In [1]:
```python
from sklearn.datasets import load_iris
```

In [2]:
```python
iris=load_iris()
```

In [4]:
```python
print(iris.DESCR)
```

```
.. _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:

    ============== ==== ==== ======= ===== ====================
                    Min  Max   Mean    SD   Class Correlation
    ============== ==== ==== ======= ===== ====================
    sepal length:   4.3  7.9   5.84   0.83    0.7826
    sepal width:    2.0  4.4   3.05   0.43   -0.4194
    petal length:   1.0  6.9   3.76   1.76    0.9490  (high!)
    petal width:    0.1  2.5   1.20   0.76    0.9565  (high!)
    ============== ==== ==== ======= ===== ====================

    :Missing Attribute Values: None
    :Class Distribution: 33.3% for each of 3 classes.
    :Creator: R.A. Fisher
    :Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
    :Date: July, 1988

The famous Iris database, first used by Sir R.A. Fisher. The dataset is taken
from Fisher's paper. Note that it's the same as in R, but not as in the UCI
Machine Learning Repository, which has two wrong data points.

This is perhaps the best known database to be found in the
pattern recognition literature.  Fisher's paper is a classic in the field and
is referenced frequently to this day.  (See Duda & Hart, for example.)  The
data set contains 3 classes of 50 instances each, where each class refers to a
type of iris plant.  One class is linearly separable from the other 2; the
latter are NOT linearly separable from each other.

.. topic:: References

   - Fisher, R.A. "The use of multiple measurements in taxonomic problems"
     Annual Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to
     Mathematical Statistics" (John Wiley, NY, 1950).
   - Duda, R.O., & Hart, P.E. (1973) Pattern Classification and Scene Analysis.
     (Q327.D83) John Wiley & Sons.  ISBN 0-471-22361-1.  See page 218.
   - Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New System
     Structure and Classification Rule for Recognition in Partially Exposed
     Environments".  IEEE Transactions on Pattern Analysis and Machine
     Intelligence, Vol. PAMI-2, No. 1, 67-71.
   - Gates, G.W. (1972) "The Reduced Nearest Neighbor Rule".  IEEE Transactions
     on Information Theory, May 1972, 431-433.
   - See also: 1988 MLC Proceedings, 54-64.  Cheeseman et al"s AUTOCLASS II
     conceptual clustering system finds 3 classes in the data.
   - Many, many more ...
```

In [5]:
```python
iris.data.shape
```

Out[5]: (150, 4)

In [6]:
```python
iris.target.shape
```

Out[6]: (150,)

In [ ]:

In [8]:
```python
import pandas as pd
```

In [9]:
```python
pd.set_option('max_columns',5)
```

In [10]:
```python
pd.set_option('display.width', None)
```

In [11]:
```python
type(iris)
```

Out[11]: sklearn.utils.Bunch

In [12]:
```python
iris_df=pd.DataFrame(iris.data, columns=iris.feature_names)
```

In [13]:
```python
iris_df.head()
```

Out[13]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

In [14]:
```python
iris_df.describe()
```

Out[14]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
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 [15]:
```python
iris_df.tail()
```

Out[15]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

In [16]:
```python
iris_df['species']=[iris.target_names[i] for i in iris.target]
```

In [17]:
```python
iris_df.head()
```

Out[17]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [18]:
```python
iris_df.sample(20)
```

Out[18]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
32	5.2	4.1	1.5	0.1	setosa
145	6.7	3.0	5.2	2.3	virginica
120	6.9	3.2	5.7	2.3	virginica
87	6.3	2.3	4.4	1.3	versicolor
88	5.6	3.0	4.1	1.3	versicolor
98	5.1	2.5	3.0	1.1	versicolor
36	5.5	3.5	1.3	0.2	setosa
128	6.4	2.8	5.6	2.1	virginica
85	6.0	3.4	4.5	1.6	versicolor
122	7.7	2.8	6.7	2.0	virginica
65	6.7	3.1	4.4	1.4	versicolor
143	6.8	3.2	5.9	2.3	virginica
92	5.8	2.6	4.0	1.2	versicolor
6	4.6	3.4	1.4	0.3	setosa
97	6.2	2.9	4.3	1.3	versicolor
105	7.6	3.0	6.6	2.1	virginica
56	6.3	3.3	4.7	1.6	versicolor
12	4.8	3.0	1.4	0.1	setosa
3	4.6	3.1	1.5	0.2	setosa
67	5.8	2.7	4.1	1.0	versicolor

In [19]:
```python
type(iris_df)
```

Out[19]: pandas.core.frame.DataFrame

In [21]:
```python
pd.set_option('precision',2)
```

In [22]:
```python
iris_df.describe()
```

Out[22]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
count	150.00	150.00	150.00	150.00
mean	5.84	3.06	3.76	1.20
std	0.83	0.44	1.77	0.76
min	4.30	2.00	1.00	0.10
25%	5.10	2.80	1.60	0.30
50%	5.80	3.00	4.35	1.30
75%	6.40	3.30	5.10	1.80
max	7.90	4.40	6.90	2.50

In [23]:
```python
iris_df['species'].describe()
```

Out[23]:
```
count           150
unique            3
top       virginica
freq             50
Name: species, dtype: object
```

In [24]:
```python
# Visualizing the dataset with seaborn pairplot
```

In [25]:
```python
import seaborn as sns
```

In [26]:
```python
sns.set(font_scale=1.1)
```

In [27]:
```python
sns.set_style('whitegrid')
```

In [30]:
```python
grid=sns.pairplot(data=iris_df,vars=iris_df.columns[0:4],hue='species')
```

In [31]:
```python
#Using a KMeans Estimator
```

In [32]:
```python
from sklearn.cluster import KMeans
```

In [33]:
```python
kmeans=KMeans(n_clusters=3,random_state=11)
```

In [34]:
```python
kmeans.fit(iris.data)
```

Out[34]: KMeans(n_clusters=3, random_state=11)

In [35]:
```python
# psudorandoms -
```

In [36]:
```python
print(kmeans.labels_[0:50])
```

```
[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]
```

In [37]:
```python
print(kmeans.labels_[50:100])
```

```
[0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

In [38]:
```python
print(kmeans.labels_[100:150])
```

```
[2 0 2 2 2 2 0 2 2 2 2 2 2 0 0 2 2 2 2 0 2 0 2 0 2 2 0 0 2 2 2 2 2 0 2 2 2
 2 0 2 2 2 0 2 2 2 0 2 2 0]
```

In [39]:
```python
iris_df.head(60)
```

Out[39]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5.0	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa
10	5.4	3.7	1.5	0.2	setosa
11	4.8	3.4	1.6	0.2	setosa
12	4.8	3.0	1.4	0.1	setosa
13	4.3	3.0	1.1	0.1	setosa
14	5.8	4.0	1.2	0.2	setosa
15	5.7	4.4	1.5	0.4	setosa
16	5.4	3.9	1.3	0.4	setosa
17	5.1	3.5	1.4	0.3	setosa
18	5.7	3.8	1.7	0.3	setosa
19	5.1	3.8	1.5	0.3	setosa
20	5.4	3.4	1.7	0.2	setosa
21	5.1	3.7	1.5	0.4	setosa
22	4.6	3.6	1.0	0.2	setosa
23	5.1	3.3	1.7	0.5	setosa
24	4.8	3.4	1.9	0.2	setosa
25	5.0	3.0	1.6	0.2	setosa
26	5.0	3.4	1.6	0.4	setosa
27	5.2	3.5	1.5	0.2	setosa
28	5.2	3.4	1.4	0.2	setosa
29	4.7	3.2	1.6	0.2	setosa
30	4.8	3.1	1.6	0.2	setosa
31	5.4	3.4	1.5	0.4	setosa
32	5.2	4.1	1.5	0.1	setosa
33	5.5	4.2	1.4	0.2	setosa
34	4.9	3.1	1.5	0.2	setosa
35	5.0	3.2	1.2	0.2	setosa
36	5.5	3.5	1.3	0.2	setosa
37	4.9	3.6	1.4	0.1	setosa
38	4.4	3.0	1.3	0.2	setosa
39	5.1	3.4	1.5	0.2	setosa
40	5.0	3.5	1.3	0.3	setosa
41	4.5	2.3	1.3	0.3	setosa
42	4.4	3.2	1.3	0.2	setosa
43	5.0	3.5	1.6	0.6	setosa
44	5.1	3.8	1.9	0.4	setosa
45	4.8	3.0	1.4	0.3	setosa
46	5.1	3.8	1.6	0.2	setosa
47	4.6	3.2	1.4	0.2	setosa
48	5.3	3.7	1.5	0.2	setosa
49	5.0	3.3	1.4	0.2	setosa
50	7.0	3.2	4.7	1.4	versicolor
51	6.4	3.2	4.5	1.5	versicolor
52	6.9	3.1	4.9	1.5	versicolor
53	5.5	2.3	4.0	1.3	versicolor
54	6.5	2.8	4.6	1.5	versicolor
55	5.7	2.8	4.5	1.3	versicolor
56	6.3	3.3	4.7	1.6	versicolor
57	4.9	2.4	3.3	1.0	versicolor
58	6.6	2.9	4.6	1.3	versicolor
59	5.2	2.7	3.9	1.4	versicolor

In [ ]:
```python
# Try K-means clustering on the iris dataset with the
# following clusters 2,4,8,10
# comment about the results
```