Alexis Coffer
Samuel Peoples

scikit learn

Overview

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Installation

Scikit-learn requires:

- Python (>= 2.7 or >= 3.3),
- NumPy (>= 1.8.2),
- SciPy (>= 0.13.3).

Install using either *pip* or *conda*:

```
pip install -U scikit-learn
```

```
conda install scikit-learn
```

What can SKLearn do?

1. Classification: Identifying to which category an object belongs.

2. Regression: Predicting a continuous-valued attribute associated with an object.

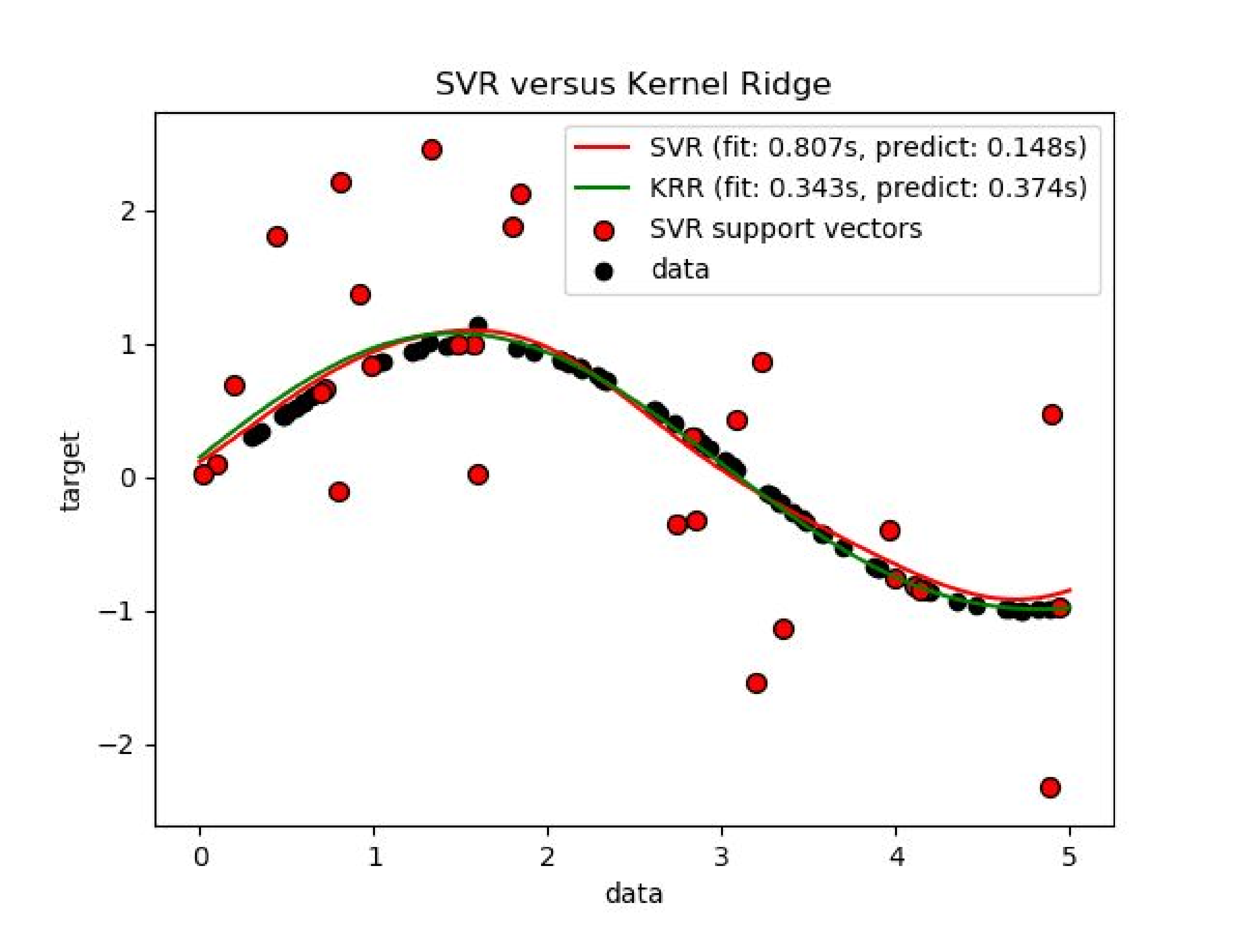
3. Clustering: Automatic grouping of similar objects into sets.

4. Data Preprocessing and Model Refinement: Feature extraction, normalization, and dimensionality reduction.

Example Uses

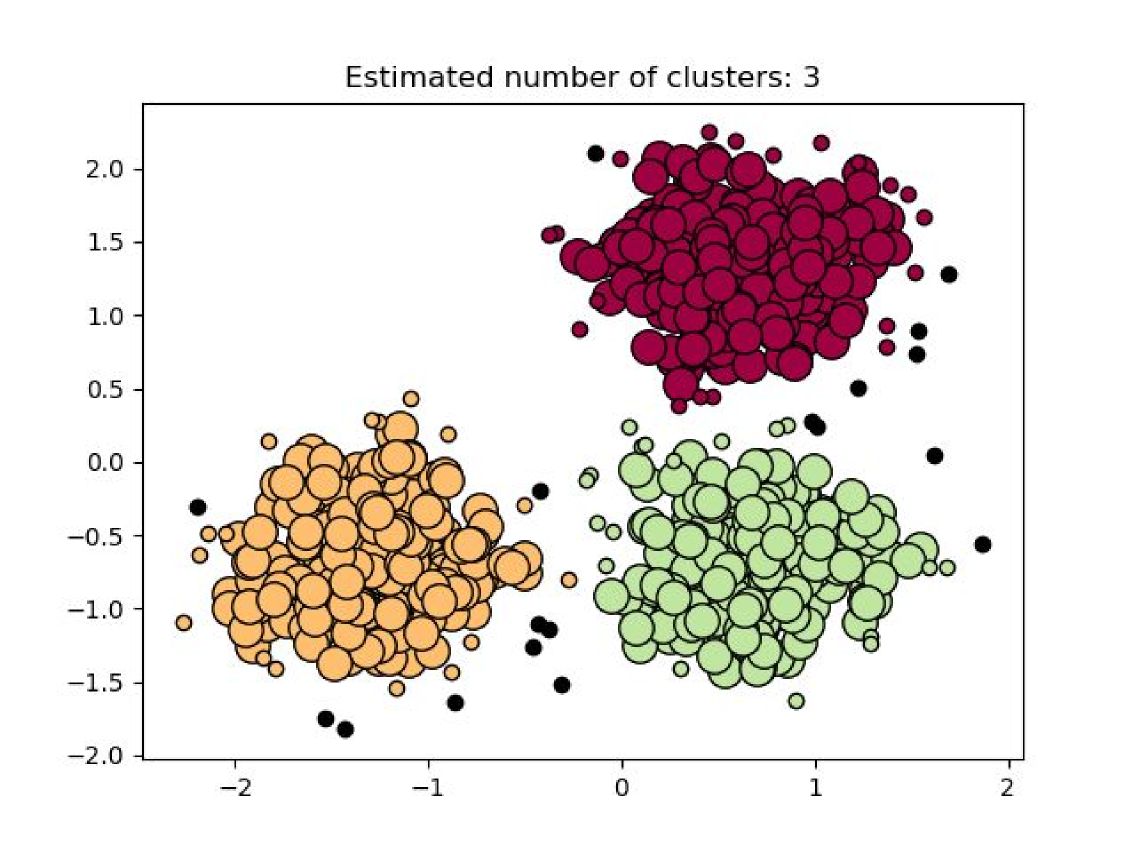
Regression

https://goo.gl/sEynCJ



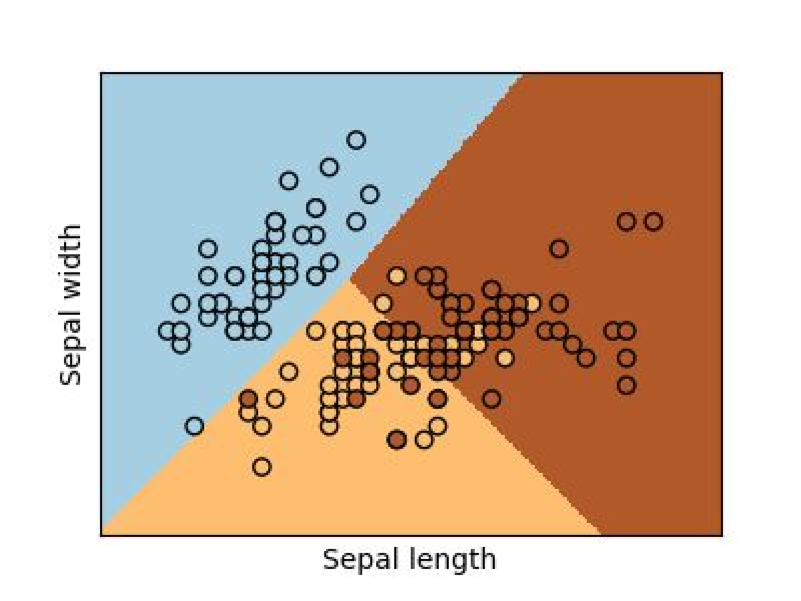
Clustering

https://goo.gl/4sdp4T



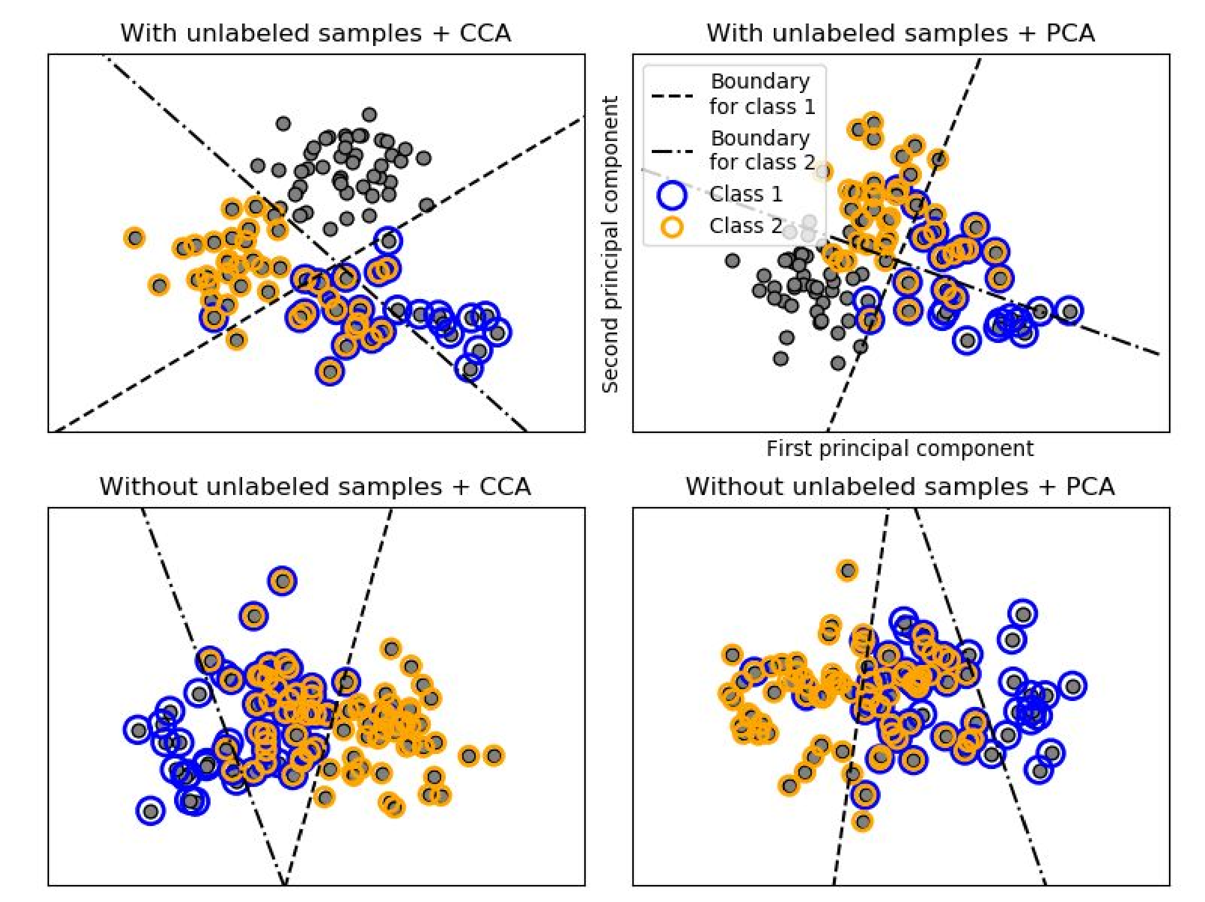
Regression (With Classification)

https://goo.gl/4pFdTq



Classification

https://goo.gl/ToQfyP



Typical Machine Learning Flow

1. Import libraries
2. Load dataset and assign x, y variables
3. Split variables into training and test sets
4. Feature-scale the data if it is highly variable
5. Fit the classifier to the training data
6. Predict the output for the test data
7. Verify accuracy of predictions, and optionally visualize results

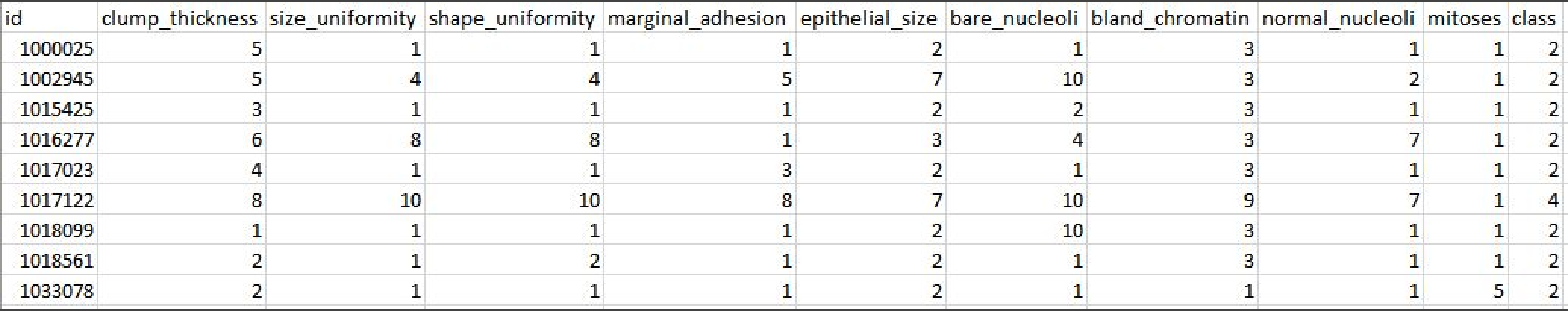
Walkthrough Example

breastCancer.csv, https://goo.gl/4uwY4k

University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg

breastCancer.py, https://goo.gl/RHsyne
Goal: Predict whether a tumor sample is malignant (4) or benign (2)

If you'd like to follow along, feel free to open a command prompt and navigate to the same directory as breastCancer.csv. The example can be run from the Python Command Line Interface.

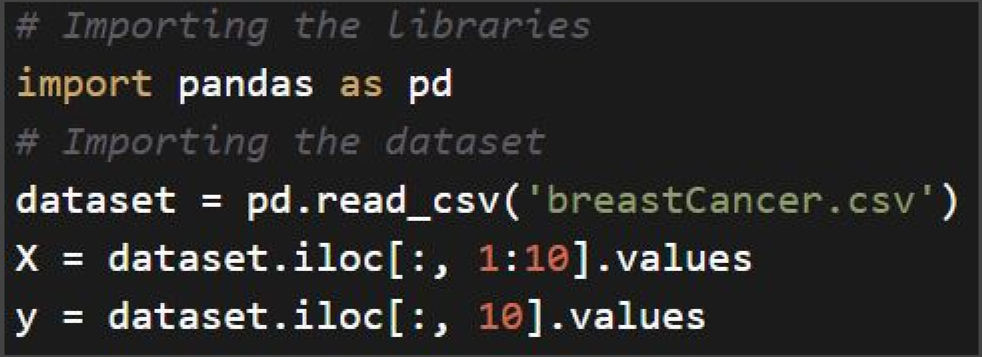


id	clump_thickness	size_uniformity	shape_uniformity	marginal_adhesion	epithelial_size	bare_nucleoli	bland_chromatin	normal_nucleoli	mitoses	class
1000025	5	1	1	1	2	1	3	1	1	2
1002945	5	4	4	5	7	10	3	2	1	2
1015425	3	1	1	1	2	2	3	1	1	2
1016277	6	8	8	1	3	4	3	7	1	2
1017023	4	1	1	3	2	1	3	1	1	2
1017122	8	10	10	8	7	10	9	7	1	4
1018099	1	1	1	1	2	10	3	1	1	2
1018561	2	1	2	1	2	1	3	1	1	2
1033078	2	1	1	1	2	1	1	1	5	2

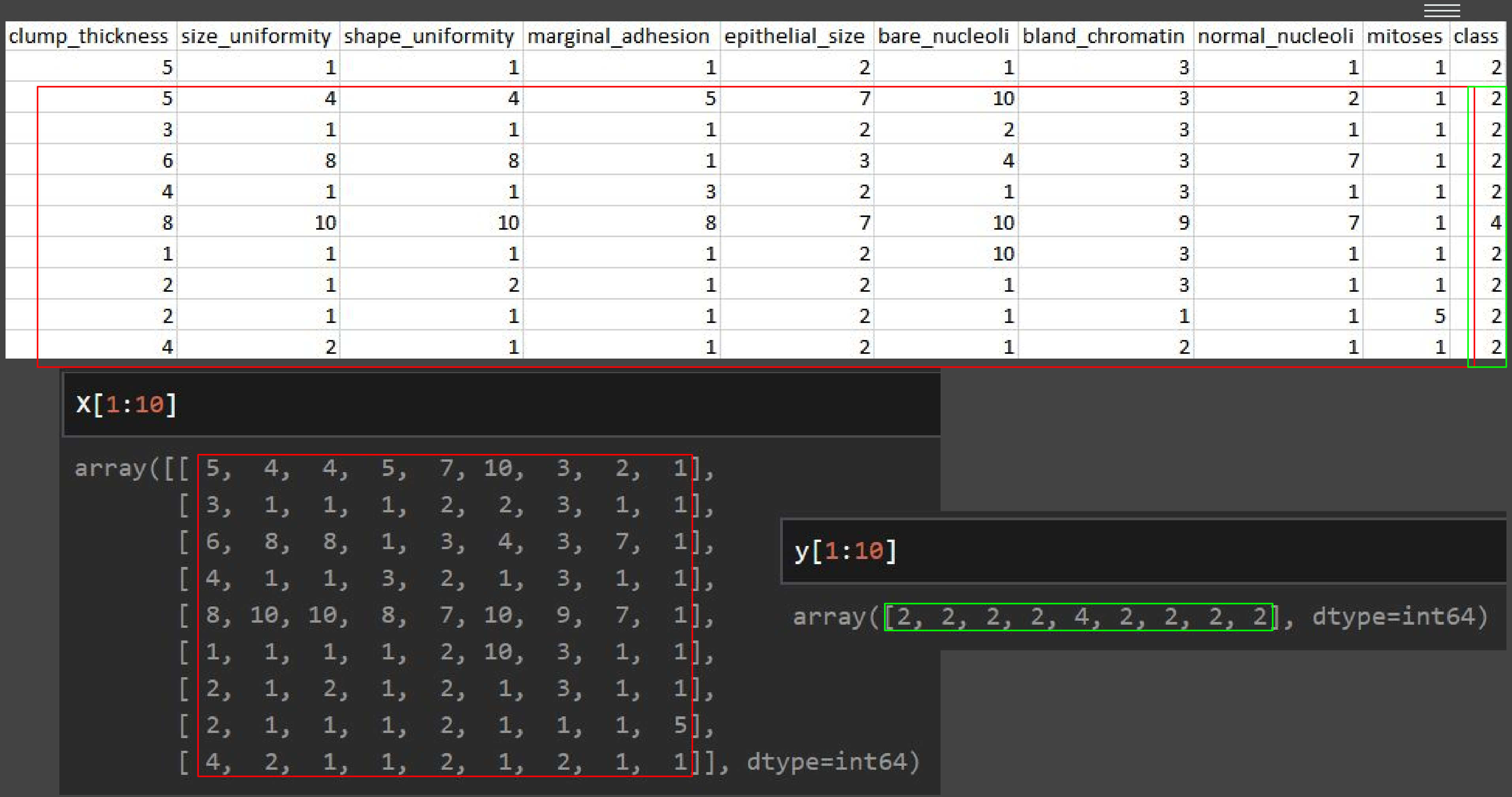
Importing libraries and defining variables.

Exclude the unique identifier, and the column we are predicting, so [clump_thickness]:[mitoses] will be our training data. Assign this to X.

Assign y as the final column.



```
# Importing the libraries
import pandas as pd
# Importing the dataset
dataset = pd.read_csv('breastCancer.csv')
X = dataset.iloc[:, 1:10].values
y = dataset.iloc[:, 10].values
```



clump_thickness	size_uniformity	shape_uniformity	marginal_adhesion	epithelial_size	bare_nucleoli	bland_chromatin	normal_nucleoli	mitoses	class
5	1	1	1	2	1	3	1	1	2
5	4	4	5	7	10	3	2	1	2
3	1	1	1	2	2	3	1	1	2
6	8	8	1	3	4	3	7	1	2
4	1	1	3	2	1	3	1	1	2
8	10	10	8	7	10	9	7	1	4
1	1	1	1	2	10	3	1	1	2
2	1	2	1	2	1	3	1	1	2
2	1	1	1	2	1	1	1	5	2
4	2	1	1	2	1	2	1	1	2

```
X[1:10]
```

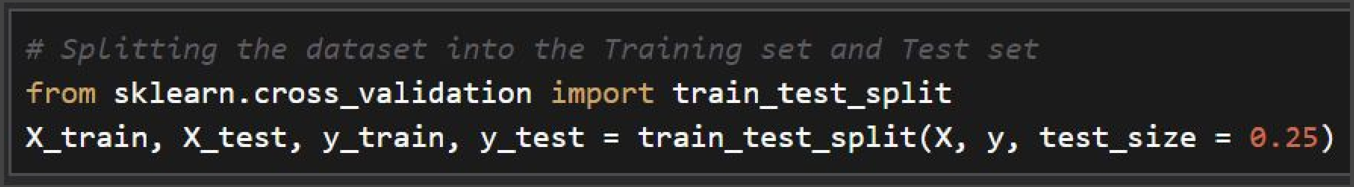
```
array([[ 5,  4,  4,  5,  7, 10,  3,  2,  1],
       [ 3,  1,  1,  1,  2,  2,  3,  1,  1],
       [ 6,  8,  8,  1,  3,  4,  3,  7,  1],
       [ 4,  1,  1,  3,  2,  1,  3,  1,  1],
       [ 8, 10, 10,  8,  7, 10,  9,  7,  1],
       [ 1,  1,  1,  1,  2, 10,  3,  1,  1],
       [ 2,  1,  2,  1,  2,  1,  3,  1,  1],
       [ 2,  1,  1,  1,  2,  1,  1,  1,  5],
       [ 4,  2,  1,  1,  2,  1,  2,  1,  1]], dtype=int64)
```

```
y[1:10]
```

```
array([2, 2, 2, 2, 4, 2, 2, 2, 2], dtype=int64)
```

Splitting the dataset into the Training set and Test set

Split the X and y variables into two sets so to have a more accurate machine learning model. In this instance 75% of our original data will be training data, and 25% will be our test data.

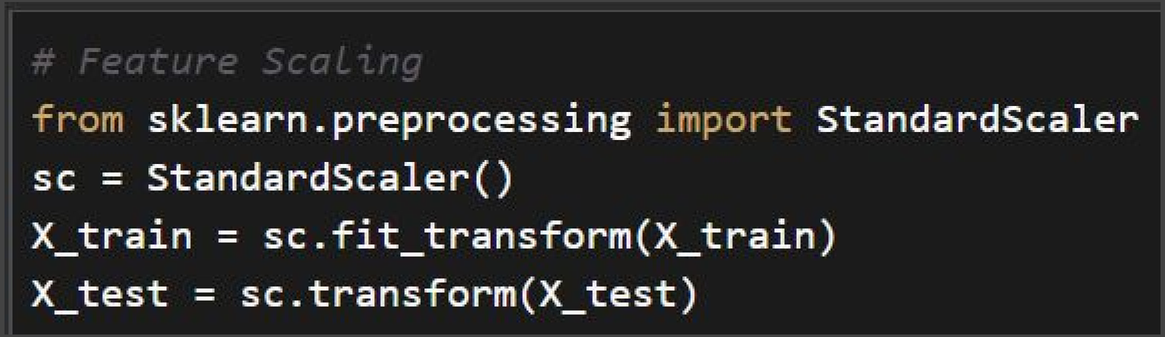


```
# Splitting the dataset into the Training set and Test set
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)
```

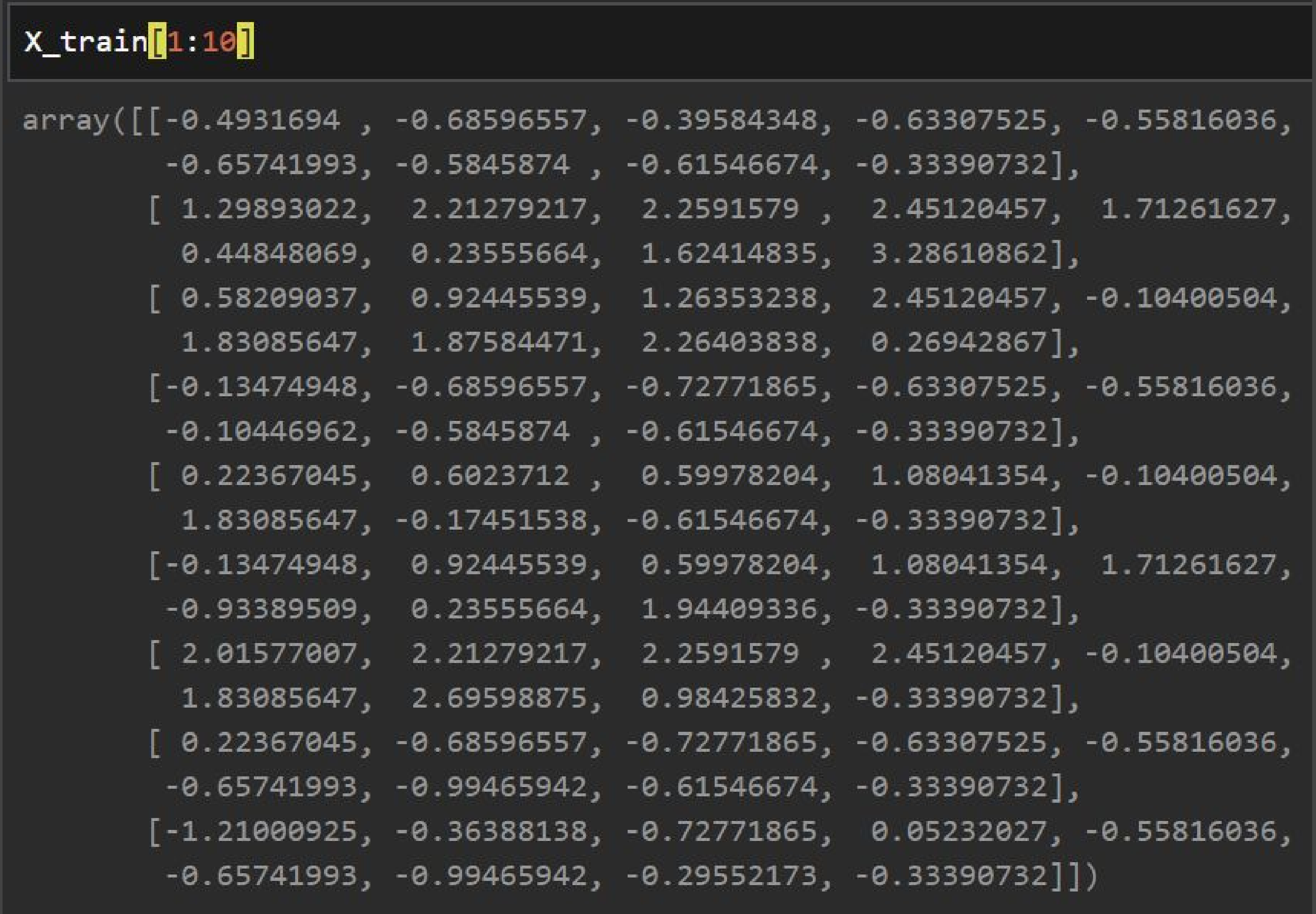
Feature scale the Data

Assign a variable to SKLearn’s Standard Scaler and scale the X variable.

Feature scaling through standardization can be an important preprocessing step for many machine learning algorithms. Standardization involves rescaling the features such that they have the properties of a standard normal distribution with a mean of zero and a standard deviation of one.



```
# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```



```
X_train[1:10]
```

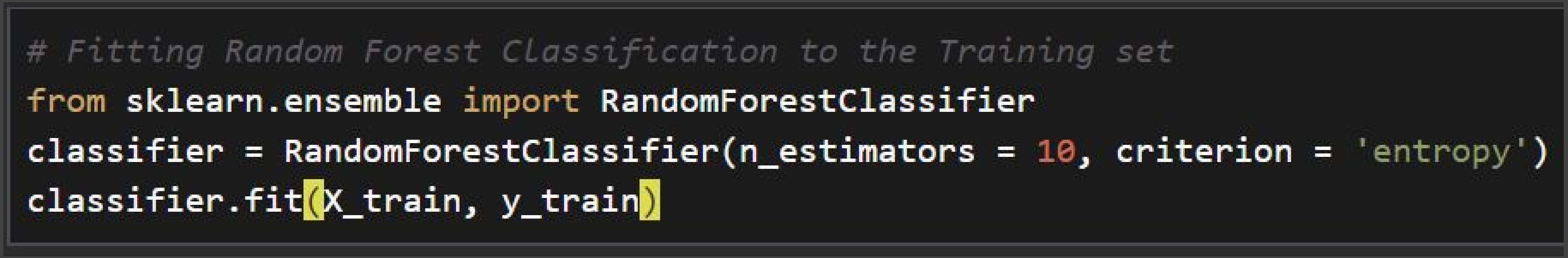
```
array([[-0.4931694 , -0.68596557, -0.39584348, -0.63307525, -0.55816036,
        -0.65741993, -0.5845874 , -0.61546674, -0.33390732],
       [ 1.29893022,  2.21279217,  2.2591579 ,  2.45120457,  1.71261627,
         0.44848069,  0.23555664,  1.62414835,  3.28610862],
       [ 0.58209037,  0.92445539,  1.26353238,  2.45120457, -0.10400504,
         1.83085647,  1.87584471,  2.26403838,  0.26942867],
       [-0.13474948, -0.68596557, -0.72771865, -0.63307525, -0.55816036,
        -0.10446962, -0.5845874 , -0.61546674, -0.33390732],
       [ 0.22367045,  0.6023712 ,  0.59978204,  1.08041354, -0.10400504,
         1.83085647, -0.17451538, -0.61546674, -0.33390732],
       [-0.13474948,  0.92445539,  0.59978204,  1.08041354,  1.71261627,
        -0.93389509,  0.23555664,  1.94409336, -0.33390732],
       [ 2.01577007,  2.21279217,  2.2591579 ,  2.45120457, -0.10400504,
         1.83085647,  2.69598875,  0.98425832, -0.33390732],
       [ 0.22367045, -0.68596557, -0.72771865, -0.63307525, -0.55816036,
        -0.65741993, -0.99465942, -0.61546674, -0.33390732],
       [-1.21000925, -0.36388138, -0.72771865,  0.05232027, -0.55816036,
        -0.65741993, -0.99465942, -0.29552173, -0.33390732]])
```

Fit the classifier to the Training Set

Using the entropy criterion for information gain, fitting to the training data.

n_estimators : integer, optional (default=10) The number of trees in the forest.

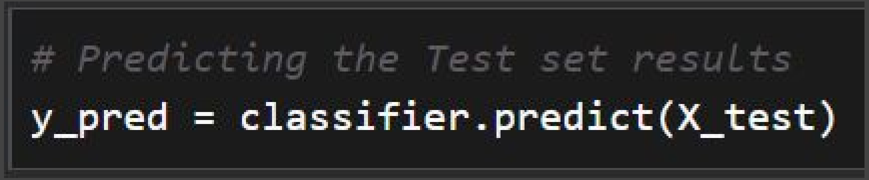
criterion : string, optional (default=”gini”) The function to measure the quality of a split. Supported criteria are “gini” for the Gini impurity and “entropy” for the information gain. Note: this parameter is tree-specific.



```
# Fitting Random Forest Classification to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entropy')
classifier.fit(X_train, y_train)
```

Predict the output using the trained model

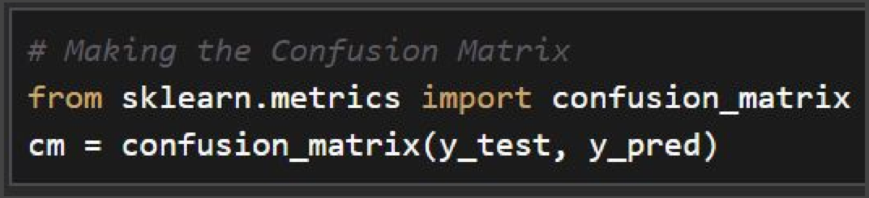
Using the trained classifier, assign the output variable using the test set.



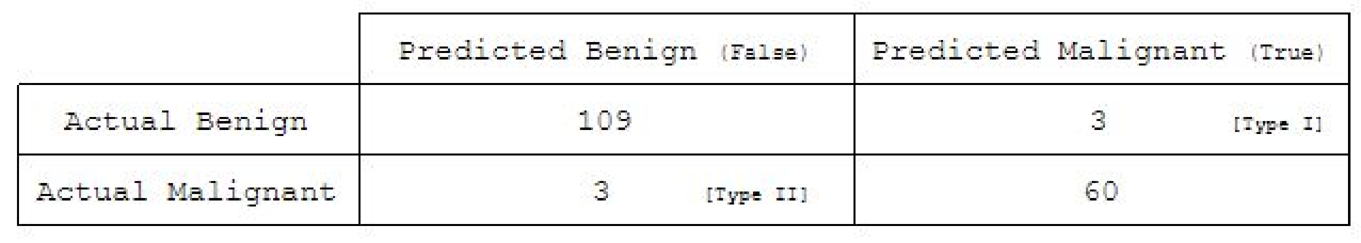
```
# Predicting the Test set results
y_pred = classifier.predict(X_test)
```

Create the Confusion Matrix

Measure the model’s accuracy with a confusion matrix

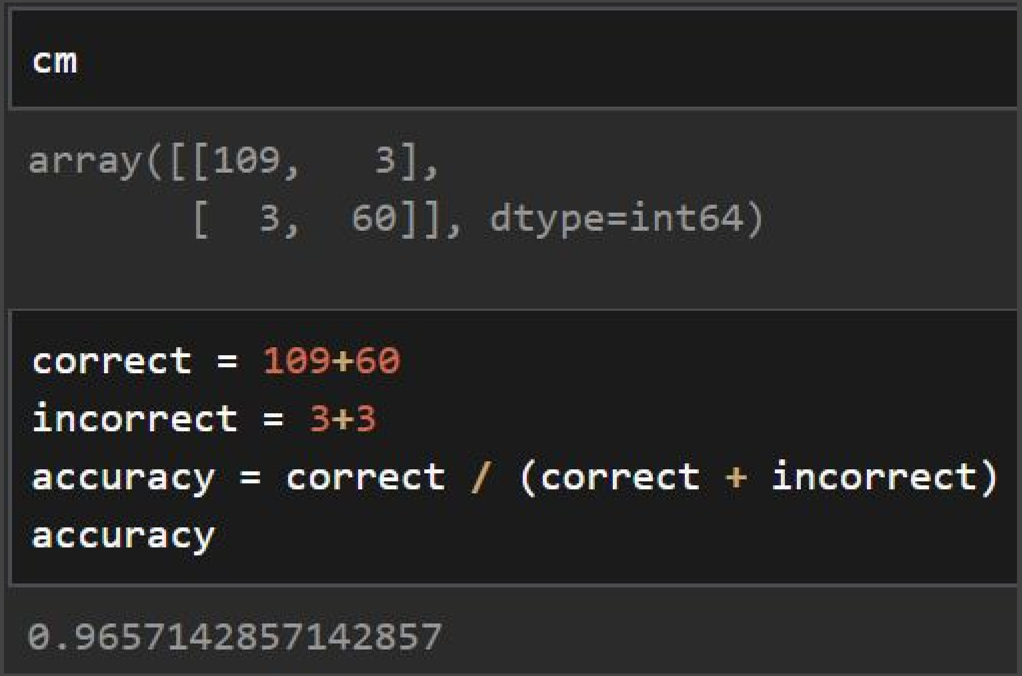


```python
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```



	Predicted Benign (False)	Predicted Malignant (True)
Actual Benign	109	3 [Type I]
Actual Malignant	3 [Type II]	60

96.6% Accuracy! This machine learning model has been able to predict whether a tumor sample is benign or malignant with high precision. This is an example of *boolean classification*!



```
cm
```

```
array([[109,   3],
       [  3,  60]], dtype=int64)
```

```
correct = 109+60
incorrect = 3+3
accuracy = correct / (correct + incorrect)
accuracy
```

```
0.9657142857142857
```

Conclusion

This python library allows users to conduct statistical and machine learning operations on given datasets with minimal work. The library has uses for *classification, regression, clustering, dimensionality reduction,* and *data preprocessing*. These operations can allow you to extract features, normalize data, conduct forecasts, and various other (progressively more advanced) machine learning techniques.

Alexis Coffer
Samuel Peoples

scikit
learn

Any Questions?