Machine Learning with Scikit-Learn

Andreas Mueller (NYU Center for Data Science, scikit-learn)

Classification
Regression
Clustering
Semi-Supervised Learning
Feature Selection
Feature Extraction
Manifold Learning
Dimensionality Reduction
Kernel Approximation
Hyperparameter Optimization
Evaluation Metrics
Out-of-core learning
…….
Get the notebooks!

Scikit-Learn is simple

Classification

```python
In [4]:
    from sklearn.datasets import load_iris
    from sklearn.cross_validation import train_test_split

    iris = load_iris()
    X, y = iris.data, iris.target
    X_train, X_test, y_train, y_test = train_test_split(X, y)
```

```python
In [5]:
    from sklearn.svm import SVC
    clf = SVC()
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
```

Transformations

```python
In [6]:
    from sklearn.decomposition import PCA

In [7]:
    pca = PCA(n_components=2)
    pca.fit(X)
    X_pca = pca.transform(X)
```

Tools

Cross-validation scoring

```python
In [36]:
    from sklearn.cross_validation import cross_val_score, StratifiedKFold

    scores = cross_val_score(SVC(), X_train, y_train, cv=5)
    print(scores)
```

```
[ 0.95652174  1.0  0.95652174  0.91304348  0.9  ]
```

Hi Andy,

I just received an email from the first tutorial speaker, presenting right before you, saying he's ill and won't be able to make it.

I know you have already committed yourself to two presentations, but is there anyway you could increase your tutorial time slot, maybe just offer time to try out what you've taught? Otherwise I have to do some kind of modern dance interpretation of Python in data :-)

-Leah

Hi Andreas,

I am very interested in your Machine Learning background. I work for X Recruiting who have been engaged by Z, a worldwide leading supplier of Y. We are expanding the core engineering team and we are looking for really passionate engineers who want to create their own story and help millions of people.

Can we find a time for a call to chat for a few minutes about this?

Thanks
Hi Andy,

I just received an email from the first tutorial speaker, presenting right before you, saying he's ill and won't be able to make it.

I know you have already committed yourself to two presentations, but is there anyway you could increase your slot a little, maybe just offer time to try out what you've taught? Otherwise I'll have to do some kind of modern dance interpretation of Python in data :-)

-Leah

Hi Andreas,

I am very interested in your Machine Learning background. I work for X Recruiting who have been engaged by Z, a worldwide leading supplier of Y. We are expanding the core engineering team and we are looking for really passionate engineers who want to create their own story and help millions of people.

Can we find a time to chat for a few minutes about this?

Thanks
Supervised Machine Learning

- Training Data
- Training Labels
- Model
Supervised Machine Learning

- Training Data
- Training Labels
- Test Data
- Model
- Prediction
Supervised Machine Learning

- Training Data
- Training Labels
- Test Data
- Test Labels

Model → Prediction → Evaluation
Supervised Machine Learning

Training Data

Training Labels

Test Data

Test Labels

Model

Prediction

Evaluation

Training

Generalization
clf = RandomForestClassifier()

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)
clf = RandomForestClassifier()

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)

clf.score(X_test, y_test)
IPython Notebook:
Chapter 1 - Introduction to Scikit-learn
Unsupervised Machine Learning

Training Data → Model
Unsupervised Machine Learning

- Training Data → Model
- Test Data → New View
Unsupervised Transformations

```python
pca = PCA()
pca.fit(X_train)
X_new = pca.transform(X_test)
```
IPython Notebook:
Chapter 2 – Unsupervised Transformers
All Data

Training data

Test data
All Data

Training data

Test data

Fold 1  Fold 2  Fold 3  Fold 4  Fold 5
IPython Notebook:
Chapter 3 - Cross-validation
In [2]:
clf = SVC()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
In [2]:

```python
clf = SVC()
clf.fit(X_train, y_train)
```

```
SVC(self, C=1.0, kernel='rbf', degree=3, gamma=0.0, coef0=0.0,
    shrinking=True, probability=False, tol=0.001, cache_size=200,
    class_weight=None, verbose=False, max_iter=-1, random_state=None)
```
All Data

Training data

Test data

Fold 1  Fold 2  Fold 3  Fold 4  Fold 5

Split 1
Fold 1  Fold 2  Fold 3  Fold 4  Fold 5

Split 2
Fold 1  Fold 2  Fold 3  Fold 4  Fold 5

Split 3
Fold 1  Fold 2  Fold 3  Fold 4  Fold 5

Split 4
Fold 1  Fold 2  Fold 3  Fold 4  Fold 5

Split 5
Fold 1  Fold 2  Fold 3  Fold 4  Fold 5

Test data
SVC(C=0.001, gamma=0.001)
SVC(C=0.001, gamma=0.001)  SVC(C=0.01, gamma=0.001)  SVC(C=0.1, gamma=0.001)  SVC(C=1, gamma=0.001)  SVC(C=10, gamma=0.001)
<table>
<thead>
<tr>
<th>SVC(C=0.001, gamma=0.001)</th>
<th>SVC(C=0.01, gamma=0.001)</th>
<th>SVC(C=0.1, gamma=0.001)</th>
<th>SVC(C=1, gamma=0.001)</th>
<th>SVC(C=10, gamma=0.001)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVC(C=0.001, gamma=0.01)</td>
<td>SVC(C=0.01, gamma=0.01)</td>
<td>SVC(C=0.1, gamma=0.01)</td>
<td>SVC(C=1, gamma=0.01)</td>
<td>SVC(C=10, gamma=0.01)</td>
</tr>
</tbody>
</table>
SVC(C=0.001, gamma=0.001)  SVC(C=0.01, gamma=0.001)  SVC(C=0.1, gamma=0.001)  SVC(C=1, gamma=0.001)  SVC(C=10, gamma=0.001)
SVC(C=0.001, gamma=0.01)  SVC(C=0.01, gamma=0.01)  SVC(C=0.1, gamma=0.01)  SVC(C=1, gamma=0.01)  SVC(C=10, gamma=0.01)
SVC(C=0.001, gamma=0.1)  SVC(C=0.01, gamma=0.1)  SVC(C=0.1, gamma=0.1)  SVC(C=1, gamma=0.1)  SVC(C=10, gamma=0.1)
<table>
<thead>
<tr>
<th>SVC(C=0.001, gamma=0.001)</th>
<th>SVC(C=0.01, gamma=0.001)</th>
<th>SVC(C=0.1, gamma=0.001)</th>
<th>SVC(C=1, gamma=0.001)</th>
<th>SVC(C=10, gamma=0.001)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVC(C=0.001, gamma=0.01)</td>
<td>SVC(C=0.01, gamma=0.01)</td>
<td>SVC(C=0.1, gamma=0.01)</td>
<td>SVC(C=1, gamma=0.01)</td>
<td>SVC(C=10, gamma=0.01)</td>
</tr>
<tr>
<td>SVC(C=0.001, gamma=0.1)</td>
<td>SVC(C=0.01, gamma=0.1)</td>
<td>SVC(C=0.1, gamma=0.1)</td>
<td>SVC(C=1, gamma=0.1)</td>
<td>SVC(C=10, gamma=0.1)</td>
</tr>
<tr>
<td>SVC(C=0.001, gamma=1)</td>
<td>SVC(C=0.01, gamma=1)</td>
<td>SVC(C=0.1, gamma=1)</td>
<td>SVC(C=1, gamma=1)</td>
<td>SVC(C=10, gamma=1)</td>
</tr>
<tr>
<td>SVC(C=0.001, gamma=10)</td>
<td>SVC(C=0.01, gamma=10)</td>
<td>SVC(C=0.1, gamma=10)</td>
<td>SVC(C=1, gamma=10)</td>
<td>SVC(C=10, gamma=10)</td>
</tr>
</tbody>
</table>
IPython Notebook:
Chapter 4 – Grid Searches
Training Labels

Training Data

Feature Extraction

Model
Training Labels \rightarrow Training Data
\hspace{1cm} \downarrow
\hspace{1cm} \downarrow
Feature Extraction \rightarrow Scaling
\hspace{1cm} \downarrow
Model
Training Labels

Feature Extraction

Scaling

Feature Selection

Model
Training Labels

Training Data

Feature Extraction

Scaling

Feature Selection

Model

Cross Validation
IPython Notebook:
Chapter 5 - Preprocessing and Pipelines
Do cross-validation over all steps jointly. Keep a separate test set until the very end.
Bag Of Word Representations

CountVectorizer / TfidfVectorizer
Bag Of Word Representations

CountVectorizer / TfidfVectorizer

“You better call Kenny Loggins”
Bag Of Word Representations

CountVectorizer / TfidfVectorizer

“You better call Kenny Loggins”

tokenizer

['you', 'better', 'call', 'kenny', 'loggins']
Bag Of Word Representations

CountVectorizer / TfidfVectorizer

"You better call Kenny Loggins"

tokenizer

['you', 'better', 'call', 'kenny', 'loggins']

Sparse matrix encoding

aardvak  better  call  you  zyxst
[0, ..., 0, 1, 0, ..., 0, 1, 0, ..., 0, 1, 0, ..., 0]
Application: Insult detection
Application: Insult detection

i really don't understand your point. It seems that you are mixing apples and oranges.
Application: Insult detection

i really don't understand your point. It seems that you are mixing apples and oranges.

Clearly you're a fucktard.
IPython Notebook:
Chapter 6 - Working With Text Data
Overfitting and Underfitting

Model complexity

Accuracy

Training

Generalization
Overfitting and Underfitting

- Accuracy
- Underfitting
- Overfitting
- Sweet spot
- Training
- Generalization

Model complexity
Linear SVM

\[ \hat{y} = \text{sign}(w_0 + \sum_i w_i x_i) \]
Linear SVM

\[ \hat{y} = \text{sign}(w_0 + \sum_i w_i x_i) \]
(RBF) Kernel SVM

\[ \hat{y} = \text{sign}(\alpha_0 + \sum_j \alpha_j y_j k(x^{(j)}, x)) \]
(RBF) Kernel SVM

\[ \hat{y} = \text{sign}(\alpha_0 + \sum_j \alpha_j y_j k(x^{(j)}, x)) \]

\[ k(x, x') = \exp(-\gamma \|x - x'\|^2) \]
(RBF) Kernel SVM

\[ \hat{y} = \text{sign}(\alpha_0 + \sum_j \alpha_j y_j k(x^{(j)}, x)) \]

\[ k(x, x') = \exp(-\gamma ||x - x'||^2) \]
(RBF) Kernel SVM

\[ \hat{y} = \text{sign}(\alpha_0 + \sum_j \alpha_j y_j k(x^{(j)}, x)) \]

\[ k(x, x') = \exp(-\gamma \|x - x'\|^2) \]
Decision Trees
Decision Trees
Decision Trees

```
max_depth = 2

X[0] <= 0.9963
  X[1] <= 1.5284
    value = [ 23, 4. ]
  X[1] <= -1.2058
    value = [ 0, 2. ]
    value = [ 1, 0. ]
    value = [ 1, 19. ]
```

Decision Trees

max_depth = 3

X[0] <= 0.9963

X[1] <= 1.5284

value = [0. 2.]

X[1] <= -1.2058

value = [1. 0.]

X[1] <= 0.2131

value = [1. 3.]

value = [0. 16.]
Decision Trees

max_depth = 5

X[0] <= 0.9963
X[1] <= 1.5284
X[1] <= -1.2058
X[1] <= 0.1055
value = [ 0.  2.]
value = [ 1.  0.]
X[0] <= 0.3755
value = [ 11.  0.]
X[1] <= 0.8215
value = [ 5.  0.]
X[0] <= 0.6284
value = [ 6.  2.]
X[1] <= 0.2131
value = [ 0.  2.]
value = [ 1.  0.]
value = [ 0.  16.]
value = [ 0.  3.]
value = [ 1.  0.]

66
Decision Trees

max_depth = 7

```
X[0] <= 0.9963
  |      
    |        |
X[1] <= 0.1055 X[0] <= 0.3755
    |        |
X[1] <= 0.8215 X[0] <= 0.6284
    |        |
value = [1. 0.] value = [0. 2.] value = [1. 0.]
X[0] <= 1.1913
  |      
X[0] <= -0.1072
    |        |
value = [0. 2.] value = [5. 0.]
    |        |
value = [0. 2.] value = [1. 0.]
```
Random Forests
Random Forests
Random Forests
Documentation of scikit-learn 0.16-git

Quick Start
A very short introduction into machine learning problems and how to solve them using scikit-learn. Introduced basic concepts and conventions.

User Guide
The main documentation. This contains an in-depth description of all algorithms and how to apply them.

Other Versions
- scikit-learn 0.15 (stable)
- scikit-learn 0.16 (development)
- scikit-learn 0.14
- scikit-learn 0.13
- scikit-learn 0.12
- Older versions

Tutorials
Useful tutorials for developing a feel for some of scikit-learn’s applications in the machine learning field.

API
The exact API of all functions and classes, as given by the docstrings. The API documents expected types and allowed features for all functions, and all parameters available for the algorithms.

Additional Resources
Talks given, slide-sets and other information relevant to scikit-learn.

Contributing
Information on how to contribute. This also contains useful information for advanced users, for example how to build their own estimators.

Flow Chart
A graphical overview of basic areas of machine learning, and guidance which kind of algorithms to use in a given situation.

FAQ
Frequently asked questions about the project and contributing.
Thank you for your attention.

@t3kcit
@amueller
importamueller@gmail.com