Machine Learning in Python with scikit-learn

O'Reilly Webcast
Aug. 2014

Olivier Grisel
@ogrisel
Datageek, contributor to scikit-learn, works with Python / Java / Clojure / Pig, interested in Machine Learning, NLProc, {Big|Linked|Open} Data and braaains!
Paris, France · http://github.com/ogrisel

Outline

- Machine Learning refresher
- scikit-learn
- How the project is structured
- Some improvements released in 0.15
- Ongoing work for 0.16

Predictive modeling ~= machine learning

- Make predictions of outcome on new data
- Extract the structure of historical data
- Statistical tools to summarize the training data into a executable predictive model
- Alternative to hard-coded rules written by experts

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

samples (train)

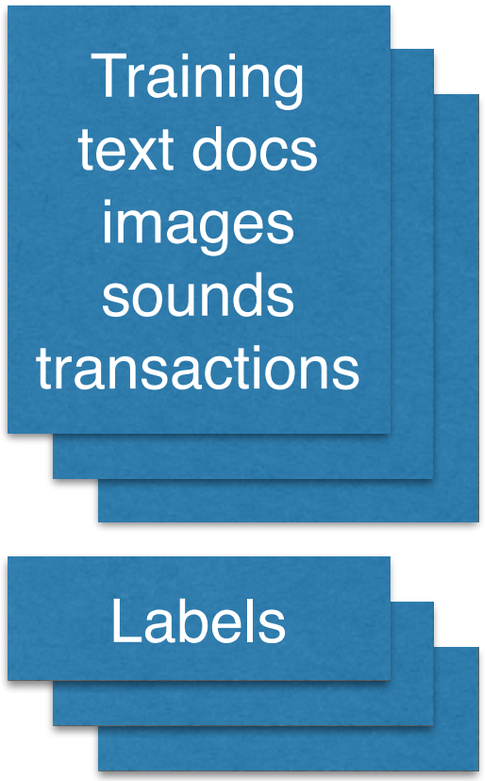
type (category)	# rooms (int)	surface (float m2)	public trans (boolean)	sold (float k€)
Apartment	3	50	TRUE	450
House	5	254	FALSE	430
Duplex	4	68	TRUE	712
Apartment	2	32	TRUE	234

features

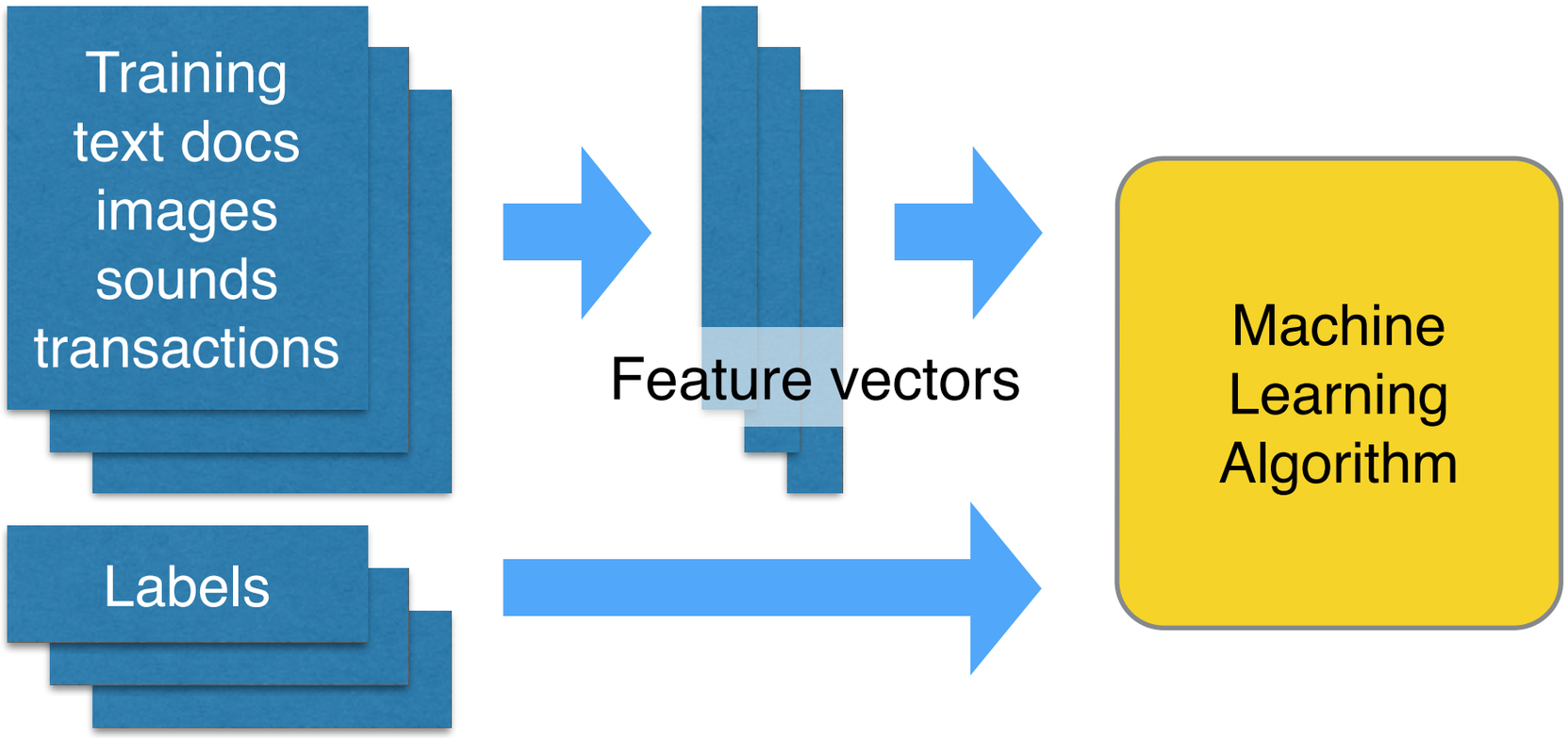
target

	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
samples (test)	Apartment	2	33	TRUE	?
	House	4	210	TRUE	?

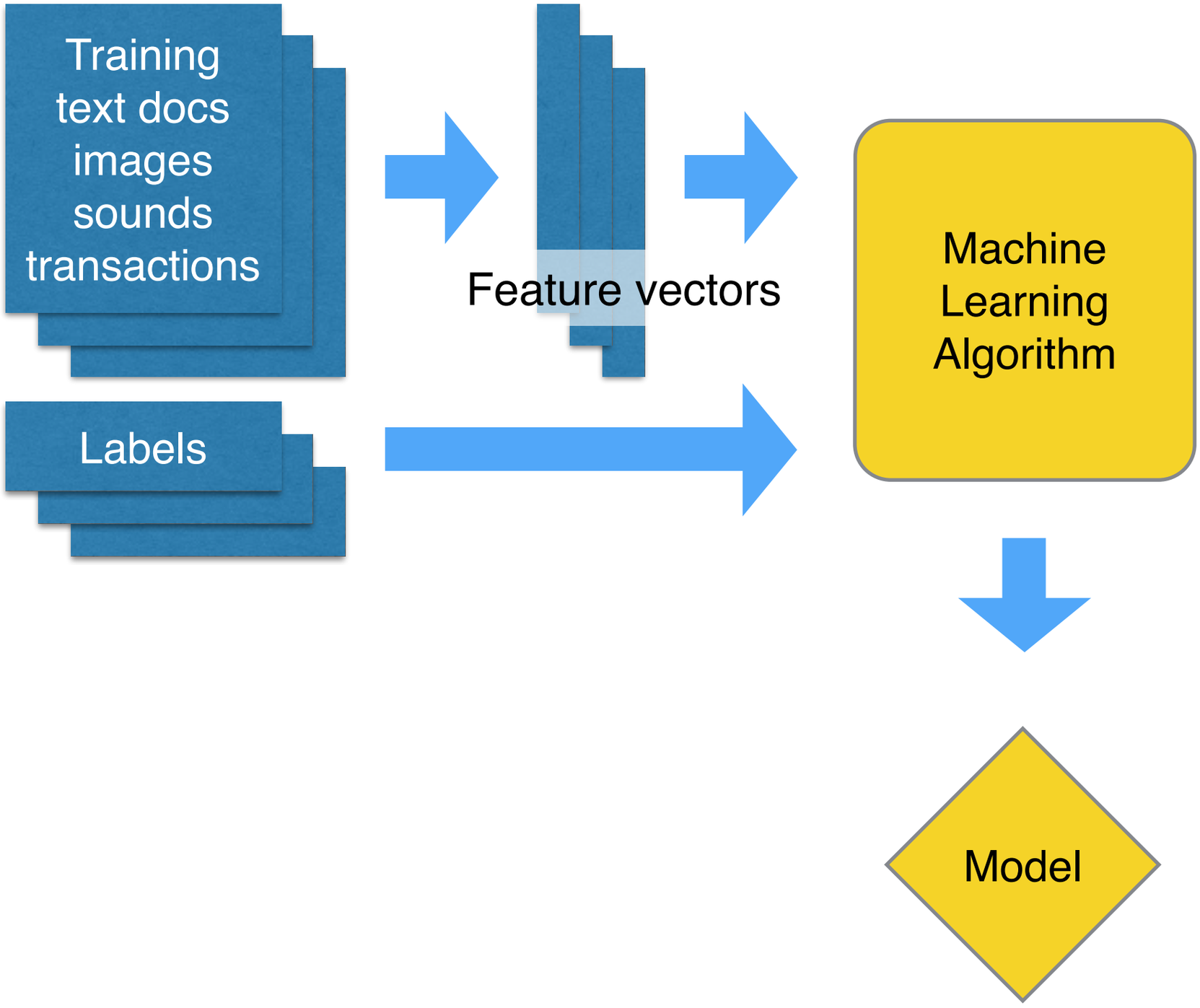
Predictive Modeling Data Flow



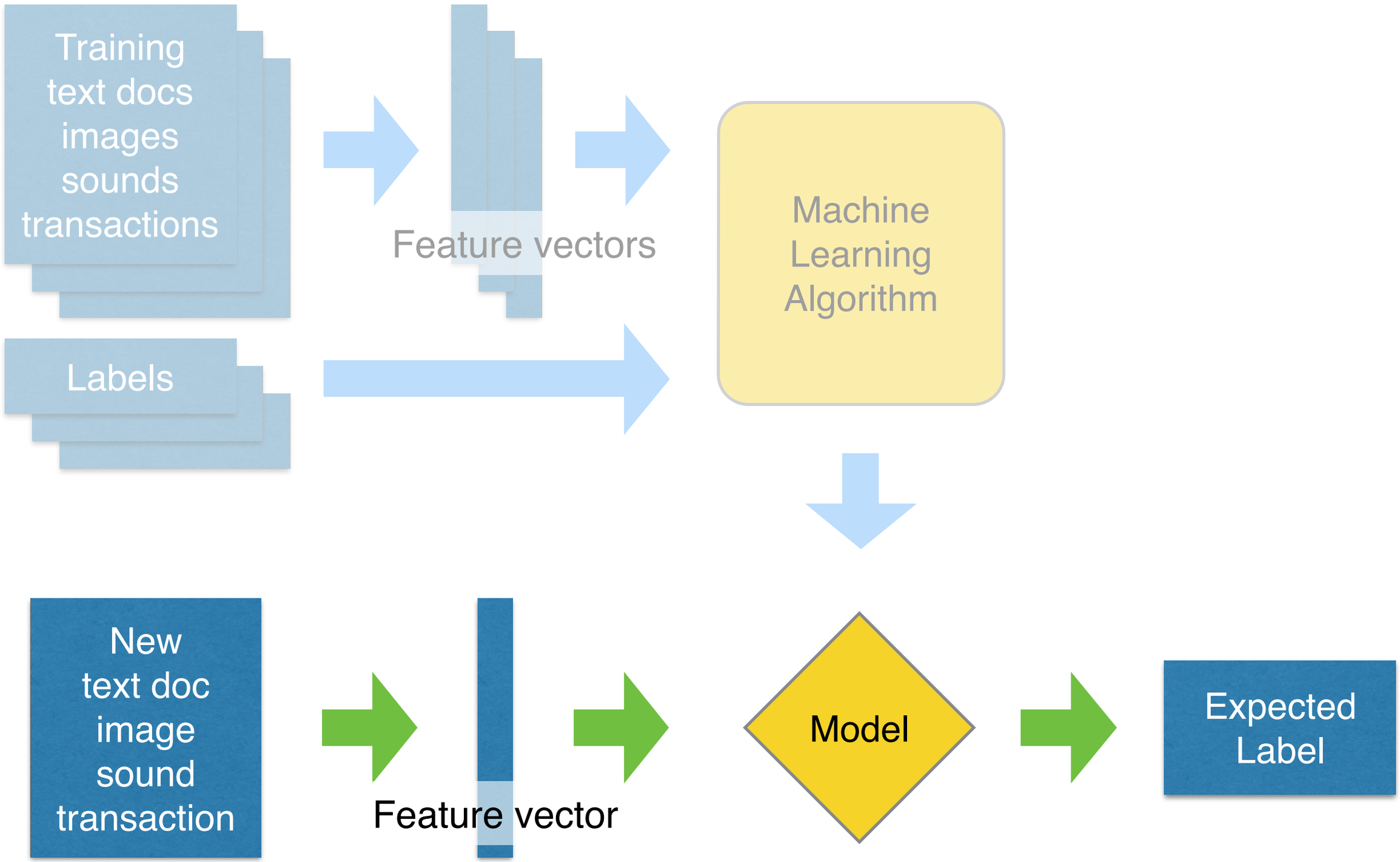
Predictive Modeling Data Flow



Predictive Modeling Data Flow



Predictive Modeling Data Flow



Predictive Modeling Data Flow

Applications in Business

- Forecast sales, customer churn, traffic, prices
- Predict CTR and optimal bid price for online ads
- Build computer vision systems for robots in the industry and agriculture
- Detect network anomalies, fraud and spams
- Recommend products, movies, music

Applications in Science

- Decode the activity of the brain recorded via fMRI / EEG / MEG
- Decode gene expression data to model regulatory networks
- Predict the distance of each star in the sky
- Identify the Higgs boson in proton-proton collisions

- Library of Machine Learning algorithms
- Focus on established methods (e.g. ESL-II)
- Open Source (BSD)
- Simple **`fit`** / **`predict`** / **`transform`** API
- Python / NumPy / SciPy / Cython
- Model Assessment, Selection & Ensembles

Support Vector Machine

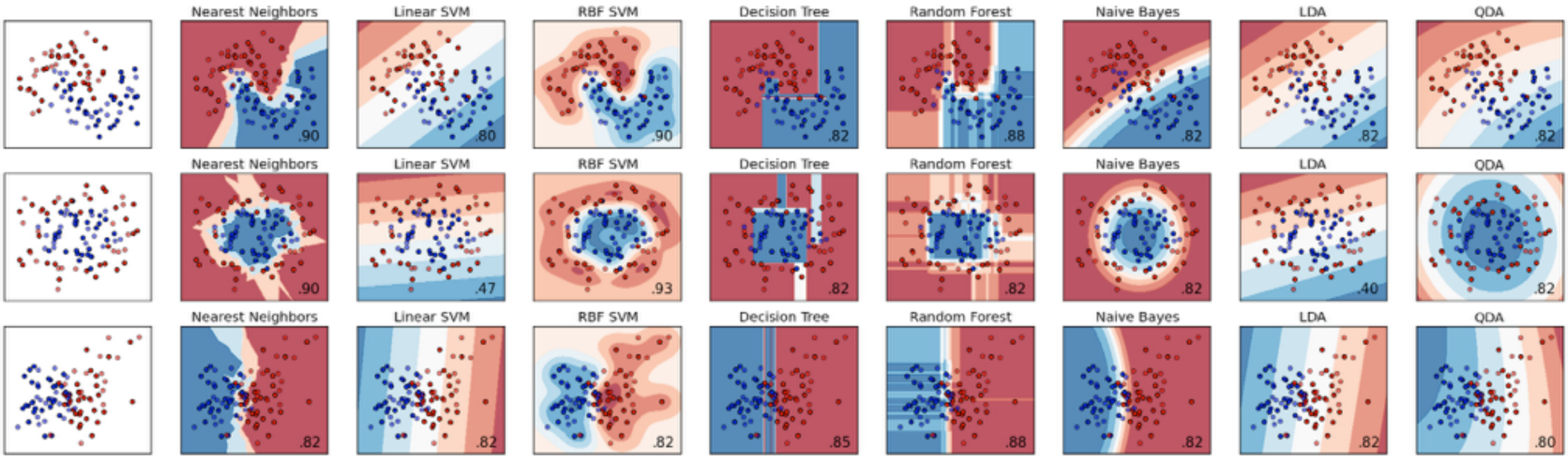
```
from sklearn.svm import SVC

model = SVC(kernel="rbf", C=1.0, gamma=1e-4)
model.fit(X_train, y_train)


y_predicted = model.predict(X_test)

from sklearn.metrics import f1_score
f1_score(y_test, y_predicted)
```

Linear Classifier

```
from sklearn.linear_model import SGDClassifier

model = SGDClassifier(alpha=1e-4, penalty="elasticnet")
model.fit(X_train, y_train)


y_predicted = model.predict(X_test)

from sklearn.metrics import f1_score
f1_score(y_test, y_predicted)
```

Random Forests

```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier(n_estimators=200)
model.fit(X_train, y_train)


y_predicted = model.predict(X_test)

from sklearn.metrics import f1_score
f1_score(y_test, y_predicted)
```


Nearest Neighbors
Linear SVM
RBF SVM
Decision Tree
Random Forest
Naive Bayes
LDA
QDA
.90
.80
.90
.82
.88
.82
.82
.82
Nearest Neighbors
Linear SVM
RBF SVM
Decision Tree
Random Forest
Naive Bayes
LDA
QDA
.90
.47
.93
.82
.82
.82
.40
.82
Nearest Neighbors
Linear SVM
RBF SVM
Decision Tree
Random Forest
Naive Bayes
LDA
QDA
.82
.82
.82
.85
.88
.82
.82
.80

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

scikit-learn: machine learning i...

scikit-learn.org/stable/

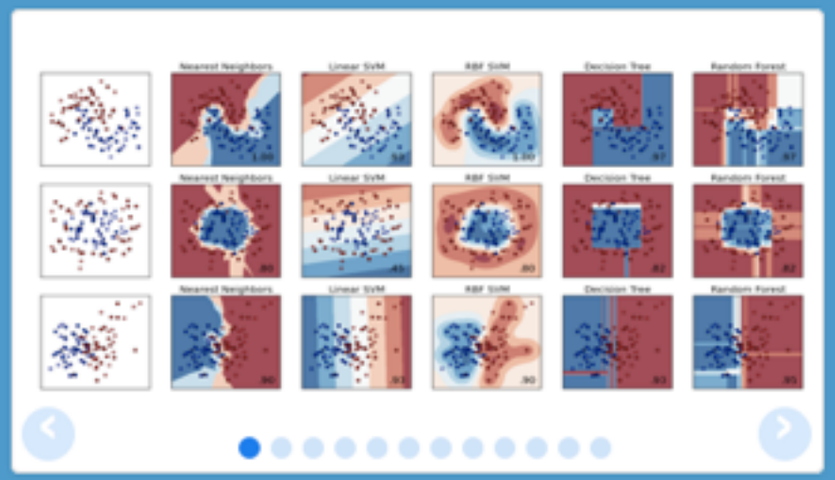
Google

scikit learn

Home | Installation | Documentation | Examples

Google™ Custom Search | Search

Fork me on GitHub



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*

1. Supervised learning

- ▸ 1.1. Generalized Linear Models
- ▸ 1.2. Support Vector Machines
- ▸ 1.3. Stochastic Gradient Descent
- ▸ 1.4. Nearest Neighbors
- ▸ 1.5. Gaussian Processes
- 1.6. Cross decomposition
- ▸ 1.7. Naive Bayes
- ▸ 1.8. Decision Trees
- ▸ 1.9. Ensemble methods
- ▸ 1.10. Multiclass and multilabel algorithms
- ▸ 1.11. Feature selection
- ▸ 1.12. Semi-Supervised
- ▸ 1.13. Linear and quadratic discriminant analysis
- 1.14. Isotonic regression

2. Unsupervised learning

- ▸ 2.1. Gaussian mixture models
- ▸ 2.2. Manifold learning
- ▸ 2.3. Clustering
- ▸ 2.4. Biclustering
- ▸ 2.5. Decomposing signals in components (matrix factorization problems)
- ▸ 2.6. Covariance estimation
- ▸ 2.7. Novelty and Outlier Detection
- ▸ 2.8. Hidden Markov Models
- ▸ 2.9. Density Estimation
- ▸ 2.10. Neural network models (unsupervised)

scikit-learn contributors

- GitHub-centric contribution workflow
  - each pull request needs 2 x [+1] reviews
  - code + tests + doc + example
  - ~94% test coverage / Continuous Integration
- 2-3 major releases per years + bug-fix
- 150+ contributors for release 0.15

Pull Requests · scikit-learn/scikit-learn

Pull Requests · scikit-learn/sci...

GitHub, Inc. (US) https://github.com/scikit-learn/scikit-learn/pulls

Google

This repository | Search or type a command | Explore | Gist | Blog | Help | ogrisel

scikit-learn / scikit-learn

Unwatch 285 | Unstar 2,263 | Fork 1,171

All Requests 128

Yours 3

Find a user…

jnothman 13
amueller 9
larsmans 5
kemaleren 4
vene 3
GaelVaroquaux 3
IssamLaradji 3
kpysniak 3
jakevdp 3
jcrudy 2
robertlayton 2
pprett 2

Open | Closed | Sort: Newest | ◂ 1 2 3 … 6 ▸ | New pull request

[WIP] faster sorting in trees; random forests almost 2× as fast #2747
Changed the heapsort in the tree learners into a quicksort and gave it cache-friendlier data acce…
by larsmans 32 minutes ago 1 comment

[MRG] Optimize mean square criterion #2745
The mse criterion could be simplified and optimized. The main idea is to avoid an online algorith…
by arjoly 5 hours ago 9 comments

Add user option to use SVD for orthogonalizing the mixing matrix in FastICA #2738
It seems that the current eigendecomposition-based method for orthogonalizing the mixing matrix in …
by alimuldal 3 days ago 2 comments

[MRG] Refactor CV and grid search #2736
1 tasks (1 completed, 0 remaining)
by AlexanderFabisch 4 days ago 66 comments

CV Attributes #2733
Partially completed #2709. I'm opening this pull request to get feedback on the implementation of…
by eshilts 5 days ago 4 comments

[MRG] ENH/FIX Change Tree underlying data structure #2732

[WIP] first cut at LambdaMART by jwkvam · Pull Request #2580 · scikit-learn/scikit-learn

[WIP] first cut at LambdaMART ...

GitHub, Inc. (US) https://github.com/scikit-learn/scikit-learn/pull/2580

Google

This repository | Search or type a command | Explore | Gist | Blog | Help | ogrisel

scikit-learn / scikit-learn

Unwatch 285 | Unstar 2,263 | Fork 1,171

Open | jwkvam wants to merge 3 commits into `scikit-learn:master` from `jwkvam:lambdamart` | 4,353 | #2580

Conversation | Commits 3 | Files Changed 3

jwkvam opened this pull request 2 months ago

[WIP] first cut at LambdaMART

Edit

Open

+ 3,198 additions

- 1,155 deletions

No one is assigned

No milestone

This PR is an attempt to implement LambdaMART [1]. I imagine the biggest hurdle will be coming to some conclusion over the correct API since we need to include the queries somehow. In my implementation I use an extra keyword argument, I don't know if this causes problems elsewhere. My hope is that this PR can serve as a catalyst to resolve that, and then I can finish up the PR.

Some items on the todo list:

- [ ] need tests
- [ ] need docs
- [ ] compare performance with gbm, ranklib, jforests

Connected to s3.amazonaws.com...

There was also a brief discussion on the mailing list [2]. Pinging **@mblondel** **@ogrisel** **@pprett** from that discussion.

[1] https://research.microsoft.com/en-us/um/people/cburges/tech_reports/msr-tr-2010-82.pdf
[2] http://sourceforge.net/mailarchive/forum.php?thread_name=CAP%2B3rpGVbSux5u4dZiatV3p1f1zUvndHXoi-oh4CjMRtSpjFsw%40mail.gmail.com&forum_name=scikit-learn-general

✔ **All is well** — The Travis CI build passed (**Details**)

jwkvam added some commits — 2 months ago

jwkvam	`first cut at lambdamart, only supports NDCG`	`a115d18`
jwkvam	`check that queries are grouped together`	✔ `1afaf09`

coveralls commented — 2 months ago

coverage 94%

Coverage remained the same when pulling `1afaf09` **on jwkvam:lambdamart** into `c0e686b` **on scikit-learn:master**.

agramfort commented — 2 months ago

one option we considered with **@fabianp** to support the query without changing too much the API is to have a 2 columns y where the second column is the query index. It just means that y.ndim can be 2 now.

pprett commented — 2 months ago

[WIP] first cut at LambdaMART by jwkvam · Pull Request #2580 · scikit-learn/scikit-learn

[WIP] first cut at LambdaMART ...

GitHub, Inc. (US) https://github.com/scikit-learn/scikit-learn/pull/2580

Google

ogrisel commented on the diff 2 months ago

sklearn/ensemble/gradient_boosting.py View full changes

```
@@ -585,6 +665,22 @@ def fit(self, X, y):
585 665         X, = check_arrays(X, dtype=DTYPE, sparse_format="dense",
586 666                           check_ccontiguous=True)
587 667         y = column_or_1d(y, warn=True)
    668 +       ranking = self.loss in ('ndcg')
    669 +       if ranking:
    670 +           if query is None:
    671 +               raise ValueError("query must not be none with ranking measure")
```

2

ogrisel repo collab 2 months ago

As **@mblondel** said, I think we should treat the `X`, `y` data as stemming from a single query in that case.

jwkvam 2 months ago

Thanks for the reminder, I will get around to it eventually.

Add a line note

jwkvam commented 12 days ago

Sorry for the lack of updates, admittedly I've been kind of lazy. I benchmarked the code (there are some unpushed changes) against GBM and the performance seems comparable outside of the execution times. I used 50 trees, a depth of 3 for

scikit-learn
International Sprint
Paris - 2014

scikit-learn users

- We support users on stackoverflow & ML
- 1500+ questions tagged with [scikit-learn]
- Many kaggle.com competitors + benchmarks
- Many data-driven startups use sklearn
- 500+ answers on 0.13 release user survey
  - 60% academics / 40% from industry

PeerIndex

fnrs
LA LIBERTÉ DE CHERCHER

Inria
INVENTORS FOR THE DIGITAL WORLD

TELECOM ParisTech

DataRobot

lovely

Spotify

ŷhat

how about we...

okc

Dataiku

airbnb

AWeber COMMUNICATIONS

c.

Bestof MEDIA GROUP

RANGE SPAN

EVERNOTE

New in 0.15

Fit time improvements in Ensembles of Trees

- Large refactoring of the Cython code base
- Better internal data structures to optimize CPU cache usage
- Leverage constant features detection
- Optimized MSE loss (for GBRT and regression forests)
- Cached features for Extra Trees
- Custom pure Cython PRNG and sort routines

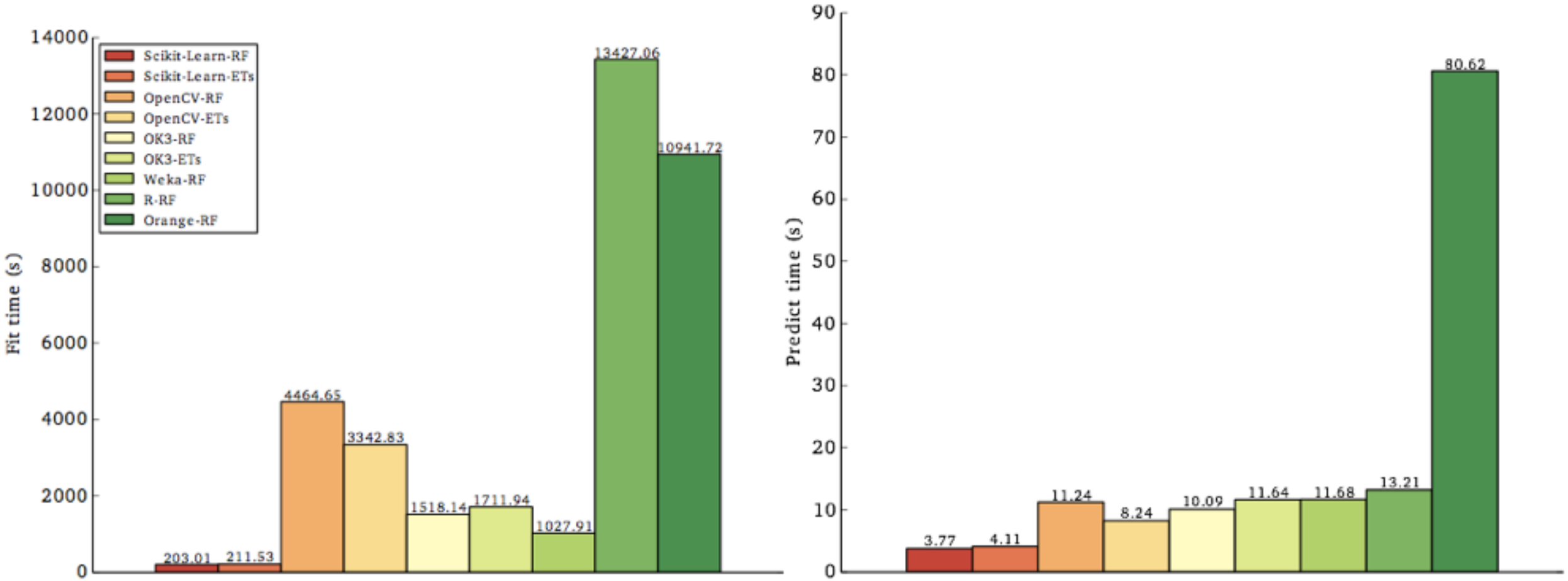


Figure 5.9: Average time required for building a forest on the MNIST dataset (left) and average time required for making predictions (right).

source: Understanding Random Forests by Gilles Louppe

Dataset	wiseRF 1.5.11	scikit-learn 0.14	scikit-learn 0.15	CudaTree 0.6
ImageNet subset	23s	50s	**13s**	25s
CIFAR-100 (raw)	**160s**	502s	181s	197s
covertype	107s	463s	73s	**67s**
poker	117s	415s	99s	**59s**
PAMAP2	1,066s	7,630s	1,683s	**934s**
intrusion	667s	1,528s	241s	**199s**

source: Blog post by Alex Rubinsteyn

Massive memory usage by parallel RandomForestClassifier #936

Closed **jni** opened this issue on Jul 7, 2012 · 32 comments

jni commented on Jul 7, 2012

I think this will be hard to fix without swapping out joblib (or maybe even the GIL ;), but basically the amount of memory used by RandomForestClassifier is exorbitant for n_jobs > 1. In my case, I have a dataset of about 1GB (300,000 samples by 415 features by 64-bit float), but doing fit() on a RandomForestClassifier having n_jobs=16 results in 45GB of memory being used.

Does anyone have any ideas or is this hopeless without moving everything to C?

Optimized memory usage for parallel training of ensembles of trees

- Extensive use of **`with nogil`** blocks in Cython
- **`threading`** backend for **`joblib`** in addition to the **`multiprocessing`** backend
- Also brings fit-time improvements when training many small trees in parallel
- Memory usage is now:
  sizeofdata(training_data) + sizeof(all_trees)

Other memory usage improvements

- Chunked euclidean distances computation in **`KMeans`** and **`Neighbors`** estimators

- Support of **`numpy.memmap`** input data for shared memory (e.g. with **`GridSearchCV`** w/ **`n_jobs=16`**)

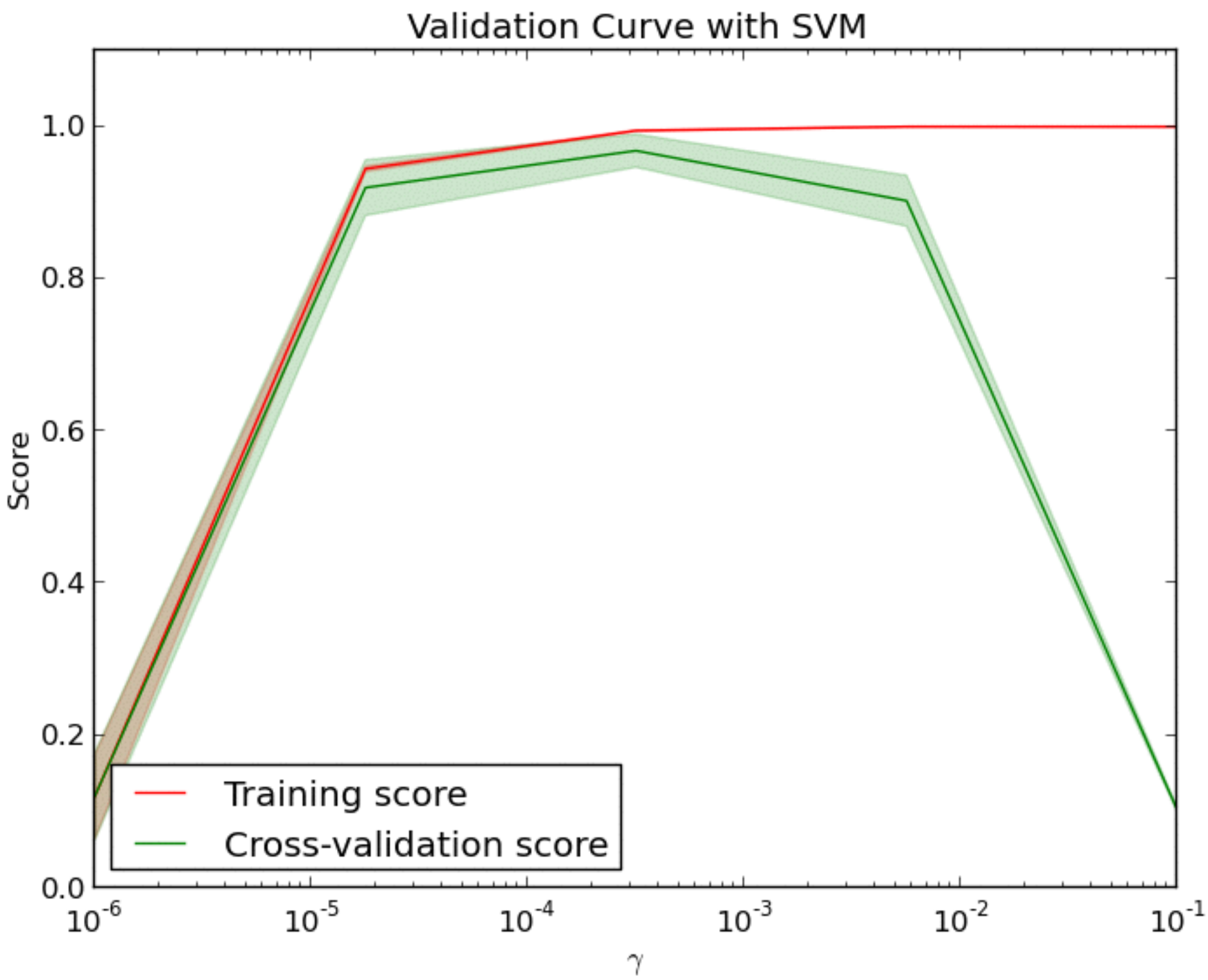
- GIL-free threading backend for multi-class **`SGDClassifier`**.

- Much more: scikit-learn.org/stable/whats_new.html

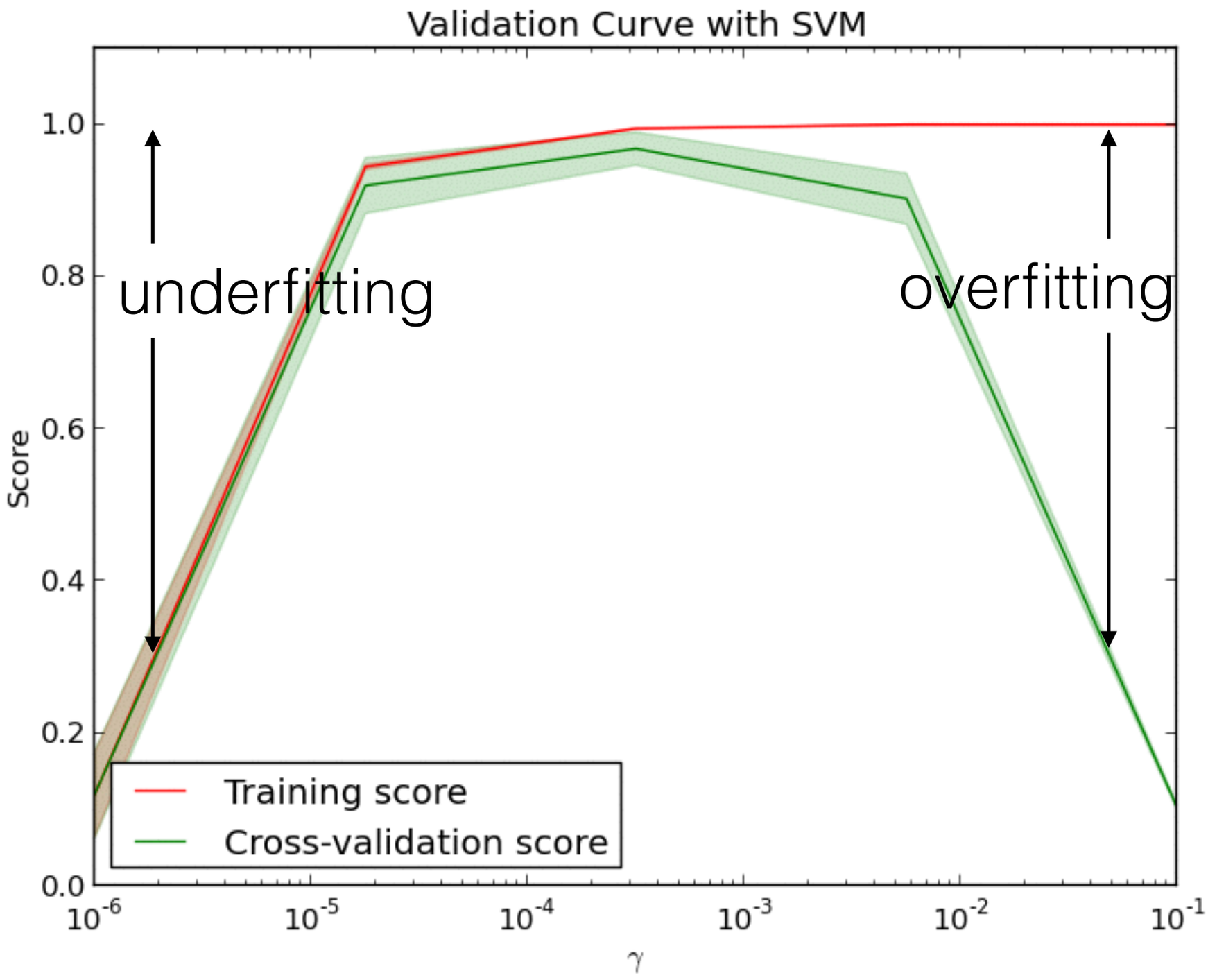
Cool new tools

to better understand your models

Validation Curves



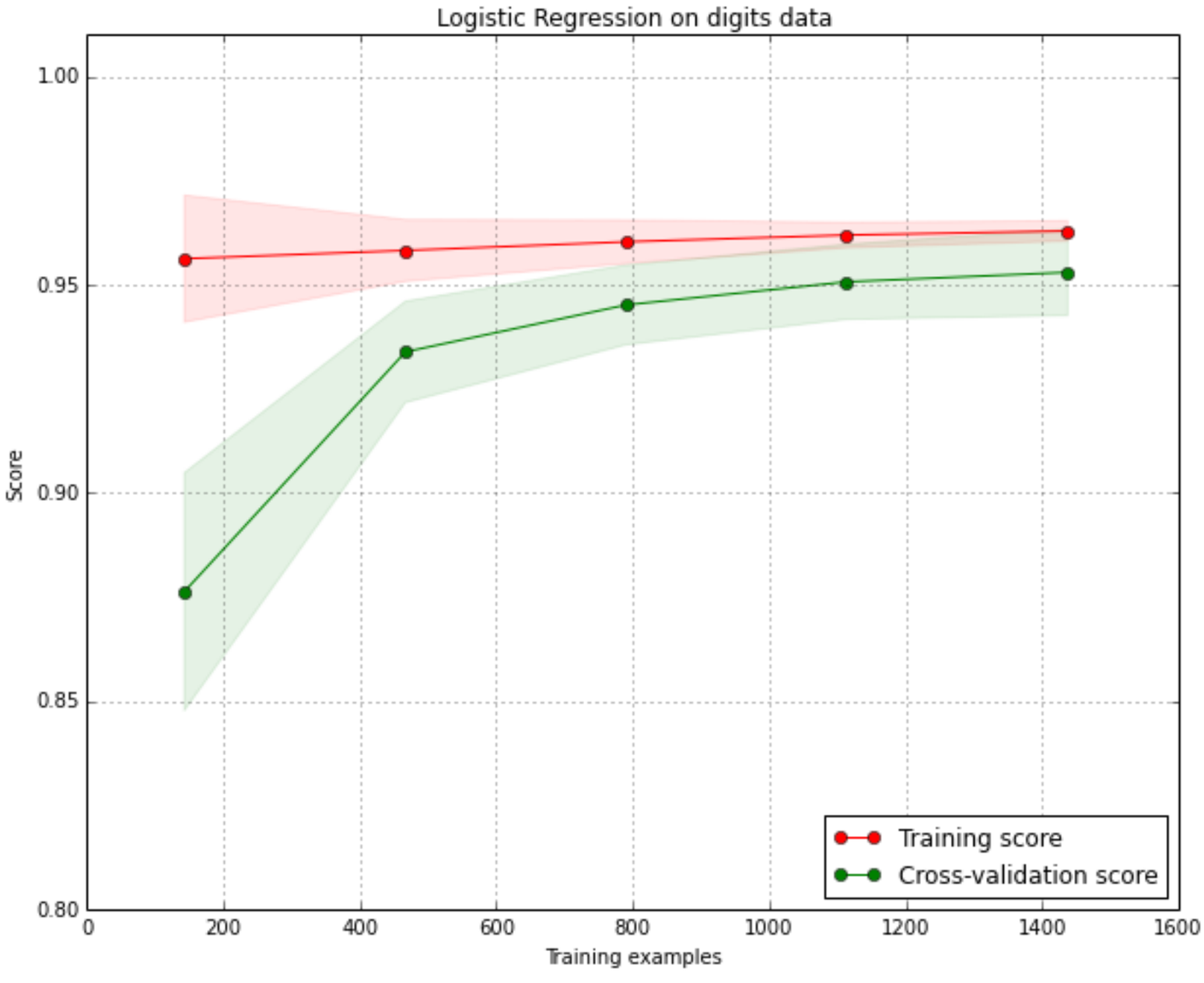
Validation Curves



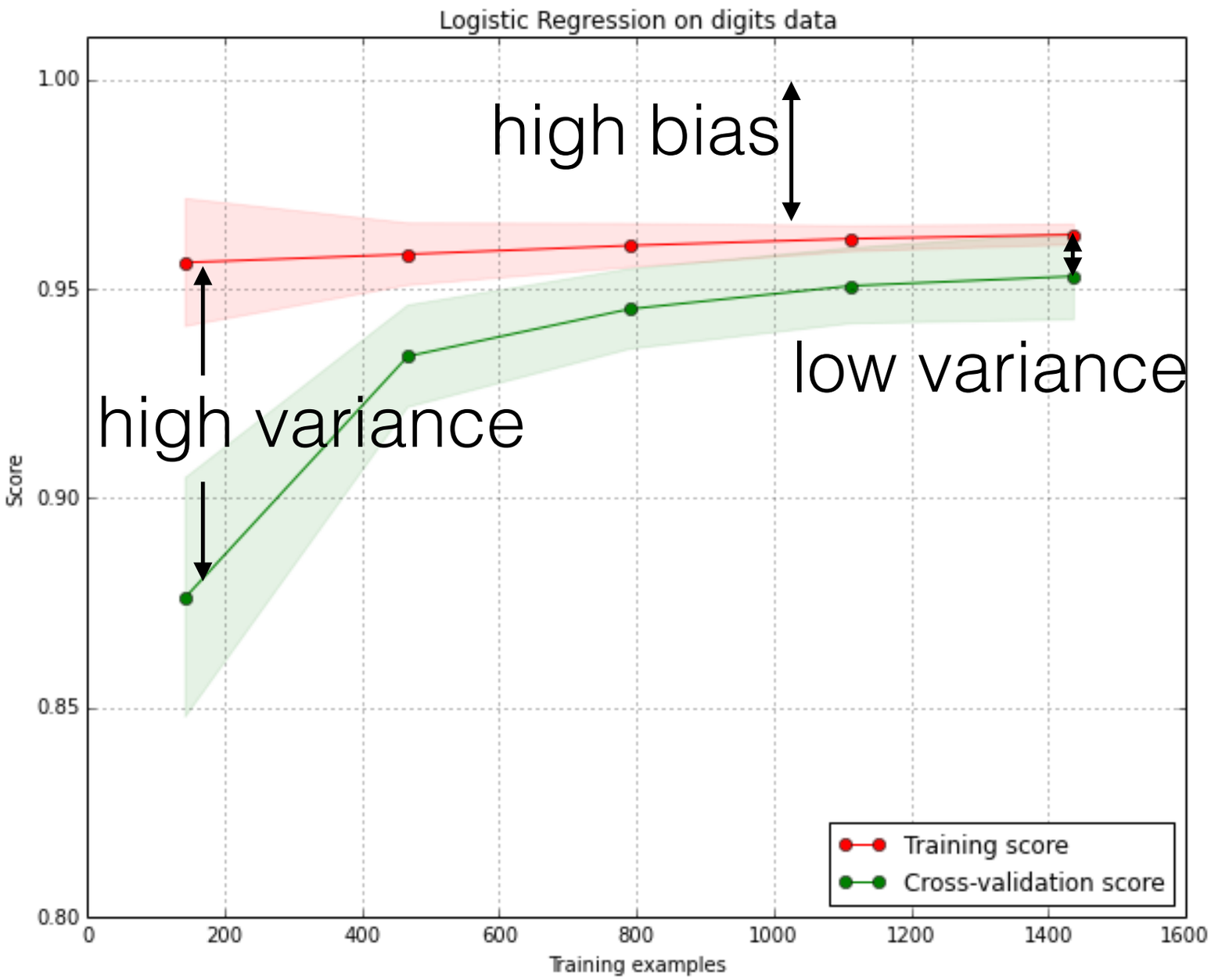
```
>>> import numpy as np
>>> from sklearn.learning_curve import validation_curve
>>> from sklearn.datasets import load_iris
>>> from sklearn.linear_model import Ridge

>>> np.random.seed(0)
>>> iris = load_iris()
>>> X, y = iris.data, iris.target
>>> indices = np.arange(y.shape[0])
>>> np.random.shuffle(indices)
>>> X, y = X[indices], y[indices]

>>> train_scores, valid_scores = validation_curve(Ridge(), X, y, "alpha",
...                                               np.logspace(-7, 3, 3))
>>> train_scores
array([[ 0.94...,  0.92...,  0.92...],
       [ 0.94...,  0.92...,  0.92...],
       [ 0.47...,  0.45...,  0.42...]])
>>> valid_scores
array([[ 0.90...,  0.92...,  0.94...],
       [ 0.90...,  0.92...,  0.94...],
       [ 0.44...,  0.39...,  0.45...]])
```

Online documentation on validation curves

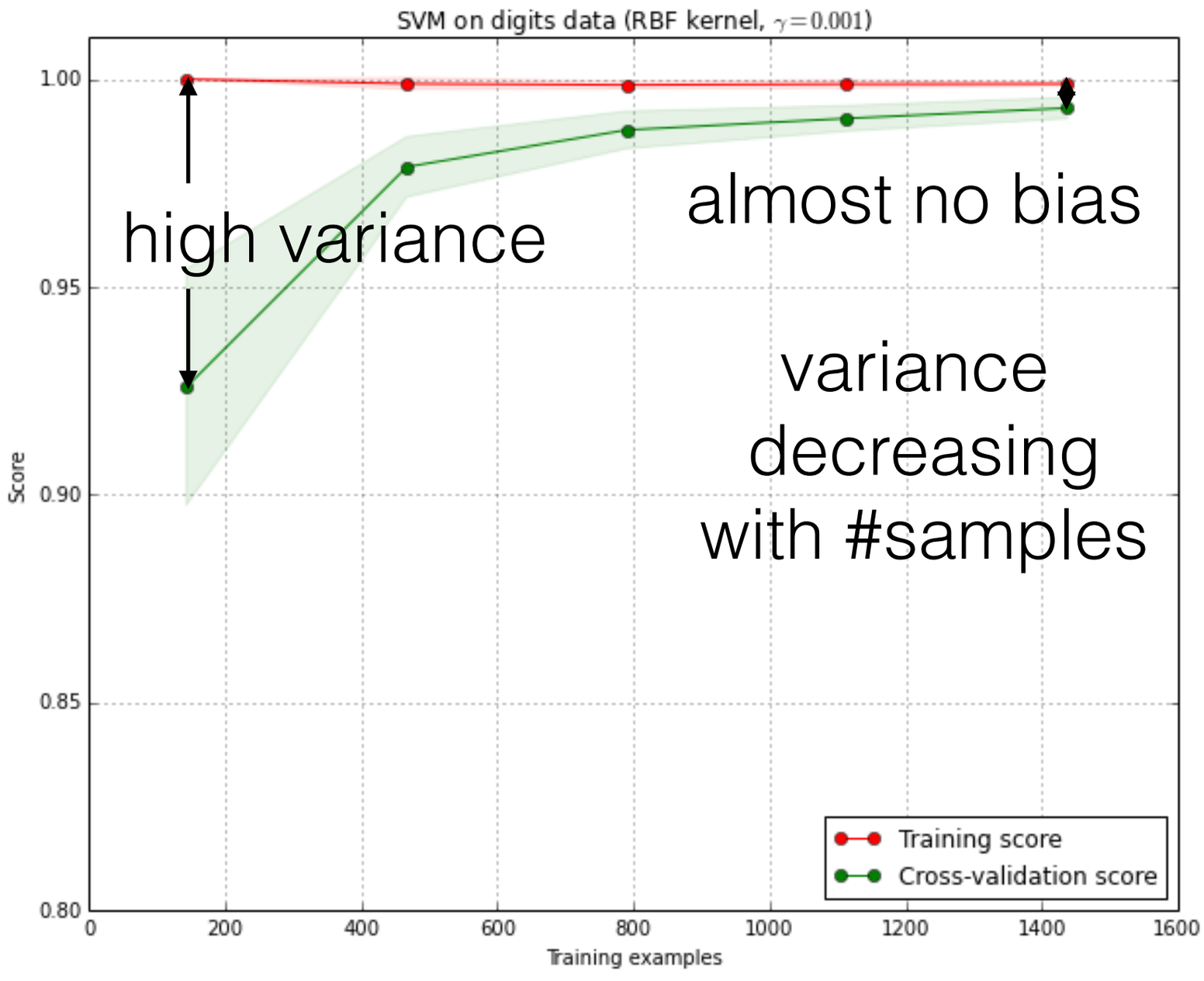
Learning curves for logistic regression



Learning curves for logistic regression



Learning curves on kernel SVM



```
>>> from sklearn.learning_curve import learning_curve
>>> from sklearn.svm import SVC

>>> train_sizes, train_scores, valid_scores = learning_curve(
...     SVC(kernel='linear'), X, y, train_sizes=[50, 80, 110], cv=5)
>>> train_sizes
array([ 50, 80, 110])
>>> train_scores
array([[ 0.98...,  0.98 ,  0.98...,  0.98...,  0.98...],
       [ 0.98...,  1.   ,  0.98...,  0.98...,  0.99...]])
>>> valid_scores
array([[ 1. ,  0.93...,  1. ,  1. ,  0.96...],
       [ 1. ,  0.96...,  1. ,  1. ,  0.96...],
       [ 1. ,  0.96...,  1. ,  1. ,  0.96...]])
```

Online documentation on learning curves

make_pipeline

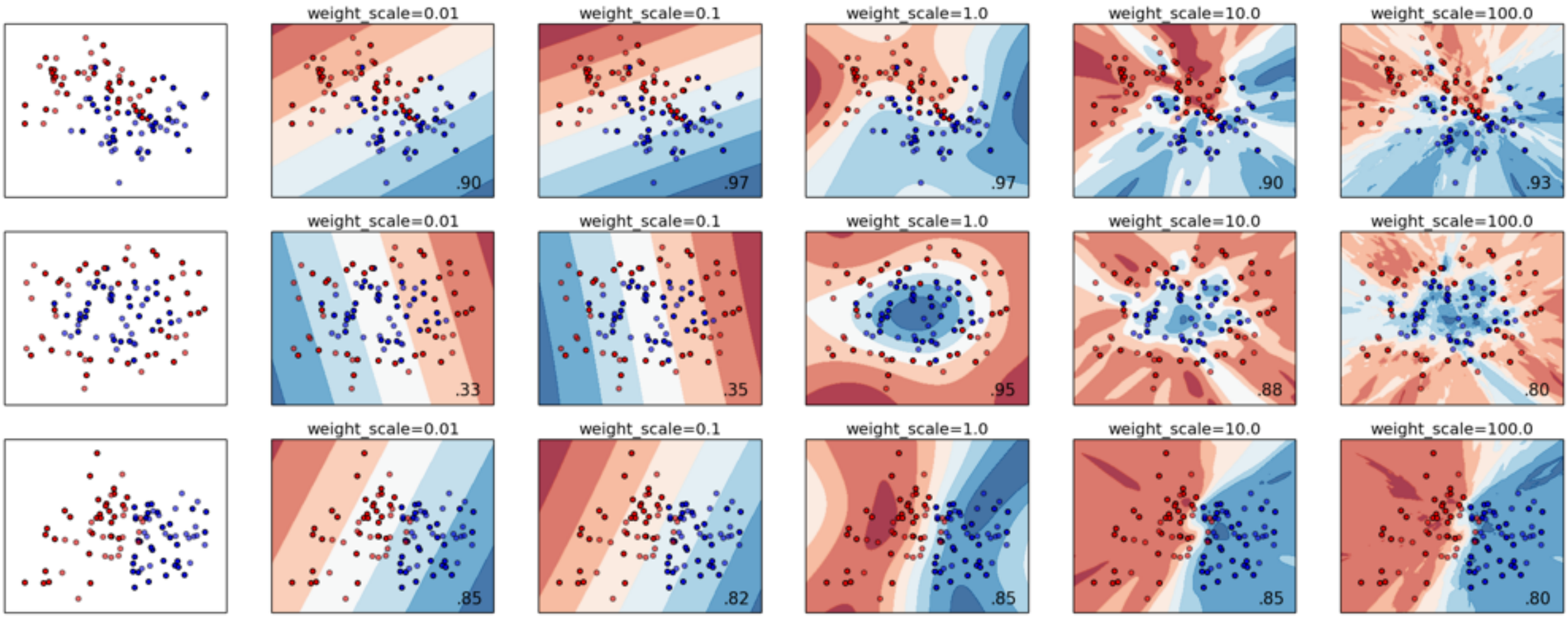
```
>>> from sklearn.pipeline import make_pipeline
>>> from sklearn.naive_bayes import GaussianNB
>>> from sklearn.preprocessing import StandardScaler

>>> p = make_pipeline(StandardScaler(), GaussianNB())
```

Ongoing work in the master branch

Neural Networks (GSoC)

- Multiple Layer Feed Forward neural networks (MLP)
  - `lbgfs` or `sgd` solver with configurable number of hidden layers
  - `partial_fit` support with `sgd` solver
    - scikit-learn/scikit-learn#3204
- Extreme Learning Machine
  - RP + non-linear activation + linear model
  - Cheap alternative to MLP, Kernel SVC or even `Nystroem`
  - scikit-learn/scikit-learn#3204

Impact of RP weight scale on ELMs



Incremental PCA

- PCA class with a `partial_fit` method
- Constant memory usage, supports for out-of-core learning e.g. from the disk in one pass.
- To be extended to leverage the `randomized_svd` trick to speed up when:

  `n_components << n_features`

- PR scikit-learn/scikit-learn#3285

Better pandas support

- CV-related tools now leverage `.iloc` based indexing without array conversion

- Estimators now leverage NumPy's `__array__` protocol implemented by `DataFrame` and `Series`

- Homogeneous feature extraction still required, e.g. using sklearn_pandas transformers in a `Pipeline`

Much much more

- Better sparse feature support, in particular for ensembles of trees (GSoC)

- Fast Approximate Nearest neighbors search with LSH Forests (GSoC)

- Many linear model improvements, e.g. `LogisticRegressionCV` to fit on a regularization path with warm restarts (GSoC)

- https://github.com/scikit-learn/scikit-learn/pulls

Personal plans for future work

Refactored joblib concurrency model

- Use pre-spawned workers without multiprocessing fork (to avoid issues with 3rd party threaded libraries)

- Make workers scheduler-aware to support nested parallelism: e.g. cross-validation of GridSearchCV

- Automatically batch short-running tasks to hide dispatch overhead, see joblib/joblib#157

- Make it possible to delegate queueing scheduling to 3rd party cluster runtime:

  - SGE, IPython.parallel, Kubernetes, PySpark

Thank you!

- http://scikit-learn.org

- https://github.com/scikit-learn/scikit-learn

@ogrisel