LetsGrowMore Data science Internship

Beginner Level - Task 1

Iris Flower Classification ML Project:

By: Ashwinraj G

Importing Libraries

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
get_ipython().run_line_magic('matplotlib','inline')
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
```

Loading Dataset

In [2]:

```
df=pd.read_csv("IRIS.csv")
df
```

Out[2]:

	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

Get the size of Dataset

In [3]:

```
data_size=df.shape
print(f"Number of rows:{data_size[0]}")
print(f"Number of colums:{data_size[1]}")
```

```
Number of rows:150
Number of colums:5
```

In [4]:

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

Out[4]:

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

In [5]:

```
df.info
```

Out[5]:

```
<bound method DataFrame.info of      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]>
```

In [6]:

```
df.describe()
```

Out[6]:

	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

In [7]:

```
df.tail
```

Out[7]:

```
<bound method NDFrame.tail of      sepal_length  sepal_width  petal_length  petal_wi
dth          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]>
```

In [8]:

```
df.head()
```

Out[8]:

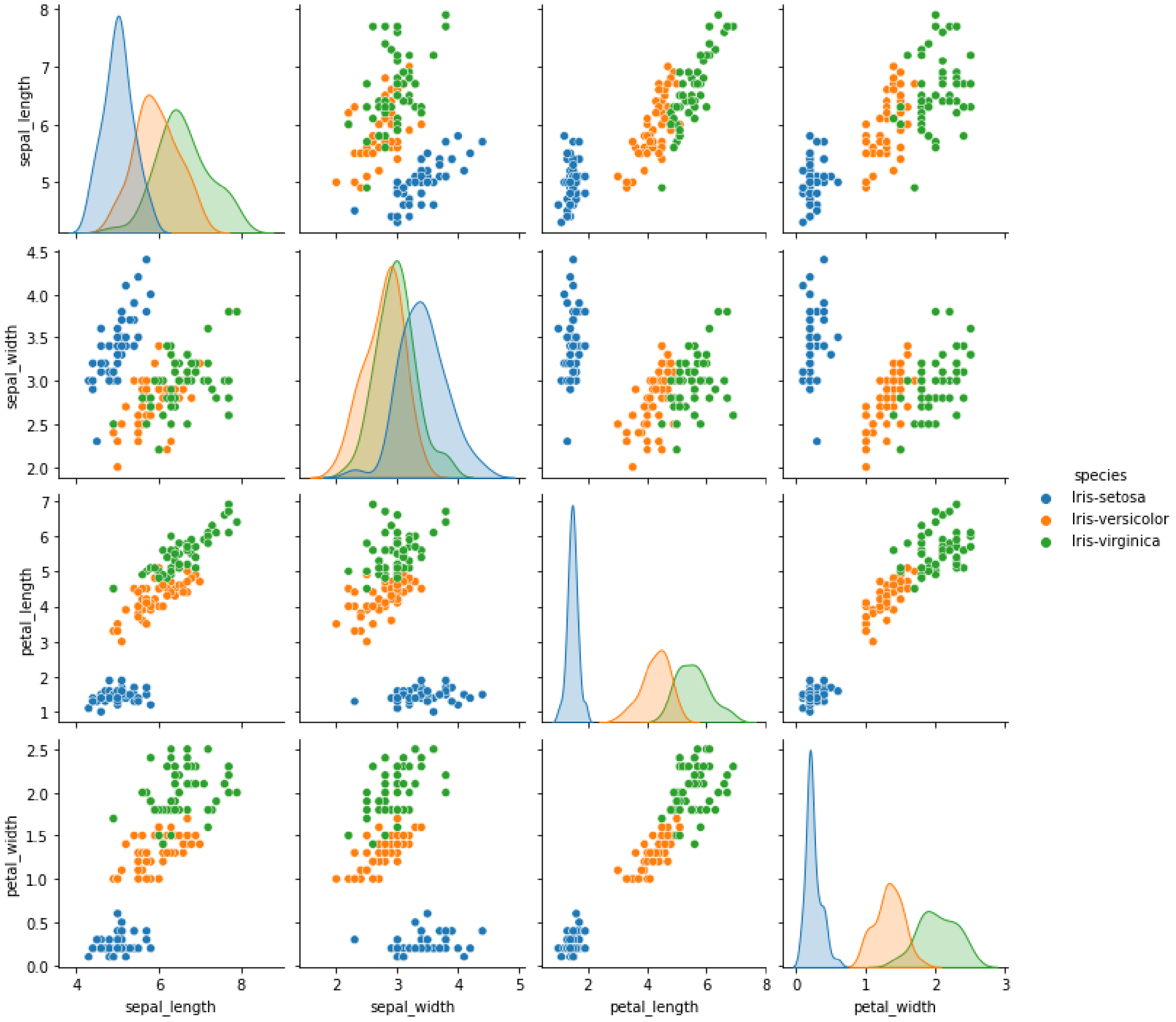
	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [11]:

```
sns.pairplot(df,hue='species')
```

Out[11]:

```
<seaborn.axisgrid.PairGrid at 0x5b635f8>
```



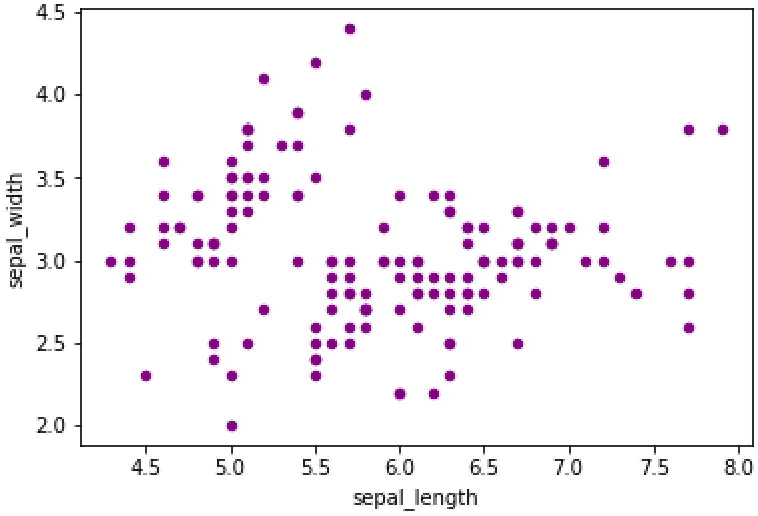
Scatter plot

In [12]:

```
df.plot(kind="scatter",x="sepal_length",y="sepal_width",color="purple",alpha=1)
```

Out[12]:

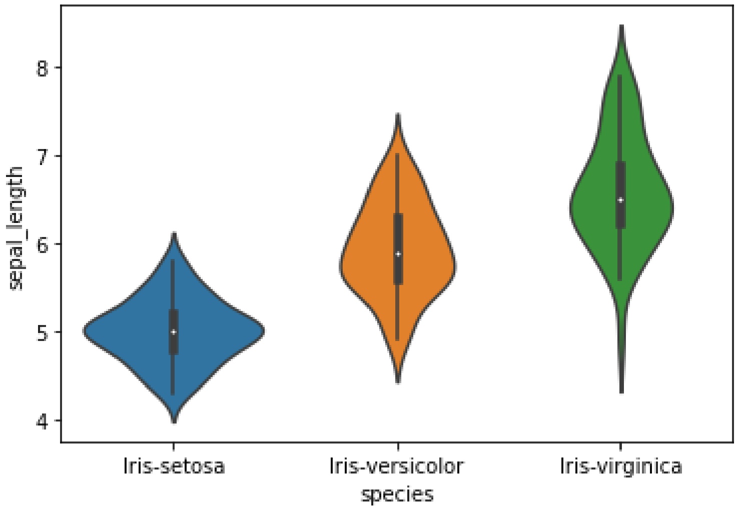
```
<AxesSubplot:xlabel='sepal_length', ylabel='sepal_width'>
```



Violine Plot

In [13]:

```
sns.violinplot(x='species',y='sepal_length',data=df)
plt.show()
```



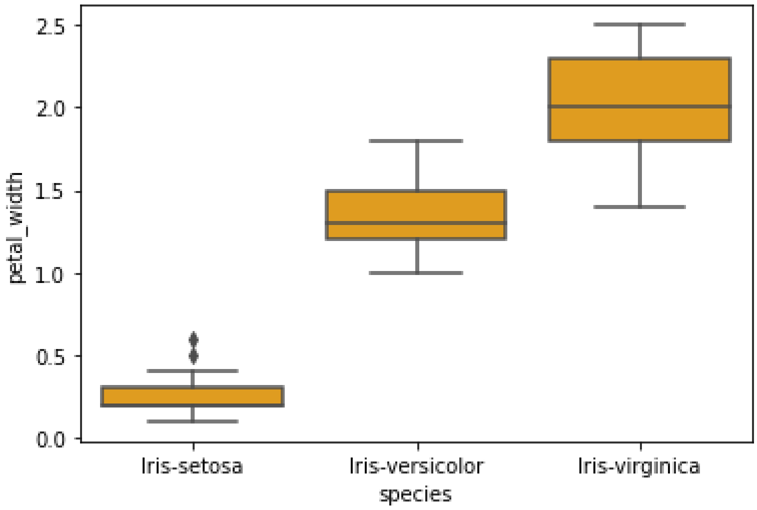
Box Plot

In [14]:

```
sns.boxplot(x="species",y="petal_width",data=df,color="orange")
```

Out[14]:

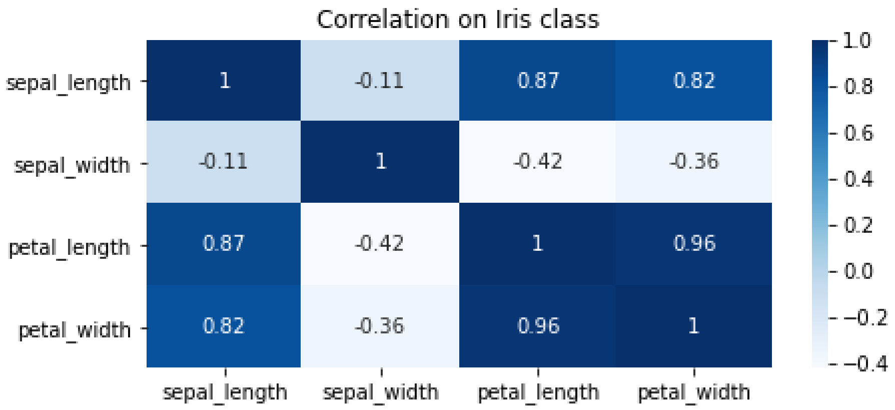
```
<AxesSubplot:xlabel='species', ylabel='petal_width'>
```



Heat Map

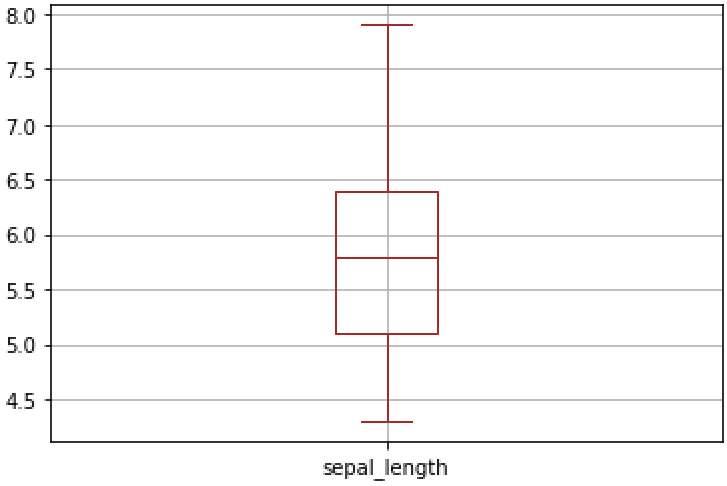
In [15]:

```
plt.subplots(figsize = (7,3))
sns.heatmap(df.corr(),annot=True,cmap="Blues").set_title("Correlation on Iris class"
plt.show()
```



In [16]:
```
df.boxplot(column=['sepal_length'],color="brown")
```

Out[16]:
```
<AxesSubplot:>
```



In [17]:
```
df.cov
```

Out[17]:
```
<bound method DataFrame.cov of      sepal_length  sepal_width  petal_length  petal_w
idth         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]>
```

In [18]:
```
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=0.4,random_state=0)
```

Logistic Regression

In [19]:

```
log_reg=LogisticRegression()
log_reg.fit(x_train,y_train)
predictions=log_reg.predict(x_test)
print("Logistic Regression")
print("The Accuracy score",accuracy_score(y_test,predictions))
print(confusion_matrix(y_test,predictions))
print(classification_report(y_test,predictions))
```

```
Logistic Regression
The Accuracy score 0.9166666666666666
[[16  0  0]
 [ 0 22  1]
 [ 0  4 17]]
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        16
Iris-versicolor       0.85      0.96      0.90        23
 Iris-virginica       0.94      0.81      0.87        21

       accuracy                           0.92        60
      macro avg       0.93      0.92      0.92        60
   weighted avg       0.92      0.92      0.92        60
```

SVM

In [21]:

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn import svm
model=SVC()
clf=svm.SVC(gamma=0.001,C=100.)
model.fit(x_train,y_train)
clf.fit(x_train,y_train)
predicitons=model.predict(x_test)
print("Support vector MAchine")
print('Train_The accuracy of the SVM is:',accuracy_score(predictions,y_test))
```

```
Support vector MAchine
Train_The accuracy of the SVM is: 0.9166666666666666
```

In [22]:

```
model = SVC()
model.fit(x_train,y_train)
prediction=model.predict(x_train)
print("Support Vector Machines")
print("Train-Ther accuracy of the SVM is:",accuracy_score(y_test,predictions))
print(classification_report(y_test,predictions))
```

```
Support Vector Machines
Train-Ther accuracy of the SVM is: 0.9166666666666666
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        16
Iris-versicolor       0.85      0.96      0.90        23
 Iris-virginica       0.94      0.81      0.87        21

       accuracy                           0.92        60
      macro avg       0.93      0.92      0.92        60
   weighted avg       0.92      0.92      0.92        60
```

In [23]:

```
print("Test - Accuracy :",accuracy_score(y_test,clf.predict(x_test)))
print("Test-Confusion matrix:\n",confusion_matrix(y_test,clf.predict(x_test)))
print(classification_report(y_test,predictions))
```

```
Test - Accuracy : 0.9333333333333333
Test-Confusion matrix:
 [[16  0  0]
 [ 0 22  1]
 [ 0  3 18]]
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        16
Iris-versicolor       0.85      0.96      0.90        23
 Iris-virginica       0.94      0.81      0.87        21

       accuracy                           0.92        60
      macro avg       0.93      0.92      0.92        60
   weighted avg       0.92      0.92      0.92        60
```