Classification

Domain: Email and messaging

Business Context:

Text messaging has revolutionized communication in the modern world. However, it comes with its own issues. Most of us deal with so many unwanted promotional messages which are of very little importance. If mobile companies could have an inbuilt feature which would be able to identify such spam messages and take necessary actions to block, then there would be a significant satisfaction among the consumers.

Here, we have a set of SMS tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,574 messages, tagged as ham (legitimate) or spam.

Objective:

We will try to build a classifier using Naive Bayes algorithm which will be able to classify text messages as spam or ham.

Dataset description:

Dataset has 2 columns and 5572 rows

v1 - class (label)
v2 - sms (text)

Steps to be taken:

1. Read the dataset and perform EDA
    a. Rename column names to sms, and label
    b. Check shape of dataset and info.
    c. Check value count of label and comment on its frequency. Is the class balanced?
    d. Check summary statistics. Observe the unique values for both label and sms
    e. Check duplicate values. [Hint - df.duplicated()]
    f. Remove duplicate rows. [Hint - df.drop_duplicates()]
    g. Check the length of each message and append it in the dataframe as a column.
    h. Check distribution of the length column and mention your comments.

2. Prepare data for modelling.
    a. Change label into categories [Hint - ham = 0, spam =1]

b. Compute word count for each sms and save it in the dataframe. [Hint - use split(“ “)] i.e -> df['word_count'] = df['sms'].apply(lambda x: len(str(x).split(" ")))
c. Convert all the words into lower case
d. Remove punctuations
e. Remove stop words. Use stopwords from nltk.
Ref - https://www.nltk.org/
https://pythonprogramming.net/stop-words-nltk-tutorial/

f. Try removing frequently used words and rarely used words. [Hint - compute frequency of the words and remove top 10 frequently used words]
g. Use either tokenization or stemming to get the root words.
[Hint - from nltk.stem import PorterStemmer]
https://www.nltk.org/_modules/nltk/stem/porter.html

3. Model Building
   a. Split the data into train and testing set
   b. Create document term matrix using either count vectorizer or tfidfvectorizer
https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html
   c. Fit Naive Bayes model on the training data
   d. Fit other classification models like logistic regression, svm, and check the performance.
   e. Predict outcome for testing set
   f. Evaluate the model

4. Mention your conclusion and inferences.

Learning Outcomes:

- Naive Bayes
- Exploratory Data Analysis
- Text Preprocessing
- NLTK
- Countvectorizer
- Confusion Matrix