Assignment 5

Travis Lloyd

4/2/2021

```
## Loading required package: DBI

## corrplot 0.84 loaded
```

Question 1.1

How many countries became independent in the twentieth century?

```
dbGetQuery(con,
'SELECT Count(Name) FROM Country WHERE IndepYear >=1900');

##   Count(Name)
## 1         149
```

Question 1.2

How many people in the world are expected to live for 75 years or more?

```
dbGetQuery(con,
'SELECT SUM(Population) FROM Country WHERE LifeExpectancy >= 75');

##   SUM(Population)
## 1       982470200
```

Question 1.3

List the 10 most populated countries in the world with their populations as a percentage of the world population.

```
SELECT Name, Population/6078749450 FROM Country ORDER BY Population DESC
LIMIT 10;
```

Displaying records 1 - 10

Name	Population/6078749450
China	0.2102
India	0.1668
United States	0.0458
Indonesia	0.0349
Brazil	0.0280
Pakistan	0.0257

Russian Federation	0.0242
Bangladesh	0.0212
Japan	0.0208
Nigeria	0.0183

Question 1.4

List the top 10 countries with the highest population density.

```
dbGetQuery(con, 'SELECT Name, population/SurfaceArea FROM Country ORDER BY
population/SurfaceArea DESC LIMIT 10;
');
```

```
##                              Name population/SurfaceArea
## 1                           Macao             26277.7778
## 2                          Monaco             22666.6667
## 3                       Hong Kong              6308.8372
## 4                       Singapore              5771.8447
## 5                       Gibraltar              4166.6667
## 6  Holy See (Vatican City State)              2500.0000
## 7                         Bermuda              1226.4151
## 8                           Malta              1203.1646
## 9                        Maldives               959.7315
## 10                     Bangladesh               896.9222
```

Question 1.5

How many countries are there in each “Region” ? Write a SQL query that produces a list of regions with a column for country counts for each region and order the count descending.

```
dbGetQuery(con, 'SELECT Region, count(Region) FROM Country GROUP BY Region
ORDER BY count(Region) DESC')
```

```
##                       Region count(Region)
## 1                  Caribbean            24
## 2             Eastern Africa            20
## 3                Middle East            18
## 4             Western Africa            17
## 5            Southern Europe            15
## 6  Southern and Central Asia            14
## 7              South America            14
## 8             Southeast Asia            11
## 9                  Polynesia            10
## 10            Eastern Europe            10
## 11            Central Africa             9
## 12            Western Europe             9
## 13           Central America             8
## 14              Eastern Asia             8
## 15          Nordic Countries             7
## 16          Northern Africa             7
## 17               Micronesia             7
## 18               Antarctica             5
## 19 Australia and New Zealand             5
## 20            North America             5
## 21          Southern Africa             5
## 22                Melanesia             5
## 23         Baltic Countries             3
## 24          British Islands             2
## 25     Micronesia/Caribbean             1
```

Question 1.6

What countries have more than 10 languages represented? Write a SQL query, using the “HAVING” clause, that produces the list of countries that have greater than 10 languages. Group by “CountryCode” and order by language count descending.

```
dbGetQuery(con, 'SELECT CountryCode, count(Language) FROM CountryLanguage
GROUP BY CountryCode HAVING count(language) > 10 ORDER BY count(CountryCode)
DESC;');
```

```
##   CountryCode count(Language)
## 1         CAN              12
## 2         CHN              12
## 3         IND              12
## 4         RUS              12
## 5         USA              12
## 6         TZA              11
## 7         ZAF              11
```

Question 2.2

Use R to understand how horsepower and weights are related to each other. Plot them using a scatter plot and color the data points using mpg. Do you see anything interesting/useful here? Report your observations with this plot. Now let us cluster the data on this plane in a “reasonable” number of groups. Show your plot where the data points are now colored with the cluster information and provide your interpretations.

To understand how horsepower and weight are related in regards to vehicls, we must first subset Horsepower, Weight, and MPG.

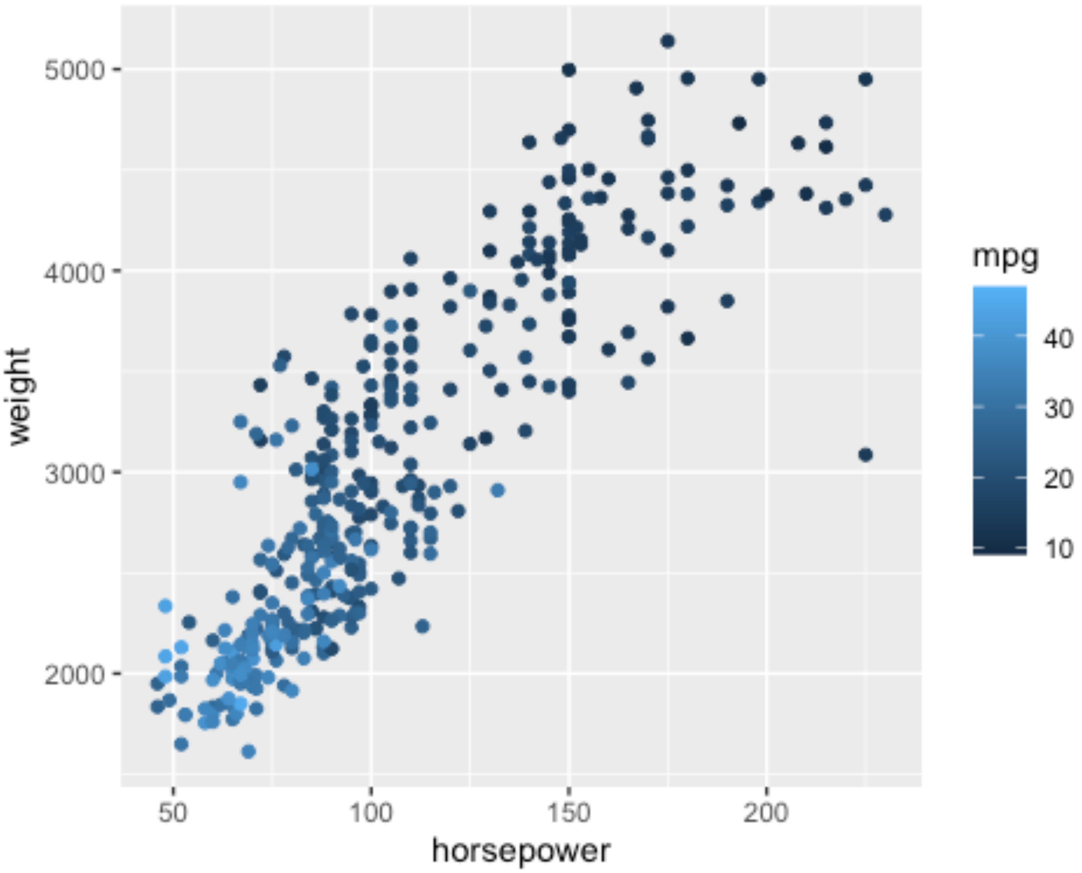
```
data <- dbGetQuery(con, "SELECT * FROM mpg")
power.weight <- dbGetQuery(con, "SELECT horsepower, weight, mpg FROM mpg")
```

Now that we have our neccesary subset, we can run a correlation test to better understand how closely correlated horsepower and weight are. Once the correlation coefficient is observed, a scatterplot can be run to gain a visual understanding of the correlation. Each point is coloured by MPG range for this scatterplot.

```
cor.test(power.weight$horsepower, power.weight$weight)
```

```
##
##  Pearson's product-moment correlation
##
## data:  power.weight$horsepower and power.weight$weight
## t = 33.972, df = 390, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8371778 0.8875815
## sample estimates:
##       cor
## 0.8645377
```

```
ggplot(power.weight, aes(horsepower, weight, colour = mpg)) + geom_point()
```

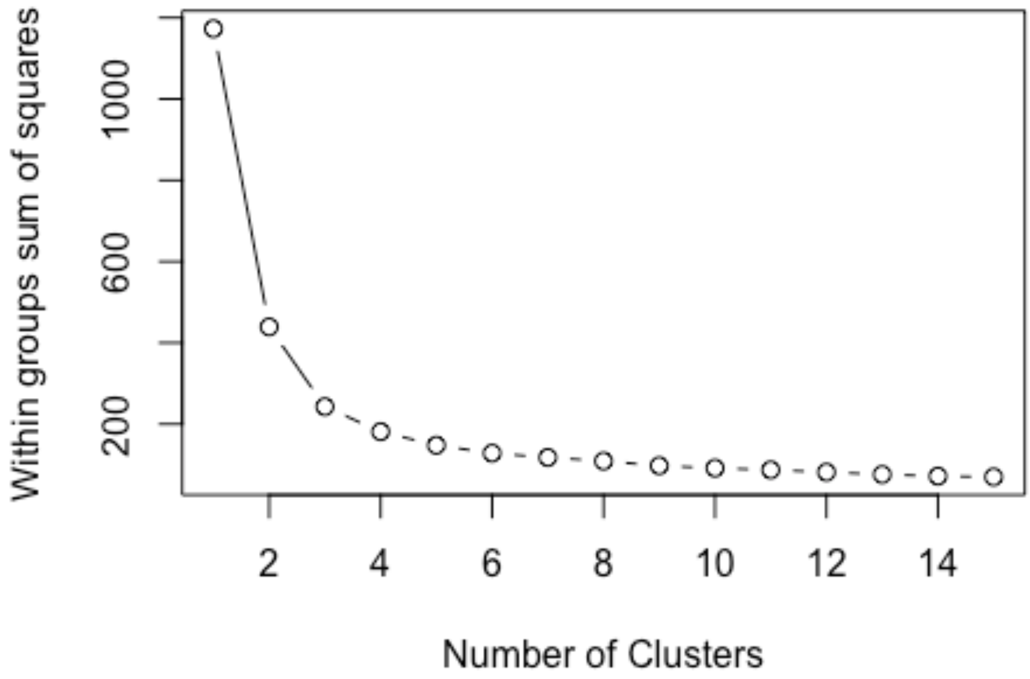


This scatterplot is very revealing with information regarding how closely related weight, horsepower and MPG are related. As Weight increases, horse power increases along with a decreasing MPG.

Using a kmeans clustering approach, we can find similar information. We must first normalize the data using the scale function. This is followed by a graph to have an educated understanding of how many clusters to use when grouping.

```
power.weight.norm <- scale(power.weight)

wss <- (nrow(power.weight.norm)-1)*sum(apply(power.weight.norm,2,var))
for (i in 2:15) wss[i] <- sum(kmeans(power.weight.norm,
                                     centers=i)$withinss)
plot(1:15, wss, type="b", xlab="Number of Clusters",
     ylab="Within groups sum of squares")
```



From the plot we can se that the arm bends at 4 clusters so a cluster analysis with 4 clusters will be used

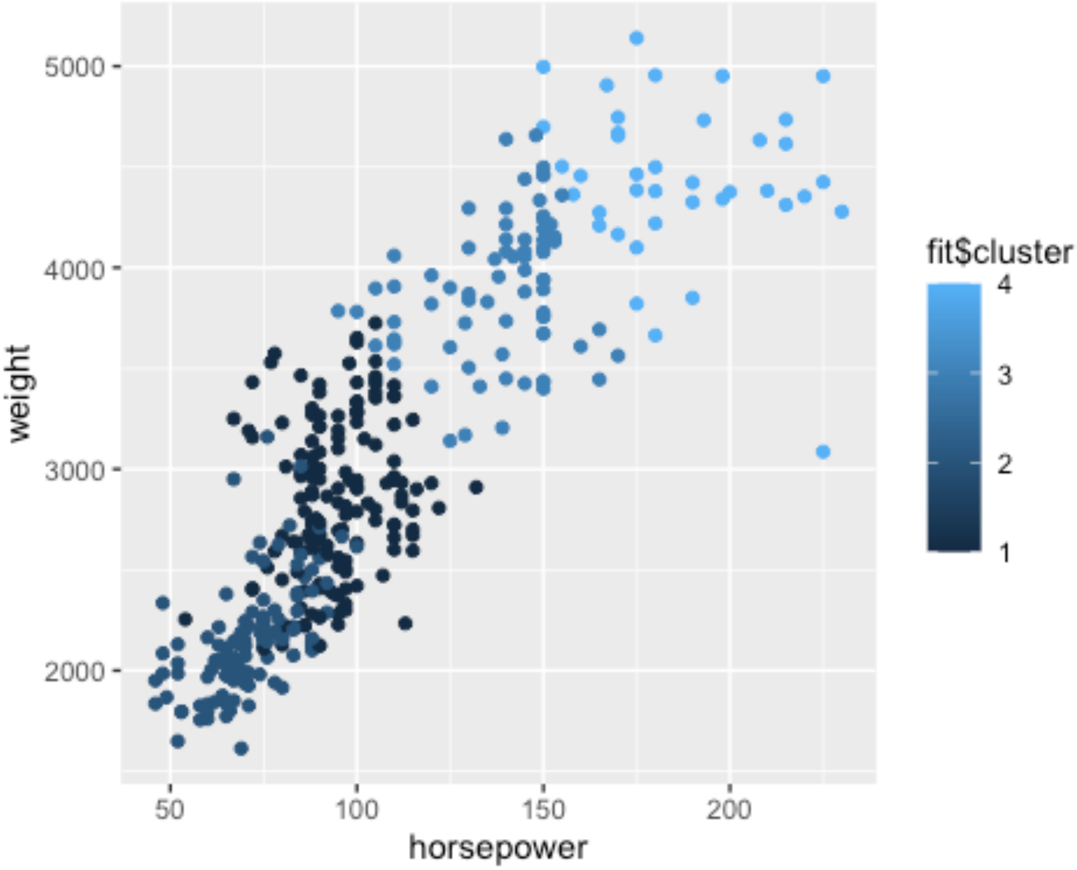
```
# K-Means Cluster Analysis
fit <- kmeans(power.weight.norm, 4)
# get cluster means
aggregate(power.weight.norm,by=list(fit$cluster),FUN=mean)

##   Group.1 horsepower     weight        mpg
## 1       1 -0.2599509 -0.1580559 -0.1454798
## 2       2 -0.8656168 -0.9828066  1.1629786
## 3       3  0.8584769  1.0618392 -0.9588797
## 4       4  2.1301554  1.7179561 -1.3254617

# append cluster assignment
power.weight.norm <- data.frame(power.weight.norm, fit$cluster)
```

Plotting the same scatterplot as previously used but with colour set for clust groups. This analysis displays that there could be some hidden underlying information

```
ggplot(power.weight, aes(horsepower, weight, colour = fit$cluster)) +
geom_point()
```



This plot is very similar to our original scatterplot, however we can see that the groups are now colored together. The

```
lapply( dbListConnections( dbDriver( drv = "MySQL")), dbDisconnect)

## [[1]]
## [1] TRUE
```

Question 2.2

This Question is on the pages to follow.

In [106…

```
import os
import mysql.connector
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import linear_model
from sklearn.linear_model import LinearRegression

%matplotlib inline
```

Use Python to explore the relationship of different variables to models per gallon (mpg). Find out which of the variables have high correlation with mpg. Report those values. Build a regression model using one of those variables to predict mpg. Do the same using two of those variables. Report your models along with the regression line equations.

In [139…

```
username = os.environ.get('DB_USER')
password = os.environ.get('DB_PASS')
```

In [108…

```
cnx = mysql.connector.connect(host='127.0.0.1',
                              user = username,
                              password = password,
                              port = int(3306),
                              db = 'module_5')
```

In [109…

```
mpg_table = pd.read_sql_query('SELECT * FROM mpg', cnx)
```

In [110…

```
mpg_table.corr()
```

Out[110…

	mpg	cylinders	displacement	horsepower	weight	model year	origin
mpg	1.000000	-0.776796	-0.804304	-0.777683	-0.831535	0.582750	0.563667
cylinders	-0.776796	1.000000	0.950823	0.842983	0.897527	-0.345647	-0.568932
displacement	-0.804304	0.950823	1.000000	0.897259	0.932993	-0.369873	-0.614551
horsepower	-0.777683	0.842983	0.897259	1.000000	0.864538	-0.416361	-0.455171
weight	-0.831535	0.897527	0.932993	0.864538	1.000000	-0.309120	-0.585005
model year	0.582750	-0.345647	-0.369873	-0.416361	-0.309120	1.000000	0.181528
origin	0.563667	-0.568932	-0.614551	-0.455171	-0.585005	0.181528	1.000000

Variable with high correlation with MPG include: cylinders, displacement, horsepower and weight.

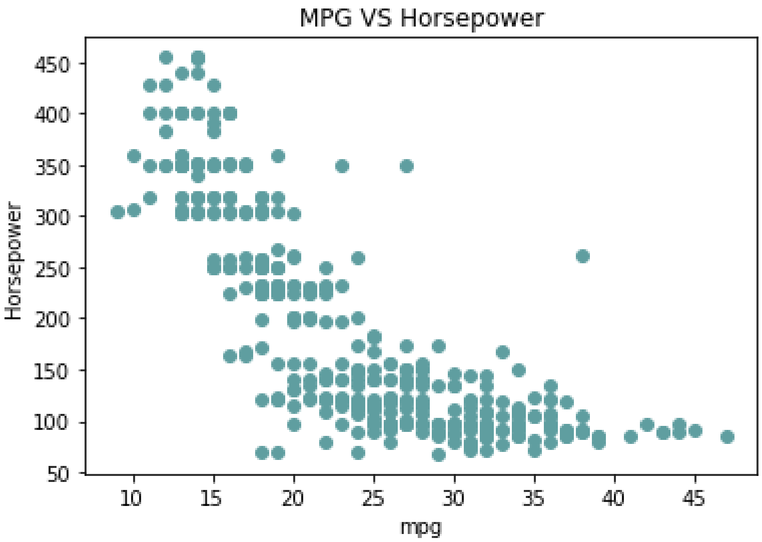
In [112…

```
x = mpg_table['mpg']
y = mpg_table['displacement']
plt.plot(x, y, 'o', color = 'cadetblue')

plt.title("MPG VS Horsepower")
plt.xlabel("mpg")
plt.ylabel("Horsepower")
```

Out[112…

```
Text(0, 0.5, 'Horsepower')
```



In [114…

```
y = mpg_table[['mpg']]
x1 = mpg_table[['displacement']]
x2 = mpg_table[['displacement', 'weight']]
```

In [121…

```
reg = LinearRegression().fit(y, x1)
reg_multi = LinearRegression().fit(y, x2)
```

In [122…

```
reg.coef_
```

Out[122…

```
array([[-10.79044089]])
```

In [126…

```
reg.intercept_
```

Out[126…

```
array([447.90604637])
```

Regression Line Equation MPG VS Horsepower

y = 447.90 - 10.79x

In [124…

```
reg_multi.coef_
```

Out[124…

```
array([[-10.79044089],
       [-90.55321282]])
```

In [127…

```
reg_multi.intercept_
```

Out[127…

```
array([ 447.90604637, 5104.89167558])
```

Regression Line Equation MPG VS (Horsepower & Weight)

y = 447.90 - 10.79x1 - 90.55x2