# Assignment 1

CS 750/850 Machine Learning

Landon Buell

29 January 2020

- **Due**: Monday 2/3 at 11:59PM
- **Submisssion**: Turn in as a **PDF** and the **source code** (R,Rmd,py,ipynb) on MyCourses
- **Questions**: Piazza and Office hours: *Marek*: Wed 1:30-3:00pm, Soheil: Mon 2-4pm, Xihong: Thu 1:30-3:30pm
- **Extra credit**: Especially good questions or helpful answers on Piazza regarding the assignment earn up to 5 points extra credit towards the assignment grade.

The instructions are geared towards R users. The comments in [P: xxx] are meant as hints to Python users.

Feel free to achieve the results using commands other than the ones mentioned. R, especially, shines when it comes to processing and visualization of structured data, but you would not be able to tell from the ISL book. It uses the oldest and simplest (and ugliest) subset of R. I recommend checking out `dplyr` for data processing [3,4] and GGPlot [1,4] for plotting.

## Problem 1 [25%]

In this exercise you will create some simulated data and will fit simple linear regression models to it. Make sure to use `set.seed(1)` [P: `np.random.seed(1)`] prior to starting part (1) to ensure consistent results.

1. Using the `rnorm()` [P: `np.random.normal`] function, create a vector, `x`, containing 100 observations drawn from a $\mathcal{N}(0, 3)$ distribution (Normal distribution with the mean 0 and the **standard deviation** $\sqrt{3}$). This represents a feature, $X$.

```
set.seed(1)
x <- rnorm(n=100,mean=0,sd=sqrt(3))
```

2. Using the `rnorm()` function, create a vector, `eps`, containing 100 observations drawn from a $\mathcal{N}(0, 0.5)$ distribution i.e. a normal distribution with mean zero and standard deviation $\sqrt{0.5}$.

```
eps <- rnorm(n=100,mean=0,sd=sqrt(0.5))
```

3. Using `x` and `eps`, generate a vector `y` according to the model $Y$:

$$Y = -2 + 0.6X + \epsilon$$

What is the length (number of elements) of `y`? What are the values of $\beta_0, \beta_1$ in the equation above (intercept and slope)?

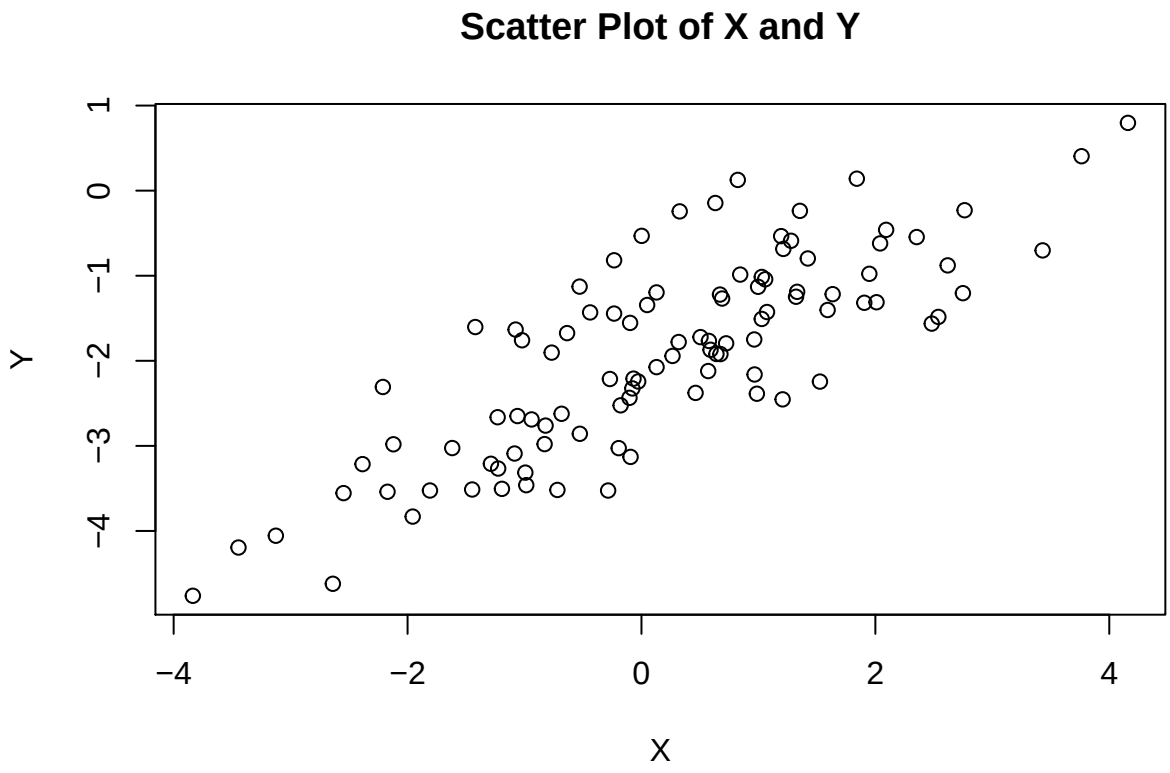
```r
Y <- -2 + 0.6*x + eps
message("Elements in Y:",length(Y))
```

```
## Elements in Y:100
```

In this model above, the intercept is given by $\beta_0 = -2$ and the slope is given by $\beta_1 = +0.6$.

4. Create a scatterplot displaying the relationship between `x` and `y`. Comment on what you observe. [P: see [2]]

```r
plot(x,Y,main='Scatter Plot of X and Y',
     xlab='X',ylab='Y')
```




The data seemes to be losely correlated, following a roughly linear relationship. (As I would expect it to because of our linear model)

5. Fit a least squares linear model to predict `y` using `x`. Comment on the model obtained. How do $\hat{\beta}_0, \hat{\beta}_1$ compare to $\beta_0, \beta_1$?

```r
lin_fit <- lm(Y~x) # lm = linear model
lin_fit
```

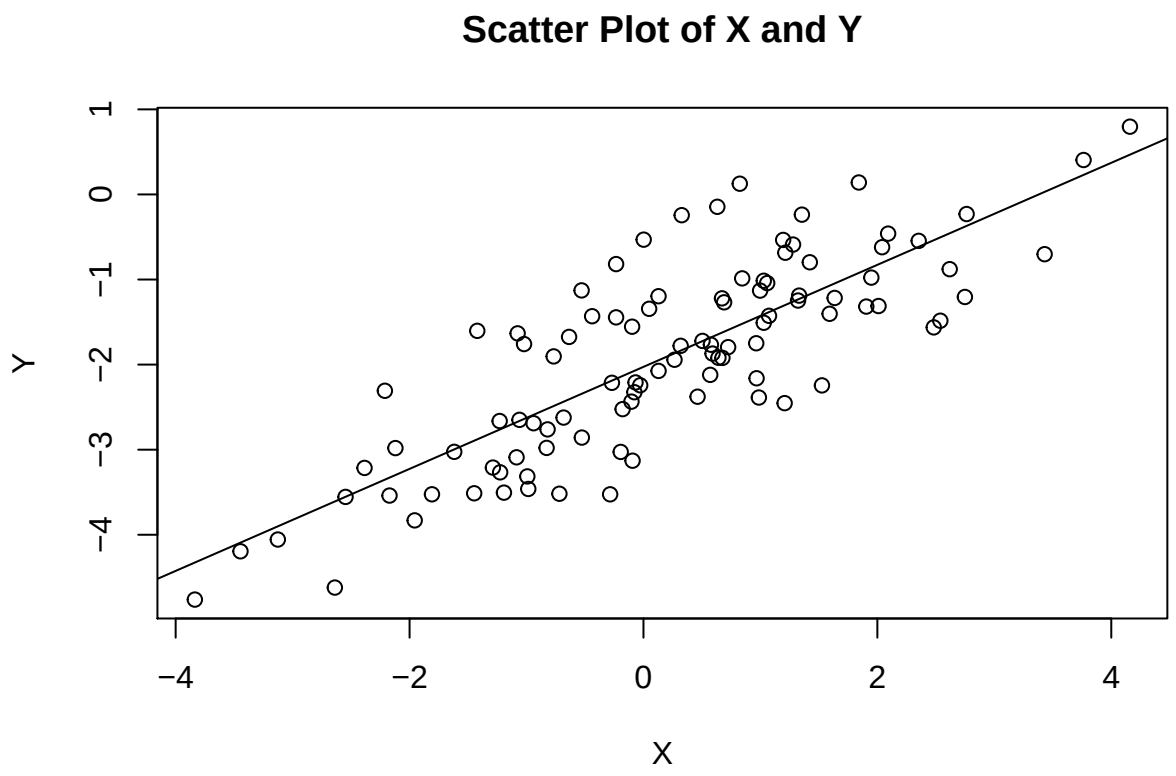
```
##
## Call:
```

```
## lm(formula = Y ~ x)
##
## Coefficients:
## (Intercept)            x
##     -2.0267       0.5996
```

The regression line is a linear relationsip (1st degree polynomial) as anticipated. It produces a fit such that the sum of the vertical distances between each data point and the line has been minimized.
In our equation, we set the intercept and slope to be: $\beta_0 = -2$ and $\beta_1 = +0.6$. repectivly. In the fit model, the linear model found the approximations of the slpoe and intercept to be roughly: $\hat{\beta_0} = -2.0267$ and $\hat{\beta_1} = +0.5996$ as shown by the coeffefcients in the fit variable above.

6. Display the least squares line on the scatterplot obtained in 4.

```
plot(x,Y,main='Scatter Plot of X and Y',
     xlab='X',ylab='Y')
abline(lin_fit)
```




7. Now fit a polynomial regression model that predicts $y$ using $x$ and $x^2$. Is there evidence that the quadratic term improves the model fit? Explain your answer.

```
quad_fit = lm(Y~poly(x,2))
quad_fit
```

```
##
```

```
## Call:
## lm(formula = Y ~ poly(x, 2))
##
## Coefficients:
## (Intercept)  poly(x, 2)1  poly(x, 2)2
##     -1.9136       9.2809      -0.9504
```

The quadratic model seems to be much further removed from the data set. Ideally, I would have expected the intercept and slope coefficients to remain fairly close to the repesctive linear model and then the quadratic coefficient to be close to zero. TOo me this would indicate that the quadratic term was not important, but still allow for the retention of the properties of the linear model. Instead, the addition of another polynomial term completely changes the best fit model to a point where is differs greatly from the expected value. Now we have the predictions: $\hat{\beta}_0 = -19.136$, $\hat{\beta}_1 = 9.2809$ and $\hat{\beta}_2 = -0.9504$.

## Optional Problem O1 [30%]

This problem can be substituted for Problem 1 above, for up to 5 points extra credit. At most one of the problems 1 and O1 will be considered.

Read Chapter 1 and solve Exercises *1.6* and *1.10* in [Bishop, C. M. (2006). Pattern Recognition and Machine Learning].

## Problem 2 [25%]

Read through Section 2.3 in ISL. Load the `Auto` data set and *make sure to remove missing values from the data*. Then answer the following questions:

```
# Below is from James, pg. 49
autodata <- read.csv(file='Auto.csv',header=T,na.strings="?")
#fix(autodata) # I'm not sure what "fix()" does
dim(autodata) # dimensions of array (nrows,ncols)
```

```
## [1] 397   9
```

```
autodata = na.omit(autodata) # eliminate "na's"
dim(autodata) # dimensions of array (nrows,ncols)
```

```
## [1] 392   9
```

```
# check variable names:
names(autodata)
```

```
## [1] "mpg"          "cylinders"    "displacement" "horsepower"   "weight"
## [6] "acceleration" "year"         "origin"       "name"
```

1. Which predictors are *quantitative* and which ones are *qualitative*?

```
summary(autodata)
```

```
##       mpg          cylinders      displacement     horsepower        weight
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140
##
##   acceleration        year           origin                      name
##  Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador       :  5
##  1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto        :  5
##  Median :15.50   Median :76.00   Median :1.000   toyota corolla    :  5
##  Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin       :  4
##  3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet        :  4
##  Max.   :24.80   Max.   :82.00   Max.   :3.000   chevrolet chevette:  4
##                                                  (Other)           :365
```

The *name*, *origin*, and perhaps even the *year* are all quantitative. The *MPG*, *Cylinders*, *Displacement*, *horsepower*, *weight* and *acceleration* categories are all made up of quantitative data. 2. What is the range, mean, and standard deviation of each predictor? Use `range()` [`pandas.DataFrame.min` and `max`] function.

```
print("Mins & Maxes:")
```

```
## [1] "Mins & Maxes:"
```

```
# I tried to do this in a 'for loop' but I couldn't get it to work!
# I just hard-coded it, sorry (I'm a Python guy)
cat("MPG:",range(autodata$mpg),"\n")
```

```
## MPG: 9 46.6
```

```
cat("Cylinders:",range(autodata$cylinders),"\n")
```

```
## Cylinders: 3 8
```

```
cat("Displacement:",range(autodata$displacement),"\n")
```

```
## Displacement: 68 455
```

```
cat("Horsepower:",range(autodata$horsepower),"\n")
```

```
## Horsepower: 46 230
```

```
cat("Weight:",range(autodata$weight),"\n")
```

```
## Weight: 1613 5140
```

```
cat("Acceleration:",range(autodata$acceleration),"\n")
```

```
## Acceleration: 8 24.8
```

```
cat("Year:",range(autodata$year),"\n")
```

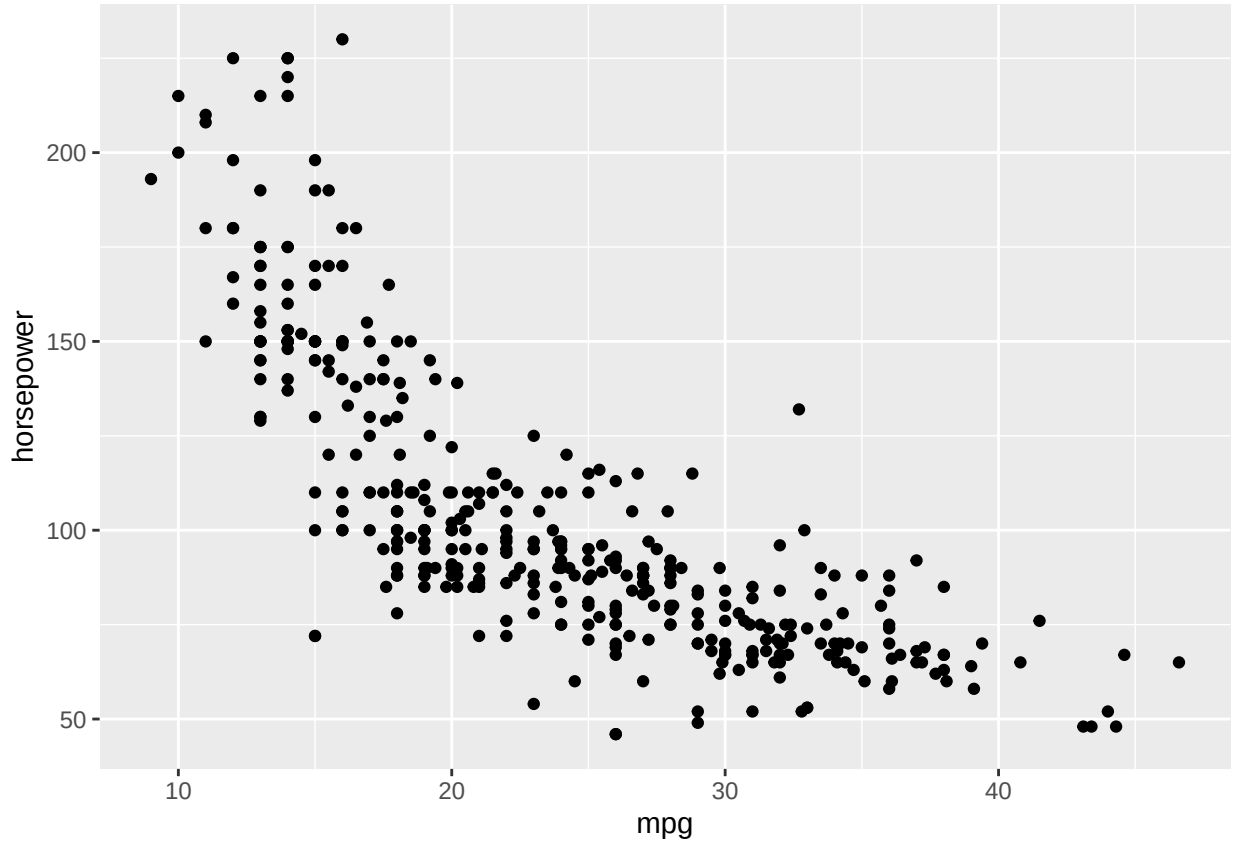
```
## Year: 70 82
```

```
cat("Origin",range(autodata$origin),"\n")
```

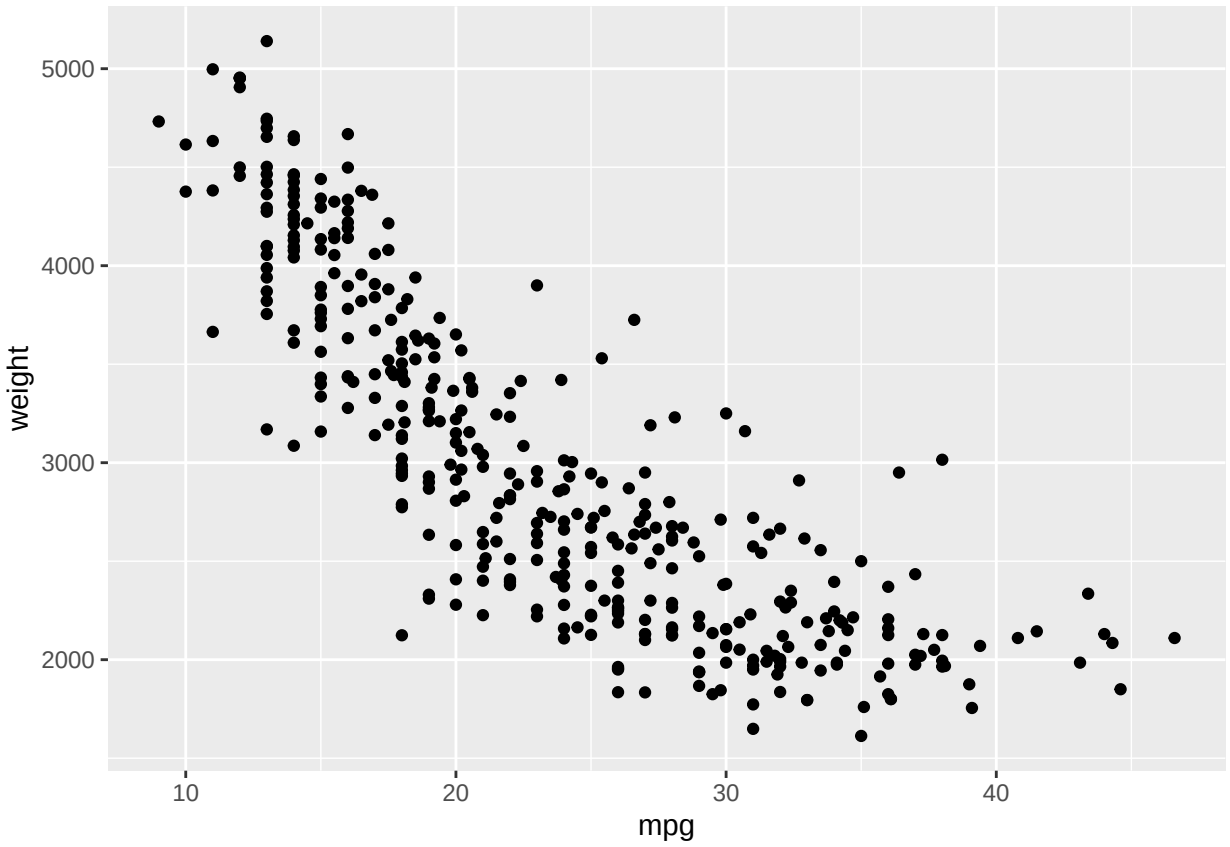
```
## Origin 1 3
```

3. Investigate the predictors graphically using plots. Create plots highlighting relationships between predictors. See [1] for a ggplot cheatsheet.

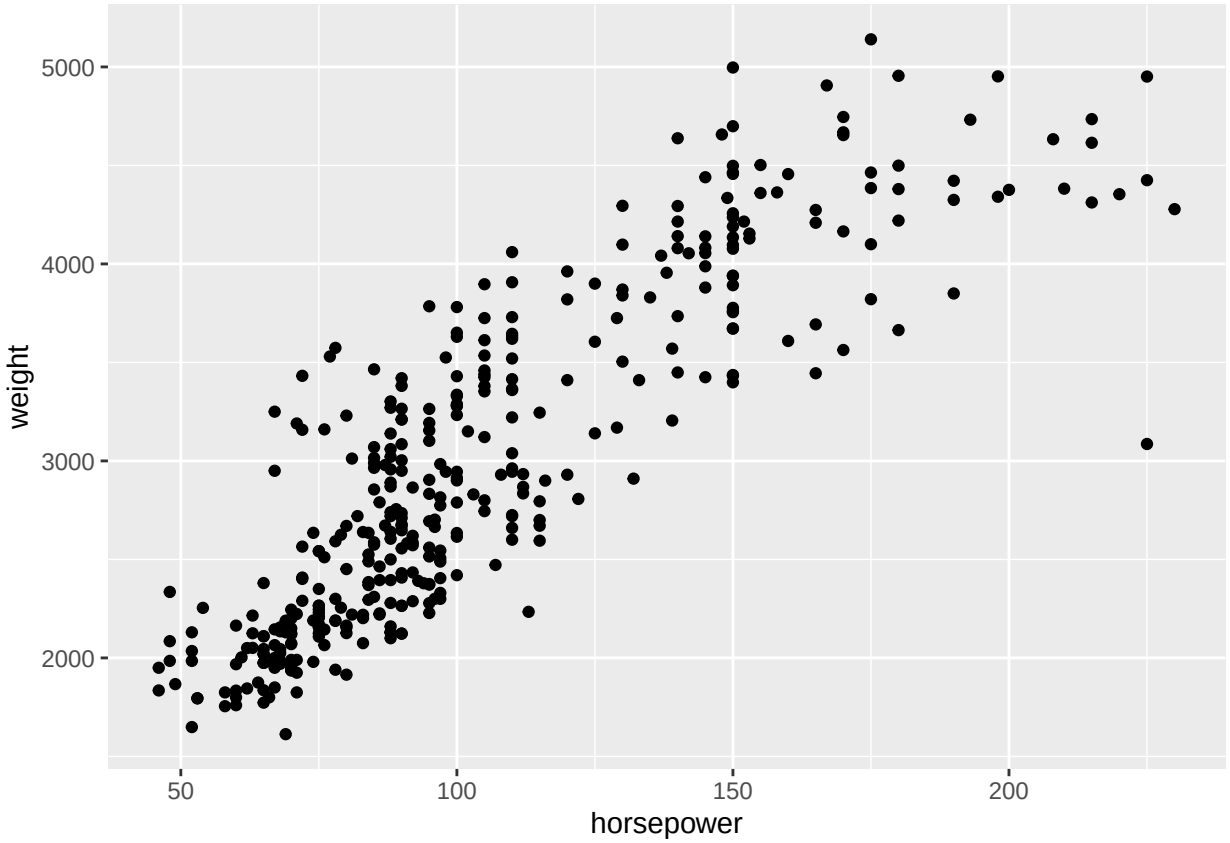
```
library(ggplot2)
# ggplot(autodata,aes(x=mpg,y=year))
# how you you get ggplot to show points? It just produces an empty background!

qplot(x=mpg,y=horsepower,data=autodata)
```



```
qplot(x=mpg,y=weight,data=autodata)
```




```
qplot(x=horsepower,y=weight,data=autodata)
```




4. Compute the matrix of correlations between variables using the function `cor()` [P: `pandas.DataFrame.corr`]. Exclude the `name` variable.
Matrix of Coreelations:

```
drops <- c("name")
auto2 <- autodata[ , !(names(autodata) %in% drops)]
# modifed this from Stack Overflow
cor(auto2)
```

```
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
## horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
## weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
## acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
##              acceleration       year     origin
## mpg             0.4233285  0.5805410  0.5652088
## cylinders      -0.5046834 -0.3456474 -0.5689316
## displacement   -0.5438005 -0.3698552 -0.6145351
## horsepower     -0.6891955 -0.4163615 -0.4551715
## weight         -0.4168392 -0.3091199 -0.5850054
## acceleration    1.0000000  0.2903161  0.2127458
## year            0.2903161  1.0000000  0.1815277
## origin          0.2127458  0.1815277  1.0000000
```

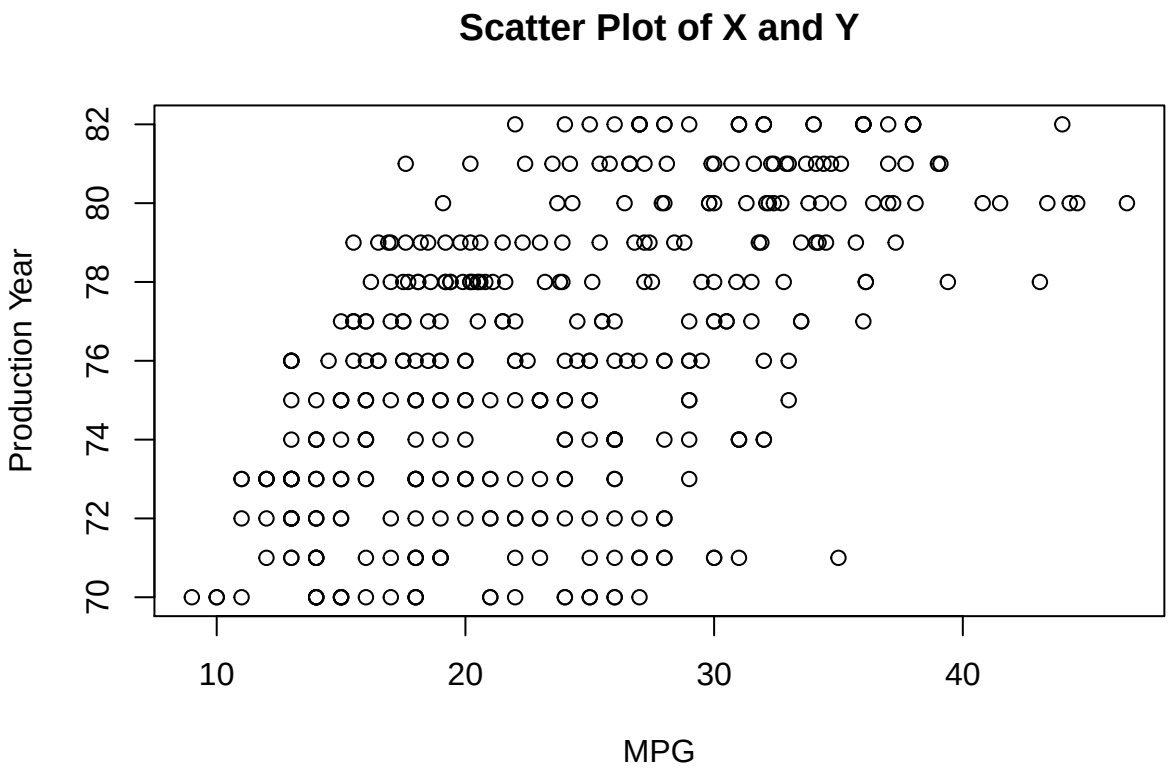
5. Use the `lm()` function to perform a multiple linear regression with `mpg` as the response. [P: using `rpy` package is acceptable] Exclude `name` as a predictor, since it is qualitative. Briefly comment on the output: What is the relationship between the predictors? What does the coefficient for `year` variable suggest?

```
fits <- lm(mpg ~ cylinders+displacement + horsepower+weight+acceleration+year+origin ,data=autodata)
summary(fits)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + year + origin, data = autodata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.5903 -2.1565 -0.1169  1.8690 13.0604
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -17.218435   4.644294  -3.707  0.00024 ***
## cylinders     -0.493376   0.323282  -1.526  0.12780
## displacement   0.019896   0.007515   2.647  0.00844 **
## horsepower    -0.016951   0.013787  -1.230  0.21963
## weight        -0.006474   0.000652  -9.929  < 2e-16 ***
## acceleration   0.080576   0.098845   0.815  0.41548
## year           0.750773   0.050973  14.729  < 2e-16 ***
## origin         1.426141   0.278136   5.127 4.67e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.328 on 384 degrees of freedom
## Multiple R-squared:  0.8215, Adjusted R-squared:  0.8182
## F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

```
plot(autodata$mpg,autodata$year,main='Scatter Plot of X and Y',
     xlab='MPG',ylab='Production Year')
abline(fits)
```

```
## Warning in abline(fits): only using the first two of 8 regression coefficients
```




None of the predictors seem to relate too much to the mpg feature. The coefficient for the *year* category (around 0.75) indicates that the y-intercept of the best fit line crosses the y-axis at the year 1975. This would indicate that Cars made before 1975 would have a negative horsepower - which is obviously not the case.

6. Use the symbols * and : to fit linear regression models with interaction effects. What do you observe?

```
fits2 <- lm(mpg ~ cylinders:displacement:horsepower:weight:acceleration:year:origin ,data=autodata)
summary(fits2)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders:displacement:horsepower:weight:acceleration:year:origin,
##     data = autodata)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.0525  -3.8068  -0.8598   3.3632  19.2309
##
## Coefficients:
##                                                                    Estimate
## (Intercept)                                                       2.902e+01
## cylinders:displacement:horsepower:weight:acceleration:year:origin -8.136e-12
##                                                                  Std. Error
## (Intercept)                                                       3.782e-01
```

```
## cylinders:displacement:horsepower:weight:acceleration:year:origin  3.866e-13
##                                                                   t value
## (Intercept)                                                         76.72
## cylinders:displacement:horsepower:weight:acceleration:year:origin  -21.05
##                                                                   Pr(>|t|)
## (Intercept)                                                         <2e-16 ***
## cylinders:displacement:horsepower:weight:acceleration:year:origin   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.347 on 390 degrees of freedom
## Multiple R-squared:  0.5318, Adjusted R-squared:  0.5306
## F-statistic:   443 on 1 and 390 DF,  p-value: < 2.2e-16
```

Once again, there seems to be no correction between MPG and the other features, wven when we build regression models with interactions.

7. Try a few different transformations of variables, such as $\log(X)$, $\sqrt{X}$, $X^2$. What do you observe?

```
fit4 <- lm(mpg~(weight)**2,data=autodata)
summary(fit4)
```

```
##
## Call:
## lm(formula = mpg ~ (weight)^2, data = autodata)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.9736  -2.7556  -0.3358   2.1379  16.5194
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 46.216524   0.798673   57.87   <2e-16 ***
## weight      -0.007647   0.000258  -29.64   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.333 on 390 degrees of freedom
## Multiple R-squared:  0.6926, Adjusted R-squared:  0.6918
## F-statistic: 878.8 on 1 and 390 DF,  p-value: < 2.2e-16
```

## Problem 3 [25%]

Using equation (3.4) in ISL, argue that in the case of simple linear regression, the least squares line always passes through the point $(\bar{x}, \bar{y})$.

Equation (3.4) in ISL:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

and

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

Simple linear regression produces a 'line of best fit' using the two coefficients, $\beta_0$ and $\beta_1$ as defined above. This fit takes the form of $f(X) = \beta_0 + \beta_1 X + \epsilon$. Given the difinition of $\beta_0$, we can rewrite this fit as: $f(X) = \bar{y} - \hat{\beta}_1(\bar{X} + X)$. Considering the defintion of $\bar{X}$ and $\bar{y}$ (as averages), the subsequenct math then follows that the regression will pass through some point given by $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$, or some simply,$(\bar{x}, \bar{y})$.

## Problem 4 [25%]

It is claimed in the ISL book that in the case of simple linear regression of $Y$ onto $X$, the $R^2$ statistic (3.17) is equal to the square of the correlation between $X$ and $Y$ (3.18). Prove that this is the case. For simplicity, you may assume that $\bar{x} = \bar{y} = 0$.
The $R^2$ metric is defined as (James,70-3.17):

$$R^2 \equiv 1 - \frac{RSS}{TSS}$$

Where RSS is the *residual sum of squares* and TSS is the *total sum of squares*. The correlation coefficient is defined as:

$$Cor(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Thus if we take $\hat{x} = \hat{y} = 0$ and square the function, we arrive at:

$$r^2 = \big(Cor(X,Y)\big)^2 = \frac{\big(\sum_{i=1}^{n} x_i y_i\big)^2}{\sum_{i=1}^{n}(x_i)^2 \sum_{i=1}^{n}(y_i)^2}$$

Thus, the RSS metric becomes $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ and the TSS metric becomes $\sum_{i=1}^{n}(y_i)^2$. This allows for the insertion of RSS and TSS into the $(Cor(X,Y))^2$ function and thus we can see that $r^2$ and $R^2$ are identical measurements.

## References

Each reference is a link. Please open the PDF in a viewer if it is not working on the website.


1. R GGPlot cheat sheet
2. Python Pandas data visualization
3. R For Data Science
4. Cheatsheets fffTheThe