```
import pandas as pd
```

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

```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True
```

```
df=pd.read_csv('/content/Restaurant_Reviews 1.tsv',delimiter='\t',quoting=3)
```

```
df.head()
```

	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 of...	1
4	The selection on the menu was great and so wer...	1

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.shape
```

```
(1000, 2)
```

```
df["Review"][0]
```

```
'Wow... Loved this place.'
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Review  1000 non-null   object
 1   Liked   1000 non-null   int64
dtypes: int64(1), object(1)
memory usage: 15.8+ KB
```

```
df.columns
```

```
Index(['Review', 'Liked'], dtype='object')
```

```
import re
```

```
import re
```

```
import string
from nltk.stem.porter import PorterStemmer
from nltk.corpus import stopwords
import nltk
nltk.download('stopwords')

corpus=[]
for i in range(0,1000):
  review=re.sub(pattern='[^a-zA-Z]',repl=' ',string=df['Review'][i])
  review=review.lower()
  review_word=review.split()
  review_word=[word for word in review_word if not word in set(stopwords.words('english'))]
  ps=PorterStemmer()
  review1=[ps.stem(word) for word in review_word]
  # Join the stemmed words back into a single string
  review=' '.join(review1)
  corpus.append(review) # Append the string to corpus
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
corpus[:1000]
```

```
 'realli impress place close',
 'would avoid place stay mirag',
 'refri bean came meal dri crusti food bland',
 'spend money time place els',
 'ladi tabl next us found live green caterpillar salad',
 'present food aw',
 'tell disappoint',
 'think food flavor textur lack',
 'appetit instantli gone',
 'overal impress would go back',
 'whole experi underwhelm think go ninja sushi next time',
 'wast enough life pour salt wound draw time took bring check']
```

```
df.shape
```

```
(1000, 2)
```

```
from sklearn.feature_extraction.text import CountVectorizer
cv= CountVectorizer(max_features=1500)
```

```
df["Review"]
```

	Review
0	Wow... Loved this place.
1	Crust is not good.
2	Not tasty and the texture was just nasty.
3	Stopped by during the late May bank holiday of...
4	The selection on the menu was great and so wer...
...	...
995	I think food should have flavor and texture an...
996	Appetite instantly gone.
997	Overall I was not impressed and would not go b...
998	The whole experience was underwhelming, and I ...
999	Then, as if I hadn't wasted enough of my life ...

1000 rows × 1 columns

dtype: object

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

```
X.shape
```

```
(1000, 1500)
```

```
X.shape
```

```
(1000, 1500)
```

```
X[0]
```

```
array([0, 0, 0, ..., 0, 0, 0])
```

```
X[0].max()
```

```
1
```

```
y=df.iloc[:,-1].values
```

```
y.shape
```

```
(1000,)
```

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test,y_train, y_test = train_test_split(X,y,random_state=104,test_size=0.2)
```

```
from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
```

```
clf1= GaussianNB()
clf2=MultinomialNB()
clf3=BernoulliNB()
```

```
clf1.fit(X_train,y_train)
clf2.fit(X_train,y_train)
clf3.fit(X_train,y_train)
```

```
▾ BernoulliNB
BernoulliNB()
```

```
y_predG=clf1.predict(X_test)
y_predM=clf2.predict(X_test)
y_predB=clf3.predict(X_test)
```

```
from sklearn.metrics import accuracy_score
```

```
accuracy_score(y_test,y_predG)
```

```
0.665
```

```
accuracy_score(y_test,y_predM)
```

```
0.785
```

```
accuracy_score(y_test,y_predB)
```

```
0.785
```

```
print("Gaussian",accuracy_score(y_test,y_predG))
print("Multinomial",accuracy_score(y_test,y_predG))
print("Bernoulli",accuracy_score(y_test,y_predG))
```

```
Gaussian 0.665
Multinomial 0.665
Bernoulli 0.665
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
rf = RandomForestClassifier()
rf.fit(X_train,y_train)
y_pred=rf.predict(X_test)
accuracy_score(y_test,y_pred)
```

```
0.735
```

```
from xgboost import XGBClassifier
xgb = XGBClassifier()
xgb.fit(X_train,y_train)
y_pred1 = xgb.predict(X_test)
accuracy_score(y_test,y_pred1)
```

```
0.75
```

```
from sklearn.metrics import confusion_matrix
```

```
confusion_matrix(y_test,y_pred)
```

```
array([[94, 10],
       [43, 53]])
```

```
confusion_matrix(y_test,y_pred1)
```

```
array([[82, 22],
       [28, 68]])
```