Stat 292 - Recitation 13

Review

Orçun Oltulu

23 / 6 / 2021

Required Packages:

```
inst_pack_func <- function(list.of.packages){
  new.packages <- list.of.packages[!(list.of.packages %in%
                                       installed.packages()[,"Package"])]
  if(length(new.packages)) install.packages(new.packages)
  lapply(list.of.packages,function(x){library(x,character.only=TRUE)})
}

list.of.packages <- c("tidyverse","magrittr","dplyr",
                      "ISLR", "sqldf","ggplot2")

inst_pack_func(list.of.packages)
```

Exercise 1: "apply" family

Part A:

Create the matrix with given code block;

```
q1a_matrix <- matrix(data = c(6,34,923,5,0, 112:116,
                              5,9,34,76,2, 545:549),
                     nrow = 5)
```

Then, get the mean of each row and column,

```
apply(q1a_matrix, MARGIN = 1, FUN = mean)
```

```
## [1] 167.00 175.50 404.50 186.00 166.75
```

```
apply(q1a_matrix, MARGIN = 2, FUN = mean)
```

```
## [1] 193.6 114.0  25.2 547.0
```

Part B:

Write a function that checks if a value is above a threshold value. The function should have two parameters: **x** which is the numeric value to check, and **threshold** which is the numeric threshold. Have the function return a logical value, TRUE if the value is above the threshold and FALSE if it is equal to or below the threshold.

Generate 20 random variables from standard normal distribution. Then apply the function on this random vector.

```
q1b_function <- function(x,threshold){
  x > threshold
}
set.seed(292)
random_sample <- rnorm(20)

sapply(random_sample, q1b_function, threshold = 0)
```

```
##  [1] FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE
## [13] FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
```

Part C:

Use three 'apply' family functions ("l","s") to get the minimum values of each column of the 'mtcars'.

```
data("mtcars")
lapply(mtcars, FUN = min)
```

```
## $mpg
## [1] 10.4
##
## $cyl
## [1] 4
##
## $disp
## [1] 71.1
##
## $hp
## [1] 52
##
## $drat
## [1] 2.76
##
## $wt
## [1] 1.513
##
```

```
## $qsec
## [1] 14.5
##
## $vs
## [1] 0
##
## $am
## [1] 0
##
## $gear
## [1] 3
##
## $carb
## [1] 1
```

```
sapply(mtcars, FUN = min)
```

```
##    mpg    cyl   disp     hp   drat     wt   qsec     vs     am   gear   carb
## 10.400  4.000 71.100 52.000  2.760  1.513 14.500  0.000  0.000  3.000  1.000
```

Exercise 2: sqldf

Use Auto data set from ISLR package for this exercise. Prepare the data before you work on.

```
data(Auto)
Auto$year <- NULL
Auto$origin <- factor(Auto$origin,
                      levels = 1:3,
                      labels=c("American","European","Japanese"))
Auto$cylinders <- factor(Auto$cylinders)
```

Part A:

Using only one "sqldf" command, select "name", "mpg", "origin", and "horsepower", print first 10 rows.

```
sqldf("SELECT name,mpg,origin,horsepower
      FROM Auto
      LIMIT 10")
```

```
##                         name mpg   origin horsepower
## 1  chevrolet chevelle malibu  18 American        130
## 2          buick skylark 320  15 American        165
## 3         plymouth satellite  18 American        150
## 4              amc rebel sst  16 American        150
```

```
## 5                ford torino  17 American        140
## 6           ford galaxie 500  15 American        198
## 7           chevrolet impala  14 American        220
## 8          plymouth fury iii  14 American        215
## 9           pontiac catalina  14 American        225
## 10        amc ambassador dpl  15 American        190
```

Part B:

Using only one "sqldf" command, print the minimum and maximum 'horsepower'.

```
sqldf("SELECT MIN(horsepower), MAX(horsepower)
      FROM Auto")
```

```
##   MIN(horsepower) MAX(horsepower)
## 1              46             230
```

Part C:

Using only one "sqldf" command, find the average MPG for each origin.

```
sqldf("SELECT origin, AVG(mpg) AS AverageMPG
      FROM Auto
      GROUP BY origin")
```

```
##     origin AverageMPG
## 1 American   20.03347
## 2 European   27.60294
## 3 Japanese   30.45063
```

Part D:

Using only one "sqldf" command, list of the car names whose weight changes between 2850-2950, sort them by their weights in decreasing order.

```
sqldf("SELECT name, weight
      FROM Auto
      WHERE weight BETWEEN 2850 AND 2950
      ORDER BY weight DESC")
```

```
##                       name weight
## 1     audi 5000s (diesel)   2950
## 2        chevrolet camaro   2950
## 3              amc hornet   2945
## 4             volvo 244dl   2945
## 5   buick century limited   2945
## 6         volvo 145e (sw)   2933
```

```
## 7          toyota mark ii   2930
## 8       datsun 810 maxima   2930
## 9             amc gremlin   2914
## 10          datsun 280-zx   2910
## 11        plymouth duster   2904
## 12             amc hornet   2901
## 13        toyota cressida   2900
## 14        ford fairmont 4   2890
## 15          ford fairmont   2870
## 16            volvo 144ea   2868
## 17   ford fairmont futura   2865
## 18 oldsmobile starfire sx   2855
```

Exercise 3: piping

For this exercise, again use the Auto data set.

Part A:

Using a pipeline, obtain a correlation matrix between numeric variables.

```
Auto %>% select_if(is.numeric) %>% cor()
```

```
##                     mpg displacement horsepower     weight acceleration
## mpg           1.0000000   -0.8051269 -0.7784268 -0.8322442    0.4233285
## displacement -0.8051269    1.0000000  0.8972570  0.9329944   -0.5438005
## horsepower   -0.7784268    0.8972570  1.0000000  0.8645377   -0.6891955
## weight       -0.8322442    0.9329944  0.8645377  1.0000000   -0.4168392
## acceleration  0.4233285   -0.5438005 -0.6891955 -0.4168392    1.0000000
```

Part B:

Using a pipeline, drop cylinders, displacement and acceleration variables and filter only American or European cars whose weight is less than 2800. Print first 10 observations.

```
Auto %>%
  select(-c(cylinders, displacement, acceleration)) %>%
  filter(origin %in% c("American","European"),
         weight < 2800) %>% head(.,10)
```

```
##    mpg horsepower weight   origin                       name
## 17  18         97   2774 American                 amc hornet
## 18  21         85   2587 American               ford maverick
## 20  26         46   1835 European volkswagen 1131 deluxe sedan
## 21  25         87   2672 European                 peugeot 504
## 22  24         90   2430 European                 audi 100 ls
```

```
## 23  25          95   2375 European                     saab 99e
## 24  26         113   2234 European                     bmw 2002
## 25  21          90   2648 American                  amc gremlin
## 31  28          90   2264 American          chevrolet vega 2300
## 34  19         100   2634 American                  amc gremlin
```

Part C:

Using a pipeline, create a new column km_per_litre by multiplying mpg by 0.425.

```
Auto %<>% mutate(km_per_litre = 0.425*mpg)
```

Part D:

Using a pipeline, print the average km_per_litre for each origin.

```
Auto %>% aggregate(km_per_litre ~ origin, ., mean)
```

```
##     origin km_per_litre
## 1 American     8.514224
## 2 European    11.731250
## 3 Japanese    12.941519
```

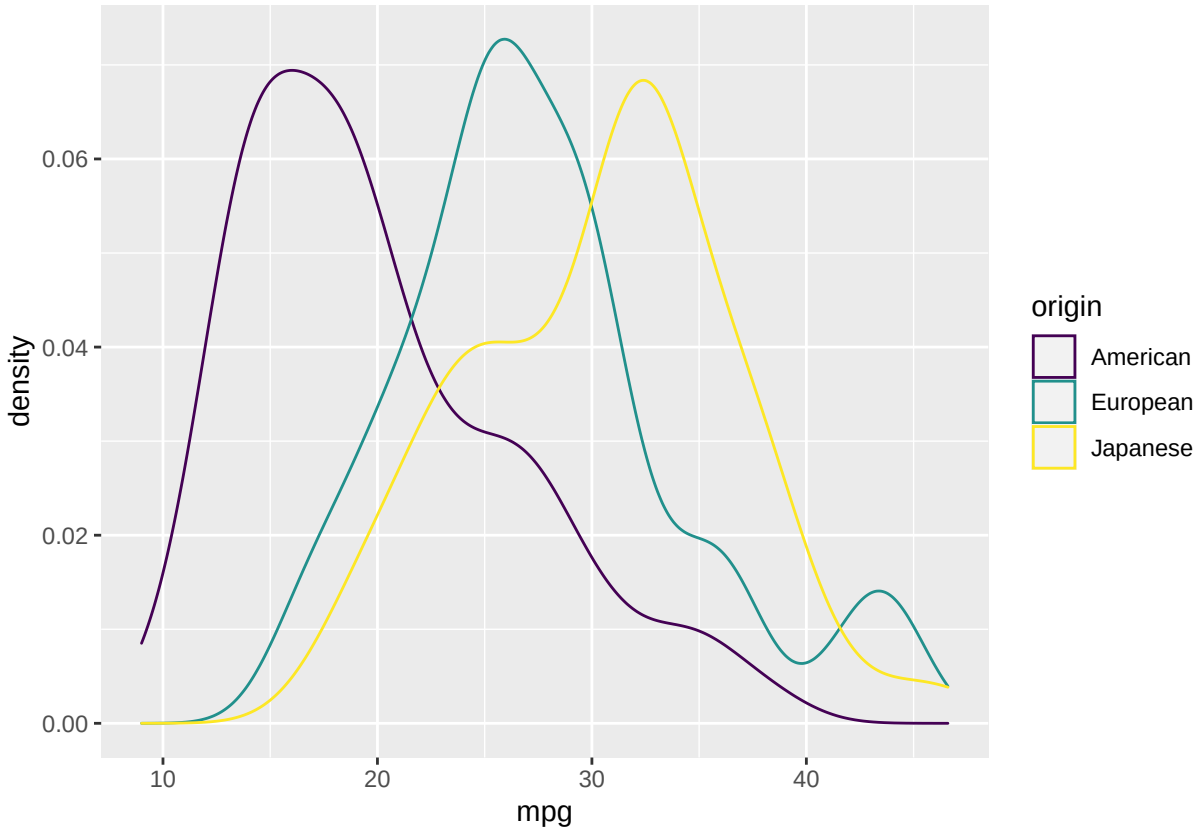
Exercise 4: ggplot

For this exercise, again use the Auto data set.

Part A:

Draw density curves for 'mpg' for each origin on a single plot.

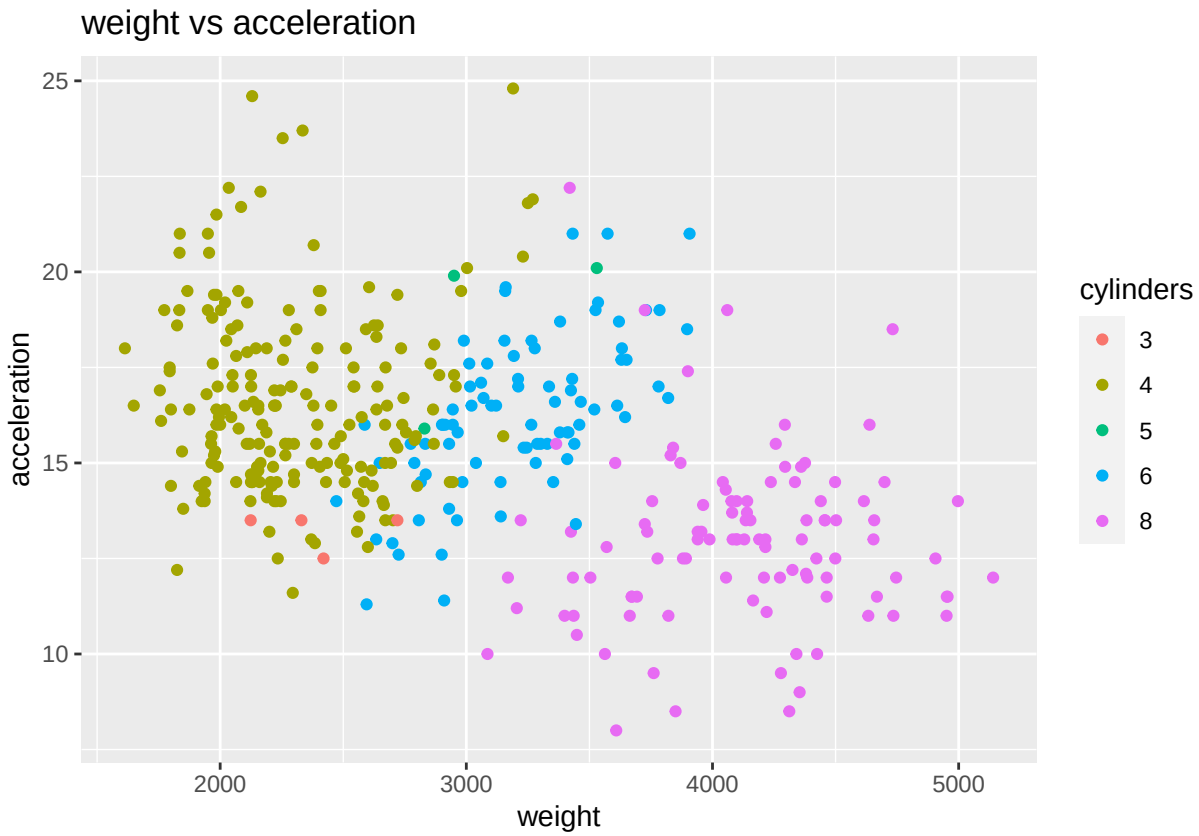
```
ggplot(Auto, aes(mpg, color = origin)) +
  geom_density() +
  scale_color_viridis_d()
```



Part B:

Obtain a scatter-plot for weight vs acceleration and color by "cylinders".

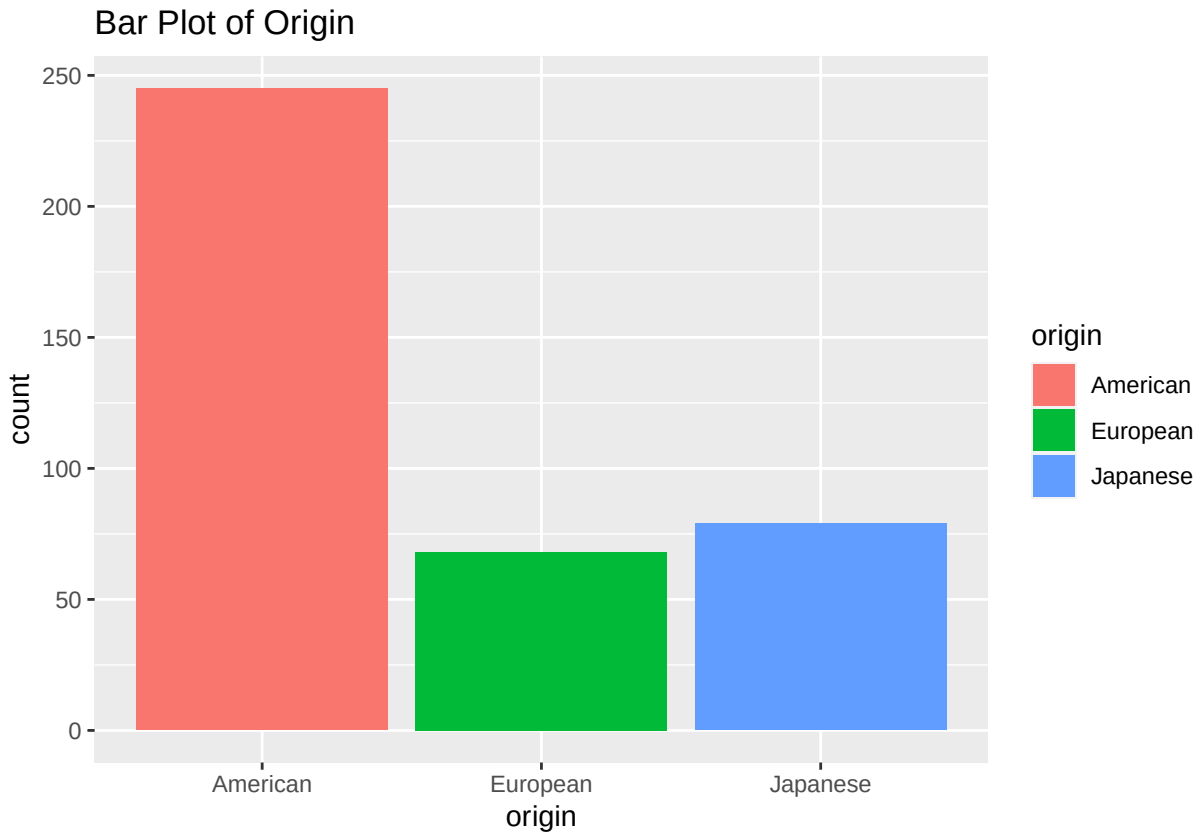
```
ggplot(Auto, aes(weight,acceleration, color = cylinders)) +
  geom_point() +
  labs(title = "weight vs acceleration")
```



Part C:

Obtain a bar plot for 'origin'.Color each bar for each level of 'origin'.

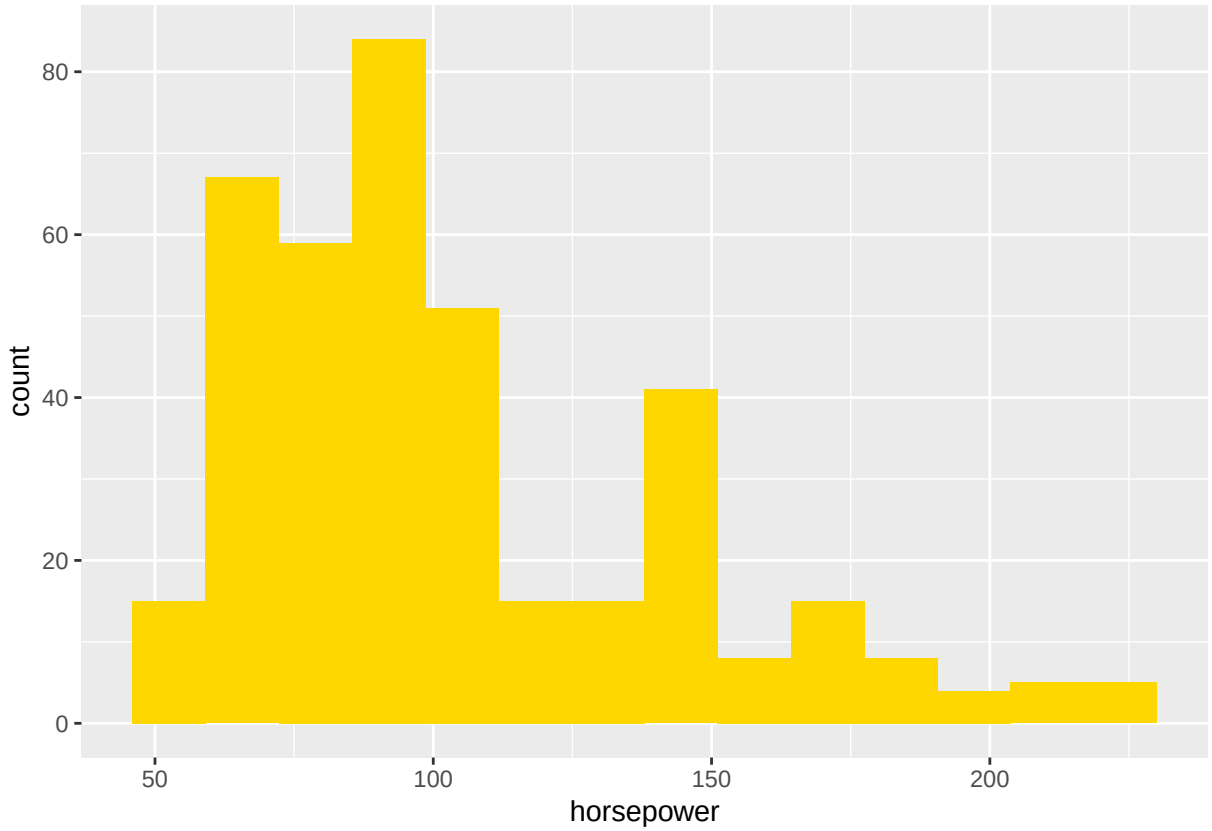
```
ggplot(data = Auto) +
  geom_bar(mapping = aes(x = origin, fill = origin)) +
  labs(title = "Bar Plot of Origin")
```



Part D:

Create a histogram for horsepower variable.

```
ggplot(data = Auto, aes(x = horsepower)) +
  geom_histogram(fill = "Gold", bins = 15)
```


80
60
40
20
0
count
50
100
150
200
horsepower