Data Collection & Preparation

ML depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So this section allows you to download the required dataset.

Collect The Dataset

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the dataset.

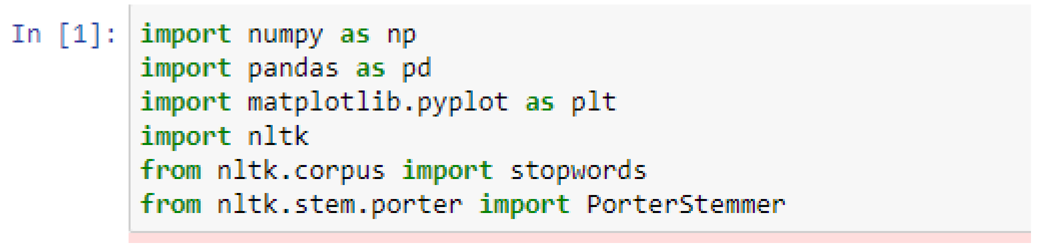
Link: https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualisation techniques and some analysing techniques.

Note: There are a number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

Importing The Libraries

Import the necessary libraries as shown in the image. (optional) Here we have used visualisation style as fivethirtyeight.

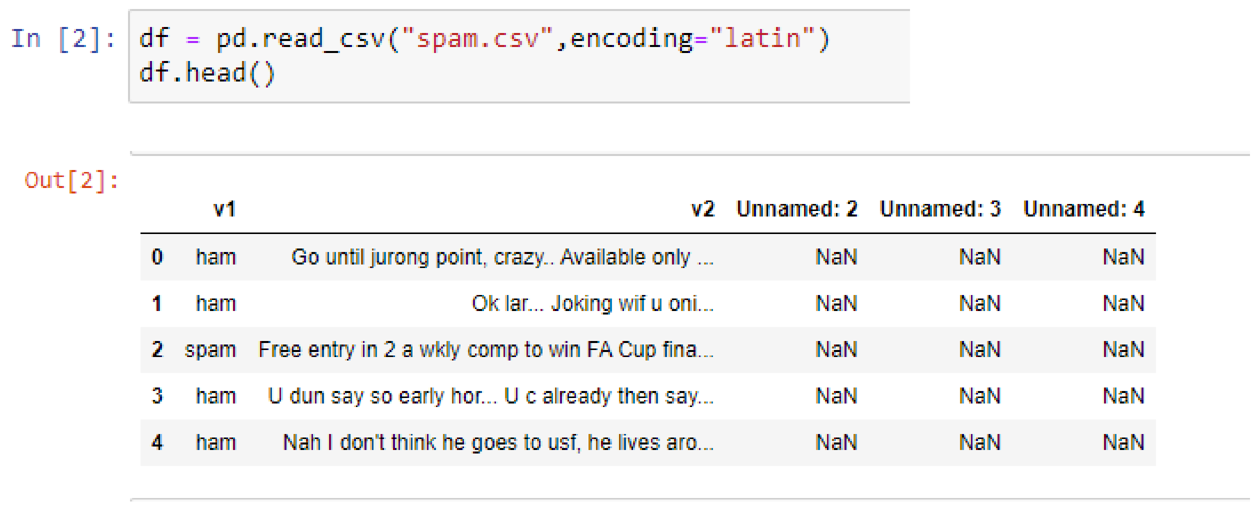


```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import nltk
        from nltk.corpus import stopwords
        from nltk.stem.porter import PorterStemmer
```

Read The Dataset

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas.

In pandas we have a function called read_csv() to read the dataset. As a parameter we have to give the directory of the csv file.



```
In [2]: df = pd.read_csv("spam.csv",encoding="latin")
        df.head()
```

Out[2]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

Data Preparation

As we have understood how the data is, let's pre-process the collected data.

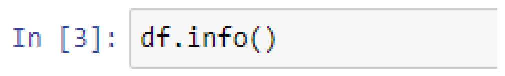
The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

- Handling missing values
- Handling categorical data
- Handling Imbalance Data

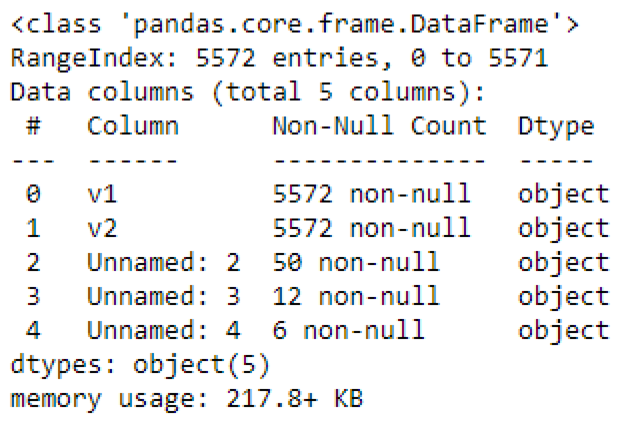
Note: These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

Handling Missing Values

- Let’s find the shape of our dataset first. To find the shape of our data, the df.shape method is used. To find the data type, df.info() function is used.

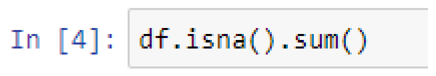


```
In [3]: df.info()
```

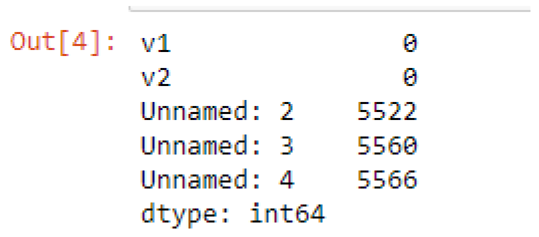


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

- For checking the null values, df.isnull() function is used. To sum those null values we use .sum() function. From the below image we found that there are no null values present in our dataset. So we can skip handling the missing values step.



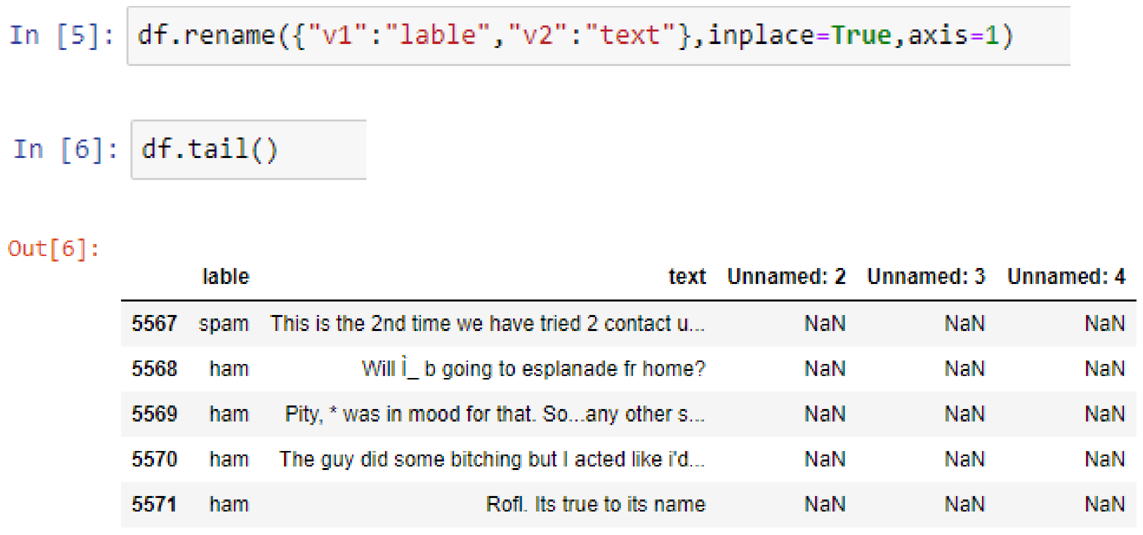
```
In [4]: df.isna().sum()
```



```
Out[4]: v1               0
        v2               0
        Unnamed: 2    5522
        Unnamed: 3    5560
        Unnamed: 4    5566
        dtype: int64
```

- From the above code of analysis, we can infer that columns such as V1 and v2 are not having missing columns,unnamed columns are not required for analysis

- Renaming the columns according the requirement



```
In [5]: df.rename({"v1":"lable","v2":"text"},inplace=True,axis=1)
```

```
In [6]: df.tail()
```

Out[6]:

	lable	text	Unnamed: 2	Unnamed: 3	Unnamed: 4
5567	spam	This is the 2nd time we have tried 2 contact u...	NaN	NaN	NaN
5568	ham	Will Ì_ b going to esplanade fr home?	NaN	NaN	NaN
5569	ham	Pity, * was in mood for that. So...any other s...	NaN	NaN	NaN
5570	ham	The guy did some bitching but I acted like i'd...	NaN	NaN	NaN
5571	ham	Rofl. Its true to its name	NaN	NaN	NaN

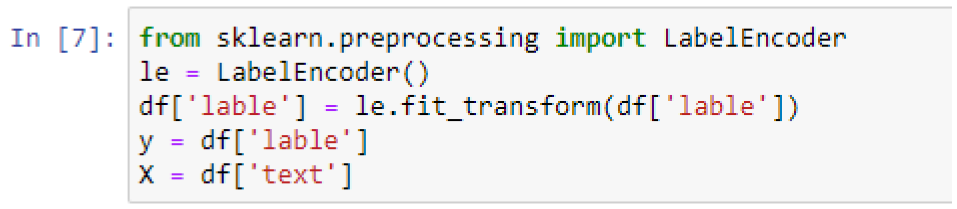
Handling Categorical Values

As we can see our dataset has categorical data we must convert the categorical data to integer encoding or binary encoding.

To convert the categorical features into numerical features we use encoding techniques.

There are several techniques but in our project we are using manual encoding with the help of list comprehension.

- In our project,we have text column so we will be using natural language processing for processing the data. Output column is having classes we
- Converting into 0 and 1 by applying label encoding



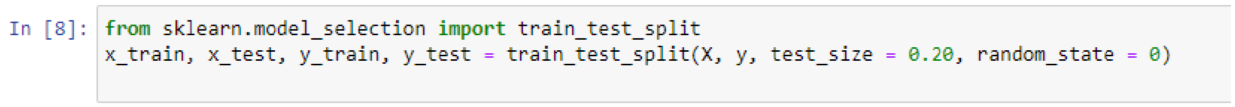
```
In [7]: from sklearn.preprocessing import LabelEncoder
        le = LabelEncoder()
        df['lable'] = le.fit_transform(df['lable'])
        y = df['lable']
        X = df['text']
```

Handling Imbalance Data

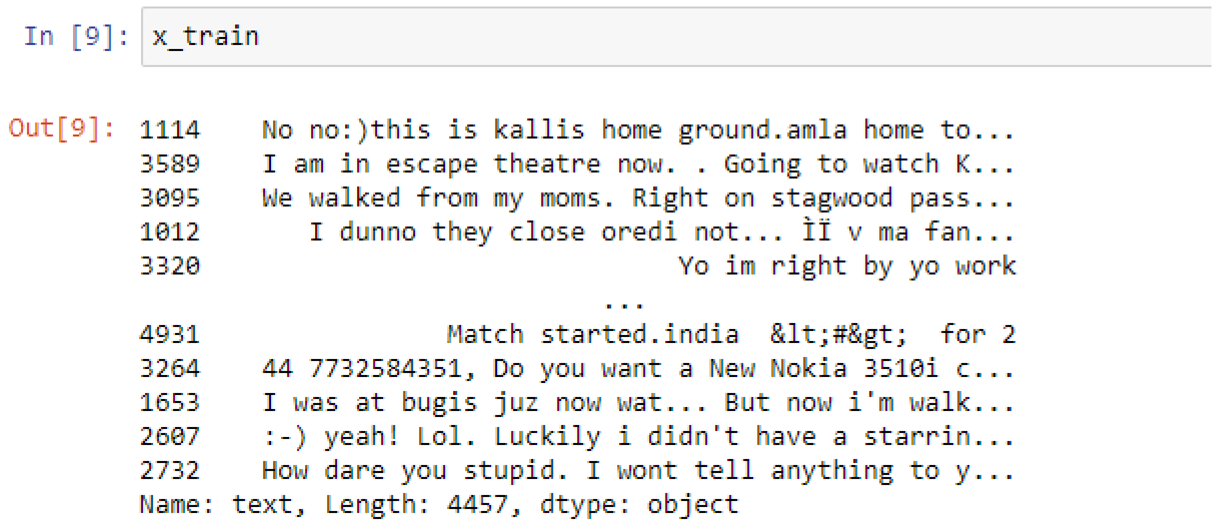
Data Balancing is one of the most important step, which need to be performed for classification models, because when we train our model on imbalanced dataset ,we will get biassed results, which means our model is able to predict only one class element .

For Balancing the data we are using the SMOTE Method.

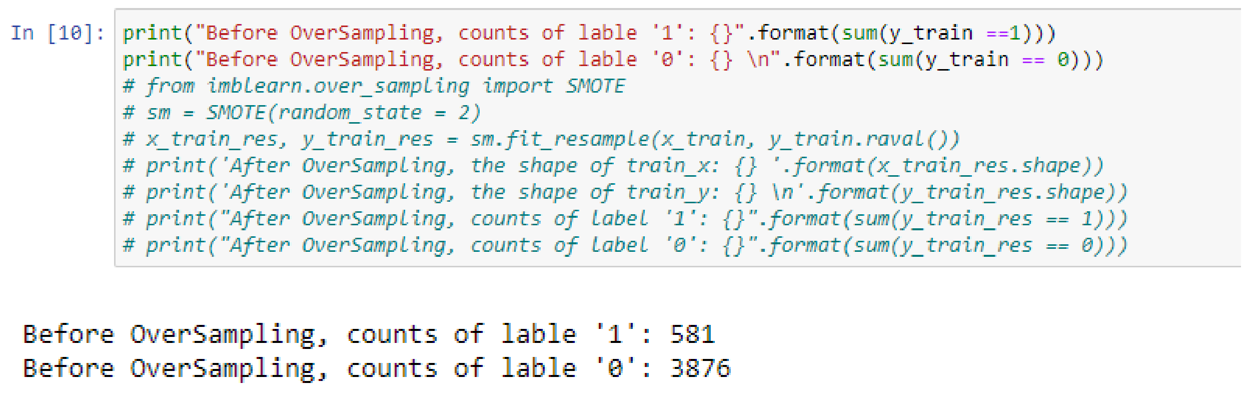
SMOTE: Synthetic minority over sampling technique, which will create new synthetic data points for under class as per the requirements given by us using KNN method.



```
In [8]: from sklearn.model_selection import train_test_split
        x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)
```



```
In [9]: x_train

Out[9]: 1114    No no:)this is kallis home ground.amla home to...
        3589    I am in escape theatre now. . Going to watch K...
        3095    We walked from my moms. Right on stagwood pass...
        1012       I dunno they close oredi not... ÌÏ v ma fan...
        3320                               Yo im right by yo work
                                 ...
        4931                  Match started.india  &lt;#&gt;  for 2
        3264    44 7732584351, Do you want a New Nokia 3510i c...
        1653    I was at bugis juz now wat... But now i'm walk...
        2607    :-) yeah! Lol. Luckily i didn't have a starrin...
        2732    How dare you stupid. I wont tell anything to y...
        Name: text, Length: 4457, dtype: object
```

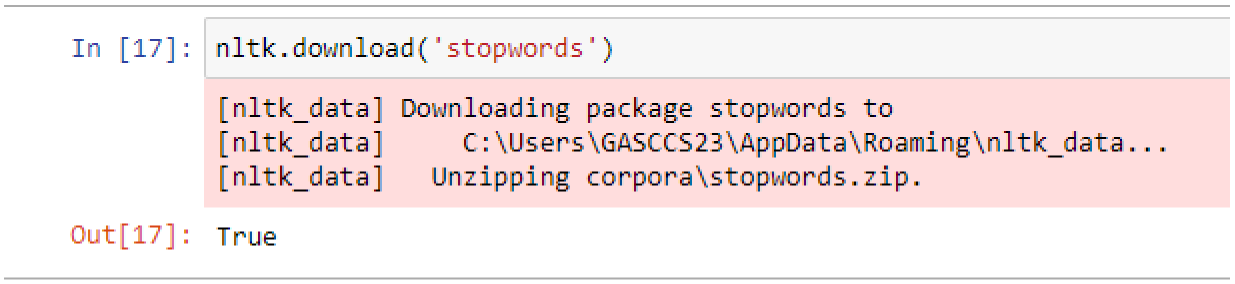


```
In [10]: print("Before OverSampling, counts of lable '1': {}".format(sum(y_train ==1)))
         print("Before OverSampling, counts of lable '0': {} \n".format(sum(y_train == 0)))
         # from imblearn.over_sampling import SMOTE
         # sm = SMOTE(random_state = 2)
         # x_train_res, y_train_res = sm.fit_resample(x_train, y_train.raval())
         # print('After OverSampling, the shape of train_x: {} '.format(x_train_res.shape))
         # print('After OverSampling, the shape of train_y: {} \n'.format(y_train_res.shape))
         # print("After OverSampling, counts of label '1': {}".format(sum(y_train_res == 1)))
         # print("After OverSampling, counts of label '0': {}".format(sum(y_train_res == 0)))
```

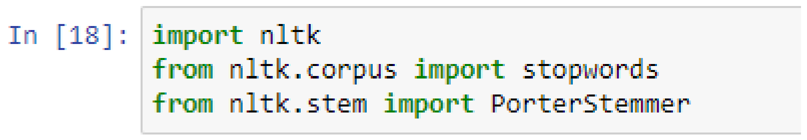
```
Before OverSampling, counts of lable '1': 581
Before OverSampling, counts of lable '0': 3876
```

From the above picture, we can infer that ,previously our dataset had 581 class 1, and 3876 class 0 items, after applying smote technique on the dataset the size has been changed for minority class.

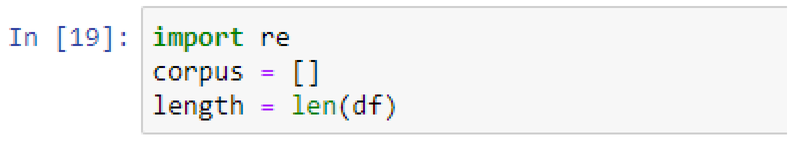
Cleaning The Text Data



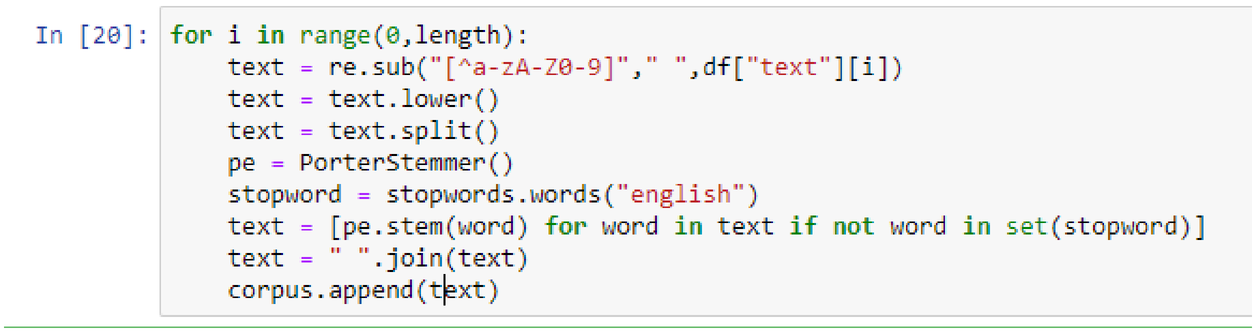
```
In [17]: nltk.download('stopwords')

         [nltk_data] Downloading package stopwords to
         [nltk_data]     C:\Users\GASCCS23\AppData\Roaming\nltk_data...
         [nltk_data]   Unzipping corpora\stopwords.zip.

Out[17]: True
```



```
In [18]: import nltk
         from nltk.corpus import stopwords
         from nltk.stem import PorterStemmer
```



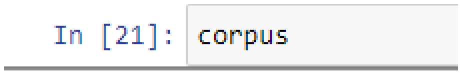
```
In [19]: import re
         corpus = []
         length = len(df)
```



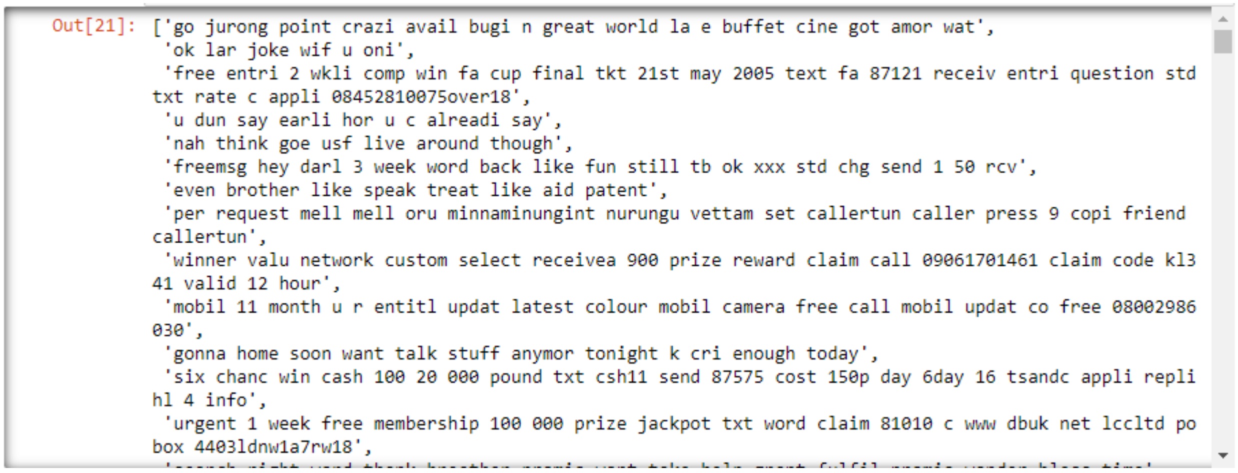
```
In [20]: for i in range(0,length):
             text = re.sub("[^a-zA-Z0-9]"," ",df["text"][i])
             text = text.lower()
             text = text.split()
             pe = PorterStemmer()
             stopword = stopwords.words("english")
             text = [pe.stem(word) for word in text if not word in set(stopword)]
             text = " ".join(text)
             corpus.append(text)
```

Text pre-processing includes

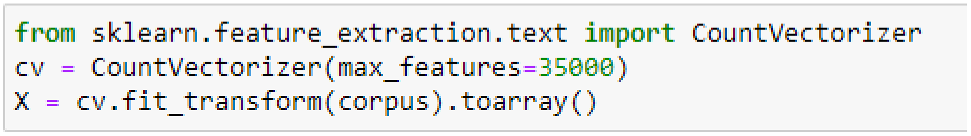
- Removing punctuation from the text using regular expression library
- Converting the sentence into lower case
- Tokenization – splitting the sentence into words
- Removing stop words from the data
- Stemming – stemming is the process of brining all the words into base form



```
In [21]: corpus
```

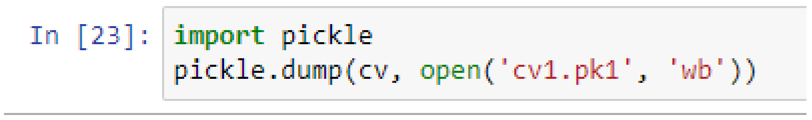


```
Out[21]: ['go jurong point crazi avail bugi n great world la e buffet cine got amor wat',
          'ok lar joke wif u oni',
          'free entri 2 wkli comp win fa cup final tkt 21st may 2005 text fa 87121 receiv entri question std
         txt rate c appli 08452810075over18',
          'u dun say earli hor u c alreadi say',
          'nah think goe usf live around though',
          'freemsg hey darl 3 week word back like fun still tb ok xxx std chg send 1 50 rcv',
          'even brother like speak treat like aid patent',
          'per request mell mell oru minnaminungint nurungu vettam set callertun caller press 9 copi friend
         callertun',
          'winner valu network custom select receivea 900 prize reward claim call 09061701461 claim code kl3
         41 valid 12 hour',
          'mobil 11 month u r entitl updat latest colour mobil camera free call mobil updat co free 08002986
         030',
          'gonna home soon want talk stuff anymor tonight k cri enough today',
          'six chanc win cash 100 20 000 pound txt csh11 send 87575 cost 150p day 6day 16 tsandc appli repli
         hl 4 info',
          'urgent 1 week free membership 100 000 prize jackpot txt word claim 81010 c www dbuk net lccltd po
         box 4403ldnw1a7rw18',
```



```
In [22]: from sklearn.feature_extraction.text import CountVectorizer
         cv = CountVectorizer(max_features=35000)
         X = cv.fit_transform(corpus).toarray()
```

- After applying all the above functions, we will get corpus
- Converting the corpus into Document Term matrix using Count vectorizer



```
In [23]: import pickle
         pickle.dump(cv, open('cv1.pk1', 'wb'))
```

Saving the count vectorizer function for future use