codsoft-ds-2

August 13, 2024

TASK-2 CLASSIFICATION OF IRIS FLOWER

[1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
pd.set_option('display.max_columns', None)
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

[2]:
```
iris = pd.read_csv('/content/Iris.csv')
```

[3]:
```
iris.head()
```

[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

[4]:
```
iris.shape
```

[4]:
```
(150, 6)
```

[5]:
```
iris.columns
```

[5]:
```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

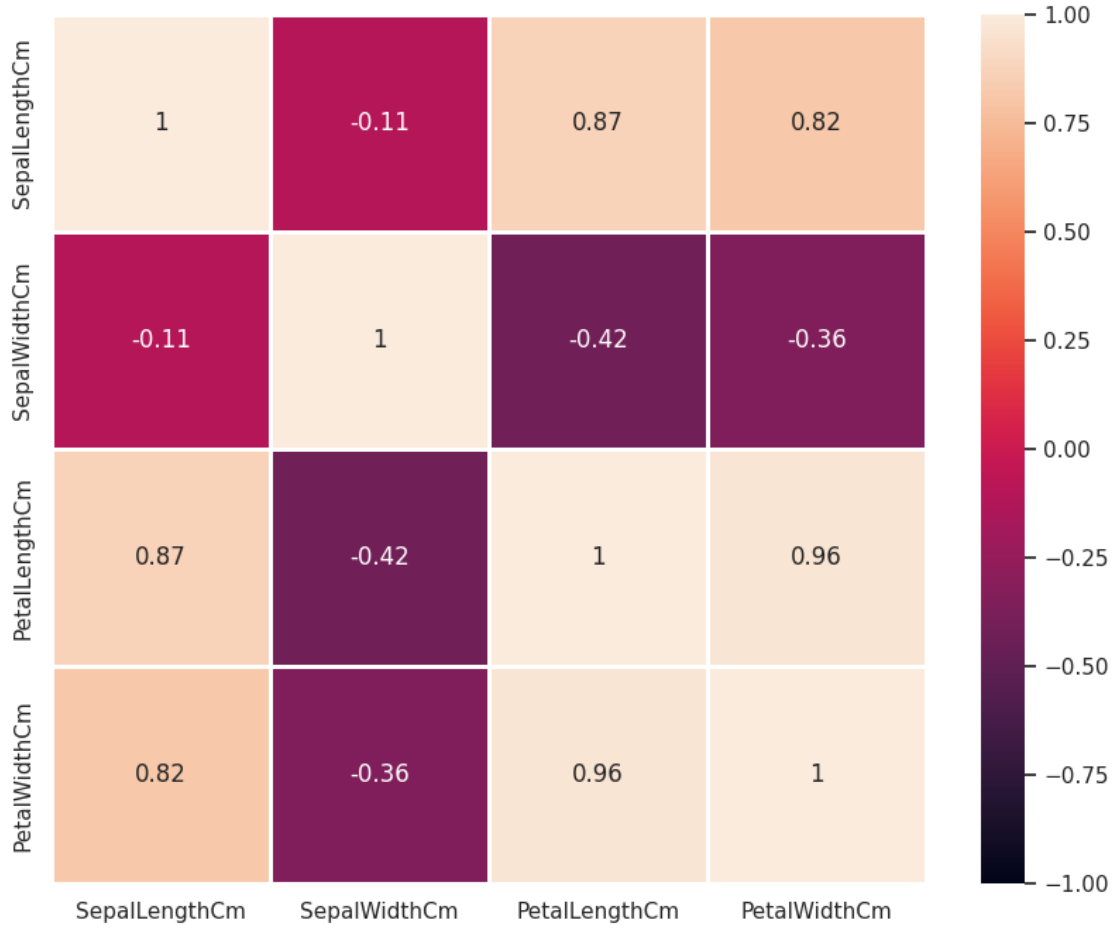
[6]:
```
iris.describe()
```

[6]:

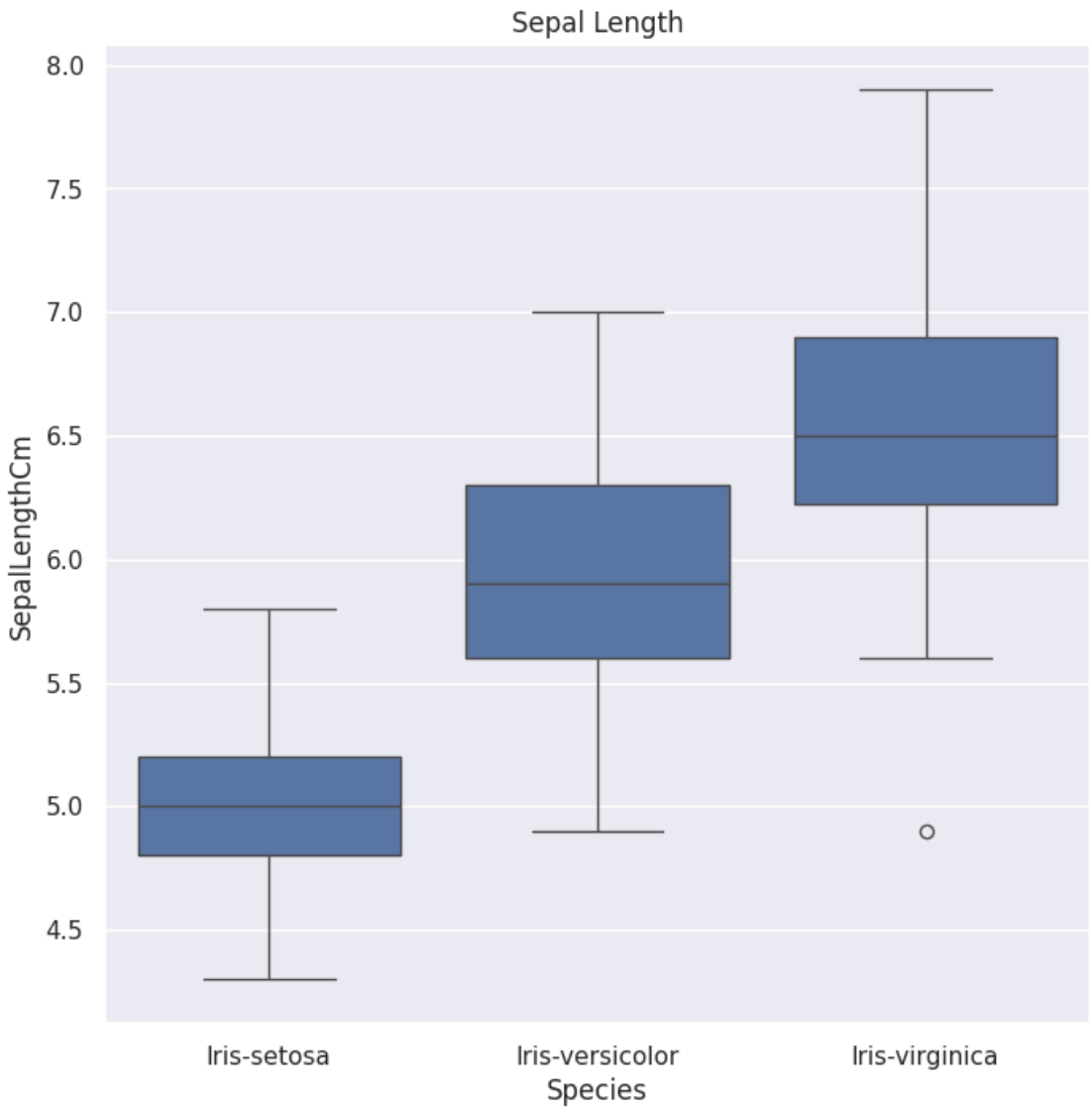
	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000

50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

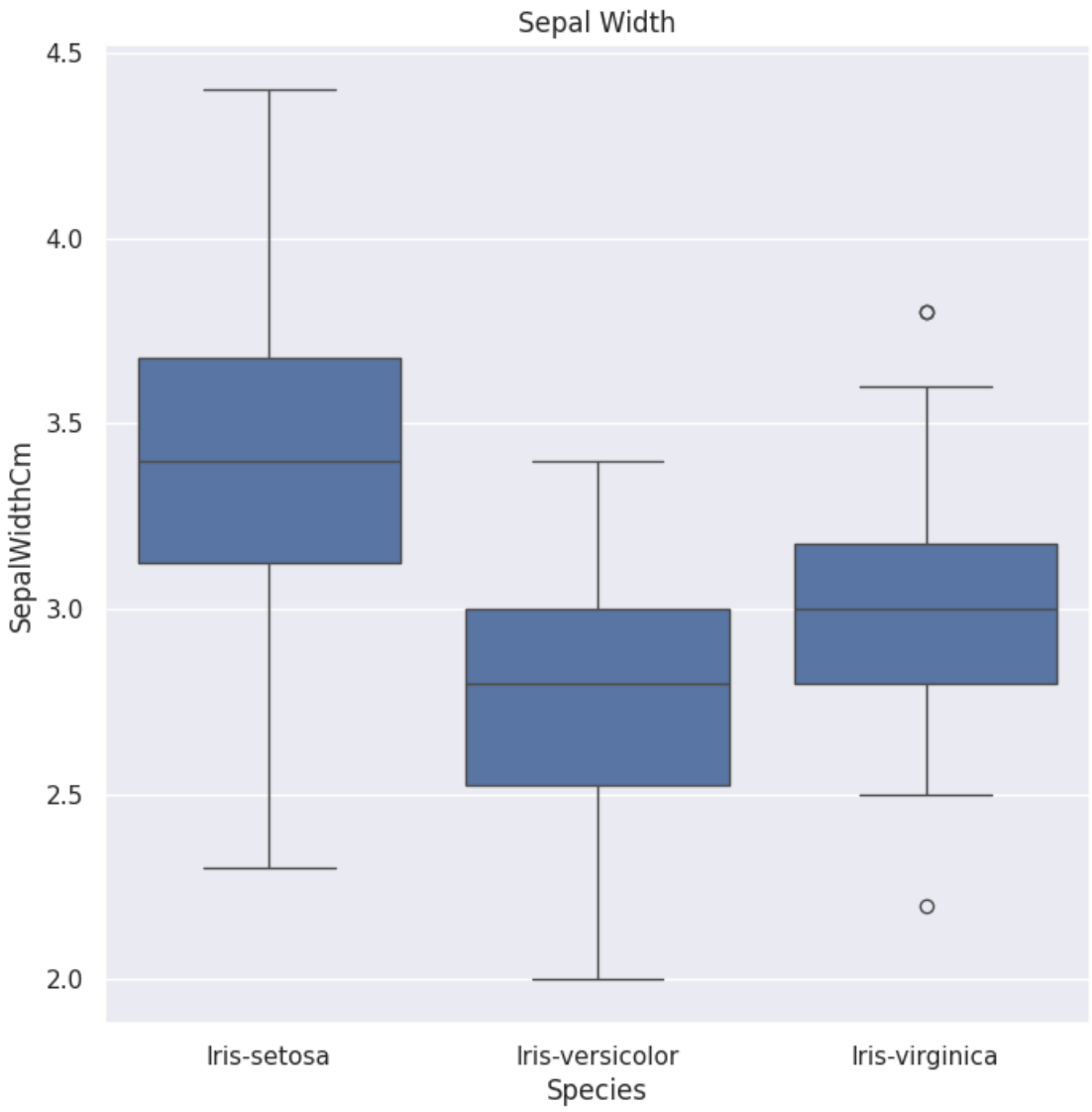
[7]:
```
plt.figure(figsize=(10,8))
sns.heatmap(iris[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm',␣
 ↪'PetalWidthCm']].corr(), vmin=-1.0, vmax=1.0, annot=True, linewidths=2)
plt.show()
```



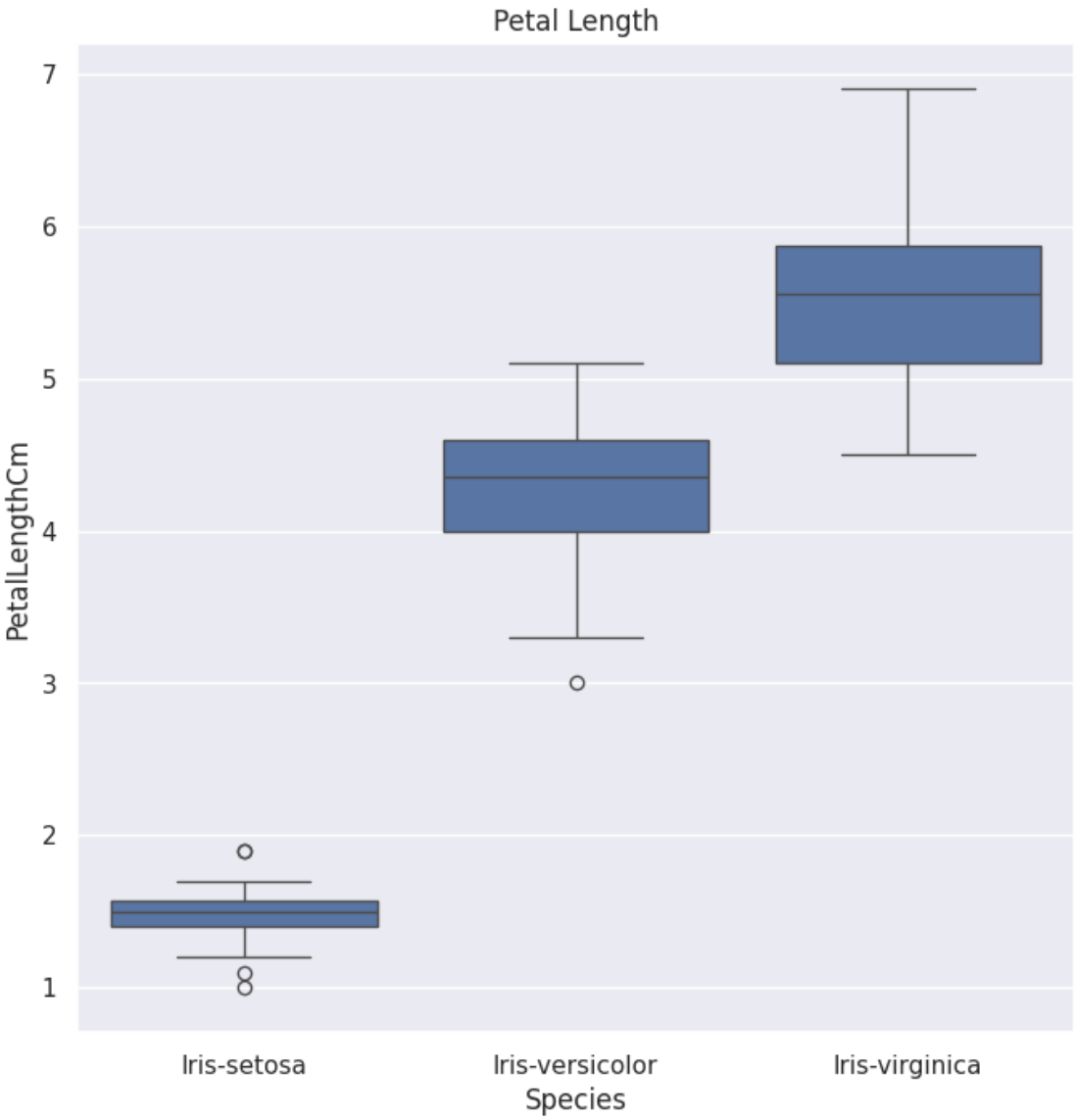
[8]:
```
plt.figure(figsize=(8,8))
ax = sns.boxplot(x="Species", y="SepalLengthCm", data=iris).set_title('Sepal␣
 ↪Length')
plt.show()
```



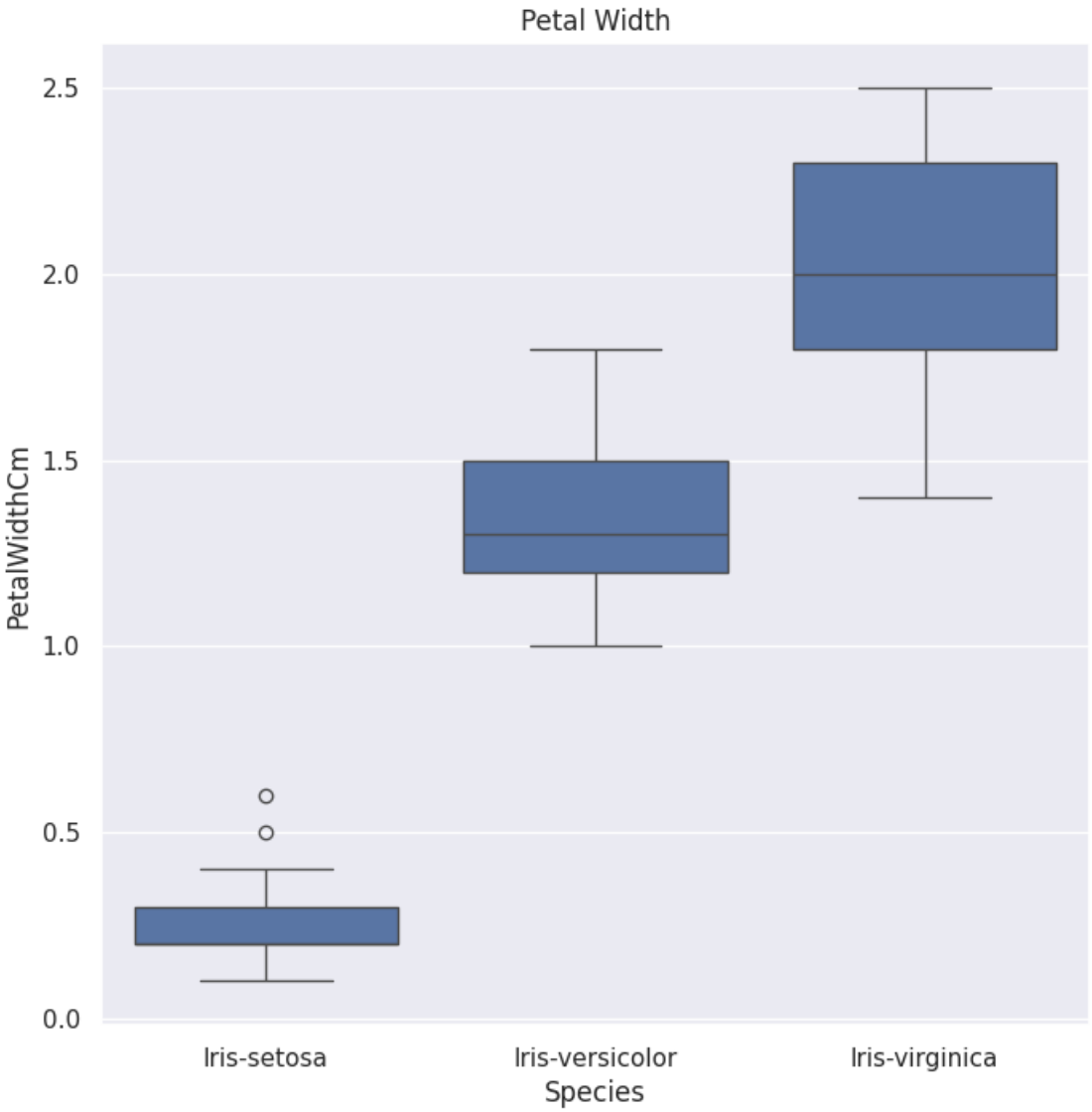
[9]:
```
plt.figure(figsize=(8,8))
ax = sns.boxplot(x="Species", y="SepalWidthCm", data=iris).set_title('Sepal␣
↪Width')
plt.show()
```



[10]:
```
plt.figure(figsize=(8,8))
ax = sns.boxplot(x="Species", y="PetalLengthCm", data=iris).set_title('Petal␣
↪Length')
plt.show()
```



[11]:
```
plt.figure(figsize=(8,8))
ax = sns.boxplot(x="Species", y="PetalWidthCm", data=iris).set_title('Petal␣
↪Width')
plt.show()
```



[12]:
```
iris.drop(['Id'], axis=1, inplace=True)
training = pd.concat([iris[:40], iris[50:90], iris[100:140]])
test = pd.concat([iris[40:50], iris[90:100], iris[140:]])
training_X = training[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm',␣
 ↪'PetalWidthCm']]
training_y = training['Species']
test_X  = test[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm',␣
 ↪'PetalWidthCm']]
test_y = test['Species']
```

[13]:
```
print('Training set:', training_X.shape)
print('Test set:', test_X.shape)
```

```
Training set: (120, 4)
Test set: (30, 4)
```

[14]:
```
from sklearn.svm import LinearSVC
SVC_classifier = LinearSVC(multi_class='crammer_singer', max_iter=3000).
 ↪fit(training_X, training_y)
SVC_classifier.score(training_X, training_y)
print('Training accuracy:', SVC_classifier.score(training_X, training_y))
print('Test accuracy:', SVC_classifier.score(test_X, test_y))
```

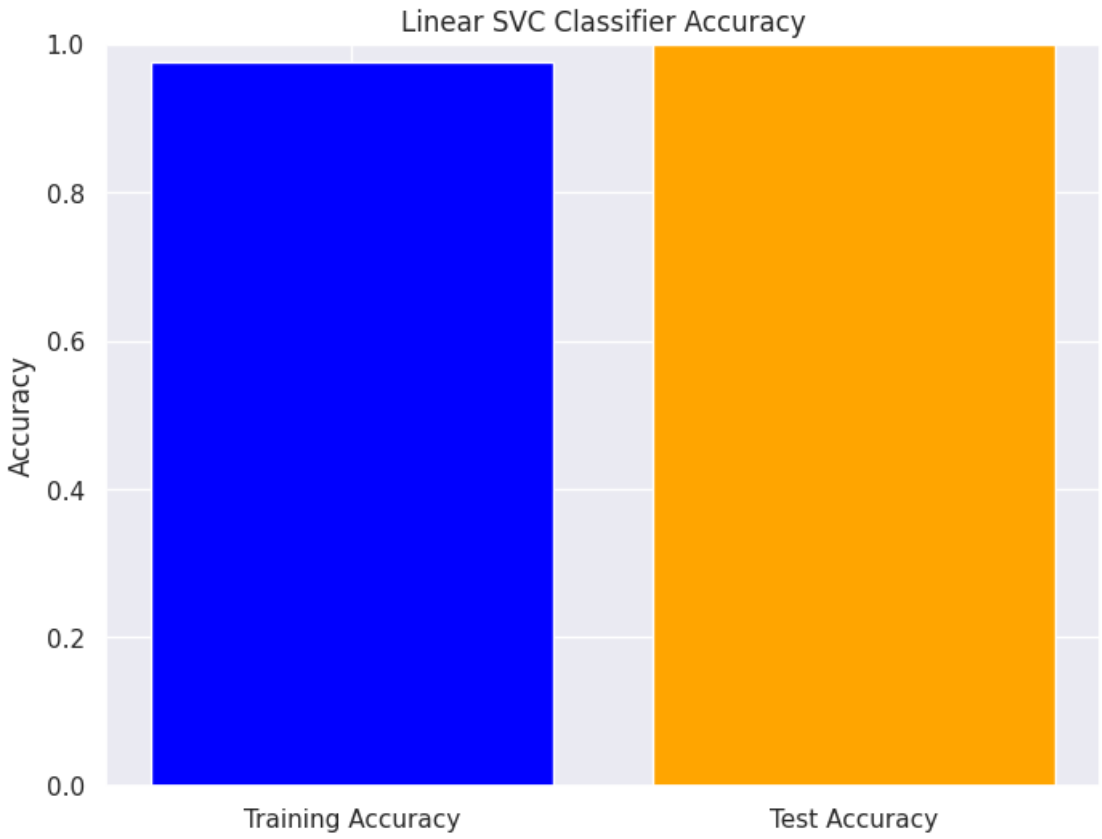
```
Training accuracy: 0.975
Test accuracy: 1.0
/usr/local/lib/python3.10/dist-packages/sklearn/svm/_classes.py:32:
FutureWarning: The default value of `dual` will change from `True` to `'auto'`
in 1.5. Set the value of `dual` explicitly to suppress the warning.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/svm/_base.py:1250:
ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
```

[16]:
```
from sklearn.svm import LinearSVC
SVC_classifier = LinearSVC(multi_class='crammer_singer', max_iter=3000).
 ↪fit(training_X, training_y)
train_accuracy = SVC_classifier.score(training_X, training_y) # Store the␣
 ↪training accuracy
print('Training accuracy:', train_accuracy)
test_accuracy = SVC_classifier.score(test_X, test_y) # Store the test accuracy
print('Test accuracy:', test_accuracy)

# Plot accuracies
plt.figure(figsize=(8, 6))
plt.bar(['Training Accuracy', 'Test Accuracy'], [train_accuracy,␣
 ↪test_accuracy], color=['blue', 'orange']) # Use the stored values
plt.ylim(0, 1)
plt.ylabel('Accuracy')
plt.title('Linear SVC Classifier Accuracy')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/svm/_classes.py:32:
FutureWarning: The default value of `dual` will change from `True` to `'auto'`
in 1.5. Set the value of `dual` explicitly to suppress the warning.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/svm/_base.py:1250:
ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn(
```

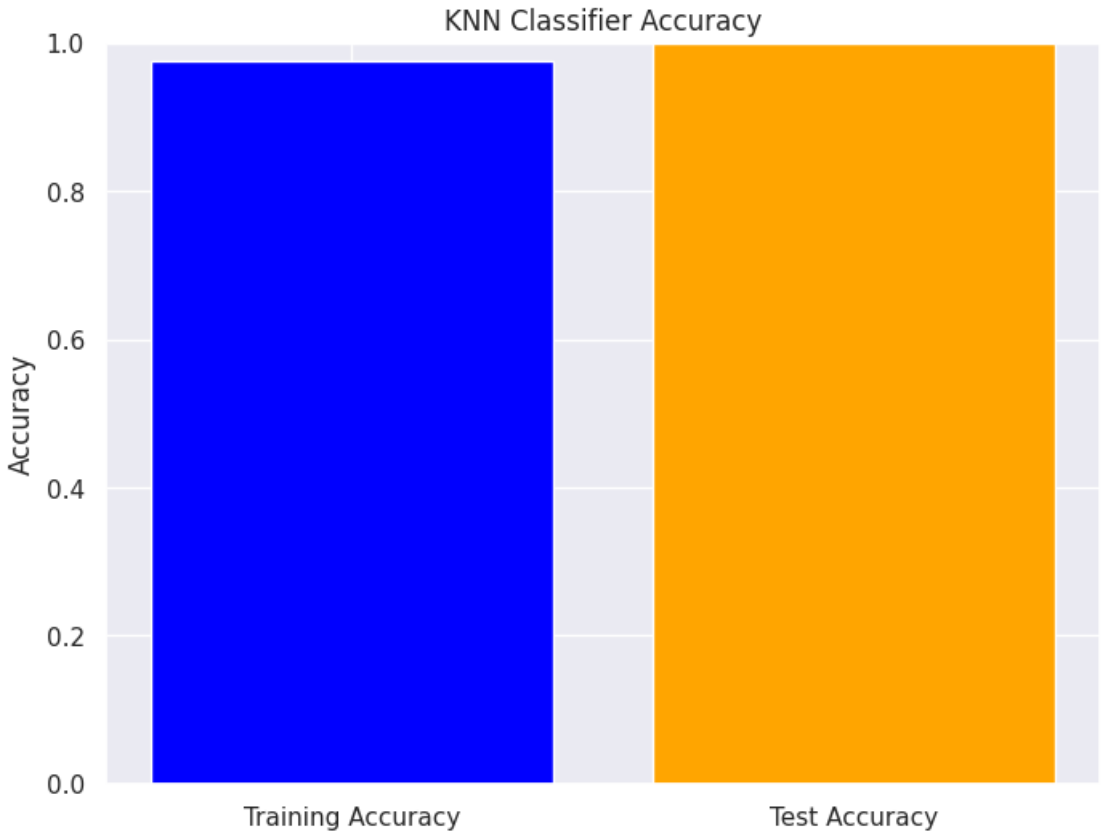
```
Training accuracy: 0.975
Test accuracy: 1.0
```



[18]:
```
from sklearn.neighbors import KNeighborsClassifier
KNN_classifier = KNeighborsClassifier().fit(training_X, training_y)
print('Training accuracy:', KNN_classifier.score(training_X, training_y))
print('Test accuracy:', KNN_classifier.score(test_X, test_y))
```

```
Training accuracy: 0.9666666666666667
Test accuracy: 1.0
```

[19]:
```
# Plot accuracies
plt.figure(figsize=(8, 6))
plt.bar(['Training Accuracy', 'Test Accuracy'], [train_accuracy,␣
  ↪test_accuracy], color=['blue', 'orange'])
plt.ylim(0, 1)
plt.ylabel('Accuracy')
plt.title('KNN Classifier Accuracy')
plt.show()
```


KNN Classifier Accuracy
1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
Training Accuracy
Test Accuracy