Data Mining In Modern Astronomy Sky Surveys:

Supervised Learning & Astronomy Applications

Ching-Wa Yip

cwyip@pha.jhu.edu; Bloomberg 518

Machine Learning

- We want computers to perform tasks.
- It is difficult for computers to “learn” like the human do.
- We use algorithms:
  - Supervised, e.g.:
    - Classification
    - Regression
  - Unsupervised, e.g.:
    - Density Estimation
    - Clustering
    - Dimension Reduction

Machine Learning

- We want computers to perform tasks.
- It is difficult for computers to “learn” like the human do.
- We use algorithms:
  - Supervised, e.g.:
    - Classification
    - Regression
  - Unsupervised, e.g.:
    - Density Estimation
    - Clustering
    - Dimension Reduction

Unsupervised vs. Supervised Learning

- Unsupervised:
  - Given data $\{\boldsymbol{x}^1, \boldsymbol{x}^2, \boldsymbol{x}^3, \cdots, \boldsymbol{x}^n\}$ find patterns.
  - The description of a pattern may come in the form of a function (say, $g(\boldsymbol{x})$).
- Supervised:
  - Given data $\{(\boldsymbol{x}^1, \boldsymbol{y}^1), (\boldsymbol{x}^2, \boldsymbol{y}^2), (\boldsymbol{x}^3, \boldsymbol{y}^3), \cdots, (\boldsymbol{x}^n, \boldsymbol{y}^n)\}$ find a function such that $f(\boldsymbol{x}) = \boldsymbol{y}$.
  - $\boldsymbol{y}$ are the labels.

Types of Label

- Class
  - Binary: 0, 1
  - Galaxy types: E, S, Sa, … etc.
- Physical Quantities
  - E.g. Redshift of a galaxy

Basic Concepts in Machine Learning

- Label and Unlabeled data
- Datasets: training set and test set
- Feature space
- Distance between points
- Cost Function (or called Error Function)
- Shape of the data distribution
- Outliers

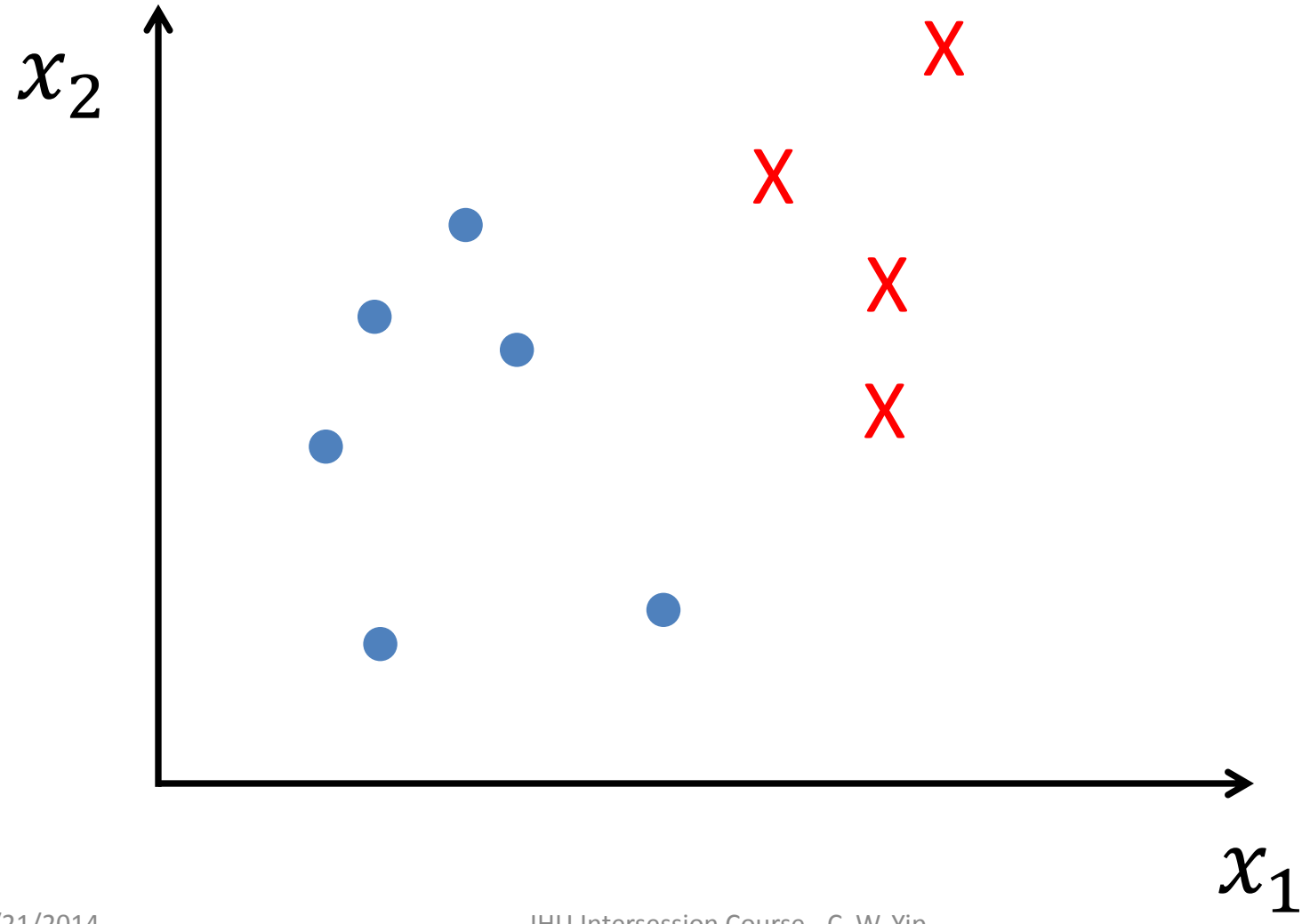
Training Set & Test Set

- Training Set
  - Data that are used to build the model.
- Validation Set
  - Data that used to evaluate the model.
  - Data were *not used in training*.
  - (Sometimes omitted.)
- Test Set
  - Similar to Validation Set but for the final model.

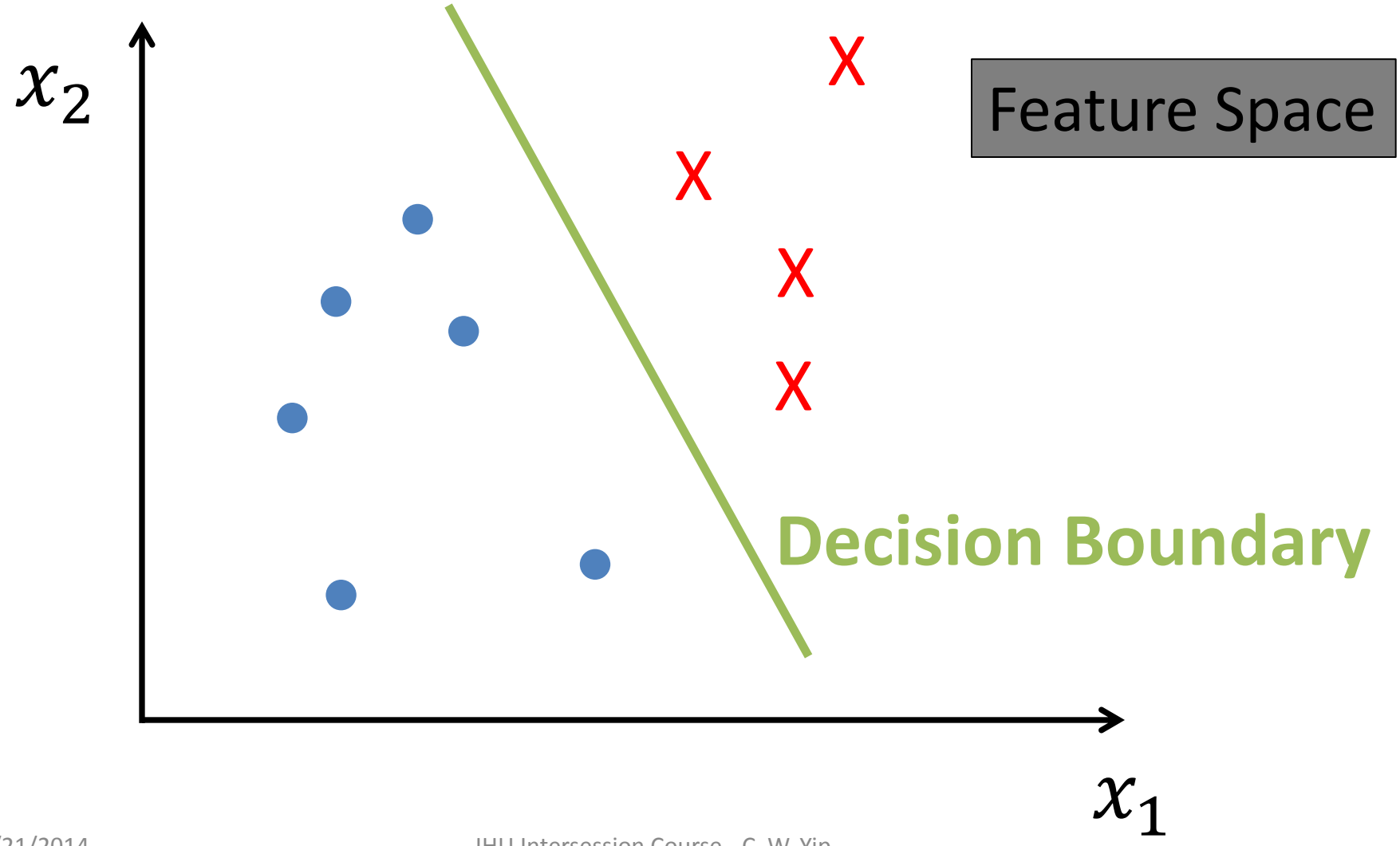
Aspects in Supervised Learning

- What are the popular algorithms?
- How to minimize cost function (numerically)?
  - Where a model is already given.
- How to select the appropriate model?
  - Where we have many candidate models.

Supervised Learning: Find Decision Boundary in Labeled Data



Supervised Learning: Find Decision Boundary in Labeled Data



Algorithms for Supervised Learning: Classification and Regression

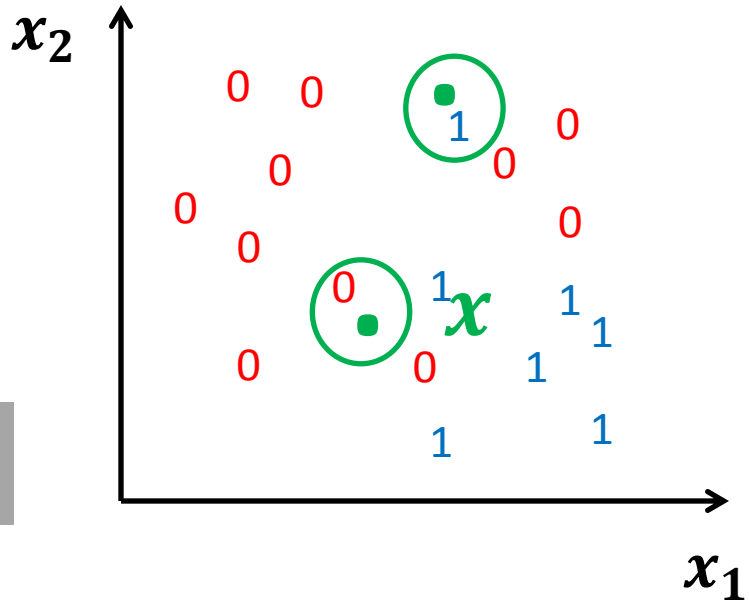
- Principal Component Analysis
- KNN
- Support Vector Machine
- Decision Tree
- Regression Tree
- Random Forest
- Etc.

K Nearest Neighbor (KNN)

- One of the simplest supervised algorithms.
- Idea: use k-neighbors to estimate $\boldsymbol{y}$ given $\boldsymbol{x}$.
- The value of k can be determined (see “Model Selection”).

KNN

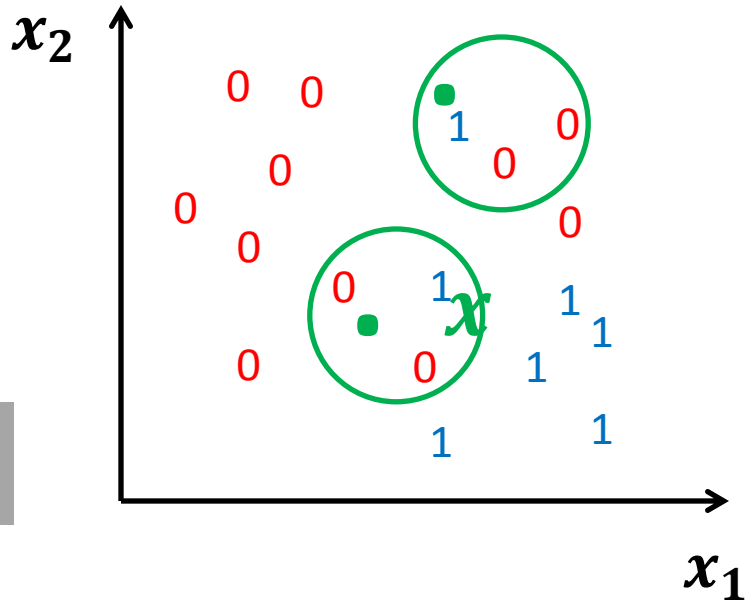
- Given $\{(\boldsymbol{x}^1, \boldsymbol{y}^1), (\boldsymbol{x}^2, \boldsymbol{y}^2), (\boldsymbol{x}^3, \boldsymbol{y}^3), \cdots, (\boldsymbol{x}^n, \boldsymbol{y}^n)\}$, find $\boldsymbol{y}$ for a given $\boldsymbol{x}$.
- E.g., $\boldsymbol{x}$ has two components ($\boldsymbol{x} \in R^2$), $\boldsymbol{y}$ is 0 or 1.
- Estimated Class:
  - From the majority vote of k nearest neighbors



k = 1

KNN

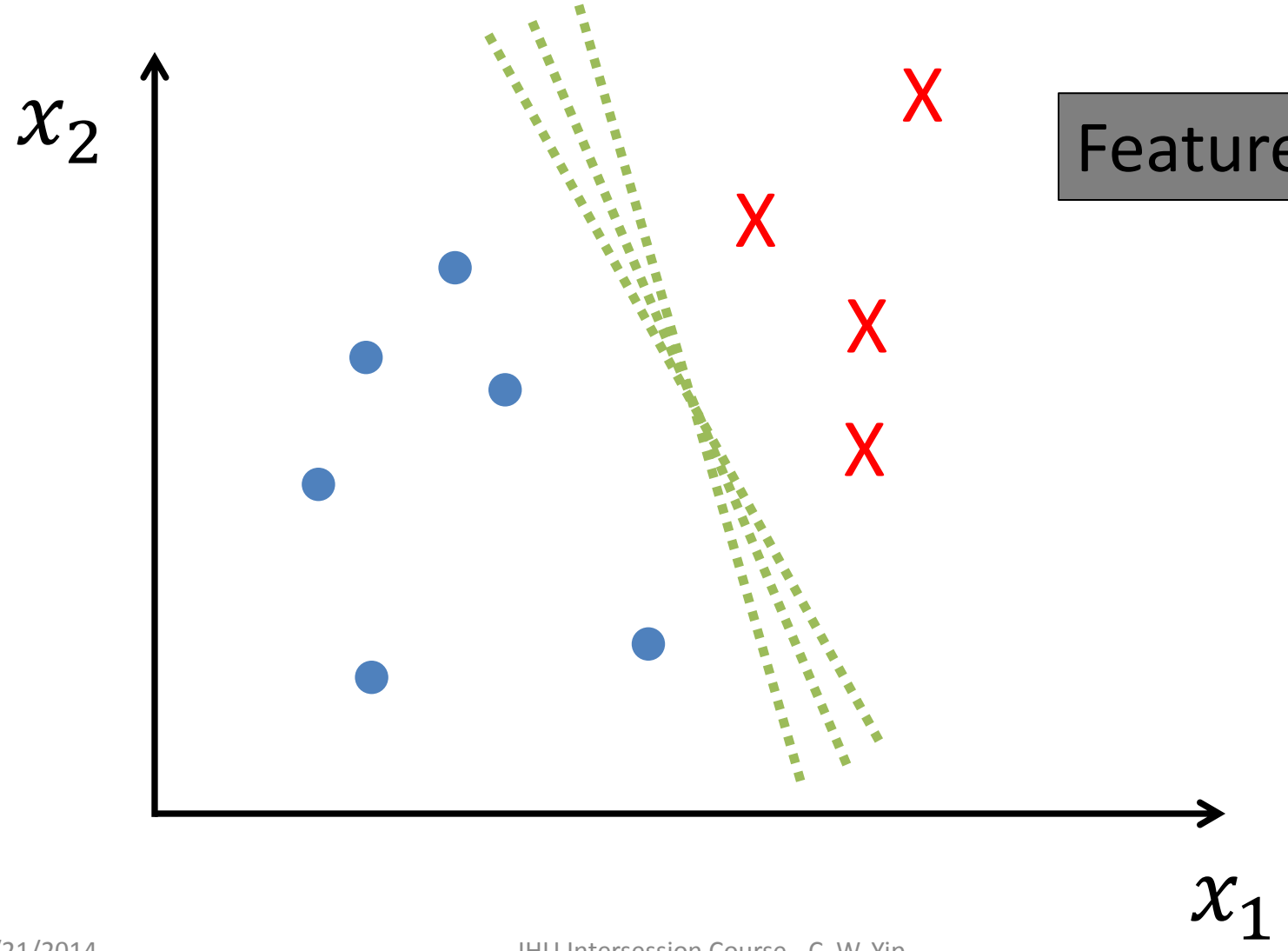
- Given $\{(\boldsymbol{x}^1, \boldsymbol{y}^1), (\boldsymbol{x}^2, \boldsymbol{y}^2), (\boldsymbol{x}^3, \boldsymbol{y}^3), \cdots, (\boldsymbol{x}^n, \boldsymbol{y}^n)\}$, find $\boldsymbol{y}$ for a given $\boldsymbol{x}$.
- E.g., $\boldsymbol{x}$ has two components ($\boldsymbol{x} \in R^2$), $\boldsymbol{y}$ is 0 or 1.
- Estimated Class:
  - From the majority vote of k nearest neighbors



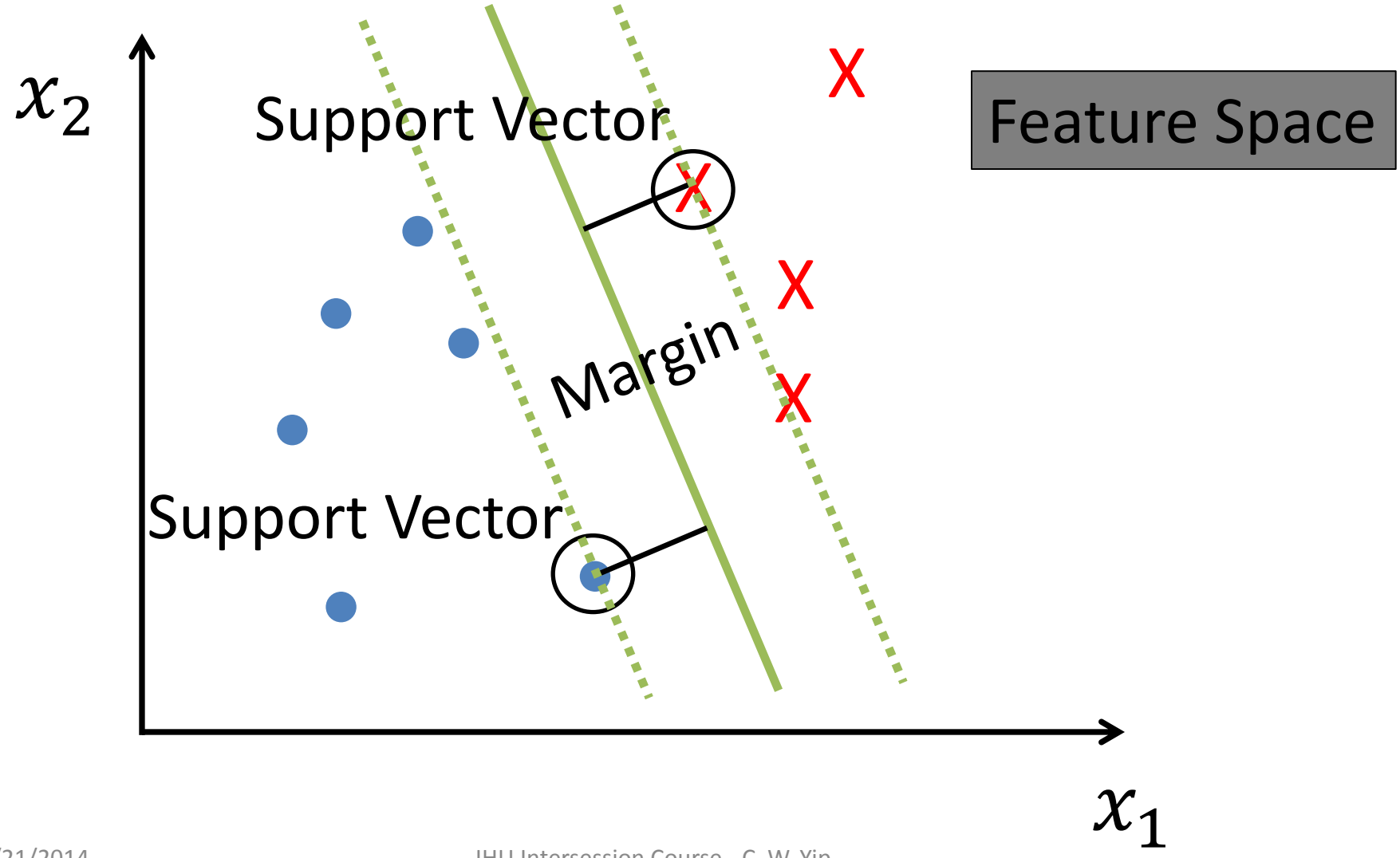
Support Vector Machine (SVM): Finding Hyperplanes in the Feature Space

- Map data into higher dimensional feature space.
- The decision boundaries, or the hyperplanes, separate the feature into classes.
  - 1D data: a point
  - 2D data: a line
  - Higher-dimensional data: a hyperplane
- More than one hyperplane can do the job.
- Support vectors are data points located closest to the hyperplanes.
  - They are the most difficult to classify.
- SVM chooses the hyperplanes which maximize the margin of the support vectors.

Many Hyperplanes can Separate the Data

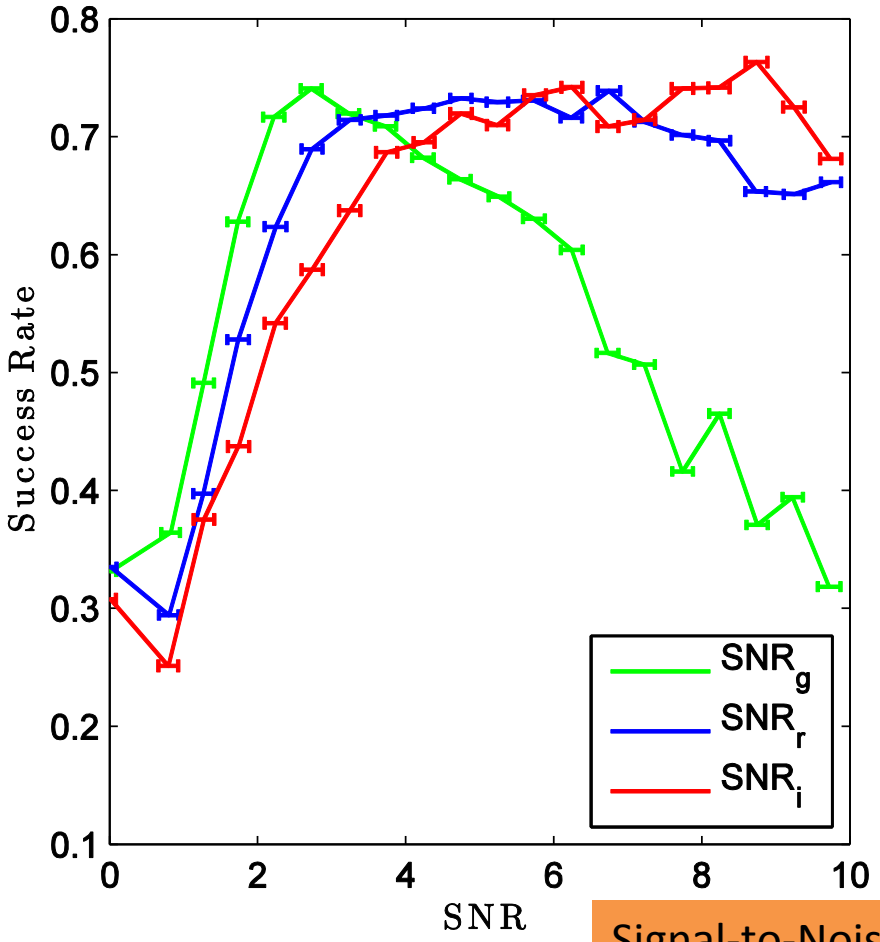


SVM finds the Hyperplane which Maximize the Margin (Perpendicular to Hyperplane)

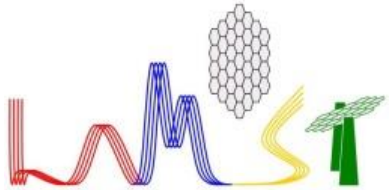


SVM in Galaxy Classification

$$\frac{\text{Correct Classification Count}}{\text{Total Data Count}}$$

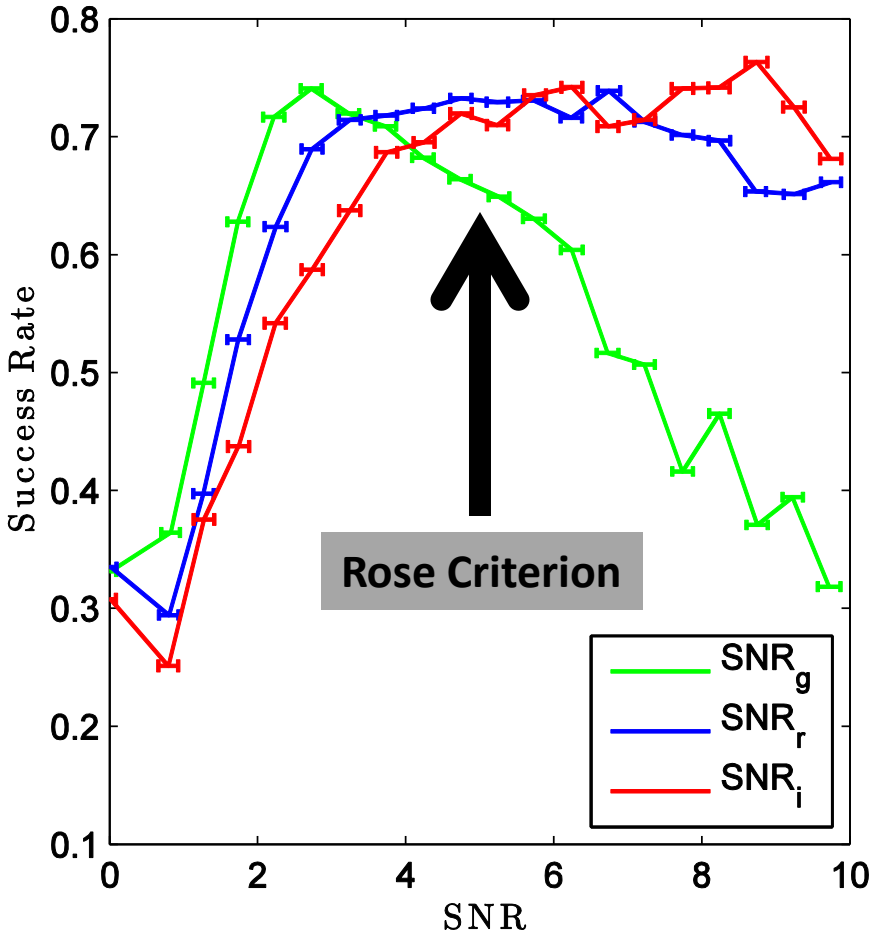


Signal-to-Noise Ratio of Data



(Classification of LAMOST galaxy spectra, Haijun Tian, 2014 in prep.)

SVM in Galaxy Classification



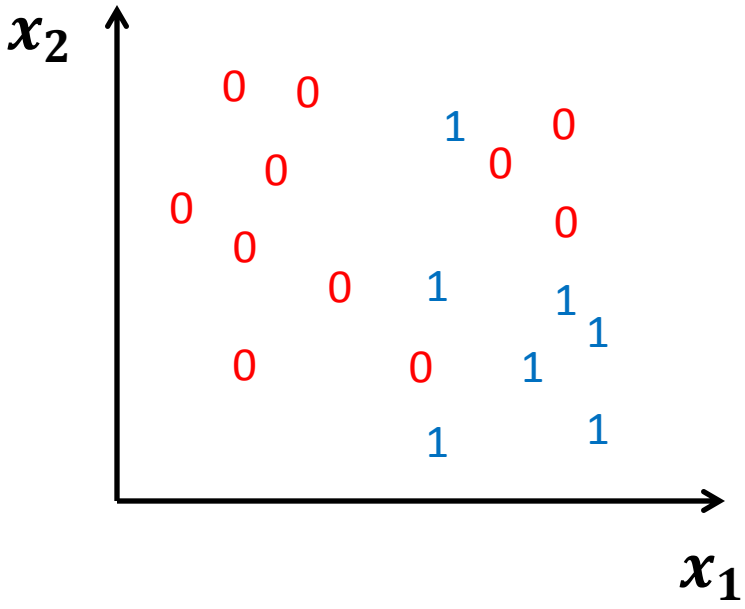
(Classification of LAMOST galaxy spectra, Haijun Tian, 2014 in prep.)

Decision Tree

- A decision tree partitions the data by features.
- A decision tree has
  - Root Node: top most decision node
  - Decision Nodes: has two or more branches (“YES” or “NO”; “x > 0.8” or “x = 0.8” or “x < 0.8”).
  - Left Node: where we make decision (classification or regression).

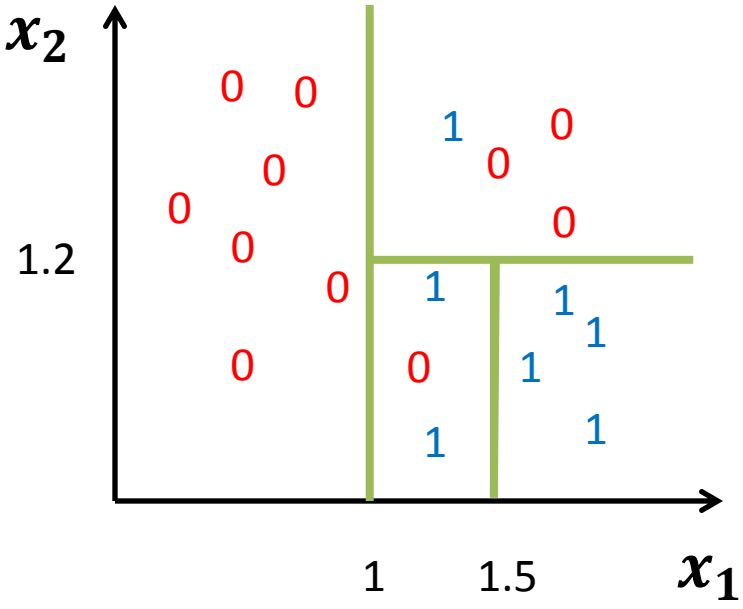
Decision Tree

- Given $\{(\boldsymbol{x}^1, \boldsymbol{y}^1), (\boldsymbol{x}^2, \boldsymbol{y}^2), (\boldsymbol{x}^3, \boldsymbol{y}^3), \cdots, (\boldsymbol{x}^n, \boldsymbol{y}^n)\}$, find $\boldsymbol{y}$ for a given $\boldsymbol{x}$.
- E.g., $\boldsymbol{x}$ has two components, $\boldsymbol{y}$ is 0 or 1.



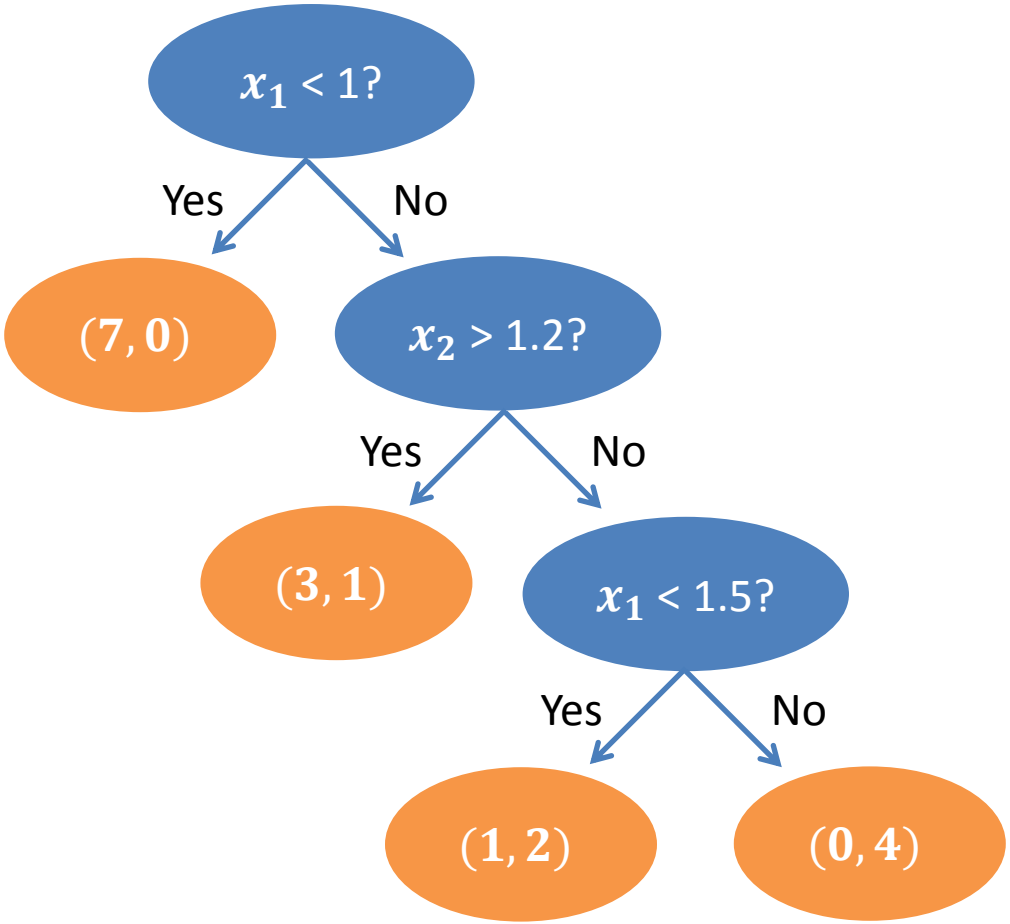
Decision Tree

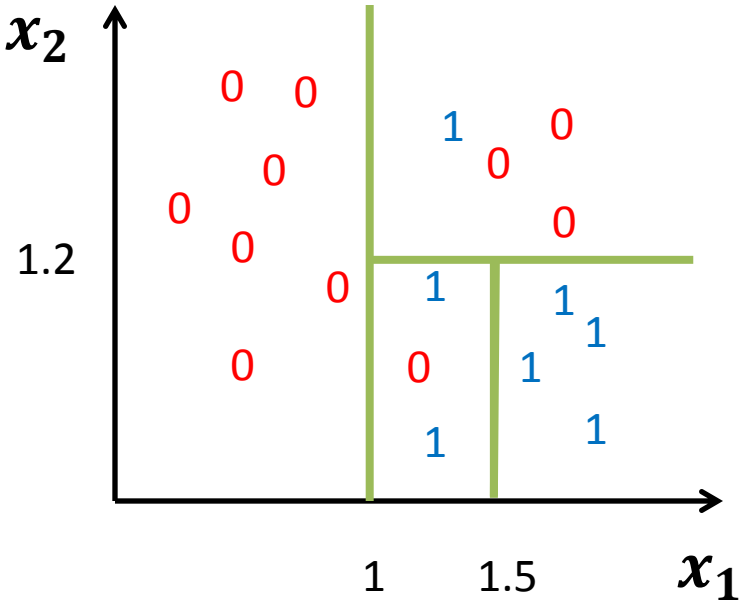
- Given $\{(\boldsymbol{x}^1, \boldsymbol{y}^1), (\boldsymbol{x}^2, \boldsymbol{y}^2), (\boldsymbol{x}^3, \boldsymbol{y}^3), \cdots, (\boldsymbol{x}^n, \boldsymbol{y}^n)\}$, find $\boldsymbol{y}$ for a given $\boldsymbol{x}$.
- E.g., $\boldsymbol{x}$ has two components, $\boldsymbol{y}$ is 0 or 1.



Binary-Decision Tree

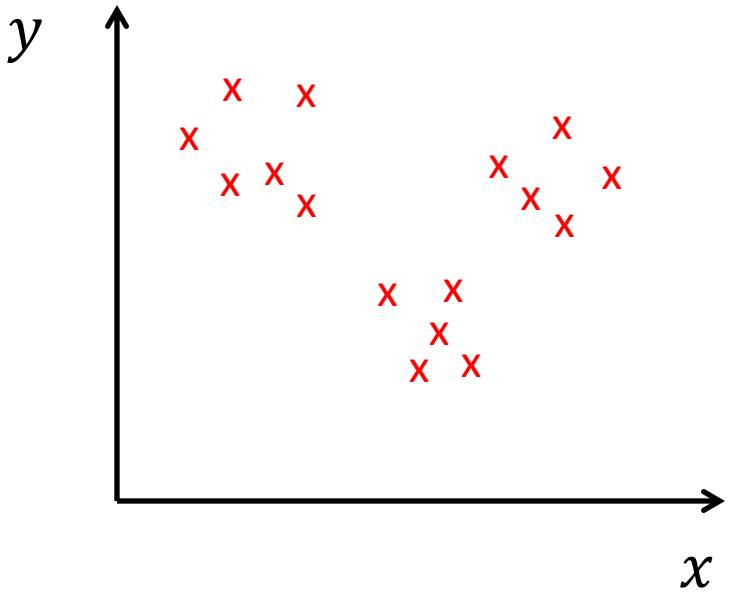
Here $(3, 1)$ represents:
3 "0" votes and 1 "1" vote.





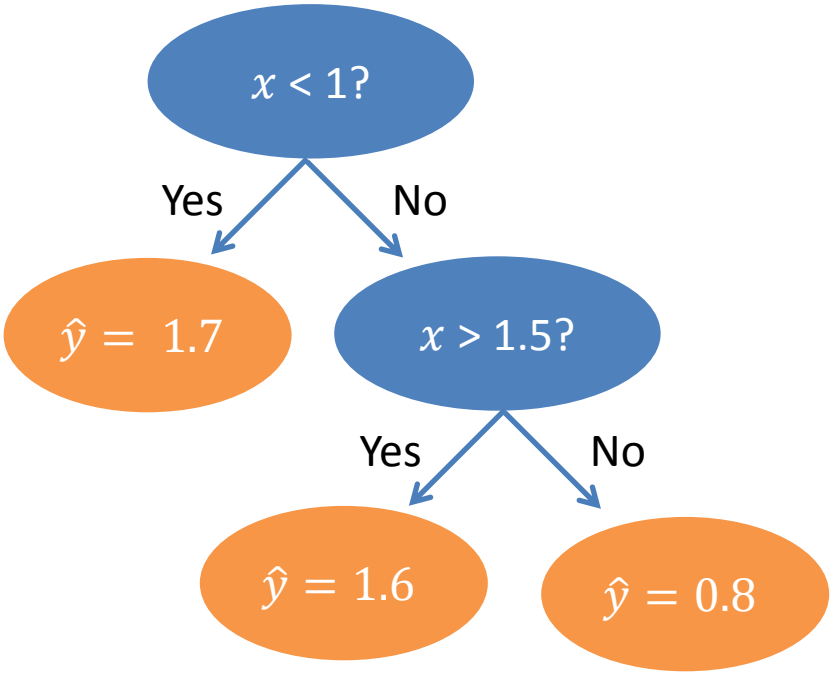
Regression Tree

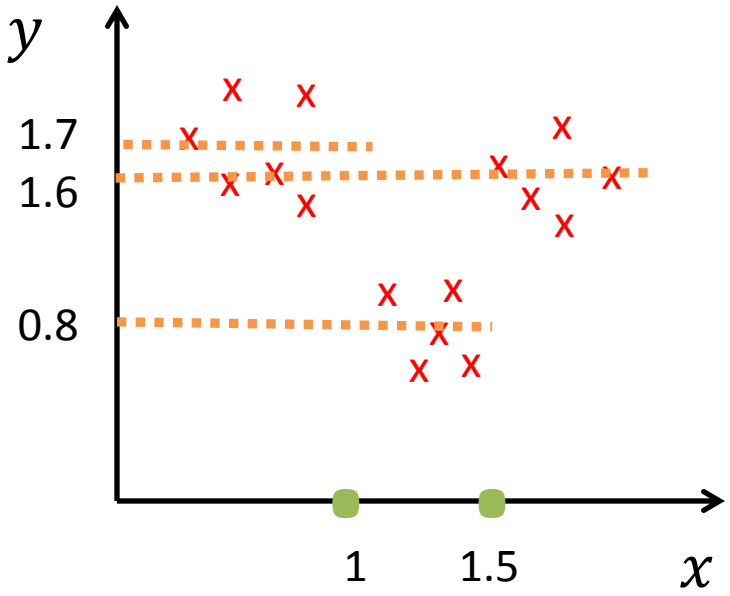
- E.g., $\boldsymbol{x}$ has one component ($= x$); $\boldsymbol{y}$ is real.



Regression Tree

- E.g., $\boldsymbol{x}$ has one component ($= x$); $\boldsymbol{y}$ is real.





Random Forest: Reduce the Variance of Estimator

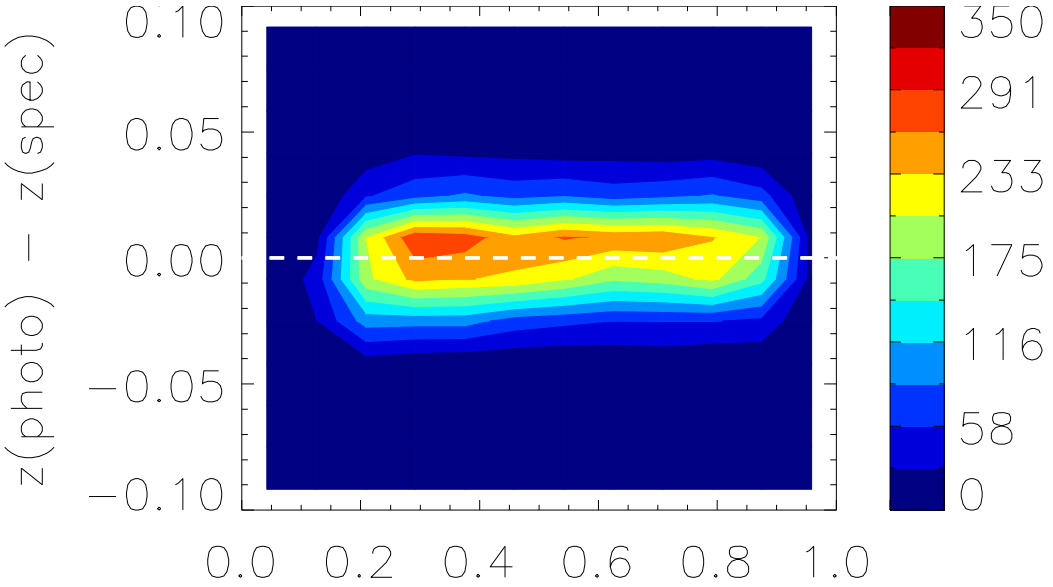
- Idea: use many trees instead of just one for estimation.
- Algorithm:
  - Sample a subset from data.
  - Construct i-th tree from the subset.
  - At each node, choose a random subset of m features. The split the data based on those features.
  - Repeat for $\mathrm{i} = 1, 2, \ldots, \mathrm{B}$ trees.
  - Given x
    - Take majority vote from trees (Classification)
    - Take average from trees (Regression)

Photometric Redshift Using Random Forest

- Use photometry (e.g., using SDSS filters u,g,r,i,z, or 5 data points) and galaxy inclination to estimate redshift (distance) of galaxies (Yip, Carliles & Szalay et al. 2011).

z(spec) is the true redshift, the redshift we obtained from galaxy spectra.

z(photo) is the estimated redshift, the redshift we obtained from galaxy photometry.



Galaxy Inclination

Comparison of Supervised Learning Algorithms

- Caruana & Niculescu-Mizil (2006) compared algorithms over 11 problems and 8 performance metrics.

Columns are the probability that an algorithm would perform at 1st, 2nd , 3rd, …, etc.

MODEL	1ST	2ND	3RD	4TH	5TH	6TH	7TH	8TH	9TH	10TH
BST-DT	0.580	0.228	0.169	0.023	0.009	0.000	0.000	0.000	0.000	0.000
RF	0.390	0.525	0.084	0.001	0.000	0.000	0.000	0.000	0.000	0.000
BAG-DT	0.030	0.232	0.571	0.150	0.017	0.000	0.000	0.000	0.000	0.000
SVM	0.000	0.008	0.148	0.574	0.240	0.029	0.001	0.000	0.000	0.000
ANN	0.000	0.007	0.035	0.230	0.606	0.122	0.000	0.000	0.000	0.000
KNN	0.000	0.000	0.000	0.009	0.114	0.592	0.245	0.038	0.002	0.000
BST-STMP	0.000	0.000	0.002	0.013	0.014	0.257	0.710	0.004	0.000	0.000
DT	0.000	0.000	0.000	0.000	0.000	0.000	0.004	0.016	0.291	0.089
LOGREG	0.000	0.000	0.000	0.000	0.000	0.000	0.040	0.312	0.423	0.225
NB	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.030	0.284	0.686

Cost Function

- In many machine learning algorithms, the idea is to find the model parameters $\boldsymbol{\theta}$ which minimize the cost function $J(\boldsymbol{\theta})$:

$$J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m}\left(model\left(data^{i}\right) - target^{i}\right)^{2}$$

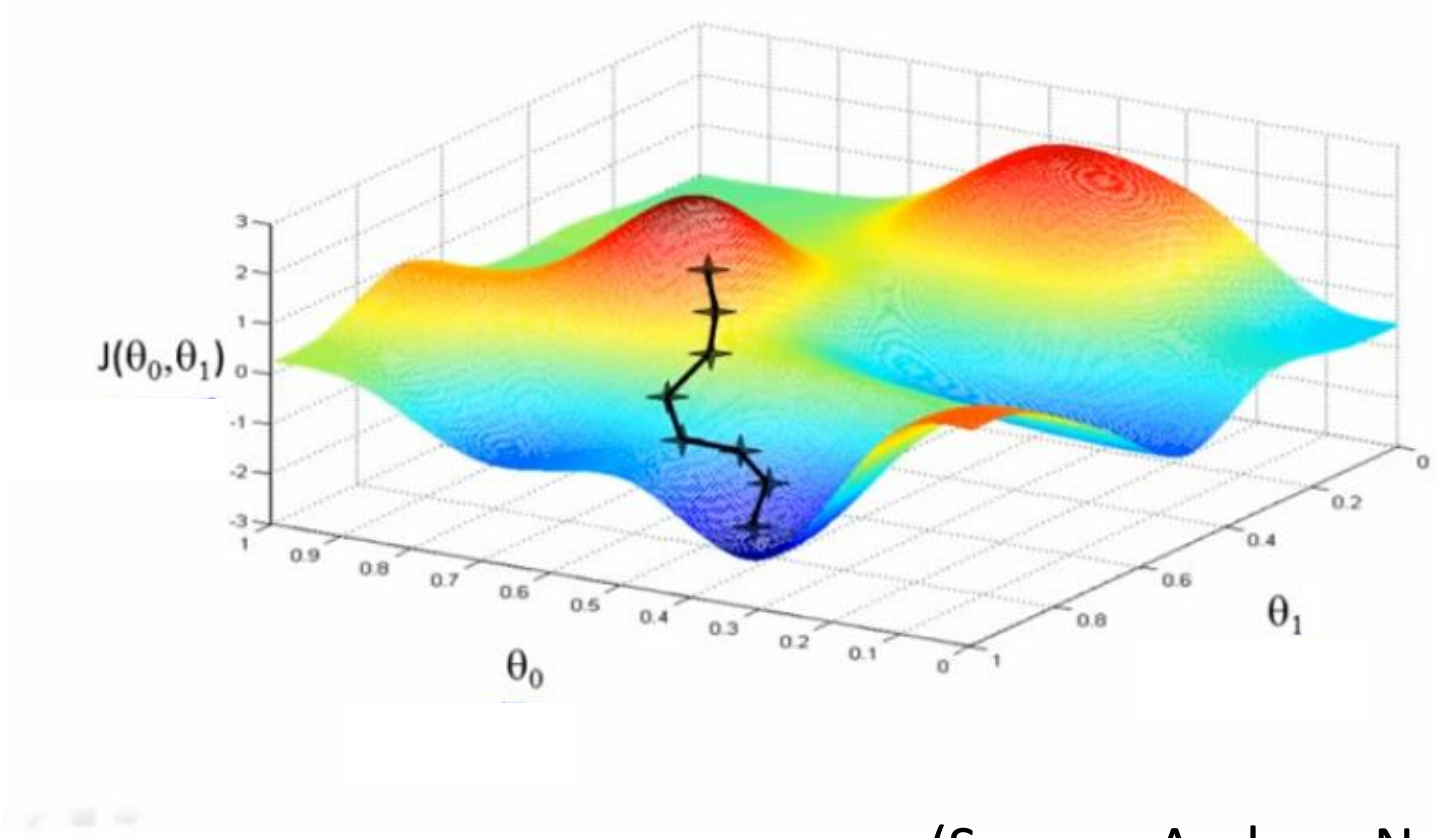
  m is the size of the training set.
- That is, we want $model$ as close to $target$ as possible.
- Note that $model$ depends on $\boldsymbol{\theta}$.

Minimization of Cost Function in order to Find Modeled Parameters

- Maximum Likelihood Estimate
- Bayesian Parameter Estimate
- Gradient Descent
- Etc.

Gradient Descent: Idea

The minimum is local – for a different starting point, we may get different local minimum.



(Source: Andrew Ng)

Minimization of Cost Function: Gradient Descent Algorithm

- Start at a point in the parameter space.
- Look around locally, take a baby step which has the largest descent.
- Repeat until we find the minimum cost.
- Mathematically, we keep refining θ_j until convergence, in accord to:

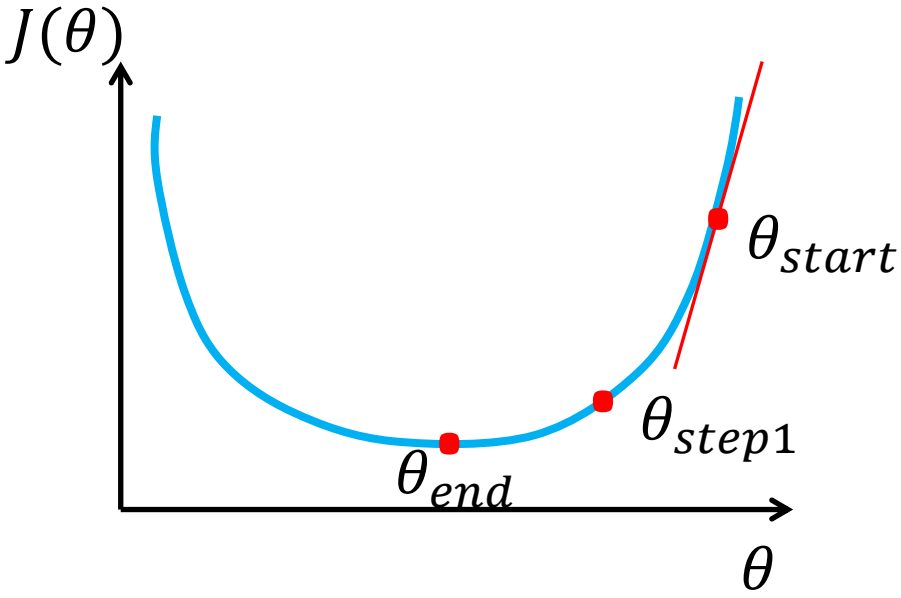
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

- α is the Learning rate.
- j runs from 1 to the number of parameters in the model.

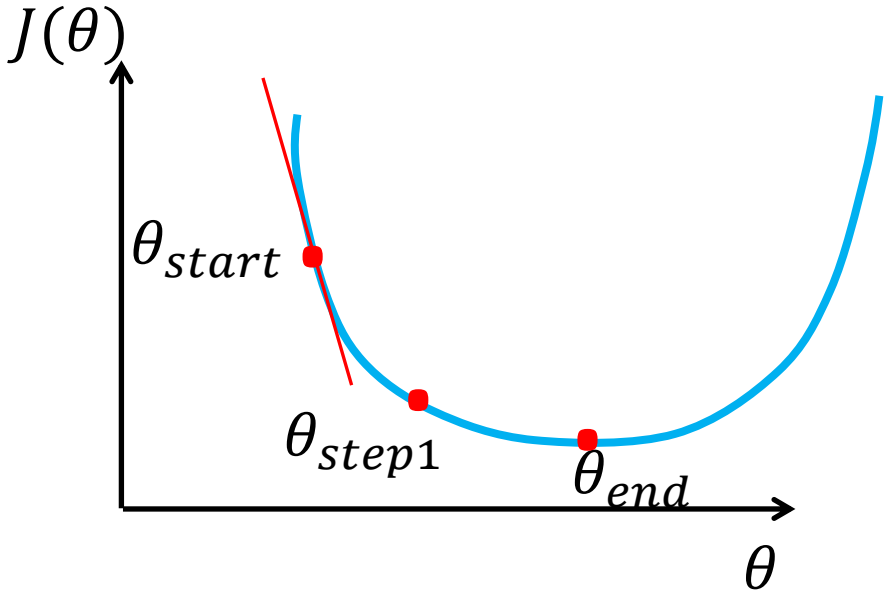
Example: Gradient Descent in 1D Data

$$\theta := \theta - \alpha \frac{d}{d\theta} J(\theta)$$

- The learning rate is positive.



Start from the right of the minimum.
$\frac{\partial}{\partial\theta} J(\theta)$ = slope > 0.
θ decreases as the algorithm progresses.



Start from the left of the minimum.

Model Selection in Machine Learning

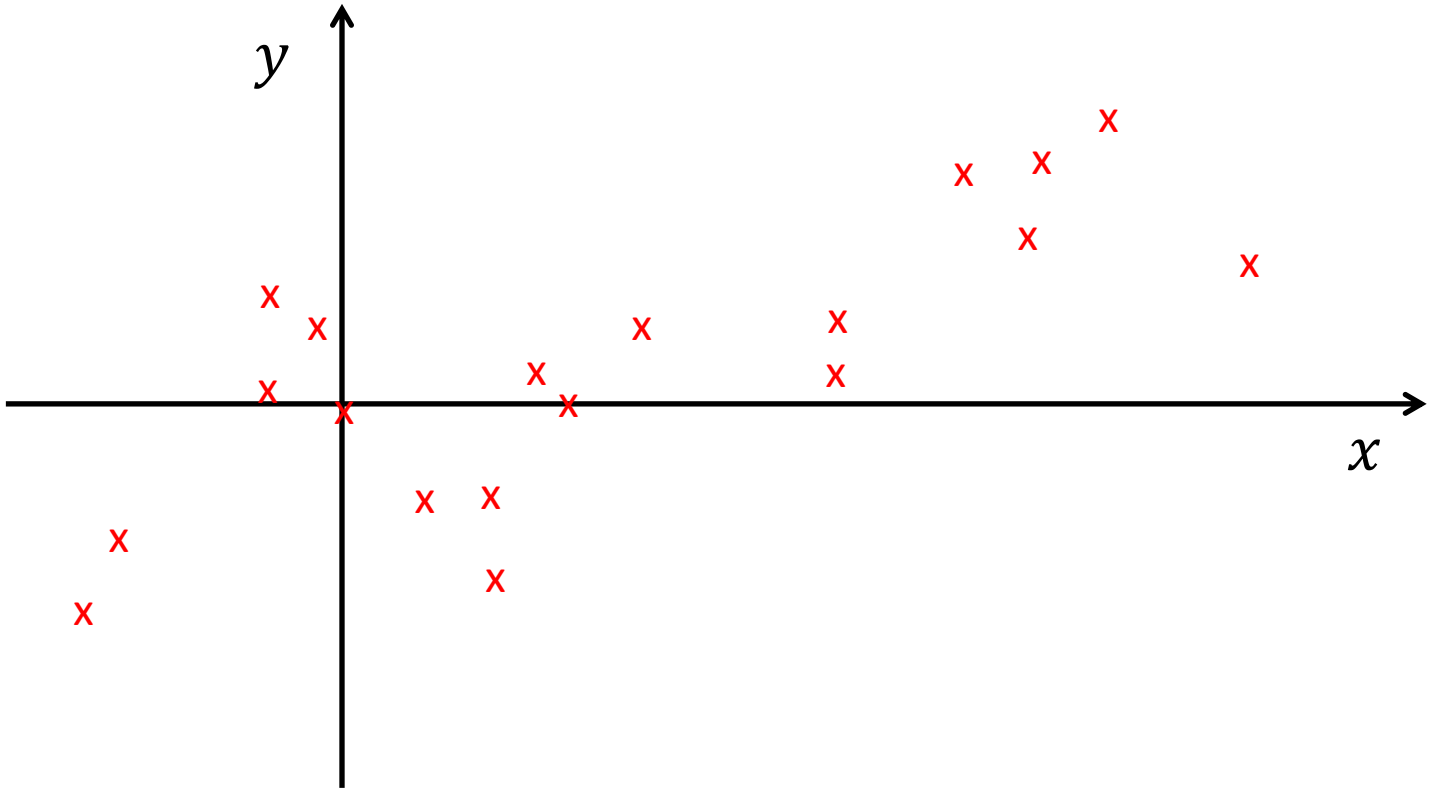
- The goal of Cost Function minimization is to select a model.
- Occam's Razor: the best explanation is usually the simplest one.
- Some model selection approaches:
  - Bayesian Approach
  - Cross Validation
  - Akaike Information Criterion (AIC)
  - Bayesian Information Criterion (BIC)

Model Selection: How to Select from Models with Different Complexity?

- Notice that the complexity parameters are not well defined and may need user's judgment.
- Complexity parameter of a model, e.g.:
  - The number of nearest neighbors (k) in KNN.
  - The number of basis functions in regression.

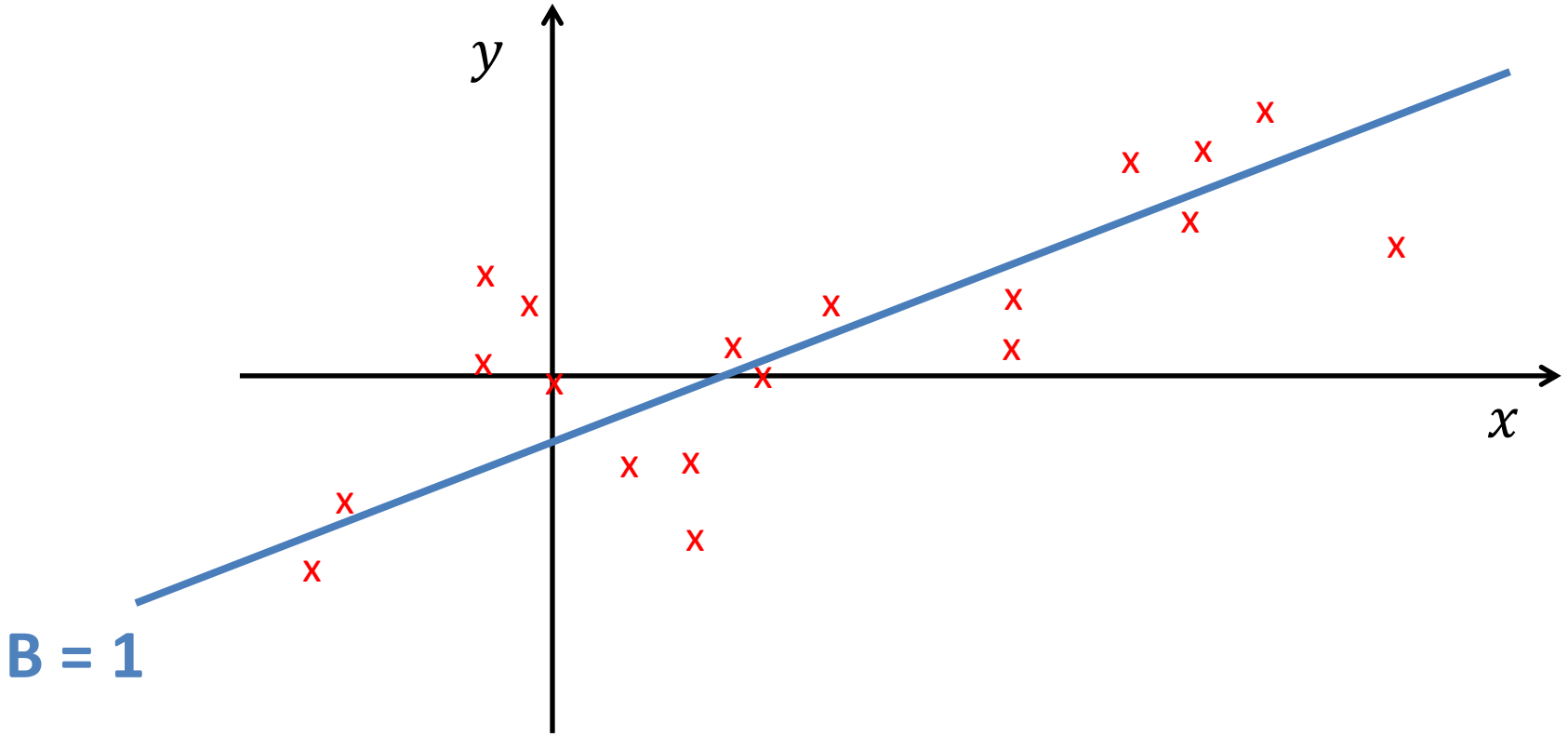
Example: Fitting a combination of Basis Functions

- E.g., Polynomial functions: $\{\emptyset_0, \emptyset_1, \emptyset_2, \cdots, \emptyset_B\}$ where $\emptyset_k = x^k$



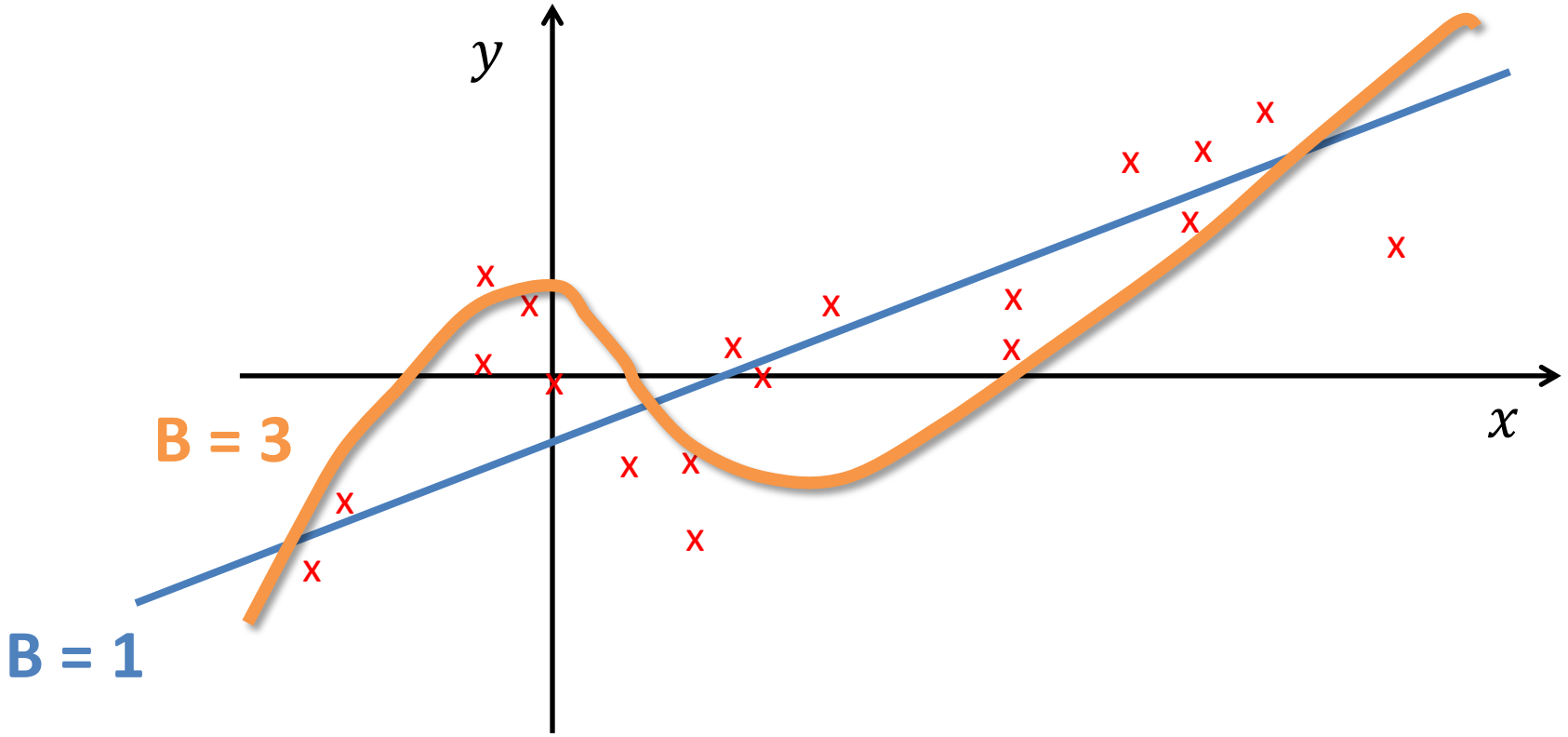
Schematics: Fitting a combination of Basis Functions

- E.g., Polynomial functions: $\{\emptyset_0, \emptyset_1, \emptyset_2, \cdots, \emptyset_B\}$ where $\emptyset_k = x^k$



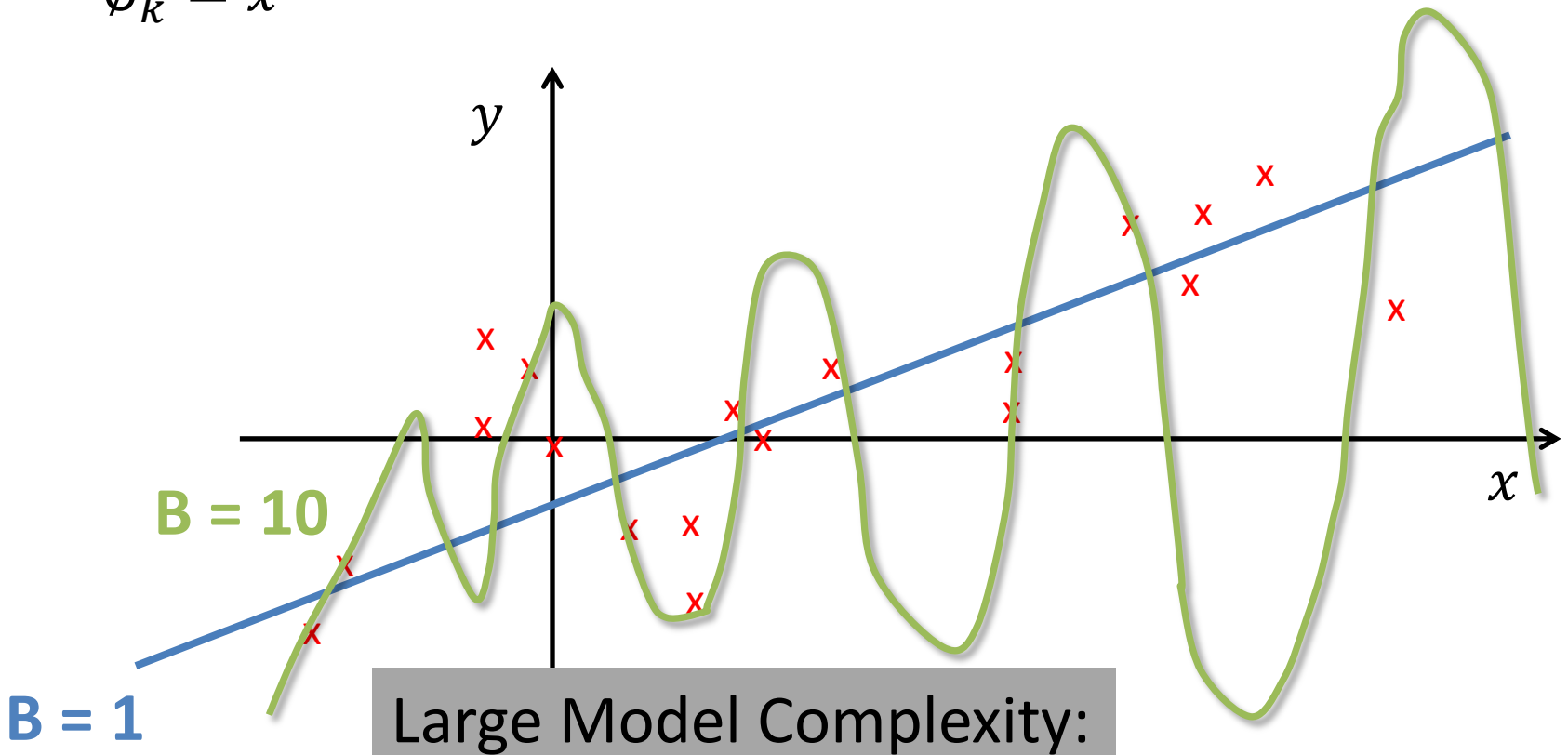
Schematics: Fitting a combination of Basis Functions

- E.g., Polynomial functions: $\{\emptyset_0, \emptyset_1, \emptyset_2, \cdots, \emptyset_B\}$ where $\emptyset_k = x^k$



Schematics: Fitting a combination of Basis Functions

- E.g., Polynomial functions: $\{\emptyset_0, \emptyset_1, \emptyset_2, \cdots, \emptyset_B\}$ where $\emptyset_k = x^k$



Large Model Complexity: Potential Overfitting

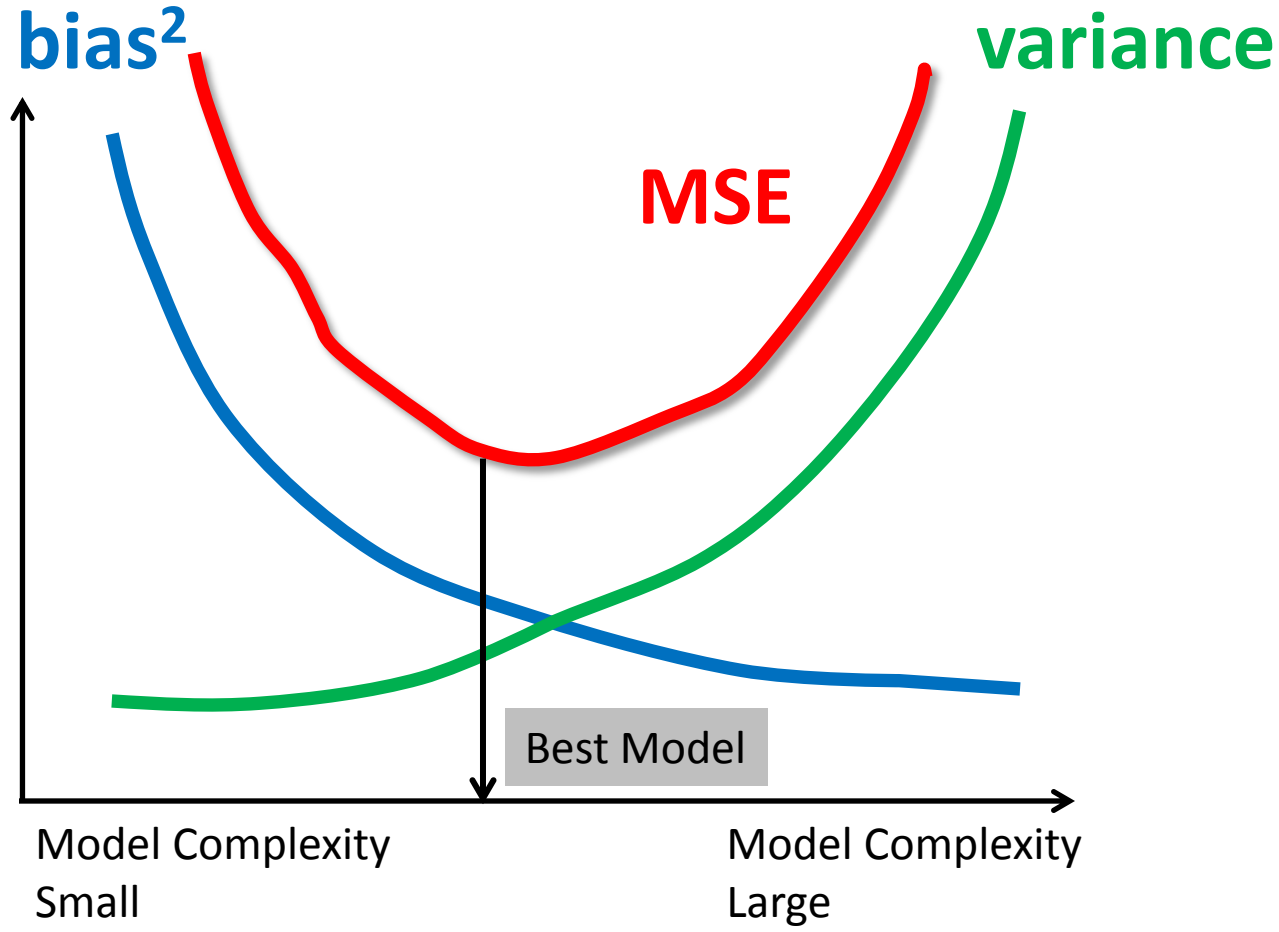
Bias-Variance Decomposition

- Suppose we have an estimator of the model parameters.
- An estimator is denoted as $\hat{\theta}$.
- Mean Square Error of an estimator can be expressed as:

$$\text{MSE} = \text{bias}^2 + \text{var}$$

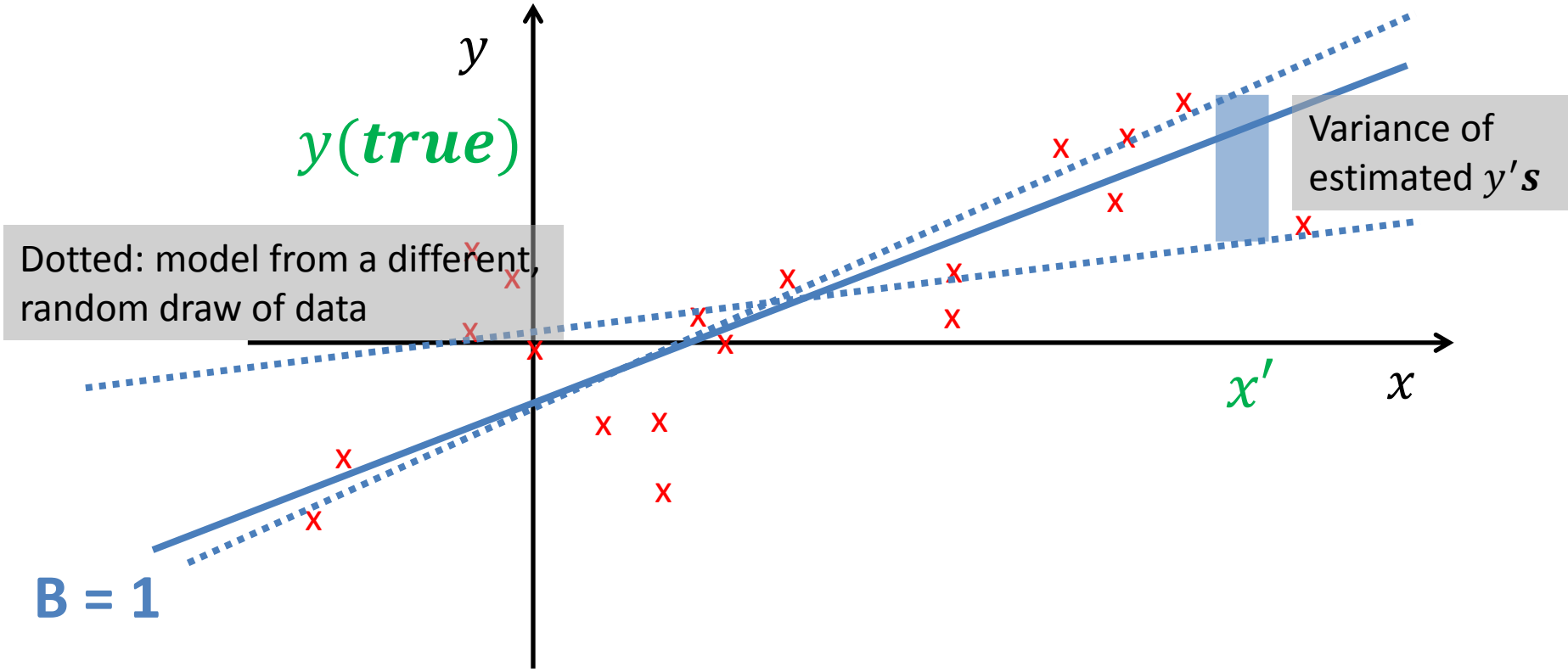
where $MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{\theta} - \theta_{True})^2$

The Best Model: Tradeoff between Bias and Variance



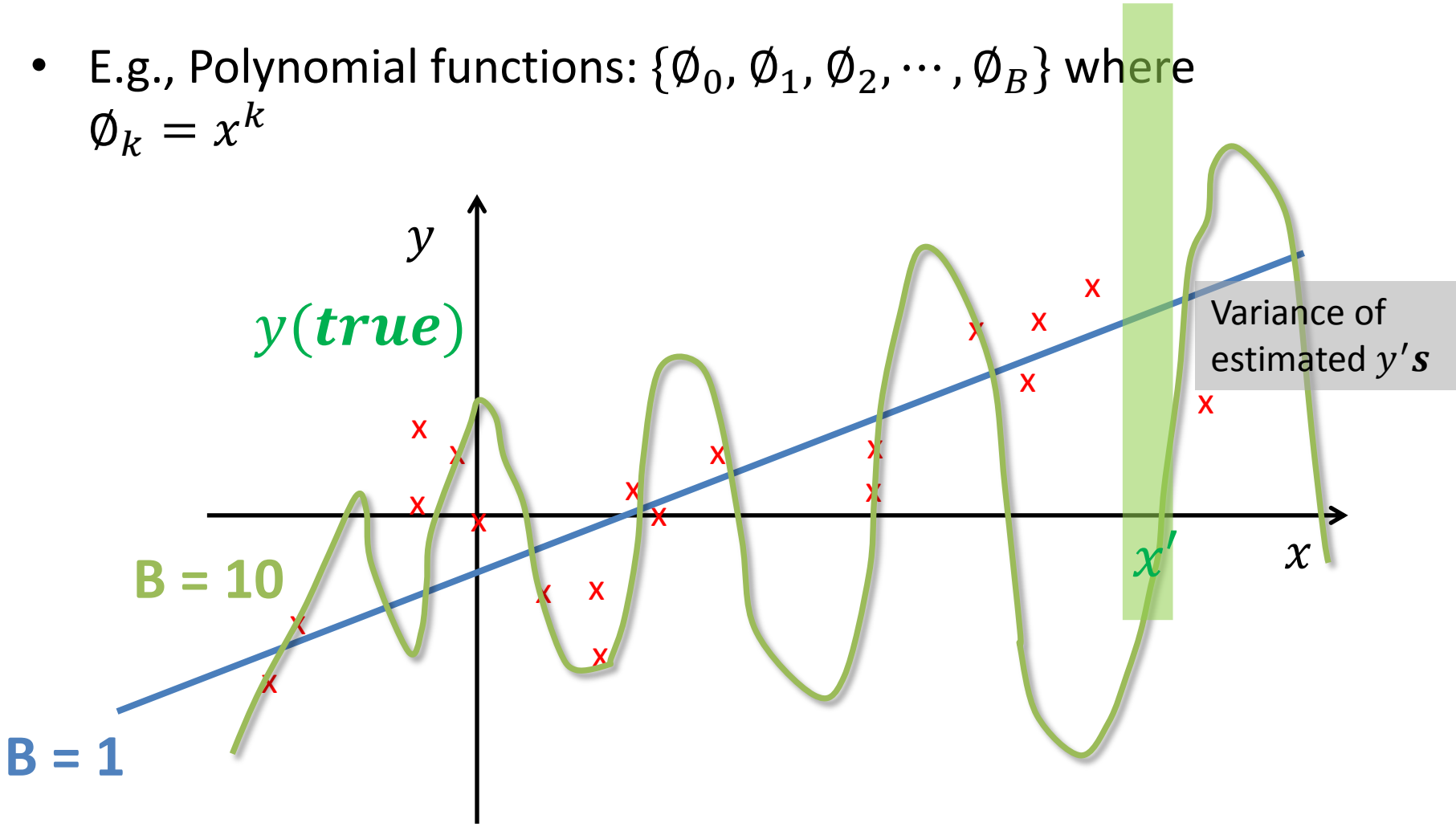
Intuition: Bias-Variance Tradeoff

- E.g., Polynomial functions: $\{\emptyset_0, \emptyset_1, \emptyset_2, \cdots, \emptyset_B\}$ where $\emptyset_k = x^k$



Intuition: Bias Variance Tradeoff

- E.g., Polynomial functions: $\{\emptyset_0, \emptyset_1, \emptyset_2, \cdots, \emptyset_B\}$ where $\emptyset_k = x^k$



Demonstration of Overfitting in R

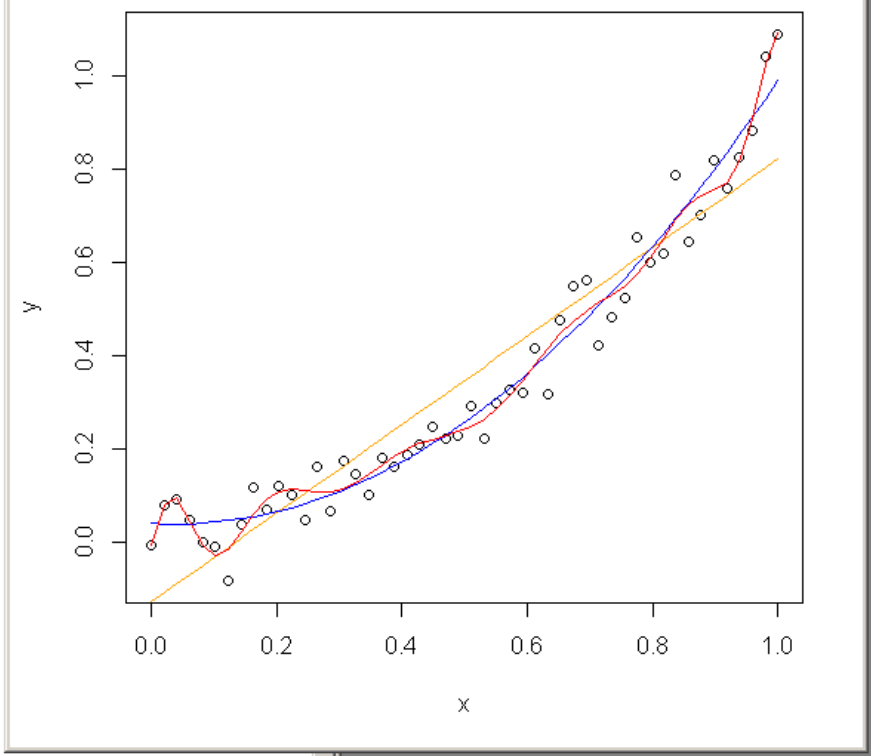
RGui (64-bit)

File History Resize Windows

R Console

```
> # Set number of data points.
> n <- 50
> # Set x.
> x <- seq(0, 1, length = n)
> # Fix y to be 2nd order polynomial of x (with randomness).
> noise = runif(n, -0.1, 0.1)
> ytrue = x^2
> y <- ytrue + noise
> # Fit straight line, 2nd order poly, and 10th order poly.
> fit1 <- lm(y ~ poly(x, 1, raw = TRUE))
> fit2 <- lm(y ~ poly(x, 2, raw = TRUE))
> fit3 <- lm(y ~ poly(x, 20, raw = TRUE))
> # Calculate y estimate
> yhead1 <- predict(fit1, data.frame(x = x))
> yhead2 <- predict(fit2, data.frame(x = x))
> yhead3 <- predict(fit3, data.frame(x = x))
Warning message:
In predict.lm(fit3, data.frame(x = x)) :
  prediction from a rank-deficient fit may be misleading
> # Plot data and predictions.
> # The 10th order polynomial fit demonstrates overfitting.
> plot(x, y)
> lines(x, yhead1, col = "orange")
> lines(x, yhead2, col = "blue")
> lines(x, yhead3, col = "red")
> # Calculate Bias.
> bias1 <- mean(yhead1 - ytrue)
> bias2 <- mean(yhead2 - ytrue)
> bias3 <- mean(yhead3 - ytrue)
> # Calculate Var.
> var1 <- var(yhead1)
> var2 <- var(yhead2)
> var3 <- var(yhead3)
> # Calculate MSE.
> mse1 = bias1^2 + var1
> mse2 = bias2^2 + var2
> mse3 = bias3^2 + var3
> # Print MSE for each fit.
> mse1
[1] 0.07977807
> mse2
[1] 0.0862601
> mse3
[1] 0.08740793
> # Summary: MSE2 has minimum MSE.
> # End
>
```

R Graphics: Device 2 (ACTIVE)



Some Variations of Machine Learning

- Semi-Supervised Learning:
  - Given data $\{(\boldsymbol{x}^1, \boldsymbol{y}^1), (\boldsymbol{x}^2, \boldsymbol{y}^2), (\boldsymbol{x}^3, \boldsymbol{y}^3), \cdots, (\boldsymbol{x}^k, \boldsymbol{y}^k), \boldsymbol{x}^{k+1}, \cdots, \boldsymbol{x}^n\}$ predicts labels $\boldsymbol{y}^{k+1}, \cdots, \boldsymbol{y}^n$ for $\boldsymbol{x}^{k+1}, \cdots, \boldsymbol{x}^n$.
- Active Learning:
  - Similar to Semi-Supervised but we can ask for extra labels $\boldsymbol{y}^i$ for particular data points $\boldsymbol{x}^i$ as the algorithm runs.