Machine Learning

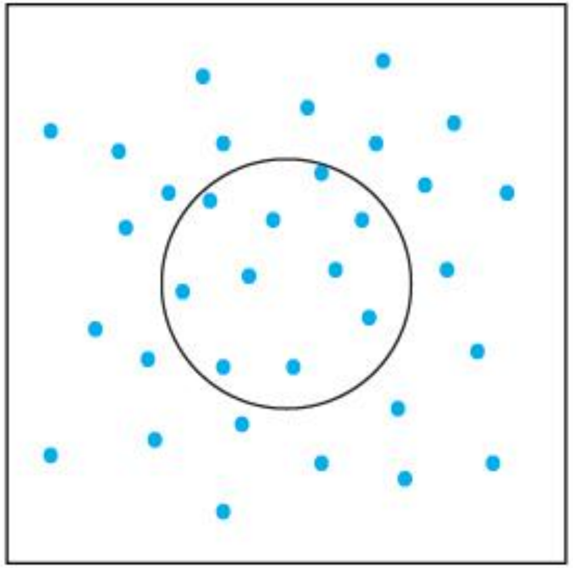
By EvolkAI

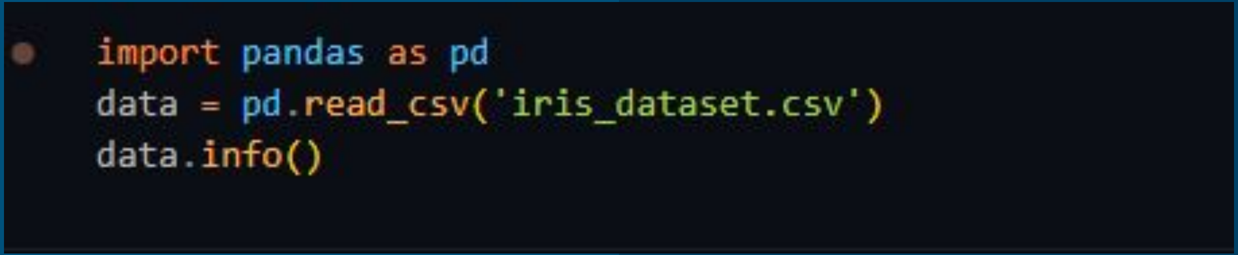
Model Name:
K-neighbors Classifier

The k-Neighbors Classifier is a simple and intuitive machine learning algorithm used for classification tasks. It works based on the idea that objects that are similar tend to belong to the same class.

Here's how it works: When given a new data point, the algorithm looks at its k nearest neighbors in the training dataset. The value of k is predefined by the user. The algorithm then assigns the class label to the new data point based on the majority class of its neighbors. For example, if k=5 and three of the five nearest neighbors belong to class A, the new data point will be classified as class A.

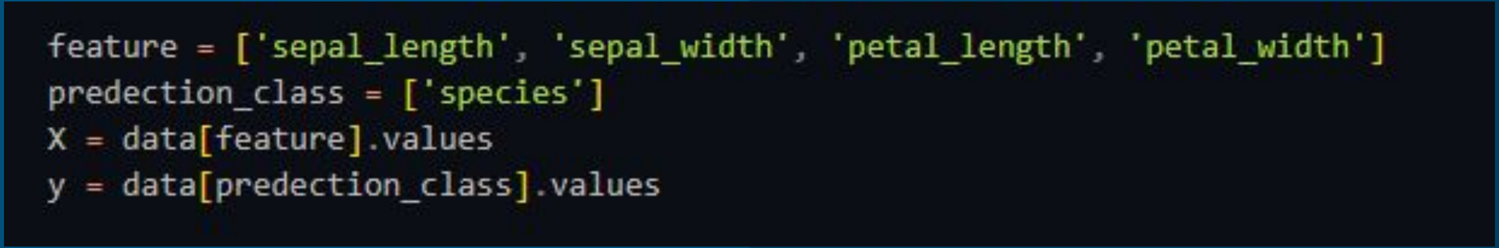
The k-Neighbors Classifier doesn't make any assumptions about the underlying data distribution, making it versatile and applicable to various domains. However, it's important to choose an appropriate value for k, as it can impact the accuracy of the algorithm. Overall, it's a straightforward and effective approach for classification tasks.





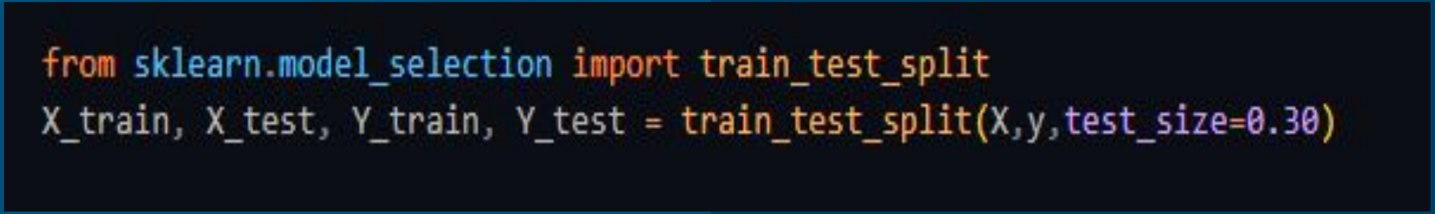
```
import pandas as pd
data = pd.read_csv('iris_dataset.csv')
data.info()
```

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.



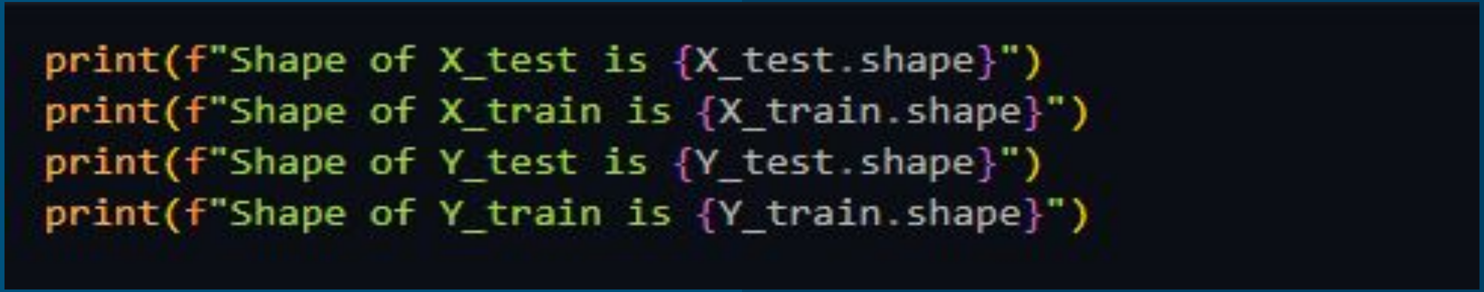
```
feature = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width']
predection_class = ['species']
X = data[feature].values
y = data[predection_class].values
```

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)
```

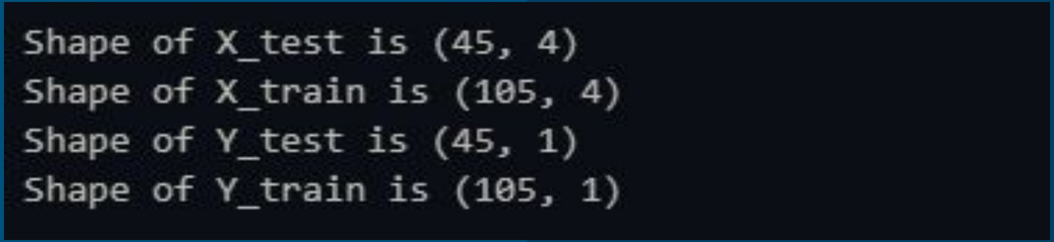
This imports the train_test_split function from the sklearn.model_selection module and uses it to split the data into training and testing sets. The test_size parameter specifies that 30% of the data should be used for testing.



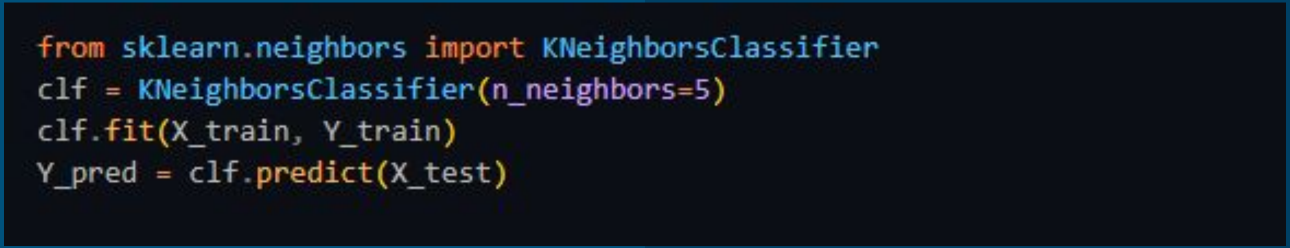
```
print(f"Shape of X_test is {X_test.shape}")
print(f"Shape of X_train is {X_train.shape}")
print(f"Shape of Y_test is {Y_test.shape}")
print(f"Shape of Y_train is {Y_train.shape}")
```

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

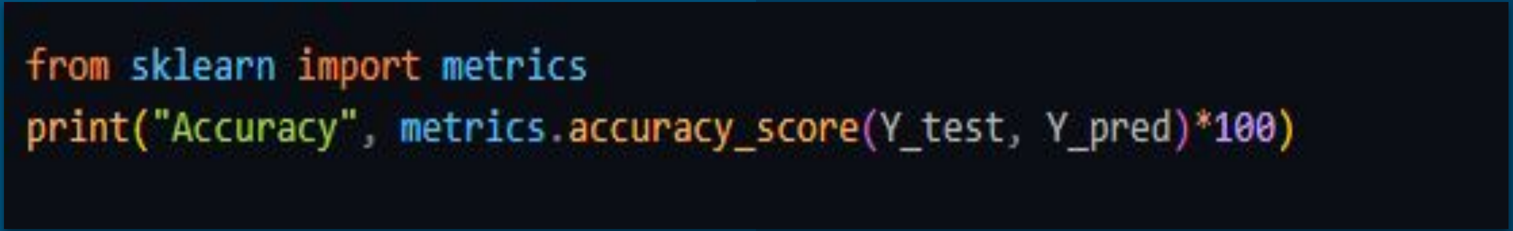


```
Shape of X_test is (45, 4)
Shape of X_train is (105, 4)
Shape of Y_test is (45, 1)
Shape of Y_train is (105, 1)
```



```python
from sklearn.neighbors import KNeighborsClassifier
clf = KNeighborsClassifier(n_neighbors=5)
clf.fit(X_train, Y_train)
Y_pred = clf.predict(X_test)
```

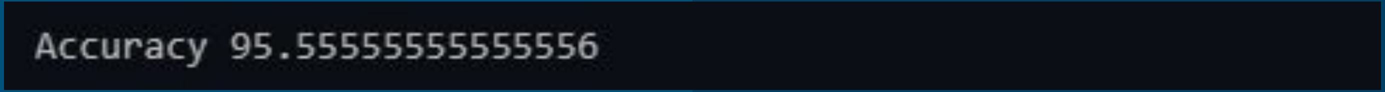
This imports the *KNeighborsClassifier* module from the sklearn neighbors library, creates a *KNeighborsClassifier* classifier object, fits the classifier with the training data using the *fit()* method, and use the *predict()* method to generate predictions for the testing data.



```python
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 95.55555555555556
```

Conclusion

In conclusion, the k-Neighbors Classifier is a straightforward and intuitive algorithm for classification tasks. It relies on the idea that similar objects belong to the same class. By examining the k nearest neighbors of a new data point, it determines its class label based on the majority class among the neighbors. This algorithm doesn't make assumptions about the data distribution, making it versatile across domains.

However, choosing an appropriate value for k is important for optimal performance. The k-Neighbors Classifier provides a simple yet effective approach to classification, enabling researchers and practitioners to make accurate predictions and classify new data points based on their similarity to existing instances.

Thank You