Natural Language Processing(NLP)

Importing Libraries

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

Importing Dataset

```
df = pd.read_csv('Restaurant_Reviews.tsv', delimiter = '\t', quoting = 3)
```

Cleaning the texts

```
corpus = []
for i in range(0,1000):
  review = re.sub('[^a-zA-Z]', ' ', df['Review'][i])
  review = review.lower().split()
  ps = PorterStemmer()
  all_stopwords = stopwords.words('english')
  all_stopwords.remove('not')
  review = [ps.stem(word) for word in review if not word in set(all_stopwords)]
  review = ' '.join(review)
  corpus.append(review)
```

To find out the max_features for CountVectorization

```
cv = CountVectorizer()
x = cv.fit_transform(corpus).toarray()
y = df.iloc[:, -1].values
```

Double-click (or enter) to edit

```
display(corpus)
```

```
['wow love place',
 'crust not good',
 'not tasti textur nasti',
 'stop late may bank holiday rick steve recommend love',
 'select menu great price',
 'get angri want damn pho',
 'honeslti tast fresh',
 'potato like rubber could tell made ahead time kept warmer',
 'fri great',
 'great touch',
 'servic prompt',
 'would not go back',
 'cashier care ever say still end wayyy overpr',
 'tri cape cod ravoli chicken cranberri mmmm',
 'disgust pretti sure human hair',
 'shock sign indic cash',
 'highli recommend',
 'waitress littl slow servic',
 'place not worth time let alon vega',
 'not like',
 'burritto blah',
 'food amaz',
 'servic also cute',
 'could care less interior beauti',
 'perform',
 'right red velvet cake ohhh stuff good',
 'never brought salad ask',
 'hole wall great mexican street taco friendli staff',
 'took hour get food tabl restaur food luke warm sever run around like total overwhelm',
 'worst salmon sashimi',
 'also combo like burger fri beer decent deal',
 'like final blow',
 'found place accid could not happier',
 'seem like good quick place grab bite familiar pub food favor look elsewher',
 'overal like place lot',
 'redeem qualiti restaur inexpens',
 'ampl portion good price',
 'poor servic waiter made feel like stupid everi time came tabl',
 'first visit hiro delight',
 'servic suck',
 'shrimp tender moist',
 'not deal good enough would drag establish',
 'hard judg whether side good gross melt styrofoam want eat fear get sick',
 'posit note server attent provid great servic',
 'frozen puck disgust worst peopl behind regist',
 'thing like prime rib dessert section',
 'bad food damn gener',
 'burger good beef cook right',
 'want sandwich go firehous',
 'side greek salad greek dress tasti pita hummu refresh',
 'order duck rare pink tender insid nice char outsid',
 'came run us realiz husband left sunglass tabl',
 'chow mein good',
 'horribl attitud toward custom talk one custom enjoy food',
 'portion huge',
 'love friendli server great food wonder imagin menu',
 'heart attack grill downtown vega absolut flat line excus restaur',
```

```
len(x[0])
```

```
1566
```

```
 'place receiv star appet',
```

Creating the Bag of Words model

```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features = 1500)
X = cv.fit_transform(corpus).toarray()
y = df.iloc[:, -1].values
```

```
 'realli realli good nice time'
```

Splitting the dataset into training and test set

```
 'guess known place would suck insid excalibur use common sens',
```

```
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

```
 'sweet potato fri good season well',
```

Training the Naive Bayes model on training set

```
 'come like experienc underwhelm relationship parti wait person ask break',
```

```
classifier = GaussianNB()
classifier.fit(x_train, y_train)
```

```
GaussianNB
GaussianNB()
```

Predicting the test set results

```
y_pred = classifier.predict(x_test)
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),1)),1))
```

```
[[1 0]
 [1 0]
 [1 0]
 [0 0]
 [0 0]
 [1 0]
 [1 1]
 [1 0]
 [1 0]
 [1 1]
 [1 1]
 [1 1]
 [1 0]
 [1 1]
 [1 1]
 [1 1]
 [0 0]
 [0 0]
 [0 0]
 [1 1]
 [0 0]
 [0 1]
 [1 1]
 [1 0]
 [1 0]
 [0 1]
 [1 1]
 [1 1]
 [1 1]
 [0 0]
 [1 1]
 [1 1]
 [1 1]
 [1 1]
 [1 1]
 [0 0]
 [1 0]
 [0 0]
 [1 0]
 [1 1]
 [1 1]
 [1 0]
 [1 1]
 [0 0]
 [0 0]
 [0 0]
 [1 0]
 [1 0]
 [0 0]
 [0 0]
 [1 1]
 [1 1]
 [1 1]
 [1 1]
 [1 0]
 [0 0]
 [1 1]
 [1 1]]
```

Making the Confusion Matrix

```
cf = confusion_matrix(y_test, y_pred)
print(cf)
accuracy_score(y_test, y_pred)
```

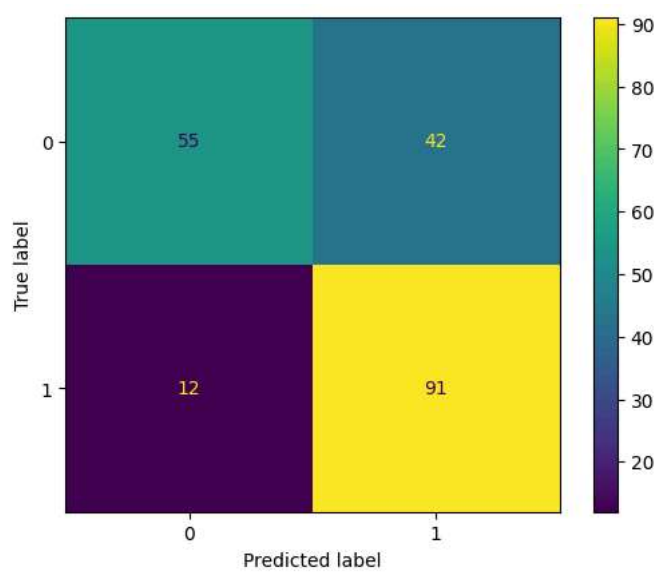
```
[[55 42]
 [12 91]]
0.73
```

Visualization of the result

```
'also order spinach avocado salad ingredi sad dress liter zero tast',

# Creating the confusion matrix visualization
cm = confusion_matrix(y_test, y_pred, labels=classifier.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=classifier.classes_)

# Plot the confusion matrix
disp.plot()
plt.show()
```




```
'worst experi ever',
'must night place',
'side delish mix mushroom yukon gold pure white corn beateou',
'bug never show would given sure side wall bug climb kitchen',
```