

**UNIVERSITY INSTITUTE OF TECHNOLOGY  
BARKATULLAH UNIVERSITY  
BHOPAL**

**Department of Computer Science & Engineering**



**MAJOR PROJECT  
ON  
“STUDENT PERFORMANCE PREDICTION SYSTEM USING  
MACHINE LEARNING”**

**Submitted for the Partial Fulfillment of the Requirement for the**

**Award of Degree of**

**Bachelor of Engineering (B.E)**

**Year 2021**

**University Institute of Technology**

**Barkatullah University, Bhopal**

**By**

**NEHA AHIRWAR**

**Under the Guidance**

**Of**

**Ms. Jagriti Chand**

Assistant Professor  
CSE, UIT-BU

**Dr. Divakar Singh**

Head of Department  
CSE, UIT-BU

# UNIVERSITY INSTITUTE OF TECHNOLOGY BARKATULLAH UNIVERSITY, BHOPAL

Department of Computer Science & Engineering



## CERTIFICATE

YEAR 2021

This is to certify that **ku. Neha Ahirwar** have successfully completed this project work titled “ **Student Performance Prediction System Using Machine Learning**” for the partial fulfillment of the award of degree of Bachelor of Engineering in Computer Science & Engineering in the year 2021 Barkatullah University Institute of Technology , Bhopal.

**Ms. Jagriti Chand**

Assistant Professor

CSE,UIT-BU

**Dr.Divakar Singh**

Head of Department

CSE, UIT-BU

**Dr.N.K Gaur**

Director

UIT-BU(Bhopal)

**UNIVERSITY INSTITUTE OF TECHNOLOGY  
BARKATULLAH UNIVERSITY, BHOPAL**

**Department of Computer Science & Engineering**



**DECLARATION**

**YEAR 2021**

I hereby declare that the work which is being presented in this report entitled “**Student Performance Prediction System Using Machine Learning**” submitted in partial fulfillment of the requirement for the award of degree of Bachelor of Engineering in Computer Science & Engineering from University Institute of Technology BU; Bhopal is an authentic record of our project work.

This is my original work and has not been submitted earlier for the award of any other degree, diploma or any other certificate.

**Name**

Neha Ahirwar

**Enrollment no.**

R188237200019

**Signature**

## **ACKNOWLEDGMENT**

I would like to express my deepest appreciation to all those who provided us the possibility to complete this report.

Before I get into the thick of the things I would like to add a few heartfelt words for the teachers who are part of this project in a numerous way. Teacher who gave us support right from the project idea were convinced.

First of all I would like to thank our project guide **Ms. Jagriti Chand** for allowing us to undergo this major project idea. I would also like to pay our sincere regards to our **Dr. Divakar Singh (H.O.D, CSE)** for providing effective platform and support in the development of this project and finally I would like to render our thanks to our **Honorable Director Dr.N.K Gaur** for his guidance in this project titled **“Student Performance Prediction System Using Machine Learning”**.

Last but not least I would like to thank our parent and friends for their support and cooperation. Regardless of the source I wish to express our gratitude to those who may have contributed to this work; even though anonymously.

**Submitted by**

Neha Ahirwar

## **ABSTRACT**

Analyzing and prediction of academic performance is important for any education institutions. Predicting student performance can help teachers to take steps in developing strategy for improving performance at early stages. With the advancement of machine learning supervised techniques developing these kinds of applications are helping teachers to analyze students in better way compare to existing methods.

This system aims to predict student's marks using machine learning. Through this project we can solve the problem like how many hours need to do the study to get 99% marks? Or how many hours need to do the study to pass the exam? If I will do study x (4) hours per day so how many marks I will get.

Student performance prediction is a method for focusing an understudies presentation dependent on his/her study hours. This additionally makes the student know whether he/she is in a situation to arrive at his/her normal or expected marks or not. So this system developed to analyze and predict the student's performance only.

We are using linear regression algorithm to create machine learning model, Flask server to run web portal, Python as a programming language, HTML, CSS and Javascript are used for user interface.

.

# **TABLE OF CONTENTS**

	PAGE NO
LIST OF FIGURES.....	i
LIST OF TABLES.....	ii
LIST OF ABBREVIATIONS.....	iii
<b>1. INTRODUCTION.....</b>	<b>1</b>
<b>2. SYSTEM REQUIREMENTS .....</b>	<b>2-4</b>
2.1 Software requirements .....	2-3
2.2 Hardware requirements .....	3-4
<b>3. TECHNICAL DESCRIPTION.....</b>	<b>5-7</b>
3.1 Supervised machine learning .....	5
3.2 How supervised machine learning works.....	6
3.3 The proposed system.....	6
3.4 Supervised machine learning algorithm.....	6-7
<b>4. PROJECT DESCRIPTION.....</b>	<b>8-10</b>
<b>5. CODING &amp; SNAPSHOT.....</b>	<b>11-24</b>
5.1 Instruction to set up the environment.....	11-13
5.2 Coding.....	14-21
5.3 Deploy ML Model On Local Machine.....	22-24
<b>6. TESTING.....</b>	<b>25</b>
<b>7. CONCLUSION.....</b>	<b>26</b>
7.1 Limitation.....	26
7.2 Benefits.....	26
7.3 Future work.....	26
<b>REFERENCE.....</b>	<b>27</b>

## LIST OF FIGURES

<b>S.No.</b>	<b>Figure No.</b>	<b>Figure Name</b>	<b>Page No.</b>
1.	Fig. 3.1	Types Of Supervised Learning	5
2.	Fig. 3.3	Proposed System	6
3.	Fig. 3.4	Linear Regression	7
4.	Fig. 4.2	Machine Learning	8

## **LIST OF TABLES**

<b>S.No.</b>	<b>Table Number</b>	<b>Table Name</b>	<b>Page No.</b>
1.	Table 2.2	Software Requirements	2
2.	Table 2.3	Hardware Requirements	4



## **LIST OF ABBREVIATIONS**

1. HTML    Hyper Text Markup Language
2. SML    Supervised Machine learning
3. KNN    K-nearest neighbors
4. SVM    Support Vector Machine
5. NBN    Naïve Bayesian model
6. OS    Operating System
7. VCS    Version Control System
8. CPU    Central processing unit
9. RAM    Random Access memory
10. AI    Artificial intelligence
11. ML    Machine learning
12. CSS    Cascading Style Sheets

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 Introduction**

Performance analysis of outcome based on learning is a system which will strive for excellence at different levels and diverse dimensions in the field of student's interests. This system developed to analyze and predict the student's performance only.

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 performance and there by taking early steps to improve student's performance and teaching quality.

This system aims to predict student's marks using linear regression. The idea behind this analysis is to predict the marks of students by their studying hours. Through this project we can determine:

- How many hours need to do the study to get 99% marks?
- If I will do study  $x$  ( ) hours per day so how many marks I will get.

Through these points the school / institutes can determine the performance of the student.

## **CHAPTER 2**

### **SYSTEM REQUIREMENTS**

#### **2.1 Introduction**

To be used efficiently, all computer software needs certain hardware component or other software resources to be present on a computer these prerequisites are known as system requirements.

#### **2.2 Functional Requirements**

Software is that part of a computer system that consists of data and computer instructions, in contrast to the physical hardware from which the system is build. In computer science and software engineering, computer software is all information processed by computer systems, programs and data. Computer software includes computer programs, libraries and related non executable data, such as online documentation or digital media. Here table 2.1 shows that the used tools & software in project.

**Table 2.2 Software Requirements**

Technology	Machine Learning
Tools	Anaconda Navigator
Programming language	Python
Framework	Flask
UI Designing	HTM,CSS, JavaScript
Dataset	Excel
Text Editor	Jupyter Notebook & Spyder

##### **2.2.1 Machine learning**

Machine learning is the field of study that gives computers the capability to learn without being explicitly programmed. Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information.

##### **2.2.2 Anaconda Navigator**

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. It is available for windows ,macOS, and Linux.

##### **2.2.3 Python**

Python is an interpreted, interactive, object oriented high-level and general- purpose programming language. Python is not only programming language its also a scripting language .So what we can do with python

1. System Programming
2. Internet Scripting
3. Database Programming
4. Gaming And So On.

#### **2.2.4 Flask**

**Flask** is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself.

#### **2.2.5 Html, Css, Javascript**

1. **HTML** provides the basic structure of sites, which is enhanced and modified by other technologies like CSS and JavaScript.
2. **CSS** is used to control presentation, formatting, and layout.
3. **JavaScript** is used to control the behavior of different elements

#### **2.2.6 Excel**

Microsoft Excel is a spreadsheet developed by Microsoft for Windows, macOS, Android and iOS. It features calculations, graphing tools, pivot tables, and a macro programming language called Visual Basic for Applications (VBA)

#### **2.2.7 Spyder & Jupyter Notebook**

Spyder is a free and open source scientific environment written in Python, for Python, and designed by and for scientists, engineers and data analysts. It features a unique combination of the advanced editing, analysis, debugging, and profiling functionality of a comprehensive development tool with the data exploration, interactive execution, deep inspection, and beautiful visualization capabilities of a scientific package.

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation. Statistical modeling, data visualization, machine learning, and much more.

**2.3 Hardware Requirements:** The common set of requirements defined by any operating system is the physical computer resources, also known as hardware or we can say that hardware is the collection of all parts which we can touch.

#### **Table 2.3 Hardware Requirements**

Processor	Intel Core
RAM	4GB

1. RAM 4GB is used as it will provide fast reading and writing capabilities and will in turn support in processing.
2. Intel Core 2 duo CPU is used as a processor because it is faster than other processors and provide reliable and stable and we can run our PC for long –time, by using this processors , we can keep on developing our project without any worries.

## CHAPTER 3

### TECHNICAL DESCRIPTION

#### 3.1 SUPERVISED MACHINE LEARNING (SML)

**3.1.1 Supervised machine learning:** Supervised machine learning is the subcategory of machine learning. Supervised learning, as the name indicates, has the presence of a supervisor as a teacher. So Supervised machine learning is when we teach or train the machine using data that are well labeled. The main advantages of supervised learning; It allows collecting data and produces data output from previous experiences. Supervised learning is also known as supervised machine learning (SML).

**Types of Supervised Learning:** There are two types of supervised learning problems:

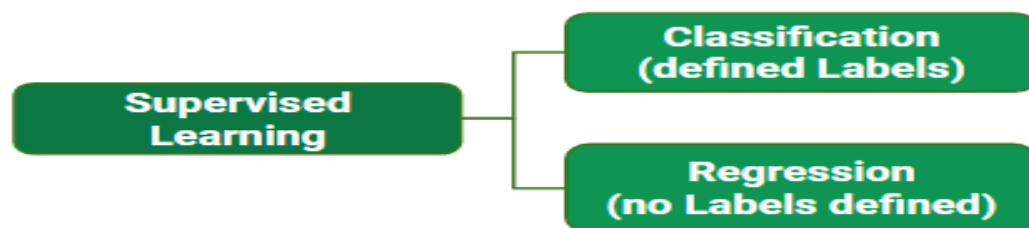


Figure3. 1– Types of supervised learning

##### 1. Classification

Classification is a subcategory of supervised learning. Basically classification is a supervised learning task where output is having defined labels or discrete value. There are two types of classification problems: binary and multi-class classification.

**1.1 Binary classifications:** when the supervised learning algorithms labels input data into two distinct classes.

Or binary model predicts either yes or no. So these types of classification are called binary classification.

**Example:** email spam detection, which each email is spam->1 spam; or is not->0.

**1.2 Multi-class classifications:** a multi-class classification means categorizing data into more than two classes or we can say that multi-class classification, model predicts more than one class. Like hand written character recognition where classes go from 0 to 9.

##### 2. Regression

It's a supervised learning task where output is having continuous or real value, such as "salary" or "weight". For example: This is the regression task 1. Predicting age of a person

2. Predicting nationality of a person. Solution: predicting age of a person because it is a real value, predicting nationality is categorical.

## 3.2 How Supervised Machine Learning Works?

Machine learning uses two types of techniques: supervised learning, which trains a model on known input and output data so that it can predict future outputs, and unsupervised learning, which finds hidden patterns or intrinsic structures in input data.

The majority of practical machine learning uses supervised learning. Supervised learning is where you have input variables ( $x$ ) and an output variable ( $Y$ ) and you use an algorithm to learn the mapping function from the input to the output  $Y = f(X)$ . The goal is to approximate the mapping function so well that when you have new input data ( $x$ ) that you can predict the output variables ( $Y$ ) for that data

.

## 3.3 The Proposed System

### 3.3.1 The system components

The following diagram, figure , shows the main steps and components of the proposed machine learning system.

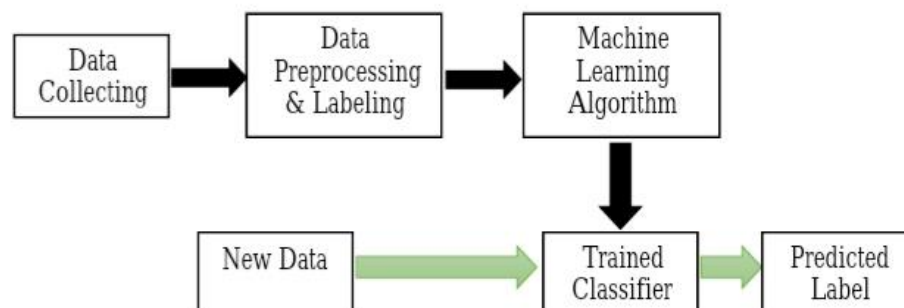


Figure3.3 - The main steps and components of the proposed system

The first step is collecting the data from the data sources. The second step is preprocessing the data in order to get a normalized dataset and then labeling the data rows. In the third step, the result of the second step, the training and testing dataset, is fed to the Machine learning algorithm. The Machine Learning Algorithm builds a model using the training data and tests the model using the test data. Finally, the Machine Learning Algorithm produces a trained model or a trained classifier that can take as an input a new data row and predicts its label.

## 3.4 Supervised Machine Learning Algorithms

1. Regression: Regression algorithms are used if there is a relationship between the input variable and the output variable. Below are some Regression algorithms which come under supervised learning:

- Linear Regression
- Regression Tree

- Non-Linear Regression
- Bayesian Linear Regression
- Polynomial Regression.

2. Classification: Classification algorithms are used when the output variable is categorical. Below are some Regression algorithms which come under supervised learning:

- Random forest
- Decision Tree
- Logistic Regression
- Support vector machines etc.

In this project we have used linear Regression algorithm.

### 3.4.1 Linear Regression Algorithm

**Linear Regression** is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.

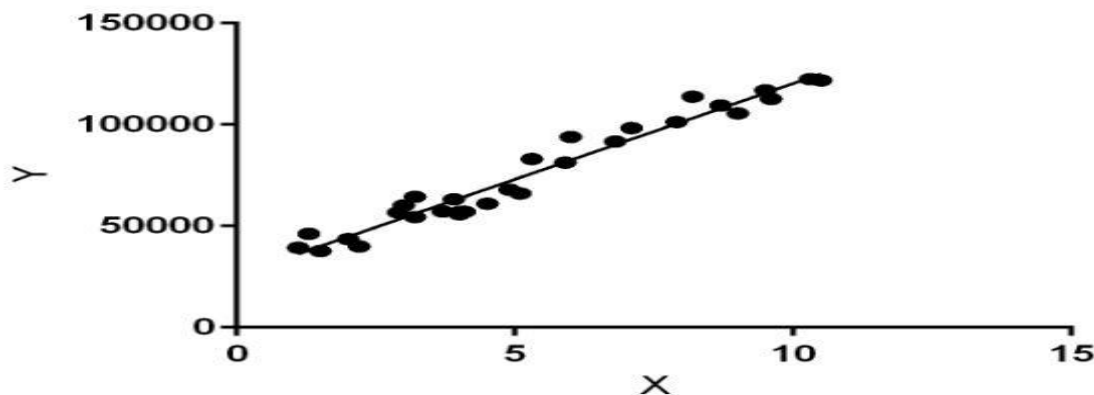


Figure 3.4 - Linear Regression

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between X(input) and Y(output). Hence, the name is Linear Regression. In the figure above, X (input) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

#### Formula

$$Y = m * x + c$$



# CHAPTER 4

## PROJECT DESCRIPTION

### Project Description

The essential ideas utilized in the student's performance prediction model are:

**4.1 Python as a programming language:** Python is a broadly utilized universally useful, elevated level programming language .It permits programming in object- oriented and procedural ideal models. In this venture, we have used to utilize diverse python libraries Like Numpy, Pandas and so on.

**4.2 Machine learning:** Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. It is one of the most powerful & emerging technologies, Here we have used machine learning to improve the precision of the model.

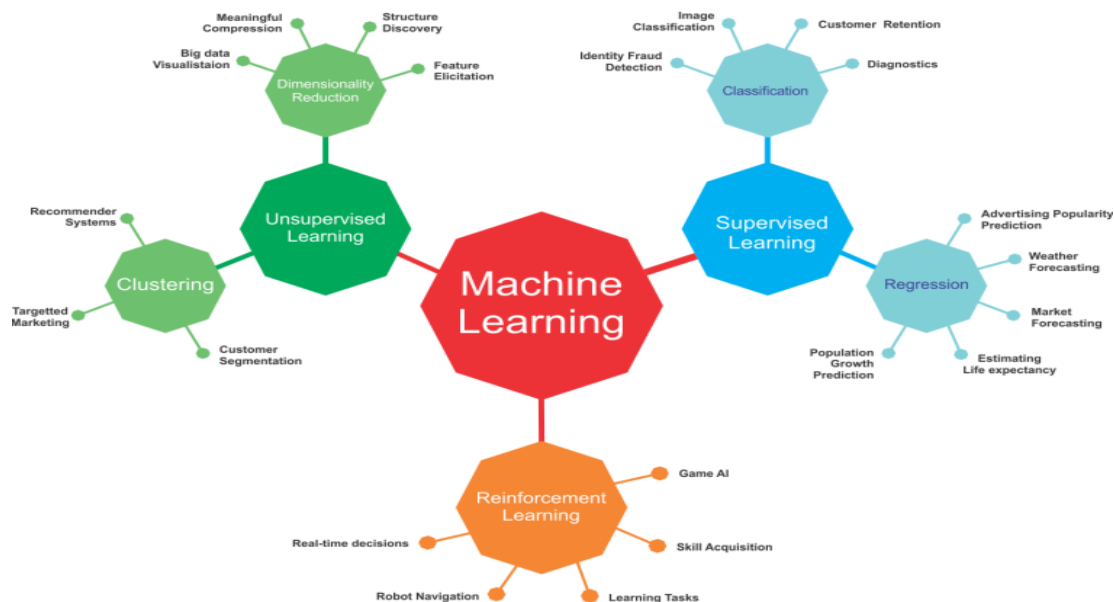


Figure 4.2- Machine Learning

**4.3 Dataset:** A data set (or dataset) is a collection of data. In the case of tabular data, a data set corresponds to one or more database tables, where every column of a table represents a particular variable, and each row corresponds to a given record of the data set in question. The data set lists values for each of the variables, such as height and weight of an object, for each member of the data set. Each value is known as a datum. Data sets can also consist of a collection of documents or files.

**4.4 This project involves a different strategy which comprises:**

**4.4.1 Data Collection:** Collecting data for training the ML model is the basic step in the machine learning pipeline. The predictions made by ML systems can only be as good as the

data on which they have been trained. Following are some of the problems that can arise in data collection:

- Inaccurate data: The collected data could be unrelated to the problem statement.
- Missing data: Sub-data could be missing. That could take the form of empty values in columns or missing images for some class of prediction.
- Data imbalance: Some classes or categories in the data may have a disproportionately high or low number of corresponding samples. As a result, they risk being under-represented in the model.
- Data bias: Depending on how the data, subjects and labels themselves are chosen, the model could propagate inherent biases on gender, politics, age or region, for example. Data bias is difficult to detect and remove.

**4.4.2 Data Preprocessing:** Real-world raw data and images are often incomplete, inconsistent and lacking in certain behaviors or trends. They are also likely to contain many errors. So, once collected, they are pre-processed into a format the machine learning algorithm can use for the model.

Pre-processing includes a number of techniques and actions:

- Data cleaning: These techniques, manual and automated, remove data incorrectly added or classified.
- Data imputations: Most ML frameworks include methods and APIs for balancing or filling in missing data. Techniques generally include imputing missing values with standard deviation, mean, median and k-nearest neighbors (k-NN) of the data in the given field.

**4.4.3 Generating Training And Testing Dataset:** Some algorithms used for testing for the assessment of input variables: LMT, Decision tree, Naive Bayesian and Support Vector Machine, linear regression etc.

## **4.5 Model Generation**

## **4.6 Prediction & Result**

## **4.7 Deploy ML Model On Local Machine**

**4.7.1 Flask Server:** Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Pocco. Flask is based on the Werkzeug WSGI toolkit and the Jinja2 template engine. Both are Pocco projects.

**4.7.2 WSGI:** The Web Server Gateway Interface (Web Server Gateway Interface, WSGI) has been used as a standard for Python web application development. WSGI is the specification of a common interface between web servers and web applications.

**4.7.3 Werkzeug:** Werkzeug is a WSGI toolkit that implements requests, response objects, and utility functions. This enables a web frame to be built on it. The Flask framework uses Werkzeug as one of its bases.

**4.7.4 Jinja2:** Jinja2 is a popular template engine for Python. A web template system combines a template with a specific data source to render a dynamic web page.

**4.7.5 Websit :** Create website using HTML ,CSS , JavaScript etc.

# CHAPTER 5

## CODING & SNAPSHOT

### 5.1 “Instruction to Set up the Environment”

1. Download Anaconda For Python. Make sure to download the “Python 3.7 Version” for the appropriate architecture

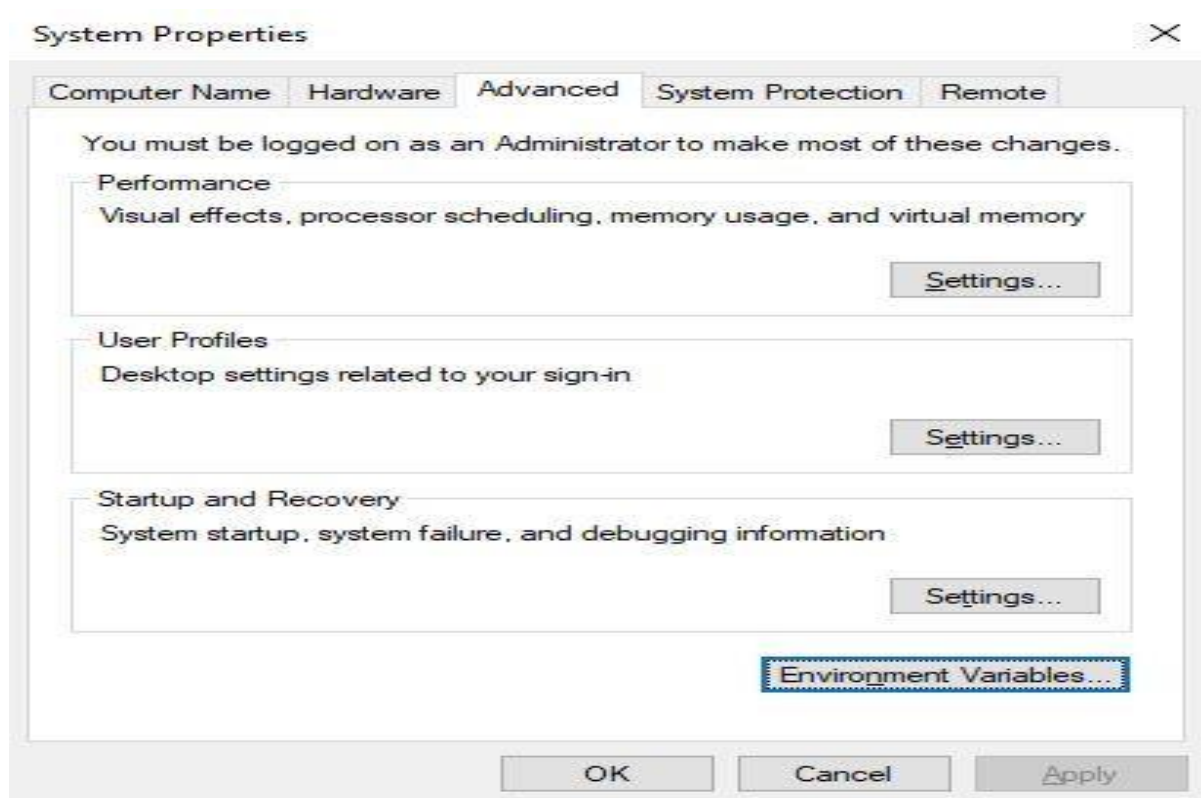


2. After the download is over, go through how to install anaconda on windows? And follow the instructions.

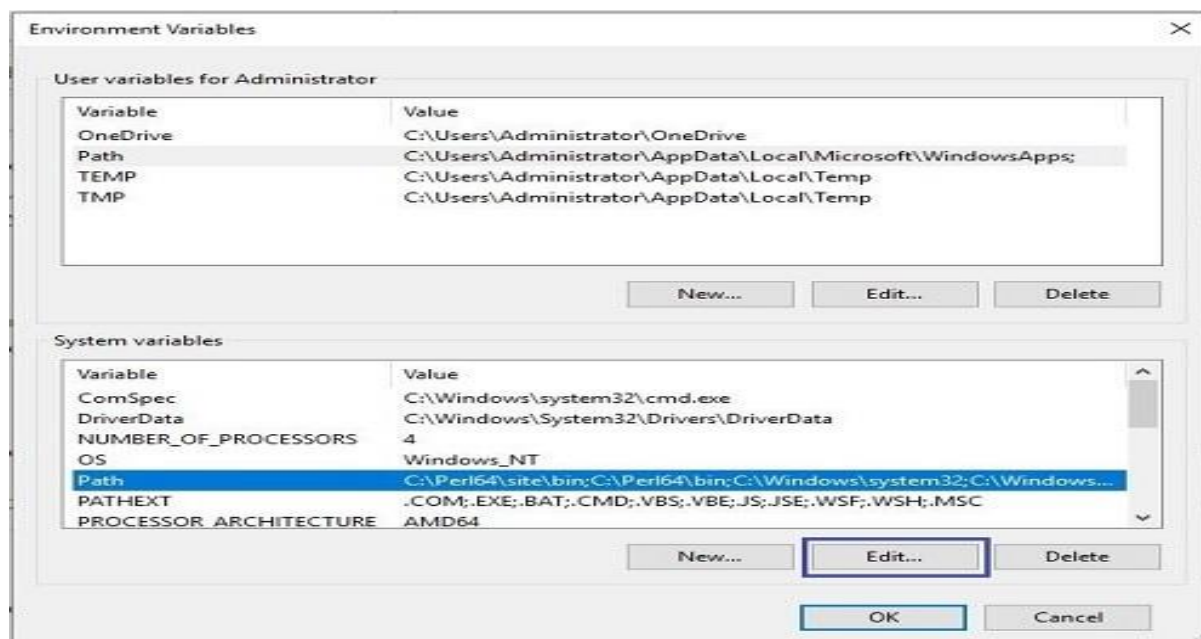
3. After the installation is done, we need to setup the environment variable.

Go to **control panel -> system and security -> system**. Under Advanced system setting option click on Environment Variables as shown below:



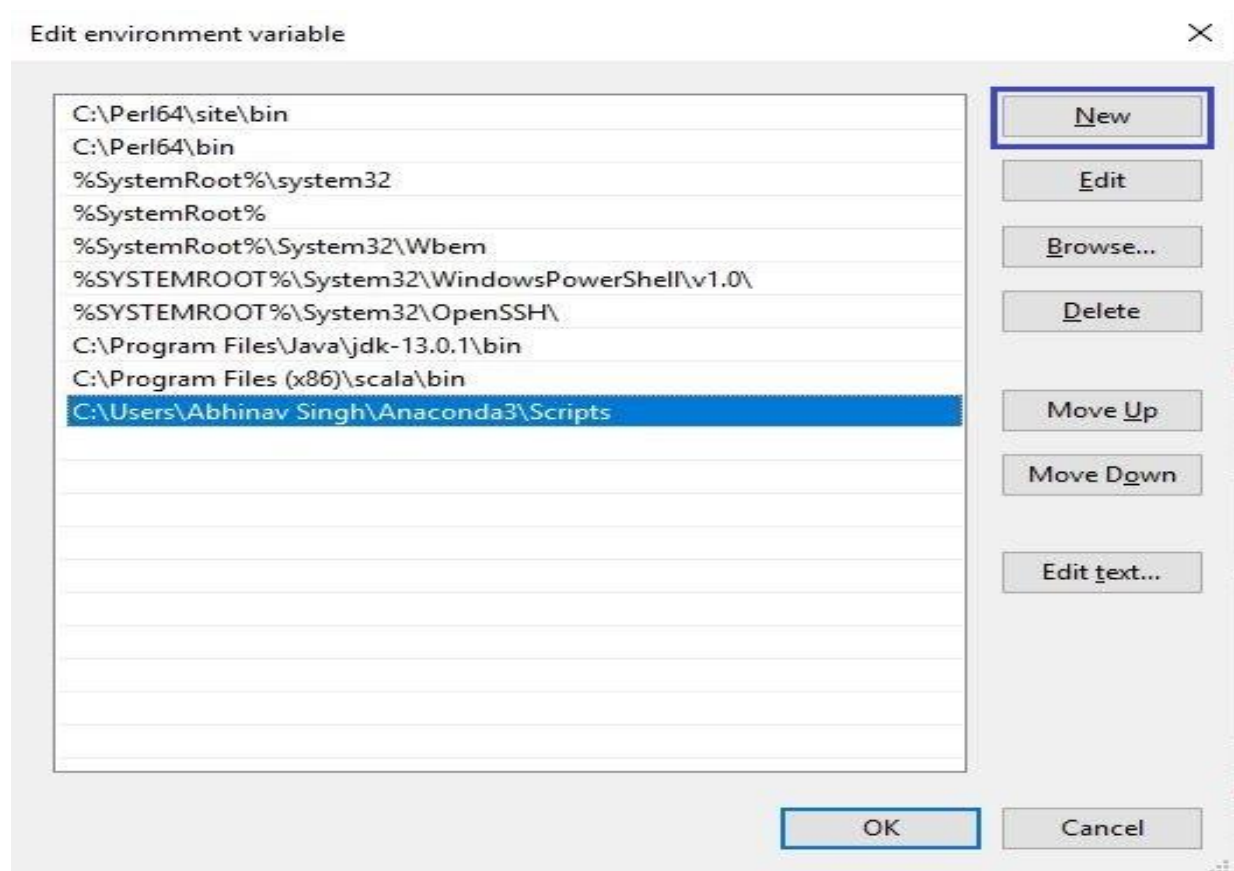


4. Now, we have to alter the “path” variable under system variables so that it also contains the path to the Anaconda environment. Select the “path” variable and click on the edit button as shown below:



5. We will see a list of different paths, click on the new button and then add the path where anaconda is installed.


6. Click on OK, save the settings and it is done!! Now to check whether the installation is done correctly, open command prompt and type anaconda-navigator. It will start the anaconda navigator App, if installed correctly.



## 5.2 Coding

we can use anaconda jupyter notebook or google colaboratory ( It's a jupyter notebook environment that requires no setup to use and runs entirely in the cloud).

### ▼ Business Problem

```
✓ 0s  #Import libraries  
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt
```

### ▼ Load Dataset

```
✓ 1s [2] path = r"https://drive.google.com/uc?export=download&id=13ZTYmL3E8S0nz-UKl4aaTZJaI3DVBGHM"  
df = pd.read_csv(path)
```

```
✓ 0s [4] df.head()
```

	study_hours	student_marks
0	6.83	78.50
1	6.56	76.74

```
[4]
```

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

```
[6] df.shape
```

```
(200, 2)
```

## Discover and visualize the data to gain insights

```
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
```

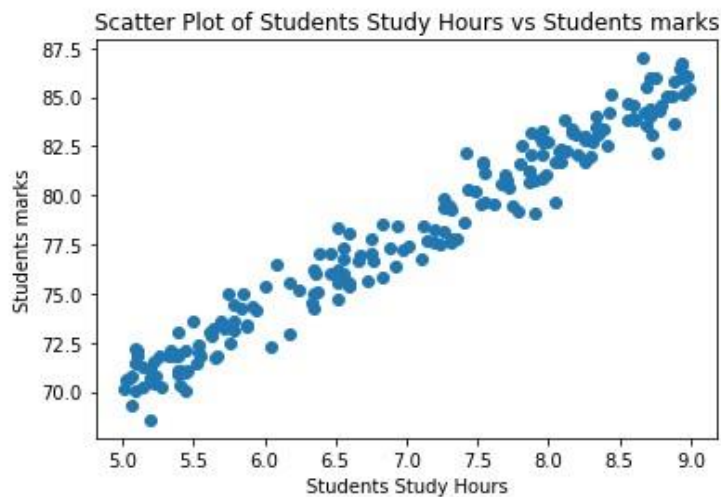
```
[8] df.describe()
```

	study_hours	student_marks
<b>count</b>	195.000000	200.000000
<b>mean</b>	6.995949	77.93375
<b>std</b>	1.253060	4.92570
<b>min</b>	5.010000	68.57000
<b>25%</b>	5.775000	73.38500
<b>50%</b>	7.120000	77.71000
<b>75%</b>	8.085000	82.32000



✓  
0s

```
[9] 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()
```



## ▼ Prepare the data for Machine Learning algorithms

✓  
1s

```
[10] # Data Cleaning
```

✓  
1s

```
df.isnull().sum()
```

```
study_hours      5
student_marks    0
dtype: int64
```

✓  
1s

```
[12] df.mean()
```

```
study_hours      6.995949
student_marks    77.933750
dtype: float64
```

✓  
1s

```
[13] df2 = df.fillna(df.mean())
```

✓  
1s

```
[14] df2.isnull().sum()
```

```
study_hours      0
student marks    0
```

```
✓ [15]      study_hours  student_marks
0s
      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
```

```
✓ [16]
0s      # split dataset
```

```
✓ [17] X = df2.drop("student_marks", axis = "columns")
0s      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)
```

```
✓ [18] from sklearn.model_selection import train_test_split
0s      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)
```

✓ 0s completed at 13:30

```
✓ [18] from sklearn.model_selection import train_test_split
0s      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)
```

## ▼ Select a model and train it

```
✓ [19] # y = m * x + c
0s      from sklearn.linear_model import LinearRegression
      lr = LinearRegression()
```

```
✓ [20] lr.fit(X_train, y_train)
```

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

```
✓ [21] lr.coef_
0s
```



0s



```
lr.intercept_
```



```
array([50.44735504])
```



0s

```
[23] m = 3.93
      c = 50.44
      y = m * 5 + c
      y
```

```
70.09
```



1s

```
[24] lr.predict([[6]])[0][0].round(2)
```

```
74.06
```



0s

```
[25] y_pred = lr.predict(X_test)
      y_pred
```

```
array([[83.11381458],
       [78.9025963 ],
       [84.57003024],
       [85.82946001],
```



3s



```
[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],
 [81.38209865],
 [81.73631327],
 [83.15317176],
```

✓ 0s  
[83.15317176],  
[82.20859943],  
[81.10659839],  
[73.58937697],  
[71.1492318 ],  
[71.89701823],  
[81.53952737],  
[72.60544747],  
[71.93637541]]

✓ 0s [26] pd.DataFrame(np.c\_[X\_test, y\_test, y\_pred], columns = ["study\_hours", "student\_marks\_original", "student\_marks\_predicted"])

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

✓ 0s

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

```
[26]
```

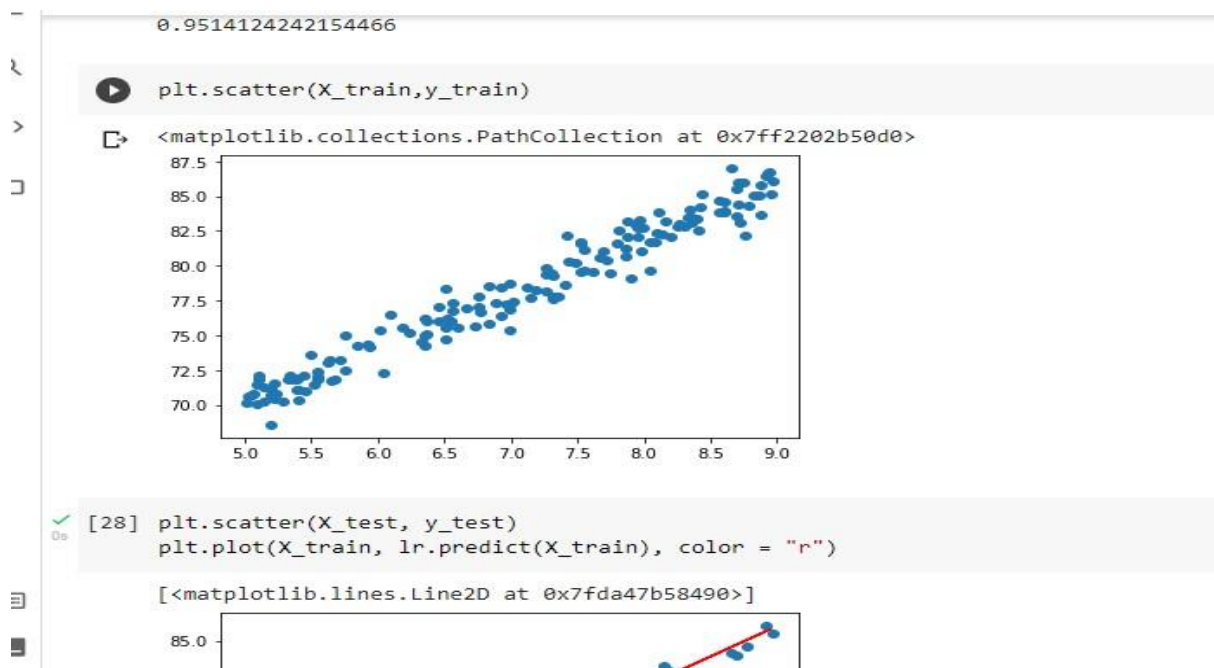
32	8.070000	82.30	82.208599
33	7.790000	79.17	81.106598
34	5.880000	73.34	73.589377
35	5.260000	71.86	71.149232
36	5.450000	70.06	71.897018
37	7.900000	80.76	81.539527
38	5.630000	72.87	72.605447
39	5.460000	71.10	71.936375

## Fine-tune our model

```
lr.score(X_test,y_test)
```

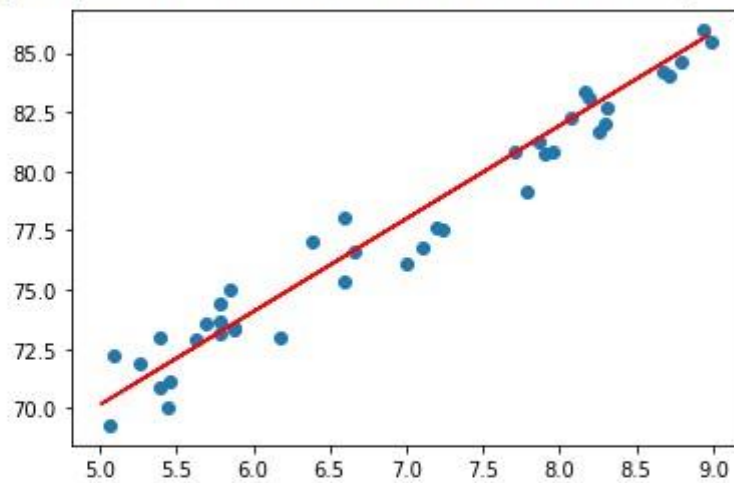
```
0.9514124242154466
```

```
plt.scatter(X_train,y_train)
```



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

[<matplotlib.lines.Line2D at 0x7fda47b58490>]



Present your solution

## Save ML Model

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

['student\_mark\_predictor.pkl']

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

```
✓ ▶ model.predict([[12]])[0][0]
```

📄 97.67597123672232



## 5.3 Deploy ML Model On Local System:

### 1. Create Flask server

### 2. Create website

```
1  # -*- coding: utf-8 -*-
2
3  import numpy as np
4  import pandas as pd
5  from flask import Flask, request, render_template
6  import joblib
7
8  app = Flask(__name__)
9
10 model = joblib.load("student_mark_predictor.pkl")
11
12 df = pd.DataFrame()
13
14 @app.route('/')
15 def home():
16     return render_template('index.html')
17
18 @app.route('/predict',methods=['POST'])
19 def predict():
20     global df
21
22     input_features = [int(x) for x in request.form.values()]
23     features_value = np.array(input_features)
24
25     #validate input hours
26     if input_features[0] <0 or input_features[0] >24:
27         return render_template('index.html', prediction_text='Please enter valid hours between 1 to 24 if
28
29
30     output = model.predict([features_value])[0][0].round(2)
31
32     # input and predicted value store in df then save in csv file
33     df= pd.concat([df,pd.DataFrame({'Study Hours':input_features,'Predicted Output':[output]}),ignore_in
34     print(df)
35     df.to_csv('smp_data_from_app.csv')
36
```

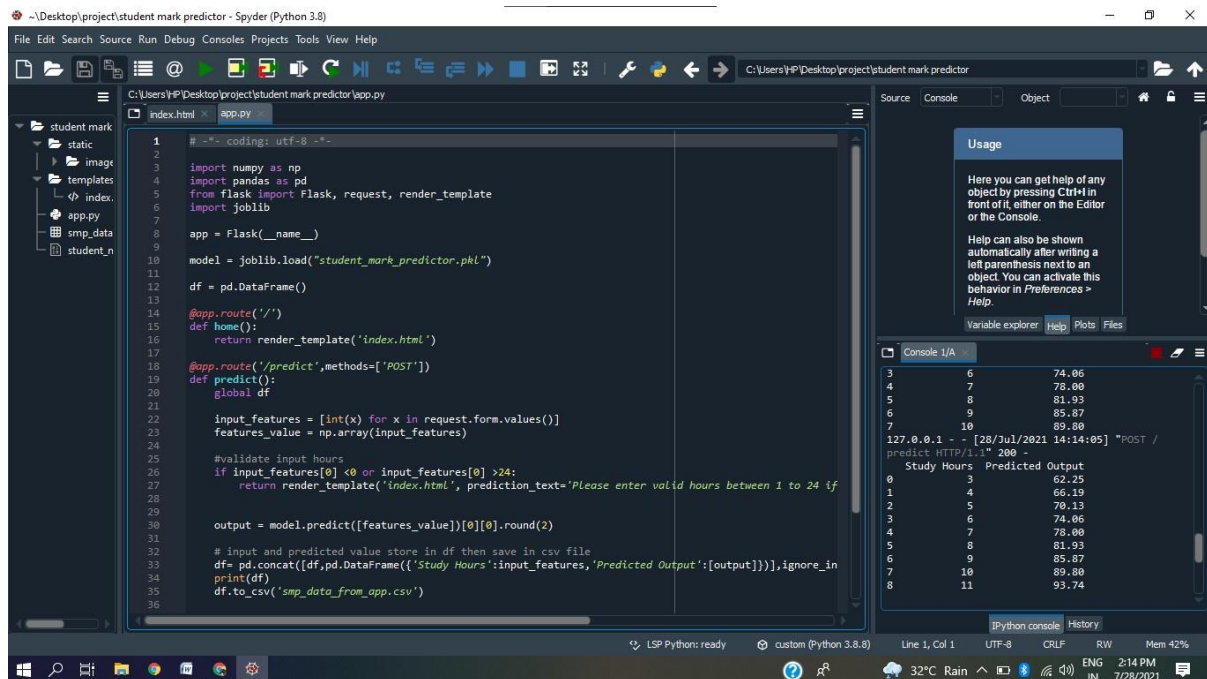
```
model = joblib.load("student_mark_predictor.pkl")
df = pd.DataFrame()
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/predict',methods=['POST'])
def predict():
    global df
    input_features = [int(x) for x in request.form.values()]
    features_value = np.array(input_features)
    #validate input hours
    if input_features[0] <0 or input_features[0] >24:
        return render_template('index.html', prediction_text='Please enter valid hours between 1 to 24 if
    output = model.predict([features_value])[0][0].round(2)
    # input and predicted value store in df then save in csv file
    df= pd.concat([df,pd.DataFrame({'Study Hours':input_features,'Predicted Output':[output]}),ignore_in
    print(df)
    df.to_csv('smp_data_from_app.csv')
    return render_template('index.html', prediction_text='You will get [{}%] marks, when you do study [{}
if __name__ == "__main__":
    app.run()
```

```
~\Desktop\project\student mark predictor - Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\HP\Desktop\project\student mark predictor\templates\index.html
index.html x app.py x
student mark
├── static
│   └── image
├── templates
│   ├── index.html
│   ├── app.py
│   ├── smp_data
│   └── student_n
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="UTF-8">
5 <title> Student Mark Predictor ML App </title>
6
7 <style>
8 h1 {color:red;} /*CSS code for heading h1*/
9
10 /*CSS code for button*/
11 .button_css{
12 color: #494949 !important;
13 text-transform: uppercase;
14 text-decoration: none;
15 background: #ffffff;
16 padding:20px;
17 border: 4px solid #494949 !important;
18 display: inline-block;
19 transition : all 0.4s ease 0s;
20 }
21 .button_css:hover{
22 color: #ffffff !important;
23 background: #f6b93b;
24 border-color: #f6b93b !important;
25 transition: all 0.4s ease 0s;
26 }
27 </style>
28 </head>
29
30 <body>
31 <!--show oxstandhard university banner-->
32 <div>
33 
35
36 <div class = "login">
```

```
~\Desktop\project\student mark predictor - Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\HP\Desktop\project\student mark predictor\templates\index.html
index.html x app.py x
student mark
├── static
│   └── image
├── templates
│   ├── index.html
│   ├── app.py
│   ├── smp_data
│   └── student_n
30 <body>
31 <!--show oxstandhard university banner-->
32 <div>
33 
35
36 <div class = "login">
37 <!-- form get input to predict marks-->
38 <center>
39 <form action="{{ url_for('predict')}}"method="POST">
40 <h1> ****Enter Your Study Hours to Predict Marks**** </h1>
41 <input align="center" type="number" name="study_hours" placeholder="study hours" required= "required" wi
42 <br>
43
44
45 <br>
46
47 <!--show button-->
48 <div class="button_cont" align="center"> <a class="button_css" href="" target="_blank" rel= nofollow noop
49 <button type="submit" class="btn btn-primary btn-block btn-Large"><strong> predict marks </strong></but
50 </div>
51 </form>
52 </center>
53
54 <!--show predicted output using ML Model --->
55
56 <div>
57
58 <center>
59 <h2>{{ prediction_text}}</h2>
60 </center>
61 </div>
62 </div>
63 </body>
64 </html>
```



## Final Output:



```
1 # -*- coding: utf-8 -*-
2
3 import numpy as np
4 import pandas as pd
5 from flask import Flask, request, render_template
6 import joblib
7
8 app = Flask(__name__)
9
10 model = joblib.load("student_mark_predictor.pkl")
11 df = pd.DataFrame()
12
13 @app.route('/')
14 def home():
15     return render_template('index.html')
16
17 @app.route('/predict', methods=['POST'])
18 def predict():
19     global df
20
21     input_features = [int(x) for x in request.form.values()]
22     features_value = np.array(input_features)
23
24     #validate input hours
25     if input_features[0] < 0 or input_features[0] > 24:
26         return render_template('index.html', prediction_text='Please enter valid hours between 1 to 24')
27
28     output = model.predict([features_value])[0][0].round(2)
29
30     # input and predicted value store in df then save in csv file
31     df = pd.concat([df, pd.DataFrame({'Study Hours': input_features, 'Predicted Output': [output]})], ignore_index=True)
32     print(df)
33     df.to_csv('smp_data_from_app.csv')
```

Console I/A

```
127.0.0.1 - - [26/Jul/2021 14:14:05] "POST /predict HTTP/1.1" 200 -
Study Hours Predicted Output
0 3 62.25
1 4 66.19
2 5 70.13
3 6 74.06
4 7 78.00
5 8 81.93
6 9 85.87
7 10 89.80
8 11 93.74
```



\*\*\*Enter Your Study Hours to Predict Marks\*\*\*

You will get [74.06%] marks, when you do study [6] hours per day



## CHAPTER 6

### TESTING

To train any machine learning model irrespective what type of dataset is being used you have to split the dataset into training data and testing data. so here I have used the “train\_test\_split” to split the data in 80:20 ratio ie , 80% of the data will be used for training the model while 20% will be used for testing the model that is built out of it.

```
✓ [16] # split dataset
```

```
✓ [17] 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)
```

```
✓ [18] 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)
```

✓ 0s completed at 13:30

```
✓ [18] 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)
```

#### ▼ Select a model and train it

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

```
✓ [20] lr.fit(X_train, y_train)
```

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

```
✓ [21] lr.coef_
```

## CHAPTER 7

### CONCLUSION

**7.1 Limitation of the project :** This system will help student used to predict the student marks by their studying hours only not involved other characteristics of the students like extracurricular activities, number of collage absences etc.

**7.2 Benefits:** Analyzing and prediction of academic performance is important for any education institutions. Predicting student performance can help teachers to take steps in developing strategy for improving performance at early stages. This system helps to minimize the failure ratio and to take acceptable action for career. In future additional features are added to our dataset to acquire better accuracy.

**7.3 Future work:** This system will help student used to predict the student marks by their studying hours.

We can use similar data sets to determine student dropout's rates. Given a larger data set with more features could provides better insights we can integrate these predictions with university LTI modules, Such as Canvas to update regularly students about their progress at various stages during a course of a semester.

## REFERENCES

- 1) [www.jetir.org\(ISSN-2349-5162\)](http://www.jetir.org(ISSN-2349-5162))
- 2) <https://github.com/furkhan26/predicting>
- 3) [www.researchgate.net](http://www.researchgate.net)
- 4) [https://youtu.be/U\\_oJgcvc0eI](https://youtu.be/U_oJgcvc0eI)
- 5) <https://www.goeduhub.com/10513/student-mark-predictor-using-machine-learning>