## IMPORTING LIBARIRES

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import confusion\_matrix,accuracy\_score

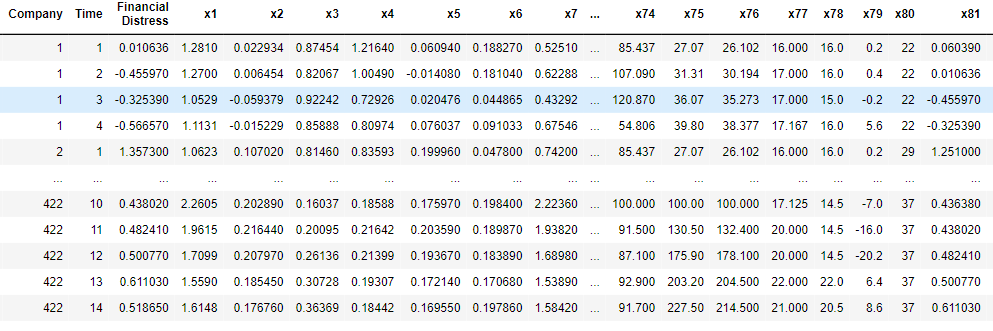
from sklearn.ensemble import RandomForestRegressor

## REDAING DATASET

df=pd.read\_csv('Financial Distress.csv.zip')

df

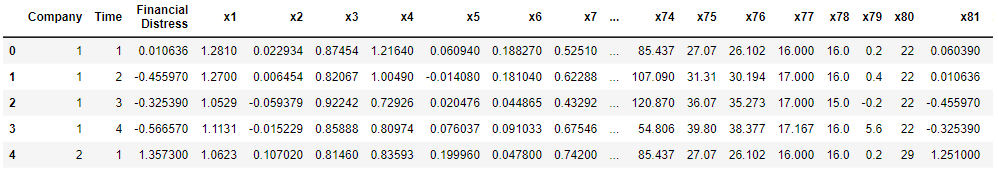
## OUTPUT



## TO CHECK FIRST FIVE COLUMNS

df.head()

## output



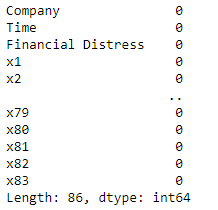
df.shape

## output

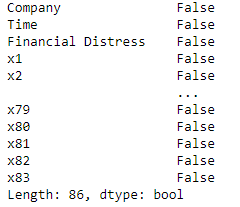


## #CHECKING MISSING VALUE

df.isnull().sum()

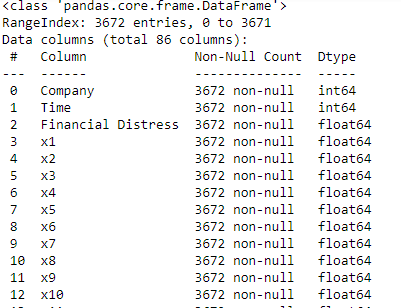


df.isnull().any()

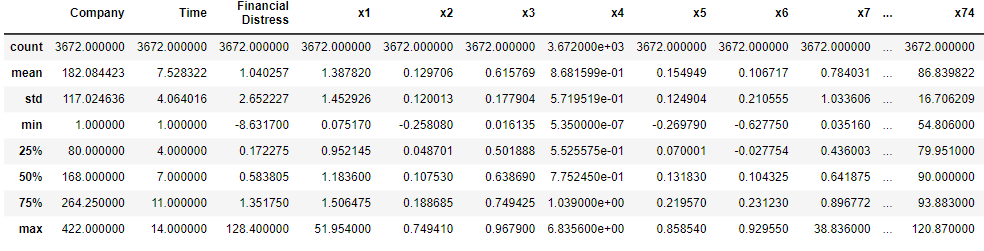


## #INFORMATION ABOUT DATASET

df.info()



df.describe()

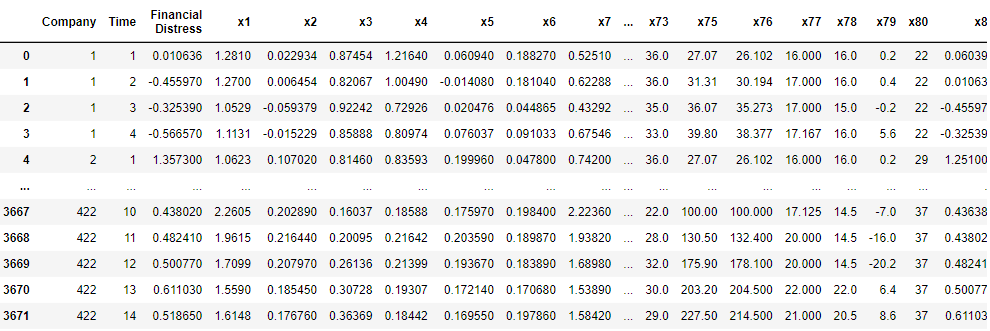


print('lenghth of dataset:',len(df))



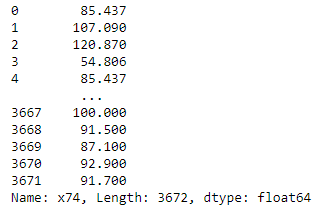
## #SPLITTING THEDATASET

x=df.drop('x74',axis=1)

x 

y=df['x74']

y



from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,random\_state=0)

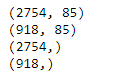
print(x\_train.shape)

print(x\_test.shape)

print(y\_train.shape)

print(y\_test.shape)

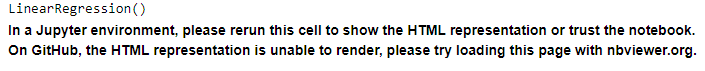
output

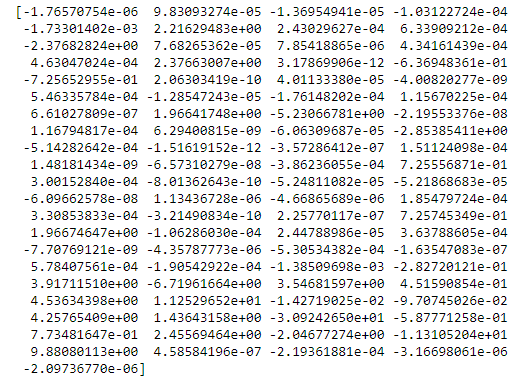


from sklearn.linear\_model import LinearRegression

regressor=LinearRegression()

regressor.fit(x\_train,y\_train)



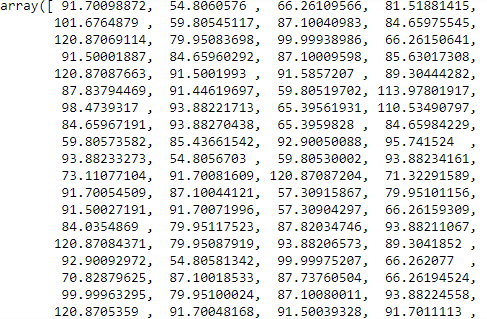
print(regressor.coef\_) 

print(regressor.intercept\_)

-46.268254777815955

predicted=regressor.predict(x\_test)

predicted



print(x\_test)

Company Time Financial Distress x1 x2 x3 x4 \

1386 125 14 0.55637 1.31570 0.069369 0.39010 0.22676

117 11 4 -0.31752 0.50287 -0.041789 0.72860 0.37339

3637 416 6 0.54565 1.36240 0.113770 0.51348 0.22232

2985 307 8 -0.42233 0.49211 -0.042668 0.86992 0.45702

3120 331 2 0.69600 1.34810 0.110400 0.79424 1.32880

... ... ... ... ... ... ... ...

2082 196 1 0.16158 2.32650 0.098068 0.52487 0.76046

2537 242 9 0.60401 1.29240 0.070652 0.34188 0.27065

1651 149 10 0.40885 0.82371 0.082287 0.66878 1.68880

1697 152 14 4.78400 2.05840 0.051552 0.57748 1.01250

229 24 11 2.23120 1.31530 0.079773 0.68381 0.95030

x5 x6 x7 ... x73 x75 x76 x77 x78 \

1386 0.070217 0.12312 0.65850 ... 29.0 227.500 214.500 21.000 20.5

117 -0.051924 -0.31306 0.24850 ... 33.0 39.800 38.377 17.167 16.0

3637 0.069435 0.13207 1.10420 ... 29.0 52.740 49.206 15.500 12.0

2985 0.038475 -0.38395 0.12969 ... 28.0 68.480 60.941 17.333 12.0

3120 0.112230 0.23539 0.74100 ... 36.0 30.250 29.171 16.750 16.0

... ... ... ... ... ... ... ... ... ...

2082 0.078434 0.53871 1.72730 ... 36.5 25.235 24.551 15.500 16.5

2537 0.054996 0.09639 1.11070 ... 27.0 82.300 74.497 16.500 13.0

1651 0.147350 -0.11352 0.21805 ... 22.0 100.000 100.000 17.125 14.5

1697 0.097953 0.46088 1.05350 ... 29.0 227.500 214.500 21.000 20.5

229 0.081231 0.20012 0.38455 ... 28.0 130.500 132.400 20.000 14.5

x79 x80 x81 x82 x83

1386 8.6000 21 2.13180 25 57

117 5.6000 14 -0.31275 13 42

3637 -6.4000 8 0.30759 4 15

2985 -8.5333 19 -0.43115 13 44

3120 0.3500 27 0.85633 11 31

... ... ... ... ... ...

2082 2.9000 21 0.27483 11 37

2537 0.6000 18 0.39899 7 16

1651 -7.0000 18 0.81935 20 44

1697 8.6000 14 3.35090 19 34

229 -16.0000 22 1.29320 18 45

predicted.shape

(918,)

dframe=pd.DataFrame(y\_test,predicted)

dframe

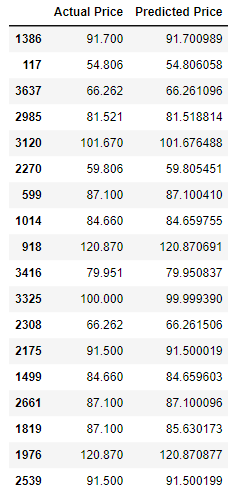
| **x74** |
| --- |
| **91.700989** | NaN |
| **54.806058** | NaN |
| **66.261096** | NaN |
| **81.518814** | NaN |
| **101.676488** | NaN |
| **...** | ... |
| **78.786461** | NaN |
| **93.882140** | NaN |
| **100.000214** | NaN |
| **91.700595** | NaN |
| **91.500303** | NaN |
|  |  |
|  |  |

dfr=pd.DataFrame({'Actual Price':y\_test,'Predicted Price':predicted})

dfr

|  | **Actual Price** | **Predicted Price** |
| --- | --- | --- |
| **1386** | 91.700 | 91.700989 |
| **117** | 54.806 | 54.806058 |
| **3637** | 66.262 | 66.261096 |
| **2985** | 81.521 | 81.518814 |
| **3120** | 101.670 | 101.676488 |
| **...** | ... | ... |
| **2082** | 78.786 | 78.786461 |
| **2537** | 93.883 | 93.882140 |
| **1651** | 100.000 | 100.000214 |
| **1697** | 91.700 | 91.700595 |
| **229** | 91.500 | 91.500303 |

dfr.head(60)



from sklearn.metrics import confusion\_matrix,accuracy\_scorein\_accuracy

train\_accuracy=regressor.score(x\_train,y\_train)

print('train\_accuracy',train\_accuracy)

test\_accuracy=regressor.score(x\_test,y\_test)

print('test\_accuracy:',test\_accuracy)



import math

from sklearn import metrics

print('Mean Absolute Error:',metrics.mean\_absolute\_error(y\_test,predicted))

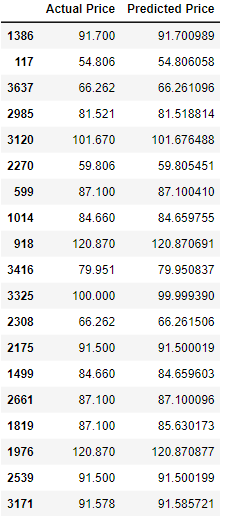
print('Mean Squared Error:',metrics.mean\_squared\_error(y\_test,predicted))

print('Root Mean Squared Error:',math.sqrt(metrics.mean\_squared\_error(y\_test,predicted)))



graph=dfr.head(20)

graph



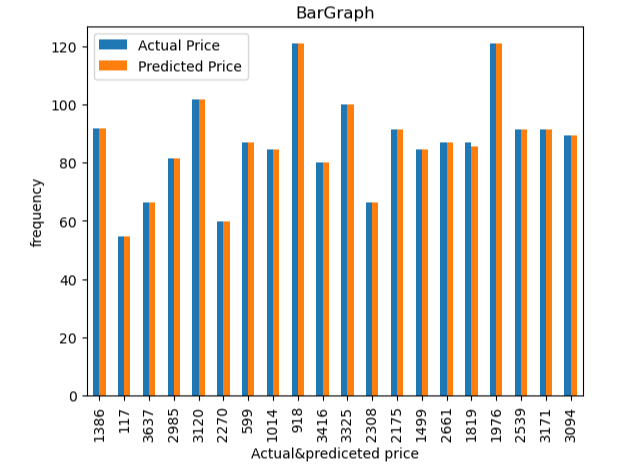
graph.plot(kind='bar')

plt.title('BarGraph')

plt.xlabel('Actual&prediceted price')

plt.ylabel('frequency')

plt.show()



from sklearn.metrics import confusion\_matrix,accuracy\_scorein\_accuracy

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.90,random\_state=100)

print(x\_train.shape)

print(x\_test.shape)

print(y\_train.shape)

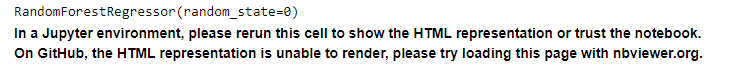
print(y\_test.shape)



from sklearn.ensemble import RandomForestRegressor

regressor = RandomForestRegressor(n\_estimators=100,random\_state=0)

regressor.fit(x\_train,y\_train)



predicted=regressor.predict(x\_test)

predicted



from sklearn.metrics import confusion\_matrix,accuracy\_score

train\_accuracy=regressor.score(x\_train,y\_train)

print('train\_accuracy:',train\_accuracy)

R\_test\_accuracy=regressor.score(x\_test,y\_test)

print('test\_accuracy:',R\_test\_accuracy)



import matplotlib.pyplot as plt

linear\_regression\_accuracy =0.9999999825628604

random\_forest\_accuracy =0.992675587067571

accuracy\_scores = [linear\_regression\_accuracy, random\_forest\_accuracy]

model\_names = ['Linear Regression', 'Random Forest Regression']

plt.bar(model\_names, accuracy\_scores)

plt.xlabel('Regression Models')

plt.ylabel('Test Accuracy')

plt.title('Comparison of Test Accuracy: Linear Regression vs Random Forest Regression')

plt.show()

