```
from sklearn.datasets import fetch_20newsgroups

# Load the dataset
newsgroups_data = fetch_20newsgroups(subset='train', remove=('headers', 'footers', 'quotes'))
print(newsgroups_data.data[0])
print(newsgroups_data.target_names[0])
print(newsgroups_data.target[0])
print(newsgroups_data.filenames[0])
print(f"Loaded {len(newsgroups_data.data)} documents with {len(newsgroups_data.target_names)} categories.")
```

```
I was wondering if anyone out there could enlighten me on this car I saw
the other day. It was a 2-door sports car, looked to be from the late 60s/
early 70s. It was called a Bricklin. The doors were really small. In addition,
the front bumper was separate from the rest of the body. This is
all I know. If anyone can tellme a model name, engine specs, years
of production, where this car is made, history, or whatever info you
have on this funky looking car, please e-mail.
alt.atheism
7
/root/scikit_learn_data/20news_home/20news-bydate-train/rec.autos/102994
Loaded 11314 documents with 20 categories.
```

+ Code + Text

```
import spacy

nlp = spacy.load("en_core_web_sm")


def preprocess(text):
  doc = nlp(text)
  tokens = [token.text for token in doc if not token.is_stop and not token.is_punct]
  return ' '.join(tokens)



preprocess_data = [preprocess(doc) for doc in newsgroups_data.data]


print("Newsgroup Data:- ", newsgroups_data.data[0])
print("----------------------------------------")
print("Preprocessed data:- ", preprocess_data[0])
```

```
Newsgroup Data:-  I was wondering if anyone out there could enlighten me on this car I saw
the other day. It was a 2-door sports car, looked to be from the late 60s/
early 70s. It was called a Bricklin. The doors were really small. In addition,
the front bumper was separate from the rest of the body. This is
all I know. If anyone can tellme a model name, engine specs, years
of production, where this car is made, history, or whatever info you
have on this funky looking car, please e-mail.
----------------------------------------
Preprocessed data:-  wondering enlighten car saw
 day 2 door sports car looked late 60s/
 early 70s called Bricklin doors small addition
 bumper separate rest body
 know tellme model engine specs years
 production car history info
 funky looking car e mail
```

```
## Vecotrization

from sklearn.feature_extraction.text import TfidfVectorizer


vectorizer = TfidfVectorizer(max_features=5000)
X = vectorizer.fit_transform(preprocess_data)
y = newsgroups_data.target


from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train the classifier
nb_model = MultinomialNB()
nb_model.fit(X_train, y_train)

# Make predictions
y_pred = nb_model.predict(X_test)
```

```
# Evaluate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"Naive Bayes Accuracy: {accuracy:.4f}")
```

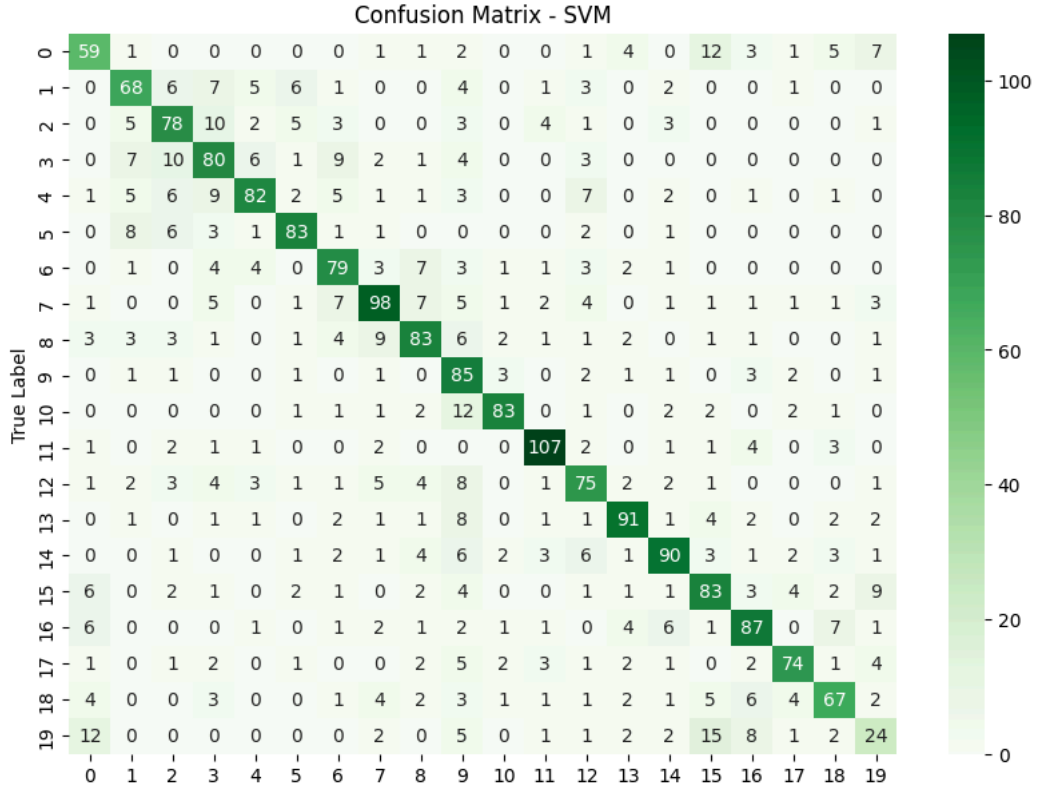
```
Naive Bayes Accuracy: 0.6969
```

```
from sklearn.svm import LinearSVC

# Train the classifier
svm_model = LinearSVC()
svm_model.fit(X_train, y_train)

# Make predictions
y_pred_svm = svm_model.predict(X_test)

# Evaluate accuracy
accuracy_svm = accuracy_score(y_test, y_pred_svm)
print(f"SVM Accuracy: {accuracy_svm:.4f}")
```

```
SVM Accuracy: 0.6964
```

```
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Generate classification report
print("Naive Bayes Classification Report:\n", classification_report(y_test, y_pred))
print("SVM Classification Report:\n", classification_report(y_test, y_pred_svm))

# Confusion Matrix for SVM
conf_matrix = confusion_matrix(y_test, y_pred_svm)
plt.figure(figsize=(10, 7))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Greens")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix - SVM")
plt.show()
```

```
Naive Bayes Classification Report:
              precision    recall  f1-score   support

           0       0.66      0.44      0.53        97
           1       0.62      0.69      0.65       104
           2       0.71      0.62      0.66       115
           3       0.55      0.70      0.62       123
           4       0.85      0.60      0.70       126
           5       0.70      0.85      0.77       106
           6       0.66      0.76      0.71       109
           7       0.74      0.64      0.69       139
           8       0.75      0.75      0.75       122
           9       0.51      0.87      0.64       102
          10       0.86      0.85      0.86       108
          11       0.83      0.87      0.85       125
          12       0.73      0.62      0.67       114
          13       0.76      0.75      0.75       119
          14       0.89      0.73      0.80       127
          15       0.52      0.90      0.66       122
          16       0.73      0.77      0.75       121
          17       0.75      0.75      0.75       102
          18       0.81      0.50      0.62       107
          19       0.33      0.01      0.03        75

    accuracy                           0.70      2263
   macro avg       0.70      0.68      0.67      2263
weighted avg       0.71      0.70      0.69      2263

SVM Classification Report:
              precision    recall  f1-score   support

           0       0.62      0.61      0.61        97
           1       0.67      0.65      0.66       104
           2       0.66      0.68      0.67       115
           3       0.61      0.65      0.63       123
           4       0.77      0.65      0.71       126
           5       0.78      0.78      0.78       106
           6       0.67      0.72      0.70       109
           7       0.73      0.71      0.72       139
           8       0.70      0.68      0.69       122
           9       0.51      0.83      0.63       102
          10       0.86      0.77      0.81       108
          11       0.84      0.86      0.85       125
          12       0.65      0.66      0.65       114
          13       0.80      0.76      0.78       119
          14       0.76      0.71      0.73       127
          15       0.64      0.68      0.66       122
          16       0.71      0.72      0.72       121
          17       0.80      0.73      0.76       102
          18       0.71      0.63      0.66       107
          19       0.42      0.32      0.36        75

    accuracy                           0.70      2263
   macro avg       0.70      0.69      0.69      2263
weighted avg       0.70      0.70      0.70      2263
```



Predicted Label

```
print(newsgroups_data.target_names)
```

```
['alt.atheism', 'comp.graphics', 'comp.os.ms-windows.misc', 'comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware', 'comp.w
```