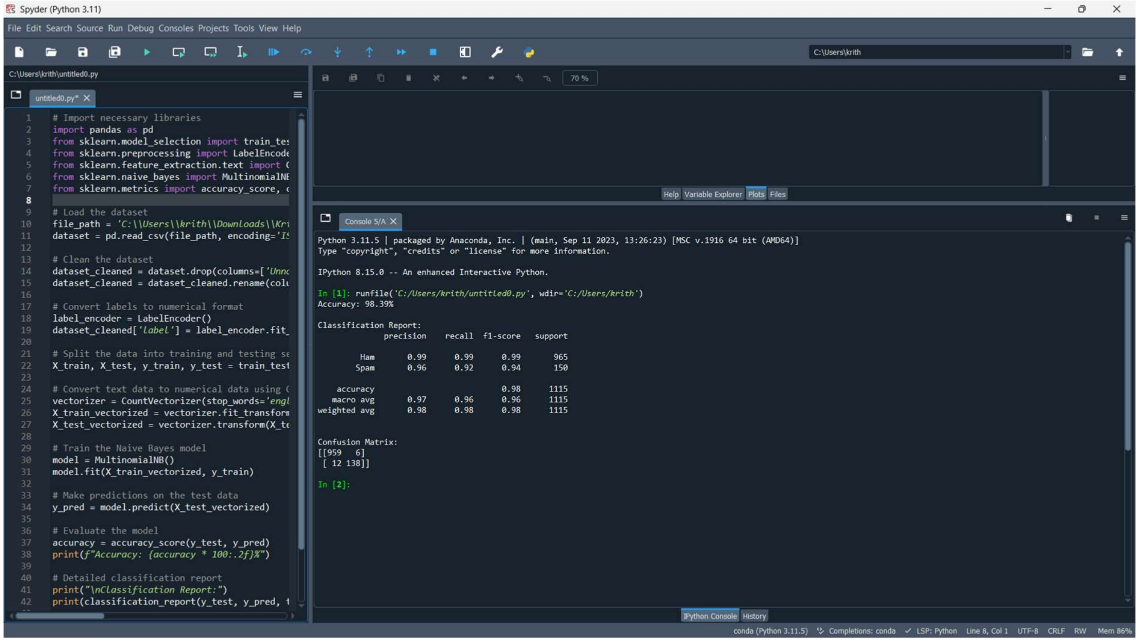
# TASK 3:EMAIL SPAM DETECTION WITH ML

- **Program:**

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

# Load the dataset
file_path = 'C:\\Users\\krith\\Downloads\\   BV   \\spam.csv'
dataset = pd.read_csv(file_path, encoding='ISO-8859-1')

# Clean the dataset
dataset_cleaned = dataset.drop(columns=['Unnamed: 2', 'Unnamed: 3',
'Unnamed: 4'])
dataset_cleaned = dataset_cleaned.rename(columns={'v1': 'label', 'v2':
'message'})

# Convert labels to numerical format
label_encoder = LabelEncoder()
dataset_cleaned['label'] =
label_encoder.fit_transform(dataset_cleaned['label'])  # 0 for ham, 1 for spam

# Split the data into training and testing sets
```

```
X_train, X_test, y_train, y_test = train_test_split(dataset_cleaned['message'],
dataset_cleaned['label'], test_size=0.2, random_state=42)


# Convert text data to numerical data using CountVectorizer
vectorizer = CountVectorizer(stop_words='english')
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)


# Train the Naive Bayes model
model = MultinomialNB()
model.fit(X_train_vectorized, y_train)


# Make predictions on the test data
y_pred = model.predict(X_test_vectorized)


# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")


# Detailed classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred, target_names=['Ham', 'Spam']))


# Confusion matrix
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

- **Output:**


Spyder (Python 3.11)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\krith
C:\Users\krith\untitled0.py
untitled0.py*
70 %
Help Variable Explorer Plots Files
Console 5/A
IPython Console History
conda (Python 3.11.5) Completions: conda LSP: Python Line 8, Col 1 UTF-8 CRLF RW Mem 86%

```python
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_tes
from sklearn.preprocessing import LabelEncode
from sklearn.feature_extraction.text import (
from sklearn.naive_bayes import MultinomialNE
from sklearn.metrics import accuracy_score, c

# Load the dataset
file_path = 'C:\\Users\\krith\\Downloads\\Kri
dataset = pd.read_csv(file_path, encoding='IS

# Clean the dataset
dataset_cleaned = dataset.drop(columns=['Unnc
dataset_cleaned = dataset_cleaned.rename(colu

# Convert labels to numerical format
label_encoder = LabelEncoder()
dataset_cleaned['label'] = label_encoder.fit_

# Split the data into training and testing se
X_train, X_test, y_train, y_test = train_test

# Convert text data to numerical data using (
vectorizer = CountVectorizer(stop_words='engl
X_train_vectorized = vectorizer.fit_transform
X_test_vectorized = vectorizer.transform(X_te

# Train the Naive Bayes model
model = MultinomialNB()
model.fit(X_train_vectorized, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test_vectorized)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")

# Detailed classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred, t
```

```
Python 3.11.5 | packaged by Anaconda, Inc. | (main, Sep 11 2023, 13:26:23) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.15.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/krith/untitled0.py', wdir='C:/Users/krith')
Accuracy: 98.39%

Classification Report:
              precision    recall  f1-score   support

         Ham       0.99      0.99      0.99       965
        Spam       0.96      0.92      0.94       150

    accuracy                           0.98      1115
   macro avg       0.97      0.96      0.96      1115
weighted avg       0.98      0.98      0.98      1115


Confusion Matrix:
[[959   6]
 [ 12 138]]

In [2]:
```