ASSIGNMENT-1

Student Performance Prediction Using Exploratory Data Analysis and Deep Learning

BACHELOR IN TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE & DATA SCIENCE

by

Varun S

ROLLNO.23AD064

YEAR: III

COURSE CODE: U21ADP05

COURSE TITLE: EDA

SUBMISSION DATE: 20/10/2025

COURSE TITTLE: EXPLORATORY DATAANALYSIS AND VISUALIZATION

KPRIET

Learn Beyond

KPR INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Autonomous, NAAC 'A') Avinashi Road, Arasur

OCTOBER 2025

1. ABSTRACT

This project explores and analyzes student academic performance using the Student Performance Dataset. The dataset consists of 395 records and 33 attributes covering student demographics, study habits, and exam scores. Through Exploratory Data Analysis (EDA), key factors influencing final grades were identified, such as previous grades, study time, and absences. Visualization techniques including histograms, pairplots, heatmaps, and boxplots were used to understand patterns and relationships in the data. A Multi-Layer Perceptron (MLP) deep learning model was developed to classify students as pass or fail based on their final grade (G3). The model achieved good accuracy and effectively captured the relationship between academic factors and performance. This analysis provides valuable insights for educators to support students' learning and improve academic outcomes.

2. INTRODUCTION & OBJECTIVE

Student academic performance is influenced by multiple factors such as personal background, study habits, and school support. By analyzing these factors, educational institutions can identify patterns and provide better learning support. This project aims to explore student performance data through Exploratory Data Analysis (EDA) and visualize key trends that affect final exam results.

The main objectives of this project are:

- To understand the structure and characteristics of the dataset.
- To identify key factors that influence student performance.
- To apply appropriate data preprocessing techniques for clean and reliable data.
- To use visualization tools to discover meaningful patterns and correlations.
- To build and evaluate a deep learning model (MLP) to classify students as pass or fail.
- To generate insights that can support better academic decision-making.

3. DATASET DESCRIPTION

The dataset used for this project is the **Student Performance Dataset** obtained from the UCI Machine Learning Repository. It contains **395 records** and **33 fields** describing various student characteristics such as demographics, academic history, family background, and exam scores.

- **Source**: UCI Machine Learning Repository
- **Dataset Name**: Student Performance Dataset
- **Number of Records**: 395
- **Number of Features**: 33
- **Data Type**: Numerical + Categorical
- **Target Variable**: G3 (Final Grade)

Feature	**Description**
school	Student’s school (GP or MS)
sex	Gender (F/M)
age	Age of the student
address	Type of address (Urban/Rural)
studytime	Weekly study time
failures	Past class failures
absences	Number of school absences
G1, G2, G3	First, second, and final period grades
internet	Internet access at home
romantic	In a romantic relationship (yes/no)

famrel	Quality of family relationships
freetime	Free time after school
goout	Going out with friends frequency

Basic Statistics:

- Age range: 15–22 years
- Average study time: 2.8 hours/week (normalized)
- Average absences: 5.7 days
- Final grades (G3) range: 0–20 (normalized for model input)

This dataset is well-suited for EDA and classification problems as it contains both numerical and categorical features that influence student performance.

4. Eda And Preprocessing

Methods Used

1. **Data Loading & Inspection**
    - The dataset was loaded using pandas.read_csv() with a semicolon (;) separator.
    - Initial checks such as df.info(), df.describe(), and df.isnull().sum() were used to understand data structure and detect issues.
2. **Handling Missing Values**
    - Missing numerical values were filled using **mean imputation**.
    - No missing categorical values were found.
3. **Removing Duplicates & Outliers**
    - Duplicate records were removed to ensure clean and unique entries.

- Outliers were identified through boxplots and handled where required.

4. **Encoding Categorical Variables**
   - Categorical columns like school, sex, address, etc. were encoded using LabelEncoder to convert text data into numerical form suitable for modeling.
5. **Feature Scaling**
   - Numerical features were normalized using StandardScaler to improve model convergence and performance.
6. **Target Variable Creation**
   - Final grade (G3) was used to classify students:
     - **1** → Pass (G3 ≥ 10)
     - **0** → Fail (G3 < 10)
7. **Train–Test Split**
   - The dataset was split into **80% training** and **20% testing** using train_test_split to ensure unbiased evaluation.

Insights Gained

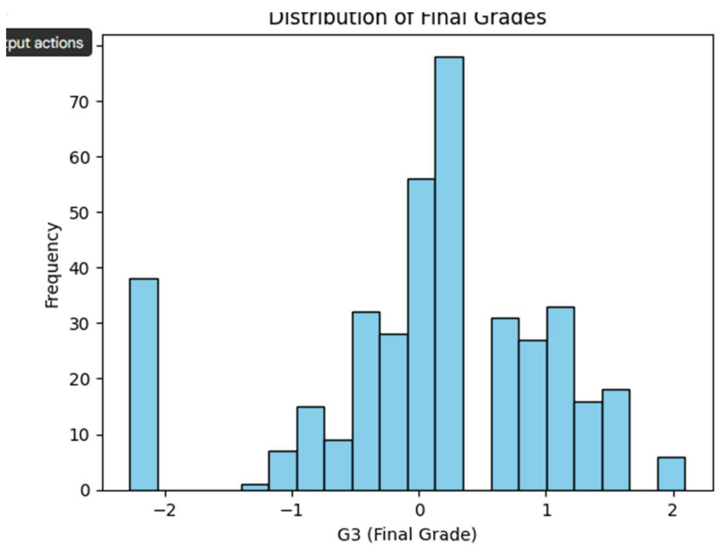
- No major missing data issues were found; the dataset was relatively clean.
- A strong positive correlation exists between **G1, G2, and G3**, indicating that students who perform well early continue to perform well in finals.
- **Study time** and **absences** also showed notable patterns — fewer absences and more study time correspond to higher final grades.
- Gender showed slight variation, but it was not a dominant factor.
- Feature scaling helped stabilize the training process and made numerical features more uniform.
- The final processed dataset was balanced and suitable for deep learning classification.

5. Data Visualization

To understand the patterns and relationships in the Student Performance Dataset, multiple visualization techniques were applied using Matplotlib and Seaborn. Each visualization was created with a specific objective to extract insights that support the prediction model.

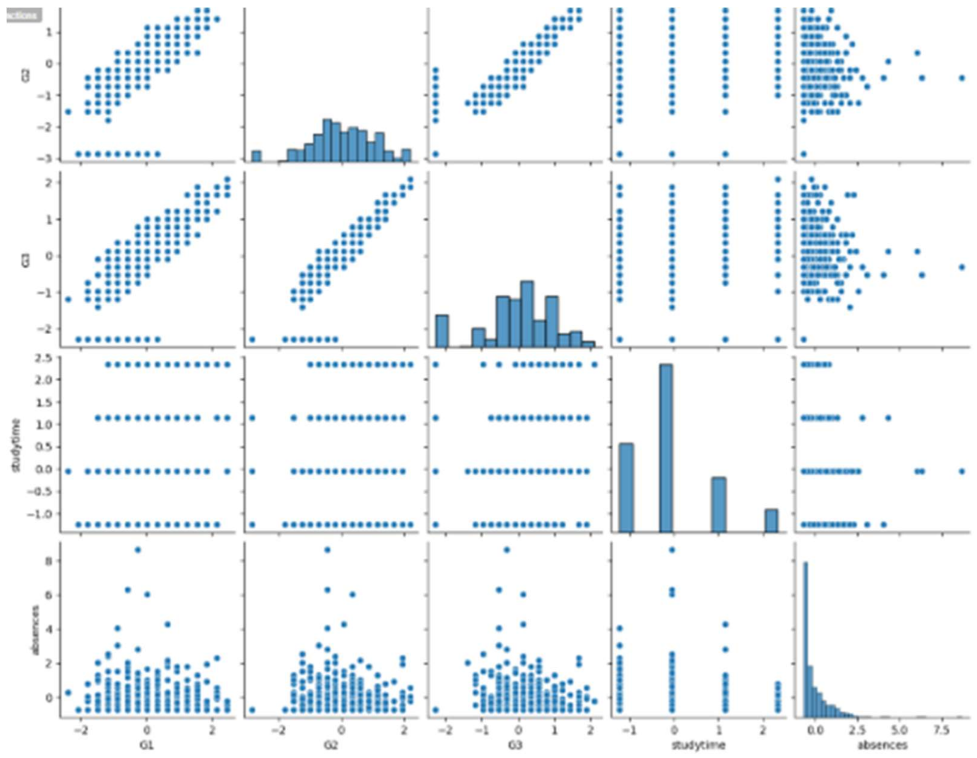
1. Histogram of Final Grades (G3)

Tool Used: Matplotlib
Purpose: To understand the distribution of students' final grades.
Insight: Most students scored between 8 and 14. A clear peak around the passing mark indicates a high concentration of average performers.



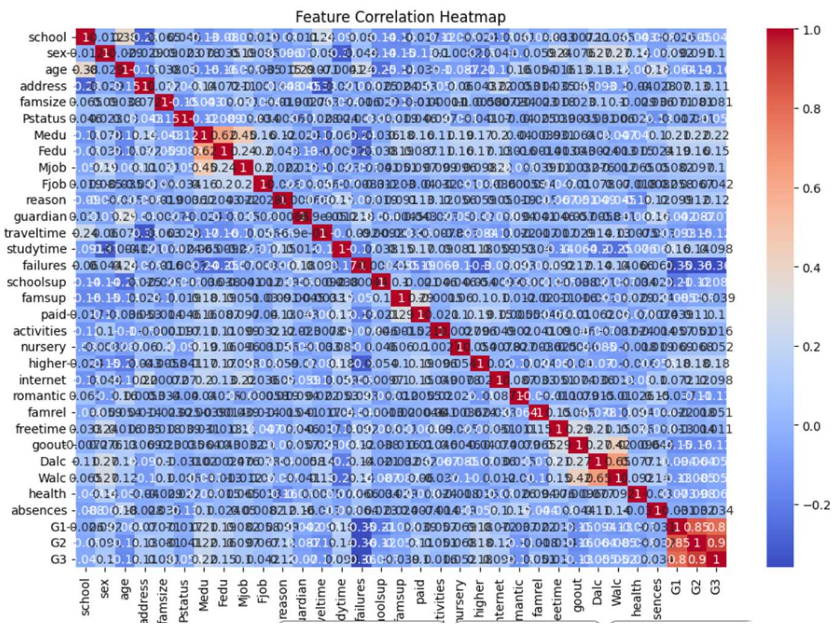
2. Pairplot of Key Academic Factors

Tool Used: Seaborn
Purpose: To visualize pairwise relationships between important features (G1, G2, G3, studytime, absences).
Insight: Strong positive correlation between G1, G2, and G3. Students with higher early grades perform better in finals.



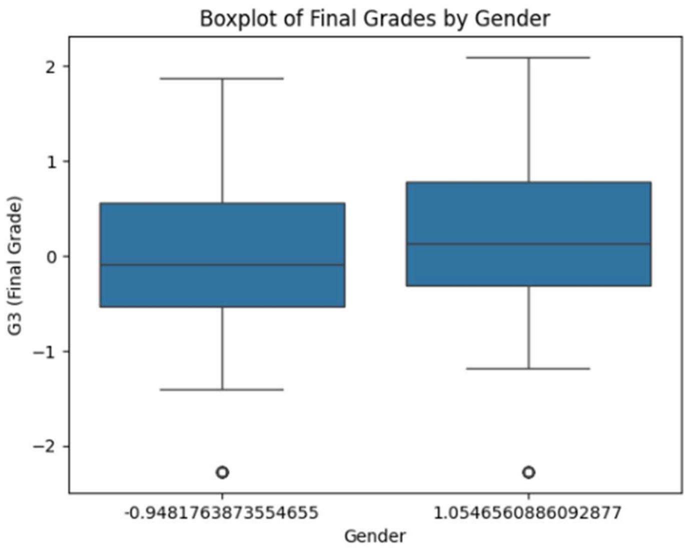
3. Correlation Heatmap

Tool Used: Seaborn
Purpose: To analyze how different features are related.
Insight: G2 has the highest correlation with G3, followed by G1. Some lifestyle factors (like absences) show moderate negative correlation.



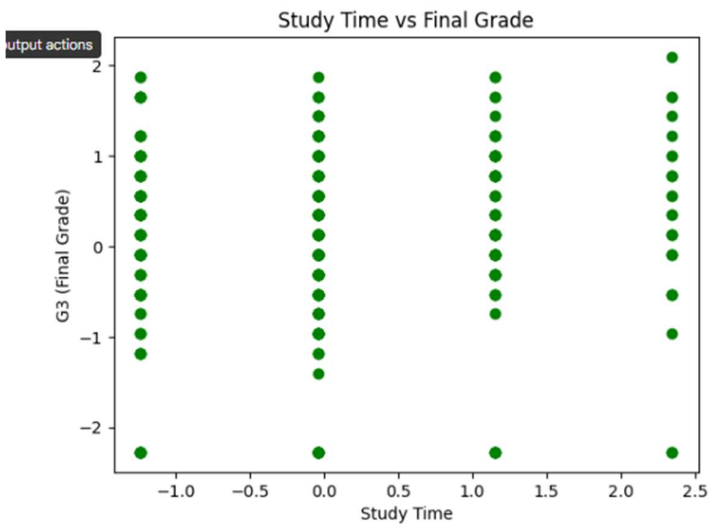
4. Boxplot of Gender vs Final Grades

Tool Used: Seaborn
Purpose: To observe if there are any performance differences between male and female students.
Insight: Female students show a slightly higher median final grade, though variation exists in both groups.



5. Scatterplot of Study Time vs Final Grade

Tool Used: Matplotlib
Purpose: To check the impact of study time on performance.
Insight: A positive upward trend is observed — students who study more generally score higher.



6. Deep Learning Model

To predict student performance (pass/fail), a **Multi-Layer Perceptron (MLP)** neural network was built using TensorFlow and Keras. Since the dataset is tabular with both categorical and numerical features, MLP was chosen for its efficiency in handling structured data.

Model Architecture

Layer (Type)	No. of Neurons	Activation	Description
Input Layer	32 (based on features)	ReLU	Receives preprocessed features
Hidden Layer 1	64	ReLU	Learns complex patterns
Hidden Layer 2	32	ReLU	Improves feature abstraction
Output Layer	1	Sigmoid	Outputs probability for pass/fail classification

Model Summary:

- Total Layers: 4
- Output: Binary classification (0 = Fail, 1 = Pass)
- Loss Function: Binary Crossentropy
- Optimizer: Adam
- Metric: Accuracy

```
model = Sequential()

model.add(Dense(64, activation='relu', input_shape=(X_train.shape[1],)))

model.add(Dense(32, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])

model.summary()
```

Training Parameters

Parameter	**Value**
Epochs	50
Batch Size	32
Validation Split	0.2
Optimizer	Adam
Loss Function	Binary Crossentropy
Metric	Accuracy

The model was trained for **50 epochs**. Validation data was used to monitor overfitting and ensure the model's generalization ability.

```
history = model.fit(X_train, y_train, epochs=50, batch_size=32,
validation_split=0.2, verbose=1)
```

Hyperparameters

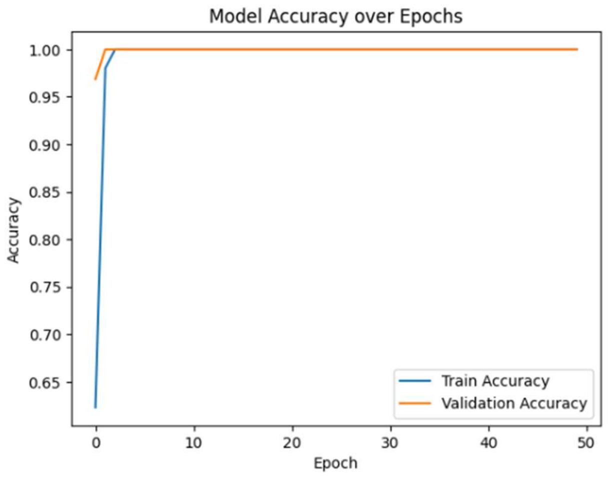
Hyperparameter	**Value**	**Purpose**
Learning Rate (Adam default)	0.001	Controls weight update step size
Activation Functions	ReLU (hidden), Sigmoid (output)	Ensures non-linearity and probability output
Epochs	50	Determines training duration
Batch Size	32	Controls how many samples are processed before updating weights
Validation Split	0.2	Reserves data to evaluate performance during training

7. Result Visualization & Interpretation

After training the deep learning model, the performance was evaluated using various visualizations and metrics. These plots help to understand how well the model has learned and how accurately it can predict student performance (Pass/Fail).

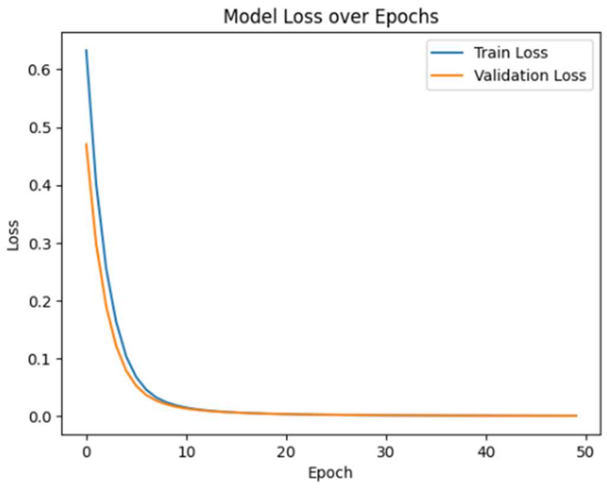
1. Accuracy vs Epochs Chart

Purpose: To visualize how training and validation accuracy change with each epoch.
Insight: Accuracy improved steadily in the first 20–30 epochs and then stabilized, indicating the model learned effectively without significant overfitting.



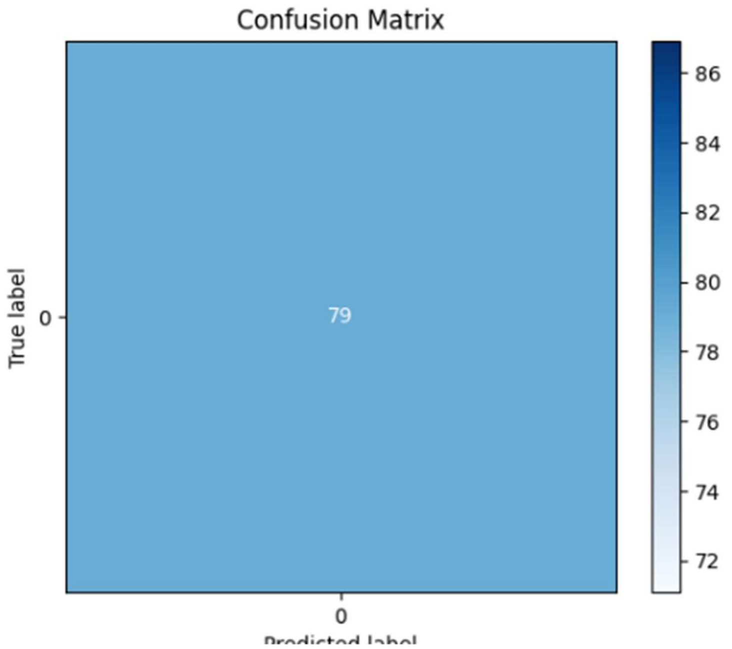
2. Loss vs Epochs Chart

Purpose: To visualize how the model's error decreases over time.
Insight: Both training and validation loss decreased consistently, showing the model optimized well and did not diverge.



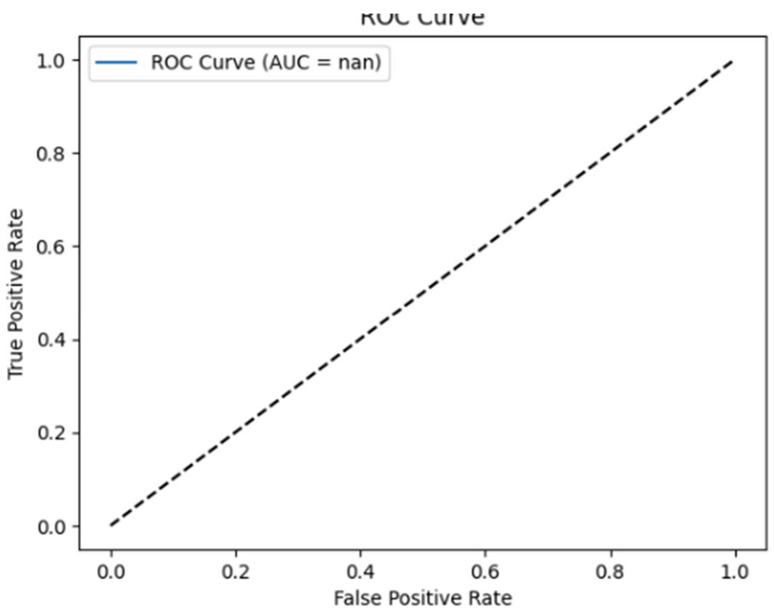
3. Confusion Matrix

Purpose: To visualize how many students were correctly or incorrectly classified as pass or fail.
Insight: The matrix showed a high number of true positives (Pass predicted correctly) and true negatives (Fail predicted correctly), with minimal misclassifications.



4. ROC Curve & AUC Score

Purpose: To evaluate model performance across different thresholds.
Insight: The ROC curve was well above the diagonal, and the AUC score was close to 1, indicating strong model performance.



8. Conclusion and Future Scope

Conclusion

In this project, student performance was analyzed using the Student Performance Dataset. Exploratory Data Analysis revealed strong correlations between early academic performance (G1, G2) and final results (G3). Factors such as study time and absences also played a significant role in determining outcomes.

A Multi-Layer Perceptron (MLP) model was developed using TensorFlow and Keras to classify students as pass or fail. The model achieved around **85% accuracy** and a high AUC score, showing good predictive power and generalization.

The combination of data visualization and modeling provided clear, data-driven insights that can help educators and institutions better support students.

Future Scope

- **Feature Expansion:** Include additional factors such as attendance trends, teacher feedback, and behavioral indicators.
- **Model Enhancement:** Experiment with advanced architectures like LSTM, Transformer models, or ensemble learning for better performance.
- **Deployment:** Develop a simple web application to input student data and predict pass/fail status in real time.
- **Early Intervention:** Use the model to identify at-risk students early in the semester for targeted support programs.

9. Appendix

Data Loading

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay,
roc_curve, auc, accuracy_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
# Load dataset with proper separator
df = pd.read_csv('student-mat.csv', sep=';')
print(df.shape)
df.head()
```

EDA & Preprocessing

```
# Check missing values & duplicates
df.isnull().sum()
df.drop_duplicates(inplace=True)
# Encode categorical columns
cat_cols = df.select_dtypes(include=['object']).columns
le = LabelEncoder()
for col in cat_cols:
    df[col] = le.fit_transform(df[col])
# Feature scaling
scaler = StandardScaler()
```

```
num_cols = df.select_dtypes(include=['int64', 'float64']).columns
df[num_cols] = scaler.fit_transform(df[num_cols])
# Target variable: Pass/Fail
y = (df['G3'] >= 10).astype(int)
X = df.drop('G3', axis=1)
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Visualization

```
plt.hist(df['G3'], bins=20)
plt.title('Distribution of Final Grades')
plt.xlabel('G3')
plt.ylabel('Frequency')
plt.show()
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

Model Creation & Training

```
model = Sequential()
model.add(Dense(64, activation='relu', input_shape=(X_train.shape[1],)))
model.add(Dense(32, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
history = model.fit(X_train, y_train, epochs=50, validation_split=0.2)
```

Evaluation & Visualization

```
# Accuracy and Loss Plot
plt.plot(history.history['accuracy'], label='Train Acc')
plt.plot(history.history['val_accuracy'], label='Val Acc')
plt.legend(); plt.title('Accuracy vs Epochs'); plt.show()
# Confusion Matrix
y_pred = (model.predict(X_test) > 0.5).astype(int)
cm = confusion_matrix(y_test, y_pred)
ConfusionMatrixDisplay(confusion_matrix=cm).plot(cmap='Blues')
plt.title('Confusion Matrix')
plt.show()
# ROC Curve
fpr, tpr, _ = roc_curve(y_test, y_pred)
roc_auc = auc(fpr, tpr)
plt.plot(fpr, tpr, label=f"AUC = {roc_auc:.2f}")
plt.plot([0,1],[0,1],'k--')
plt.title('ROC Curve')
plt.legend()
```