HW10

111078513

Help By 108078467

Question 1) We will use the interactive_regression() function from CompStatsLib again – Windows users please make sure your desktop scaling is set to 100% and RStudio zoom is 100%; alternatively, run R from the Windows Command Prompt.

a. Comparing scenarios 1 and 2, which do we expect to have a stronger R^2 ?

Because in Scenario 1, most of the points are close to the regression line, which predicts a larger R^2 value.

b. Comparing scenarios 3 and 4, which do we expect to have a stronger R^2 ?

Because in Scenario 3, most of the points are close to the regression line, which predicts a larger R^2 value.

c. Comparing scenarios 1 and 2, which do we expect has bigger/smaller SSE, SSR, and SST? (intuitively)

We expect scenarios 1 has bigger SSR and smaller SSE; on the other hand, We expect scenarios 2 has smaller SSR and bigger SSE. However, it's hard to expect which one's SST is bigger.

d. Comparing scenarios 3 and 4, which do we expect has bigger/smaller SSE, SSR, and SST? (intuitively)

We expect scenarios 3 has bigger SSR and smaller SSE; on the other hand, We expect scenarios 4 has smaller SSR and bigger SSE. However, it's hard to expect which one's SST is bigger.

Question 2) Let's analzye the programmer_salaries.txt dataset we saw in class.

```
prog_sal <- read.csv("programmer_salaries.txt", sep="\t")
```

a. Use the lm() function to estimate the regression model [Salary ~ Experience + Score + Degree]Show the beta coefficients, R2, and the first 5 values of y $(fitted.values)and($residuals)

```
prog_sal_lm <- lm(Salary~Experience + Score + Degree, data = prog_sal)
prog_sal_lm$coefficients
```

```
## (Intercept)  Experience       Score      Degree
##    7.944849    1.147582    0.196937    2.280424
```

```
summary(prog_sal_lm)$r.squared
```

```
## [1] 0.8467961
```

```
head(prog_sal_lm$fitted.values)
```

```
##        1        2        3        4        5        6 
## 27.89626 37.95204 26.02901 32.11201 36.34251 38.24380
```

```
head(prog_sal_lm$residuals)
```

```
##          1          2          3          4          5          6 
## -3.8962605  5.0479568 -2.3290112  2.1879860 -0.5425072 -0.2437966
```

b. Use only linear algebra and the geometric view of regression to estimate the regression yourself:

(i) Create an X matrix that has a first column of 1s followed by columns of the independent variables

```
x <- as.matrix(data.frame("1" = rep(1, length.out = nrow(prog_sal))
                          , prog_sal[, 1:3]))
```

(ii) Create a y vector with the Salary values

```
y <- as.vector(prog_sal[, 4])
```

(iii) Compute the beta_hat vector of estimated regression coefficients

```
tanspose_x <- t(x)
beta_hat <- solve(tanspose_x %*% x) %*% tanspose_x %*% y; beta_hat
```

```
##                [,1]
## X1         7.944849
## Experience 1.147582
## Score      0.196937
## Degree     2.280424
```

(iv) Compute a y_hat vector of estimated y values, and a res vector of residuals

```
y_hat <- x %*% beta_hat; head(y_hat)
```

```
##          [,1]
## [1,] 27.89626
## [2,] 37.95204
## [3,] 26.02901
## [4,] 32.11201
## [5,] 36.34251
## [6,] 38.24380
```

```
res <- y - y_hat; head(res)
```

```
##            [,1]
## [1,] -3.8962605
## [2,]  5.0479568
## [3,] -2.3290112
## [4,]  2.1879860
## [5,] -0.5425072
## [6,] -0.2437966
```

(v) Using only the results from (i) – (iv), compute SSR, SSE and SST

```
SSR <- sum((y_hat - mean(y))**2); SSR
```

```
## [1] 507.896
```

```
SSE <-  sum((y - y_hat)**2); SSE
```

```
## [1] 91.88949
```

```
SST <- SSR + SSE; SST
```

```
## [1] 599.7855
```

c. Compute R2 for in two ways, and confirm you get the same results

(i) Use any combination of SSR, SSE, and SST

```
R2 <- SSR/SST; R2
```

```
## [1] 0.8467961
```

(ii) Use the squared correlation of vectors y and y_hat

```
R2 <- (cor(y, y_hat)^2)[1, 1] ; R2
```

```
## [1] 0.8467961
```

Question 3) We're going to take a look back at the early heady days of global car manufacturing, when American, Japanese, and European cars competed to rule the world. Take a look at the data set in file auto-data.txt. We are interested in explaining what kind of cars have higher fuel efficiency (mpg).

```
auto <- read.table("auto-data.txt", header=FALSE, na.strings = "?")
names(auto) <- c("mpg", "cylinders", "displacement", "horsepower", "weight",
                 "acceleration", "model_year", "origin", "car_name")
```

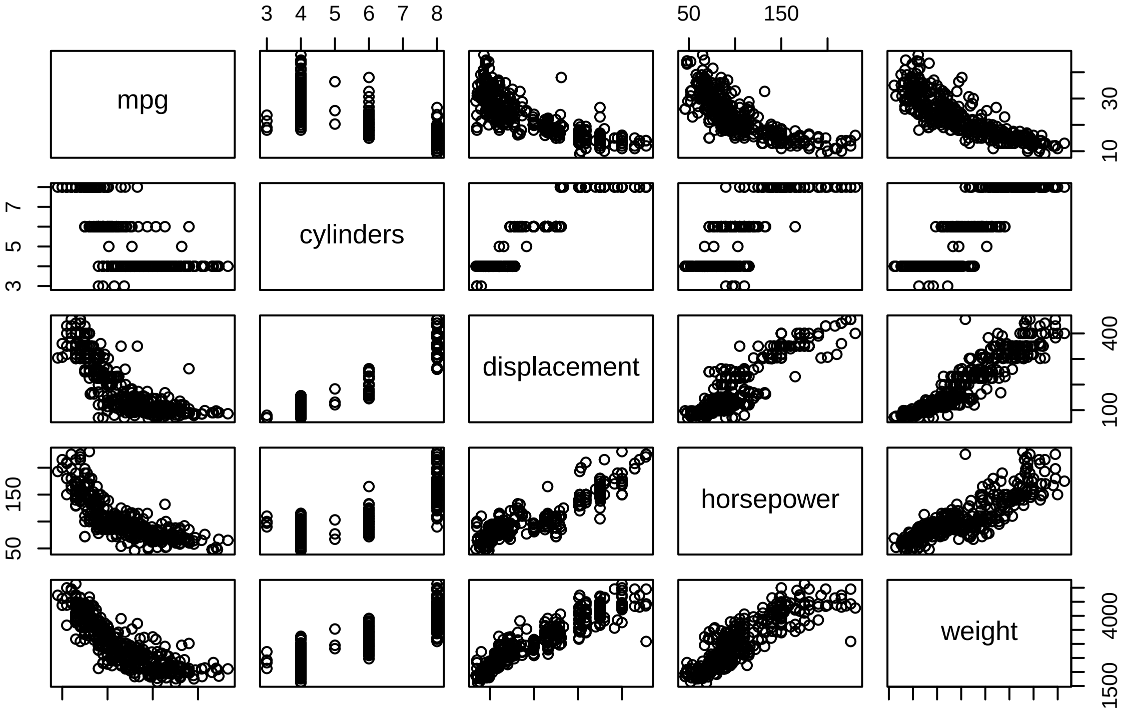
a. Let's first try exploring this data and problem:

(i) Visualize the data as you wish

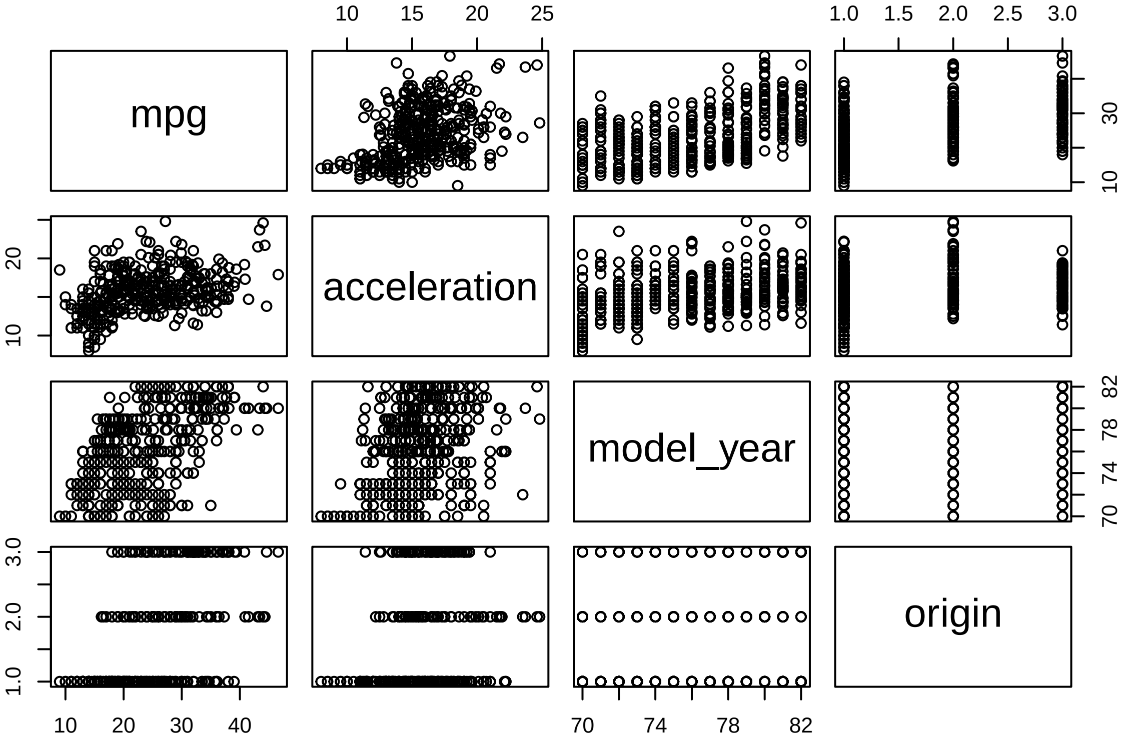
```
library(car)
```

```
## Loading required package: carData
```

```
pairs(~mpg + cylinders + displacement + horsepower + weight ,data=auto)
```



```
pairs(~mpg + acceleration + model_year + origin ,data=auto)
```



(ii) Report a correlation table of all variables, rounding to two decimal places

```
auto_cor <- cor(auto[, 1:8], use="pairwise.complete.obs"); round(auto_cor, 2)
```

```
##                   mpg cylinders displacement horsepower weight acceleration
## mpg            1.00     -0.78        -0.80      -0.78  -0.83         0.42
## cylinders     -0.78      1.00         0.95       0.84   0.90        -0.51
## displacement -0.80      0.95         1.00       0.90   0.93        -0.54
## horsepower   -0.78      0.84         0.90       1.00   0.86        -0.69
## weight       -0.83      0.90         0.93       0.86   1.00        -0.42
## acceleration  0.42     -0.51        -0.54      -0.69  -0.42         1.00
## model_year    0.58     -0.35        -0.37      -0.42  -0.31         0.29
## origin        0.56     -0.56        -0.61      -0.46  -0.58         0.21
##              model_year origin
## mpg                0.58   0.56
## cylinders         -0.35  -0.56
## displacement      -0.37  -0.61
## horsepower        -0.42  -0.46
## weight            -0.31  -0.58
## acceleration       0.29   0.21
## model_year         1.00   0.18
## origin             0.18   1.00
```

(iii) From the visualizations and correlations, which variables appear to relate to mpg?

From correlations, it seems like “cylinders”, “displacement”, “horsepower”, and “weight” are highly related to mpg.

(iv) Which relationships might not be linear?

According to the plots we drew, “cylinders” and “origin” might not be linear.

(v) Are there any pairs of independent variables that are highly correlated?

```
for(i in 2:8){
  for(j in 2:i){
    data = auto_cor[i, j]
      if(abs(auto_cor[i, j]) > 0.7 & i != j){
        print(paste(rownames(auto_cor)[i], "and", colnames(auto_cor)[j]))
      }
  }
}
```

```
## [1] "displacement and cylinders"
## [1] "horsepower and cylinders"
## [1] "horsepower and displacement"
## [1] "weight and cylinders"
## [1] "weight and displacement"
## [1] "weight and horsepower"
```

b. Let’s create a linear regression model where mpg is dependent upon all other suitable variables (Note: origin is categorical with three levels, so use factor(origin) in lm(. . .) to split it into two dummy variables)

(i) Which independent variables have a ‘significant’ relationship with mpg at 1% significance?

```
auto_lm <- lm(mpg~ cylinders + displacement + horsepower + weight
              + acceleration + model_year + factor(origin) ,data=auto)
auto_lm_coef <- summary(auto_lm)$coefficients[2:9, ]
rownames(auto_lm_coef)[auto_lm_coef[, 4]< 0.01]
```

```
## [1] "displacement"    "weight"          "model_year"      "factor(origin)2"
## [5] "factor(origin)3"
```

(ii) Looking at the coefficients, is it possible to determine which independent variables are the most effective at increasing mpg? If so, which ones, and if not, why not? (hint: units!)

It is possible to determine which IVs have the strongest linear relationship with the mpg (aka, our DV) by looking at the magnitudes of the coefficients in the linear regression model. However, it is not appropriate to compare the magnitudes of the coefficients for variables that have different units. Therefore, we can't determine which IV are the most effective at increasing mpg as all the IVs have different unit.

c. Let's try to resolve some of the issues with our regression model above.

(i) Create fully standardized regression results: are these slopes easier to compare?

```
auto_std <- data.frame(scale(auto[1:8]))
auto_regr_std <- lm(mpg~ cylinders + displacement + horsepower + weight
              + acceleration + model_year + factor(origin) ,data=auto_std)
summary(auto_regr_std)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + model_year + factor(origin), data = auto_std)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.15270 -0.26593 -0.01257  0.25404  1.70942
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -0.13323    0.03174  -4.198 3.35e-05 ***
## cylinders                         -0.10658    0.06991  -1.524  0.12821
## displacement                       0.31989    0.10210   3.133  0.00186 **
## horsepower                        -0.08955    0.06751  -1.326  0.18549
## weight                            -0.72705    0.07098 -10.243  < 2e-16 ***
## acceleration                       0.02791    0.03465   0.805  0.42110
## model_year                         0.36760    0.02450  15.005  < 2e-16 ***
## factor(origin)0.532551687239475    0.33649    0.07247   4.643 4.72e-06 ***
## factor(origin)1.7793491667766      0.36505    0.07072   5.162 3.93e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.423 on 383 degrees of freedom
##   (6 observations deleted due to missingness)
## Multiple R-squared:  0.8242, Adjusted R-squared:  0.8205
## F-statistic: 224.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

As the standardized coefficients are all within -1 to 1 without any significantly larger values compared to others, it is harder to distinguish.

(ii) Regress mpg over each nonsignificant independent variable, individually. Which ones become significant when we regress mpg over them individually?

```
nonsignificant_coef <- rownames(auto_lm_coef)[auto_lm_coef[, 4] > 0.01]
for (item in nonsignificant_coef) {
  formula_str <- paste("mpg~", item)
  temp_regr <- lm(as.formula(formula_str), data = auto_std)
  p_value <- summary(temp_regr)$coefficients[2, 4]
  if(p_value < 0.01){
    print(paste("P-value for", item, "is", p_value, ": significant"))
  }
}
```

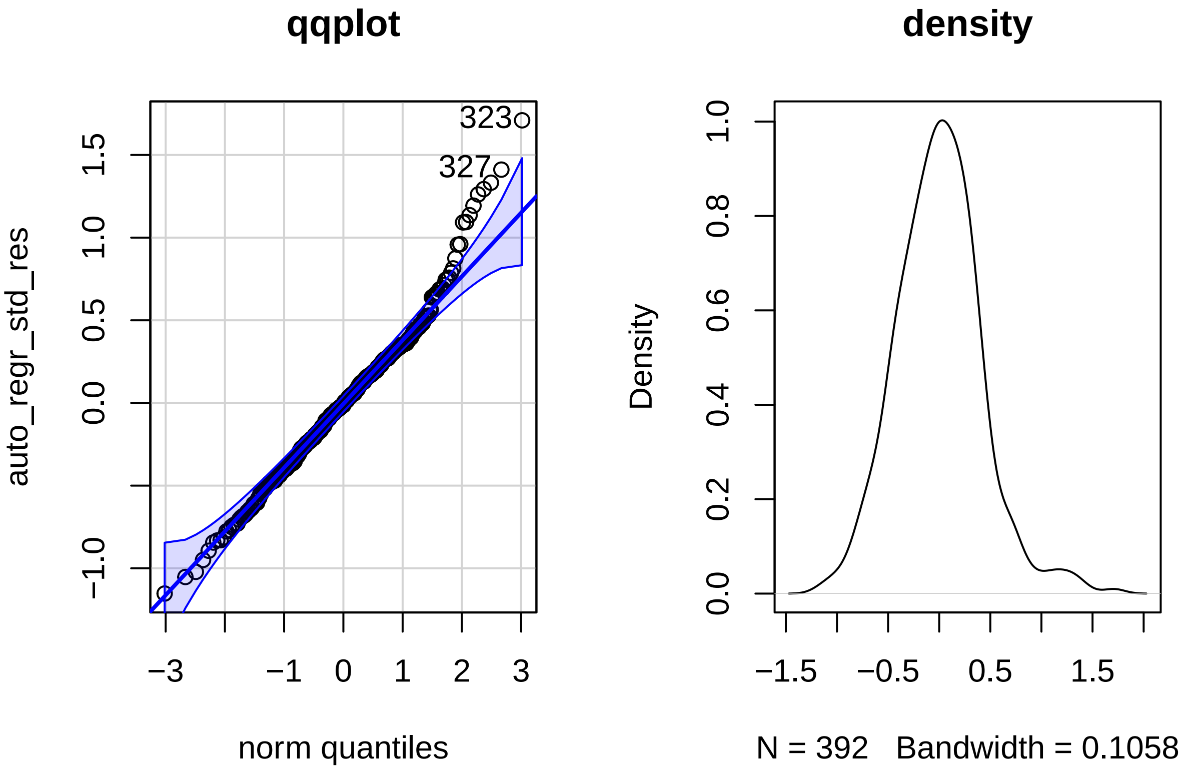
```
## [1] "P-value for cylinders is 4.50399224617619e-81 : significant"
## [1] "P-value for horsepower is 7.03198902940446e-81 : significant"
## [1] "P-value for acceleration is 1.82309153507868e-18 : significant"
```

(iii) Plot the distribution of the residuals: are they normally distributed and centered around zero?

```
par(mfrow = c(1, 2))
auto_regr_std_res <- auto_regr_std$residuals
qqPlot(auto_regr_std_res, main = "qqplot")
```

```
## 323 327
## 321 325
```

```
plot(density(auto_regr_std_res), main = "density")
```



According to the qqplot we drew, we can see that, except for some deviations in the upper right corner of the plot, the rest of the plot is generally in the shape of a normal distribution. In addition, we can also see from the density plot that the distribution of residuals is roughly bell-shaped. Based on these two plots, we can say that the residuals are normally distributed.