http://scikit-learn.org

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

Alexis Coffer
Samuel Peoples

http://scikit-learn.org

Example: Predicting Malignancy of Breast Cancer Sample

Dataset : University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg
https://goo.gl/EWAzsM

```
"""
Random Forest Classification
Predicting Malignancy of Breast Cancer data
2: Benign, 4: Malignant
"""
#      Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
#      Importing the dataset
dataset = pd.read_csv('breastCancer.csv')
X = dataset.iloc[:, 1:10].values
y = dataset.iloc[:, 10].values
#      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 Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
#      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)
#      Predicting the Test set results
y_pred = classifier.predict(X_test)
#      Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

Confusion matrix:

109	3
3	60

- Only 3 Type-I errors and 3 Type-II errors
- Correct / Total = $\frac{109+60}{109+60+3+3}$ = 96.57% accuracy.

Alexis Coffer
Samuel Peoples