End-to-End Machine Learning Project

Prepared by: Ruba Al Omari

Instructions

- Use the Students Performance dataset available at https://raw.githubusercontent.com/rubaomari/data/main/student/student-mat-modified-RA.csv to perform the tasks below.

- Note that the above dataset is a modified version of the original dataset: Cortez, Paulo. (2014). Student Performance. UCI Machine Learning Repository: **https://doi.org/10.24432/C5TG7T**

- When a submission is required, your notebook should load the dataset directly from the URL above. Do not clone or copy the dataset locally to your machine. If you do so, your notebook can't be run on another machine for marking.

- We will use this modified dataset to predict the student's performance based on a set of features. The data dictionary is available at: https://github.com/rubaomari/data/blob/main/student/student-mat-modified-RA.txt

- The notebook provided along this document is for guidance only. You are required to troubleshoot any errors and warnings you run into.

Tasks

1. Frame the problem.

2. Load the dataset directly from the URL above.
   2.1 Take a quick look at the data structure using `head()`, `describe()`, `info()`, and `value_counts()` for categorical data.
   2.2 Create a test set.

3. Explore and visualize the data to gain insights:
   3.1 Plot a histogram of the data using `hist()`.
   3.2 Look for correlations using Pearson correlation coefficient. Plot `G1` vs. `G3` using `sns.lineplot`, and create a scatterplot for `G2` and `G3`.

4. Prepare the data for Machine Learning algorithms:
   4.1 Check for duplicate rows, and remove them.

4.2 Handle the missing values: find the missing values, drop the `guardian` feature, and delete instances with null `famsize` attribute.

4.3 Create and apply a preprocessing pipeline to:

- Fill in the missing numerical values with the `mean` using a `SimpleImputer`.
- Scale the numerical columns using `StandardScaler`. Do not scale the target.
- Fill in the missing categorical values with the `most_frequent` value using `SimpleImputer`.
- Encode the categorical columns using `OneHotEncoder`.
- Display your pipeline.

5. Select a model and train it:

5.1 Split the data into 80% training set and 20% testing set.
Print the shape of X_train, y_train, X_test, and y_test using one print statement.
Note: Shapes should be `(317, 55) (317,) (80, 55) (80,)`

5.2 Train a `LinearRegression` model to predict `G3`, test using the test set, and report on the `MSE`.

5.3 Train a `LinearRegression` model using K-fold cross-validation with **5** folds, and report on the `cross_validation_score`.
Use negative mean squared error `neg_mean_squared_error` as the cross-validation scoring metric.

5.4 Calculate the `mean` of the cross-validation scores to get an overall assessment of the model's performance.

5.5 Train a `Ridge` regression model and a `Lasso` regression model with `alpha=1`. Test using the test set, and report on the `MSE`.

5.6 Plot the prediction vs. actual for the best-performing model.