Center for Complex
Engineering Systems

Introduction to Applied Machine Learning

Abdullah Almaatouq
(amaatouq@mit.edu)

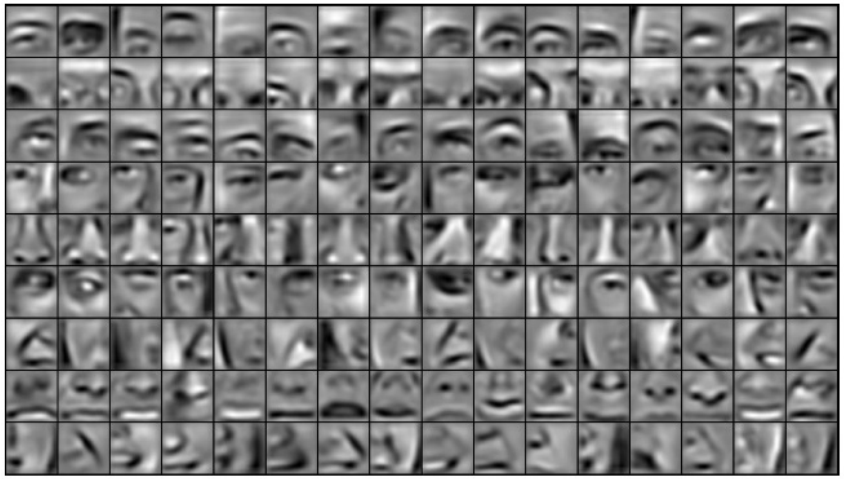
Abdulaziz Alhassan
(a.alhassan@cces-kacst-mit.org)

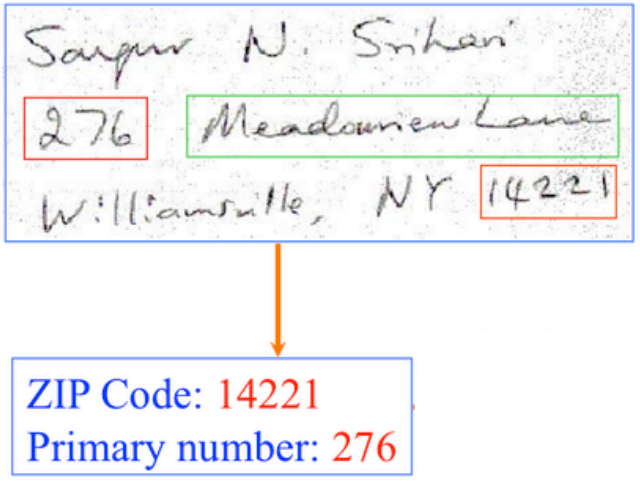
Ahmad Alabdulkareem
(kareem@mit.edu)

Tutorial Outline

- What is machine learning
- How to learn
- How to learn well
- Practical Session

Examples of Machine Learning





What is Machine Learning

- Given computers the ability to learn without explicitly being programmed. – Arthur Samuel 1959

- Machine learning is a computer program that improves its performance in a certain task with experience. – Tom Mitchell (1998)

What is Machine Learning

- Making predictions, based on learned patterns from historical data.

- Not to be confused with data mining
  - Data mining focuses on exploiting patterns in historical data.

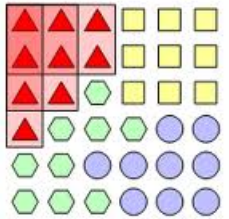
- New field?
  - Statistical Generalization
  - Computational Statistics

Why Machine learning?

- Huge amounts of data !!!!
- Humans are expensive, computers are cheap.
  - 88ECU, 32CPU and 244GB RAM ≈ $3.50/h
  - Minimum worker wage in the US is $7.25/h
- Rule based is (often):
  - Difficult to capture all relevant patterns.
  - You can't add rules to patterns you don't know exist
  - Very hard to maintain
  - (often) doesn't work well
- Psychic superpowers (sometimes!)

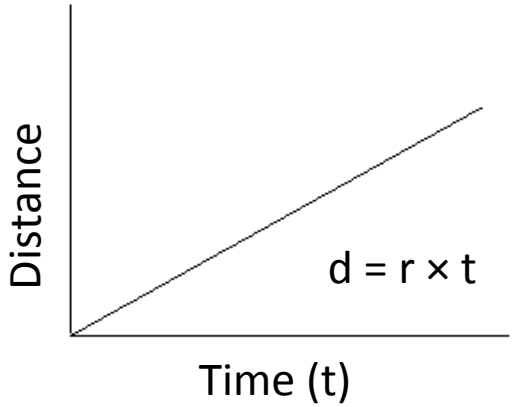
When to use Machine Learning

- There is a pattern to be detected



- You can NOT pin it mathematically

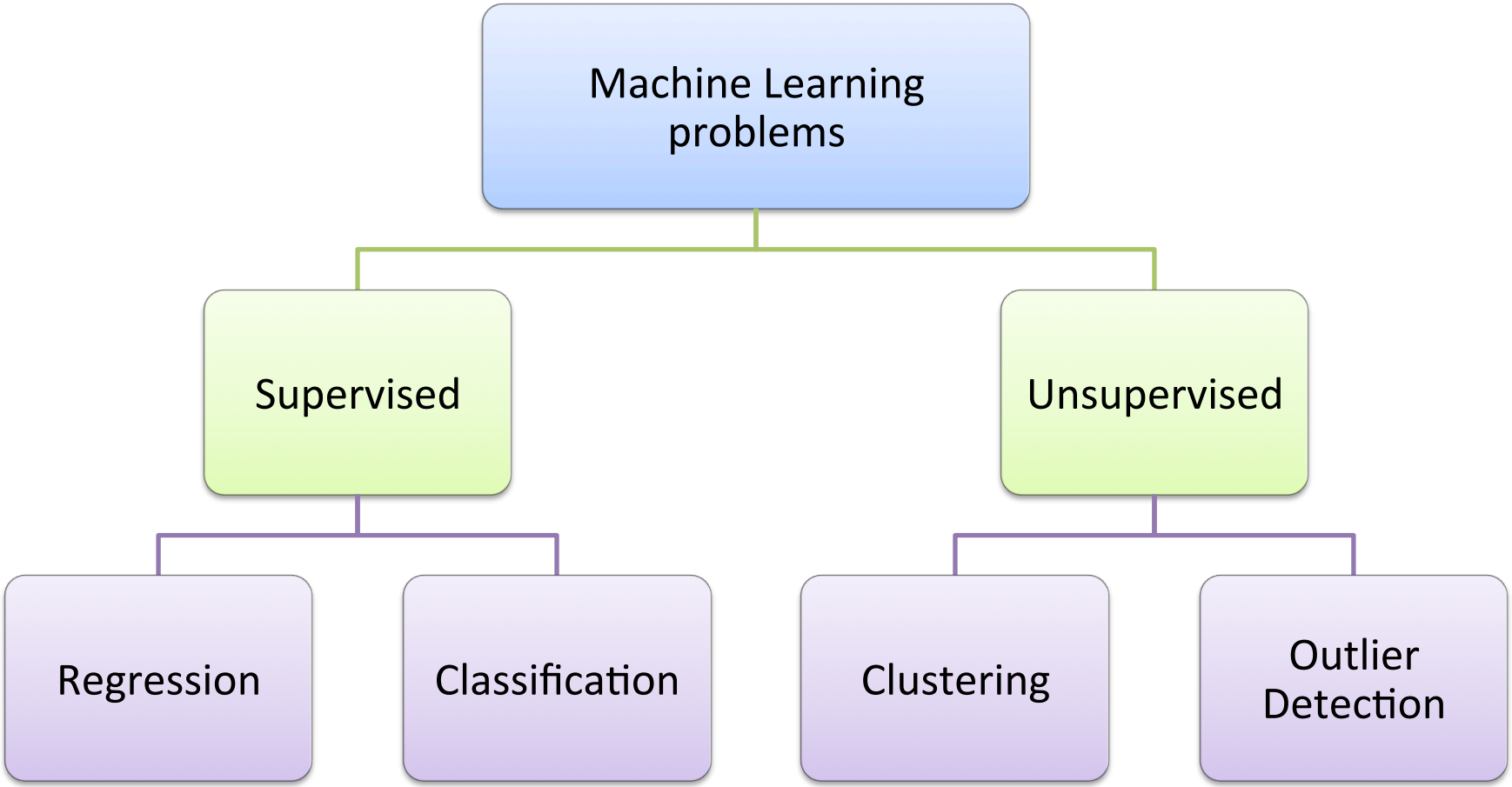
- You have data



Types of Problems

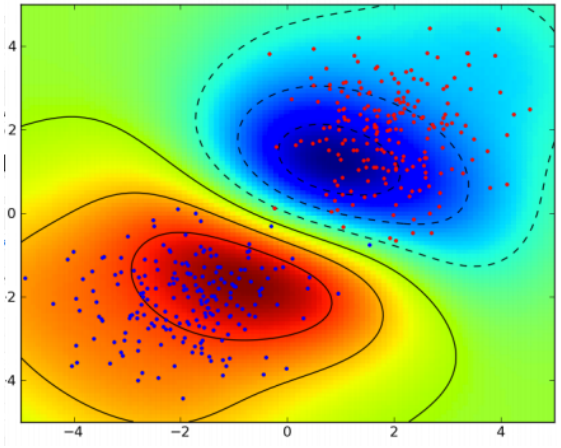
- Supervised
  - Training from seen labeled examples to generalize to unseen new observations.
  - Given some input x_i predict 'class/value' of y_i
- Unsupervised
  - Find hidden structures in unlabeled data

Types of Problems

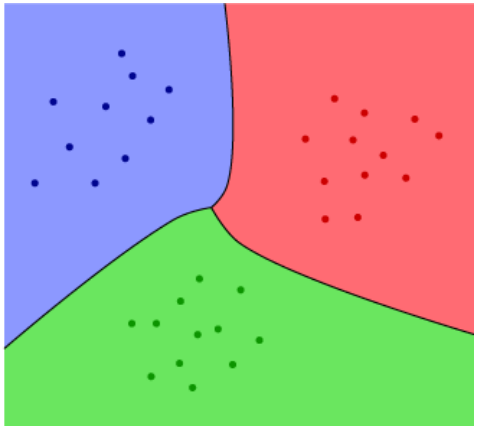


Supervised Learning

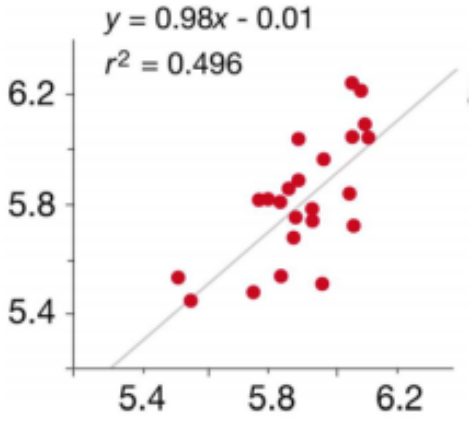
- **Classification**
  - Binary: Given x_i find y_i in {-1, 1}
  - Multicategory : Given x_i find y_i in {-1, .. K}
- **Regression**
  - Given x_i find y_i in R (or R^d)



Spam VS Not Spam

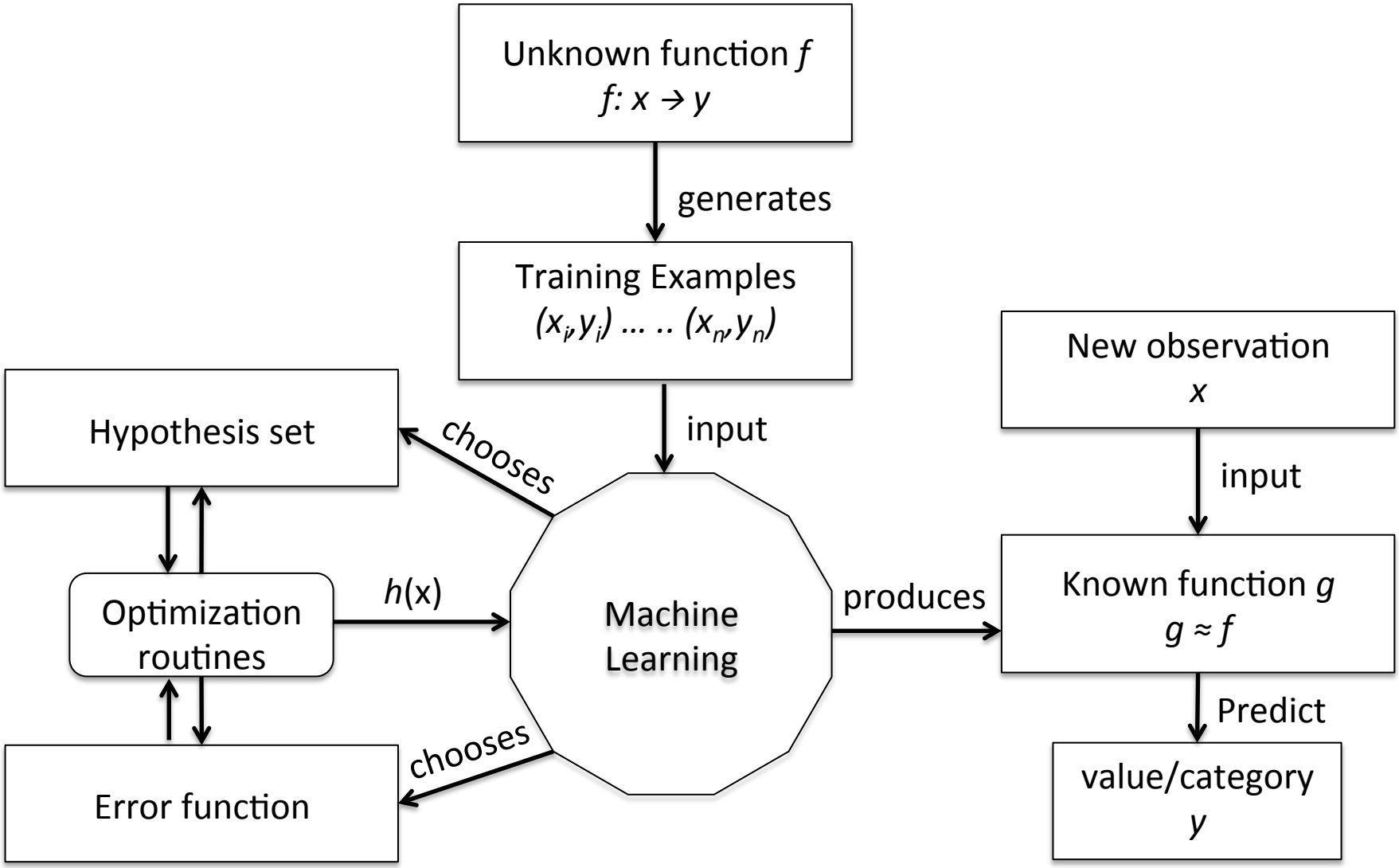


Images to digits



Beak depth prediction

Supervised Learning Framework



Linear Regression

Linear form: y = ax+b

- Capture the collective behavior of credit officers
- Predict the credit line for new customers

Input x =

Bias	1
Age	23 years
Annual salary	$144000
Years in residence	4 years
Years in job	2 years
…..	…

= [1,23,144000,4,2..]

$$h(x) = \theta_1 x_1 + \ldots + \theta_d x_d + \theta_0 x_0$$

Linear regression output: $h(x) = \sum_{i=0}^{d} \theta_i x_i = \theta^T x$

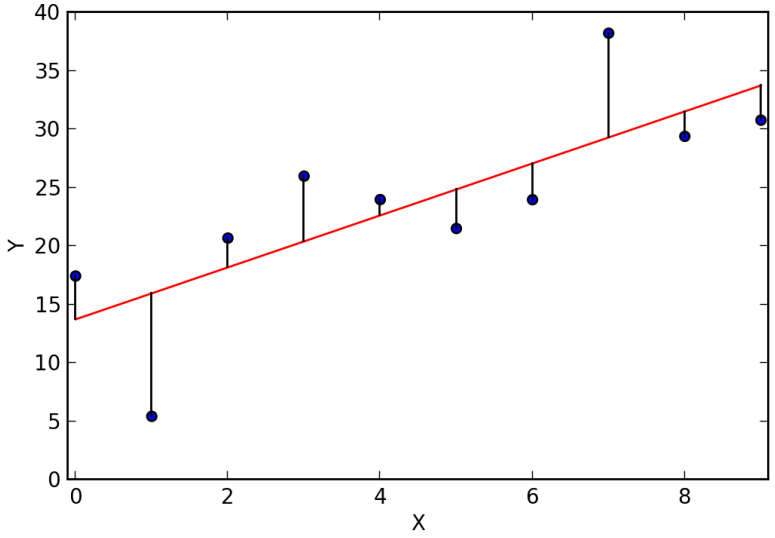
Linear Regression

- The historical dataset

  $(x_1,y_1), (x_2,y_2),\ldots, (x_N,y_N)$

- $y_n \in R$ is the credit line for customer x_n
- How well our $h(x) = \theta^T x$ approximate $f(x)$
  - Squared error $(h(x) - f(x))^2$
  - $E(h) = \frac{1}{N}\sum_{n=1}^{N}(h(x_n) - y_n)^2$
- Goal is to choose h that minimizes the $E(h)$

Linear Regression



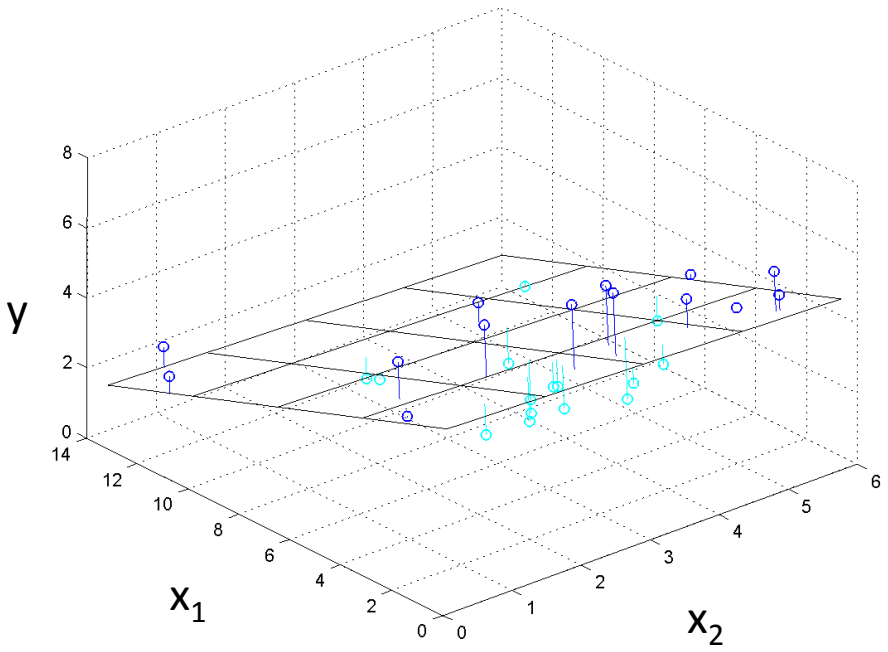


Illustration of 2d and 3d regression line and plane. It works the same in higher dimensions

Setting-up the problem

$$E(h) = \frac{1}{N}\sum_{n=1}^{N}(h(x_n) - y_n)^2$$

$$X = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{pmatrix}$$

$$E(\theta) = \frac{1}{N}\sum_{n=1}^{N}(\theta^T x_n - y_n)^2$$

$$= \frac{1}{N} \| X\theta - y \|^2$$

$$X = \begin{pmatrix} & \text{Features} & \\ \text{Observations} & & \end{pmatrix}$$

Minimizing *E*(h)

$$E(\theta) = \frac{1}{N} \| X\theta - y \|^2$$

$$\bigtriangledown E(\theta) = \frac{2}{N} X^T (X\theta - y)$$

$$= \frac{2}{N} X^T (X\theta - y) = 0$$

$$= \frac{2}{N} (X^T X\theta - X^T y) = 0$$

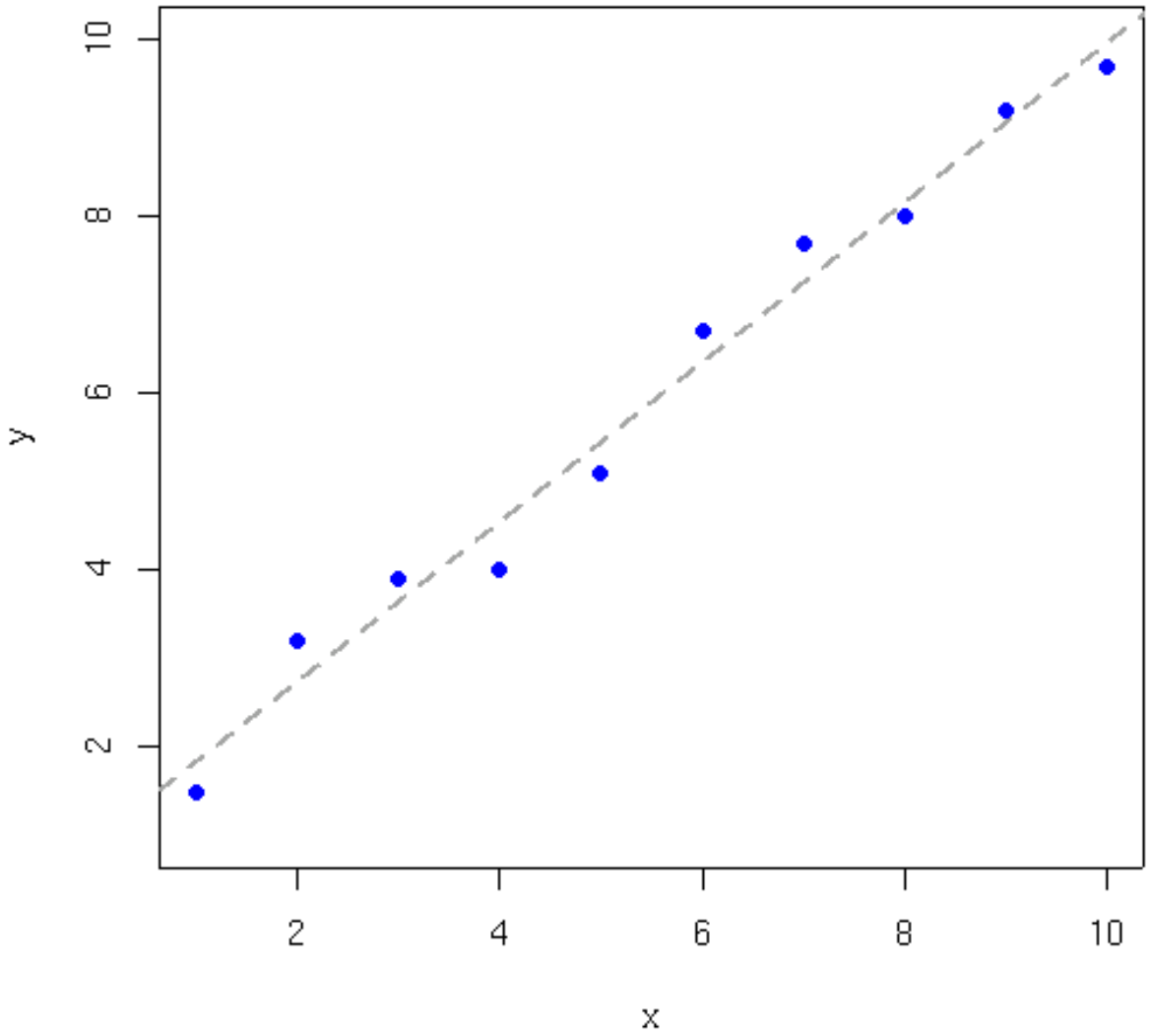
$$\theta = (X^T X)^{-1} X^T y$$

Python:
theta = linalg.inv (X.T*X)*X.T*y

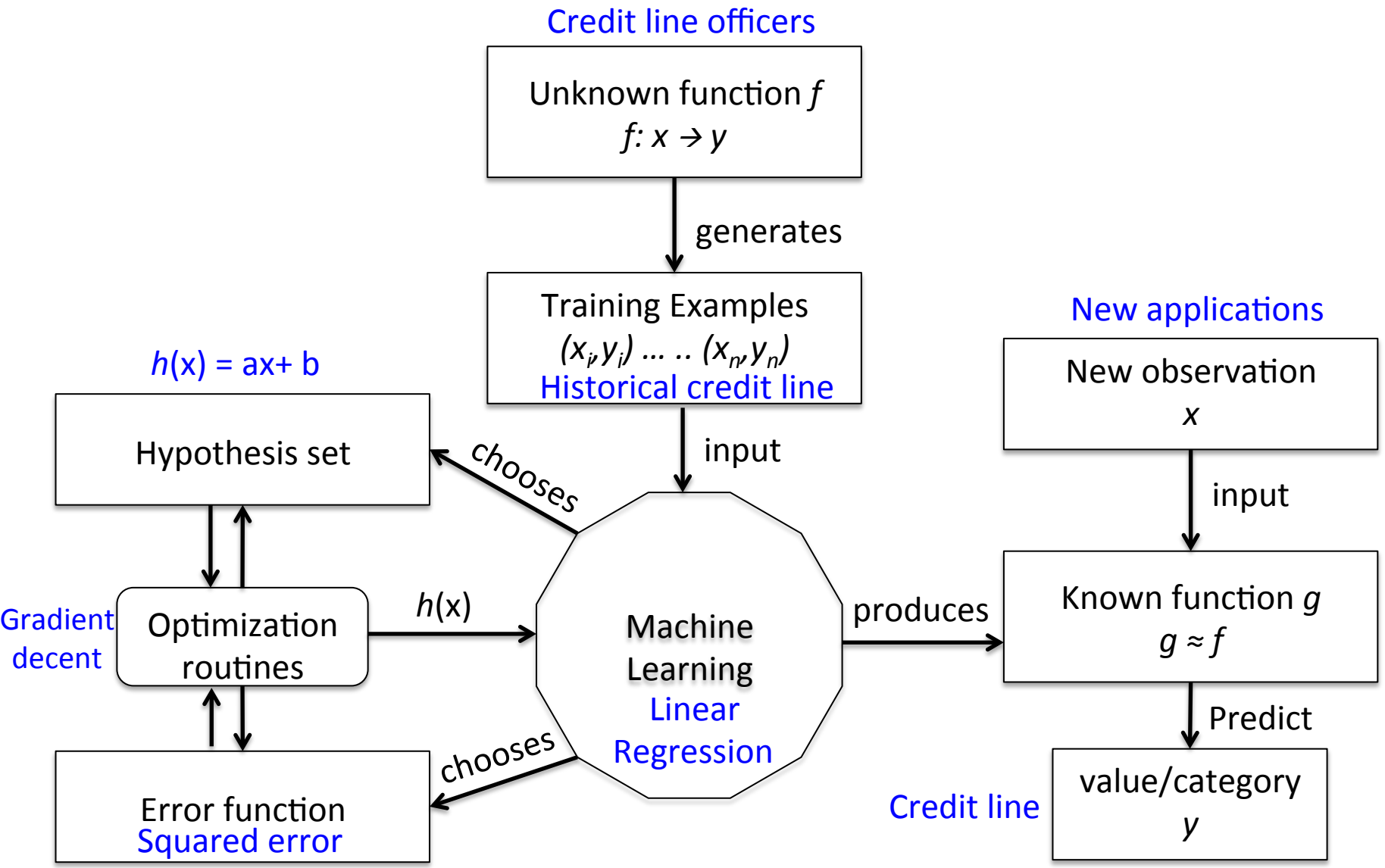
MATLAB:
theta = inv(X'*X)*X'*y

Inverse here is pseudo-inverse

Linear Regression

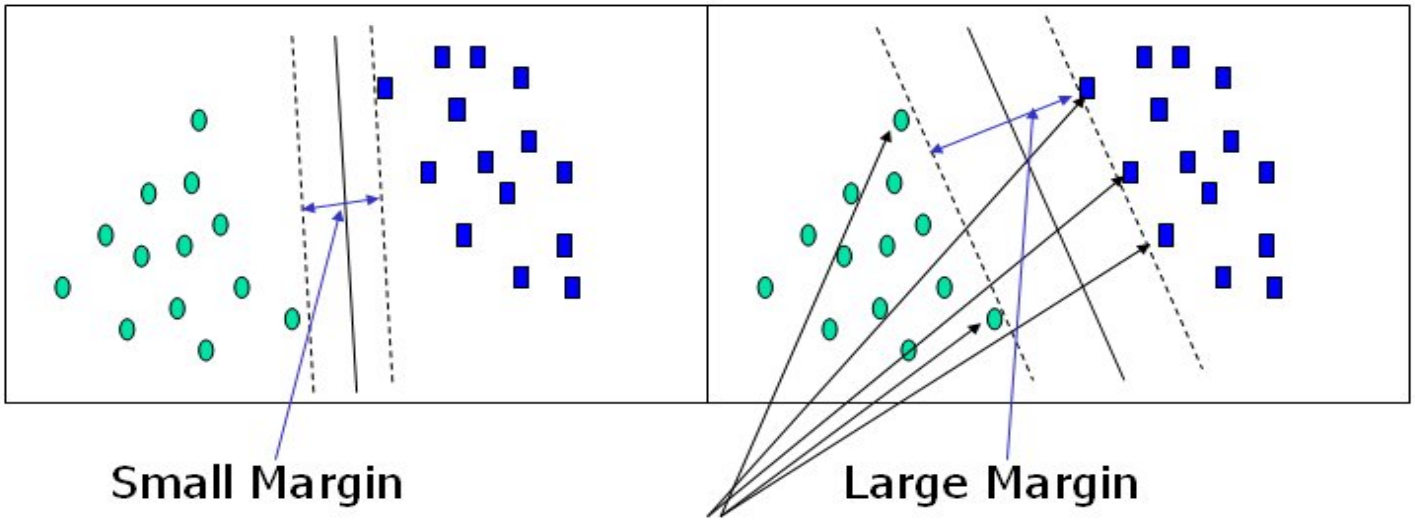


Supervised Learning Framework



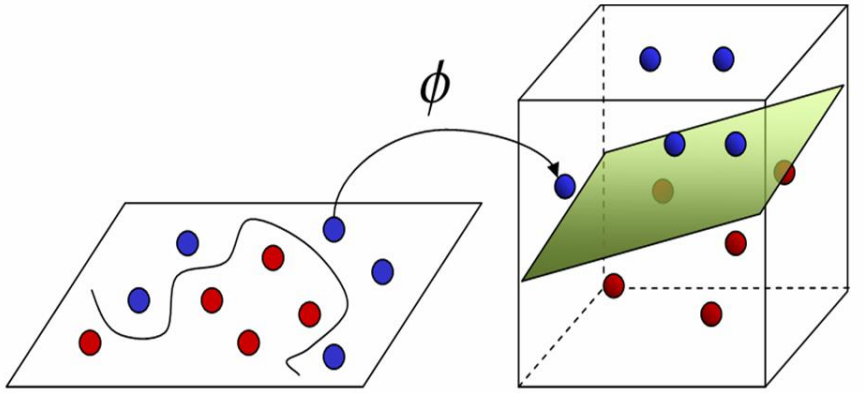
Support Vector Machines

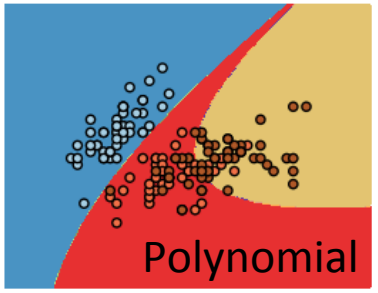
- Separating plane with maximum margin
- Margins are configurable with parameters to allow for mistakes
- Gold standard blackbox for many practitioners

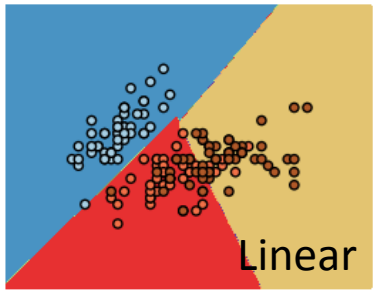


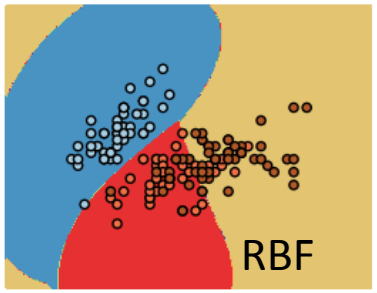
Support Vector Machines

- Effective in high dimensional spaces
- effective even when #features > #observations
- It can uses different kernel functions

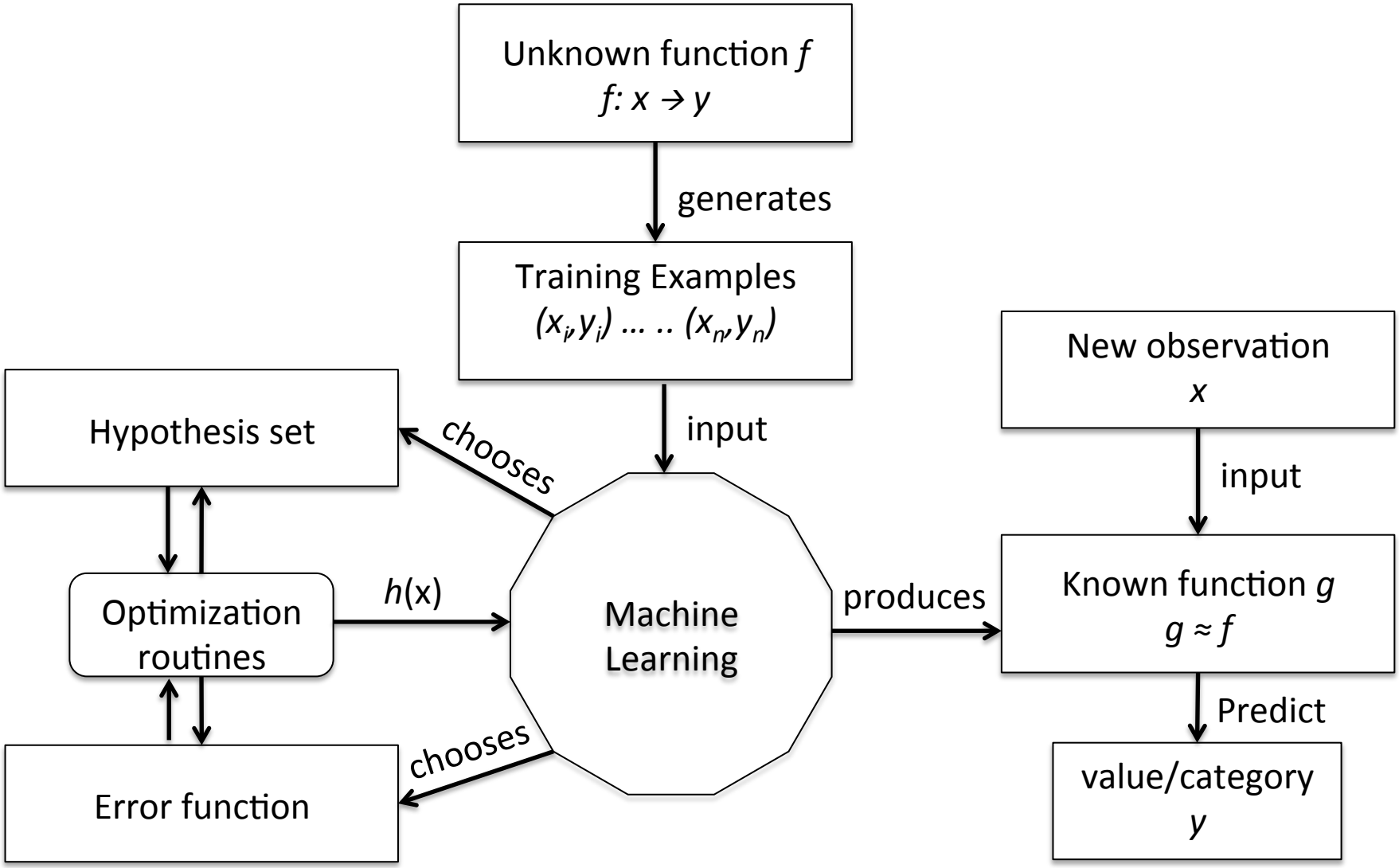




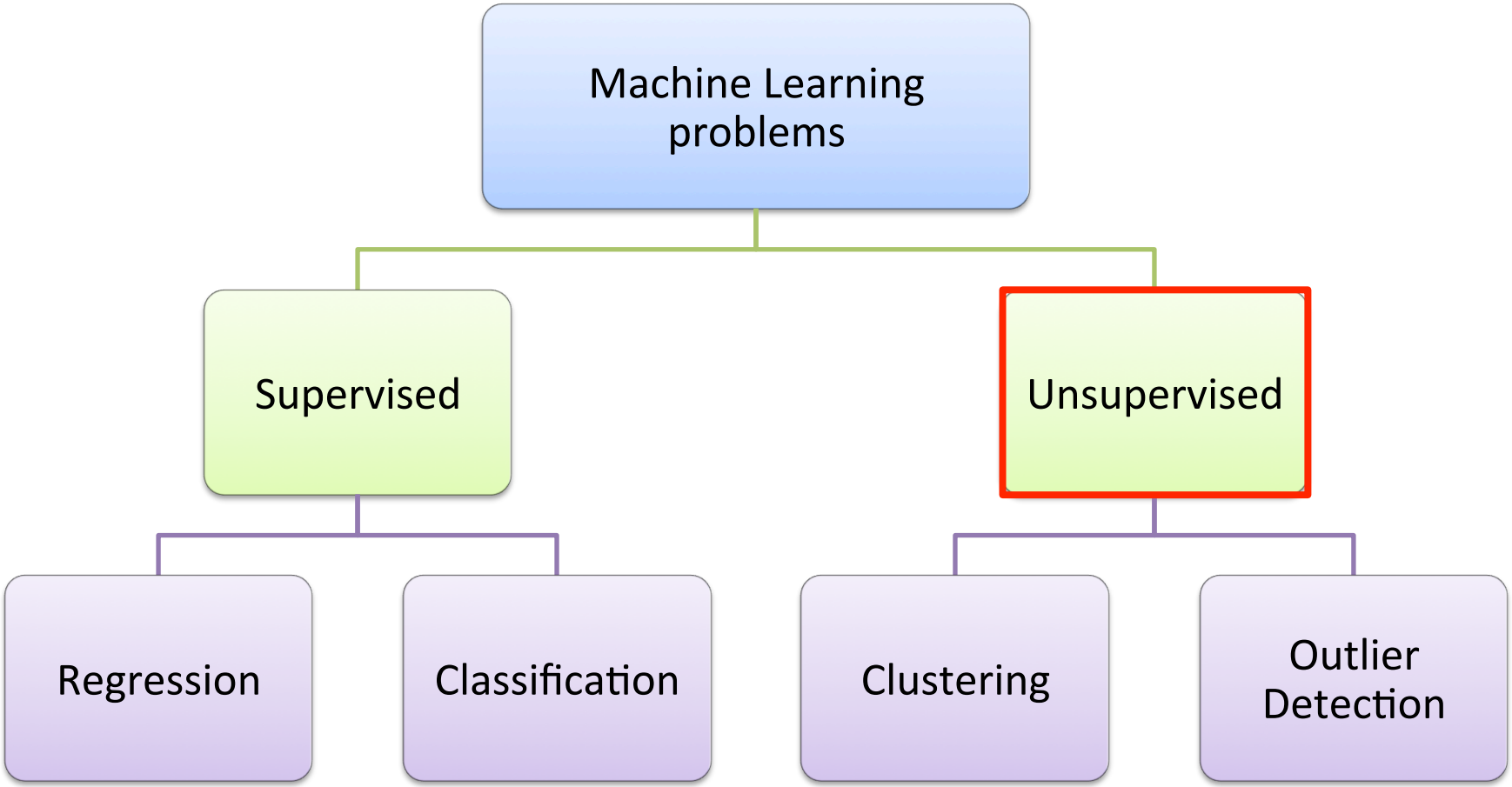




Supervised Learning Framework

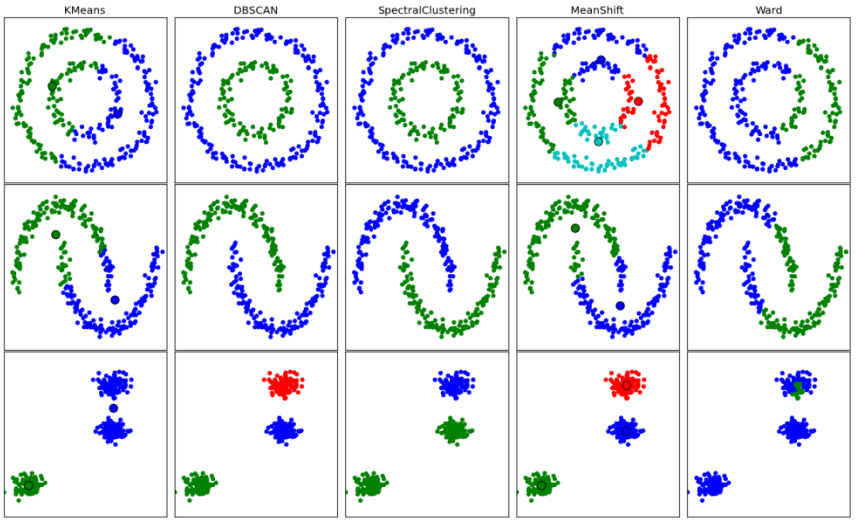


Types of Problems



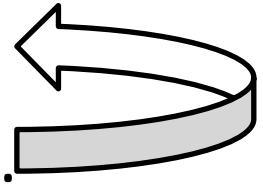
Unsupervised Learning

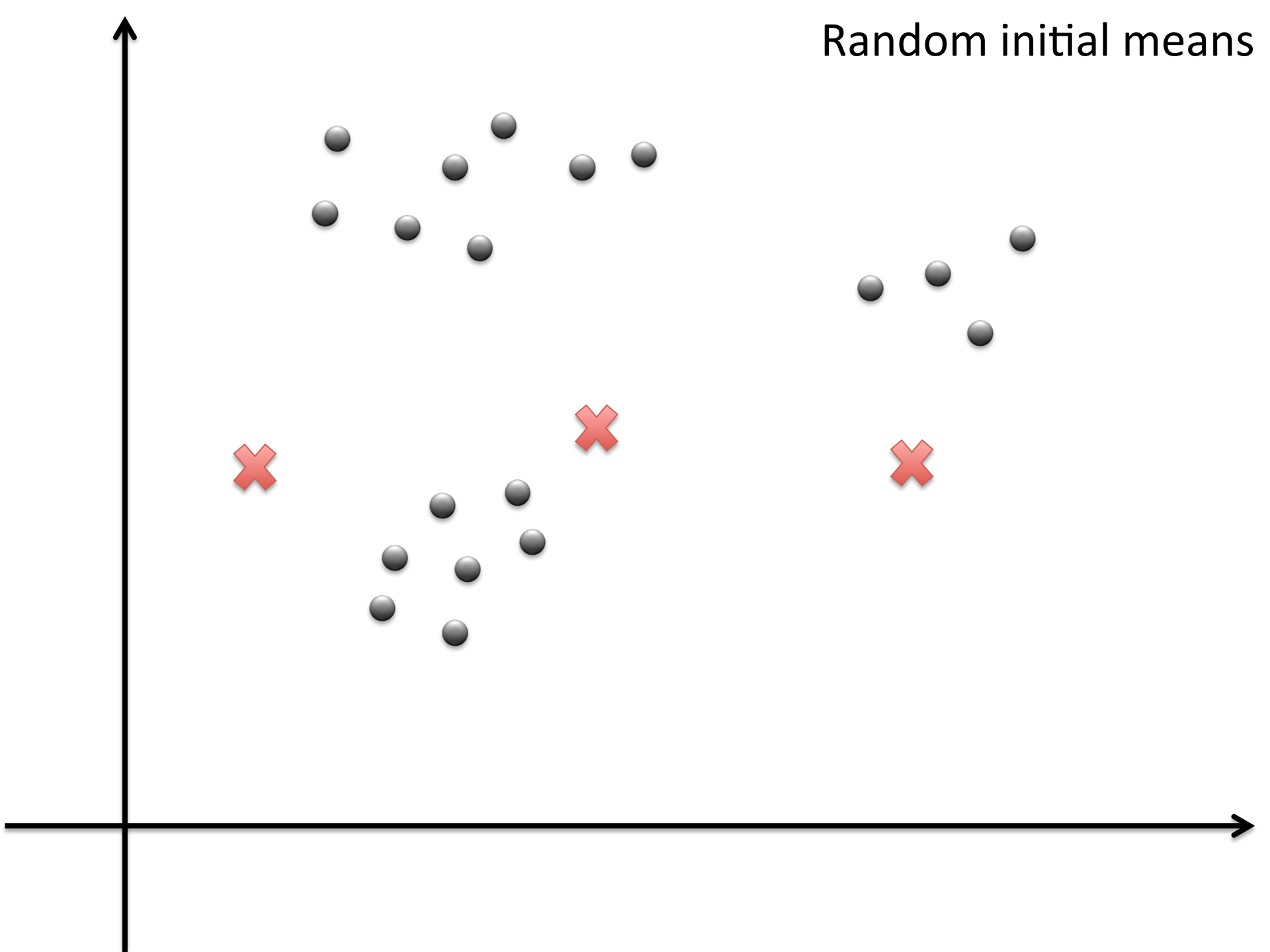
- Trying to find hidden structure in unlabeled data.
- Clustering
  - Group similar data points in “clusters”
- k-means
- Mixture models
- hierarchical clustering

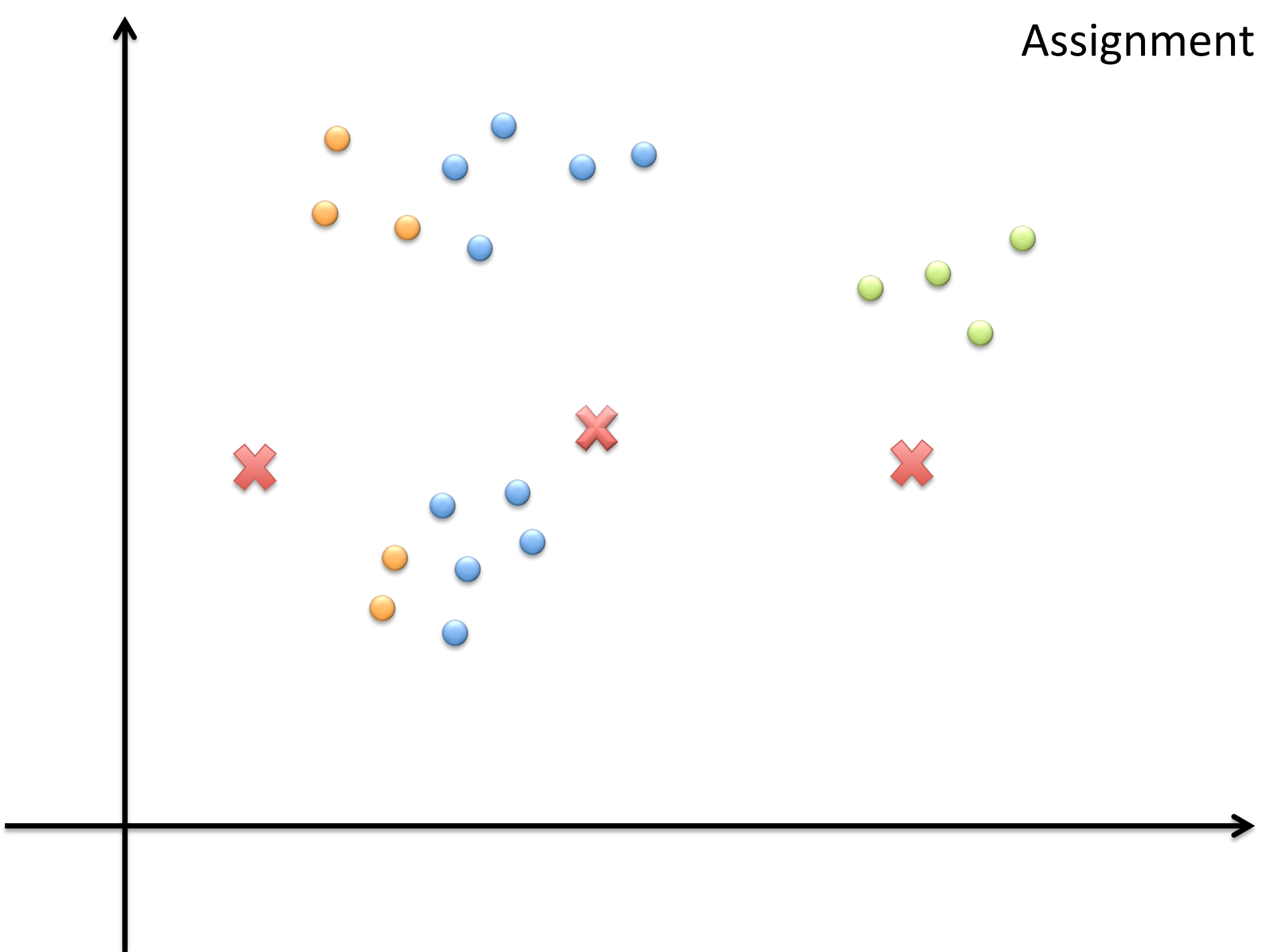


Unsupervised Learning

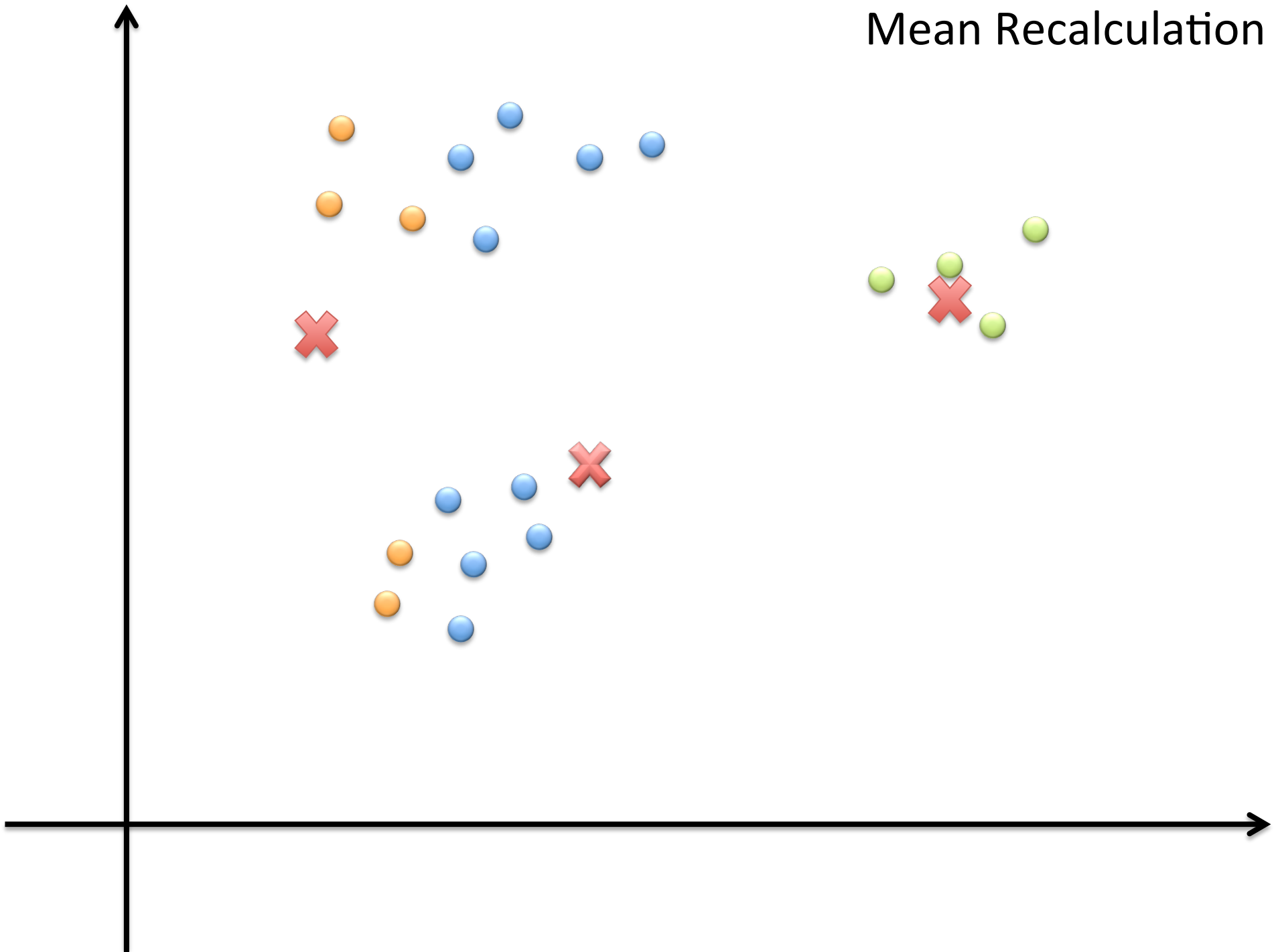
- K-means
- “Birds of a feather flock together”
- Group samples around a “mean”
  - Start with random “means” and assign each point to the nearest “mean”
  - Calculate new “means”
  - Re-assign points to new “means”…



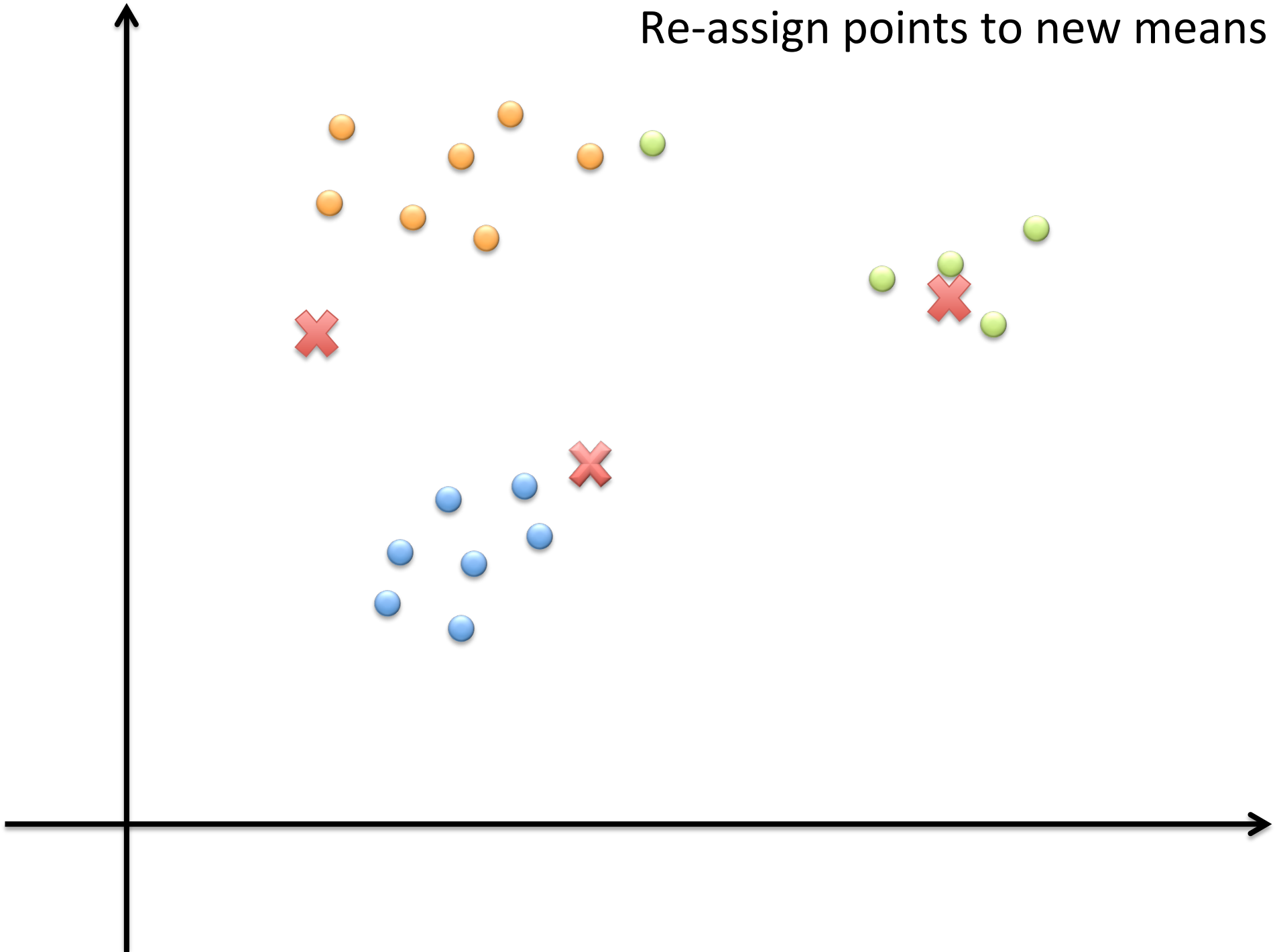

Random initial means


Assignment

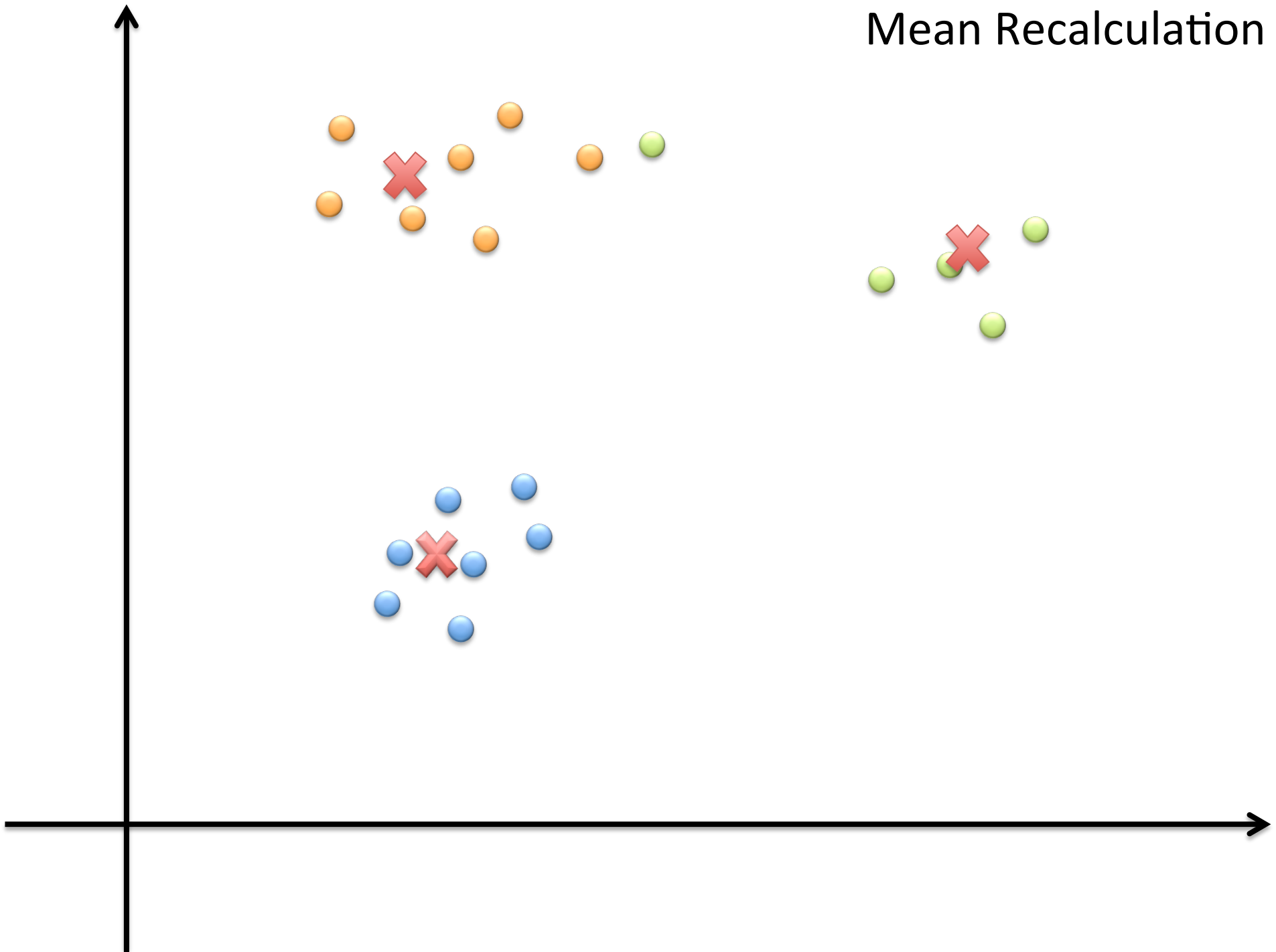
Mean Recalculation



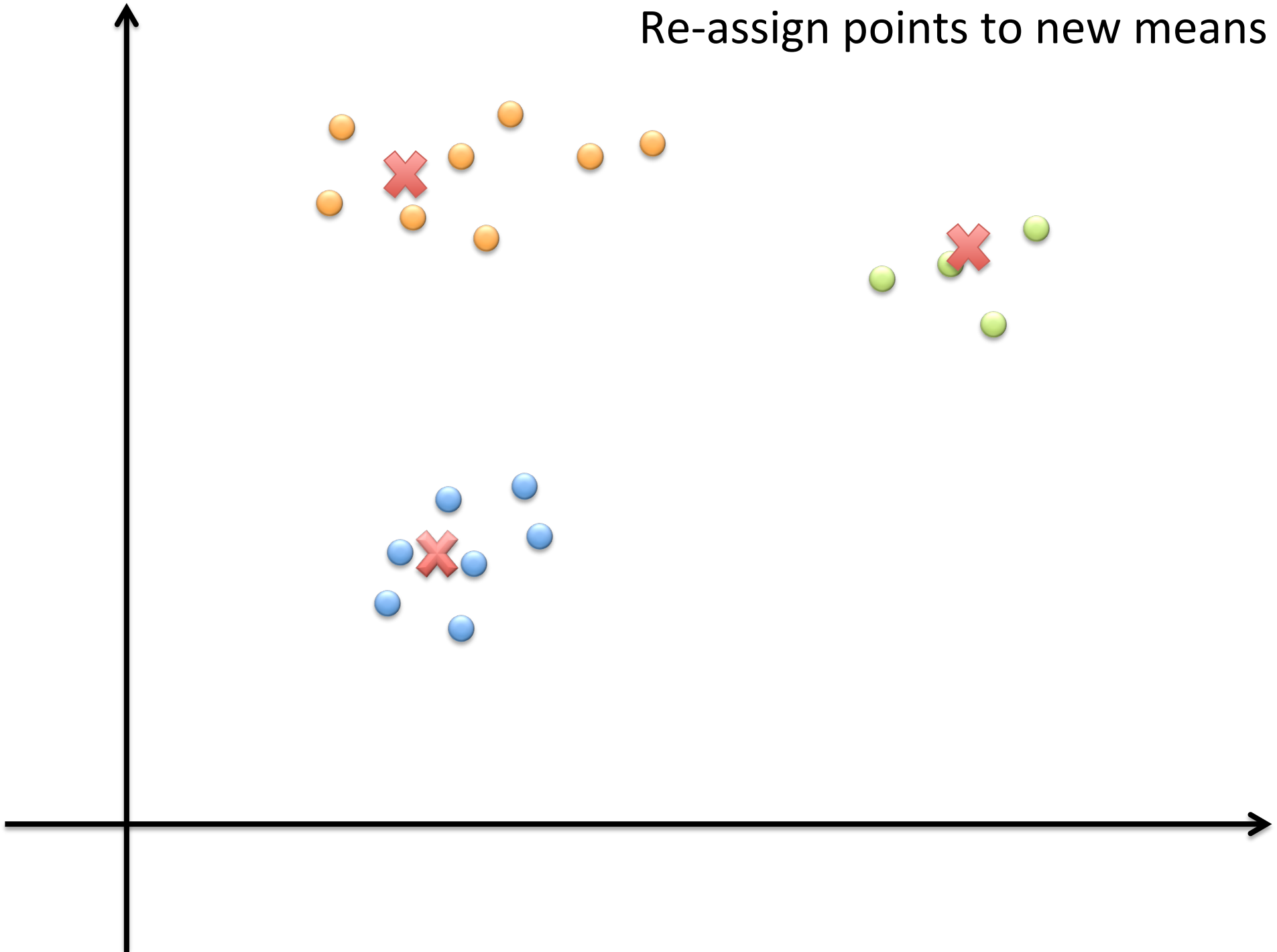
Re-assign points to new means



Mean Recalculation

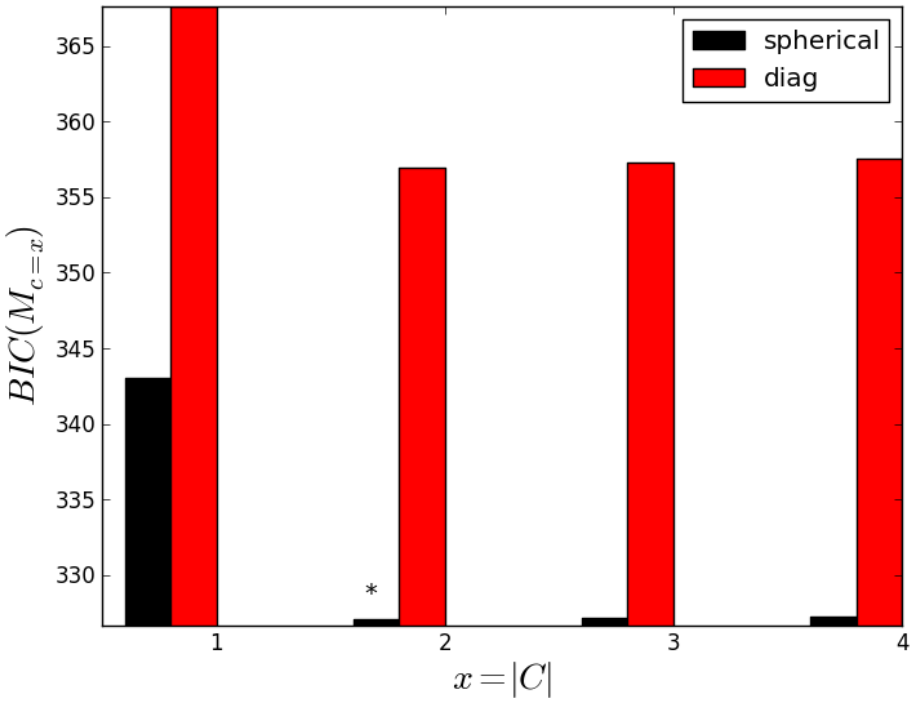


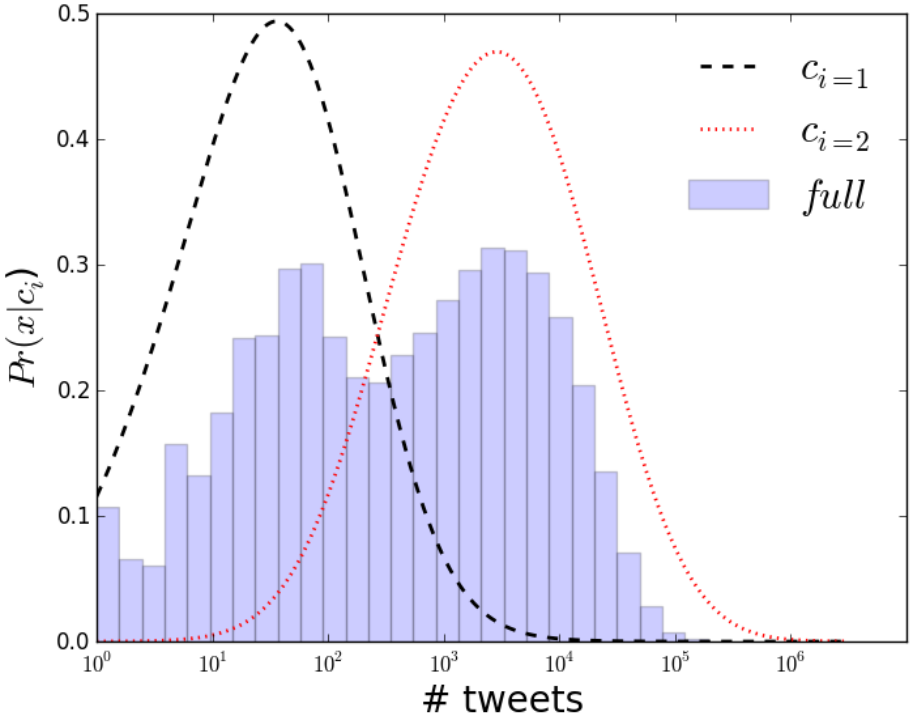
Re-assign points to new means



Unsupervised Learning

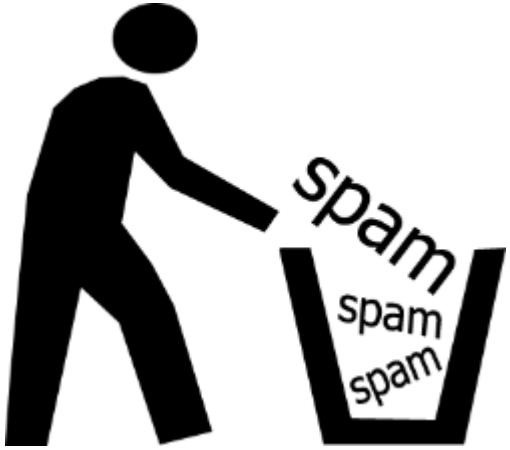
- Mixture Models
  - Presence of subpopulations within an overall population





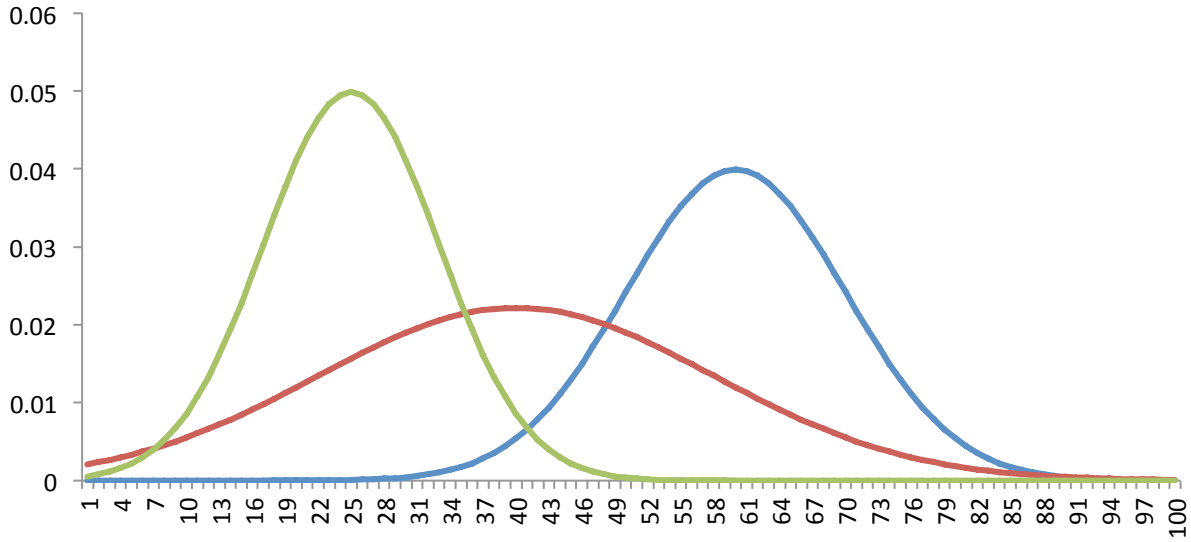
Anomaly Detection

- Detect abnormal data
- Whitelisting good behavior



Anomaly Detection

- Statistical Anomaly Detection

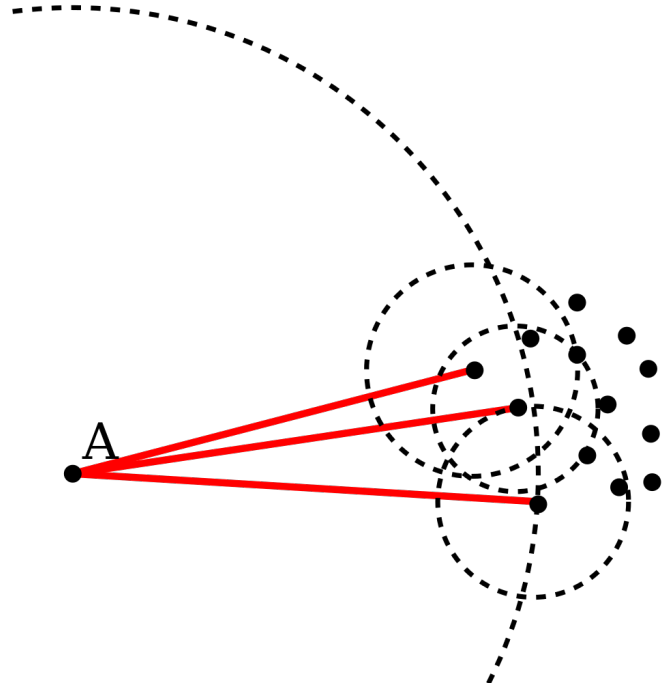


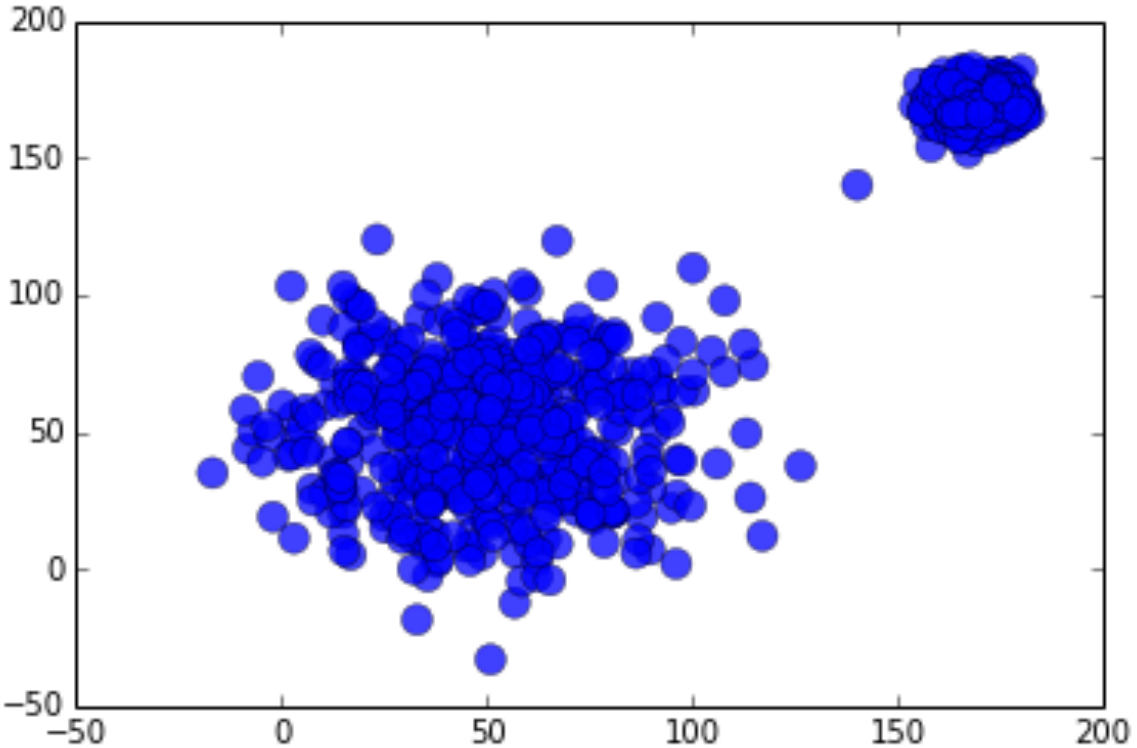
#	X_1	X_2	…	X_n
1	12	8	…	48
2	14	7	…	52
3	16	8	…	57
4	11	6	…	53
…	…	…	…	…
m	13	7	…	56

$$P(x) = p(x_1, \alpha_1, \mu_1)\, p(x_2, \alpha_2, \mu_2) \ldots p(x_n, \alpha_n, \mu_n)$$

Anomaly Detection

- Local Outlier Factor (LOF)
  - Local density
  - Locality is given by nearest neighbors





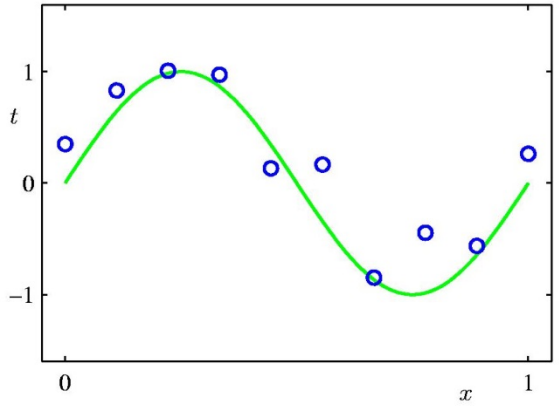
How to learn (well)

Initial critical obstacles

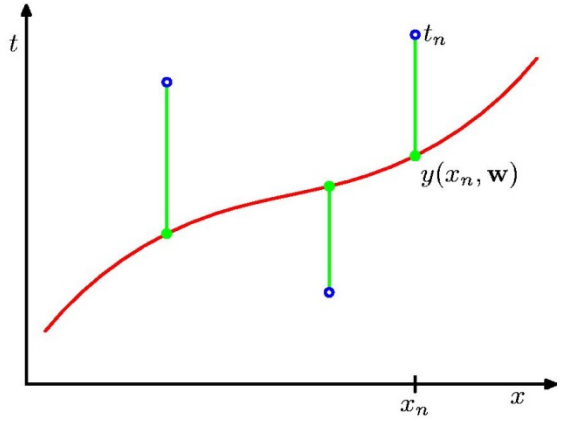
- Choosing Features:
  - By domain experts knowledge and expertise
  - Or by Feature extraction algorithms
- Choosing Parameters, for example:
  - Degree of the regression equation
  - SVM parameters (c, gamma)
  - Number of clusters
    - K-mean (pre)
    - Agglomerative (post)

A simple example: Fitting a polynomial

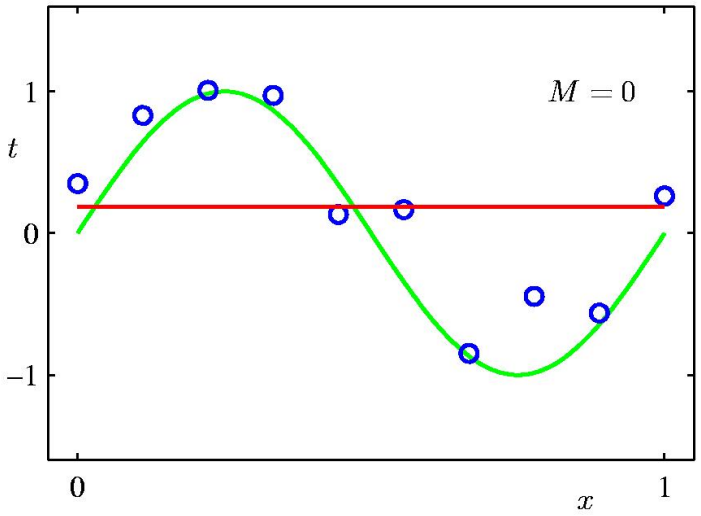
- The green curve is the true function (which is not a polynomial)

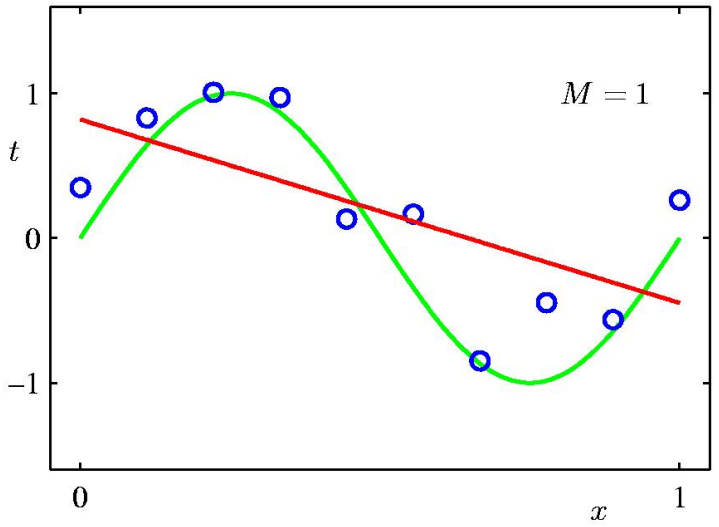


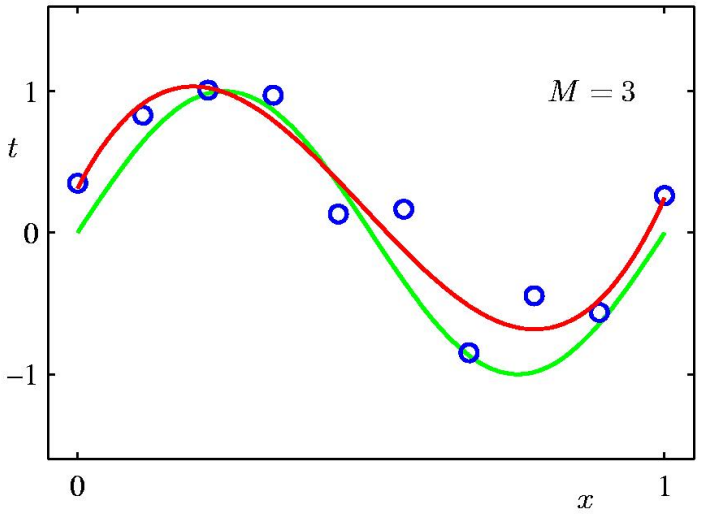
- We will use a loss function that measures the squared error in the prediction of y(x) from x.

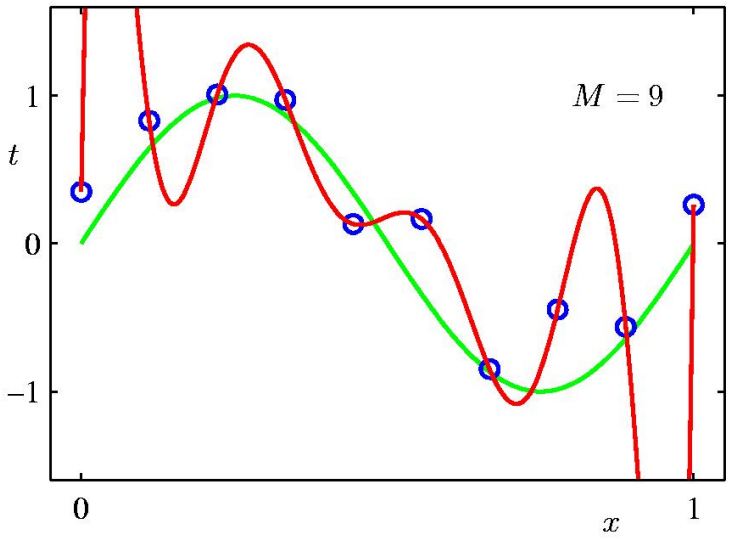


Some fits to the data: which is best?

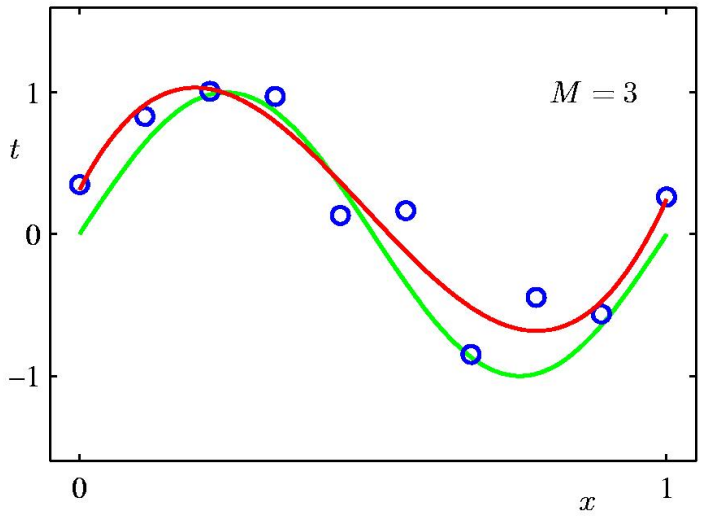


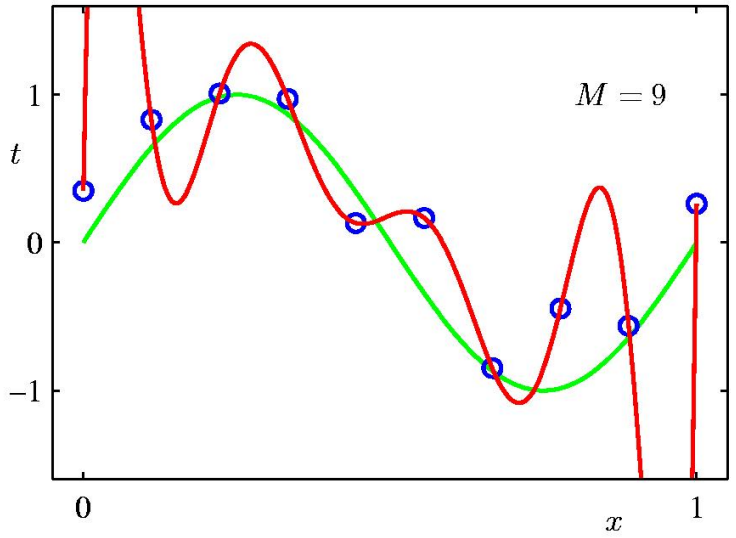






which is best?

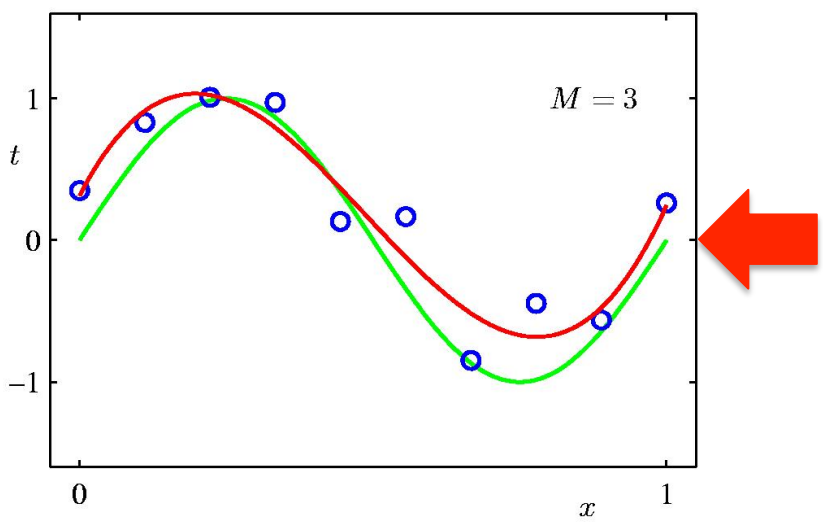


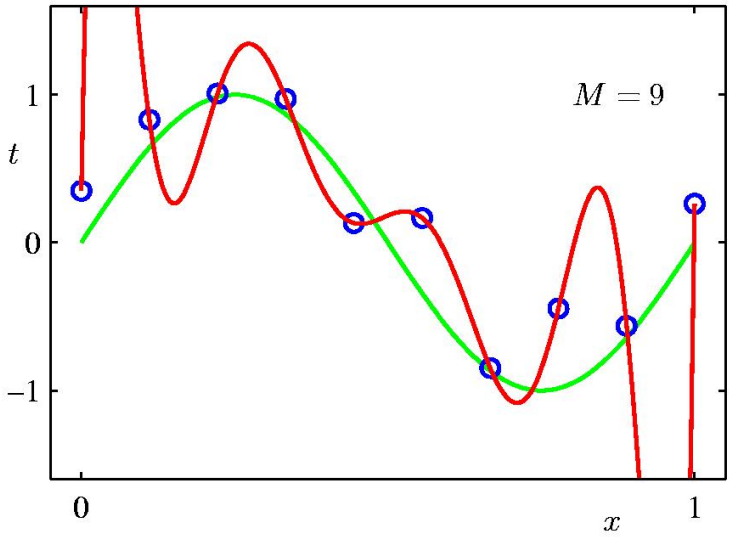


Occam's Razor

- **"The simplest explanation for some phenomenon is more likely to be accurate than more complicated explanations."**

which is best?

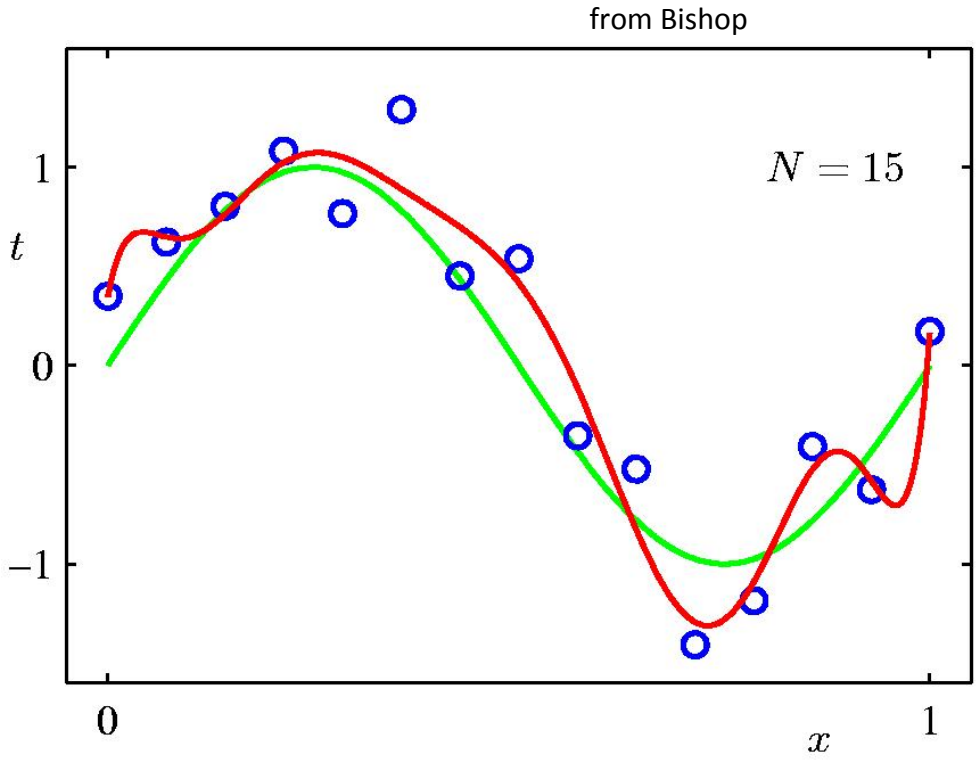




A simple way to reduce model complexity

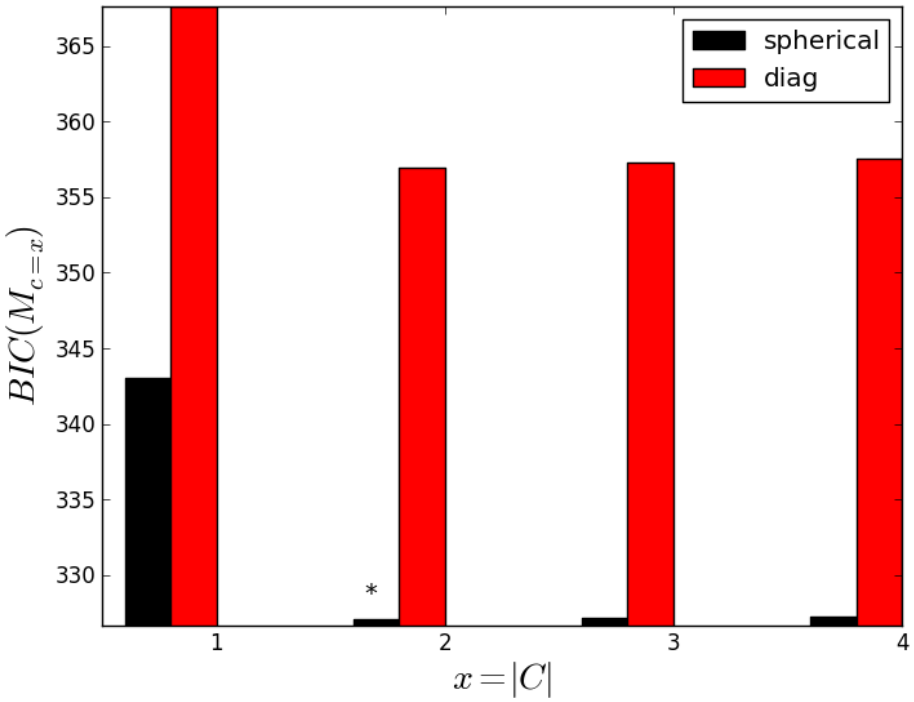
Add penalties

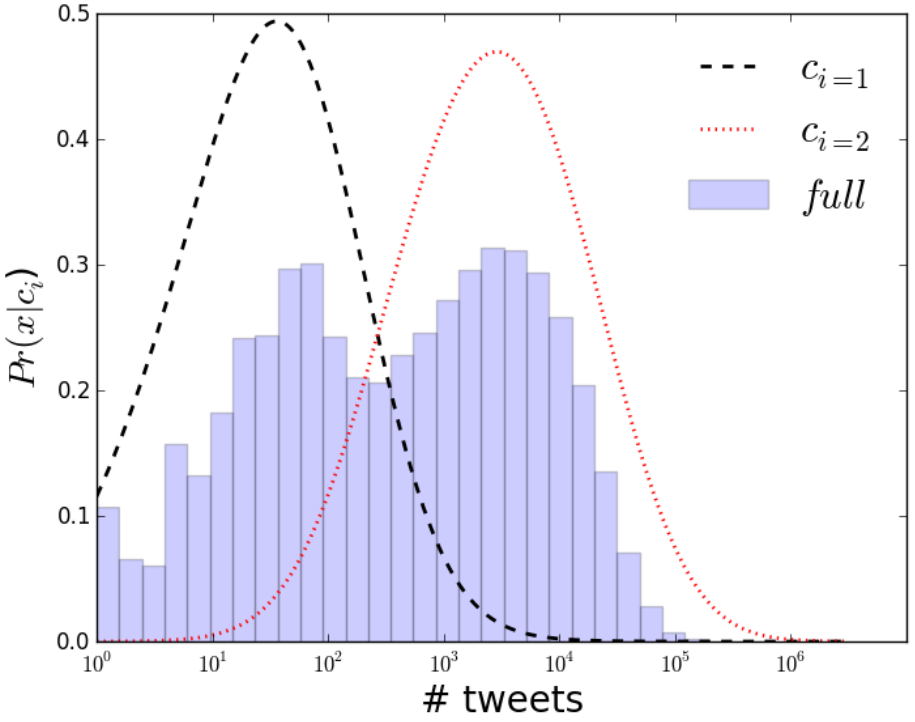
from Bishop



Example

- Mixture Models





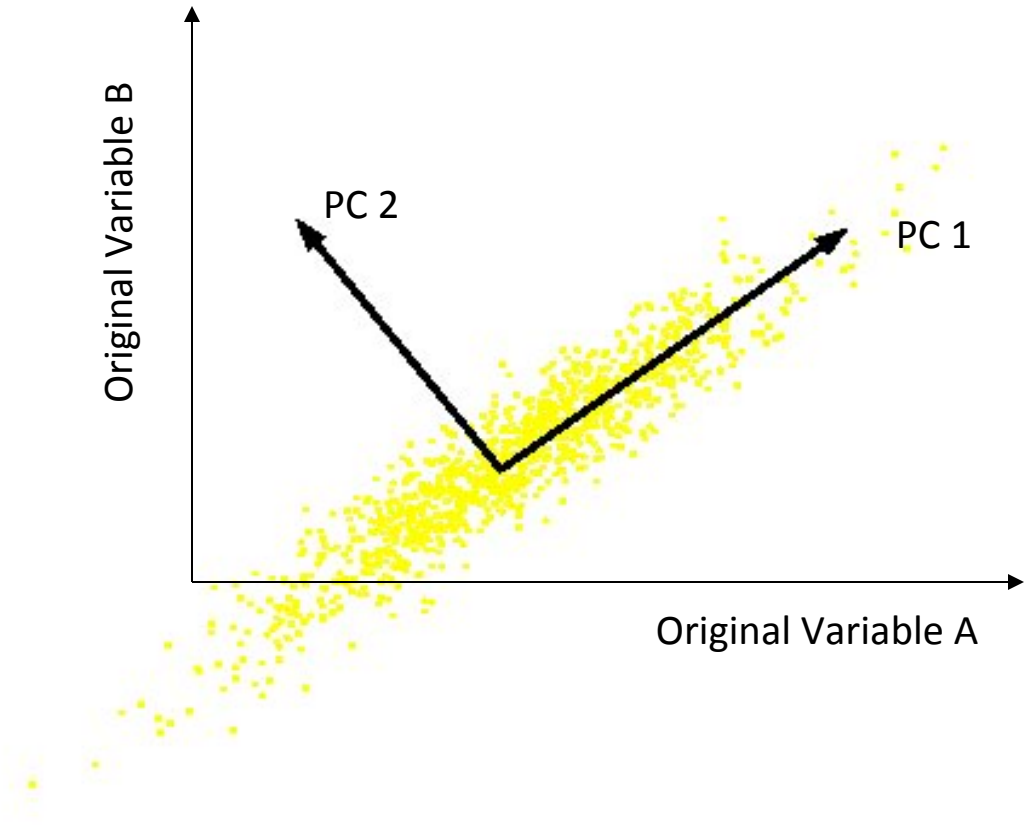
Using a validation set

- Divide the total dataset into three subsets:
  - Training data: for learning the parameters of the model.
  - Validation data for deciding what type of model and what amount of regularization works best.
  - Test data is used to get a final, unbiased estimate of how well the algorithm works. We expect this estimate to be worse than on the validation data.

PCA

- Data visualization
- Noise reduction
- Data reduction
- Can help with the curse of dimensionality
- And more

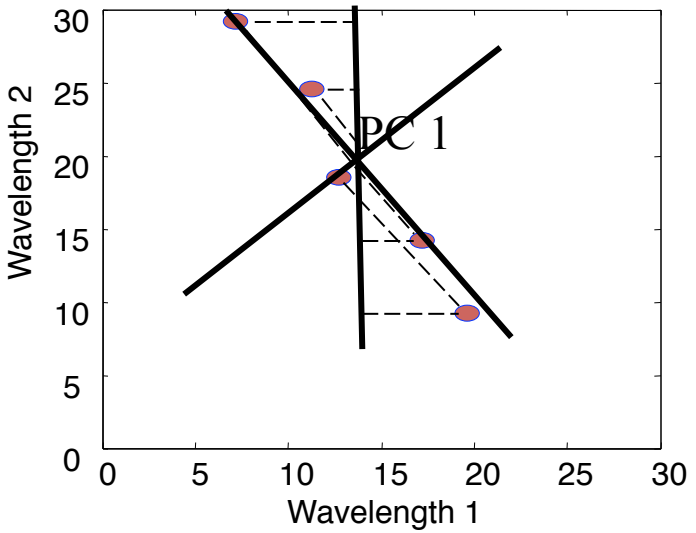
What are PCA



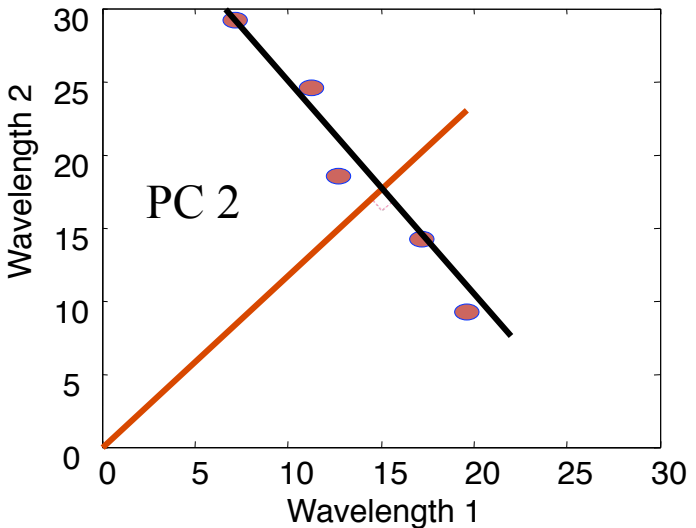
- Orthogonal directions of greatest variance in data

Principal Components

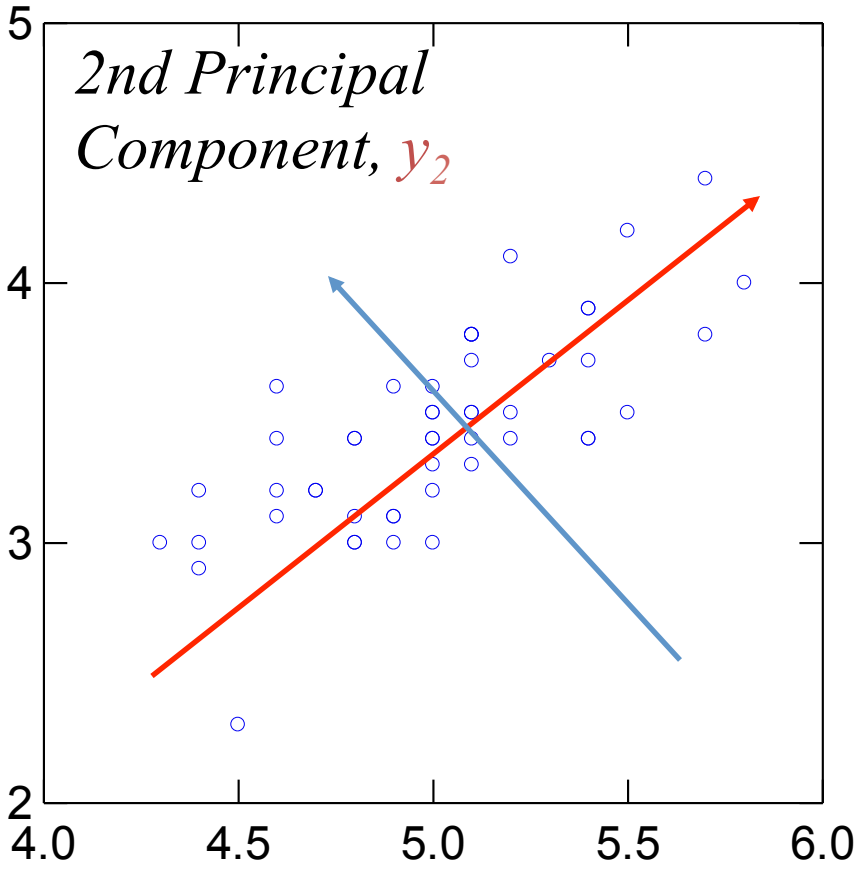
- First PC is direction of maximum variance from origin



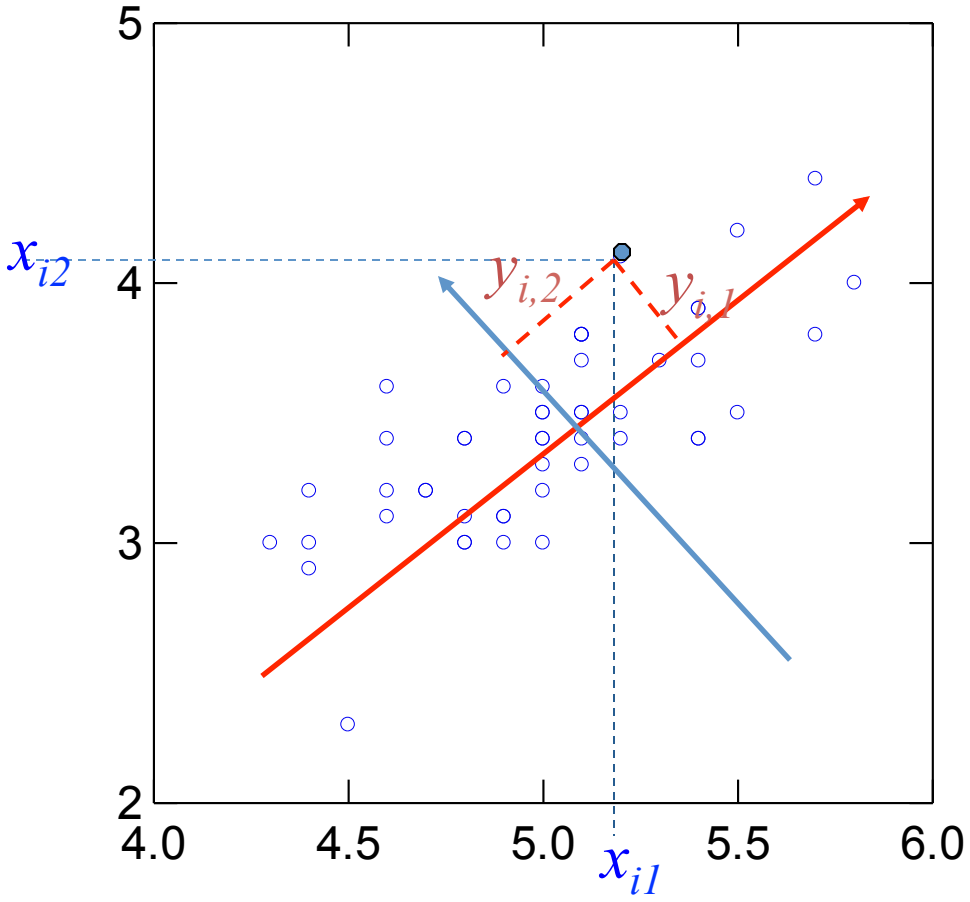
- Subsequent PCs are orthogonal to 1st PC and describe maximum residual variance



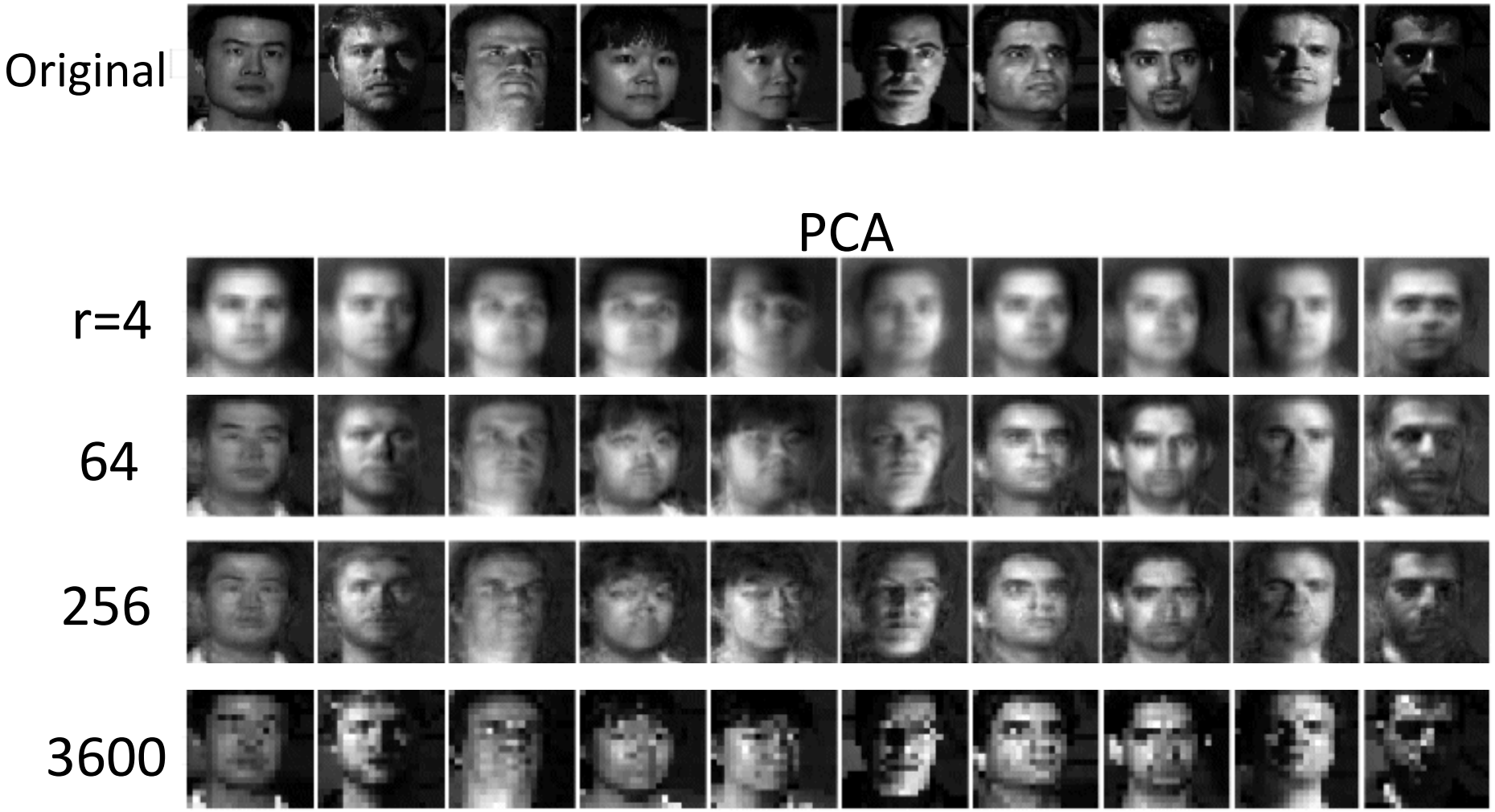
Principal Components



Principal Components



PCA on real example



Thank you

- Make sure you have python and IPython notebook
  - Easiest way to get this running http://continuum.io/downloads

- Download the tutorial IPython notebook http://www.amaatouq.com/notebook.zip