DATA SCIENCE PROJECT DOCUMENTATION:

TOPIC:-

STUDENT PERFORMANCE PREDICTION:-

USING LINEAR REGRESSION MODEL.

SUBMITTED BY:-

Chetanya Agarwal (211141)

Ishita Mishra (211151)

SUBMITTED TO:-

Ms. Diksha Hooda

INDEX

ACKNOWLEDGEMENTS:

I extend my heartfelt gratitude to everyone who contributed to the success of this GPA Predictor project. This journey wouldn't have been possible without the support, guidance, and inspiration from various individuals and communities.

Firstly, a massive thank you to the scikit-learn development team for creating a powerful machine learning library that made implementing linear regression and model evaluation seamless. The extensive documentation and community support were invaluable.

Special appreciation goes to the pandas community for providing a robust and user-friendly data manipulation tool. Handling the dataset with ease greatly streamlined the preprocessing phase.

I want to express my gratitude to the developers behind NumPy for providing a fundamental library for numerical operations in Python. NumPy played a crucial role in manipulating and processing the numerical aspects of the dataset.

Heartfelt thanks to the Matplotlib team for creating a versatile plotting library. The captivating visualizations in this project owe much to Matplotlib, enhancing the understanding of the relationship between study hours and GPA.

I also want to acknowledge the educators and researchers whose work laid the foundation for the concepts applied in this project. Their contributions to the field of machine learning and data science have been instrumental in shaping the landscape.

Last but not least, I want to thank my peers and friends who provided encouragement and constructive feedback throughout the development of this project. Your insights and discussions were invaluable in refining the approach and enhancing the overall quality.

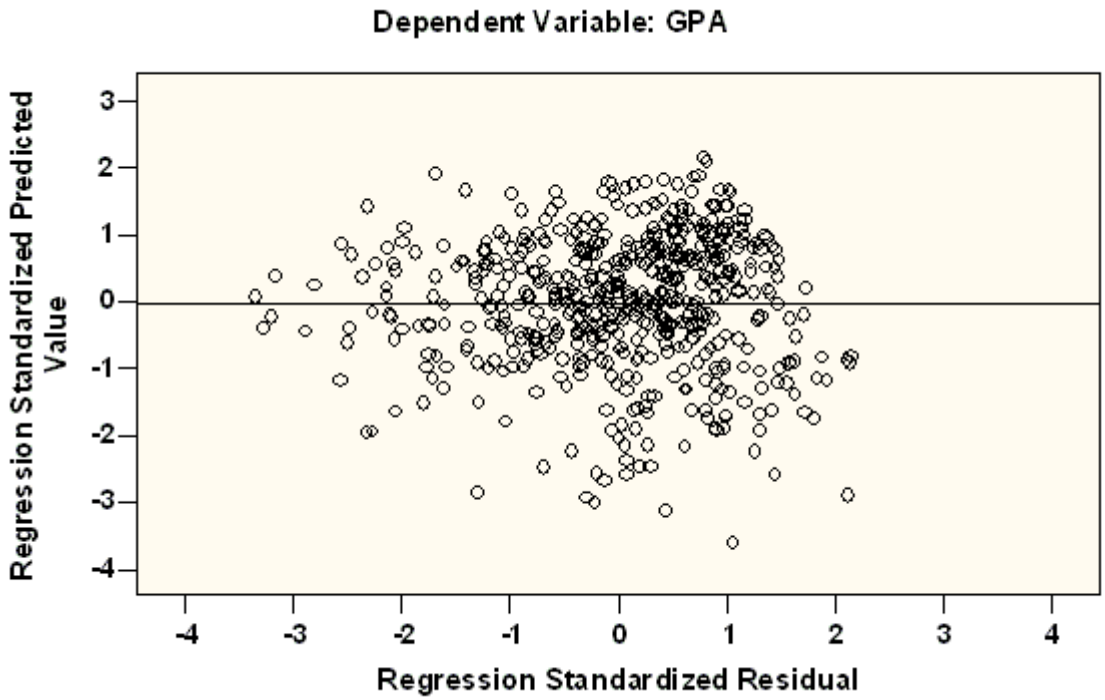
This project stands as a collaborative effort, and I'm grateful for the collective knowledge and support that fueled its success.

INTRODUCTION:

Welcome to the GPA Predictor - a data-driven exploration into the intriguing relationship between study hours and academic success. In the fast-paced world of academia, understanding the correlation between time invested in studying and resulting GPA can be a key factor in student success.

This project harnesses the power of machine learning, specifically linear regression, to unravel the patterns hidden within a dataset. By leveraging the capabilities of libraries such as scikit-learn, pandas, NumPy, and Matplotlib, we embark on a journey to predict GPA based on study hours.

As a computer science enthusiast and aspiring data scientist, I set out to answer a fundamental question: How strong is the connection between the hours dedicated to studying and the academic performance reflected in the GPA? The exploration involves loading and preprocessing data, training a linear regression model, and evaluating its predictive accuracy.



ABSTRACT:

The GPA Predictor project represents a foray into the dynamic intersection of data science and academic performance. In a landscape where time is a precious resource, understanding the relationship between study hours and GPA becomes paramount. This endeavour employs machine learning, specifically linear regression, to unravel the nuances hidden within a dataset.

The journey begins with the meticulous loading and preprocessing of data using the powerful tools provided by pandas and NumPy. Leveraging scikit-learn, a linear regression model is trained to discern patterns between study hours and GPA. Matplotlib steps onto the stage, translating data into compelling visualizations that illuminate the correlation.

Results indicate a compelling association between study time and academic achievement, paving the way for accurate GPA predictions.

LIBRARIES USED:

Several Python libraries were utilized to facilitate various tasks. Here's a list of the libraries used along with their roles in the project:

pandas:

Role: Pandas is a powerful data manipulation and analysis library. In this project, it is used for reading and handling the dataset, as well as organizing and preprocessing the data.

NumPy:

Role: NumPy is a fundamental library for numerical operations in Python. It is employed in this project for numerical manipulation and processing of the dataset.

scikit-learn:

Role: Scikit-learn is a machine learning library that provides various tools for data mining and data analysis. In this project, scikit-learn is used for

implementing the linear regression model, splitting the dataset into training and testing sets, and evaluating the model's performance.

matplotlib:

Role: Matplotlib is a plotting library in Python. It is used in this project to create visualizations, including scatter plots and regression lines, to help understand the relationship between study hours and GPA.

StandardScaler from scikit-learn:

Role: StandardScaler is a part of scikit-learn's preprocessing module. It is specifically used for standardizing the features by removing the mean and scaling to unit variance. In this project, it is applied to standardize the study hours before training the linear regression model.

These libraries collectively provide a robust foundation for data handling, analysis, visualization, and machine learning model implementation in the context of the GPA Predictor project.

SOURCE CODE:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt

# Load data from the CSV file
file_path = '/content/gpa_study_hours.csv'  # Replace with the actual path
df = pd.read_csv(file_path)

# Display the first few rows of the dataset
print(df.head())

# Separate features (X) and target variable (y)
X = df[['study_hours']]
```

```
y = df['gpa']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Standardize the features using StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Create a linear regression model
model = LinearRegression()

# Train the model
model.fit(X_train_scaled, y_train)

# Make predictions on the scaled testing set
y_pred = model.predict(X_test_scaled)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')

# Print the coefficients and intercept
coefficients = model.coef_
intercept = model.intercept_
print(f'Coefficients: {coefficients}')
print(f'Intercept: {intercept}')

# Plot the training graphs
plt.figure(figsize=(12, 6))

# Plot the scatter plot of training data
plt.subplot(1, 2, 1)
plt.scatter(X_train, y_train, color='blue', label='Training Data')
plt.xlabel('Study Hours')
plt.ylabel('GPA')
plt.title('Training Data')
```

```
# Plot the regression line on the training data
plt.subplot(1, 2, 2)
plt.scatter(X_train, y_train, color='blue', label='Training Data')
plt.plot(X_train, model.predict(X_train_scaled), color='red', linewidth=2,
label='Regression Line')
plt.xlabel('Study Hours')
plt.ylabel('GPA')
plt.title('Training Data with Regression Line')

plt.tight_layout()
plt.show()


# Testing model
# Assume you have a new student with 4 study hours
new_data = {'study_hours': [4]}
new_df = pd.DataFrame(new_data)

# Standardize the new data using the same scaler
new_data_scaled = scaler.transform(new_df)

# Make predictions on the new data
predicted_gpa = model.predict(new_data_scaled)

print(f'Predicted GPA for 4 study hours: {predicted_gpa[0]}')
```

CODE STRUCTURE:

Import Libraries:
- Import necessary Python libraries for data manipulation, analysis, visualization, and machine learning.

Load Data:
- Read the dataset from a CSV file into a panda DataFrame.

Data Exploration:
Display the first few rows of the dataset to get an overview.

Data Preparation:
- Separate the features (X) and target variable (y).

- Split the dataset into training and testing sets.

Data Standardization:
- Use StandardScaler to standardize the features, ensuring they have a mean of 0 and a standard deviation of 1.

Model Creation and Training:
- Create a linear regression model using scikit-learn.
- Train the model using the standardized training data.

Model Evaluation:
- Make predictions on the scaled testing set.
- Evaluate the model's performance using mean squared error.

Visualization:
- Plot training graphs, including scatter plots of training data and the regression line.

New Data Prediction:
- Assume new data (4 study hours) and standardize it.
- Make predictions on the new data and print the predicted GPA.

This structured approach ensures clarity in data processing, model training, evaluation, and visualization within the GPA Predictor project.

OUTPUT:

```
    gpa  study_hours
0  4.00         10.0
1  3.80         25.0
2  3.93         45.0
3  3.40         10.0
4  3.20          4.0
Mean Squared Error: 0.06592396536461444
Coefficients: [0.04082264]
Intercept: 3.5830194805194813
```


Training Data
GPA
Study Hours
4.25
4.00
3.75
3.50
3.25
3.00
2.75
0
10
20
30
40
50
60
70


Training Data with Regression Line
GPA
Study Hours
4.25
4.00
3.75
3.50
3.25
3.00
2.75
0
10
20
30
40
50
60
70

SOFTWARE REQUIREMENTS SPECIFICATION (SRS):

The GPA Predictor project requires the following software:

Python 3.x: The code is written in Python, and the project assumes the availability of Python 3.x.

scikit-learn, pandas, NumPy, matplotlib: These Python libraries are essential for data manipulation, numerical operations, machine learning, and data visualization.

COMPILER AND IDE:

The project was developed using the following compiler and integrated development environment (IDE):

Compiler: Not applicable (Python is an interpreted language).

IDE: The code was developed using the online Google Colab Platform.

LOW-LEVEL DESIGN (LLD):

1.Data Loading and Exploration:

Submodule 1: Load Data
- Responsible for reading the dataset from a CSV file using pandas.

Submodule 2: Display Dataset
- Displays the first few rows of the dataset to provide an overview.

2. Data Preparation:

Submodule 1: Feature Separation
- Separates the dataset into features (study_hours) and the target variable (gpa).

Submodule 2: Data Splitting
- Splits the data into training and testing sets using scikit-learn's train_test_split.

3. Data Standardization:

Submodule 1: Standardize Features
- Applies StandardScaler from scikit-learn to standardize the study_hours feature.

4. Linear Regression Modeling:

Submodule 1: Model Creation

- Creates a linear regression model using scikit-learn's LinearRegression.

Submodule 2: Model Training

- Trains the model using the standardized training data.

5. Model Evaluation:

Submodule 1: Predictions

- Generates predictions on the scaled testing set.

Submodule 2: Mean Squared Error

- Computes the mean squared error to evaluate the model's performance.

6. Visualization:

Submodule 1: Training Graphs

- Creates a figure with two subplots: scatter plot of training data and a subplot with the regression line.

7. New Data Prediction:

Submodule 1: New Data Preparation

- Assumes new data (4 study hours) and creates a DataFrame.

Submodule 2: Standardize New Data

- Standardizes the new data using the same scaler.

Submodule 3: Prediction

- Uses the trained model to predict GPA for the new data.

CONCLUSION:

The GPA Predictor project provides insights into the relationship between study hours and academic performance. Through the utilization of machine learning and data visualization, we gain a deeper understanding of the factors influencing GPA. This project stands as a testament to the power of data science in unraveling educational insights.