

CS109 – Data Science

Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau

vkaynig@seas.harvard.edu

staff@cs109.org

Announcements

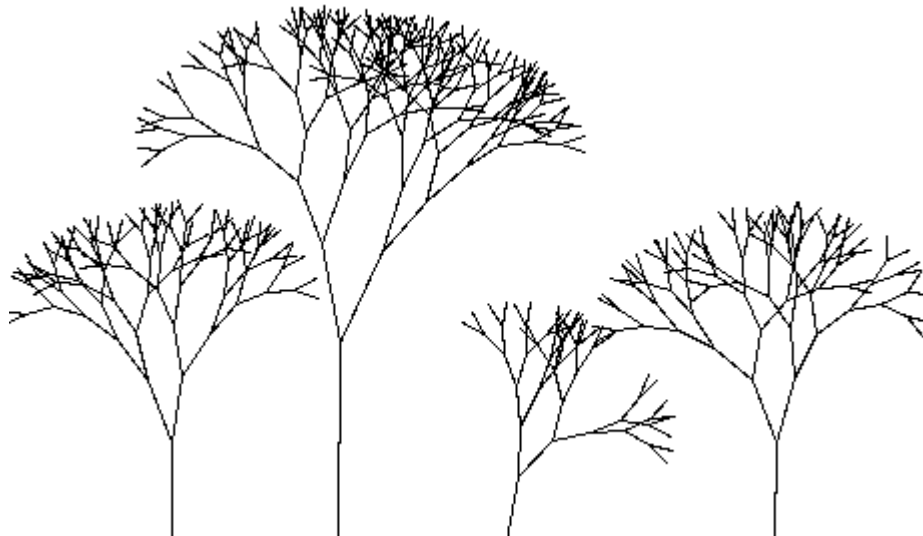
- Story telling is important for data scientists
- Please make sure to target your audience as directed in the homework

Next Topics

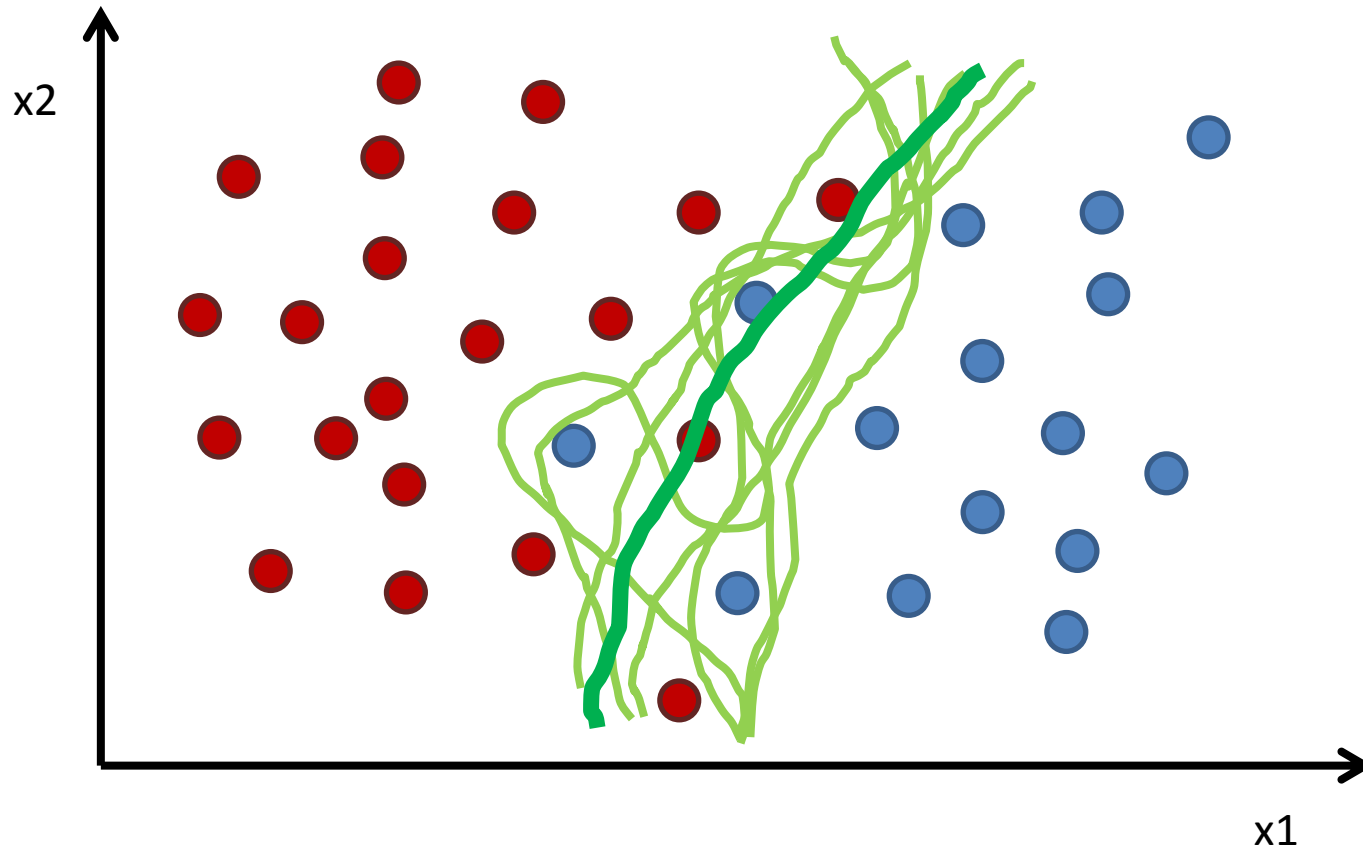
- Classifier wrapup:
 - Some RF things
 - Regression
- ML best practices
 - model choice
 - imbalanced data
 - missing values
- Recommender systems
 - collaborative filtering
 - content-based filtering

Random Forest

- Builds upon the idea of bagging
- Each tree build from bootstrap sample
- Node splits calculated from **random feature subsets**



Bagging Idea



Random Forest

- All trees are fully grown
- No pruning
- Two parameters
 - Number of trees
 - Number of features

What is the difference between Bagging and Random forest?

Random Forest Error Rate

- Error depends on:
 - Correlation between trees (higher is worse)
 - Strength of single trees (higher is better)
- Increasing number of features for each split:
 - Increases correlation
 - Increases strength of single trees

Extreme Scenario

y	x			
1	1	@	&	...
0	0	#	%	...
1	1	\$	#	...
1	1	%	^	...
0	0	^	!	...
1	1	*)	...
0	0)	%	...
...

only have 1 split because the first feature allows it to classify exactly

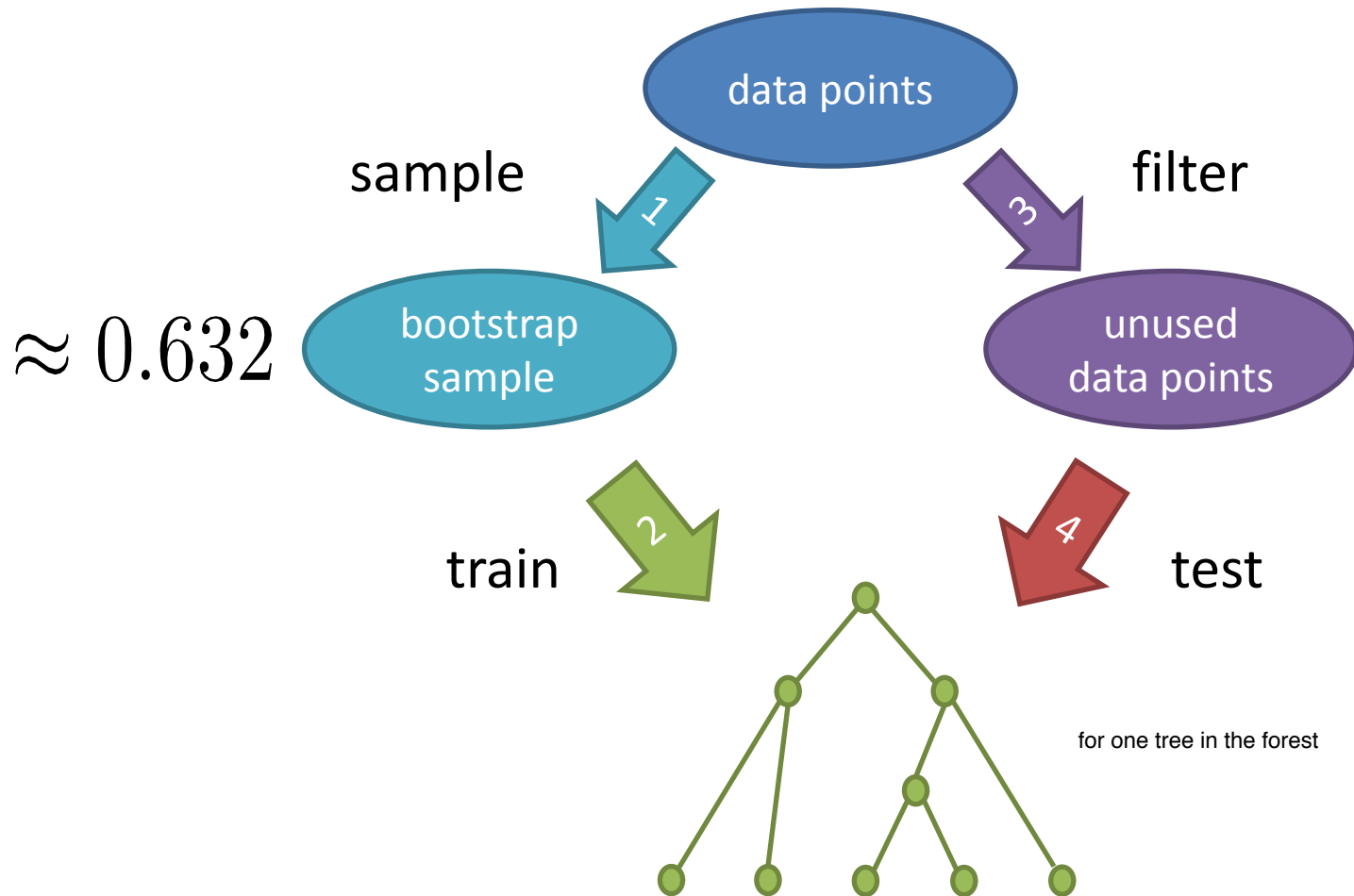
What would a single fully grown tree learn? How deep would it be?

What would Bagging learn? How would it differ from the single tree?

What would a Random Forest learn with `max_feature=1`?

may never get to see the first column, will fit to the noise most of the time. each single tree in the forest will be bad

Out of Bag Error



Out of Bag Error

- Very similar to cross-validation
- Measured during training
- Can be too optimistic

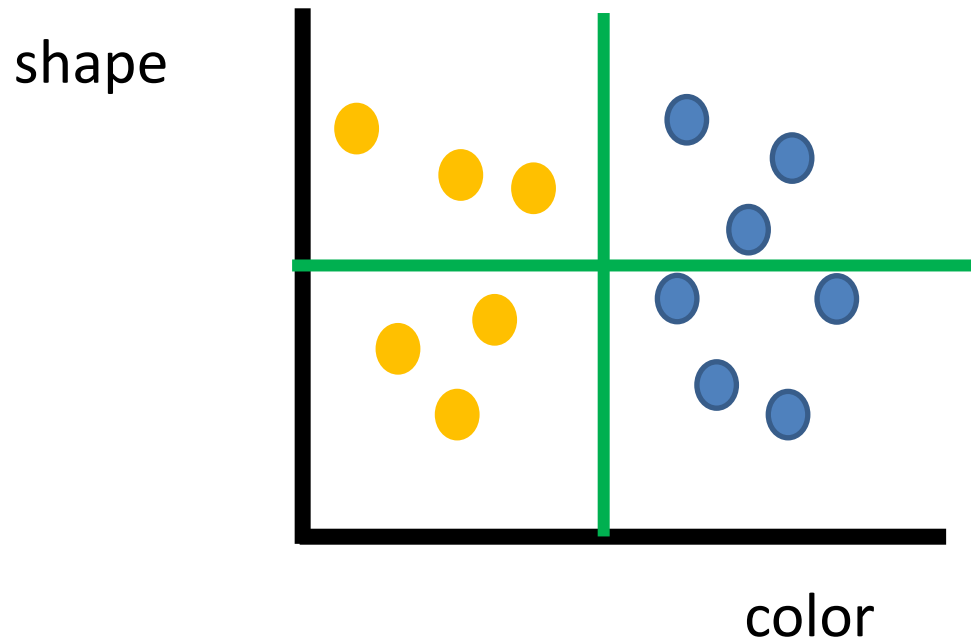
From a Kaggle Forum

... it feels weird to be using cross-validation type methods with random forests since they are already an ensemble method using random samples with a lot of repetition. Using cross-validation on random forests feels redundant.

Variable Importance - 1

- Again use out of bag samples
- Predict class for these samples
- Randomly permute values of one feature
- if destroying that feature decreases the accuracy a lot, then that variable is important
Predict classes again
- Measure decrease in accuracy

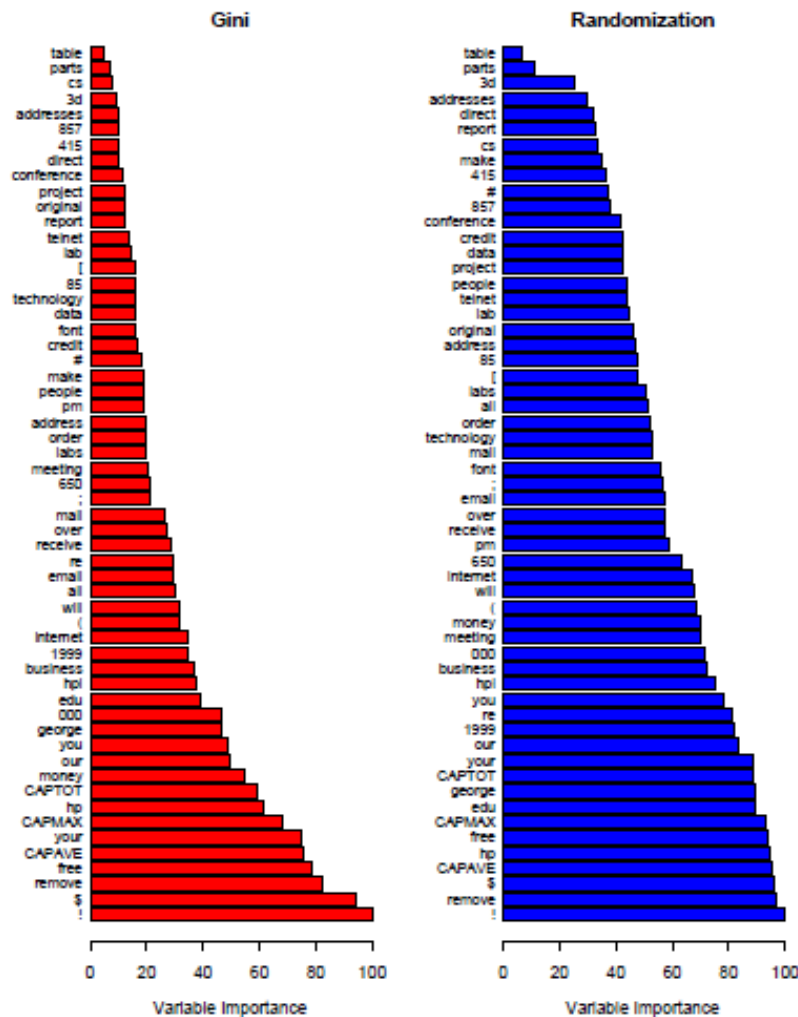
Variable Importance - 1



Variable Importance - 2

- Measure split criterion improvement
- Record improvements for each feature
- Accumulate over whole ensemble

Example: Spam classification



Randomization tends to spread out the variable importance more uniformly.

Overall both methods mainly agree on what is important.

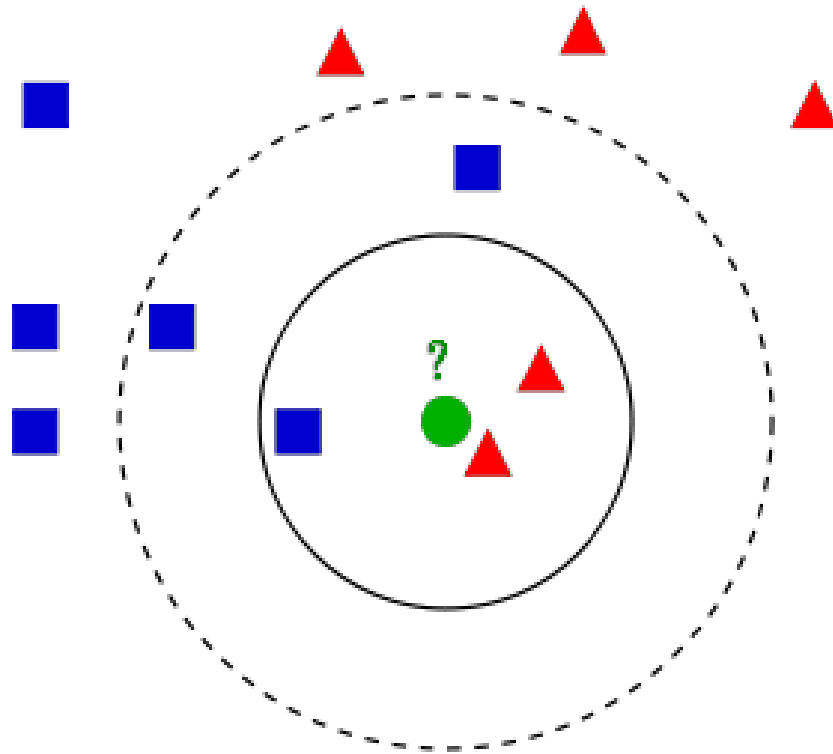
Regression

- What is the difference between regression and classification?

in classification, there is no notion of distance. a wrong classification is a wrong classification. eg. 2 stars is closer to 1 star than 5 stars. but if we misclassify it, then it is wrong without weighting of the error. regression preserves the distance

- Think about handling 5 star ratings as classification or as regression problem.

KNN Recap

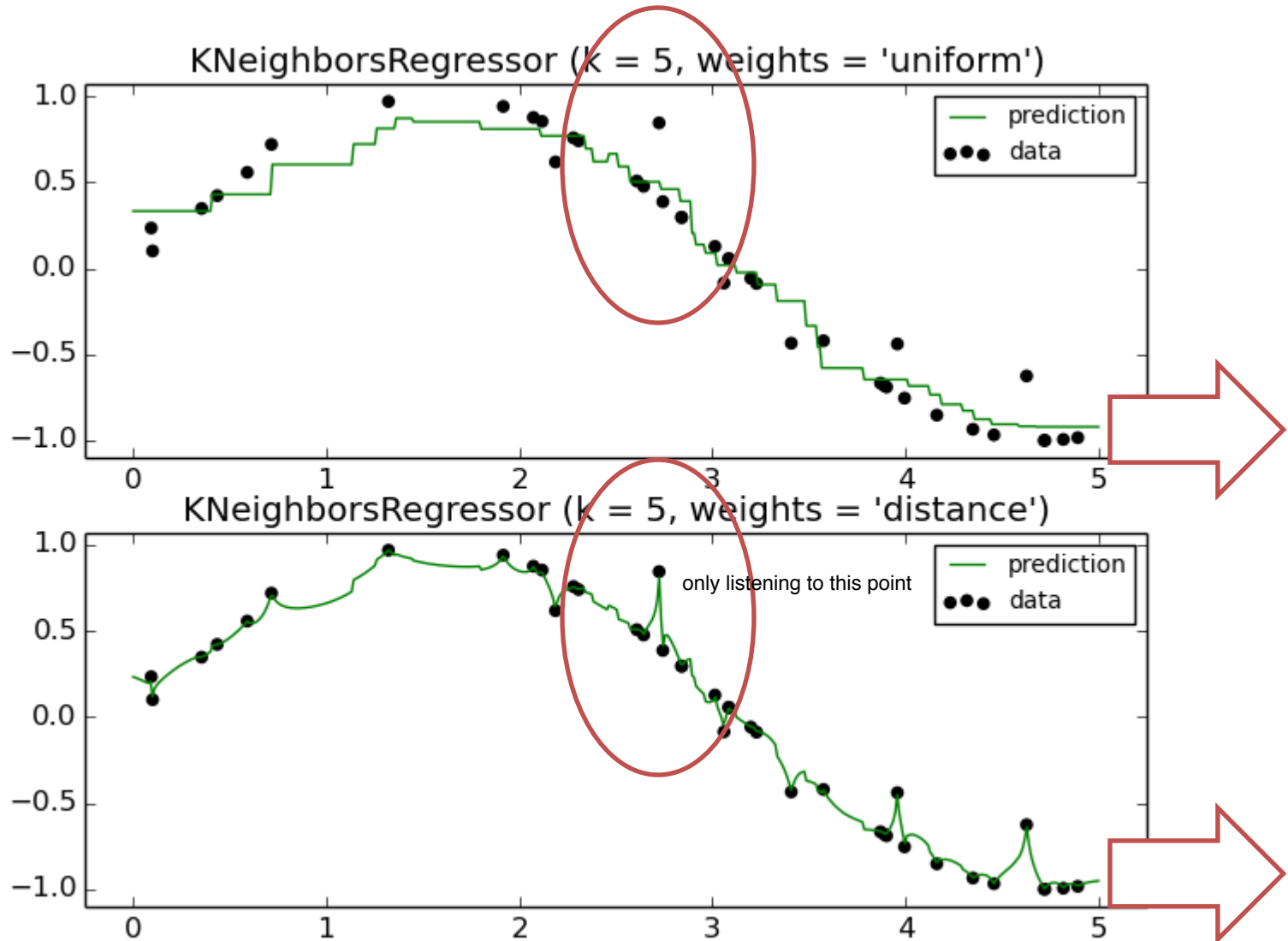


How would you modify KNN to do regression?

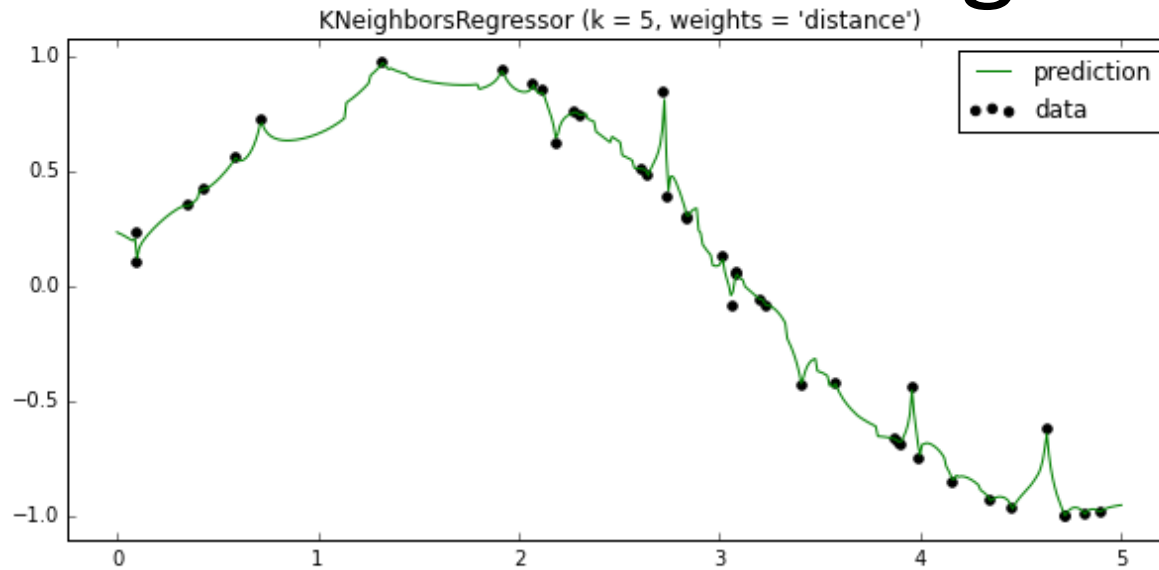
KNN for Regression

- Average the values of the K nearest neighbors
- Or build a weighted average of the K nearest neighbors

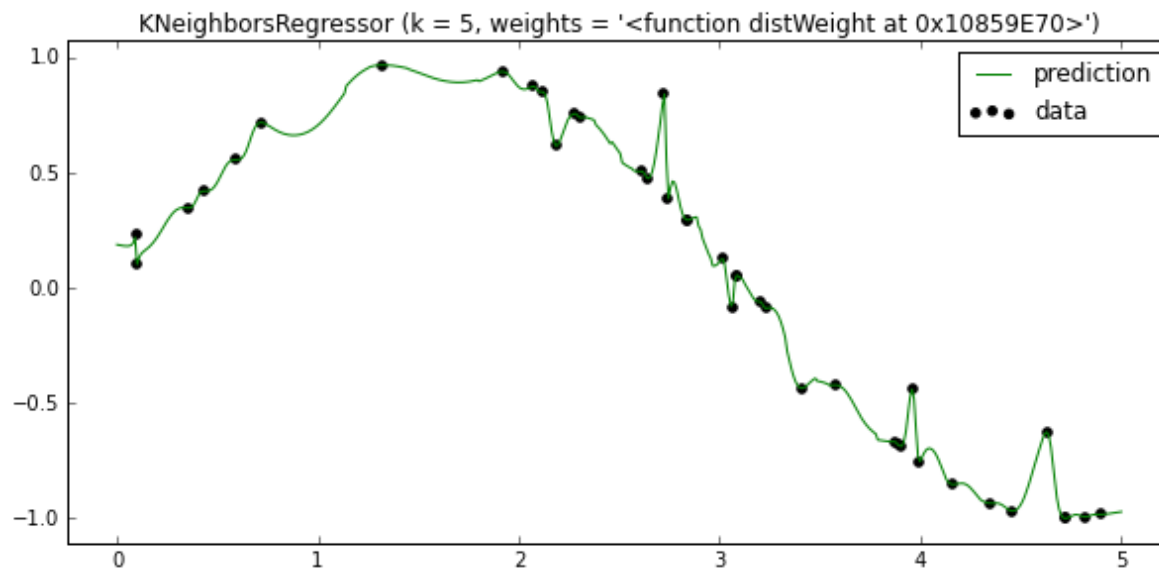
KNN Example



Distance Weighting

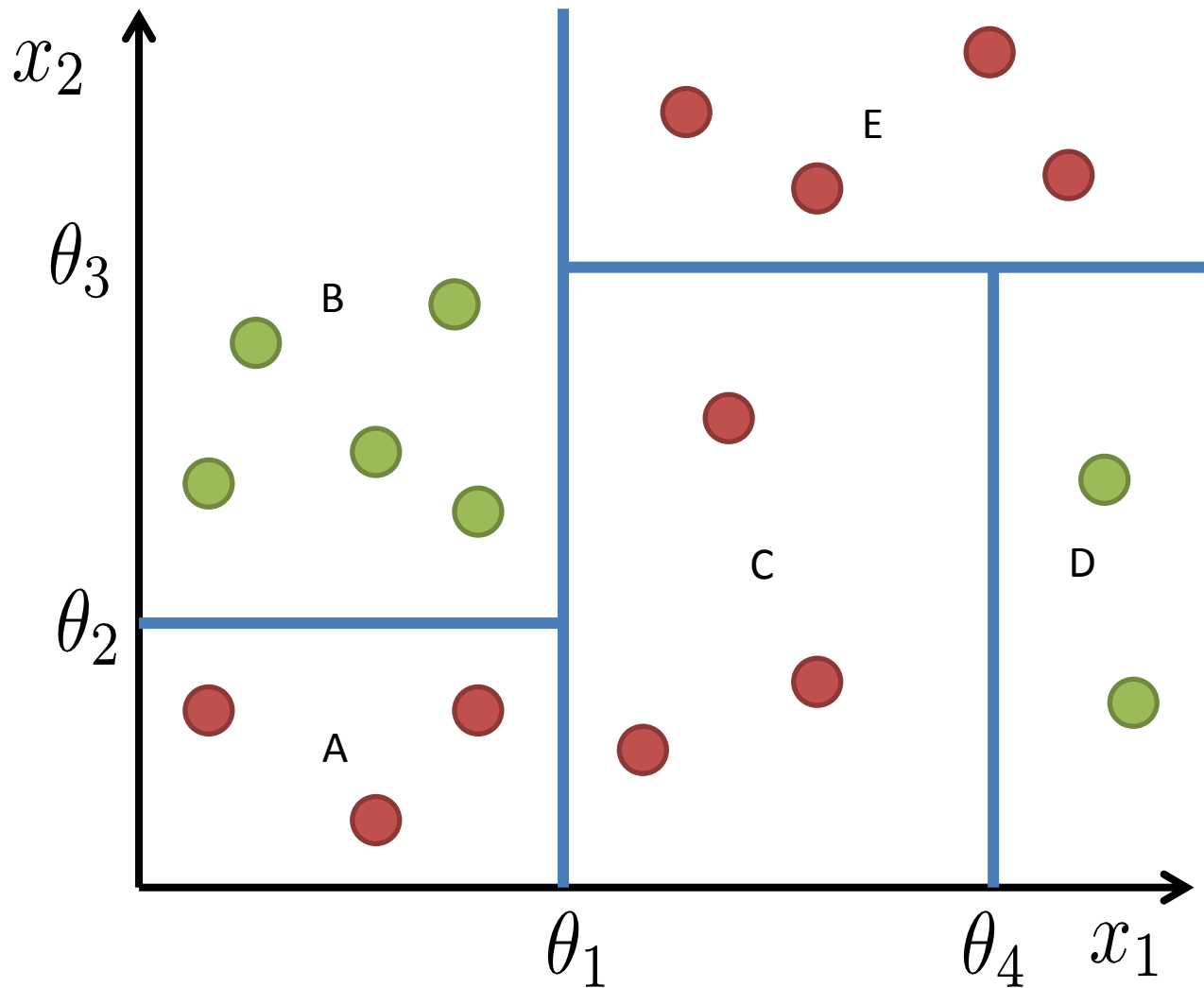


Linear weights

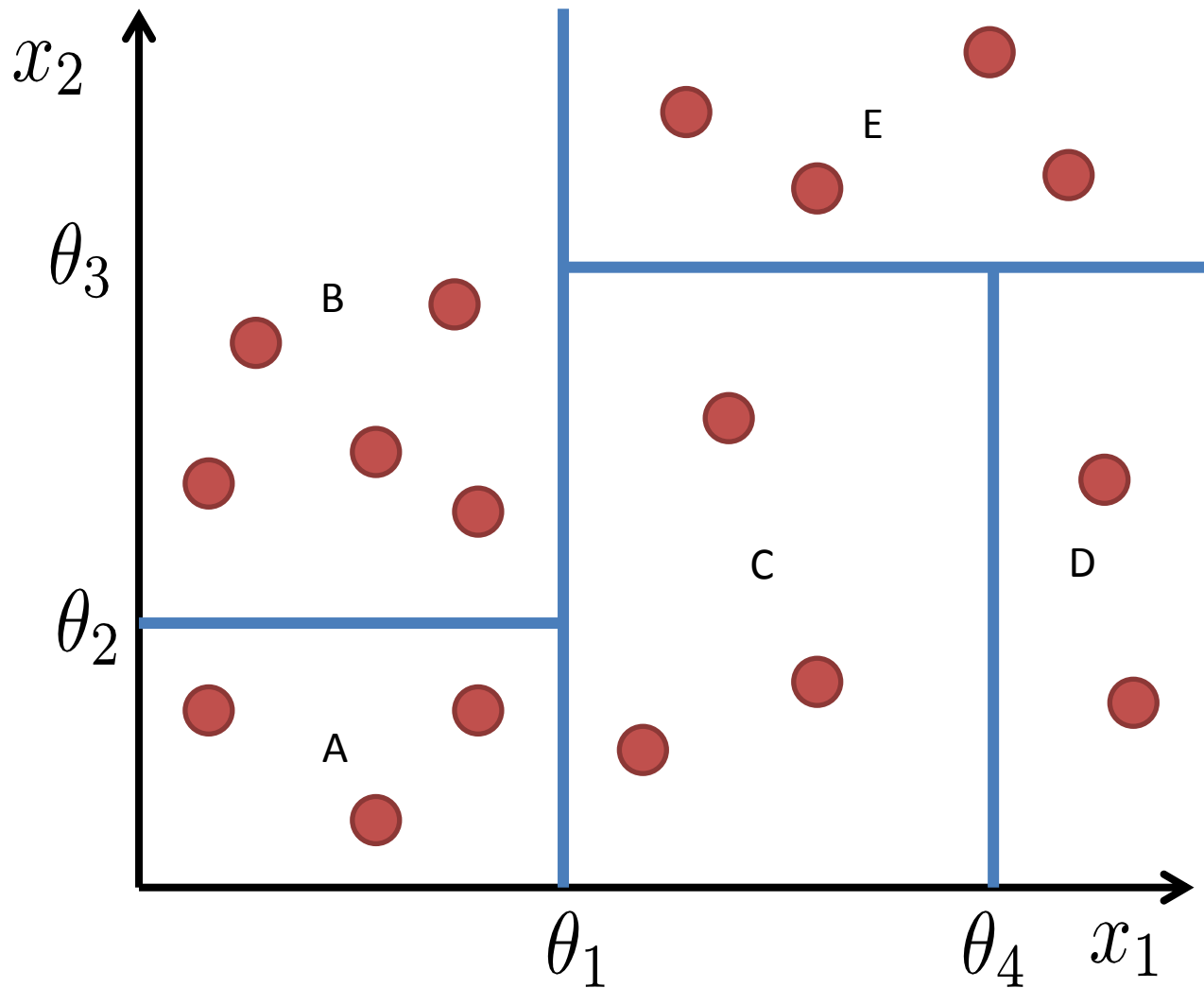


Quadratic weights

Decision Tree



Regression Tree

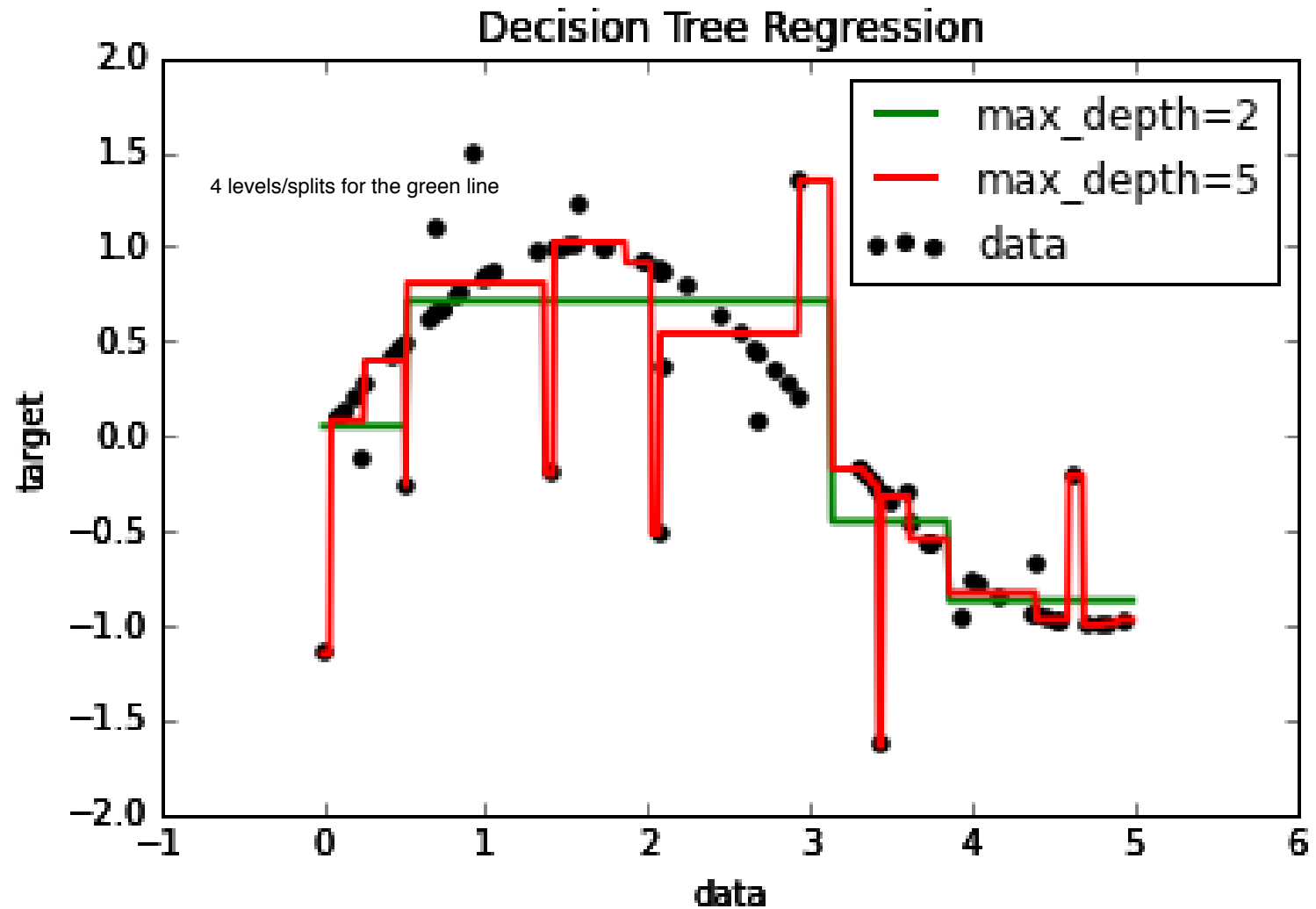


Regression Tree

- Again we average, this time over all points in one of the cells.
- During training, split in the way that reduces the squared error the most.

not using Gini

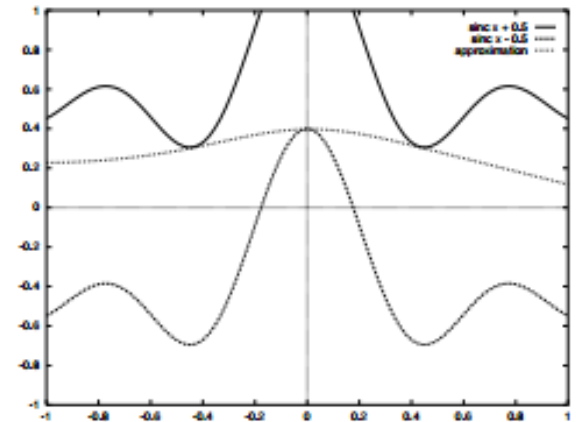
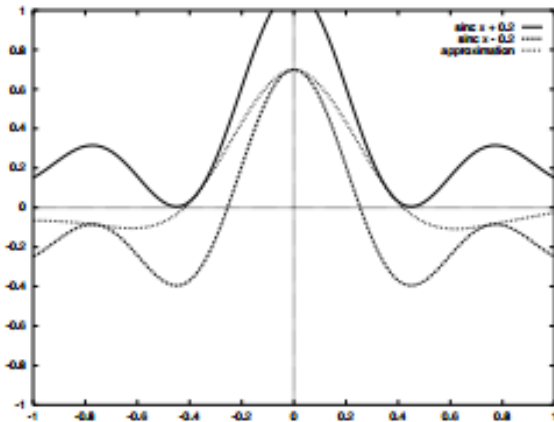
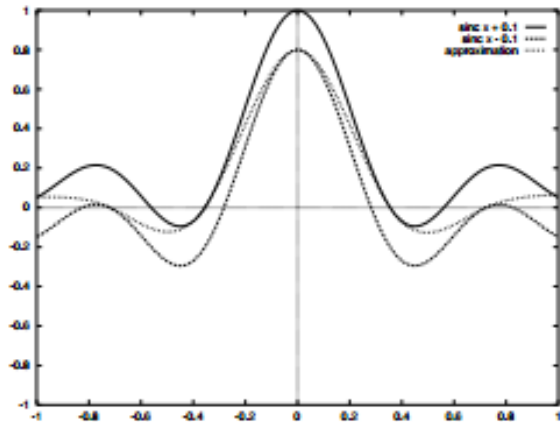
Regression Tree Training



Random Forest for Regression

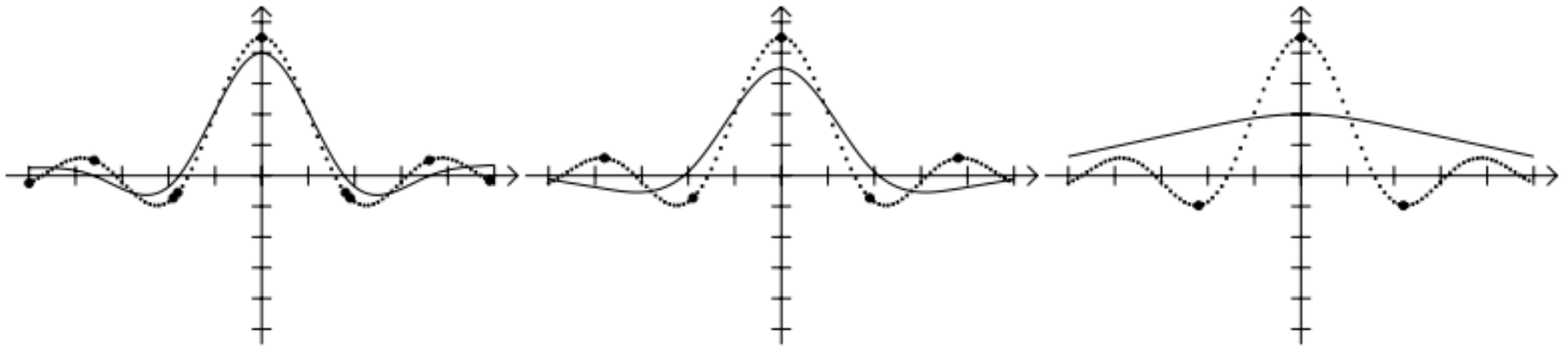
- Same idea as before
- Train multiple trees in parallel and average
- Different defaults
- `max_features = n_features`
- square error

SVM for Regression

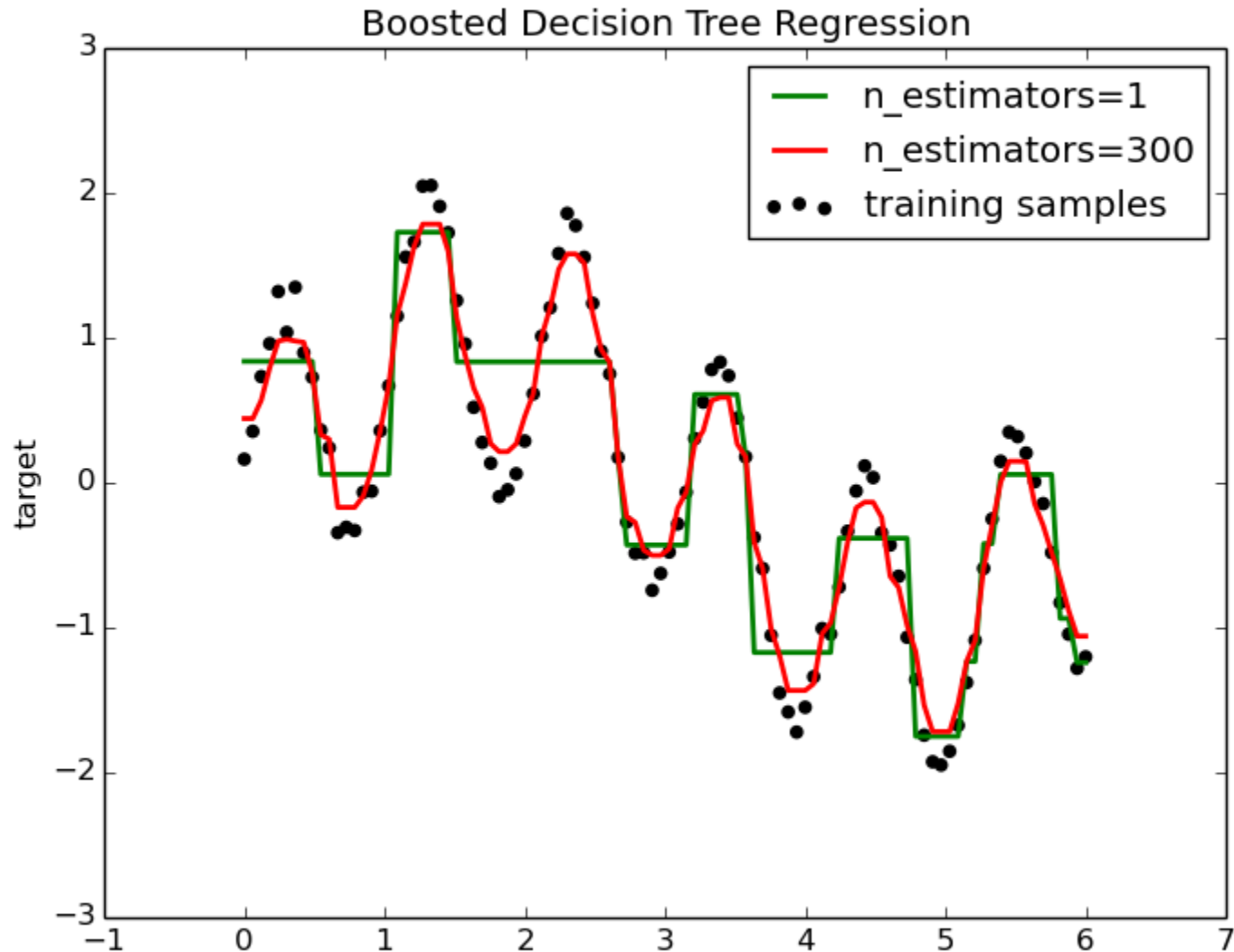


larger band for larger slack

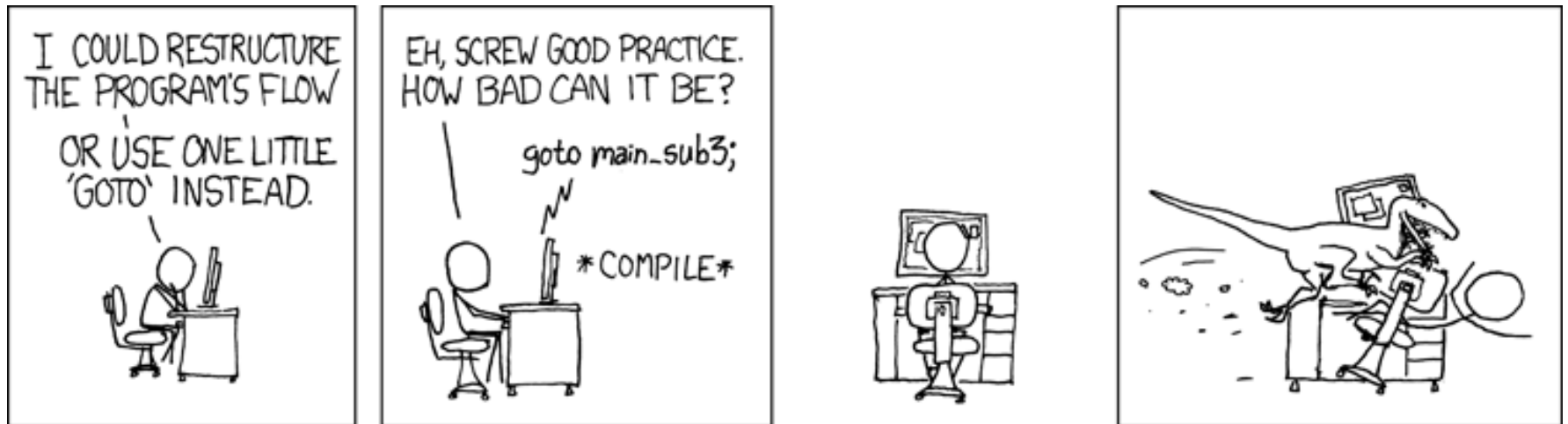
less support vectors (dots) for larger slack. get a smoother regression curve (solid black line)



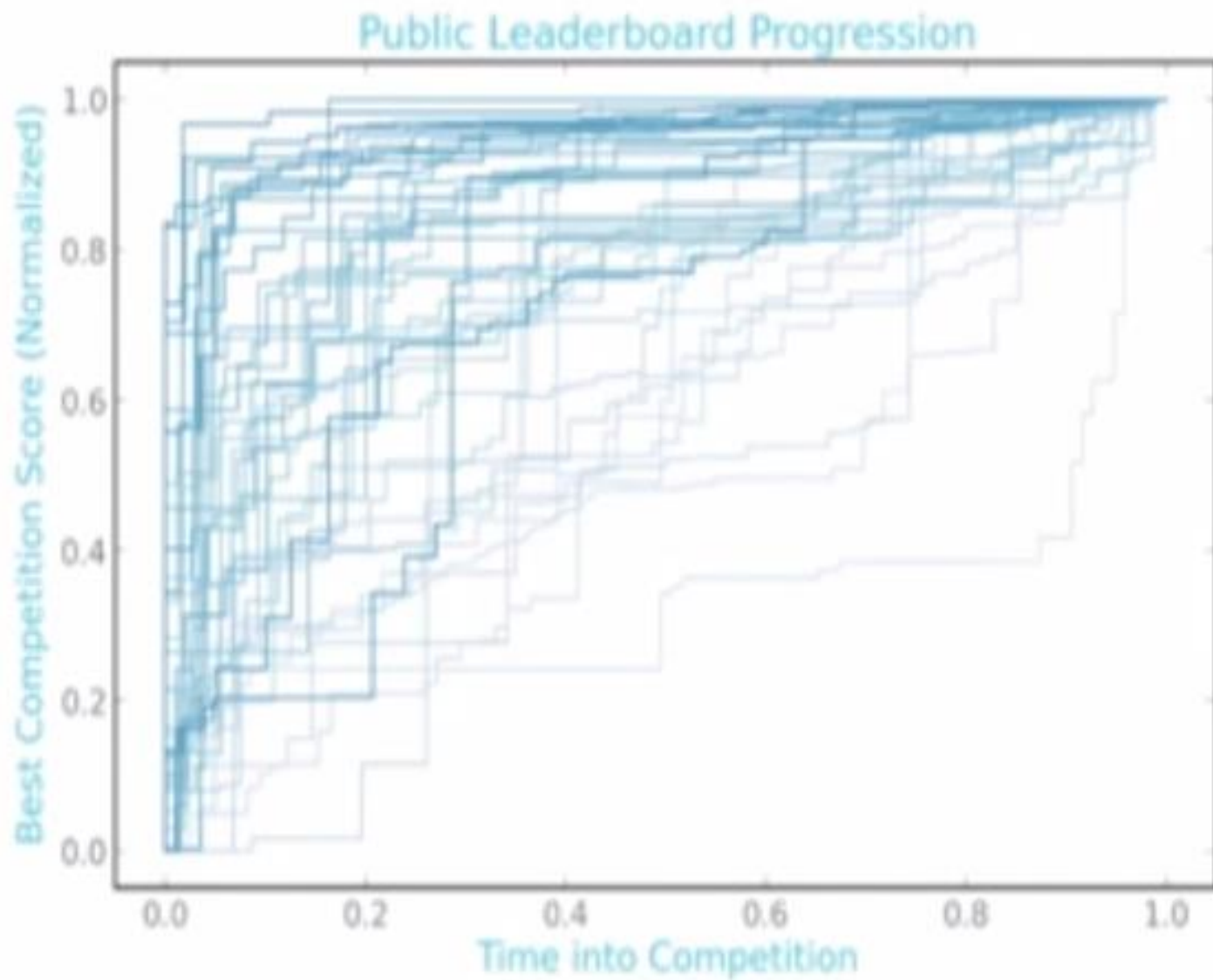
Boosting



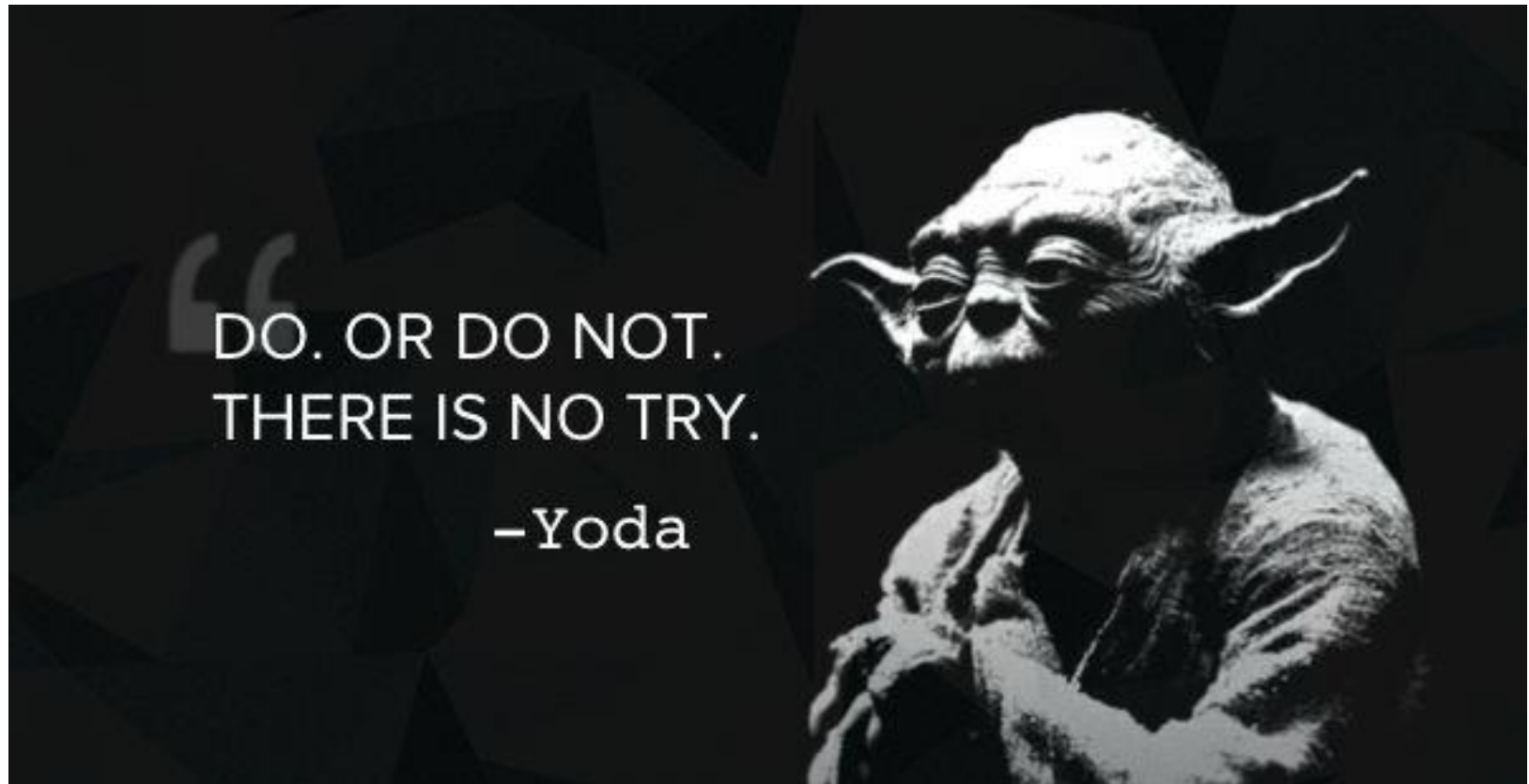
Best Practices



Typical Progress



Under Promise, Over Deliver!

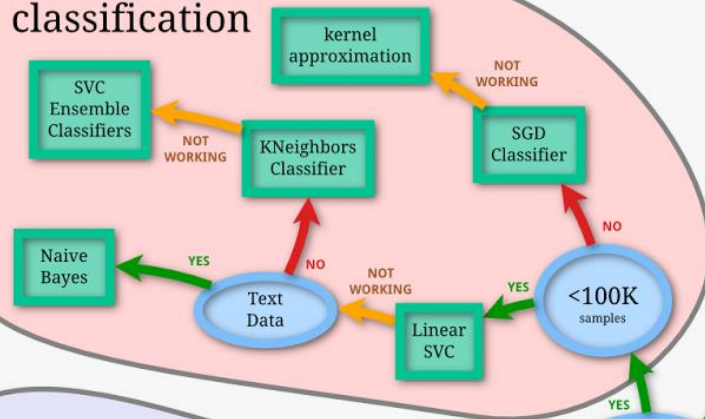


General Best Practices

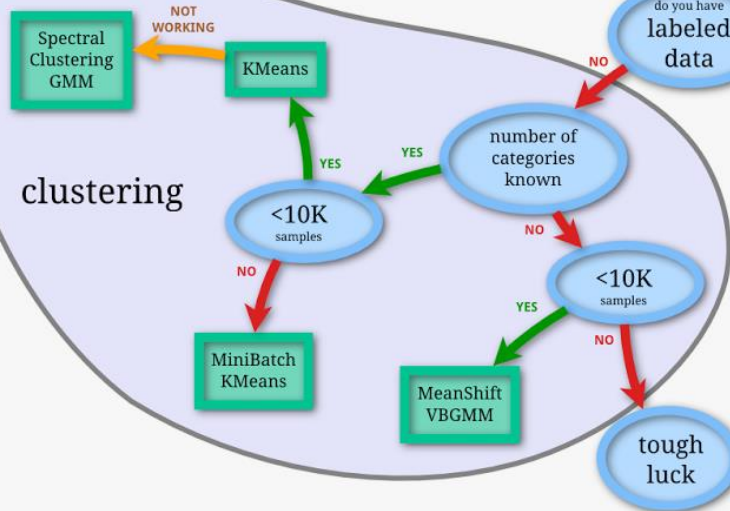
- it will be harder than it looks
- know your application:
 - zero values
 - outliers
 - where do labels come from
- Document, document, document
 - for yourself! And for others
- commit, pull, push, repeat

scikit-learn algorithm cheat-sheet

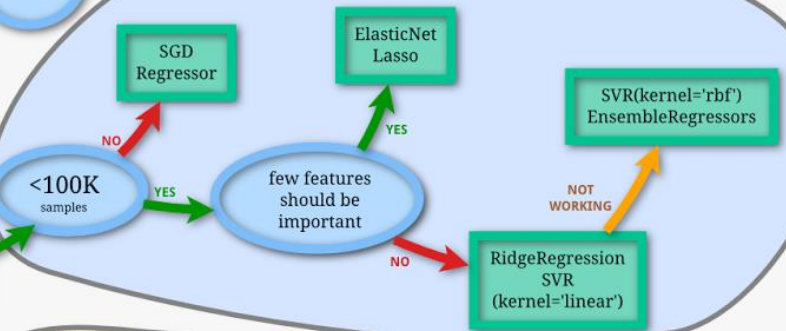
classification



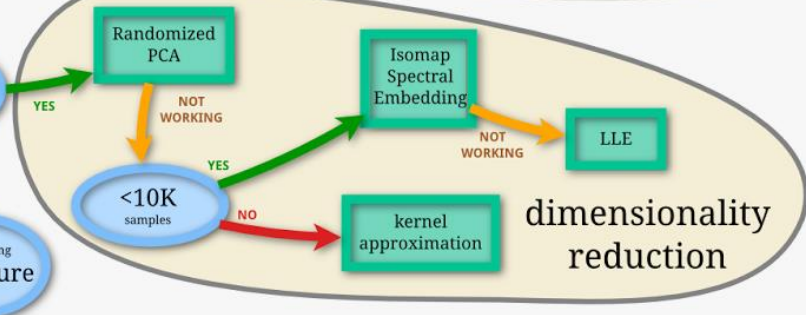
clustering

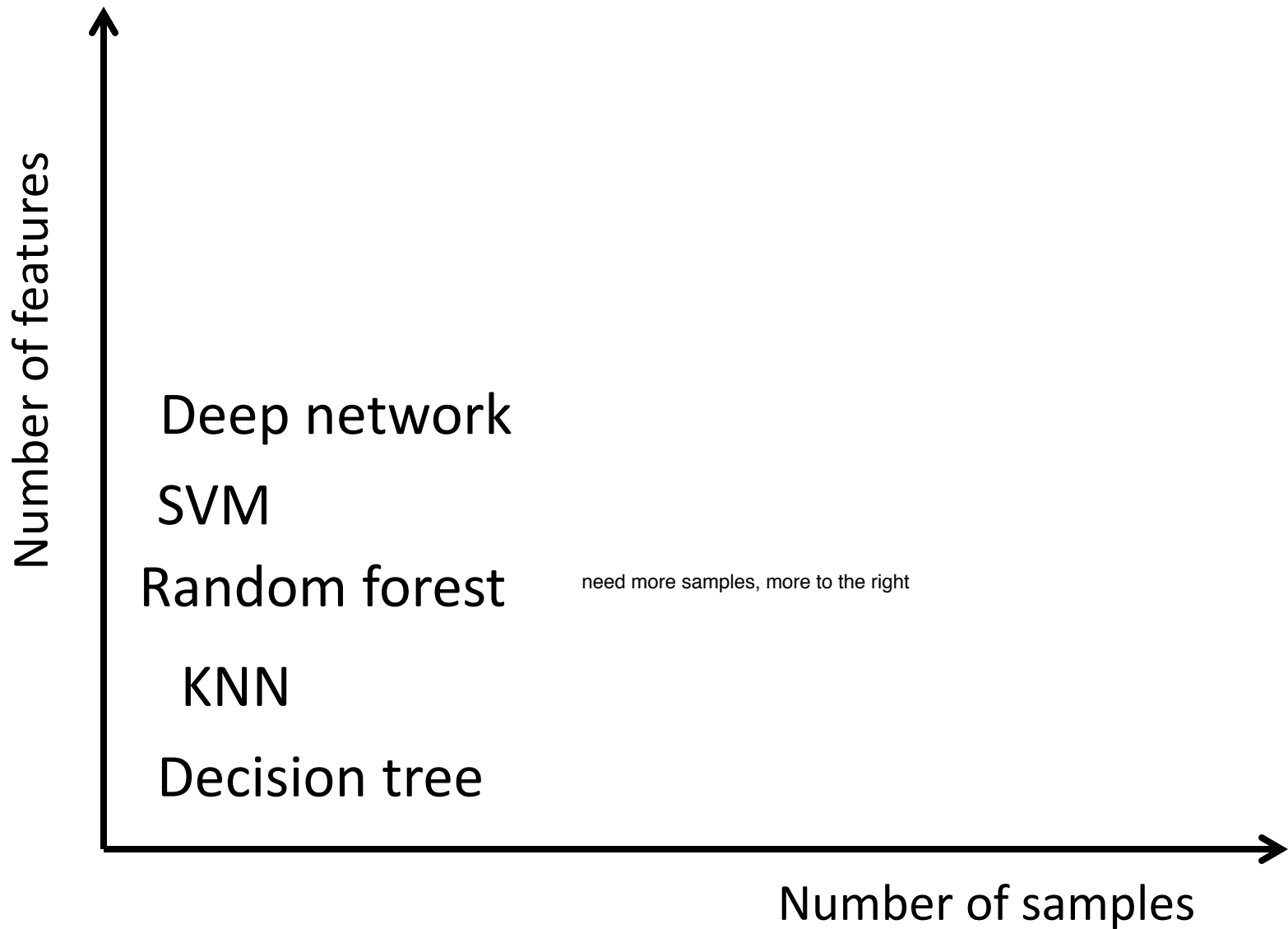


regression



dimensionality reduction





Cross Validation



training
data



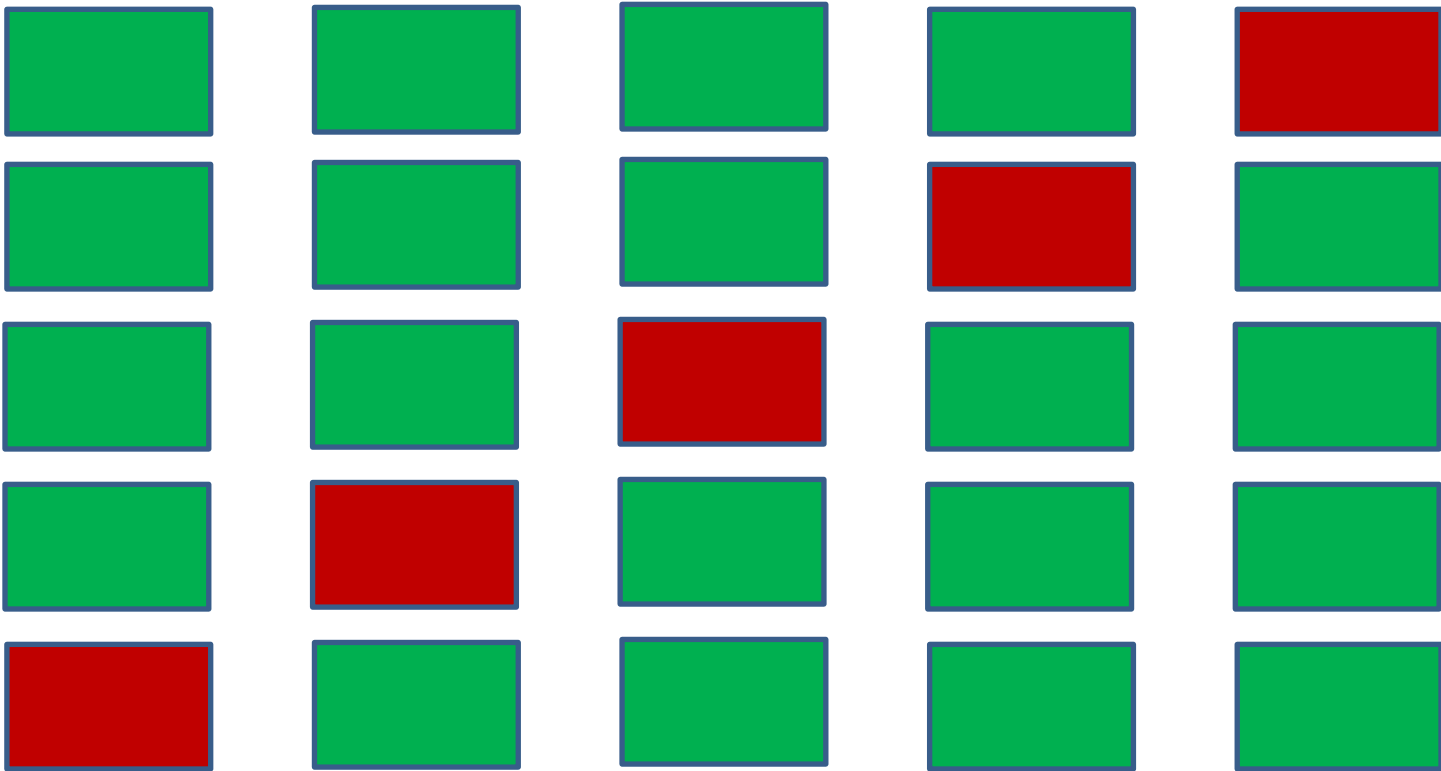
validation
data



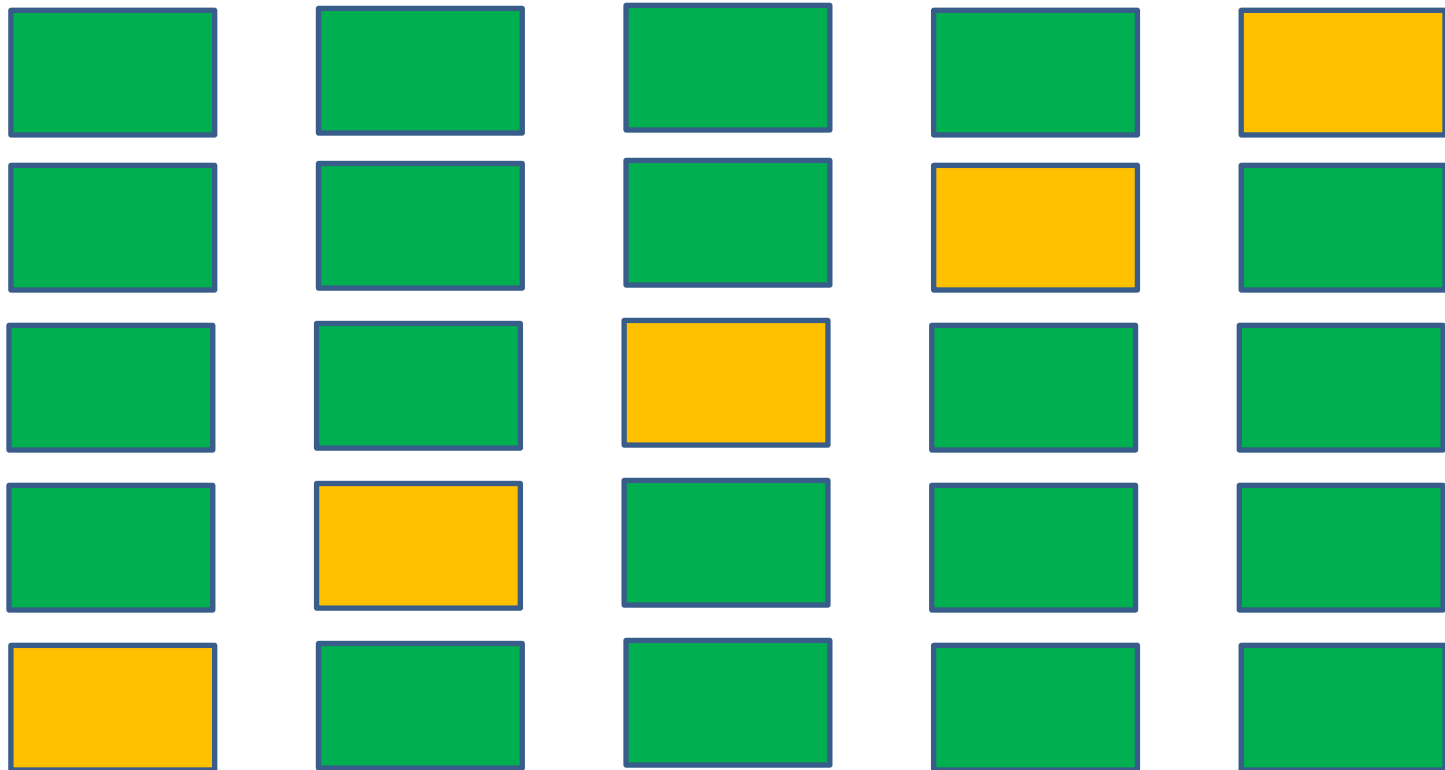
test
data

- Training data: train classifier
 - Validation data: estimate hyper parameters
 - Test data: estimate performance
-
- Be mindful of validation and test set, validation set might refer to test set in some papers.

5 – Fold Cross Validation




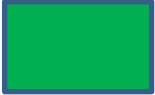


5 – Fold Cross Validation



Then test

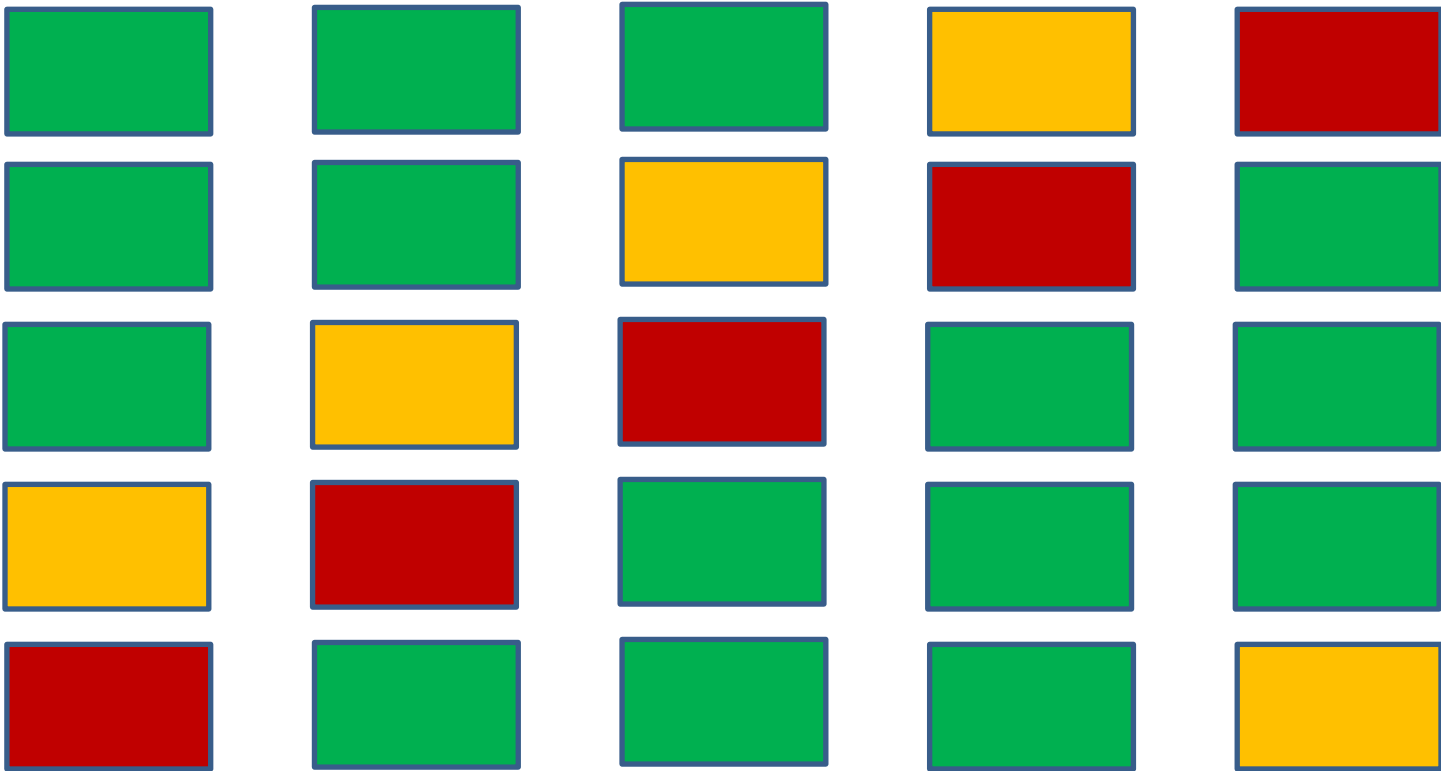


Last Step of Each Fold

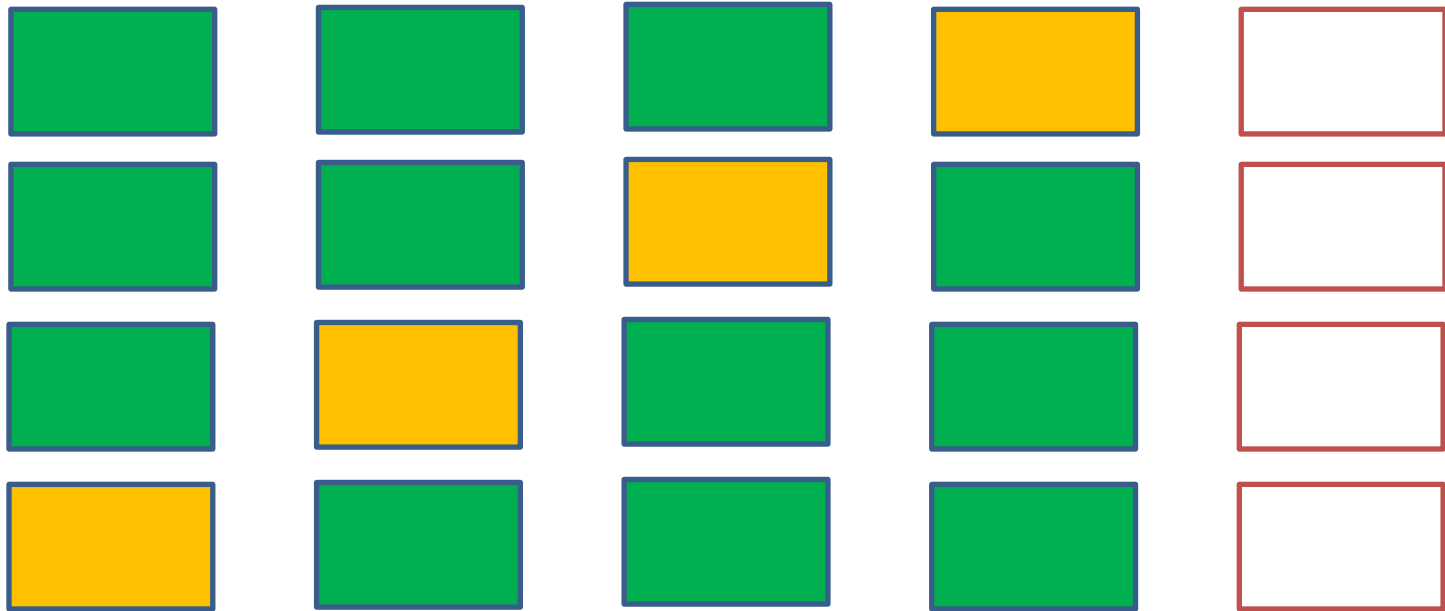
1. Take best parameters 
2. Train on training data and validation data together  
3. Test performance on test data 

This is the **final** result of your method.

5 – Fold Cross Validation



5 – Fold Cross Validation



Things to Keep in Mind

- How do you aggregate the parameters?
- What if the hyperparameters are all over the place?
each fold is too small?
- What if the hyperparameters are at the border of your grid search window?
widen window and go again

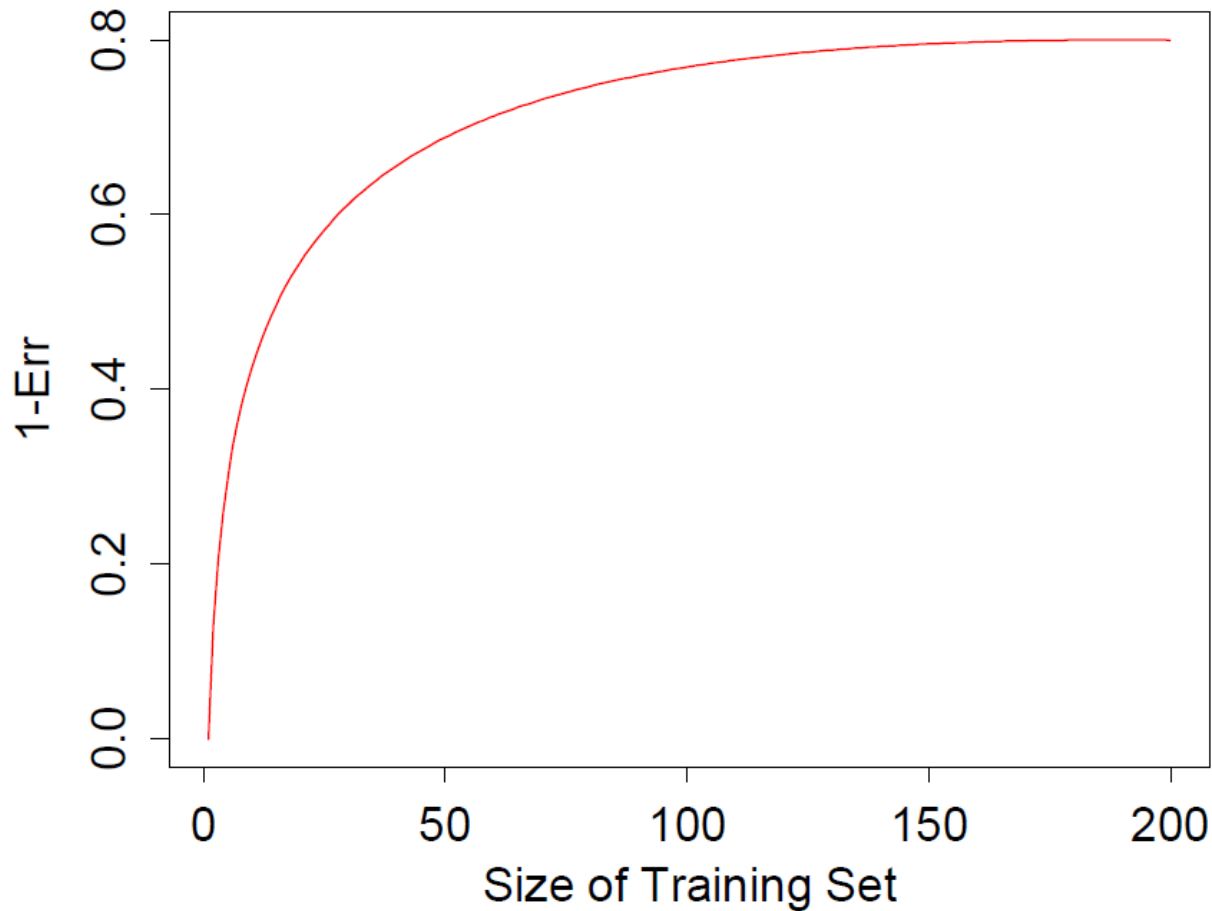
Scenario - 1

- 1. Screen the predictors: find a subset of “good” predictors that show fairly strong (univariate) correlation with the class labels
- 2. Using just this subset of predictors, build a multivariate classifier.
- 3. Use cross-validation to estimate the unknown tuning parameters and to estimate the prediction error of the final model.

Scenario - 2

- 1. Divide the samples into K cross-validation folds (groups) at random.
- 2. For each fold $k = 1, 2, \dots, K$
 - Find a subset of “good” predictors that show fairly strong (uni-variate) correlation with the class labels, using all of the samples except those in fold k .
 - Using just this subset of predictors, build a multivariate classifier, using all of the samples except those in fold k .
 - Use the classifier to predict the class labels for the samples in fold k .

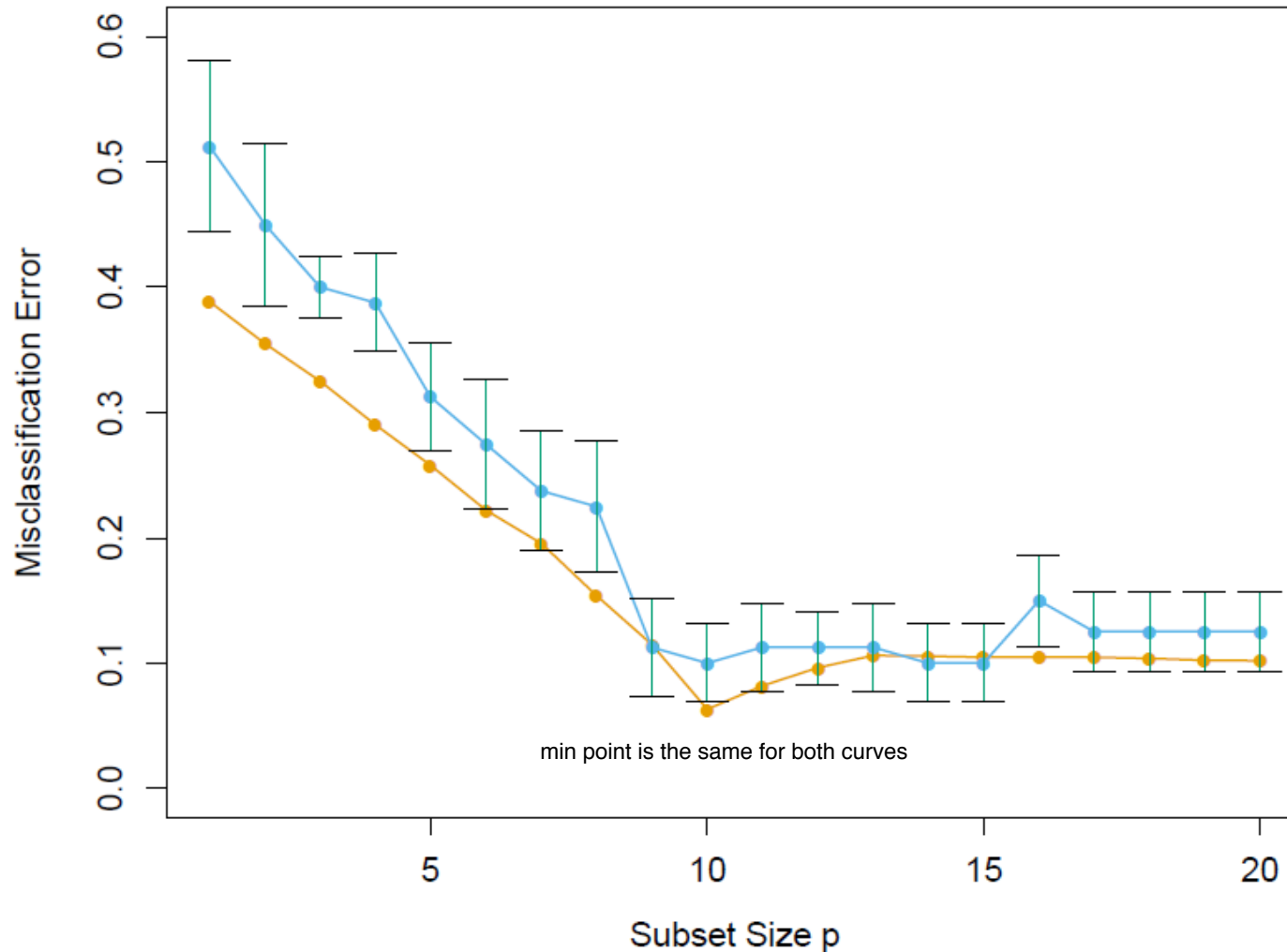
Effect of Sample Size



5-fold cross validation:

- $n=200 \Rightarrow 160$ samples
- $n=50 \Rightarrow 40$ samples

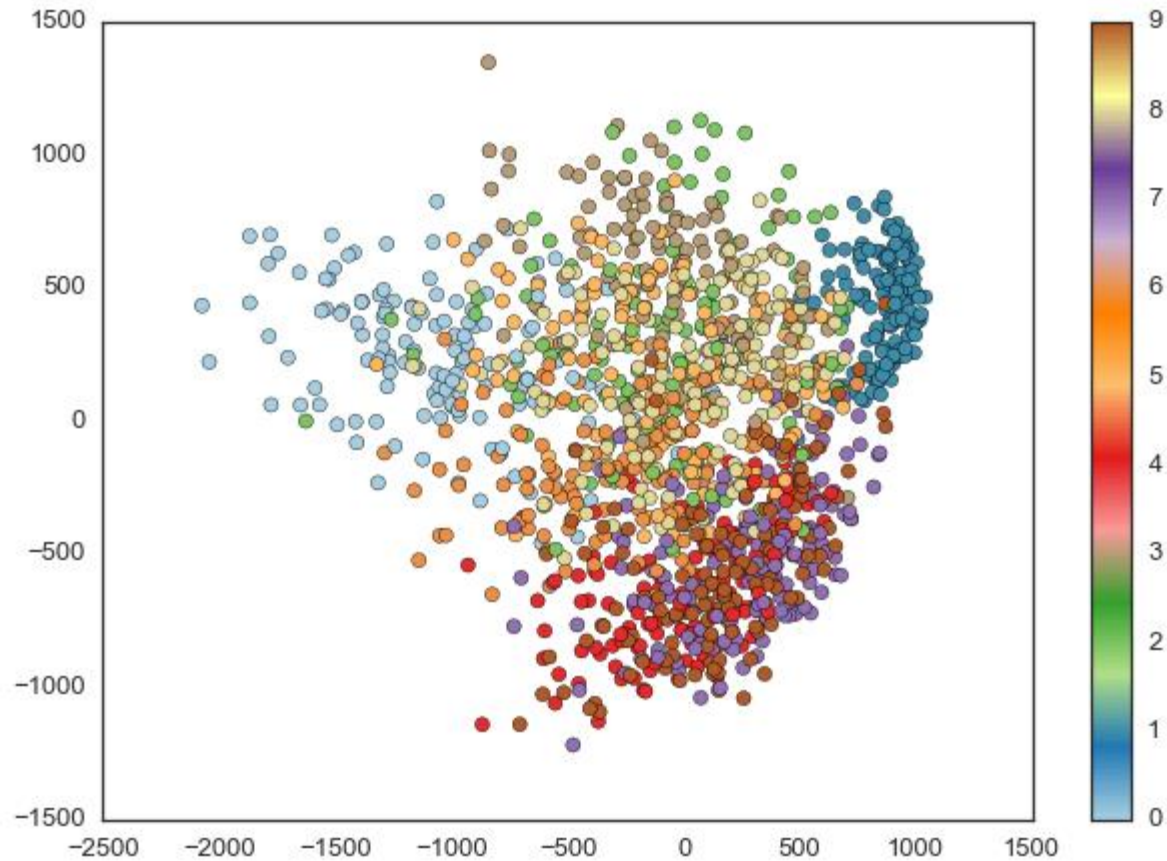
Cross Validation Over Estimates Error



Normalization

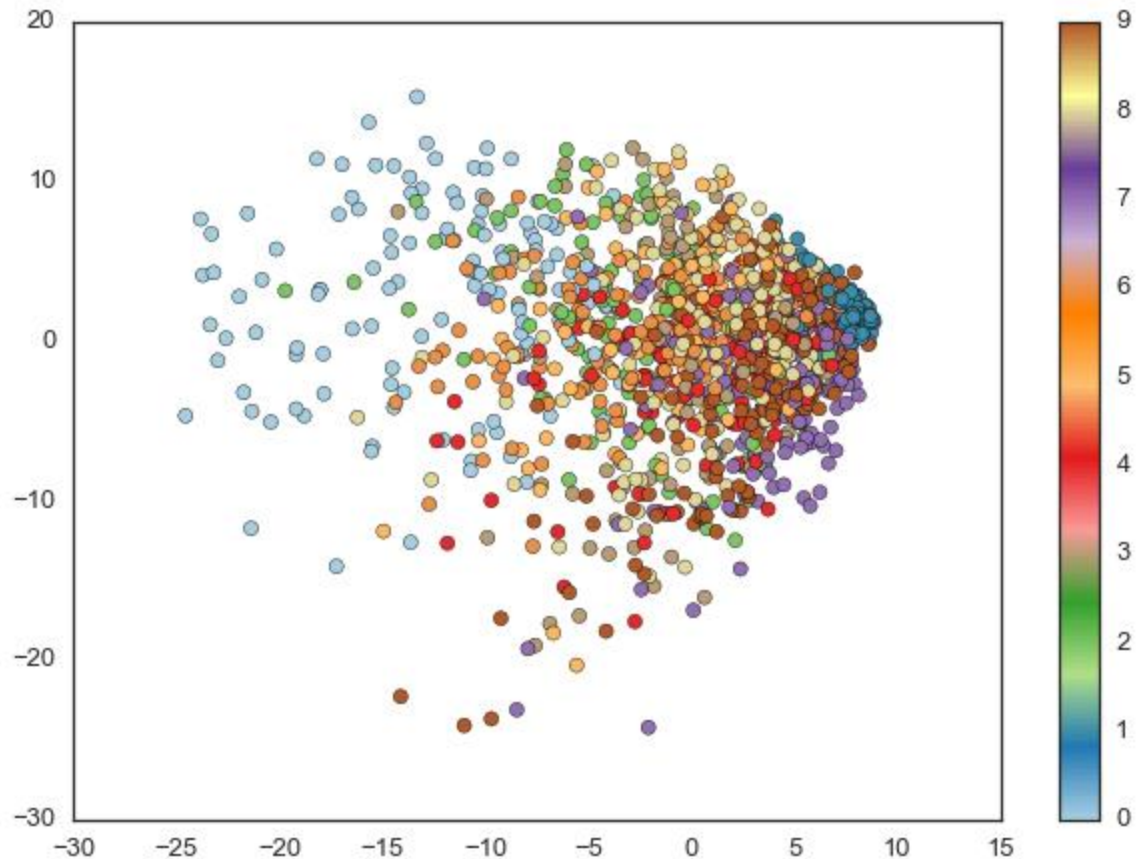
- Be very careful.
- Do not leak into the test data.
- Think about what is useful.

Example PCA on MNIST



standard PCA

Example PCA on MNIST



PCA with normalized std dev

Normalization - 1



training



Estimate
mean
values and
normalize.



validation



Estimate
mean
values and
normalize.

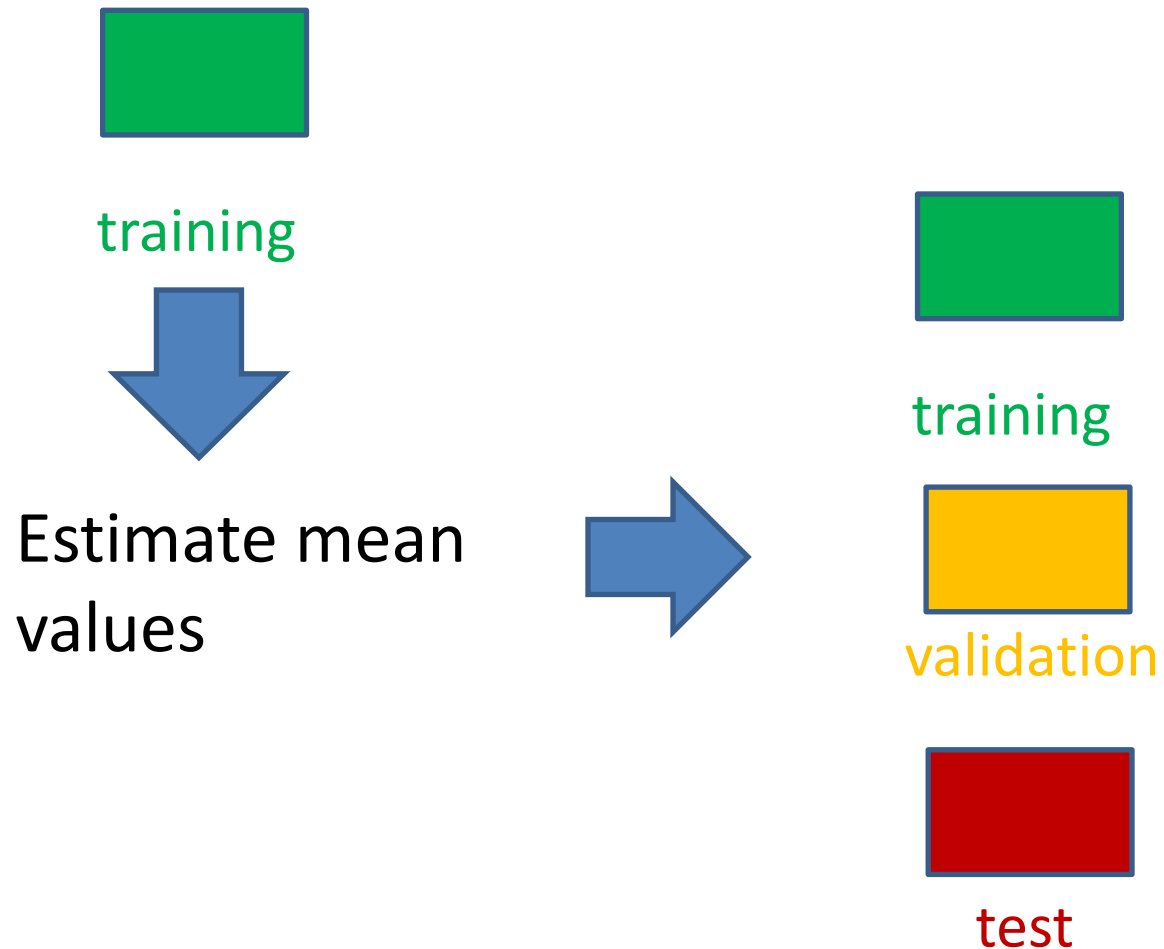


test

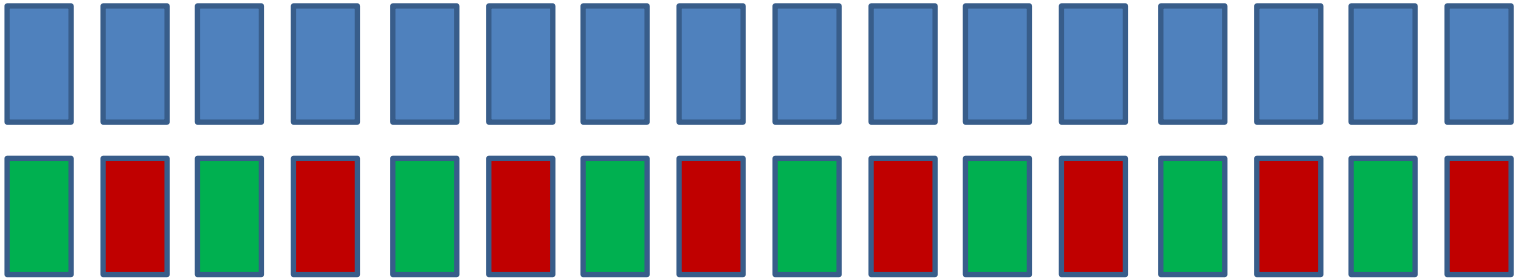


Estimate
mean
values and
normalize.

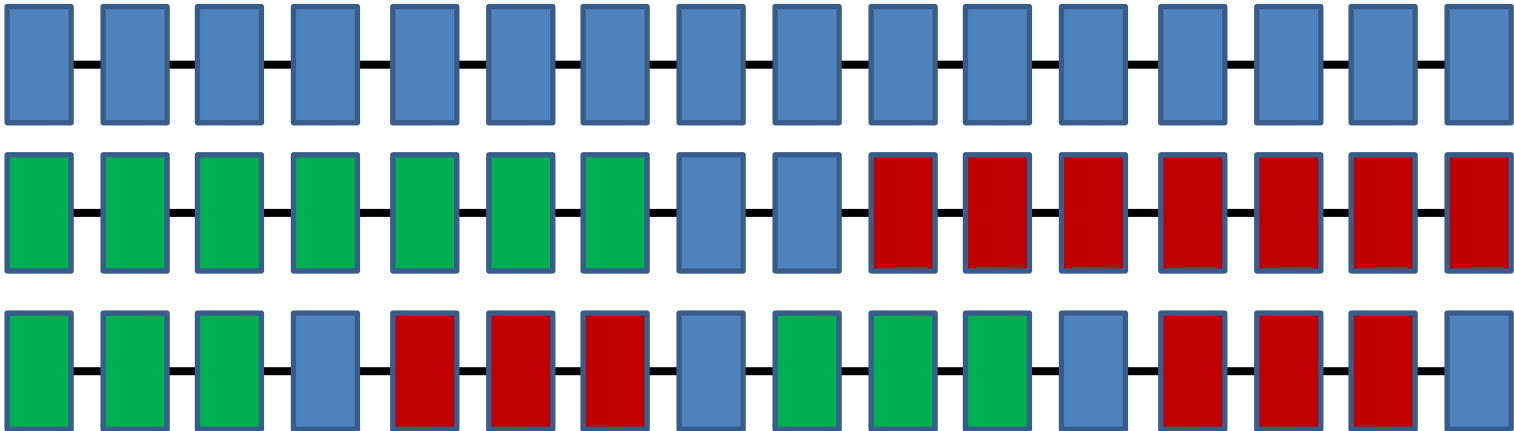
Normalization - 2



Know Your Data



make the test and training data uncorrelated

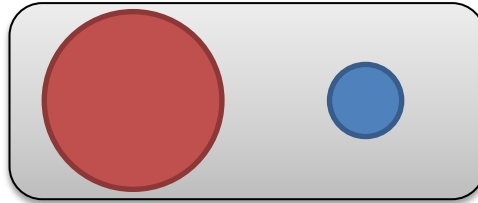


Imbalanced Data

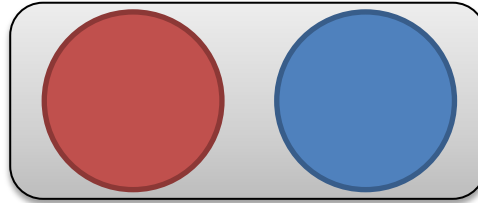
- subsample
- oversample
- re-weight sample points
- use clustering to reduce majority class
- re-calibrate classifier output
- Beware the easy true negatives

Imbalanced Classes

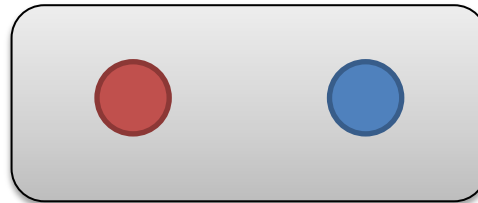
- The Problem:



- Oversample:

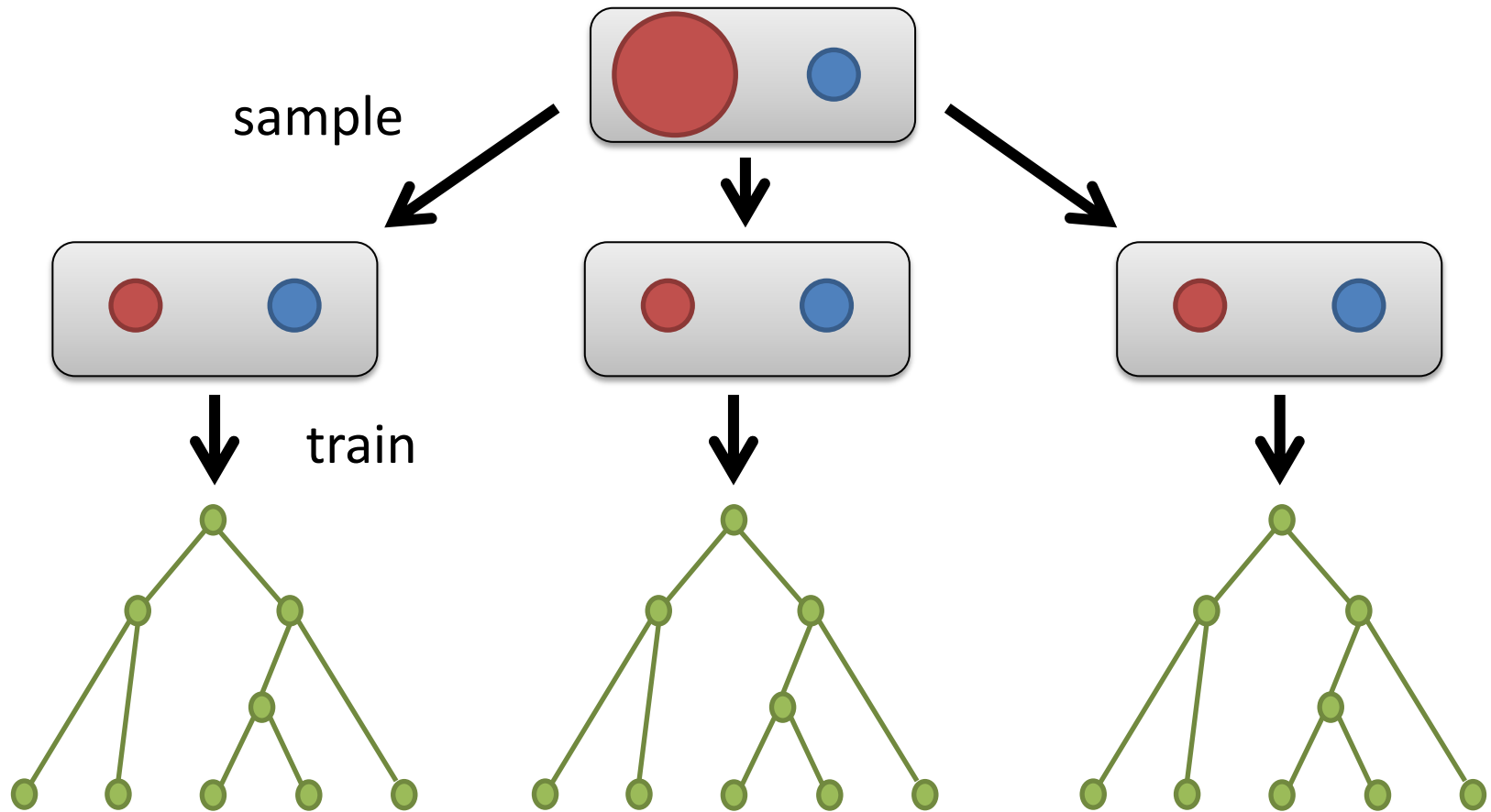


- Subsample:

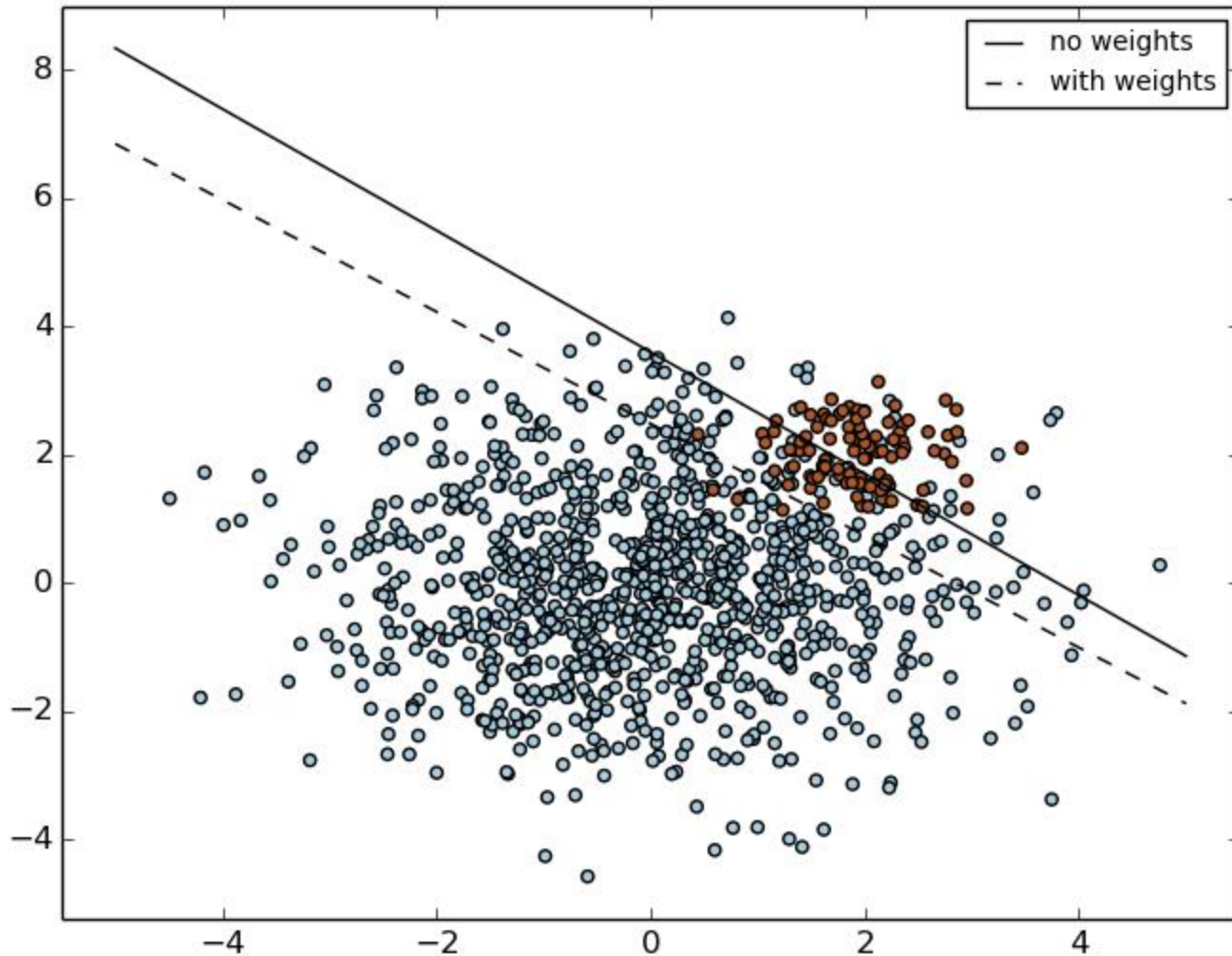


- Subsample for each tree in a random forest

Example: Random Forest Subsampling



Class Weights



http://scikit-learn.org/stable/_images/plot_separating_hyperplane_unbalanced_0011.png

Cross Validation with Imbalanced Classes

- Think about using stratified sampling to generate the folds
- The goal is to have the same class ratio in training, validation and test set.

Missing data

- Delete data points
 - Can cause sample size to be way too small
- Use the mean of the feature
 - Does not change the sample mean, but is independent of the other features.
- Use regression to estimate the value
 - Values will be deterministic

Recommender Systems

- We are already surrounded by them



Good Resources (also for this lecture)


























Survey on recommender systems by Michael D. Ekstrand et al.

- <http://files.grouplens.org/papers/FnT%20CF%20Recsys%20Survey.pdf>

Good slides from Stanford lecture by Lester Mackey

- <http://web.stanford.edu/~lmackey/papers/cf-slides-pml09.pdf>

Rating Matrix Completion Problem

Collaborative Filtering

Insight: Personal preferences are correlated

- If Jack loves A and B, and Jill loves A, B, and C, then Jack is more likely to love C
- Does not rely on item or user attributes (e.g. demographic info, author, genre)

Content-based Filtering

- Each item is described by a set of features
- Measure similarity between items
- Recommend items that are similar to the items the User liked

Comparison

- Collaborative filtering:
 - Items entirely described by user ratings
 - Good for new discoveries
 - People who like SciFi maybe also like Fantasy
- Content-based filtering:
 - Predictions are in users comfort zone
 - Can start with a single item
- Can do a hybrid approach

User Based Collaborative Filtering


























Intuition:

- I like what people similar to me like
- Users give ratings
- People with similar ratings in the past assumed to have similar ratings in the future

Item-based Collaborative Filtering

- Similar, but looks at the items instead of the users
- Useful if the user base is way larger than the number of items.
- More useful: Items are relatively stable in their rating, users vary more.

Short Recap of Terminology

We Could Use Missing Data Strategies

All that we talked about earlier:

- Omitting samples
- Using the mean rating of an item
- Doing regression



CF as Regression

- Choose favorite regression algorithm
- Train a predictor for each item
- Each user who rated that item provides one sample
- To predict rating of an item A , apply predictor for A to the user's incomplete ratings vector.

Recommendation by Regression

- **Pros:**









- Reduces recommendations to a well-studied problem
- Many good prediction algorithms available

- **Cons:**

- Have to handle tons of missing data
- Training M predictors is expensive

KNN for Collaborative Filtering

- Widely used
- Item-based and User-based focus
- Represent each user as incomplete vector of item ratings
- Compute similarity between query user and all other users
- Find K most similar users who rated the query item
- Predict weighted average of ratings

Similarity Measures

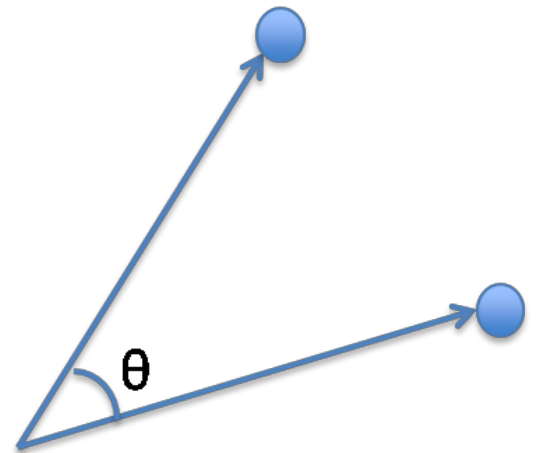
- Pearson Correlation Coefficient
 - bound between 1 and -1
 - suffers from computing high similarity between users with few ratings in common
 - set threshold for minimum number of co-rated items suffers from computing high similarity between users with few ratings in common

$$s(u, v) = \frac{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{i \in I_u \cap I_v} (r_{v,i} - \bar{r}_v)^2}}$$

Similarity Measures

- Cosine similarity
 - vector-space approach based on linear algebra
 - Unknown ratings are considered to be 0
 - this causes them to effectively drop out of the numerator

$$\text{sim}(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$



Dimensionality Reduction

- We have treated item or user ratings as vectors
- In many dimensions
- With lots of missing data
- Can we find a low-dimensional sub-space that captures our data?

Eigentaste

- <http://eigentaste.berkeley.edu/index.html>



A dog goes into a bar and orders a martini. The bartender says, "You don't see a dog in here drinking a martini very often."

The dog says, "At these prices, I'm not surprised."

Less Funny

More Funny



Next

PCA for Recommendations

- Compute PCA of a dense rating matrix
- Keep the k largest components
- Project users into k -dimensional subspace
- Cluster users in k -dimensions
- Recommend jokes based on users cluster

Singular Value Decomposition

The diagram illustrates the Singular Value Decomposition (SVD) of a matrix \mathbf{R} . The matrix \mathbf{R} is represented by a large rectangle with dimensions $|U|$ (height) and $|I|$ (width). It is equal to the product of three matrices: \mathbf{U} , Σ , and \mathbf{T}^T . Matrix \mathbf{U} is a tall, narrow rectangle. Matrix Σ is a small square with dimension k indicated above it. Matrix \mathbf{T}^T is a wide, short rectangle.

$$\begin{matrix} |I| \\ \boxed{\mathbf{R}} \\ |U| \end{matrix} = \boxed{\mathbf{U}} \begin{matrix} k \\ \boxed{\Sigma} \end{matrix} \boxed{\mathbf{T}^T}$$

- This is what Simon Funk did for the Netflix prize:
<http://sifter.org/~simon/journal/20061027.2.htm>
|

Singular Value Decomposition

The diagram illustrates the Singular Value Decomposition (SVD) of a matrix \mathbf{R} . The matrix \mathbf{R} is represented by a large rectangle with dimensions $|U|$ (height) and $|I|$ (width). It is equal to the product of three matrices: \mathbf{U} , Σ , and \mathbf{T}^T . Matrix \mathbf{U} is a tall, narrow rectangle. Matrix Σ is a small square with dimension k indicated above it. Matrix \mathbf{T}^T is a wide, short rectangle.

$$\begin{matrix} |I| \\ \boxed{\mathbf{R}} \\ |U| \end{matrix} = \boxed{\mathbf{U}} \begin{matrix} k \\ \boxed{\Sigma} \end{matrix} \boxed{\mathbf{T}^T}$$

- Decomposes each matrix into three components
- Σ is diagonal and the entries are the singular values of the decomposition

Singular Value Decomposition

The diagram illustrates the Singular Value Decomposition (SVD) of a matrix \mathbf{R} . The matrix \mathbf{R} is represented by a large rectangle with dimensions $|U|$ (height) and $|I|$ (width). It is equal to the product of three matrices: \mathbf{U} , Σ , and \mathbf{T}^T . Matrix \mathbf{U} is a tall, narrow rectangle. Matrix Σ is a small square with dimension k indicated above it. Matrix \mathbf{T}^T is a wide, short rectangle.

$$\begin{matrix} |I| \\ \boxed{\mathbf{R}} \end{matrix} = \begin{matrix} \boxed{\mathbf{U}} \\ |U| \end{matrix} \begin{matrix} k \\ \boxed{\Sigma} \end{matrix} \begin{matrix} \boxed{\mathbf{T}^T} \end{matrix}$$

- \mathbf{U} and \mathbf{T} are orthogonal
- As in PCA we can truncate Σ to compute a lower rank approximation of \mathbf{R} .

Singular Value Decomposition

The diagram illustrates the Singular Value Decomposition (SVD) of a matrix \mathbf{R} . The matrix \mathbf{R} is represented by a large rectangle with dimensions $|U|$ (height) and $|I|$ (width). It is equal to the product of three matrices: \mathbf{U} , Σ , and \mathbf{T}^T . Matrix \mathbf{U} is a tall, narrow rectangle. Matrix Σ is a small square with dimension k above it. Matrix \mathbf{T}^T is a wide, short rectangle.

$$\begin{matrix} |I| \\ \boxed{\mathbf{R}} \\ |U| \end{matrix} = \boxed{\mathbf{U}} \begin{matrix} k \\ \boxed{\Sigma} \end{matrix} \boxed{\mathbf{T}^T}$$

- Rows of \mathbf{U} are users interest in the k inferred topics
- Rows of \mathbf{T} are the items relevance for each topic

Singular Value Decomposition

The diagram illustrates the Singular Value Decomposition (SVD) of a matrix \mathbf{R} . The matrix \mathbf{R} is represented by a large rectangle with dimensions $|U|$ (height) and $|I|$ (width). It is equal to the product of three matrices: \mathbf{U} , Σ , and \mathbf{T}^T . Matrix \mathbf{U} is a tall, narrow rectangle. Matrix Σ is a small square with dimension k indicated above it. Matrix \mathbf{T}^T is a wide, short rectangle.

$$\begin{matrix} |I| \\ \boxed{\mathbf{R}} \end{matrix} = \begin{matrix} \boxed{\mathbf{U}} \\ |U| \end{matrix} \begin{matrix} k \\ \boxed{\Sigma} \end{matrix} \begin{matrix} \boxed{\mathbf{T}^T} \end{matrix}$$

- A user's preference for an item, therefore, is the weighted sum of the user's interest in each of the topics times that item's relevance to the topic.

Singular Value Decomposition

The diagram illustrates the Singular Value Decomposition (SVD) of a matrix \mathbf{R} . It shows the equation $\mathbf{R} = \mathbf{U} \mathbf{\Sigma} \mathbf{T}^T$. Matrix \mathbf{R} is a large rectangle with dimensions $|U|$ (height) and $|I|$ (width). Matrix \mathbf{U} is a tall, narrow rectangle. Matrix $\mathbf{\Sigma}$ is a small square with dimension k (width). Matrix \mathbf{T}^T is a wide, short rectangle.

- If we know the SVD, we could compute the missing values in \mathbf{R} .
- Try to infer SVD from matrix with missing data, and reconstruct full matrix \mathbf{R}

