Project

Ranjini

21/12/2020

Task 1: Analysis

The data is loaded using the following command.

```
cardata<-read.csv("~/Desktop/R-Programming/USA_cars_datasets.csv")
```

The following command calculates of Frequency of cars with its brand name

```
table(cardata$brand)
```

```
##
##           acura            audi             bmw           buick        cadillac
##               3               4              17              13              10
##       chevrolet        chrysler           dodge            ford             gmc
##             297              18             432            1235              42
## harley-davidson       heartland           honda         hyundai        infiniti
##               1               5              12              15              12
##          jaguar            jeep             kia            land           lexus
##               1              30              13               4               2
##         lincoln        maserati           mazda   mercedes-benz          nissan
##               2               1               2              10             312
##       peterbilt             ram          toyota
##               4               1               1
```

We can notice that car of the brand Ford is highly used by people(1235).

The following command calculates the number of observations having car price greater than 10000 dollars

```
table(cardata$price>10000.0)
```

```
##
## FALSE  TRUE
##   615  1884
```

which says 1884 observations with price >10000.0 615 observations with price <=10000.0

The following command calculates the average price and average mileage(miles travelled) of cars

```
apply(cardata[2],2,mean)
```

```
##    price
## 18767.67
```

```
apply(cardata[7],2,mean)
```

```
##  mileage
## 52298.69
```

The following command calculates the standard deviation and quautiles of the car price

```
sd(cardata$price)
```

```
## [1] 12116.09
```

```
quantile(cardata$price)
```

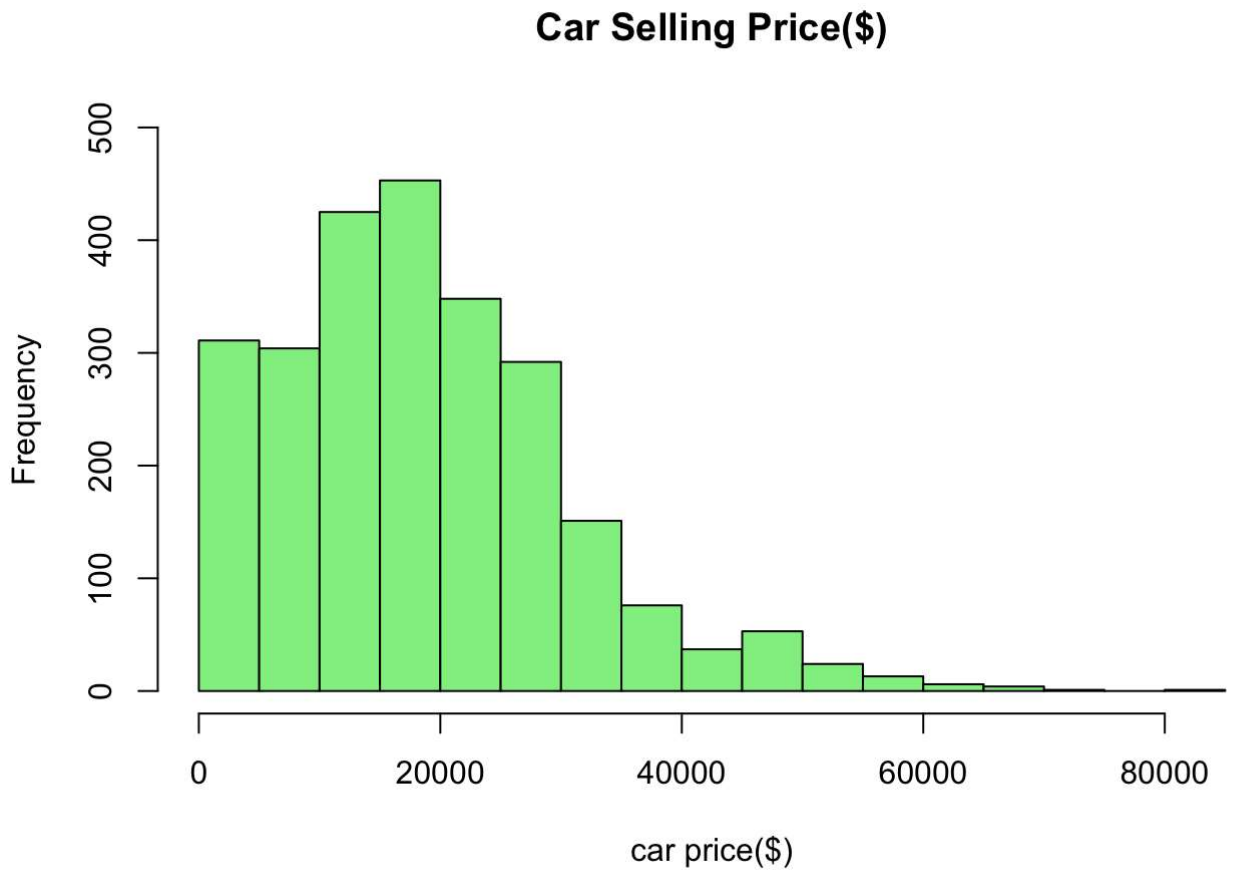
```
##      0%     25%     50%     75%    100%
##     0.0 10200.0 16900.0 25555.5 84900.0
```

```
# Command showing 10th and 90th percentiles
quantile(cardata$price,c(0.1,0.9))
```

```
##   10%   90%
##  4140 33720
```

The following command produces a histogram of car selling price

```
hist(cardata$price,main="Car Selling Price($)",xlab="car price($)",col="light green",ylim = c(0,
500))
```



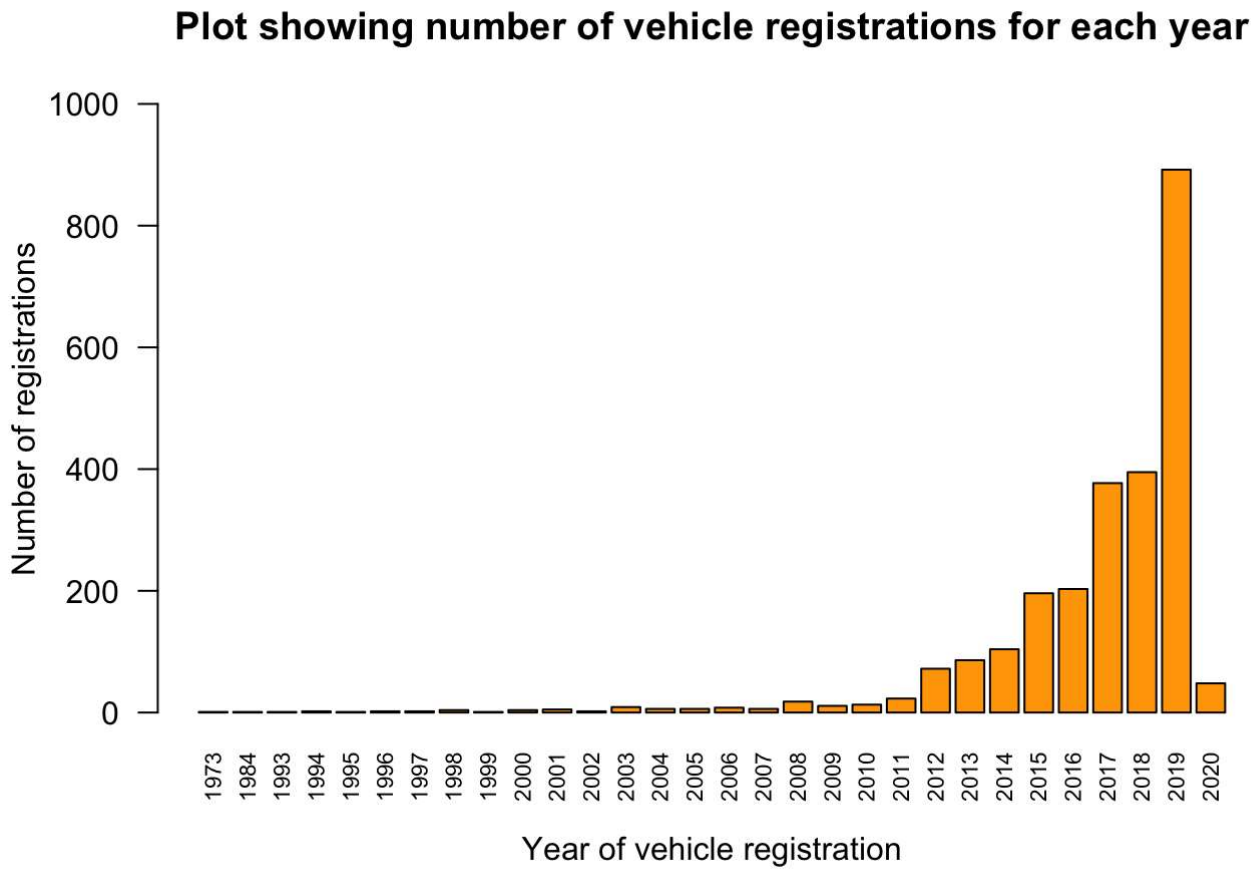
We can see that Selling price of majority of the cars is less than 20000$

The following command produces a table of the number of vehicle registrations for each year and a barplot.

```
tab1<-table(cardata$year)
tab1
```

```
##
## 1973 1984 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006
##    1    1    1    2    1    2    2    4    1    4    5    2    9    6    6    8
## 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020
##    6   18   11   13   23   72   86  104  196  203  377  395  892   48
```

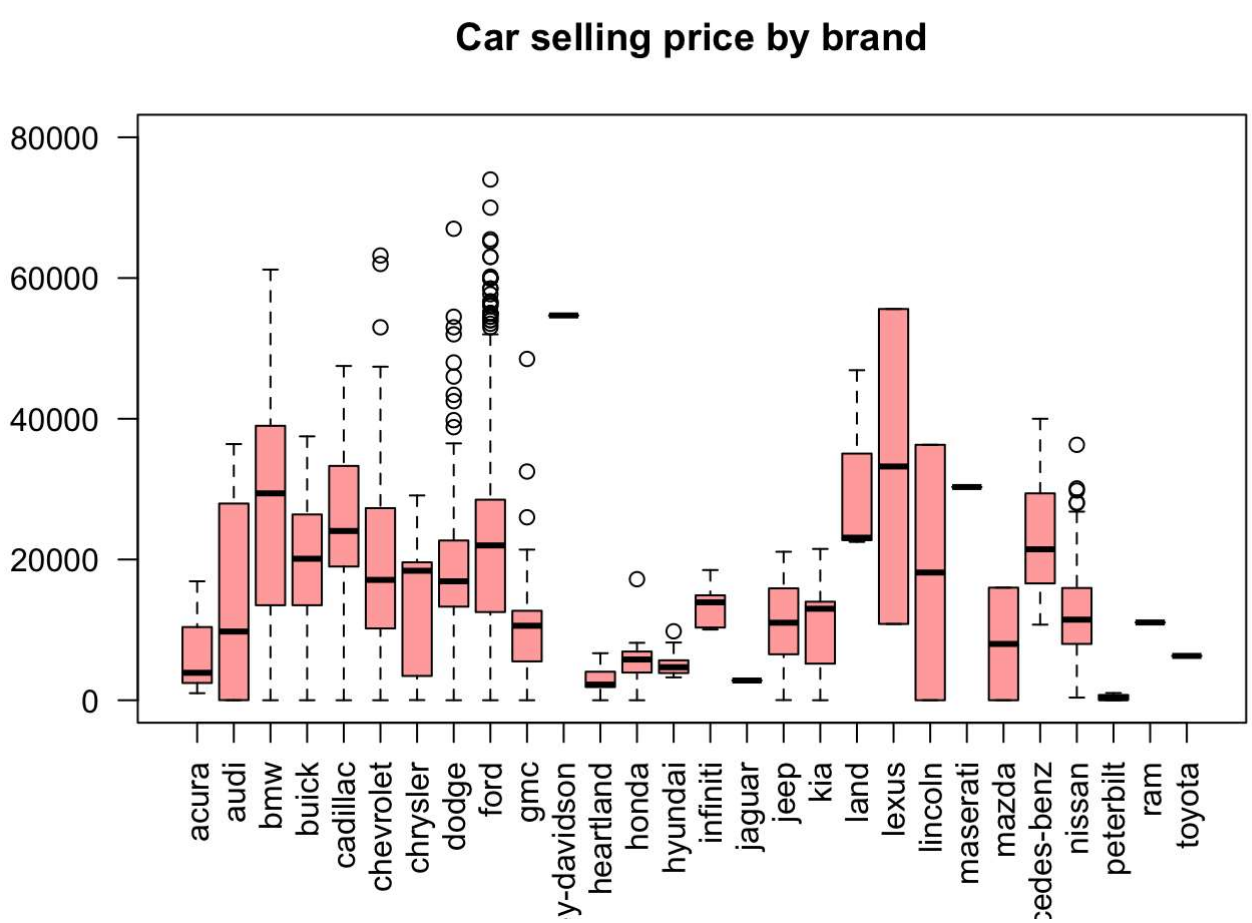
```
barplot(tab1, xlab ="Year of vehicle registration",ylab="Number of registrations",las=2 ,cex.nam
es=0.7,
        main ="Plot showing number of vehicle registrations for each year",ylim=c(0,1000),col="o
range")
```



We can see that majority of the vehicles are registered in the year 2019.

The following command produces a boxplot of car selling price for each car brand.

```
splitdat<-split(cardata$price,cardata$brand)
boxplot(splitdat,las=2 ,cex.names=1.0,
        main="Car selling price by brand",
       ylim=c(0,80000),col=rgb(1,0,0,alpha = 0.4))
```



We install tidyverse library as ggplot2 is included in the tidyverse package.We can produce a ggplot to see car status.

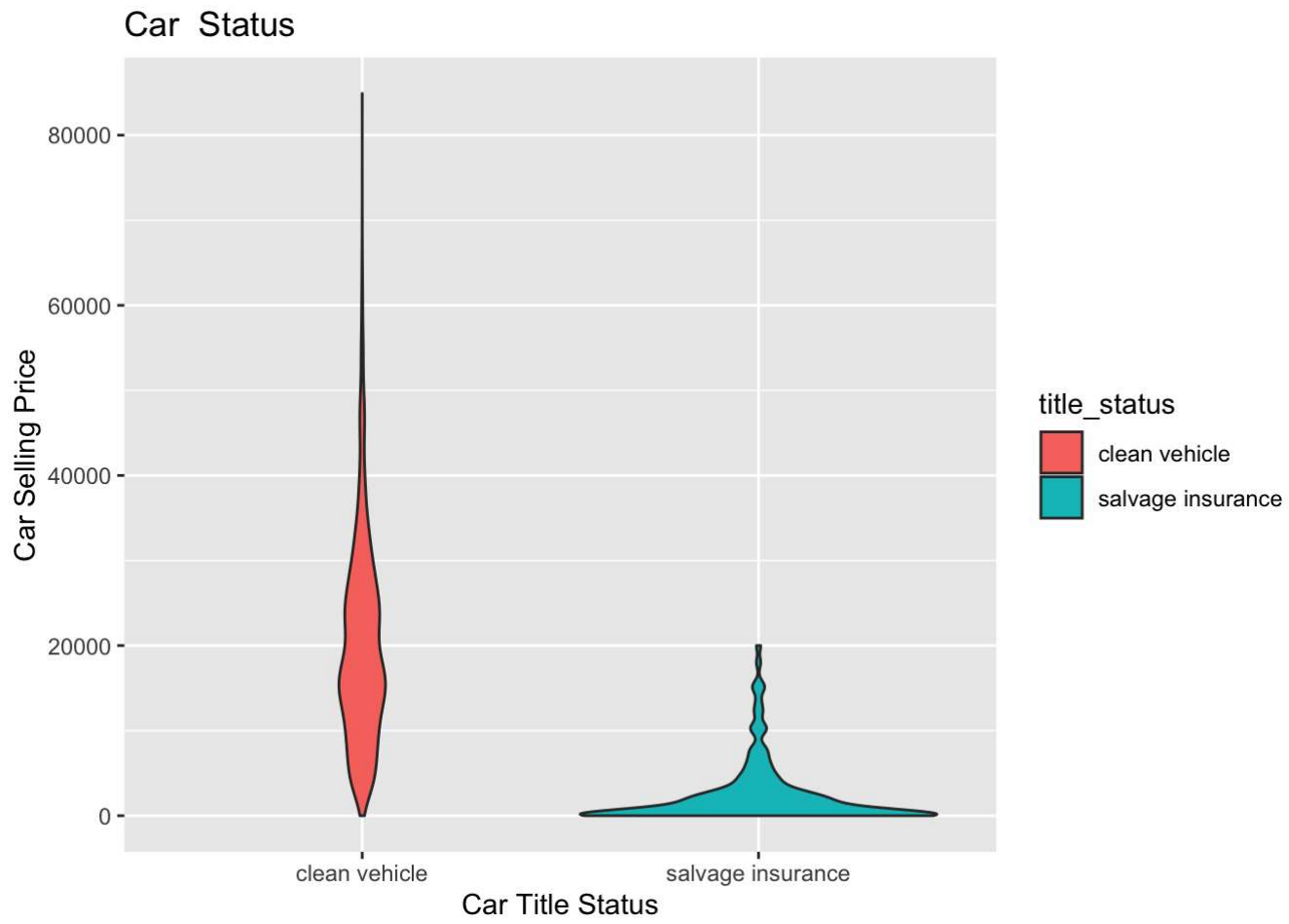
```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──
```

```
## ✓ ggplot2 3.3.2     ✓ purrr   0.3.4
## ✓ tibble  3.0.4     ✓ dplyr   1.0.2
## ✓ tidyr   1.1.2     ✓ stringr 1.4.0
## ✓ readr   1.4.0     ✓ forcats 0.5.0
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
ggplot(data=cardata,aes(x=title_status,y=price,fill=title_status))+ggtitle("Car  Status")+geom_v
iolin(scale="area")+labs(x="Car Title Status",y="Car Selling Price")
```

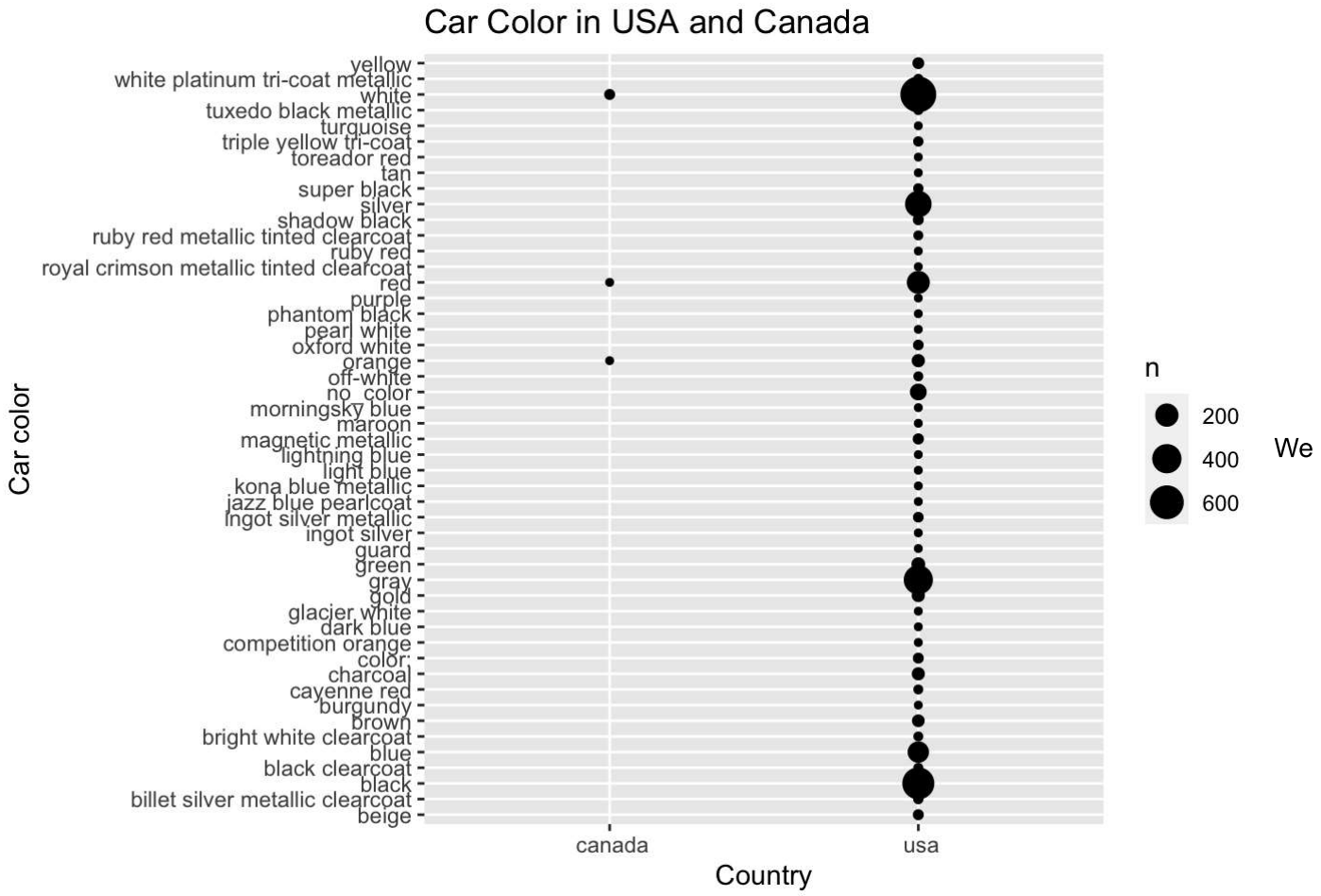


Clean Vehicle:A clean title is one you would receive in most cases when you purchase a vehicle. A brand new vehicle has a clean title and most pre-owned vehicles that can be driven safely and are insurable. Salvage insurance:A salvage title is given when a vehicle is no longer drivable.

Here the salvage insurance is more when the price of the car is low(0-20000)but on the other hand we can see that the clean title is lesser when the price of the car is 40000($) or above.

The following command will tell us which car color is mostly preferred in USA and Canada.

```
ggplot(data=cardata,aes(country,color),width=500,height=4000)+geom_count()+ggtitle("Car Color in USA and Canada")+labs(x="Country",y="Car color")
```



can say that the most preferred car color is White in USA and Canada.And USA has more number of White Color cars than Canada.

The The following command will give us summary of USA car dataset

```
summary(cardata)
```

```
##        X              price          brand              model
##  Min.   :   0.0   Min.   :    0   Length:2499        Length:2499
##  1st Qu.: 624.5   1st Qu.:10200   Class :character   Class :character
##  Median :1249.0   Median :16900   Mode  :character   Mode  :character
##  Mean   :1249.0   Mean   :18768
##  3rd Qu.:1873.5   3rd Qu.:25556
##  Max.   :2498.0   Max.   :84900
##       year      title_status          mileage           color
##  Min.   :1973   Length:2499        Min.   :      0   Length:2499
##  1st Qu.:2016   Class :character   1st Qu.:  21466   Class :character
##  Median :2018   Mode  :character   Median :  35365   Mode  :character
##  Mean   :2017                      Mean   :  52299
##  3rd Qu.:2019                      3rd Qu.:  63472
##  Max.   :2020                      Max.   :1017936
##      vin                 lot               state             country
##  Length:2499        Min.   :159348797   Length:2499        Length:2499
##  Class :character   1st Qu.:167625331   Class :character   Class :character
##  Mode  :character   Median :167745058   Mode  :character   Mode  :character
##                     Mean   :167691389
##                     3rd Qu.:167779772
##                     Max.   :167805500
##   condition
##  Length:2499
##  Class :character
##  Mode  :character
##
##
##
```

Task 2:

#I have used PURRR package in the following section.Purrr package provides an excellent complete and consistent set of tools for working with functions and vectors.I have focused on functions provided by the purrr package of Purr package.

loading the purr library

```
library(purrr)
```

#Functions:

map()

The map functions transform their input by applying a function to each element and returning a vector the same length as the input. Eg1:splits a data frame into pieces, fit a model to each piece and summarise the data

```
cardata %>%
  split(.$brand) %>%
  map(~ lm(price ~ mileage, data = .)) %>%
  map(summary)
```

```
## $acura
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##   375   391   596
## -6499  3070  3429
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.040e+04  1.256e+04   1.624    0.351
## mileage     -1.091e-01  9.713e-02  -1.123    0.463
##
## Residual standard error: 7964 on 1 degrees of freedom
## Multiple R-squared:  0.5578, Adjusted R-squared:  0.1155
## F-statistic: 1.261 on 1 and 1 DF,  p-value: 0.4631
##
##
## $audi
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##      410      421      430      445
##   -47.35   976.48 -6732.03  5802.90
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.167e+04  6.806e+03   6.122   0.0257 *
## mileage     -2.344e-01  5.104e-02  -4.593   0.0443 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6323 on 2 degrees of freedom
## Multiple R-squared:  0.9134, Adjusted R-squared:  0.8701
## F-statistic:  21.1 on 1 and 2 DF,  p-value: 0.04428
##
##
## $bmw
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -23896 -12136  -2281   9782  25240
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.774e+04  5.117e+03   7.375 2.31e-06 ***
## mileage     -2.371e-01  7.673e-02  -3.090  0.00747 **
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14700 on 15 degrees of freedom
## Multiple R-squared:  0.3889, Adjusted R-squared:  0.3482
## F-statistic: 9.547 on 1 and 15 DF,  p-value: 0.007471
##
##
## $buick
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9947.7 -2917.4   189.6  4132.6 12571.8
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.746e+04  2.877e+03   9.545 1.17e-06 ***
## mileage     -2.042e-01  5.796e-02  -3.524  0.00476 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6693 on 11 degrees of freedom
## Multiple R-squared:  0.5303, Adjusted R-squared:  0.4876
## F-statistic: 12.42 on 1 and 11 DF,  p-value: 0.004763
##
##
## $cadillac
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -11014  -5383  -3232   2706  18050
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 38236.0758  6028.3250   6.343 0.000222 ***
## mileage        -0.3308     0.1278  -2.587 0.032258 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9966 on 8 degrees of freedom
## Multiple R-squared:  0.4555, Adjusted R-squared:  0.3875
## F-statistic: 6.693 on 1 and 8 DF,  p-value: 0.03226
##
##
## $chevrolet
##
## Call:
## lm(formula = price ~ mileage, data = .)
```

```
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -25765  -7312  -2318   6600  38349
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.577e+04  8.715e+02   29.56   <2e-16 ***
## mileage     -1.090e-01  9.782e-03  -11.14   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10250 on 295 degrees of freedom
## Multiple R-squared:  0.2961, Adjusted R-squared:  0.2937
## F-statistic: 124.1 on 1 and 295 DF,  p-value: < 2.2e-16
##
##
## $chrysler
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14031.0  -2512.1    449.1   1753.1   9194.4
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.320e+04  2.114e+03  10.975 7.42e-09 ***
## mileage     -1.303e-01  2.207e-02  -5.905 2.22e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5807 on 16 degrees of freedom
## Multiple R-squared:  0.6855, Adjusted R-squared:  0.6658
## F-statistic: 34.87 on 1 and 16 DF,  p-value: 2.216e-05
##
##
## $dodge
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -19352  -4098  -2298   3802  44131
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.454e+04  5.528e+02   44.40   <2e-16 ***
## mileage     -1.530e-01  9.524e-03  -16.07   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 7451 on 430 degrees of freedom
## Multiple R-squared:  0.3752, Adjusted R-squared:  0.3737
## F-statistic: 258.2 on 1 and 430 DF,  p-value: < 2.2e-16
##
##
## $ford
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -27213  -7835  -1316   4546  82472
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.721e+04  4.530e+02   60.07   <2e-16 ***
## mileage     -1.065e-01  5.985e-03  -17.79   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11550 on 1233 degrees of freedom
## Multiple R-squared:  0.2043, Adjusted R-squared:  0.2036
## F-statistic: 316.6 on 1 and 1233 DF,  p-value: < 2.2e-16
##
##
## $gmc
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -15377  -3599   -408    541  34395
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.538e+04  1.856e+03   8.283 3.28e-10 ***
## mileage     -8.060e-02  2.389e-02  -3.374  0.00166 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7911 on 40 degrees of freedom
## Multiple R-squared:  0.2215, Adjusted R-squared:  0.2021
## F-statistic: 11.38 on 1 and 40 DF,  p-value: 0.001656
##
##
## $`harley-davidson`
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
## ALL 1 residuals are 0: no residual degrees of freedom!
```

```
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    54680         NA      NA       NA
## mileage           NA         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
##
##
## $heartland
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##   326   334   343   350   358
##  1084 -1086  3714 -2966  -746
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)     2966       1129   2.627   0.0584 .
## mileage           NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2525 on 4 degrees of freedom
##
##
## $honda
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5371.0 -1758.1 -1104.7   700.1 10836.0
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7263.29815 2280.47169   3.185  0.00974 **
## mileage       -0.01240    0.02108  -0.588  0.56934
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4206 on 10 degrees of freedom
## Multiple R-squared:  0.03346,    Adjusted R-squared:  -0.0632
## F-statistic: 0.3462 on 1 and 10 DF,  p-value: 0.5693
##
##
## $hyundai
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
```

```
## Residuals:
##    Min     1Q Median     3Q    Max
##  -1887  -1372   -541    503   4571
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.272e+03  9.507e+02   5.545 9.46e-05 ***
## mileage     -1.213e-03  1.394e-02  -0.087    0.932
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2049 on 13 degrees of freedom
## Multiple R-squared:  0.0005822,  Adjusted R-squared:  -0.0763
## F-statistic: 0.007573 on 1 and 13 DF,  p-value: 0.932
##
##
## $infiniti
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3190.6 -2576.0   853.7  2124.3  2914.5
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.126e+04  1.422e+03   7.919 1.29e-05 ***
## mileage     7.358e-02  4.799e-02   1.533    0.156
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2636 on 10 degrees of freedom
## Multiple R-squared:  0.1903, Adjusted R-squared:  0.1094
## F-statistic: 2.351 on 1 and 10 DF,  p-value: 0.1562
##
##
## $jaguar
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
## ALL 1 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)     2800         NA      NA       NA
## mileage           NA         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
##
##
## $jeep
```

```
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -7876  -3110  -1611   4650   9517
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.343e+04  1.208e+03   11.12 8.77e-12 ***
## mileage     -6.446e-02  2.053e-02   -3.14  0.00396 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4979 on 28 degrees of freedom
## Multiple R-squared:  0.2604, Adjusted R-squared:  0.234
## F-statistic:  9.86 on 1 and 28 DF,  p-value: 0.003961
##
##
## $kia
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9220.2 -3631.8    16.8  3580.8  8952.2
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.740e+04  2.999e+03   5.801 0.000119 ***
## mileage     -1.108e-01  4.503e-02  -2.461 0.031608 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5697 on 11 degrees of freedom
## Multiple R-squared:  0.3551, Adjusted R-squared:  0.2965
## F-statistic: 6.058 on 1 and 11 DF,  p-value: 0.03161
##
##
## $land
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##    452    453    463    476
##   3206   9925  -1688 -11444
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8322.669  30273.246  -0.275    0.809
## mileage         1.399      1.119   1.250    0.338
```

```
##
## Residual standard error: 11010 on 2 degrees of freedom
## Multiple R-squared:  0.4388, Adjusted R-squared:  0.1581
## F-statistic: 1.564 on 1 and 2 DF,  p-value: 0.3376
##
##
## $lexus
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
## ALL 2 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 68497.056         NA      NA       NA
## mileage        -1.576         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:    NaN
## F-statistic:   NaN on 1 and 0 DF,  p-value: NA
##
##
## $lincoln
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
## ALL 2 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 43322.4882         NA      NA       NA
## mileage        -0.4829         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:    NaN
## F-statistic:   NaN on 1 and 0 DF,  p-value: NA
##
##
## $maserati
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
## ALL 1 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    30300         NA      NA       NA
## mileage           NA         NA      NA       NA
```

```
##
## Residual standard error: NaN on 0 degrees of freedom
##
##
## $mazda
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
## ALL 2 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 26948.1286         NA      NA       NA
## mileage        -0.2293         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:    NaN
## F-statistic:   NaN on 1 and 0 DF,  p-value: NA
##
##
## $`mercedes-benz`
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -20359 -10126  -3684   2892  45332
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 48950.9296 12603.9207   3.884  0.00465 **
## mileage        -0.3708     0.2026  -1.831  0.10453
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19120 on 8 degrees of freedom
## Multiple R-squared:  0.2952, Adjusted R-squared:  0.2071
## F-statistic: 3.351 on 1 and 8 DF,  p-value: 0.1045
##
##
## $nissan
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15913.1  -3018.4   -530.5   2802.8  21626.5
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  1.629e+04  5.252e+02   31.02   <2e-16 ***
## mileage     -9.952e-02  1.014e-02   -9.82   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5326 on 310 degrees of freedom
## Multiple R-squared:  0.2373, Adjusted R-squared:  0.2348
## F-statistic: 96.43 on 1 and 310 DF,  p-value: < 2.2e-16
##
##
## $peterbilt
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
##      491      505      517      529
##   -2.347   18.119 -512.615  496.843
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.188e+01  5.031e+02   0.163    0.886
## mileage     4.384e-04  5.997e-04   0.731    0.541
##
## Residual standard error: 505 on 2 degrees of freedom
## Multiple R-squared:  0.2109, Adjusted R-squared:  -0.1837
## F-statistic: 0.5345 on 1 and 2 DF,  p-value: 0.5408
##
##
## $ram
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
## ALL 1 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    11050         NA      NA       NA
## mileage           NA         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
##
##
## $toyota
##
## Call:
## lm(formula = price ~ mileage, data = .)
##
## Residuals:
## ALL 1 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (1 not defined because of singularities)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)     6300         NA      NA       NA
## mileage           NA         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
```

#map_lgl() The map_lgl() will return a logical vector.This map_lgl() checks according to the function given to it. In first example,we check whether cardata has any Double data type.In the second example,map_lgl() checks whether dataset has integer values or not,if present shows as TRUE else false.Map function always returns lists.

Eg1:

```
map_lgl(cardata, is.double)
```

```
##           X       price       brand       model        year title_status
##       FALSE       FALSE       FALSE       FALSE       FALSE       FALSE
##     mileage       color         vin         lot       state     country
##        TRUE       FALSE       FALSE       FALSE       FALSE       FALSE
##   condition
##       FALSE
```

Eg2:

```
map_lgl(cardata, is.integer)
```

```
##           X       price       brand       model        year title_status
##        TRUE        TRUE       FALSE       FALSE        TRUE       FALSE
##     mileage       color         vin         lot       state     country
##       FALSE       FALSE       FALSE        TRUE       FALSE       FALSE
##   condition
##       FALSE
```

map_chr()

map_chr() will return a character vector

```
map_chr(cardata, typeof)
```

```
##           X       price       brand       model        year title_status
##    "integer"   "integer" "character" "character"   "integer" "character"
##      mileage       color         vin         lot       state     country
##     "double" "character" "character"   "integer" "character" "character"
##   condition
## "character"
```

as_mapper()

The as_mapper() function arguments will allow you to refer to . for one argument functions, .x and .y for two argument functions, and ..1, ..2, ..3, etc, for functions with an arbitrary number of arguments.

```
as_mapper(~ length(unique(.cardata)))
```

```
## <lambda>
## function (..., .x = ..1, .y = ..2, . = ..1)
## length(unique(.cardata))
## attr(,"class")
## [1] "rlang_lambda_function" "function"
```

map_if

map_if() maps a function over the elements of dataset satisfying a predicate.

Eg1:Convert integers to characters

```
cardata %>%
    map_if(is.integer, as.character) %>%
    str()
```

```
## List of 13
##  $ X           : chr [1:2499] "0" "1" "2" "3" ...
##  $ price       : chr [1:2499] "6300" "2899" "5350" "25000" ...
##  $ brand       : chr [1:2499] "toyota" "ford" "dodge" "ford" ...
##  $ model       : chr [1:2499] "cruiser" "se" "mpv" "door" ...
##  $ year        : chr [1:2499] "2008" "2011" "2018" "2014" ...
##  $ title_status: chr [1:2499] "clean vehicle" "clean vehicle" "clean vehicle" "clean vehicle"
...
##  $ mileage     : num [1:2499] 274117 190552 39590 64146 6654 ...
##  $ color       : chr [1:2499] "black" "silver" "silver" "blue" ...
##  $ vin         : chr [1:2499] "  jtezu11f88k007763" "  2fmdk3gc4bbb02217" "  3c4pdcgg5jt34641
3" "  1ftfw1et4efc23745" ...
##  $ lot         : chr [1:2499] "159348797" "166951262" "167655728" "167753855" ...
##  $ state       : chr [1:2499] "new jersey" "tennessee" "georgia" "virginia" ...
##  $ country     : chr [1:2499] " usa" " usa" " usa" " usa" ...
##  $ condition   : chr [1:2499] "10 days left" "6 days left" "2 days left" "22 hours left" ...
```

map_at

This function will modify the elements corresponding to a character vector of names or a mumeric vector of positions. Eg1: Specify which columns to map with a numeric vector of positions:

```
cardata %>%
    map_at(c(5, 9, 10), as.character) %>% #year,vin and lot
    str()
```

```
## List of 13
##  $ X           : int [1:2499] 0 1 2 3 4 5 6 7 8 9 ...
##  $ price       : int [1:2499] 6300 2899 5350 25000 27700 5700 7300 13350 14600 5250 ...
##  $ brand       : chr [1:2499] "toyota" "ford" "dodge" "ford" ...
##  $ model       : chr [1:2499] "cruiser" "se" "mpv" "door" ...
##  $ year        : chr [1:2499] "2008" "2011" "2018" "2014" ...
##  $ title_status: chr [1:2499] "clean vehicle" "clean vehicle" "clean vehicle" "clean vehicle"
...
##  $ mileage     : num [1:2499] 274117 190552 39590 64146 6654 ...
##  $ color       : chr [1:2499] "black" "silver" "silver" "blue" ...
##  $ vin         : chr [1:2499] "  jtezu11f88k007763" "  2fmdk3gc4bbb02217" "  3c4pdcgg5jt34641
3" "  1ftfw1et4efc23745" ...
##  $ lot         : chr [1:2499] "159348797" "166951262" "167655728" "167753855" ...
##  $ state       : chr [1:2499] "new jersey" "tennessee" "georgia" "virginia" ...
##  $ country     : chr [1:2499] " usa" " usa" " usa" " usa" ...
##  $ condition   : chr [1:2499] "10 days left" "6 days left" "2 days left" "22 hours left" ...
```

Eg2: Specify which columns to map with a vector of names:

```
cardata %>% map_at(c("year", "condition"), as.character) %>% str()
```

```
## List of 13
##  $ X           : int [1:2499] 0 1 2 3 4 5 6 7 8 9 ...
##  $ price       : int [1:2499] 6300 2899 5350 25000 27700 5700 7300 13350 14600 5250 ...
##  $ brand       : chr [1:2499] "toyota" "ford" "dodge" "ford" ...
##  $ model       : chr [1:2499] "cruiser" "se" "mpv" "door" ...
##  $ year        : chr [1:2499] "2008" "2011" "2018" "2014" ...
##  $ title_status: chr [1:2499] "clean vehicle" "clean vehicle" "clean vehicle" "clean vehicle"
...
##  $ mileage     : num [1:2499] 274117 190552 39590 64146 6654 ...
##  $ color       : chr [1:2499] "black" "silver" "silver" "blue" ...
##  $ vin         : chr [1:2499] "  jtezu11f88k007763" "  2fmdk3gc4bbb02217" "  3c4pdcgg5jt34641
3" "  1ftfw1et4efc23745" ...
##  $ lot         : int [1:2499] 159348797 166951262 167655728 167753855 167763266 167655771 167
753872 167692494 167763267 167656121 ...
##  $ state       : chr [1:2499] "new jersey" "tennessee" "georgia" "virginia" ...
##  $ country     : chr [1:2499] " usa" " usa" " usa" " usa" ...
##  $ condition   : chr [1:2499] "10 days left" "6 days left" "2 days left" "22 hours left" ...
```

map_dfr

If we want each element of the output to be data frame, we can use map_dfr to row-bind them together:

```
cardata %>%
  split(.$brand) %>%
  map(~ lm(price ~ mileage, data = .x)) %>%
  map_dfr(~ as.data.frame(t(as.matrix(coef(.)))))
```

```
##    (Intercept)       mileage
## 1  20398.23838 -0.1090846326
## 2  41665.19091 -0.2344288803
## 3  37740.13957 -0.2370727570
## 4  27462.09596 -0.2042438935
## 5  38236.07584 -0.3307570136
## 6  25765.35281 -0.1089513803
## 7  23200.29356 -0.1303241254
## 8  24544.20937 -0.1530438349
## 9  27213.10155 -0.1064853097
## 10 15376.57227 -0.0806027845
## 11 54680.00000            NA
## 12  2966.00000            NA
## 13  7263.29815 -0.0123996785
## 14  5271.94860 -0.0012128424
## 15 11257.14399  0.0735793901
## 16  2800.00000            NA
## 17 13430.06958 -0.0644634728
## 18 17397.65093 -0.1108260810
## 19 -8322.66920  1.3986254924
## 20 68497.05597 -1.5755015839
## 21 43322.48816 -0.4829439625
## 22 30300.00000            NA
## 23 26948.12862 -0.2292657764
## 24 48950.92956 -0.3708432616
## 25 16288.17633 -0.0995222941
## 26    81.88076  0.0004384128
## 27 11050.00000            NA
## 28  6300.00000            NA
```

Task 3:

Creating an S3 object

```
s <- list(brand = 'Ford',price = 10300,model = "fiesta",year='2019' , title_status = "clean vehi
cle" ,
          mileage =   18937 ,color= "black", vin = "3fadp4bjxkm123238" , lot=  167759377,state="
missouri")

class(s) <- 'cardat1'
s # or print(j)
```

```
## $brand
## [1] "Ford"
##
## $price
## [1] 10300
##
## $model
## [1] "fiesta"
##
## $year
## [1] "2019"
##
## $title_status
## [1] "clean vehicle"
##
## $mileage
## [1] 18937
##
## $color
## [1] "black"
##
## $vin
## [1] "3fadp4bjxkm123238"
##
## $lot
## [1] 167759377
##
## $state
## [1] " missouri"
##
## attr(,"class")
## [1] "cardat1"
```

```
attributes(s)
```

```
## $names
##  [1] "brand"        "price"        "model"        "year"         "title_status"
##  [6] "mileage"      "color"        "vin"          "lot"          "state"
##
## $class
## [1] "cardat1"
```

print method

```
print.cardat1 <- function(wkr) {
  cat(wkr$brand, '\n')
  cat('Price', wkr$price, '\n')
  cat('model', wkr$model, '\n')
  cat('year', wkr$year, '\n')
  cat('Car_status', wkr$title_status, '\n')
  cat('Car_mileage', wkr$mileage, '\n')
  cat('color', wkr$color, '\n')
  cat('vin', wkr$vin, '\n')
  cat('lot', wkr$lot, '\n')
  cat('state', wkr$state, '\n')
}
print.cardat1(s)
```

```
## Ford
## Price 10300
## model fiesta
## year 2019
## Car_status clean vehicle
## Car_mileage 18937
## color black
## vin 3fadp4bjxkm123238
## lot 167759377
## state  missouri
```

#Function gives the list of available cars when given the price is $4200

```
dat<-cardata
class(dat)<-"Car Details"
print.car<-function(p)
{
    {cat("Data about particular given price($)", p, "\n")
    print("\n")
    p1<-filter(cardata,cardata$price == p)
    print(p1)
  }
}
print.car(4200)
```

```
## Data about particular given price($) 4200
## [1] "\n"
##      X price     brand   model year      title_status mileage    color
## 1  193  4200      ford    door 2017     clean vehicle   60219      red
## 2  493  4200 chevrolet cutaway 2006     clean vehicle  118009    white
## 3  763  4200 chevrolet    door 2014     clean vehicle  102500    black
## 4  901  4200     dodge    door 2014     clean vehicle  144725     gray
## 5 1227  4200      ford    door 2016     clean vehicle  104383     blue
## 6 1236  4200      ford    door 2013     clean vehicle       0 no_color
## 7 1848  4200      ford cutaway 2015 salvage insurance   80032   color:
##                   vin       lot        state country     condition
## 1   3fa6p0hd7hr273550 167656522        texas     usa   2 days left
## 2   1gbhg31u661113654 167781216    wisconsin     usa 21 hours left
## 3   1g1p75sz3e7332279 167799775       nevada     usa   2 days left
## 4   2c3cdxbg3eh170784 167519895       nevada     usa 22 hours left
## 5   3fa6p0hd5gr285811 167773291 pennsylvania     usa   2 days left
## 6   1fadp3j23dl155179 167773673 pennsylvania     usa   2 days left
## 7   1fdfe4fs3fda25516 167529792     illinois     usa 16 hours left
```

#summary method

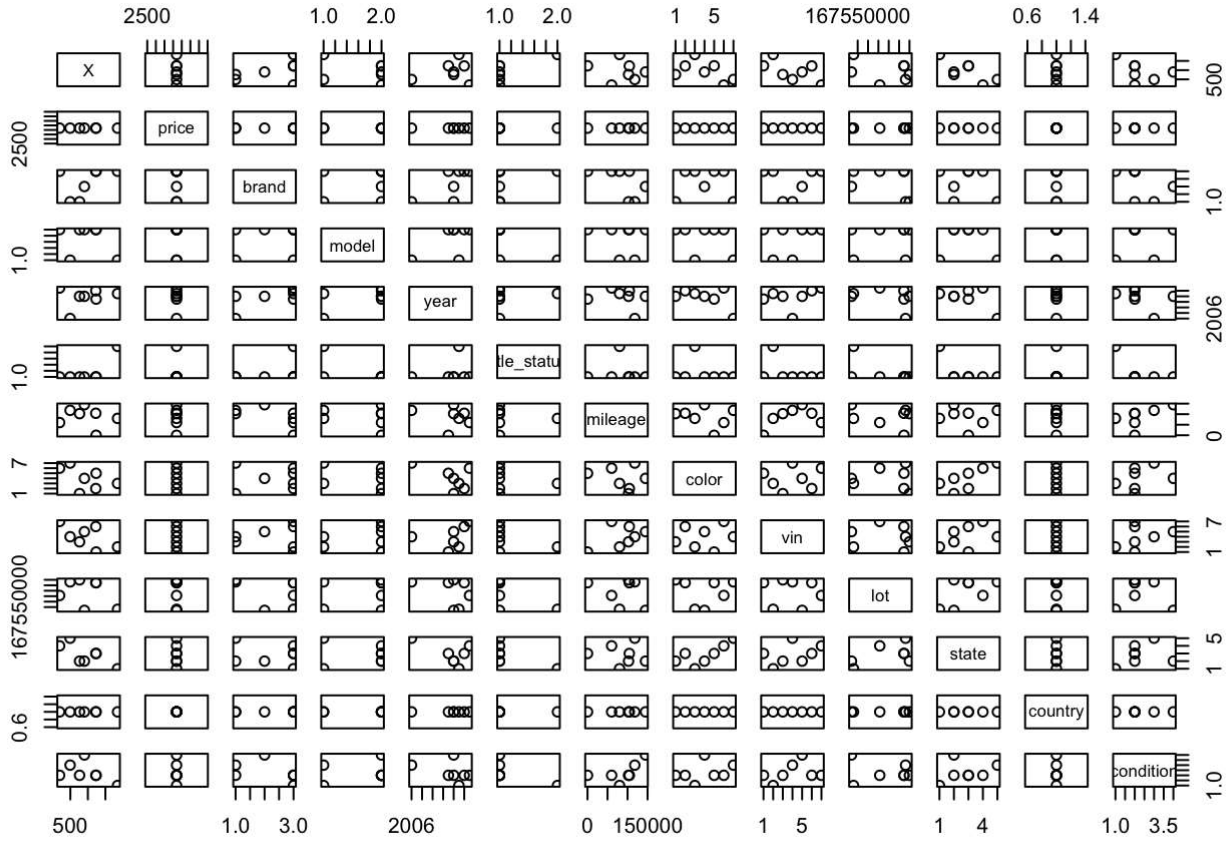
```
print.summary1<-function(pr)
{
print(summary(print.car(pr)))
}
print.summary1(4200)
```

```
## Data about particular given price($) 4200
## [1] "\n"
##      X price     brand   model year      title_status mileage    color
## 1  193  4200      ford    door 2017     clean vehicle   60219      red
## 2  493  4200 chevrolet cutaway 2006     clean vehicle  118009    white
## 3  763  4200 chevrolet    door 2014     clean vehicle  102500    black
## 4  901  4200     dodge    door 2014     clean vehicle  144725     gray
## 5 1227  4200      ford    door 2016     clean vehicle  104383     blue
## 6 1236  4200      ford    door 2013     clean vehicle       0 no_color
## 7 1848  4200      ford cutaway 2015 salvage insurance   80032   color:
##                   vin       lot        state country     condition
## 1   3fa6p0hd7hr273550 167656522        texas     usa   2 days left
## 2   1gbhg31u661113654 167781216    wisconsin     usa 21 hours left
## 3   1g1p75sz3e7332279 167799775       nevada     usa   2 days left
## 4   2c3cdxbg3eh170784 167519895       nevada     usa 22 hours left
## 5   3fa6p0hd5gr285811 167773291 pennsylvania     usa   2 days left
## 6   1fadp3j23dl155179 167773673 pennsylvania     usa   2 days left
## 7   1fdfe4fs3fda25516 167529792     illinois     usa 16 hours left
##        X              price         brand              model
##  Min.   : 193.0   Min.   :4200   Length:7           Length:7
##  1st Qu.: 628.0   1st Qu.:4200   Class :character   Class :character
##  Median : 901.0   Median :4200   Mode  :character   Mode  :character
##  Mean   : 951.6   Mean   :4200
##  3rd Qu.:1231.5   3rd Qu.:4200
##  Max.   :1848.0   Max.   :4200
##       year      title_status          mileage          color
##  Min.   :2006   Length:7           Min.   :     0   Length:7
##  1st Qu.:2014   Class :character   1st Qu.: 70126   Class :character
##  Median :2014   Mode  :character   Median :102500   Mode  :character
##  Mean   :2014                      Mean   : 87124
##  3rd Qu.:2016                      3rd Qu.:111196
##  Max.   :2017                      Max.   :144725
##      vin                lot               state             country
##  Length:7           Min.   :167519895   Length:7           Length:7
##  Class :character   1st Qu.:167593157   Class :character   Class :character
##  Mode  :character   Median :167773291   Mode  :character   Mode  :character
##                     Mean   :167690595
##                     3rd Qu.:167777444
##                     Max.   :167799775
##   condition
##  Length:7
##  Class :character
##  Mode  :character
##
##
##
```

#plot method

```
plot(print.car(4200))
```

```
## Data about particular given price($) 4200
## [1] "\n"
##      X price     brand   model year      title_status mileage    color
## 1  193  4200      ford    door 2017     clean vehicle   60219      red
## 2  493  4200 chevrolet cutaway 2006     clean vehicle  118009    white
## 3  763  4200 chevrolet    door 2014     clean vehicle  102500    black
## 4  901  4200     dodge    door 2014     clean vehicle  144725     gray
## 5 1227  4200      ford    door 2016     clean vehicle  104383     blue
## 6 1236  4200      ford    door 2013     clean vehicle       0 no_color
## 7 1848  4200      ford cutaway 2015 salvage insurance   80032   color:
##                   vin       lot        state country     condition
## 1   3fa6p0hd7hr273550 167656522        texas     usa   2 days left
## 2   1gbhg31u661113654 167781216    wisconsin     usa 21 hours left
## 3   1g1p75sz3e7332279 167799775       nevada     usa   2 days left
## 4   2c3cdxbg3eh170784 167519895       nevada     usa 22 hours left
## 5   3fa6p0hd5gr285811 167773291 pennsylvania     usa   2 days left
## 6   1fadp3j23dl155179 167773673 pennsylvania     usa   2 days left
## 7   1fdfe4fs3fda25516 167529792     illinois     usa 16 hours left
```



#help document

install.packages(“devtools”) install.packages(“roxygen2”) library(devtools) create(“myRpackage”)

#’ @param p A number. #’ @param p1 A vector #’ @return Cars of specified price value #’ @examples #’ print.car(4200) print.car<-function(p) { p1<-filter(cardata,cardata$price == p) print(p1) }