[22]:
```
#imports
from sklearn.datasets import load_iris
from sklearn import tree
import pandas as pd
```

[23]:
```
data = load_iris()
```

[24]:
```
type(data)
```

[24]:
```
sklearn.utils.Bunch
```

[34]:
```
data = load_iris()
df = pd.DataFrame(data=data.data, columns=data.feature_names)
df['target'] = pd.DataFrame(data = data.target)
```

[35]:
```
df.head()
```

[35]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

[48]:
```
X = df[["sepal length (cm)" , "sepal width (cm)","petal length (cm)","petal width (cm)"]]
y = pd.DataFrame(data = df["target"])
```

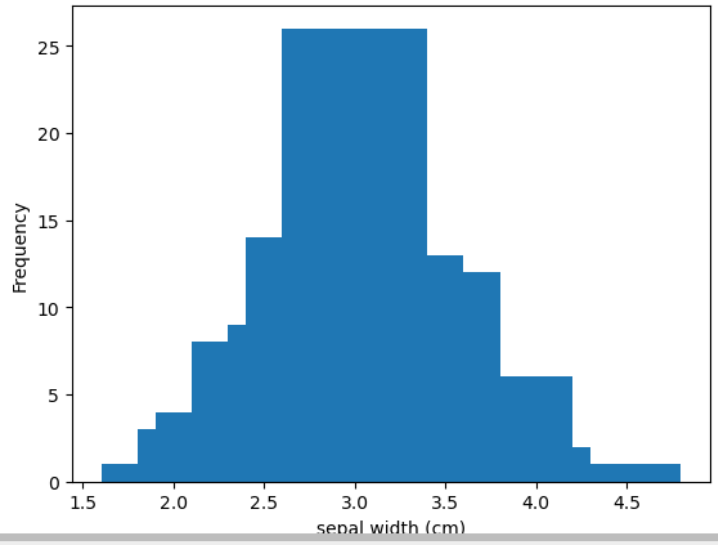
[49]:
```
for column in df.columns:
    print(f"Number of null values in {column}: {df[column].isnull().sum()}")
```

```
Number of null values in sepal length (cm): 0
Number of null values in sepal width (cm): 0
Number of null values in petal length (cm): 0
Number of null values in petal width (cm): 0
Number of null values in target: 0
```

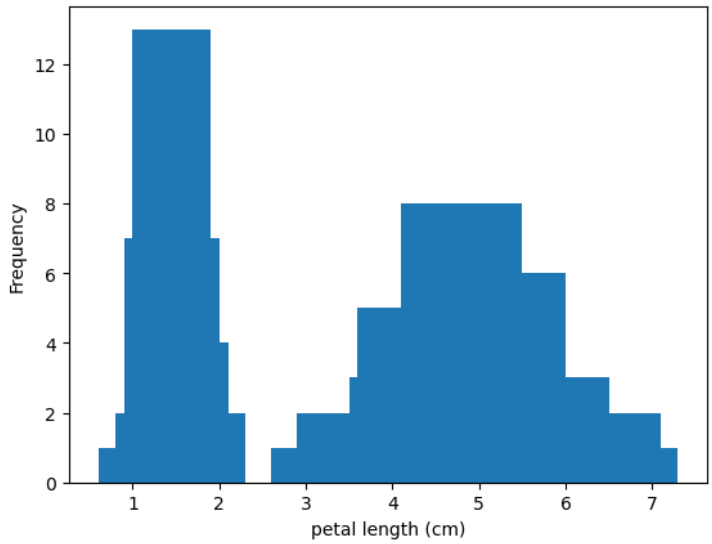
[50]:
```
fig, ax = plt.subplots()
ax.bar(X["sepal length (cm)"].value_counts().index, X["sepal length (cm)"].value_counts() )
ax.set_xlabel("sepal length (cm)")
ax.set_ylabel('Frequency')
```

[50]:
```
Text(0, 0.5, 'Frequency')
```

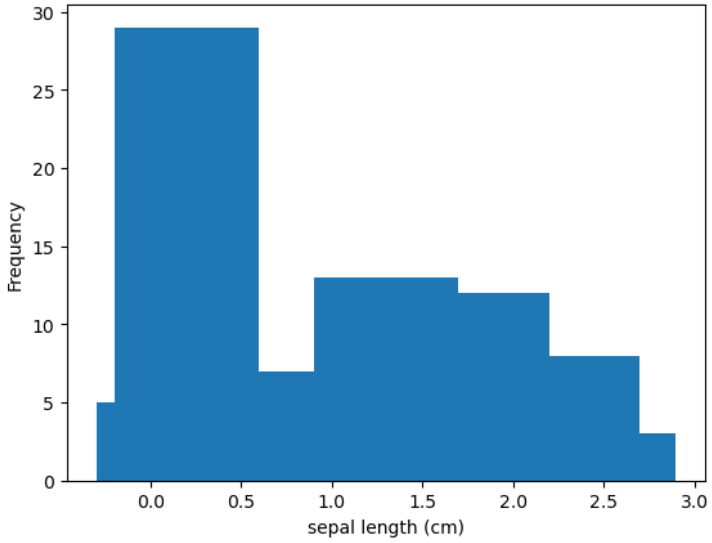
[51]:
```
fig, ax = plt.subplots()
ax.bar(X["sepal width (cm)"].value_counts().index, X["sepal width (cm)"].value_counts().values)
ax.set_xlabel("sepal width (cm)")
ax.set_ylabel('Frequency')
plt.show()
```



[52]:
```
fig, ax = plt.subplots()
ax.bar(X["petal length (cm)"].value_counts().index, X["petal length (cm)"].value_counts().values)
ax.set_xlabel("petal length (cm)")
ax.set_ylabel('Frequency')
plt.show()
```

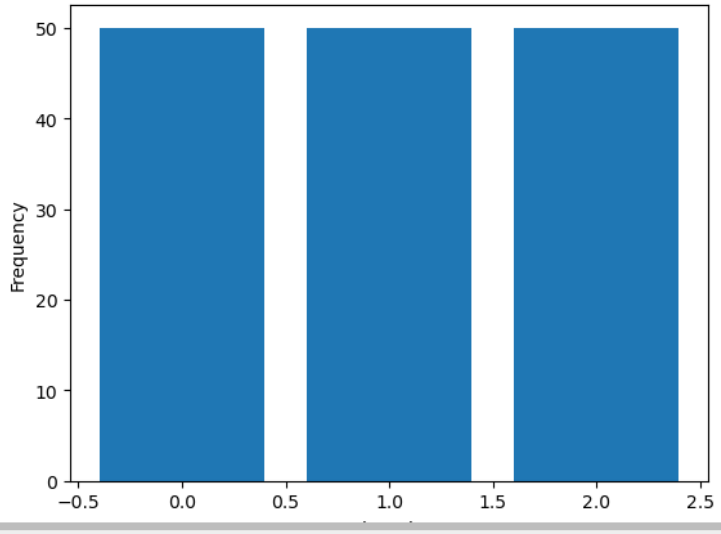


[53]:
```
fig, ax = plt.subplots()
ax.bar(X["petal width (cm)"].value_counts().index, X["petal width (cm)"].value_counts().values)
ax.set_xlabel("sepal length (cm)")
ax.set_ylabel('Frequency')
plt.show()
```



sepal length (cm)

[54]:
```
fig, ax = plt.subplots()
ax.bar(y["target"].value_counts().index, y["target"].value_counts().values)
ax.set_xlabel("target")
ax.set_ylabel('Frequency')
plt.show()
```



[57]:
```
X_train, X_test, y_train ,y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

[58]:
```
DTC_Model =DecisionTreeClassifier(criterion= 'entropy', max_depth=6, random_state=10)

DTC_Model.fit(X_train, y_train)
```

[58]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=6, random_state=10)
```

[62]:
```
DTC_Model.score(X_train , y_train)
```

[62]:
```
1.0
```

[63]:
```
y_pred = DTC_Model.predict(X_test)
```

[64]:
```
print(y_pred)

# now print the actual labels
print(y_test)
```

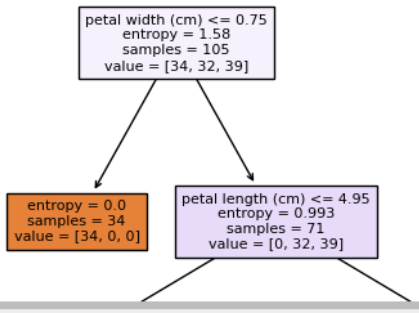
```
[2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0 2 1 0 2 2 1 0
 2 1 1 2 0 2 0 0]
     target
114       2
62        1
33        0
107       2
7         0
100       2
40        0
10        0
```

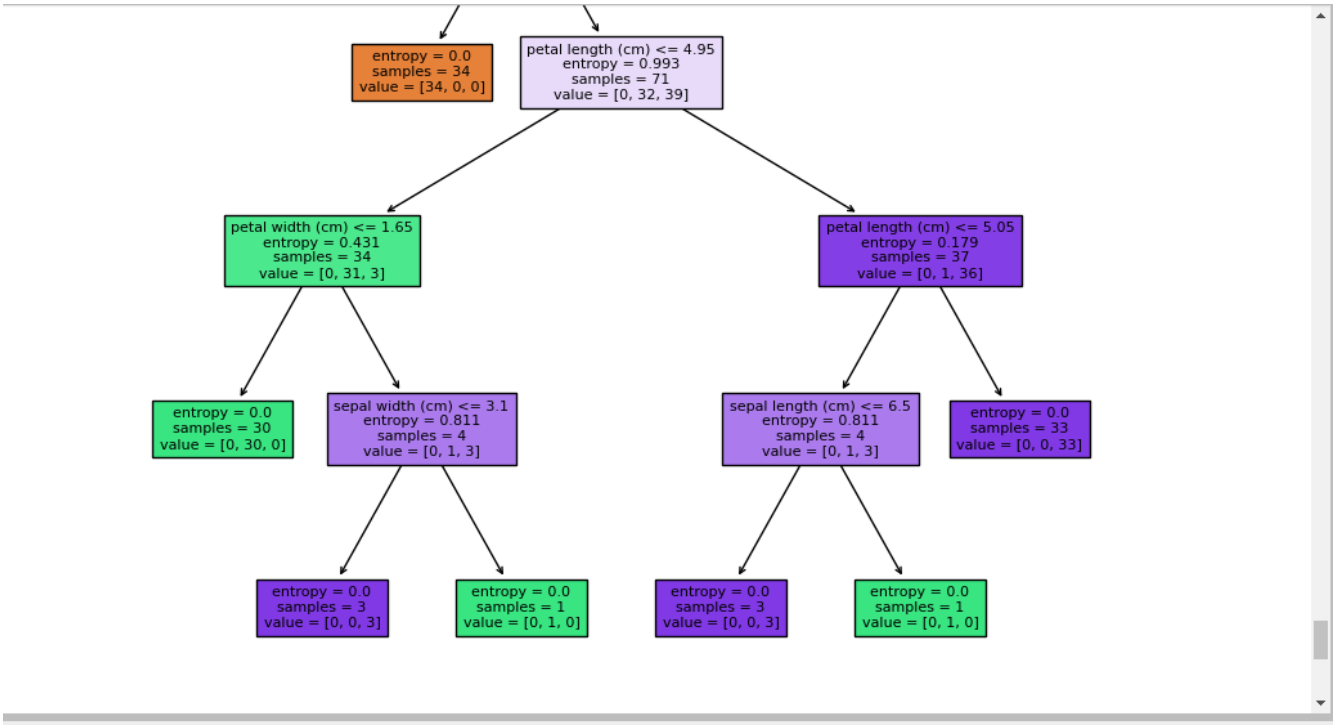
[65]:
```
DTC_Model.score(X_test , y_test)
```

[65]:
```
0.9777777777777777
```

[67]:
```
plt.figure(figsize=(10, 9))

DTC_tree = tree.plot_tree(DTC_Model, filled=True,
                          feature_names=['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (
                          fontsize=8,
                          )
```





[68]:
```
DTC_Model =DecisionTreeClassifier(criterion= 'gini', max_depth=6, random_state=10)

DTC_Model.fit(X_train, y_train)
```

[68]:
```
DecisionTreeClassifier(max_depth=6, random_state=10)
```

[69]:
```
y_pred = DTC_Model.predict(X_test)
```

[70]:
```
print(y_pred)
```

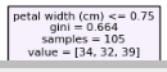
```
[2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0 2 1 0 2 2 1 0
 2 1 1 2 0 2 0 0]
```

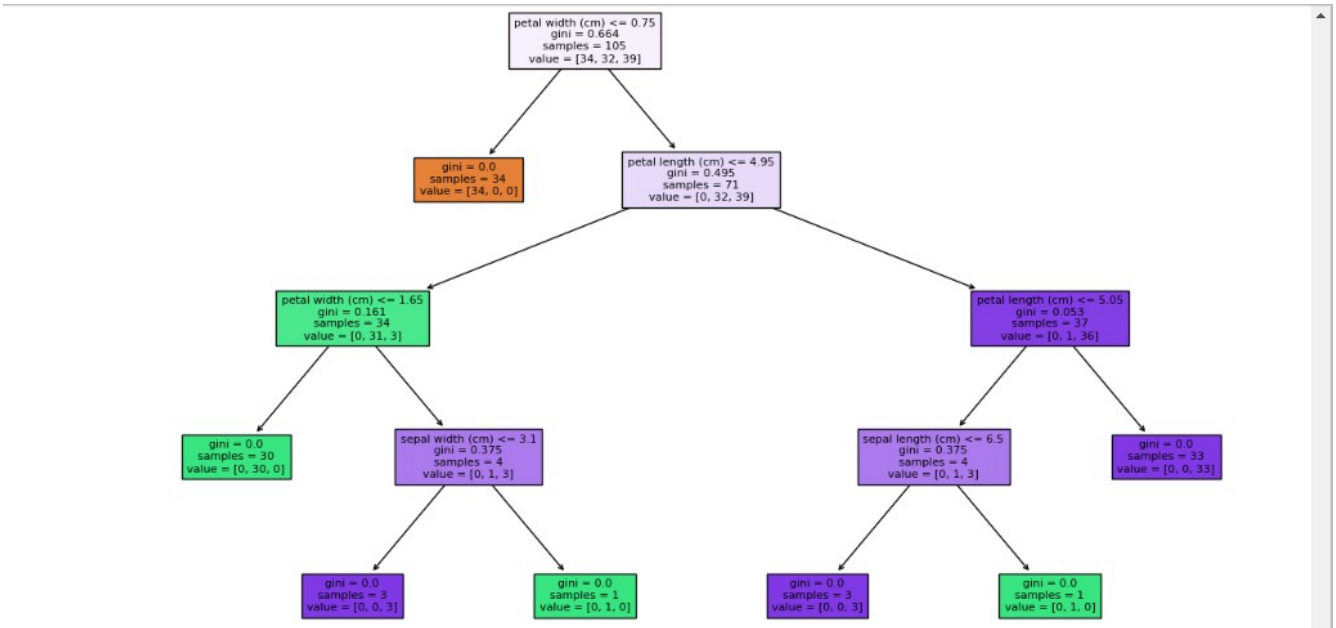
[71]:
```
DTC_Model.score(X_test , y_test)
```

[71]:
```
0.9777777777777777
```

[72]:
```
plt.figure(figsize=(15, 9))

DTC_tree = tree.plot_tree(DTC_Model, filled=True,
                          feature_names=['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (
                          fontsize=8,
                          )
```





Both have same accuracy