

**A
SYNOPSIS
of
MINOR PROJECT
on
Student Marks Prediction
(In Machine Learning)**



Submitted by

Kuldeep Nath Chouhan (21EGICS058)

**Project guide
Ms. Payal Sachdev**

**Head of Department
Dr. Mayank Patel**

**Geetanjali Institute of Technical Studies, Dabok, Udaipur (Raj.)
Department of Computer Science and Engineering**

july,2024

INTRODUCTION

INTRODUCTION

In this project, the data collected from an students while they are studying for their exam that is how much hour a student can study based on which the student mark is predicted. This overview of the data collected helps in determining How much hour need to do the study to get 99% mark and if a student study $x()$ hours per day so how much mark he will get through these points and the school or college can determine the performance of the student. Data collected includes student study hours and student mark, which will help the prediction process. There are two columns in the dataset which includes numerical columns. Different machine learning methods are used to analyze the dataset and to predict the output. This is a time saving process for banks. After performing the evaluation with the three different machine learning models, the model with more accuracy is considered for further prediction.

OBJECTIVES

- Machine learning based data mining techniques are used to automate process of student performance prediction using linear regression technique. Education is very important issue regarding development of a country. The main objective of educational institutions is to provide high quality education to its students. One way to accomplish this is by predicting student's academic performance and thereby taking early steps to improve student's performance and teaching quality. Students marks of other subjects are taken as input for evaluation students'

performance. Data set is pre-processed and features and labels are extracted from dataset then dataset is split in to test and train sets then linear regression is applied to dataset for prediction.

OVERVIEW OF THE PROJECT

The project provides a quick and easy way to predict student mark based on the how many hour they study. It makes the selection process easier. A set of training data is given to the machine learning model, on the basis of this data set is trained. All new student details collected act as predetermined test data. After performance of the test, the model guesses how much mark is obtained by the student based on their studying hour it concludes on the basis of training data. Three different machine learning models are used to train the data. The model with the higher accuracy value is used for evaluating the test dataset. Thus it concludes predicting whether the student can get above 99 percentage mark.

CHAPTERWISE SUMMARY

The first chapter is an introductory chapter, which gives an overview of the project. It includes four divisions - introduction, objectives, overview and chapterwise summary. The second chapter is data analysis, where the dataset is analyzed and studied for further classifications. Third chapter deals with the different machine learning models used. The three different models used are Decision Tree Classifier, Random Forest Classifier and Logistic Regression. The last chapter gives an elaborate idea about the results of different models. Let's get to know more about the dataset in the upcoming chapter.

DATA ANALYSIS

❖ STRUCTURE OF DATASET

There are two different columns including the target variable. These include
Numerical data

Numerical_columns=[`study_hours`, `student_marks`]

```
[7] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 200 entries, 0 to 199  
Data columns (total 2 columns):  
#   Column          Non-Null Count  Dtype  
---  ---  
0   study_hours     195 non-null   float64  
1   student_marks   200 non-null   float64  
dtypes: float64(2)  
memory usage: 3.2 KB
```

Loading the dataset

In the second step we are loading the dataset. I have taken a csv file and we are loading it with the help of pandas which helps us to read the csv files, then I have given the path path of the csv file and assigned it a variable called df.

▼ Load Dataset



```
path = r"/content/student_info.csv"  
df = pd.read_csv(path)
```

If we want to see the tail or head of our dataset we can give commands like `df.head` and `df.tail` and `df.shape` , from this we can get to know how our dataset looks like.



```
df.head()
```



	study_hours	student_marks
--	-------------	---------------

0	6.83	78.50
---	------	-------

1	6.56	76.74
---	------	-------

2	NaN	78.68
---	-----	-------

3	5.67	71.82
---	------	-------

4	8.67	84.19
---	------	-------

```
[5] df.tail()
```

	study_hours	student_marks
195	7.53	81.67
196	8.56	84.68
197	8.94	86.75
198	6.60	78.05
199	8.35	83.50

```
df.shape
```

```
(200, 2)
```

❖ DATA VISUALIZATION

In the third step we want to visualize our data. We use seaborn and matplotlib.pyplot libraries to visualize and thereby analyze the dataset. Here are the visualizations of the data in the dataset.

▼ Discover and visualize the data to gain insights

```
[7] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 200 entries, 0 to 199  
Data columns (total 2 columns):  
#   Column          Non-Null Count  Dtype  
---  ---  
0   study_hours     195 non-null    float64  
1   student_marks   200 non-null    float64  
dtypes: float64(2)  
memory usage: 3.2 KB
```

In the above diagram we are able to get the details like we have two columns which are student hour and student mark, we are also able to get the info that we have 2 columns and the student hour and student mark belongs to the float datatype. The memory usage is 3.2kb

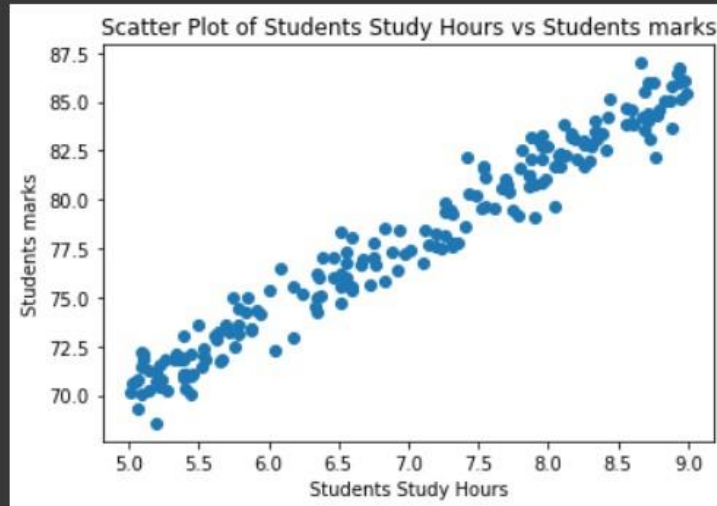
```
[8] df.describe()
```

	study_hours	student_marks
count	195.000000	200.000000
mean	6.995949	77.93375
std	1.253060	4.92570
min	5.010000	68.57000
25%	5.775000	73.38500
50%	7.120000	77.71000
75%	8.085000	82.32000
max	8.990000	86.99000

In this diagram we are able to find parameters like count ,mean, standard deviation , maximum and minimum values of the dataset. Along with that we are also given with some percentage information about the dataset.



```
plt.scatter(x =df.study_hours, y = df.student_marks)
plt.xlabel("Students Study Hours")
plt.ylabel("Students marks")
plt.title("Scatter Plot of Students Study Hours vs Students marks")
plt.show()
```

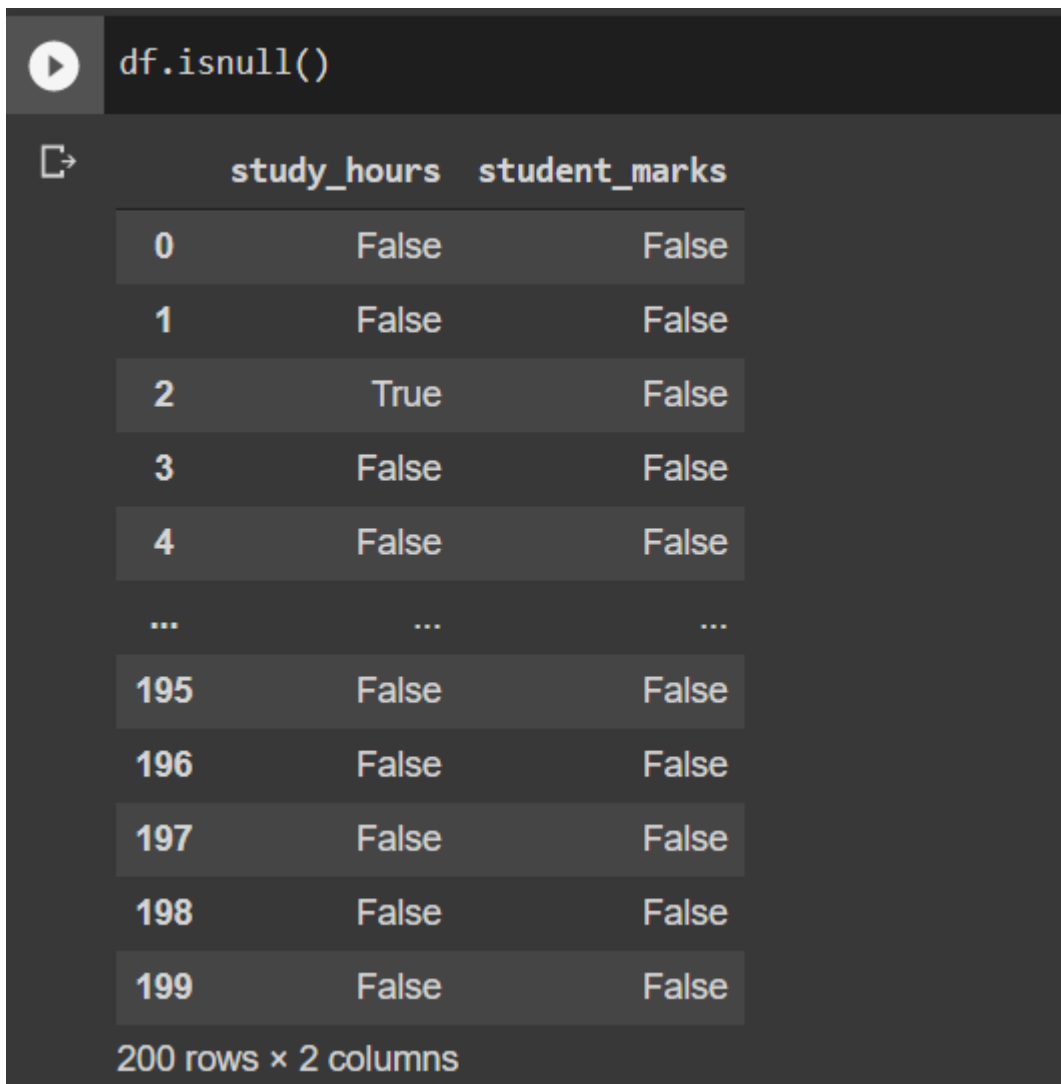


In the above diagram we are using scatterplot for the visualizing the data .We are using scatterplot by matplotlib library. Scatterplot needs a x axis and y axis. We are giving the x axis and y axis name as plt.xlabel and plt.ylabel. From the graph we can come to know that the blue point are the student mark and student hour .Also as the student hour is increasing ,the student mark is also increasing. Also the data is in linear model.

Prepare the data for Machine Learning algorithms

First step in preparing the data for machine learning is data cleaning .In our dataset there can be some point where we can find null values. When we are training our machine learning model we always have to give clean data.

Now we will check how many values in our dataset are null so that we can include some value in that place



A screenshot of a Jupyter Notebook interface. At the top, a code cell contains the command `df.isnull()` with a play button icon to its left. Below the code cell, the output is displayed as a table with two columns: `study_hours` and `student_marks`. The table shows rows 0 through 199. Row 2 has a `True` value in the `study_hours` column, while all other rows have `False`. All values in the `student_marks` column are `False`. The table is truncated with ellipses in the middle. At the bottom of the output, it says "200 rows x 2 columns".

	study_hours	student_marks
0	False	False
1	False	False
2	True	False
3	False	False
4	False	False
...
195	False	False
196	False	False
197	False	False
198	False	False
199	False	False

200 rows x 2 columns

Here we are not getting the exact count, now for checking the exact numbers we can give the command like `df.isnull().sum()` ,from which we can get the exact number where we are getting null values.

```
df.isnull().sum()
study_hours      5
student_marks    0
dtype: int64
```

Here in the study hour columns we have five values as null values which we have to insert some values

For solving that we will take the mean of the dataframe and then replace that mean value in the null value places so that our machine learning model works correctly.

```
df.mean()
study_hours      6.995949
student_marks    77.933750
dtype: float64
```

```
df2 = df.fillna(df.mean())
```

Now the null value is replaced by the mean values for checking that we can give the command like `df2 = df.isnull().sum`

```
df2.isnull().sum()
study_hours      0
student_marks    0
dtype: int64
```

Now if want to check the value we can check by giving `df2.head`

	study_hours	student_marks
0	6.830000	78.50
1	6.560000	76.74
2	6.995949	78.68
3	5.670000	71.82
4	8.670000	84.19

Comparison of data frame after cleaning and before cleaning

Before cleaning


After cleaning


	study_hours	student_marks
0	6.83	78.50
1	6.56	76.74
2	NaN	78.68
3	5.67	71.82
4	8.67	84.19

	study_hours	student_marks
0	6.830000	78.50
1	6.560000	76.74
2	6.995949	78.68
3	5.670000	71.82
4	8.670000	84.19

2nd index value in student hour is replaced by the mean value.

❖ SPLITTING THE DATA

```
 x = df2.drop("student_marks", axis = "columns")  
y = df2.drop("study_hours", axis = "columns")  
print("shape of X = ", X.shape)  
print("shape of y = ", y.shape)
```

```
 shape of X = (200, 1)  
shape of y = (200, 1)
```

Here we are taking two variables X and y. We know that X variable is for the student hour and y is for the student mark so in the above diagram we are dropping student mark from the X variable and giving axis is equal to column so X equal to student hour similarly for the y variable will be equal to the student mark. Finally after executing we get the shape of X and y

IMPLEMENTATION

LIBRARIES USED

Pandas ,Numpy ,Matplotlib.pyplot and sklearn are the major libraries used in the project. Pandas is mainly used for data analysis. Pandas allows importing data from various file formats such as comma-separated values, JSON, SQL, Microsoft Excel. Pandas allows various data manipulation operations such as merging, reshaping, selecting, as well as data cleaning, and data wrangling features. Numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. Moreover Numpy forms the foundation of the Machine Learning stack. Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. SciPy makes use of Matplotlib. Scikit-learn is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy.

MODELS USED

❖ SPLITTING THE DATA INTO TRAINING AND TESTING

Train/Test is a method to measure the accuracy of your model. It is called Train/Test because you split the the data set into two sets: a training set and

a testing set You test the model using the testing set. Train the model means create the model. By using similar data for training and testing, you can minimize the effects of data discrepancies and better understand the characteristics of the model. After a model has been processed by using the training set, you test the model by making predictions against the test set.

```
[23] from sklearn.model_selection import train_test_split
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state=51)
      print("shape of X_train = ", X_train.shape)
      print("shape of y_train = ", y_train.shape)
      print("shape of X_test = ", X_test.shape)
      print("shape of y_test = ", y_test.shape)
```

```
shape of X_train = (160, 1)
shape of y_train = (160, 1)
shape of X_test = (40, 1)
shape of y_test = (40, 1)
```

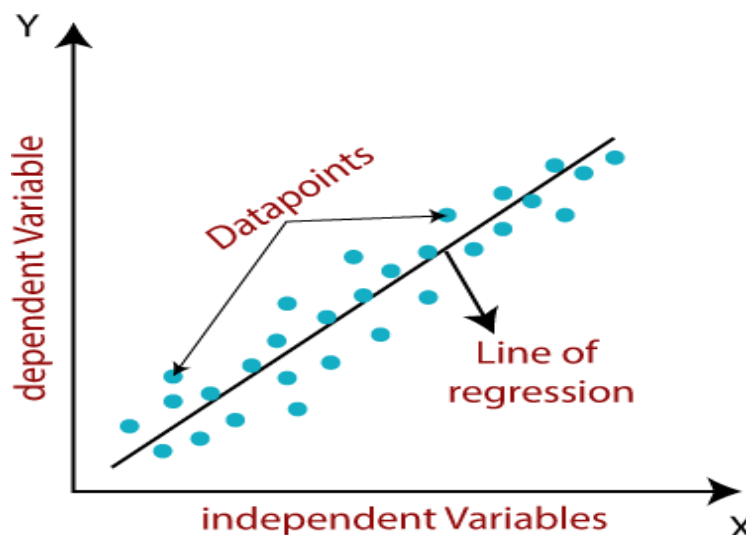
We have a class called `train_test_split` which we import `sklearn model selection` which is used for testing our machine learning module. In this class we have to give to two variable called the independent variable and independent variable. In our project the student hour is the independent variable and student mark is the dependent variable which are `X, y` respectively. In the above diagram we are giving test size equal to 0.2 which means we are giving 20 % of our data for testing. The `train_test_split` gives us 4 dataframe called `x_train, x_test, y_train` and `y_train`.

❖ Selecting a model and training it(LINEAR REGRESSION)

LINEAR REGRESSION is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, product price, etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (x) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables.




```
[26] # y = m * x + c
      from sklearn.linear_model import LinearRegression
      lr = LinearRegression()
```

We are importing linear regression from sklearn linear model and then create an object for the linear regression.

Now we should train this model from x_train and y_train

```
lr.fit(x_train,y_train)

LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

Now if we want to get m and c value that is the coefficient and intercept value

```
[28] lr.coef_

array([[3.93571802]])
```

```
lr.intercept_

array([50.44735504])
```

So we have got the m and c value we can predict the y value.

```
m = 3.93
c = 50.44
y = m * 4 + c
y
```

66.16

```
lr.predict([[4]])[0][0].round(2)
```

66.19

Now we are predicting the y value by using the y pred by using the x_test

```
y_pred = lr.predict(x_test)
y_pred
```

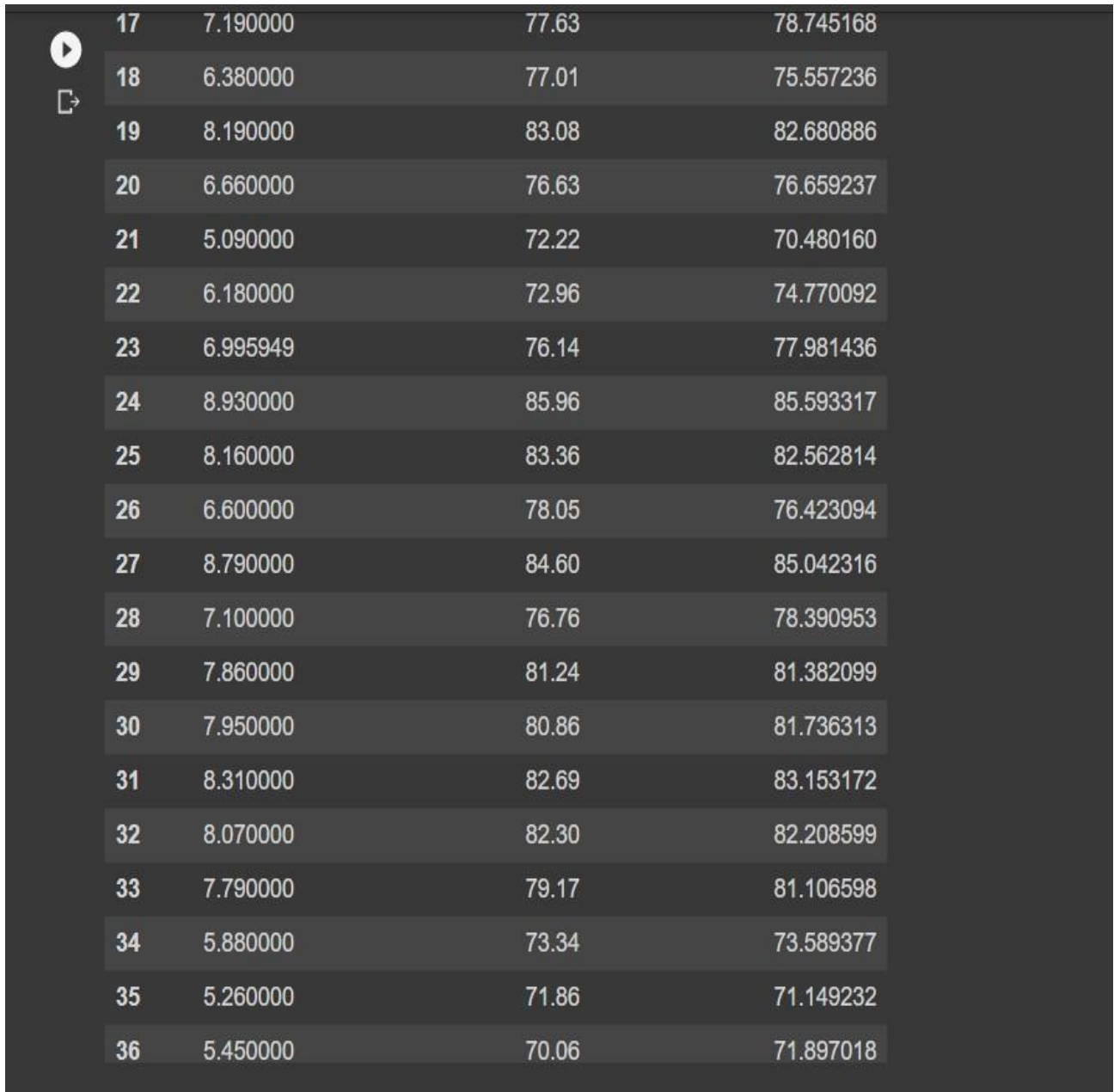
```
array([[83.11381458],
       [78.9025963 ],
       [84.57003024],
       [85.82946001],
       [84.72745896],
       [80.75238377],
       [72.84159055],
       [71.66087515],
       [73.23516235],
       [71.66087515],
       [73.47130543],
       [76.38373677],
       [73.23516235],
       [73.58937697],
       [82.95638585],
       [70.40144538],
       [73.23516235],
       [78.74516758],
       [75.55723598],
       [82.68088559],
       [76.65923703],
       [70.48015974],
       [74.77009238],
       [77.98143645],
       [85.59331693],
       [82.56281405],
       [76.42309395],
       [85.0423164 ],
       [78.39095296],
```

```
[75.55723598],
[82.68088559],
[76.65923703],
[70.48015974],
[74.77009238],
[77.98143645],
[85.59331693],
[82.56281405],
[76.42309395],
[85.0423164 ],
[78.39095296],
[81.38209865],
[81.73631327],
[83.15317176],
[82.20859943],
[81.10659839],
[73.58937697],
[71.1492318 ],
[71.89701823],
[81.53952737],
[72.60544747],
[71.93637541]]])
```

Now with the help of numpy library we will combine the actual and predicted value and concadinate all the 3 values and also give the columns name also create a dataframe for it

```
[33] pd.DataFrame(np.c_[X_test, y_test, y_pred], columns = ["study_hours", "student_marks_original", "student_marks_predicted"])
```

	study_hours	student_marks_original	student_marks_predicted
0	8.300000	82.02	83.113815
1	7.230000	77.55	78.902596
2	8.670000	84.19	84.570030
3	8.990000	85.46	85.829460
4	8.710000	84.03	84.727459
5	7.700000	80.81	80.752384
6	5.690000	73.61	72.841591
7	5.390000	70.90	71.660875
8	5.790000	73.14	73.235162
9	5.390000	73.02	71.660875
10	5.850000	75.02	73.471305
11	6.590000	75.37	76.383737
12	5.790000	74.44	73.235162
13	5.880000	73.40	73.589377
14	8.260000	81.70	82.956386
15	5.070000	69.27	70.401445
16	5.790000	73.64	73.235162



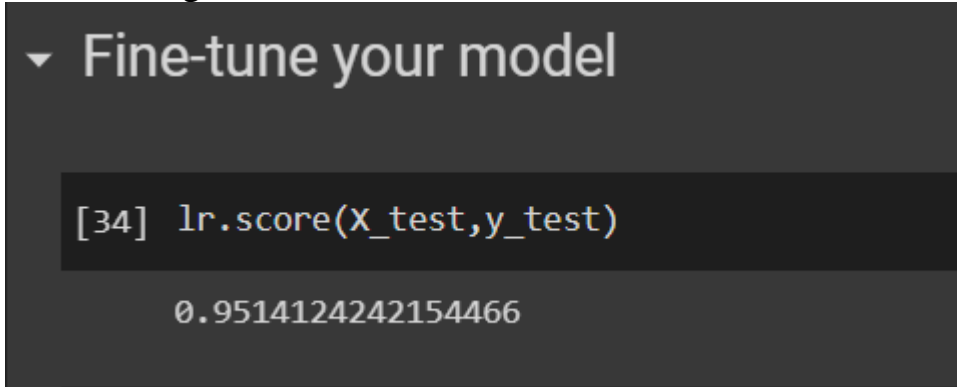
17	7.190000	77.63	78.745168
18	6.380000	77.01	75.557236
19	8.190000	83.08	82.680886
20	6.660000	76.63	76.659237
21	5.090000	72.22	70.480160
22	6.180000	72.96	74.770092
23	6.995949	76.14	77.981436
24	8.930000	85.96	85.593317
25	8.160000	83.36	82.562814
26	6.600000	78.05	76.423094
27	8.790000	84.60	85.042316
28	7.100000	76.76	78.390953
29	7.860000	81.24	81.382099
30	7.950000	80.86	81.736313
31	8.310000	82.69	83.153172
32	8.070000	82.30	82.208599
33	7.790000	79.17	81.106598
34	5.880000	73.34	73.589377
35	5.260000	71.86	71.149232
36	5.450000	70.06	71.897018

TEST RESULTS

RESULTS

❖ ACCURACY CHECK

We are using a method called `lr.score` for which we will check for `x_test` and `y_test`



```
▼ Fine-tune your model

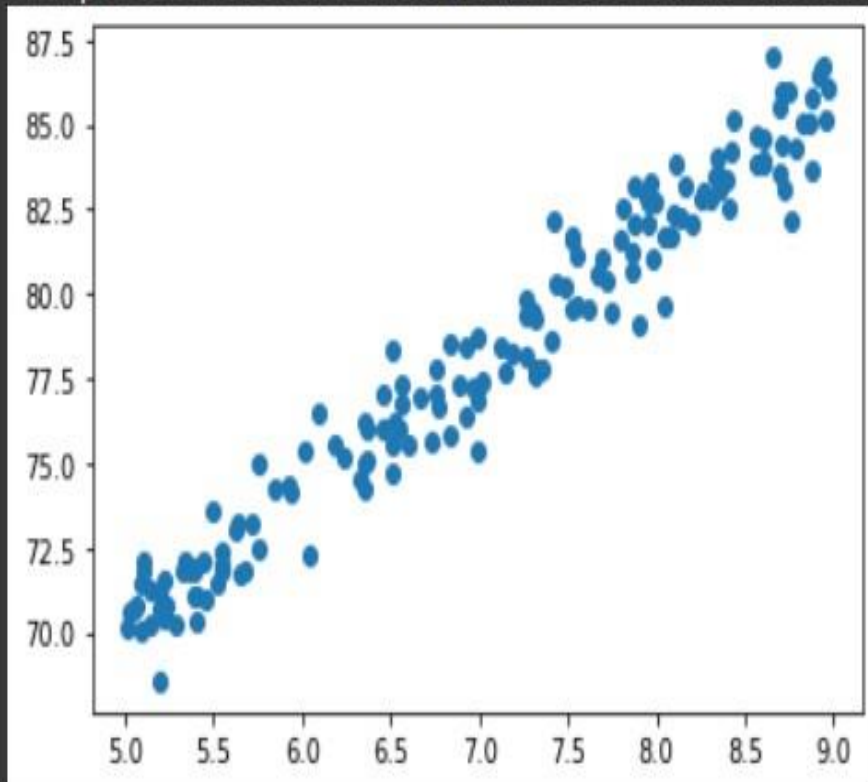
[34] lr.score(x_test,y_test)

0.9514124242154466
```

Based on this accuracy how our machine learning model predicted the line we have to use the below command

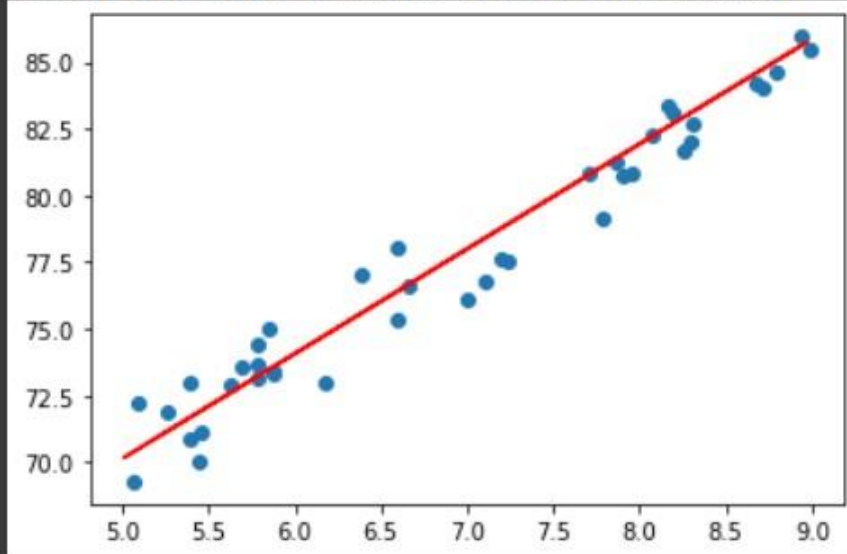
```
[35] plt.scatter(X_train,y_train)
```

```
<matplotlib.collections.PathCollection at 0x7f5975d44e90>
```



```
[36] plt.scatter(X_test, y_test)
      plt.plot(X_train, lr.predict(X_train), color = "r")
```

```
[<matplotlib.lines.Line2D at 0x7f5975ccb690>]
```



The dots in the the graph on the red line and our model has predicted 95 % accuracy

❖ Saving our model

For saving our model we use a library called joblib, with the help of dump method we can save the linear regression model and also give a name for the model as student mark prediction.pkl

Now we have to import the model for which we use joblib.load() and assign it to a variable and then predict the model

```
[40] import joblib
      joblib.dump(lr, "student_mark_predictor.pkl")

      ['student_mark_predictor.pkl']

[41] model = joblib.load("student_mark_predictor.pkl")
```

❖ Final result student mark prediction

If studying hour =10

```
▶ model.predict([[10]])[0][0]

89.80453520342567
```

If studying hour =2



```
model.predict([[2]])[0][0]
```

```
58.31879107023909
```

PROJECT CONTRIBUTION

For a student marks prediction project, there are several areas where you can make significant contributions, depending on your skills and interests. Here are some ways you can contribute:

Data Collection and Preprocessing

- **Collect Data:** Gather data on student marks, attendance, socio-economic background, study hours, and other relevant factors.
- **Clean Data:** Handle missing values, correct errors, and normalize the data.
- **Feature Engineering:** Create new features from the raw data that might help improve the prediction accuracy.

Exploratory Data Analysis (EDA)

- **Visualizations:** Create graphs and plots to understand the distribution of marks, correlation between different features, and trends.
- **Statistical Analysis:** Perform statistical tests to identify significant factors affecting student performance.

Model Building

- **Algorithm Selection:** Choose appropriate algorithms (e.g., linear regression, decision trees, random forests, neural networks) for predicting student marks.
- **Training Models:** Train the chosen models on the dataset and fine-tune their hyperparameters.
- **Model Evaluation:** Evaluate the models using metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2 score.

Deployment

- **Build a Web Application:** Create a user-friendly web application where teachers or students can input data and get predictions.
- **API Development:** Develop APIs to serve the prediction model for integration with other systems.

Documentation and Reporting

- **Project Documentation:** Document the entire project including data collection, preprocessing steps, model building, and evaluation.
- **Report Writing:** Write a detailed report or research paper on the findings and methodology of the project.

Ethical Considerations

- **Bias Mitigation:** Ensure the model does not have biases against any particular group of students.
- **Privacy:** Make sure that student data is anonymized and handled with care to maintain privacy.

Examples of Contributions

1. Data Scientist Role:

- Focus on collecting and preprocessing the data, building and evaluating models.
- Provide insights from the data through EDA.

2. Software Developer Role:

- Develop the web application and APIs for deploying the prediction model.
- Ensure the system is scalable and user-friendly.

3. Researcher Role:

- Conduct a literature review to understand existing methods for student marks prediction.
- Experiment with different algorithms and report the findings.