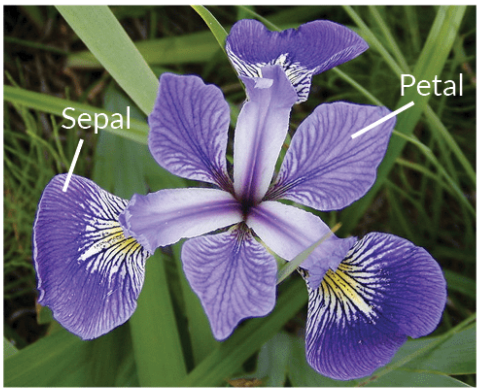
In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [3]:
```
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

The Iris Dataset contains four features (length and width of sepals and petals) of 50 samples of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). These measures were used to create a linear discriminant model to classify the species. The dataset is often used in data mining, classification and clustering examples and to test algorithms.



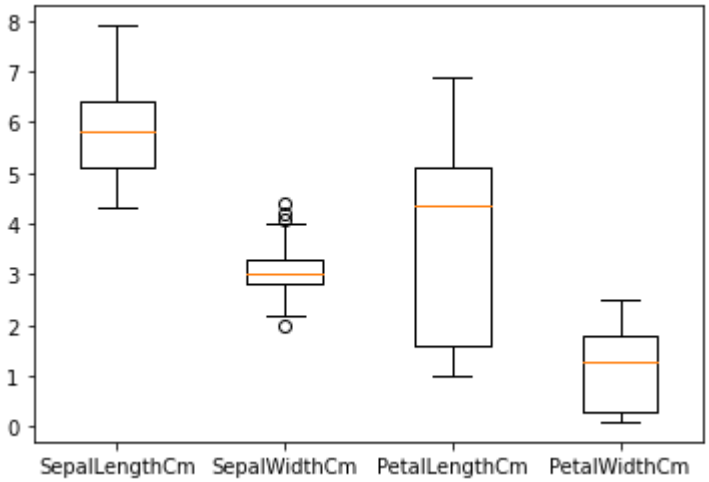
Iris Versicolor

Iris Setosa

Iris Virginica

Analysis of fetures for classification

In [4]:
```
plt.boxplot([df.iloc[:,1],df.iloc[:,2],df.iloc[:,3],df.iloc[:,4]],labels=df.columns[1:-
plt.show()
```



In [ ]:
```
plt.title('Sepal Length distribution')
sl_setosa = df.loc[df['Species']=='Iris-setosa',df.columns[1]] #2,3,4
sl_versi = df.loc[df['Species'] =='Iris-versicolor',df.columns[1]] #2,3,4
sl_virgi = df.loc[df['Species'] =='Iris-virginica',df.columns[1]] #2,3,4
plt.boxplot([sl_setosa,sl_versi,sl_virgi],labels=df['Species'].unique())
plt.show()
```

In [ ]:
```
import seaborn as sns
sns.pairplot(df.iloc[:,1:],hue='Species')
```

sklearn

In [ ]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
```

Prepare train, test datasets

In [ ]:
```
X = df.iloc[:,[1,4]]
y = df.iloc[:,5]
txf = dict(zip(df['Species'].unique(),[0,1,2]))
y = y.map(txf)
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2)
```

Train, Test, Evaluate

In [ ]:
```
clf = LogisticRegression()
clf.fit(X_train,y_train)
```

In [ ]:
```
y_pred = clf.predict(X_test)
y_pred
```

In [ ]:
```
misclf = np.count_nonzero(y_pred - y_test)
accuracy = 1 - misclf/len(y_test)
print(accuracy)
```

In [ ]:
```
confusion_matrix(y_test,y_pred)
```

Decision boundary

In [ ]:
```
px = []
py = []
z = []
for sl in np.arange( df.iloc[:,1].min()-0.5,df.iloc[:,1].max()+0.5,0.01):
    for pw in np.arange( df.iloc[:,4].min()-0.5,df.iloc[:,4].max()+0.5,0.01):
        px.append(sl)
        py.append(pw)
```

In [ ]:
```
XX = np.array([x for x in zip(px,py)])
yy = clf.predict(XX)
```

In [ ]:
```
plt.scatter(XX[:,0],XX[:,1],c=yy)
plt.scatter(X.iloc[:,0],X.iloc[:,1],c=y,cmap='hot')
```

In [ ]:
```

```