Spam Email Classification

1. Dataset Overview:

The dataset used for the spam classification task consists of emails labeled as either "spam" or "not spam." The spam column contains binary labels, with 1 indicating spam and 0 indicating non-spam. The primary text data in the text column represents the body of the emails. The data was initially loaded from a CSV file and previewed to ensure that the structure is correct.

- The dataset was imbalanced, with a significant number of non-spam emails compared to spam emails. This imbalance posed a challenge for training machine learning models effectively.

2. Text Preprocessing:

To prepare the text data for machine learning models, several preprocessing steps were applied:

- **Lowercasing**: All email text was converted to lowercase using a function, ensuring uniformity and eliminating discrepancies caused by case differences.
- **Removing Special Characters**: A function was created to remove any non-alphanumeric characters from the text to avoid noise that may affect model accuracy.
- **HTML Tag Removal**: Emails containing HTML tags had those tags removed using a regular expression. This step ensured that any HTML formatting did not interfere with the analysis.
- **Stopwords Removal**: Common words such as "the", "is", "in" that do not carry significant meaning were removed using the NLTK library's stopwords list. This step helped reduce the dimensionality of the data and focused on the more meaningful words.

3. Handling Imbalanced Data:

The dataset had a significant imbalance between spam and non-spam emails, with fewer spam emails. To address this, **SMOTE (Synthetic Minority Over-sampling Technique)** was used to generate synthetic data for the minority class (spam). This oversampling technique increased the number of spam emails to balance the dataset, which helped improve the model's ability to predict spam effectively.

4. Feature Extraction:

- **TF-IDF (Term Frequency-Inverse Document Frequency)** was used to convert the email text into numerical vectors, capturing the importance of words in the context of the entire dataset. This method helps in reducing the impact of frequently occurring words that are less informative and focuses more on unique terms.

5. Model Training:

The following models were trained on the preprocessed data:

- **Multinomial Naive Bayes** (Naive Bayes classifier for multinomial distributions)
- **Logistic Regression**
- **Random Forest Classifier**

- **Support Vector Classifier (SVC)**
- **Decision Tree Classifier**

All models were trained on the training set created using SMOTE, and predictions were made for both training and test datasets.

6. Model Prediction and Evaluation:

- **Sample Predictions**: The models were tested on sample emails, and each model made predictions whether the email was spam or not.
- **Confusion Matrix**: A confusion matrix was generated for each model, showing the number of true positives, true negatives, false positives, and false negatives. This allowed for a detailed comparison of how each model performed in terms of accuracy, precision, recall, and F1-score.
- **Classification Report**: The classification report, which includes precision, recall, and F1-score for each class (spam and not spam), was generated for each model. This provided deeper insight into the performance of each model beyond simple accuracy.

7. Model Performance Comparison:

- **Accuracy Scores**: The accuracy scores of each model were calculated on the test dataset. A bar chart was created to visually compare the accuracy of the models, revealing the most effective models for spam classification.
- **KFold Cross-Validation**: To further assess the robustness of the models, KFold cross-validation (with 5 splits) was used. This technique splits the dataset into 5 parts, training the model on 4 parts and testing it on the remaining part. Cross-validation scores were recorded for each model, and the results were plotted to observe how consistently each model performed across different data splits.
- **Sorted Model Scores**: The models were ranked based on their accuracy scores, and the sorted results were displayed in a bar chart. This provided a clear view of which models performed the best.

9. Visualizations:

- **Confusion Matrices** were plotted using heatmaps for each model to visually represent performance on the test set.
- **Classification Reports** were visualized as heatmaps to show precision, recall, and F1-score for each class.
- **Model Comparison Bar Charts** displayed the accuracy scores of each model.
- **KFold Cross-Validation Results** were plotted to show how each model performed across multiple splits of the data.