Project 3

**Perdition of Student Marks with Linear Regression**

**OVERVIEW**

A project to understand and implement the concepts of Linear Regression that will outline how the regression concept works . the prediction will be determined on the number of hours a student will study and the scores he will receive accordingly.

**Software Requirements**

1. Programming Language : Python

2. Environemnt: Jupyter Notebooks / Google Collab

3. Database: CSV(export type)

4. Operation System: Windows XP or above

5. Librarires Used: Pandas,Folium, Seaborn, Scikit, SKLEARN

6.Datasets used: Student Dataset

1. **Open a New Notebook and import the required libraires and read the csv file**

|  |  |
| --- | --- |
|  | **import** **numpy** **as** **np**  **import** **pandas** **as** **pd**  **import** **matplotlib.pyplot** **as** **plt**  **import** **seaborn** **as** **sns**  **import** **scipy.stats** **as** **stats**  **from** **sklearn.model\_selection** **import** train\_test\_split |

Description :

The above part of the program is used in order to import various functions.

“NumPy” is used for mathematical operations.

“pandas” is used for operations on data sets.

“matplotlib.pyplot” is used for plotting the data.

“seaborn” is used in visualizing the data.

“scipy.stats” is used for optimization of the data.

“sklearn.model\_selection” is used for training the model and testing the model.

1. **Importing the Student Dataset**

df = pd.read\_csv('/data.csv')

Description :

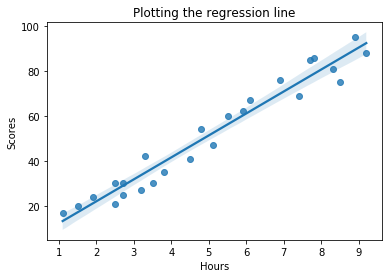
The above line of the code is used to read the data from the location path provided.

1. **Viewing and Exploring the Data**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. print("Now our data is loaded") 2. df   Output:      Description :  The above part of the code is printing the data set.  df.shape  (25, 2)  df.info()  <class 'pandas.core.frame.DataFrame'>  RangeIndex: 25 entries, 0 to 24  Data columns (total 2 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 Hours 25 non-null float64  1 Scores 25 non-null int64  dtypes: float64(1), int64(1)  memory usage: 528.0 bytes  Description :  “df.shape” is used to find No. of. Row’s and No. of. Column’s of data set.  “df.info()” is used to find Column’s, Data type of each column, values filled in Column’s.  df.describe()   |  | **Hours** | **Scores** | | --- | --- | --- | | **count** | 25.000000 | 25.000000 | | **mean** | 5.012000 | 51.480000 | | **std** | 2.525094 | 25.286887 | | **min** | 1.100000 | 17.000000 | | **25%** | 2.700000 | 30.000000 | | **50%** | 4.800000 | 47.000000 | | **75%** | 7.400000 | 75.000000 | | **max** | 9.200000 | 95.000000 |   df.corr()    Description :  “df.describe()” is used to find count, mean, standard deviation, min, 25%, 50%, 75% and max of every column in the data set.  “df.corr()” is used to find the co-relation of every column mapping to all the column.   1. **Visualizing the Linear Relation between Hours & Scores ( Drawing a joint Plot**   sns.jointplot(df['Hours'], df['Scores'], kind = "reg").annotate(stats.pearsonr) plt.show()      Description :  “sns.jointplot(df['Hours'], df['Scores'], kind = "reg").annotate(stats.pearsonr) plt.show()”is used to visualize the linear relation between hours & scores by plotting a joint plot and we got the information that there is no drastic change with respect of hours and also scores. |  |

1. **Visualizing the Correlation**

sns.regplot(x="Hours", y="Scores", data=df) plt.title("Plotting the regression line")



Description :

“sns.regplot(x="Hours", y="Scores", data=df) plt.title("Plotting the regression line")” is used in plotting the regression line, observing the plot gives us the information that most of the values in the data set are near to the line.

## Using Simple linear regression to predict the data as we only have two columns.

Dividing Our Dataset into training and testing

X = df.iloc[:, :-1].values

y = df.iloc[:, -1].values

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.30, random\_state = 0)

**from** **sklearn.linear\_model** **import** LinearRegression

regressor = LinearRegression()

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

Out[27]:

LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=False)

Description :

The above piece of code is having a significant importance of training the data and testing the data which give predictions according to which the machine is trained.

In order to train the data ‘train\_test\_split’ is imported from the function ‘sklearn.model\_selection’.

From the imported function the data will be spilt into 4 parts in which training data will be of (1-test\_size)% of data and remaining data will be in training data.

“**from** **sklearn.linear\_model** **import** LinearRegression

regressor = LinearRegression()

regressor.fit(X\_train, y\_train)” is used to fit the data in linear regression model to train the data.

**After Training now performing Prediction**

y\_pred = regressor.predict(X\_test)

y\_pred

Out[28]:

array([17.05366541, 33.69422878, 74.80620886, 26.8422321 , 60.12335883,

39.56736879, 20.96909209, 78.72163554])

Description :

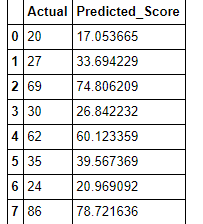
The above part of the code is used to predict the data from the knowledge the machine learnt.

The predicted data is stored under the variable “y\_pred” which predicts the marks.

## Comparing Actual vs Predicted Value

df1 = pd.DataFrame({'Actual': y\_test, 'Predicted\_Score': y\_pred})

df1



Description :

The above part of the code is having the following operation,

“df1 = pd.DataFrame({'Actual': y\_test, 'Predicted\_Score': y\_pred})” is used to create a new data set that is having 2 columns named “Actual” & “Predicted\_Score” which shows the original scores of students and predicted scores.

**Conclusion**

**The conclusion that can be drawn from this project is that using linear regression and training using 70% of the data in the dataset we were able to achieve accuracy of 95.5% and also able to find the coefficients of b0 and b1.**