Machine Learning with Scikit-Learn

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

Material: http://bit.ly/nycmlsklearn

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

Home Installation Documentation Examples

Google™ Custom Search Search ×

Documentation of scikit-learn 0.17

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 0.18 (development)
- scikit-learn 0.17 (stable)
- scikit-learn 0.16
- scikit-learn 0.15

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.

http://scikit-learn.org/

Doing Machine Learning With Scikit-Learn

Representing Data

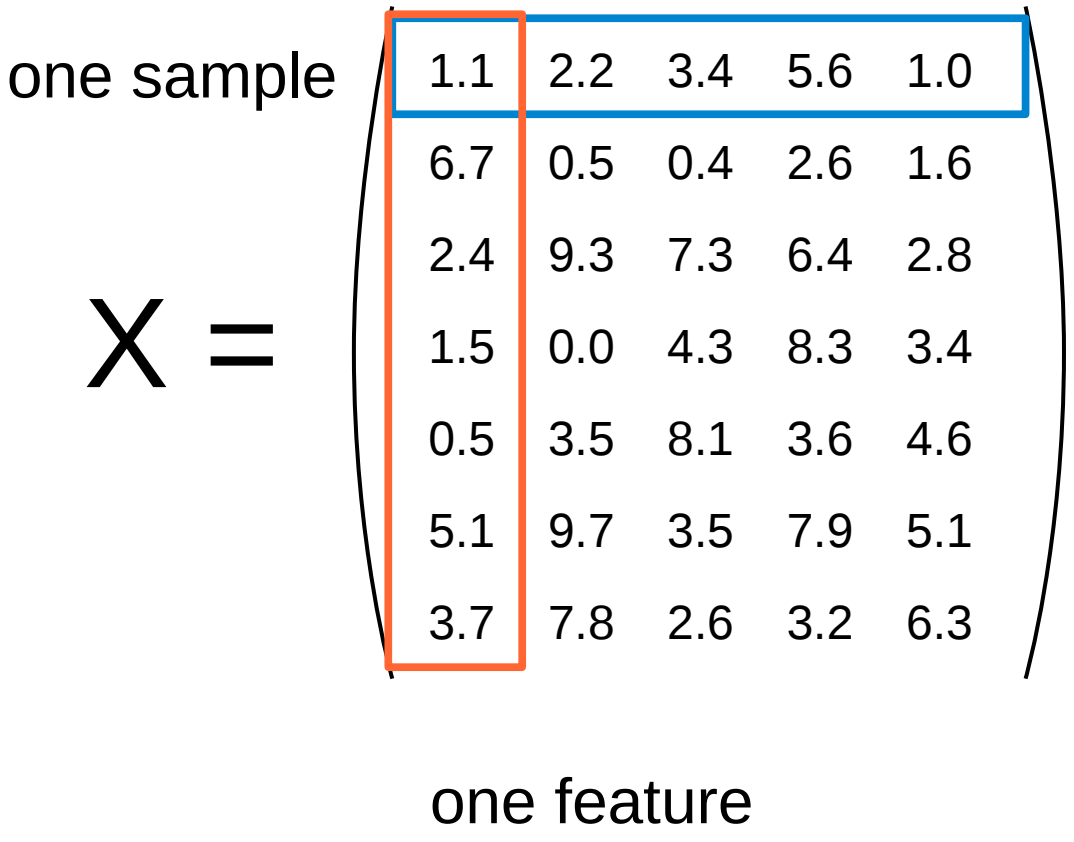
$$X = \begin{pmatrix} 1.1 & 2.2 & 3.4 & 5.6 & 1.0 \\ 6.7 & 0.5 & 0.4 & 2.6 & 1.6 \\ 2.4 & 9.3 & 7.3 & 6.4 & 2.8 \\ 1.5 & 0.0 & 4.3 & 8.3 & 3.4 \\ 0.5 & 3.5 & 8.1 & 3.6 & 4.6 \\ 5.1 & 9.7 & 3.5 & 7.9 & 5.1 \\ 3.7 & 7.8 & 2.6 & 3.2 & 6.3 \end{pmatrix}$$

Representing Data

one sample

$$X = \begin{pmatrix} 1.1 & 2.2 & 3.4 & 5.6 & 1.0 \\ 6.7 & 0.5 & 0.4 & 2.6 & 1.6 \\ 2.4 & 9.3 & 7.3 & 6.4 & 2.8 \\ 1.5 & 0.0 & 4.3 & 8.3 & 3.4 \\ 0.5 & 3.5 & 8.1 & 3.6 & 4.6 \\ 5.1 & 9.7 & 3.5 & 7.9 & 5.1 \\ 3.7 & 7.8 & 2.6 & 3.2 & 6.3 \end{pmatrix}$$

Representing Data



Representing Data

one sample

$$X = \begin{pmatrix} 1.1 & 2.2 & 3.4 & 5.6 & 1.0 \\ 6.7 & 0.5 & 0.4 & 2.6 & 1.6 \\ 2.4 & 9.3 & 7.3 & 6.4 & 2.8 \\ 1.5 & 0.0 & 4.3 & 8.3 & 3.4 \\ 0.5 & 3.5 & 8.1 & 3.6 & 4.6 \\ 5.1 & 9.7 & 3.5 & 7.9 & 5.1 \\ 3.7 & 7.8 & 2.6 & 3.2 & 6.3 \end{pmatrix}$$

one feature

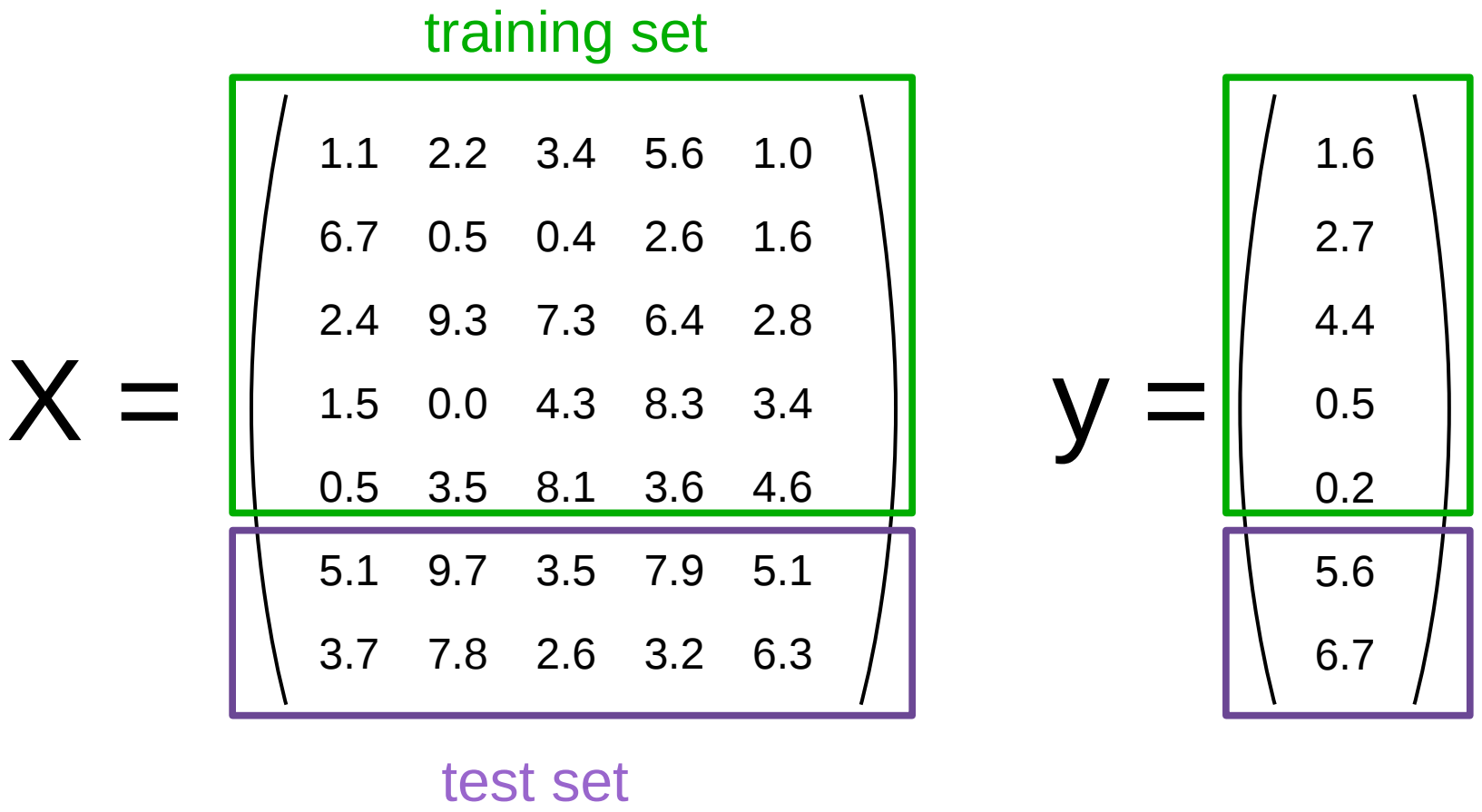
$$y = \begin{pmatrix} 1.6 \\ 2.7 \\ 4.4 \\ 0.5 \\ 0.2 \\ 5.6 \\ 6.7 \end{pmatrix}$$

outputs / labels

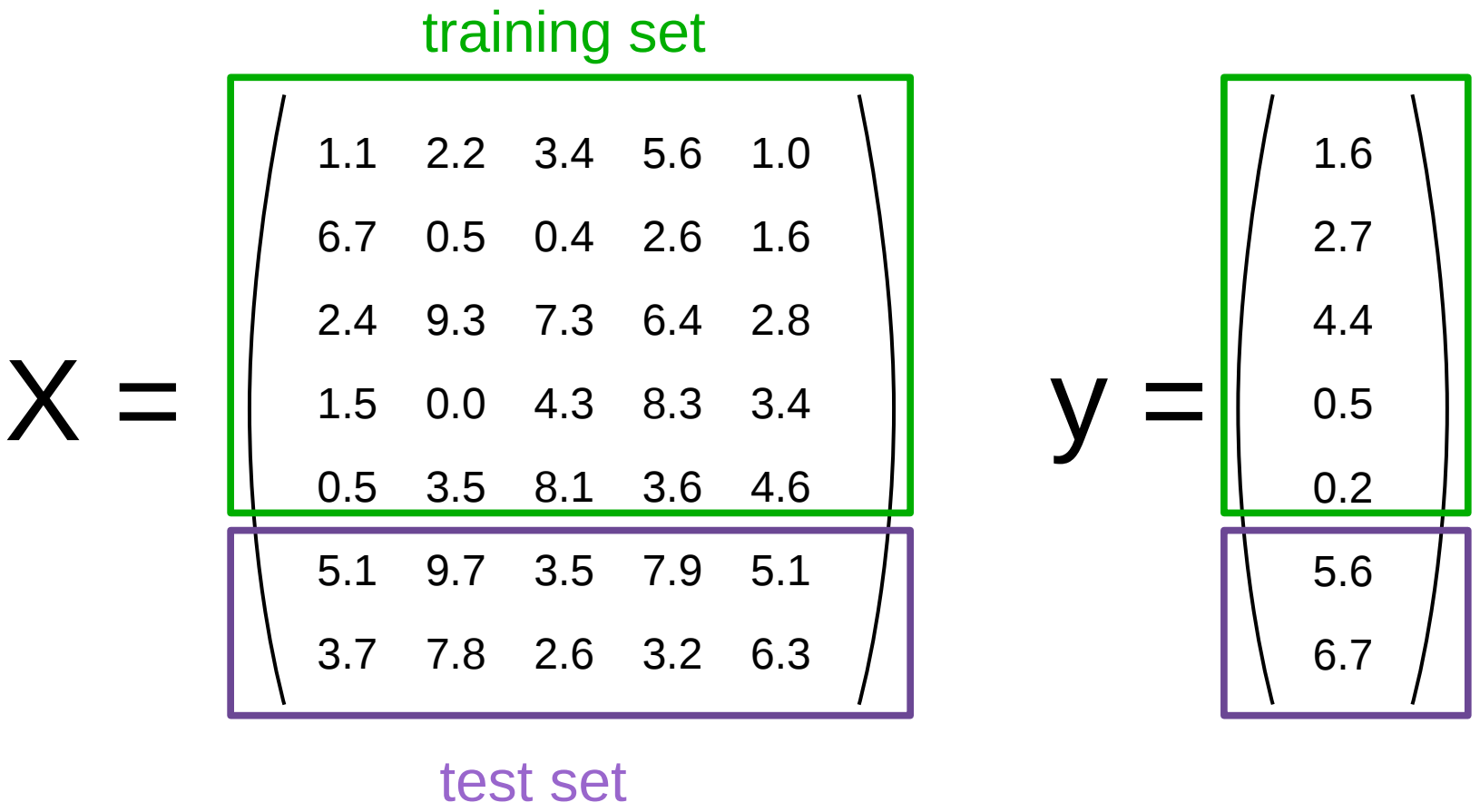
Training and Testing Data

$$X = \begin{pmatrix} 1.1 & 2.2 & 3.4 & 5.6 & 1.0 \\ 6.7 & 0.5 & 0.4 & 2.6 & 1.6 \\ 2.4 & 9.3 & 7.3 & 6.4 & 2.8 \\ 1.5 & 0.0 & 4.3 & 8.3 & 3.4 \\ 0.5 & 3.5 & 8.1 & 3.6 & 4.6 \\ 5.1 & 9.7 & 3.5 & 7.9 & 5.1 \\ 3.7 & 7.8 & 2.6 & 3.2 & 6.3 \end{pmatrix} \quad y = \begin{pmatrix} 1.6 \\ 2.7 \\ 4.4 \\ 0.5 \\ 0.2 \\ 5.6 \\ 6.7 \end{pmatrix}$$

Training and Testing Data

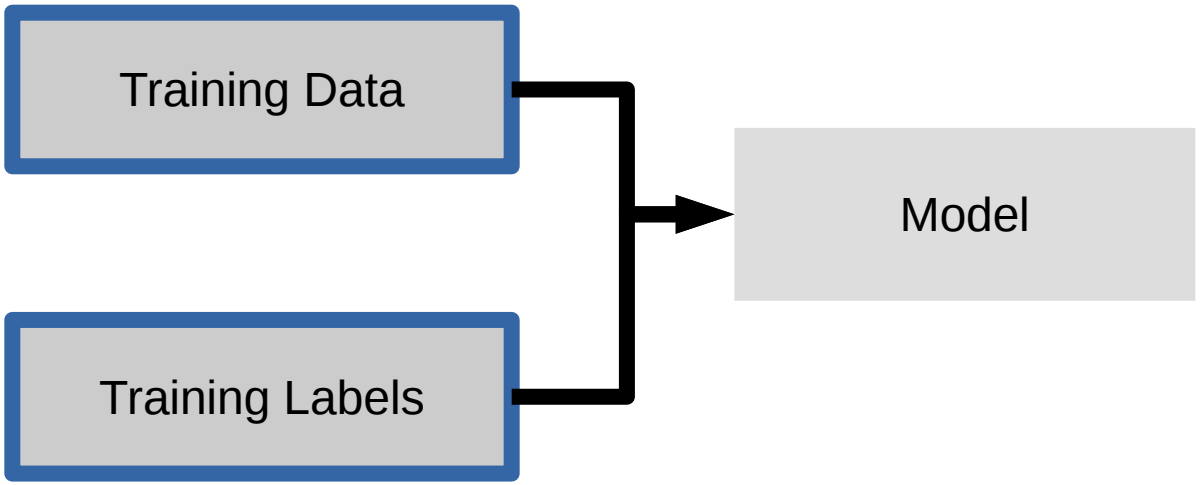


Training and Testing Data

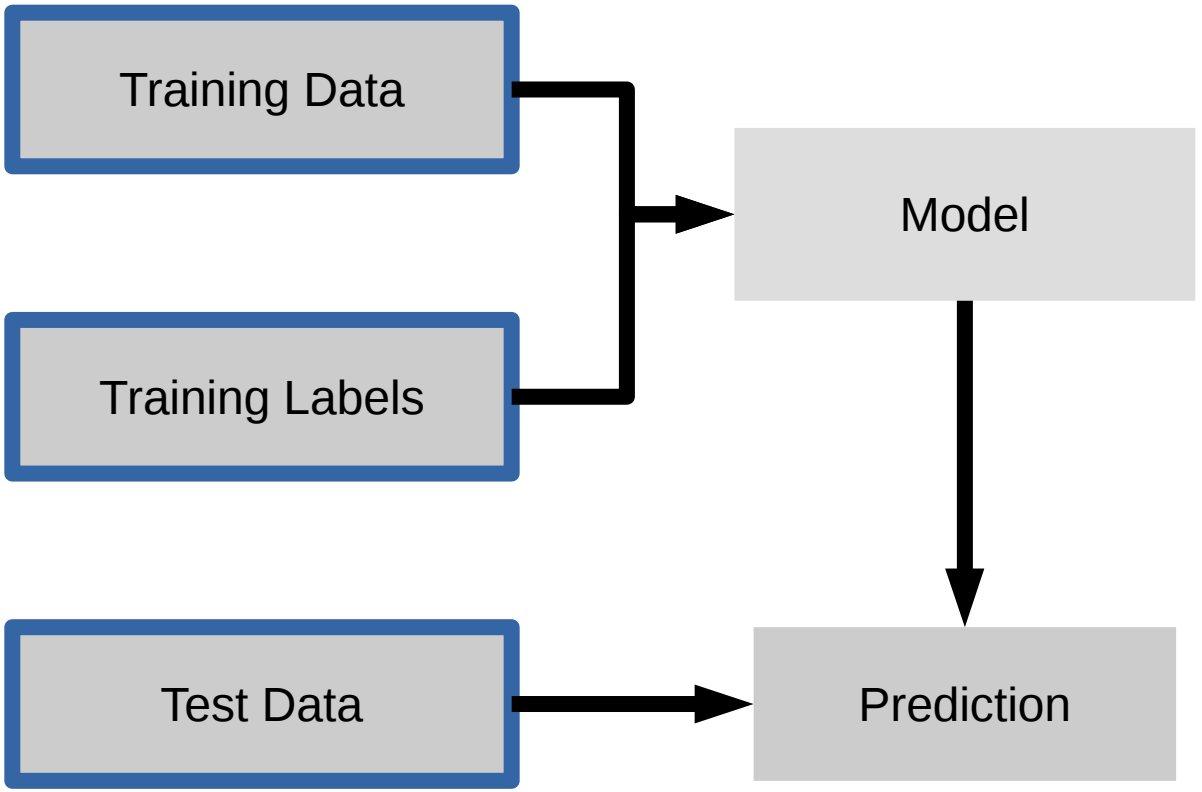



```
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y)
```

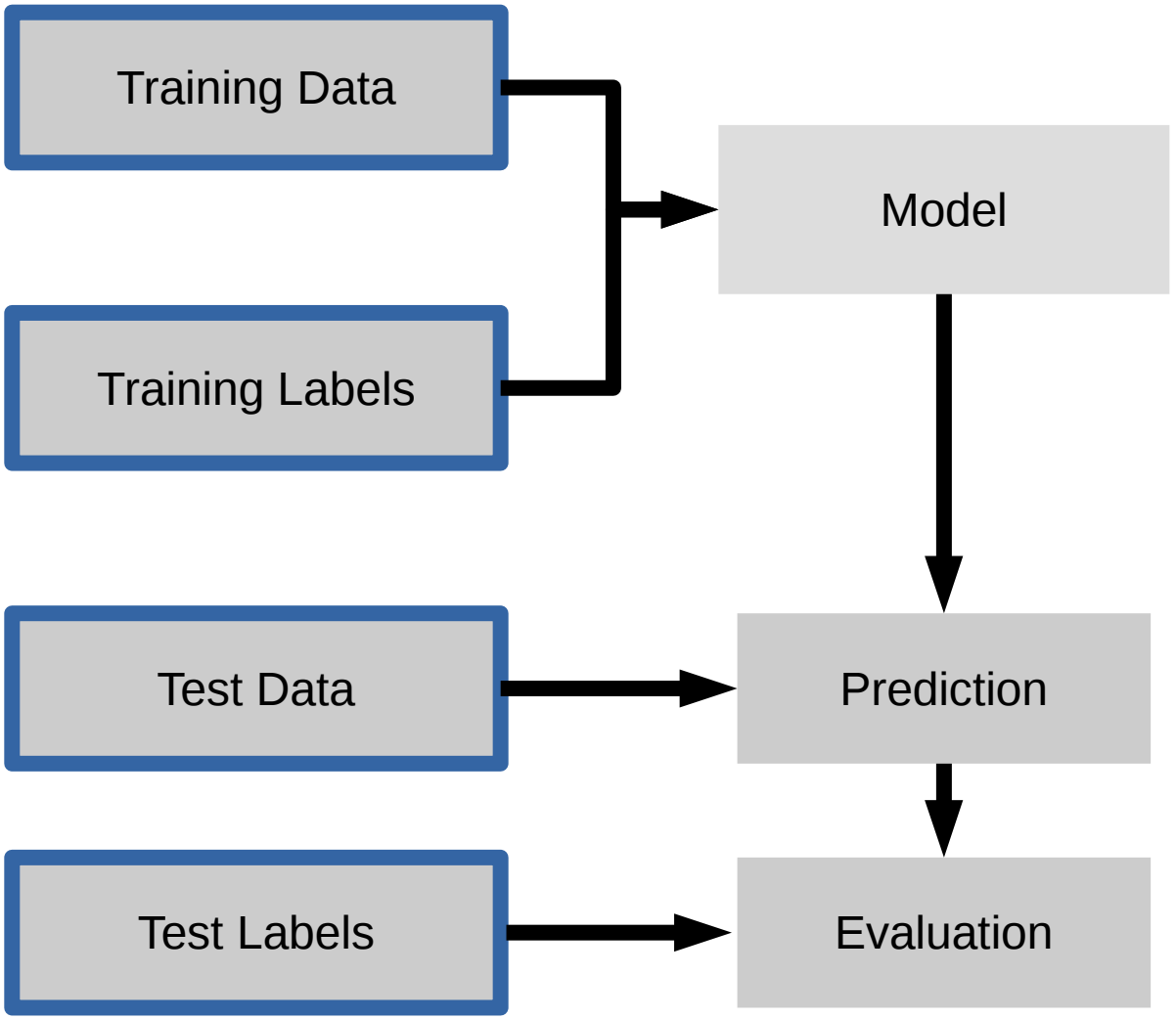
Supervised Machine Learning



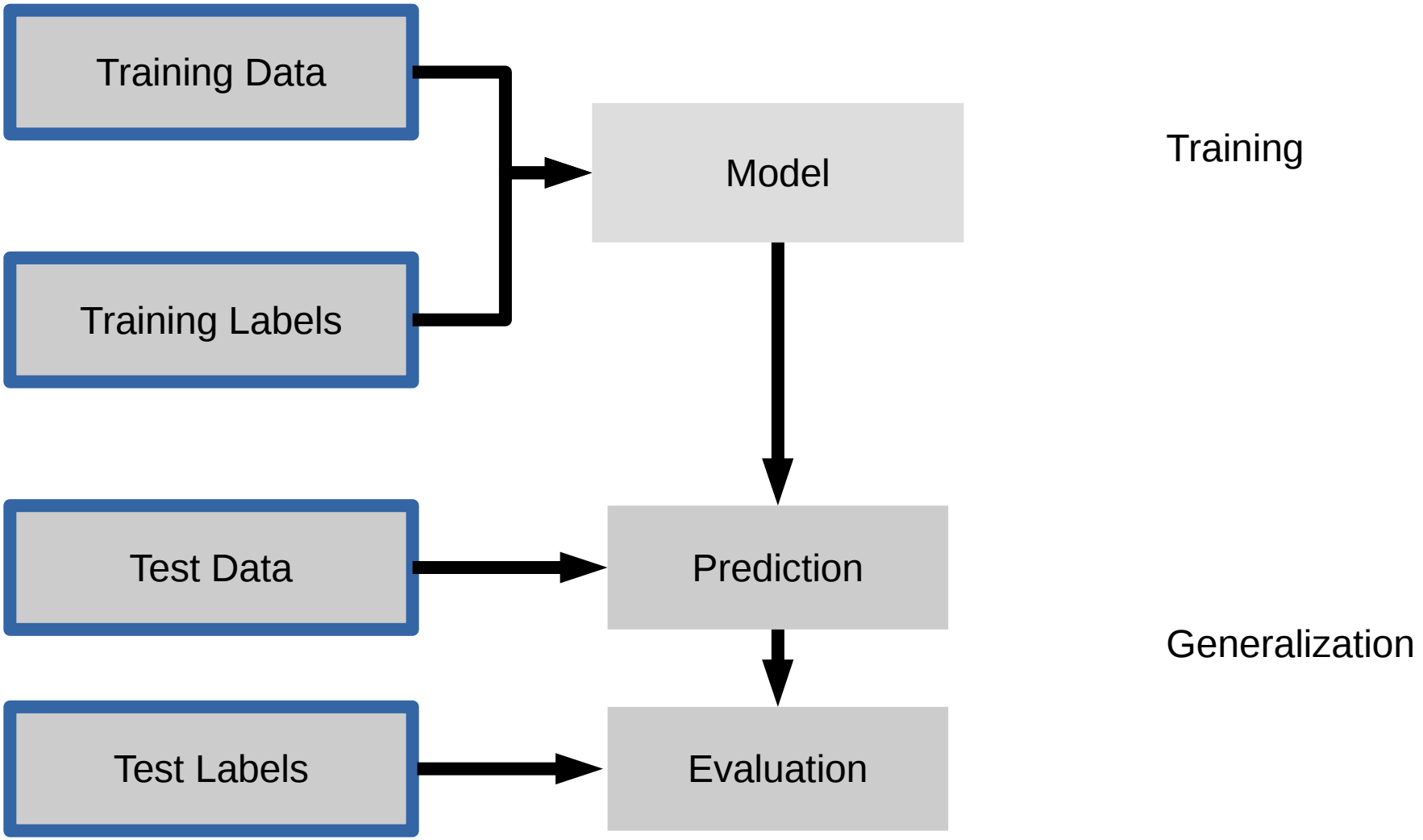
Supervised Machine Learning



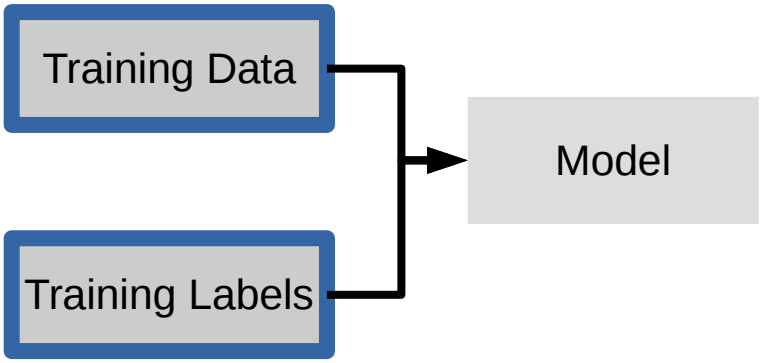
Supervised Machine Learning



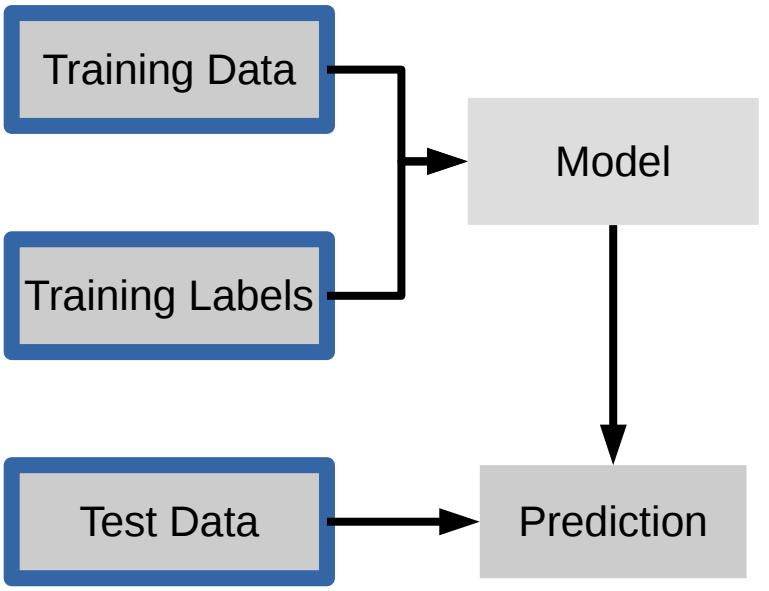
Supervised Machine Learning



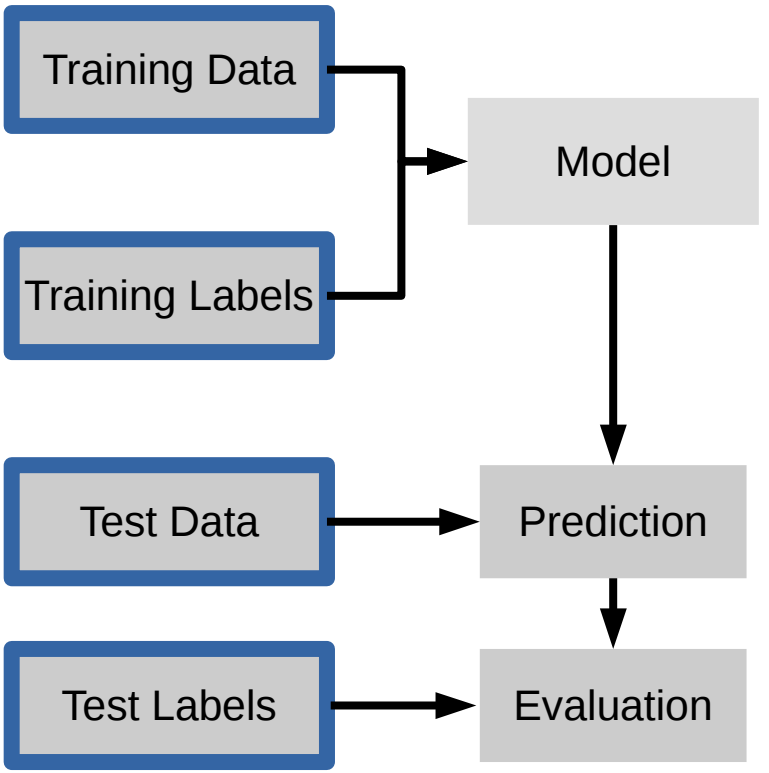
```
clf = RandomForestClassifier()

clf.fit(X_train, y_train)
```



```
clf = RandomForestClassifier()

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)
```

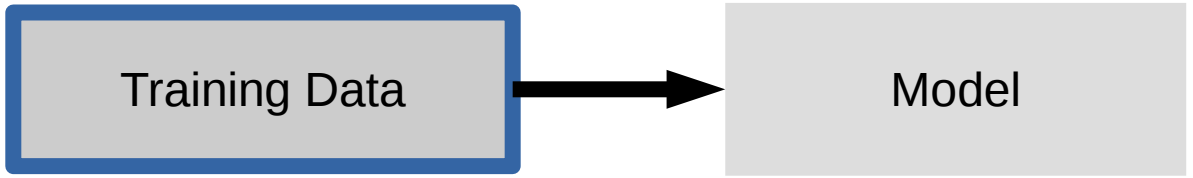


```
clf = RandomForestClassifier()

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)

clf.score(X_test, y_test)
```

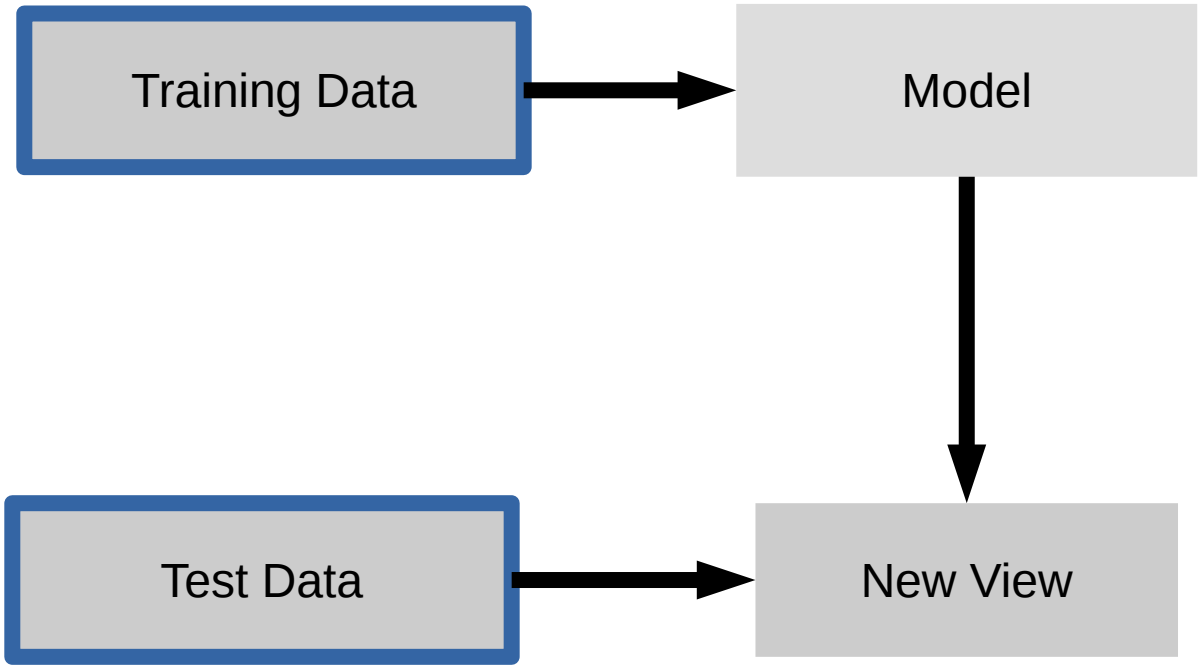


IPython Notebook:
Part 1 - Introduction to Scikit-learn

Unsupervised Machine Learning



Unsupervised Machine Learning

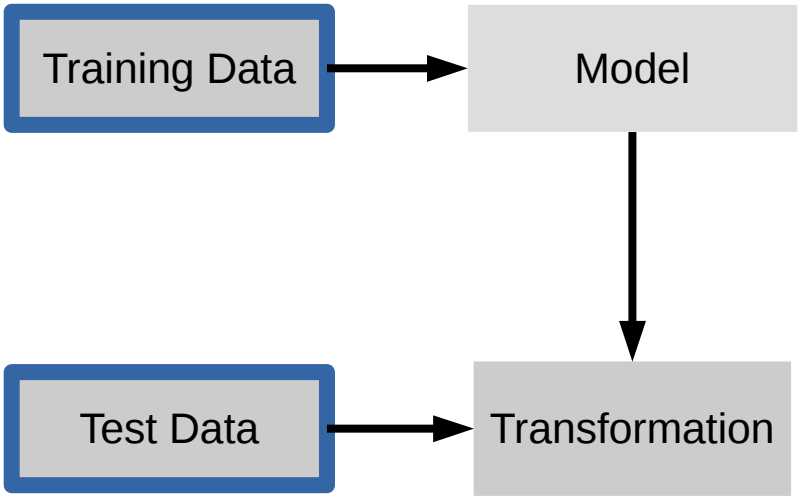


Unsupervised Transformations

```
pca = PCA()
```

```
pca.fit(X_train)
```

```
X_new = pca.transform(X_test)
```



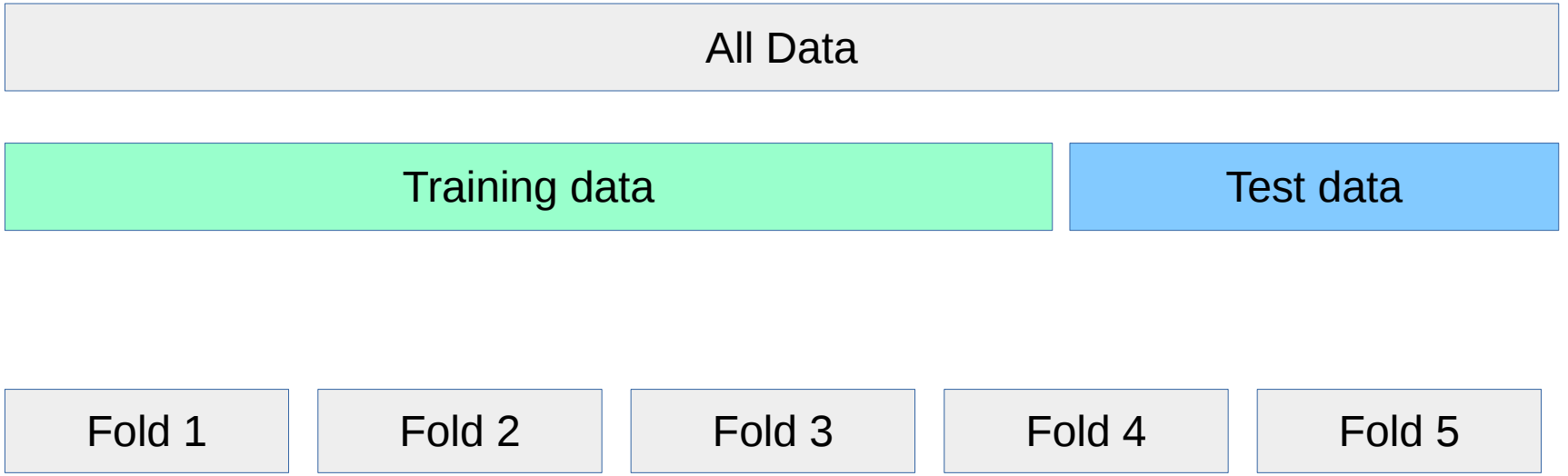
IPython Notebook:
Part 2 – Unsupervised Transformers

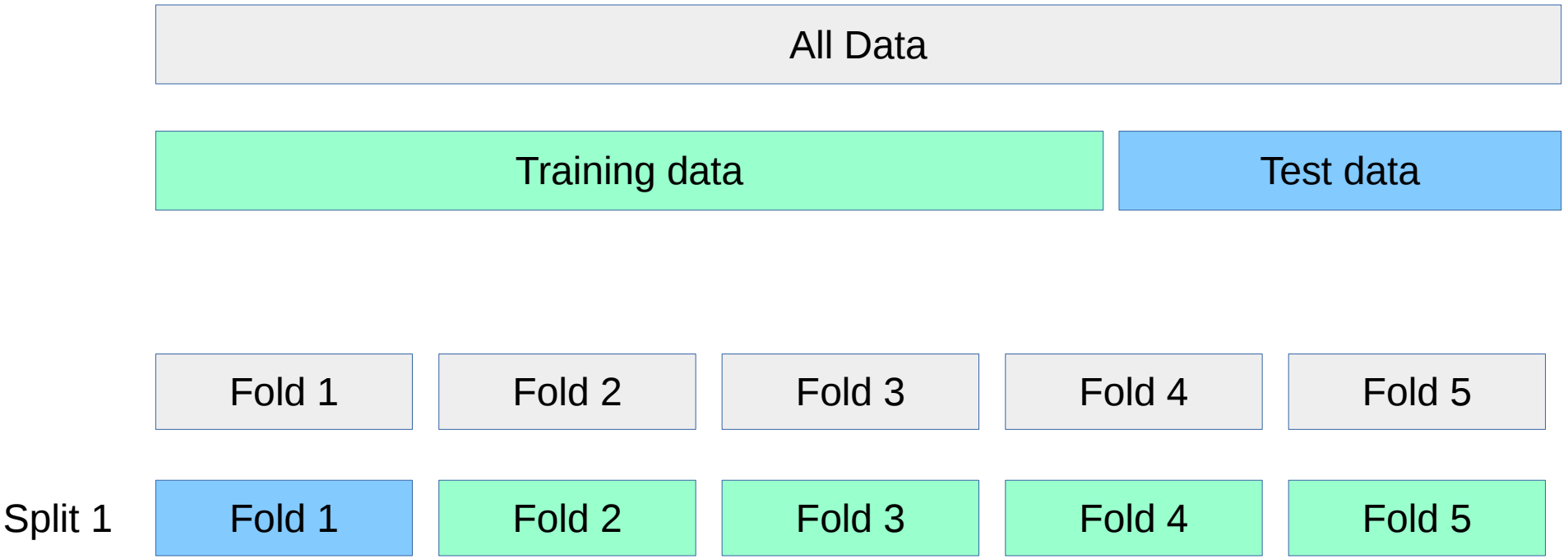
Basic API

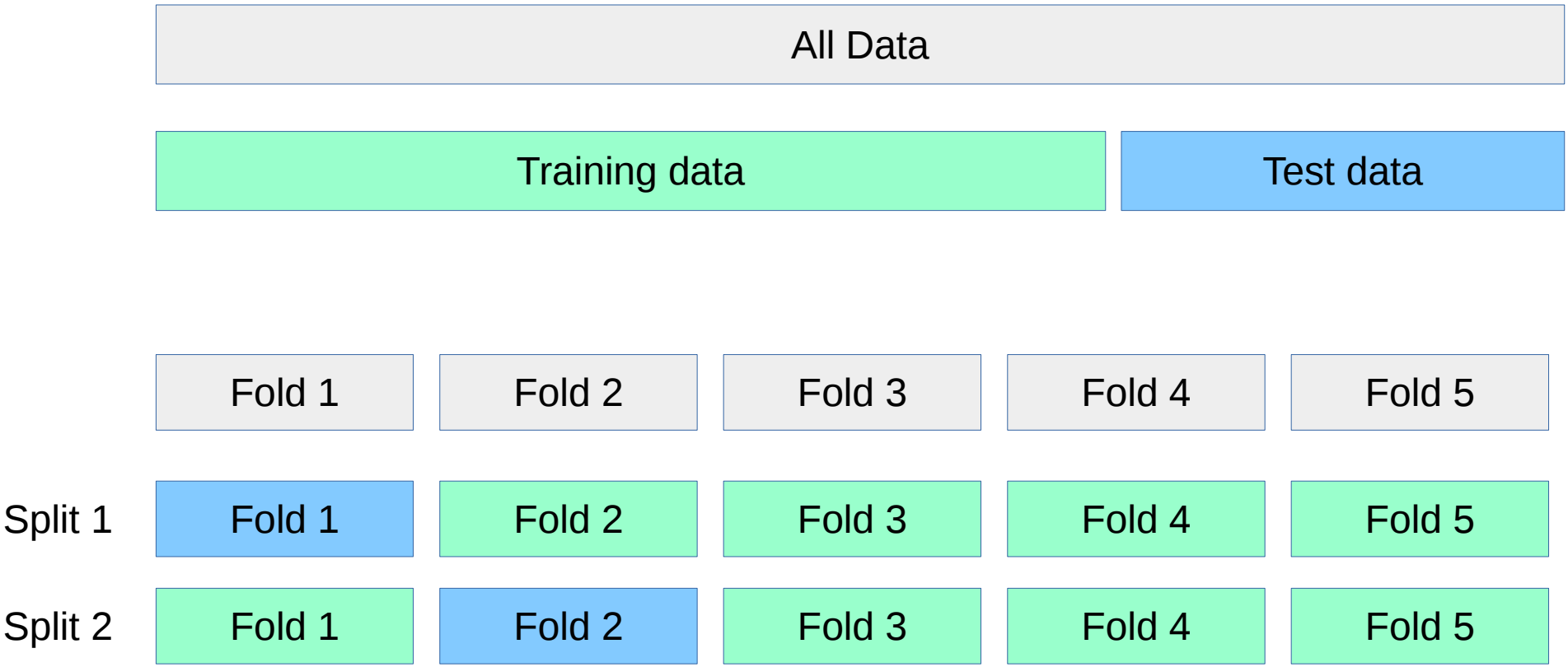
`estimator.fit(X, [y])`

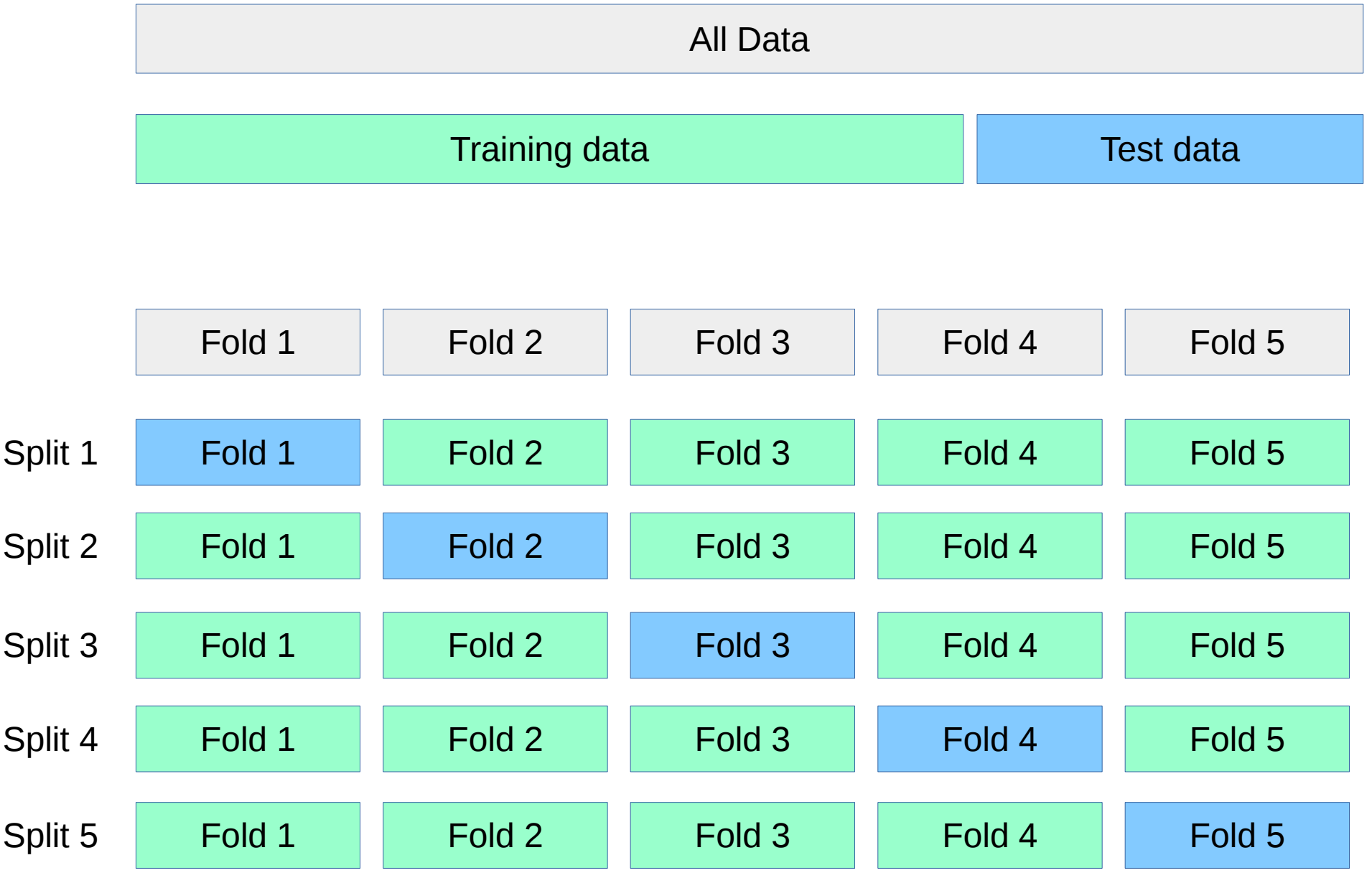
`estimator.predict`	**`estimator.transform`**
Classification	Preprocessing
Regression	Dimensionality reduction
Clustering	Feature selection
	Feature extraction


All Data
Training data
Test data


All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5

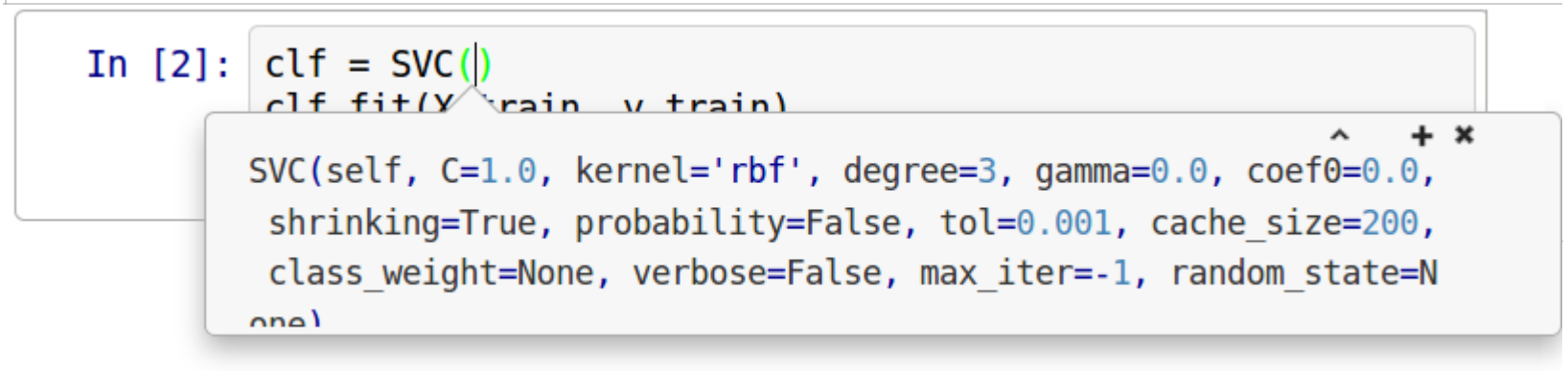

All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5


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

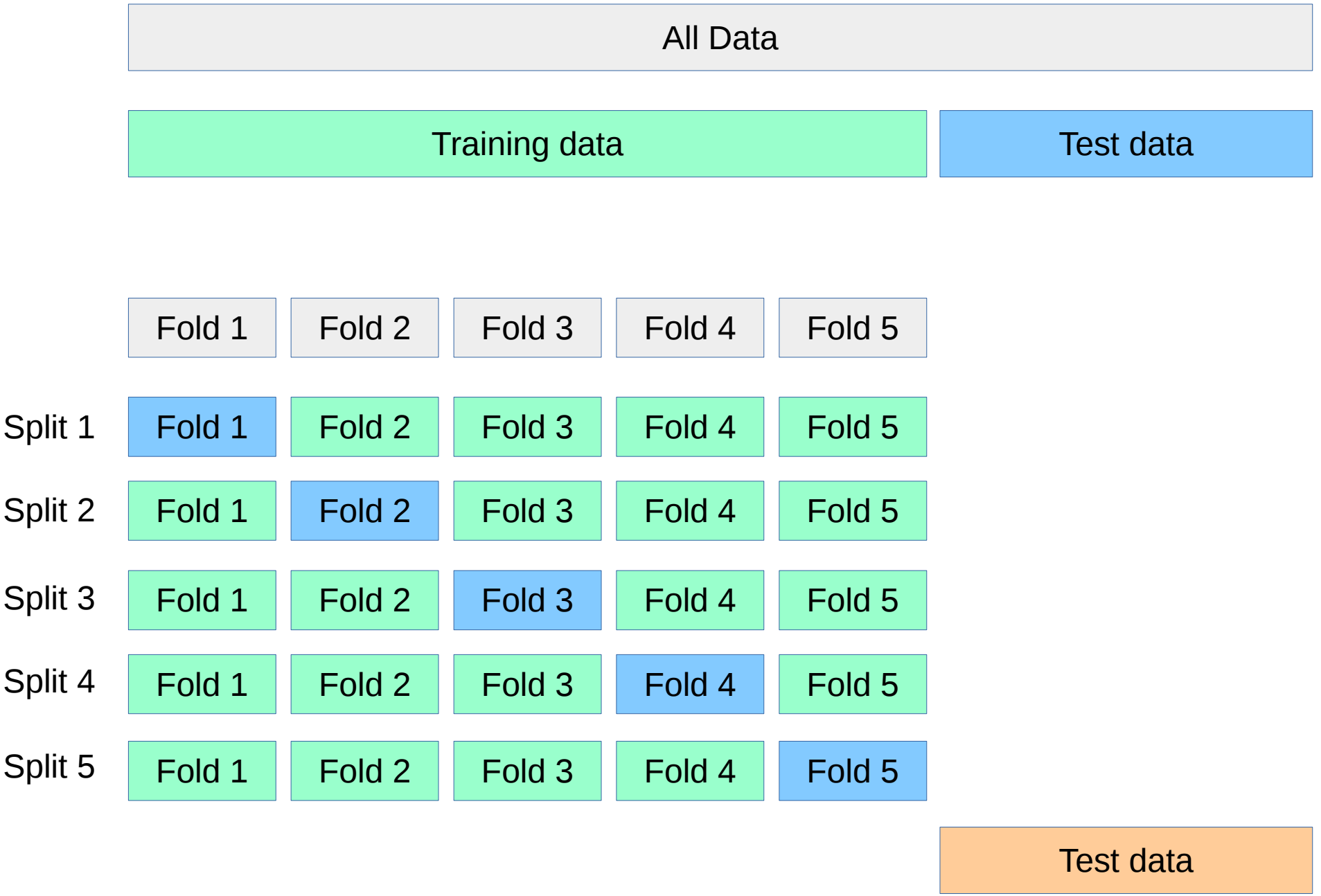

All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 2
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 3
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 4
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 5
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5

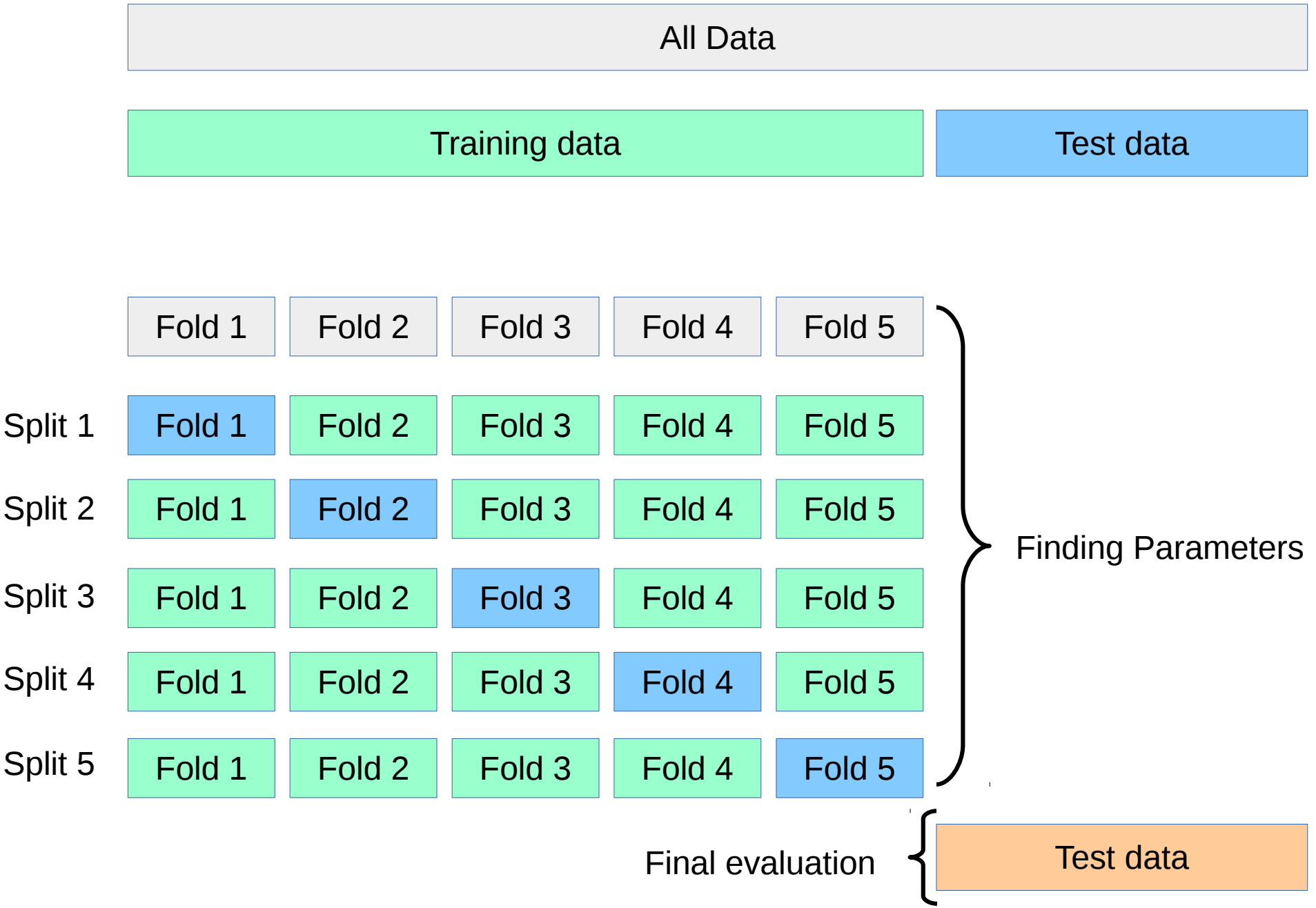
IPython Notebook:
Part 3 - Cross-validation

In [2]:
```
clf = SVC()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
```


In [2]:
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=N
one)


All Data
Training data
Test data


All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Split 2
Split 3
Split 4
Split 5
Test data


All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Split 2
Split 3
Split 4
Split 5
Finding Parameters
Final evaluation
Test data

SVC(C=0.001,
gamma=0.001)

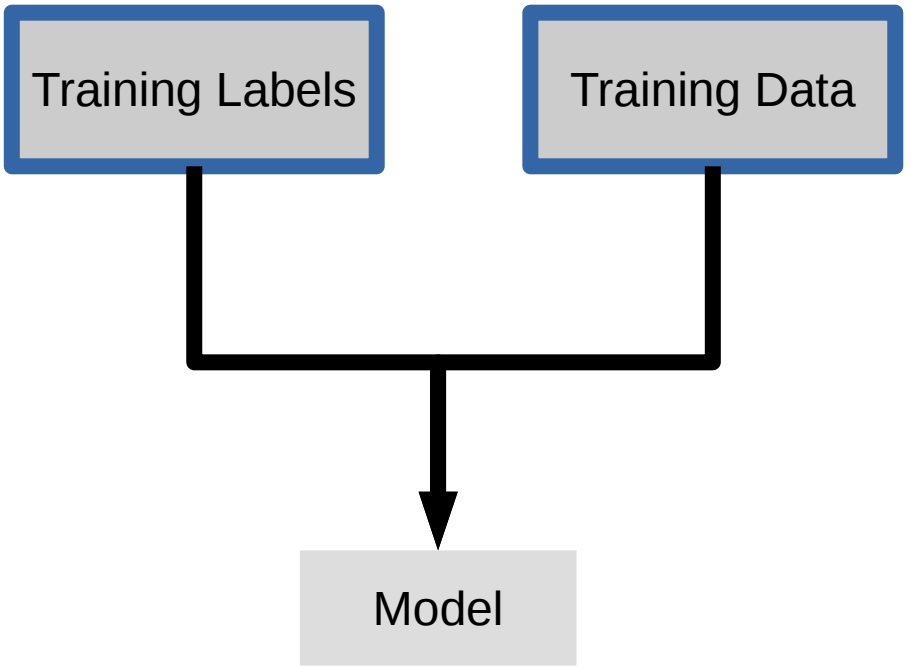
SVC(C=0.001, gamma=0.001)	SVC(C=0.01, gamma=0.001)	SVC(C=0.1, gamma=0.001)	SVC(C=1, gamma=0.001)	SVC(C=10, gamma=0.001)

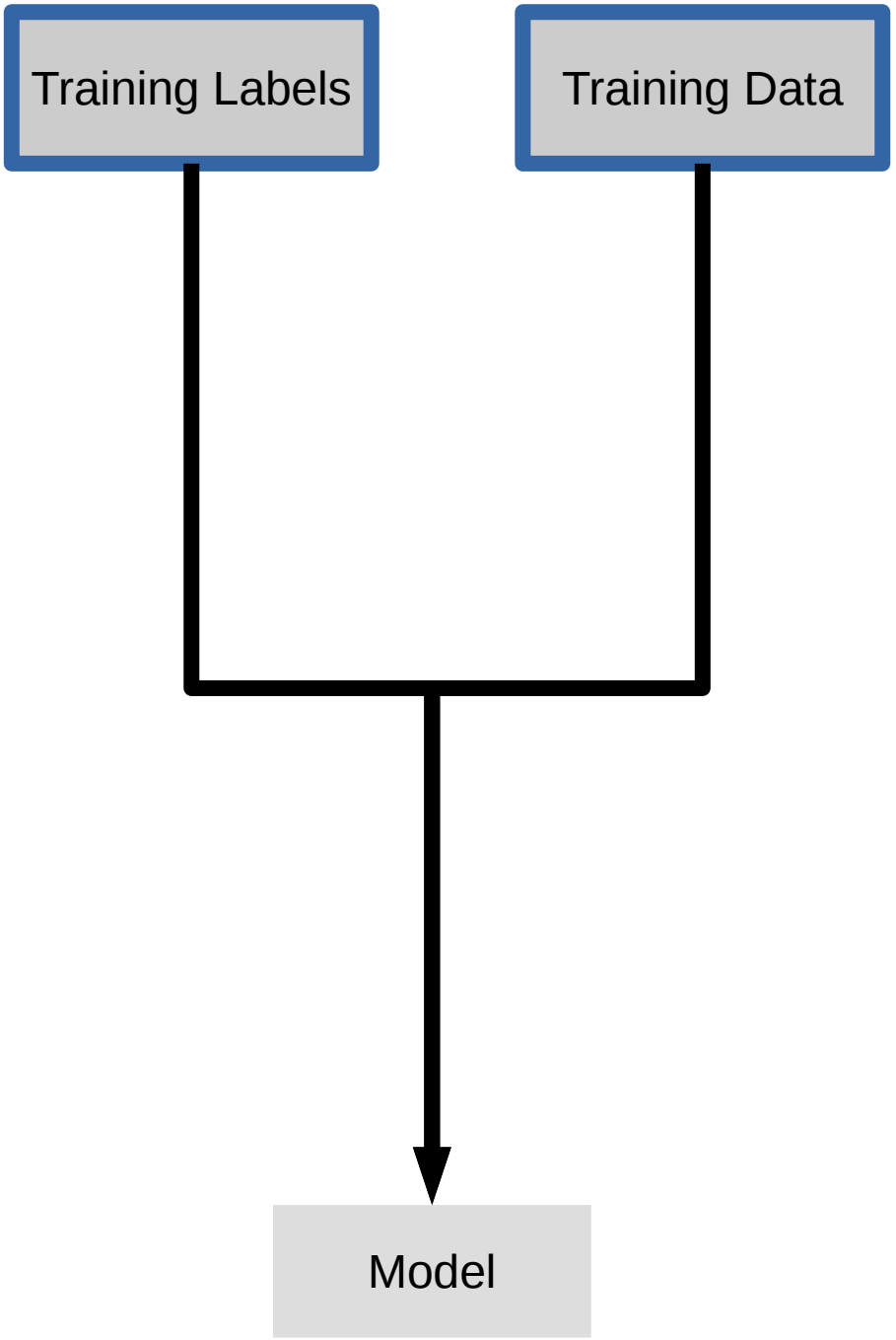
SVC(C=0.001, gamma=0.001)	SVC(C=0.01, gamma=0.001)	SVC(C=0.1, gamma=0.001)	SVC(C=1, gamma=0.001)	SVC(C=10, gamma=0.001)
SVC(C=0.001, gamma=0.01)	SVC(C=0.01, gamma=0.01)	SVC(C=0.1, gamma=0.01)	SVC(C=1, gamma=0.01)	SVC(C=10, gamma=0.01)

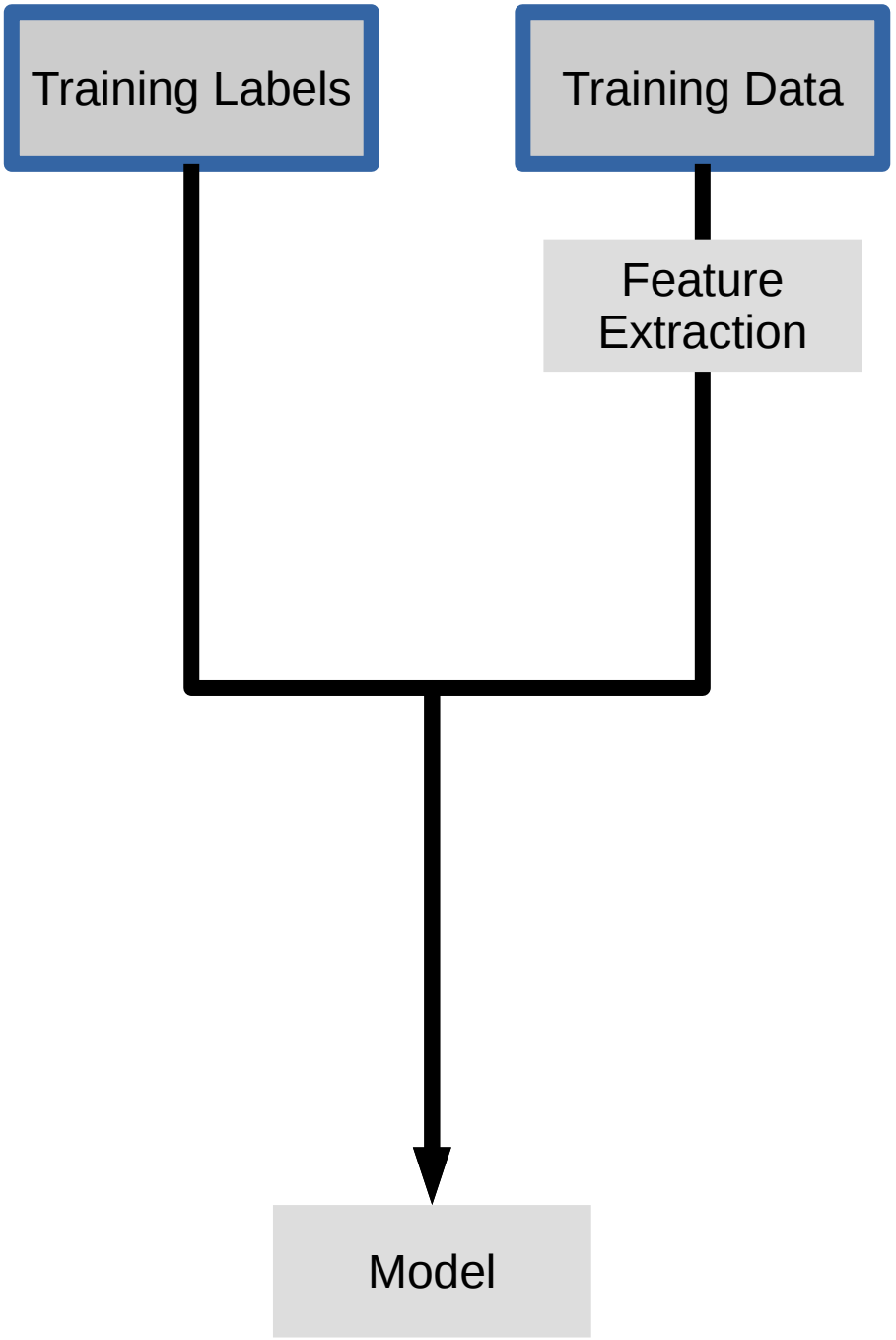
SVC(C=0.001, gamma=0.001)	SVC(C=0.01, gamma=0.001)	SVC(C=0.1, gamma=0.001)	SVC(C=1, gamma=0.001)	SVC(C=10, gamma=0.001)
SVC(C=0.001, gamma=0.01)	SVC(C=0.01, gamma=0.01)	SVC(C=0.1, gamma=0.01)	SVC(C=1, gamma=0.01)	SVC(C=10, gamma=0.01)
SVC(C=0.001, gamma=0.1)	SVC(C=0.01, gamma=0.1)	SVC(C=0.1, gamma=0.1)	SVC(C=1, gamma=0.1)	SVC(C=10, gamma=0.1)

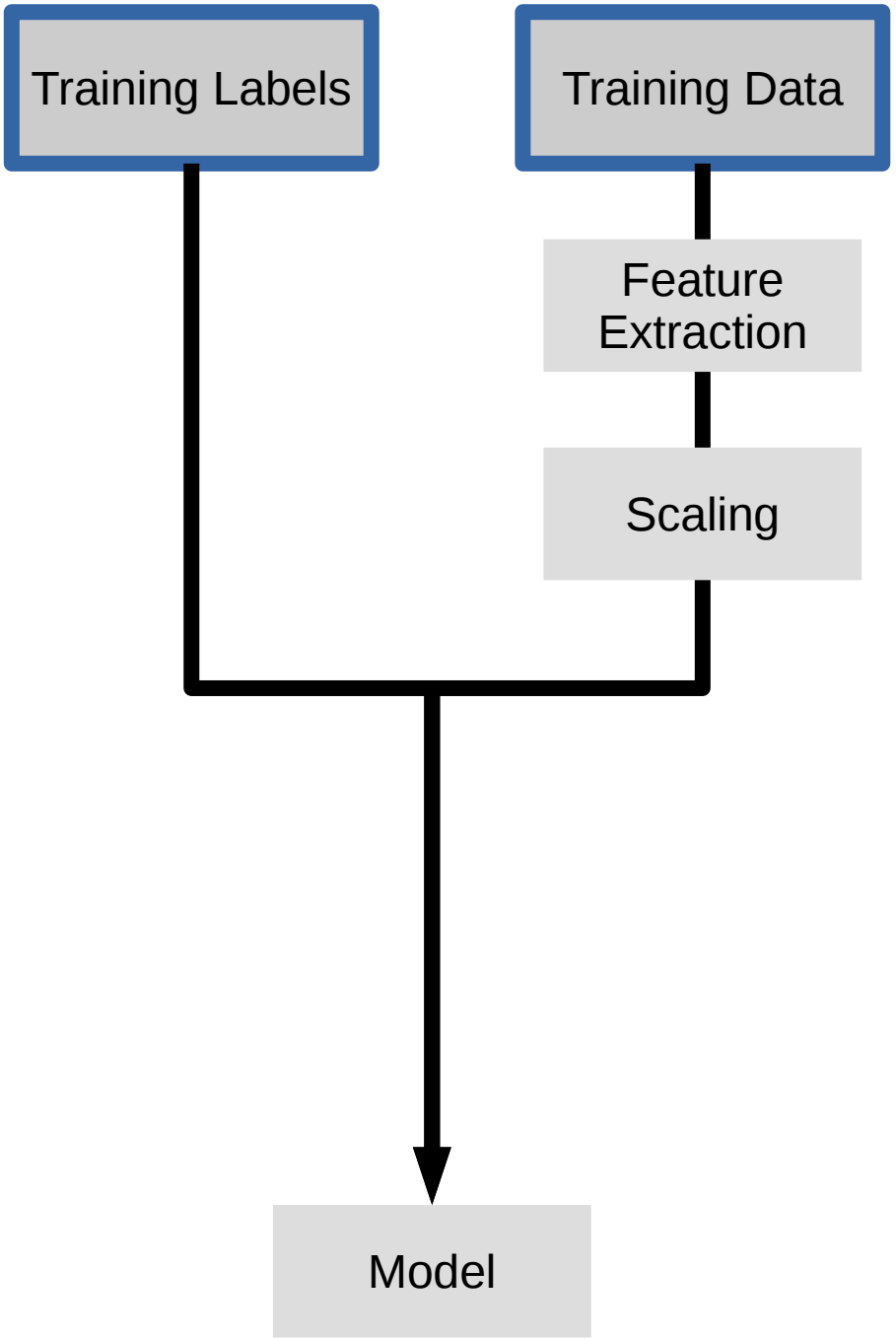
SVC(C=0.001, gamma=0.001)	SVC(C=0.01, gamma=0.001)	SVC(C=0.1, gamma=0.001)	SVC(C=1, gamma=0.001)	SVC(C=10, gamma=0.001)
SVC(C=0.001, gamma=0.01)	SVC(C=0.01, gamma=0.01)	SVC(C=0.1, gamma=0.01)	SVC(C=1, gamma=0.01)	SVC(C=10, gamma=0.01)
SVC(C=0.001, gamma=0.1)	SVC(C=0.01, gamma=0.1)	SVC(C=0.1, gamma=0.1)	SVC(C=1, gamma=0.1)	SVC(C=10, gamma=0.1)
SVC(C=0.001, gamma=1)	SVC(C=0.01, gamma=1)	SVC(C=0.1, gamma=1)	SVC(C=1, gamma=1)	SVC(C=10, gamma=1)
SVC(C=0.001, gamma=10)	SVC(C=0.01, gamma=10)	SVC(C=0.1, gamma=10)	SVC(C=1, gamma=10)	SVC(C=10, gamma=10)

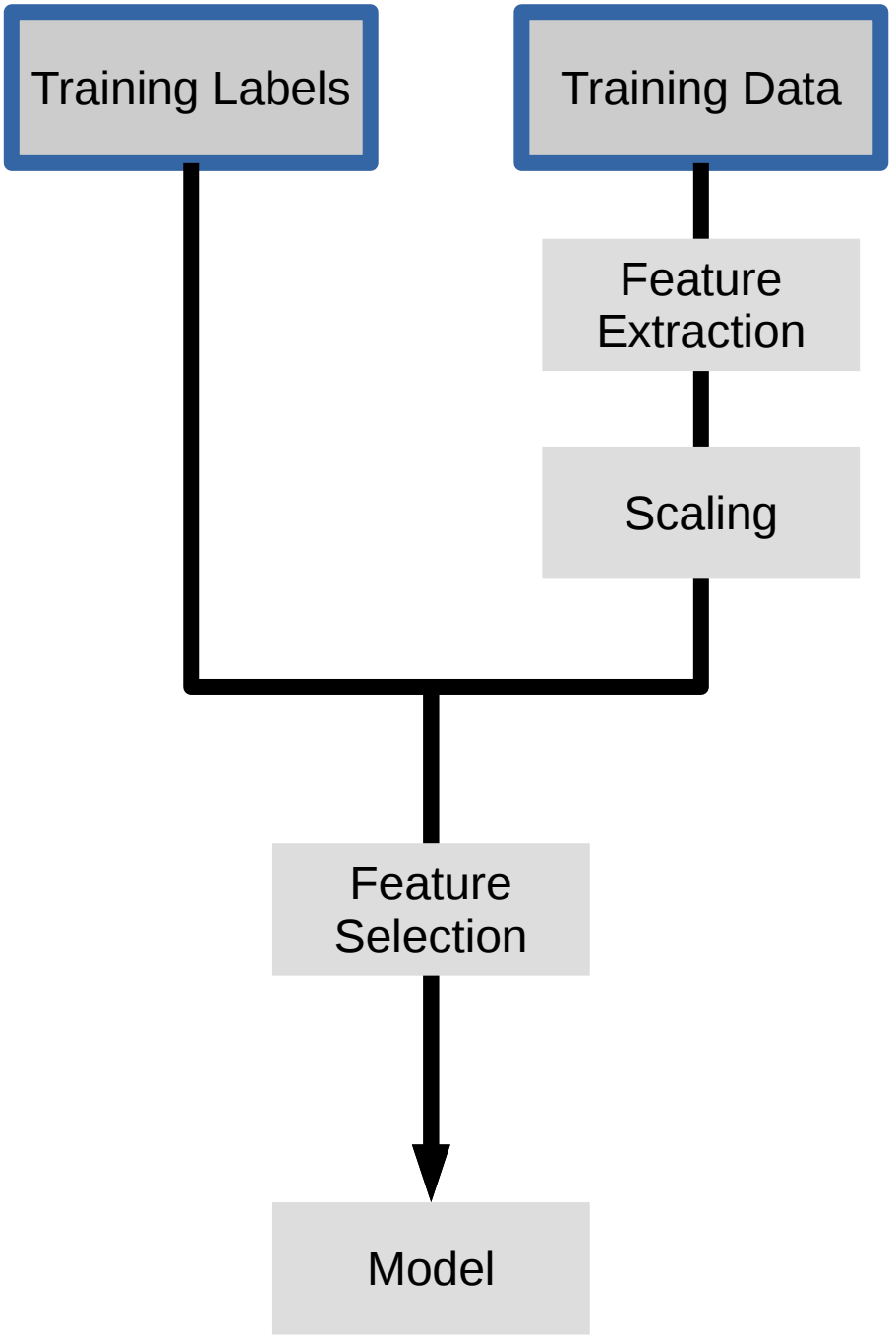
IPython Notebook:
Part 4 – Grid Searches

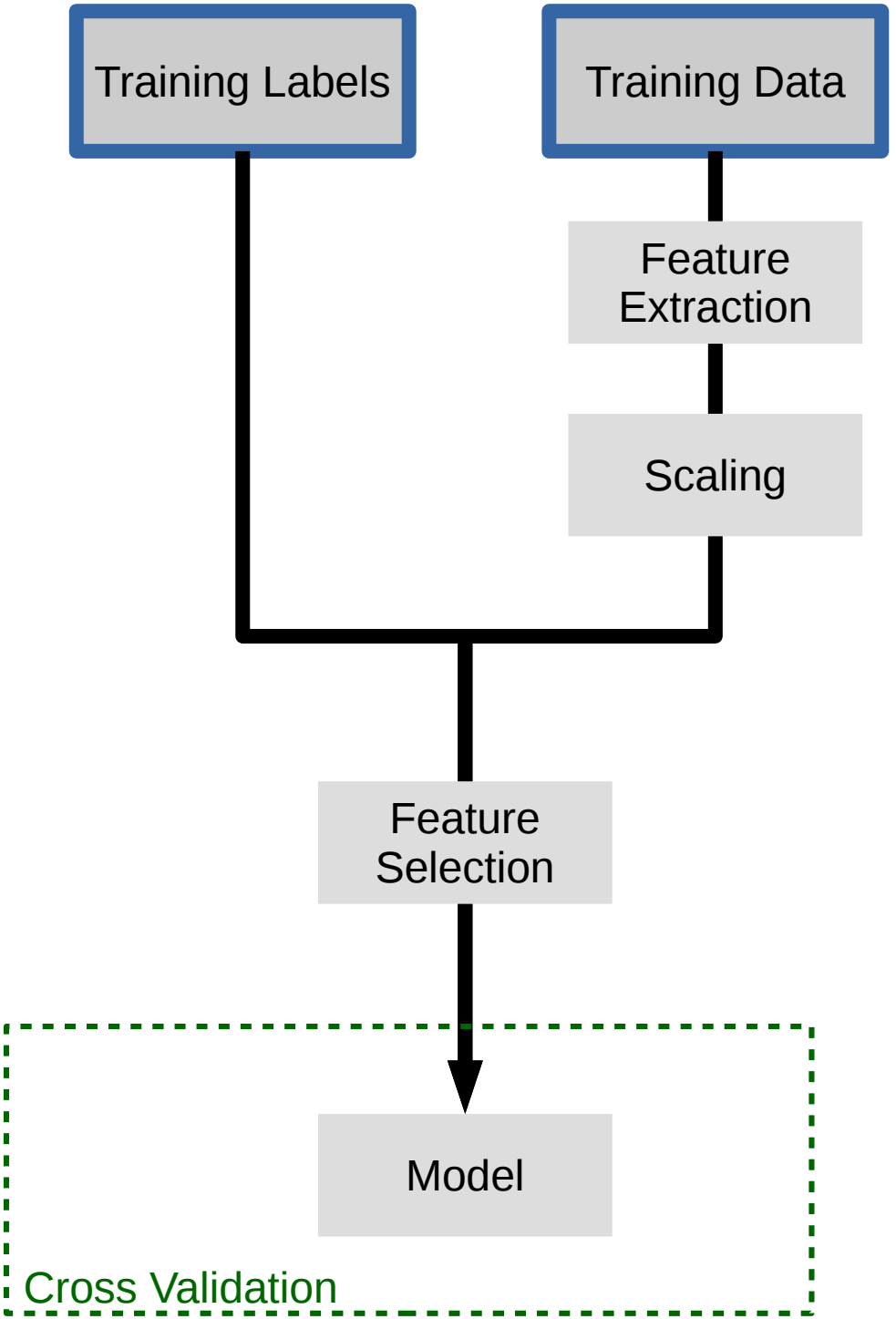

Training Labels
Training Data
Model

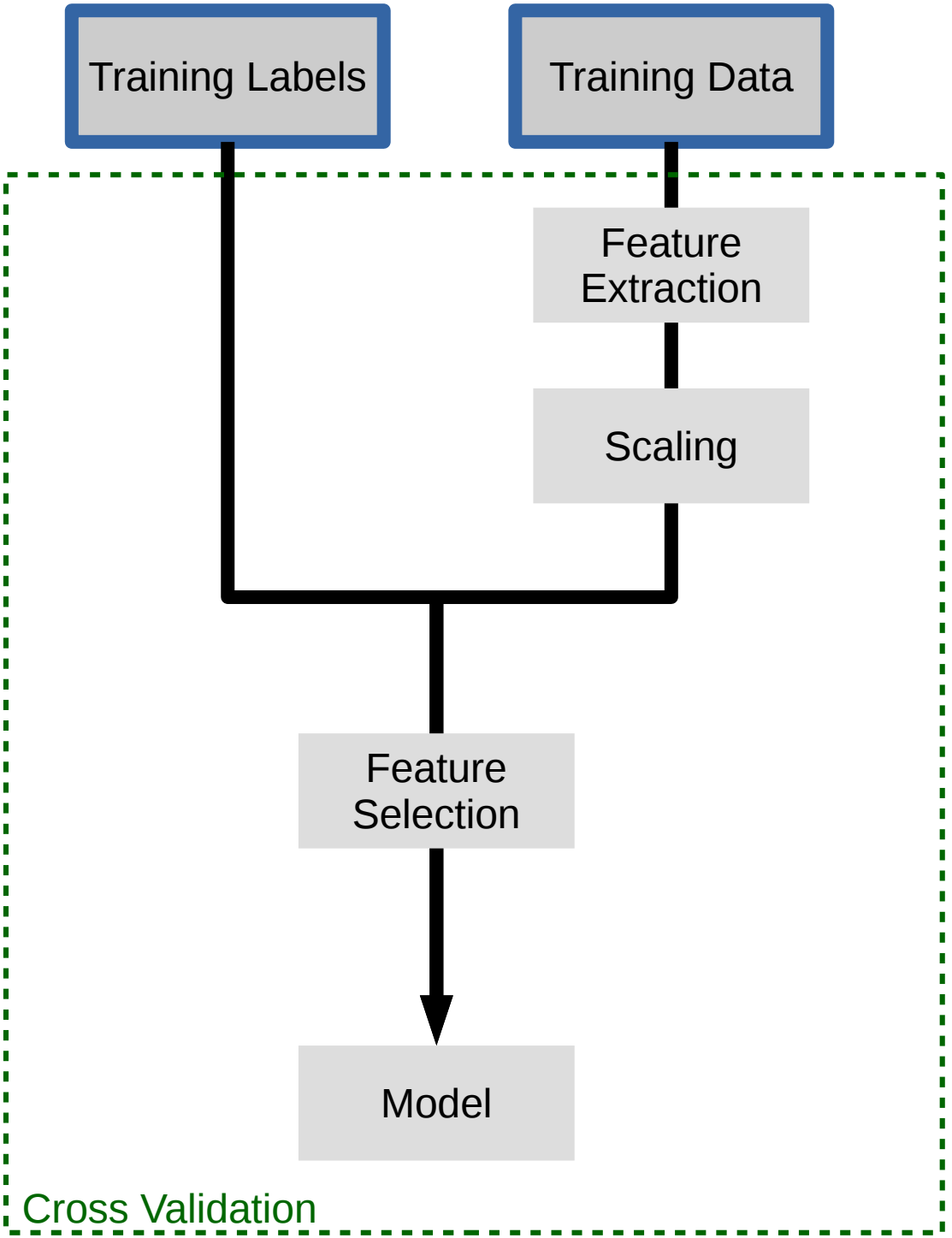

Training Labels
Training Data
Model


Training Labels
Training Data
Feature Extraction
Model


Training Labels
Training Data
Feature Extraction
Scaling
Model

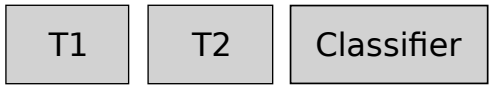

Training Labels
Training Data
Feature Extraction
Scaling
Feature Selection
Model


Training Labels
Training Data
Feature Extraction
Scaling
Feature Selection
Model
Cross Validation

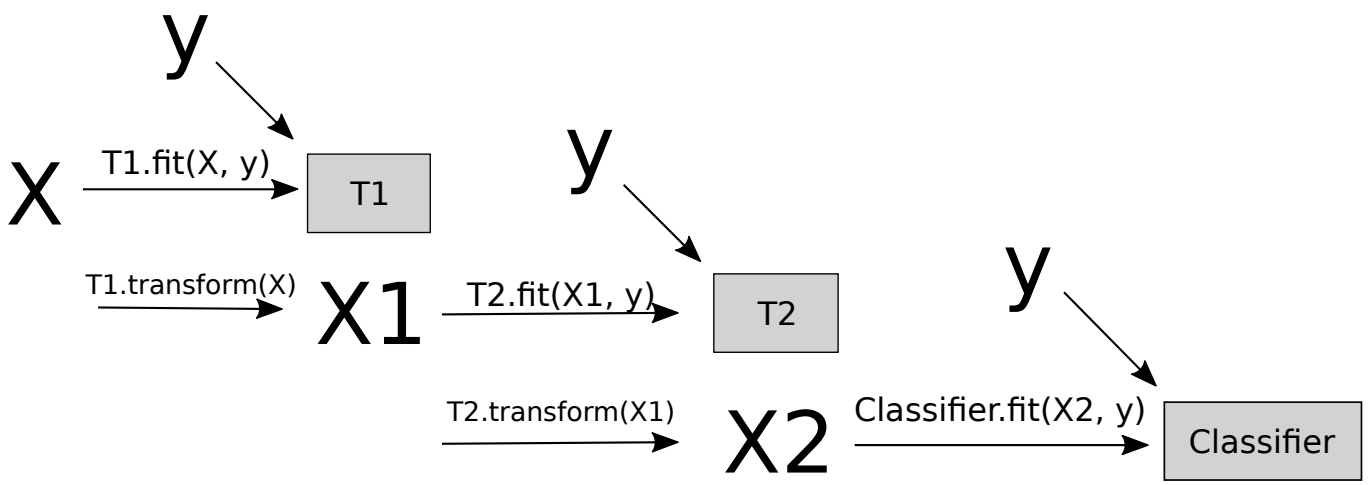

Training Labels
Training Data
Feature Extraction
Scaling
Feature Selection
Model
Cross Validation

Pipelines

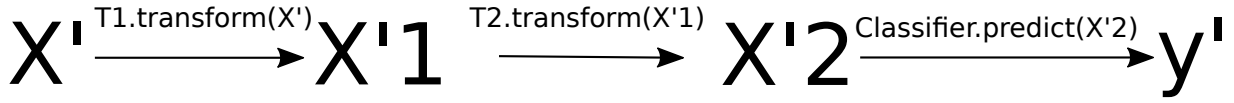
pipe = make_pipeline(T1(), T2(), Classifier())



pipe.fit(X, y)



pipe.predict(X')



IPython Notebook:
Part 5 - Preprocessing and Pipelines

Do cross-validation and parameter selection
over all steps jointly.
Keep a separate test set until the very end.

If you are not using pipelines,
your probably doing it wrong.

Sample application: Sentiment Analysis

IMDB Movie Reviews Data

Review:
One of the worst movies I've ever rented. Sorry it had one of my favorite actors on it (Travolta) in a nonsense role. In fact, anything made sense in this movie.

Who can say there was true love between Eddy and Maureen? Don't you remember the beginning of the movie ?

Is she so lovely? Ask her daughters. I don't think so.

Label: negative

Training data: 12500 positive, 12500 negative

Bag Of Word Representations

`CountVectorizer / TfidfVectorizer`

Bag Of Word Representations

`CountVectorizer / TfidfVectorizer`

`“This is how you get ants.”`

Bag Of Word Representations

`CountVectorizer / TfidfVectorizer`

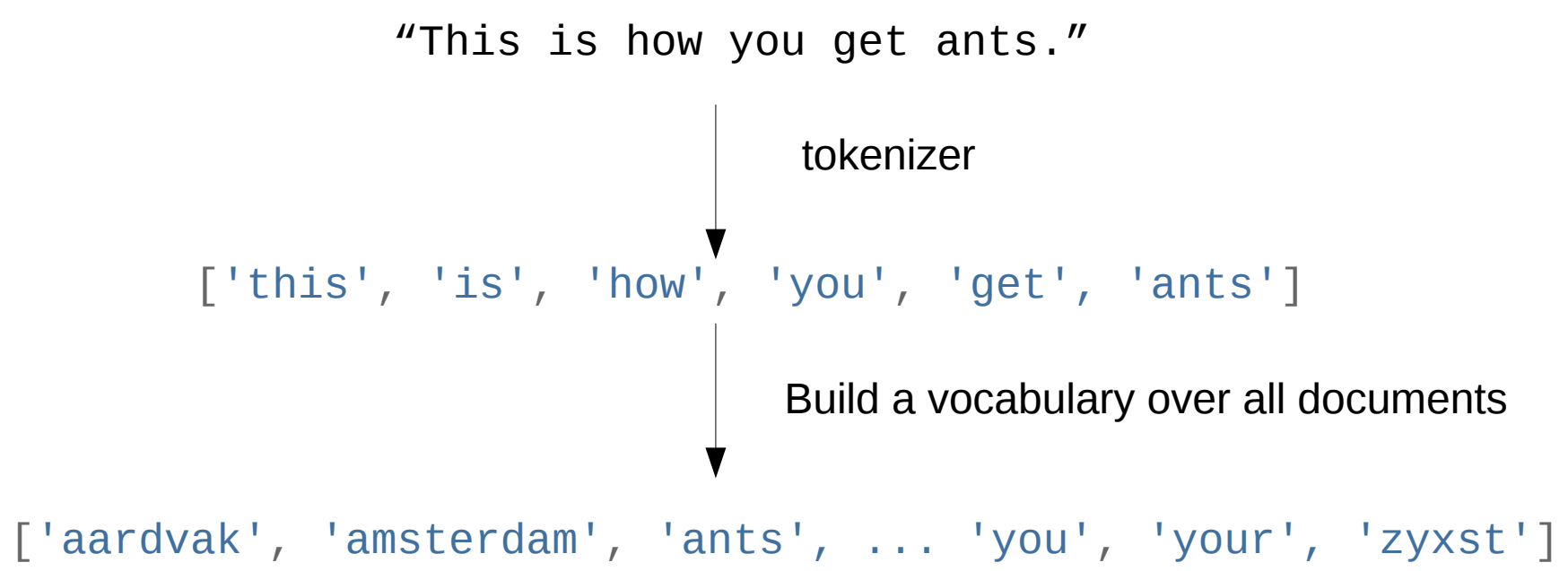
`“This is how you get ants.”`

tokenizer

```
['this', 'is', 'how', 'you', 'get', 'ants']
```

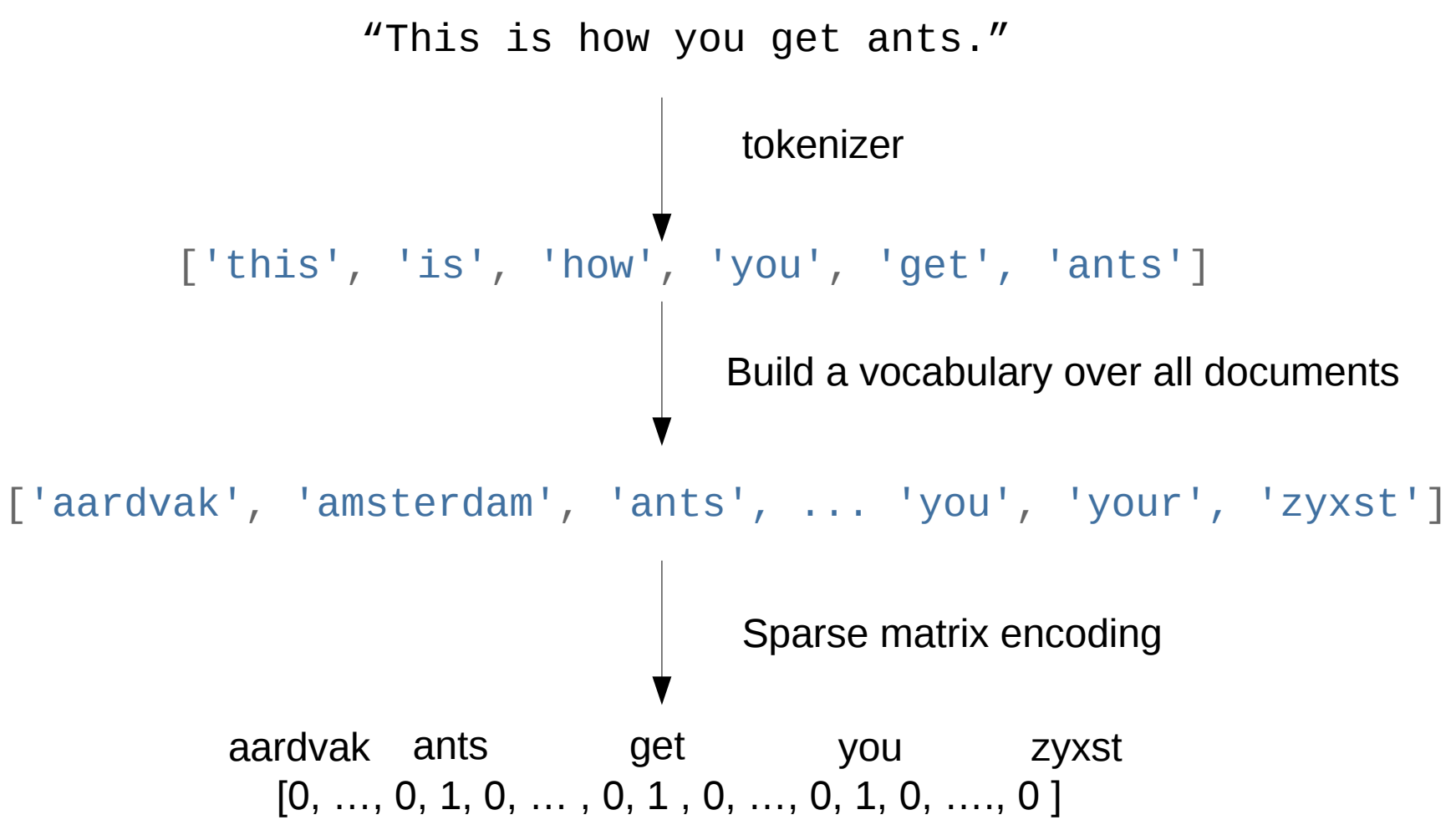
Bag Of Word Representations

`CountVectorizer / TfidfVectorizer`



Bag Of Word Representations

`CountVectorizer / TfidfVectorizer`



IPython Notebook:
Part 6 - Working With Text Data

Scaling Up

Three regimes of data

- Fits in RAM
- Fits on a Hard Drive
- Doesn't fit on a single PC

Three regimes of data

- Fits in RAM (up to 256 GB?)
- Fits on a Hard Drive (up to 6TB?)
- Doesn't fit on a single PC

Nobody ever got fired for using Hadoop on a cluster

Antony Rowstron, Dushyanth Narayanan, Austin Donnelly, Greg O'Shea, and Andrew Douglas

10 April 2012

	vCPU	ECU	Memory (GiB)	Instance Storage (GB)	Linux/UNIX Usage
Memory Optimized - Current Generation					
r3.large	2	6.5	15	1 x 32 SSD	$0.195 per Hour
r3.xlarge	4	13	30.5	1 x 80 SSD	$0.39 per Hour
r3.2xlarge	8	26	61	1 x 160 SSD	$0.78 per Hour
r3.4xlarge	16	52	122	1 x 320 SSD	$1.56 per Hour
r3.8xlarge	32	104	244	2 x 320 SSD	$3.12 per Hour
Storage Optimized - Current Generation					
i2.xlarge	4	14	30.5	1 x 800 SSD	$0.938 per Hour
i2.2xlarge	8	27	61	2 x 800 SSD	$1.876 per Hour
i2.4xlarge	16	53	122	4 x 800 SSD	$3.751 per Hour
i2.8xlarge	32	104	244	8 x 800 SSD	$7.502 per Hour

"256Gb ought to be enough for anybody."

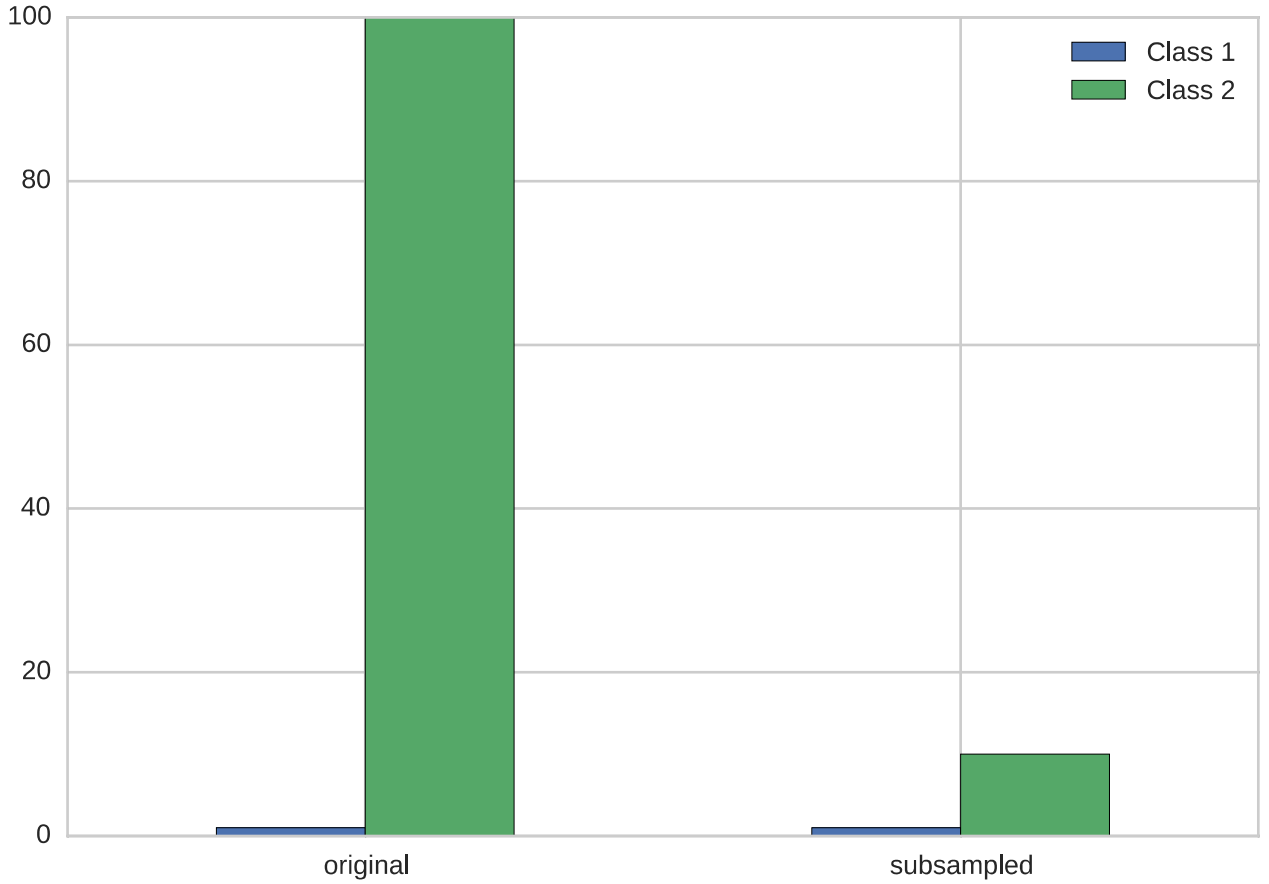
- me

"256Gb ought to be enough for anybody."
- me

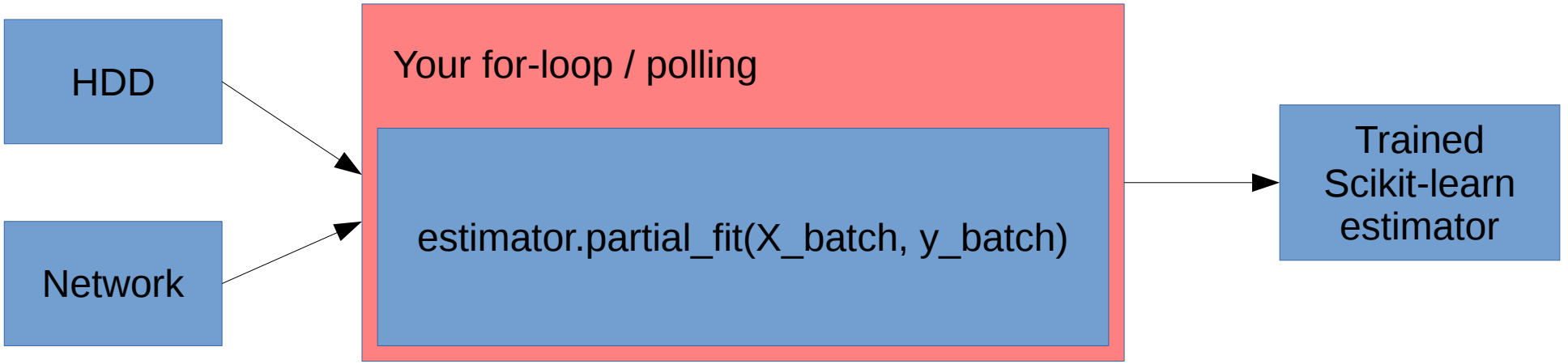
(for machine learning)

Subsample!

Subsample!



The scikit-learn way


HDD
Network
Your for-loop / polling
estimator.partial_fit(X_batch, y_batch)
Trained
Scikit-learn
estimator

Supported Algorithms

- All `SGDClassifier` derivatives
- Naive Bayes
- `MinibatchKMeans`
- `Birch`
- `IncrementalPCA`
- `MiniBatchDictionaryLearning`
- `...`

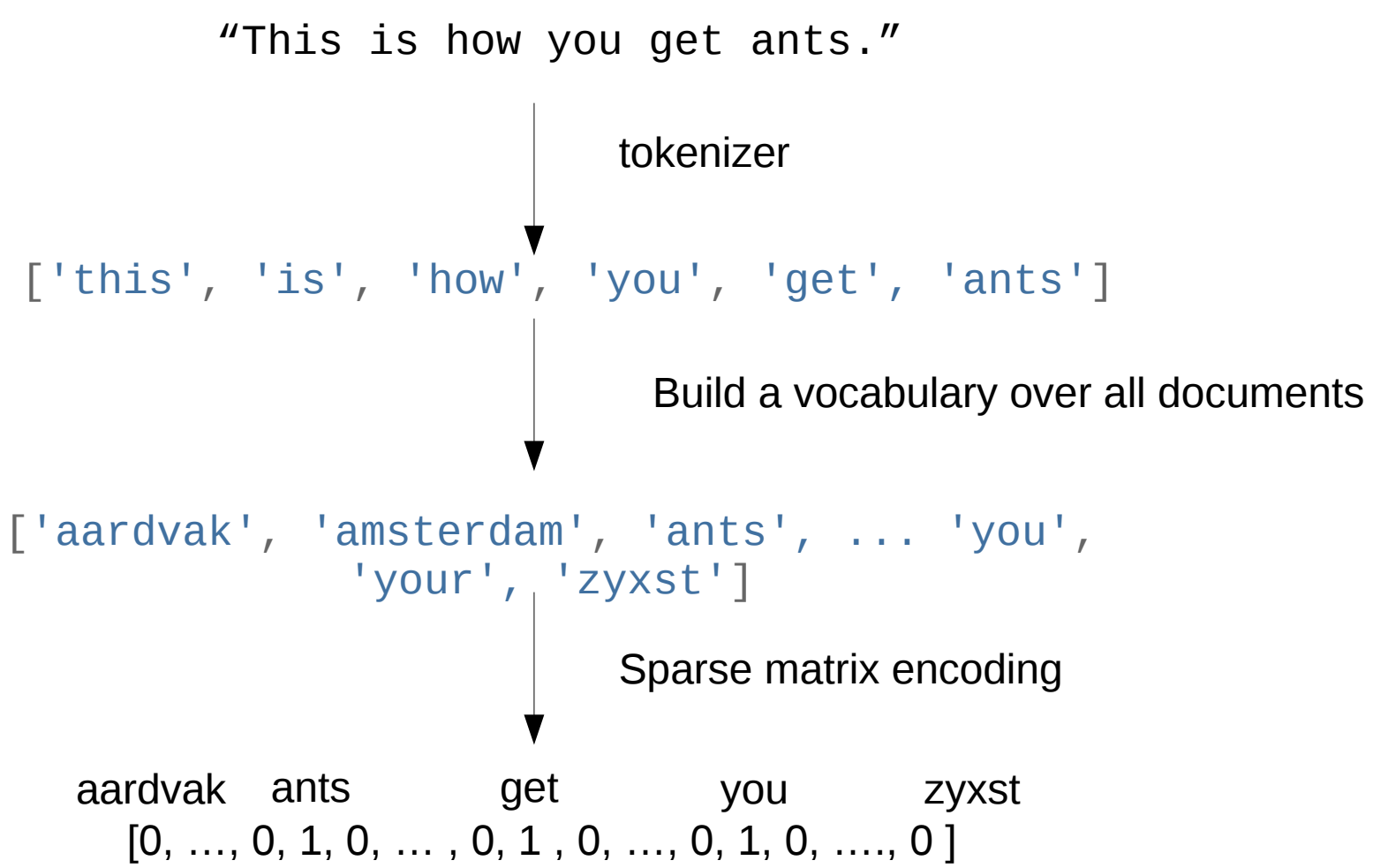
IPython Notebook:
Part 8 – Out Of Core Learning

Stateless Transformers

- `Normalizer`
- `HashingVectorizer`
- `RBFSampler` (and other kernel approx)

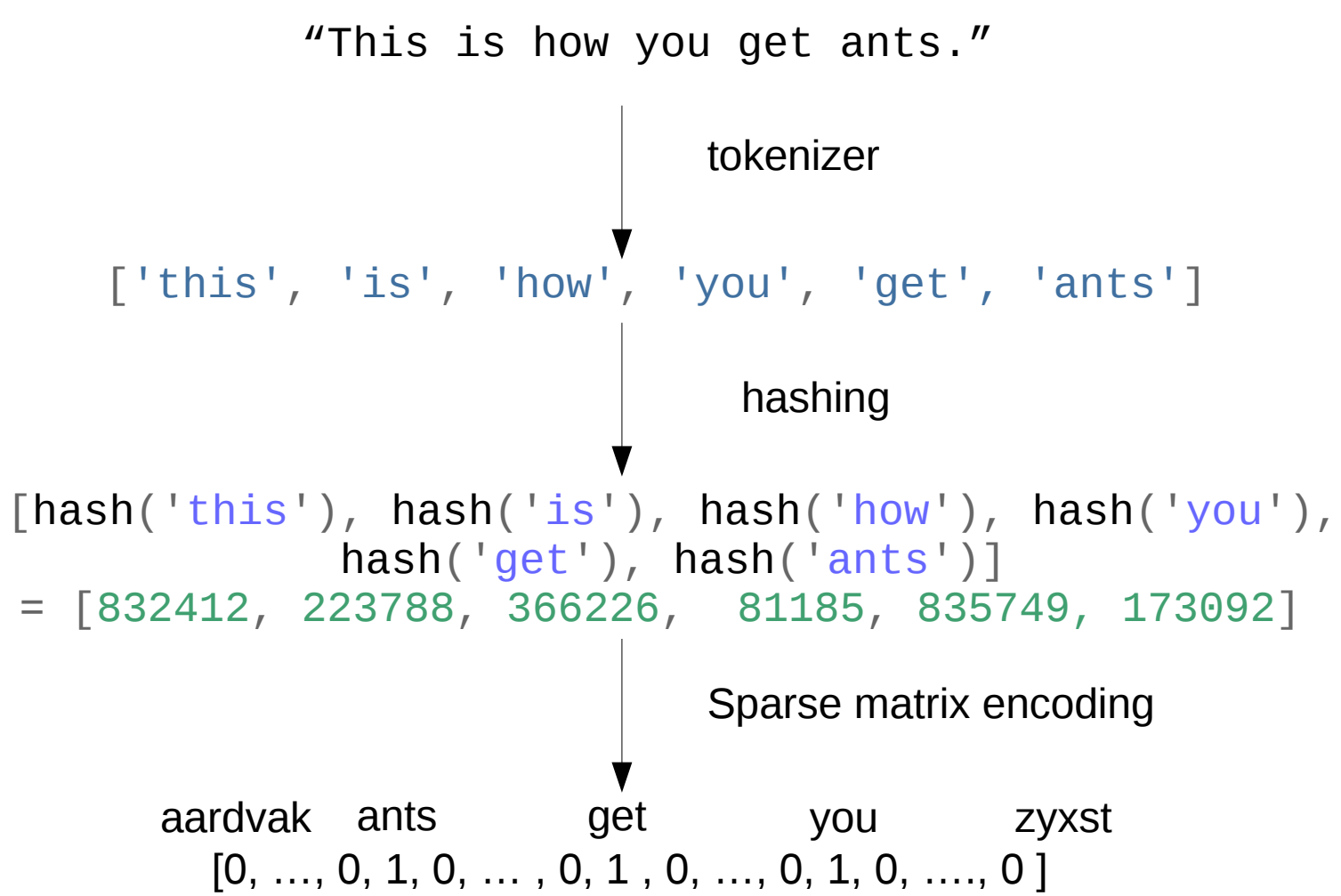
Bag Of Word Representations

`CountVectorizer / TfidfVectorizer`



Hashing Trick

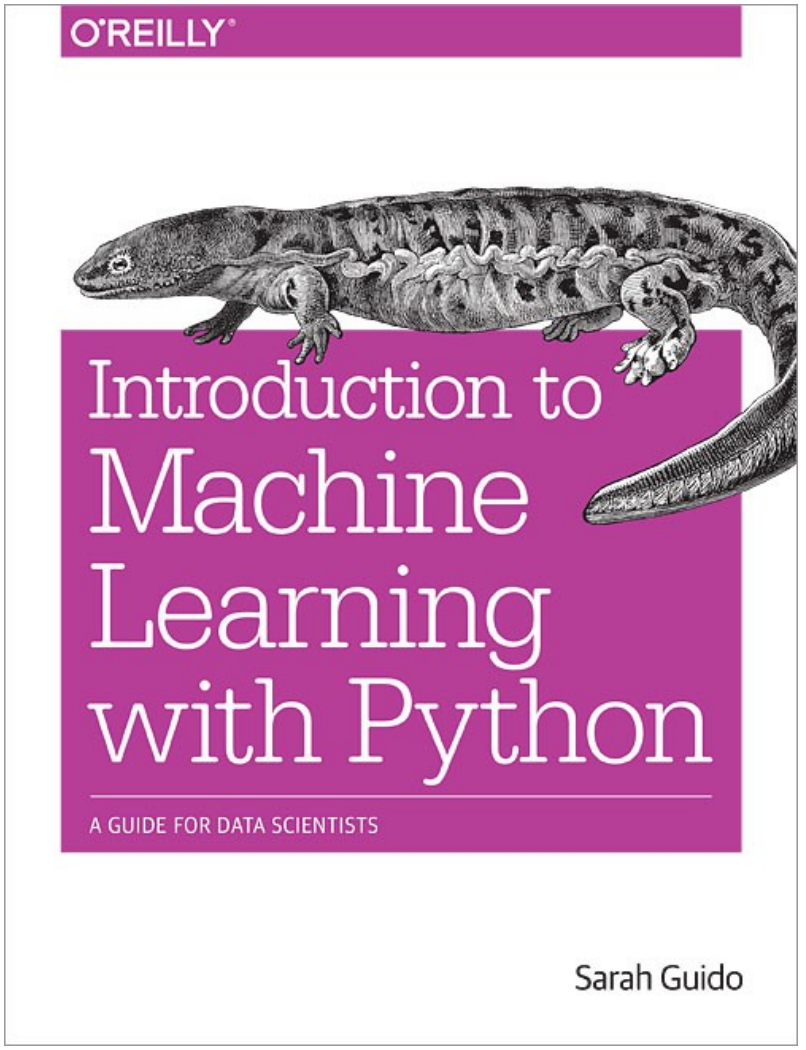
`HashingVectorizer`



IPython Notebook:
Part 9 – Out Of Core Learning for Text

Video Series
Advanced Machine Learning with scikit-learn

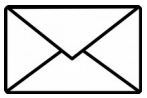
Video Series
Advanced Machine Learning with scikit-learn



Thank you for your attention.


@amuellerml


@amueller


importamueller@gmail.com


http://amueller.github.io