CODESOFT TASK 2 IRIS FLOWER CLASSIFICATION

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df = pd.read_csv('/content/IRIS.csv')
df
```

	sepal_length	sepal_width	petal_length	petal_width	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

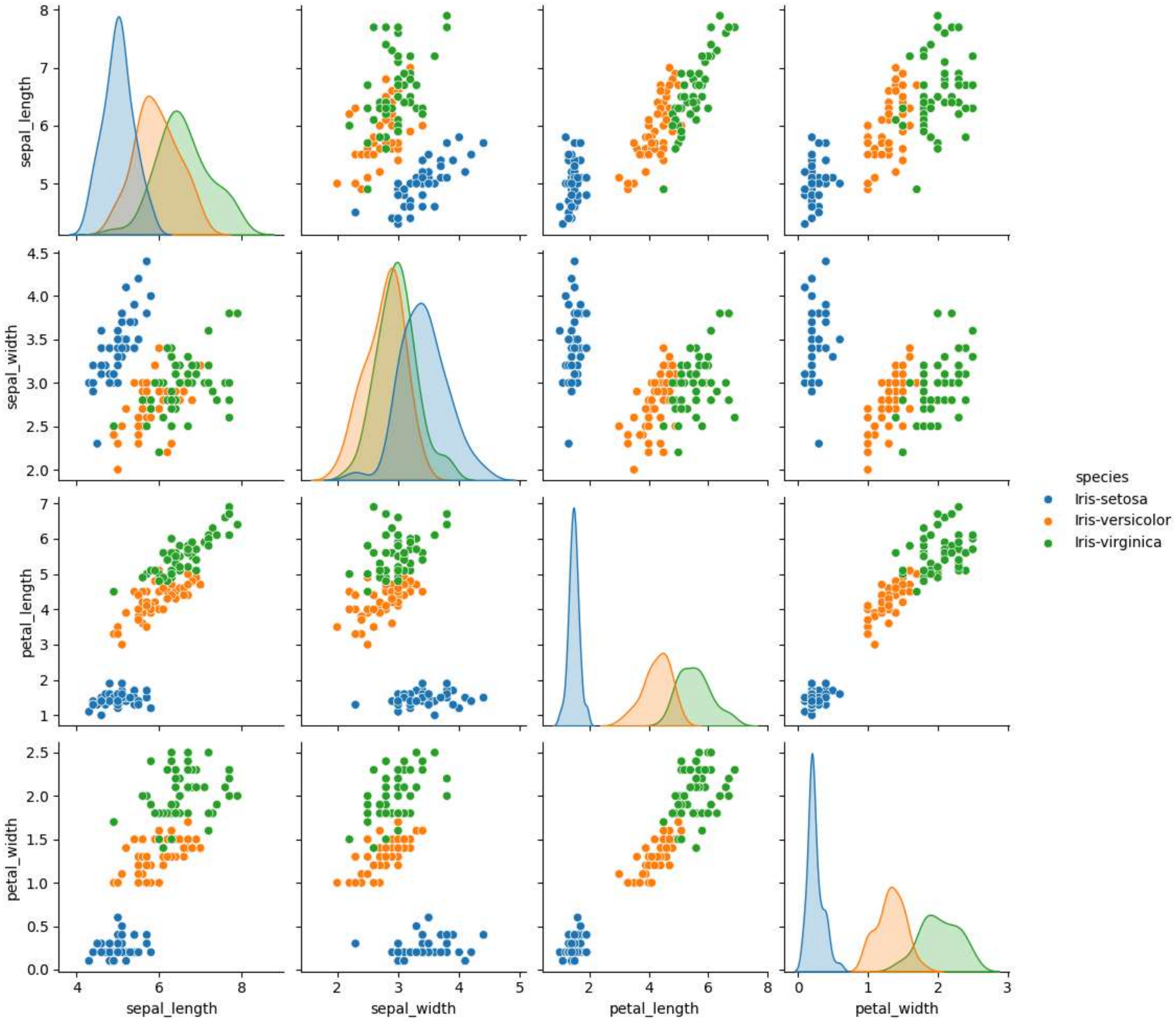
```
df.isnull().sum()
```

	0
sepal_length	0
sepal_width	0
petal_length	0
petal_width	0
species	0

dtype: int64

```
import warnings
warnings.filterwarnings('ignore')
sns.pairplot(df, hue='species')
```

```
<seaborn.axisgrid.PairGrid at 0x7f6b29693190>
```


sepal_length
sepal_width
petal_length
petal_width
8
7
6
5
4.5
4.0
3.5
3.0
2.5
2.0
7
6
5
4
3
2
1
2.5
2.0
1.5
1.0
0.5
0.0
4
6
8
2
3
4
5
2
4
6
8
0
1
2
3
sepal_length
sepal_width
petal_length
petal_width
species
Iris-setosa
Iris-versicolor
Iris-virginica

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['species'] = le.fit_transform(df['species'])
df
```

	sepal_length	sepal_width	petal_length	petal_width	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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2
146	6.3	2.5	5.0	1.9	2
147	6.5	3.0	5.2	2.0	2
148	6.2	3.4	5.4	2.3	2
149	5.9	3.0	5.1	1.8	2

150 rows × 5 columns

```
df.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   species       150 non-null    int64  
dtypes: float64(4), int64(1)
memory usage: 6.0 KB
```

```
df.describe()
```

	sepal_length	sepal_width	petal_length	petal_width	species
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667	1.000000
std	0.828066	0.433594	1.764420	0.763161	0.819232
min	4.300000	2.000000	1.000000	0.100000	0.000000
25%	5.100000	2.800000	1.600000	0.300000	0.000000
50%	5.800000	3.000000	4.350000	1.300000	1.000000
75%	6.400000	3.300000	5.100000	1.800000	2.000000
max	7.900000	4.400000	6.900000	2.500000	2.000000

```
x = df.drop('species', axis=1)
y = df['species']
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_state=42)
```

```
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC
GG = GridSearchCV(SVC(gamma='auto'),
  {
    'C': [i for i in range(1,100)],
    'kernel': ['rbf', 'linear', 'poly']
}, cv=5, return_train_score=False)
GG.fit(x, y)
```

```
▸ GridSearchCV
▸ best_estimator_:
        SVC
    ▸ SVC
```

```
print("Best Hyperparameters for your moedel are ",GG.best_params_)
```

```
Best Hyperparameters for your moedel are  {'C': 4, 'kernel': 'rbf'}
```

```
from sklearn.svm import SVC
model = SVC(C=4, kernel='rbf')
model.fit(x_train, y_train)
```

SVC

```
print('Testing accuracy sore is', model.score(x_test, y_test)*100,'%')
print('Training accuracy sore is', model.score(x_train, y_train)*100,"%")
```

```
Testing accuracy sore is 100.0 %
Training accuracy sore is 99.16666666666667 %
```

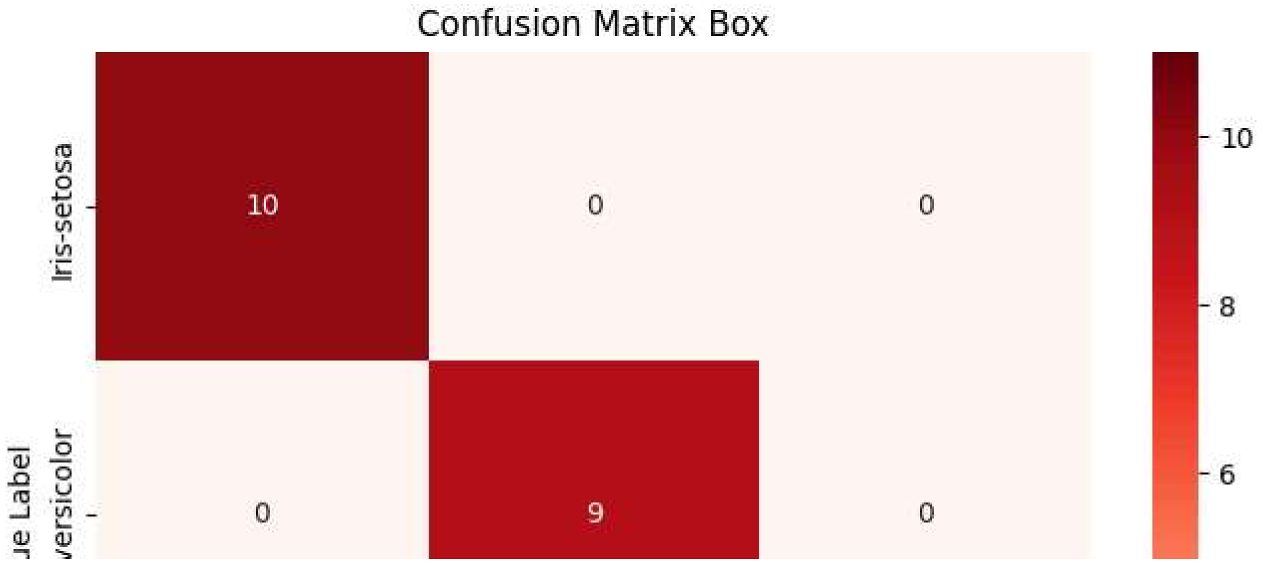
```
from sklearn.metrics import classification_report
y_pred = model.predict(x_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
cm
import matplotlib.pyplot as plt
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Reds", xticklabels=le.classes_, yticklabels=le.cl
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix Box")
plt.show()
```


Confusion Matrix Box
Iris-setosa
ue Label
versicolor