```
## importing libraries

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd


# importing dataset
data=pd.read_csv(r"D:\Data Science with AI\30th-jan-2024\4.CUSTOMERS REVIEW DATASET\Restaurant_Revi

##cleaning the texts
import re
import nltk
#nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer

corpus = []

for i in range(0, 1000):
    review = re.sub('[^a-zA-Z]', ' ', data['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)


##creating bag of words model
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer()
x=cv.fit_transform(corpus).toarray()
y=data.iloc[:,-1].values


##splitting the dataset into traingi and testing set
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)


from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB,BernoulliNB
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix,accuracy_score
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import GradientBoostingClassifier

from lightgbm import LGBMClassifier



from sklearn.metrics import accuracy_score
```

```
classifier={
    'Logistic Regression':LogisticRegression(),
    'Decision Tree' :DecisionTreeClassifier(),
    'RandomForest':RandomForestClassifier(),
    'Support Vector Machine':SVC(gamma=1),
    'K-nearest Neighbors':KNeighborsClassifier(),
    'BernoulliNB':BernoulliNB(),
    'GaussianNB':GaussianNB(),
    'GradientBoostingClassifier':GradientBoostingClassifier(),
    'Light GBM':LGBMClassifier()
    }

results=pd.DataFrame(columns=['accuracy','bias','variance'])


for method, cls in classifier.items():
     #train the classifier
     cls.fit(x_train,y_train)
     # Predicting the Test set results
     y_pred = cls.predict(x_test)

     ac = accuracy_score(y_test, y_pred)

     bias = cls.score(x_train,y_train)

     variance = cls.score(x_test,y_test)

     results.loc[method]=[ac,bias,variance]


     # Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)




results
```