

Practical No:7

Title:

Implement text classifier using logistic regression model

Aim:

To implement a text classifier using logistic regression, which can classify text data into distinct categories based on the provided input features.

Pre-requisites:

1. Understanding of machine learning concepts, especially classification.
 2. Familiarity with logistic regression and its applications in binary and multi-class classification.
 3. Basic knowledge of text preprocessing techniques (tokenization, vectorization, etc.).
 4. Experience with Python libraries such as `scikit-learn` for model building and `pandas` for data handling.
-

Theory:

Text classification is a natural language processing (NLP) task where text data is categorized into predefined labels. A logistic regression model, a linear model widely used for binary classification, estimates the probability of a text sample belonging to a particular class based on feature input.

Logistic Regression

Logistic regression is a supervised learning algorithm that uses a linear combination of features, applies the logistic function, and outputs a probability between 0 and 1. It's

particularly effective in binary classification tasks but can also handle multi-class classification using techniques such as one-vs-rest.

Given features X and a binary target variable y , the logistic regression model is defined as:

$$P(y = 1|X) = \frac{1}{1 + e^{-(wX+b)}}$$

where:

- w is the weight vector,
- b is the bias term, and
- e is the base of the natural logarithm.

Steps to Implement a Text Classifier with Logistic Regression:

1. **Data Collection:** Use a labeled dataset for text classification (e.g., spam vs. not spam, sentiment analysis).
2. **Text Preprocessing:**
 - **Tokenization:** Split the text into words (tokens).
 - **Stop Word Removal:** Remove commonly used words that may not add meaningful information.
 - **Vectorization:** Convert tokens to numerical features using techniques like Bag of Words or TF-IDF (Term Frequency-Inverse Document Frequency).
3. **Train-Test Split:** Divide the dataset into training and testing sets to evaluate the model.
4. **Model Building:** Train the logistic regression model on the vectorized features.

5. Model Evaluation: Evaluate the model's accuracy, precision, recall, and F1-score.

Conclusion:

Using logistic regression for text classification is efficient and provides robust results for binary classification tasks. This implementation demonstrates basic NLP preprocessing steps like tokenization and TF-IDF vectorization, which are essential for converting text data into numerical form. The logistic regression model successfully learns from these features to classify text data into predefined categories.