

```

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import ComplementNB
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report

# 1. Load and Split the data
data = pd.read_csv('spam_ham_dataset.csv')
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5171 entries, 0 to 5170
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Unnamed: 0    5171 non-null   int64
1   label         5171 non-null   object
2   text          5171 non-null   object
3   label_num     5171 non-null   int64
dtypes: int64(2), object(2)
memory usage: 161.7+ KB

df = data[['label', 'text']].rename(columns={'label': 'Category',
                                             'text': 'Message'})

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5171 entries, 0 to 5170
Data columns (total 2 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Category    5171 non-null   object
1   Message     5171 non-null   object
dtypes: object(2)
memory usage: 80.9+ KB

x_train, x_test, y_train, y_test = train_test_split(
    df['Message'],
    df['Category'],
    test_size=0.2,
    random_state=42,
    stratify=df['Category']
)

# 2. Create and Train the Pipeline
spam_filter_pipeline = Pipeline([
    ('tfidf', TfidfVectorizer(stop_words='english')),
    ('classifier', ComplementNB(alpha=0.1))
])

```

```
spam_filter_pipeline.fit(x_train, y_train)

Pipeline(steps=[('tfidf', TfidfVectorizer(stop_words='english')),
                 ('classifier', ComplementNB(alpha=0.1))])
```

3. Evaluate the Model

```
predictions = spam_filter_pipeline.predict(x_test)
```

```
print(f"Classification Report:\n {classification_report(y_test,
predictions)}")
```

Classification Report:

	precision	recall	f1-score	support
ham	0.98	0.98	0.98	735
spam	0.96	0.96	0.96	300
accuracy			0.97	1035
macro avg	0.97	0.97	0.97	1035
weighted avg	0.97	0.97	0.97	1035

4. Test with New Emails

```
print("\n Testing with new emails:")
new_emails = ["Congratulations! You've won a $1,000 Walmart gift card.
Go to http://bit.ly/claim-yours to claim now.",
"Hi team, are we still on for the 3 PM meeting today? Please
confirm.",
"URGENT: Your account has been compromised. Please click here to reset
your password immediately!"]
predictions_new = spam_filter_pipeline.predict(new_emails)
for email, prediction in zip(new_emails, predictions_new):
    print(f"-> Email: \"{email[:60]}...\" \n Prediction:
**{prediction.upper()}**\n")
```

Testing with new emails:

```
-> Email: "Congratulations! You've won a $1,000 Walmart gift card. Go
t..."
```

```
Prediction: **SPAM**
```

```
-> Email: "Hi team, are we still on for the 3 PM meeting today? Please
..."
```

```
Prediction: **HAM**
```

```
-> Email: "URGENT: Your account has been compromised. Please click
here..."
```

```
Prediction: **SPAM**
```