OPTIMAL MODEL SEARCH: SMS SPAM CLASSIFICATION MACHINE LEARNING PROJECT

The primary objective of this project is to discover the most effective classification model for detecting text message spam.

Dexter Astorga

This document contains:

- Project Overview
- Project Architecture Diagrams
- Project Discussion (Process Walkthrough)
- Results and Discussions
- Recommendations

PROJECT OVERVIEW

This project focuses on finding the optimal model for text message spam classification through a detailed process:

1. The Dataset

   The dataset used in this project is a culmination of 3 different datasets. This is done to introduce data diversity and overcome high imbalance.

2. Dataset Partition

   Three new different training datasets are created. They are divided into these ham to spam partitions: 70-30, 60-40, and 50-50. These datasets are used for training and testing machine learning models, independent of each other.

3. Model Selection in Each Partition

   Out of 11 different classification algorithms, the best model was identified by training all the models using default parameters and cross validating the results, (hold out validation was done but not considered in selecting best model because of bias).
   Certainly, here are the names of the classification algorithms you mentioned, numbered for convenience:

   a. Logistic Regression
   b. Support Vector Classifier
   c. Bernoulli Naive Bayes
   d. Decision Tree Classifier
   e. K-Neighbors Classifier
   f. Random Forest Classifier
   g. AdaBoost Classifier
   h. Bagging Classifier
   i. Extra Trees Classifier
   j. Gradient Boosting Classifier
   k. XGBoost Classifier

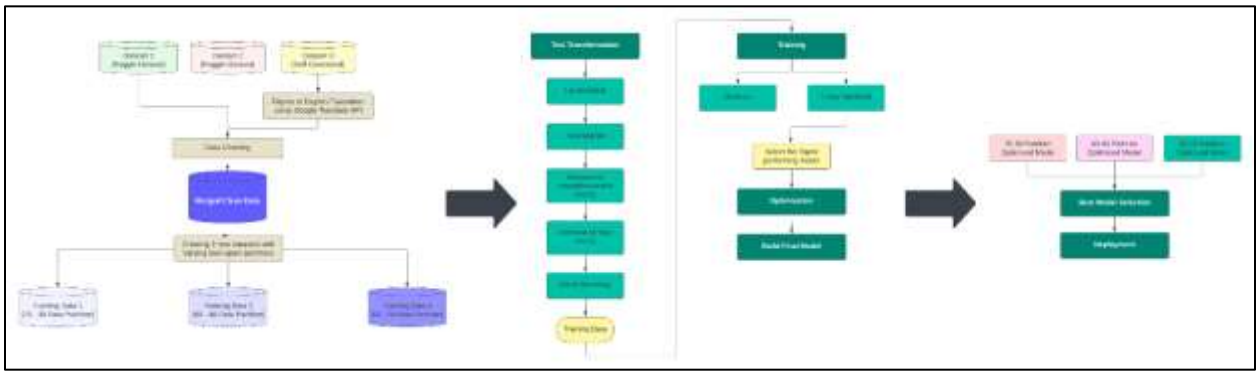
4. Model Optimization

   All the best models from each partition were further optimized to enhance their performance.

5. Model Comparison

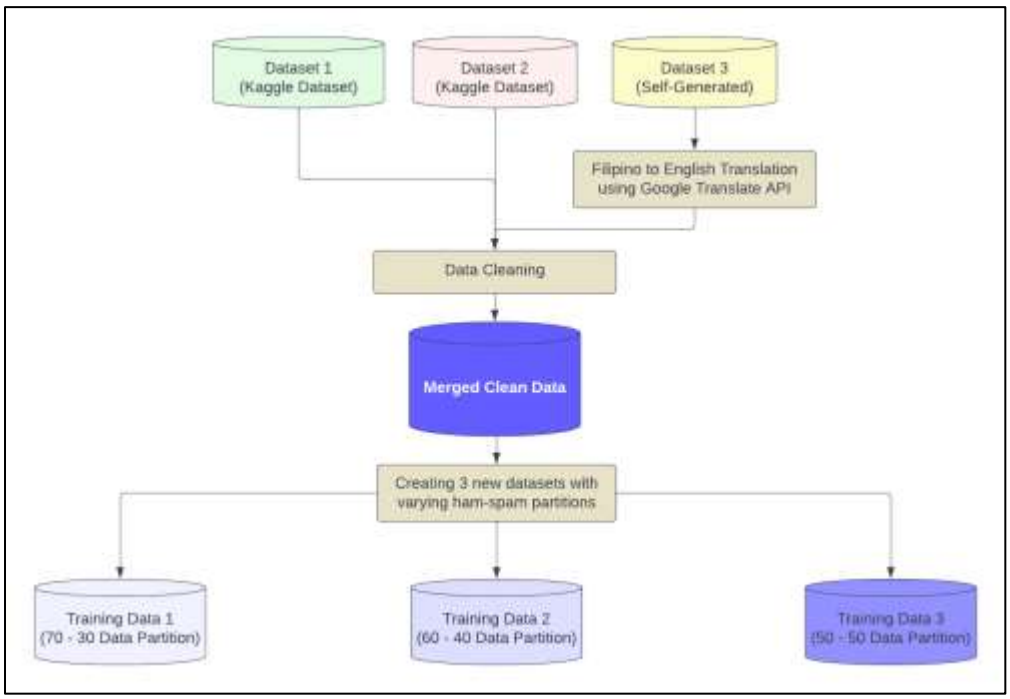
   After optimization, I compared the performance of these models to select the best-performing one among them.

PROJECT ARCHITECTURE DIAGRAMS



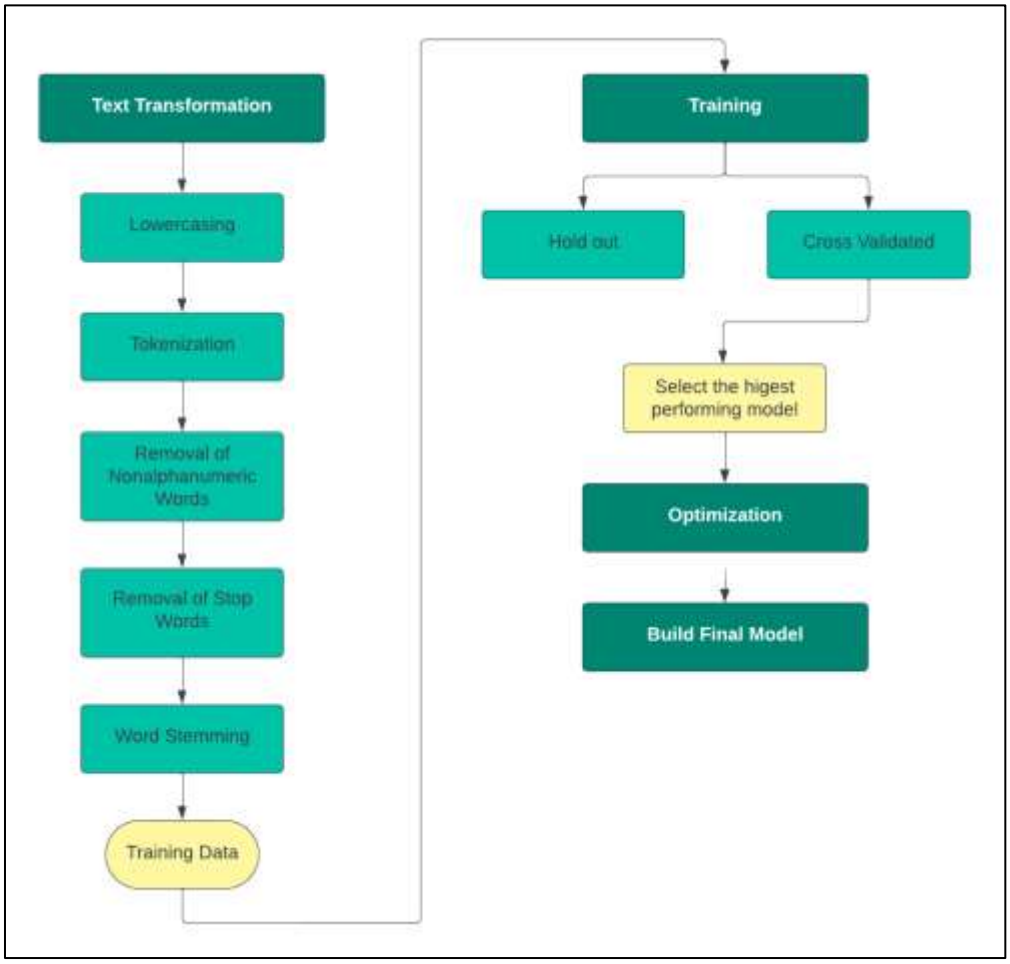
MAIN PROJECT ARCHITECTURE

This diagram is available here: https://github.com/Dex-Astorga/ml-spam-detection/blob/main/project_diagram.png



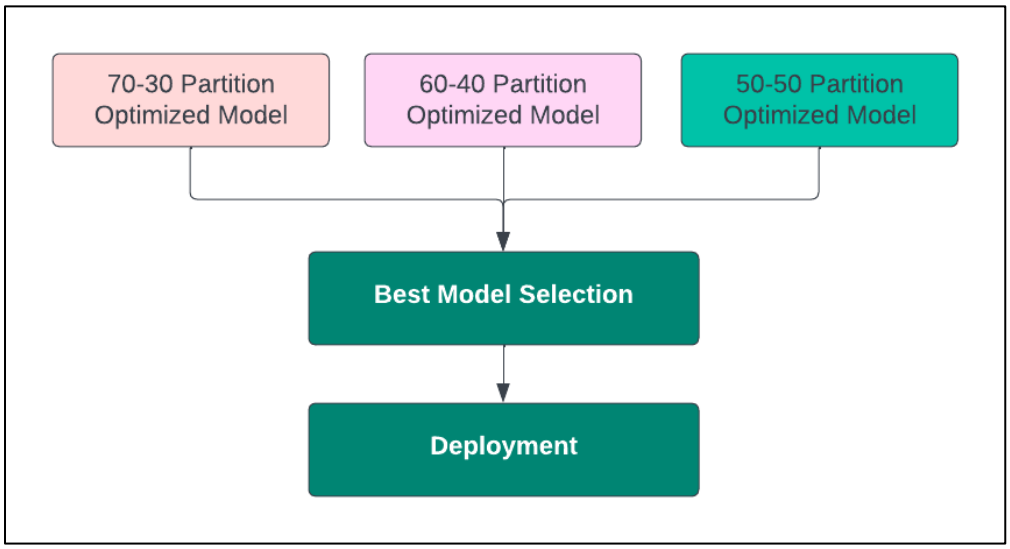
DATASET PIPELINE

This diagram is available here: https://github.com/Dex-Astorga/ml-spam-detection/blob/main/diagrams/dataset_pipeline.png



TRAINING PIPELINE

This diagram is available here: https://github.com/Dex-Astorga/ml-spam-detection/blob/main/diagrams/machine_learning_pipeline.png



FINAL MODEL

This diagram is available here: https://github.com/Dex-Astorga/ml-spam-detection/blob/main/diagrams/best_model.png

PROJECT DISCUSSION (PROCESS WALKTHROUGH)

Datasets:

https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset

https://www.kaggle.com/datasets/bwandowando/philippine-spam-sms-messages

https://github.com/Dex-Astorga/ml-spam-detection/blob/main/original_datasets/dataset3.csv

Data Wrangling:

1. Cleaned each dataset.
2. The third dataset needed to be translated to English and it is done by using Google Translate API. The NLTK Library uses English words and since the third dataset has Filipino SMS messages, we needed the translations.
3. Merging the three datasets into a master dataset.
4. Creation of three different training datasets, each with different ham to spam partitions: 70-30, 60-40, 50-50. This approach allowed us to identify if data imbalance is a key point in the machine learning model performance.

***Model Training*:**

Each dataset is trained in 11 different machine learning classification algorithms. The following metrics are recorded: Accuracy, Precision, Recall, F1-Score, but the best performing models are evaluated on the average of Accuracy and Precision. Only models trained using cross validation are qualified to be best performing models because hold out validation introduces bias.

Accuracy: Evaluating accuracy ensures that your model correctly predicts both classes, which aligns with your goal of high correctness.

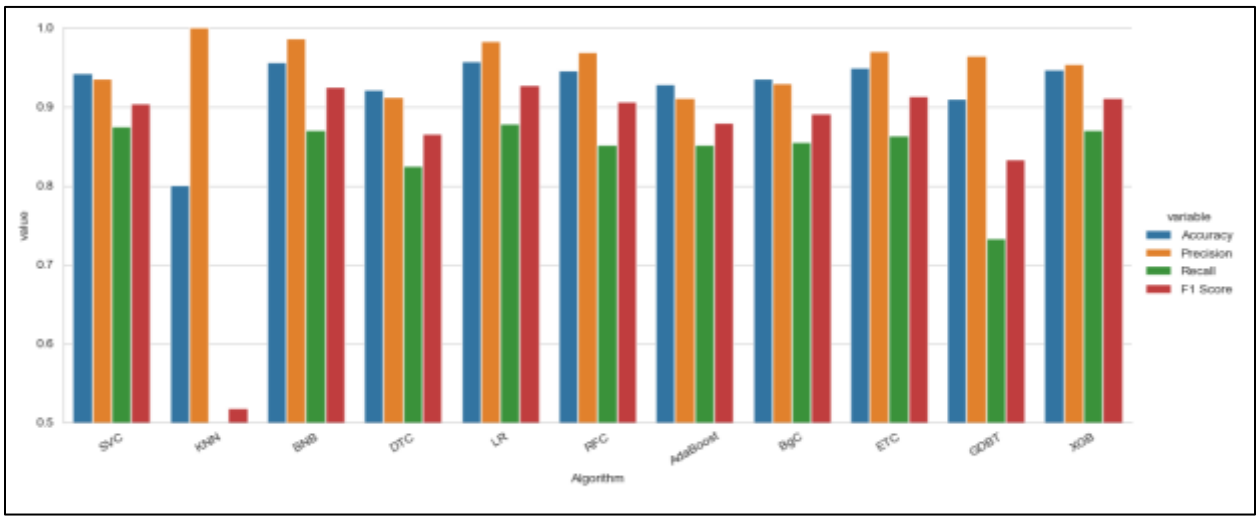
Precision: Precision focuses on the correctness of positive predictions. By considering precision, you are emphasizing the importance of correctly identifying instances of the positive class. This can be particularly relevant when false positives are costly.

Training Performances:

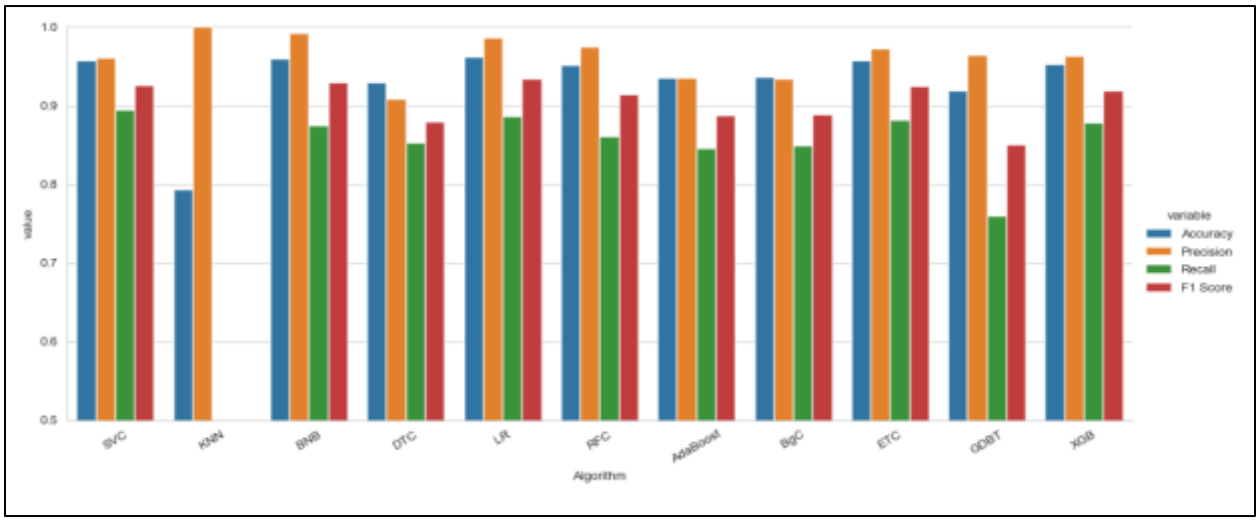
Across the three differently-partitioned datasets, Bernoulli Naive Bayes Algorithm hailed the best classification model.

70 – 30 Partition Training Dataset:

Hold Out Performances:



Cross Validation Performances:

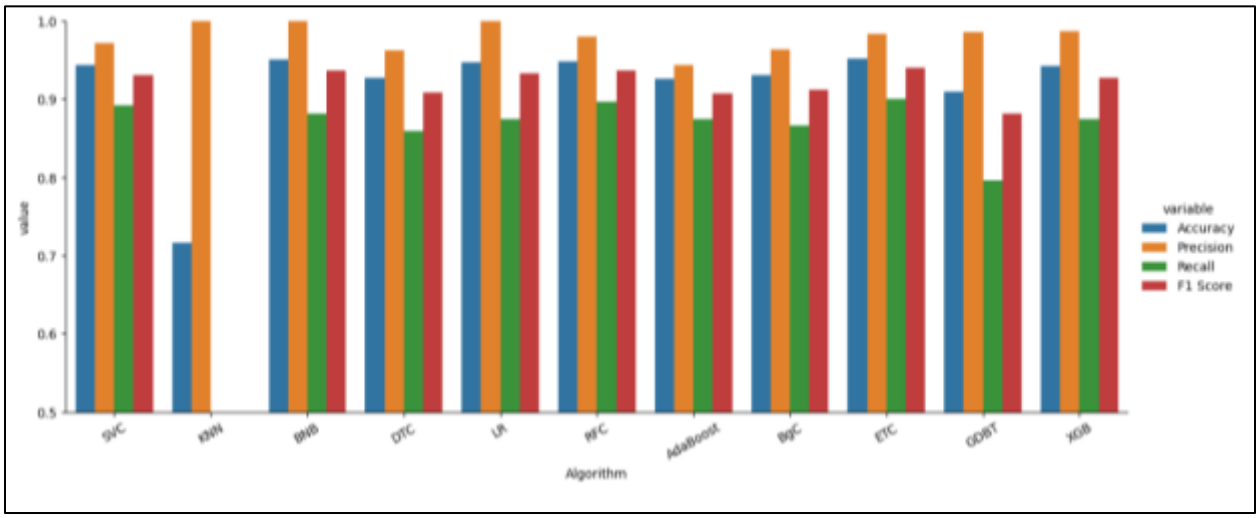


Best Model:

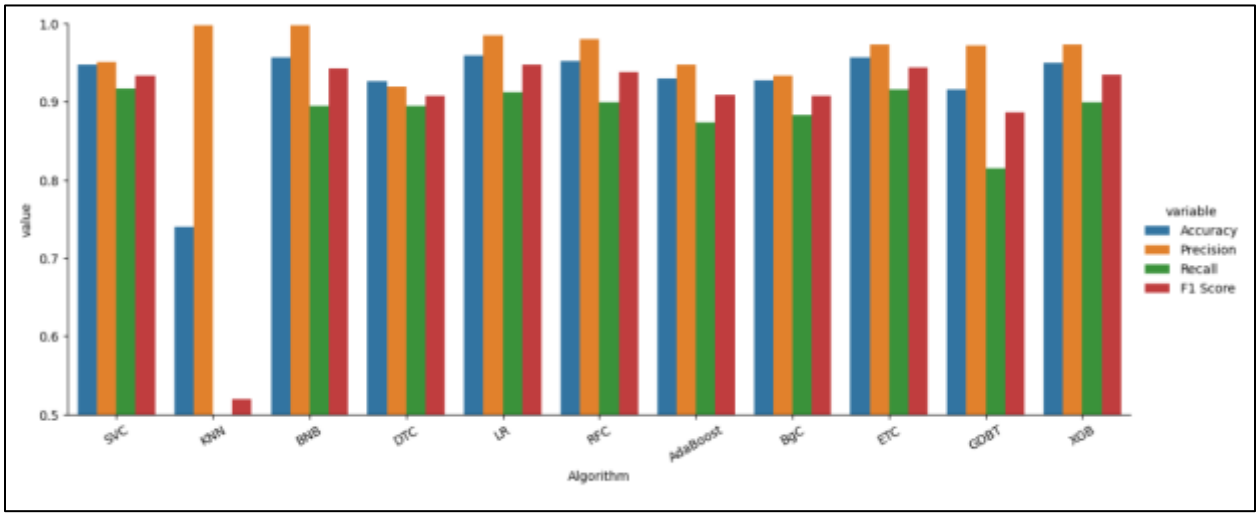
With mean accuracy of 96.05% and mean precision of 99.21%, the best-performing model is the Bernoulli Naive Bayes Algorithm (BNB).

60 – 40 Partition Training Dataset:

Hold Out Performances:



Cross Validation Performances:

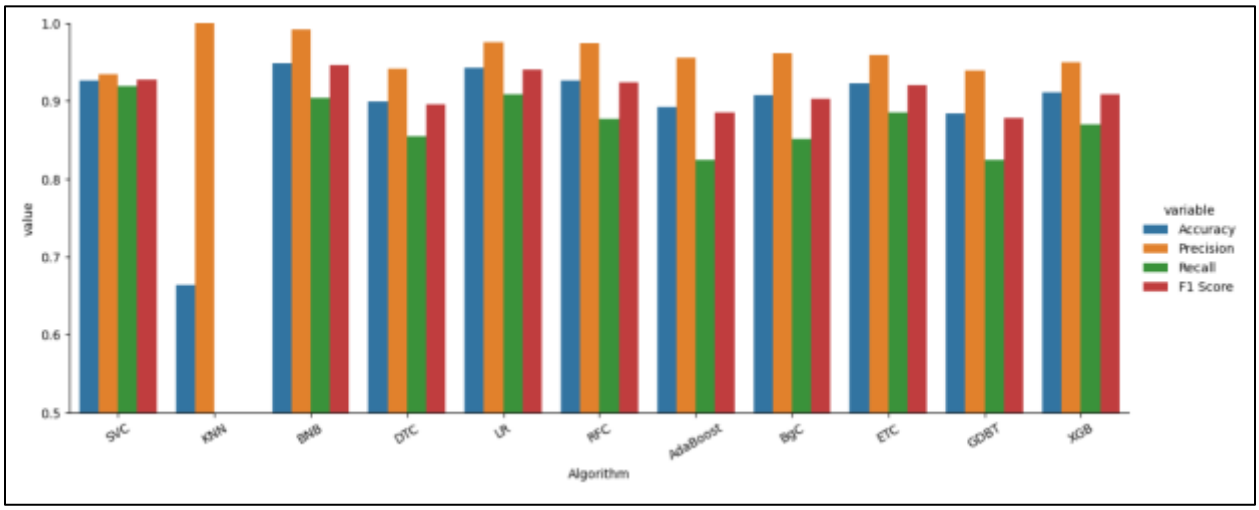


Best Model:

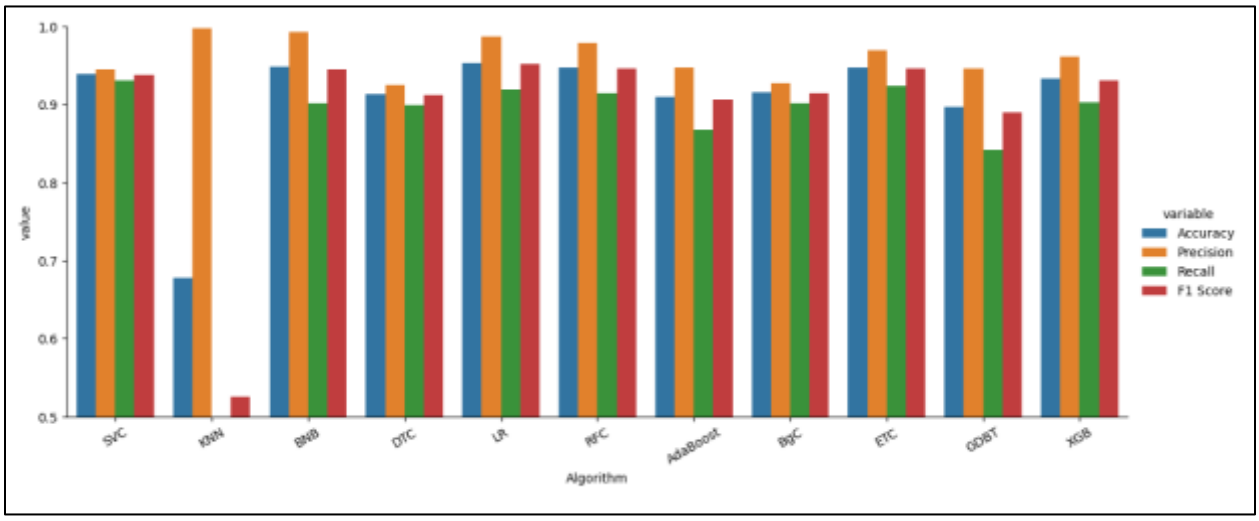
With mean accuracy of 95.66% and mean precision of 99.74%, the best-performing model is the Bernoulli Naive Bayes Algorithm (BNB).

50 – 50 Partition Training Dataset:

Hold Out Performances:



Cross Validation Performances:



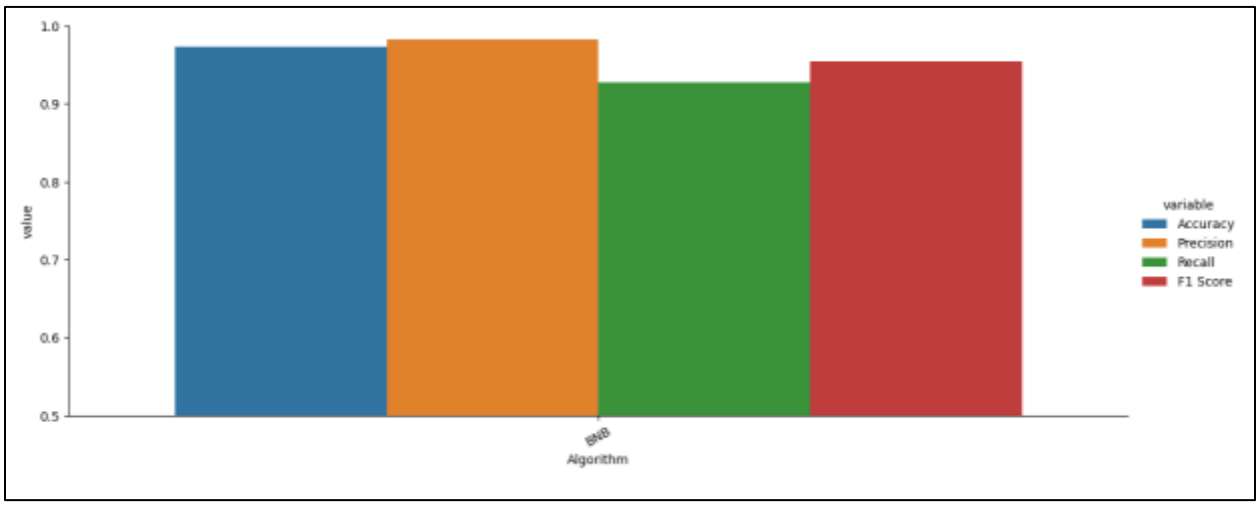
Best Model:

With mean accuracy of 94.77% and mean precision of 99.32%, the best-performing model is the Bernoulli Naive Bayes Algorithm (BNB).

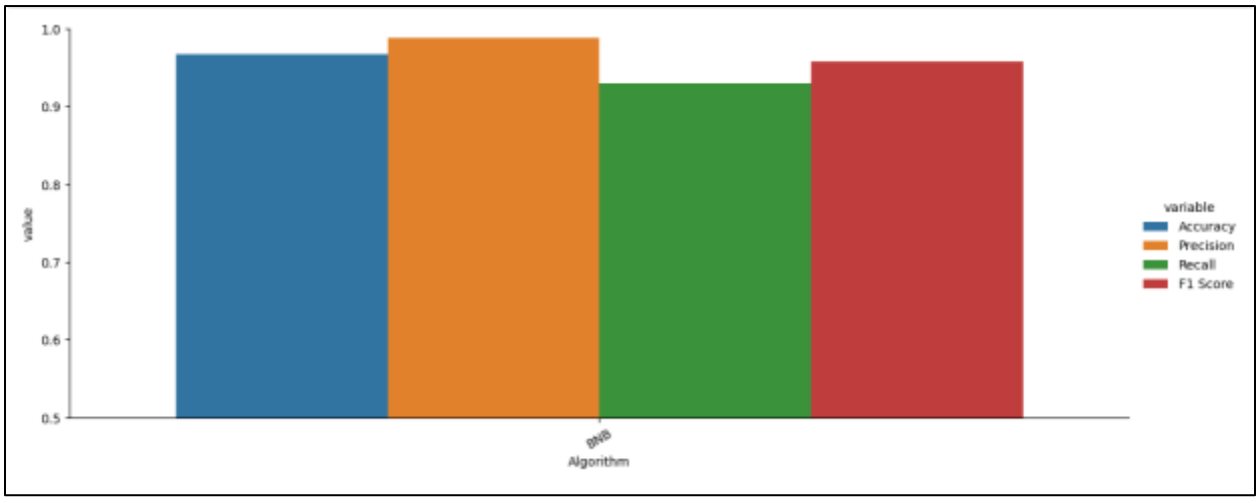
Optimization:

Optimization techniques are applied to each training dataset using BNB. The highest performing model is the BNB trained model for 50 – 50 ham to spam partition dataset.

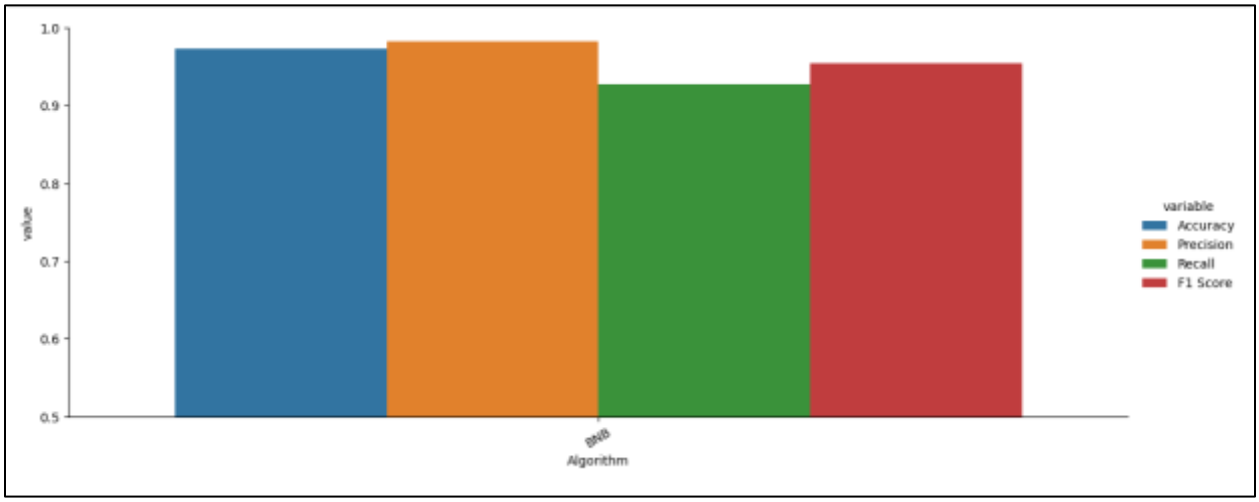
70 – 30 Partition Training Dataset Optimized Model Cross Val Results:



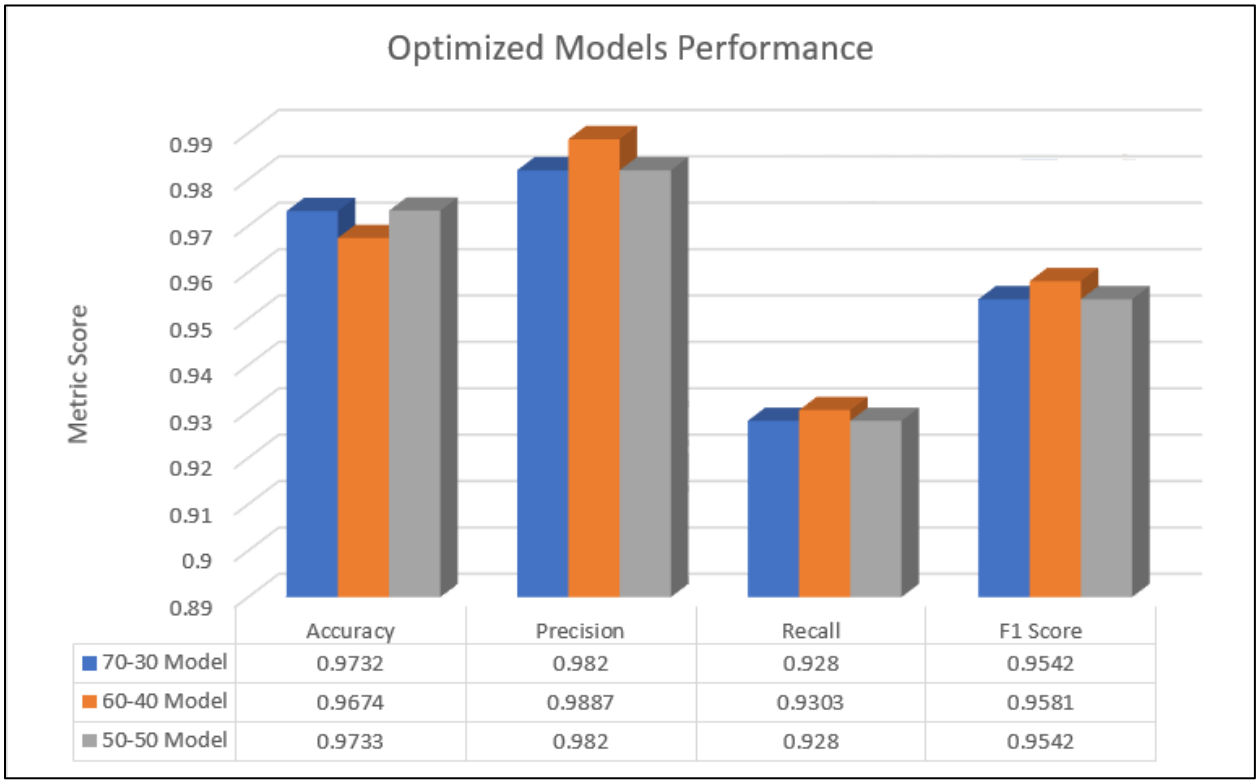
60 – 40 Partition Training Dataset Optimized Model Cross Val Results:



50 – 50 Partition Training Dataset Optimized Model Cross Val Results:



Performance Summary:



	Accuracy	Precision	Recall	F1 Score
70-30 Model	0.9732	0.982	0.928	0.9542
60-40 Model	0.9674	0.9887	0.9303	0.9581
50-50 Model	0.9733	0.982	0.928	0.9542

RESULTS AND DISCUSSIONS

Based on the results of the optimized models, the model with highest performance is Bernoulli Naive Bayes trained in 50 – 50 ham to spam partition. The performance of 70 – 30 dataset partition is not far from the 50 – 50 model results so it is inferred that data imbalance is not a huge factor. The fact that BNB scored the highest scores in all training datasets proves that it is the most effective classification algorithm for this job.

RECOMMENDATIONS

1. Larger Datasets

   Consider expanding your dataset by collecting a larger volume of text messages for training and testing. A larger dataset can help improve the model's ability to generalize and detect spam patterns effectively.

2. Data Diversity

   In addition to increasing the dataset size, focus on enhancing data diversity. Collect text messages from various sources and languages to create a more representative dataset that reflects real-world scenarios.

3. Alternative Vectorization Methods

   Explore alternative text vectorization techniques beyond the ones you've used. Consider methods such as TF-IDF (Term Frequency-Inverse Document Frequency) and word embeddings (e.g., Word2Vec, GloVe) to represent text data differently and potentially improve model performance.

4. Deep Learning

   Investigate the use of deep learning models for text message spam detection. Techniques like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) can capture complex patterns in text data. Experiment with architectures like LSTM (Long Short-Term Memory) and Transformer-based models (e.g., BERT) for improved performance.

5. Transfer Learning:

   Explore the application of transfer learning in NLP. Pre-trained language models like GPT-3 and BERT can be fine-tuned on your spam detection task, potentially providing significant performance gains.

I hope this one helps you. If you have any questions you can reach me at codex1727@gmail.com.

Thank you!

Dexter Astorga