ask-3-iris-flower-classification

July 8, 2024

[ ]:
```
Task 3 - IRIS FLOWER CLASSIFICTAION

The Iris flower dataset consists of three species: setosa, versicolor, and␣
↪virginica.
These species can be distinguished based on their measurements.
Now, imagine that you have the measurements of Iris flowers categorized by␣
↪their respective species.
Your objective is to train a machine learning model that can learn from these␣
↪measurements and
accurately classify the Iris flowers into their respective species.

Use the Iris dataset to develop a model that can classify iris flowers into␣
↪different species
based on their sepal and petal measurements.
This dataset is widely used for introductory classification tasks.
```

[ ]:
```

```

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt


from warnings import filterwarnings
filterwarnings(action='ignore')
```

[3]:
```
iris = pd.read_csv(r"C:\Users\divya\OneDrive\Documents\CodSoft␣
↪Internship\iris_dataset.csv")
print(iris)
```

```
     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 x 5 columns]
```

[4]:
```
print(iris.shape)
```

```
(150, 5)
```

[5]:
```
print(iris.describe())
```

```
       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
```

[6]:
```
print(iris.isna().sum())
print(iris.describe())
```

```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
       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
```

[7]:
```
iris.head()
```

[7]:
```
   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
```

[8]:
```python
iris.head(150)
```

[8]:
```
     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 x 5 columns]
```

[9]:
```python
iris.tail(100)
```

[9]:
```
     sepal_length  sepal_width  petal_length  petal_width          species
50            7.0          3.2           4.7          1.4  Iris-versicolor
51            6.4          3.2           4.5          1.5  Iris-versicolor
52            6.9          3.1           4.9          1.5  Iris-versicolor
53            5.5          2.3           4.0          1.3  Iris-versicolor
54            6.5          2.8           4.6          1.5  Iris-versicolor
..            …            …             …            …                …
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

[100 rows x 5 columns]
```

[22]:
```python
n = len(iris[iris['species'] == 'Iris-versicolor'])
print("No of Iris-Versicolor in Dataset:",n)
```

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

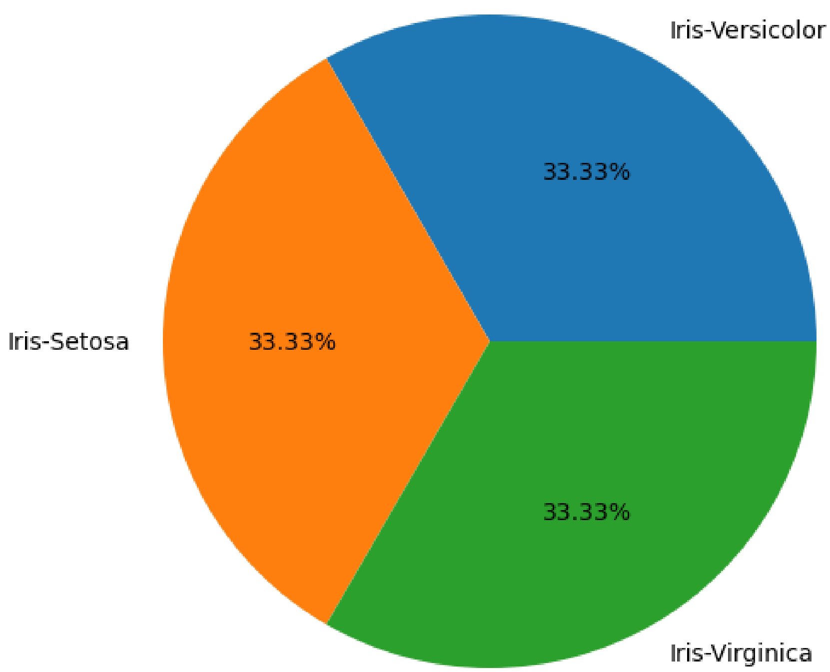
[23]:
```python
n2 = len(iris[iris['species'] == 'Iris-setosa'])
print("No of Iris-Setosa in Dataset:",n2)
```

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

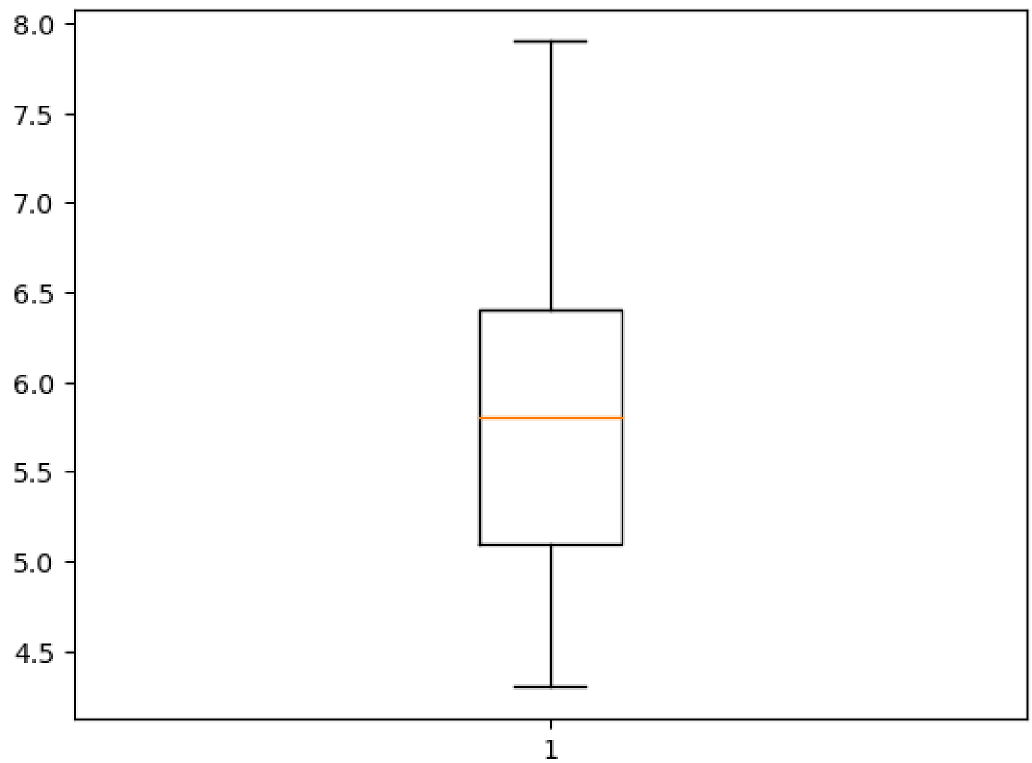
[25]:
```
n1 = len(iris[iris['species'] == 'Iris-virginica'])
print("No of Iris-Virginica in Dataset:",n1)
```

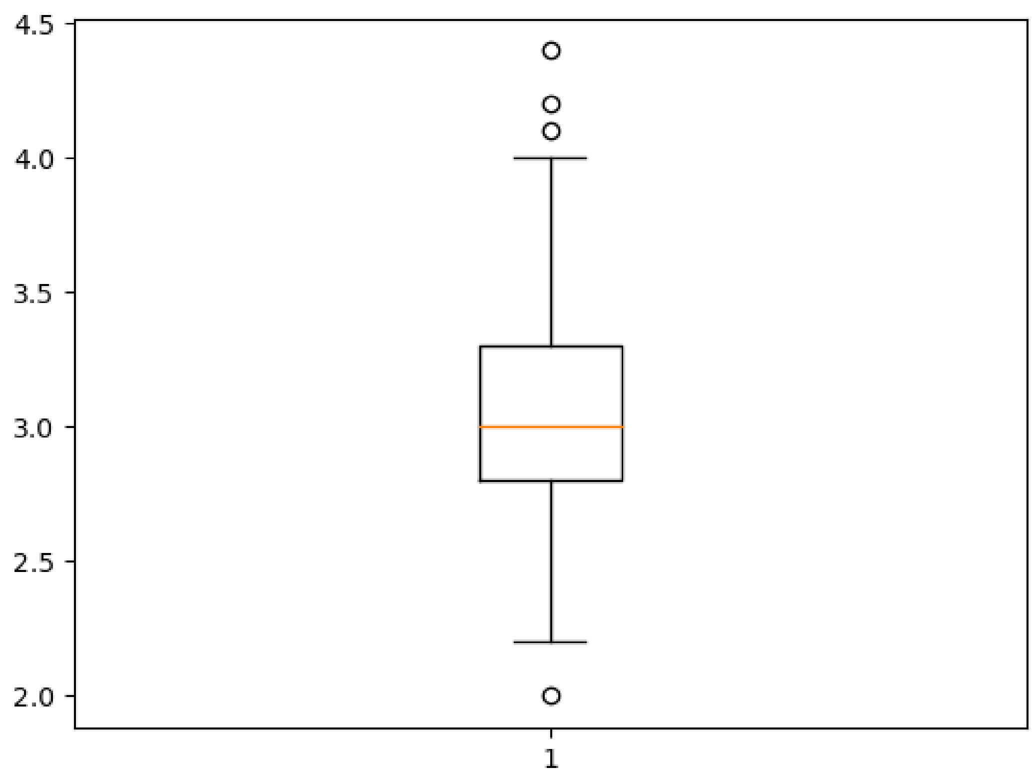
```
No of Iris-Virginica in Dataset: 50
```

[26]:
```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.axis('equal')
l = ['Iris-Versicolor', 'Iris-Setosa', 'Iris-Virginica']
s = [50,50,50]
ax.pie(s, labels = l,autopct='%1.2f%%')
plt.show()
```

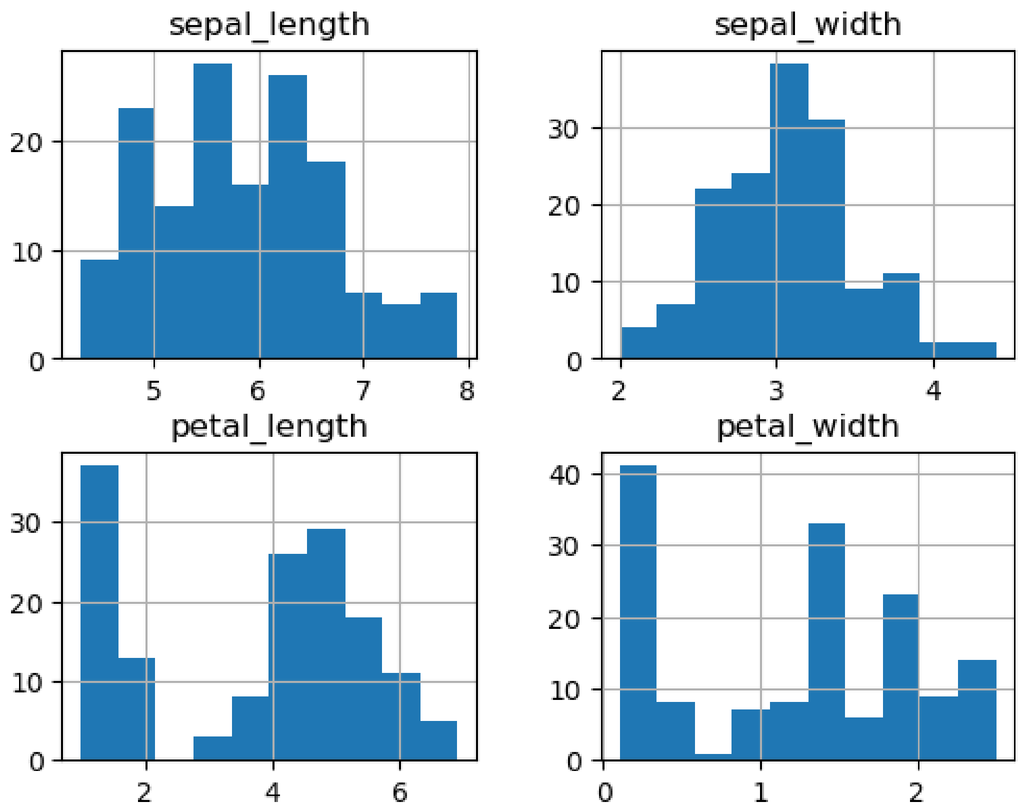


[27]:
```
import matplotlib.pyplot as plt
plt.figure(1)
plt.boxplot([iris['sepal_length']])
plt.figure(2)
plt.boxplot([iris['sepal_width']])
plt.show()
```


8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
1

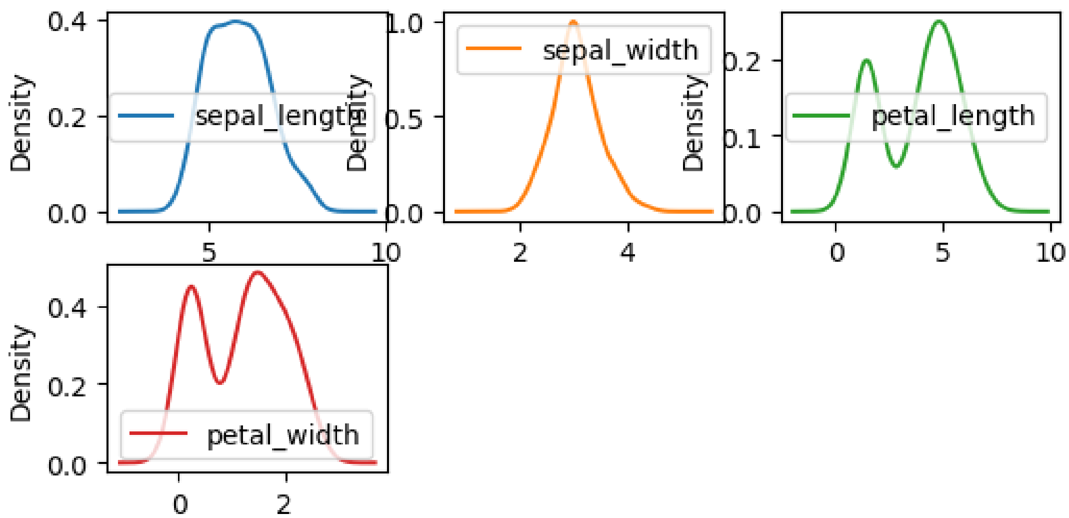

4.5
4.0
3.5
3.0
2.5
2.0
1

[28]:
```
iris.hist()
plt.show()
```



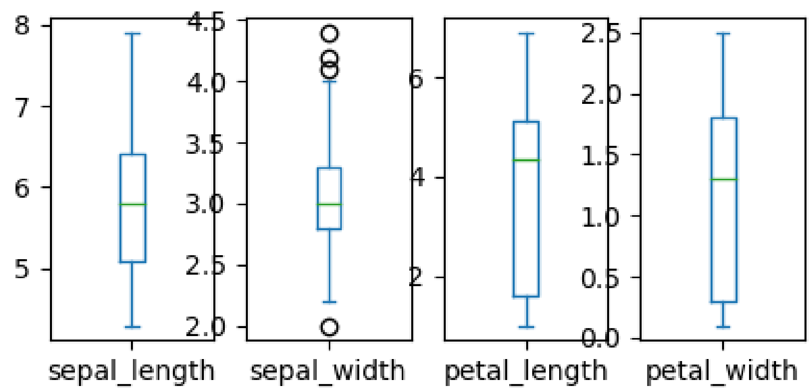
[29]:
```
iris.plot(kind ='density',subplots = True, layout =(3,3),sharex = False)
```

[29]:
```
array([[<Axes: ylabel='Density'>, <Axes: ylabel='Density'>,
        <Axes: ylabel='Density'>],
       [<Axes: ylabel='Density'>, <Axes: ylabel='Density'>,
        <Axes: ylabel='Density'>],
       [<Axes: ylabel='Density'>, <Axes: ylabel='Density'>,
        <Axes: ylabel='Density'>]], dtype=object)
```



[30]:
```
iris.plot(kind ='box',subplots = True, layout =(2,5),sharex = False)
```

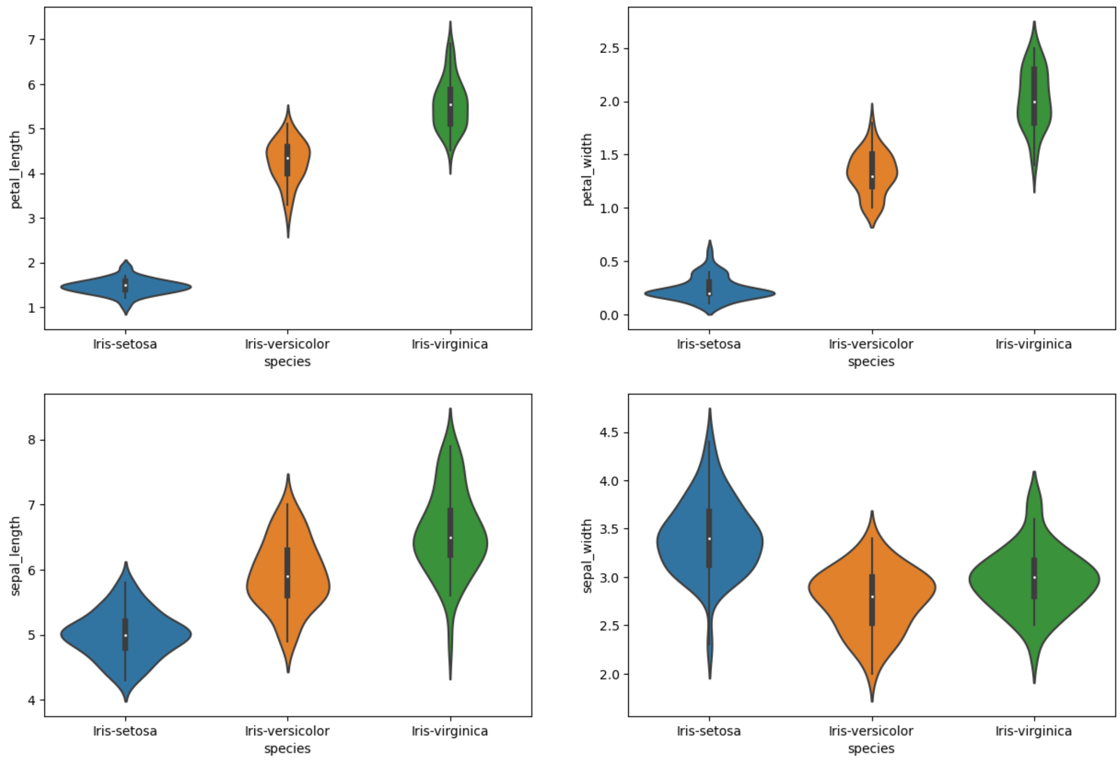
[30]:
```
sepal_length        Axes(0.125,0.53;0.133621x0.35)
sepal_width      Axes(0.285345,0.53;0.133621x0.35)
petal_length      Axes(0.44569,0.53;0.133621x0.35)
petal_width      Axes(0.606034,0.53;0.133621x0.35)
dtype: object
```



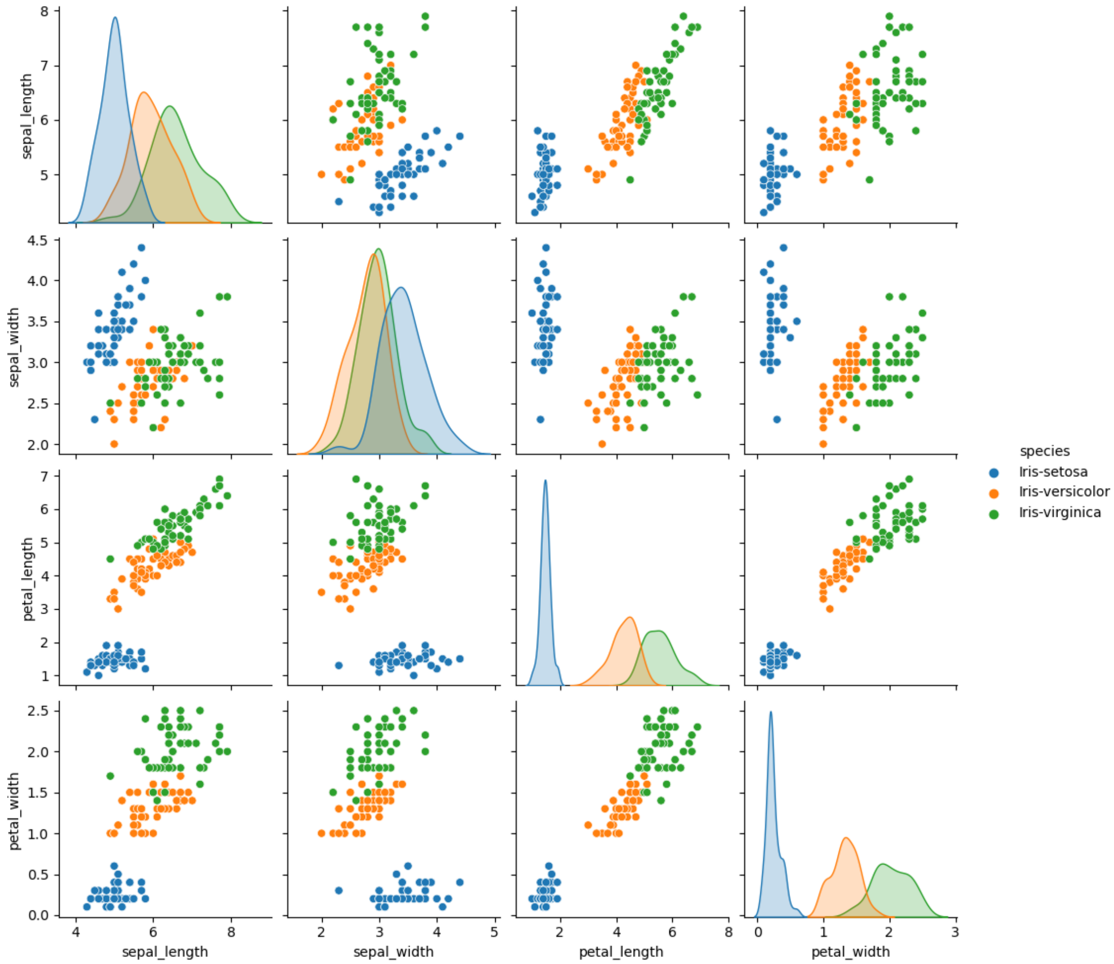
[31]:
```
plt.figure(figsize=(15,10))
plt.subplot(2,2,1)
sns.violinplot(x='species',y='petal_length',data=iris)
plt.subplot(2,2,2)
sns.violinplot(x='species',y='petal_width',data=iris)
plt.subplot(2,2,3)
```

```
sns.violinplot(x='species',y='sepal_length',data=iris)
plt.subplot(2,2,4)
sns.violinplot(x='species',y='sepal_width',data=iris)
```

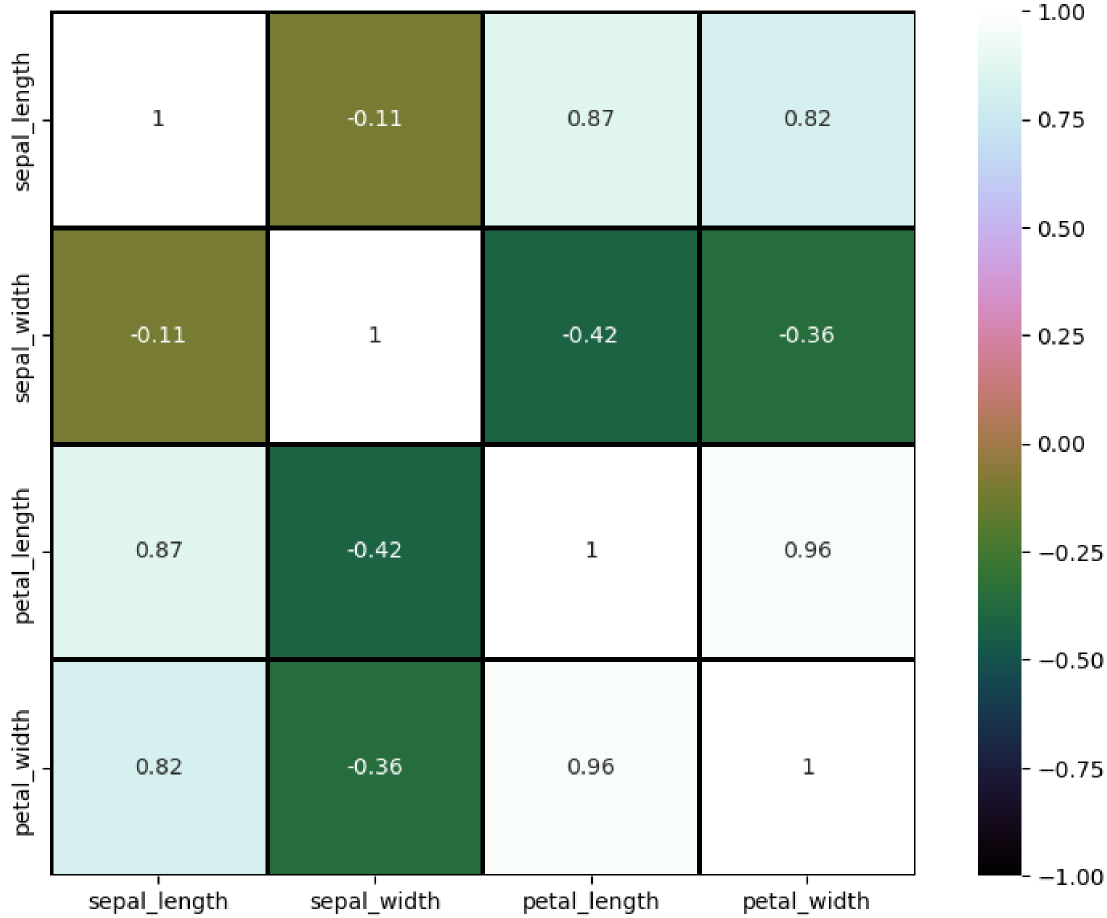
[31]: <Axes: xlabel='species', ylabel='sepal_width'>



[32]:
```
sns.pairplot(iris,hue='species');
```



[33]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.heatmap(iris.
 ↪corr(),annot=True,cmap='cubehelix',linewidths=1,linecolor='k',square=True,mask=False,
 ↪vmin=-1, vmax=1,cbar_kws={"orientation": "vertical"},cbar=True)
```



[34]:
```python
X = iris['sepal_length'].values.reshape(-1,1)
print(X)
```

```
[[5.1]
 [4.9]
 [4.7]
 [4.6]
 [5. ]
 [5.4]
 [4.6]
 [5. ]
 [4.4]
 [4.9]
 [5.4]
 [4.8]
 [4.8]
 [4.3]
 [5.8]
```

```
[5.7]
[5.4]
[5.1]
[5.7]
[5.1]
[5.4]
[5.1]
[4.6]
[5.1]
[4.8]
[5. ]
[5. ]
[5.2]
[5.2]
[4.7]
[4.8]
[5.4]
[5.2]
[5.5]
[4.9]
[5. ]
[5.5]
[4.9]
[4.4]
[5.1]
[5. ]
[4.5]
[4.4]
[5. ]
[5.1]
[4.8]
[5.1]
[4.6]
[5.3]
[5. ]
[7. ]
[6.4]
[6.9]
[5.5]
[6.5]
[5.7]
[6.3]
[4.9]
[6.6]
[5.2]
[5. ]
[5.9]
[6. ]
```

```
[6.1]
[5.6]
[6.7]
[5.6]
[5.8]
[6.2]
[5.6]
[5.9]
[6.1]
[6.3]
[6.1]
[6.4]
[6.6]
[6.8]
[6.7]
[6. ]
[5.7]
[5.5]
[5.5]
[5.8]
[6. ]
[5.4]
[6. ]
[6.7]
[6.3]
[5.6]
[5.5]
[5.5]
[6.1]
[5.8]
[5. ]
[5.6]
[5.7]
[5.7]
[6.2]
[5.1]
[5.7]
[6.3]
[5.8]
[7.1]
[6.3]
[6.5]
[7.6]
[4.9]
[7.3]
[6.7]
[7.2]
[6.5]
```

```
 [6.4]
 [6.8]
 [5.7]
 [5.8]
 [6.4]
 [6.5]
 [7.7]
 [7.7]
 [6. ]
 [6.9]
 [5.6]
 [7.7]
 [6.3]
 [6.7]
 [7.2]
 [6.2]
 [6.1]
 [6.4]
 [7.2]
 [7.4]
 [7.9]
 [6.4]
 [6.3]
 [6.1]
 [7.7]
 [6.3]
 [6.4]
 [6. ]
 [6.9]
 [6.7]
 [6.9]
 [5.8]
 [6.8]
 [6.7]
 [6.7]
 [6.3]
 [6.5]
 [6.2]
 [5.9]]
```

[40]:

```
Y = iris['sepal_width'].values.reshape(-1,1)
print(Y)
```

```
[[3.5]
 [3. ]
 [3.2]
 [3.1]
 [3.6]
```

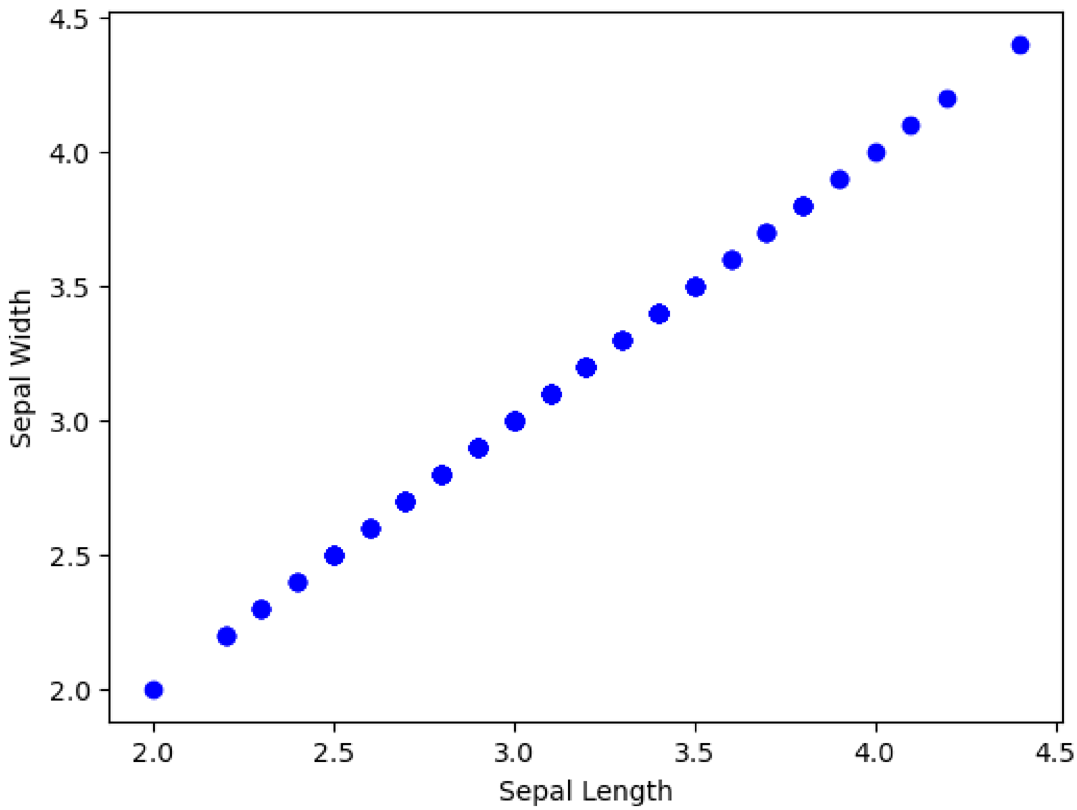
```
[3.9]
[3.4]
[3.4]
[2.9]
[3.1]
[3.7]
[3.4]
[3. ]
[3. ]
[4. ]
[4.4]
[3.9]
[3.5]
[3.8]
[3.8]
[3.4]
[3.7]
[3.6]
[3.3]
[3.4]
[3. ]
[3.4]
[3.5]
[3.4]
[3.2]
[3.1]
[3.4]
[4.1]
[4.2]
[3.1]
[3.2]
[3.5]
[3.1]
[3. ]
[3.4]
[3.5]
[2.3]
[3.2]
[3.5]
[3.8]
[3. ]
[3.8]
[3.2]
[3.7]
[3.3]
[3.2]
[3.2]
[3.1]
```

```
[2.3]
[2.8]
[2.8]
[3.3]
[2.4]
[2.9]
[2.7]
[2. ]
[3. ]
[2.2]
[2.9]
[2.9]
[3.1]
[3. ]
[2.7]
[2.2]
[2.5]
[3.2]
[2.8]
[2.5]
[2.8]
[2.9]
[3. ]
[2.8]
[3. ]
[2.9]
[2.6]
[2.4]
[2.4]
[2.7]
[2.7]
[3. ]
[3.4]
[3.1]
[2.3]
[3. ]
[2.5]
[2.6]
[3. ]
[2.6]
[2.3]
[2.7]
[3. ]
[2.9]
[2.9]
[2.5]
[2.8]
[3.3]
```

```
[2.7]
[3. ]
[2.9]
[3. ]
[3. ]
[2.5]
[2.9]
[2.5]
[3.6]
[3.2]
[2.7]
[3. ]
[2.5]
[2.8]
[3.2]
[3. ]
[3.8]
[2.6]
[2.2]
[3.2]
[2.8]
[2.8]
[2.7]
[3.3]
[3.2]
[2.8]
[3. ]
[2.8]
[3. ]
[2.8]
[3.8]
[2.8]
[2.8]
[2.6]
[3. ]
[3.4]
[3.1]
[3. ]
[3.1]
[3.1]
[3.1]
[2.7]
[3.2]
[3.3]
[3. ]
[2.5]
[3. ]
[3.4]
```

```
[3. ]]
```

[41]:
```
plt.xlabel("Sepal Length")
plt.ylabel("Sepal Width")
plt.scatter(X,Y,color='b')
plt.show()
```



[42]:
```
corr_mat = iris.corr()
print(corr_mat)
```

```
              sepal_length  sepal_width  petal_length  petal_width
sepal_length      1.000000    -0.109369      0.871754     0.817954
sepal_width      -0.109369     1.000000     -0.420516    -0.356544
petal_length      0.871754    -0.420516      1.000000     0.962757
petal_width       0.817954    -0.356544      0.962757     1.000000
```

[43]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from sklearn import metrics
```

```
from sklearn.tree import DecisionTreeClassifier
```

[44]:
```
train, test = train_test_split(iris, test_size = 0.25)
print(train.shape)
print(test.shape)
```

```
(112, 5)
(38, 5)
```

[46]:
```
train_X = train[['sepal_length', 'sepal_width', 'petal_length',
                 'petal_width']]
train_y = train.species

test_X = test[['sepal_length', 'sepal_width', 'petal_length',
                 'petal_width']]
test_y = test.species
```

[47]:
```
train_X.head()
```

[47]:

	sepal_length	sepal_width	petal_length	petal_width
109	7.2	3.6	6.1	2.5
123	6.3	2.7	4.9	1.8
80	5.5	2.4	3.8	1.1
106	4.9	2.5	4.5	1.7
45	4.8	3.0	1.4	0.3

[48]:
```
test_y.head()
```

[48]:
```
139    Iris-virginica
19        Iris-setosa
12        Iris-setosa
30        Iris-setosa
34        Iris-setosa
Name: species, dtype: object
```

[49]:
```
model = LogisticRegression()
model.fit(train_X, train_y)
prediction = model.predict(test_X)
print('Accuracy:',metrics.accuracy_score(prediction,test_y))
```

```
Accuracy: 0.9736842105263158
```

[50]:
```
from sklearn.metrics import confusion_matrix,classification_report
confusion_mat = confusion_matrix(test_y,prediction)
print("Confusion matrix: \n",confusion_mat)
print(classification_report(test_y,prediction))
```

```
Confusion matrix:
 [[13  0  0]
 [ 0 11  1]
 [ 0  0 13]]
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        13
Iris-versicolor       1.00      0.92      0.96        12
 Iris-virginica       0.93      1.00      0.96        13

       accuracy                           0.97        38
      macro avg       0.98      0.97      0.97        38
   weighted avg       0.98      0.97      0.97        38
```

[51]:
```
from sklearn.svm import SVC
model1 = SVC()
model1.fit(train_X,train_y)

pred_y = model1.predict(test_X)

from sklearn.metrics import accuracy_score
print("Acc=",accuracy_score(test_y,pred_y))
```

```
Acc= 0.9736842105263158
```

[52]:
```
from sklearn.neighbors import KNeighborsClassifier
model2 = KNeighborsClassifier(n_neighbors=5)
model2.fit(train_X,train_y)
y_pred2 = model2.predict(test_X)

from sklearn.metrics import accuracy_score
print("Accuracy Score:",accuracy_score(test_y,y_pred2))
```

```
Accuracy Score: 0.9736842105263158
```

[53]:
```
from sklearn.naive_bayes import GaussianNB
model3 = GaussianNB()
model3.fit(train_X,train_y)
y_pred3 = model3.predict(test_X)

from sklearn.metrics import accuracy_score
print("Accuracy Score:",accuracy_score(test_y,y_pred3))
```

```
Accuracy Score: 0.9473684210526315
```

[54]:
```
from sklearn.tree import DecisionTreeClassifier
model4 = DecisionTreeClassifier(criterion='entropy',random_state=7)
```

```
model4.fit(train_X,train_y)
y_pred4 = model4.predict(test_X)

from sklearn.metrics import accuracy_score
print("Accuracy Score:",accuracy_score(test_y,y_pred4))
```

```
Accuracy Score: 0.9736842105263158
```

[55]:
```
results = pd.DataFrame({
    'Model': ['Logistic Regression','Support Vector Machines', 'Naive␣
 ↪Bayes','KNN' ,'Decision Tree'],
    'Score': [0.947,0.947,0.947,0.947,0.921]})

result_df = results.sort_values(by='Score', ascending=False)
result_df = result_df.set_index('Score')
result_df.head(9)
```

[55]:

Score	Model
0.947	Logistic Regression
0.947	Support Vector Machines
0.947	Naive Bayes
0.947	KNN
0.921	Decision Tree

[ ]: