scikit-learn
From Zero to Hero!

Alexandre Gramfort

http://alexandre.gramfort.net

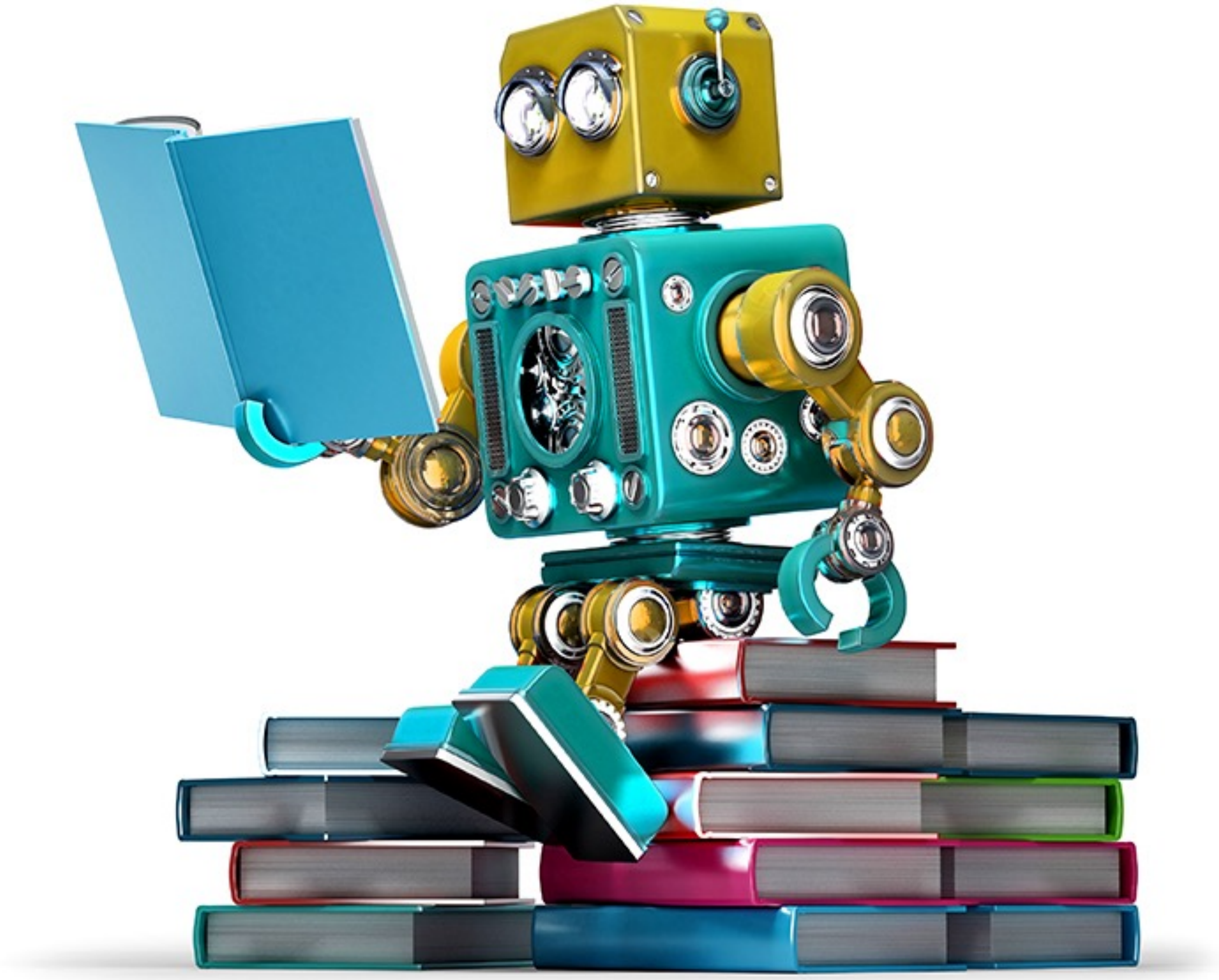
https://github.com/agramfort/blend2017

GitHub : @agramfort

Twitter : @agramfort

scikit learn is a ML library in python

but what is machine learning?



Artificial Intelligence

IBM Watson

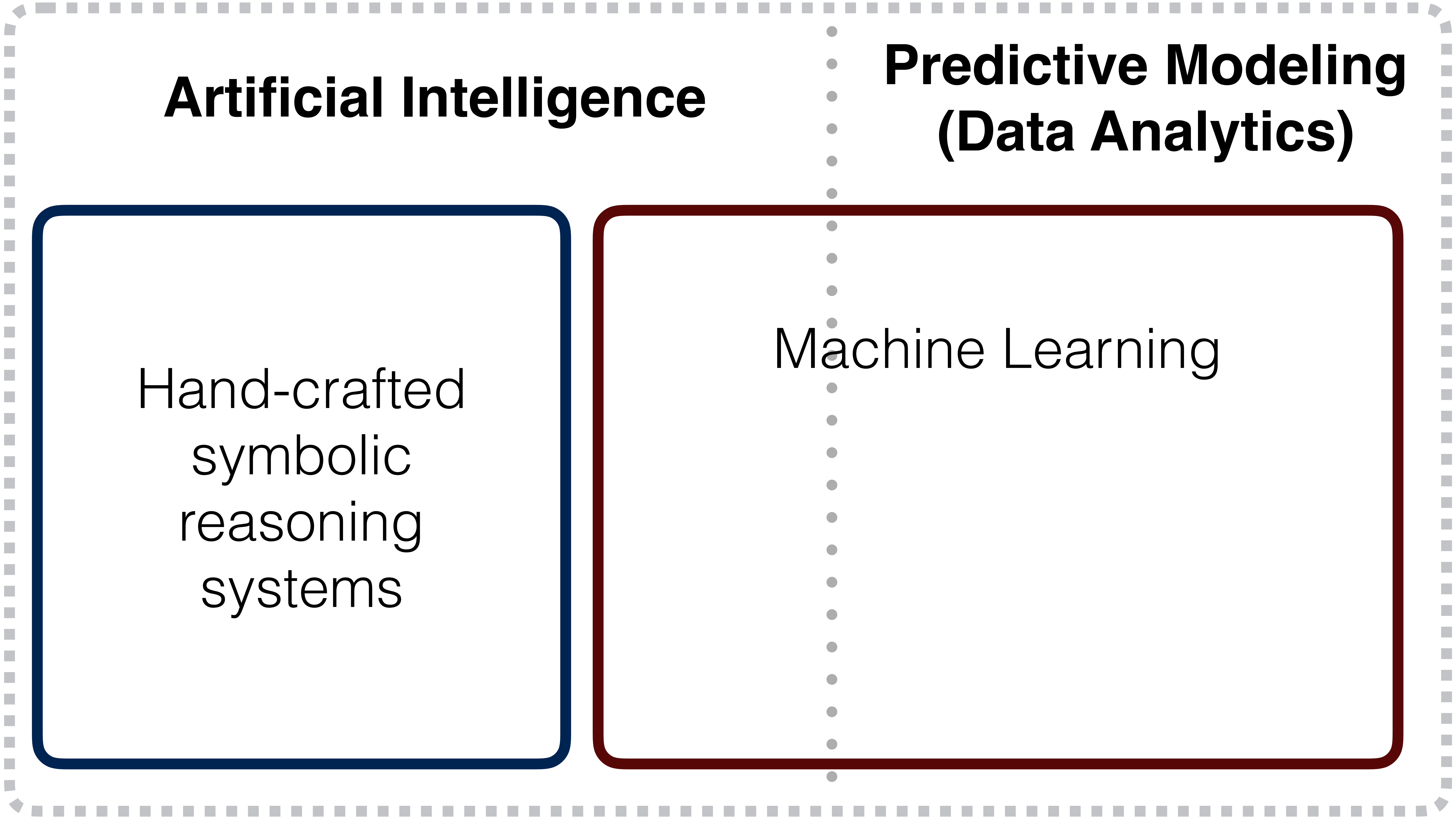
Self-driving cars

Predictive Modeling (Data Analytics)

Movie recommendations

Predictive Maintenance

Slide courtesy of @ogrisel



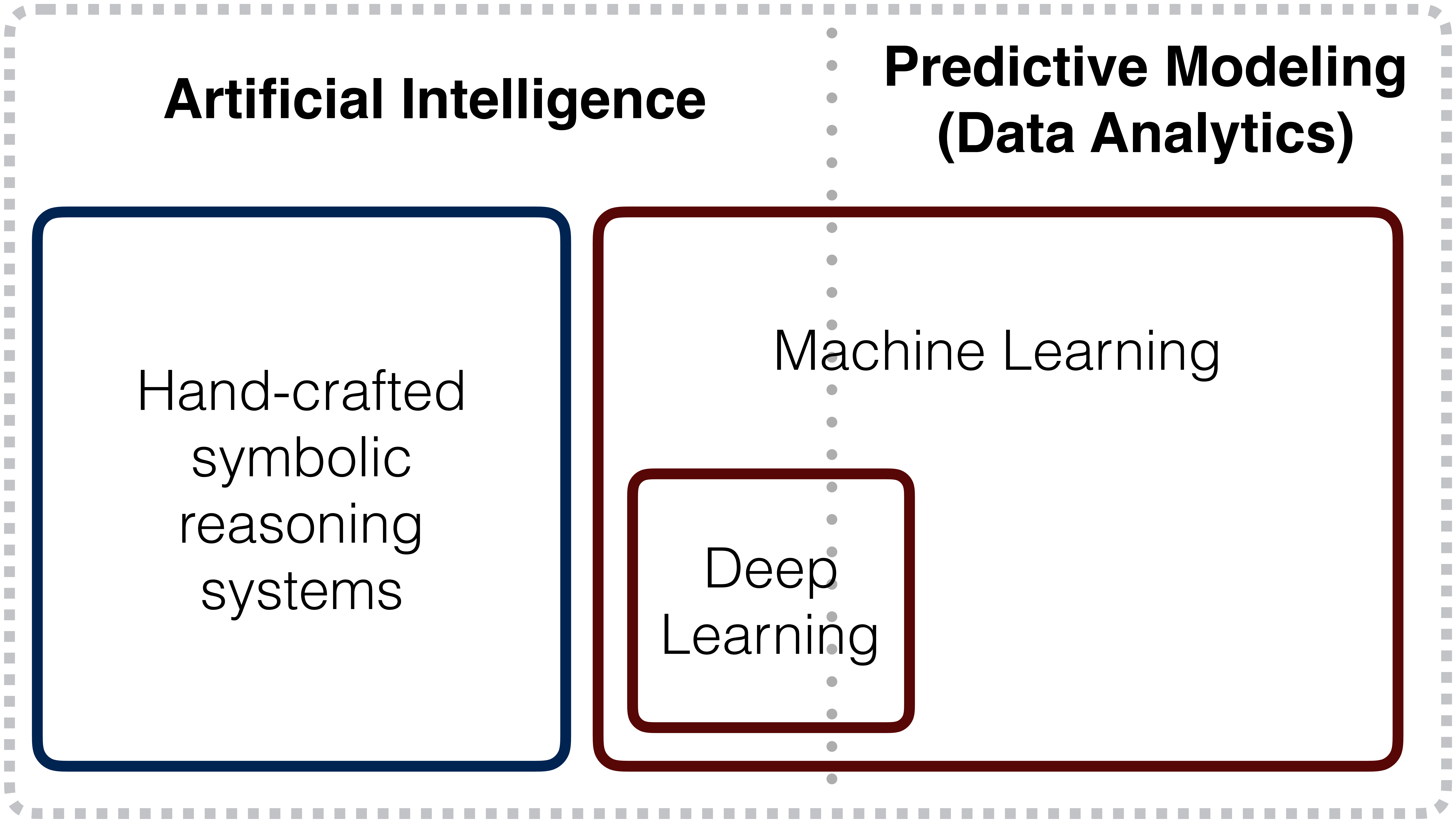
Artificial Intelligence

Predictive Modeling (Data Analytics)

Hand-crafted symbolic reasoning systems

Machine Learning

Slide courtesy of @ogrisel



Artificial Intelligence

Predictive Modeling
(Data Analytics)

Hand-crafted symbolic reasoning systems

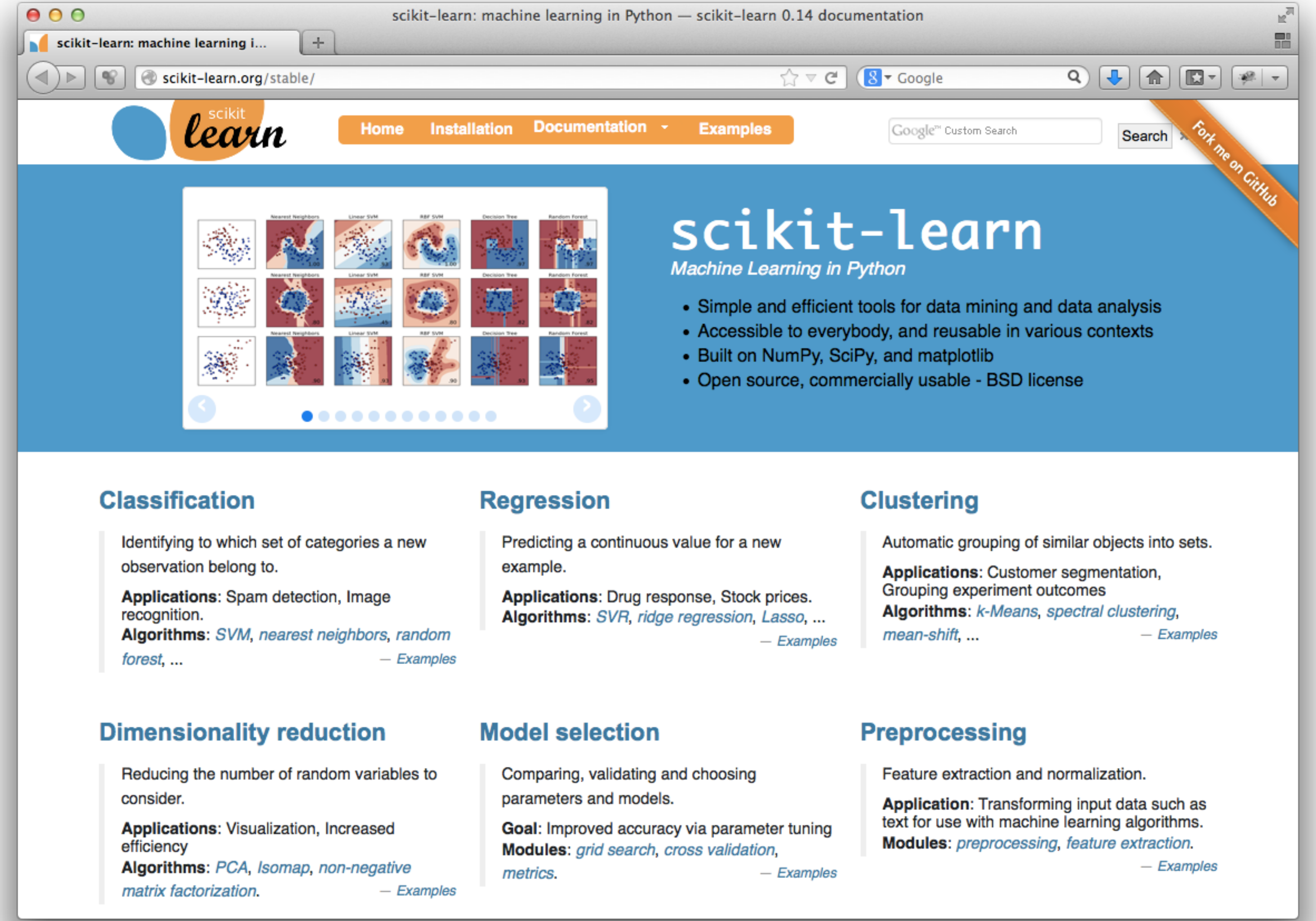
Machine Learning

Deep Learning

Slide courtesy of @ogrisel

Definition: Machine learning consists in teaching a computer to make decisions based on examples

machine learning is one (very effective) way to solve AI problems



scikit-learn

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

Classification

Identifying to which set of categories a new observation belong to.

Applications: Spam detection, Image recognition.
Algorithms: *SVM*, *nearest neighbors*, *random forest*, ... — *Examples*

Regression

Predicting a continuous value for a new example.

Applications: Drug response, Stock prices.
Algorithms: *SVR*, *ridge regression*, *Lasso*, ... — *Examples*

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: *k-Means*, *spectral clustering*, *mean-shift*, ... — *Examples*

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: *PCA*, *Isomap*, *non-negative matrix factorization*. — *Examples*

Model selection

Comparing, validating and choosing parameters and models.

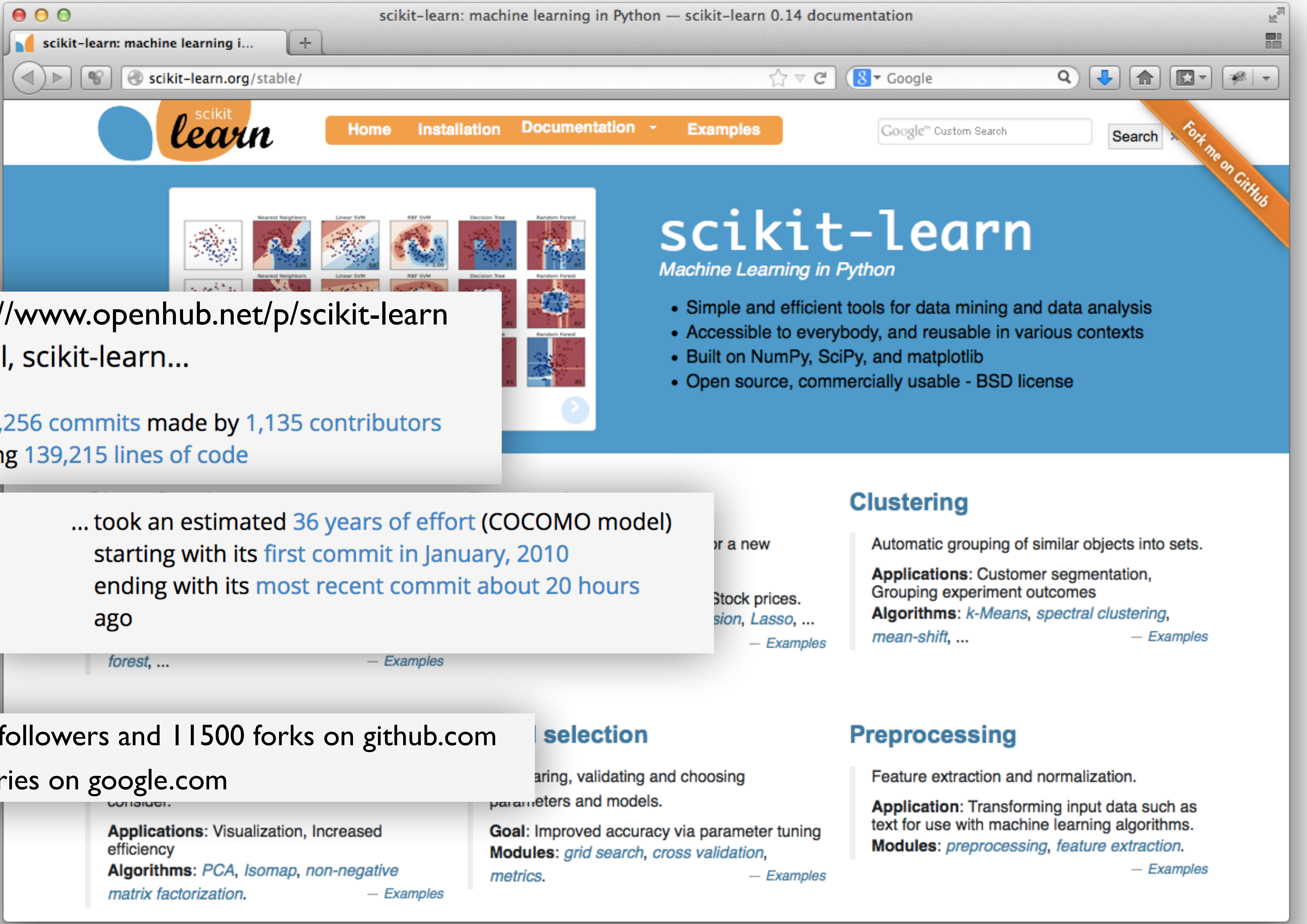
Goal: Improved accuracy via parameter tuning
Modules: *grid search*, *cross validation*, *metrics*. — *Examples*

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.
Modules: *preprocessing*, *feature extraction*. — *Examples*

http://scikit-learn.org



scikit-learn

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

r a new

Stock prices.

sion, *Lasso*, ...

— *Examples*

forest, ...

— *Examples*

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: *k-Means*, *spectral clustering*, *mean-shift*, ...

— *Examples*

consider.

Applications: Visualization, Increased efficiency

Algorithms: *PCA*, *Isomap*, *non-negative matrix factorization*.

— *Examples*

selection

aring, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: *grid search*, *cross validation*, *metrics*.

— *Examples*

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: *preprocessing*, *feature extraction*.

— *Examples*

Source: https://www.openhub.net/p/scikit-learn

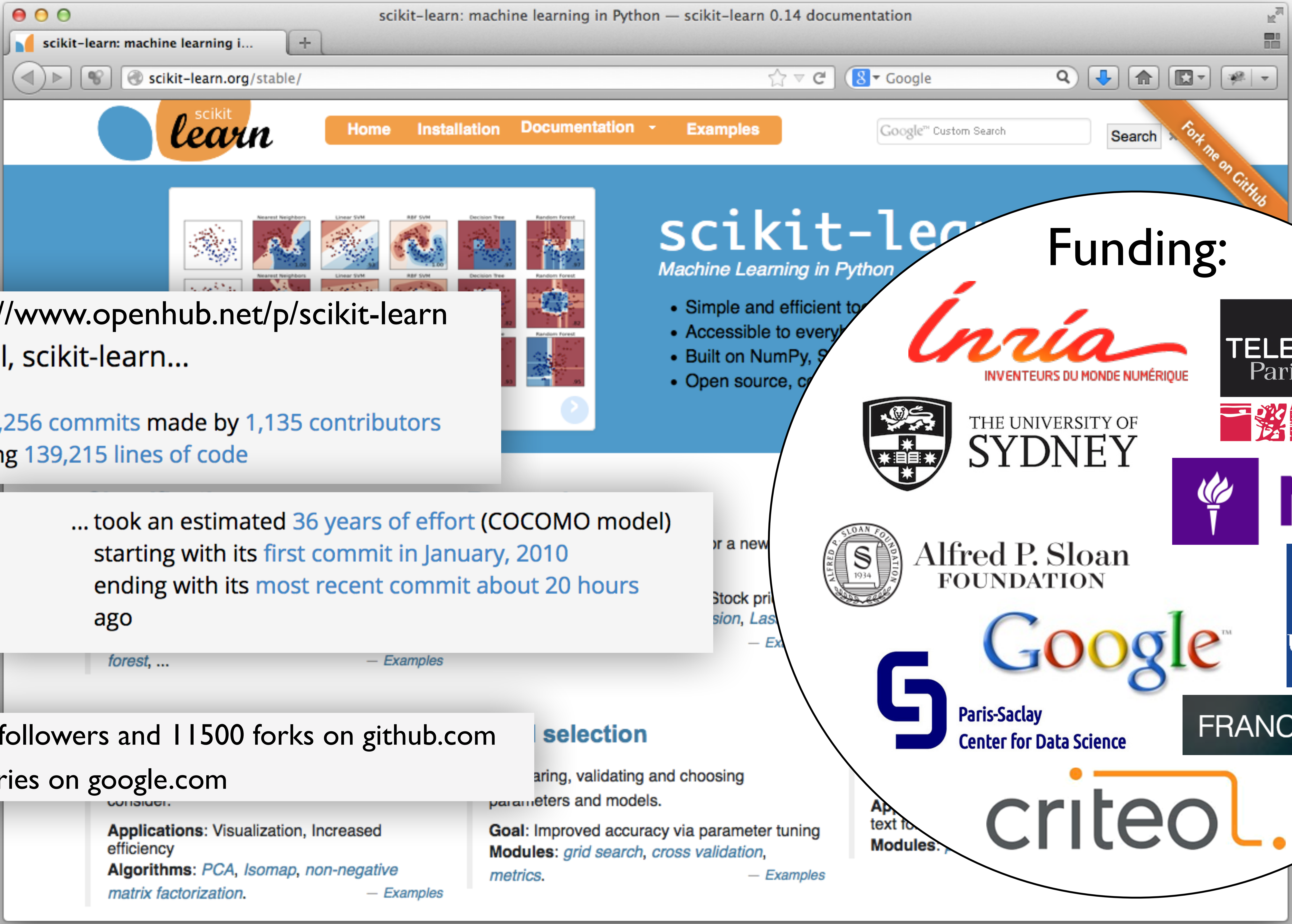
In a Nutshell, scikit-learn...

... has had 22,256 commits made by 1,135 contributors representing 139,215 lines of code

... took an estimated 36 years of effort (COCOMO model) starting with its first commit in January, 2010 ending with its most recent commit about 20 hours ago

- > 21500 followers and 11500 forks on github.com
- 474k entries on google.com

http://scikit-learn.org



scikit-learn: machine learning in Python — scikit-learn 0.14 documentation

scikit-learn: machine learning i...

scikit-learn.org/stable/

scikit-lea

Machine Learning in Python

- Simple and efficient to
- Accessible to every
- Built on NumPy, S
- Open source, c

r a new

Stock pri

sion, Las

— Ex

forest, ... — Examples

selection

aring, validating and choosing parameters and models.

Applications: Visualization, Increased efficiency

Algorithms: PCA, Isomap, non-negative matrix factorization. — Examples

Goal: Improved accuracy via parameter tuning

Modules: grid search, cross validation, metrics. — Examples

Ap text fo Modules:

Source: https://www.openhub.net/p/scikit-learn

In a Nutshell, scikit-learn...

... has had 22,256 commits made by 1,135 contributors representing 139,215 lines of code

... took an estimated 36 years of effort (COCOMO model) starting with its first commit in January, 2010 ending with its most recent commit about 20 hours ago

- > 21500 followers and 11500 forks on github.com
- 474k entries on google.com

Funding:

Inria INVENTEURS DU MONDE NUMÉRIQUE

TELECOM ParisTech

THE UNIVERSITY OF SYDNEY

NYU

Alfred P. Sloan FOUNDATION

Google

COLUMBIA UNIVERSITY

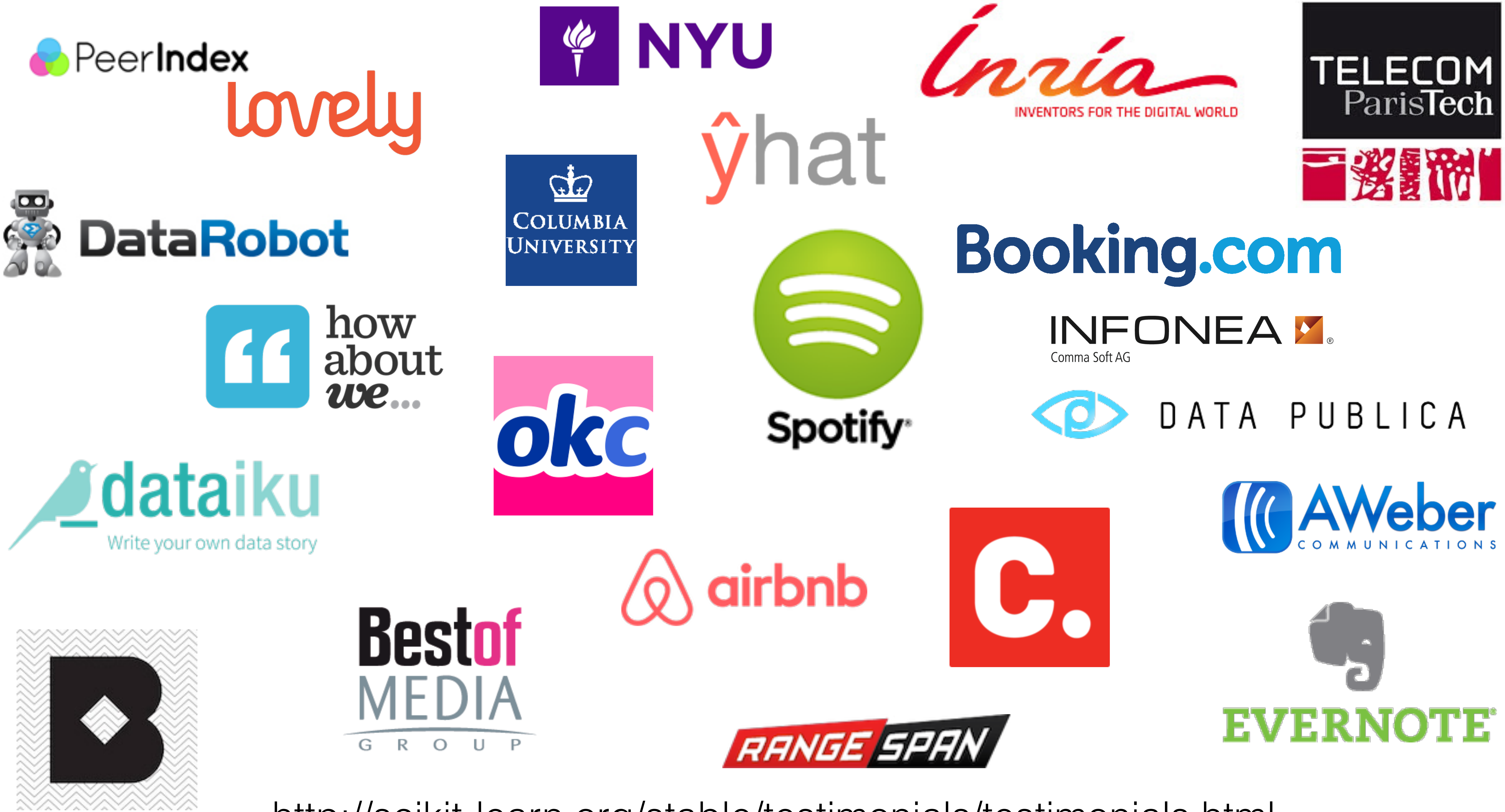
Paris-Saclay Center for Data Science

FRANCE IS AI

criteo

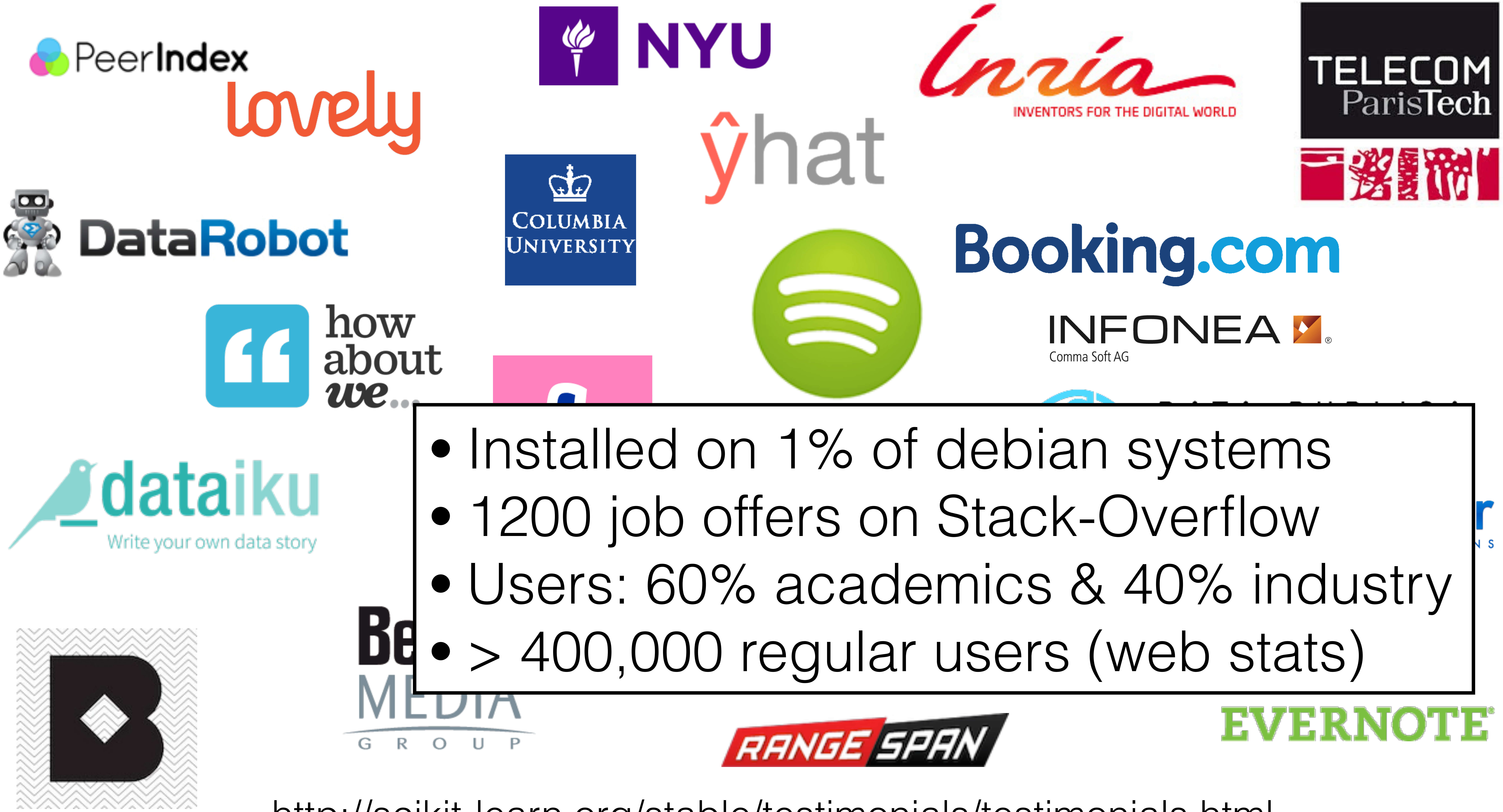
http://scikit-learn.org

scikit-learn impact: education, research, industry



http://scikit-learn.org/stable/testimonials/testimonials.html

scikit-learn impact: education, research, industry



- Installed on 1% of debian systems
- 1200 job offers on Stack-Overflow
- Users: 60% academics & 40% industry
- > 400,000 regular users (web stats)

http://scikit-learn.org/stable/testimonials/testimonials.html

Machine Learning Taxonomy

Machine Learning

Unsupervised learning

Supervised learning

Clustering

Dimensionality Reduction

Anomaly Detection

Regression

Classification

Ranking

Machine Learning Taxonomy

Machine Learning
Unsupervised learning
Supervised learning
Clustering
Dimensionality Reduction
Anomaly Detection
Regression
Classification
Ranking

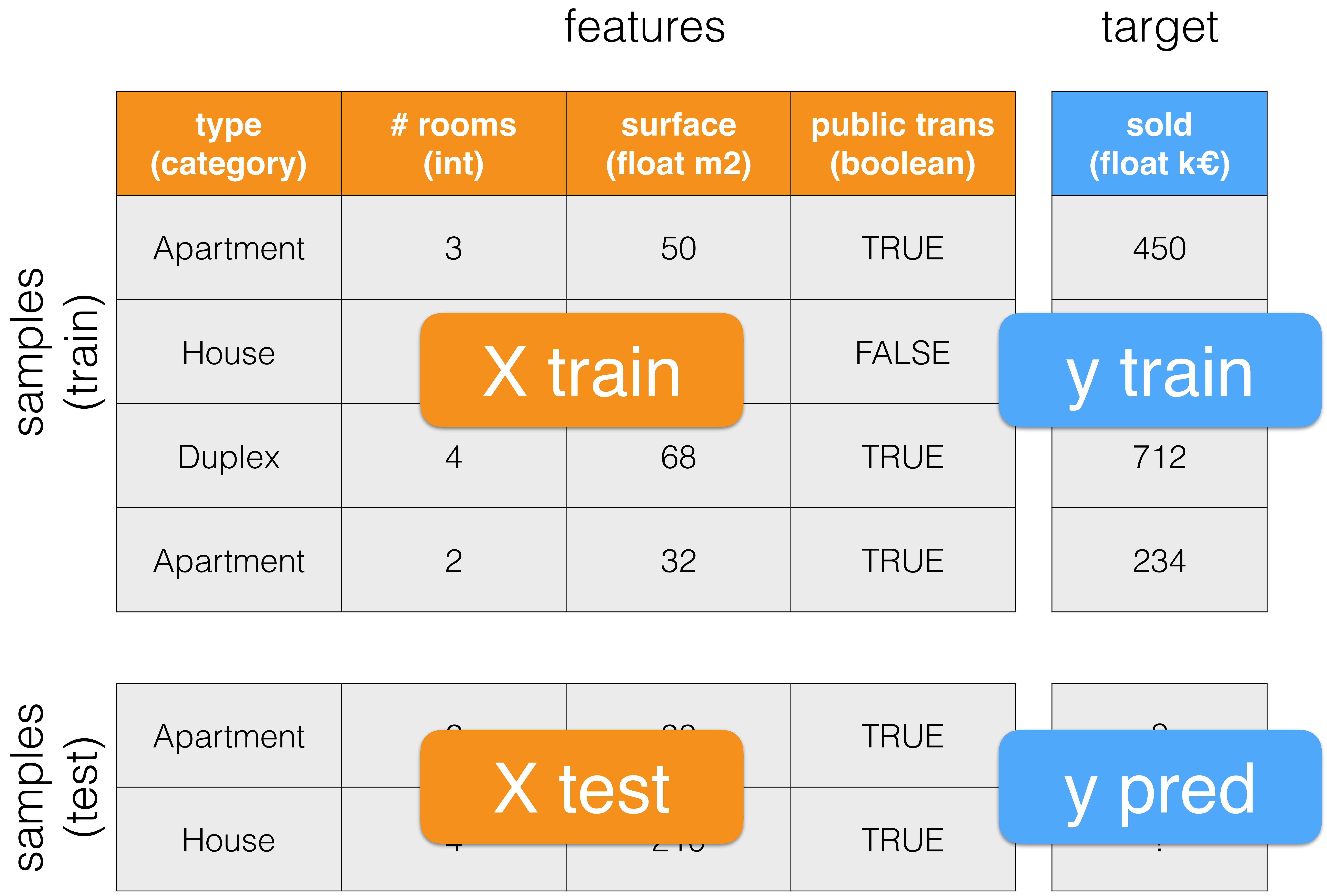
let's start with supervised learning workflow….

type (category)	# rooms (int)	surface (float m2)	public trans (boolean)
Apartment	3	50	TRUE
House	5	254	FALSE
Duplex	4	68	TRUE
Apartment	2	32	TRUE

type (category)	# rooms (int)	surface (float m2)	public trans (boolean)	sold (float k€)
Apartment	3	50	TRUE	450
House	5	254	FALSE	430
Duplex	4	68	TRUE	712
Apartment	2	32	TRUE	234

	features				target
	type (category)	**# rooms (int)**	**surface (float m2)**	**public trans (boolean)**	**sold (float k€)**
samples (train)	Apartment	3	50	TRUE	450
	House	5	254	FALSE	430
	Duplex	4	68	TRUE	712
	Apartment	2	32	TRUE	234

	features				target
	type (category)	**# rooms (int)**	**surface (float m2)**	**public trans (boolean)**	**sold (float k€)**
samples (train)	Apartment	3	50	TRUE	450
	House	5	254	FALSE	430
	Duplex	4	68	TRUE	712
	Apartment	2	32	TRUE	234
samples (test)	Apartment	2	33	TRUE	?
	House	4	210	TRUE	?



features

target

samples (train)

type (category)	# rooms (int)	surface (float m2)	public trans (boolean)	sold (float k€)
Apartment	3	50	TRUE	450
House	[illegible]	[illegible]	FALSE	[illegible]
Duplex	4	68	TRUE	712
Apartment	2	32	TRUE	234

X train

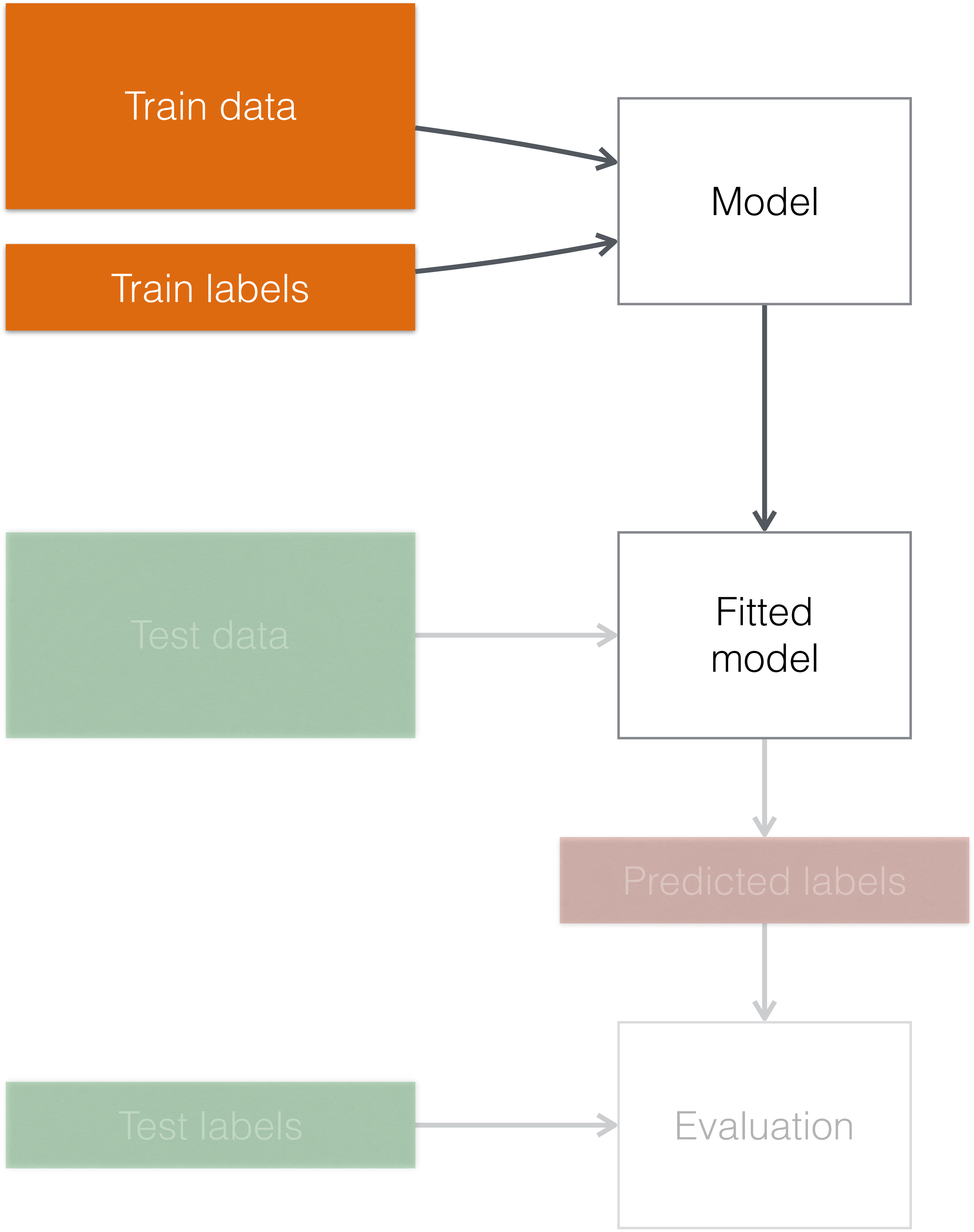
y train

samples (test)

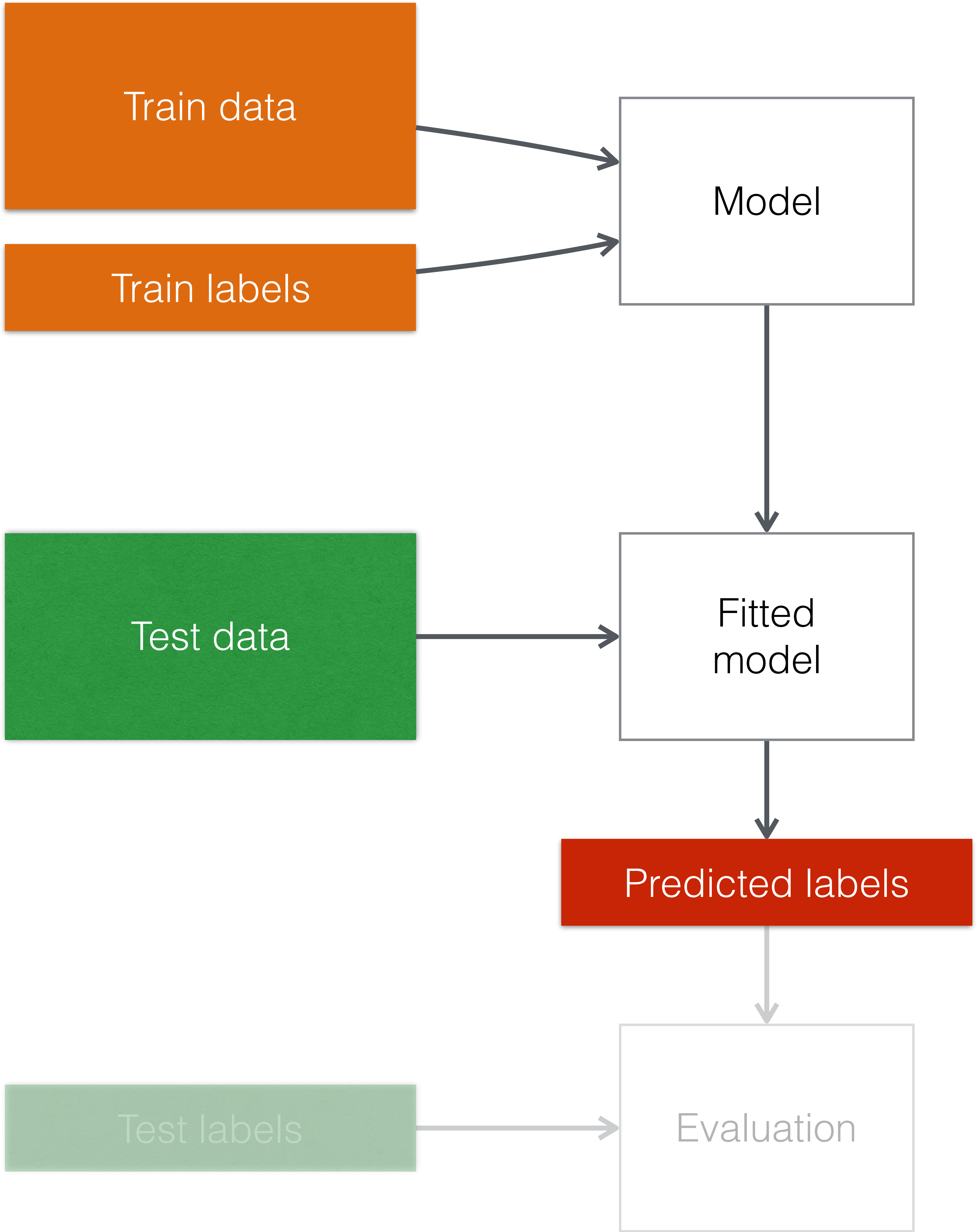
Apartment	[illegible]	[illegible]	TRUE	?
House	4	210	TRUE	?

X test

y pred

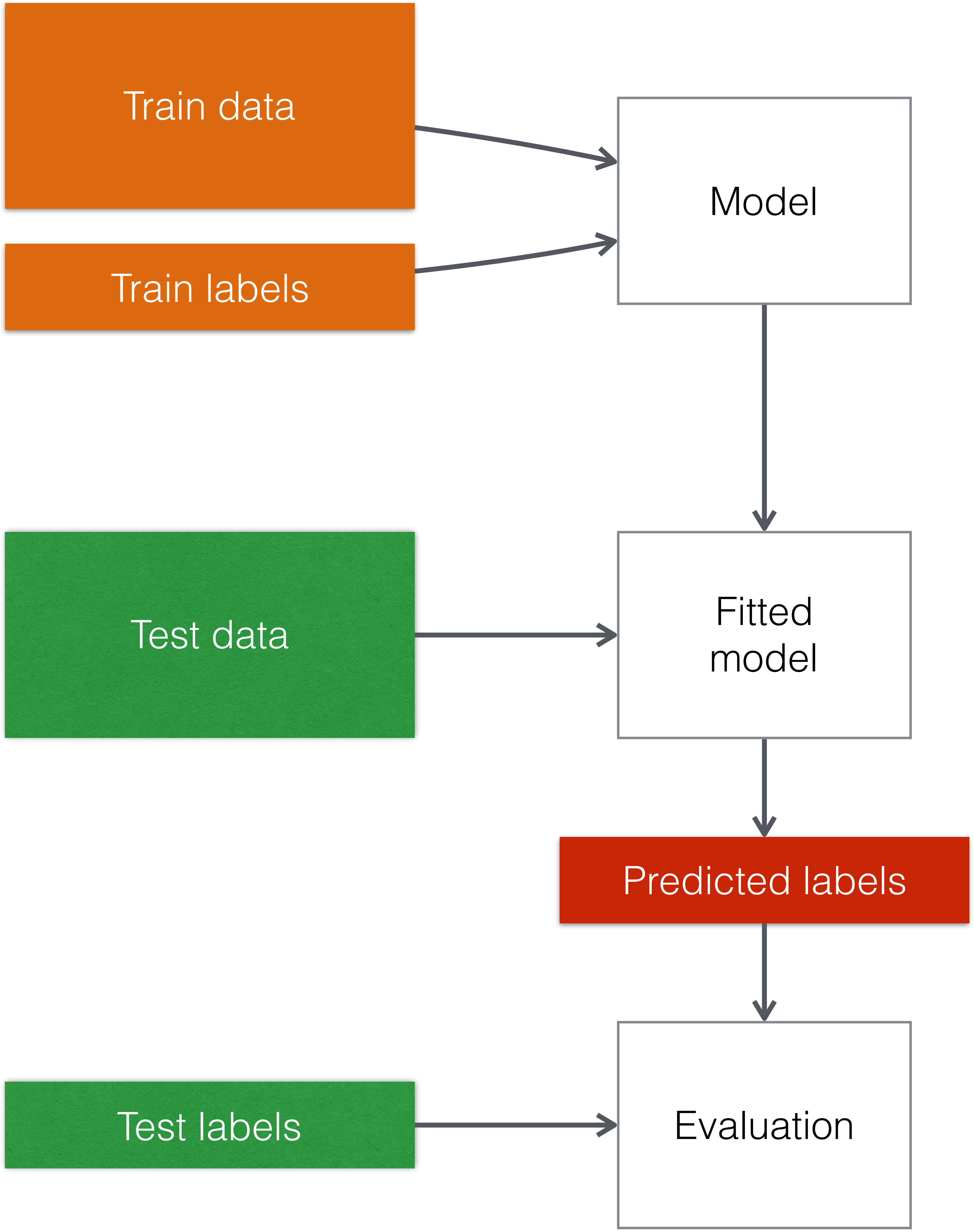


```
>>> model = LogisticRegression(C=1)
>>> model.fit(X_train, y_train)
```



```
>>> model = LogisticRegression(C=1)
>>> model.fit(X_train, y_train)
```

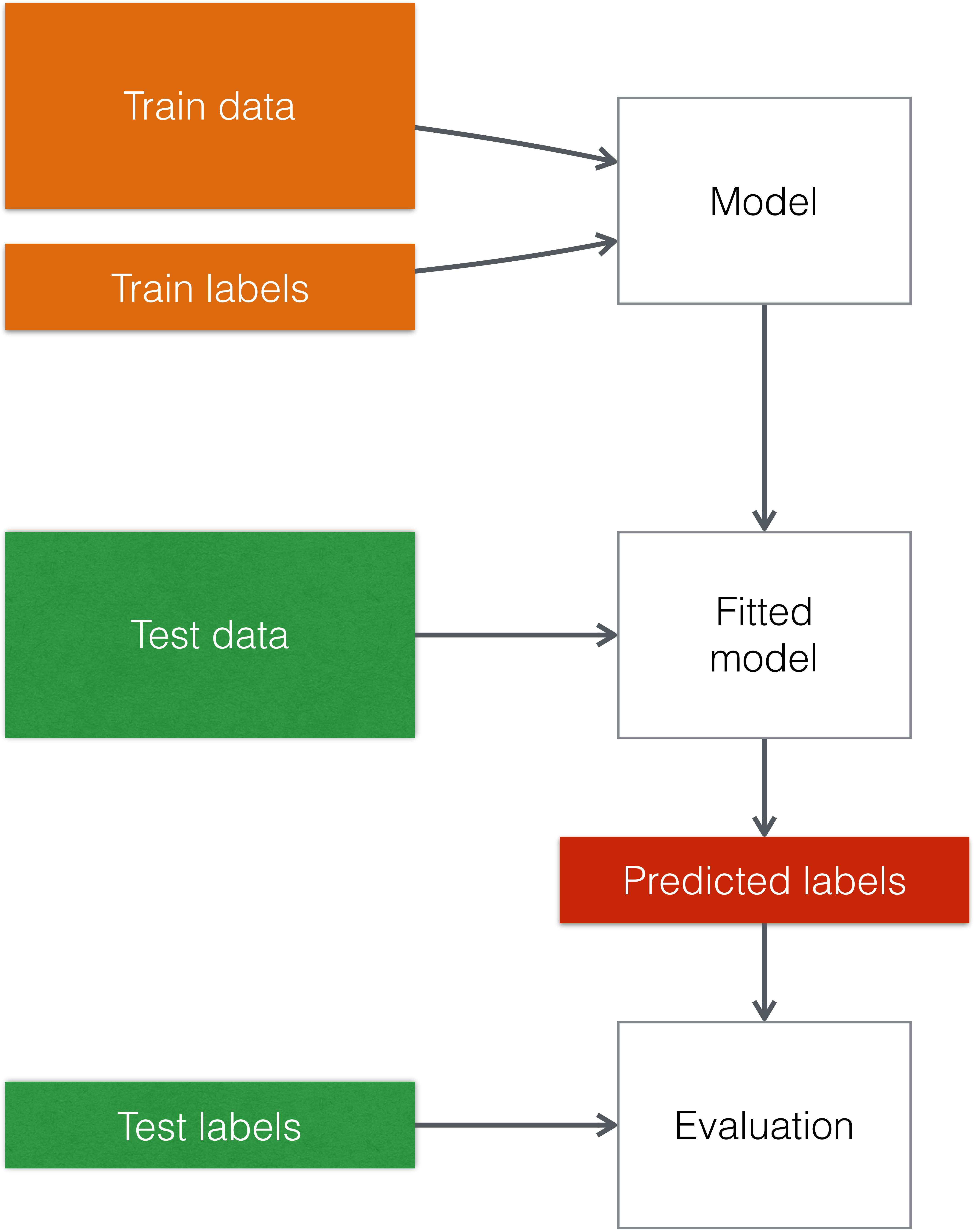
```
>>> y_pred = model.predict(X_test)
```



```
>>> model = LogisticRegression(C=1)
>>> model.fit(X_train, y_train)
```

```
>>> y_pred = model.predict(X_test)
```

```
>>> accuracy_score(y_test, y_pred)
```



```
>>> model = LogisticRegression(C=1)
>>> model.fit(X_train, y_train)
```

fit

```
>>> y_pred = model.predict(X_test)
```

predict

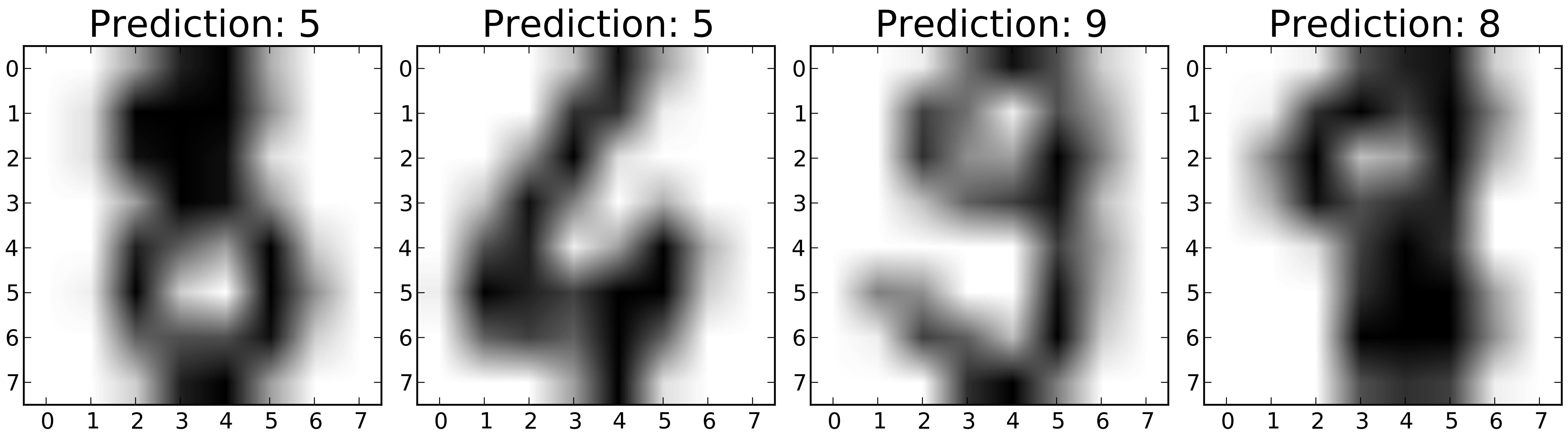
```
>>> accuracy_score(y_test, y_pred)
```

score

Putting this together…

Classification of images of digits in a few lines of code

```python
import matplotlib.pyplot as plt
from sklearn import datasets, svm
# Load data
digits = datasets.load_digits()
n_samples = len(digits.images)
data = digits.images.reshape((n_samples, -1))
# Learn ie. fit
classifier = svm.SVC()
classifier.fit(data[:n_samples // 2], digits.target[:n_samples // 2])
# Predict and plot
for index, image in enumerate(digits.images[n_samples // 2:n_samples // 2 + 4]):
    plt.subplot(1, 4, index)
    plt.imshow(image, cmap=plt.cm.gray_r)
    plt.title('Prediction: %i' % classifier.predict(image.ravel()), fontsize=20)
```



powerful yet easy!

the (supervised learning) API

```
>>> from sklearn import Model
>>> model = Model(param1=1e-8, param2="auto")
>>> print(model.param2)
"auto"
>>> model.fit(X_train, y_train)  # learn from training data
>>> y_pred = model.predict(X_test)  # predict from new data
>>> model.score(X_test, y_test)  # evaluate performance new data
0.96
```

API: fit, predict, score

Unsupervised learning

samples

features				~~target~~
type (category)	**# rooms (int)**	**surface (float m2)**	**public trans (boolean)**	~~**sold (float k€)**~~
Apartment	3	50	TRUE	~~450~~
House	5	254	FALSE	~~430~~
Duplex	4	68	TRUE	~~712~~
Apartment	2	32	TRUE	~~234~~

We have just X (no y)

the (unsupervised learning) API

```
>>> from sklearn import Model
>>> model = Model(param1=1e-8, param2="auto")
>>> print(model.param2)
"auto"
>>> model.fit(X)  # learn from training data (no y)
>>> Xt = model.transform(X)  # transform new or same data
```

API: fit, transform

What to do with scikit-learn?

Supervised

- Trees / Random Forest
- SVM
- Ridge Regression
- Logistic regression
- Nearest Neighbors
- LDA / GDA
- Canonical component analysis
- Partial Least Squares
- Naive Bayes
- Gaussian process
- Lasso (sparse models)

Unsupervised

- Clustering:
  - Mean-shift
  - Affinity propagation
  - Hierarchical Clustering
  - K-Means
  - Gaussian mixtures
  - DBSCAN / BIRCH
- Factorization / decomposition
  - PCA / ICA / NMF
  - Latent Dirichlet Allocation
  - Dictionary learning
- Dimensionality Reduction:
  - PCA
  - Manifold learning
  - TSNE
- Anomaly/Novelty detection
  - One-class SVM
  - Isolation Forest
  - Local Outlier Factor (LOF)

BSD License
Python / NumPy / SciPy / Cython

http://scikit-learn.org/stable

Now some demos …

Contact:

Alexandre Gramfort
http://alexandre.gramfort.net

GitHub : @agramfort

Twitter : @agramfort