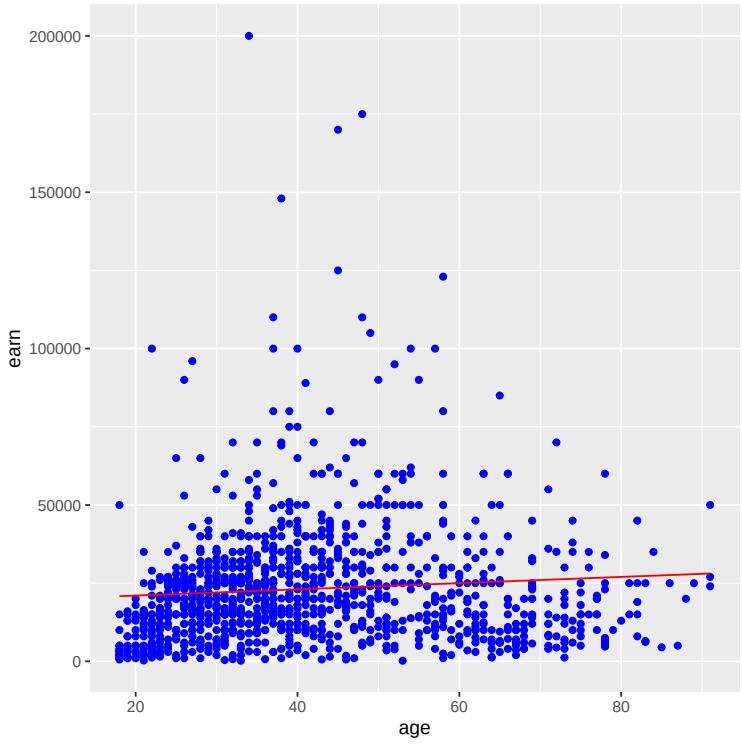
July 25, 2023

The results below are generated from an R script.

```
# Assignment: ASSIGNMENT 6
# Name: Tang, Xin
# Date: 2023-07-25

## Set the working directory to the root of your DSC 520 directory
setwd("~/dsc520")

## Load the `data/r4ds/heights.csv` to
heights_df <- read.csv("data/r4ds/heights.csv")

## Load the ggplot2 library
library(ggplot2)

## Fit a linear model using the `age` variable as the predictor and `earn` as the outcome
age_lm <- lm(formula = earn ~ age, data = heights_df)

## View the summary of your model using `summary()`
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()`
age_predict_df <- data.frame(earn = predict(age_lm, newdata = heights_df), age = heights_df$age)
head(age_predict_df$earn)

## [1] 23514.79 24807.06 21924.29 28087.45 22918.35 21626.08
```

```r
View(age_predict_df)


## Plot the predictions against the original data
ggplot(data = heights_df, aes(y = earn, x = age)) +
  geom_point(color='blue') +
  geom_line(color='red',data = age_predict_df, aes(y = earn, x = age))
```




```r
mean_earn <- mean(heights_df$earn)
## Corrected Sum of Squares Total
sst <- sum((mean_earn - heights_df$earn)^2)
## Corrected Sum of Squares for Model
ssm <- sum((mean_earn - age_predict_df$earn)^2)
## Residuals
residuals <- heights_df$earn - age_predict_df$earn
## Sum of Squares for Error
sse <- sum(residuals^2)
## R Squared R^2 = SSM\SST
r_squared <- ssm/sst

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

## Number of regression parameters
p <- 2
## Corrected Degrees of Freedom for Model (p-1)
dfm <- p-1
## Degrees of Freedom for Error (n-p)
dfe <- n-p
```

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

## Mean of Squares for Model:   MSM = SSM / DFM
msm <- ssm / dfm
## Mean of Squares for Error:   MSE = SSE / DFE
mse <- sse / dfe
## Mean of Squares Total:   MST = SST / DFT
mst <- sst / dft
## F Statistic F = MSM/MSE
f_score <- msm / mse

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

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

The R session information (including the OS info, R version and all packages used):

```r
sessionInfo()

## R version 4.3.1 (2023-06-16 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19045)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_United States.utf8  LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8 LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/Chicago
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] ggplot2_3.4.2
##
## loaded via a namespace (and not attached):
##  [1] vctrs_0.6.2      cli_3.6.1        knitr_1.43       rlang_1.1.1      xfun_0.39
##  [6] highr_0.10       generics_0.1.3   glue_1.6.2       labeling_0.4.2   colorspace_2.1-0
## [11] scales_1.2.1     fansi_1.0.4      grid_4.3.1       munsell_0.5.0    evaluate_0.21
## [16] tibble_3.2.1     lifecycle_1.0.3  compiler_4.3.1   dplyr_1.1.2      pkgconfig_2.0.3
## [21] rstudioapi_0.14  farver_2.1.1     R6_2.5.1         tidyselect_1.2.0 utf8_1.2.3
## [26] pillar_1.9.0     magrittr_2.0.3   tools_4.3.1      withr_2.5.0      gtable_0.3.3

Sys.time()

## [1] "2023-07-25 20:36:39 CDT"
```