Name : Syed Riaz Ali

Email : syedriazali1997@gmail.com

Whatsapp : 923002502513

Day_26 : Random Forest Algorithm

Random Forest Classification

In [30]:
```
# Importing libraries

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [31]:
```
# Loading the data set

df = sns.load_dataset("iris")
df.head()
```

Out[31]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [32]:
```
# Separating the input and output variables

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

In [33]:
```
# Fitting the model

from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators= 100)
model.fit(X,y)
model.predict([[5,4,2,6]])
```

```
C:\Users\syedriaz\AppData\Local\Temp/ipykernel_6768/258399899.py:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  model.fit(X,y)
C:\Users\syedriaz\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but RandomForestClassifier was fitted with feature names
  warnings.warn(
```

Out[33]:
```
array(['setosa'], dtype=object)
```

Definition of n_estimator

> This is the number of trees you want to build before taking the maximum voting or averages of predictions. Higher number of trees give you better performance but makes your code slower.

In [34]:
```
# Testing the model using train_test_split Technique

from sklearn.model_selection import train_test_split

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

predictions = model.predict(X_test)
predictions
```

Out[34]:
```
array(['virginica', 'virginica', 'virginica', 'virginica', 'setosa',
       'virginica', 'setosa', 'versicolor', 'virginica', 'setosa',
       'setosa', 'virginica', 'virginica', 'versicolor', 'versicolor',
       'versicolor', 'versicolor', 'virginica', 'versicolor', 'virginica',
       'versicolor', 'virginica', 'versicolor', 'setosa', 'virginica',
       'setosa', 'setosa', 'setosa', 'virginica', 'setosa'], dtype=object)
```

In [35]:
```
# Accuracy test

score = model.score(X_test, y_test)
print("The accuracy score is :  ", score)
```

```
The accuracy score is :  1.0
```

In [36]:
```
from sklearn import metrics
print("Accuracy :", metrics.accuracy_score(y_test, predictions))
```

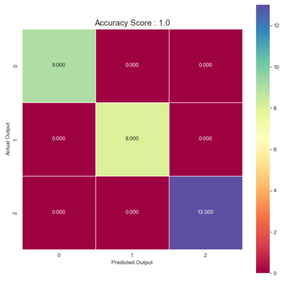
```
Accuracy : 1.0
```

In [37]:
```
# Confusion Matrix

from sklearn import metrics
cm = metrics.confusion_matrix(y_test, predictions)
cm
```

Out[37]:
```
array([[ 9,  0,  0],
       [ 0,  8,  0],
       [ 0,  0, 13]], dtype=int64)
```

In [38]:
```
# Plotting the confusion using Heatmap plot

plt.figure(figsize=(9,9))
sns.set_style("darkgrid")
sns.heatmap(cm, annot = True, fmt= ".3f", linewidth = .5, square = True, cmap = "Spectral")
plt.ylabel("Actual Output")
plt.xlabel("Predicted Output")
all_sample_title = "Accuracy Score : {0}".format(score)
plt.title(all_sample_title, size = 15)
```

Out[38]:
```
Text(0.5, 1.0, 'Accuracy Score : 1.0')
```



Random Forest Regressor

In [39]:
```
# Loading the data set

df2 = sns.load_dataset("iris")
df2.head()
```

Out[39]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [45]:
```
# Replacing Categorical data in label variable into numerical data
# 0 for setosa
# 1 for versicolor
# 2 for virginica

df2["species"] = df2["species"].replace("setosa", 0)
df2["species"] = df2["species"].replace("versicolor", 1)
df2["species"] = df2["species"].replace("virginica", 2)
df2.tail()
```

Out[45]:

	sepal_length	sepal_width	petal_length	petal_width	species
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

In [46]:
```
# Separating the input and output variables

X = df2.iloc[:, :-1]
y = df2.iloc[:, -1:]
```

In [47]:
```
y.head()
```

Out[47]:

	species
0	0
1	0
2	0
3	0
4	0

In [48]:
```
# Training and Fitting the data

from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor(n_estimators= 100)
model.fit(X,y)
model.predict([[5,4,2,6]])
```

```
C:\Users\syedriaz\AppData\Local\Temp/ipykernel_6768/1412340738.py:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  model.fit(X,y)
C:\Users\syedriaz\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but RandomForestRegressor was fitted with feature names
  warnings.warn(
```

Out[48]:
```
array([0.67])
```

In [50]:
```
# Testing the data using train_test_split technique

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor

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

model = RandomForestRegressor().fit(X_train,y_train)

predictions = model.predict(X_test)
predictions
```

```
C:\Users\syedriaz\AppData\Local\Temp/ipykernel_6768/2307444712.py:6: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  model = RandomForestRegressor().fit(X_train,y_train)
```

Out[50]:
```
array([2.  , 1.99, 0.  , 1.99, 2.  , 0.  , 2.  , 0.  , 1.  , 0.  , 1.98,
       1.92, 0.  , 1.  , 1.02, 0.  , 1.93, 0.  , 0.  , 0.  , 2.  , 1.01,
       1.  , 1.53, 1.98, 0.  , 1.01, 1.01, 1.48, 2.  ])
```

In [51]:
```
# Accuracy test

score = model.score(X_test, y_test)
print("The accuracy score is :  ", score)
```

```
The accuracy score is :  0.9787667682926829
```

In [ ]:
```

```