```
import pandas as pd
import re
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report

# Load the dataset
file_path = '/content/emails.csv'
emails_df = pd.read_csv(file_path)

# Text preprocessing function
def preprocess_text(text):
    text = text.lower()  # Convert to lowercase
    text = re.sub(r'\d+', '', text)  # Remove digits
    text = re.sub(r'[^\w\s]', '', text)  # Remove punctuation
    text = re.sub(r'\s+', ' ', text).strip()  # Remove extra spaces and strip tex
    return text

# Apply the text preprocessing function to the email texts
emails_df['cleaned_text'] = emails_df['text'].apply(preprocess_text)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    emails_df['cleaned_text'], emails_df['spam'], test_size=0.2, random_state=42
)

# Feature extraction using TF-IDF
vectorizer = TfidfVectorizer(stop_words='english')
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

# Train a Naive Bayes classifier
nb_classifier = MultinomialNB()
nb_classifier.fit(X_train_tfidf, y_train)

# Predict and evaluate the Naive Bayes model
y_pred_nb = nb_classifier.predict(X_test_tfidf)
nb_accuracy = accuracy_score(y_test, y_pred_nb)
nb_report = classification_report(y_test, y_pred_nb)

# Train a Support Vector Machine classifier
svm_classifier = SVC()
svm_classifier.fit(X_train_tfidf, y_train)

# Predict and evaluate the SVM model
y_pred_svm = svm_classifier.predict(X_test_tfidf)
svm_accuracy = accuracy_score(y_test, y_pred_svm)
svm_report = classification_report(y_test, y_pred_svm)
```

```
# Print results
```

Start coding or generate with AI.

```
print(nb_report)
```