Table of Contents

Week 4 Project: Spam Email Detection

Objective

Email spam detection is a crucial task in natural language processing (NLP) and machine learning. This project aims to build a model that can classify emails as spam or ham (non-spam) using Logistic Regression and Support Vector Machine (SVM).

The dataset used contains 5,572 emails, with a significant class imbalance: 4825 ham and 747 spam emails. To address this imbalance, resampling techniques were applied.

Data Preprocessing

1. Loading Data

The dataset `email_spam_detection.csv` was loaded using `pandas` with `latin-1` encoding to handle special characters.

2. Handling Missing Values

No missing values were found in the `Category` column.

```
data.isna().sum()
```

3. Class Distribution

Initial counts showed an imbalanced dataset:

- Ham: 4825
- Spam: 747

To balance the dataset, **resampling** was applied:

```
from sklearn.utils import resample

ham = data[data['Category']=='ham']
spam = data[data['Category']=='spam']

resample_spam = resample(spam,
                         replace=True,
                         n_samples=len(ham),
                         random_state=42)
```

```
data = pd.concat([resample_spam, ham])
```

After resampling, the dataset was balanced:

- Ham: 4825
- Spam: 4825

Feature Engineering

1. Encoding Target

The target column was encoded as:

- Ham → 0
- Spam → 1

```
data['Category'] = data['Category'].map({'ham': 0, 'spam': 1})
```

2. Train-Test Split

The dataset was split into training and testing sets (70% train, 30% test).

3. Text Vectorization

Text data was converted to numerical features using **Bag-of-Words** representation with CountVectorizer.

```
from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer(stop_words='english', max_features=3000)
x_train_vec = vectorizer.fit_transform(x_train)
x_test_vec = vectorizer.transform(x_test)
```

Model Training

1. Logistic Regression

A Logistic Regression model was trained on the vectorized data:

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter=1000)
model.fit(x_train_vec, y_train)
y_pred = model.predict(x_test_vec)
```

2. Support Vector Machine (SVM)

A linear SVM model was also trained for comparison:

```
from sklearn.svm import SVC

svc_model = SVC(kernel='linear')
svc_model.fit(x_train_vec, y_train)
y_pred_svc = svc_model.predict(x_test_vec)
```

Model Evaluation

1. Logistic Regression Results

- Accuracy: **99.52%**
- Classification metrics (Precision, Recall, F1-score) were all near **1.0**.

```
--------------LOGISTIC REGRESSION RESULTS--------------

Accuracy:  0.9951640759930915

Classification Report:                precision    recall  f1-score   support

           0       1.00      1.00      1.00      1433
           1       1.00      1.00      1.00      1462

    accuracy                           1.00      2895
   macro avg       1.00      1.00      1.00      2895
weighted avg       1.00      1.00      1.00      2895


Confusion Matrix:  [[1426    7]
 [    7 1455]]
```

2. SVM Results

- Accuracy: **99.59%**
- Classification metrics were similarly excellent.

```
--------------SVM RESULTS--------------

Accuracy:  0.9958549222797928

Classification Report:                precision    recall  f1-score   support

           0       1.00      1.00      1.00      1433
           1       1.00      1.00      1.00      1462

    accuracy                           1.00      2895
   macro avg       1.00      1.00      1.00      2895
weighted avg       1.00      1.00      1.00      2895


Confusion Matrix:  [[1426    7]
 [    5 1457]]
```

Both models performed exceptionally well due to the balanced dataset and robust feature representation.

Testing with New Emails

The trained Logistic Regression model was tested on a new email:

```
input = ["Congratulations, you won free 100 Bitcoins in a lottery"]
new_vec = vectorizer.transform(input)
result = model.predict(new_vec)

if result == 0:
    print("Not Spam")
else:
    print("Spam")


Output: Spam
```

Conclusion

- Both Logistic Regression and SVM achieved near-perfect accuracy on this dataset.
- Balancing the dataset using **resampling** improved model performance.
- The Bag-of-Words approach effectively converted text into features suitable for ML models.

- This project demonstrates a practical approach to **email spam detection**, which is essential for improving email security and filtering unwanted messages.