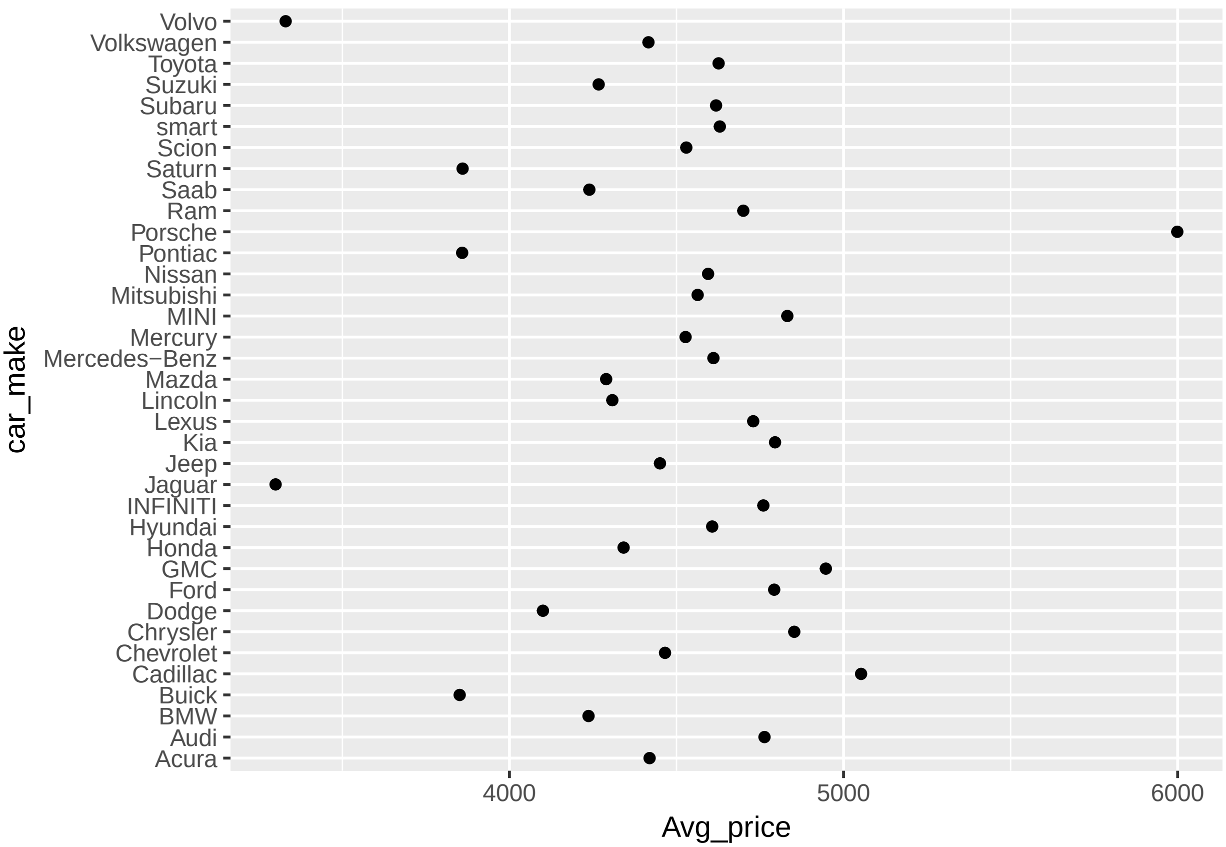
```r
car_make <- dbGetQuery(used_cars, "select [car_make] from scraped_used_cars")

ggplot(data = car_make)+
  geom_bar(mapping = aes(x = (car_make)))+
  coord_flip()
```



From the chart we can see that Ford makes cars are the cars with highest listings followed by Toyota, Chevrolet, Honda and Nissan

2. Comparing Average Prices of Different Makes:

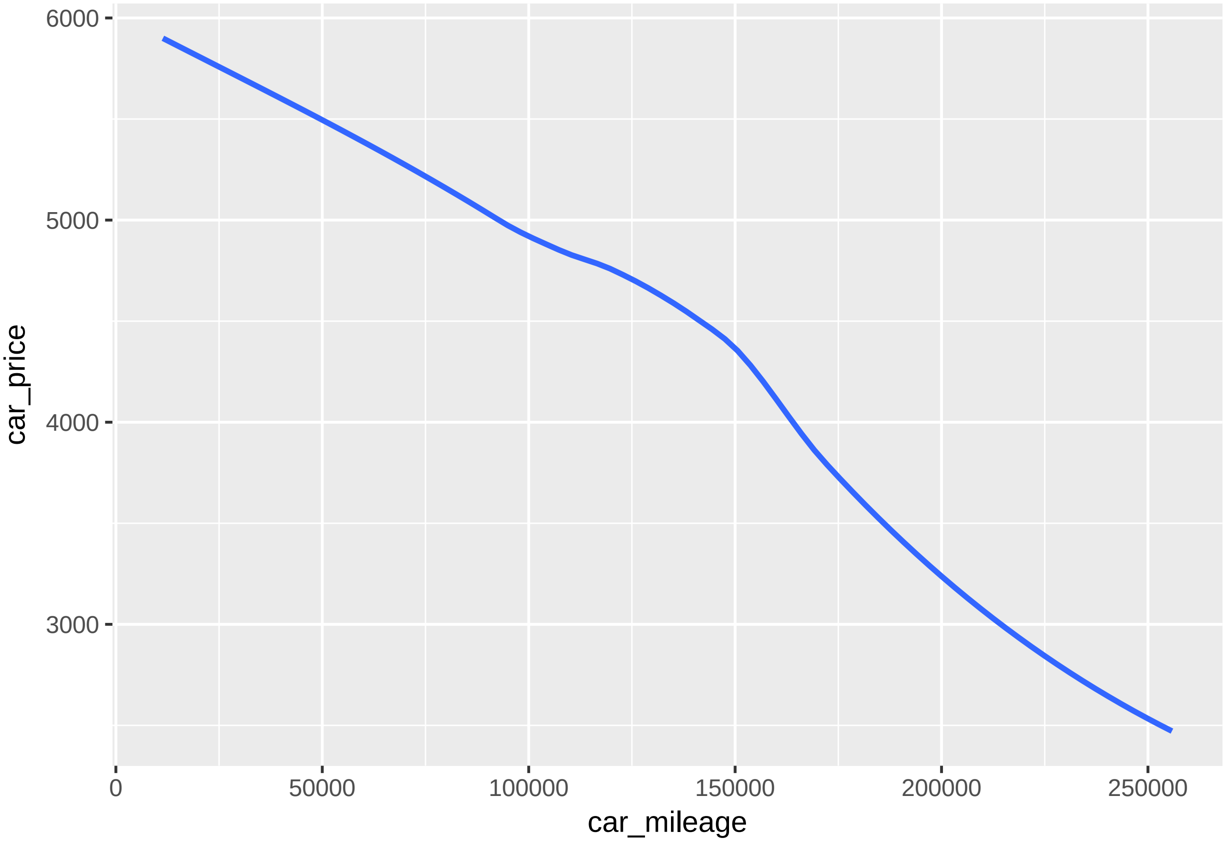
```
Make_AVGprice_Comparision <- dbGetQuery(used_cars, "select [car_make],
                                        avg([car_price]) as [Avg_price] from
                                        scraped_used_cars
                                        group by [car_make]")

ggplot(data = Make_AVGprice_Comparision)+
  geom_point(mapping = aes(x = car_make, y = Avg_price)) +
  coord_flip()
```



Here we can see that prices of maximum number of used cars vary between a range of 4000 to 6000 dollars From the graph we can see that though Ford, Toyota, Chevrolet, Honda and Nissan are the most frequently listed makes, their prices are almost in the same range. Hence we can say that there is no deflation of price because of most number of listings. That the the total number of listings of a car make does not effect it's price

3. In General, Relation Showing how the car_price is varying with car mileage(The distance a car travelled before being listed in the website:

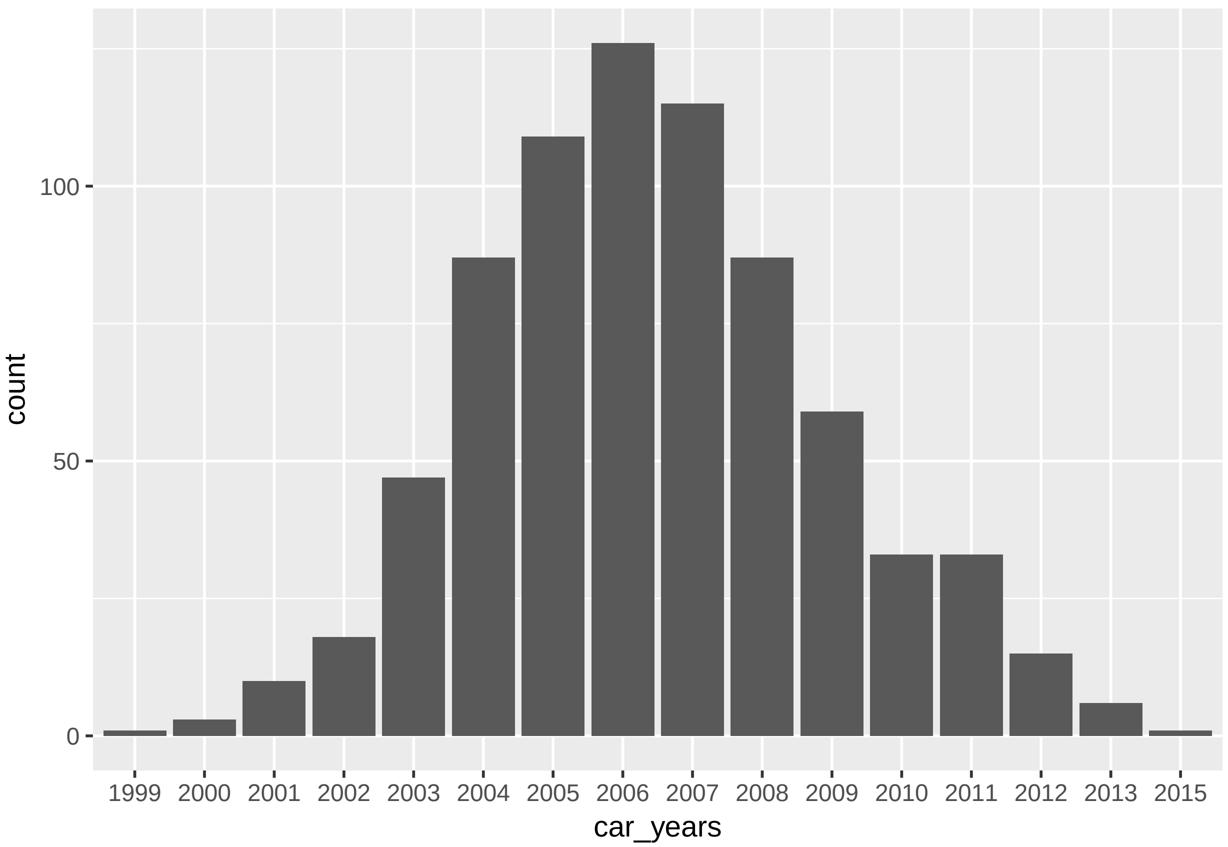
```
miles_price <- dbGetQuery(used_cars, "select [car_mileage], [car_price]
                          from scraped_used_cars")
ggplot(data = miles_price)+
  geom_smooth(mapping = aes(x = car_mileage, y = car_price), se = FALSE)
```



From the Graph, we can see that more the car travels before being listed, the lesser its price will be. That is the price at which the car is listed and the total miles it travelled before being listed are inversely proportional to each other.
In common terms we can say that, if a car travels more distance before being listed for sale, it will be sold at a very lesser price.
If a car travels less distance before it is listed for sale, then it will be sold at a relatively good price
The above visualization can also help us in predicting what price we can expect for a used car that travelled specific number of miles.
For example, we can say that if a car 50,000 miles then we can expect it's price to be above 5000 dollars when listed for sale. If a car travelled more that 250,000 miles we cannot expect a price of more than 3000 dollars

4. Which year cars are the most frequently listed cars?

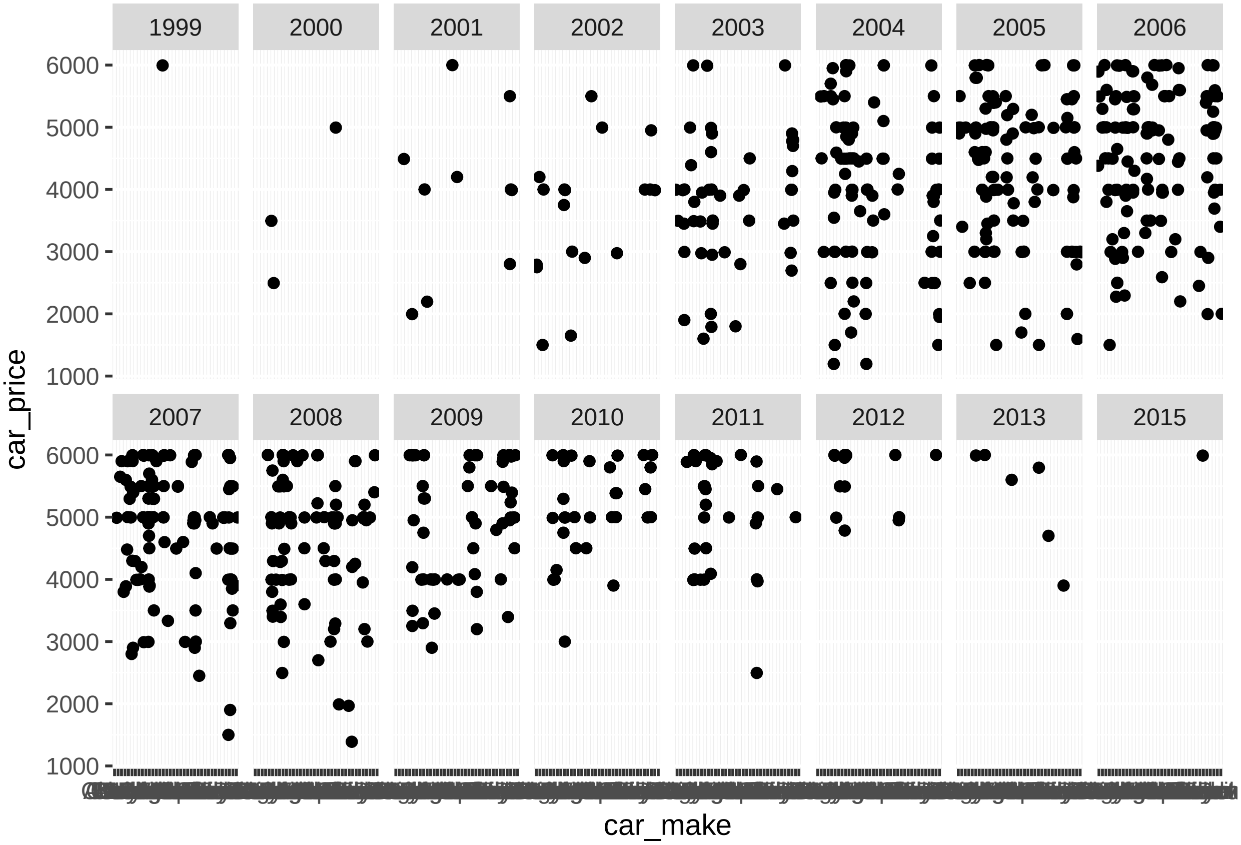
```
years <- dbGetQuery(used_cars, "select [car_years]from scraped_used_cars")

ggplot(data = years)+
  geom_bar(mapping = aes(x = car_years))
```



Here we can see that year"2006" has the highest listings.
Here we can see that as we move from 2015 to 2006 the cars listed for sale are increasing. this can be interpreted as "More the car gets old, more likely are people ready to sell"
From year 2006 to 1998, we can see that the number of listing decrease this can be interpreted as "Older the car gets, more likely that the car is either sold or car is given as scrap."

5. In General How the Prices of Different makes of cars are varying along the years:

```
price_variation_along_years <- dbGetQuery(used_cars, "select [car_years],
                                          [car_make], [car_price] from
                                          scraped_used_cars")

ggplot(data = price_variation_along_years)+
  geom_point(mapping = aes(x = car_make, y = car_price), position = 'jitter')+
  facet_wrap(~car_years, nrow = 2)
```



Here we can see that as 2006, 2007 are most frequently listed cars there are wide range of cars which are available from a range of $2000 to $6000
From years "2006" to years "2015" we can see a clear trend in increase of prices.
We can interpret the above result as "Older the car greater the chance that the price will be lesser"
There are few outliers after year"2006" this can be because of older costly cars like "Porsche", "cadillac" etc.,

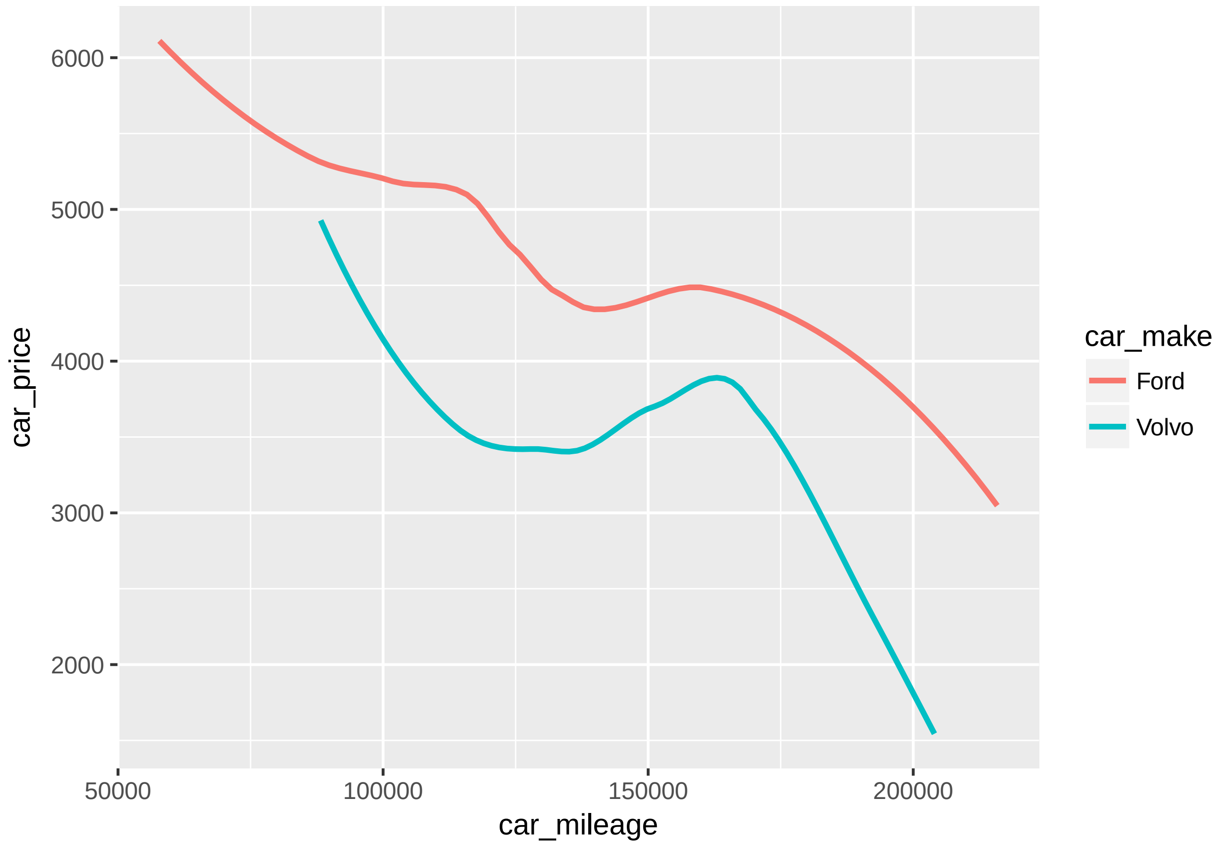
6. Specific Comparision between Two car Makes to evaluate and choose a better option:

```
compare_two_makes <- function(make1,make2){

  f1 <- "select [car_make],[car_price],
  [car_mileage] from scraped_used_cars
  where [car_make] = '%s' or [car_make] = '%s'"

  f2 <- sprintf(f1, make1, make2)

  f3 <- dbGetQuery(used_cars, f2)

  ggplot(data = f3)+
    geom_smooth(mapping = aes(x = car_mileage, y = car_price, color = car_make), se = FALSE)

}
```

Here the above function is used to evaluate two makes based on their listed prices and the miles they travelled

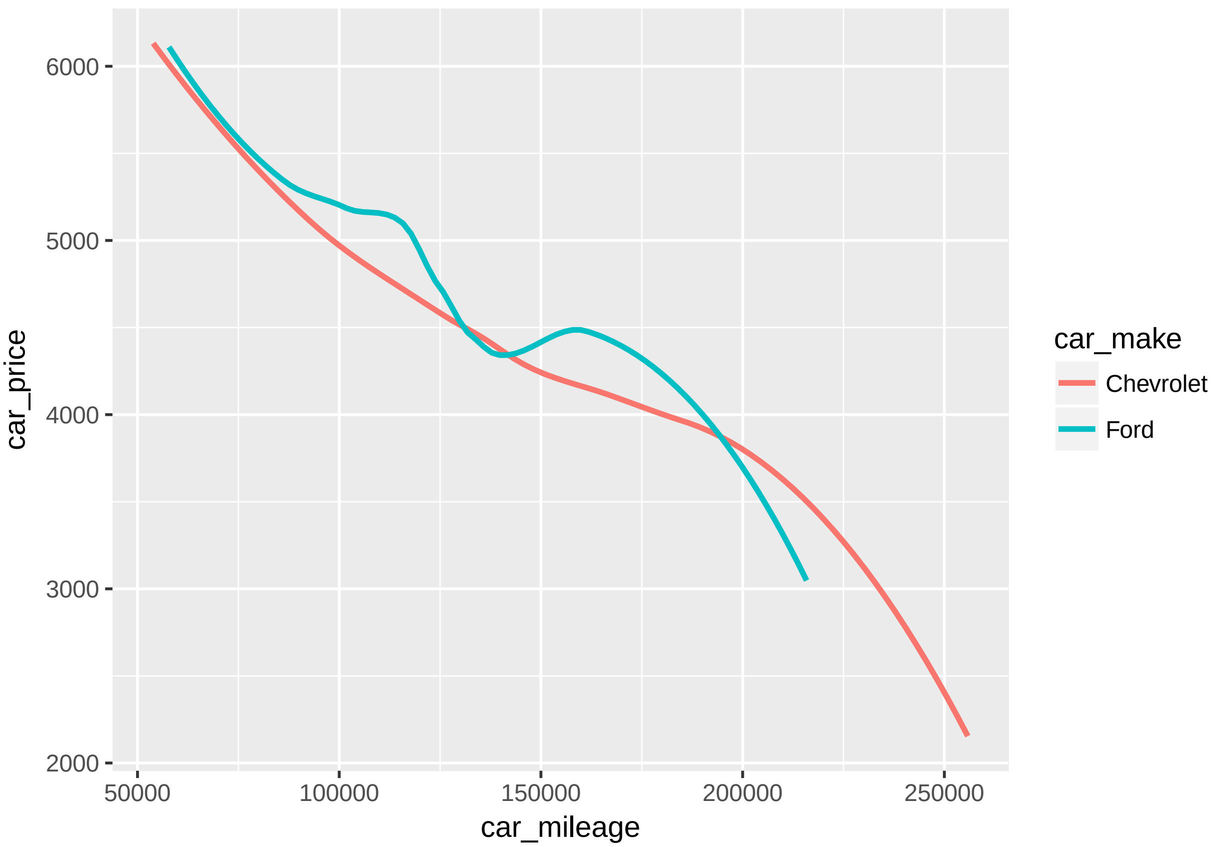
before being listed on the website.
Here, make1, and make2 arguments take the names of makes we would like to compare

```
compare_two_makes(make1 = 'Volvo', make2 = 'Ford')
```



Based on the above visualization, we can say that Ford cars are more costly than volvo cars for the same miles travelled by them.

```
compare_two_makes(make1 = 'Ford', make2 = 'Chevrolet')
```



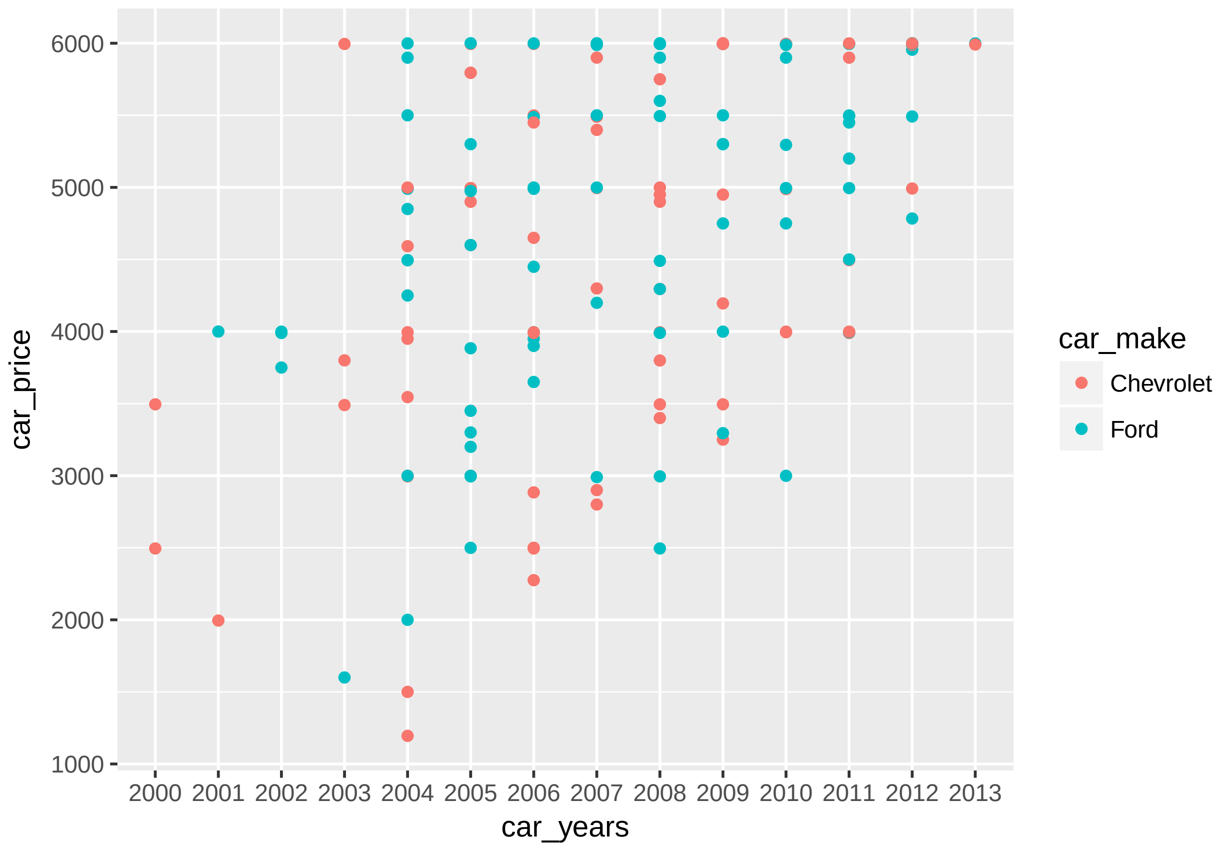
Here we can say that the price of Ford made cars are almost similar to chevrolet cars for the total number of miles they travelled before being listed in the website.

7. Price we range expected for two make of cars along years:

```
make_price_comprsn_along_time <- function(make1,make2){

  p1 <- "select [car_make],[car_price],
  [car_years] from scraped_used_cars
  where [car_make] = '%s' or [car_make] = '%s'"

  p2 <- sprintf(p1, make1, make2)

  p3 <- dbGetQuery(used_cars, p2)

  ggplot(data = p3)+
    geom_point(mapping = aes(x = car_years, y = car_price, color = car_make))

}
```

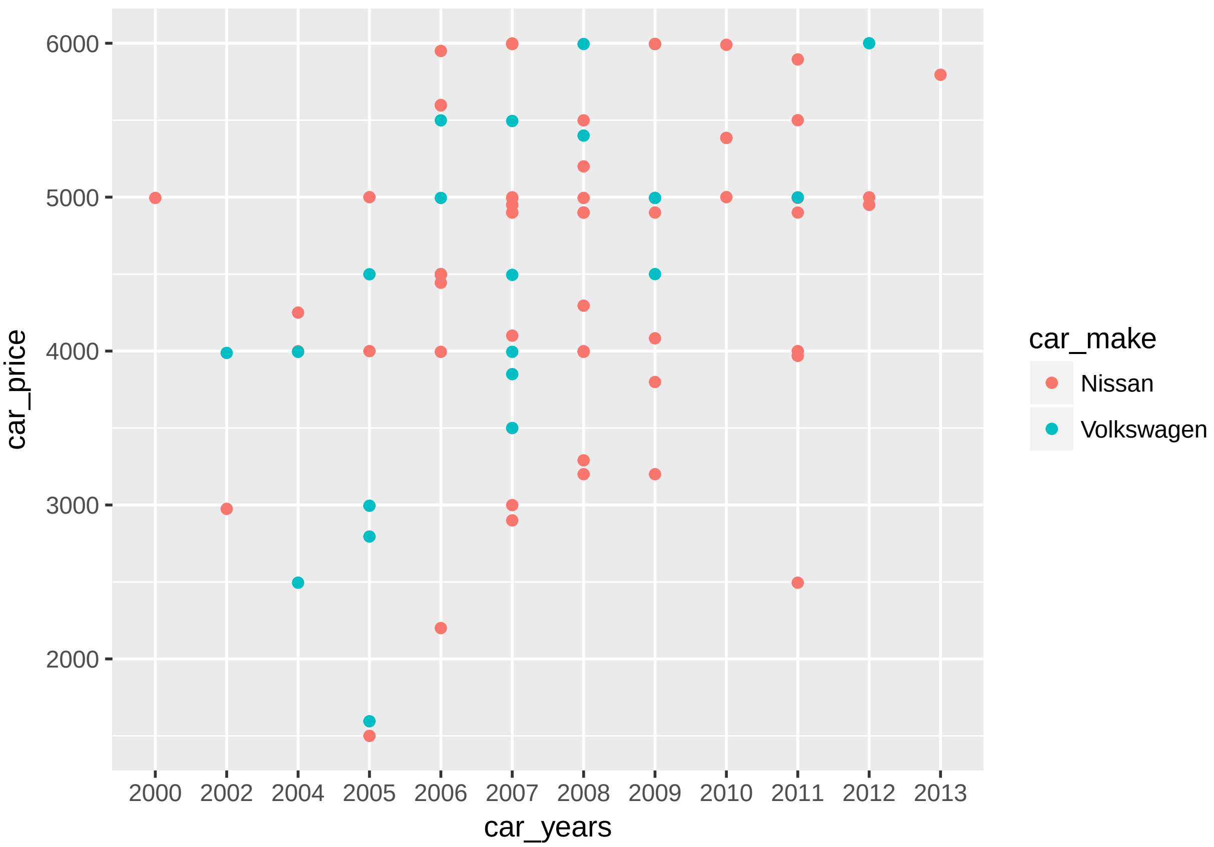
Here the above function can be used to evalute or predict an estimate price range for a make along different years

```
make_price_comprsn_along_time(make1 = 'Ford', make2 = 'Chevrolet')
```



Here from the pattern we can say that Ford make cars are slightly costly than Chevrolet make cars as their count is slightly higher at price = $6000
For Year 2002, we can see that the price range for chevrolet is between 5000 dollars to 3500 dollars, where as the price range for Ford cars is around 4000 dollars
Like wise, we can see that in general the in each year the Chevrolet cars are relatively cheaper than Ford cars. We can see a pattern that as we move towards year 2013, from year 2004 there is decrease in range of prices and increase in price of each makes which can be interpret as: "New cars are sold at greater prices than the Old cars"

```r
make_price_comprsn_along_time(make1 = 'Volkswagen', make2 = 'Nissan')
```



Like the above, we can say that Nissan make cars are costly than Volkswagen cars
As we move from year 2005 to year 2013, we can see that there is an increase in prices of cars

Issues Faced During the course of Project And Methods employed to resolve them:

Issue1. I Had to switch to "www.cargurus.com" from "www.enterprisecarsales.com" as there are Pagination issues with enterprise car sales website. Inspite of changing the values of pages, I was getting the same information.
I could figure it out and resolve pagination issues with enterprise car sales by observing the change in url by using import.io when made an attempt to scraped multiple pages of website. Import.io is such a beautiful tool not only for scraping data from multiple pages, but also for helping me in constructing the url manually in the function "get_one_page_information" used to scrape data from single page. Using Import.io, I could construct the url to scrape cars in and around Boston
Issue2 Another issue i faced during analysis phase is, after scraping and collecting car_price and car_mileage columns, they were in character format. When I was trying to visualize them, I could not get any output
So, to resolve the issue, I used `guess_parser` *function to get to know about their data type, and Parsed them in them into numeric format while scraping itself which helped in data visualization to predict trends*

Scope and Future work:

In this I could scrape date only in and around "Boston". We can build a scraper that can be used to scrape used cars listing across various cities in U.S. to see if location does effect car prices?

The above analysis can not only be used for evaluating a used car while purchasing, but also while selling a used car
We can also construct a mathematical model to Predict car price of a used car based on it's year of make and the total number of miles it travelled before being listed for sale.

Conclusion And Leason's Learnt:

1. From the above analysis in general we can say that older the car or greater the car miles travelled, Price will be lower.
By writting a function we can both scrape and clean data altogether resulting in a clean data at the end 2. Really enjoyed writing functions. They really made work easy while scraping and comparing two different makes of cars. During comparision of cars based on make, functions really made comparision easy by just providing make names, avoiding a very tidious work of filtering querying databased every time for a specific make.

References:

1. Website used for scraping used cars: www.cargurus.com
2. R for data sciences by : Hadley wickham for data cleaning
3. Data Collection, Integration And Analysis by Kathleen Durant for Data Scraping Using "rvest" Package and Storing data in SQLite database
4.regexr.com for testing the regular expression while cleaning the scraped data
Import.io for helping with pagination issues and constructing url to scrape manually