Model Documentation

Introduction

This document provides the explanation of a machine learning model that uses a Logistic Regression classifier to categorize customer complaints based on product categories. The model was trained on a dataset of customer complaints with associated product categories.

Modules used

- `pandas`: Used for data loading and manipulation.
- `sklearn.model_selection`: `train_test_split` function was used to split the data into training and testing sets.
- `sklearn.feature_extraction.text`: `CountVectorizer` was used to convert the consumer complaints into a matrix of token counts.
- `sklearn.linear_model`: `LogisticRegression` was used to classify the complaints into product categories.
- `sklearn.metrics`: `classification_report` was used to evaluate the model's performance.
- `pickle`: Used to serialize and deserialize the model and the vectorizer, allowing them to be saved and loaded.

Instructions

1. **Data Loading**: The first step is to load the data using pandas. We assume that the dataset is a CSV file with a 'Consumer complaint narrative' column for the complaints and a 'Product' column for the product categories. Missing values in 'Consumer complaint narrative' are replaced with empty strings.

2. **Feature Extraction**: The 'Consumer complaint narrative' column is transformed into a matrix of token counts using `CountVectorizer`. This is our feature matrix, X.

3. **Model Training**: The data is split into training and testing sets. The Logistic Regression model is then trained on the training set.

4. **Model Evaluation**: The model is evaluated on the test set using the classification report, which provides key metrics such as precision, recall, f1-score, and support for each category, as well as overall averages.

5. **Prediction Function**: A prediction function `classify_complaint_classical` is defined. This function takes a raw complaint as input, transforms it using the vectorizer, makes a prediction with the model, and returns the predicted product category.

6. **Saving the Model and Vectorizer**: The model and vectorizer are saved using pickle to be loaded later.

Model Training

The model is trained on a dataset of consumer complaints. The 'Consumer complaint narrative' is used as the input feature, and the 'Product' is used as the target.

The dataset is first split into a training set and a test set, with 80% of the data used for training and 20% used for testing. The model is then trained using the `fit` method of the `LogisticRegression` object.

Evaluation

The model's performance is evaluated using the test set. The evaluation metric used is the classification report, which provides the precision, recall, f1-score, and support for each class, as well as overall averages.

Prediction

The prediction function takes a complaint as input, transforms it using the `CountVectorizer` that was fitted on the training data, and then makes a prediction using the Logistic Regression model. The prediction is the product category associated with the complaint.

API

A Flask API can be created to deploy the model. The API takes a complaint as input in JSON format, processes it using the vectorizer and the model, and returns the predicted category also in JSON format. This allows the model to be used as a service that can be accessed via HTTP.

Note: The API requires the Flask library and pickle for loading the serialized model and vectorizer.

The model is now ready to make predictions on new, unseen complaints.