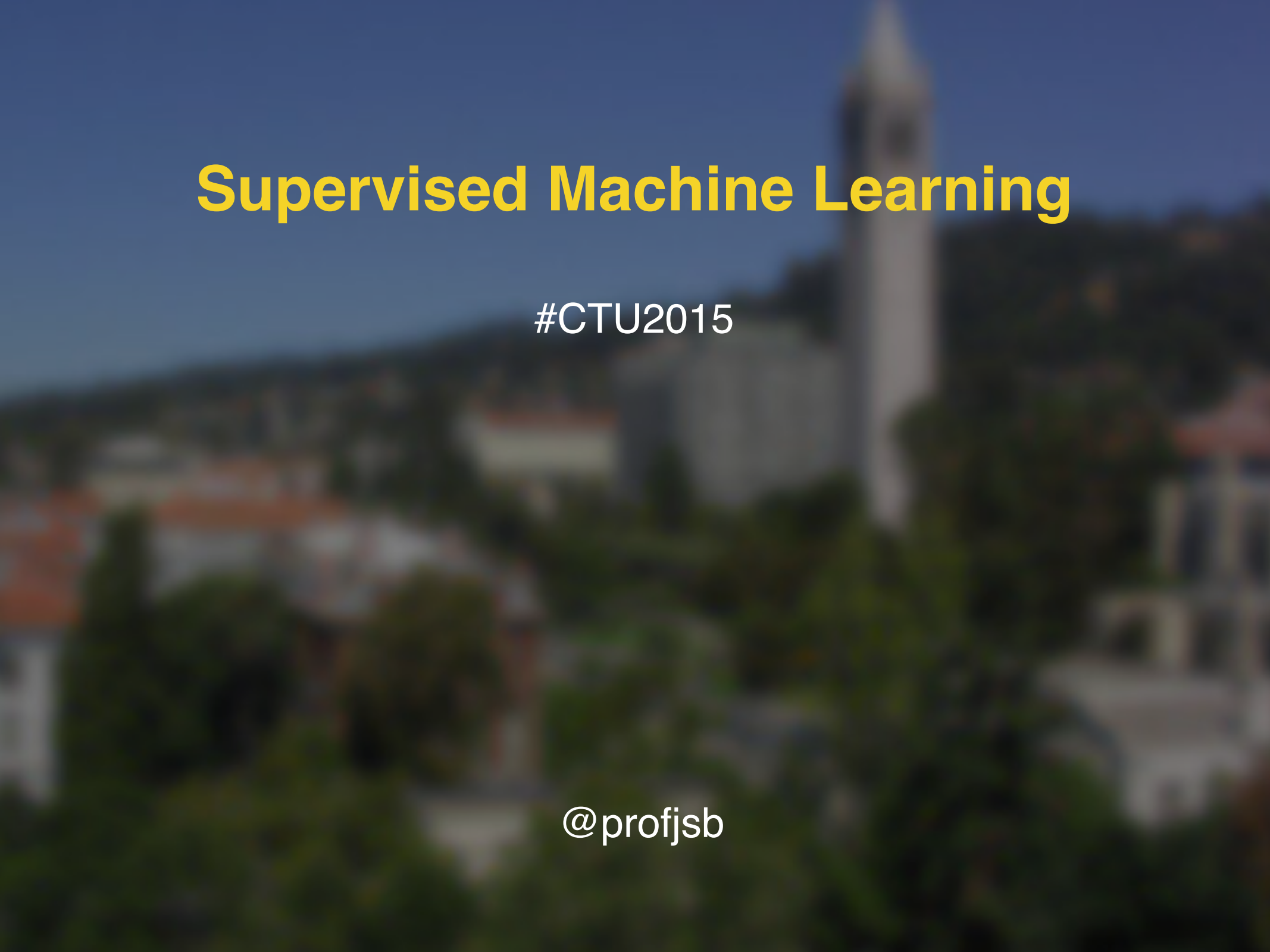
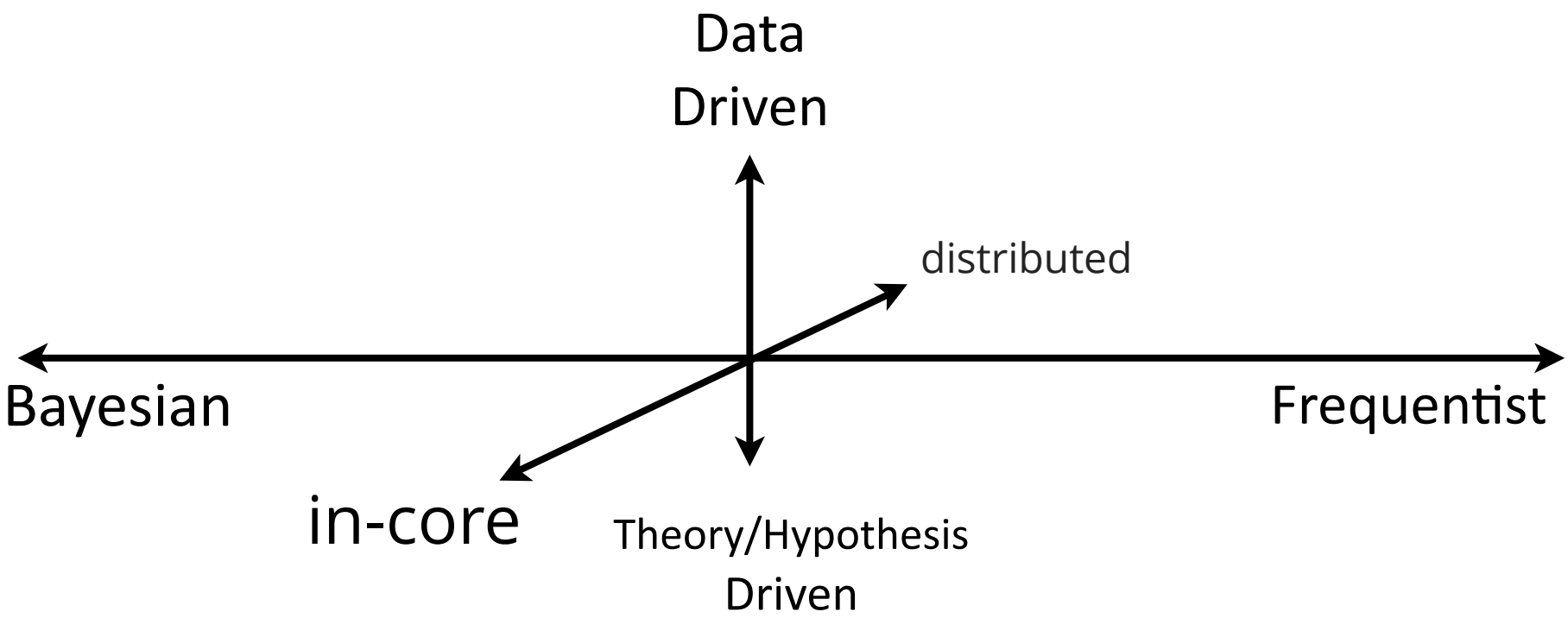

Supervised Machine Learning
#CTU2015
@profjsb

Data Inference Space



Hardware	laptops → clusters/supercomputers
Software	Spark, Sklearn, Python/Scipy, R, ...
Carbonware	(astro) grad students, postdocs

1. What is machine learning?

- Flavors and facets of machine learning
  - supervised, semi-supervised, clustering, ...
  - classification / regression
- When to use it, when not to
- scikit-learn
- testing/validation sets, cross-validation
- metrics: ROC, AUC, confusion matrix

2. Regression

- Logistic regression, kNN, random forest

[breakout: predict quasar redshifts from photometric data]

3. Classification

- SVM, random forest, deep learning

[breakout: predict Star/Galaxy/QSO from photometric data]

4. Improving your models

- hyperparameter optimization
  ```GridSearchCV```
- dealing with missing data
- Feature selection / feature importance
- feature engineering

[breakout: redo Star/Galaxy/QSO from photometric data]

5. Considerations in getting into production

- multicore / multimachine
- scikit-learn pipelines
- Bigdata machine learning: Graphlab, MLlib (Spark)
  ```

What is Machine Learning?

Short Answer: The offspring of Statistics and Computer Science

Better Answer: A set of models which aim to learn something about a data set to apply that knowledge to new data

- Using labels from *training* data to classify new objects (e.g. images, digits, webpages)
- Learning the relationship between explanatory *features* and response variable to predict for new data (e.g., stock market)
- Discovering natural clustering structure in data
- Detecting low-dimensional structure in high-dimensional data
- Finding outliers in large data sets

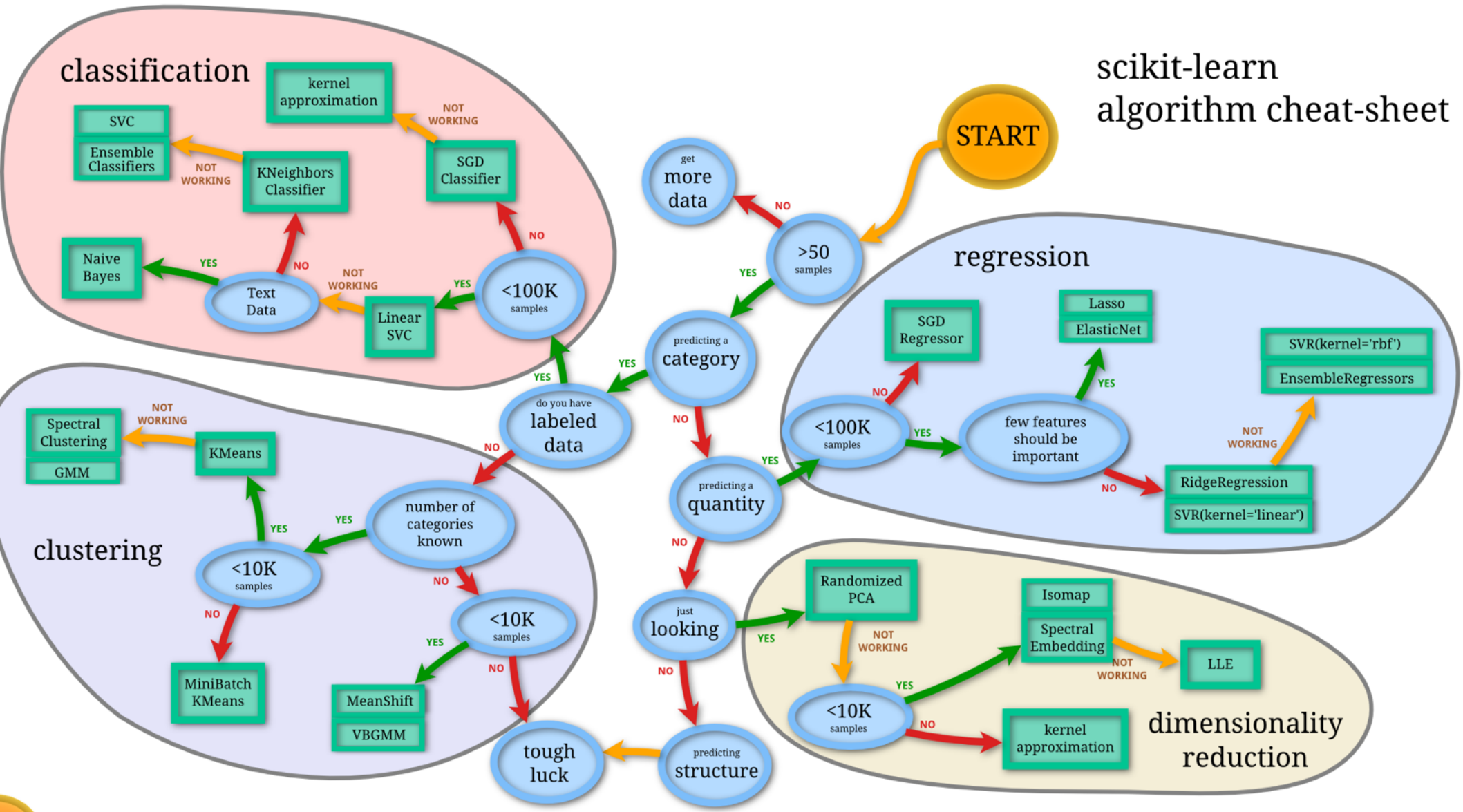
Machine Learning with scikit-learn

Make sure you have version >= 0.15.1 installed

```
conda update scikit-learn
pip install scikit-learn
```

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

also: http://mlpy.sourceforge.net/, http://orange.biolab.si/


scikit-learn
algorithm cheat-sheet
START
get more data
>50 samples
predicting a category
do you have labeled data
predicting a quantity
just looking
predicting structure
tough luck
classification
kernel approximation
SVC
Ensemble Classifiers
KNeighbors Classifier
SGD Classifier
Naive Bayes
Text Data
Linear SVC
<100K samples
regression
SGD Regressor
Lasso
ElasticNet
SVR(kernel='rbf')
EnsembleRegressors
<100K samples
few features should be important
RidgeRegression
SVR(kernel='linear')
clustering
Spectral Clustering
GMM
KMeans
number of categories known
<10K samples
MiniBatch KMeans
<10K samples
MeanShift
VBGMM
dimensionality reduction
Randomized PCA
Isomap
Spectral Embedding
LLE
<10K samples
kernel approximation
YES
NO
NOT WORKING

Supervised Learning

Use *training* set of (*x*,y) pairs to learn to predict y for new *x*

Regression - predicting continuous *outcome* (y) variable from a vector of input *features* (*x*)

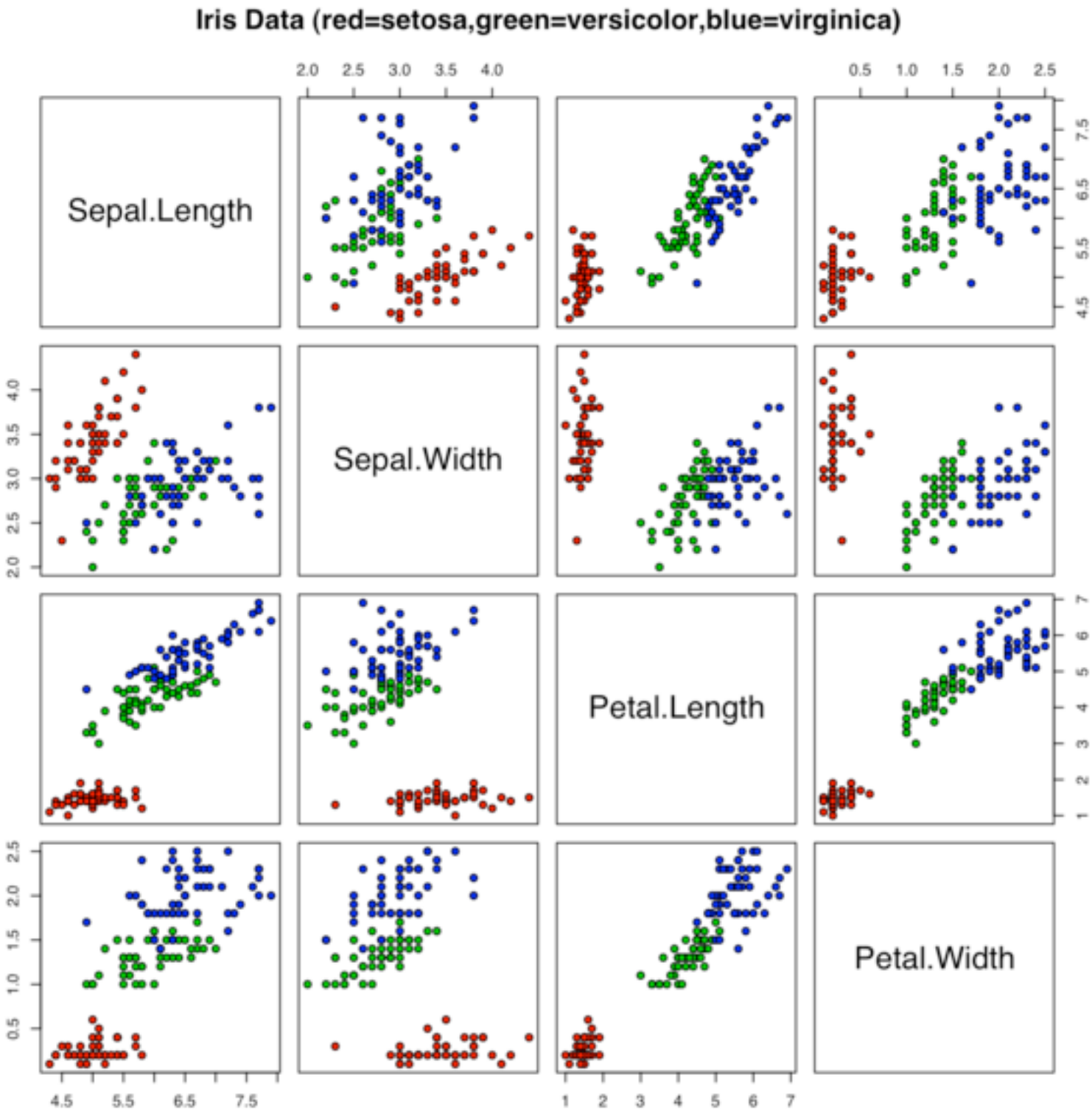
- Linear Regression: `linear_model.LinearRegression`
- Lasso & Ridge Reg.: `linear_model.Lasso / linear_model.Ridge`
- Gaussian Process Regression: `gaussian_process.GaussianProcess`
- Nearest Neighbor Regression: `neighbors.KNeighborsRegressor`
- Support Vector Regression: `svm.SVR`
- Regression Trees: `tree.DecisionTreeRegressor`

... and more!

Notebook

Supervised Learning

Classification - predicting the discrete class (y) of an object from a vector of input *features* (***x***)



setosa

veriscolor

virginica

Supervised Learning

Classification - predicting the discrete class (y) of an object from a vector of input *features* (***x***)

e.g. $\mathbf{x}_i$ = [5.1, 3.5, 1.4, 0.2]

[sepal length (cm), sepal width (cm), petal length (cm), petal width (cm)]

e.g. y_1 = "Iris-Setosa"

For Iris: Number of (x, y) "instances" = 150
Number of classes = 3

Notebook...

Supervised Learning, cont.

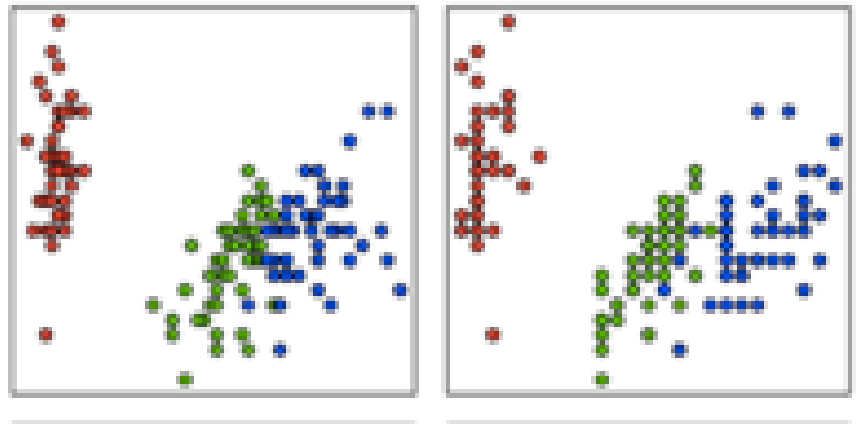
Classification - predicting the discrete class (y) of an object from a vector of input *features* (*x*)

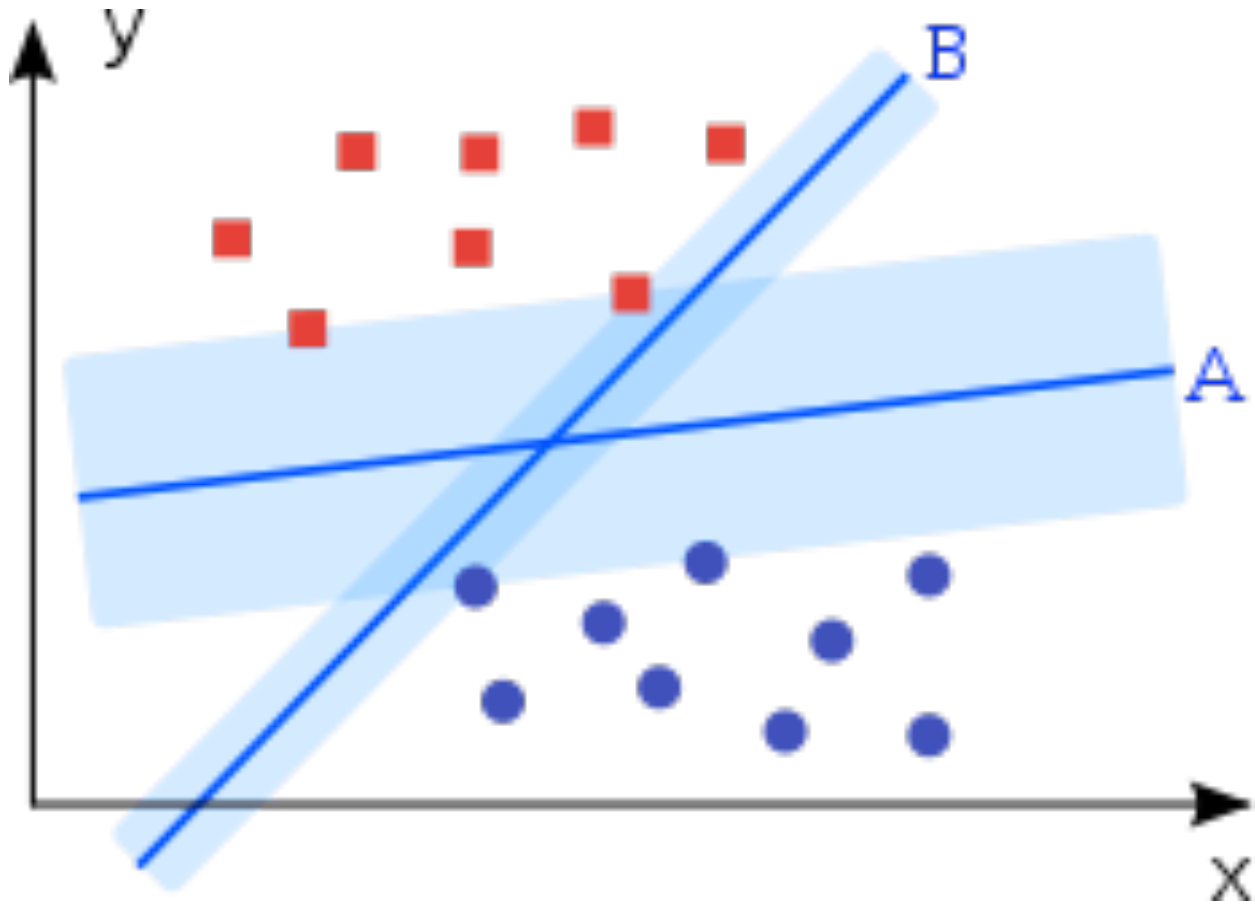
- Logistic Regression: `linear_model.LogisticRegression`
- KNN Classification: `neighbors.KNeighborsClassifier`
- LDA / QDA: `lda.LDA / lda.QDA`
- Naive Bayes: `naive_bayes.GaussianNB`
- Support Vector Machines: `svm.SVC`
- Classification Trees: `tree.DecisionTreeClassifier`
- Random Forest: `ensemble.RandomForestClassifier`
- Multi-class & multi-label Classification is supported:

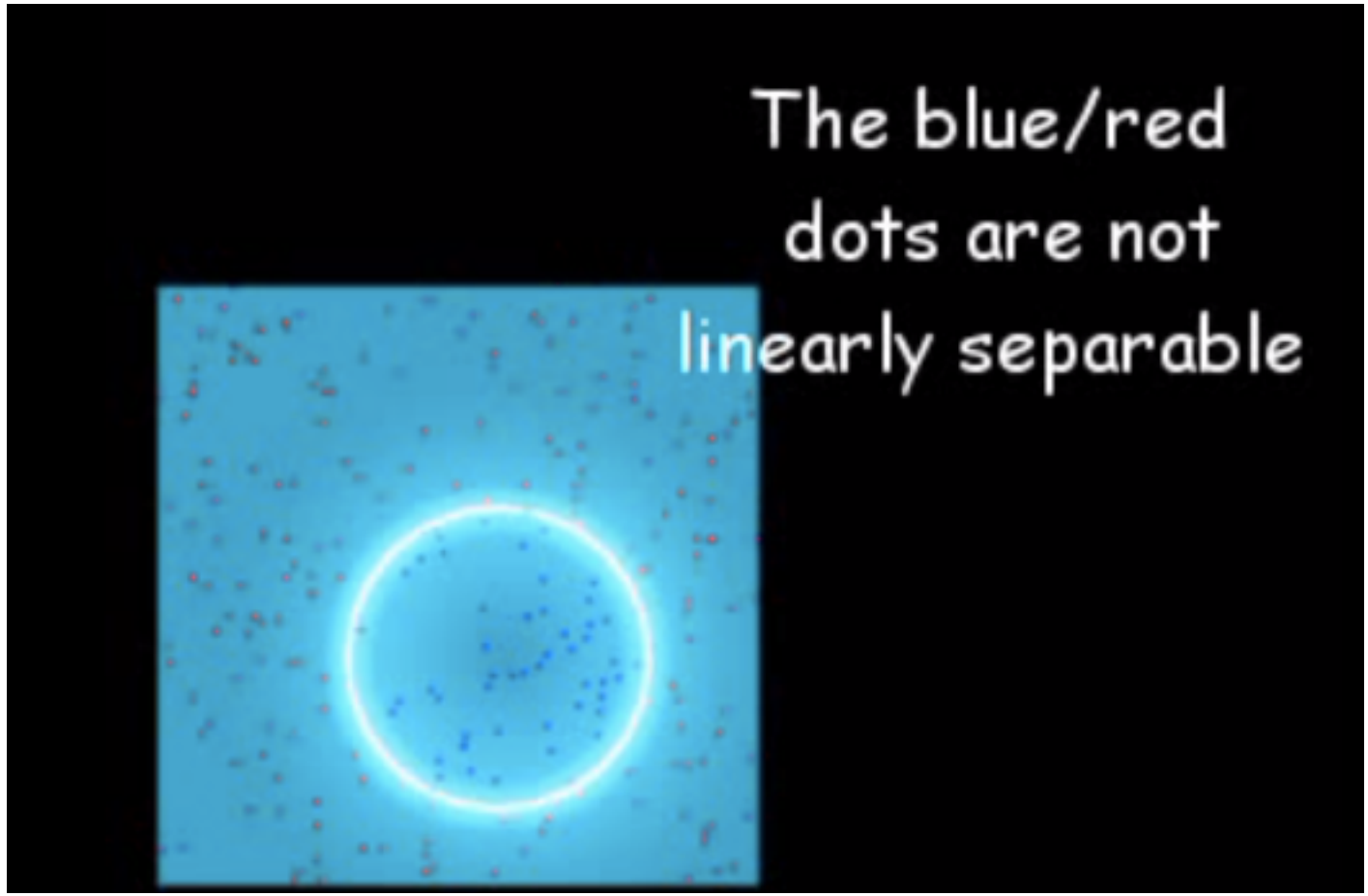
`multiclass.OneVsRest` `multiclass.OneVsOne`

- **`feature_selection:`** recursive, L1, and tree-based

Support Vector Machines (1992)





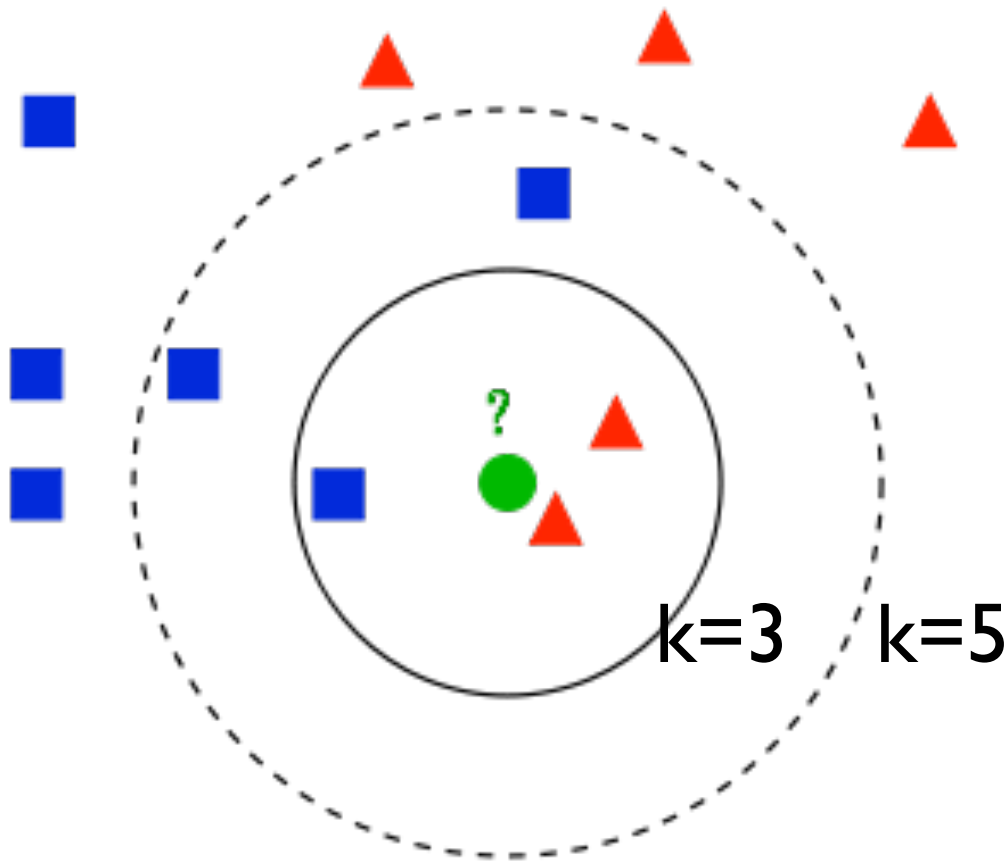


SVM with polynomial kernel visualization

http://www.youtube.com/watch?v=3liCbRZPrZA

Supervised Learning

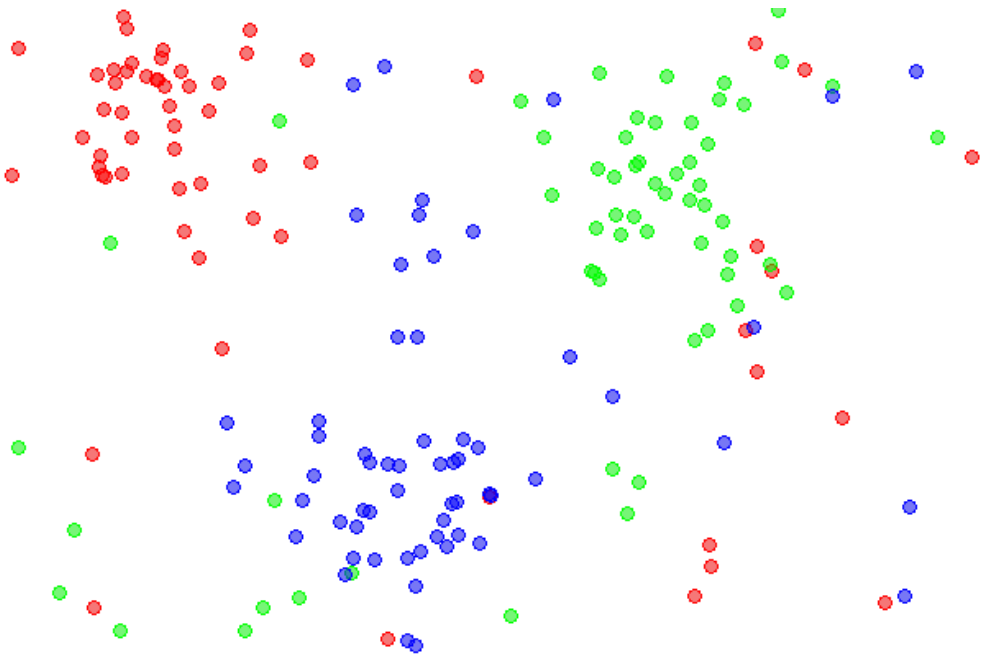
kNearestNeighbors (kNN)



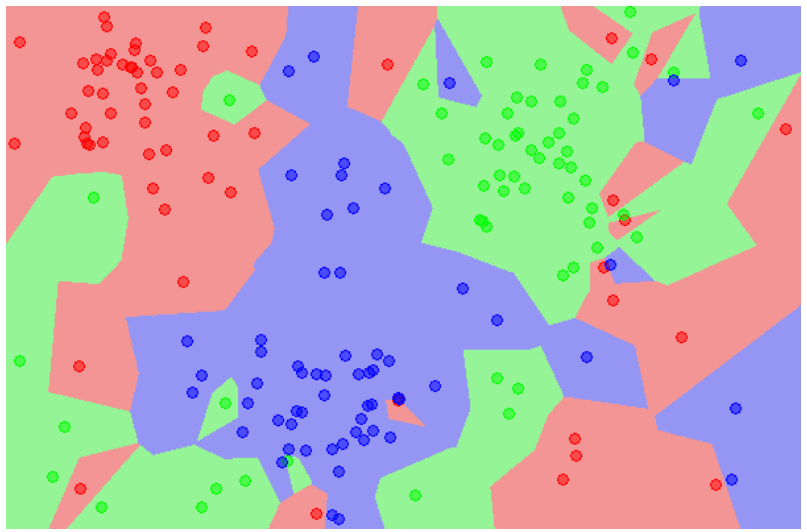
For each test point, **x** find the k-nearest instances in the training data

Classify the point according to the majority vote of their class labels

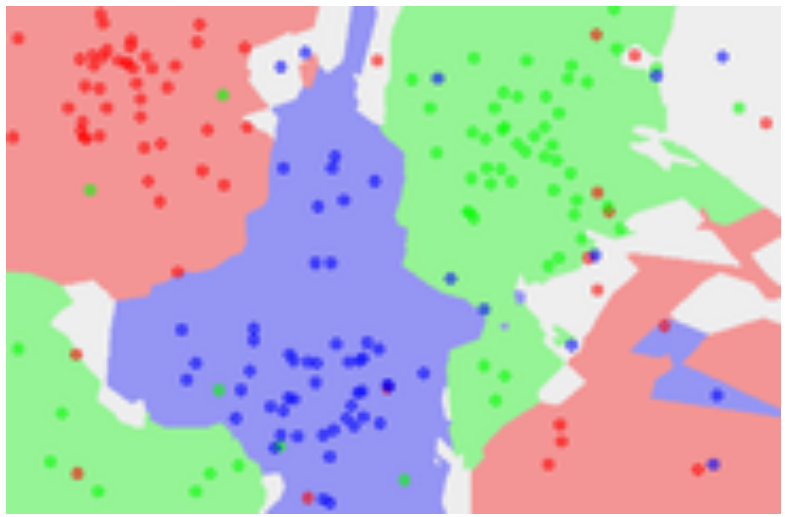
kNearestNeighbors (kNN)



Dataset



k=1

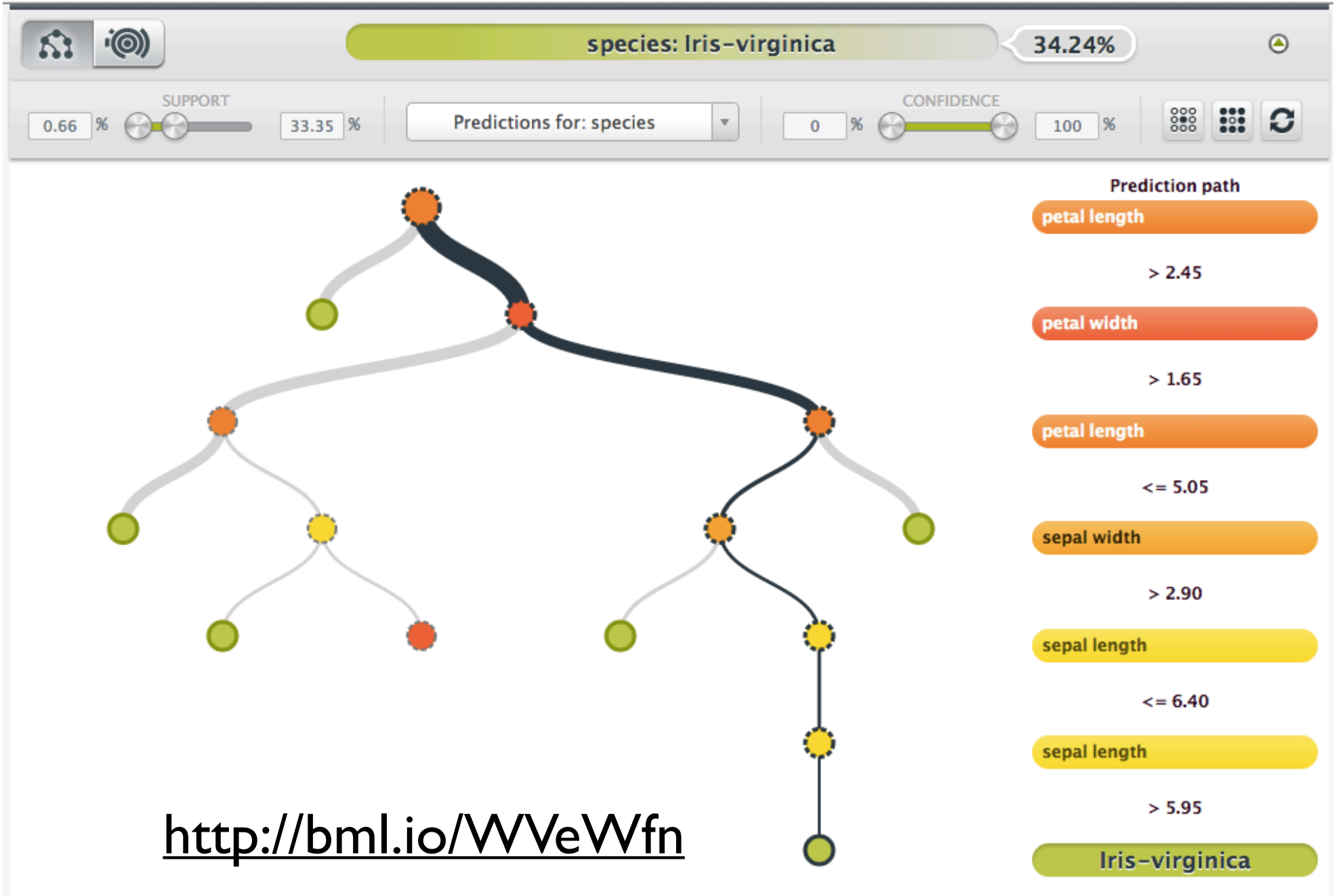


k=5

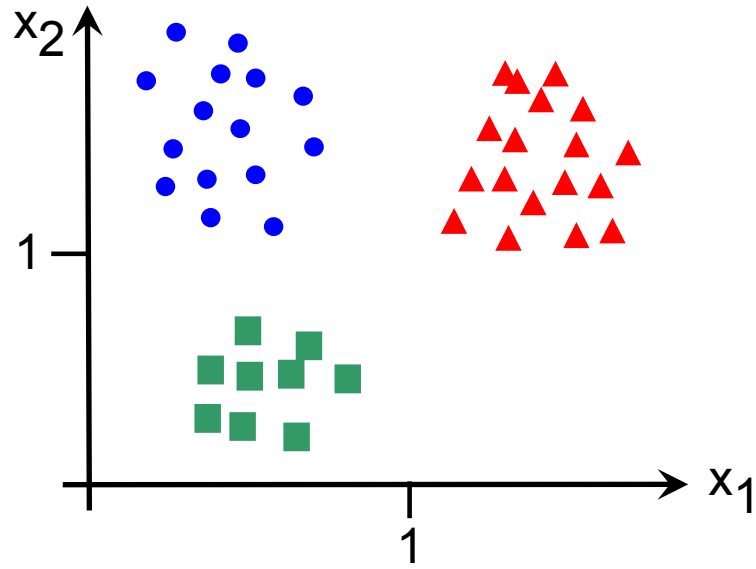
`neighbors.KNeighborsClassifier`

Decision Trees

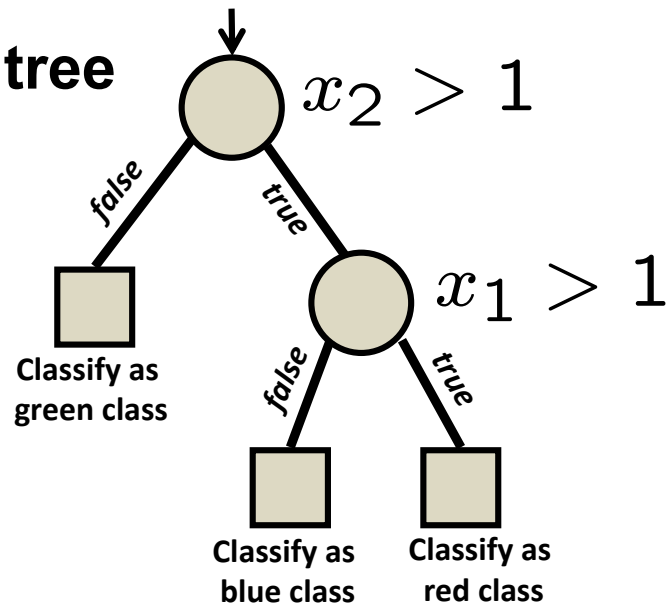
Classification and regression trees (CART)



http://bml.io/WVeWfn

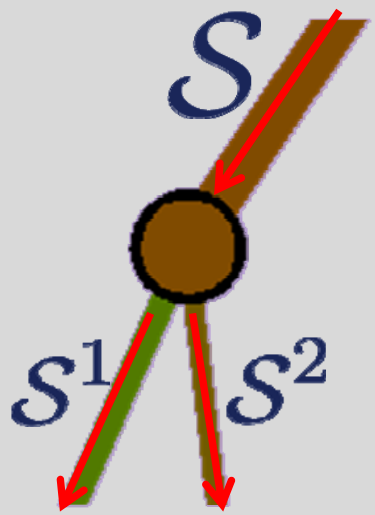


A decision tree



http://www.robots.ox.ac.uk/~az/lectures/ml/lect5.pdf

Building Trees Rigorously (Node Splitting Criteria)



Information gain

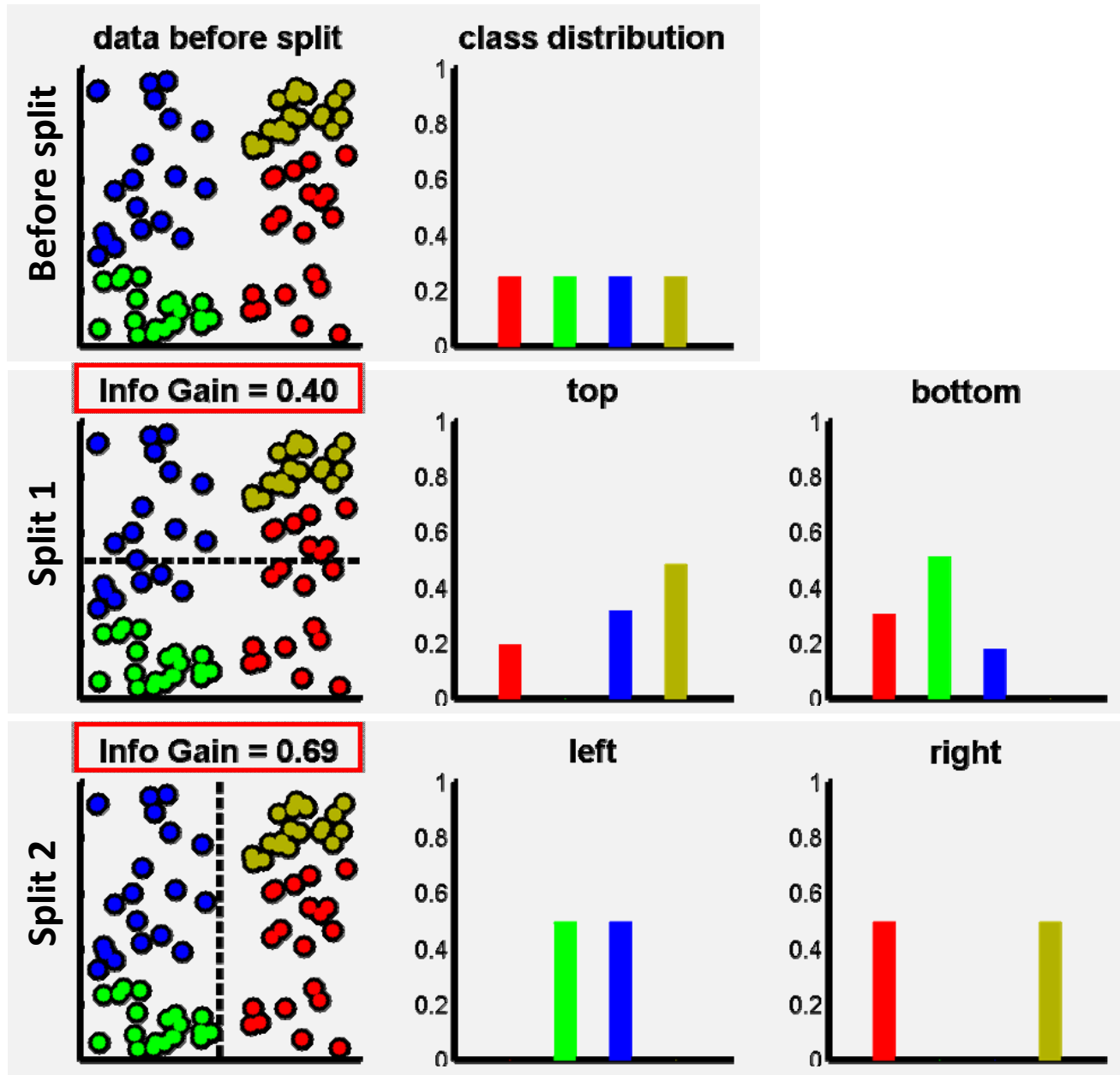
$$I = H(S) - \sum_{i \in \{1,2\}} \frac{|S^i|}{|S|} H(S^i)$$

Shannon's entropy

$$H(S) = -\sum_{c \in C} p(c) \log(p(c))$$

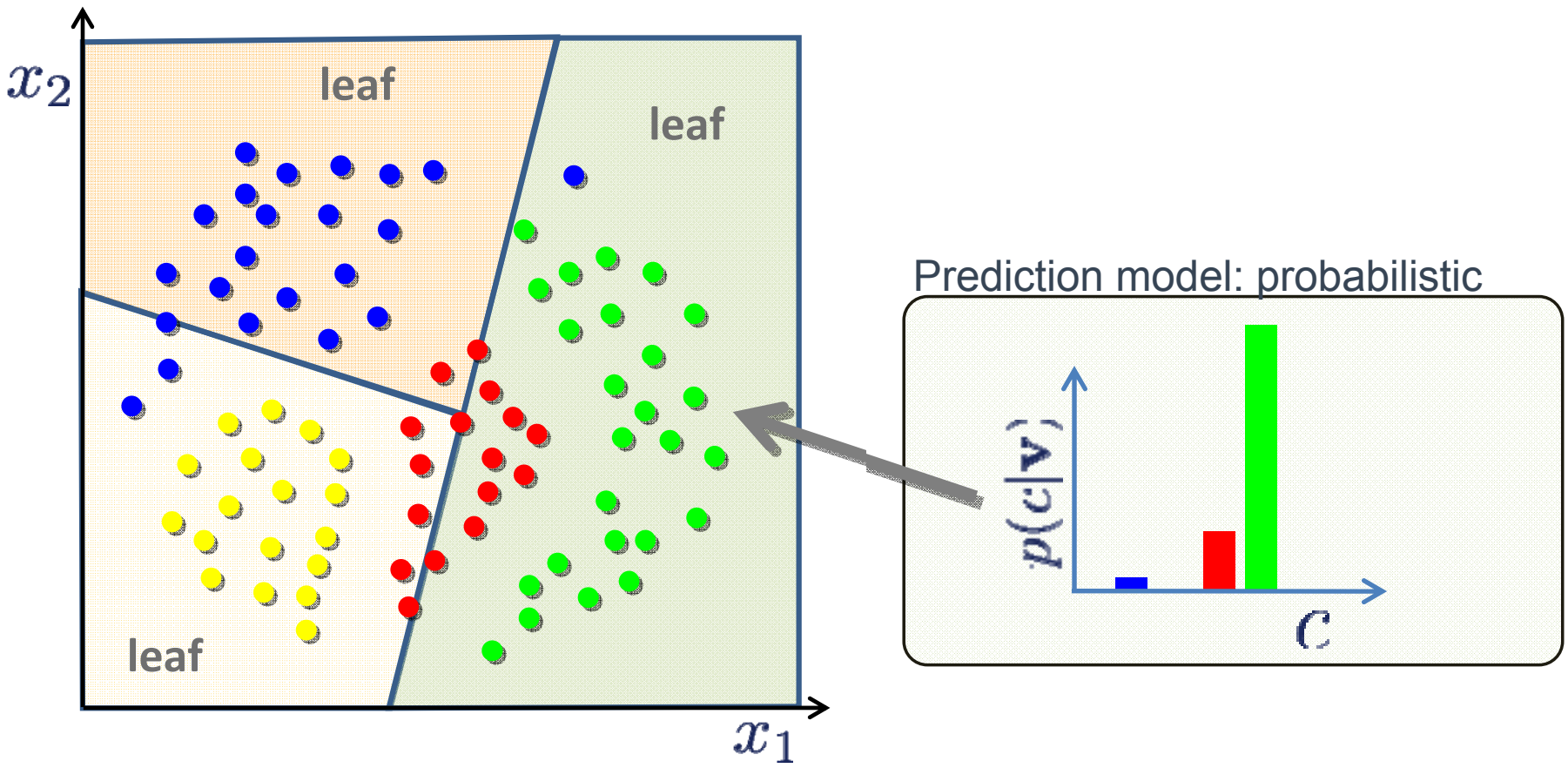
Node training

$$\theta^* = \arg\max_{\theta \in T} I$$



http://www.robots.ox.ac.uk/~az/lectures/ml/lect5.pdf

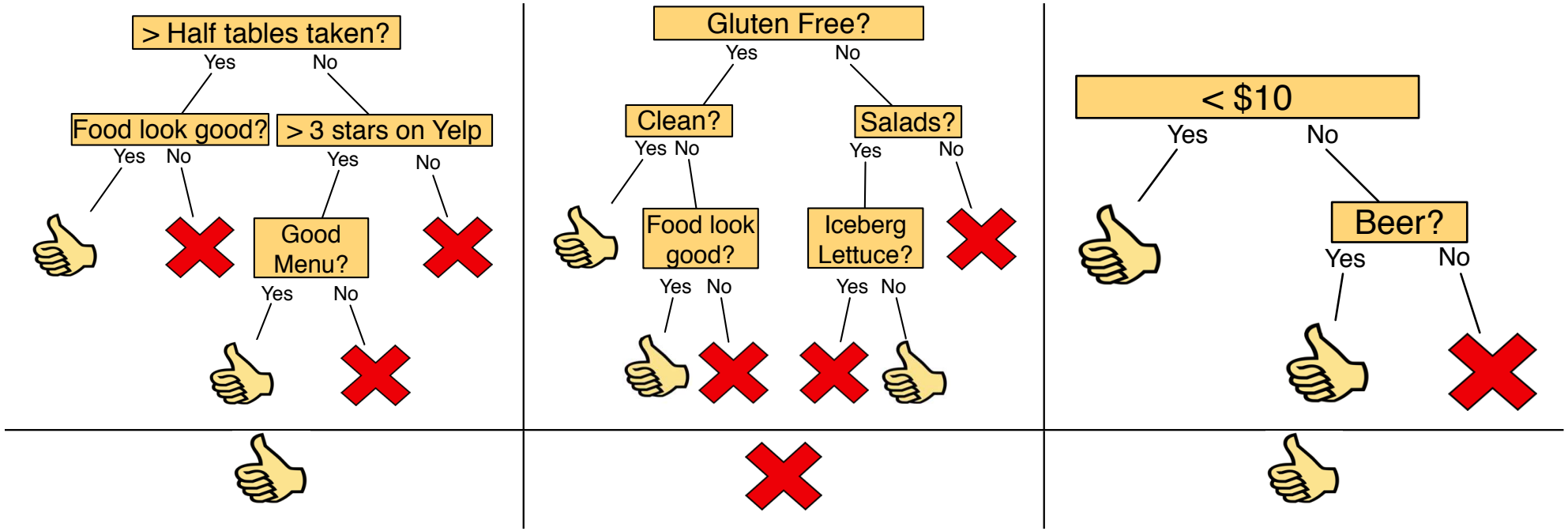
Making predictions from the Learned Tree



http://cs.stanford.edu/people/karpathy/svmjs/demo/

Collections of Trees ("Random Forests")

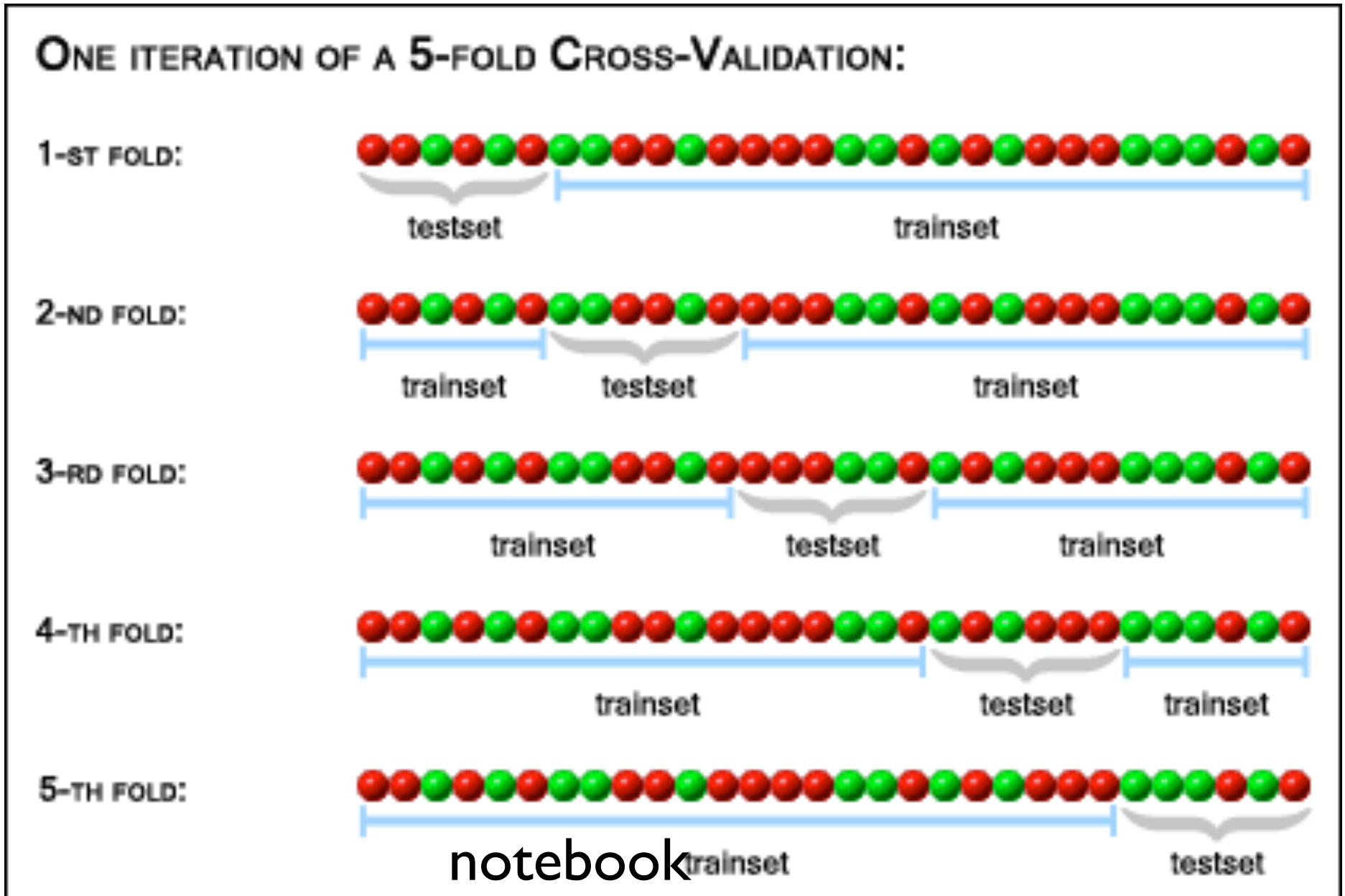
ensembles generally increase robustness



D. Eads

Cross-validation generators

`KFold` (n, k)	`StratifiedKFold` (y, k)	`LeaveOneOut` (n)	`LeaveOneLabelOut` (labels)
Split it K folds, train on K-1 and then test on left-out	It preserves the class ratios / label distribution within each fold.	Leave one observation out	Takes a label array to group observations



notebook

Error Estimation & Model Selection

Q: How do I choose which model and parameters to use?

KNN with what # of neighbors?
SVM which what kernel & bandwidth?
RF with how many trees and which mtry?
GP with what kernel & bandwidth?

Solution: use grid_search.GridSearchCV

`grid_search.`**`GridSearchCV`**(*estimator*, *param_grid*, *loss_func*, *n_jobs*, *cv=None*)

Computes *cv*-fold cross-validated *loss_func* (or *score_func*) of estimator over a *param_grid* on *n_jobs* cores, and returns the best model!

Error Estimation & Model Selection

Q: What evaluation metrics are available?

Loss Functions

metrics.zero_one(y_true, y_pred)
Zero-One classification loss
metrics.hinge_loss(y_true, pred_decision[, ...])
Cumulated hinge loss (non-regularized).
metrics.mean_square_error(y_true, y_pred)
Mean square error regression loss

Or write your own!!

Score Functions

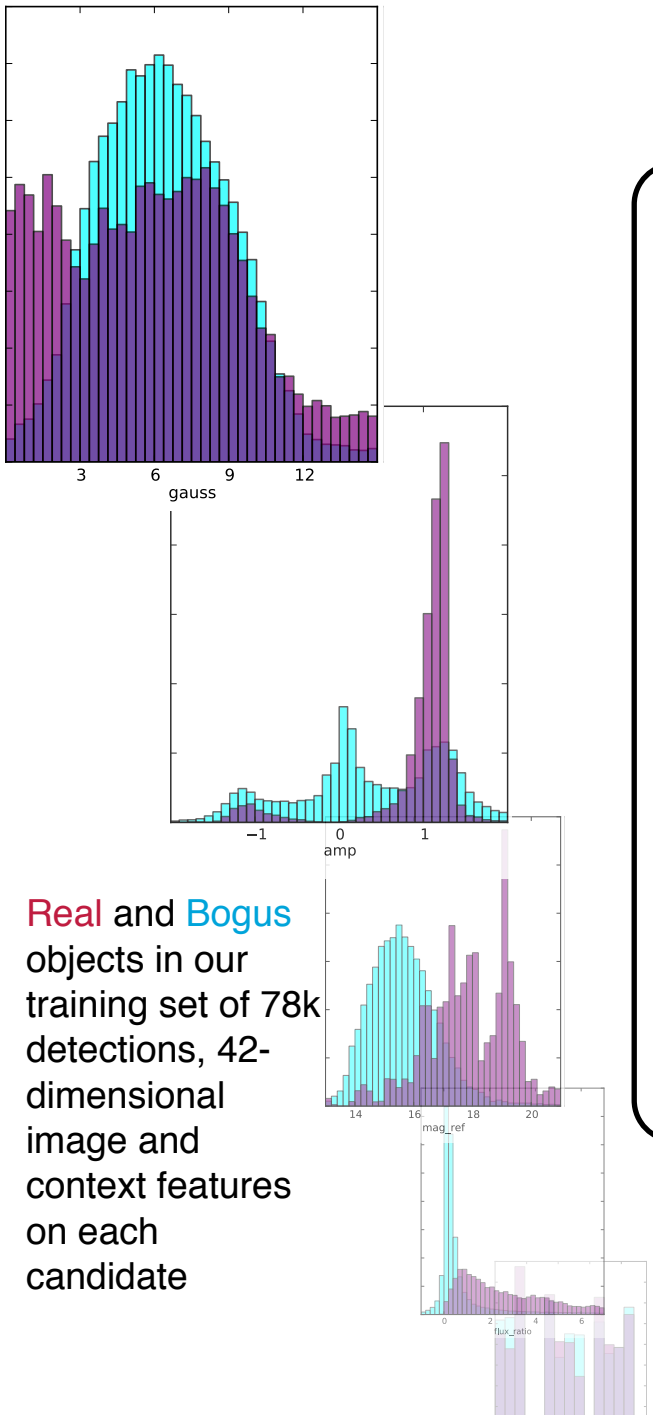
metrics.zero_one_loss(y_true, y_pred)
Zero-One classification score
metrics.auc(x, y)
Compute Area Under the Curve (AUC)
metrics.precision_score(y_true, y_pred[, ...])
Compute the precision
metrics.recall_score(y_true, y_pred[, pos_label])
Compute the recall
metrics.fbeta_score(y_true, y_pred, beta[, ...])
Compute fbeta score
metrics.f1_score(y_true, y_pred[, pos_label])
Compute f1 score

Evaluation Plots

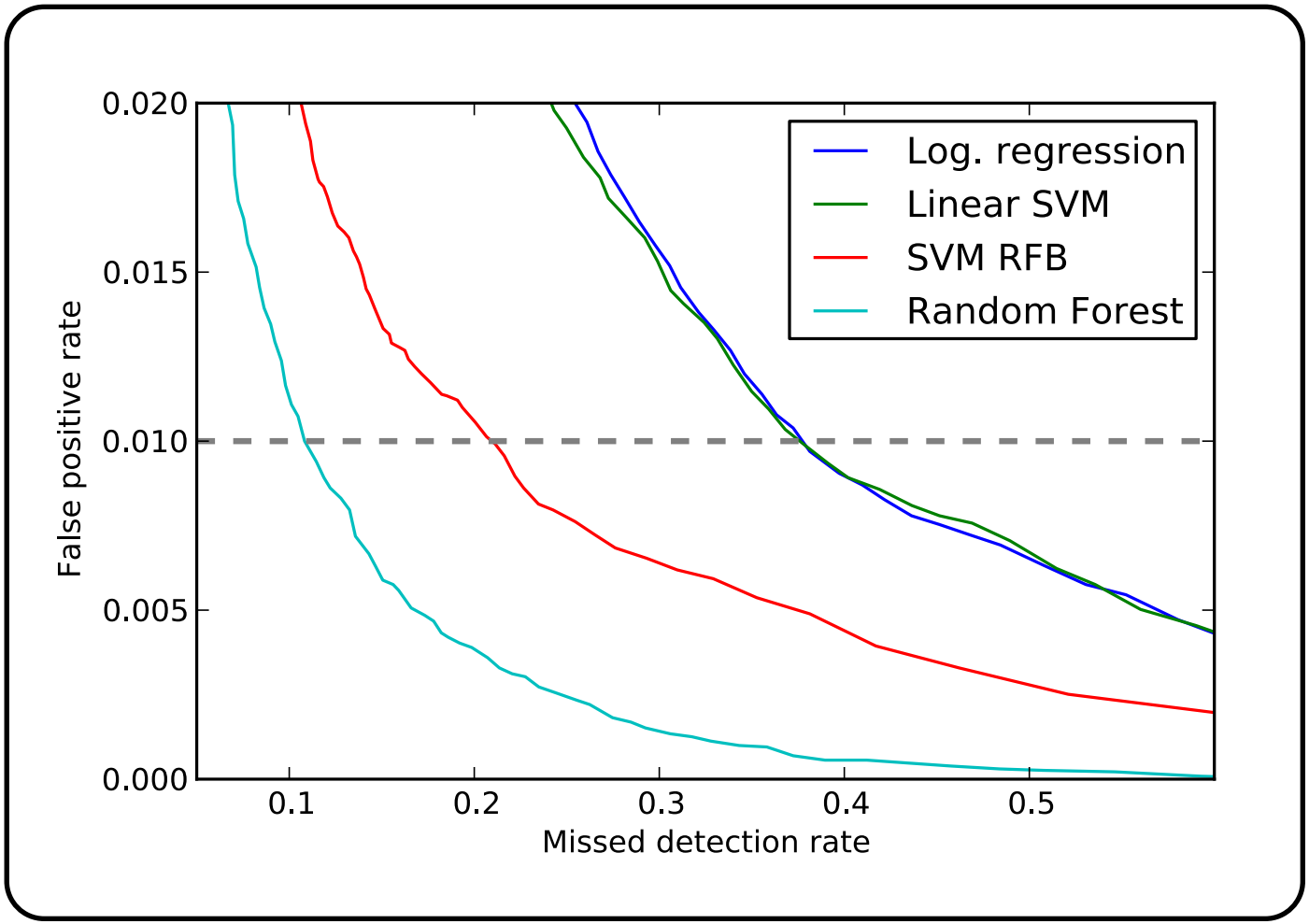
metrics.confusion_matrix(y_true, y_pred[, ...]) Compute confusion matrix to evaluate the accuracy of a classification
metrics.roc_curve(y_true, y_score) Compute Receiver operating characteristic (ROC)
metrics.precision_recall_curve(y_true, ...) Compute precision-recall pairs for different probability thresholds

Some classifiers work better than others

"ROC Curve"



Real and Bogus objects in our training set of 78k detections, 42-dimensional image and context features on each candidate



Brink+2012

arXiv.org > astro-ph > arXiv:1209.3775

To the Notebook!

Breakout

Classify the famous **Iris** data, first used by R.A Fisher.

To load in the data and split into training / testing sets:

```
iris = datasets.load_iris()
Xtr = iris.data[::2,:]
Xte = iris.data[1::2,:]
Ytr = iris.target[::2]
Yte = iris.target[1::2]
```

1. Choose your favorite classification model.
2. Find the parameters that minimize the 3-fold cross-validation misclassification rate over the training set.
3. Apply this optimized model to predict the class of each object in the held-out set.

a) What is your best 3-fold CV 0-1 score?

b) What is your 0-1 score when applying it to testing set?

Kinect – random forest classifier

Input data for each frame

Output

RGB image

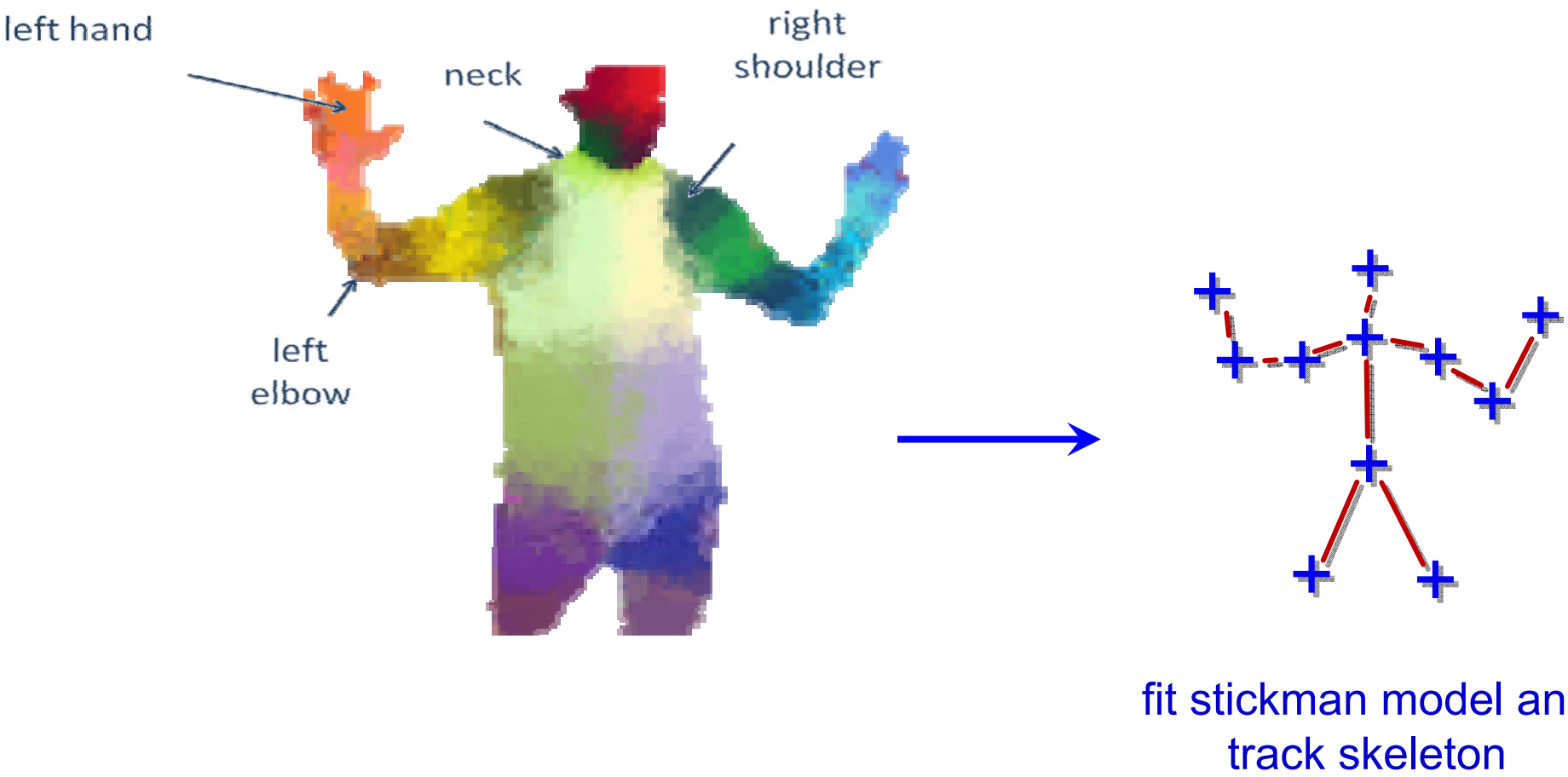
Depth image

Multi-class classification

inferred body parts

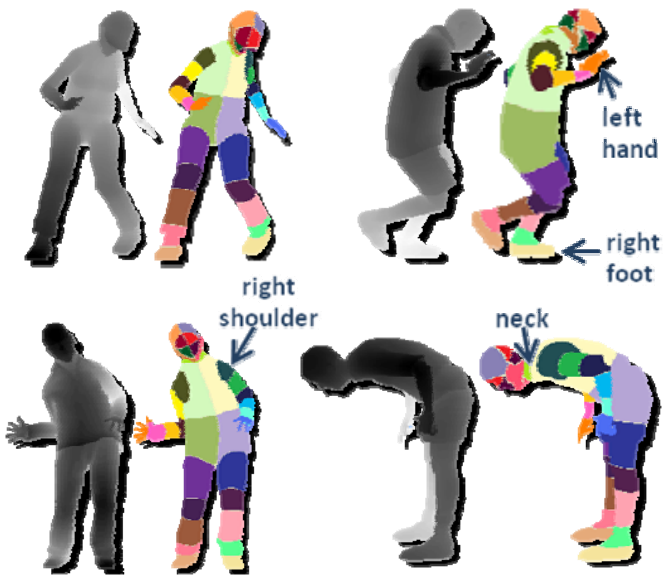
http://www.robots.ox.ac.uk/~az/lectures/ml/lect5.pdf

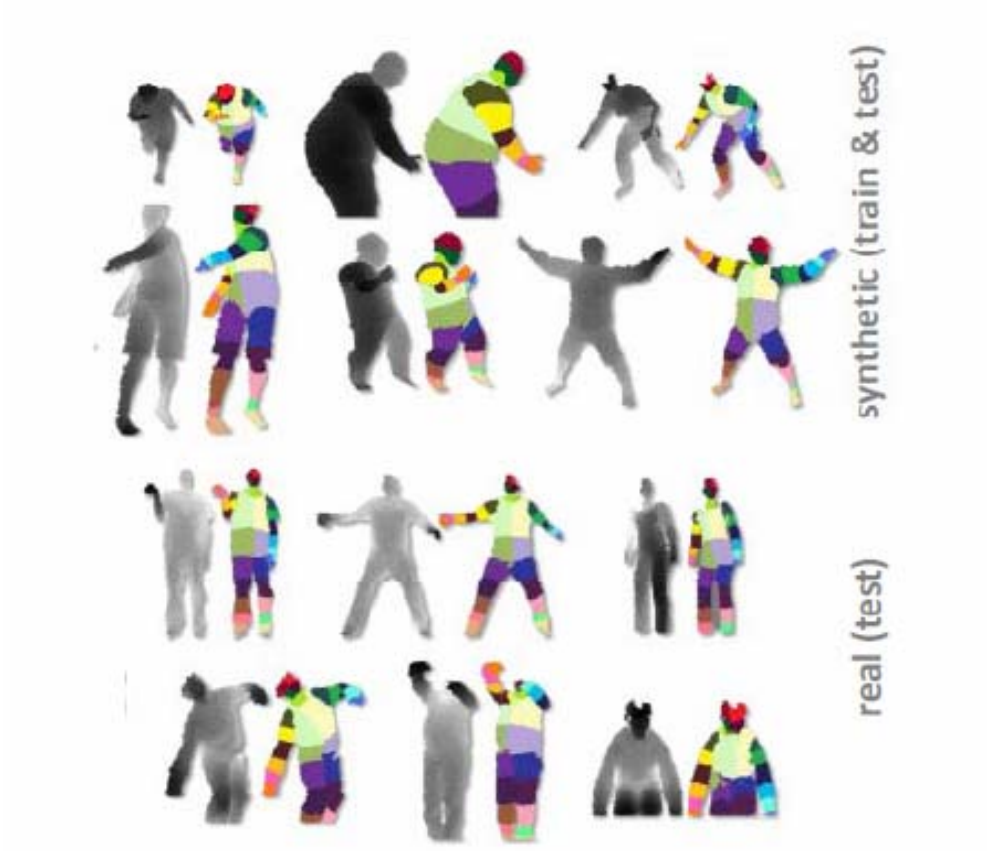
Goal: train a random forest classifier to predict body parts



Training/test Data

From motion capture system





e.g. 1 million training examples

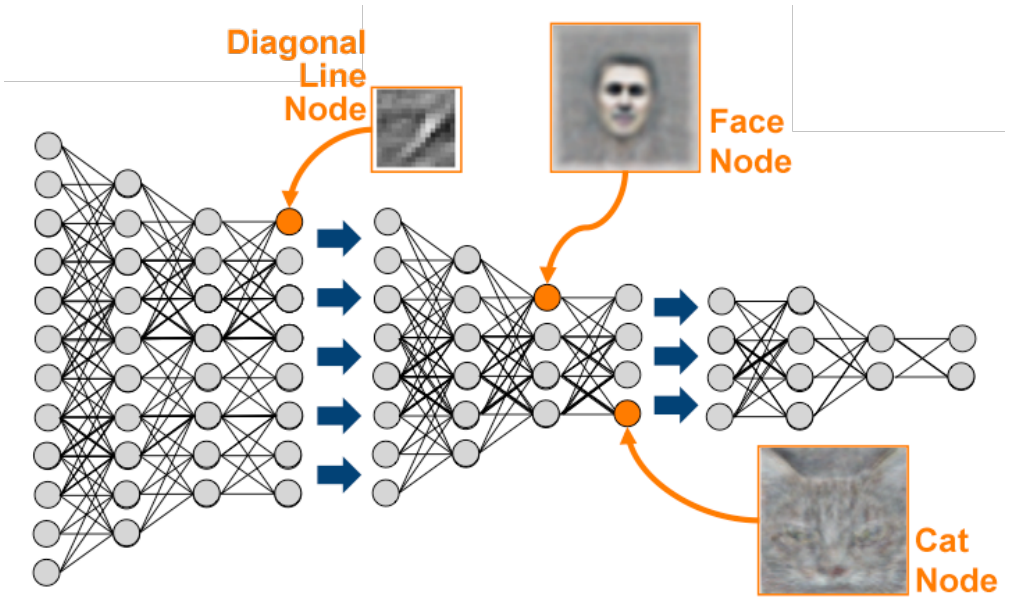
Example result

Input depth image (bg removed)

Inferred body parts (posterior)

body parts in 3D

Deep Learning (Neural Nets + marketing)



Network of weak learning layers, constructed to mimic the human perception biology

Good:
Outperforms on image recognition heading toward natural language domains

Challenge:
immensely expensive to train, no obvious hyperparameter optimization

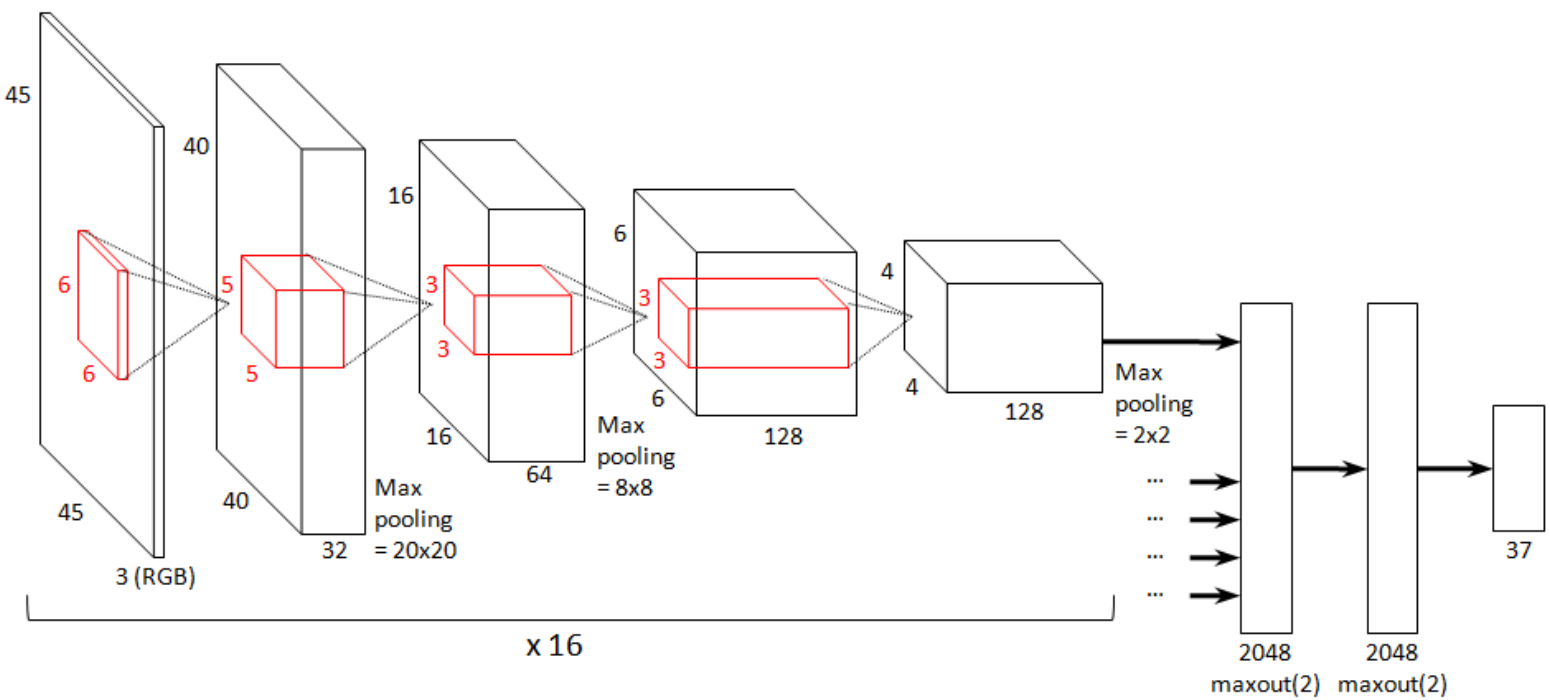
https://pythonhosted.org/nolearn/

My solution for the Galaxy Zoo challenge

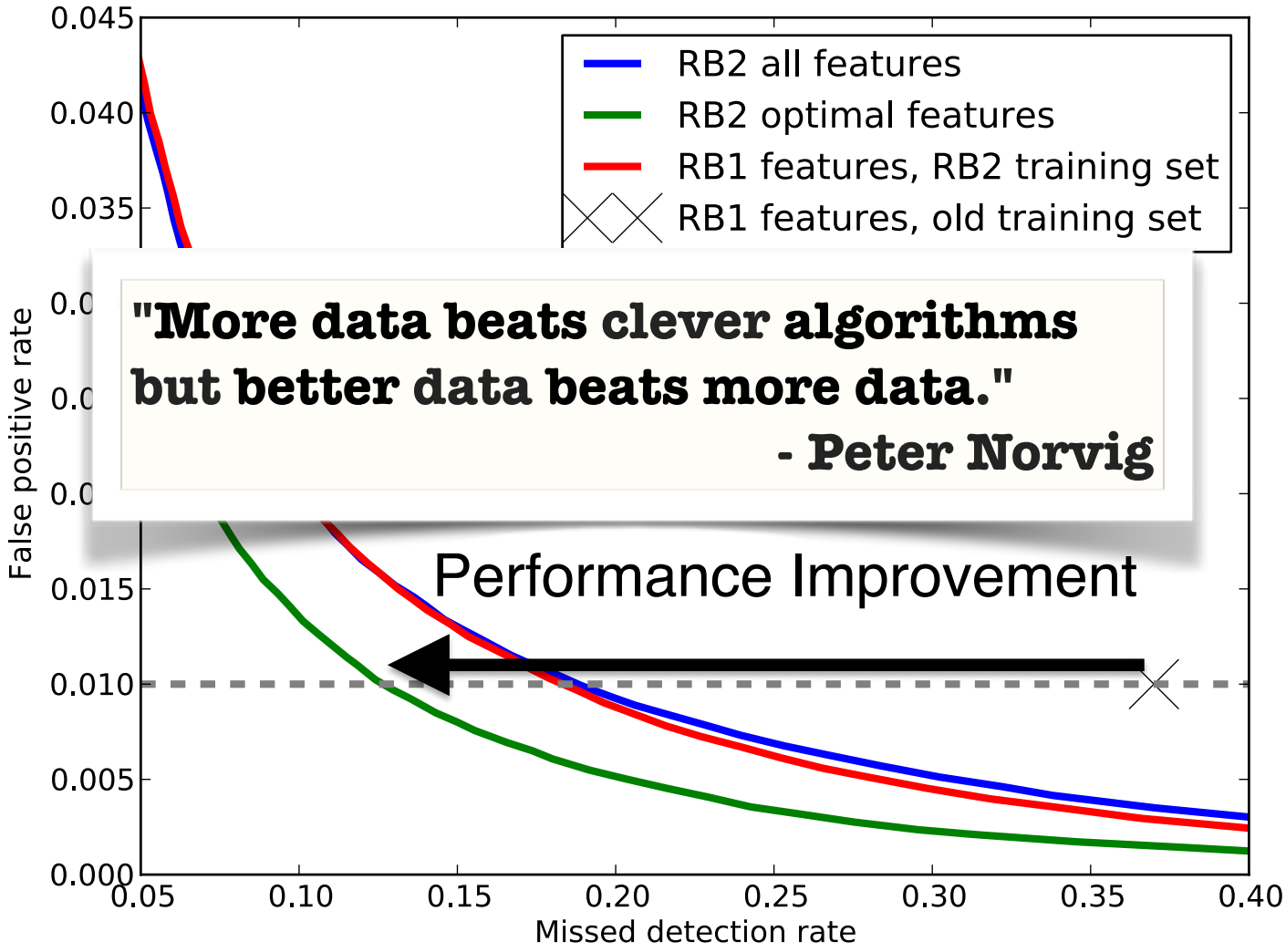
APRIL 05, 2014

http://benanne.github.io/2014/04/05/galaxy-zoo.html

"The goal of the competition was to predict how Galaxy Zoo users (zooites) would classify images of galaxies from the Sloan Digital Sky Survey. I finished in 1st place and in this post I'm going to explain how my solution works."



Classification Improvements



Brink+2012

4. Improving your models

- hyperparameter optimization
  `GridSearchCV`
- dealing with missing data
- Feature selection / feature importance
- feature engineering

Machine-Learning Approach to Classification

Engineered **features** homogenize data → $\mathbb{R}^p$

Describe time-domain characteristics & context of a source

$p \approx 100$ features computed in < 1 sec per 8-core machine (including periodogram analysis)

variability metrics:
e.g. Stetson indices, χ^2/do
(constant hypothesis)

periodic metrics:
e.g. dominant frequencies in Lomb-Scargle, phase offsets between periods

shape analysis
e.g. skewness, kurtosis, Gaussianity

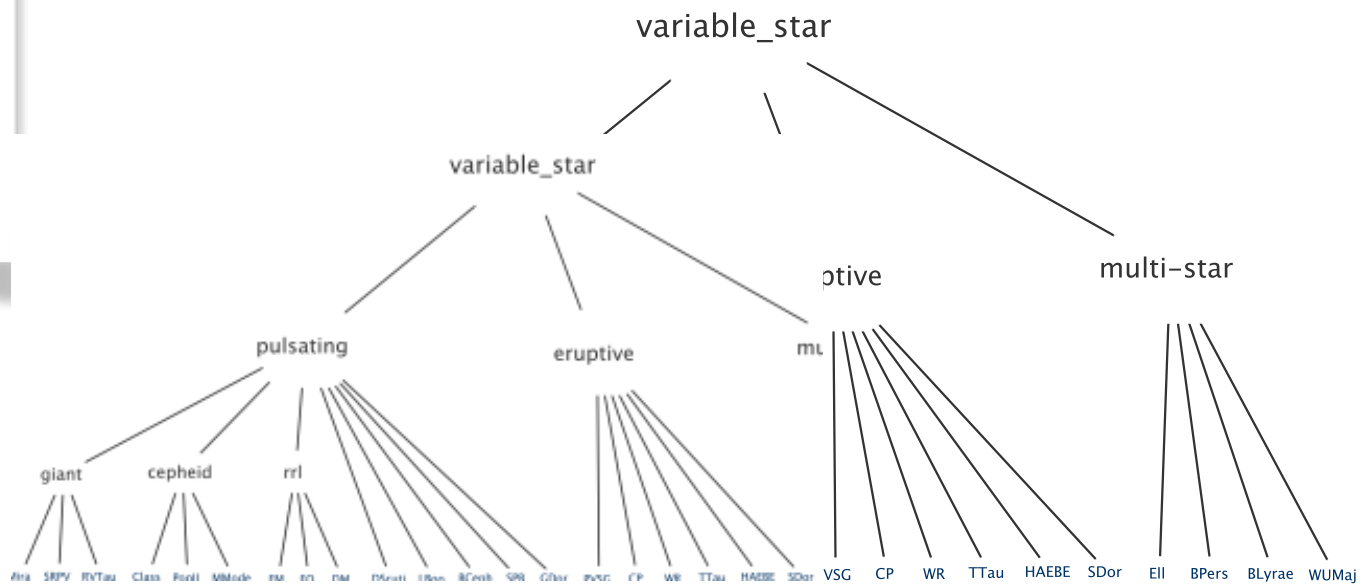
context metrics
e.g. distance to nearest galaxy, type of nearest galaxy, location in the ecliptic plane

Wózniak et al. 2004; Protopapas+06, Willemsen & Eyer 2007; Debosscher et al. 2007; Mahabal et al. 2008; Sarro et al. 2009; Blomme et al. 2010; Kim+11, Richards+11

Structured Learning

Structured Classification: Let class taxonomy guide clas

5% gross mis-classification rate!



HSC: Hierarchical single-label classification.

- Fit separate classifier at each non-terminal node.

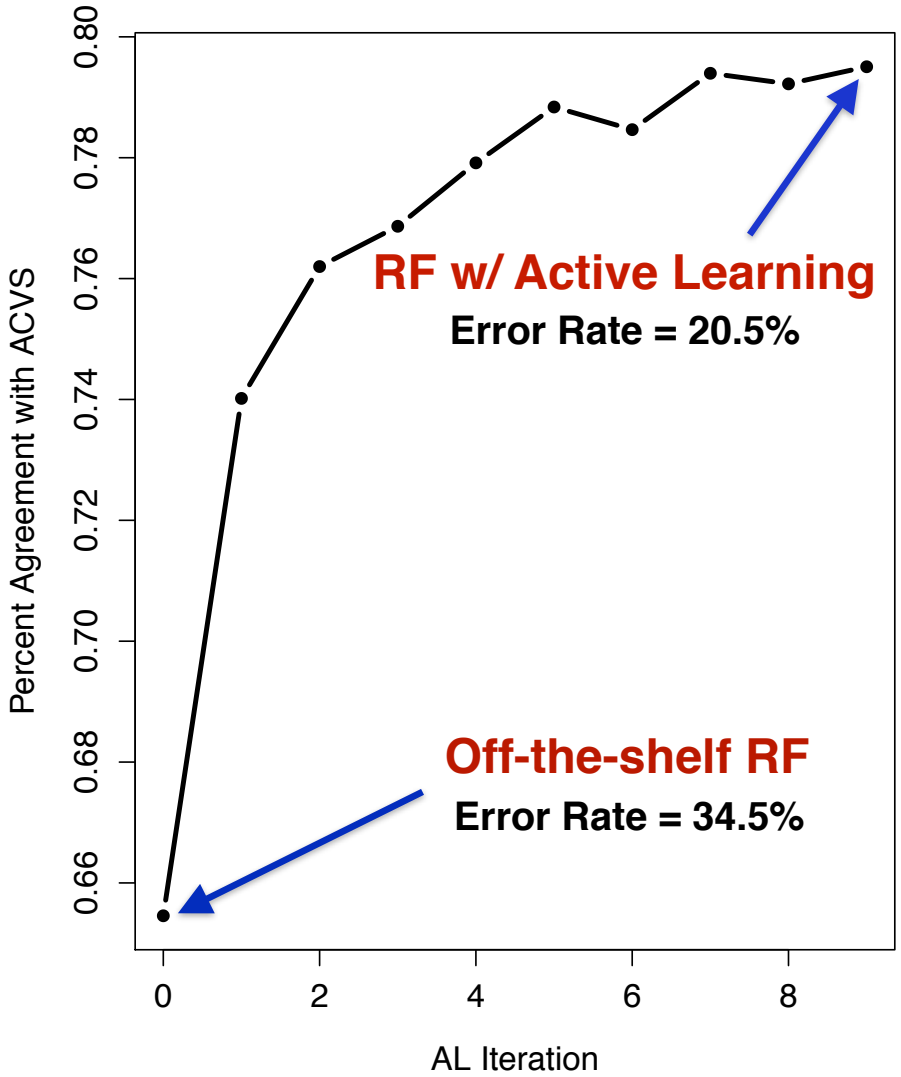
HMC: Hierarchical mu classification.

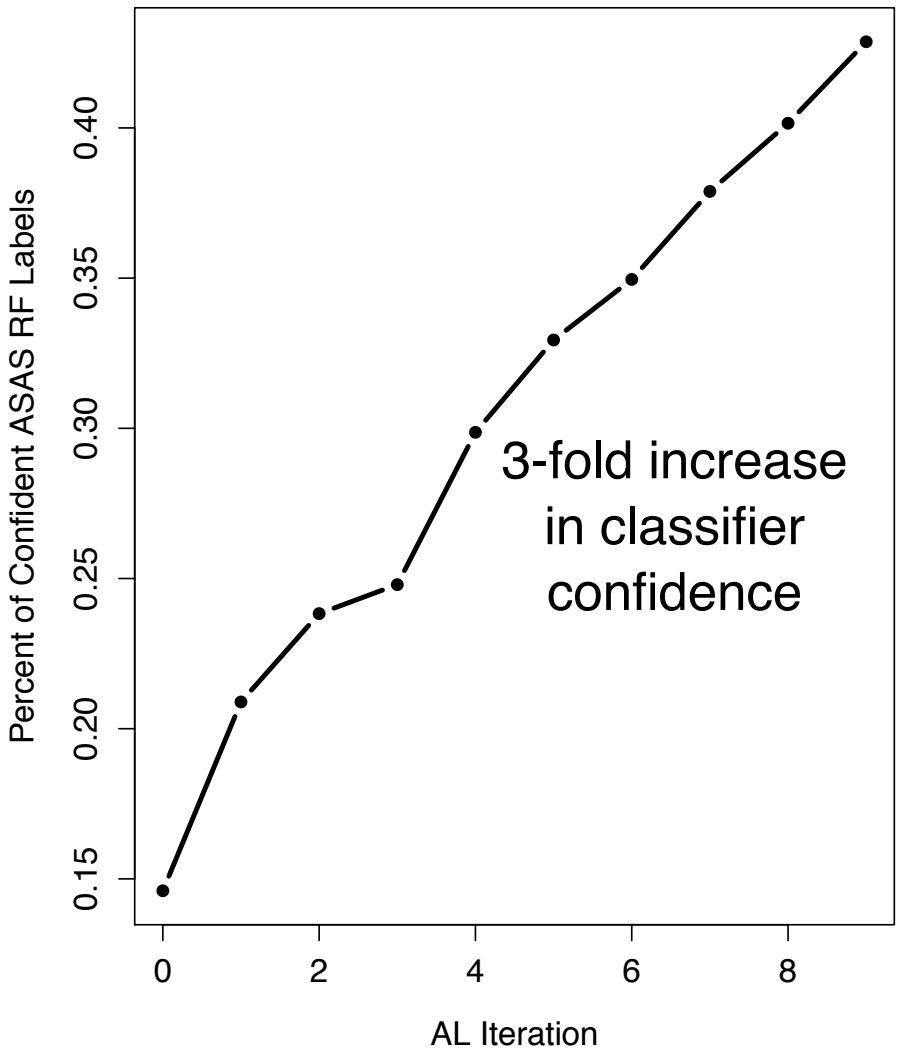
- Fit one classifier, $L(y, \widehat{f}(\mathbf{x})) \propto w_0^{\text{de}}$

Richards+11

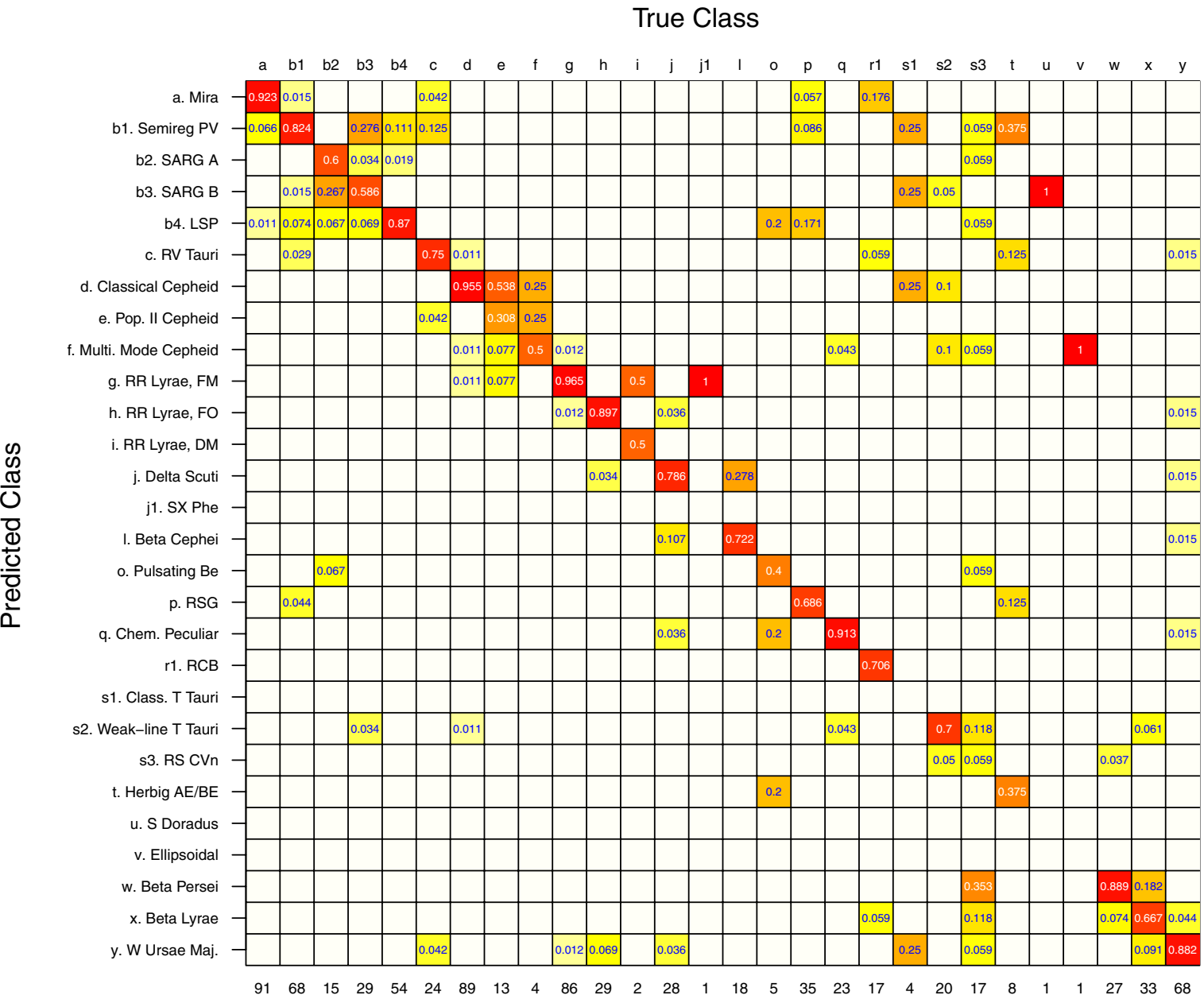
Active Learning

28-class variable star classification problem with 50,000 stars





Machine Learned Classification



74 dimensional feature set for learning

featurization is the bottleneck (but embarrassingly parallel)

Richards+12

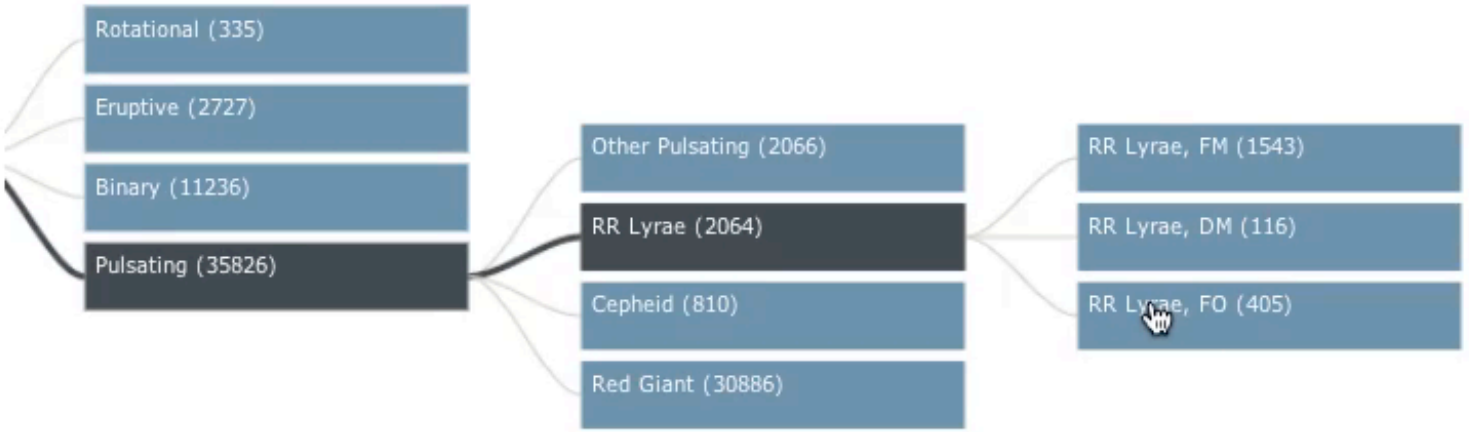
SELECT CLASS:

RR Lyrae

Specify RA/Dec

Customize Table

Export Query Results



Displaying sources 1 to 500

Previous 500 Next 500

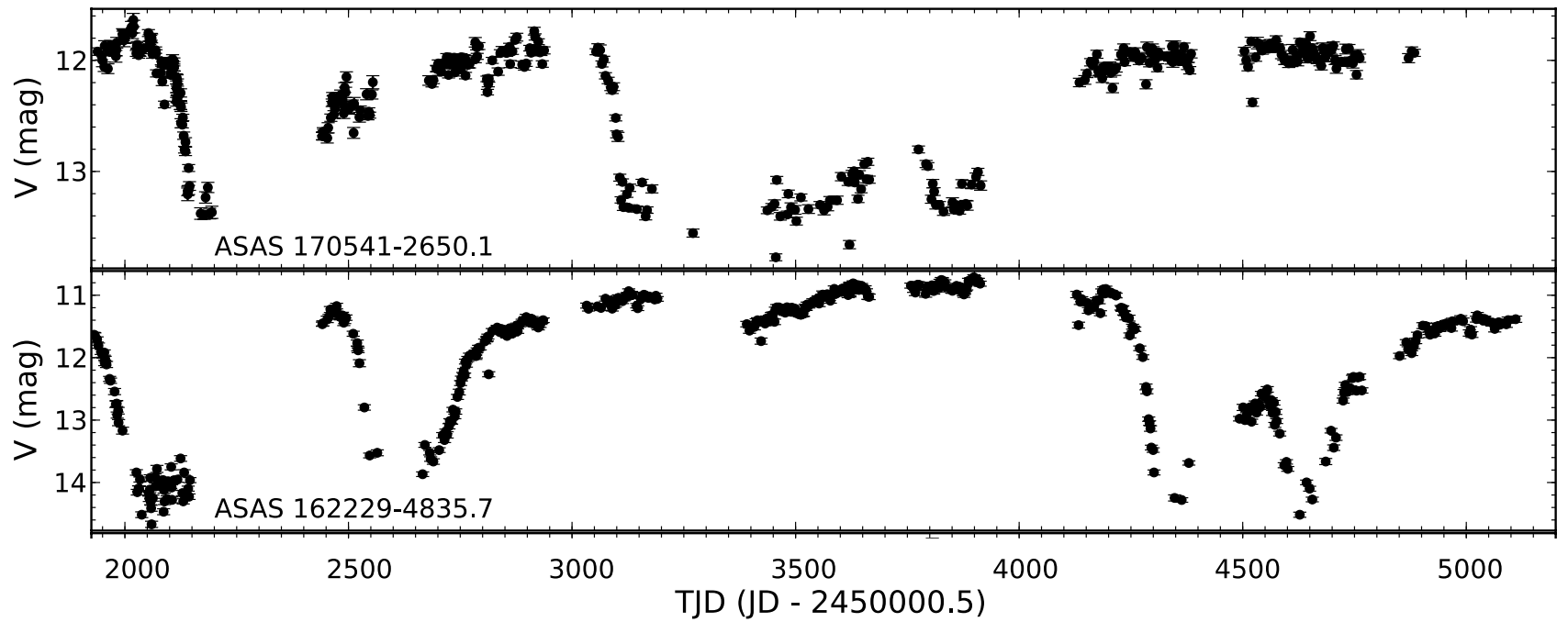
ASAS_ID	dotAstro_ID	RA	DEC	Class	P_Class	Anomaly	ACVS_Class	Train_Class	P	P_signif	N_epochs	V	d
012842-5356.2	216035	22.18476	-53.940074	RR_Lyrae_DM		14.625	ESD/EC		0.512	10.988	515	10.95	
020419+1318.9	216342	31.079265	13.316235	RR_Lyrae_DM		3.000	MISC		0.870	4.485	1640	12.01	
024721+2505.4	216754	41.83305	25.091205	RR_Lyrae_DM		6.143	MISC		0.385	6.719	350	13.32	
030528-3058.7	216961	46.369725	-30.974181	RR_Lyrae_DM		1.370	RRC		0.439	13.671	682	12.69	
030600+2308.3	216967	46.500615	23.138132	RR_Lyrae_DM		6.246	MISC		0.330	6.200	540	12.34	
032439-0209.5	217178	51.1608	-2.15906	RR_Lyrae_DM		2.817	EC		0.826	15.553	356	12.31	
032820-6458.7	217226	52.077465	-64.978341	RR_Lyrae_DM		2.185	RRC		0.355	22.242	489	12.63	
033244+2523.2	217281	53.18397	25.386141	RR_Lyrae_DM		3.202	EC/DSCT		0.177	11.107	1370	12.12	
040054-4923.8	217649	60.216645	-49.400343	RR_Lyrae_DM		1.874	RRAB:v		0.417	9.201	693	13.77	
042745-5812.0	218045	66.92889	-58.195033	RR_Lyrae_DM		10.628	EC		0.455	18.977	308	12.57	
043659-2244.2	218187	69.2505	-22.736578	RR_Lyrae_DM		2.165	RRAB/DCEP-FO/EC		0.365	11.578	660	13.68	
051446-0041.5	219041	78.688065	-0.691387	RR_Lyrae_DM		9.417	MISC		21.390	4.857	615	10.7	
061035+2338.9	220981	92.6		RR_Lyrae_DM		7.621	MISC			11.559	408	11.72	
061821+1045.9	221359	94		RR_Lyrae_DM		5.579	MISC			11.162	690	9.98	
063414+2412.5	222147	98		RR_Lyrae_DM		7.621	MISC			8.673	630	11.25	
064131+1036.7	222551	100		RR_Lyrae_DM		6.353	ESD/RRC/EC			6.352	610	13.16	

Machine-learned Variable Star catalog: **http://bigmacc.info**

Discovery of Bright Galactic R Coronae Borealis and DY Persei Variables: Rare Gems Mined from ASAS

A. A. Miller[1,*], J. W. Richards[1,2], J. S. Bloom[1], S. B. Cenko[1], J. M. Silverman[1], D. L. Starr[1], and K. G. Stassun[3,4]

arXiv.org > astro-ph > arXiv:1204.4181



17 known Galactic RCB/DY Per

5. Considerations in getting into production

- scikit-learn pipelines
- multicore / multimachine
- Bigdata machine learning: Graphlab, MLlib (Spark)

Pipelining in Sklearn

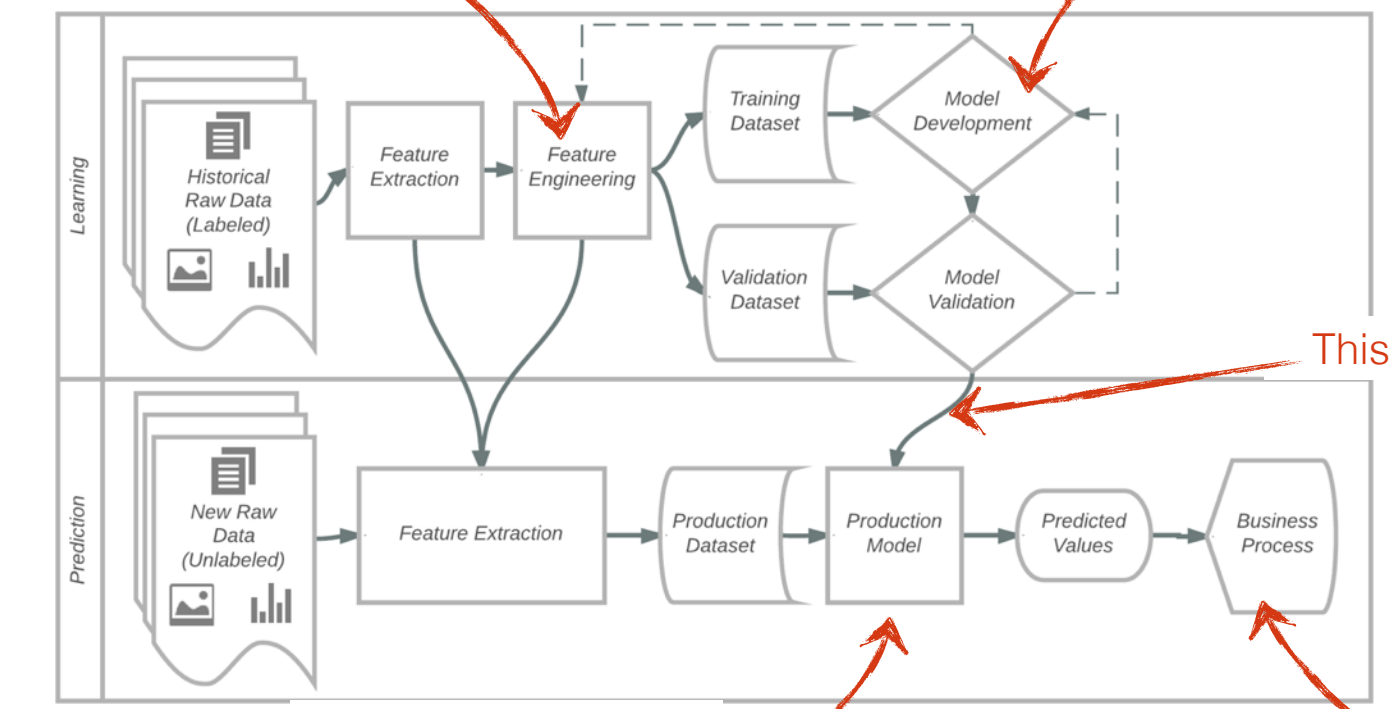
```
pipeline = Pipeline([
    ('vect', CountVectorizer()),
    ('tfidf', TfidfTransformer()),
    ('clf', SGDClassifier()),
])
```

```
parameters = {
    'vect__max_df': (0.5, 0.75, 1.0),
   'vect__ngram_range': ((1, 1), (1, 2)),  # unigrams or bigrams
    'clf__alpha': (0.00001, 0.000001),
    'clf__penalty': ('l2', 'elasticnet'),
}
```

```
grid_search = GridSearchCV(pipeline, parameters, verbose=1)
```

http://scikit-learn.org/stable/modules/pipeline.html

Data-driven Workflow with ML



The most time consuming part…

Most academic CS work & ML companies selling toolkits only focus here…

This is broken…

Hard to understand for non-experts...

Actionable insight What really matters…

cf. Ben Lorica's "Data Science Workflow"

Multicore *Usually RAM is the bottleneck*

- sklearn can make use of multiple cores
makes most sense when doing cross-validation
or when doing grid search optimization

`njobs=-1`

- Streaming learning: out-of-core learning / incremental learning

Multimachine *Usually I/O is the bottleneck*

- IPython cluster http://nbviewer.ipython.org/github/ogrisel/parallel_ml_tutorial/tree/master/

Some Tutorials

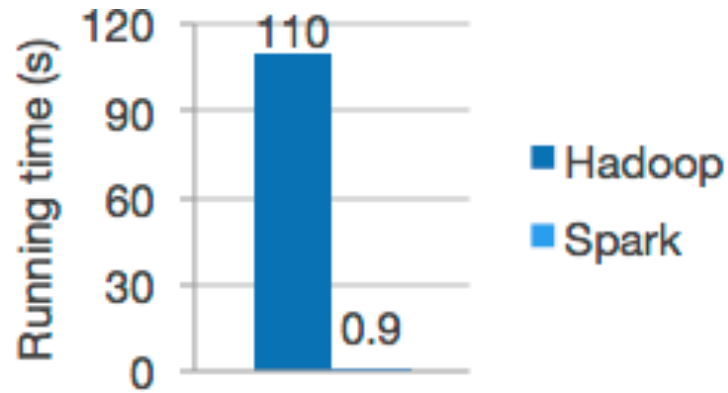
http://scikit-learn.org/stable/modules/scaling_strategies.html

https://www.youtube.com/watch?v=iFkRt3BCctg

If you have REALLY Big Data……

Spark MLlib

MLlib is Apache Spark's scalable machine learning library.



Dato
Create Intelligence™

```
pip install graphlab-create==1.2.1
```

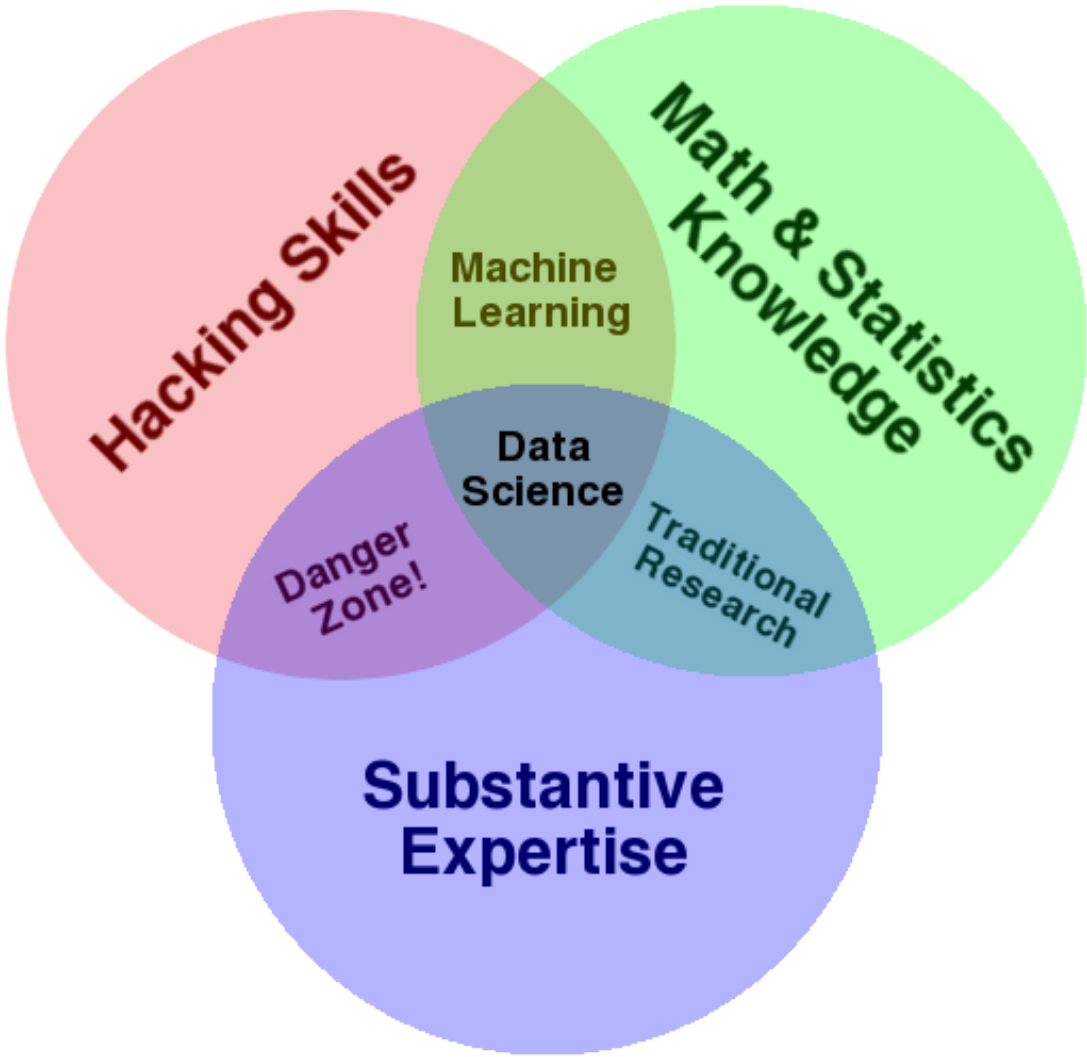
https://www.youtube.com/watch?v=4TjxMCdA8po

http://graphlab.com/learn/gallery/notebooks/intro-regression.html

sklearn is missing a few things...

- Kernel smoothing / Loess
- Multivariate Adaptive Regression Splines (MARS)
- Neural Networks / Self-Organizing Maps
- Kalman Filtering / Particle Filtering
- Missing data imputation / surrogate splting
- Bayesian model fitting / non-parametrics

Machine Learning



http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram