Text Classification for Spam Detection using SVM

Import necessary libraries

```
import nltk
from nltk.corpus import stopwords
import re
import subprocess
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
```

Example to test functions

```
test_text = "Hi, Check our website http://example.com for daily coupons! Don't miss promotions."
```

Function to remove stop words

```
stop_words = stopwords.words('english') # Outside the function for code optimization

def stop_words_removal(text):
    text = text.lower()
    tokens = [token for token in text.split() if token not in stop_words]
    text_no_sw = " ".join(tokens)
    return text_no_sw

# Test
print("input:",test_text)
test_text = stop_words_removal(test_text)
print("output:",test_text)
```

Function to remove urls, punctuations and numbers

```
url_pattern = re.compile(r'http\S+|www\S+')
cleaning_pattern = re.compile(r'[^\w\s]|[\d]')

def text_no_urls_puncs_nums(text):
    text = text.lower()
    # Remove URLs
    text = url_pattern.sub('', text)
    # Remove punctuations and numbers
    text = cleaning_pattern.sub('', text)
    return text

# Test
```

```
print("input:",test_text)
test_text = text_no_urls_puncs_nums(test_text)
print("output:",test_text)
```

Function to lemmatize words

```
# Download and unzip wordnet
try:
    nltk.data.find('wordnet.zip')
except:
    nltk.download('wordnet', download_dir='/kaggle/working/')
    command = "unzip /kaggle/working/corpora/wordnet.zip -d /kaggle/working/corpora"
    subprocess.run(command.split())
    nltk.data.path.append('/kaggle/working/')

# Now you can import the NLTK resources as usual
from nltk.corpus import wordnet


from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

# Lemmatization
def lemmatize_text(text):
    text = text.lower()
    lemmatized_tokens = [lemmatizer.lemmatize(token) for token in nltk.word_tokenize(text)]
    lemmatized_text = ' '.join(lemmatized_tokens)
    return lemmatized_text

# Test
print("input:",test_text)
test_text = lemmatize_text(test_text)
print("output:",test_text)
```

Function to clean text using previous functions

```
# Merge the previous functions
def clean_text(text):
    clean_text = lemmatize_text(text_no_urls_puncs_nums(stop_words_removal(text)))
    # Remove one letter tokens
    clean_text = " ".join([token for token in clean_text.split() if len(token)>1])
    return clean_text

# Test
test_text_2 = "Hi, Check our website http://example.com for daily coupons! Don't miss
promotions."
print("input:",test_text_2)
test_text_2 = clean_text(test_text_2)
print("output:",test_text_2)
```

Loading Dataset

```
# Read Dataset
```

```
data_orig =
pd.read_csv('/kaggle/input/sms-spam-collection-dataset/spam.csv',encoding='latin1')
Rename columns
data.columns = ['class', 'text']
data
Convert class column to spam=1 ham=0
data['class'] = data['class'].apply(lambda x: 1 if x == 'spam' else 0)
data
Clean Text
data['text'] = data['text'].apply(lambda x: clean_text(x))
data.head()
Split data to train and test parts
x_train, x_test, y_train, y_test = train_test_split(data['text'], data['class'], test_size=0.2,
random_state=20)
# Check Test samples percent
len(x_test)/len(data['text']
Transform text to numerical data that SVM can work with
cv = CountVectorizer()
x_train = cv.fit_transform(x_train)
x_test = cv.transform(x_test)
Train the SVM classifier using the training data
model = SVC(random_state = 20)
model.fit(x_train, y_train)
Calculate the accuracy of the SVM model on the test data
model.score(x_test,y_test)
Confusion Matrix
y_pred = model.predict(x_test)
y_pred
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt="d")
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()


# Verify Confusion matrix output

check_res = pd.DataFrame(y_test)
check_res['y_pred'] = y_pred

count = 0
for index, row in check_res.iterrows():
    if row['class']==1 and row['y_pred']==0:
        count += 1

print(count)
```