In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn import metrics
import seaborn as sns
```

In [3]:
```
dataset = pd.read_csv('irisdatset.csv')
```

In [4]:
```
dataset
```

Out[4]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

In [5]:
```
x = dataset.iloc[:, [0,1,2,3]].values
x
```

Out[5]:
```
array([[  1. ,   5.1,   3.5,   1.4],
       [  2. ,   4.9,   3. ,   1.4],
       [  3. ,   4.7,   3.2,   1.3],
       [  4. ,   4.6,   3.1,   1.5],
       [  5. ,   5. ,   3.6,   1.4],
       [  6. ,   5.4,   3.9,   1.7],
       [  7. ,   4.6,   3.4,   1.4],
       [  8. ,   5. ,   3.4,   1.5],
       [  9. ,   4.4,   2.9,   1.4],
       [ 10. ,   4.9,   3.1,   1.5],
       [ 11. ,   5.4,   3.7,   1.5],
       [ 12. ,   4.8,   3.4,   1.6],
       [ 13. ,   4.8,   3. ,   1.4],
       [ 14. ,   4.3,   3. ,   1.1],
       [ 15. ,   5.8,   4. ,   1.2],
       [ 16. ,   5.7,   4.4,   1.5],
       [ 17. ,   5.4,   3.9,   1.3],
       [ 18. ,   5.1,   3.5,   1.4],
       [ 19. ,   5.7,   3.8,   1.7],
```

In [6]:
```
y = dataset.iloc[:, 5].values
y
```

```
Out[6]: array(['Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
               'Iris-setosa', 'Iris-setosa', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
               'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
```

```
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica'], dtype=object)
```

In [7]:
```
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.25, random_state=0)
sc = StandardScaler()
xtrain = sc.fit_transform(xtrain)
xtest = sc.transform(xtest)
```

In [9]:
```
dtree_gini = DecisionTreeClassifier(criterion = "gini", random_state = 100,max_depth=3,
min_samples_leaf=5)
dtree_gini.fit(xtrain, ytrain)
```

Out[9]:
```
DecisionTreeClassifier(max_depth=3, min_samples_leaf=5, random_state=100)
```

In [10]:
```
y_pred1 = dtree_gini.predict(xtest)
print("Predicted values:")
y_pred1
```

```
Predicted values:
```

Out[10]:
```
array(['Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-virginica', 'Iris-setosa', 'Iris-virginica', 'Iris-setosa',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-virginica', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',
       'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-setosa', 'Iris-virginica',
       'Iris-versicolor', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-versicolor'], dtype=object)
```

In [11]:
```
accgini= accuracy_score(ytest,y_pred1)*100
print ("\n\nAccuracy using Gini Index: ", accgini)
```

```
Accuracy using Gini Index:  100.0
```

In [12]:
```
cm = confusion_matrix(ytest, y_pred1)
print ("\n\n Confusion Matrix -using Gini Index: \n", cm)
```

```
 Confusion Matrix -using Gini Index:
 [[13  0  0]
 [ 0 16  0]
 [ 0  0  9]]
```

In [13]:
```
fig, ax = plt.subplots(figsize=(6, 6))
ax.imshow(cm)
ax.grid(False)
ax.xaxis.set(ticks=(0,1,2), ticklabels=('Predicted Setosa', 'Predicted Versicolor', 'Predicted Virginica'))
ax.yaxis.set(ticks=(0,1,2), ticklabels=('Actual Setosa', 'Actual Versicolor', 'Actual Virginica'))
ax.set_ylim(2.5, -0.5)
for i in range(3):
    for j in range(3):
        ax.text(j, i, cm[i, j], ha='center', va='center', color='white')
plt.show()
```

In [14]:
```
print("\n\nClassification Report - Using Gini Index: \n",classification_report(ytest, y_pred1))
```

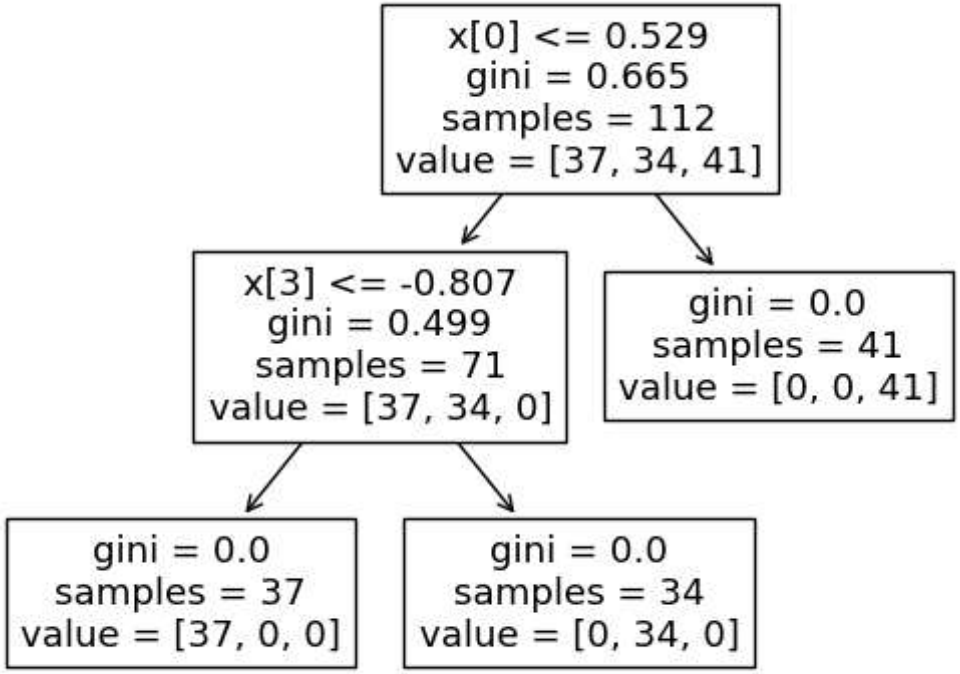
```
Classification Report - Using Gini Index:
                  precision    recall  f1-score   support

     Iris-setosa       1.00      1.00      1.00        13
 Iris-versicolor       1.00      1.00      1.00        16
  Iris-virginica       1.00      1.00      1.00         9

        accuracy                           1.00        38
       macro avg       1.00      1.00      1.00        38
    weighted avg       1.00      1.00      1.00        38
```

In [15]:
```
tree.plot_tree(dtree_gini)
```

Out[15]:
```
[Text(0.6, 0.8333333333333334, 'x[0] <= 0.529\ngini = 0.665\nsamples = 112\nvalue = [37, 34, 41]'),
 Text(0.4, 0.5, 'x[3] <= -0.807\ngini = 0.499\nsamples = 71\nvalue = [37, 34, 0]'),
 Text(0.2, 0.16666666666666666, 'gini = 0.0\nsamples = 37\nvalue = [37, 0, 0]'),
 Text(0.6, 0.16666666666666666, 'gini = 0.0\nsamples = 34\nvalue = [0, 34, 0]'),
 Text(0.8, 0.5, 'gini = 0.0\nsamples = 41\nvalue = [0, 0, 41]')]
```



In [16]:
```
#using Entropy
dtree_entropy = DecisionTreeClassifier(criterion = "entropy", random_state = 100,
max_depth = 3, min_samples_leaf = 5)
dtree_entropy.fit(xtrain, ytrain)
y_pred2 = dtree_entropy.predict(xtest)
```

In [17]:
```
print("Predicted values:")
print(y_pred2)
```

```
Predicted values:
['Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-virginica'
 'Iris-setosa' 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-setosa' 'Iris-setosa' 'Iris-versicolor'
 'Iris-versicolor' 'Iris-setosa' 'Iris-virginica' 'Iris-versicolor'
 'Iris-setosa' 'Iris-virginica' 'Iris-virginica' 'Iris-versicolor'
 'Iris-setosa' 'Iris-versicolor']
```

In [18]:
```
acc_entropy= accuracy_score(ytest,y_pred2)*100
print ("\n\nAccuracy using Entropy: ", acc_entropy)
```

```
Accuracy using Entropy:  100.0
```

In [19]:
```
cm = confusion_matrix(ytest, y_pred2)
print ("\n\n Confusion Matrix -using Entropy: \n", cm)
```

```
 Confusion Matrix -using Entropy:
 [[13  0  0]
 [ 0 16  0]
 [ 0  0  9]]
```

In [20]:
```
fig, ax = plt.subplots(figsize=(6, 6))
ax.imshow(cm)
ax.grid(False)
ax.xaxis.set(ticks=(0,1,2), ticklabels=('Predicted Setosa', 'Predicted Versicolor', 'Predicted Virginica'))
ax.yaxis.set(ticks=(0,1,2), ticklabels=('Actual Setosa', 'Actual Versicolor', 'Actual Virginica'))
ax.set_ylim(2.5, -0.5)
for i in range(3):
    for j in range(3):
        ax.text(j, i, cm[i, j], ha='center', va='center', color='white')
plt.show()
```

In [21]:
```
print("Classification Report - Using Entropy: \n",classification_report(ytest, y_pred2))
```

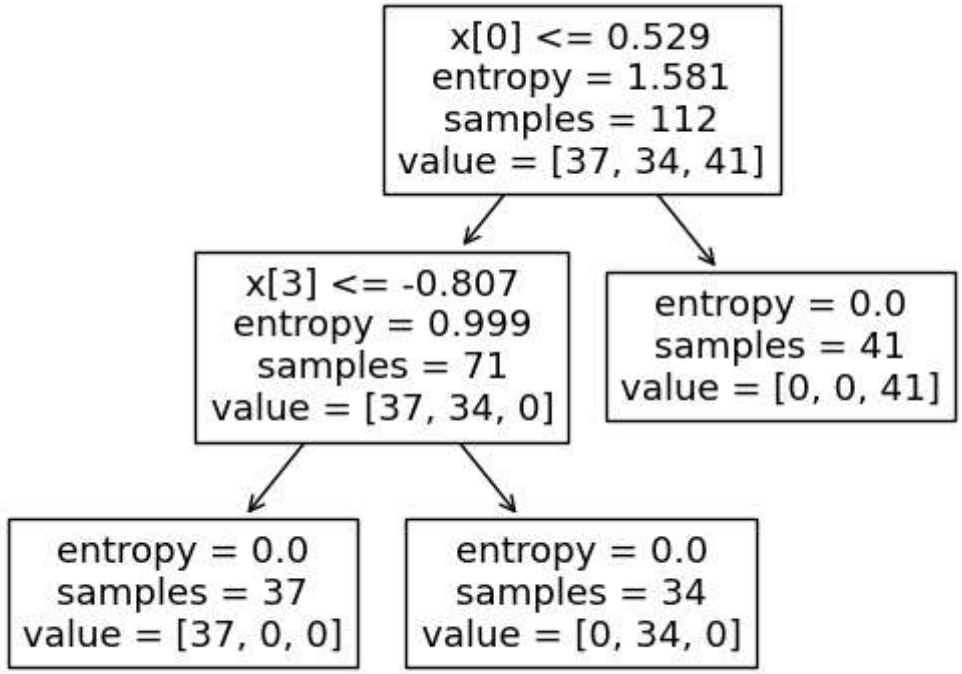
```
Classification Report - Using Entropy:
                  precision    recall  f1-score   support

     Iris-setosa       1.00      1.00      1.00        13
 Iris-versicolor       1.00      1.00      1.00        16
  Iris-virginica       1.00      1.00      1.00         9

        accuracy                           1.00        38
       macro avg       1.00      1.00      1.00        38
    weighted avg       1.00      1.00      1.00        38
```

In [22]:
```
tree.plot_tree(dtree_entropy)
```

Out[22]:
```
[Text(0.6, 0.8333333333333334, 'x[0] <= 0.529\nentropy = 1.581\nsamples = 112\nvalue = [37, 34, 41]'),
 Text(0.4, 0.5, 'x[3] <= -0.807\nentropy = 0.999\nsamples = 71\nvalue = [37, 34, 0]'),
 Text(0.2, 0.16666666666666666, 'entropy = 0.0\nsamples = 37\nvalue = [37, 0, 0]'),
 Text(0.6, 0.16666666666666666, 'entropy = 0.0\nsamples = 34\nvalue = [0, 34, 0]'),
 Text(0.8, 0.5, 'entropy = 0.0\nsamples = 41\nvalue = [0, 0, 41]')]
```



In [ ]:

In [ ]: