Supervised Learning for Predictive Analytics

More Classification Algorithms

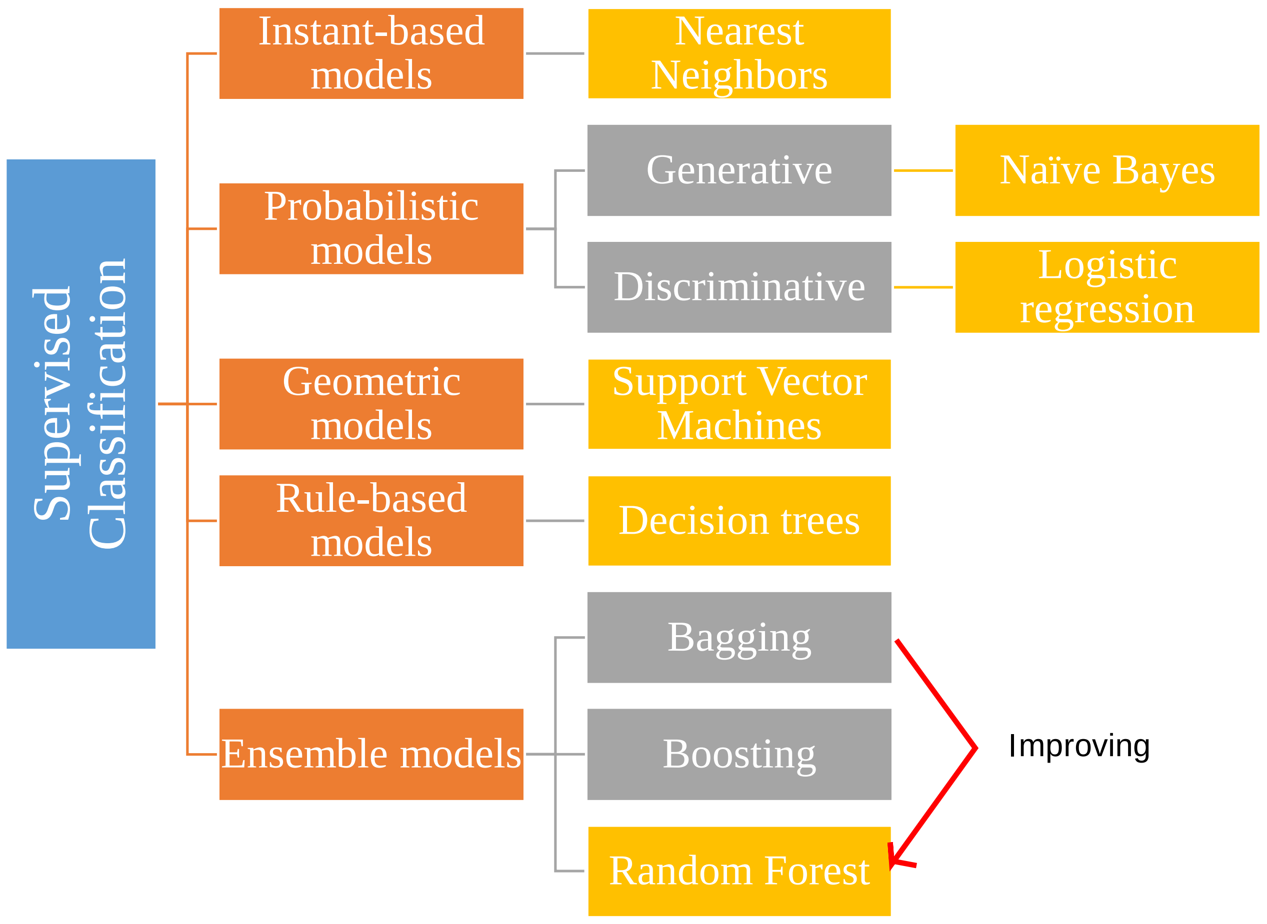
Dr. Unchalisa Taetragool

Department of Computer Engineering, Faculty of Engineering
King Mongkut's University of Technology Thonburi

Classification Recap

- Two-step process:
  - **Learning** – a model construction
  - **Applying** – a model usage
- In model construction, we describe a set of pre-determined classes:
  - Each record is assumed to belong to a predefined class based on its features
  - The set of records is used for model construction is a **training set**
- The trained model is then applied to **unseen data** to classify those records into the predefined classes
- Model should fit well to training data and have strong predictive power
  - Do NOT overfit a model, as that results in low predictive power

Classification Methods



Classification Methods

- There is no “BEST” method
- Methods can be selected based on metrics (accuracy, precision, recall, F-measure), speed, robustness, and scalability

Outline

- Ensemble methods
  - Bagging and Boosting trees
  - Random forest and XGBoost algorithms
- Support Vector Machines

Ensemble Learning

- Ensemble learning
  - The process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem.
- Ensemble classification
  - Aggregation of predictions of multiple classifiers with the goal of improving accuracy.

Machine learning competition with a $1 million prize

Leaderboard

Display top 20 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
1	The Ensemble	0.8553	10.10	2009-07-26 18:38:22
2	BellKor's Pragmatic Chaos	0.8554	10.09	2009-07-26 18:18:28
Grand Prize - RMSE <= 0.8563				
3	Grand Prize Team	0.8571	9.91	2009-07-24 13:07:49
4	Opera Solutions and Vandelay United	0.8573	9.89	2009-07-25 20:05:52
5	Vandelay Industries !	0.8579	9.83	2009-07-26 02:49:53
6	PragmaticTheory	0.8582	9.80	2009-07-12 15:09:53
7	BellKor in BigChaos	0.8590	9.71	2009-07-26 12:57:25
8	Dace_	0.8603	9.58	2009-07-24 17:18:43
9	Opera Solutions	0.8611	9.49	2009-07-26 18:02:08
10	BellKor	0.8612	9.48	2009-07-26 17:19:11
11	BigChaos	0.8613	9.47	2009-06-23 23:06:52
12	Feeds2	0.8613	9.47	2009-07-24 20:06:46
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				
13	xiangliang	0.8633	9.26	2009-07-21 02:04:40
14	Gravity	0.8634	9.25	2009-07-26 15:58:34
15	Ces	0.8642	9.17	2009-07-25 17:42:38
16	Invisible Ideas	0.8644	9.14	2009-07-20 03:26:12
17	Just a guy in a garage	0.8650	9.08	2009-07-22 14:10:42
18	Craig Carmichael	0.8656	9.02	2009-07-25 16:00:54
19	J Dennis Su	0.8658	9.00	2009-03-11 09:41:54
20	acmehill	0.8659	8.99	2009-04-16 06:29:35
Progress Prize 2007 - RMSE = 0.8712 - Winning Team: KorBell				
Cinematch score on quiz subset - RMSE = 0.9514				

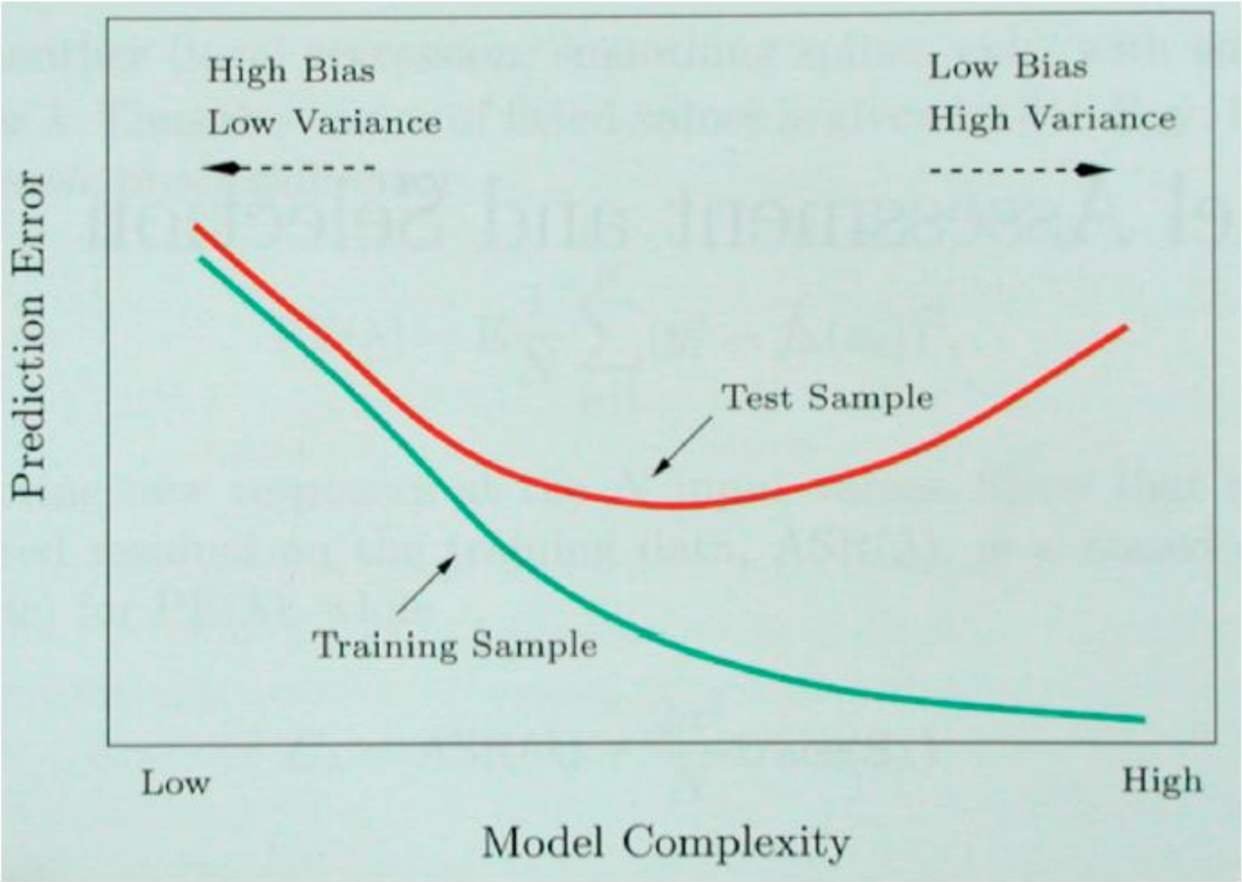
users

movies						
	1		?	3	5	?
	?	1				2
		4		4	5	?

Ensemble Learning

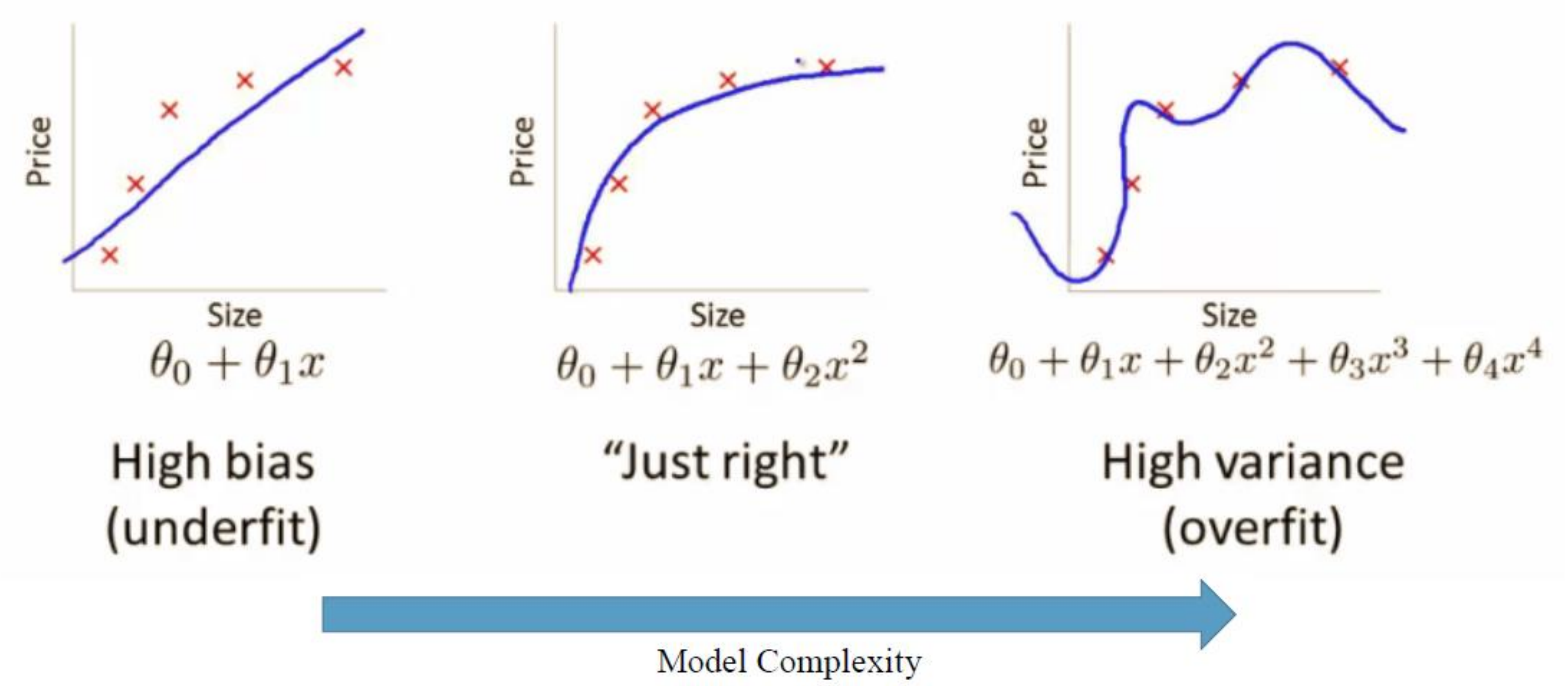
- Utility of combining diverse, independent opinions in human decision making
- Majority vote
  - Suppose we have 5 completely independent classifiers…
  - If accuracy is 70% for each
    - $10\ (.7^3)(.3^2) + 5\ (.7^4)(.3) + (.7^3)$
    - 83.7% accuracy
  - 101 such classifiers
    - 99.9% accuracy

Bias-Variance Tradeoff



Hastie, Tibshirani, Friedman "Elements of Statistical Learning" 2001

Bias-Variance Tradeoff



Reducing variance without increasing bias

- Averaging reduces variance:

$$\mathrm{var}\left(\bar{X}\right) = \frac{\mathrm{var}\left(X\right)}{n}$$

  - Average models to reduce model variance
  - One problem
    - Only one training set
    - Where do multiple models come from?

Bagging: bootstrap aggregation

- Take repeated bootstrap samples from training set D.
  - Bootstrap sampling: Given set D containing N training examples, create D' by drawing N examples at random with replacement from D.
- Bagging: Procedure
  - Create k bootstrap samples $D_1 \dots D_k$
  - Train distinct classifier on each Di < Different training set >
  - Classify new instance by majority vote (classification) or average (regression)

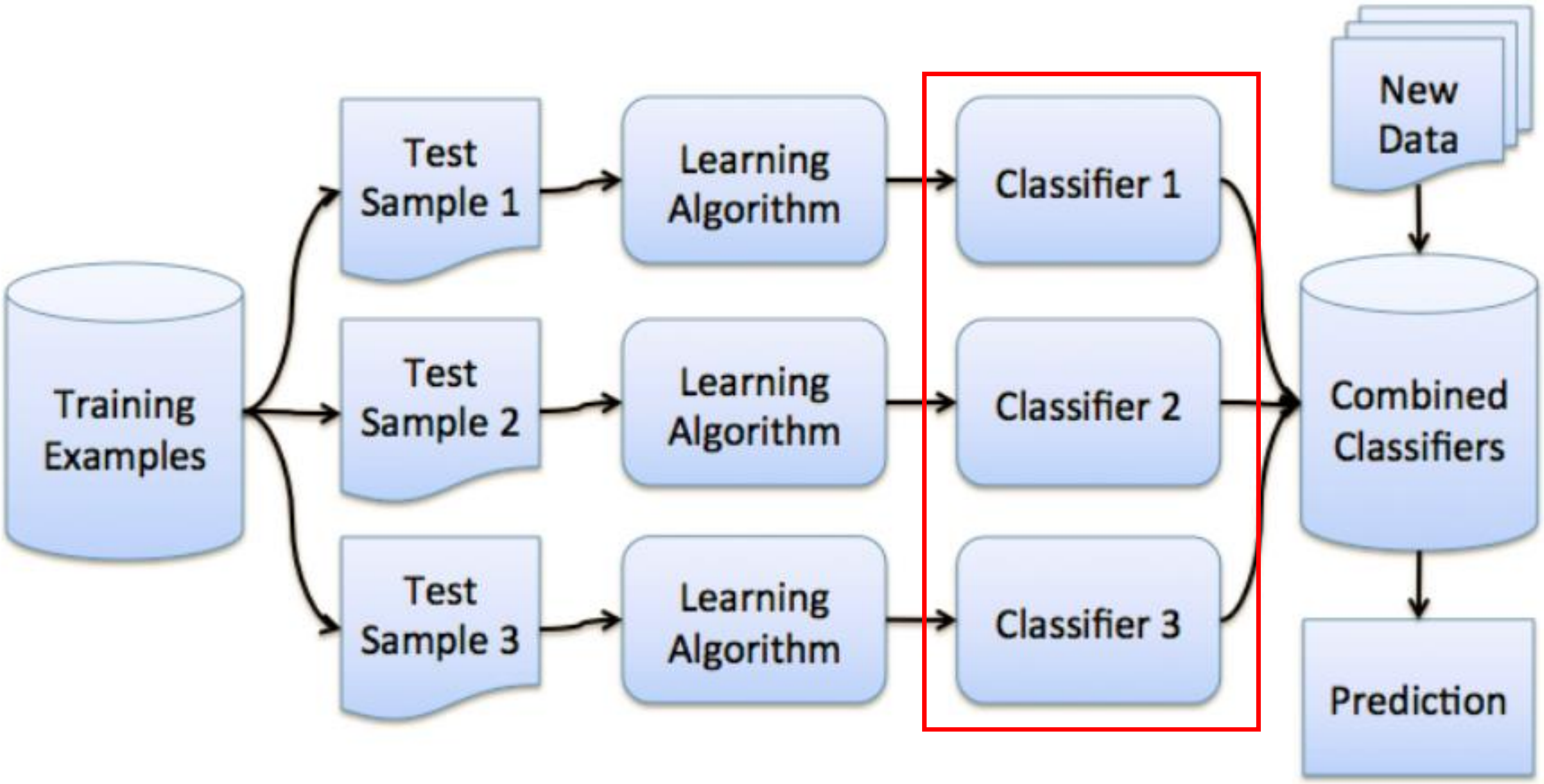
Bagging

- Best case

$$\mathrm{var}\left(bagging\left(L\left(x,D\right)\right)\right) \rightarrow \frac{\mathrm{var}\left(L\left(x,D\right)\right)}{N}$$

- In practice, models are correlated, so the reduction is smaller than 1/N
- Variance of the models trained on fewer training cases usually somewhat larger

Bagging



Training Examples
Test Sample 1
Test Sample 2
Test Sample 3
Learning Algorithm
Learning Algorithm
Learning Algorithm
Classifier 1
Classifier 2
Classifier 3
New Data
Combined Classifiers
Prediction

Reduce bias and decrease variance?

Reduce over fitting effect

- Bagging reduces variance by averaging
- In practice, bagging has little effect on bias
- Can we average and reduce bias?
- Yes: Boosting

Reduce under fitting

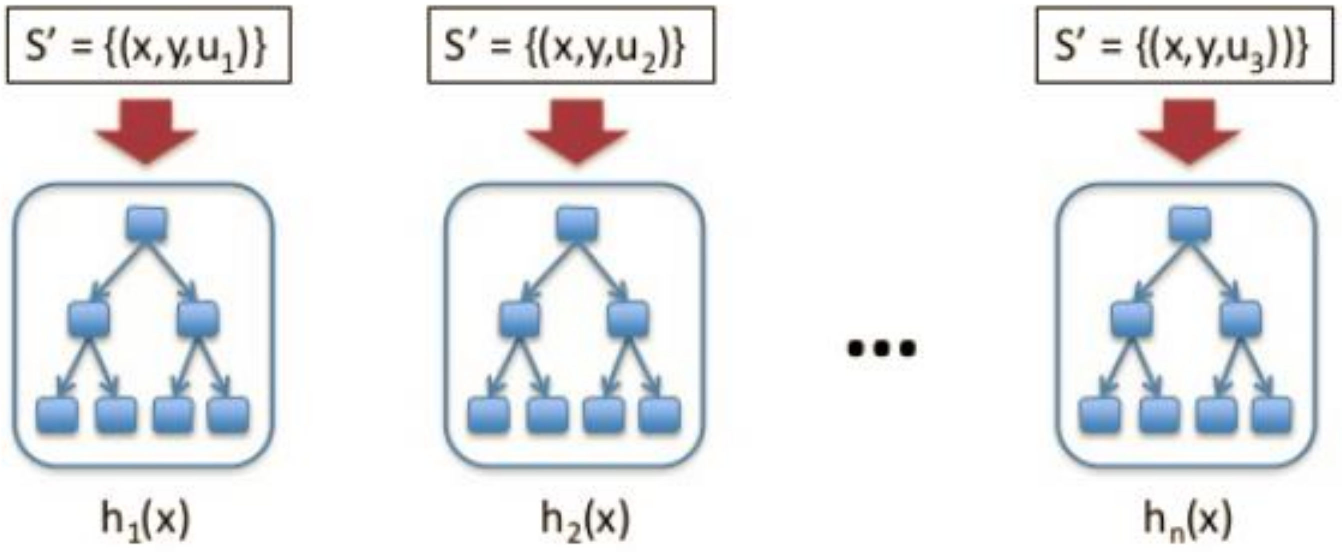
Boosting

- Boosting aims to reduce bias.
- Can a set of weak learners create a single strong learner?
- A weak learner is defined to be a classifier which is only slightly correlated with the true classification (it can label examples better than random guessing).
- In contrast, a strong learner is a classifier that is arbitrarily well-correlated with the true classification.
- It make examples currently misclassified more important (or less, in some cases)

Boosting (AdaBoost)

ปรับ weight ไปเรื่อยๆจนกว่าจะดี

$$h(x) = a_1h_1(x) + a_2h_2(x) + \ldots + a_3h_n(x)$$



u – weighting on data points

a – weight of linear combination

https://www.cs.princeton.edu/~schapire/papers/explaining-adaboost.pdf

Stop when validation performance plateaus (will discuss later)

Boosting

- Create a sequence of classifiers, giving higher influence to more accurate classifiers
- At each iteration, make examples currently misclassified more important (get larger weight in the construction of the next classifier)

ตัวที่ทายผิด

- Then, combine classifiers by weighted vote (weight given by classifier accuracy)

AdaBoost

- Advantages
  - Very little code
  - Reduce variance

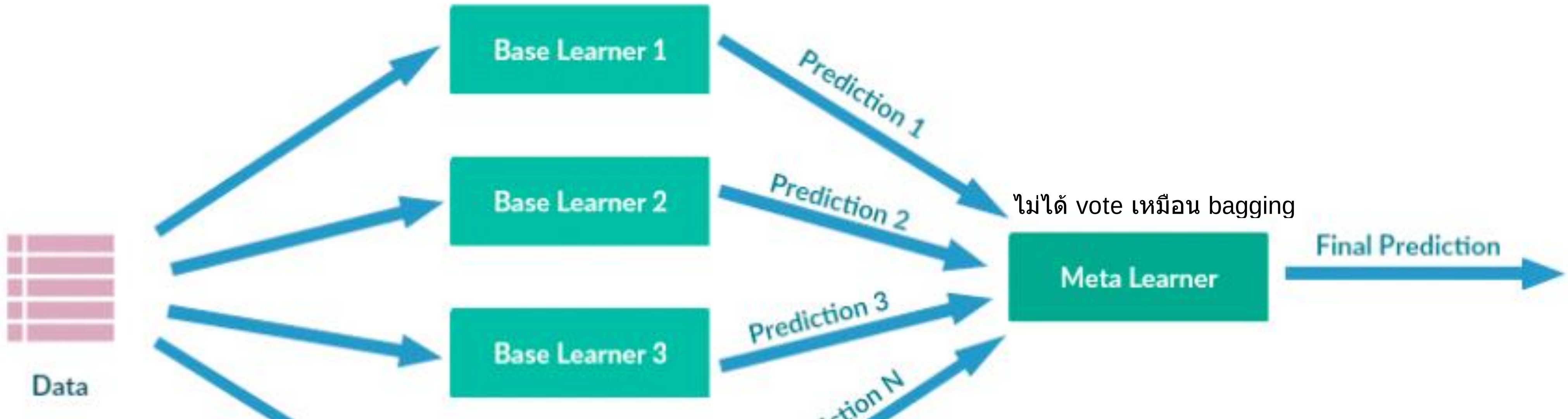
- Disadvantages
  - Sensitive to noise and outliers

เวลาไปทำให้ noise สำคัญกว่า data จริง

Stacking

- Stacking (sometimes called stacked generalization) involves training a learning algorithm to combine the predictions of several other learning algorithms
- First, all of the other algorithms are trained using the available data, then a combiner algorithm is trained to make a final prediction using all the predictions of the other algorithms as additional inputs

Stacking

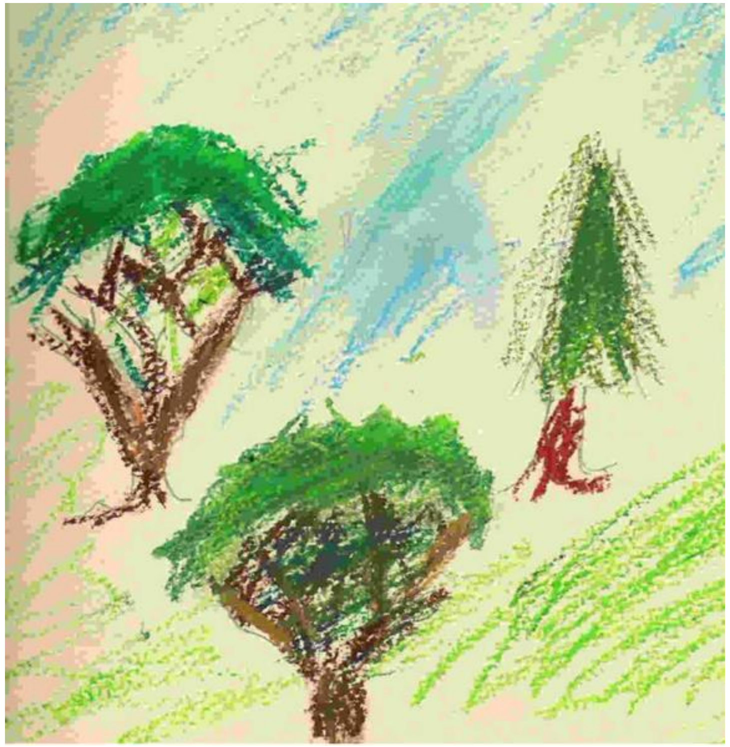


ไม่ได้ vote เหมือน bagging

credit:supunsetunga.blogspot.com

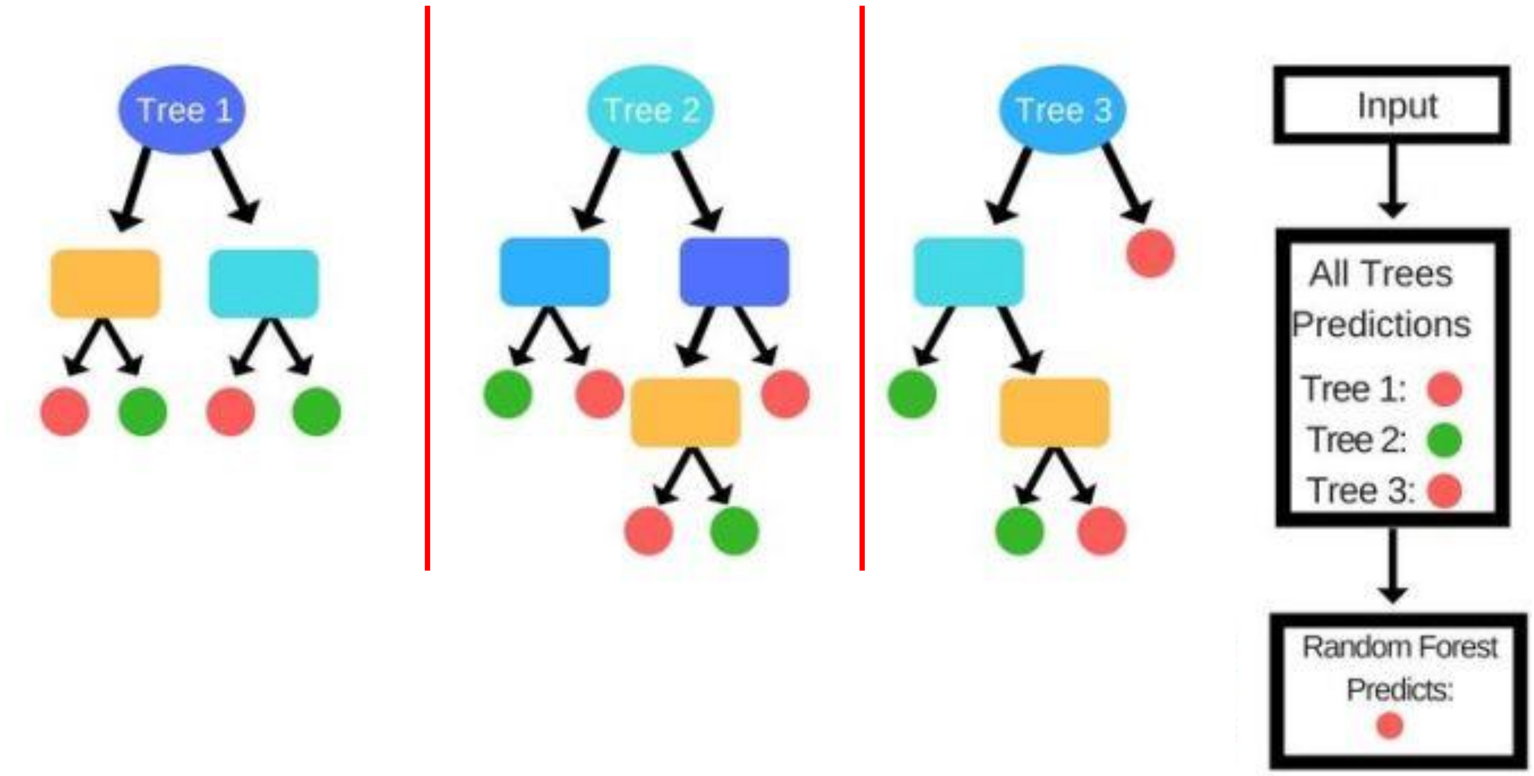
Random Forest

for regression and classification



Random Forest

Separating model



majority vote

Random forest

random both data, and feature selection

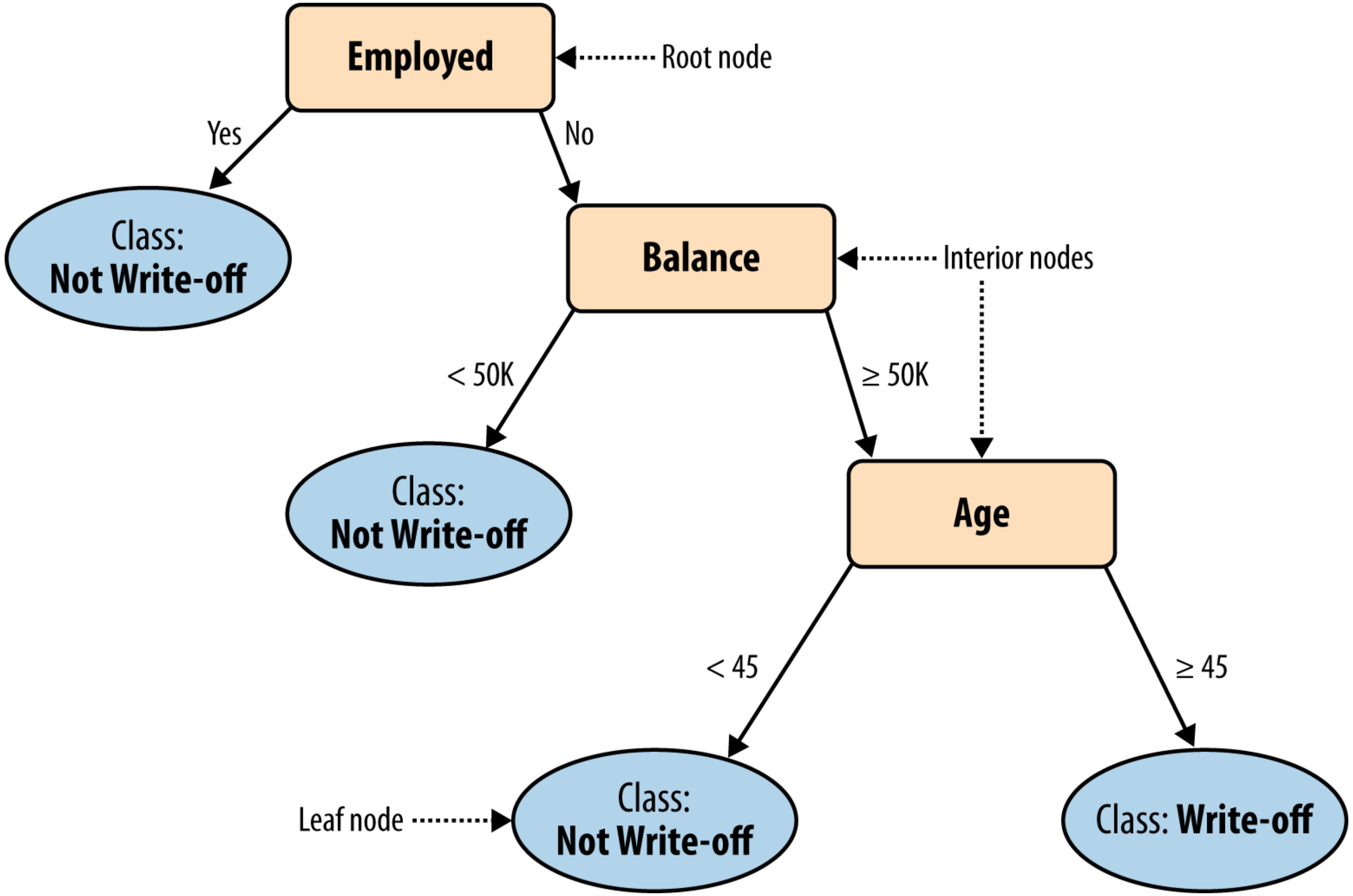
- An ensemble of decision tree trained by bootstrap sampling and random feature selection
- It is based on decision trees: classifiers constructed greedily using the conditional entropy
- The extension hinges on two ideas:
  - building an ensemble of trees by training on subsets of data
  - considering a reduced number of possible variables at each node

Classification and Regression Trees Pioneers

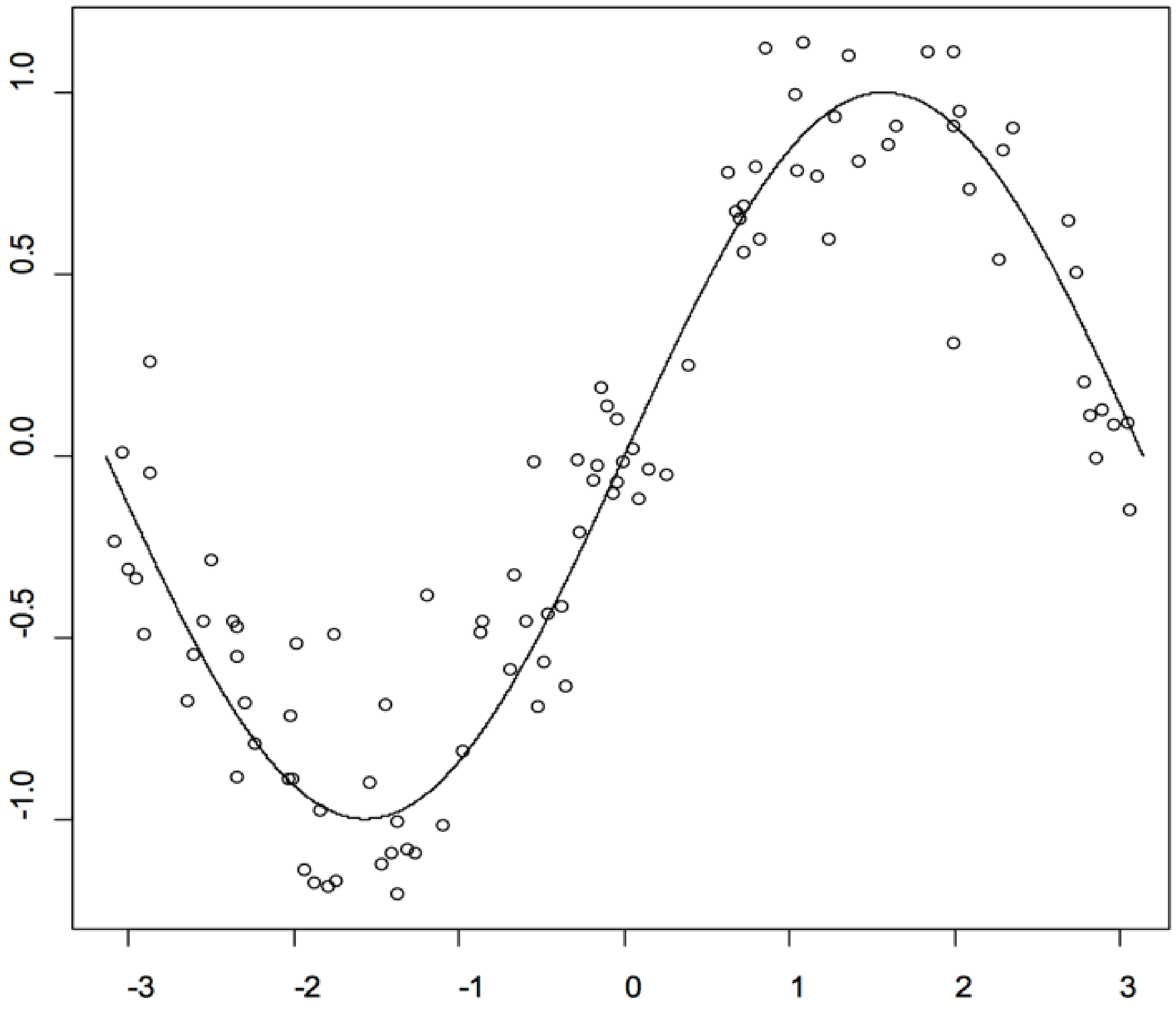
Pioneers:

- Morgan and Sonquist (1963).
- **Breiman, Friedman, Olshen, Stone (1984). *CART***
- Quinlan (1993). *C4.5*

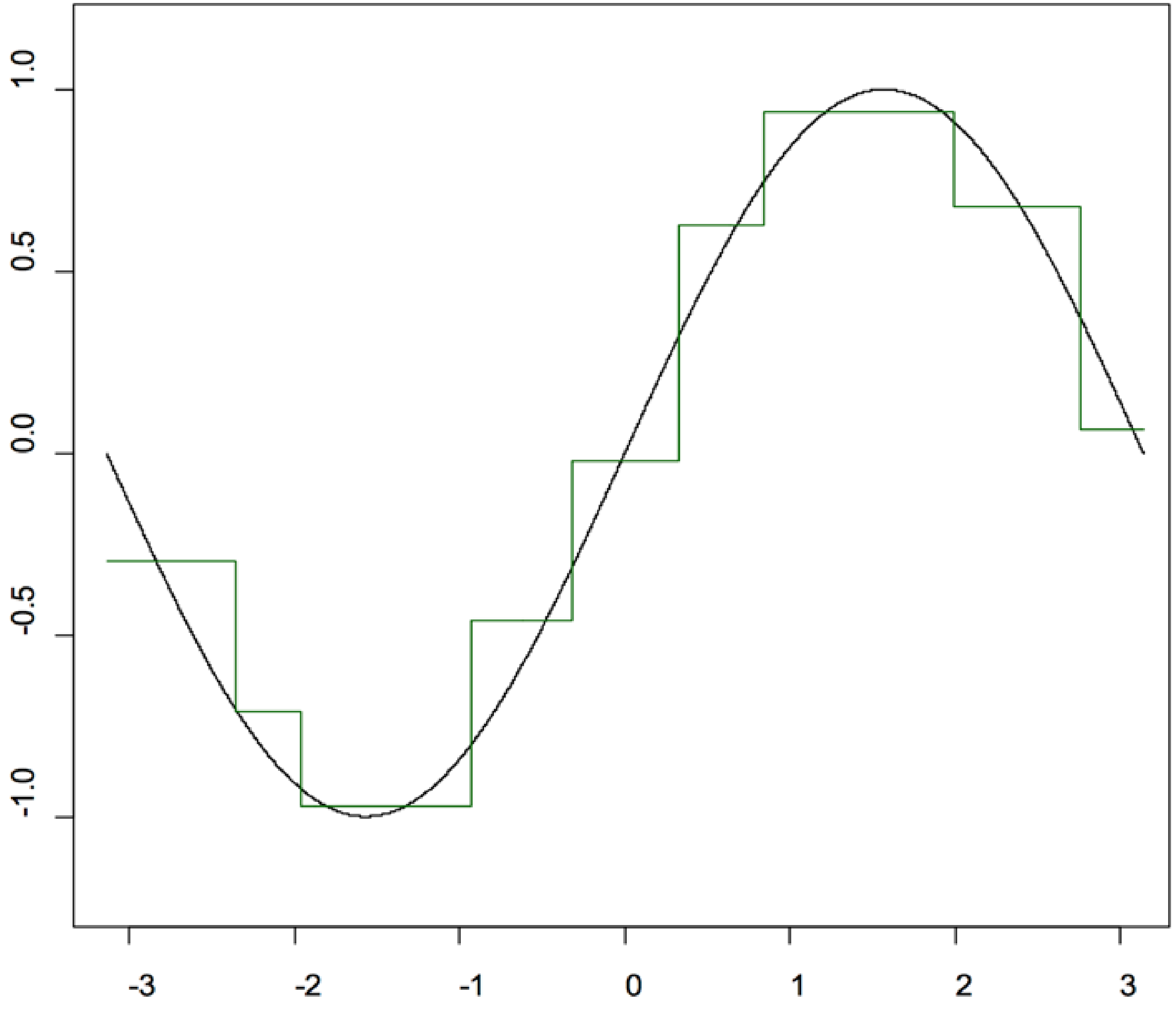
A Classification Tree



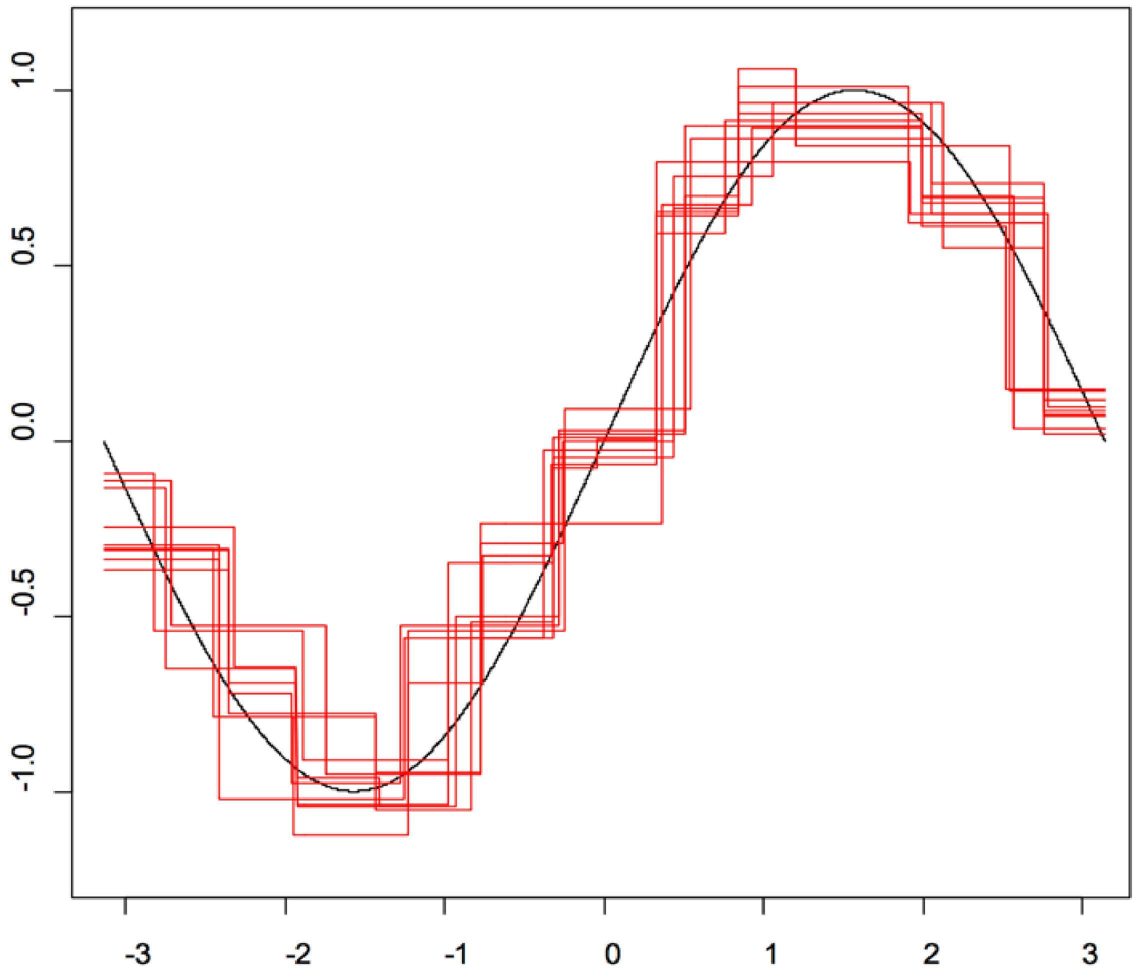
Data and Underlying Function



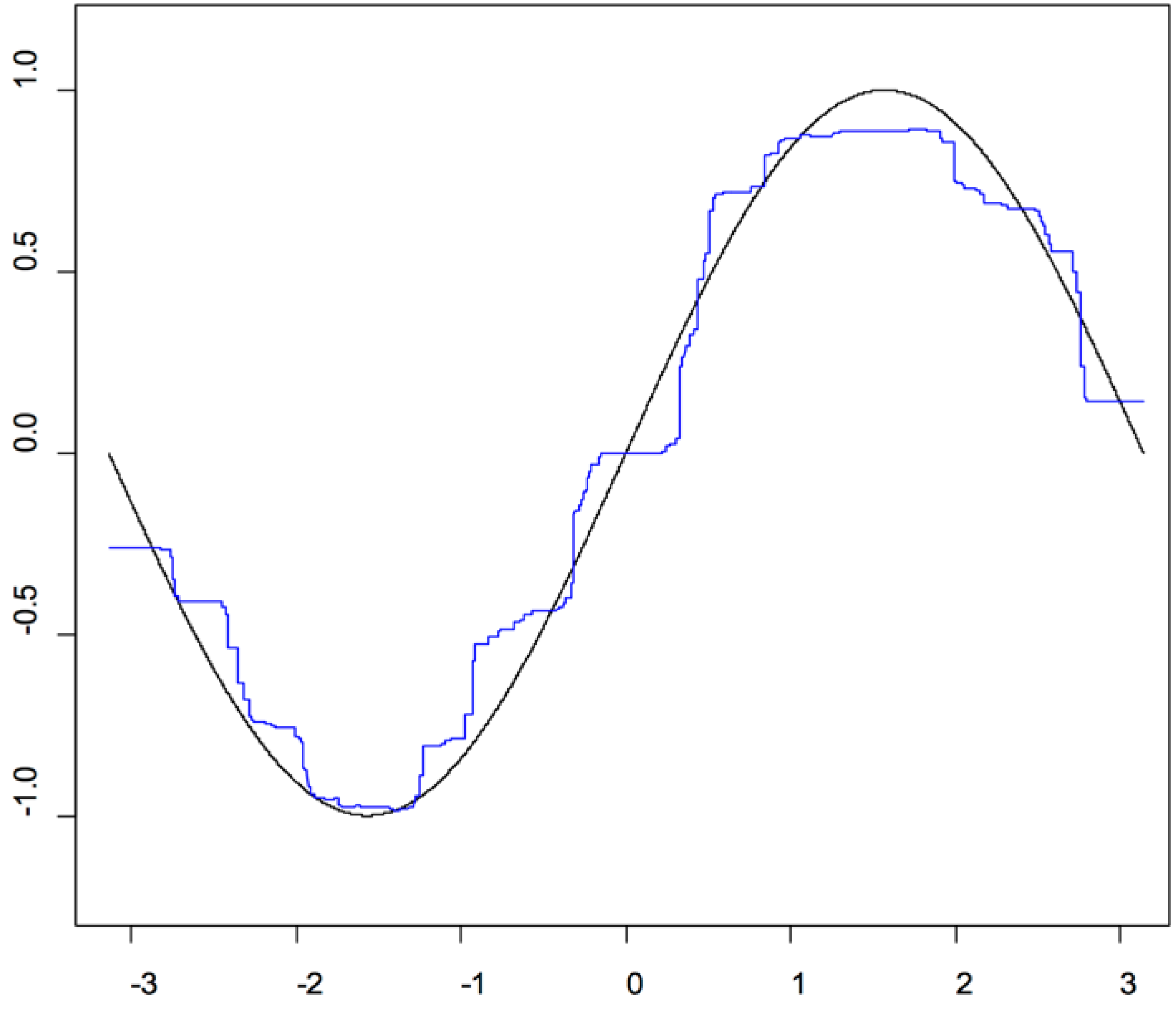
Single Regression Trees



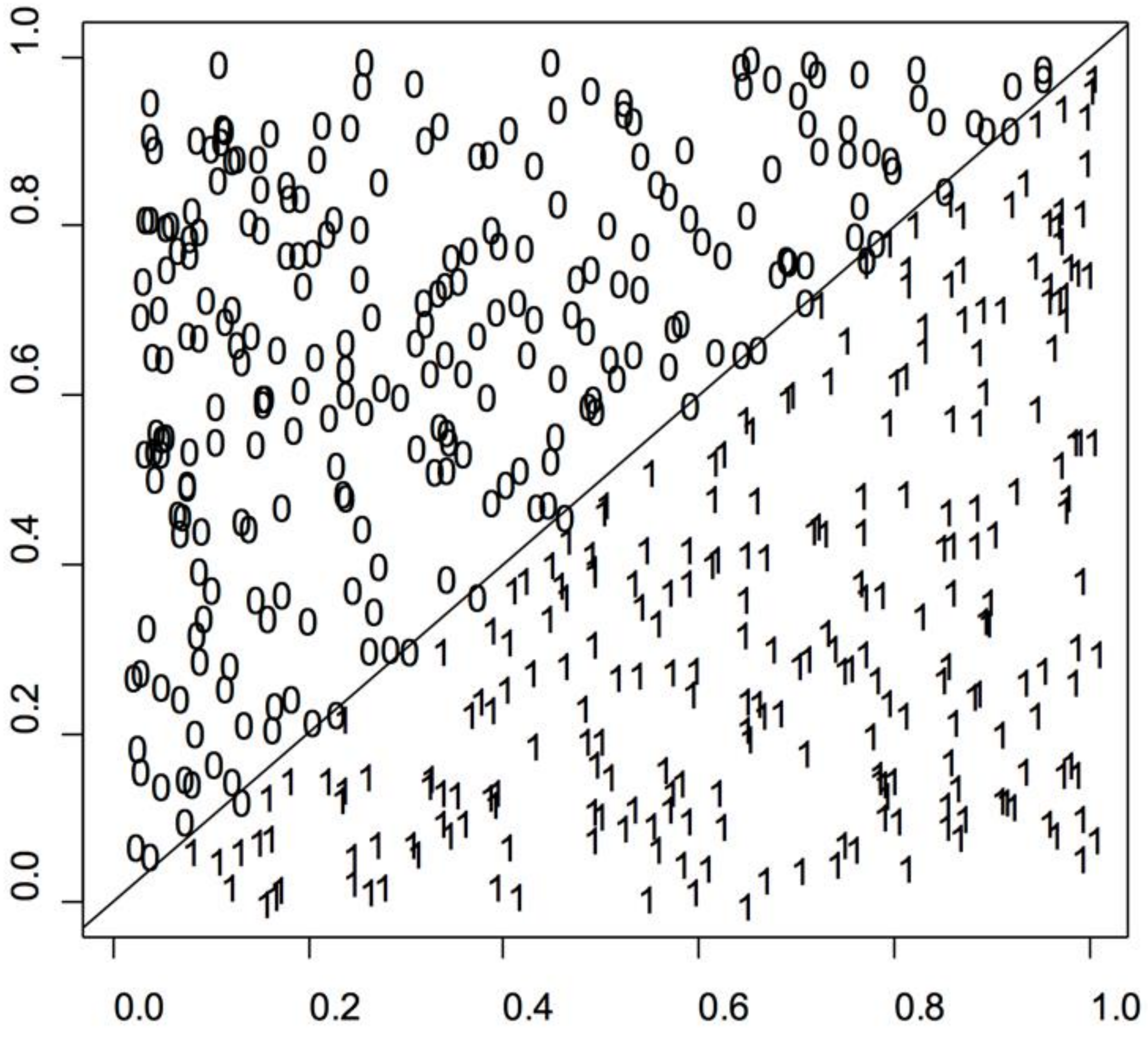
10 Regression Trees from bagging



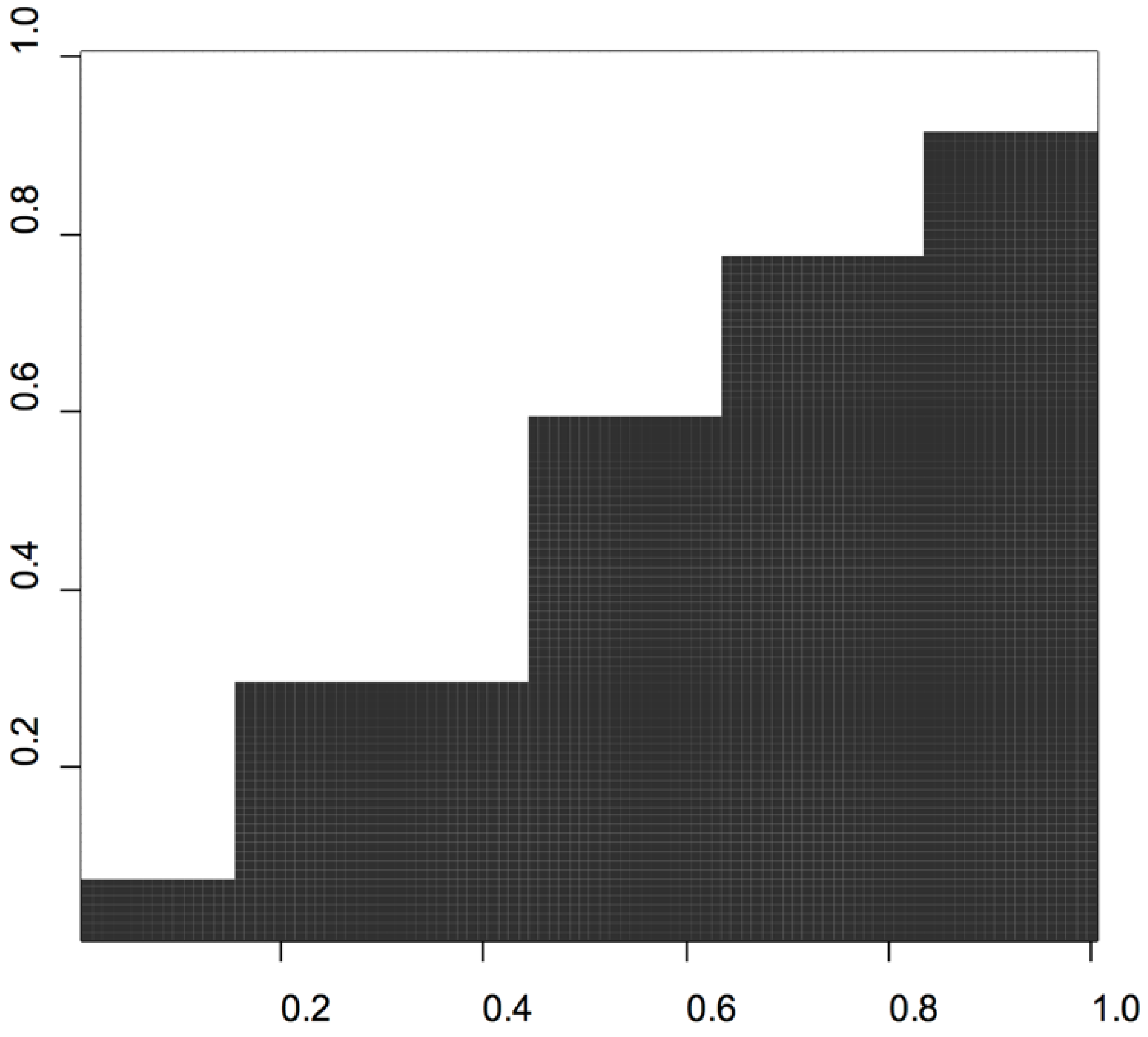
Average of 100 Regression Trees



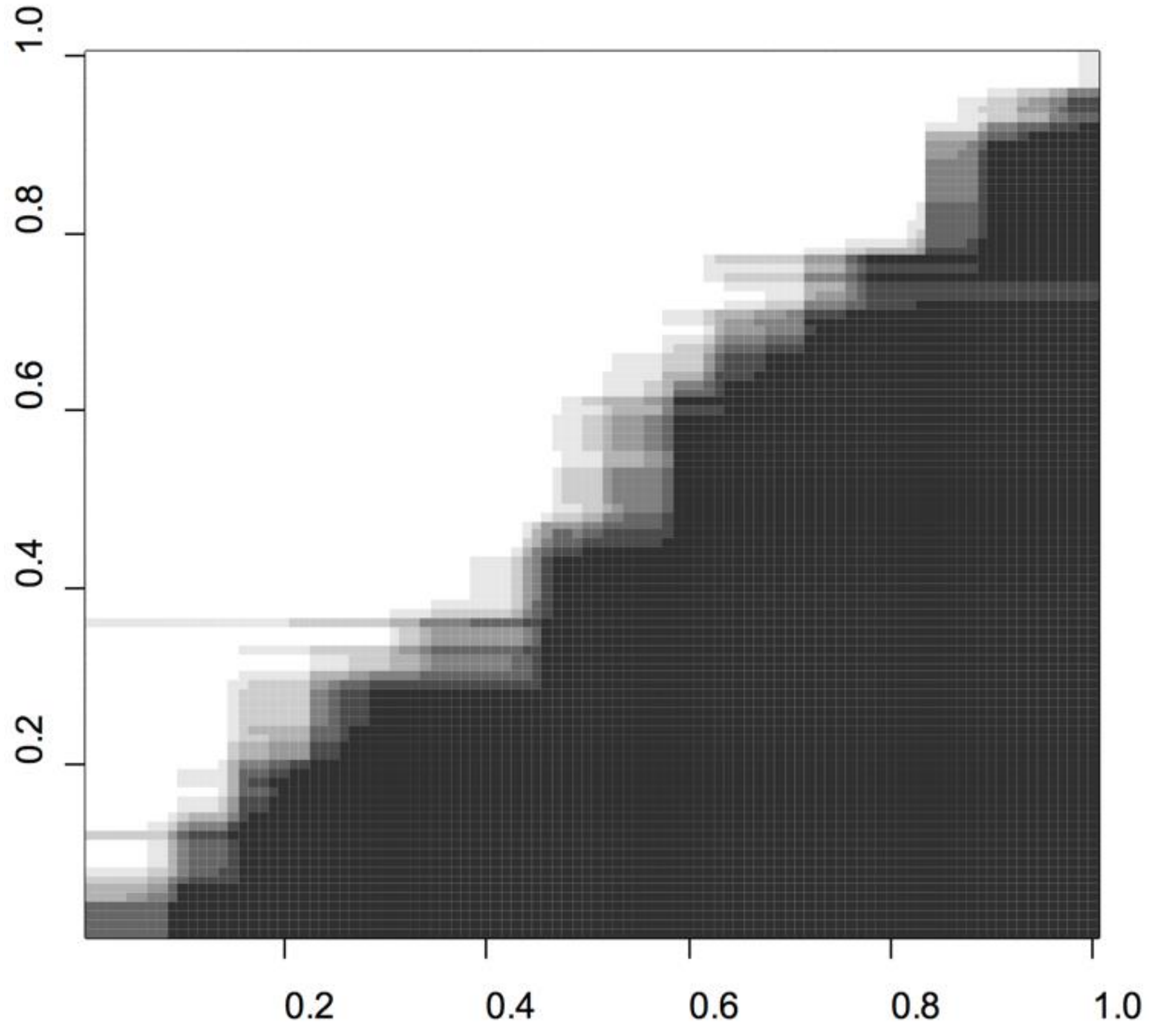
Hard problem for a single tree



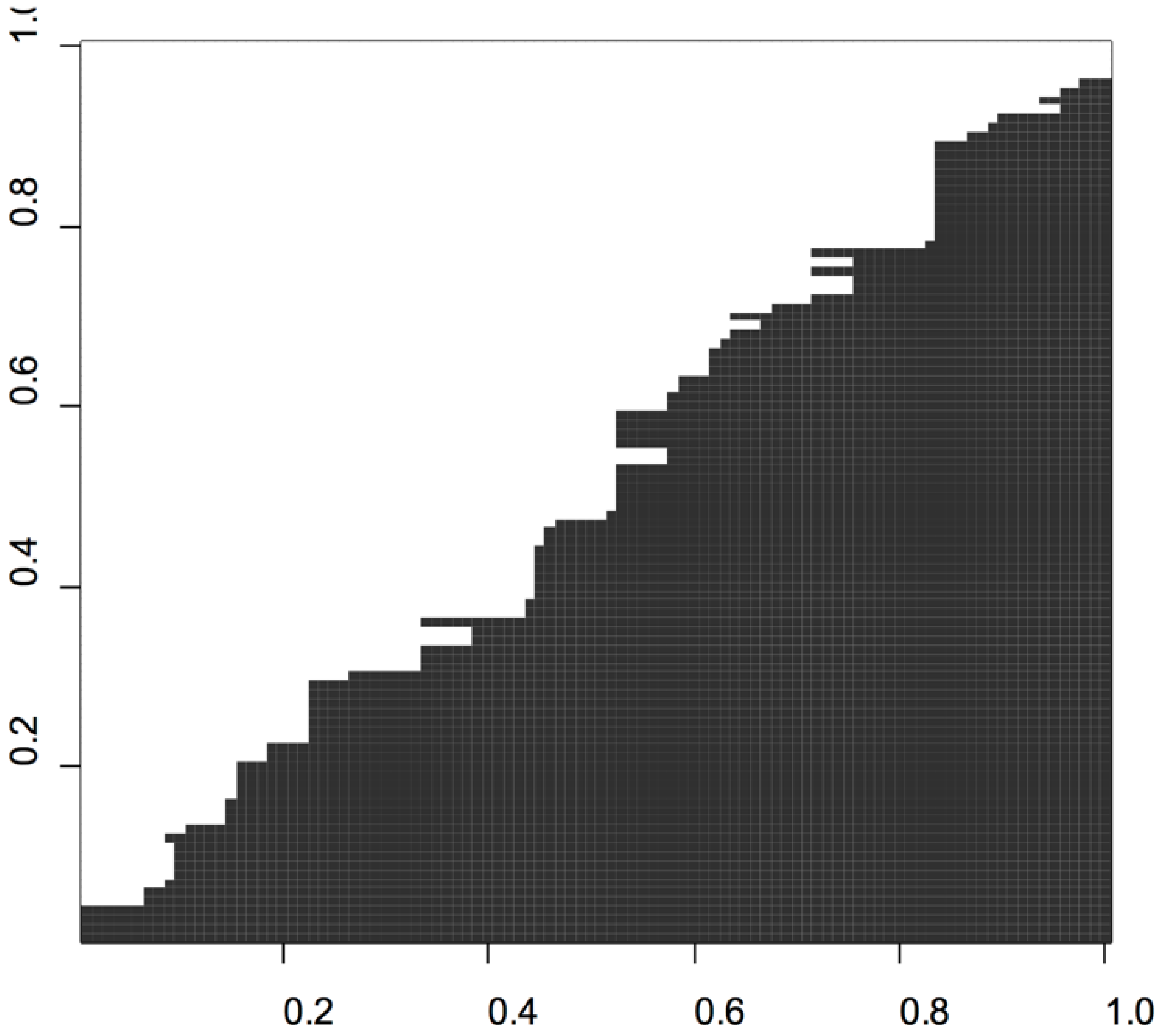
Single Tree



25 Averaged Trees



25 Voted Trees



Random Forests

- Grow a **forest** of many trees.
  - Number of trees could be 10, 50, 500. The more trees, the less overfitted.
- Grow each tree on an independent **bootstrap sample*** from the training data.
- At each node:
  1. Select ***m*** variables **at random** out of all ***M*** possible variables (independently for each node).
  2. Find the best split on the selected ***m*** variables.
- Grow the trees to maximum depth (classification). Vote/average the trees to get predictions for new data.

*Sample N cases at random with replacement.

Random Forests

Inherit many of the advantages of CART:

- Applicable to both regression and classification problems. Yes.
- Handle categorical predictors naturally. Yes.
- Computationally simple and quick to fit, even for large problems. Yes.
- No formal distributional assumptions (non-parametric). Yes.
- Can handle highly non-linear interactions and classification boundaries. Yes.
- Automatic variable selection. Yes. But need variable importance too.
- Handles missing values ~~through surrogate variables~~. Using proximities.
- ~~Very easy to interpret if the tree is small~~. NO! It's not easy to interprete

BIG DATA EXPERIENCE

Random Forests

Improve on CART with respect to:

- *Accuracy* – Random Forests is competitive with the best known machine learning methods.
- *Instability* – if we change the data a little, the individual trees may change but the forest is relatively stable because it is a combination of many trees.

Two natural questions

- ***Why bootstrap? (Why subsample?)***
  - Bootstrapping → out-of-bag data →
    - Estimated error rate and confusion matrix
    - Variable importance
- ***Why trees?***
  - Trees → proximities →
    - Missing value fill-in
    - Outlier detection
    - Illuminating pictures of the data (clusters, structure, outliers)

eXtreme Gradient Boosting

XGBoost

just only improving method that can apply with any algorithms

- stands for **eXtreme Gradient Boosting**
- has become a widely used and really popular tool among Kaggle competitors and Data Scientists in industry
- highly flexible and versatile tool that can work through most regression, classification and ranking problems as well as user-built objective functions

XGBoost

- a scalable and accurate implementation of gradient boosting machines
- It has proven to push the limits of computing power for boosted trees algorithms
  - it was built and developed for the sole purpose of model performance and computational speed

How does it work?

- Concept of boosting: ทำให้ model แย่ๆ นั้นดีขึ้นเรื่อยๆ จากการสร้างโมเดลที่ไม่ดีหลายอัน
    - an ensemble method that seeks to create a strong classifier (model) based on “weak” classifiers
    - **weak** and **strong** refer to a measure of how correlated are the learners to the actual target variable
- **Gradient Boosting = Gradient Descent + Boosting**
- In each state, introduce a weak learner to compensate shortcomings of existing weak learners.
    - **AdaBoost**: shortcomings are identified by high-weight data points
    - **Gradient boosting**: shortcomings are identified by gradients

Gradient Boosting

ทำการ fit residue แล้วเอามาบวกกับ prediction แล้วก็ fit residue ... and so on

Fit model to initial data

It can be:

- same algorithm as for further steps
- or something very simple (like uniform probabilities or average target in regression)

Fit pseudo-residuals

For any function that:

- agrregates the error from examples (e.g. log-loss, RMSE, but not AUC)
- you can calculate gradient on example level (it is called pseudo-residual)

Finalization

Sum up all the models

Credit: Jaroslaw Szymczak, Machine Learning Scientist at OLX Group

how to deal with unbalanced
1) Re sampling
2) Ensemble techniques

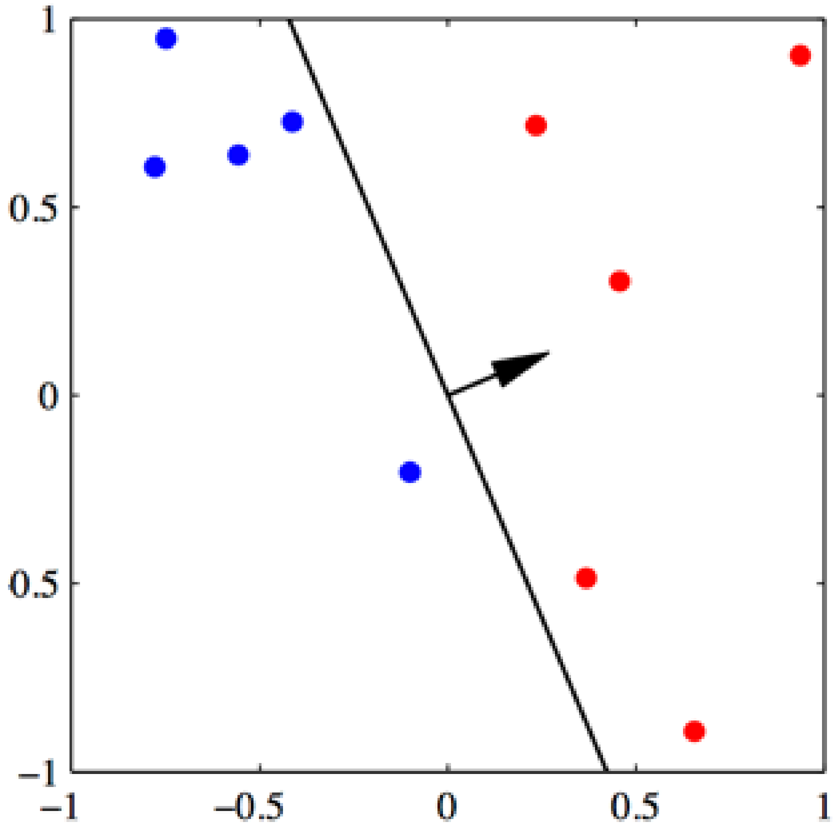
using K-mean clustering and over sampling every group to replicate the data
minimum event rate should be around 20%

Random Forest: Lab

Support Vector Machines

Linear Discriminant Analysis (LDA)

- Focus on linear classification model, i.e., the decision boundary is a linear function of $\boldsymbol{x}$
  - Defined by (D – 1)-dimensional hyperplane
- If the data can be separated exactly by linear decision surfaces, they are called linearly separable
- Implicit assumption: Classes can be modeled well by Gaussians
- Simply speaking, treat classification as a projection problem

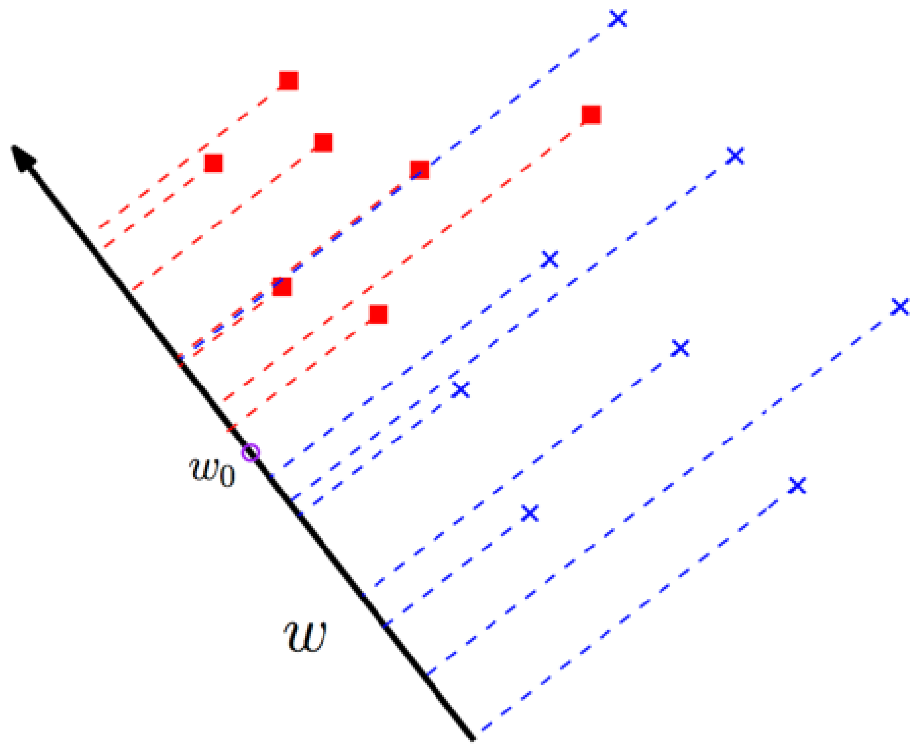


From PRML (Bishop, 2006)

Projection

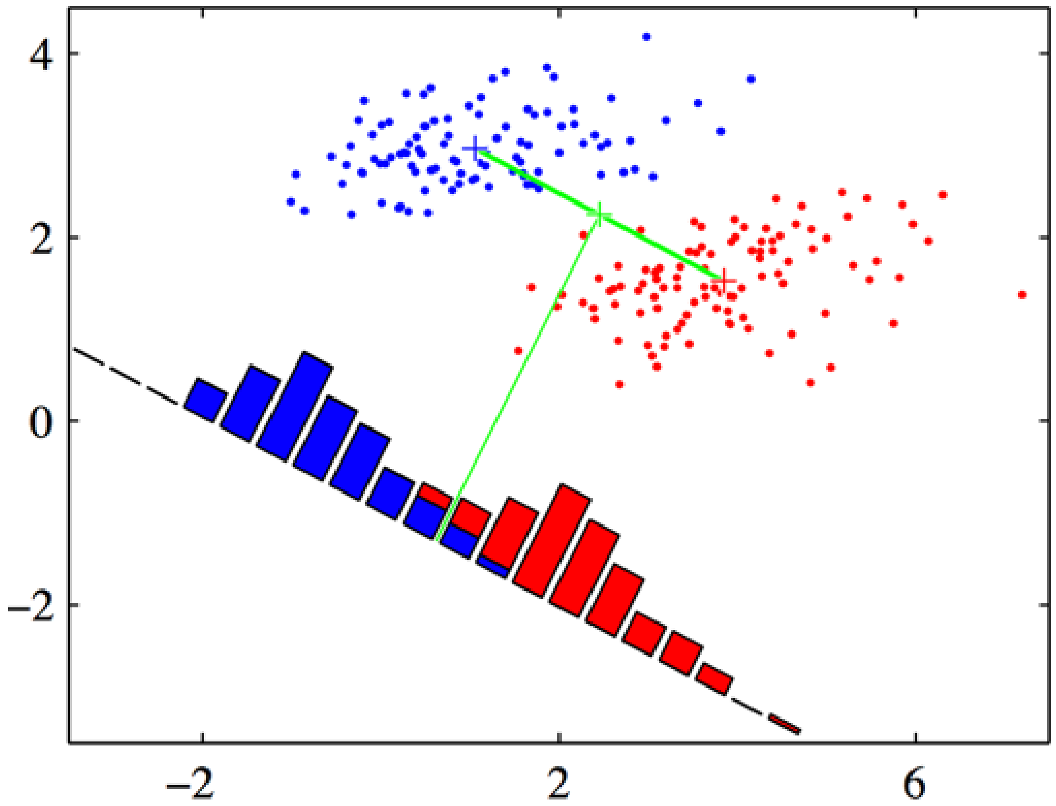
- Assume we know the basic vector ***w***, we can compute the projection, *y*, of any points, *x*.

- Threshold w_0, such that we decide on C_1 if $y \geqq w_0$ and C_2 otherwise.



Potential issues

- Considerable loss of information when projecting
- Even if data was linearly separable, we may lose this separability
- Find good basis vector ***w*** that spans the subspace we project onto



Approach: Maximize Class Separation

- Adjust components of basis vector $\boldsymbol{w}$

  ⏩ Select projection that maximizes the class separation

- Consider two classes: C_1 with N_1 points and C_2 with N_2 points
- Corresponding mean vectors:

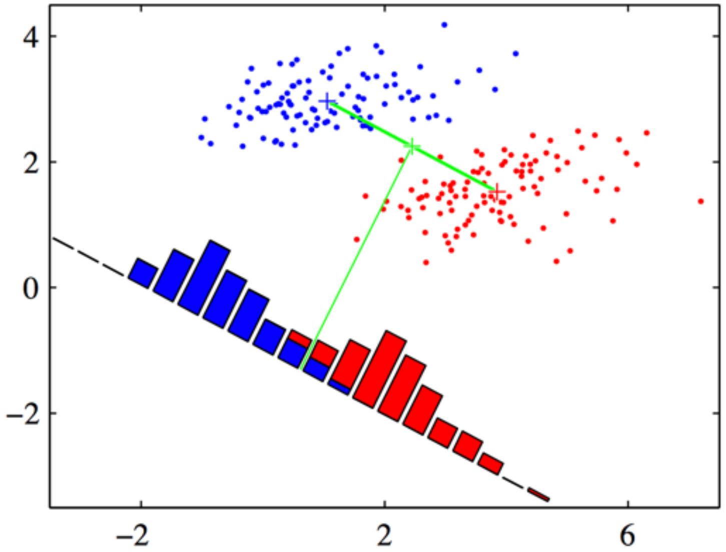
$$\boldsymbol{m}_1 = \frac{1}{N_1} \sum_{n \in C_1} \boldsymbol{x}_n \,, \qquad \boldsymbol{m}_2 = \frac{1}{N_2} \sum_{n \in C_2} \boldsymbol{x}_n$$

- Measure class separation as the distance of the projected class means:

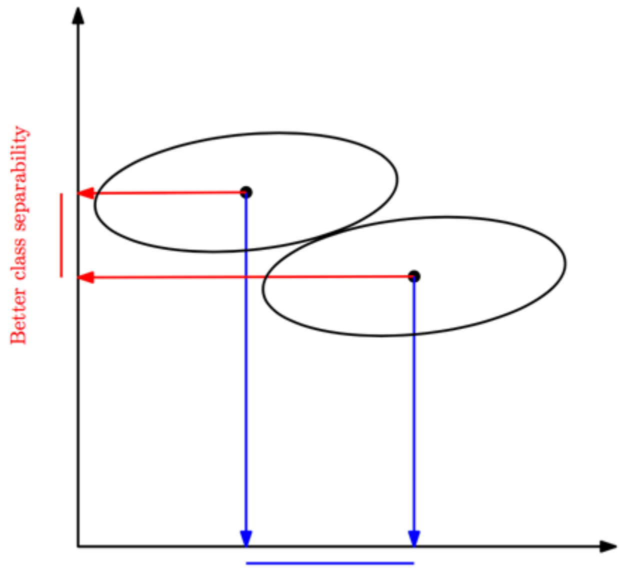
$$m_2 - m_1 = \boldsymbol{w}^\top \boldsymbol{m}_2 - \boldsymbol{w}^\top \boldsymbol{m}_1 = \boldsymbol{w}^\top (\boldsymbol{m}_2 - \boldsymbol{m}_1)$$

and maximize this w.r.t. $\boldsymbol{w}$ with the constraint $\|\boldsymbol{w}\| = 1$

Maximum Class Separation

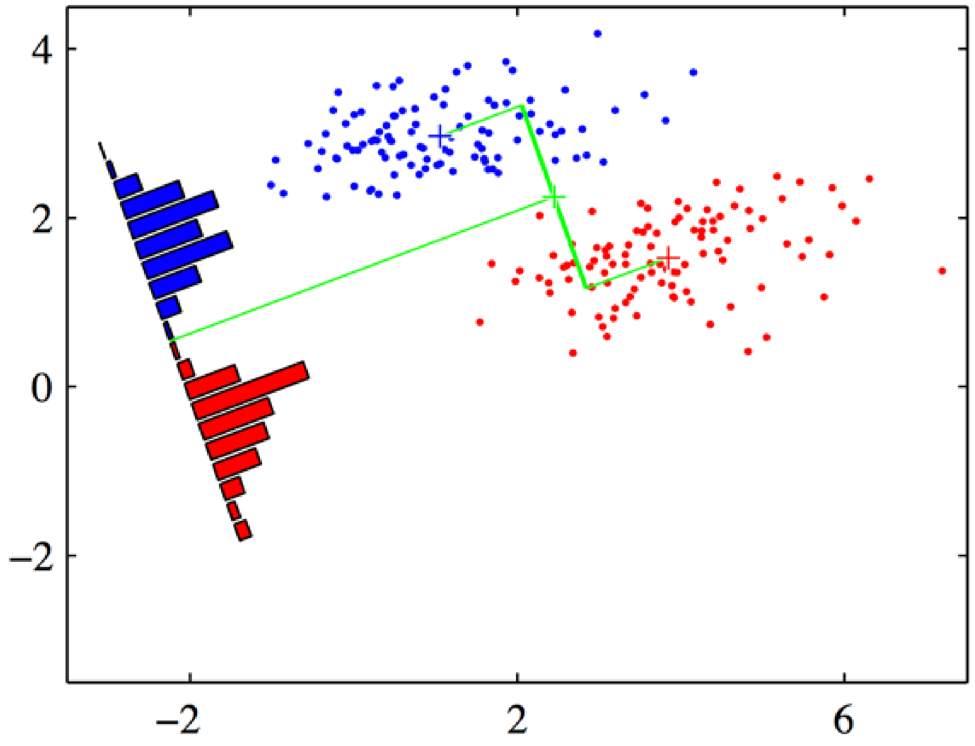


From PRML (Bishop, 2006)



- Find $\mathbf{w} \propto (\mathbf{m}_2 - \mathbf{m}_1)$
- LDA: Large separation of projected class means and small within-class variation (small overlap of classes)

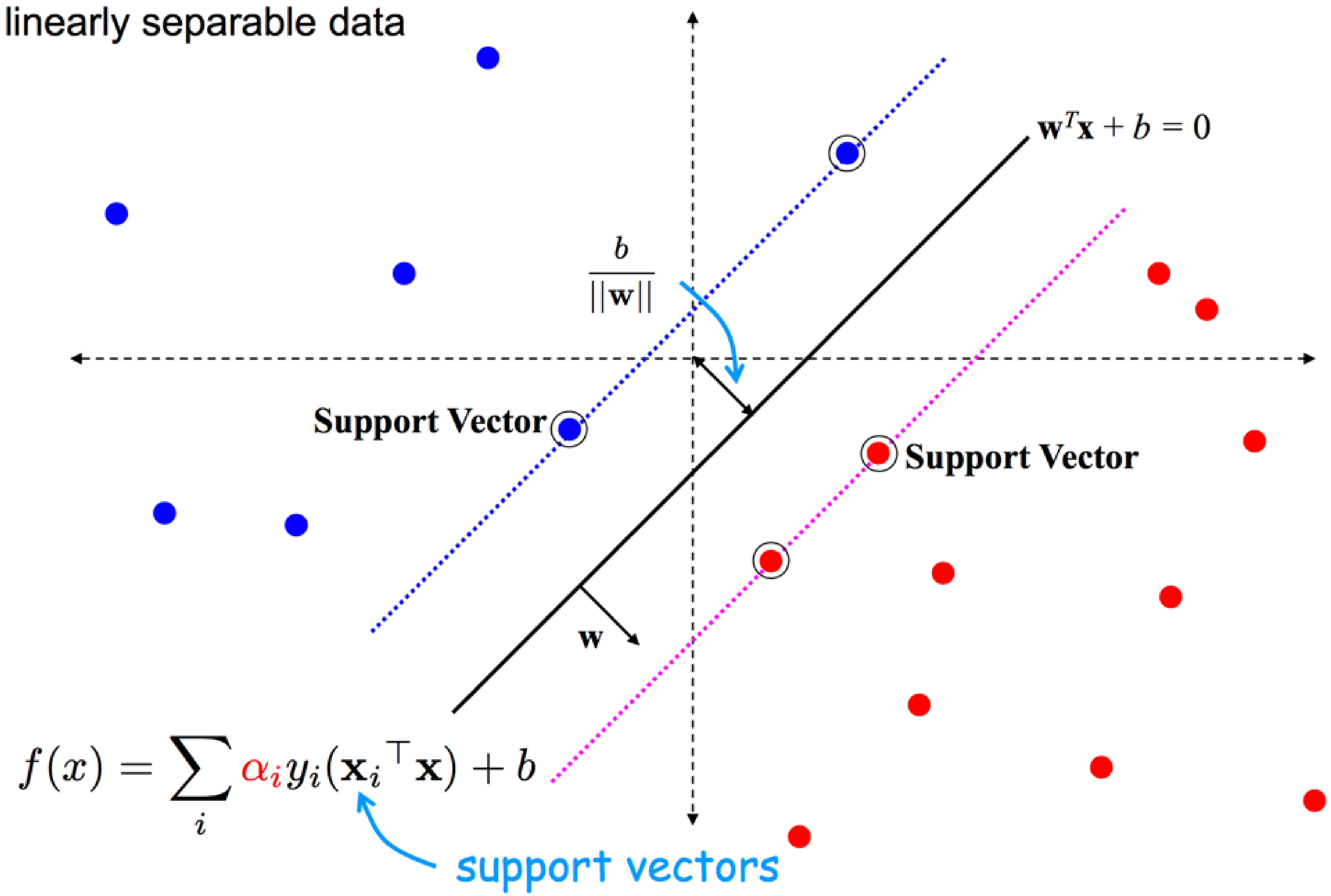
LDA



From PRML (Bishop, 2006)

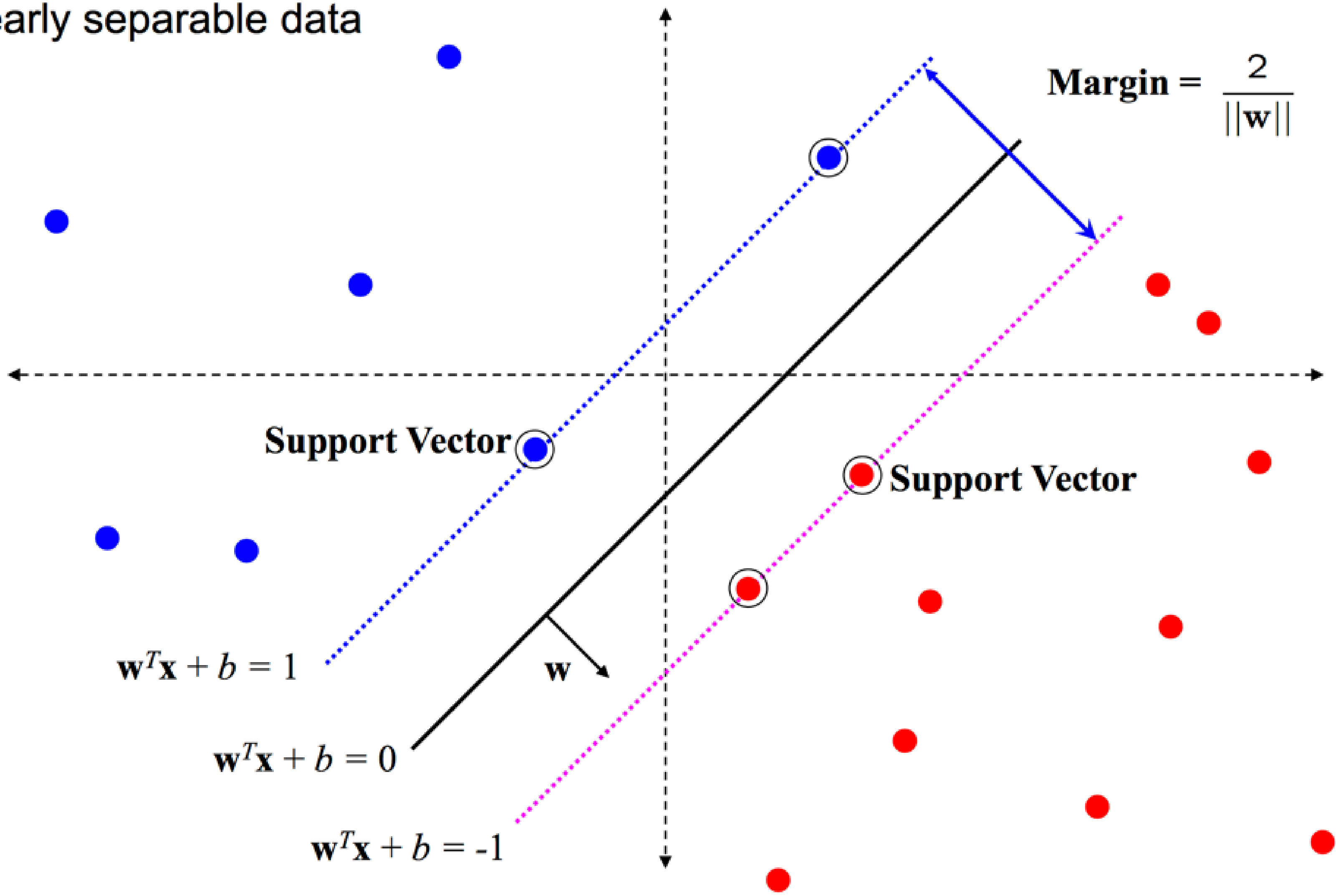
- Separate samples of distinct groups by projecting them onto a space that
  - Maximize their between-class separability while
  - Minimize their within-class variability

SVM

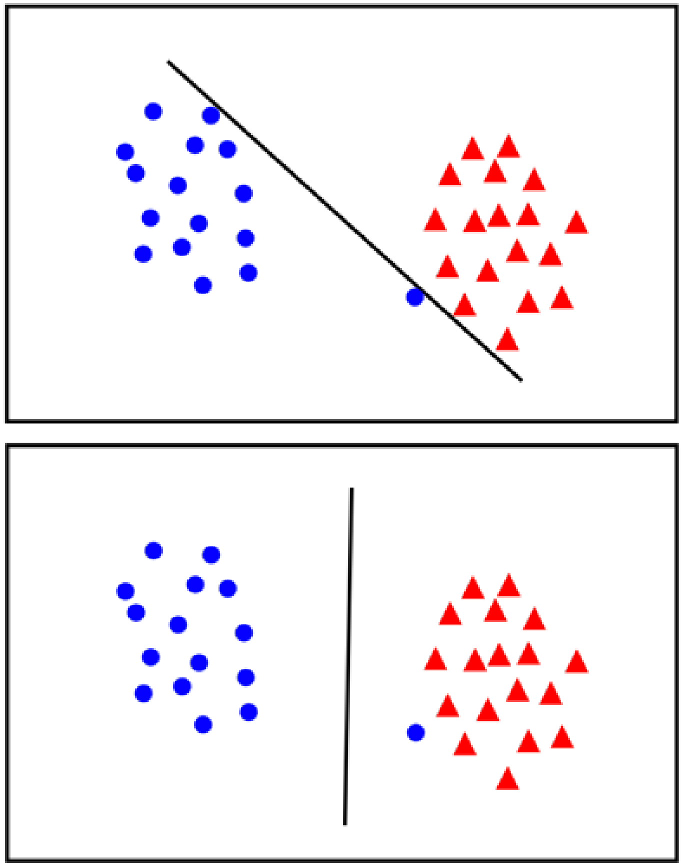

linearly separable data
$\mathbf{w}^T\mathbf{x} + b = 0$
$\frac{b}{||\mathbf{w}||}$
Support Vector
Support Vector
$\mathbf{w}$
$f(x) = \sum_i \alpha_i y_i (\mathbf{x}_i{}^\top \mathbf{x}) + b$
support vectors

SVM Margin

linearly separable data



Linearly separable: What is the best w?



- the points can be linearly separated but there is a very narrow margin

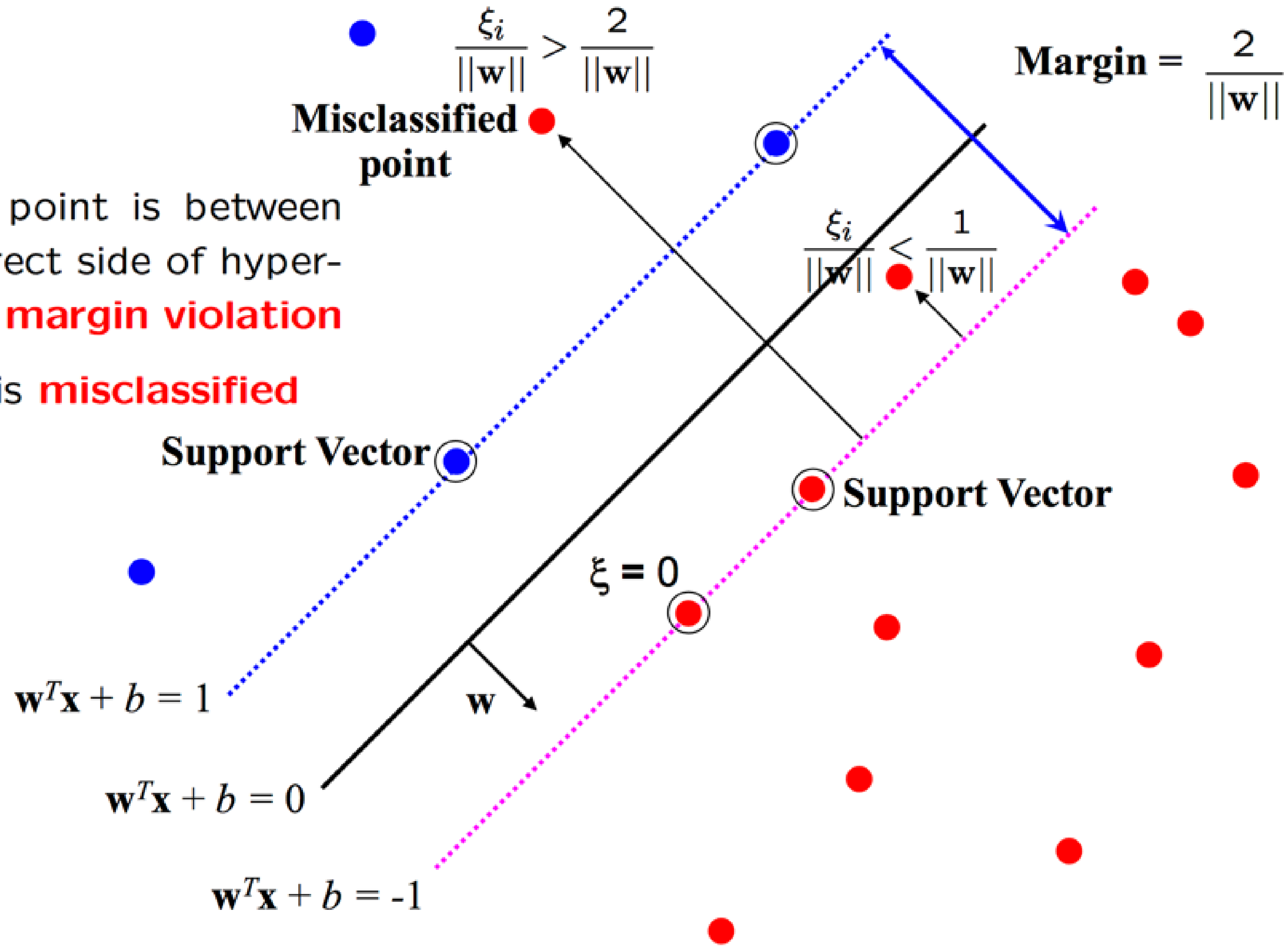
- but possibly the large margin solution is better, even though one constraint is violated

In general there is a trade off between the margin and the number of mistakes on the training data

Error variables

$$\xi_i \geq 0$$

- for $0 < \xi \leq 1$ point is between margin and correct side of hyper-plane. This is a **margin violation**
- for $\xi > 1$ point is **misclassified**



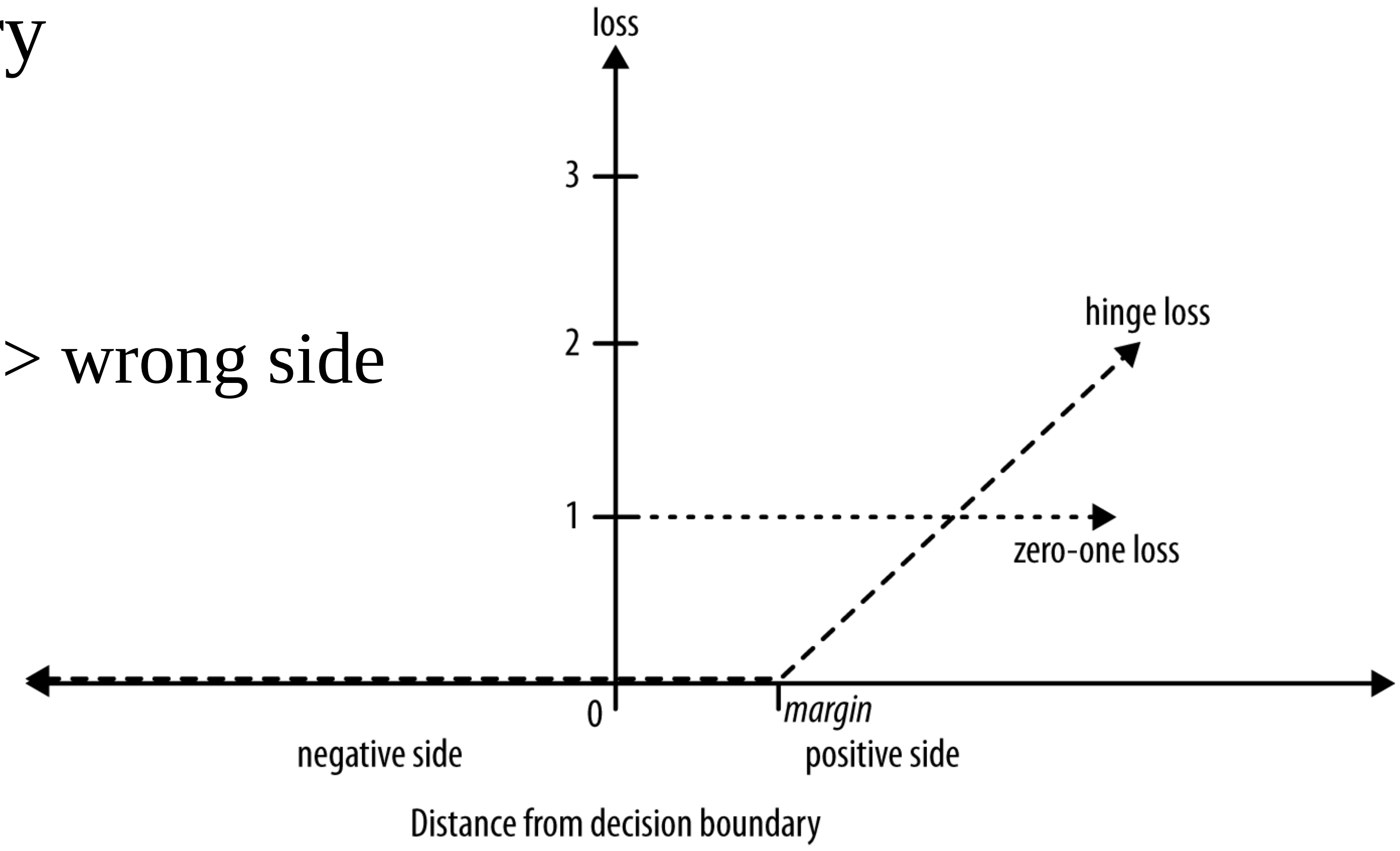
SVM, (2) error handling

- Real world data are not well separated
- In SVM, it simply penalize a training point for being on the wrong side.
- If data are linearly separable no penalty will incur.
- If data are not linearly separable, the best fit is some balance between a fat margin and a low total error penalty

SVM error handling

- The penalty for a misclassified point is proportional to the distance from the margin boundary

Positive => wrong side



Soft margin solution

The optimization problem becomes

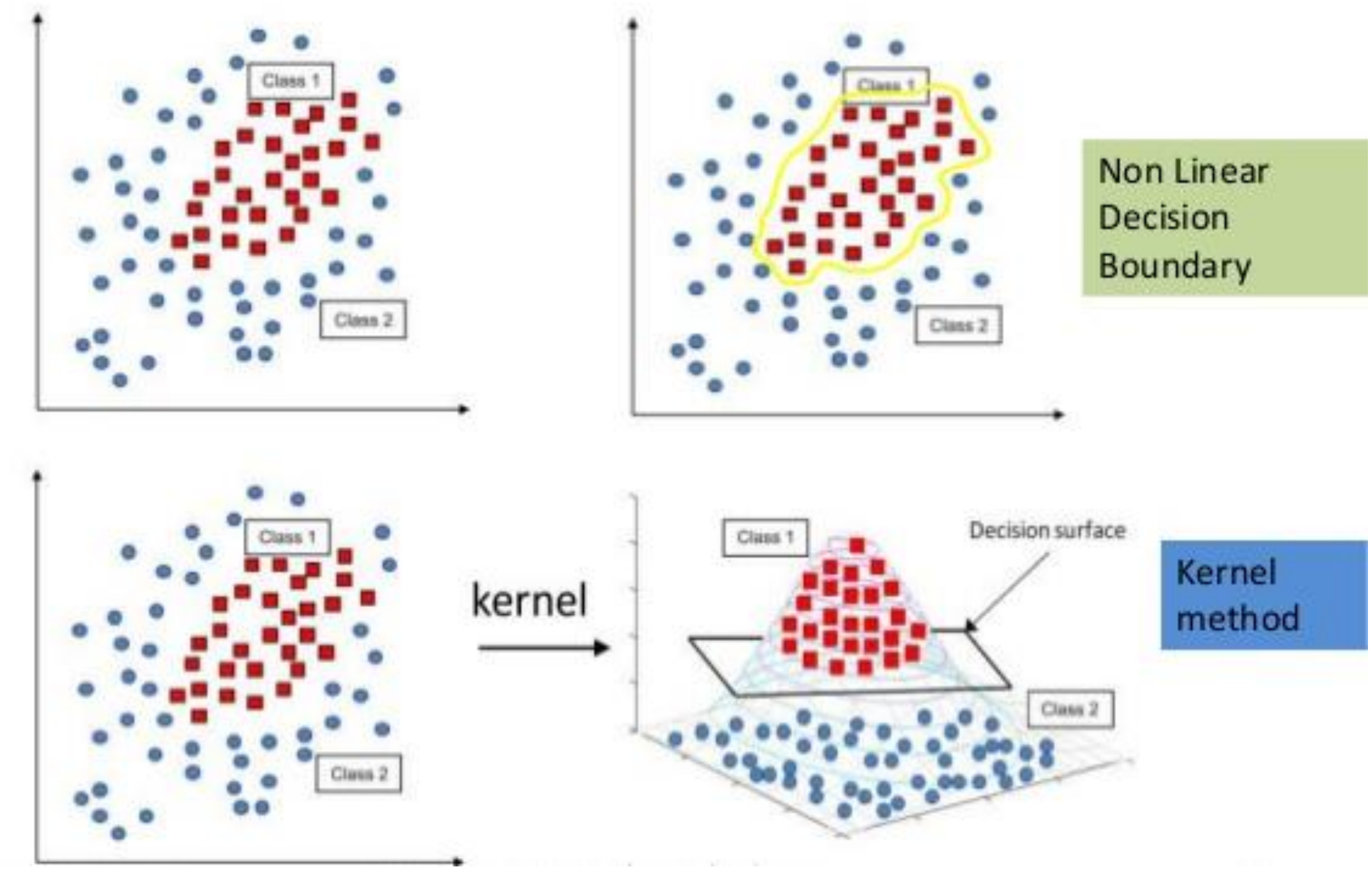
$$\min_{\mathbf{w}\in\mathbb{R}^d, \xi_i \in \mathbb{R}^+} ||\mathbf{w}||^2 + C\sum_i^N \xi_i$$

subject to

$$y_i\left(\mathbf{w}^\top \mathbf{x}_i + b\right) \geq 1 - \xi_i \text{ for } i = 1 \ldots N$$

- Every constraint can be satisfied if ξ_i is sufficiently large
- C is a regularization parameter:
  - small C allows constraints to be easily ignored $\rightarrow$ large margin
  - large C makes constraints hard to ignore $\rightarrow$ narrow margin
  - $C = \infty$ enforces all constraints: hard margin

SVM Kernel


Class 1
Class 2
Class 1
Class 2
Non Linear Decision Boundary
Class 1
Class 2
kernel
Class 1
Decision surface
Class 2
Kernel method

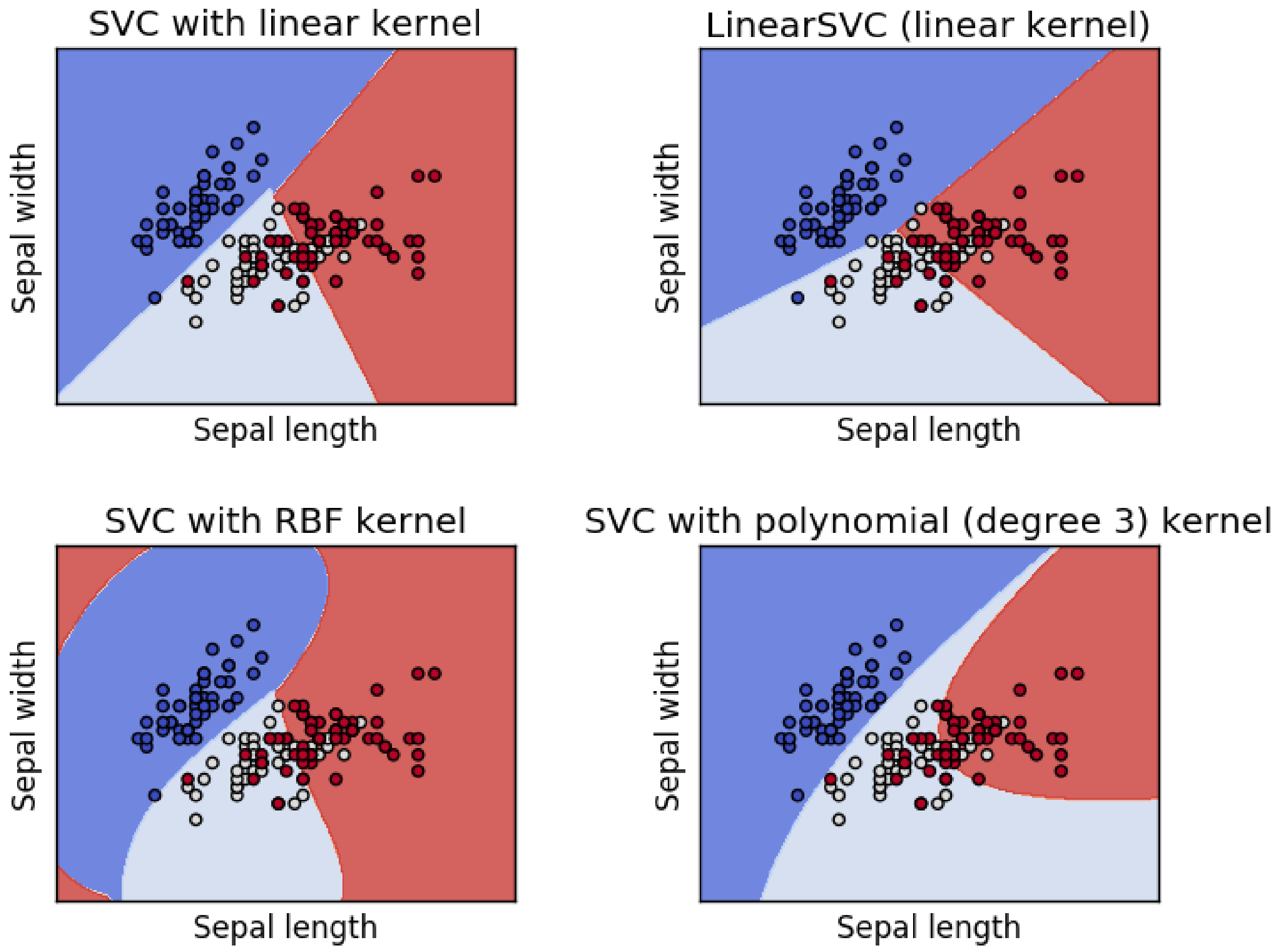
Different types of SVM kernels

linear: $u'v$

polynomial: $(\gamma u'v + coef0)^{degree}$

radial basis: $e^{(}-\gamma|u-v|^2)$

sigmoid: $tanh(\gamma u'v + coef0)$



SVM: Lab

Thank you

Question?