Spam E-mail Filter Report

1. Notebook Flow Description:

1. Data Loading: The dataset, "Spam_Email_Data.csv," is loaded into a pandas DataFrame.
2. Data Preprocessing & Features Extraction: Text data preprocessing techniques such as converting text to lowercase, removing special characters and numbers, tokenization, removing stopwords, and lemmatization are applied to clean the text data. This ensures that the text data is in a suitable format for further processing and analysis.
3. Data Splitting: The dataset is split into training and testing sets using a 60-40 ratio. This ensures that an adequate amount of data is available for both training and evaluating the models.
4. Text Embedding: Various text embedding techniques are employed, including Word2Vec, Doc2Vec, Bag of Words, and TF-IDF. These techniques transform text data into numerical vectors, which can be used as input features for machine learning models.
5. Model Training: Two classifiers, Logistic Regression and Decision Tree, are chosen for training. Each classifier is trained using the different text embedding techniques mentioned above.
6. Model Evaluation: The performance of each model is evaluated using accuracy and F1-score metrics.

Following this order ensures a logical progression in the notebook flow, from data loading and preprocessing to model training and evaluation. It allows for a comprehensive analysis of the text data and helps in making informed decisions throughout the text classification pipeline.

2. Data Preprocessing & Features Extraction:

Data preprocessing techniques such as converting text to lowercase, removing special characters and numbers, tokenization, removing stopwords, and lemmatization are chosen to clean the text data. These techniques help in standardizing the text data and reducing noise, making it more suitable for analysis. Additionally, the choice of lemmatization over stemming ensures that words are transformed into their root forms, preserving the semantic meaning of the text.

3. Data Splitting:

The dataset is divided into training and testing sets using a 60-40 ratio. This ratio ensures that there is a sufficient amount of data for both training and testing the models. Splitting the data in this manner helps in evaluating the generalization performance of the models. A larger portion of the data is allocated to the training set to ensure that the models are trained on an adequate amount of data.

4. Model Training:

Logistic Regression and Decision Tree classifiers are chosen for training due to their simplicity, and effectiveness for text classification tasks. These classifiers are suitable for handling both linear and non-linear relationships in the data. Additionally, various text embedding techniques such as Word2Vec, Doc2Vec, Bag of Words, and TF-IDF are employed to represent text data as numerical vectors for model training.

5. Model Evaluation:

The models are evaluated using accuracy and F1-score metrics. Accuracy measures the overall correctness of the predictions, while the F1-score provides a balance between precision and recall. These metrics are suitable for evaluating the performance of classifiers in binary classification tasks.

6. Dominant Models:

These dominant models are chosen based on their ability to achieve high accuracy and F1-score on the test dataset. (highlighted in red)

Model	Accuracy	F1-Score
Logistic Regression with Word2vec	0.987495	0.987464
Logistic Regression with doc2vec	0.974558	0.974477
Logistic Regression with Bag of Words	0.996119	0.996117
Logistic Regression with TF-IDF	0.985339	0.985254
Decision Tree with Word2vec	0.974127	0.974183
Decision Tree with doc2vec	0.868047	0.868377
Decision Tree with Bag of Words	0.978008	0.977987
Decision Tree with TF-IDF	0.978439	0.978439