In [11]:
```
import pandas as pd
import numpy as np
import matplotlib as mt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,confusion_matrix
```

In [12]:
```
df=pd.read_csv('Iris.csv')
```

In [13]:
```
df.shape
```

Out[13]:
```
(150, 6)
```

In [14]:
```
df.info()
# count This shows the number of non-null values in each numerical column
#mean
#std ,
#25%= Q1  value below which 25% of the data falls.
# 50% median
#75%= Q3   value below which 75% of the data falls.
#max and min val in that column
```

```
<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
```

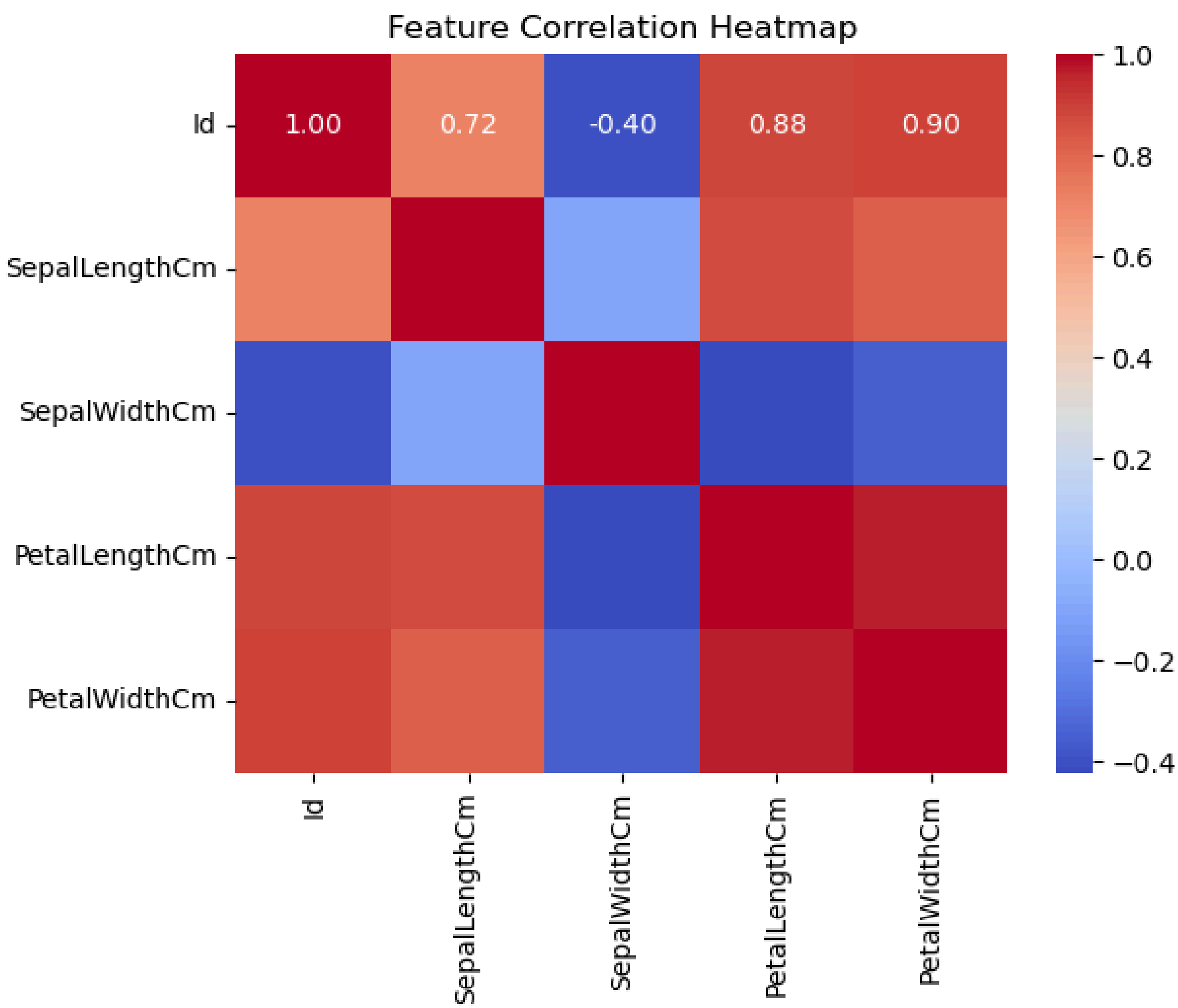
In [15]:
```
df.head()
#df.corr()
```

Out[15]:

	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 [ ]:

In [16]:
```
# Exclude non-numeric columns from the DataFrame
numeric_df = df.select_dtypes(include=['number'])

# Calculate the correlation matrix for numeric columns only
correlation_matrix = numeric_df.corr()

# Visualize the correlation matrix
import seaborn as sns
import matplotlib.pyplot as plt

sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Feature Correlation Heatmap")
plt.show()
```



In [17]:
```
X=df.drop(['Id','Species'],axis=1)
y=df['Species']
print(X)
print(y)
print(X.shape)
print(y.shape)
```

```
     SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
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 x 4 columns]
0         Iris-setosa
1         Iris-setosa
2         Iris-setosa
3         Iris-setosa
4         Iris-setosa
            ...
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
Name: Species, Length: 150, dtype: object
(150, 4)
(150,)
```

In [18]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test =train_test_split(X,y,test_size=0.2,random_
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(120, 4)
(30, 4)
(120,)
(30,)
```

In [19]:

```
from sklearn.naive_bayes import GaussianNB
model=GaussianNB()
model.fit(X_train,y_train)
```

Out[19]:

```
▾ GaussianNB
GaussianNB()
```

In [20]:

```
print(y)
```

```
0          Iris-setosa
1          Iris-setosa
2          Iris-setosa
3          Iris-setosa
4          Iris-setosa
              ...
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
Name: Species, Length: 150, dtype: object
```

In [21]:
```
df.drop('Id',axis=1)
```

Out[21]:

	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
...	...	...	...	...	...
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 [22]:
```
print(y)
```

```
0          Iris-setosa
1          Iris-setosa
2          Iris-setosa
3          Iris-setosa
4          Iris-setosa
              ...
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
Name: Species, Length: 150, dtype: object
```

In [23]:
```
y_pred=model.predict(X_test)
model.score(X_test,y_test)
```

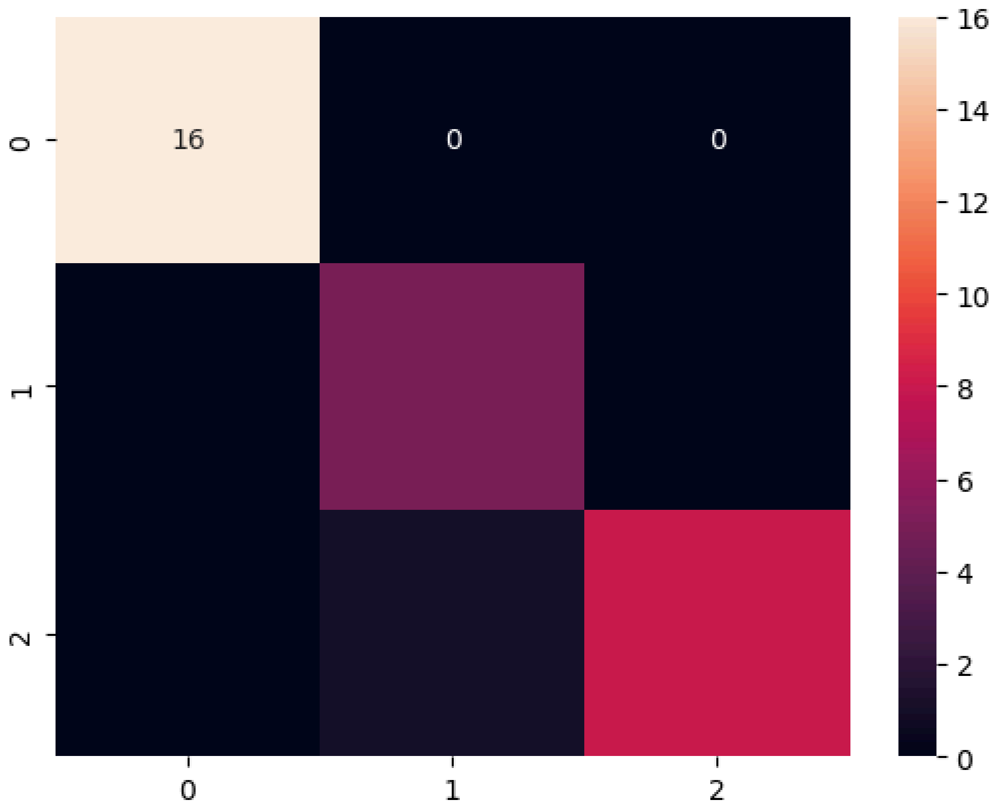
Out[23]: 0.9666666666666667

In [24]:
```
cm=confusion_matrix(y_test,y_pred)
print(cm)
```

```
[[16  0  0]
 [ 0  5  0]
 [ 0  1  8]]
```

In [25]:
```
sns.heatmap(cm,annot=True)
```

Out[25]: <Axes: >



In [26]:
```
TN=cm[0][0]
FP=cm[0][1]
FN=cm[0][1]
TP=cm[1][1]
accuracy=TP+FP/TP+FP+TN+FN
error_rate=1 - accuracy
precision=TP/(TP+FP)
recall=TP/ (TP+FN)
print("Accuracy:",accuracy)
print("error rate:",error_rate)
print("precision:",precision)
print("Recall:",recall)

print('TN:',TN)
print('TP:',TP)
print('FN:',FN)
print('FP:',FP)
```

```
Accuracy: 21.0
error rate: -20.0
precision: 1.0
Recall: 1.0
TN: 16
TP: 5
FN: 0
FP: 0
```

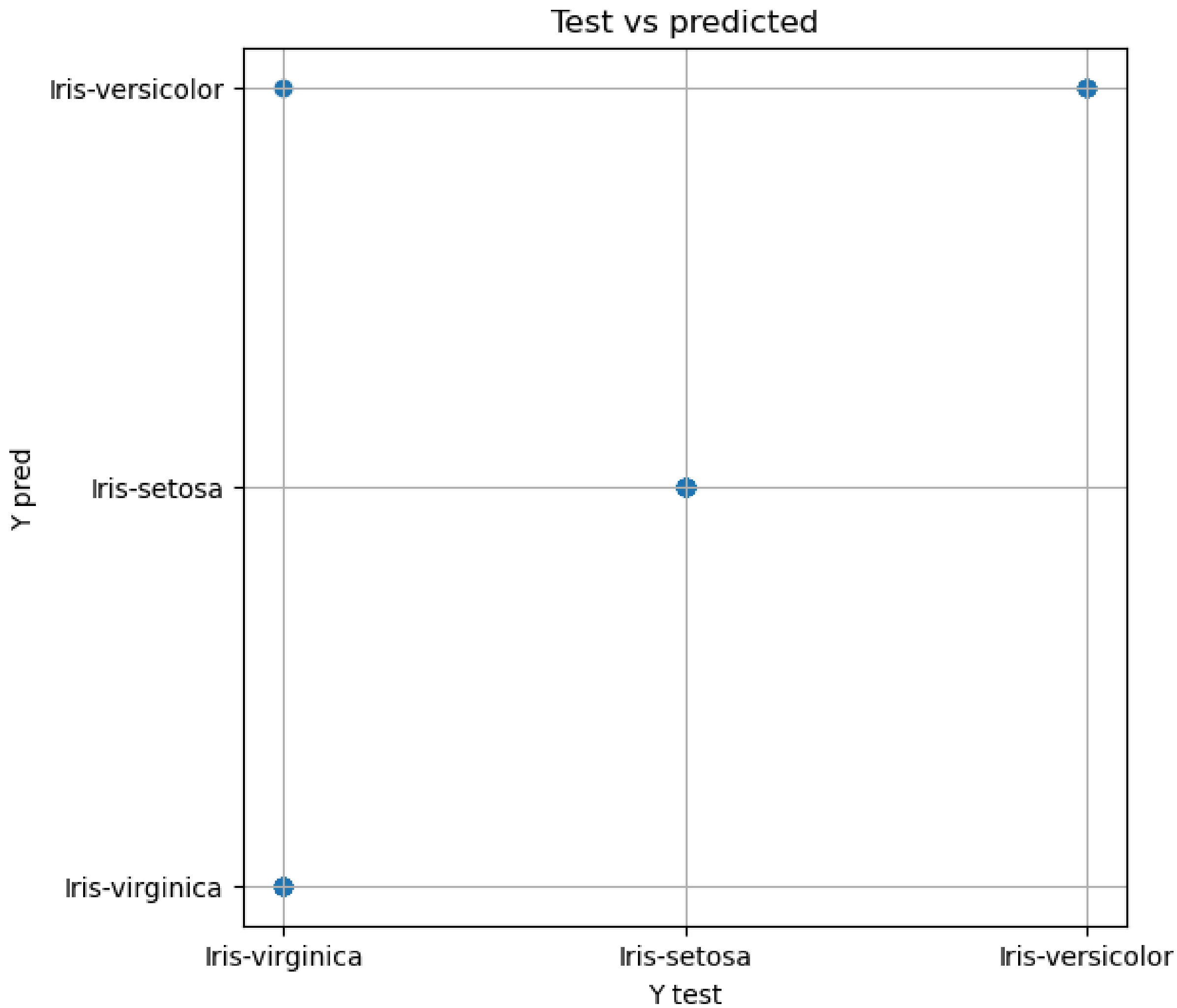
In [27]:
```
from sklearn.metrics import classification_report
report=classification_report(y_test,y_pred)
print(report)
```

```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        16
Iris-versicolor       0.83      1.00      0.91         5
 Iris-virginica       1.00      0.89      0.94         9

       accuracy                           0.97        30
      macro avg       0.94      0.96      0.95        30
   weighted avg       0.97      0.97      0.97        30
```

In [30]:
```
f1_score=(2*precision*recall)/precision+recall
print('F1 score:',f1_score)
```

```
F1 score: 3.0
```

In [33]:
```
plt.figure(figsize=(6,6))
plt.scatter(y_test,y_pred)
plt.xlabel("Y test")
plt.ylabel("Y pred")
plt.title('Test vs predicted')
plt.grid(True)
plt.show()
```


Test vs predicted
Iris-versicolor
Iris-setosa
Iris-virginica
Y pred
Iris-virginica
Iris-setosa
Iris-versicolor
Y test

In [ ]: