**SMS Spam Detection Using Machine Learning – Project Report**

**🔹 Project Title:**

**SMS Spam Classification Using NLP & Naive Bayes**

**🔹 Objective:**

To develop a machine learning model that can classify SMS messages as **spam** or **ham (not spam)** using text preprocessing and natural language processing (NLP) techniques.

**🔹 Dataset Description:**

* **Source**: Public SMS spam dataset (UCI or Kaggle)
* **Rows**: 5572 SMS messages
* **Columns**:
  + label: Indicates whether the message is "ham" or "spam"
  + message: The actual SMS content

**🔹 Tools & Libraries Used:**

* **Python**
* **Pandas** – Data manipulation
* **Matplotlib / Seaborn** – Data visualization
* **Scikit-learn** – ML model, evaluation, vectorization
* **Regex, String** – Text cleaning

**🔹 Step-by-Step Workflow:**

**✅ Step 1: Load Data**

Loaded the dataset using pandas.read\_csv() and explored its structure using info() and head().

**✅ Step 2: Data Cleaning**

* Removed missing/null values.
* Checked class distribution (ham vs spam).
* Added message length column for basic EDA.

**✅ Step 3: Exploratory Data Analysis (EDA)**

* **Pie chart** and **countplot** to compare spam vs ham frequency.
* **Histogram** to compare message lengths.
* **WordCloud** to visualize frequent words in spam and ham messages.
* **Barplots** to show most common words in each category.

**✅ Step 4: Text Preprocessing**

* Converted all messages to lowercase.
* Removed punctuation and digits.
* Removed extra spaces.

Output stored in a new column: cleaned.

**✅ Step 5: Feature Extraction**

* Used **TF-IDF Vectorizer** to convert cleaned messages into numerical format.
* Limited features to **3000** most important words.

**✅ Step 6: Label Encoding**

* Converted ham to 0 and spam to 1 using .map().

**✅ Step 7: Train-Test Split**

* Split data into training and test sets (80% train, 20% test).

**✅ Step 8: Model Training**

* Used **Multinomial Naive Bayes** (ideal for text classification).
* Trained the model on the training data.

**✅ Step 9: Model Evaluation**

* Achieved **Accuracy**: ~0.97 – 0.99
* Displayed:
  + Confusion Matrix
  + Classification Report (Precision, Recall, F1-Score)

**📊 Key Results:**

* **High precision and recall** in spam detection.
* Balanced performance for both classes.
* Confusion Matrix and WordClouds enhanced interpretability.

**🔮 Future Scope:**

* Use advanced models like **Logistic Regression**, **SVM**, or **XGBoost**.
* Add more features (e.g., special character frequency, capital letter ratio).
* Deploy as a **web app** using Flask or Streamlit.
* Add a feedback loop for **active learning** and **continual improvement**.

**📌 Conclusion:**

This project successfully demonstrates how to build a spam classifier using simple NLP and machine learning techniques. It showcases the full pipeline from data preprocessing to evaluation — making it an excellent foundational project in data science and NLP.