```
# Importing Packages
from sklearn.svm import SVC
import pandas as pd
from sklearn.metrics import
accuracy_score,classification_report,confusion_matrix
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.svm import SVC

#importing csv files
path="/content/drive/MyDrive/Dataset/Iris.csv"
df=pd.read_csv(path)
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 150,\n  \"fields\": [\
n    {\n      \"column\": \"Id\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 43,\n        \"min\": 1,\n
\"max\": 150,\n        \"num_unique_values\": 150,\n
\"samples\": [\n          74,\n          19,\n          119\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"SepalLengthCm\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.8280661279778629,\n        \"min\":
4.3,\n        \"max\": 7.9,\n        \"num_unique_values\": 35,\n
\"samples\": [\n          6.2,\n          4.5,\n          5.6\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"SepalWidthCm\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.4335943113621737,\n        \"min\": 2.0,\n        \"max\": 4.4,\n
\"num_unique_values\": 23,\n        \"samples\": [\n          2.3,\n
4.0,\n          3.5\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"PetalLengthCm\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1.7644204199522617,\n        \"min\":
1.0,\n        \"max\": 6.9,\n        \"num_unique_values\": 43,\n
\"samples\": [\n          6.7,\n          3.8,\n          3.7\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"PetalWidthCm\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.7631607417008414,\n        \"min\": 0.1,\n        \"max\": 2.5,\n
\"num_unique_values\": 22,\n        \"samples\": [\n          0.2,\n
1.2,\n          1.3\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Species\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 3,\n        \"samples\":
[\n          \"Iris-setosa\",\n          \"Iris-versicolor\",\n
\"Iris-virginica\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

```
df.shape
```

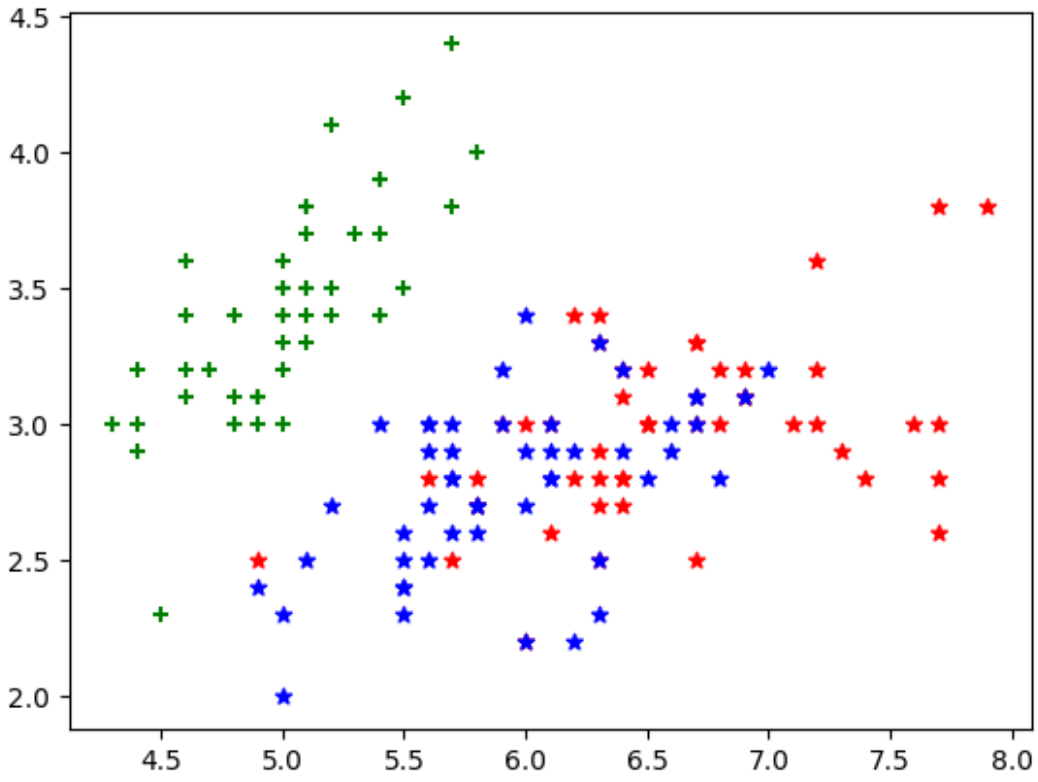
```
(150, 6)
```

```
df.Species.unique()
```

```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'],
dtype=object)
```

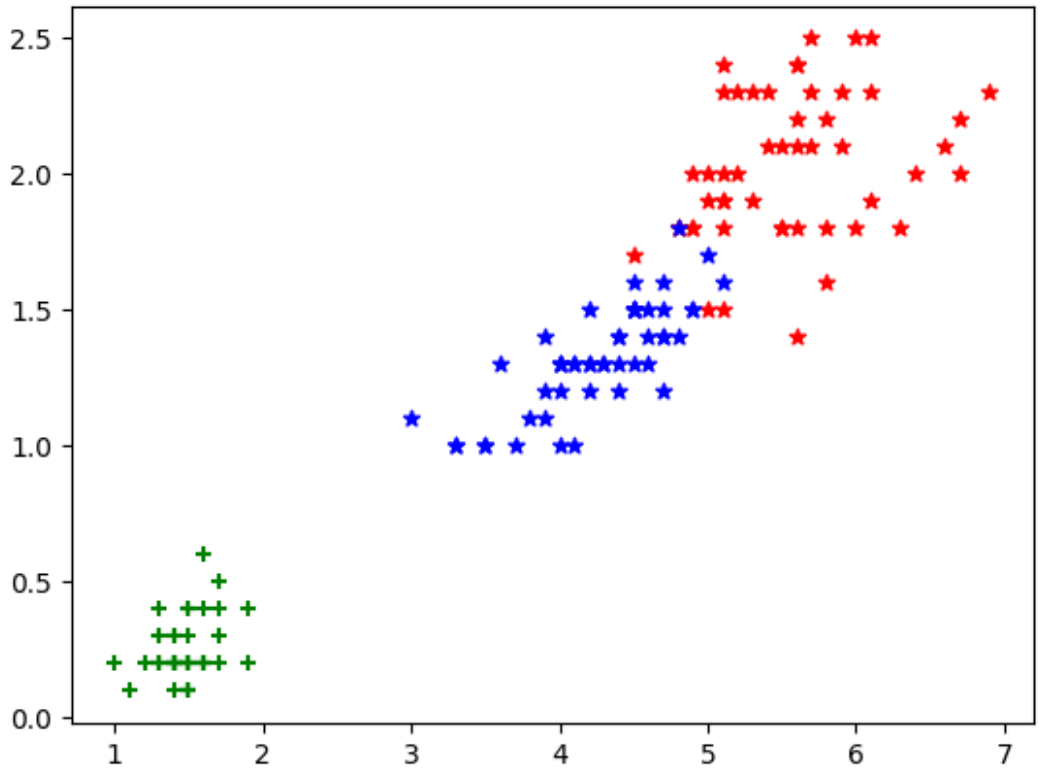
```
#Visualization
df0 = df[:50]
df1 = df[50:100]
df2 = df[100:150]

x = df.iloc[:100,:2]
y = df.Species[:100]

plt.scatter(df0['SepalLengthCm'],df0['SepalWidthCm'],color='green',mar
ker='+')
plt.scatter(df2['SepalLengthCm'],df2['SepalWidthCm'],color='red',marke
r='*')
plt.scatter(df1['SepalLengthCm'],df1['SepalWidthCm'],color='blue',mark
er='*')
```

```
<matplotlib.collections.PathCollection at 0x7e49c56b85d0>
```



```
plt.scatter(df0['PetalLengthCm'],df0['PetalWidthCm'],color='green',marker='+')
plt.scatter(df2['PetalLengthCm'],df2['PetalWidthCm'],color='red',marker='*')
plt.scatter(df1['PetalLengthCm'],df1['PetalWidthCm'],color='blue',marker='*')
```

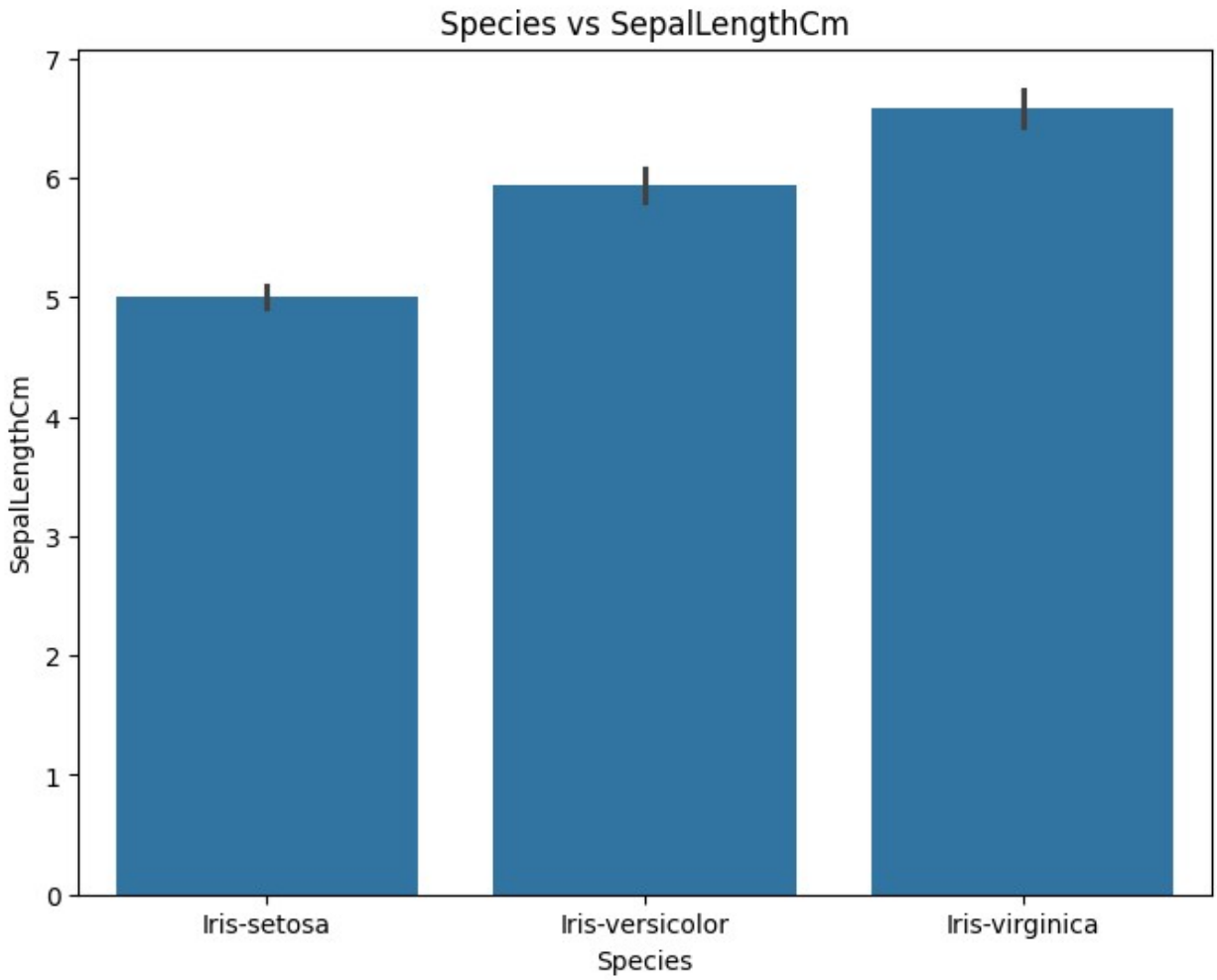
```
<matplotlib.collections.PathCollection at 0x7e49c3356450>
```




```
print(df['Species'].unique())
```

```
['Iris-setosa' 'Iris-versicolor' 'Iris-virginica']
```

```
plt.figure(figsize=(8,6))
sns.barplot(x='Species',y='SepalLengthCm',data=df)
plt.title('Species vs SepalLengthCm')
plt.xlabel('Species')
plt.ylabel('SepalLengthCm')
plt.show()
```




```
x=df.drop('Species',axis=1)
y=df['Species']

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,rando
m_state=42)
print(x_train.shape)
print(x_test.shape)
```

```
(120, 5)
(30, 5)
```

```
model = SVC()# c=1.0, kernel=rbf
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
accuracy_score(y_test,y_predict)
```

```
1.0
```

```
print(classification_report(y_test,y_predict))
```

```
                  precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        10
Iris-versicolor       1.00      1.00      1.00         9
 Iris-virginica       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
```

```
confusion_matrix(y_test,y_predict)
```

```
array([[10,  0,  0],
       [ 0,  9,  0],
       [ 0,  0, 11]])
```

```
# Parameter Tuning
model = SVC(C=0.1)# c=0.1
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
accuracy_score(y_test,y_predict)
```

```
1.0
```

```
model = SVC(C=10)# c=10
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
accuracy_score(y_test,y_predict)
```

```
1.0
```

```
model = SVC(C=100)# c=100
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
accuracy_score(y_test,y_predict)
```

```
1.0
```

```
#gamma
model = SVC(gamma=0.01)# gamma=0.01
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
accuracy_score(y_test,y_predict)
```

```
1.0
```

```
#gamma
model = SVC(gamma=100)# gamma=0.01
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
accuracy_score(y_test,y_predict)
```

```
0.3
```

```
model = SVC(C=100,gamma=0.1,kernel='poly')# gamma=1
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
accuracy_score(y_test,y_predict)
```

```
1.0
```

```
#Kernel
model = SVC(kernel='linear')# gamma=1
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
accuracy_score(y_test,y_predict)
```

```
1.0
```

```
model = SVC(kernel='poly')
model.fit(x_train,y_train)
y_predict = model.predict(x_test)
accuracy_score(y_test,y_predict)
```

```
1.0
```