Machine Learning

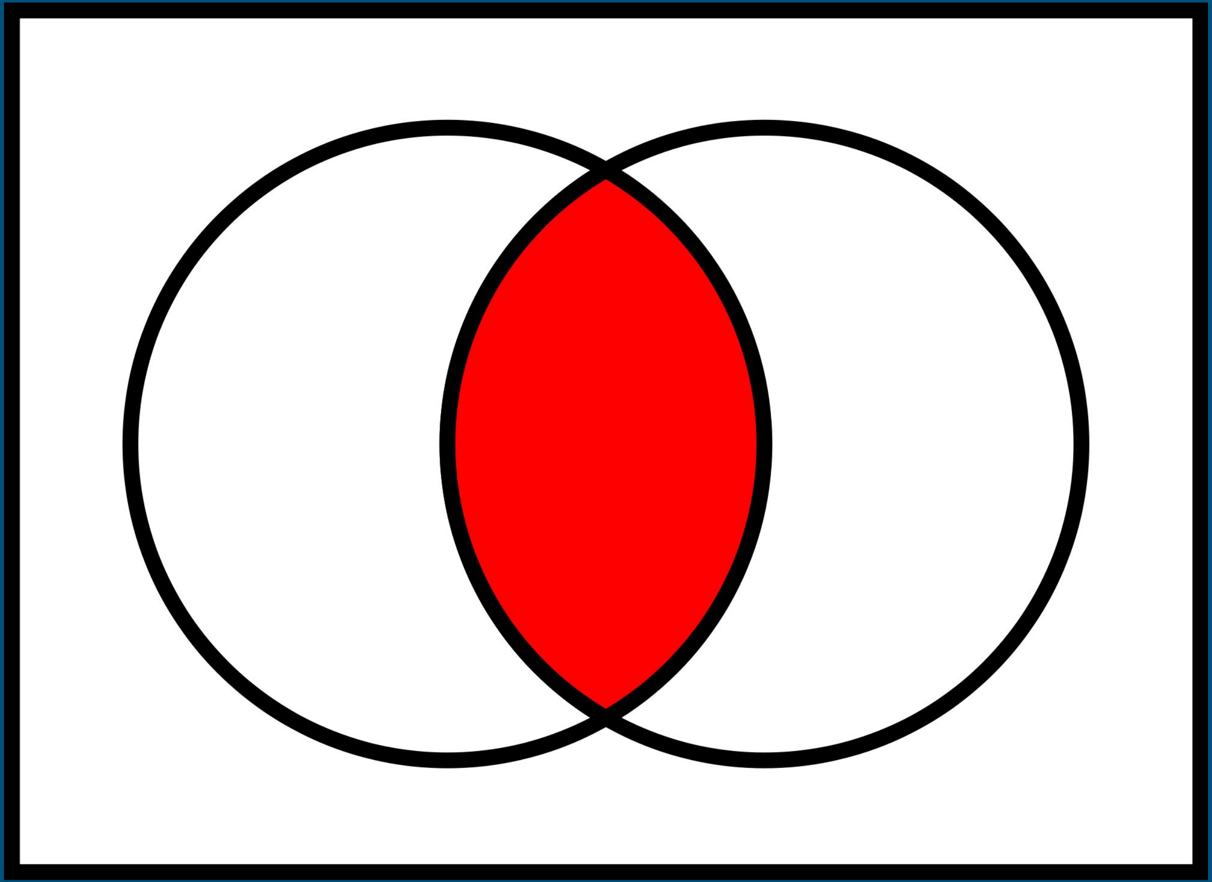
By EvolkAI

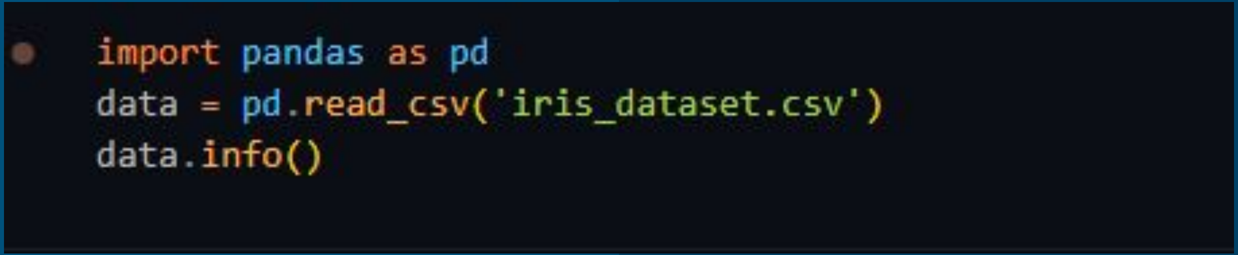
Model Name: ComplementNB

Complement Naive Bayes (NB) is a type of algorithm used for classification tasks, which means it helps us categorize things into different groups based on certain features. It is called "complement" because it takes a different approach compared to regular Naive Bayes.

Instead of directly calculating the probabilities of each class based on the features, complement NB focuses on the probabilities of the non-class features. It assumes that the absence of certain features in a particular class is more important in determining the class than the presence of specific features. It's like looking at what's missing rather than what's there.

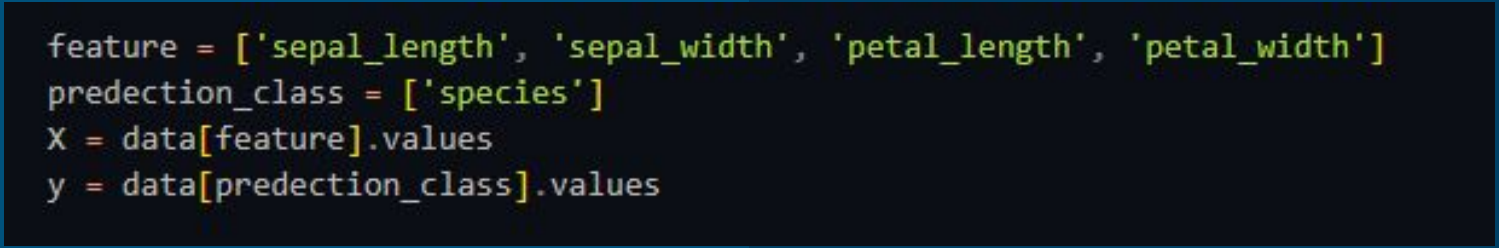
By considering the complement probabilities, this algorithm can be useful when dealing with imbalanced datasets, where some classes have significantly more instances than others. Complement NB aims to address the issue of under-represented classes by focusing on the less common features.





```
import pandas as pd
data = pd.read_csv('iris_dataset.csv')
data.info()
```

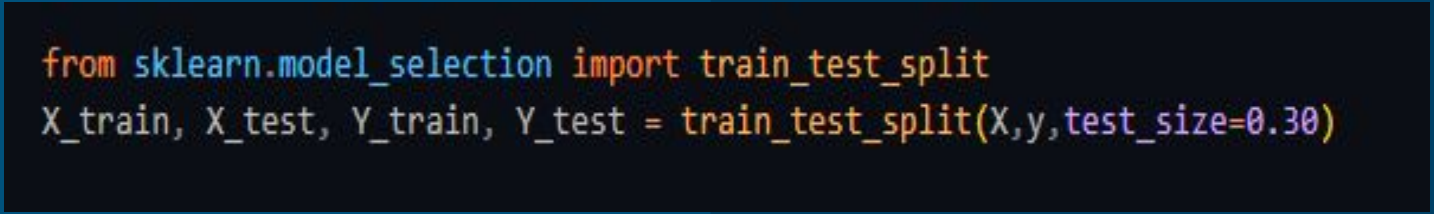
In First line we Import pandas library as pd, then we read iris_dataset.csv file using the read_csv() function, and prints information about the data using the info() method.



```
feature = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width']
predection_class = ['species']
X = data[feature].values
y = data[predection_class].values
```

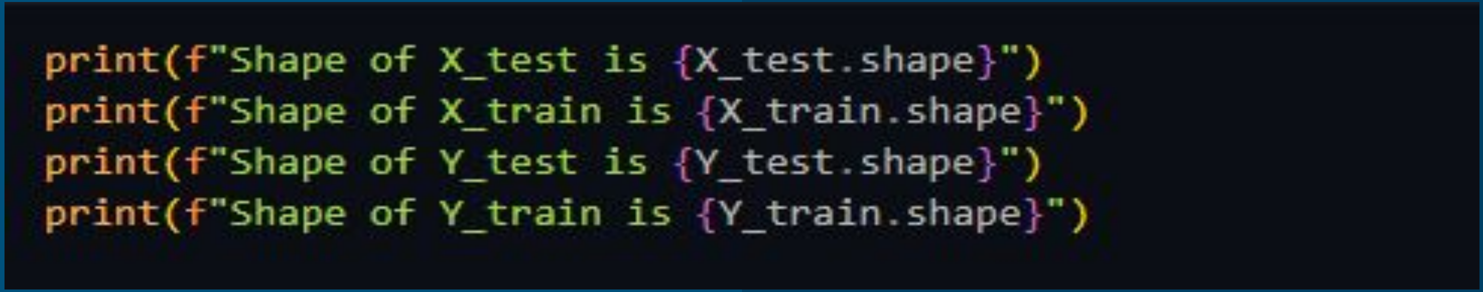
This defines the ‘feature’ and ‘predection_class’ variables, which specify the columns of the data to use as ‘features’ and the column to use as the prediction target.

The code then creates ‘X’ and ‘y’ arrays containing the values of these columns from the data DataFrame.



```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X,y,test_size=0.30)
```

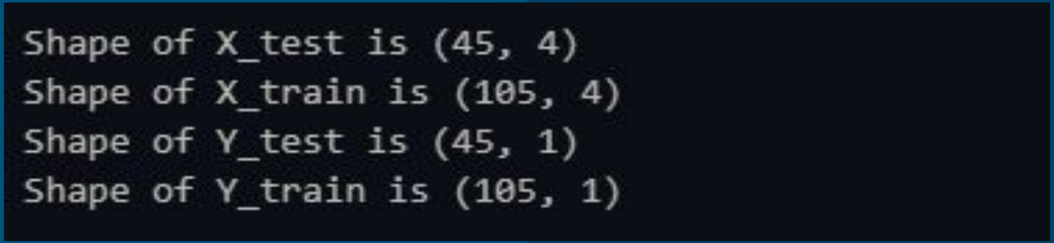
This imports the train_test_split function from the sklearn.model_selection module and uses it to split the data into training and testing sets. The test_size parameter specifies that 30% of the data should be used for testing.



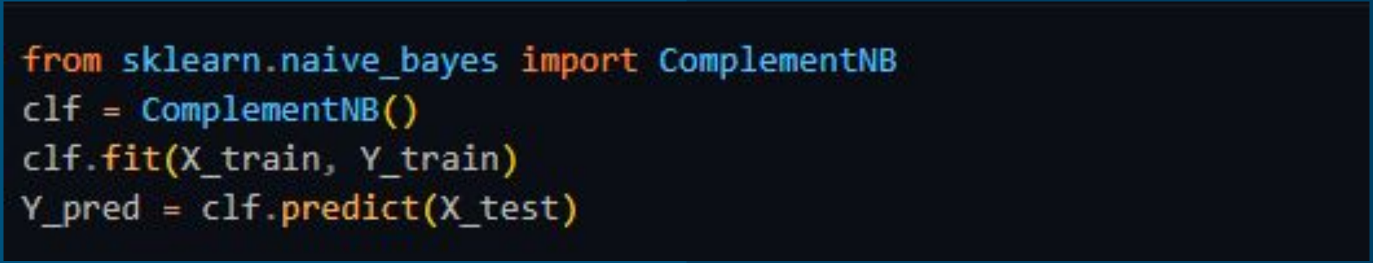
```
print(f"Shape of X_test is {X_test.shape}")
print(f"Shape of X_train is {X_train.shape}")
print(f"Shape of Y_test is {Y_test.shape}")
print(f"Shape of Y_train is {Y_train.shape}")
```

These lines print the shapes of the training and testing data arrays. This output totally depends on the test size we took while train_test_split.

OUTPUT

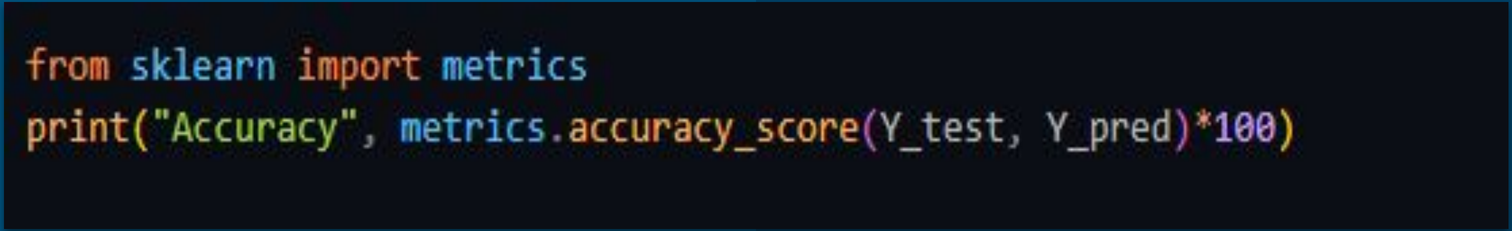


```
Shape of X_test is (45, 4)
Shape of X_train is (105, 4)
Shape of Y_test is (45, 1)
Shape of Y_train is (105, 1)
```



```
from sklearn.naive_bayes import ComplementNB
clf = ComplementNB()
clf.fit(X_train, Y_train)
Y_pred = clf.predict(X_test)
```

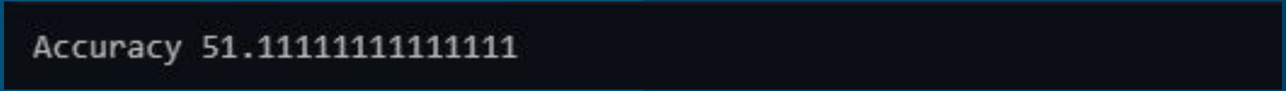
This imports the *ComplementNB* module from the sklearn tree library, creates a *ComplementNB* classifier object, fits the classifier with the training data using the *fit()* method, and use the *predict()* method to generate predictions for the testing data.



```
from sklearn import metrics
print("Accuracy", metrics.accuracy_score(Y_test, Y_pred)*100)
```

This imports the metrics module from sklearn and uses the accuracy_score() function to calculate the accuracy of the model on the testing data. The result is printed in the console.

OUTPUT



```
Accuracy 51.11111111111111
```

Conclusion

In conclusion, Complement Naive Bayes (NB) is a valuable algorithm for classification tasks, particularly in dealing with imbalanced datasets. It takes a different approach compared to regular Naive Bayes by focusing on the probabilities of non-class features.

By considering what is missing rather than what is present, Complement NB addresses the issue of under-represented classes and provides more balanced predictions. Its effectiveness in handling imbalanced datasets makes it a useful tool in various applications, allowing researchers and practitioners to make accurate categorizations and improve their understanding of complex datasets.

Thank You

🙏