Project Name : SMS Spam Classifier (Natural Language Processing)

Problem Statement : To build a prediction model which classifies which text is spam.

Dataset link : https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection

Context :

The SMS Spam Collection is 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 acording being ham (legitimate) or spam.

Libraries :

- Importing the dataset.
- Importing pandas library for reading the csv file adding a seperator '\t' it is basically used to seperate the the tab seperated values in the dataset & it will divide the column into two columns Independent feature(input) & Dependent feature(output).

Procedure :

In the given dataset the values are seperated by '\t' but the column names are not assigned to the seperated values inside the dataset. The unassigned columns inside the dataset are assigned the name as "label"(Dependent variable) & "message"(Independent variable).

Data Cleaning & Data Preprocessing :

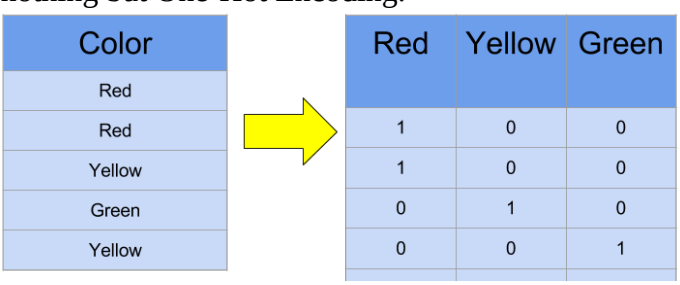
In this step we need to eliminate the stop words which are in the dataset & removing all the unnecessary words which are present inside the dataset. The dataset can be cleaned by removing. Importing the regular expression library to remove the stop words. Importing the nltk library for processing the data. Importing the stemmer as PortStemmer to remove all the punctuation inside the dataset. Initializing the PortStemmer. Creating an empty list named corpus because after all the data preprocessing of the messages all the processed data will be stored inside the corpus

Creating a for loop and iterating through the 5572 messages with the help of regular expressions we are going to remove all the stopwords from the dataset except a-z & A-z and along with it we are passing messages of messages of i(ie. One by one). After removing the stop words we will be converting the data into lower case and we will be splitting each and every sentences where i'll be getting list of words. Writing a list comprehension, for word in review if that word is not in stopwords of english file we are going to take that word and we are going to apply the stemming process after applying the stemming process we will get a base form of words. We will get the list of words inside the review variable and we are going to join them and appending them into corpus.

Creating Bag Of Words :

Bag of words is nothing but document matrix with respect to the word. Basically the words are converted into vector's(ie. 1's and 0's). Now from sklearn's feature extraction.text we will now import CountVectorizer

CountVectorizer is nothing but One-Hot Encoding.



Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

Next after initializing the count vectorizer we will then fit transform the newly created corpus data & we'll convert into an array.

Now converting the categorical label column into binary because our model does not understands ham & spam. We will be converting this categorical column into binary by using pandas buy applying pd.get_dummies(message['label']).

Here in y column ham and spam columns are converted into the dummie variaibles. In ham column the 1 is represented as ham and 0 is represented as spam and similarly for the spam column. We can eliminate one column you can take any column the spam or the ham column because one column can specify both the information ie. If it is 0 it basically specifies that it is ham & 1 specifies that it is spam. You dont have to use two categorical features you can use one categorical feature.

Train Test Splitting :

In this step we'll do train test split because we need to train our model. So for training our model we need to import train test split from sklearn and model selection. We will be using 80% for training the data and 20% for testing the data.

Training our model using naive bayes classifier :

Naice baye's classifier works well on the nlp problem as we will not get good accuracy but in order to get good accuracy we can fine tune it. It is basically a classification technique which completely works on probabilities

Confusion Matrix :

The confusion matrix is basically used in the classification problems to check the. Confusion matrix is available inside sklearn module under metrices.

Accuracy Score :

Calculating the accuracy the accuracy score of this particular model is 98% so we've done a good job.