

Arianna Burford assignment #2 Iris Species

September 4, 2023

```
[1]: # add comment
import pandas as pd
from sklearn.datasets import load_iris
iris_dataset = load_iris()
# nothing will be printed to output but you must run this cell
```

```
[2]: # add comment
import pandas as pd
from sklearn.datasets import load_iris
iris_dataset = load_iris()
```

```
[3]: # add comment
print("Keys of iris_dataset: \n{}".format(iris_dataset.keys()))
```

```
Keys of iris_dataset:
dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names',
'filename', 'data_module'])
```

```
[4]: # comment
print(iris_dataset['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 ...

```
[5]: # comment
print("Target name: {}".format(iris_dataset['target_names']))
```

Target name: ['setosa' 'versicolor' 'virginica']

```
[6]: # comment explain features
print("Feature names: \n{}".format(iris_dataset['feature_names']))
```

Feature names:
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']

```
[7]: # comment explain "type"
print("Type of data: {}".format(type(iris_dataset['data'])))
```

Type of data: <class 'numpy.ndarray'>

```
[8]: # comment explain "shape"
print("Shape of data: {}".format(iris_dataset['data'].shape))
```

Shape of data: (150, 4)

```
[9]: # comment
print("First 5 columns of data: \n{}".format(iris_dataset['data'][:5]))
```

First 5 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]]

```
[10]: # create dataframe to store information about the features
iris_df = pd.DataFrame(iris_dataset.data)
# automatically prints out the first 5 rows of data
iris_df.head()
```

```
[10]:
```

	0	1	2	3
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

```
[11]: # add comment that explain why this was done
iris_df.columns = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width']
iris_df.head()
```

```
[11]:
```

	sepal_length	sepal_width	petal_length	petal_width
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

```
[14]: # the target frame is only one column but when combined with both dataframes,
# features and which flower it belongs to.
target_df = pd.DataFrame(iris_dataset.target)
target_df = target_df.rename(columns = {0: 'target'})
target_df.head()
```

```
[14]:      target
0         0
1         0
2         0
3         0
4         0
```

```
[16]: # combined the two dataframes to see the features and target (labels) together
df = pd.concat([iris_df, target_df], axis=1)
df
```

```
[16]:      sepal_length  sepal_width  petal_length  petal_width  target
0           5.1           3.5           1.4           0.2         0
1           4.9           3.0           1.4           0.2         0
2           4.7           3.2           1.3           0.2         0
3           4.6           3.1           1.5           0.2         0
4           5.0           3.6           1.4           0.2         0
..          ...           ...           ...           ...         ...
145          6.7           3.0           5.2           2.3         2
146          6.3           2.5           5.0           1.9         2
147          6.5           3.0           5.2           2.0         2
148          6.2           3.4           5.4           2.3         2
149          5.9           3.0           5.1           1.8         2
```

[150 rows x 5 columns]

```
[17]: # numbers given by the iris_datset array: 0 setosa, 1 versicolor and 2 virginica
print("Target: \n{}".format(iris_dataset['target']))
```

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 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]
```

```
[19]: # add comment
import matplotlib.pyplot as plt
# indices of features to plot
X = 0
y = 1
# formatter labels the color bar with the target names
formatter = plt.FuncFormatter(lambda i, *args: iris_dataset.
target_names[int(i)])
# nothing will be printed out

[22]: # Two feature scatter plot of the iris dataset using matplotlib
plt.figure(figsize=(5,4))
plt.scatter(iris_dataset.data[:, X], iris_dataset.data[:,y], c=iris_dataset.
target)
plt.colorbar(ticks=[0,1,2], format=formatter)
plt.xlabel(iris_dataset.feature_names[X])
plt.ylabel(iris_dataset.feature_names[y])
class_labels = iris_dataset.target_names
legend_aliases = [plt.Line2D([0,0],[0,0], color=['purple','green','yellow'][i],
marker='o', linestyle='', label=label)
for i, label in enumerate(class_labels)]
plt.legend(handles=legend_aliases)
plt.tight_layout()
plt.show()
```

output_15_0.png

```
[ ]:
```