CMPE252 Lab 1

Supervised Learning Problem: Movie Rating Prediction (Using Netflix or MovieLens Dataset)

Objective:
The goal of this exercise is to build a supervised learning model to predict the ratings given by users to movies. This is a regression problem where the output is a continuous rating.

You will be using the **MovieLens dataset** or **Netflix dataset** (using a publicly available third-party version) to predict movie ratings. The task will involve ingesting, cleaning the data, performing Exploratory Data Analysis (EDA), training a model, testing it, fine-tuning hyperparameters, and generating results.

Steps to Complete:

1. **Ingest the Data**:
    - Download the **MovieLens dataset** from the following link: https://grouplens.org/datasets/movielens/
    - Alternatively, you may choose to use the **Netflix dataset** available on Kaggle: Netflix Dataset on Kaggle: https://www.kaggle.com/netflix-inc/netflix-prize-data
    - Import the necessary libraries (Pandas, Numpy, Matplotlib, Seaborn, etc.) and load the dataset.
    - Ensure to load the data into a pandas DataFrame for easier manipulation.
2. **Data Cleaning**:
    - Handle missing values by checking for NaN (Not a Number) values and deciding on an approach (e.g., removing rows or filling with mean/median values).
    - Check for duplicate records and remove them if necessary.
    - Ensure the data types are correctly set (e.g., converting numeric columns to appropriate data types).
    - If applicable, remove any irrelevant or unnecessary columns that do not contribute to predicting the rating.
    - Transform categorical variables into numeric representations (e.g., one-hot encoding for categorical features like genre or user IDs).
3. **Exploratory Data Analysis (EDA)**:
    - **Summarize the dataset**: Use `.describe()` to get a quick overview of the numerical features.
    - **Visualizations**:
        - Plot the distribution of ratings.
        - Use correlation matrices to check how features relate to each other and the target variable (rating).
        - Create boxplots or histograms to explore the distribution of features like movie genres, user age groups, etc.
        - Visualize the number of ratings per user and per movie.
    - **Identify Outliers**: Look for outliers in numerical features (like ratings) and decide how to handle them.

    - **Explore the relationship between users, movies, and ratings**: Are there any observable patterns?
4. **Feature Engineering**:
    - Convert the datetime features (e.g., timestamp) to meaningful features, such as the month, year, or day of the week.
    - Aggregate ratings based on movies or users if needed.
    - Generate new features that could be helpful for predicting ratings, such as average ratings per movie, or frequency of ratings per user.
5. **Train/Test Split**:
    - Split the dataset into a **training set** and a **testing set** (e.g., 80% training, 20% testing).
    - Make sure that the data is split in such a way that there is no data leakage (e.g., ensure that a user's ratings are only in one of the datasets).
6. **Model Training**:
    - Choose a regression model for predicting the ratings (e.g., Linear Regression, Decision Tree Regressor, Random Forest Regressor, or Gradient Boosting).
    - Train the model on the training dataset.
    - Evaluate the model performance using suitable regression metrics such as **Mean Absolute Error (MAE)**, **Mean Squared Error (MSE)**, and **R^2 score**.
7. **Hyperparameter Tuning**:
    - Fine-tune the model by using techniques like **GridSearchCV** or **RandomizedSearchCV** to find the best hyperparameters.
    - For example, if using Random Forest, tune the number of estimators, max depth, and min samples per split.
    - Re-train the model with the best parameters and evaluate it again.
8. **Model Evaluation**:
    - After fine-tuning, evaluate the model's performance on the test set and compare the results to the initial model.
    - Provide a summary of how the model's performance has improved (or not) after hyperparameter tuning.
9. **Visualization of Results**:
    - Plot the **predicted vs. actual ratings** to assess the model's prediction accuracy.
    - Create residual plots (the difference between predicted and actual ratings) and any other additional visualizations that show various details (e.g., histogram or KDE plot)
10. **Save and Submit**:
    - Save the final model and any preprocessing steps as a pickle file (.pkl).
    - Ensure that all code and explanations are neatly organized in a Jupyter Notebook.
    - Convert the Jupyter Notebook into an HTML file and submit it.

Deliverables:

- A Jupyter Notebook file with all the steps implemented (ingesting data, cleaning, EDA, model training, evaluation, and hyperparameter tuning).
- An HTML file of the Jupyter Notebook for easy review. FULLY DOCUMENT your work.

Suggested Libraries:

- Pandas

- NumPy
- Matplotlib / Seaborn (for EDA and visualization)
- Scikit-learn (for regression models, data preprocessing, and hyperparameter tuning)
- Joblib or Pickle (to save the model)

Additional Information:

You may use the following datasets for this exercise:

- **MovieLens Dataset**: MovieLens Dataset
- **Netflix Dataset**: Available through Kaggle (Note: Netflix does not officially provide a public dataset, but there are third-party datasets available on platforms like Kaggle).

Evaluation Criteria:

- Correctness of data ingestion and cleaning steps.
- Completeness and clarity of Exploratory Data Analysis (EDA).
- The effectiveness of model selection and hyperparameter tuning.
- The quality and clarity of your code, visualizations, and explanations.
- Performance improvement after hyperparameter tuning.