Making Predictions

We can now make predictions using this model on new data for which we might not know the correct labels. Imagine we found an iris in the wild with a sepal length of 5 cm, a sepal width of 2.9 cm, a petal length of 1 cm, and a petal width of 0.2 cm. What species of iris would this be? We can put this data into a NumPy array, again by calculating the shape—that is, the number of samples (1) multiplied by the number of features (4):

In[27]:

```
X_new = np.array([[5, 2.9, 1, 0.2]])
print("X_new.shape: {}".format(X_new.shape))
```

Out[27]:

```
X_new.shape: (1, 4)
```

Note that we made the measurements of this single flower into a row in a two-dimensional NumPy array, as `scikit-learn` always expects two-dimensional arrays for the data.

To make a prediction, we call the `predict` method of the `knn` object:

In[28]:

```
prediction = knn.predict(X_new)
print("Prediction: {}".format(prediction))
print("Predicted target name: {}".format(
       iris_dataset['target_names'][prediction]))
```

Out[28]:

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

Our model predicts that this new iris belongs to the class 0, meaning its species is *setosa*. But how do we know whether we can trust our model? We don't know the correct species of this sample, which is the whole point of building the model!

Evaluating the Model

This is where the test set that we created earlier comes in. This data was not used to build the model, but we do know what the correct species is for each iris in the test set.

Therefore, we can make a prediction for each iris in the test data and compare it against its label (the known species). We can measure how well the model works by computing the *accuracy*, which is the fraction of flowers for which the right species was predicted: