Scikit-Learn: Machine Learning in the Python ecosystem

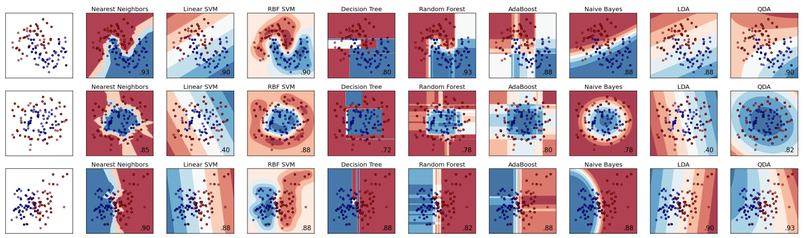
Arnaud Joly, Gilles Louppe, *and all scikit-learn contributors*

a.joly@ulg.ac.be @JolyArnaud g.louppe@ulg.ac.be @glouppe http://scikit-learn.org

Overview

The *scikit-learn* project is an increasingly popular machine learning library written in Python. It is designed to be **simple and efficient**, useful to both experts and non-experts, and reusable in a variety of contexts. The primary aim of the project is to provide a compendium of efficient implementations of classic, well-established machine learning algorithms.

Supervised learning : linear models (ridge, lasso, elastic net, ...), ensemble methods (random forests, bagging, boosting, ...), support vector machines.



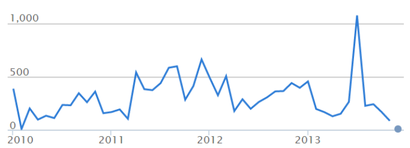
Unsupervised learning : clustering (KMeans, Ward, ...), matrix decomposition (PCA, ICA, ...), outlier detection.

Model selection and evaluation : cross-validation, grid-search, lots of metrics.

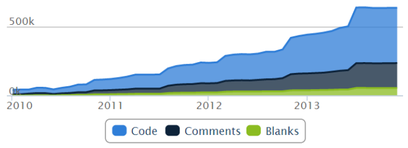
...and many more! (See our Reference).

Collaborative development

Around 10 core developers and more than 100 occasional contributors from all around the world. All are working together on GitHub with strict coding guidelines, including style consistency, unit-test coverage, documentation, examples and code review.



Number of commits



Number of line of codes

Who is using scikit-learn ?

In the Bioinformatics and Modelling research unit, *scikit-learn* is used by several researchers to perform **gene interaction discovery**, **bioinformatics image analysis** such as in the Cytomine project or more **fundamental research** in machine learning. The research unit is also actively contributing to the library with two researchers as core contributors.

In the applied science faculty of the University of Liège (ULg), *scikit-learn* is taught in the **machine learning course** and often used in students' master's theses.

scikit-learn is widely used in the industry (*e.g.* Spotify, Evernote, Phimeca) and in the academics world (*e.g.* ULg, Inria, Télécom ParisTech).

A simple and unified API

All objects in *scikit-learn* share a **uniform and limited API** consisting of three complementary interfaces :

- an estimator interface for building and fitting models ;
- a predictor interface for making predictions ;
- a transformer interface for converting data.

All of them takes as input data which is structured as Numpy arrays or Scipy sparse matrices.

```
>>> # Load data
>>> from sklearn.datasets import load_digits
>>> data = load_digits()
>>> X, y = data.data, data.target

>>> # Instantiate a classifier
>>> from sklearn.tree import DecisionTreeClassifier # Change this
>>> clf = DecisionTreeClassifier()                  # ... and that

>>> # Do stuff (and rely on the common interface across estimators)
>>> from sklearn.pipeline import Pipeline
>>> from sklearn.decomposition import PCA
>>> pipeline = Pipeline([("pca", PCA()), ("classifier", clf)])

>>> # Validate the model
>>> from sklearn.cross_validation import cross_val_score
>>> print "Accuracy = ", cross_val_score(pipeline, X, y, cv=3).mean()
Accuracy = 0.796863873387
```

Data analysis in Python

Let us illustrate the various components of the scientific Python ecosystem for the analysis of scientific data. We consider genetic data from the HapMap project which catalogs common genetic variants in human beings from 11 human populations from different parts of the world.

1) Data preprocessing with *scikit-learn* and pandas

Loading, converting and merging raw data files is easily done using *scikit-learn*, pandas, NumPy or other Python battery included modules.

Once data are loaded, *scikit-learn* and pandas provide **data preprocessing tools** to extract and to normalize data. For instance, there is missing values in the HapMap dataset that can be inferred using the most frequent values associated to each SNP.

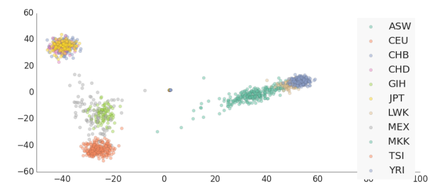
```
>>> from sklearn.preprocessing import Imputer
>>> imputer = Imputer(strategy="most_frequent", missing_values=-1)
>>> X = imputer.fit_transform(X)
```

2) Data visualization with matplotlib

In exploratory analysis, **data visualization** plays an important role in identifying interesting patterns. However data are often high dimensional. *scikit-learn* implements several **dimensionality reduction methods**. For illustration, let's study chromosome 15. First, we reduce the dimensionality of the data from 33800 to 2.

```
>>> from sklearn.decomposition import RandomizedPCA
>>> Xp = RandomizedPCA(n_components=2).fit_transform(X)
```

We rely on the matplotlib module for generating our figures and plots. In this plot, individuals from a same population are clustered together, which confirms that they share the same genetic variants.



3) Learning from the data with *scikit-learn*

In predictive analytics, the goal is to extract information to classify objects or to predict continuous value. With *scikit-learn*, you can **learn a model from the data**. We consider the learning task that consists in predicting the population of an individual given chromosome 15. Let us learn a forest of randomized tree model.

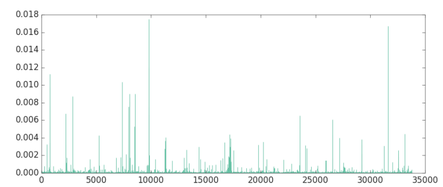
```
>>> from sklearn.cross_validation import train_test_split
>>> X_train, X_test, y_train, y_test = train_test_split(X, y)

>>> from sklearn.ensemble import ExtraTreesClassifier
>>> clf = ExtraTreesClassifier(n_estimators=100,
                               max_features=0.2).fit(X_train, y_train)

# Model prediction
>>> clf.predict(X_test)
array(['CHB', 'ASW', 'CEU', ..., 'GIH'])
```

4) Learning from a *scikit-learn* model

Once a model is fitted, you can **learn from the model**. Let us now examine the importance associated to each SNP to predict the population. The top SNP is rs1834640, which is known to be associated with skin pigmentation.



SNPs importance of chromosome 15

Conclusions

Python data-oriented packages tend to complement and integrate smoothly with each other. Together, they provide a powerful, flexible and coherent working environment for real-world scientific data analysis. Python is good and viable replacement to Matlab and R.

Scikit-Learn complements this ecosystem with machine learning algorithms and data analysis utilities.

www.scikit-learn.org