R_Project: Regression - Linear Regression/ KNN/ Decision Tree

Celio F

2022-06-30

Data info

- Data Name: Data Science and STEM Salaries
- Database source: https://www.kaggle.com/datasets/jackogozaly/data-science-and-stem-salaries
- Data type: .CSV
- Last Update: October, 2021.
- Search/ Downloaded Date: 29, June, 2022.
- Rows:62643; Columns:29

Purpose:

Use the data and the algorithms listed above to predict a base salary for Computer Science based in some variables, as years of experience, years at the company, race, gender and total yearly compensation.

Linear Regression Algorithm

Steps to get Data into R and necessary libraries.

- 1°: step: reading data set.
- 2°: libraries necessary for the algorithms.

```
DataSc_Sal <- read.csv("~/Downloads/UTD/MachineLearn/R-Project/Regression/DataScienceSalaries.csv", na.s

library(readr)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.6     v dplyr   1.0.9
## v tibble  3.1.7     v stringr 1.4.0
## v tidyr   1.2.0     v forcats 0.5.1
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(tidyr)
library(scales)
```

```
##
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':
##
##     discard

## The following object is masked from 'package:readr':
##
##     col_factor
```

```
library(viridisLite)
library(viridis)
```

```
##
## Attaching package: 'viridis'

## The following object is masked from 'package:scales':
##
##     viridis_pal
```

```
library(ggstatsplot)
```

```
## You can cite this package as:
##      Patil, I. (2021). Visualizations with statistical details: The 'ggstatsplot' approach.
##      Journal of Open Source Software, 6(61), 3167, doi:10.21105/joss.03167
```

```
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(tree)
library(rpart)
library(rpart.plot)
```

Steps to Data Cleaning

- 1°: removing unnecessary columns.
- 2°: check for NA's, and remove them.
- 3°: boxplot to check outliers.
- 4°: check min and max values.

- 5°: set salaries higher than 700.000 to median value.
- 6°: set the years of experience equal median value.

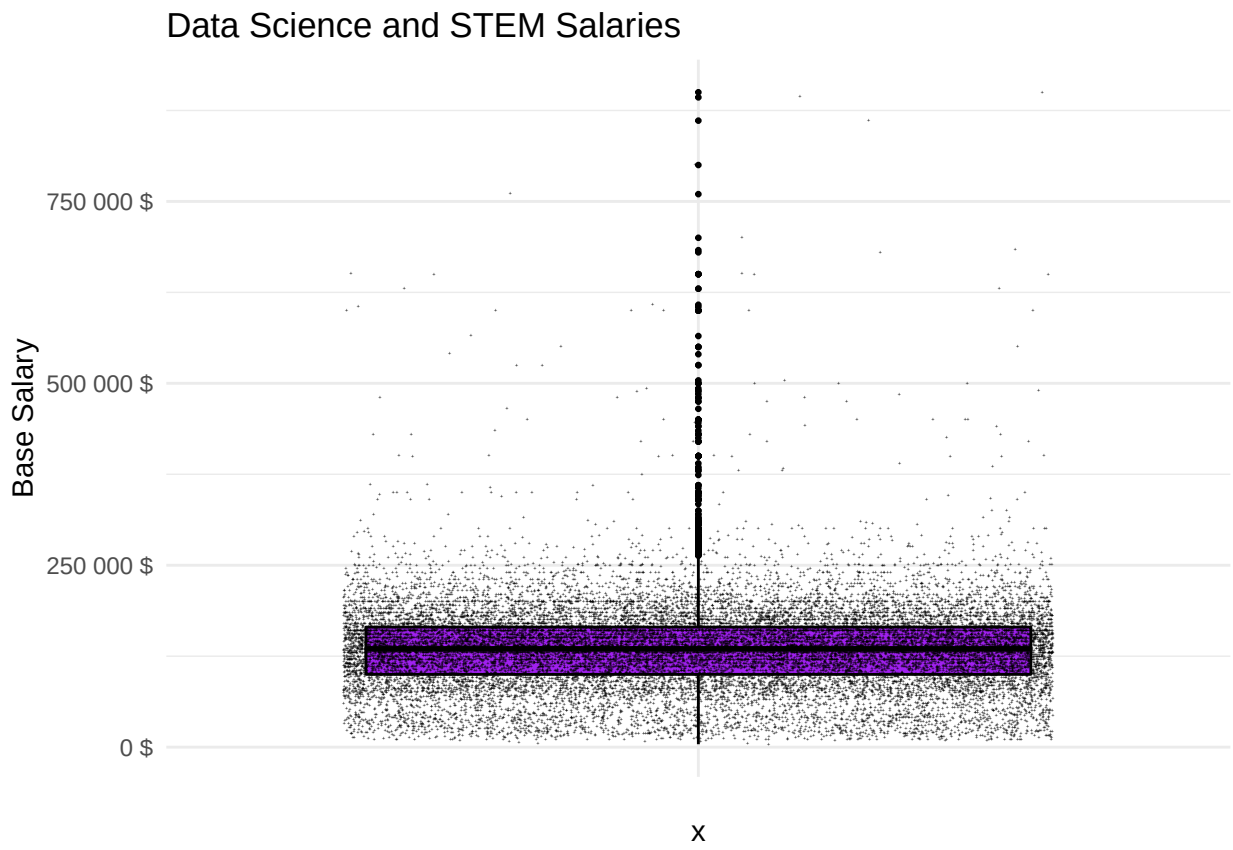
Comments: removing some columns that seems useless for this prediction. There are 15-level, 19540-gender, 808-tag, 400215-race, and 32272-education that has NA's values. Since the data sill have 21575 rows after removing, still enough data to work with the prediction. Make a base salary graph to look how the data are spread, and have an idea where the concentrated salaries are. Since there are few high salaries and few employees with a large experience, i make them equal median value.

```
#removing unnecessary columns
DataSc_Sal$timestamp <- NULL
DataSc_Sal$otherdetails <- NULL
DataSc_Sal$cityid <- NULL
DataSc_Sal$dmaid <- NULL
DataSc_Sal$rowNumber <- NULL

# checking columns with NA's
sapply(DataSc_Sal, function(x) sum(is.na(x)==TRUE))
```

```
##                  company                    level                    title
##                        0                       15                        0
## totalyearlycompensation                 location       yearsofexperience
##                        0                        0                        0
##          yearsatcompany                      tag             basesalary
##                        0                      808                        0
##         stockgrantvalue                    bonus                   gender
##                        0                        0                    19540
##          Masters_Degree        Bachelors_Degree        Doctorate_Degree
##                        0                        0                        0
##              Highschool            Some_College             Race_Asian
##                        0                        0                        0
##              Race_White        Race_Two_Or_More             Race_Black
##                        0                        0                        0
##           Race_Hispanic                     Race                Education
##                        0                    40215                    32272
```

```
# remove NA's rows
DataSc_Sal <- DataSc_Sal %>% drop_na()

# check outliers
```

```
check_outliers <- DataSc_Sal[, c(4, 6, 7, 9, 12, 23)]

# graph to visualize mean
ggplot(check_outliers, aes(x="", y= basesalary)) +
  geom_boxplot(fill='purple', color="black", outlier.size = 0.5) +
  scale_y_continuous(labels = unit_format(scale = 10e-1, unit = "$")) +
  geom_jitter(shape="+", color="black", size=0.4, alpha=1.4) +
  theme_minimal() + ggtitle("Data Science and STEM Salaries") + ylab("Base Salary")
```




```
# check rows with min and max value
min <- which.min(DataSc_Sal$basesalary)
print(DataSc_Sal[min,])
```

```
##                         company level             title totalyearlycompensation
## 17969 Tata Consultancy Services    l1 Software Engineer                   10000
##                location yearsofexperience yearsatcompany    tag basesalary
## 17969 Mumbai, MH, India                 3              1 DevOps       4000
##       stockgrantvalue bonus gender Masters_Degree Bachelors_Degree
## 17969            4000  2000   Male              1                 0
##       Doctorate_Degree Highschool Some_College Race_Asian Race_White
## 17969                0          0            0          1          0
##       Race_Two_Or_More Race_Black Race_Hispanic  Race       Education
## 17969                0          0             0 Asian Master's Degree
```

```
max <- which.max(DataSc_Sal$basesalary)
print(DataSc_Sal[max,])
```

```
##       company               level                 title totalyearlycompensation
## 11570     PwC Partner / Principal Management Consultant                  900000
##          location yearsofexperience yearsatcompany        tag basesalary
## 11570 Raleigh, NC                22              4 Cloud, IoT      9e+05
##       stockgrantvalue bonus gender Masters_Degree Bachelors_Degree
## 11570               0     0   Male              1                0
##       Doctorate_Degree Highschool Some_College Race_Asian Race_White
## 11570                0          0            0          1          0
##       Race_Two_Or_More Race_Black Race_Hispanic  Race       Education
## 11570                0          0             0 Asian Master's Degree
```

```
# set the salaries higer then 700k to median
DataSc_Sal$basesalary[DataSc_Sal$basesalary > 700000] <- median(DataSc_Sal$basesalary)

# set years of experience greater then 30 equal median
DataSc_Sal$yearsofexperience[DataSc_Sal$yearsofexperience > 35] <- median(DataSc_Sal$yearsofexperience)
```

Steps to Adjust some columns (data cleaning)

- 1°: making variables as factors.

```
#making variables as factors
DataSc_Sal$gender <- factor(DataSc_Sal$gender)
DataSc_Sal$Race <- factor(DataSc_Sal$Race)
DataSc_Sal$title <- factor(DataSc_Sal$title)
DataSc_Sal$localtion <- factor(DataSc_Sal$location)
DataSc_Sal$tag <- factor(DataSc_Sal$tag)
DataSc_Sal$Race <- factor(DataSc_Sal$Race)
```

Steps to do Data Exploration

- 1°: some data analysis.

Checking some values from the data, min, max, mean, median and also checking variables type.

```
# some analysis on the data
head(DataSc_Sal)
```

```
##     company level             title totalyearlycompensation          location
## 1    Google    L6 Software Engineer                  400000     Sunnyvale, CA
## 2 Microsoft    61 Software Engineer                  136000       Redmond, WA
## 3    Google    L5 Software Engineer                  337000     San Bruno, CA
## 4 Microsoft    62 Software Engineer                  222000       Seattle, WA
## 5     Blend   IC3 Software Engineer                  187000 San Francisco, CA
## 6    Amazon    L6 Software Engineer                  310000       Seattle, WA
```

```
##   yearsofexperience yearsatcompany                          tag basesalary
## 1                 5              5 Distributed Systems (Back-End)     210000
## 2                 3              2                          DevOps     124000
## 3                 6              6                      Full Stack     177000
## 4                 4              4     API Development (Back-End)     164000
## 5                 5              0                      Full Stack     165000
## 6                15              3                         ML / AI     160000
##   stockgrantvalue bonus gender Masters_Degree Bachelors_Degree Doctorate_Degree
## 1          145000 45000   Male              0                0                1
## 2            1000 11000   Male              0                1                0
## 3          125000 36000   Male              0                1                0
## 4           38000 20000   Male              1                0                0
## 5           22000     0   Male              0                1                0
## 6          150000     0   Male              0                1                0
##   Highschool Some_College Race_Asian Race_White Race_Two_Or_More Race_Black
## 1          0            0          1          0                0          0
## 2          0            0          0          0                1          0
## 3          0            0          1          0                0          0
## 4          0            0          1          0                0          0
## 5          0            0          0          1                0          0
## 6          0            0          1          0                0          0
##   Race_Hispanic        Race         Education         localtion
## 1             0       Asian               PhD     Sunnyvale, CA
## 2             0 Two Or More Bachelor's Degree       Redmond, WA
## 3             0       Asian Bachelor's Degree     San Bruno, CA
## 4             0       Asian   Master's Degree       Seattle, WA
## 5             0       White Bachelor's Degree San Francisco, CA
## 6             0       Asian Bachelor's Degree       Seattle, WA
```

```
summary(DataSc_Sal)
```

```
##    company             level                                    title
##  Length:21575       Length:21575       Software Engineer           :13693
##  Class :character   Class :character   Product Manager             : 1451
##  Mode  :character   Mode  :character   Software Engineering Manager: 1024
##                                        Data Scientist              :  875
##                                        Hardware Engineer           :  781
##                                        Technical Program Manager   :  633
##                                        (Other)                     : 3118
##  totalyearlycompensation   location         yearsofexperience yearsatcompany
##  Min.   :  10000         Length:21575       Min.   : 0.000    Min.   : 0.000
##  1st Qu.: 119000         Class :character   1st Qu.: 3.000    1st Qu.: 0.000
##  Median : 174000         Mode  :character   Median : 6.000    Median : 2.000
##  Mean   : 197855                            Mean   : 7.098    Mean   : 2.704
##  3rd Qu.: 245000                            3rd Qu.:10.000    3rd Qu.: 4.000
##  Max.   :4980000                            Max.   :35.000    Max.   :40.000
##
##                               tag         basesalary     stockgrantvalue
##  Full Stack                     :3697   Min.   :  4000   Min.   :     0
##  Distributed Systems (Back-End):3295   1st Qu.:100000   1st Qu.:     0
##  API Development (Back-End)    :2093   Median :135000   Median : 20000
##  Web Development (Front-End)   :1018   Mean   :133701   Mean   : 44894
##  ML / AI                       : 986   3rd Qu.:165000   3rd Qu.: 55000
##  Product                       : 894   Max.   :700000   Max.   :954000
```

```
##  (Other)                       :9592
##      bonus           gender      Masters_Degree  Bachelors_Degree
##  Min.   :     0   Female: 3876   Min.   :0.000   Min.   :0.0000
##  1st Qu.:  3000   Male  :17594   1st Qu.:0.000   1st Qu.:0.0000
##  Median : 13000   Other :  105   Median :0.000   Median :1.0000
##  Mean   : 18418                  Mean   :0.421   Mean   :0.5066
##  3rd Qu.: 25000                  3rd Qu.:1.000   3rd Qu.:1.0000
##  Max.   :900000                  Max.   :1.000   Max.   :1.0000
##
##  Doctorate_Degree    Highschool       Some_College       Race_Asian
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :0.00000   Median :0.00000   Median :0.00000   Median :1.0000
##  Mean   :0.04283   Mean   :0.01395   Mean   :0.01571   Mean   :0.5287
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:1.0000
##  Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.0000
##
##    Race_White     Race_Two_Or_More    Race_Black      Race_Hispanic
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000
##  Median :0.0000   Median :0.00000   Median :0.00000   Median :0.00000
##  Mean   :0.3559   Mean   :0.03541   Mean   :0.03068   Mean   :0.04941
##  3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000
##  Max.   :1.0000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000
##
##           Race        Education                          localtion
##  Asian      :11406   Length:21575       Seattle, WA         : 2514
##  Black      :  662   Class :character   San Francisco, CA   : 1757
##  Hispanic   : 1065   Mode  :character   New York, NY        : 1605
##  Two Or More:  764                      Redmond, WA         :  754
##  White      : 7678                      Mountain View, CA   :  647
##                                         Bangalore, KA, India:  629
##                                         (Other)             :13669
```

```
str(DataSc_Sal)
```

```
## 'data.frame':    21575 obs. of  25 variables:
##  $ company               : chr  "Google" "Microsoft" "Google" "Microsoft" ...
##  $ level                 : chr  "L6" "61" "L5" "62" ...
##  $ title                 : Factor w/ 15 levels "Business Analyst",..: 12 12 12 12 12 12 12 13 12 12
##  $ totalyearlycompensation: int  400000 136000 337000 222000 187000 310000 113000 620000 98000 180000
##  $ location              : chr  "Sunnyvale, CA" "Redmond, WA" "San Bruno, CA" "Seattle, WA" ...
##  $ yearsofexperience     : num  5 3 6 4 5 15 3 19 9 1 ...
##  $ yearsatcompany        : num  5 2 6 4 0 3 3 7 4 1 ...
##  $ tag                   : Factor w/ 1548 levels "#finance","2 Year Rotational Program",..: 450 427
##  $ basesalary            : num  210000 124000 177000 164000 165000 160000 103000 160000 78000 130000
##  $ stockgrantvalue       : num  145000 1000 125000 38000 22000 150000 0 460000 20000 30000 ...
##  $ bonus                 : num  45000 11000 36000 20000 0 0 10000 0 0 20000 ...
##  $ gender                : Factor w/ 3 levels "Female","Male",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Masters_Degree        : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ Bachelors_Degree      : int  0 1 1 0 1 1 1 1 1 1 ...
##  $ Doctorate_Degree      : int  1 0 0 0 0 0 0 0 0 0 ...
##  $ Highschool            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Some_College          : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ Race_Asian             : int  1 0 1 1 0 1 0 1 1 1 ...
##  $ Race_White             : int  0 0 0 0 1 0 0 0 0 0 ...
##  $ Race_Two_Or_More       : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ Race_Black             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Race_Hispanic          : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ Race                   : Factor w/ 5 levels "Asian","Black",..: 1 4 1 1 5 1 3 1 1 1 ...
##  $ Education              : chr  "PhD" "Bachelor's Degree" "Bachelor's Degree" "Master's Degree" ...
##  $ localtion              : Factor w/ 768 levels "Aachen, NW, Germany",..: 662 554 595 622 598 622 3
```

```
median(DataSc_Sal$basesalary)
```

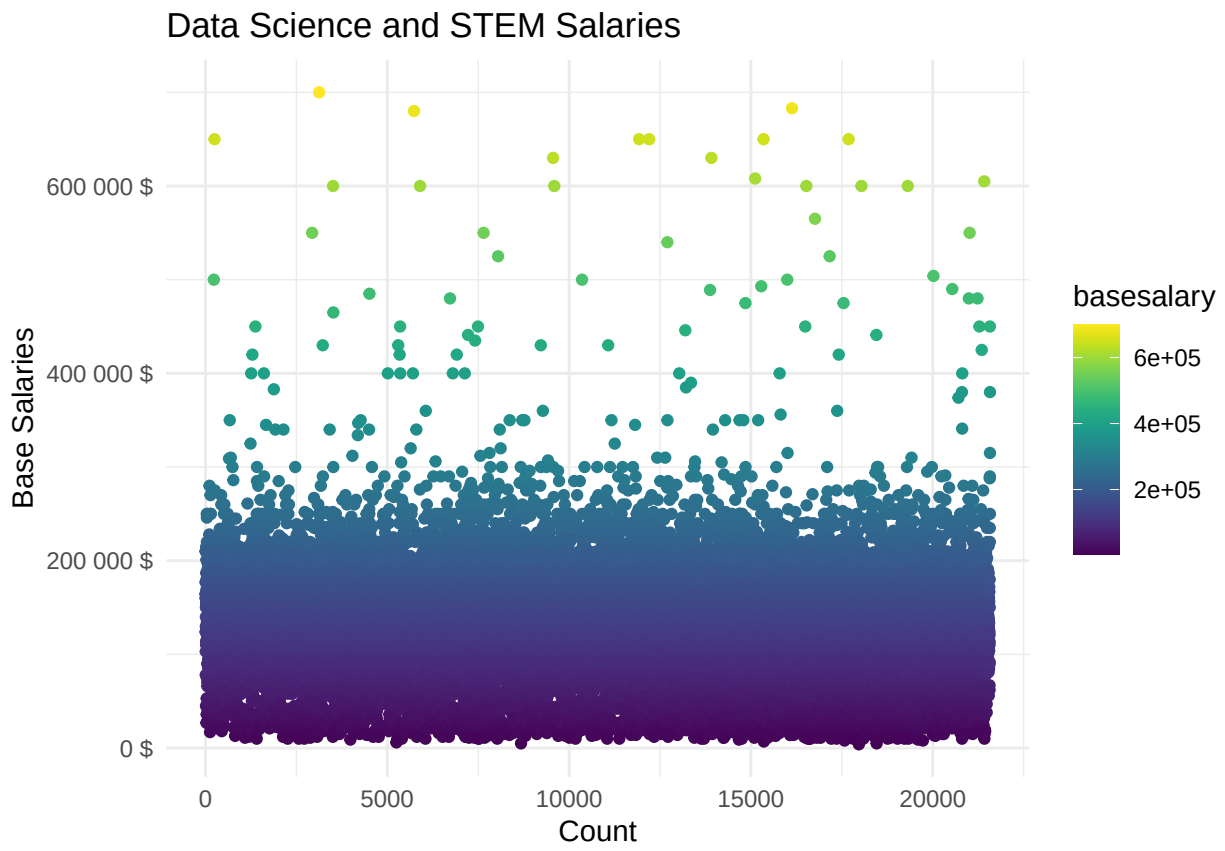
```
## [1] 135000
```

Steps to Data Exploration (graphs)

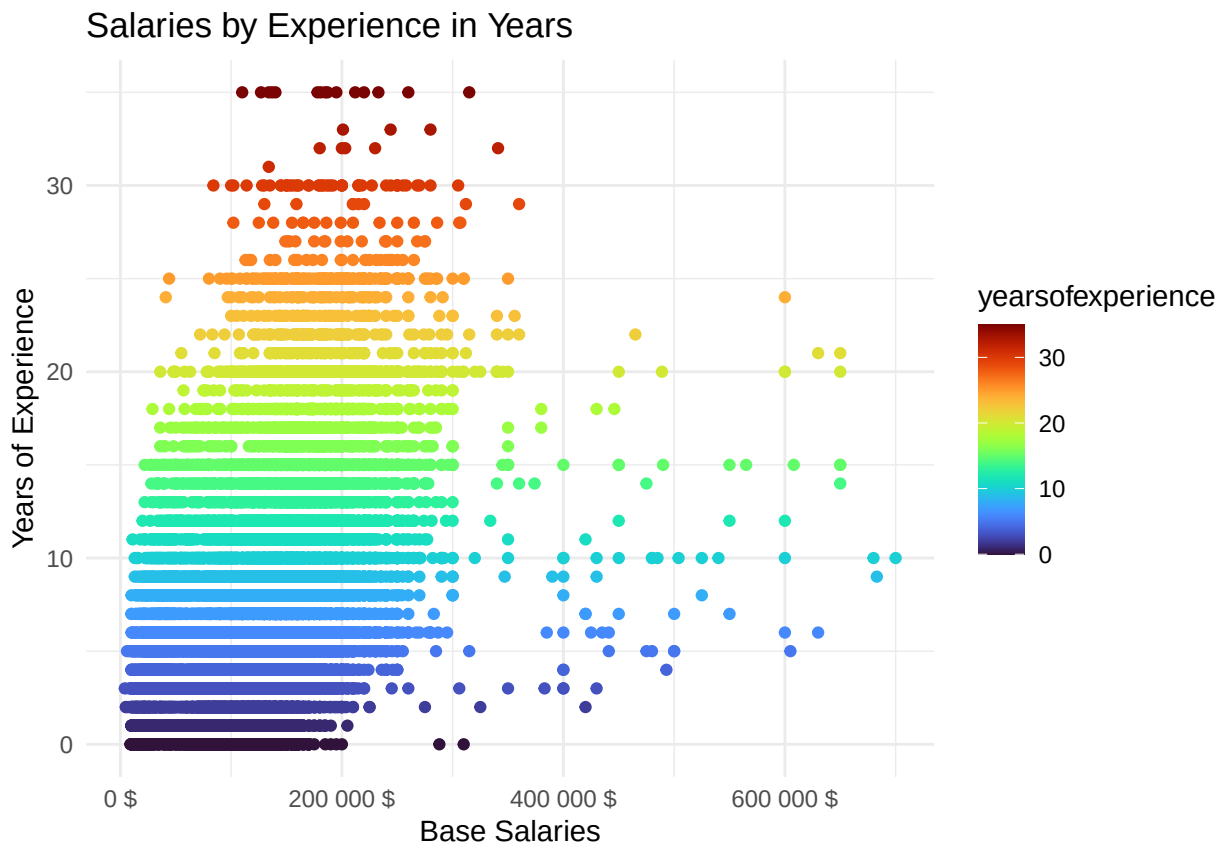
- 1°: graph to analyze base salaries and it range.
- 2°: graph analyzing base salary with years of experience.
- 3°: graph analyzing base salary with race.

Analyzing the Raw data Graphs. * 1°: graph is an analyze to check 'base salary' after updates on the raw data. * 2°: graph is an analyze on 'base salary' with the predictor 'years of experience'. * 3°: graph is an analyze on 'base salary' with predictor 'race'.

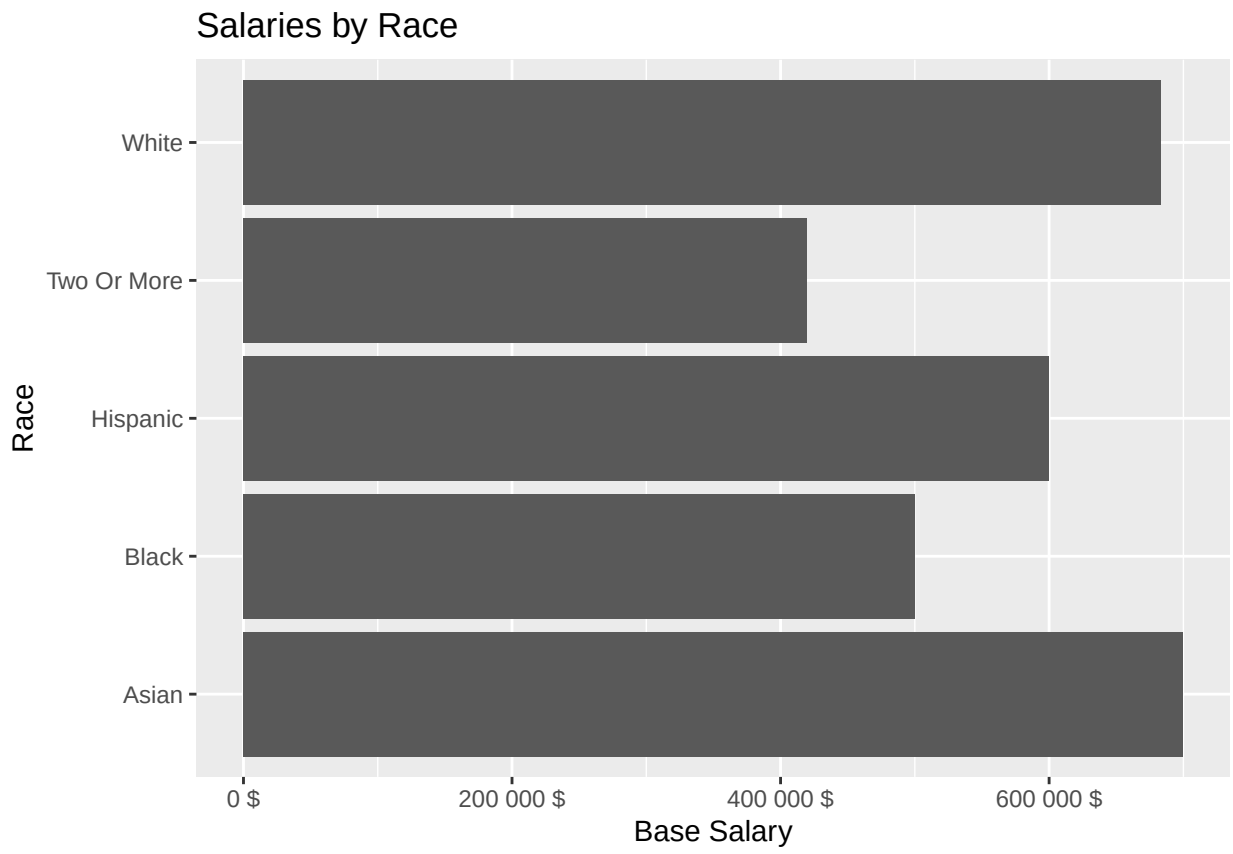
```
ggplot(DataSc_Sal, aes(x= 1:nrow(DataSc_Sal), y= basesalary)) + theme_minimal() +
  geom_point(aes(color = basesalary)) + scale_color_viridis(option = "D") +
  scale_y_continuous(labels = unit_format(scale = 10e-1, unit = "$")) +
  labs(title = "Data Science and STEM Salaries", x= "Count", y= "Base Salaries")
```




```
par(mfrow= c(1, 2))
ggplot(DataSc_Sal, aes(basesalary, yearsofexperience)) + theme_minimal()  +
  geom_point(aes(color = yearsofexperience)) + scale_color_viridis(option = "H") +
  scale_x_continuous(labels = unit_format(scale = 10e-1, unit = "$")) +
  labs(title = "Salaries by Experience in Years", x= "Base Salaries", y= "Years of Experience")
```




```
ggplot(data=DataSc_Sal, aes(x=basesalary, y=Race)) +
geom_bar(stat="identity", position=position_dodge()) +
  scale_x_continuous(labels = unit_format(scale = 10e-1, unit = "$")) +
  labs(title = "Salaries by Race", x= "Base Salary", y= "Race")
```



Steps for Linear Regression Model

- 1°: dividing the data into 80% train and 20% test.
- 2°: make a linear model with 5 predictors.
- 3°: plot and check residuals.
- 4°: make predictions about the linear model (2nd step).
- 5°: print and check what the algorithm learned.

Interpreting Summary:

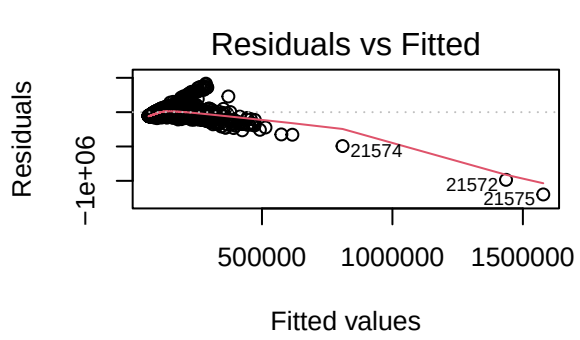
- Residual in this case is not strong symmetrical since it's far away from 0.
- Estimate: expected value is 67080, for example if we consider years of compensation would take an average 0.3 years of compensation to a 67080 salary.
- Standard Error: we can use the standard to error to calculate the average that coefficient estimates. For example; require 0.0021 years of compensation for a estimate salary 67080
- T-value: estimate how far standard deviation is from 0. In this example we are taking in consideration years of compensation, we can say that standard deviation is far of 140 from 0.
- P_vale (Pr(>|t|)): note that most the predictors used have a very small p_value close to zero, also the intercept is very small so we can reject the null hypothesis, in conclusion there are good relationship between target base salary and the predictors selected.
- Residual Standard Error: it is very hard to interpret as we learned in class.
- R-squared: measure how well the model we made will fit on the raw data. The best case is r_square = 1, in this example we have 0.6 that is a good number but not perfect.

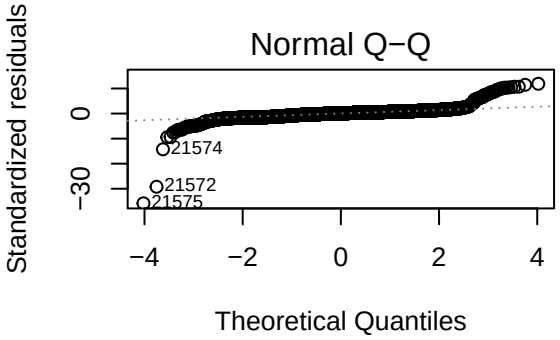
- F-statistic: indicates is there are relationship between the target and predictors selected. The higher the value is from 1 better relationship we have.This model generates a F-statistic of 3120. So we do have a good relationship between the target and the predictors picked in this example.

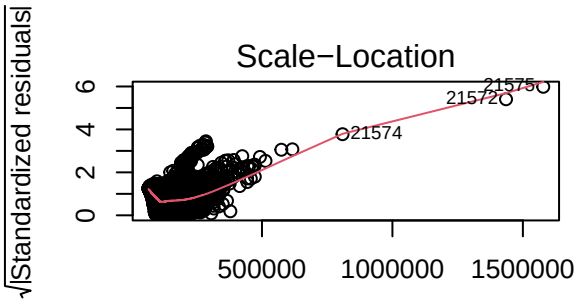
```
# divide data into train and test
set.seed(1234)
i <- sample(1:nrow(DataSc_Sal), nrow(DataSc_Sal)*0.8, replace=FALSE)
train <- DataSc_Sal[i,]
test <- DataSc_Sal[-i,]

# first model with 5 predictors
lr_start_time <- Sys.time()
lm1 <- lm(basesalary~totalyearlycompensation+yearsofexperience+yearsatcompany+gender+Race, data= train)
lr_end_time <- Sys.time()

summary(lm1)
```

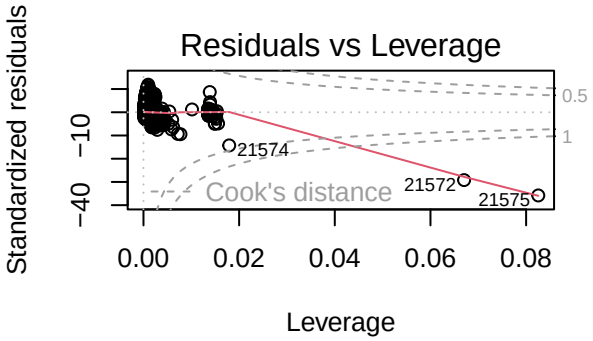
```
##
## Call:
## lm(formula = basesalary ~ totalyearlycompensation + yearsofexperience +
##     yearsatcompany + gender + Race, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1198196   -14860     3435    16812   414730
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)               6.708e+04  7.794e+02  86.068  < 2e-16 ***
## totalyearlycompensation   2.997e-01  2.140e-03 140.031  < 2e-16 ***
## yearsofexperience         1.388e+03  5.861e+01  23.686  < 2e-16 ***
## yearsatcompany           -2.161e+02  9.314e+01  -2.320 0.020346 *
## genderMale               -5.061e+03  6.962e+02  -7.269 3.78e-13 ***
## genderOther              -3.914e+03  4.086e+03  -0.958 0.338084
## RaceBlack                 5.818e+03  1.570e+03   3.707 0.000211 ***
## RaceHispanic              4.378e+03  1.246e+03   3.513 0.000444 ***
## RaceTwo Or More           4.609e+03  1.455e+03   3.167 0.001542 **
## RaceWhite                 4.496e+03  5.811e+02   7.737 1.08e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34930 on 17250 degrees of freedom
## Multiple R-squared:  0.6195, Adjusted R-squared:  0.6193
## F-statistic:  3120 on 9 and 17250 DF,  p-value: < 2.2e-16
```

```
#plotting the residuals
par(mfrow=c(2, 2))
plot(lm1)
```










```
#making some prediction on test
lm1_pred <- predict(lm1, newdata = test)
lm1_cor <- cor(lm1_pred, test$basesalary)
lm1_mse <- mean((lm1_pred - test$basesalary)^2)

# printing some linear model results
print(paste("Linear Regres. - Cor: ", lm1_cor))
```

```
## [1] "Linear Regres. - Cor:  0.83327667069042"
```

```
print(paste("Linear Regres. - Mse: ", lm1_mse))
```

```
## [1] "Linear Regres. - Mse:  933010585.366937"
```

```
print(paste("Linear Regres. - Time: ", lr_end_time - lr_start_time ))
```

```
## [1] "Linear Regres. - Time:  0.0110609531402588"
```

KNN Algorithm

Steps for KNN

- 1°: convert predictors columns as integer

- 2°: make a prediction using KNN regression algorithm
- 3°: print results for analysis

Comments: Running this first model for KNN, I could assume by analyzing the correlation/ mse between Linear Regression and KNN, I would say that KNN is better by a little improvement with higher correlation and lower mse. But with a few more models and predictions and one more algorithm we'll be able to determined if KNN it's the best model.

```
# converting predictors to integer
train$yearsofexperience <- as.integer(train$yearsofexperience)
test$yearsofexperience <- as.integer(test$yearsofexperience)

train$yearsatcompany <- as.integer(train$yearsatcompany)
test$yearsatcompany <- as.integer(test$yearsatcompany)

train$gender <- as.integer(train$gender)
test$gender <- as.integer(test$gender)

train$Race <- as.integer(train$Race)
test$Race <- as.integer(test$Race)

# make the prediction
knn3_start_time <- Sys.time()
fit <- knnreg(train[, c(4, 6, 7, 12, 23)], train[, 9], k= 3)
knn3_end_time <- Sys.time()
predic <- predict(fit, test[, c(4, 6, 7, 12, 23)])

# calculating correlation and mse
cor_knn <- cor(predic, test$basesalary)
mse_knn <- mean((predic - test$basesalary)^2)

print(paste("KNN k=3 - Cor: ", cor_knn))
```

```
## [1] "KNN k=3 - Cor:  0.871631266124787"
```

```
print(paste("KNN k=3 - Mse: ", mse_knn))
```

```
## [1] "KNN k=3 - Mse:  736143643.179047"
```

```
print(paste("KNN K=3 - Time: ", knn3_end_time - knn3_start_time ))
```

```
## [1] "KNN K=3 - Time:  0.00114798545837402"
```

Steps to Scale KNN

- 1°: set up a train scale
- 2°: calculate mean and standard deviation on train data
- 3°: apply scale on train and test data
- 4°: make the prediction
- 5°: calculate correlation and mse

Comments: Scaling the KNN improved the KNN results a little bit more the the previous, but still not a considerable improvement so let's try check KNN with different values for K.

```
# scaling train data
train_scale <- train[, c(4, 6, 7, 12, 23)]

# calculate mean and standard deviation
means <- sapply(train_scale, mean)
stdvs <- sapply(train_scale, sd)

# apply the scale on train and test data
train_scale <- scale(train_scale, center= means, scale= stdvs)
test_scale <- scale(test[, c(4, 6, 7, 12, 23)], center= means, scale= stdvs)

# make prediction on the train scaled
knns_start_time <- Sys.time()
fit <- knnreg(train_scale, train$basesalary, k= 3)
knns_end_time <- Sys.time()
predic <- predict(fit, test_scale)

# calculate correlation and mse
cor_knn2 <- cor(predic, test$basesalary)
mse_knn2 <- mean((predic - test$basesalary)^2)

print(paste("KNN Scaled k=3 - Cor: ", cor_knn2))
```

```
## [1] "KNN Scaled k=3 - Cor:  0.873669774246138"
```

```
print(paste("KNN Scaled k=3 - Mse: ", mse_knn2))
```

```
## [1] "KNN Scaled k=3 - Mse:  706303591.809512"
```

```
print(paste("KNN Scaled     - Time: ", knns_end_time - knns_start_time ))
```

```
## [1] "KNN Scaled     - Time:  0.00114202499389648"
```

Steps to Find the best K

- 1°: make loop to check different values for K.
- 2°: graph the arrays of cor_k and mse_k.
- 3°: find out the min and max K correlation and Mse.

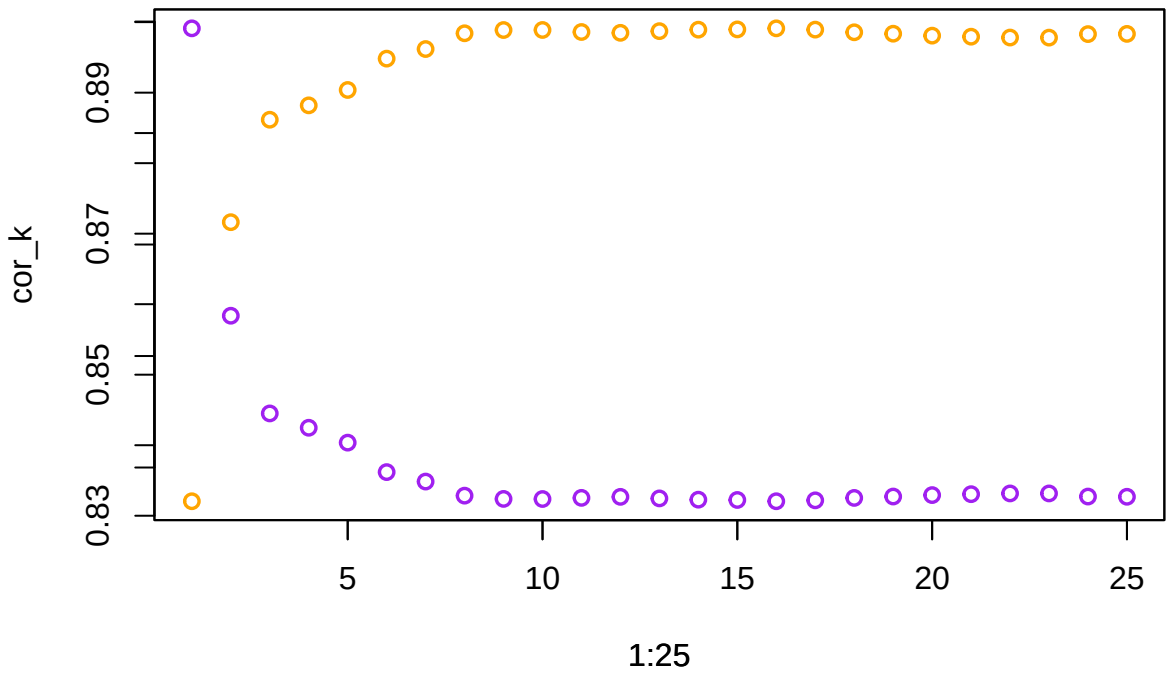
```
# setting the variables and index
cor_k <- rep(0, 20)
mse_k <- rep(0, 20)
i <- 1

# looping through the data
for (k in seq(1, 50, 2)) {
  # predicting with different values for k on each loop
```

```
  fit_k <- knnreg(train[, c(4, 6, 7, 12, 23)], train[, 9], k= k)
  pred_k <- predict(fit_k, test[, c(4, 6, 7, 12, 23)])

  # store the result for correlation and mean on array
  cor_k[i] <- cor(pred_k, test$basesalary)
  mse_k[i] <- mean((pred_k - test$basesalary)^2)

  # print each result
  print(paste("K= ", k, "Correlation: ", cor_k[i], " Mse: ", mse_k[i]))
  i <- i+1
}
```

```
## [1] "K=  1 Correlation:  0.832054011928972  Mse:  993917466.398736"
## [1] "K=  3 Correlation:  0.871631266124787  Mse:  736143643.179047"
## [1] "K=  5 Correlation:  0.886160601936087  Mse:  648501371.215949"
## [1] "K=  7 Correlation:  0.888196997770403  Mse:  635652324.915983"
## [1] "K=  9 Correlation:  0.890378978176373  Mse:  622312386.114436"
## [1] "K=  11 Correlation:  0.894825321391697  Mse:  595895452.659922"
## [1] "K=  13 Correlation:  0.896186597373795  Mse:  587401180.680076"
## [1] "K=  15 Correlation:  0.898421197472725  Mse:  574757885.638329"
## [1] "K=  17 Correlation:  0.898880496683814  Mse:  571837457.113851"
## [1] "K=  19 Correlation:  0.898887961036769  Mse:  571761099.139348"
## [1] "K=  21 Correlation:  0.898603289068096  Mse:  572785011.666603"
## [1] "K=  23 Correlation:  0.898484327414787  Mse:  573682367.500894"
## [1] "K=  25 Correlation:  0.898724886003756  Mse:  572293551.009599"
## [1] "K=  27 Correlation:  0.898932463855781  Mse:  571146965.017536"
## [1] "K=  29 Correlation:  0.898968705748743  Mse:  570888942.935234"
## [1] "K=  31 Correlation:  0.899119013130159  Mse:  569748064.645462"
## [1] "K=  33 Correlation:  0.898937671536357  Mse:  570542549.222346"
## [1] "K=  35 Correlation:  0.898559687829861  Mse:  572719250.526102"
## [1] "K=  37 Correlation:  0.898358704096807  Mse:  573969272.81481"
## [1] "K=  39 Correlation:  0.898085415969983  Mse:  575265042.031774"
## [1] "K=  41 Correlation:  0.897935775921349  Mse:  576070764.87806"
## [1] "K=  43 Correlation:  0.897814289701695  Mse:  576755363.300801"
## [1] "K=  45 Correlation:  0.897805784418473  Mse:  576740010.472035"
## [1] "K=  47 Correlation:  0.898304209502961  Mse:  573953493.198788"
## [1] "K=  49 Correlation:  0.898334954083258  Mse:  573767776.95552"
```

```
# graphing mse and cor
plot(1:25, cor_k, lwd= 1.7, col= 'orange')
par(new=TRUE)
plot(1:25, mse_k, lwd= 1.7, col= 'purple', labels= FALSE, ylab="")
```

```
## Warning in plot.window(...): "labels" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "labels" is not a graphical parameter

## Warning in box(...): "labels" is not a graphical parameter

## Warning in title(...): "labels" is not a graphical parameter
```




```
# find the min and max k cor/ mse
paste("Max K cor: ", which.max(cor_k))
```

```
## [1] "Max K cor:  16"
```

```
paste("Min K mse: ", which.min(mse_k))
```

```
## [1] "Min K mse:  16"
```

Steps to check K= 15

- 1°: lets compare K= 15

Comments: Setting k= 15 we get the best result between Logistic Regression and KNN previous implemented but still not relevant result.

```
# make new prediction with k= 15
knn15_start_time <- Sys.time()
fit <- knnreg(train[, c(4, 6, 7, 12, 23)], train[, 9], k= 15)
knn15_end_time <- Sys.time()
predic <- predict(fit, test[, c(4, 6, 7, 12, 23)])
```

```r
# calculating correlation and mse
cor_knn15 <- cor(predic, test$basesalary)
mse_knn15 <- mean((predic - test$basesalary)^2)

print(paste("KNN k=15 - Cor: ", cor_knn15))
```

```
## [1] "KNN k=15 - Cor:  0.898421197472725"
```

```r
print(paste("KNN k=15 - Mse: ", mse_knn15))
```

```
## [1] "KNN k=15 - Mse:  574757885.638329"
```

```r
print(paste("KNN k=15 - Time: ", knn15_end_time - knn15_start_time ))
```

```
## [1] "KNN k=15 - Time:  0.00122189521789551"
```

Decision Tree Algorithm

Steps to do Decision Tree

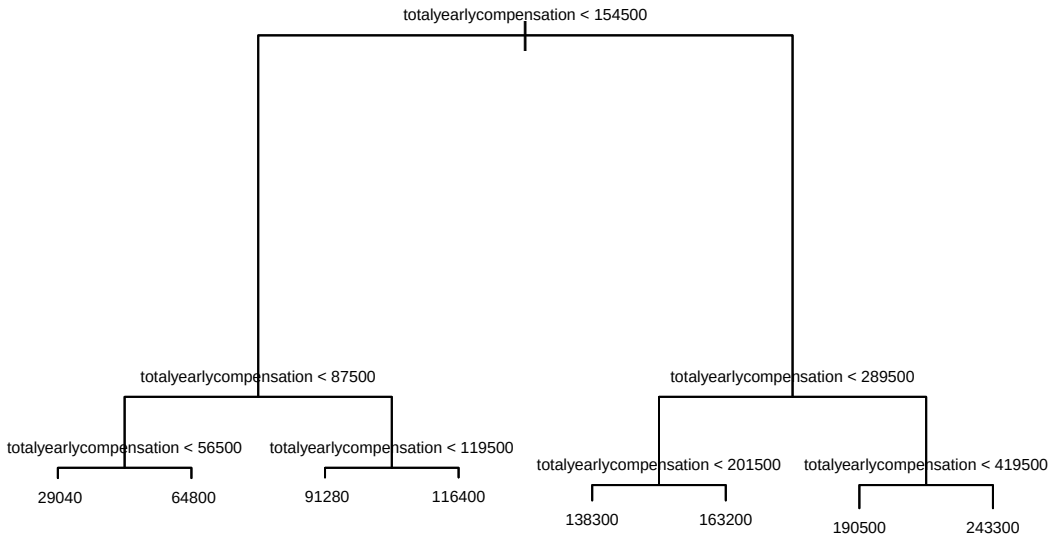
- 1°: make a prediction using 5 predictors
- 2°: plot the prediction to analyze the graph
- 3°: make a prediction and calculate correlation and mse

Comments: The graph of Decision Tree defined that the 'total years of compensation' is the best predictor for the target used, also defined that the best fit for the model is 'total years of compensation' less than 154500 for that reason it doesn't displayed the other predictors used.

```r
# making the prediction with the target and predictors
dt_start_time <- Sys.time()
tre <- tree(basesalary ~ totalyearlycompensation + yearsofexperience + yearsatcompany + gender + Race, 
dt_end_time <- Sys.time()
summary(tre)
```

```
##
## Regression tree:
## tree(formula = basesalary ~ totalyearlycompensation + yearsofexperience +
##     yearsatcompany + gender + Race, data = train)
## Variables actually used in tree construction:
## [1] "totalyearlycompensation"
## Number of terminal nodes:  8
## Residual mean deviance:  727500000 = 1.255e+13 / 17250
## Distribution of residuals:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -123300.0  -13040.0    -437.5       0.0   11660.0  456700.0
```

```r
# plotting the prediction
plot(tre)
text(tre, cex= 0.5, pretty= 0)
```




```r
# make prediction and find correlation and mse
tre_pred <- predict(tre, newdata =test)
tre_cor <- cor(tre_pred, test$basesalary)
tre_mse <- sqrt(mean((tre_pred - test$basesalary)^2))

print(paste("Dec. Tree - Cor: ", tre_cor))
```

```
## [1] "Dec. Tree - Cor:  0.8838284855868"
```

```r
print(paste("Dec. Tree - Mse: ", tre_mse))
```

```
## [1] "Dec. Tree - Mse:  25512.356873744"
```

```r
print(paste("Dec. Tree - Time: ", dt_end_time - dt_start_time ))
```

```
## [1] "Dec. Tree - Time:  0.0356040000915527"
```

Final conclusion and analyse.

-Linear Regression:

* "Cor: 0.83327667069042"
* "Mse: 933010585.366937"
* "Logistic Regres. - Time: 0.0227301120758057"

-KNN for K=3

* "KNN k=3 - Cor: 0.871631266124787"
* "KNN k=3 - Mse: 736143643.179047"
* "KNN K=3 - Time: 0.00343608856201172"

-Scaled KNN for K=3

* "KNN Scaled k=3 - Cor: 0.873669774246138"
* "KNN Scaled k=3 - Mse: 706303591.809512"
* "KNN Scaled   - Time: 0.00418591499328613"

-KNN for K=15

* "KNN k=15 - Cor: 0.898421197472725"
* "KNN k=15 - Mse: 574757885.638329"
* "KNN k=15 - Time: 0.00408005714416504"

-Decision Tree

* "Dec. Tree - Cor: 0.8838284855868"
* "Dec. Tree - Mse: 25512.356873744"
* "Dec. Tree - Time: 0.0441329479217529"

Since the best algorithm should give the higher correlation and lower mse it seems that KNN for K=15 and Decision Tree are the best they vary between one having the higher correlation and the other the lower mse, even though the difference of the correlation being 0.01. Also I set a times on each model to check if time could be something to take into a count, but also run time does not seems to be a good parameter. In conclusion, since neither of the technologies implemented had a significant difference in (correlation, mse, or time) we can conclude that all are the same for the prediction made in this example.