CONCLUSION AND FUTURE WORKS :

In conclusion, developing a smart AI-powered spam classifier involves leveraging advanced machine learning algorithms for effective detection. Future work could focus on enhancing model robustness, exploring real-time adaptability, and refining the system's ability to handle evolving spam tactics. Additionally, user feedback integration and continuous model updates would contribute to sustained effectiveness in combating spam

SOURCE CODE :

```python
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics

# Load your dataset (replace 'your_dataset.csv' with your actual dataset)
data = pd.read_csv('your_dataset.csv')

# Preprocess the data
# (You might need to clean and preprocess your text data based on your dataset)

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

# Convert text data to numerical features using CountVectorizer
vectorizer = CountVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)
```

```
# Train a Naive Bayes classifier
classifier = MultinomialNB()
classifier.fit(X_train_vectorized, y_train)

# Make predictions
predictions = classifier.predict(X_test_vectorized)

# Evaluate the model
accuracy = metrics.accuracy_score(y_test, predictions)
precision = metrics.precision_score(y_test, predictions)
recall = metrics.recall_score(y_test, predictions)
f1_score = metrics.f1_score(y_test, predictions)

# Print evaluation metrics
print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"F1 Score: {f1_score:.2f}")
```