Prediction using Unsupervised learning

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from sklearn import datasets
```

Loading Data set

In [3]:
```
# Load the iris dataset
df = pd.read_csv('Iris.csv')
df.head()
```

Out [3]:

	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

In [4]:
```
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 [5]:
```
df.isnull().sum()
```

Out [5]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [6]:
```
df = df.drop(columns= ['Id'])
df.head()
```

Out [6]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [7]:
```
df.describe()
```

Out [7]:

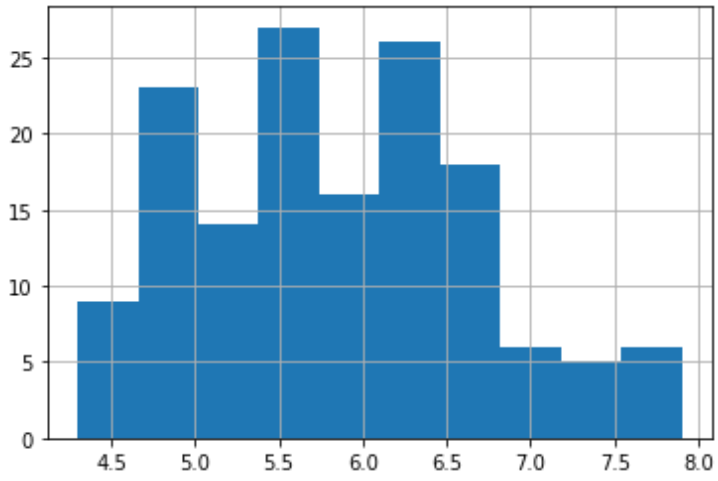
	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [8]:
```
df['Species'].value_counts()
```

Out [8]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

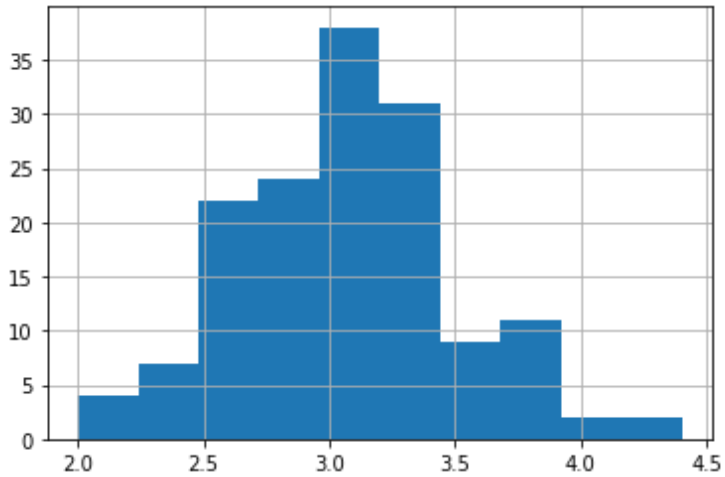
In [9]:
```
df['SepalLengthCm'].hist()
```

Out [9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdb1df72160>
```



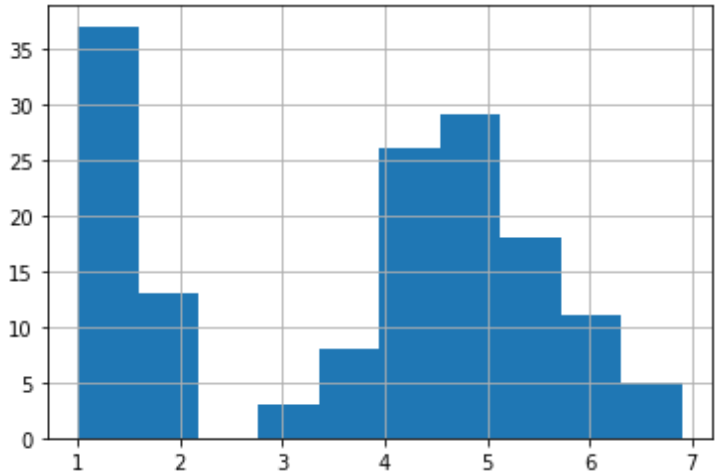
In [39]:
```
df['SepalWidthCm'].hist()
```

Out [39]: <AxesSubplot:>



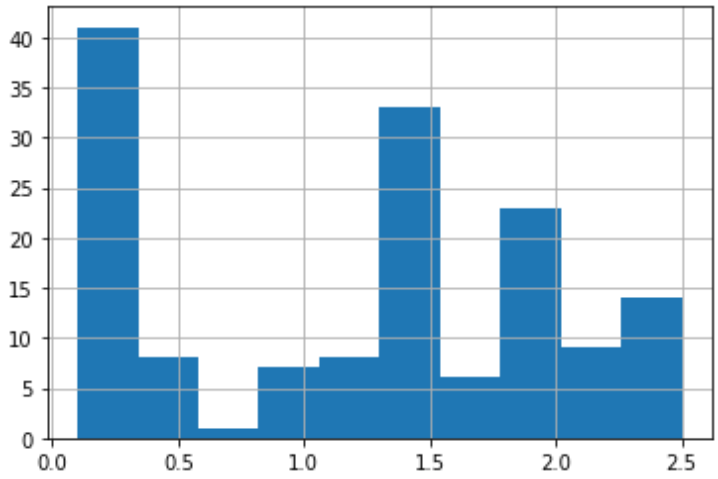
In [40]:
```
df['PetalLengthCm'].hist()
```

Out [40]: <AxesSubplot:>



In [41]:
```
df['PetalWidthCm'].hist()
```

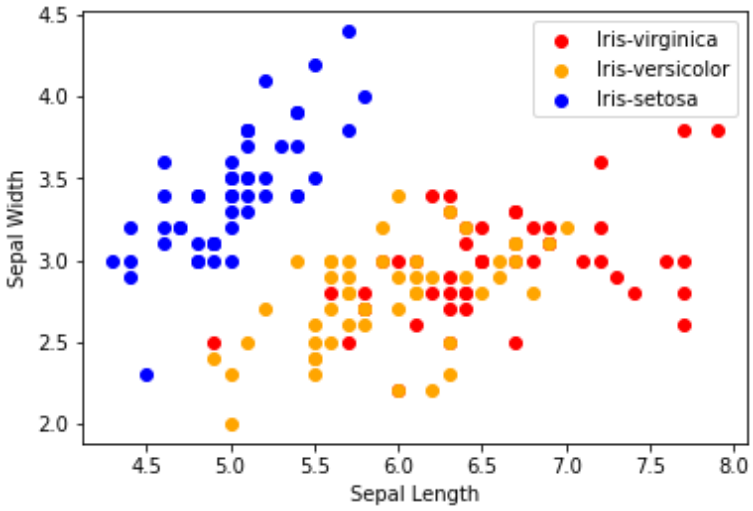
Out [41]: <AxesSubplot:>



In [42]:
```
# Scatterplot
colors = ['red', 'orange', 'blue']
species = ['Iris-virginica','Iris-versicolor','Iris-setosa']
```

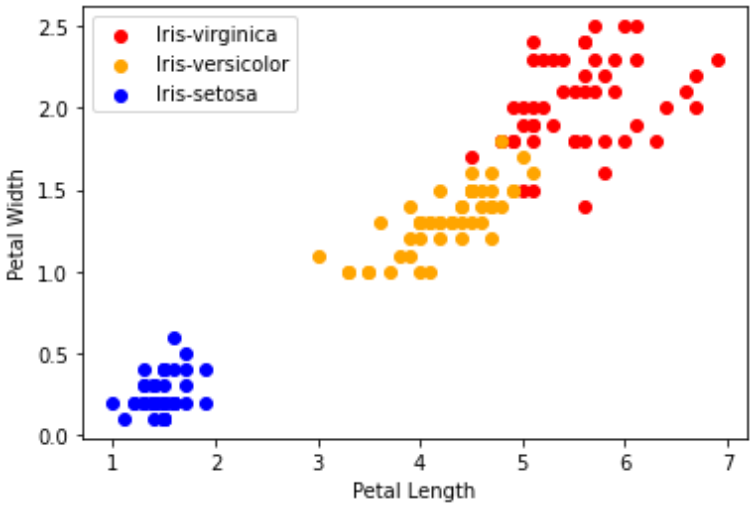
In [43]:
```
for i in range(3):
    x = df[df['Species'] == species[i]]
    plt.scatter(x['SepalLengthCm'], x['SepalWidthCm'], c = co
plt.xlabel("Sepal Length")
plt.ylabel("Sepal Width")
plt.legend()
```

Out [43]: <matplotlib.legend.Legend at 0x7f0d1374b820>



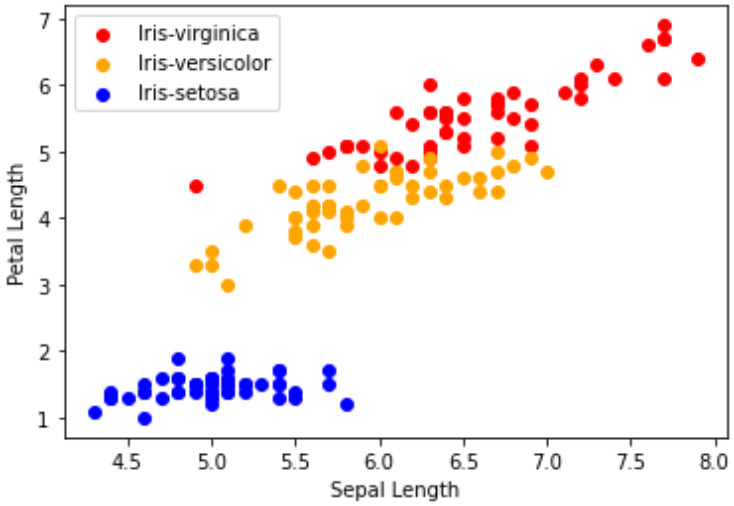
In [44]:
```
for i in range(3):
    x = df[df['Species'] == species[i]]
    plt.scatter(x['PetalLengthCm'], x['PetalWidthCm'], c = co
plt.xlabel("Petal Length")
plt.ylabel("Petal Width")
plt.legend()
```

Out [44]: <matplotlib.legend.Legend at 0x7f0d1366f070>



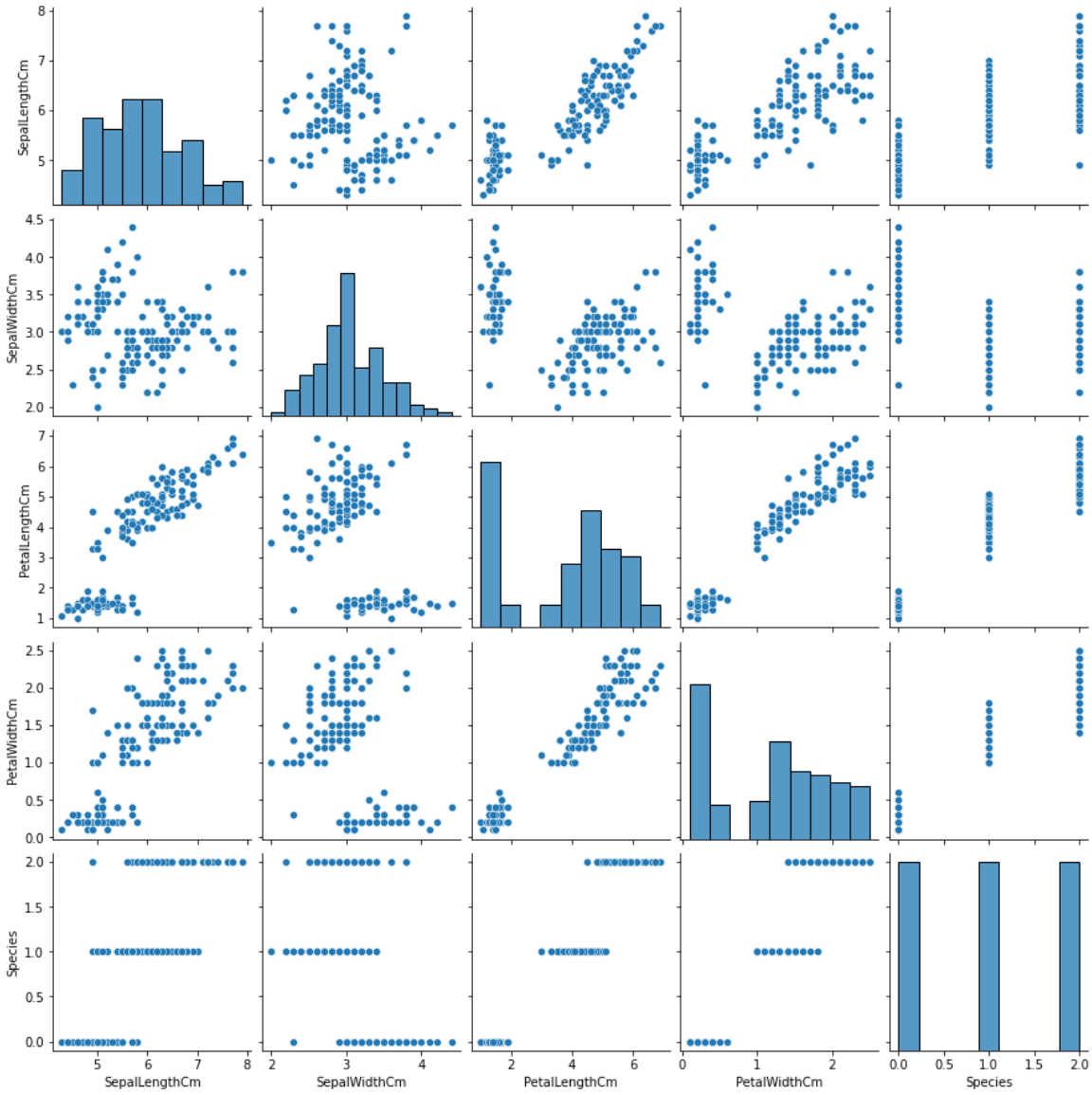
In [45]:
```
for i in range(3):
    x = df[df['Species'] == species[i]]
    plt.scatter(x['SepalLengthCm'], x['PetalLengthCm'], c = c
plt.xlabel("Sepal Length")
plt.ylabel("Petal Length")
plt.legend()
```

Out [45]: <matplotlib.legend.Legend at 0x7f0d135e1130>



In [62]:
```
sns.pairplot(df)
```

Out [62]: <seaborn.axisgrid.PairGrid at 0x7f0d1394abe0>



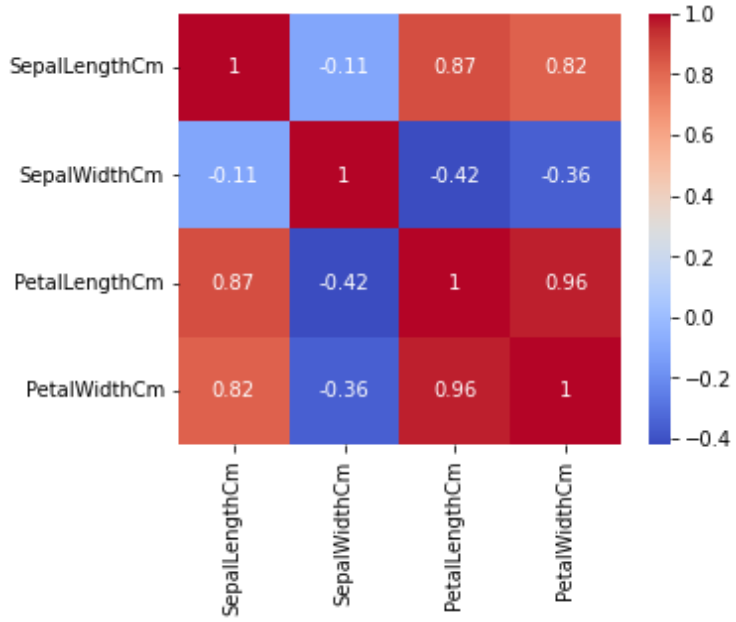
In [46]:
```
df.corr()
```

Out [46]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
SepalLengthCm	1.000000	-0.109369	0.871754	0.817954
SepalWidthCm	-0.109369	1.000000	-0.420516	-0.356544
PetalLengthCm	0.871754	-0.420516	1.000000	0.962757
PetalWidthCm	0.817954	-0.356544	0.962757	1.000000

In [47]:
```
corr = df.corr()
fig, ax = plt.subplots(figsize=(5,4))
sns.heatmap(corr, annot=True, ax=ax, cmap = 'coolwarm')
```

Out [47]: <AxesSubplot:>



In [48]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

In [49]:
```
df['Species'] = le.fit_transform(df['Species'])
df.head()
```

Out [49]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
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

In [50]:
```
from sklearn.model_selection import train_test_split
# train - 70
# test - 30
X = df.drop(columns=['Species'])
Y = df['Species']
x_train, x_test, y_train, y_test = train_test_split(X, Y, tes
```

In [54]:
```
# knn - k-nearest neighbours
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier()
```

In [55]:
```
model.fit(x_train, y_train)
```

Out [55]:
```
KNeighborsClassifier()
```

In [56]:

```
# print metric to get performance
print("Accuracy: ",model.score(x_test, y_test) * 100)
```

```
Accuracy:  95.55555555555556
```