IN [2]:	<pre>print("Keys of iris_dataset: ".format(iris_dataset.keys())) Keys of iris dataset:</pre>					
In [3]:	dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename'])					
111 [0]	<pre>print(iris_dataset['DESCR'][:193] + "\n")iris_dataset:</pre>					
	Iris plants dataset					
	Data Set Characteristics:					
	:Number of Instances: 150 (50 in each of three classes) :Number of Attributes: 4 numeric, pre					
In [4]:	<pre>print("Target names: {}".format(iris_dataset['target_names']))</pre>					
	Target names: ['setosa' 'versicolor' 'virginica']					
In [5]:	print(reacure names: .format(fris_dataset[reacure_names]))					
T. [6]	Feature names: ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']					
In [6]:	<pre>print("Type of data: {}".format(type(iris_dataset['data']))) Type of data: <class 'numpy.ndarray'=""></class></pre>					
In [7]:	<pre>print("Shape of data: {}".format(iris_dataset['data'].shape))</pre>					
	Shape of data: (150, 4)					
In [8]:	<pre>print("First five columns of data:".format(iris_dataset['data'][:5]))</pre>					
	First five columns of data: [[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 [9]:	<pre>print("Type of target: {}".format(type(iris_dataset['target'])))</pre>					
	Type of target: <class 'numpy.ndarray'=""></class>					
In [39]:	<pre>print("Shape of target: {}".format(iris_dataset['target'].shape))</pre> Shape of target: (150,)					
In [11]:						
	Target: [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0					
	$\begin{smallmatrix}0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1$					
In [12]:	2 2]					
	<pre>X_train, X_test, y_train, y_test = train_test_split(iris_dataset['data'], iris_dataset['target'], random_state=0)</pre>					
In [13]:	<pre>print("X_train shape: {}".format(X_train.shape)) print("y train shape: {}" format(y train shape))</pre>					
	<pre>print("y_train shape: {}".format(y_train.shape)) X_train shape: (112, 4) </pre>					
In [15]:	<pre>y_train shape: (112,) print("X test shape: {}".format(X test.shape))</pre>					
	<pre>print("y_test shape: {}".format(y_test.shape))</pre> <pre>X test shape: (38, 4)</pre>					
In [24]:	y_test shape: (38,)					
	<pre>import numpy as np import pandas as pd import sklearn</pre>					
	<pre>from sklearn.datasets import load_iris from sklearn.model_selection import train_test_split from sklearn.neighbors import KNeighborsClassifier</pre>					
	<pre>from sklearn.metrics import classification_report from sklearn.metrics import confusion_matrix from sklearn.metrics import accuracy_score</pre>					
In [26]:	# create dataframe from data in X_train					
	<pre># label the columns using the strings in iris_dataset.feature_names iris_dataframe = pd.DataFrame(X_train, columns=iris_dataset.feature_names)</pre>					
In [30]:	<pre># create a scatter matrix from the dataframe, color by y_train grr = pd.plotting.scatter matrix(iris dataframe, c=y train, figsize=(15, 15), marker='o')</pre>					
In [30]:	<pre># create a scatter matrix from the dataframe, color by y_train grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o')</pre>					
In [30]:	<pre>grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o')</pre>					
In [30]:	<pre>grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o')</pre>					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o') 7.5 7.0 6.5 7.0 9.55 9.55					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o') 7.5 (W) 6.5 6.6 (B) 6.5 (B) 6.5					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o') 7.5 7.0 6.5 5.0 4.5					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o') 75 70 65 50 45					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o')					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o') (b)					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o') 75 76 77 78 40 40 40 40 40 40 40 40 40 4					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o')					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o') (W)					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o')					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, cey_train, figsize=(15, 15), marker='o') 15 15 16 17 18 18 18 18 18 18 18 18 18					
In [30]:	grr = pd.plotting.scatter_matrix(iria_dataframe, owy_train, figsize=(15, 15), marker='o') 7.5 (8) (8) (8) 40 40 40 40 40 40 40 40 40 4					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, cwy_train, figsize=(15, 15), marker='o') 73 74 75 76 77 78 79 79 79 79 79 79 79 79					
In [30]:	grr = pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='0') 7.5 7.6 (W) 1.5 (W)					
In [30]:	### pd. plotting. seatter_matrix(iris_dataframe, c=y_train, figure=(15, 15), markez=** c*) 73					
In [30]:	from sklearn.neighbors import KNeighborsculassifier					
In [31]:	grr = pd.plotting.scatrer_natrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o') 72 63 64 65 66 67 68 69 69 69 69 69 69 69 69 69					
In [31]: In [32]:	from shiesen.neighbors import (Neighborschiesent) from shiesen.neighbors import (Neighborschiesent) kno. Elik(X_issin, y_issin)					
In [31]:	### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), marker=101) ### Sepal by Polithing .coshber_matrix(iris_labsframe, cmy_brazin, figeizem(15, 15), m					
In [31]: In [32]: Out[32]:	from riteran metighnors import (Anighborsclassifier metallocation) from riteran metighnors import (Anighborsclassifier metallocation) Xmm.II.(X teain, y teain) Xmm.II.(X teain, y teain) Xmey = mm.orray([(5, 2.9, 1, 0.2])) print("X new emp.orray([(5, 2.9, 1, 0.2])))					
In [31]: In [32]: Out[32]:	possi length (cm) sepal length					
In [31]: In [32]: Out[32]: In [33]:	from shinesm.maighbors import Hadighborshilassifier Mark Mar					
In [31]: In [32]: Out[32]: In [33]:	from whiteach relations: Amport (Mexichouse Classifier Mon. It (%_text) boxed (%_text) X. new = non-scarv((%_text), %_text) Y. new = non-scarv((%_text), %_text) Y. new = non-scarv((%_text), %_text) X. new = non-scarv((%_text), %_text) Y. new = non-scarv(
In [31]: In [32]: Out[32]: In [33]:	per = policionary (nontrol pontrol (series)) per = policionary (nontrol pontrol (series)) per = policionary (nontrol pontrol (series)) per = policionary (nontrol pontrol pontrol (series)) per = policionary (nontrol pontrol pon					
In [31]: In [32]: Out[32]: In [33]:	### Special Control of Control					
In [31]: In [32]: Out[32]: In [33]:	### projections of the projection of the project					
In [31]: In [32]: Out[32]: In [33]:	description of the property of					
In [31]: In [32]: Out[32]: In [33]:	The model processes of the company o					
In [31]: In [32]: Out[32]: In [33]: In [36]:	The projection of the projecti					
In [31]: In [32]: Out[32]: In [33]: In [36]:	From exhaust interpretation of the product of the p					
In [31]: In [32]: Out[32]: In [33]: In [34]:	### Special Company (p. 1.8. l. C.4.) ### Special Company (p. 1.8. l. C.4.) #### Special Company (p. 1.8. l. C.4.) ##################################					
In [31]: In [32]: Out[32]: In [33]: In [34]:	From Michael and Discovering State (1997). Annal (Incovering State (1997)) The annal (Incovering State (1997)). Annal (Incovering State (1997)) The annal (Incovering State (1997)). Annal (Incovering State (1997)) The annal (Incovering State (1997)). Annal (Incovering State (1997)). The annal (Incovering					

In [1]:
 from sklearn.datasets import load_iris
 iris_dataset = load_iris()

print("Keys of iris_dataset: \n{}".format(iris_dataset.keys()))