A

MACHINE LEARNING PROJECT REPORT

ON

Student Grade Prediction

In fulfillment of

B.Tech 3rd yr (Computer Science & Engg.)

Submitted To:

Ms. Jyoti Vyas

(Assistant Prof).

Submitted By:

Rahul Kumar

(CSE, 3rd yr)

Acknowledgment

I would like to express my gratitude to **Ms. Jyoti Vyas** for providing me with the opportunity to work on this project and for their continuous support and guidance. I would also like to thank **Ms. Mamta Garg**, Head of the Computer Science and Engineering Department at the Jodhpur Institute of Engineering and Technology, for their encouragement and moral support. Additionally, I am grateful to all the faculty members of the department, my parents, and God for their consistent blessings.

Table of Contents

1. Introduction

The goal of this project is to predict student grades using machine learning techniques. By analyzing various student attributes such as attendance, previous grades, class participation, and assignment scores, the system can estimate the performance of students in upcoming exams or overall course grades. This project uses Python and several key libraries for data manipulation, visualization, and model building.

2. Technology Used

- **Programming Language**: Python
- **Libraries**:
  - **NumPy**: For numerical operations
  - **Pandas**: For data manipulation and preprocessing
  - **Matplotlib & Seaborn**: For visualizing data trends and patterns
  - **TensorFlow & Keras**: For building and training deep learning models
  - **Scikit-learn**: For data splitting, model selection, and evaluation
- **Tools**: VS Code, Jupyter Notebooks for interactive development

3. Project Details

1. Flow Chart

Start
↓
Load Data (Student Performance Data)
↓
Preprocess Data (Handling missing values, normalizing data)
↓
Split Data into Training and Testing Sets
↓
Select Features (Attendance, test scores, class participation)
↓
Train Machine Learning Model using Training Data
↓
Evaluate Model on Testing Data (Metrics: Accuracy, MSE)
↓
Predict Grades for New Students or Upcoming Exams
↓
End

2. Functions/Modules

- **Data Loading and Preprocessing**: Load and preprocess the student performance dataset using Pandas, including feature engineering and handling missing data.
- **Model Architecture**:
  - **Feature Selection**: Selecting features like attendance, test scores, and participation.
  - **Model Types**:
    - **Linear Regression** for predicting final grades.
    - **Random Forest** for handling complex data relationships.
    - **Neural Networks** using **TensorFlow/Keras** for improved performance on larger datasets.
- **Training and Evaluation**: Train models using mean squared error or classification accuracy, and evaluate model performance on the test set.
- **Dimensionality Reduction**: Use **PCA** and **t-SNE** for visualizing the relationships between students in terms of their performance and grade predictions.
- **Grade Prediction Generation**: Predict final grades for students based on current features, providing insights for further improvement.

3. Project Code

The core code for the project includes the following steps:

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split


df = pd.read_csv('student.csv')


print('Total number of students: ', len(df))
print("Parameters are: ", df.columns)
df.info()


df = df.dropna()
```

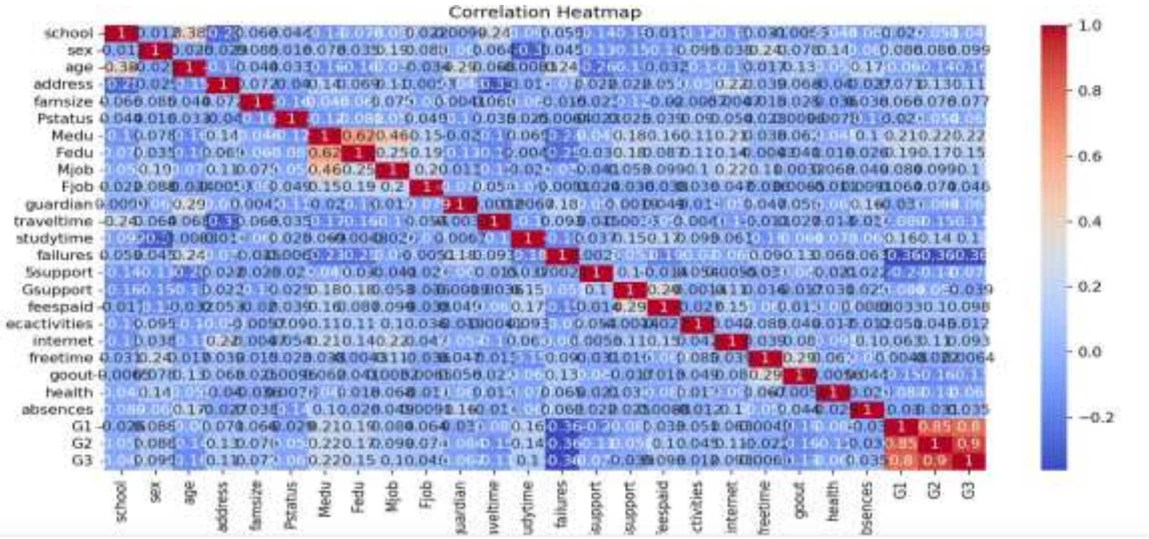
```
label_columns = df.select_dtypes(include=['object']).columns
le = LabelEncoder()

for col in label_columns:
    df[col] = le.fit_transform(df[col])

print("After encoding:")
print(df.dtypes)

plt.figure(figsize=(10, 8))
correlation_matrix = df.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()


sns.countplot(x="age", data=df)
plt.show()


most_correlated = df.corr().abs()['G3'].sort_values(ascending=False)
most_correlated = most_correlated[:9]
print("Most correlated features: \n", most_correlated)


df = df.loc[:, most_correlated.index]
df.head()


X_train, X_test, y_train, y_test = train_test_split(df.drop('G3', axis=1), df['G3'],
test_size=0.3, random_state=0)


lr = LinearRegression()
model = lr.fit(X_train, y_train)

print("Model Score (R^2):", lr.score(X_test, y_test))

predictions = lr.predict(X_test)


plt.scatter(y_test, predictions)
m, b = np.polyfit(y_test, predictions, 1)
```

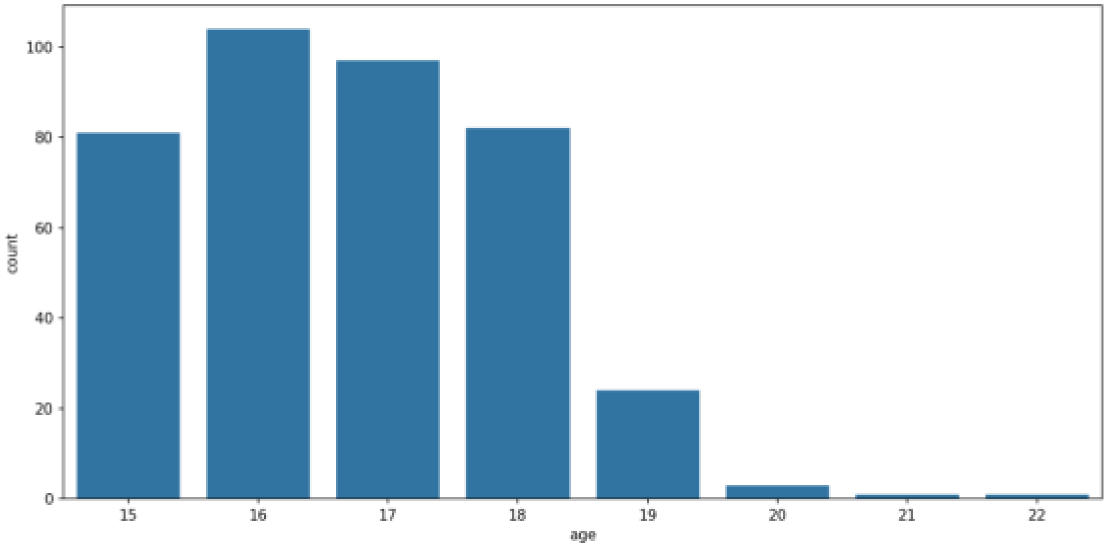
```
plt.plot(y_test, m * y_test + b, color='red')
plt.xlabel("Actual Grade")
plt.ylabel("Predicted Grade")
plt.title("Actual vs Predicted Grades")
plt.show()


user_input = {}
for col in X_train.columns:
    user_input[col] = float(input(f"Enter value for {col}: "))
user_input_df = pd.DataFrame([user_input])


predicted_grade = lr.predict(user_input_df)
print("Predicted Grade:", predicted_grade[0])


df.head()
```

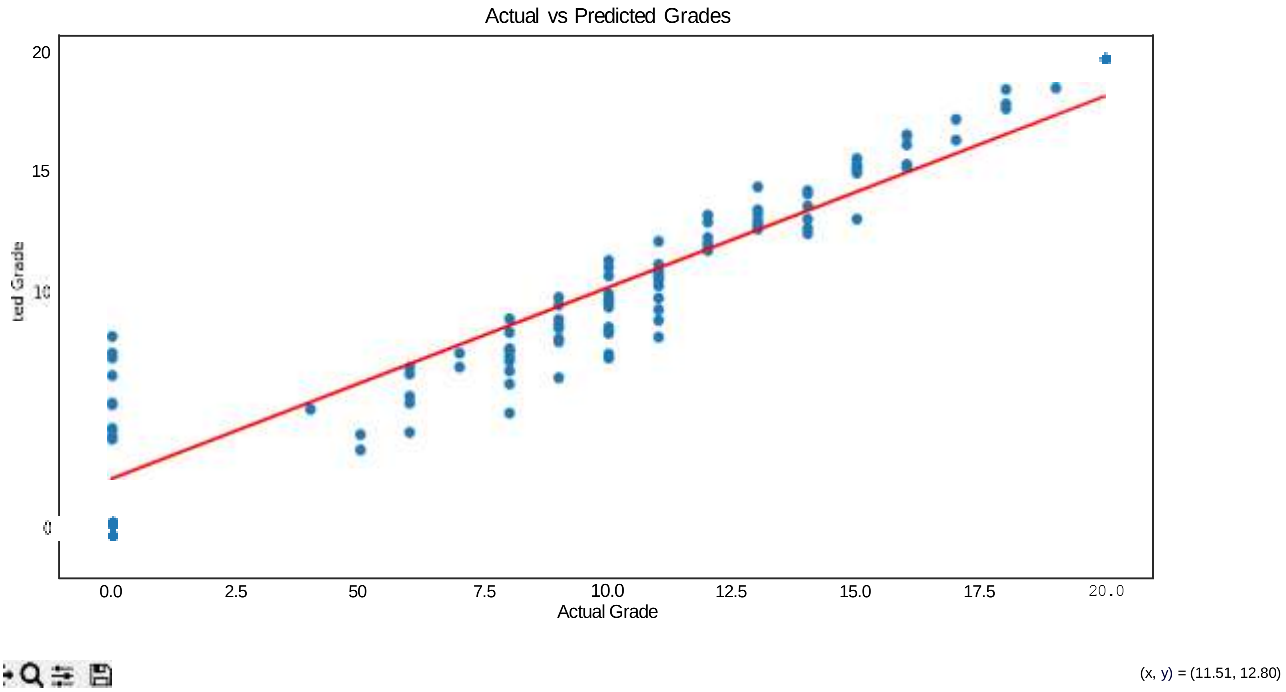
4. Project Screenshots

Include screenshots of the following:

1. Dataset head and shape






Actual vs Predicted Grades
ted Grade
20
15
10
0
0.0
2.5
50
7.5
10.0
12.5
15.0
17.5
20.0
Actual Grade

(x, y) = (11.51, 12.80)

4. Applications

The Book Recommendation System can be used in:

- **Educational Institutions**: To help teachers and administrators identify students who may need additional help or intervention.

- **E-learning Platforms**: Predicting student performance and offering personalized resources to improve grades.

- **Student Advising**: Guiding students on which areas to focus on for better performance..

5. Conclusion and Future Work

This project demonstrates how machine learning models can accurately predict student grades. The models show promising results, and future improvements can include incorporating more features such as behavioral data or advanced deep learning architectures.

6. References

- **Pandas Documentation**: https://pandas.pydata.org/
- **TensorFlow/Keras Documentation**: https://www.tensorflow.org/
- **Scikit-learn Documentation**: https://scikit-learn.org/