**Document Classification**

**1. Introduction**

This project focuses on classifying SMS messages as either **"ham" (non-spam)** or **"spam"**, using a combination of unsupervised clustering and supervised machine learning. We use a labeled SMS dataset and apply different vectorization techniques and models to improve classification accuracy.

**2. Unsupervised Learning: Clustering**

**2.1 Data Loading & Preprocessing**

* Loaded the dataset, keeping only the message text and label.
* Converted messages to strings to ensure compatibility with NLP processing.

**2.2 TF-IDF Vectorization**

* Applied **TF-IDF** (Term Frequency-Inverse Document Frequency) to convert text into numerical features.
* Limited the number of features to 1,000 (max\_features=1000) for dimensional control.

**2.3 Clustering with K-Means and Hierarchical**

* Used **PCA** to reduce TF-IDF vectors to 2D for visualization.
* Applied **K-Means** to form 2 clusters.
* Also used **Hierarchical Clustering** to compare results.

**2.4 Results & Insights**

* The 2D plot visualizes message clusters based on PCA-reduced TF-IDF vectors.
* Axes: Principal components (PC1 & PC2).
* Colors: Cluster assignments from K-Means (likely spam vs ham).

**Observations:**

* The clear “V” shape indicates natural separation.
* Some overlaps suggest ambiguous cases.
* One cluster dominates, likely due to class imbalance (more ham messages).

**Takeaway:**

* K-Means reasonably separates messages into spam and ham.
* However, unsupervised methods fall short when labeled data is available.

**3. Supervised Learning: Classification Models**

**3.1 Preprocessing**

* Converted text labels into binary numeric values (e.g., spam = 1, ham = 0).
* Split dataset into **80% training** and **20% testing**.

**3.2 Word2Vec Vectorization**

* Used **pretrained Google News Word2Vec embeddings**.
* Each message was represented by the **average of its word vectors**.
* Custom function document\_vector(doc) computed average vectors for words in each message.

**4. Model Training & Evaluation**

**A. Random Forest Classifier**

* Model: RandomForestClassifier with 100 trees (n\_estimators=100).
* Evaluation: Used classification\_report to assess precision, recall, and F1-score.

**Results:**

* **Accuracy:** 95%
* **Ham Recall:** 1.00 (perfect ham detection)
* **Spam Recall:** 0.66 (missed one-third of spam)
* **Precision:** Above 0.95 for both classes

**Conclusion:** Strong on ham detection, but spam recall could be improved.

**B. XGBoost Classifier**

* Model: XGBClassifier from the XGBoost library.
* Followed standard training and evaluation process.

**Results:**

* **Accuracy:** 97%
* **Ham Recall:** 0.99
* **Spam Recall:** 0.78
* **F1-score (Spam):** 0.85

**Conclusion:** Best overall performance, particularly effective in detecting spam.

**C. Naive Bayes (GaussianNB)**

* Model: GaussianNB assuming normal distribution of features.
* Used the same Word2Vec vectors for training and testing.

**Results:**

* **Accuracy:** 74% (lowest)
* **Spam Precision:** 0.33 (many false positives)
* **Ham Recall:** 0.72

**Conclusion:** Weakest model; struggles especially with spam detection. Word2Vec may not suit GaussianNB well.

**5. Model Performance Comparison**

| **Metric** | **Random Forest** | **XGBoost** | **Naive Bayes** |
| --- | --- | --- | --- |
| Accuracy | 0.95 | 0.97 | 0.65 |
| Spam Precision | 0.98 | 0.94 | 0.33 |
| Spam Recall | 0.66 | 0.78 | 0.87 |
| Ham Precision | 0.95 | 0.97 | 0.97 |
| Ham Recall | 1.00 | 0.99 | 0.72 |
| F1-Score (Average) | 0.95 | 0.97 | 0.74 |

**Summary:**

* **Best Overall:** XGBoost – great balance of precision and recall.
* **Best on Ham:** Random Forest – perfect ham detection.
* **Worst Performer:** Naive Bayes – poor spam precision and low accuracy.

**6. Conclusion**

This project demonstrated a complete workflow for SMS message classification using both unsupervised and supervised methods. While clustering gives a useful overview, supervised models—especially XGBoost—offer the highest accuracy and reliability.

**Key Takeaways:**

* XGBoost is the top choice for spam filtering.
* Vectorization technique greatly impacts model performance.
* Choosing the right model and feature representation is crucial for high-quality text classification.

All experiments were run in **Google Colab** using **Python**, **Scikit-learn**, **Gensim**, and **XGBoost**.