In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

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

Out[ ]:

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

In [ ]:
```
X.head()
```

Out[ ]:

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

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

Out[ ]:

	species
0	setosa
1	setosa
2	setosa
3	setosa
4	setosa

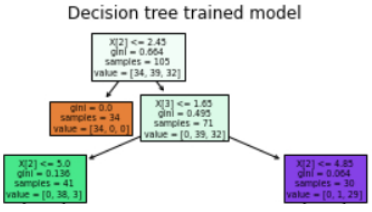
In [ ]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import plot_tree
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```

In [ ]:
```
# Splitting the dataset into train and test

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 100)
```

In [ ]:
```
# Decision tree classifier , fitting the training data
model = DecisionTreeClassifier(random_state = 100).fit(X_train, y_train)
```

In [ ]:
```
plot_tree(model, filled = True)
plt.title("Decision tree trained model")
plt.show()
```



In [ ]:
```
#plt.savefig("tiff_compressed.tiff", dpi=600, format = "tiff", facecolor = 'white', edgecolor = 'none',
#pil_kwargs = {"compression": "tiff_lzw"}  )
```

In [ ]:
```
# prediction on X_test (testing data )
y_pred  = model.predict(X_test)
y_pred
```

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

In [ ]:
```
#acuuracy score with 10/90
score = model.score(X_test, y_test)
print("Acuuracy score is : " ,score)
```

```
Acuuracy score is :  1.0
```

In [ ]:
```
#acuuracy score with 20/80
score = model.score(X_test, y_test)
print("Acuuracy score is : " ,score)
```

```
Acuuracy score is :  0.9666666666666667
```

In [ ]:
```
#acuuracy score with 30/70
score = model.score(X_test, y_test)
print("Acuuracy score is : " ,score)
```

```
Acuuracy score is :  0.9555555555555556
```

In [ ]:
```
#Accuray of the model

print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
cm= metrics.confusion_matrix(y_test, y_pred)
cm
```

```
Accuracy: 0.9555555555555556
```

Out[ ]:
```
array([[16,  0,  0],
       [ 0, 10,  1],
       [ 0,  1, 17]], dtype=int64)
```

In [ ]:
```
X_train.head(10)
```

Out[ ]:

	sepal_length	sepal_width	petal_length	petal_width
6	4.6	3.4	1.4	0.3
25	5.0	3.0	1.6	0.2
21	5.1	3.7	1.5	0.4
92	5.8	2.6	4.0	1.2
9	4.9	3.1	1.5	0.1
23	5.1	3.3	1.7	0.5
35	5.0	3.2	1.2	0.2
54	6.5	2.8	4.6	1.5
131	7.9	3.8	6.4	2.0
127	6.1	3.0	4.9	1.8

In [ ]:
```
#Sample Value Prediction:
print(model.predict([[5.1, 3.2, 1.9, 0.7], [5.1,3.5,1.4,0.2],[4.8,3.9,3.4,0.4],[4.1,3.1,2.6,0.7],[5.2, 3.6,2.9,2.3]]))
```

```
['setosa' 'setosa' 'versicolor' 'versicolor' 'versicolor']
```