Assignment: ASSIGNMENT 6

Name: Shekhar, Manish

Date: 2021-05-03

Set the working directory to the root of your DSC 520 directory

setwd("/Users/mshekhar/Desktop/R Programming/DSC520/stats_for_data_science/stats_for_data_science")

Load the `data/r4ds/heights.csv` to

```
setwd("/Users/mshekhar/Desktop/R Programming/DSC520/stats_for_data_science/stats_for_data_science")
heights_df <- read.csv("./heights.csv")

## Load the ggplot2 library
library(ggplot2)
```

Fit a linear model using the `age` variable as the predictor and `earn` as the outcome

```
# check if there are any nulls in age or earn in the heights_df
str(heights_df)
```

```
## 'data.frame':    1192 obs. of  6 variables:
##  $ earn  : num  50000 60000 30000 50000 51000 9000 29000 32000 2000 27000 ...
##  $ height: num  74.4 65.5 63.6 63.1 63.4 ...
##  $ sex   : chr  "male" "female" "female" "female" ...
##  $ ed    : int  16 16 16 16 17 15 12 17 15 12 ...
##  $ age   : int  45 58 29 91 39 26 49 46 21 26 ...
##  $ race  : chr  "white" "white" "white" "other" ...
```

```
sum(is.na(heights_df$earn))
```

```
## [1] 0
```

```
sum(is.na(heights_df$age))
```

```
## [1] 0
```

```
library(caTools)
set.seed(123)
# As there is no NA in the data, we do not need na.action argument in the lm()
# creating the linear model and storing the model object in the age_lm variable
age_lm <- lm(earn ~ age, heights_df)
```

View the summary of your model using `summary()`

```
# check the model statistics of model with all the data
summary(age_lm)
```

```
##
## Call:
## lm(formula = earn ~ age, data = heights_df)
##
## Residuals:
```

```
##    Min     1Q Median     3Q    Max
## -25098 -12622  -3667   6883 177579
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 19041.53    1571.26  12.119  < 2e-16 ***
## age            99.41      35.46   2.804  0.00514 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19420 on 1190 degrees of freedom
## Multiple R-squared:  0.006561,   Adjusted R-squared:  0.005727
## F-statistic:  7.86 on 1 and 1190 DF,  p-value: 0.005137
```

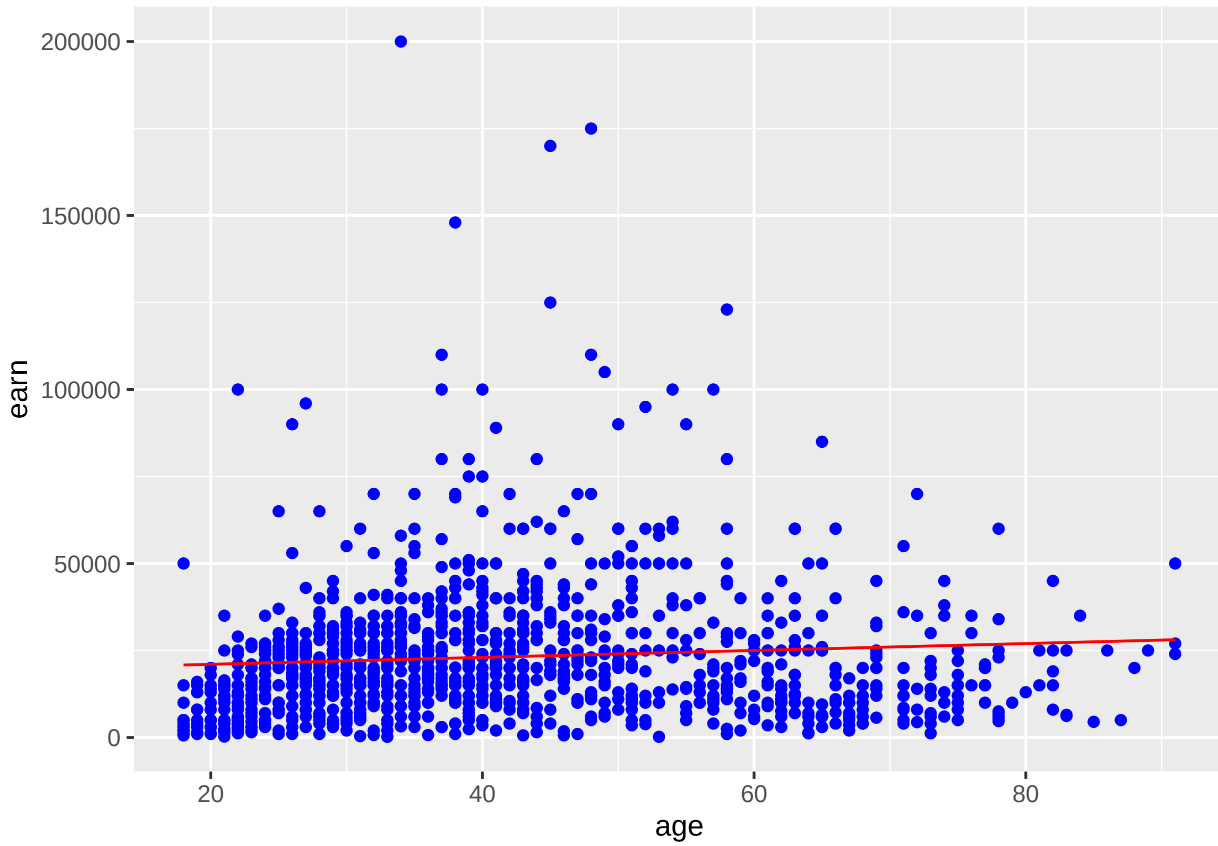
Creating predictions using `predict()`

```r
# creating an age predict data frame to predict earnings using the model
age_testing_df <- data.frame(age = c(17,22,25,35,55,51,62))
# make earning prediction using age_testing_df
age_predict_df <- data.frame(age = age_testing_df, earn = predict(age_lm, newdata = age_testing_df))
# check the data in the data frame
age_predict_df
```

```
##   age     earn
## 1  17 20731.42
## 2  22 21228.45
## 3  25 21526.67
## 4  35 22520.73
## 5  55 24508.84
## 6  51 24111.22
## 7  62 25204.68
```

Plot the predictions against the original data

```r
# plotting 1. scatterplot with age on x and earn on y-axis
# adding a line with prediction for ages in heights_df on y-axis and all ages on x-axis
ggplot(data = heights_df, aes(y = earn, x = age)) +
    geom_point(color='blue') +
    geom_line(color='red',data = heights_df, aes(y=predict(age_lm, newdata = heights_df), x=age))
```




```r
# getting the mean earning
mean_earn <- mean(heights_df$earn)
mean_earn
```

```
## [1] 23154.77
```

```r
## Corrected Sum of Squares Total
#sst <- sum((mean_earn - heights_df$earn)^2)
sst <- sum((heights_df$earn-mean_earn)^2)
sst
```

```
## [1] 451591883937
```

```r
## Corrected Sum of Squares for Model
## To be able to show the same model evaluation stats will let model predict using
## training data -> heights_df
## recreating age_predict_df by predicting on heights_df
age_predict_df <- data.frame(age = heights_df$age, earn = predict(age_lm, newdata = heights_df))
ssm <- sum((age_predict_df$earn-mean_earn)^2)
ssm
```

```
## [1] 2963111900
```

```r
## Residuals
residuals <- heights_df$earn - age_predict_df$earn

## Sum of Squares for Error
sse <- sum(residuals^2)
sse
```

```
## [1] 448628772037
```

```r
## R Squared R^2 = SSM\SST
r_squared <- ssm/sst
r_squared
```

```
## [1] 0.006561482
```

```r
## Number of observations
n <- nrow(heights_df)
n
```

```
## [1] 1192
```

```r
## Number of regression parameters
## In simple regression, when we only have one predictor, p = 1
## I am keeping p as 2 as given to avoid division by 0 later
p <- 2

## Corrected Degrees of Freedom for Model (p-1)
dfm <- p-1
dfm
```

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

```r
## Degrees of Freedom for Error (n-p)
dfe <- n-p
dfe
```

```
## [1] 1190
```

```r
## Corrected Degrees of Freedom Total:   DFT = n - 1
dft <- n-1
dft
```

```
## [1] 1191
```

```r
## Mean of Squares for Model:   MSM = SSM / DFM
msm <- ssm/dfm
msm
```

```
## [1] 2963111900
```

```r
## Mean of Squares for Error:   MSE = SSE / DFE
mse <- sse/dfe
mse
```

```
## [1] 376998968
```

```r
## Mean of Squares Total:   MST = SST / DFT
mst <- sst/dft
mst
```

```
## [1] 379170348
```

```r
## F Statistic F = MSM/MSE
f_score <- msm/mse
f_score
```

```
## [1] 7.859735
```

```r
## Adjusted R Squared R2 = 1 - (1 - R2)(n - 1) / (n - p)
adjusted_r_squared <- 1-((1-r_squared)*dft)/dfe
adjusted_r_squared
```

```
## [1] 0.005726659
```

```r
## Calculate the p-value from the F distribution
p_value <- pf(f_score, dfm, dft, lower.tail=F)
p_value
```

```
## [1] 0.005136826
```