jupyter Iris_classification Last Checkpoint: an hour ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code

In [74]:
```
#Meeting and understanding the data
```

In [2]:
```
from sklearn.datasets import load_iris
dataset=load_iris()
```

In [3]:
```
print("keys of iris dataset:\n{}".format(dataset.keys()))
```

```
keys of iris dataset:
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename'])
```

In [7]:
```
print(dataset['DESCR'] +"\n")
```

```
.. _iris_dataset:

Iris plants dataset
--------------------

**Data Set Characteristics:**

    :Number of Instances: 150 (50 in each of three classes)
    :Number of Attributes: 4 numeric, predictive attributes and the class
    :Attribute Information:
        - sepal length in cm
        - sepal width in cm
        - petal length in cm
        - petal width in cm
        - class:
                - Iris-Setosa
                - Iris-Versicolour
                - Iris-Virginica

    :Summary Statistics:

    ============== ==== ==== ======= ===== ====================
```

jupyter Iris_classification Last Checkpoint: an hour ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code

In [6]:
```
print("Target names:{}".format(dataset['target_names']))
```
```
Target names:['setosa' 'versicolor' 'virginica']
```

In [8]:
```
print("Feature Names:{}".format(dataset['feature_names']))
```
```
Feature Names:['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
```

In [13]:
```
print("Shape of data:{}".format(dataset['data'].shape))
```
```
Shape of data:(150, 4)
```

In [14]:
```
print("First five data of dataset:{}".format(dataset['data'][:5]))
```
```
First five data of dataset:[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]]
```

In [15]:
```
print("shape of target:{}".format(dataset['target'].shape))
```
```
shape of target:(150,)
```

In [18]:
```
print("shape of target:{}\n".format(dataset['target']))
```
```
shape of target:[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
```

jupyter Iris_classification Last Checkpoint: an hour ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Python

Run Code

In [ ]:
```
#Splitting the data into training data and testing data
```

In [27]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(dataset['data'],dataset['target'],random_state=0)
```

In [28]:
```
print("xtrain shape:{}".format(x_train.shape),"ytrain shape:{}".format(y_train.shape))
```

```
xtrain shape:(112, 4) ytrain shape:(112,)
```

In [30]:
```
print("xtest shape:{}".format(x_test.shape),"ytest shape:{}".format(y_test.shape))
```

```
xtest shape:(38, 4) ytest shape:(38,)
```

In [33]:
```
import pandas as pd
iris_dataframe=pd.DataFrame(x_train,columns=dataset.feature_names)
iris_dataframe
```

Out[33]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.9	3.0	4.2	1.5
1	5.8	2.6	4.0	1.2
2	6.8	3.0	5.5	2.1
3	4.7	3.2	1.3	0.2
4	6.9	3.1	5.1	2.3
...	...	...	...	...
107	4.9	3.1	1.5	0.1
108	6.3	2.9	5.6	1.8
109	5.8	2.7	4.1	1.0
110	7.7	3.8	6.7	2.2
111	4.6	3.2	1.4	0.2

jupyter Iris_classification Last Checkpoint: an hour ago (unsaved changes)

Logout

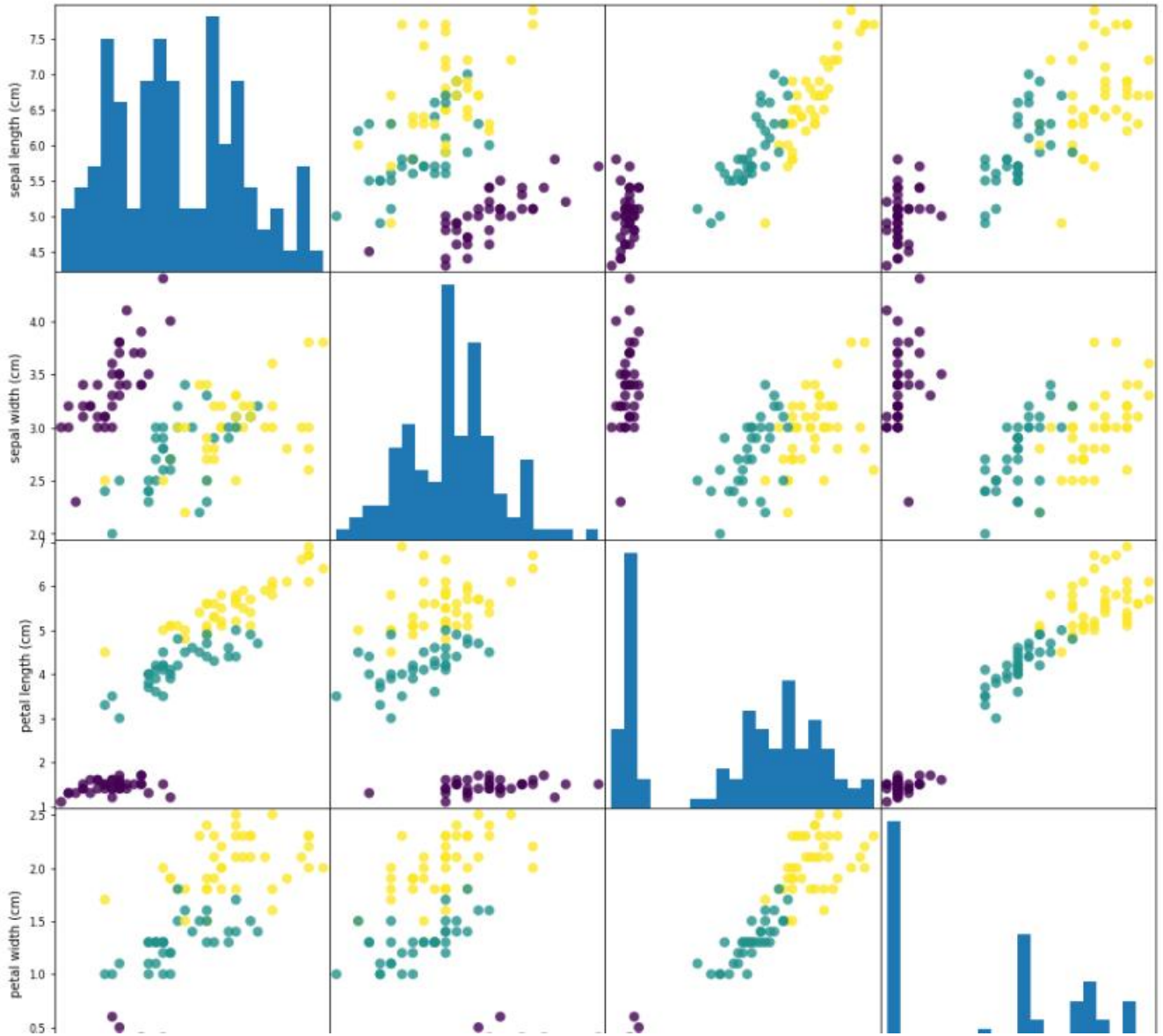
File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code

In [ ]:
```
#Visualizing the data
```

In [37]:
```
rr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o',
 hist_kwds={'bins': 20}, s=60, alpha=.8)
```



jupyter Iris_classification Last Checkpoint: 2 hours ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code

In [ ]:
```
#Implementing the KNN classifier model
```

In [39]:
```
from sklearn.neighbors import KNeighborsClassifier
knn= KNeighborsClassifier(n_neighbors=1)
```

In [40]:
```
knn.fit(x_train,y_train)
```

Out[40]:
```
KNeighborsClassifier(n_neighbors=1)
```

In [54]:
```
import numpy as np
x_new=np.array([[3,5.2,1,2.2]])
x_new
```

Out[54]:
```
array([[3. , 5.2, 1. , 2.2]])
```

In [55]:
```
prediction=knn.predict(x_new)
print("Prediction:{}".format(prediction))
print("Prediction name:{}".format(dataset['target_names'][prediction]))
```

```
Prediction:[0]
Prediction name:['setosa']
```

In [ ]:
```
#Making predictions
```

In [56]:
```
y_pred=knn.predict(x_test)
```

In [59]:
```
y_pred
y_pred.shape
```

Out[59]:
```
(38,)
```

jupyter Iris_classification Last Checkpoint: 2 hours ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code

```
print("Prediction name:{}".format(dataset['target_names'][prediction]))
```

```
Prediction:[0]
Prediction name:['setosa']
```

In [ ]:
```
#Making predictions
```

In [56]:
```
y_pred=knn.predict(x_test)
```

In [59]:
```
y_pred
y_pred.shape
```

Out[59]:
```
(38,)
```

In [ ]:
```
#Testing accuracy
```

In [73]:
```
count=0

for i in range(y_pred.shape[0]):
    if y_pred[i]==y_test[i]:
        count+=1



print("Accuracy of prediction:{}".format((count/y_pred.shape[0])*100))
```

```
Accuracy of prediction:97.36842105263158
```