Sri Sivasubramaniya Nadar College of Engineering, Kalavakkam – 603 110 (An Autonomous Institution, Affiliated to Anna University, Chennai)

UCS2612 Machine Learning Laboratory

Academic Year: 2023-2024

Even Batch: 2021-2025

Faculty In-charges: Y.V. Lokeswari & Nilu R Salim

A. No. : 1 Working with Python packages - Numpy, Scipy,Scikit-learn, Matplotlib

Name : Y.V.Ojus

Register number : 3122 21 5001 125

Class : CSE - B

Link to Access : https://colab.research.google.com/drive/1Cvc7Nn5TJjDzpPdZbjxFpxBc69SkcQze?usp=sharing

Mounting Drive

In [1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Importing Dependencies

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

Reading File

In [ ]:
```
df = pd.read_csv("/content/drive/MyDrive/Sem VI/ML Lab/Datasets/Iris.csv")
df.head()
```

Rudimentary Method

In this case we randomly sample data point and find points which we randomly sample data points belonging to each class

In [ ]:
```
sample1 = df.loc[np.random.randint(0,49)]
sample2 = df.loc[np.random.randint(50,99)]
```

```
sample3 = df.loc[np.random.randint(100,149)]
```

Filter out points which are close by to the randomly generated point Based on Sepal Data

In [ ]:
```
for idx,row in df.iterrows():
  if(((sample1['SepalLengthCm']-0.1)<= row['SepalLengthCm']) and (row['SepalLengthCm']
    if(((sample1['SepalWidthCm']-0.1)<= row['SepalWidthCm']) and (row['SepalWidthCm']<
      print(row)
      print("\n\n\n")
```

```
Id                          23
SepalLengthCm              4.6
SepalWidthCm               3.6
PetalLengthCm              1.0
PetalWidthCm               0.2
Species            Iris-setosa
Name: 22, dtype: object
```

Filter out points which are close by to the randomly generated point Based on Petal Data

In [ ]:
```
for idx,row in df.iterrows():
  if(((sample1['PetalLengthCm']-0.1)<= row['PetalLengthCm']) and (row['PetalLengthCm']
    if(((sample1['PetalWidthCm']-0.1)<= row['PetalWidthCm']) and (row['PetalWidthCm']<
      print(row)
      print("\n\n\n")
```

```
Id                          14
SepalLengthCm              4.3
SepalWidthCm               3.0
PetalLengthCm              1.1
PetalWidthCm               0.1
Species            Iris-setosa
Name: 13, dtype: object
```

```
Id                          23
SepalLengthCm              4.6
SepalWidthCm               3.6
PetalLengthCm              1.0
PetalWidthCm               0.2
Species            Iris-setosa
Name: 22, dtype: object
```

In [ ]:
```
print(sample1)
```

```
Id                      23
SepalLengthCm          4.6
SepalWidthCm           3.6
PetalLengthCm          1.0
PetalWidthCm           0.2
Species          Iris-setosa
Name: 22, dtype: object
```

In [ ]:
```
print(sample2)
```

```
Id                       70
SepalLengthCm           5.6
SepalWidthCm            2.5
PetalLengthCm           3.9
PetalWidthCm            1.1
Species          Iris-versicolor
Name: 69, dtype: object
```

In [ ]:
```
print(sample3)
```

```
Id                      103
SepalLengthCm           7.1
SepalWidthCm            3.0
PetalLengthCm           5.9
PetalWidthCm            2.1
Species          Iris-virginica
Name: 102, dtype: object
```

Exploratory Data Analysis

Basic

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

In [ ]:
```
df['Species'].value_counts()
```

Out[ ]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

In [ ]:
```
df['SepalLengthCm'].describe()
```

Out[ ]:
```
count    150.000000
mean       5.843333
std        0.828066
min        4.300000
25%        5.100000
50%        5.800000
75%        6.400000
max        7.900000
Name: SepalLengthCm, dtype: float64
```

In [ ]:
```
df['PetalLengthCm'].describe()
```

Out[ ]:
```
count    150.000000
mean       3.758667
std        1.764420
min        1.000000
25%        1.600000
50%        4.350000
75%        5.100000
max        6.900000
Name: PetalLengthCm, dtype: float64
```

In [ ]:
```
df['SepalWidthCm'].describe()
```

Out[ ]:
```
count    150.000000
mean       3.054000
std        0.433594
min        2.000000
25%        2.800000
50%        3.000000
75%        3.300000
max        4.400000
Name: SepalWidthCm, dtype: float64
```

In [ ]:
```
df['PetalWidthCm'].describe()
```

Out[ ]:
```
count    150.000000
mean       1.198667
std        0.763161
min        0.100000
25%        0.300000
50%        1.300000
75%        1.800000
max        2.500000
Name: PetalWidthCm, dtype: float64
```

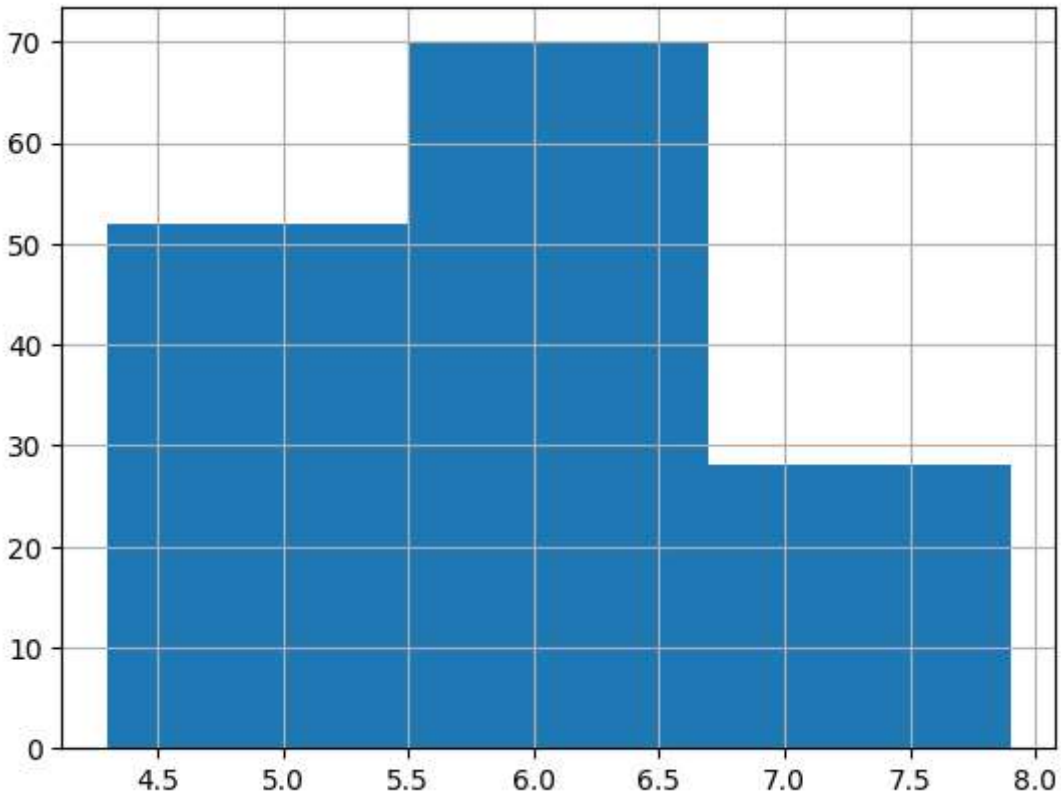
Plotting

Sepal Length

Histogram

In [ ]:
```
df['SepalLengthCm'].hist(bins=3)
```

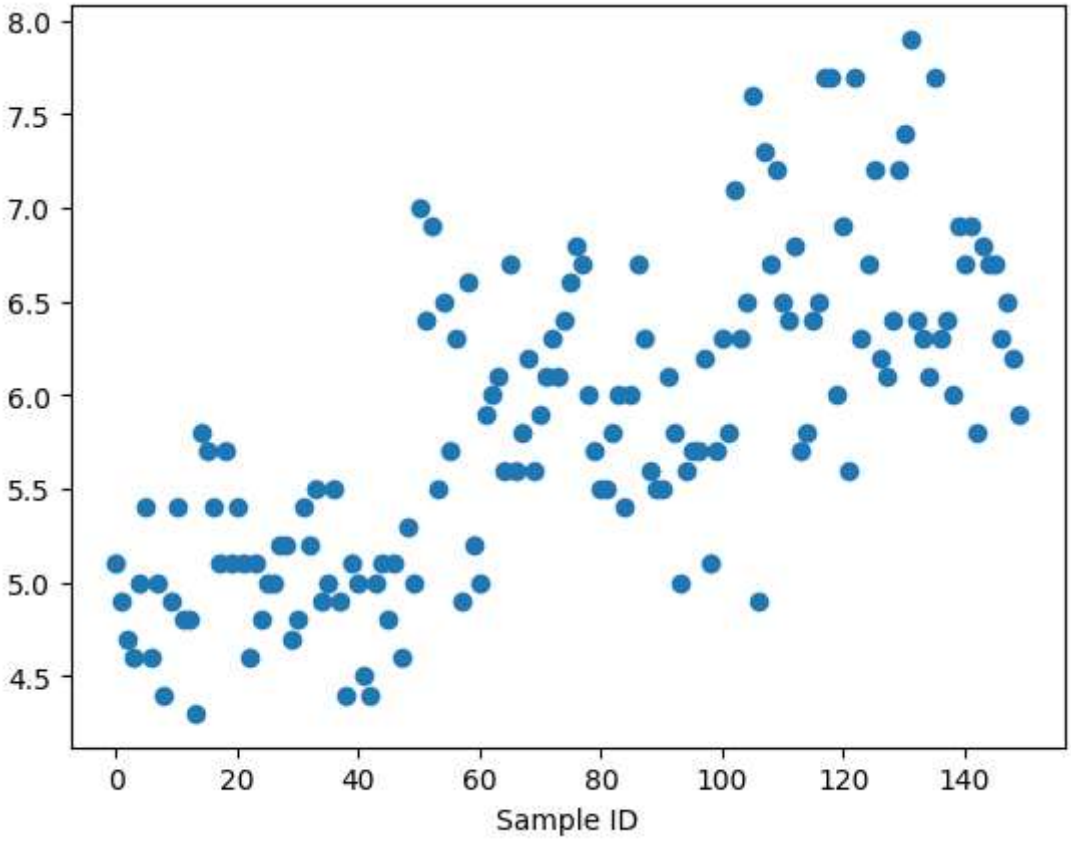
Out[ ]:
```
<Axes: >
```



Scatter Plot

In [ ]:
```
plt.plot(df['SepalLengthCm'],'o')
plt.xlabel('Sample ID')
plt.ylabel('Sepal Length')
```

Out[ ]:
```
Text(0.5, 0, 'Sample ID')
```

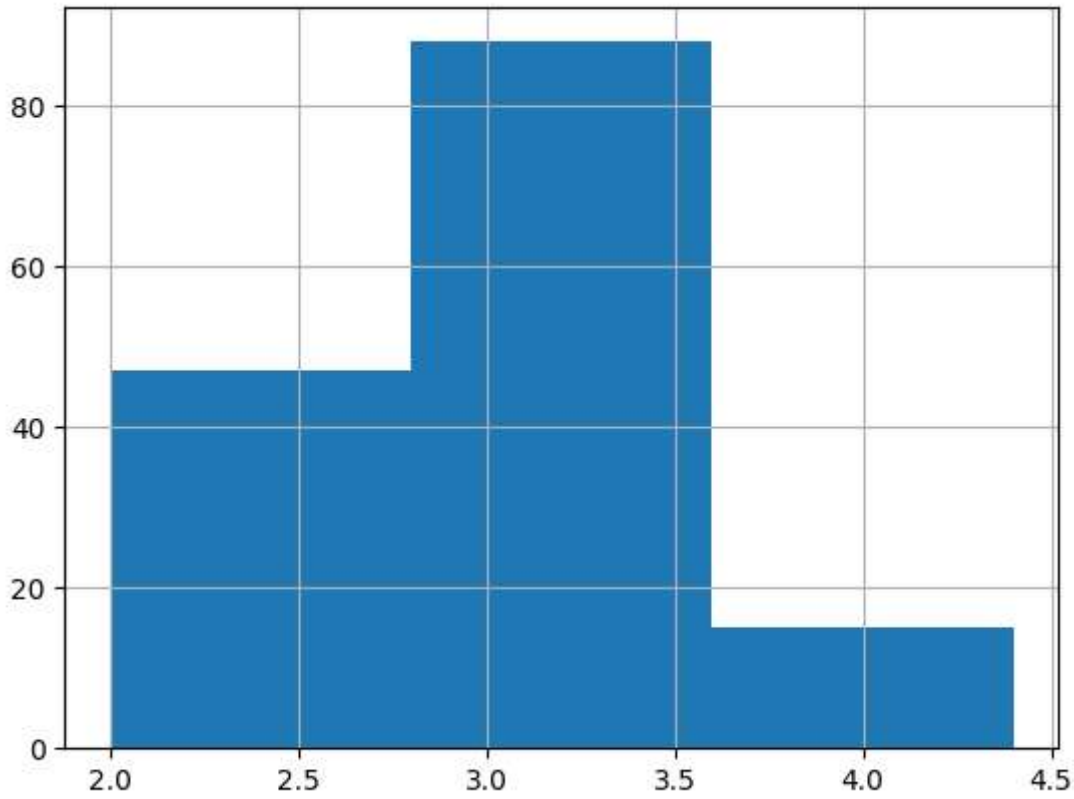


Sepal Width

Histogram

In [ ]:
```
df['SepalWidthCm'].hist(bins=3)
```

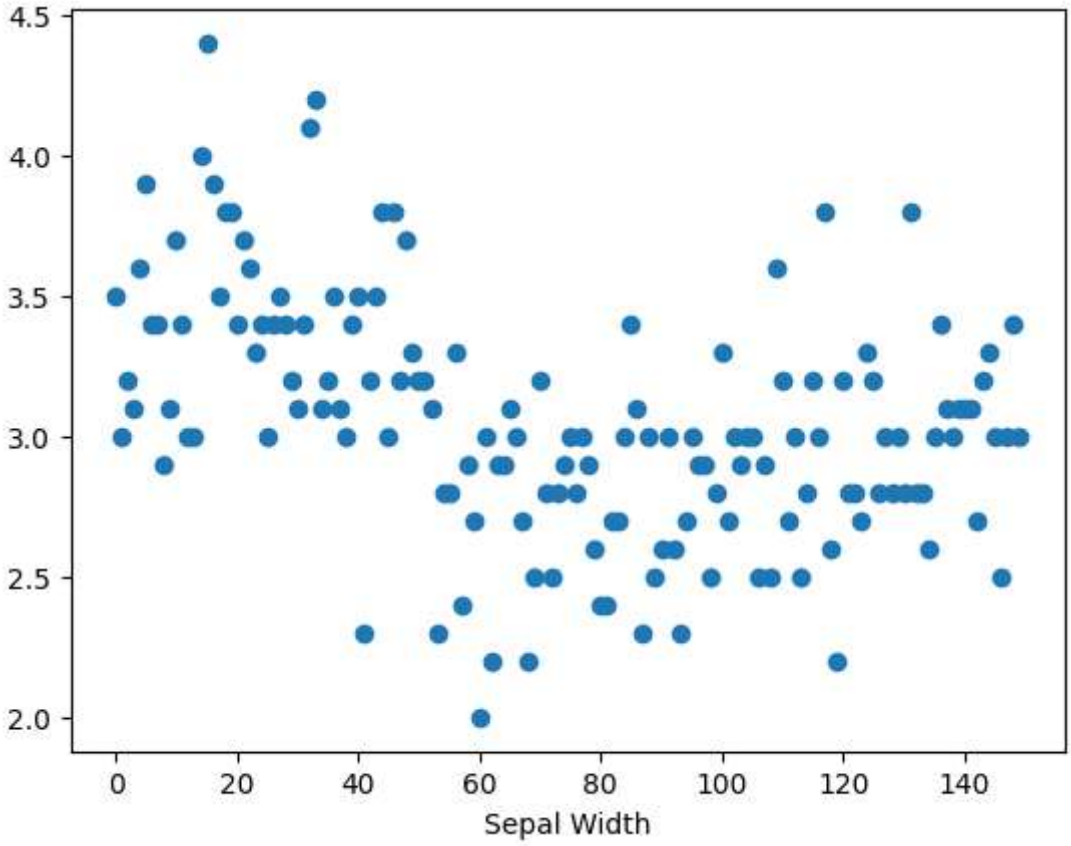
Out[ ]:
```
<Axes: >
```



Scatter Plot

In [ ]:
```
plt.plot(df['SepalWidthCm'],'o')
plt.xlabel('Sample ID')
plt.ylabel('Sepal Width')
```

Out[ ]:
```
Text(0.5, 0, 'Sepal Width')
```

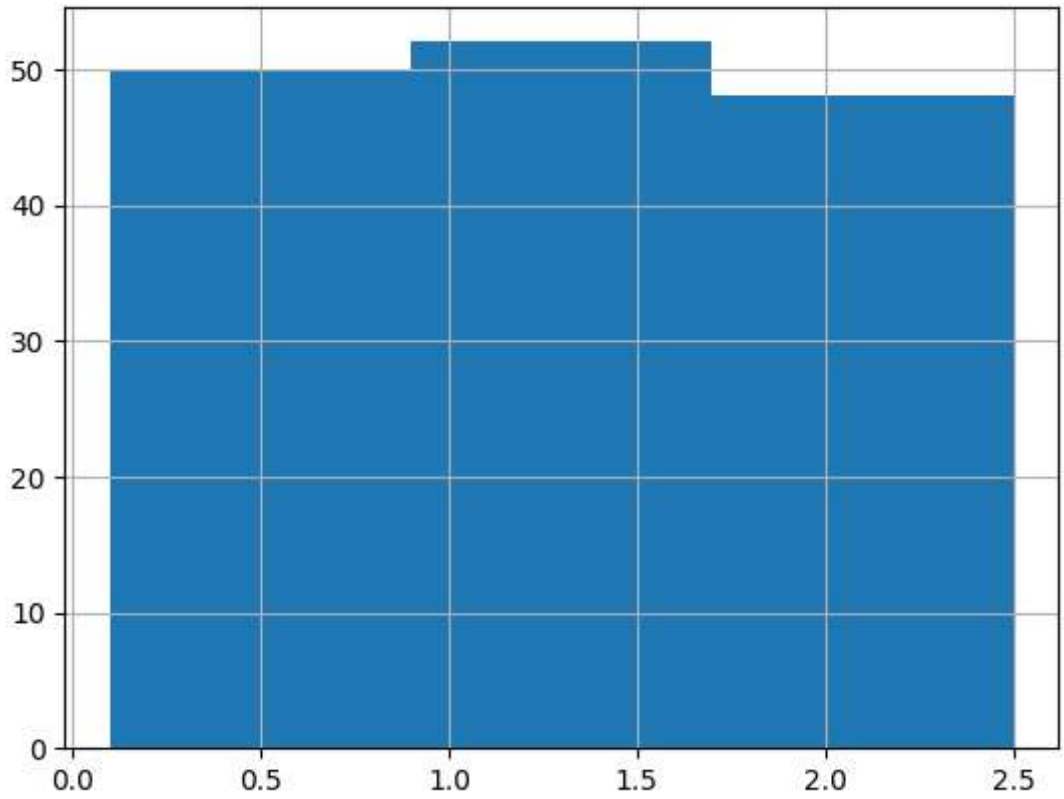


Petal Width

Histogram

In [ ]:
```
df['PetalWidthCm'].hist(bins=3)
```

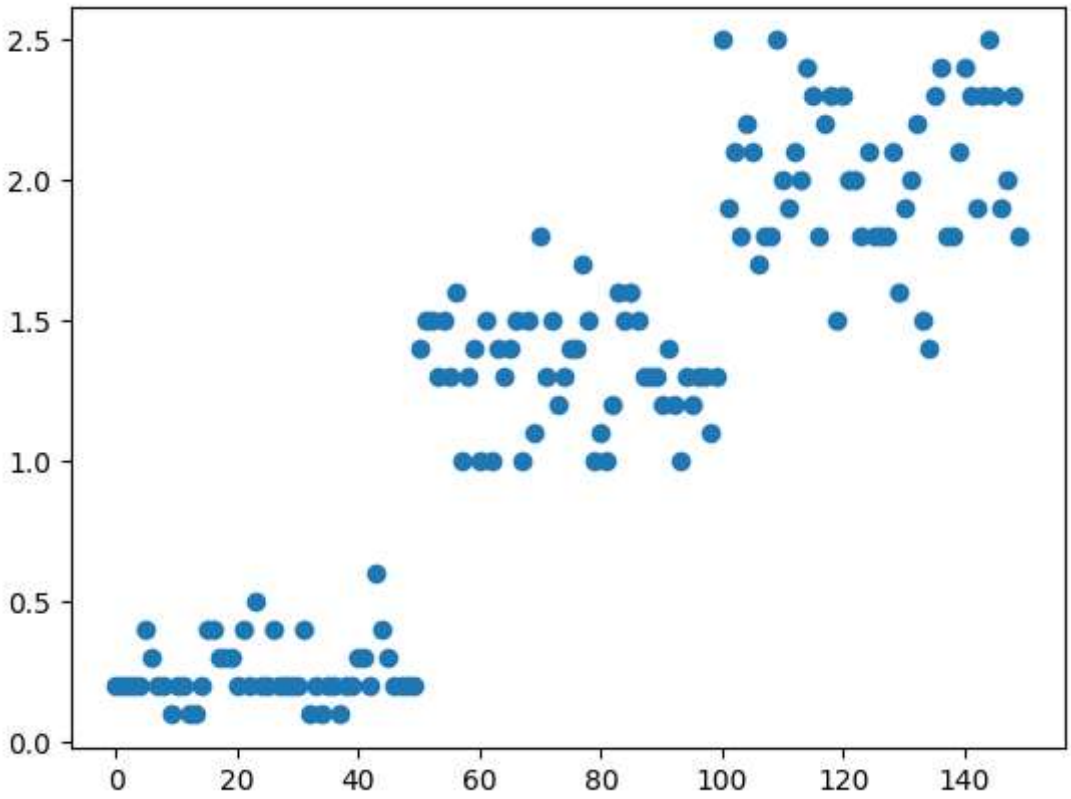
Out[ ]:
```
<Axes: >
```



Scatter Plot

In [ ]:
```
plt.plot(df['PetalWidthCm'],'o')
plt.xlabel('Sample ID')
plt.ylabel('Petal Width')
```

Out[ ]:
```
[<matplotlib.lines.Line2D at 0x7e8c471e9f90>]
```

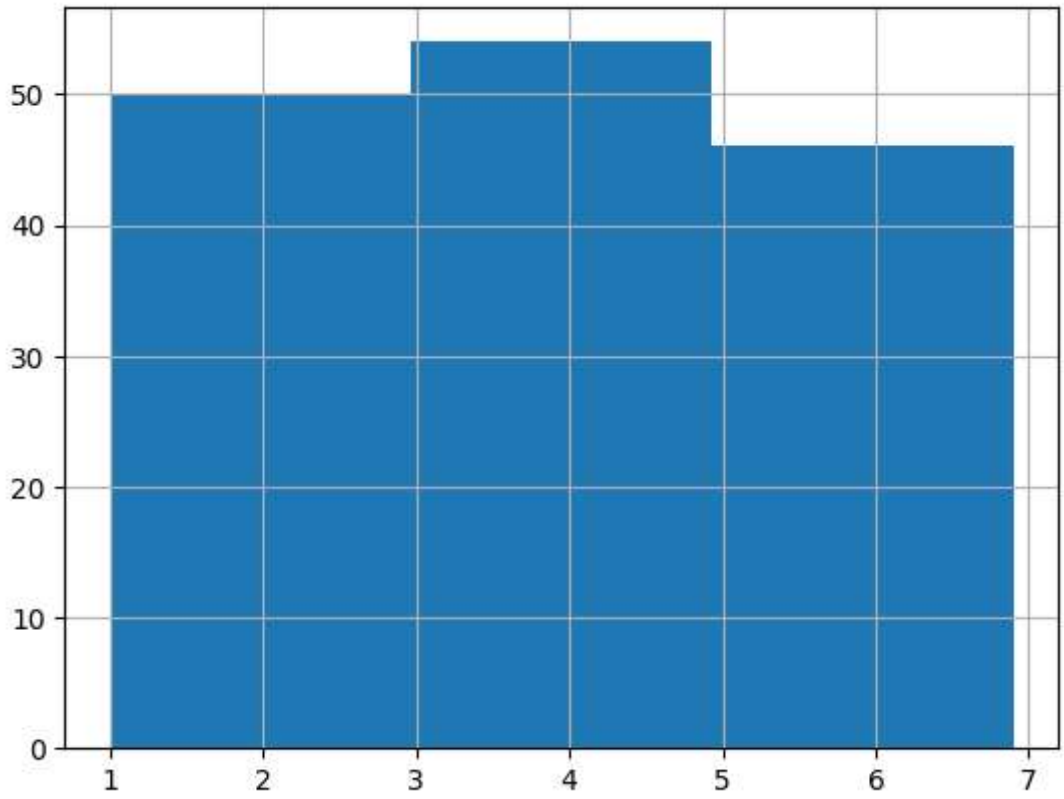


Petal Length

Histogram

In [ ]:
```
df['PetalLengthCm'].hist(bins=3)
```

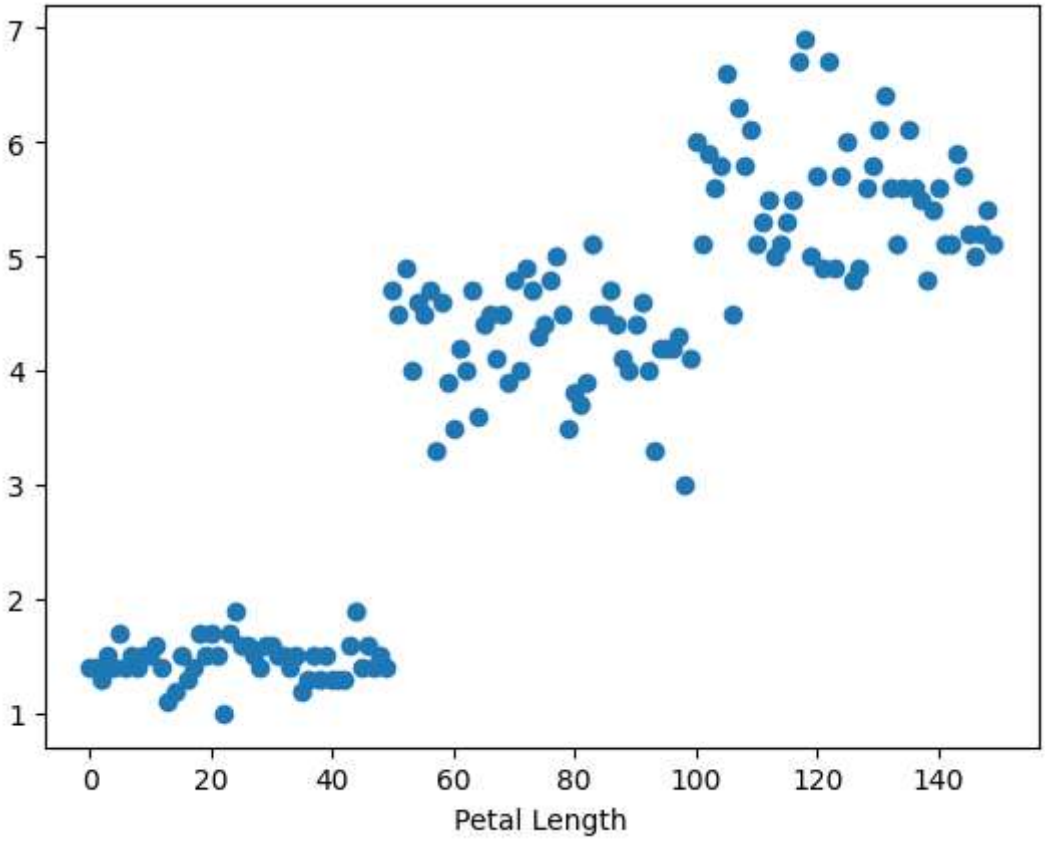
Out[ ]:
```
<Axes: >
```



Scatter Plot

In [ ]:
```
plt.plot(df['PetalLengthCm'],'o')
plt.xlabel('Sample ID')
plt.ylabel('Petal Length')
```

Out[ ]:
```
Text(0.5, 0, 'Petal Length')
```



ML Algorithm

KNN

In [ ]:
```
# Import necessary modules
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris

# Loading data
irisData = load_iris()

# Create feature and target arrays
X = irisData.data
y = irisData.target

# Split into training and test set
X_train, X_test, y_train, y_test = train_test_split(
                        X, y, test_size = 0.2, random_state=42)

knn = KNeighborsClassifier(n_neighbors=7)

knn.fit(X_train, y_train)

# Predict on dataset which model has not seen before
print(knn.predict(X_test))

# Calculate the accuracy of the model
print(knn.score(X_test, y_test))
```

```
[1 0 2 1 1 0 1 2 2 1 2 0 0 0 0 1 2 1 1 2 0 2 0 2 2 2 2 2 0 0]
0.9666666666666667
```