DEPT: SOFTWARE ENGINEERING

FUNDAMENTAL OF MACHINE LEARNING

NAME: BIRUK ASHAGRE

ID: DBU1401048

SUBMISSION DATE: 02/05/2017

Student Performance Prediction

Objective:

This project explores a machine learning problem of predicting student performance based on various features such as study time, absences, and previous grades. The goal is to guide you through the entire machine learning lifecycle, from data sourcing and preprocessing to model deployment. This will solidify your understanding of machine learning principles and provide practical experience in solving real-world problems using regression techniques.

Problem Definition & Data Acquisition

Problem Definition:

The problem is to predict the final grade (`G3`) of students based on features such as study time, absences, and previous grades (`G1` and `G2`). This is a regression problem since the target variable (`G3`) is continuous.

Data Source:

- **Dataset Name:** `student-por.csv`
- **Source:** The dataset is sourced from the UCI Machine Learning Repository, a well-known repository for machine learning datasets. Link is https://archive.ics.uci.edu/dataset/320/student+performance
- **License/Terms of Use:** The dataset is publicly available for educational and research purposes. It is free to use, modify, and distribute.
- **Data Structure:** The dataset is in CSV format and contains 33 columns (features) and 649 rows (students). Each row represents a student, and the columns include features such as `studytime`, `absences`, `G1`, `G2`, and `G3`.
- **Features**:
  - studytime : Weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, 4 - >10 hours).
  - absences: Number of school absences (numeric: from 0 to 93).
  - G1 : First period grade (numeric: from 0 to 20).
  - G2 : Second period grade (numeric: from 0 to 20).
  - G3 : Final grade (numeric: from 0 to 20, target variable).

Data Understanding & Exploration

Steps:
1. Loading and Displaying the Dataset:
   - The dataset is loaded using `pandas` and displayed using `df.head()` to show the first five rows.
   - `df.info()` is used to understand the structure of the dataset, including column names, data types, and missing values.

2. Summary Statistics:
   - Summary statistics for numerical features are generated using `df.describe()`.
   - Summary statistics for categorical features are generated using `df.describe(include=['object'])`.

3. Missing Values:
   - Missing values are checked using `df.isnull().sum()`. The dataset has no missing values.

Perform exploratory data analysis (EDA)

- Distribution of Final Grades (`G3`): A histogram is plotted to visualize the distribution of final grades.
- Correlation Heatmap: A heatmap is created to analyze the correlation between numerical features.
- Outlier Detection: Boxplots are used to identify outliers in numerical features.
- Scatter Plots: Scatter plots are used to visualize relationships between study time, absences, and final grades.
- Pairplot: A pairplot is created to visualize relationships among key features (`G1`, `G2`, `studytime`, `absences`, `G3`).

Data Preprocessing:

Steps:
1. Handling Missing Values:
   - No missing values were found in the dataset.

2. Handling Duplicates:
   - Duplicate rows are checked using `df.duplicated().sum()`. No duplicates were found.

3. Handling Outliers:
   - Outliers are identified using the Interquartile Range (IQR) method and removed from the dataset.

4. Encoding Categorical Features:
   - Ordinal categorical features (e.g., `Pstatus`) are encoded using `LabelEncoder`.

- Nominal categorical features are encoded using one-hot encoding (`pd.get_dummies()`).

5. Feature Scaling:
- Numerical features are standardized using `StandardScaler` to ensure all features are on the same scale.

6. Feature Engineering:
- An interaction term between `studytime` and `failures` is created to capture their combined effect on the final grade.
- `absences` are binned into categories (e.g., 'Very Low', 'Low', 'Medium', 'High', 'Very High') to make it easier for models to understand the data in ranges.

Model Implementation and Training

Steps:
1. Splitting Data into Training and Testing Sets:
- The dataset is split into training (80%) and testing (20%) sets using `train_test_split`.

2. Training the Linear Regression Model:
- A Linear Regression model is trained on the training data.
- The model's performance is evaluated using R^2 and Mean Squared Error (MSE) on both the training and test sets.

3. Hyperparameter Tuning with GridSearchCV:
- Hyperparameter tuning is performed for Ridge and Lasso regression models using `GridSearchCV`.
- The best model is selected based on the highest R^2 score.

4. Best Model Selection:
- The best model selected is Ridge Regression with the highest R^2 score on the test set.

Model Evaluation and Analysis

Steps:
1. Performance Metrics:
- The R^2 and MSE are calculated for both the training and test sets to evaluate the model's performance.

2. Visualization of Results:
- Actual vs. Predicted Grades: A scatter plot is created to compare actual and predicted grades.
- Residuals Distribution: A histogram of residuals is plotted to assess the model's prediction errors.

- Baseline Model Comparison: A baseline model (`DummyRegressor`) is used to compare the performance of the Ridge Regression model.

Model Deployment

Saving the Model:
- The best Ridge Regression model is saved using `joblib.dump()` to a file named `student_performance_model.pkl`.

Loading the Model and Making Predictions:
- The saved model is loaded using `joblib.load()`.
- The model is used to predict the final grade for a new student based on their study time, absences, and previous grades.

Cloud Deployment: The model has been deployed on Render Cloud, making it accessible via an
API at: https://ml-sk2t.onrender.com/docs

Frontend Web Application: To enhance user interaction, I developed a React-based web
application that provides a user-friendly interface for diabetes prediction. This allows users to input their medical data easily and receive real-time predictions without interacting directly with the API.
The website can be accessed here: https://machine-learning-roan.vercel.app/