Day_25

January 26, 2022

```
[ ]: # load sample datasets
     import pandas as pd
     import seaborn as sns
     import numpy as np
     df= sns.load_dataset('iris')
     df.head()
```

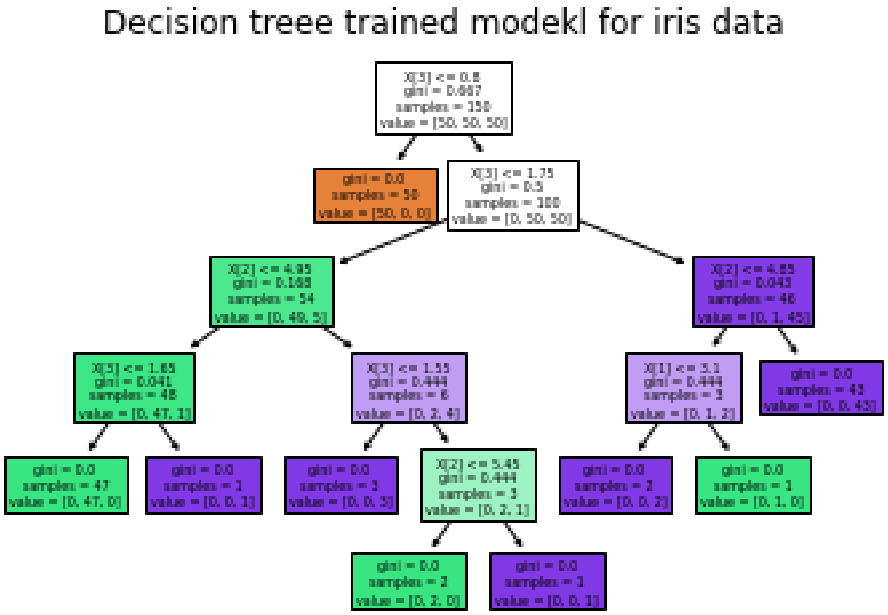
```
[ ]:    sepal_length  sepal_width  petal_length  petal_width species
     0           5.1          3.5           1.4          0.2  setosa
     1           4.9          3.0           1.4          0.2  setosa
     2           4.7          3.2           1.3          0.2  setosa
     3           4.6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

```
[ ]: import matplotlib.pyplot as plt

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

```
[ ]: from sklearn.tree import DecisionTreeClassifier
     from sklearn.tree import plot_tree
     model = DecisionTreeClassifier().fit(x,y)
     plot_tree(model, filled=True)
     plt.title('Decision treee trained modekl for iris data')
```

```
[ ]: Text(0.5, 1.0, 'Decision treee trained modekl for iris data')
```



1 Predictions from trained model

[ ]:

```
model.predict([[5,4,2,6]])
```

```
C:\Users\Haier\AppData\Local\Programs\Python\Python310\lib\site-
packages\sklearn\base.py:450: UserWarning: X does not have valid feature names,
but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

[ ]:

```
array(['versicolor'], dtype=object)
```

[ ]:

```
model.predict([[7,7,7,6]])
```

```
C:\Users\Haier\AppData\Local\Programs\Python\Python310\lib\site-
packages\sklearn\base.py:450: UserWarning: X does not have valid feature names,
but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

[ ]:

```
array(['virginica'], dtype=object)
```

[ ]:

```
# how to save this plot in png, hd , tiff hd qualities
# plt.savefig('decisiontree.png', dpi=300)
plt.savefig('tiff_compressed.tiff', dpi=600, format='tiff',
            facecolor='white', edgecolor='none',
            pil_kwargs={'compression':'tiff_lzw'})
plt.show()
```

```
<Figure size 432x288 with 0 Axes>
```

2 checking the model accuracy (20/80)

```
[ ]: 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
```

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

```
[ ]: score= model.score(x_test, y_test)
     print ('The accuracy score is', score)
```

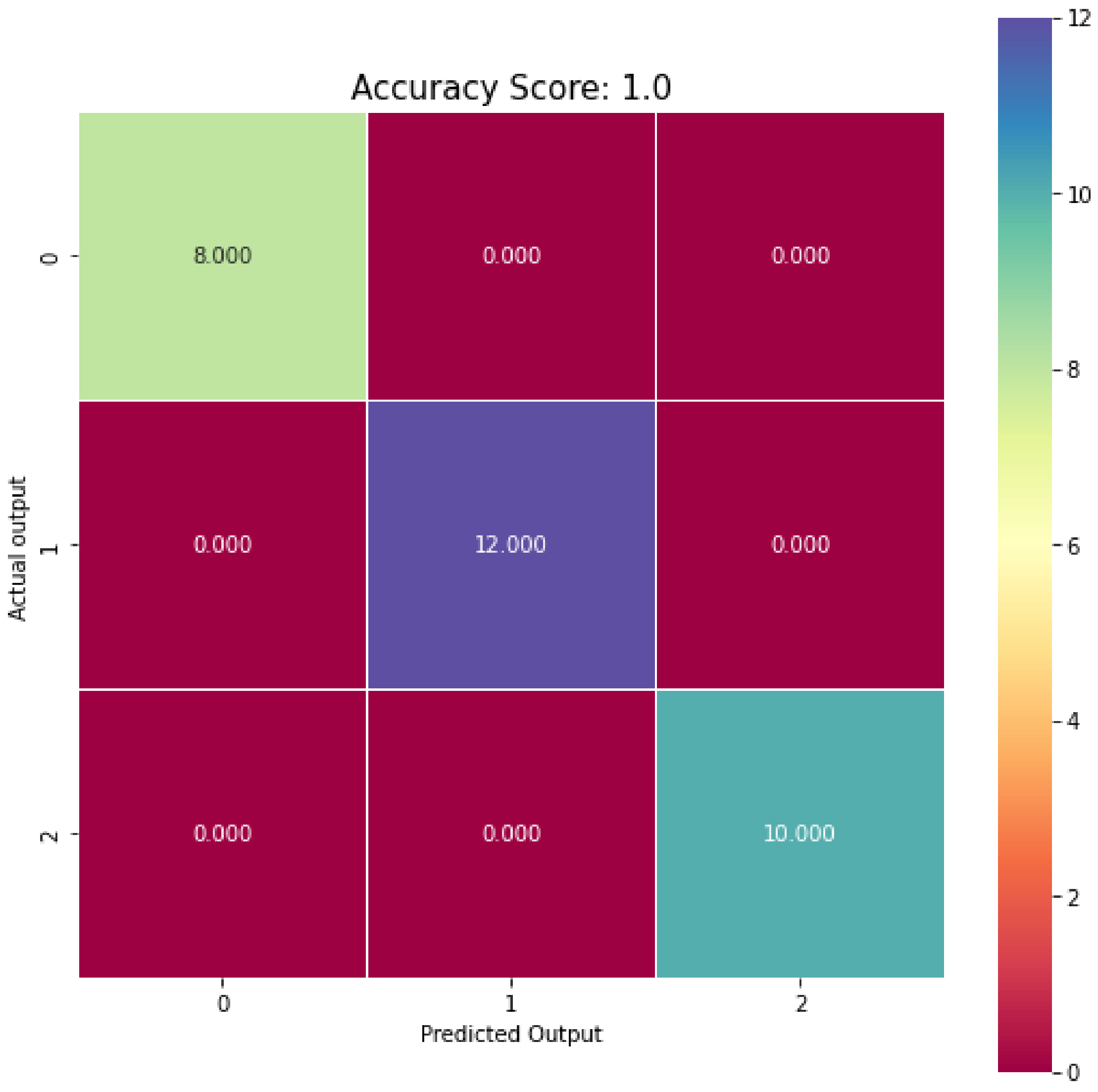
```
The accuracy score is 1.0
```

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

```
Accuracy: 1.0
```

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

```
[ ]: import seaborn as sns
     plt.figure(figsize=(9,9))
     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);
```



3 checking the model accuracy (30/70)

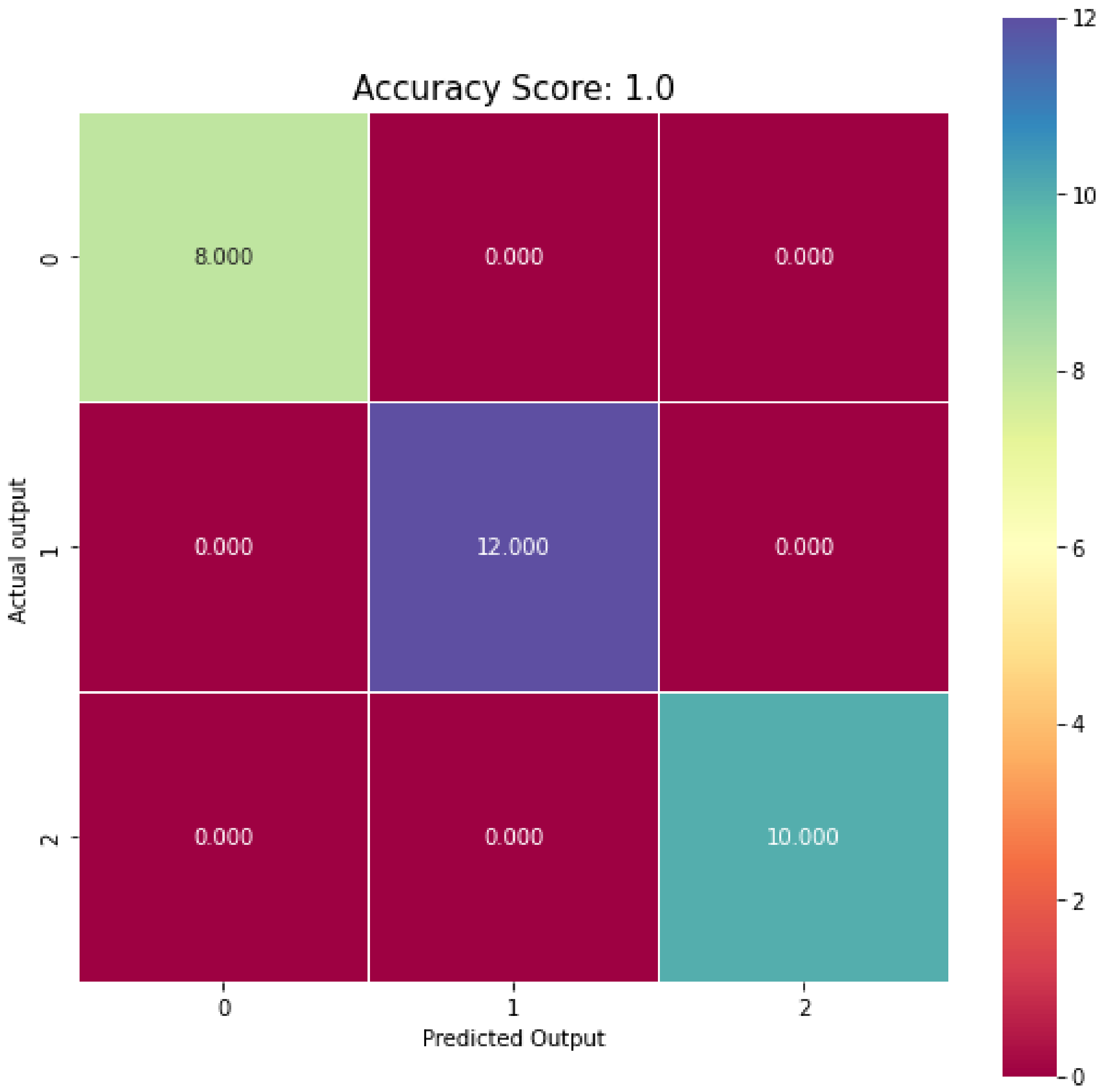
```
[ ]: from sklearn.model_selection import train_test_split
     x_train, x_test, y_train,  y_test= train_test_split(x,y, test_size=0.3)
     predictions= model.predict(x_test)
     predictions
```

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

```
[ ]: score= model.score(x_test, y_test)
     print ('The accuracy score is', score)
     from sklearn import metrics
     print("Accuracy:", metrics.accuracy_score(y_test, predictions))
```

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

```
[ ]: from sklearn import metrics
     cm = metrics.confusion_matrix(y_test, predictions)
     import seaborn as sns
     plt.figure(figsize=(9,9))
     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);
```



4 checking the model accuracy (10/90)

```
[ ]: from sklearn.model_selection import train_test_split
     x_train, x_test, y_train,  y_test= train_test_split(x,y, test_size=0.1)
     predictions= model.predict(x_test)
     predictions
```

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

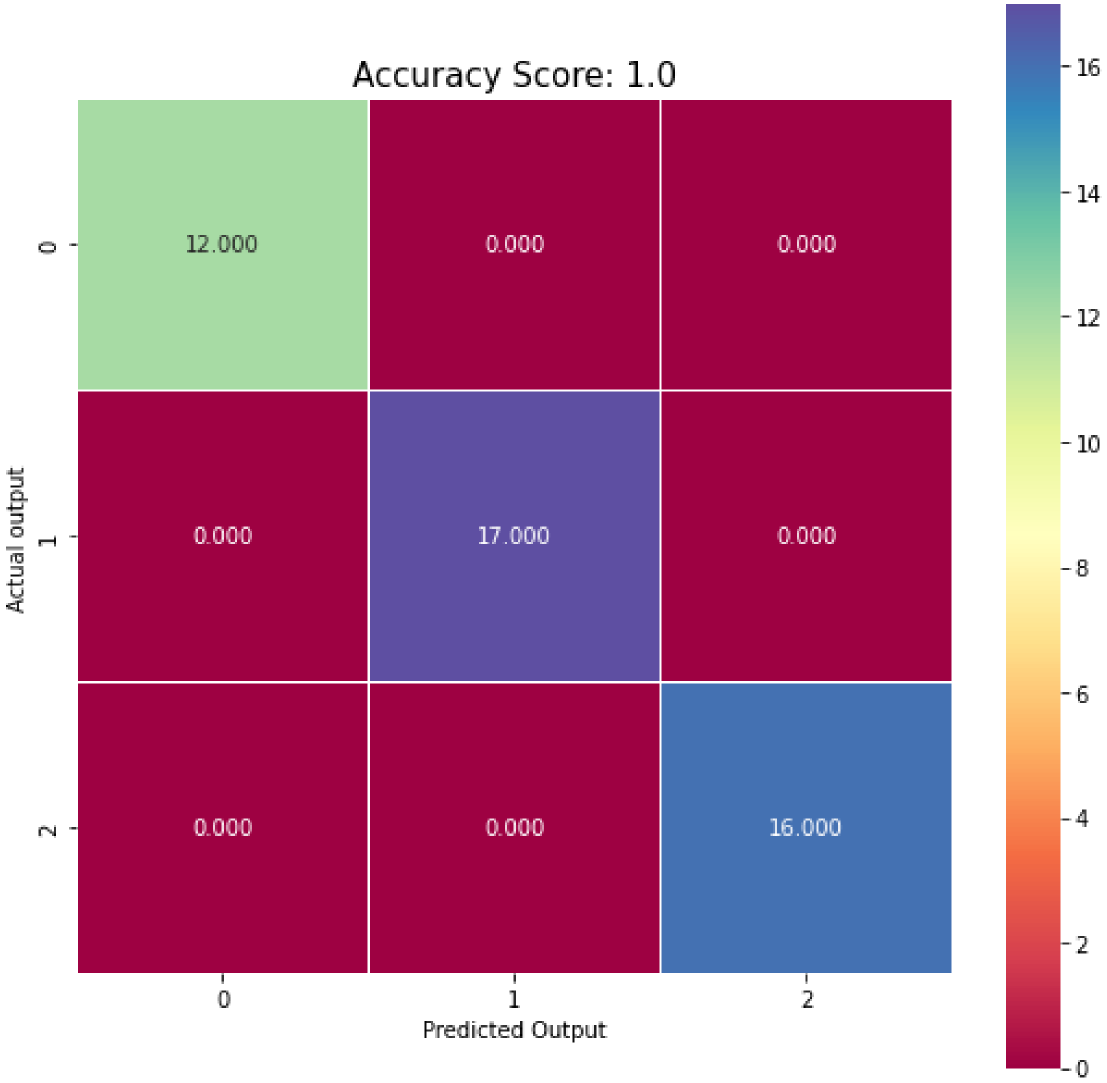
[ ]:

```python
score= model.score(x_test, y_test)
print ('The accuracy score is', score)
from sklearn import metrics
print("Accuracy:", metrics.accuracy_score(y_test, predictions))
```

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

[ ]:

```python
from sklearn import metrics
cm = metrics.confusion_matrix(y_test, predictions)
import seaborn as sns
plt.figure(figsize=(9,9))
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);
```


Accuracy Score: 1.0
12.000
0.000
0.000
0.000
17.000
0.000
0.000
0.000
16.000
Actual output
0
1
2
Predicted Output
0
1
2
16
14
12
10
8
6
4
2
0