UDACITY

PROJECT SPECIFICATION

Plagiarism Detector

All Required Files and Tests

CRITERIA	MEETS SPECIFICATIONS
Are all required files submitted?	The submission includes complete notebook files as `.ipynb`: "2_Plagiarism_Feature_Engineering" and "3_Training_a_Model". And the test and helper files are included: "problem_unittests.py", "helpers.py". The submission also includes a training directory `source_sklearn` OR `source_pytorch`.
Are all unit tests passed?	All the unit tests in project have passed.

Notebook 2: DataFrame Pre-Processing

CRITERIA	MEETS SPECIFICATIONS
For each data point, is a class label added and the `Category` converted to a numerical	The function `numerical_dataframe` should be complete, reading in the original file_information.csv file and returning a DataFrame of information with a numerical `Category` column and new, `Class` column.

value?

CRITERIA	MEETS SPECIFICATIONS
Is a `complete_df` created?	There is no code requirement here, just make sure you run all required cells to create a `complete_df` that holds pre-processed file text data and `Datatype` information.

Notebook 2: Features Created

CRITERIA	MEETS SPECIFICATIONS
Have you implemented a function to calculate containment?	The function `calculate_containment` should be complete, taking in the necessary information and returning a single, normalized containment value for a given answer file.
Did you answer the question about containment feature calculation?	Provide an answer to the question about containment feature calculation.

CRITERIA	MEETS SPECIFICATIONS
Have you implemented a function to calculate the longest common subsequence of words between two texts?	The function `lcs_norm_word` should be complete, taking in two texts and returning a single, normalized LCS value.
Have you created a DataFrame that holds at least 6 similarity features for all the text data?	Define an n-gram range to calculate multiple containment features. Run the code to calculate one LCS feature, and create a DataFrame that holds all of these feature calculations.

Notebook 2: Train and Test Files Created

CRITERIA	MEETS SPECIFICATIONS
Have you implemented the code to create train/test features and	Complete the function `train_test_data`. This should return only a *selection* of training and test features, and corresponding class labels.

CRITERIA	MEETS SPECIFICATIONS
features and labels?	
Have you selected a few "good" features to include in the train/test data?	Select at least three features to use in your final training and test data.
Did you answer the question about selecting features?	Provide an answer that describes why you chose your final features.
Have you made train/test csv files?	Implement the `make_csv` function. The class labels for train/test data should be in the first column of the csv file; selected features in the rest of the columns. Run the rest of the cells to create `train.csv` and `test.csv` files.

Notebook 3: Data Upload

CRITERIA	MEETS SPECIFICATIONS
Have you uploaded your training	Upload the `train.csv` file to a specified directory in an S3 bucket.

CRITERIA	MEETS SPECIFICATIONS
training data to S3?	

Notebook 3: Training a Custom Model

CRITERIA	MEETS SPECIFICATIONS
Have you completed a training script?	Complete at least *one* of the `train.py` files by instantiating a model, and training it in the main if statement. If you are using a custom PyTorch model, you will have to complete the `model.py` file, as well (you do not have to do so if you choose to use an imported sklearn model).
Have you defined a custom estimator?	Define a custom sklearn OR PyTorch estimator by passing in the required arguments.
Have you trained your model?	Fit your estimator (from the previous rubric item) to the training data you stored in S3.

Notebook 3: Deploying and Evaluating a Model

CRITERIA	MEETS SPECIFICATIONS
Have you deployed the trained model?	Deploy the model and create a `predictor` by specifying a deployment instance.

CRITERIA	MEETS SPECIFICATIONS
Is the test accuracy of your model greater than 90%?	Pass test data to your deployed `predictor` and evaluate its performance by comparing its predictions to the true, class labels. Your model should get at least 90% test accuracy.
Did you answer the two questions about model metrics and design considerations?	Provide an answer to the two model-related questions.

Notebook 3: Cleaning up Resources

CRITERIA	MEETS SPECIFICATIONS
Did you delete your model endpoint?	Run the code to clean up your final model resources.