splitting_the_data

August 14, 2020

1 Splitting the Data

This is a mini Jupyter notebook in which you will load your data, examine it, convert it to NumPy arrays, generate numerical labels from string labels and call the data splitting function (from scikit-learn) twice to split your data into training, validation and test datasets.

1.1 Before you start

- In order for the notebooks to function as intended, modify only between lines marked "### begin your code here (__ lines)." and "### end your code here.".
- The line count is a suggestion of how many lines of code you need to accomplish what is asked.
- You should execute the cells (the boxes that a notebook is composed of) in order.
- You can execute a cell by pressing Shift and Enter (or Return) simultaneously.
- You should have completed the previous Jupyter notebooks before attempting this one as the concepts covered there are not repeated, for the sake of brevity.

1.2 Loading the appropriate packages

We will import the required libraries.

[1]:
```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
import pandas as pd
import plotly.express as px
```

Let's turn off scientific notation.

[2]:
```
np.set_printoptions(suppress=True)
```

1.3 Loading and examining the data

We will load our data from a CSV file and put it in a pandas an object of the `DataFrame` class.

This is the Iris flower dataset, a very famous dataset which was published in 1936 by studying three different species of the flower Iris: *Iris setosa*, *Iris versicolor* and *Iris virginica*. Originally, the

dataset has 150 examples which corresponds to 150 different Iris flowers measured (50 flowers from each species). The dataset has 4 features, the sepal length, sepal width, petal length and petal width of each flower in centimeters.

- This was taken and modified from the Machine Learning dataset repository of School of Information and Computer Science of University of California Irvine (UCI):

  Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[3]:
```
df = pd.read_csv('data_splitting_the_data.csv')
```

Let's take a look at the data.

[4]:
```
df
```

[4]:

	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
5	5.4	3.9	1.7	0.4	Iris setosa
6	4.6	3.4	1.4	0.3	Iris setosa
7	5.0	3.4	1.5	0.2	Iris setosa
8	4.4	2.9	1.4	0.2	Iris setosa
9	4.9	3.1	1.5	0.1	Iris setosa
10	5.4	3.7	1.5	0.2	Iris setosa
11	4.8	3.4	1.6	0.2	Iris setosa
12	4.8	3.0	1.4	0.1	Iris setosa
13	4.3	3.0	1.1	0.1	Iris setosa
14	5.8	4.0	1.2	0.2	Iris setosa
15	5.7	4.4	1.5	0.4	Iris setosa
16	5.4	3.9	1.3	0.4	Iris setosa
17	5.1	3.5	1.4	0.3	Iris setosa
18	5.7	3.8	1.7	0.3	Iris setosa
19	5.1	3.8	1.5	0.3	Iris setosa
20	5.4	3.4	1.7	0.2	Iris setosa
21	5.1	3.7	1.5	0.4	Iris setosa
22	4.6	3.6	1.0	0.2	Iris setosa
23	5.1	3.3	1.7	0.5	Iris setosa
24	4.8	3.4	1.9	0.2	Iris setosa
25	5.0	3.0	1.6	0.2	Iris setosa
26	5.0	3.4	1.6	0.4	Iris setosa
27	5.2	3.5	1.5	0.2	Iris setosa
28	5.2	3.4	1.4	0.2	Iris setosa
29	4.7	3.2	1.6	0.2	Iris setosa
..	...	...	...	...	...
120	6.9	3.2	5.7	2.3	Iris virginica

```
121          5.6           2.8            4.9           2.0  Iris virginica
122          7.7           2.8            6.7           2.0  Iris virginica
123          6.3           2.7            4.9           1.8  Iris virginica
124          6.7           3.3            5.7           2.1  Iris virginica
125          7.2           3.2            6.0           1.8  Iris virginica
126          6.2           2.8            4.8           1.8  Iris virginica
127          6.1           3.0            4.9           1.8  Iris virginica
128          6.4           2.8            5.6           2.1  Iris virginica
129          7.2           3.0            5.8           1.6  Iris virginica
130          7.4           2.8            6.1           1.9  Iris virginica
131          7.9           3.8            6.4           2.0  Iris virginica
132          6.4           2.8            5.6           2.2  Iris virginica
133          6.3           2.8            5.1           1.5  Iris virginica
134          6.1           2.6            5.6           1.4  Iris virginica
135          7.7           3.0            6.1           2.3  Iris virginica
136          6.3           3.4            5.6           2.4  Iris virginica
137          6.4           3.1            5.5           1.8  Iris virginica
138          6.0           3.0            4.8           1.8  Iris virginica
139          6.9           3.1            5.4           2.1  Iris virginica
140          6.7           3.1            5.6           2.4  Iris virginica
141          6.9           3.1            5.1           2.3  Iris virginica
142          5.8           2.7            5.1           1.9  Iris virginica
143          6.8           3.2            5.9           2.3  Iris virginica
144          6.7           3.3            5.7           2.5  Iris virginica
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]
```

We have more than 3 features, so let's use a scatter matrix to visualize our data:

[5]:
```
data_dimensions = df.columns[:-1].to_list()

fig = px.scatter_matrix(df, dimensions=data_dimensions, color='Species')
fig.show()
```

Let's convert our data to NumPy arrays. We use LabelEncoder from scikit-learn to convert our string labels into numbers:

[6]:
```
X = df.drop('Species', axis=1).to_numpy()
y_text = df['Species'].to_numpy()
y = LabelEncoder().fit_transform(y_text)
```

Now we can check the resulting NumPy arrays and their shapes:

[7]:
```
X
```

[7]:
```
array([[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],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
       [4.3, 3. , 1.1, 0.1],
       [5.8, 4. , 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
       [5.7, 3.8, 1.7, 0.3],
       [5.1, 3.8, 1.5, 0.3],
       [5.4, 3.4, 1.7, 0.2],
       [5.1, 3.7, 1.5, 0.4],
       [4.6, 3.6, 1. , 0.2],
       [5.1, 3.3, 1.7, 0.5],
       [4.8, 3.4, 1.9, 0.2],
       [5. , 3. , 1.6, 0.2],
       [5. , 3.4, 1.6, 0.4],
       [5.2, 3.5, 1.5, 0.2],
       [5.2, 3.4, 1.4, 0.2],
       [4.7, 3.2, 1.6, 0.2],
       [4.8, 3.1, 1.6, 0.2],
       [5.4, 3.4, 1.5, 0.4],
       [5.2, 4.1, 1.5, 0.1],
       [5.5, 4.2, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.2],
       [5. , 3.2, 1.2, 0.2],
       [5.5, 3.5, 1.3, 0.2],
       [4.9, 3.6, 1.4, 0.1],
       [4.4, 3. , 1.3, 0.2],
       [5.1, 3.4, 1.5, 0.2],
       [5. , 3.5, 1.3, 0.3],
       [4.5, 2.3, 1.3, 0.3],
       [4.4, 3.2, 1.3, 0.2],
       [5. , 3.5, 1.6, 0.6],
       [5.1, 3.8, 1.9, 0.4],
       [4.8, 3. , 1.4, 0.3],
       [5.1, 3.8, 1.6, 0.2],
```

```
[4.6, 3.2, 1.4, 0.2],
[5.3, 3.7, 1.5, 0.2],
[5. , 3.3, 1.4, 0.2],
[7. , 3.2, 4.7, 1.4],
[6.4, 3.2, 4.5, 1.5],
[6.9, 3.1, 4.9, 1.5],
[5.5, 2.3, 4. , 1.3],
[6.5, 2.8, 4.6, 1.5],
[5.7, 2.8, 4.5, 1.3],
[6.3, 3.3, 4.7, 1.6],
[4.9, 2.4, 3.3, 1. ],
[6.6, 2.9, 4.6, 1.3],
[5.2, 2.7, 3.9, 1.4],
[5. , 2. , 3.5, 1. ],
[5.9, 3. , 4.2, 1.5],
[6. , 2.2, 4. , 1. ],
[6.1, 2.9, 4.7, 1.4],
[5.6, 2.9, 3.6, 1.3],
[6.7, 3.1, 4.4, 1.4],
[5.6, 3. , 4.5, 1.5],
[5.8, 2.7, 4.1, 1. ],
[6.2, 2.2, 4.5, 1.5],
[5.6, 2.5, 3.9, 1.1],
[5.9, 3.2, 4.8, 1.8],
[6.1, 2.8, 4. , 1.3],
[6.3, 2.5, 4.9, 1.5],
[6.1, 2.8, 4.7, 1.2],
[6.4, 2.9, 4.3, 1.3],
[6.6, 3. , 4.4, 1.4],
[6.8, 2.8, 4.8, 1.4],
[6.7, 3. , 5. , 1.7],
[6. , 2.9, 4.5, 1.5],
[5.7, 2.6, 3.5, 1. ],
[5.5, 2.4, 3.8, 1.1],
[5.5, 2.4, 3.7, 1. ],
[5.8, 2.7, 3.9, 1.2],
[6. , 2.7, 5.1, 1.6],
[5.4, 3. , 4.5, 1.5],
[6. , 3.4, 4.5, 1.6],
[6.7, 3.1, 4.7, 1.5],
[6.3, 2.3, 4.4, 1.3],
[5.6, 3. , 4.1, 1.3],
[5.5, 2.5, 4. , 1.3],
[5.5, 2.6, 4.4, 1.2],
[6.1, 3. , 4.6, 1.4],
[5.8, 2.6, 4. , 1.2],
[5. , 2.3, 3.3, 1. ],
```

```
[5.6, 2.7, 4.2, 1.3],
[5.7, 3. , 4.2, 1.2],
[5.7, 2.9, 4.2, 1.3],
[6.2, 2.9, 4.3, 1.3],
[5.1, 2.5, 3. , 1.1],
[5.7, 2.8, 4.1, 1.3],
[6.3, 3.3, 6. , 2.5],
[5.8, 2.7, 5.1, 1.9],
[7.1, 3. , 5.9, 2.1],
[6.3, 2.9, 5.6, 1.8],
[6.5, 3. , 5.8, 2.2],
[7.6, 3. , 6.6, 2.1],
[4.9, 2.5, 4.5, 1.7],
[7.3, 2.9, 6.3, 1.8],
[6.7, 2.5, 5.8, 1.8],
[7.2, 3.6, 6.1, 2.5],
[6.5, 3.2, 5.1, 2. ],
[6.4, 2.7, 5.3, 1.9],
[6.8, 3. , 5.5, 2.1],
[5.7, 2.5, 5. , 2. ],
[5.8, 2.8, 5.1, 2.4],
[6.4, 3.2, 5.3, 2.3],
[6.5, 3. , 5.5, 1.8],
[7.7, 3.8, 6.7, 2.2],
[7.7, 2.6, 6.9, 2.3],
[6. , 2.2, 5. , 1.5],
[6.9, 3.2, 5.7, 2.3],
[5.6, 2.8, 4.9, 2. ],
[7.7, 2.8, 6.7, 2. ],
[6.3, 2.7, 4.9, 1.8],
[6.7, 3.3, 5.7, 2.1],
[7.2, 3.2, 6. , 1.8],
[6.2, 2.8, 4.8, 1.8],
[6.1, 3. , 4.9, 1.8],
[6.4, 2.8, 5.6, 2.1],
[7.2, 3. , 5.8, 1.6],
[7.4, 2.8, 6.1, 1.9],
[7.9, 3.8, 6.4, 2. ],
[6.4, 2.8, 5.6, 2.2],
[6.3, 2.8, 5.1, 1.5],
[6.1, 2.6, 5.6, 1.4],
[7.7, 3. , 6.1, 2.3],
[6.3, 3.4, 5.6, 2.4],
[6.4, 3.1, 5.5, 1.8],
[6. , 3. , 4.8, 1.8],
[6.9, 3.1, 5.4, 2.1],
[6.7, 3.1, 5.6, 2.4],
```

```
       [6.9, 3.1, 5.1, 2.3],
       [5.8, 2.7, 5.1, 1.9],
       [6.8, 3.2, 5.9, 2.3],
       [6.7, 3.3, 5.7, 2.5],
       [6.7, 3. , 5.2, 2.3],
       [6.3, 2.5, 5. , 1.9],
       [6.5, 3. , 5.2, 2. ],
       [6.2, 3.4, 5.4, 2.3],
       [5.9, 3. , 5.1, 1.8]])
```

[8]:
```
X.shape
```

[8]:
```
(150, 4)
```

[9]:
```
y_text
```

[9]:
```
array(['Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris setosa', 'Iris setosa',
       'Iris setosa', 'Iris setosa', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris versicolor', 'Iris versicolor', 'Iris versicolor',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
```

```
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica', 'Iris virginica',
       'Iris virginica', 'Iris virginica'], dtype=object)
```

[10]:
```
y_text.shape
```

[10]:
```
(150,)
```

[11]:
```
y
```

[11]:
```
array([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, 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])
```

[12]:
```
y.shape
```

[12]:
```
(150,)
```

1.4 Splitting data

Now use `train_test_split` twice: Once to create `X_train` and `y_train`, and `X_vt` and `y_vt` to create two sets, one for training and the other for validation and test. We want 70% of our data to be training data. No other argumants need to be sepcified. You can find the documentation for `train_test_split` here:

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

Go ahead and do that function call here:

[13]:
```
### begin your code here (1 line).
X_train, X_vt, y_train, y_vt = train_test_split(X, y, test_size=0.3)
### end your code here.
```

Now, split your combined validation and test data (`X_vt` and `y_vt`) into two separate validation (`X_validation` and `y_validation`) and test (`X_test` and `y_test`) datasets. We want 50% of this data to be for validation and another 50% for test (which reserves 15% of total datapoints for each):

[14]:
```
### begin your code here (1 line).
```

```
X_validation, X_test, y_validation, y_test = train_test_split(X_vt, y_vt,␣
↪test_size=0.5)
### end your code here.
```

If everything went alright, we are done! You can use these datasets in a supervised learning algorithm now. But let's check the resulting data now:

[15]:
```
X_train
```

[15]:
```
array([[5.5, 4.2, 1.4, 0.2],
       [6.3, 3.3, 4.7, 1.6],
       [5.8, 2.8, 5.1, 2.4],
       [4.7, 3.2, 1.3, 0.2],
       [5.2, 2.7, 3.9, 1.4],
       [4.6, 3.1, 1.5, 0.2],
       [5.4, 3.7, 1.5, 0.2],
       [4.9, 2.5, 4.5, 1.7],
       [6.3, 2.9, 5.6, 1.8],
       [5. , 2. , 3.5, 1. ],
       [5.6, 2.7, 4.2, 1.3],
       [6.1, 3. , 4.9, 1.8],
       [5.4, 3.4, 1.5, 0.4],
       [5. , 3.3, 1.4, 0.2],
       [4.8, 3.1, 1.6, 0.2],
       [4.3, 3. , 1.1, 0.1],
       [5.9, 3.2, 4.8, 1.8],
       [4.4, 3. , 1.3, 0.2],
       [6.9, 3.1, 5.1, 2.3],
       [5. , 3.4, 1.6, 0.4],
       [6. , 2.2, 4. , 1. ],
       [5.1, 3.5, 1.4, 0.3],
       [6.8, 2.8, 4.8, 1.4],
       [6.5, 3. , 5.5, 1.8],
       [5.5, 2.3, 4. , 1.3],
       [7.3, 2.9, 6.3, 1.8],
       [5.8, 2.7, 3.9, 1.2],
       [6.3, 2.8, 5.1, 1.5],
       [6.3, 3.4, 5.6, 2.4],
       [6.1, 2.6, 5.6, 1.4],
       [5.1, 3.7, 1.5, 0.4],
       [6.3, 2.5, 5. , 1.9],
       [4.7, 3.2, 1.6, 0.2],
       [5.7, 2.9, 4.2, 1.3],
       [5.7, 2.8, 4.1, 1.3],
       [6.9, 3.1, 5.4, 2.1],
       [5.2, 3.5, 1.5, 0.2],
       [7.2, 3. , 5.8, 1.6],
       [6. , 2.7, 5.1, 1.6],
       [4.9, 3. , 1.4, 0.2],
```

```
[5. , 3.6, 1.4, 0.2],
[6.4, 2.8, 5.6, 2.1],
[6.7, 2.5, 5.8, 1.8],
[5.1, 3.4, 1.5, 0.2],
[5.8, 4. , 1.2, 0.2],
[7.4, 2.8, 6.1, 1.9],
[7. , 3.2, 4.7, 1.4],
[6.1, 2.9, 4.7, 1.4],
[4.9, 3.1, 1.5, 0.2],
[6.7, 3.1, 5.6, 2.4],
[5.1, 3.8, 1.9, 0.4],
[5.1, 3.8, 1.5, 0.3],
[6.9, 3.1, 4.9, 1.5],
[5.1, 2.5, 3. , 1.1],
[5. , 3.5, 1.3, 0.3],
[5. , 2.3, 3.3, 1. ],
[7.1, 3. , 5.9, 2.1],
[6.8, 3. , 5.5, 2.1],
[6.7, 3. , 5.2, 2.3],
[6.8, 3.2, 5.9, 2.3],
[5.5, 2.4, 3.7, 1. ],
[5.6, 3. , 4.5, 1.5],
[6. , 3.4, 4.5, 1.6],
[5. , 3.2, 1.2, 0.2],
[6.4, 2.9, 4.3, 1.3],
[6.4, 2.7, 5.3, 1.9],
[5.4, 3. , 4.5, 1.5],
[6.3, 2.5, 4.9, 1.5],
[5. , 3. , 1.6, 0.2],
[6.6, 3. , 4.4, 1.4],
[5.7, 2.6, 3.5, 1. ],
[4.8, 3. , 1.4, 0.3],
[5.5, 2.4, 3.8, 1.1],
[6.7, 3. , 5. , 1.7],
[6.4, 3.1, 5.5, 1.8],
[5.8, 2.7, 4.1, 1. ],
[5.8, 2.6, 4. , 1.2],
[6.2, 3.4, 5.4, 2.3],
[7.7, 2.6, 6.9, 2.3],
[5.3, 3.7, 1.5, 0.2],
[5.5, 3.5, 1.3, 0.2],
[5.6, 2.5, 3.9, 1.1],
[6. , 2.2, 5. , 1.5],
[6.3, 3.3, 6. , 2.5],
[5.1, 3.5, 1.4, 0.2],
[6.4, 3.2, 5.3, 2.3],
[7.6, 3. , 6.6, 2.1],
```

```
       [6. , 3. , 4.8, 1.8],
       [6.5, 3.2, 5.1, 2. ],
       [6.2, 2.2, 4.5, 1.5],
       [5.1, 3.3, 1.7, 0.5],
       [7.7, 3.8, 6.7, 2.2],
       [5.9, 3. , 4.2, 1.5],
       [4.4, 2.9, 1.4, 0.2],
       [6.7, 3.1, 4.4, 1.4],
       [5.7, 3. , 4.2, 1.2],
       [6.2, 2.8, 4.8, 1.8],
       [5.4, 3.9, 1.7, 0.4],
       [4.5, 2.3, 1.3, 0.3],
       [5.7, 3.8, 1.7, 0.3],
       [4.6, 3.2, 1.4, 0.2],
       [6.5, 2.8, 4.6, 1.5],
       [4.9, 3.6, 1.4, 0.1],
       [4.8, 3.4, 1.6, 0.2],
       [6.3, 2.3, 4.4, 1.3]])
```

[16]:
```
X_train.shape
```

[16]:
```
(105, 4)
```

[17]:
```
y_train
```

[17]:
```
array([0, 1, 2, 0, 1, 0, 0, 2, 2, 1, 1, 2, 0, 0, 0, 0, 1, 0, 2, 0, 1, 0,
       1, 2, 1, 2, 1, 2, 2, 2, 0, 2, 0, 1, 1, 2, 0, 2, 1, 0, 0, 2, 2, 0,
       0, 2, 1, 1, 0, 2, 0, 0, 1, 1, 0, 1, 2, 2, 2, 2, 1, 1, 1, 0, 1, 2,
       1, 1, 0, 1, 1, 0, 1, 1, 2, 1, 1, 2, 2, 0, 0, 1, 2, 2, 0, 2, 2, 2,
       2, 1, 0, 2, 1, 0, 1, 1, 2, 0, 0, 0, 0, 1, 0, 0, 1])
```

[18]:
```
y_train.shape
```

[18]:
```
(105,)
```

[19]:
```
X_validation
```

[19]:
```
array([[6.4, 2.8, 5.6, 2.2],
       [5.2, 4.1, 1.5, 0.1],
       [4.9, 2.4, 3.3, 1. ],
       [5.5, 2.5, 4. , 1.3],
       [4.8, 3.4, 1.9, 0.2],
       [5.6, 3. , 4.1, 1.3],
       [6.6, 2.9, 4.6, 1.3],
       [4.6, 3.4, 1.4, 0.3],
       [4.4, 3.2, 1.3, 0.2],
       [6.1, 2.8, 4.7, 1.2],
       [5.8, 2.7, 5.1, 1.9],
       [6.3, 2.7, 4.9, 1.8],
       [6.5, 3. , 5.8, 2.2],
       [6.4, 3.2, 4.5, 1.5],
```

```
       [5. , 3.4, 1.5, 0.2],
       [5.7, 2.8, 4.5, 1.3],
       [6. , 2.9, 4.5, 1.5],
       [4.6, 3.6, 1. , 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [6.7, 3.3, 5.7, 2.5],
       [7.9, 3.8, 6.4, 2. ],
       [5.4, 3.9, 1.3, 0.4]])
```

[20]:
```
X_validation.shape
```

[20]:
```
(22, 4)
```

[21]:
```
y_validation
```

[21]:
```
array([2, 0, 1, 1, 0, 1, 1, 0, 0, 1, 2, 2, 2, 1, 0, 1, 1, 0, 0, 2, 2, 0])
```

[22]:
```
y_validation.shape
```

[22]:
```
(22,)
```

[23]:
```
X_test
```

[23]:
```
array([[6.1, 3. , 4.6, 1.4],
       [5.9, 3. , 5.1, 1.8],
       [6.2, 2.9, 4.3, 1.3],
       [4.8, 3. , 1.4, 0.1],
       [5.7, 4.4, 1.5, 0.4],
       [7.7, 2.8, 6.7, 2. ],
       [5. , 3.5, 1.6, 0.6],
       [5.1, 3.8, 1.6, 0.2],
       [5.2, 3.4, 1.4, 0.2],
       [6.7, 3.3, 5.7, 2.1],
       [5.6, 2.8, 4.9, 2. ],
       [6.1, 2.8, 4. , 1.3],
       [5.4, 3.4, 1.7, 0.2],
       [5.5, 2.6, 4.4, 1.2],
       [5.6, 2.9, 3.6, 1.3],
       [7.2, 3.6, 6.1, 2.5],
       [7.7, 3. , 6.1, 2.3],
       [6.9, 3.2, 5.7, 2.3],
       [5.8, 2.7, 5.1, 1.9],
       [7.2, 3.2, 6. , 1.8],
       [5.7, 2.5, 5. , 2. ],
       [6.5, 3. , 5.2, 2. ],
       [6.7, 3.1, 4.7, 1.5]])
```

[24]:
```
X_test.shape
```

[24]:
```
(23, 4)
```

[25]:
```
y_test
```

[25]:
```
array([1, 2, 1, 0, 0, 2, 0, 0, 0, 2, 2, 1, 0, 1, 1, 2, 2, 2, 2, 2, 2, 2,
       1])
```

[26]:
```
y_test.shape
```

[26]:
```
(23,)
```

Let's plot the three datasets as well:

[27]:
```
df_train = pd.DataFrame(np.c_[X_train, y_train], columns=df.columns)
fig2 = px.scatter_matrix(df_train, dimensions=data_dimensions, color='Species')
fig2.show()
```

[28]:
```
df_validation = pd.DataFrame(np.c_[X_validation, y_validation], columns=df.
↪columns)
fig3 = px.scatter_matrix(df_validation, dimensions=data_dimensions,␣
↪color='Species')
fig3.show()
```

[29]:
```
df_test = pd.DataFrame(np.c_[X_test, y_test], columns=df.columns)
fig4 = px.scatter_matrix(df_test, dimensions=data_dimensions, color='Species')
fig4.show()
```

We are done!