LAB TASK 5

In [1]:

```
import pandas as pd
import numpy as np

# Visualization imports
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
from sklearn.linear_model import Perceptron
from sklearn.metrics import accuracy_score, precision_score,recall_score, f1_score



from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

from sklearn.linear_model import Perceptron


#imports
from sklearn.datasets import load_iris
from sklearn import tree
import pandas as pd
```

In [2]:

```
data = load_iris()
df = pd.DataFrame(data=data.data, columns=data.feature_names)
```

In [3]:

```
df
```

Out[3]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
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
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

In [4]:

```
data.target
```

Out[4]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [5]:

```
data.target_names
```

Out[5]:

```
array(['setosa', 'versicolor', 'virginica'], dtype='<U10')
```

In [6]:

```
data.target
```

Out[6]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [7]:

```
df['target'] = data.target
```

In [8]:

```
#dropping all the 2's

df = df[df.target != 2]
```

In [9]:

```
df['target'].unique()
```

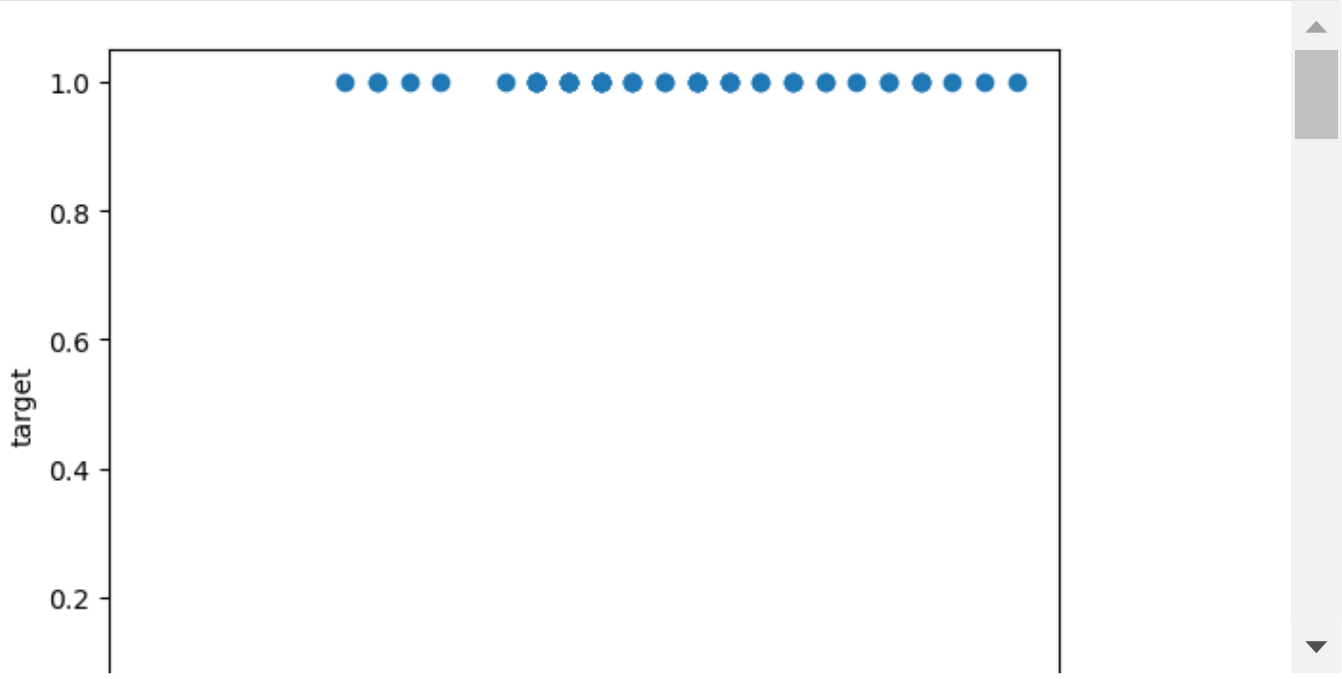
Out[9]:

```
array([0, 1])
```

In [10]:

```
for column in df.columns:

        fig, ax = plt.subplots()
        ax.scatter(df[column], df['target'])
        ax.set_xlabel(column)
        ax.set_ylabel('target')
        plt.show()
```



In [11]:

```
for i in df.columns:
    print(i)
```

```
sepal length (cm)
sepal width (cm)
petal length (cm)
petal width (cm)
target
```

In [12]:

```
X = df[['sepal length (cm)','sepal width (cm)','petal length (cm)','petal width (cm)']]
Y = df.target
```

In [13]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state=0)

p= Perceptron()
p.fit(X_train, Y_train)
```

Out[13]:

```
Perceptron()
```

In [14]:

```
y=df.target
df=df.drop('target',axis='columns')
X = df[['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)','petal width (cm)']
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.3, random_state=0)
clf=Perceptron()
clf.fit(X_train,y_train)
Perceptron()
y_pred=clf.predict(X_test)
print(f"Accuracy score: {accuracy_score(y_test, y_pred)}")
print(f"Precision score: {precision_score(y_test, y_pred,average='weighted')}")
print(f"Recall score: {recall_score(y_test, y_pred,average='weighted')}")
print(f"F1 score: {f1_score(y_test, y_pred, average='weighted')}")
```

```
Accuracy score: 1.0
Precision score: 1.0
Recall score: 1.0
F1 score: 1.0
```

In [15]:

```
x1=np.array(df)
def predict(row, weights):
 activation = weights[0]
 for i in range(len(row)-1):
     activation += weights[i + 1] * row[i]
     return 1.0 if activation >= 0.0 else 0.0
weights = [-358.00000000000995, 173.59999999999948, -1747.400000000064, 4242.90000000014
for row in x1:
 prediction = predict(row, weights)
 print("Expected=%d, Predicted=%d" % (row[-1], prediction))
```

```
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=0, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
```

```
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
Expected=1, Predicted=1
```

In [16]:

```
def train_weights(train, l_rate, n_epoch):
    weights = [0.0 for i in range(len(train[0]))]
    for epoch in range(n_epoch):
        sum_error = 0.0

        for row in train:

            prediction = predict(row, weights)

            error = row[-1] - prediction

            sum_error += error**2

            weights[0] = weights[0] + l_rate * error #bias(t+1) = bias(t) + learning_rat

            for i in range(len(row)-1):

                weights[i + 1] = weights[i + 1] + l_rate * error *row[i] #w(t+1) = w(t)

        print('epoch=%d, lrate=%.3f, error=%.3f' % (epoch, l_rate,sum_error))

    return weights
l_rate = 1
n_epoch = 100
weights = train_weights(x1, l_rate, n_epoch)
print(weights)
```

```
epoch=0, lrate=1.000, error=19.800
epoch=1, lrate=1.000, error=28.600
epoch=2, lrate=1.000, error=29.400
epoch=3, lrate=1.000, error=29.400
epoch=4, lrate=1.000, error=27.200
epoch=5, lrate=1.000, error=26.800
epoch=6, lrate=1.000, error=26.800
epoch=7, lrate=1.000, error=26.600
epoch=8, lrate=1.000, error=26.800
epoch=9, lrate=1.000, error=26.600
epoch=10, lrate=1.000, error=26.800
epoch=11, lrate=1.000, error=26.600
epoch=12, lrate=1.000, error=26.200
epoch=13, lrate=1.000, error=26.600
epoch=14, lrate=1.000, error=26.800
epoch=15, lrate=1.000, error=26.000
epoch=16, lrate=1.000, error=28.600
epoch=17, lrate=1.000, error=26.200
epoch=18, lrate=1.000, error=28.600
epoch=19, lrate=1.000, error=26.200
epoch=20, lrate=1.000, error=28.600
epoch=21, lrate=1.000, error=26.400
epoch=22, lrate=1.000, error=26.800
epoch=23, lrate=1.000, error=26.400
epoch=24, lrate=1.000, error=26.600
epoch=25, lrate=1.000, error=26.600
epoch=26, lrate=1.000, error=26.600
epoch=27, lrate=1.000, error=26.600
epoch=28, lrate=1.000, error=26.600
epoch=29, lrate=1.000, error=26.600
epoch=30, lrate=1.000, error=26.600
epoch=31, lrate=1.000, error=26.600
epoch=32, lrate=1.000, error=26.800
epoch=33, lrate=1.000, error=25.600
epoch=34, lrate=1.000, error=26.000
epoch=35, lrate=1.000, error=26.200
epoch=36, lrate=1.000, error=27.000
epoch=37, lrate=1.000, error=26.800
epoch=38, lrate=1.000, error=26.200
epoch=39, lrate=1.000, error=26.000
epoch=40, lrate=1.000, error=26.200
epoch=41, lrate=1.000, error=26.400
epoch=42, lrate=1.000, error=26.800
epoch=43, lrate=1.000, error=25.600
epoch=44, lrate=1.000, error=26.000
epoch=45, lrate=1.000, error=26.200
epoch=46, lrate=1.000, error=26.400
epoch=47, lrate=1.000, error=26.800
epoch=48, lrate=1.000, error=25.600
epoch=49, lrate=1.000, error=26.000
epoch=50, lrate=1.000, error=26.400
epoch=51, lrate=1.000, error=26.200
epoch=52, lrate=1.000, error=26.600
epoch=53, lrate=1.000, error=25.600
epoch=54, lrate=1.000, error=26.000
epoch=55, lrate=1.000, error=26.400
epoch=56, lrate=1.000, error=26.200
epoch=57, lrate=1.000, error=26.800
```

In [ ]:

In [ ]: