**Model Summary : Abstract**

**Resume Screening using Next Word Predictor**

**1. Introduction**

The dataset used in this project is taken from the UCI Machine Learning Repository, specifically the SMS Spam Collection dataset. The main objective of this project is to create an application that can automatically segregate spam SMS messages and move them to a spam folder, similar to the spam filtering systems used by major firms to avoid promotional and fraudulent messages.

**2. Model Preprocessing**

**2.1)** **Data Cleaning**

* Removed NaN values and outliers to ensure the dataset is clean and ready for analysis.

**2.2)** **Exploratory Data Analysis (EDA)**

* Performed EDA using libraries such as Matplotlib and Seaborn for visualization tasks.
* Used charts like distplot, bar chart, pie chart, pairplot, and heatmap to find hidden patterns and better understand the data.
* Discovered that the dataset is imbalanced, with a higher ratio of non-spam messages compared to spam messages.

**2.3)** **NLTK Library**

* Utilized the NLTK library for converting and counting the number of words, sentences, and characters in each message.
* Converted text to lowercase.
* Tokenized words from sentences.
* Removed special characters.
* Removed stopwords (e.g., is, the, am, are) and punctuations (e.g., , ?!)
* Performed stemming and lemmatization to reduce the chances of confusion.

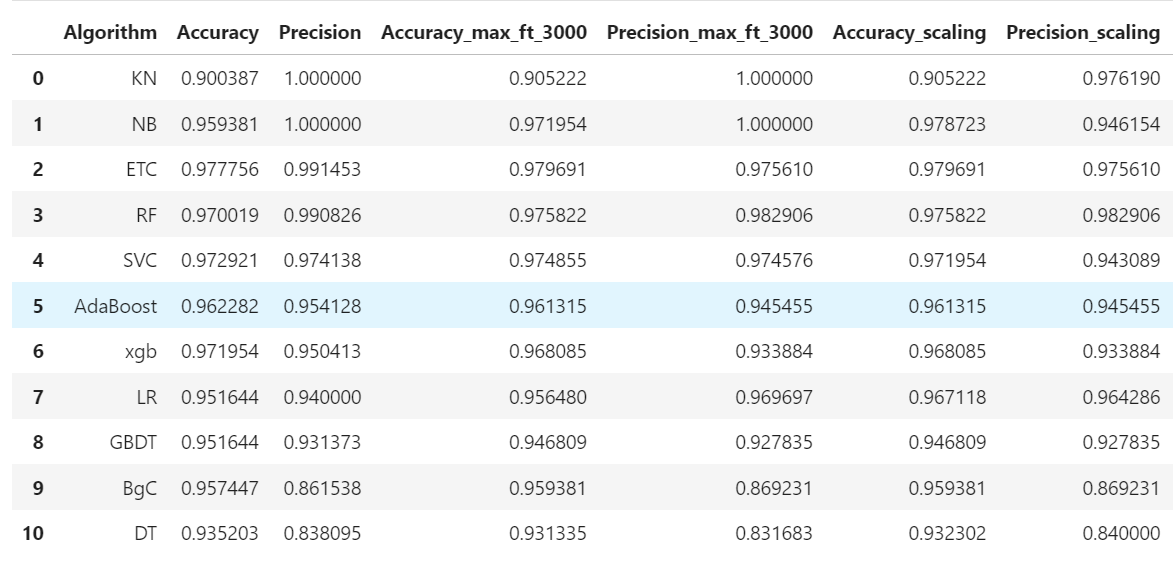
**3. Model Building**

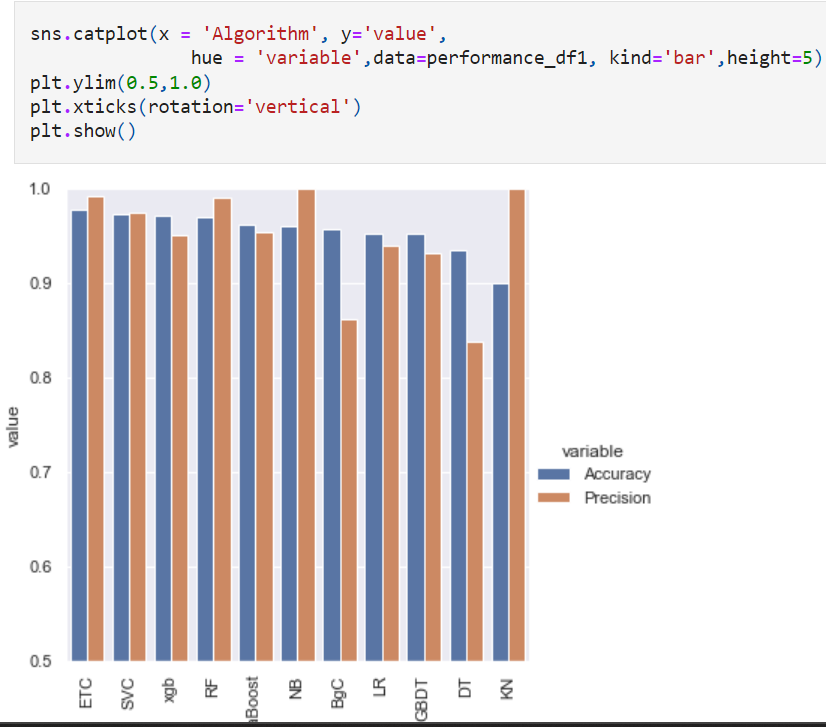
**3.1)Text Vectorization**

* Applied text vectorization techniques like Bag of Words and TF-IDF to convert strings into numerical data.
* Used various variants of Naive Bayes (Bernoulli, Multinomial, and Gaussian) to evaluate performance metrics such as accuracy, confusion matrix, and precision.
* Found that TF-IDF with Gaussian Naive Bayes provided the best accuracy and precision, whereas Bag of Words performed best with Multinomial Naive Bayes.

**3.2)** **Classification Algorithms**

* After applying Naive Bayes, achieved an accuracy of 0.95 and a precision of 1.
* Tested other classification algorithms (Logistic Regression, SVM, Decision Trees, KNN, Random Forest, AdaBoost, Gradient Boost, Bagging Classifier, Extra Trees Classifier, and Voting and Stacking Classifier).
* None of the other algorithms outperformed Naive Bayes in terms of both accuracy and precision, thus confirming Naive Bayes as the best choice for this task.



* 

**4. Model Evaluation and Frontend Formation**

* Dumped the trained model using Pickle.
* Imported the model into the Pycharm virtual environment code editor.
* Created the frontend for the SMS Spam Classifier using the Streamlit library.

**5. Conclusion**

* The development of the SMS Spam Classifier demonstrates the practical application of machine learning techniques in text classification. By using methods such as data cleaning, exploratory data analysis, text vectorization, and various classification algorithms, we were able to build a model that effectively identifies spam messages. The use of Naive Bayes algorithms proved to be the most effective, achieving high accuracy and precision. The model was then deployed in a user-friendly interface using Streamlit, making it accessible for practical use.and utility, showcasing the potential of tailored models in specific contexts.