In [ ]:

```
 VIDYA PAYGUDE
---------------------------------------------

   Data Science Intern at LetsGrowMore Virtual Internship Program (APRIL-2022)

   Beginner Level Task 1 - Iris Flowers Classification ML Project

   Algorithm Used - K nearest neighbor classifier
```

In [23]:

```
import pandas as pd #for analysis and manipulation of numerical tables
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt #for plotting graphs
from pandas.plotting import scatter_matrix
```

In [32]:

```
col_names = ['sepal-length','sepal-width','petal-length','petal-width','class']
iris = pd.read_csv(r'C:\Users\Admin\Documents\csv\iris.csv', names=col_names)
```

In [33]:

```
iris
```

Out[33]:

	sepal-length	sepal-width	petal-length	petal-width	class
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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [34]:

```
iris.head()
```

Out[34]:

	sepal-length	sepal-width	petal-length	petal-width	class
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 [35]:

```
iris.tail()
```

Out[35]:

	sepal-length	sepal-width	petal-length	petal-width	class
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

In [36]:

```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal-length  150 non-null    float64
 1   sepal-width   150 non-null    float64
 2   petal-length  150 non-null    float64
 3   petal-width   150 non-null    float64
 4   class         150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [37]:

```
iris.describe()
```

Out[37]:

	sepal-length	sepal-width	petal-length	petal-width
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 [38]:

```
print(iris.shape)
```

```
(150, 5)
```

In [39]:

```
print(iris.isna().sum())
```

```
sepal-length    0
sepal-width     0
petal-length    0
petal-width     0
class           0
dtype: int64
```

In [40]:

```
versicolor = len(iris[iris['class'] == 'Iris-versicolor'])
print("No. of Iris Versicolor in Dataset:",versicolor)
```

```
No. of Iris Versicolor in Dataset: 50
```

In [41]:

```
setosa = len(iris[iris['class'] == 'Iris-setosa'])
print("No. of Iris Setosa in Dataset:",setosa)
```

```
No. of Iris Setosa in Dataset: 50
```

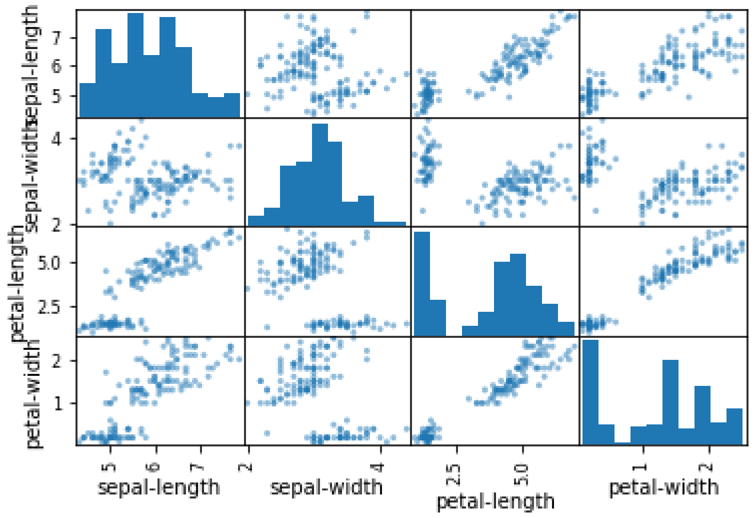
In [42]:

```
virginica = len(iris[iris['class'] == 'Iris-virginica'])
print("No. of Iris Virginica in Dataset:",virginica)
```

```
No. of Iris Virginica in Dataset: 50
```

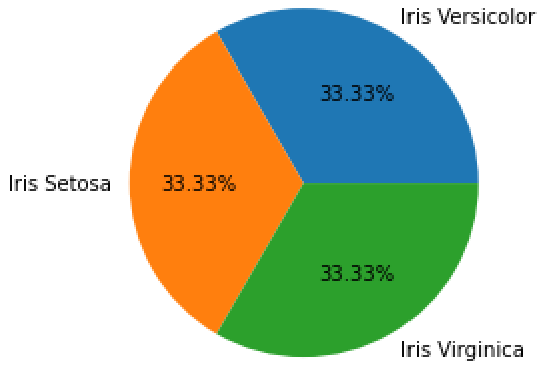
In [43]:

```
scatter_matrix(iris)
plt.show()
```



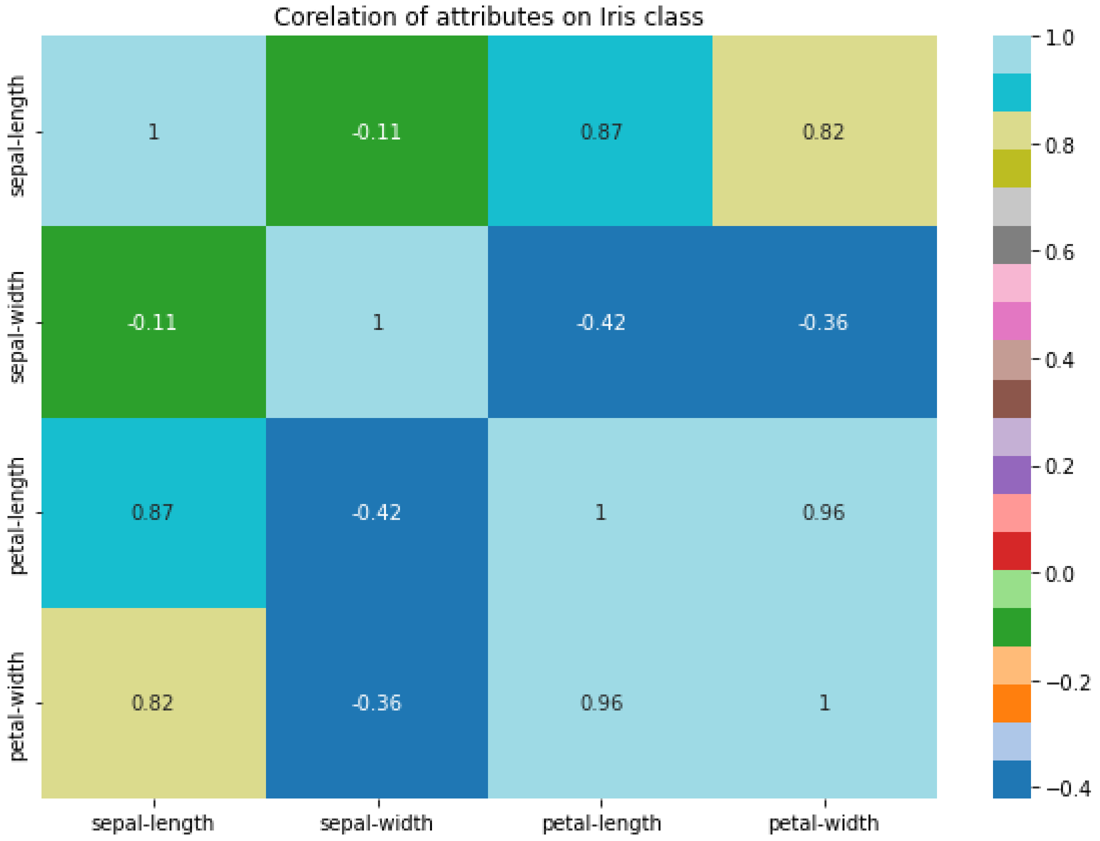
In [44]:

```
fig = plt.figure()
labels = ['Iris Versicolor', 'Iris Setosa', 'Iris Virginica']
d = [50,50,50]
plt.pie(d, labels = labels,autopct='%1.2f%%')
plt.show()
```



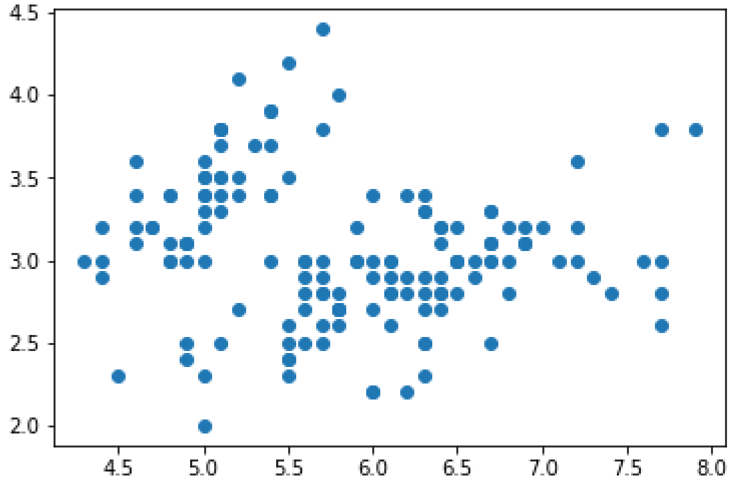
In [45]:

```
plt.subplots(figsize = (10,7))
sns.heatmap(iris.corr(),annot=True,cmap="tab20").set_title("Corelation of attributes on Iri
plt.show()
```



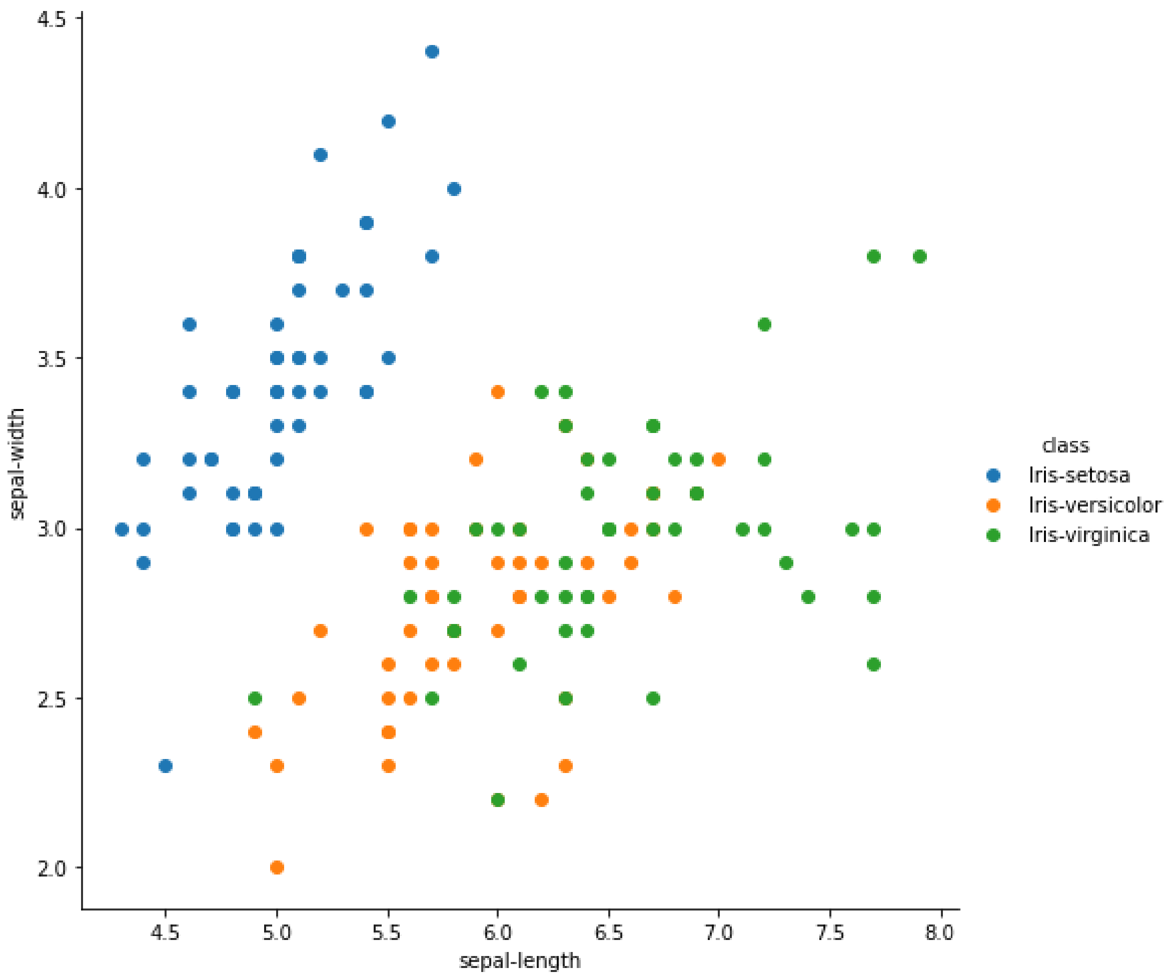
In [46]:

```
fig=plt.scatter(iris['sepal-length'],iris['sepal-width'])
```



In [47]:

```
graph=sns.FacetGrid(iris, hue ="class",height=7)
graph.map(plt.scatter,"sepal-length","sepal-width").add_legend()
plt.show()
```

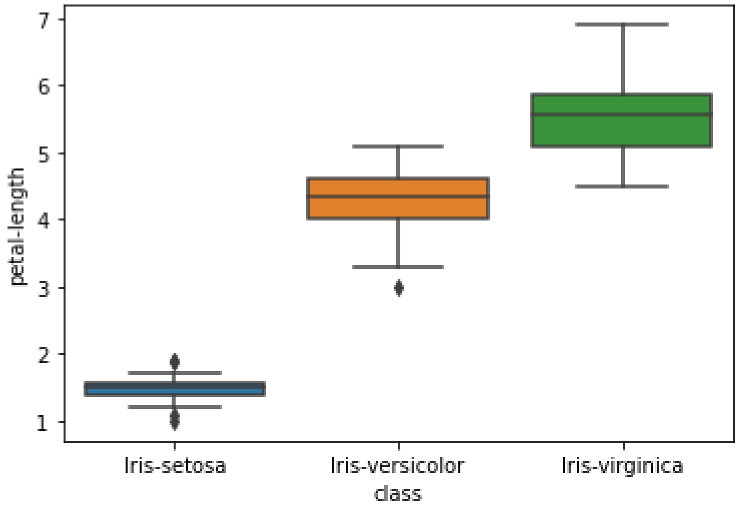


In [48]:

```
sns.boxplot(x="class",y="petal-length",data=iris)
```

Out[48]:

```
<AxesSubplot:xlabel='class', ylabel='petal-length'>
```

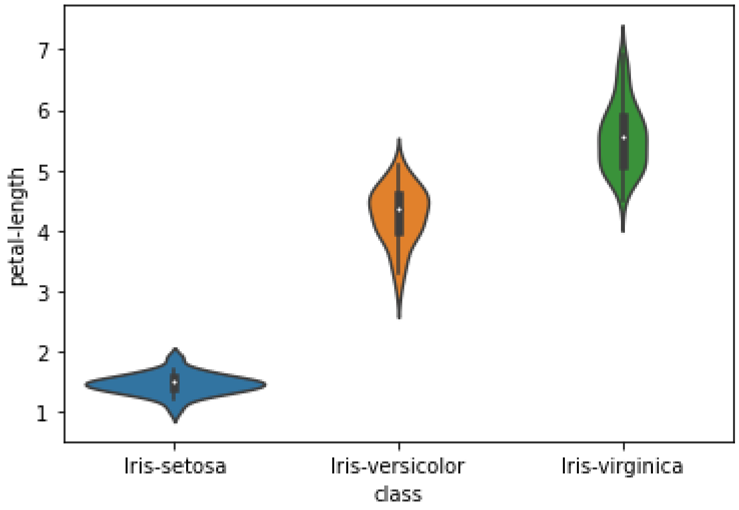


In [49]:

```
sns.violinplot(x="class",y="petal-length",data=iris,size=10)
```

Out[49]:

```
<AxesSubplot:xlabel='class', ylabel='petal-length'>
```

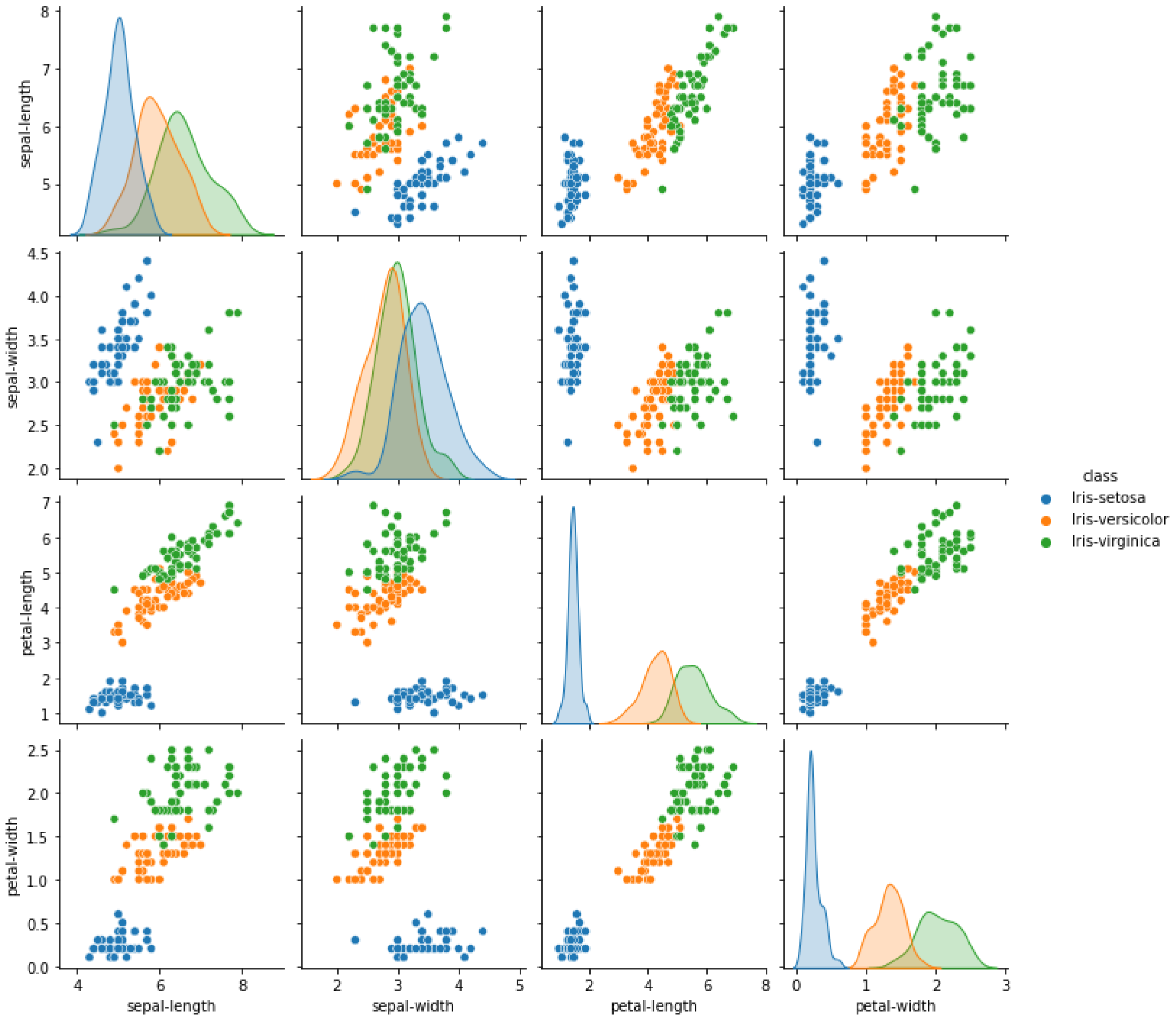


In [50]:

```
sns.pairplot(iris,hue='class')
```

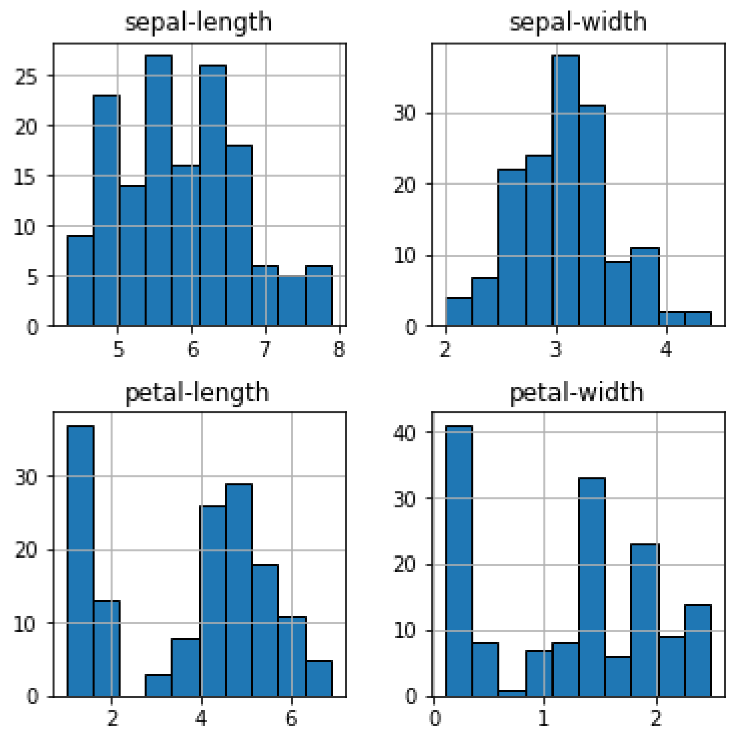
Out[50]:

```
<seaborn.axisgrid.PairGrid at 0x1f2c9945a90>
```



In [51]:

```
iris.hist(figsize=(6,6),edgecolor='black')
plt.show()
```



In [52]:

```
from  sklearn.datasets import  load_iris
iris =load_iris()
x=iris.data
y=iris.target
```

In [53]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.5)
```

In [54]:

```
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
```

In [55]:

```
from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression()
log_reg.fit(x_train, y_train)
predictions = log_reg.predict(x_test)
print ("Logistic Regression")
print ("Accuracy Score:", accuracy_score(y_test, predictions))
print (confusion_matrix(y_test, predictions))
print (classification_report(y_test, predictions))
```

```
Logistic Regression
Accuracy Score: 0.9466666666666667
[[22  0  0]
 [ 0 24  0]
 [ 0  4 25]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        22
           1       0.86      1.00      0.92        24
           2       1.00      0.86      0.93        29

    accuracy                           0.95        75
   macro avg       0.95      0.95      0.95        75
weighted avg       0.95      0.95      0.95        75
```

In [56]:

```
from sklearn.svm import SVC
svm = SVC()
svm.fit(x_train, y_train)
predictions = svm.predict(x_test)
print ("Support Vector Machines")
print ("Accuracy Score:", accuracy_score(y_test, predictions))
print (confusion_matrix(y_test, predictions))
print (classification_report(y_test, predictions))
```

```
Support Vector Machines
Accuracy Score: 0.92
[[22  0  0]
 [ 0 24  0]
 [ 0  6 23]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        22
           1       0.80      1.00      0.89        24
           2       1.00      0.79      0.88        29

    accuracy                           0.92        75
   macro avg       0.93      0.93      0.92        75
weighted avg       0.94      0.92      0.92        75
```

In [ ]: