**Structure of Your Project Report / Notebook**

**🧩 1. Title & Introduction**

* **Project**: SMS Spam Detection using NLP & Machine Learning
* **Goal**: Build a model to classify messages as spam or ham, and understand key indicators using explainable AI techniques.

**📊 2. Exploratory Data Analysis (EDA)**

* Count of spam vs ham
* Message length distributions
* Word cloud for spam vs ham

✅ **Code:**

python

CopyEdit

df['label'].value\_counts().plot(kind='bar', title='Spam vs Ham Count')

# Length distribution

df['message\_length'] = df['message'].apply(len)

df['message\_length'].hist(by=df['label'], bins=50, figsize=(12,4), color='skyblue')

**🧼 3. Text Preprocessing + TF-IDF**

python

CopyEdit

from sklearn.feature\_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(stop\_words='english', max\_df=0.95)

X = df['message']

y = df['label\_num']

# Train-test split + reset index

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

X\_train = X\_train.reset\_index(drop=True)

X\_test = X\_test.reset\_index(drop=True)

y\_train = y\_train.reset\_index(drop=True)

y\_test = y\_test.reset\_index(drop=True)

X\_train\_tfidf = vectorizer.fit\_transform(X\_train)

X\_test\_tfidf = vectorizer.transform(X\_test)

**🧠 4. Model Building & Evaluation**

Train both models and compare:

python

CopyEdit

from sklearn.linear\_model import LogisticRegression

from sklearn.naive\_bayes import MultinomialNB

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report

lr\_model = LogisticRegression()

nb\_model = MultinomialNB()

lr\_model.fit(X\_train\_tfidf, y\_train)

nb\_model.fit(X\_train\_tfidf, y\_train)

# Predict

y\_pred\_lr = lr\_model.predict(X\_test\_tfidf)

y\_pred\_nb = nb\_model.predict(X\_test\_tfidf)

# Evaluation

print("Logistic Regression:\n", classification\_report(y\_test, y\_pred\_lr))

print("Naive Bayes:\n", classification\_report(y\_test, y\_pred\_nb))

🎯 **Select best model** based on F1-score (usually logistic regression does better here).

**📌 5. Insights**

**🔹 Logistic Regression Coefficients**

python

CopyEdit

feature\_names = vectorizer.get\_feature\_names\_out()

coefficients = lr\_model.coef\_[0]

# Top spam/ham words

top\_spam = sorted(zip(feature\_names, coefficients), key=lambda x: x[1], reverse=True)[:20]

top\_ham = sorted(zip(feature\_names, coefficients), key=lambda x: x[1])[:20]

**🔹 Word Clouds**

python

CopyEdit

from wordcloud import WordCloud

import matplotlib.pyplot as plt

spam\_words = ' '.join(df[df['label\_num'] == 1]['message'])

WordCloud(width=800, height=400).generate(spam\_words).to\_image()

**🔹 Optional: SHAP (for advanced visual explainability)**

**🧳 6. Packaging for Submission**

* Export model:

python

CopyEdit

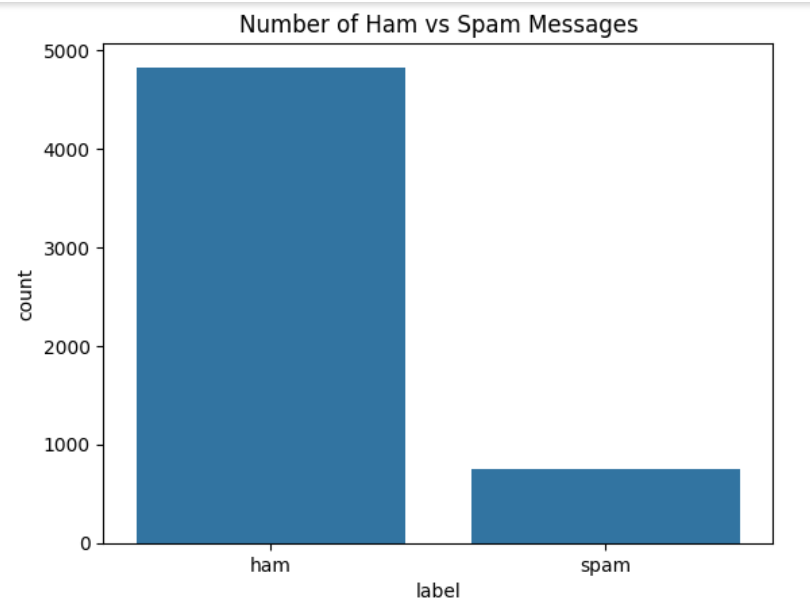
import joblib

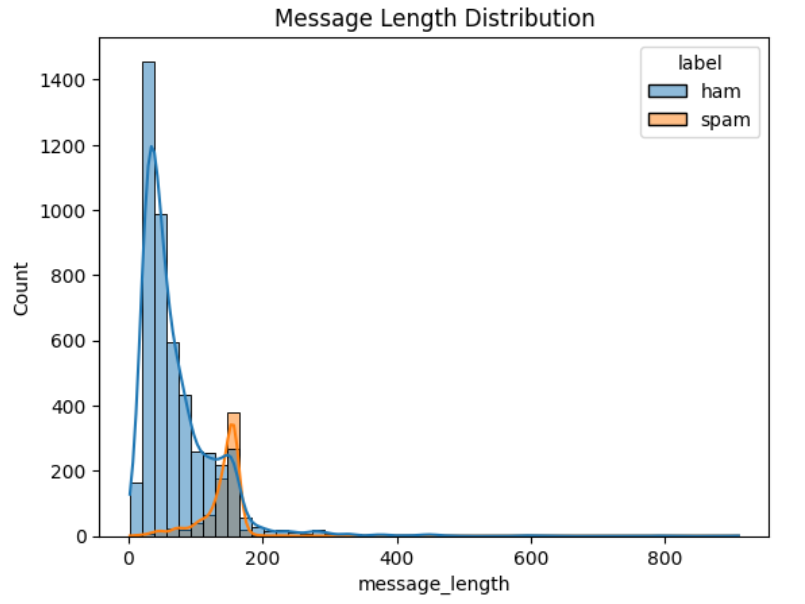
joblib.dump(lr\_model, "spam\_model.pkl")

joblib.dump(vectorizer, "tfidf\_vectorizer.pkl")

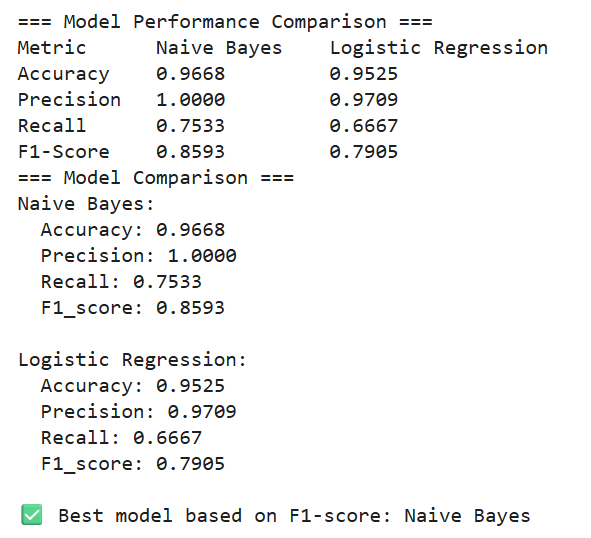
* Save notebook as PDF or HTML
* Add README with:
  + 📌 Problem Statement
  + 💡 Summary of Results
  + 📈 Accuracy / F1 Score
  + 🧠 Key Insights

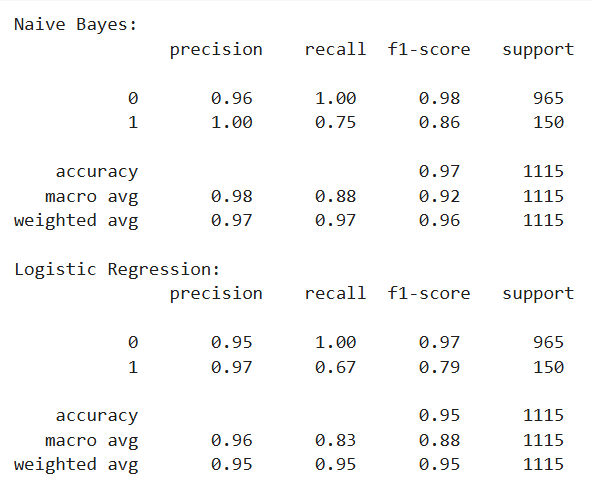
Initial data anyalyis

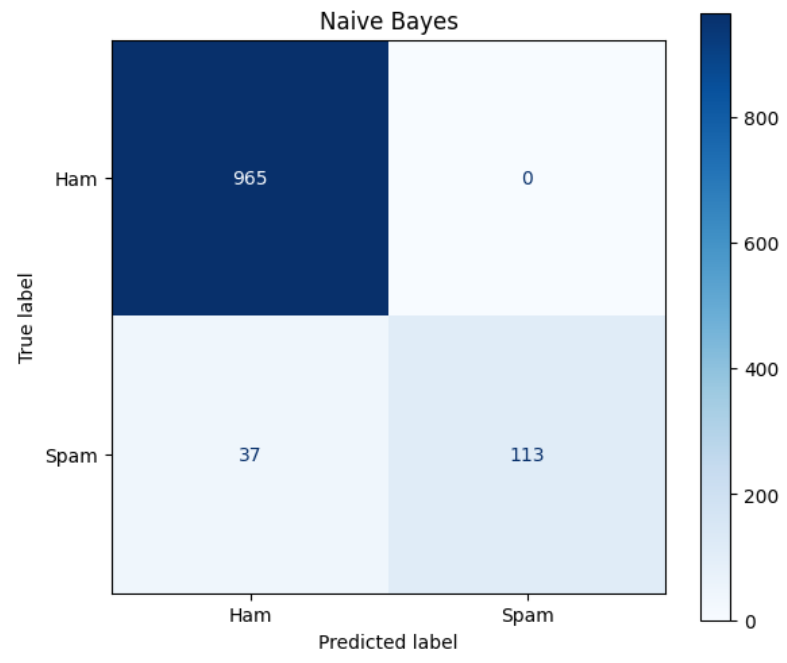


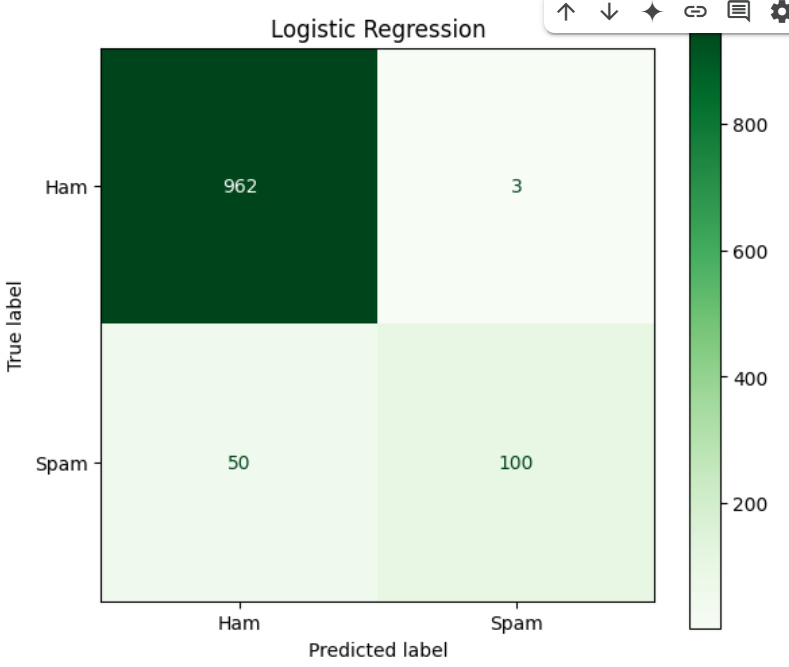


Model selection









**Insights**

