4) Salary\_hike -> Build a prediction model for Salary\_hike

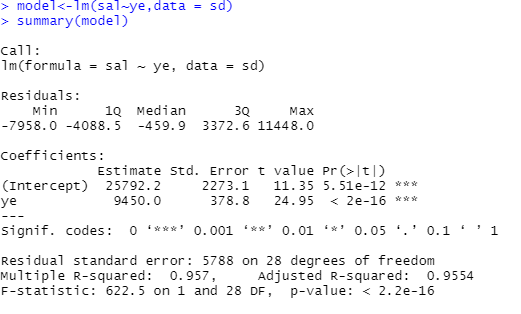
BUSSINESS PROBLEM: PREDICT SALARY HIKE USING YEARS OF EXPERIENCE

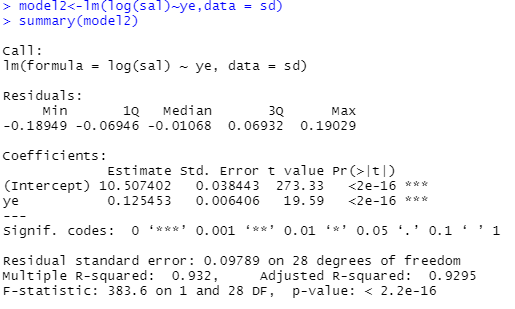
UNIVARIET ANALYSIS

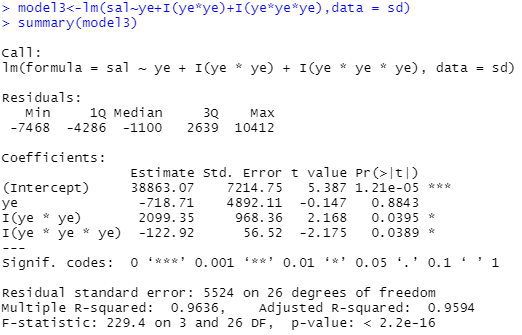
|  |  |
| --- | --- |
| Years of exerience  BAR PLOT:    multimodal plot  HISTOGRAM:    Data is right skewed  BOX-PLOT:    Data is right skewed with no outliers  More data is concentrated in first half and more spread in second half  QQPLOT    Data is normally distributed ,as more data points lies in qqline | Salary-hike    multimodal plot    Data is    Data is right skewed with no outliers  More data is concentrated in first half and more spread in second half    Data is not normally distributed and positively skewed |

BIVARIENT ANLYSIS

|  |
| --- |
| Correlation coefficient  cor(x,y)=0.97  SCATTER PLOT    SCATTER PLOT SHOWS HETROSCADASITICITY PROBLEM |







|  |  |  |  |
| --- | --- | --- | --- |
| MODEL | COR-COEF | R2 | ERROR |
| Model1 | 0.97 | 0.95 | 5592.04 |
| Model2 | 0.95 | 0.93 | 7213 |
| Model3 | 0.98 | 0.96 | 5142 |

Model 3 is the good fit model

PYTHON CODE

import pandas as pd # deals with data frame

import numpy as np # deals with numerical values

sd = pd.read\_csv("C:/Users/USER/Desktop/sd.csv")

sd.columns="x","y"

###x=years of experience and y=salary

import matplotlib.pylab as plt #for different types of plots

plt.scatter(x=sd['x'], y=sd['y'],color='green')# Scatter plot

np.corrcoef(sd.x, sd.y) #correlation

import statsmodels.formula.api as smf

model = smf.ols('y ~ x', data=sd).fit()

model.summary()

pred1 = model.predict(pd.DataFrame(sd['x']))

pred1

print (model.conf\_int(0.01)) # 99% confidence interval

res = sd.y - pred1

sqres = res\*res

mse = np.mean(sqres)

rmse = np.sqrt(mse)

rmse

###cor-coef-0.97,r2=0.95,err=5592.04

###lets try to improve model

model = smf.ols('y ~ x+I(x\*x)', data=sd).fit()

model.summary()

pred1 = model.predict(pd.DataFrame(sd['x']))

pred1

print (model.conf\_int(0.01))

res = sd.y - pred1

sqres = res\*res

mse = np.mean(sqres)

rmse = np.sqrt(mse)

rmse

###r2=0.95,err=5590.84

###model3######

model = smf.ols('y ~ x+I(x\*x)+I(x\*x\*x)', data=sd).fit()

model.summary()

pred1 = model.predict(pd.DataFrame(sd['x']))

pred1

print (model.conf\_int(0.01))

res = sd.y - pred1

sqres = res\*res

mse = np.mean(sqres)

rmse = np.sqrt(mse)

rmse

np.corrcoef(sd.y,pred1)

###r2=0.96,err=5142,cor-coef=0.98#######

###hence its the best model becoz of improved r2 value and decrease in error