

⇒ **Natural Language Processing:**

According to industry estimates, only 21% of the available data is present in structured form. Data is being generated as we speak, as we tweet, as we send messages on Whatsapp and in various other activities. Majority of this data exists in the textual form, which is highly unstructured in nature.

Few notorious examples include – tweets / posts on social media, user to user chat conversations, news, blogs and articles, product or services reviews and patient records in the healthcare sector. A few more recent ones includes chatbots and other voice driven bots.

Despite having high dimension data, the information present in it is not directly accessible unless it is processed (read and understood) manually or analysed by an automated system.

In order to produce significant and actionable insights from text data, it is important to get understanding with the techniques and principles of Natural Language Processing (NLP).

NLP is a branch of data science that consists of systematic processes for analysing, understanding, and deriving information from the text data in a smart and efficient manner. By utilizing NLP and its components, one can organize the massive chunks of text data, perform numerous automated tasks and solve a wide range of problems such as – automatic summarization, machine translation, named entity recognition, relationship extraction, sentiment analysis, speech recognition, and topic segmentation etc.

A very well-known model in NLP is the Bag of Words model. It is a model used to pre-process the texts to classify before fitting the classification algorithms on the observations containing the texts.

Main NLP library example:

Natural Language Toolkit – NLTK

SpaCy

Stanford NLP

OpenNLP

⇒ **Benefits of NLP**

As all of you know, there are millions of gigabytes every day are generated by blogs, social websites, and web pages.

There are many companies gathering all of these data for understanding users and their passions and give these reports to the companies to adjust their plans.

Suppose a person loves traveling and is regularly searching for a holiday destination, the searches made by the user is used to provide him with relative advertisements by online hotel and flight booking apps.

Example:

- You can also use NLP on a text review to predict if the review is a good one or a bad one.
- You can use NLP on an article to predict some categories of the articles that you are trying to segment.
- You can use NLP on a book to predict the genre of the book.

NLP is all about analysing text. Here, I am going to analyse text containing restaurant reviews. We will apply ML over it and will predict, whether review is positive OR negative.

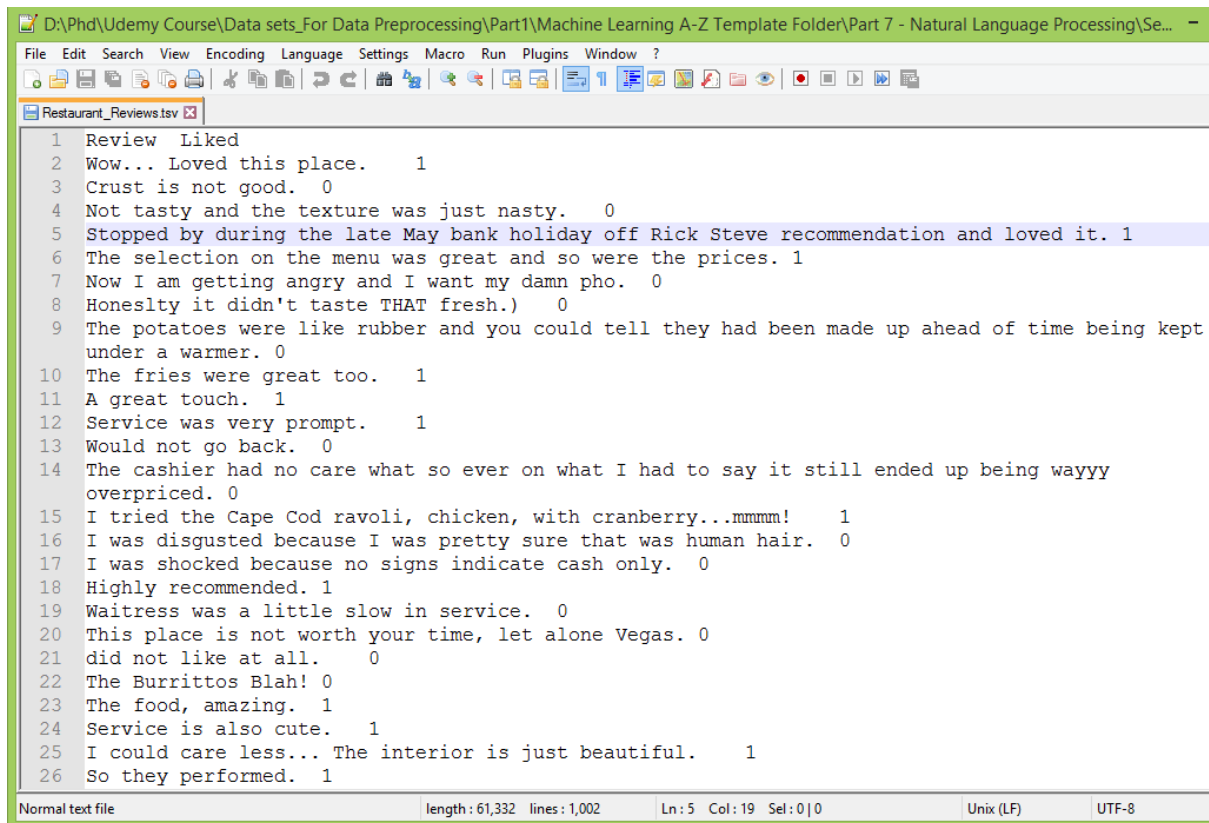
Step 1:

Install NLTK

```
pip install -U NLTK
```

Step 2:

Here, we are going to use .tsv (Tab Separated File) of reviews and it contains total 1000 review and having value 1 or 0 (means like or dislike).



```
1 Review Liked
2 Wow... Loved this place. 1
3 Crust is not good. 0
4 Not tasty and the texture was just nasty. 0
5 Stopped by during the late May bank holiday off Rick Steve recommendation and loved it. 1
6 The selection on the menu was great and so were the prices. 1
7 Now I am getting angry and I want my damn pho. 0
8 Honeslty it didn't taste THAT fresh.) 0
9 The potatoes were like rubber and you could tell they had been made up ahead of time being kept
under a warmer. 0
10 The fries were great too. 1
11 A great touch. 1
12 Service was very prompt. 1
13 Would not go back. 0
14 The cashier had no care what so ever on what I had to say it still ended up being wayyy
overpriced. 0
15 I tried the Cape Cod ravioli, chicken, with cranberry...mmmm! 1
16 I was disgusted because I was pretty sure that was human hair. 0
17 I was shocked because no signs indicate cash only. 0
18 Highly recommended. 1
19 Waitress was a little slow in service. 0
20 This place is not worth your time, let alone Vegas. 0
21 did not like at all. 0
22 The Burrittos Blah! 0
23 The food, amazing. 1
24 Service is also cute. 1
25 I could care less... The interior is just beautiful. 1
26 So they performed. 1
```

Step 3: Import dataset

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

```
dataset = pd.read_csv('D:\\Phd\\Udemy Course\\Data sets_For Data Preprocessing\\Part1\\Machine Learning A-Z Template Folder\\Part 7 - Natural Language Processing\\Section 36 - Natural Language Processing\\Restaurant_Reviews.tsv', delimiter='\\t', quoting=3)
```

Here, I have imported dataset and if we use `read_csv`, then it can import only .csv file. But, if we wish to import .tsv file, then we must specify one parameter `delimiter='\\t'`. There is also one more parameter and that is `quoting`. This `quoting` parameter is used to ignore any value. Here, I have specified `quoting = 3`, which means it can avoid all the double quotes (“), because sometimes it may cause some problem while analysing dataset.

Index	Review	Liked
0	Wow... Loved this place.	1
1	Crust is not good.	0
2	Not tasty and the texture ...	0
3	Stopped by during the l...	1
4	The selection on the menu ...	1
5	Now I am getting angr...	0
6	Honeslty it didn't taste...	0
7	The potatoes were like ru...	0
8	The fries were great t...	1
9	A great touch.	1
10	Service was very prompt.	1
11	Would not go back.	0
12	The cashier had no care ...	0
13	I tried the Cape Cod rav...	1

Step 4: Clean the dataset

Now, our next step is to clean different reviews to make it ready for future ML algorithms. Cleaning means, we are going to get rid of non-useful words which of no use to predict it's review (i.e., The, on). We will also try to avoid some punctuation like

Let's apply cleaning on first review and then we have to apply same thing trough looping on all the other reviews.

```
In [2]: dataset['Review'][0]
Out[2]: 'Wow... Loved this place.'
```

Cleaning the texts

import re

review = re.sub('[^a-zA-Z]', ' ', dataset['Review'][0])

Here with the help of sub function we are going to remove unnecessary data. In sub function, first parameter `[^a-zA-Z]` will not allow you to remove any lowercase OR uppercase letter. 2nd parameter is space `' '`, which indicates that our removed data OR character we have to replace with space and third indicates the 1st review on which first time I have applied cleaning.

```
In [6]: review
Out[6]: 'Wow    Loved this place '
```

Our next step is to convert all the uppercase letters into lowercase letters. Means, we need to convert `Wow` into `wow`.

```
review = review.lower()
```

```
In [9]: review
Out[9]: 'wow    loved this place '
```

Step 5: Apply stop words

Now, next step is to remove irrelevant words like `the`, `a`, `on`, `in` etc...

In above example, **loved this place** statement is there. In this, **this** indicates nothing. In order to do this, we need to import `nltk` and library and then have to use `stopword` tool to avoid such types of unnecessary words. For that we need to download stopwords:

```
import nltk
```

```
nltk.download('stopwords')
```

We can convert string into list of word in order to check and then clean every single word.

```
review = review.split()
```

Consider this full code which removes extra words also:

```
# Cleaning the texts
```

```
from nltk.corpus import stopwords
```

```
review = re.sub('[^a-zA-Z]', ' ', dataset['Review'][0])
```

```
review = review.lower()
```

```
review = review.split()
```

```
review = [word for word in review if not word in set(stopwords.words('english'))]
```

review	list	3	['wow', 'loved', 'place']
--------	------	---	---------------------------

Step 6: Apply Stemming

Now, let's move towards next step i.e. stemming. For that, we need to import PorterStemmer class from library nltk.stem.porter

```
from nltk.stem.porter import PorterStemmer
```

```
ps = PorterStemmer()
```

Now, we have to use this object to apply stemming on each of the word of review.

```
review = [ps.stem(word) for word in review if not word in set(stopwords.words('english'))]
```

review	list	3	['wow', 'love', 'place']
--------	------	---	--------------------------

FULL CODE UP TO THIS:

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
import pandas as pd
```

```
dataset = pd.read_csv('D:\\Phd/Udemy Course/Data sets_For Data Preprocessing/Part1/Machine Learning A-Z Template Folder/Part 7 - Natural Language Processing/Section 36 - Natural Language Processing/Restaurant_Reviews.tsv', delimiter='\t', quoting=3)
```

```
# Cleaning the texts
```

```
import re
```

```
import nltk
```

```
nltk.download('stopwords')
```

```
from nltk.corpus import stopwords
```

```
from nltk.stem.porter import PorterStemmer
```

```
review = re.sub('[^a-zA-Z]', ' ', dataset['Review'][0])
```

```
review = review.lower()
```

```
review = review.split()
```

```
ps = PorterStemmer()
```

```
review = [ps.stem(word) for word in review if not word in  
set(stopwords.words('english'))]
```

So, we are done with cleaning process. Now, final step is to apply joining step on all the words.

```
review = ' '.join(review)
```

This line join the list of words in a review and separate them by space because of ' '. Now, we have to do this same thing for all the reviews with help of loop.

```
corpus = []
```

```
for i in range(0, 1000):
```

```
    review = re.sub('[^a-zA-Z]', ' ', dataset['Review'][i])
```

```
    review = review.lower()
```

```
    review = review.split()
```

```
    ps = PorterStemmer()
```

```
    review = [ps.stem(word) for word in review if not word in  
set(stopwords.words('english'))]
```

```
    review = ' '.join(review)
```

```
    corpus.append(review)
```

Index	Review	Liked
0	Wow... Loved this place.	1
1	Crust is not good.	0
2	Not tasty and the texture was just nasty.	0
3	Stopped by during the late May bank holiday off Rick Steve reco...	1
4	The selection on the menu was great and so were the prices.	1
5	Now I am getting angry and I want my damn pho.	0
6	Honeslty it didn't taste THAT fresh.)	0
7	The potatoes were like rubber and you could tell they had been mad...	0
8	The fries were great too.	1
9	A great touch.	1
10	Service was very prompt.	1
11	Would not go back.	0
12	The cashier had no care what so ever on what I had to say it sti...	0
13	I tried the Cape Cod ravioli,	1

Index	Type	Size	Value
0	str	1	wow love place
1	str	1	crust good
2	str	1	tasti textur nasti
3	str	1	stop late may bank holiday rick steve recommend love
4	str	1	select menu great price
5	str	1	get angri want damn pho
6	str	1	honeslty tast fresh
7	str	1	potato like rubber could tell made ahead time kept warmer
8	str	1	fri great
9	str	1	great touch
10	str	1	servic prompt
11	str	1	would go back
12	str	1	cashier care ever say still end wayyy overpr
13	str	1	tri cape cod ravioli chicken cranberry mmmm

Step 7: Create bag of words model

The bag-of-words model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. The bag-of-words model is a way of representing text data when modelling text with machine learning algorithms.

In order to create Bag of Words model, it read every review, identify unique word and for each word it creates one column. So, every word stands for one column and total row remain same equal to number of reviews. Now in whichever review that word of column available, we need to put 1 over there and in rest other place it put 0. So, it is having matrix with majority of value as 0. So, this matrix is known as sparse matrix.

Creating a Bag of Words Model

```
from sklearn.feature_extraction.text import CountVectorizer
```

```
cv = CountVectorizer(max_features=1500)
```

```
X = cv.fit_transform(corpus).toarray()
```


	1293	1294	1295	1296	1297
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	1
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	1	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0
11	0	0	0	0	0

Here, it selects 1500 frequently used words out of 1000 rows.

This sparse matrix X contains only independent variables.

We need to add dependent variable also in order to predict whether review is positive or negative.

`Y = dataset.iloc[:, 1].values`

Y - NumPy array	
	0
0	1
1	0
2	0
3	1
4	1
5	0
6	0
7	0
8	1
9	1
10	1
11	0
12	0

Now, we need to select any classification algorithm in order to predict.

Prediction using Naïve Bayes Classification Algorithm:

```
#Naive Bayes Classification Algorithm

from sklearn.cross_validation import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.20, random_state
= 0)

# Fitting SVM to the training set

from sklearn.naive_bayes import GaussianNB

classifier = GaussianNB()

classifier.fit(X_train, Y_train)

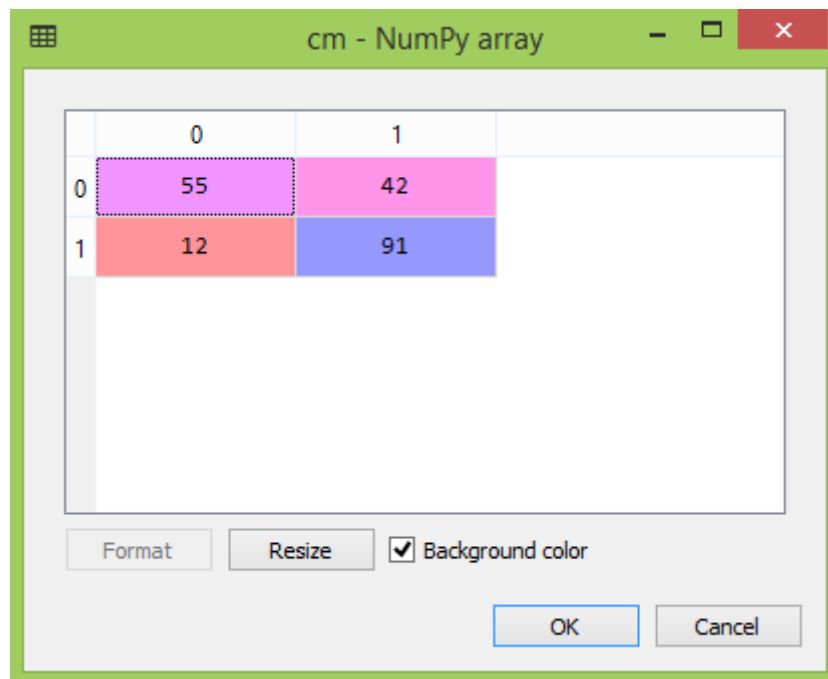
# Predicting the Test set results

Y_pred = classifier.predict(X_test)

# Making the confusion matrix

from sklearn.metrics import confusion_matrix
```

```
cm = confusion_matrix(Y_test, Y_pred)
```



⇒ Generalized mechanism of NLP

