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

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.

Documentation: <http://scikit-learn.org/stable/user_guide.html>

Installation

Scikit-learn requires:

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

Installation can be completed from the command prompt using *pip* or *conda*.

pip install -U scikit-learn

conda install scikit-learn

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

Example: Predicting Malignancy of Breast Cancer Sample

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

<https://goo.gl/EWAzsM>

1. *"""*
2. *Random Forest Classification*
3. *Predicting Malignancy of Breast Cancer data*
4. *2: Benign, 4: Malignant*
5. *"""*
6. *# Importing the libraries*
7. **import** ***numpy*** as np
8. **import** ***matplotlib.pyplot*** as plt
9. **import** ***pandas*** as pd
10. *# Importing the dataset*
11. dataset = pd.read\_csv('breastCancer.csv')
12. X = dataset.iloc[:, 1:10].values
13. y = dataset.iloc[:, 10].values
14. *# Splitting the dataset into the Training set and Test set*
15. **from** ***sklearn.cross\_validation*** **import** ***train\_test\_split***
16. X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25)
17. *# Feature Scaling*
18. **from** ***sklearn.preprocessing*** **import** ***StandardScaler***
19. sc = StandardScaler()
20. X\_train = sc.fit\_transform(X\_train)
21. X\_test = sc.transform(X\_test)
22. *# Fitting Random Forest Classification to the Training set*
23. **from** ***sklearn.ensemble*** **import** ***RandomForestClassifier***
24. classifier = RandomForestClassifier(n\_estimators = 10, criterion = 'entropy')
25. classifier.fit(X\_train, y\_train)
26. *# Predicting the Test set results*
27. y\_pred = classifier.predict(X\_test)
28. *# Making the Confusion Matrix*
29. **from** ***sklearn.metrics*** **import** ***confusion\_matrix***
30. cm = confusion\_matrix(y\_test, y\_pred)
31. **print**(cm)

Confusion matrix:

|  |  |
| --- | --- |
| 109 | 3 |
| 3 | 60 |

* Only 3 Type-I errors and 3 Type-II errors
* Correct / Total = = 96.57% accuracy.