Documentation: Student Exam Score Prediction using Linear Regression

Overview

This documentation provides a detailed explanation of how to predict student exam scores based on various personal and academic factors using Linear Regression. It covers data preprocessing, model training, evaluation, and visualization steps, followed by an analysis of feature importance.

Objective

The goal is to predict the exam scores of students based on their study habits, lifestyle choices, and demographic information. We use a Linear Regression model to achieve this.

Steps and Code Explanation

1. Importing Necessary Libraries

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

Purpose: Importing required libraries.

- **pandas**: For data manipulation and analysis.
- **matplotlib.pyplot**: For data visualization (graphs and plots).
- **train_test_split**: To split the dataset into training and testing sets.
- **LinearRegression**: For building and training the Linear Regression model.
- **mean_squared_error and r2_score**: For evaluating the model's performance.

2. Loading and Preprocessing the Data

```
# Load the dataset
df = pd.read_csv(r"C:\Users\nteny\Desktop\DATASETS\student_habits_performance.csv")
```

Purpose: Loads the dataset from a CSV file located at the specified path into a pandas DataFrame (df).

```
# Generate dummy variables for categorical columns
df = pd.get_dummies(
  df,
  columns=["gender", "part_time_job", "diet_quality",
       "parental_education_level", "internet_quality",
       "extracurricular_participation"],
  drop_first=True
)
```

Purpose: Converts categorical variables into dummy variables (binary columns).

- **pd.get_dummies()**: This function encodes categorical columns into numerical binary columns.
- **drop_first=True**: Prevents the issue of multicollinearity by dropping one category of each feature.

```
# Convert dummy columns to integers
dummy_columns = df.columns[df.columns.str.contains('|'.join([
  'gender_', 'part_time_job_', 'diet_quality_',
  'parental_education_level_', 'internet_quality_',
  'extracurricular_participation_'
]))]
```

```
df[dummy_columns] = df[dummy_columns].astype(int)
```

Purpose: Converts all the newly created dummy columns into integers for modeling purposes.

```
# Drop the student_id column
df.drop(columns=["student_id"], inplace=True)
```

Purpose: Removes the student_id column, which is not needed for prediction, from the DataFrame

```
# Define the target variable and features
y = df["exam_score"]
x = df.drop(columns="exam_score")

# 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)
```

Purpose: Splits the dataset into features (x) and target (y) variables, then divides the data into training and testing sets.

- **train_test_split()**: Divides the data into 80% for training and 20% for testing. The random state ensures reproducibility.

```
# Instantiate and train the model
model = LinearRegression()
model.fit(x_train, y_train)
```

Purpose: Creates an instance of the Linear Regression model and fits it to the training data.

```
# Make predictions on the test set
y_pred = model.predict(x_test)

# Calculate performance metrics
mse = mean_squared_error(y_test, y_pred)
re = r2_score(y_test, y_pred)

# Print the results
print(f"mse: {mse: 2f}, re: {re: 2f}")
```

Purpose: Evaluates the model using the Mean Squared Error (MSE) and R^2 Score metrics.

- **MSE**: Measures the average squared difference between the actual and predicted values.
- **R^2 Score**: Indicates how well the model explains the variance in the target variable.

```
# Get feature importance based on the coefficients
feature_importance = pd.DataFrame({
   "Feature": x.columns,
   "Coefficient": model.coef_
})


# Sort features by coefficient value
feature_importance = feature_importance.sort_values(by="Coefficient", ascending=False)


# Display the feature importance table
print(feature_importance)
```

Purpose: Retrieves the coefficients from the Linear Regression model to understand the importance of each feature.

- **Coefficient Values**: The magnitude of the coefficients indicates the impact of each feature on the target variable. Positive coefficients indicate a positive relationship, and negative coefficients indicate an inverse relationship.
- **Sorting**: Sorts the features by their coefficient values to identify the most important ones.

```
# Scatter plot of actual vs predicted exam scores
plt.figure(figsize=(18, 8))
plt.scatter(y_test, y_pred, color="blue", alpha=0.6)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], color="red", lw=2)  # Ideal line
plt.xlabel("Actual Exam Score")
plt.ylabel("Predicted Exam Score")
plt.title("Actual vs Predicted Exam Scores")
plt.grid(True)
plt.show()
```

Purpose: Visualizes the relationship between actual and predicted exam scores.

- **Scatter Plot**: Plots the actual exam scores (y_test) against the predicted values (y_pred).
- **Ideal Line**: A red line is drawn to represent the ideal scenario where the predicted scores match the actual scores perfectly.
- **Grid**: A grid is added for better readability of the plot.

Results

Model Performance

The model was evaluated using the following metrics:

- **Mean Squared Error (MSE)**: 26.464502
- **R^2 Score**: 0.896796

These results suggest that the model performs well, explaining approximately 89.68% of the variance in the exam scores.

Feature Importance

The table below shows the features sorted by their importance (based on the magnitude of their coefficients):

Conclusion

The model reveals that certain study habits, such as the number of hours spent studying, sleep, and mental health rating, have a positive impact on exam scores. However, excessive time spent on Netflix and social media negatively impacts performance. The R^2 score of 0.896 indicates that the model explains a substantial portion of the variability in exam scores.