Introduction to Machine Learning

Samrachana Adhikari
Taman Upadhaya

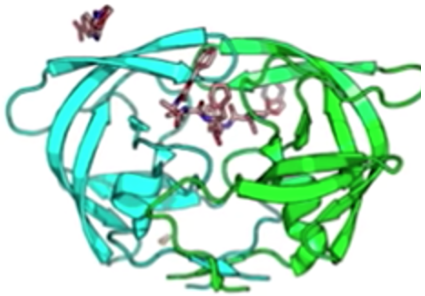
December 22, 2018

State of the Art Applications of Machine Learning

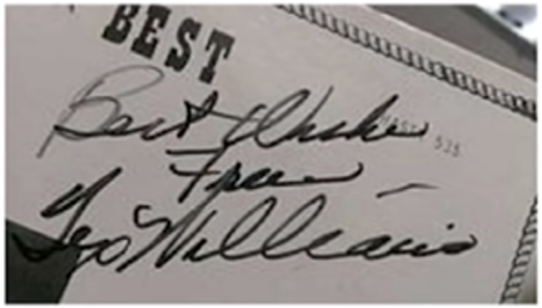
AlphaGo

NETFLIX

Recommendation systems



Drug discovery



Character recognition

2σ TWO SIGMA

Hedge fund stock predictions

Voice assistants

Assisted driving

Face detection/recognition

Cancer diagnosis

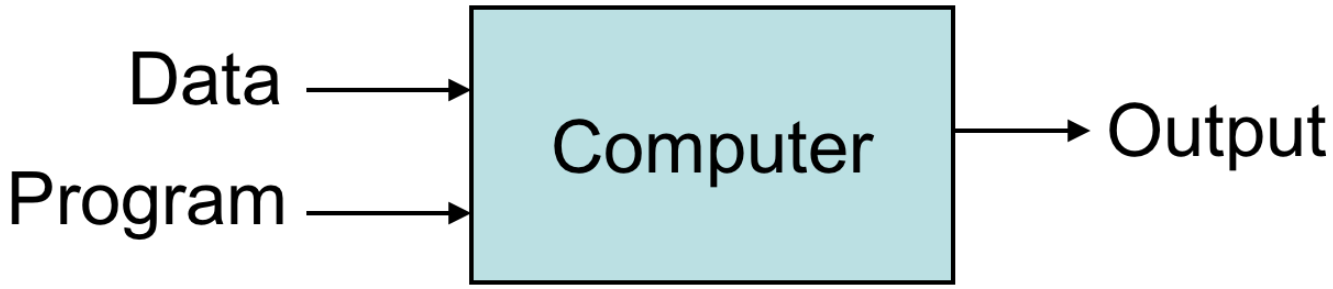
What is in This Lecture Note

- Introduction
  - examples and application
- Supervised learning
  - Regression
  - classification
- Unsupervised learning
  - Clustering
  - ...
- Evaluation
  - Performance metrics
  - Bias-variance trade-off/ Over-fitting and under-fitting
  - Regularization methods
  - Cross-validation

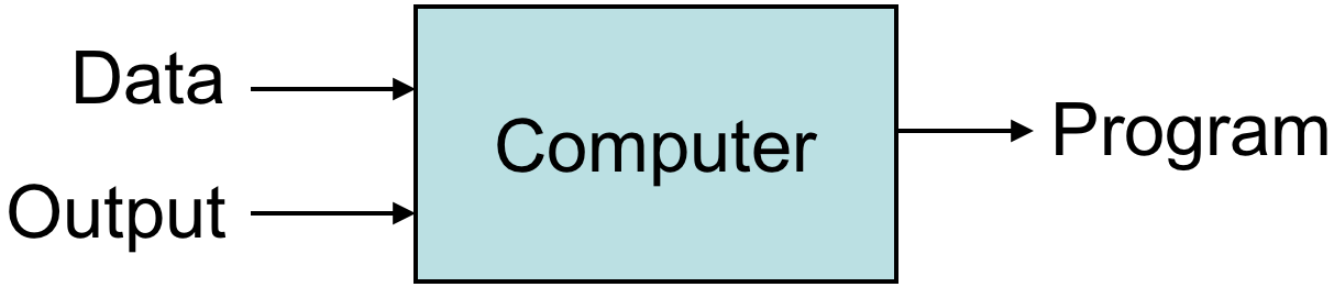
Introduction

What is Machine Learning

Traditional Programming



Machine Learning



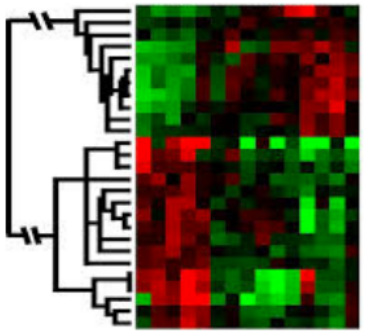
"Field of study that gives computers the ability to learn without being explicitly programmed"– Arthur Samuel 1959

When Do We Use Machine Learning?

ML is used when:

- Human expertise does not exist (navigating on Mars)
- Models must be customized (personalized medicine)
- Models are based on huge amounts of data (genomics)
- ...



Sample Applications

ML is used in:

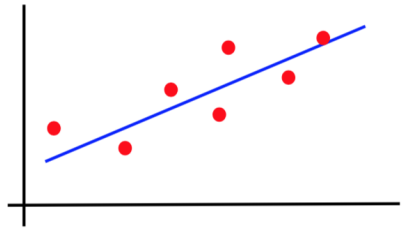
- Web search
- Computational biology
- Finance
- E-commerce
- Space exploration
- Robotics
- Information extraction
- Social networks
- Debugging software
- Your favorite area ...

Basic Paradigm

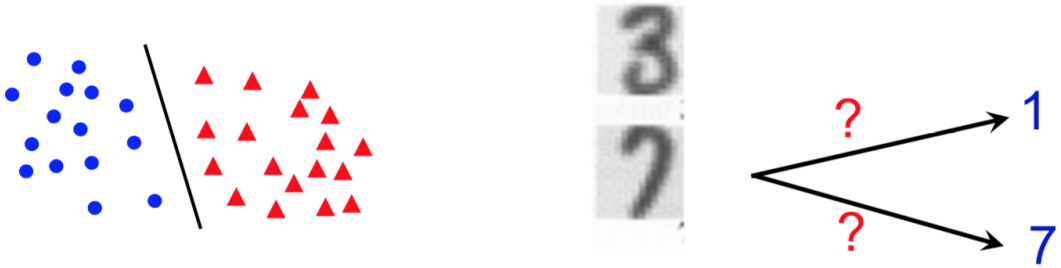
- **Training data:** set of example for observation
- **Testing data:** unseen set of example for prediction
- Variations on paradigm
  - Supervised: given a set of feature/label pairs, find a rules that predicts the label associated with a previously unseen input
  - Unsupervised: given a set of features vector (without label) group them into "natural cluster" (or create labels for groups)

Three canonical learning problems

1. Regression-supervised
   estimate parameters, e.g. of weight vs height

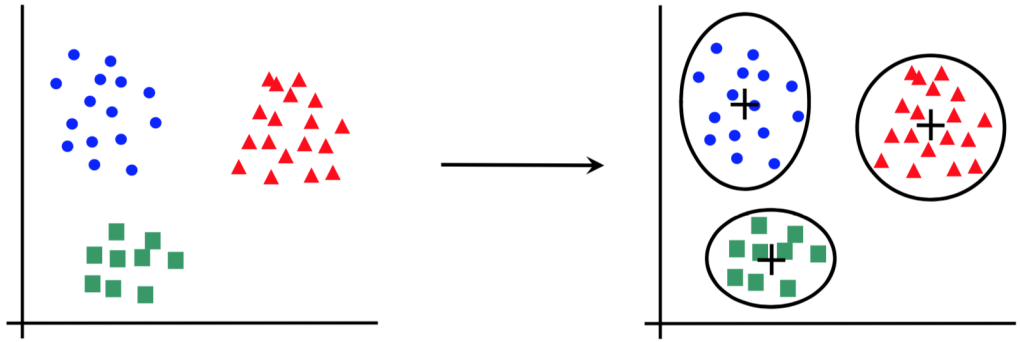


2. Classification-supervised
   estimate class, e.g. handwritten digit classification

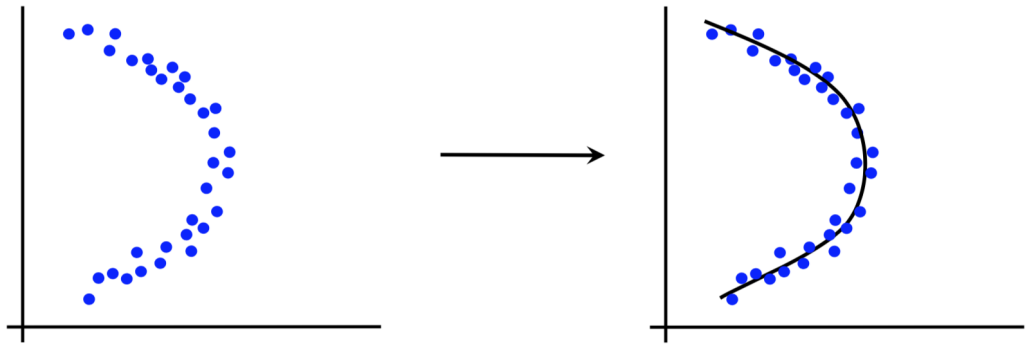


Three canonical learning problems

3. Unsupervised learning model the data
   - clustering



   - dimensionality reduction



Three components of ML

1. Representation
   - Regression model
   - Neural networks
   - Decision Tree
   - Support Vector Machine, ...
2. Evaluation
   - Accuracy
   - Squared error
   - Precision and recall
   - Posterior probability
   - Entropy, ...
3. Optimization
   - Combinatorial optimization, e.g. Greedy search
   - Convex optimization, e.g Gradient descent
   - Constrained optimization, e.g Linear programming

Three components of ML

1. Representation
   - Regression model
   - Neural networks
   - Decision Tree
   - Support Vector Machine, ...
2. Evaluation
   - Accuracy
   - Squared error
   - Precision and recall
   - Posterior probability
   - Entropy, ...
3. Optimization
   - Combinatorial optimization, e.g. Greedy search
   - Convex optimization, e.g Gradient descent
   - Constrained optimization, e.g Linear programming

Three components of ML

1. Representation
   - Regression model
   - Neural networks
   - Decision Tree
   - Support Vector Machine, ...
2. Evaluation
   - Accuracy
   - Squared error
   - Precision and recall
   - Posterior probability
   - Entropy, ...
3. Optimization
   - Combinatorial optimization, e.g. Greedy search
   - Convex optimization, e.g Gradient descent
   - Constrained optimization, e.g Linear programming

Three components of ML

1. Representation
   - Regression model
   - Neural networks
   - Decision Tree
   - Support Vector Machine, ...
2. Evaluation
   - Accuracy
   - Squared error
   - Precision and recall
   - Posterior probability
   - Entropy, ...
3. Optimization
   - Combinatorial optimization, e.g. Greedy search
   - Convex optimization, e.g Gradient descent
   - Constrained optimization, e.g Linear programming

Supervised learning

Supervised learning: Motivation

1. Predict weight from gender, height, age, ...
2. Predict Google stock price today from Google, Yahoo, Microsoft prices yesterday
3. Predict each pixel intensity in robots current camera image, from previous image and previous action
4. **Discuss other examples in small group**

Supervised Learning

Formal motivation

Given a set of measurements (x_i, y_i) for, $i = 1, ..., N$ subjects, known as the **training data**, we want to construct a prediction rule for an arbitrary input x_0.

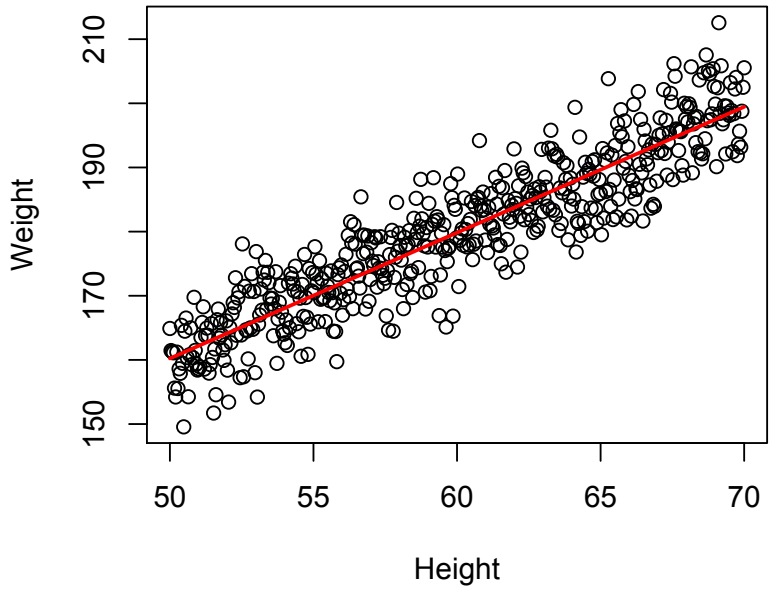
- Xs are often called **predictors or covariates**.
- Y is the **outcome variable**; can be continuous, categorical or ordinal.
- y is used to represent a realization of the random variable Y.

Supervised learning: Linear Regression

Model continuous outcome as a **Linear** function of predictors

$$
\begin{aligned}
Y &= \sum_{k=1}^{p} X^k \beta_k + \epsilon = X\beta + \epsilon \\
E(\epsilon) &= 0
\end{aligned}
$$

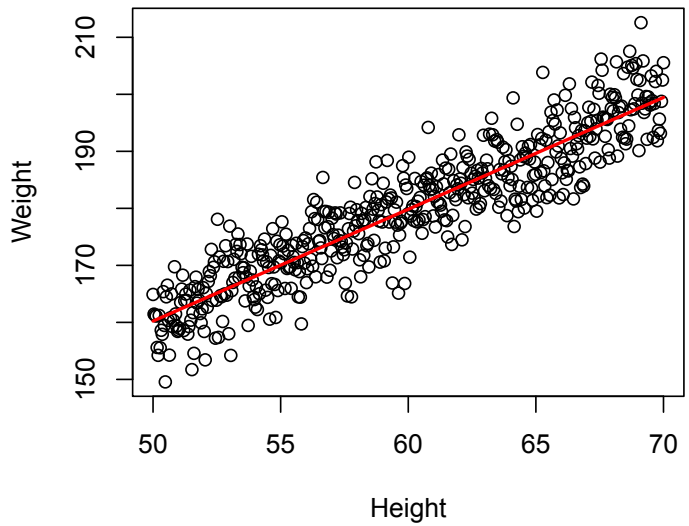
Y: A **continuous outcome**
$X = [X_1, \ldots, X_p]$: matrix of p **predictors or features**
β: a vector of **coefficients**
ϵ is sometimes called **noise or error term**



How to predict Y at an arbitrary x_0?

1. **Training:** Define objective function to estimate β using training data
2. **Prediction:** Predict outcome at x_0 as

$$E(Y|x_0) = \hat{Y}|x_0 = \hat{\beta}x_0$$



Linear Regression Estimation: Maximum Likelihood Estimate

MLE β

$$\hat{\beta} = \text{argmax}_\beta[\text{likelihood}(\beta)] \equiv \text{argmin}_\beta[\text{-likelihood}(\beta)]$$

Let for each i, $\epsilon_i \sim \text{Normal}(0, \sigma^2)$

$$\Rightarrow Y_i|X_i, \beta \sim \textbf{Normal}(X_i\beta, \sigma^2)$$

$$\text{Likelihood}(\beta) = p(Y|X, \beta) = \prod_{i=1}^{n} p(Y_i|X_i, \beta)$$

- We are assuming each Y_i is independent and identically distributed
- Log of likelihood is used in practice

Linear Regression Estimation: Maximum Likelihood Estimate

MLE β

$$\hat{\beta} = \text{argmax}_\beta[\text{likelihood}(\beta)] \equiv \text{argmin}_\beta[\text{-likelihood}(\beta)]$$

Let for each i, $\epsilon_i \sim \text{Normal}(0, \sigma^2)$

$$\Rightarrow Y_i|X_i, \beta \sim \textbf{Normal}(X_i\beta, \sigma^2)$$

$$\text{Likelihood}(\beta) = p(Y|X, \beta) = \prod_{i=1}^{n} p(Y_i|X_i, \beta)$$

- We are assuming each Y_i is independent and identically distributed
- Log of likelihood is used in practice

Linear Regression Estimation: Maximum Likelihood Estimate

MLE β

$$\hat{\beta} = \text{argmax}_\beta[\text{likelihood}(\beta)] \equiv \text{argmin}_\beta[\text{-likelihood}(\beta)]$$

Let for each i, $\epsilon_i \sim \text{Normal}(0, \sigma^2)$

$$\Rightarrow Y_i|X_i, \beta \sim \textbf{Normal}(X_i\beta, \sigma^2)$$

$$\text{Likelihood}(\beta) = p(Y|X, \beta) = \prod_{i=1}^{n} p(Y_i|X_i, \beta)$$

- We are assuming each Y_i is independent and identically distributed
- Log of likelihood is used in practice

MLE in Linear Regresssion

- $Y_1, \ldots, Y_n$ are iiid outcomes
- For each i, $\epsilon_i \sim \text{Normal}(0, \sigma^2)$

$$\Rightarrow Y_i|X_i, \beta \sim \mathbf{Normal}(X_i\beta, \sigma^2)$$

$\text{Likelihood}(\beta) = p(Y|X, \beta) = \prod_{i=1}^{n} p(Y_i|X_i, \beta)$

$\text{Log-Likelihood}(\beta) = logp(Y|X, \beta) = \sum_{i=1}^{n} logp(Y_i|X_i, \beta)$

- $logp(Y_i|X_i, \beta, \sigma^2) = log\frac{1}{\sqrt{2\pi}\sigma} + logexp[\frac{-(Y_i - X_i\beta)^2}{2\sigma^2}]$

Discussion

Derive MLE of β in linear regression.

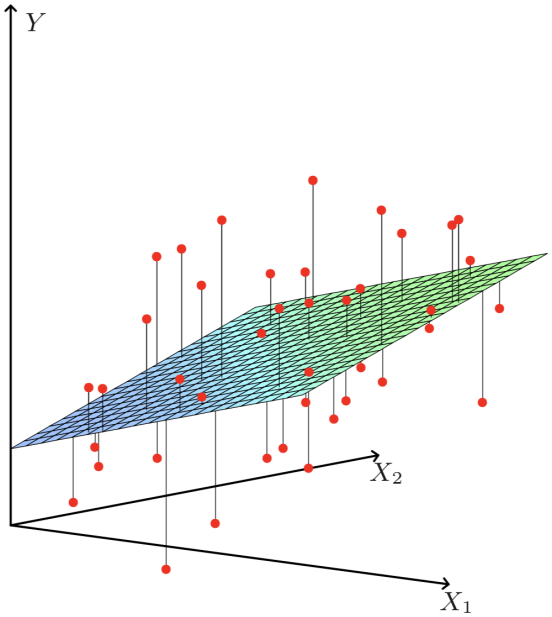
Linear Regression Estimation: MSE Loss

Mean Square Error Loss

$$\hat{\beta} = \text{argmin}_\beta ||Y - X\beta||_2^2 = \text{argmin}_\beta \sum_{i=1}^{n} (Y_i - X_i\beta)^2$$

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

- When $\epsilon \sim \text{Normal}(\mu, \sigma^2)$, minimizing squared error loss is equivalent to maximizing likelihood



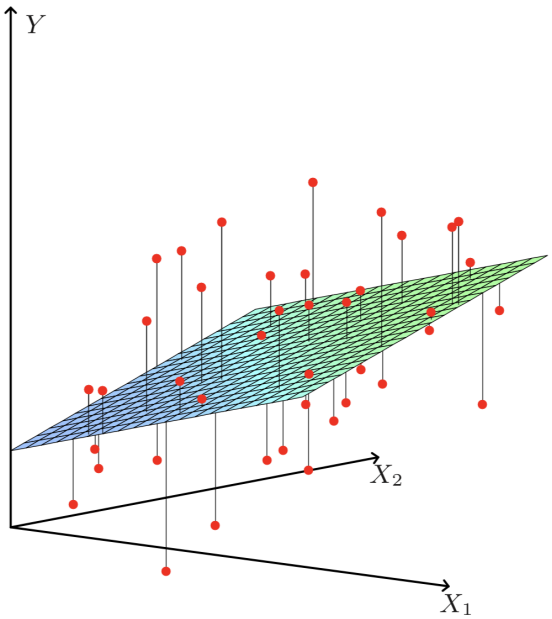
Linear Regression Estimation: MSE Loss

Mean Square Error Loss

$$\hat{\beta} = \text{argmin}_\beta ||Y - X\beta||_2^2 = \text{argmin}_\beta \sum_{i=1}^{n} (Y_i - X_i\beta)^2$$

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

- When $\epsilon \sim \text{Normal}(\mu, \sigma^2)$, minimizing squared error loss is equivalent to maximizing likelihood



Linear Regression Discussion

Pros:

- Interpretability and simplicity
- Solution exists (in most cases)
- Favorable large-sample statistical properties
- Numerous software packages to fit the model

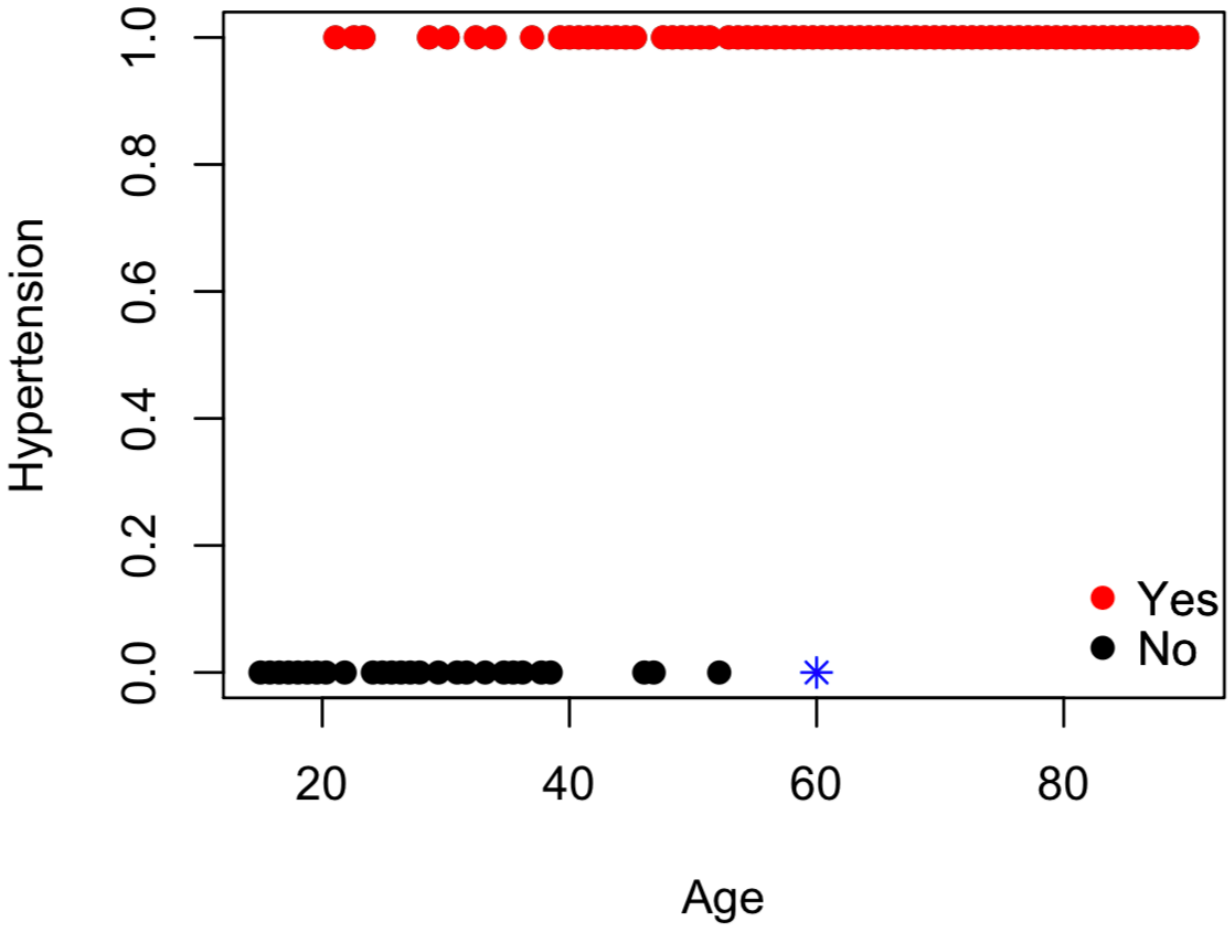
Cons:

- Real world is never that simple!!!
  - **Too restrictive:** Assumes linear relationship between features and outcomes
  - **Big data:** Not suitable when feature space is large ($p >> n$)

Supervised learning: Classification models

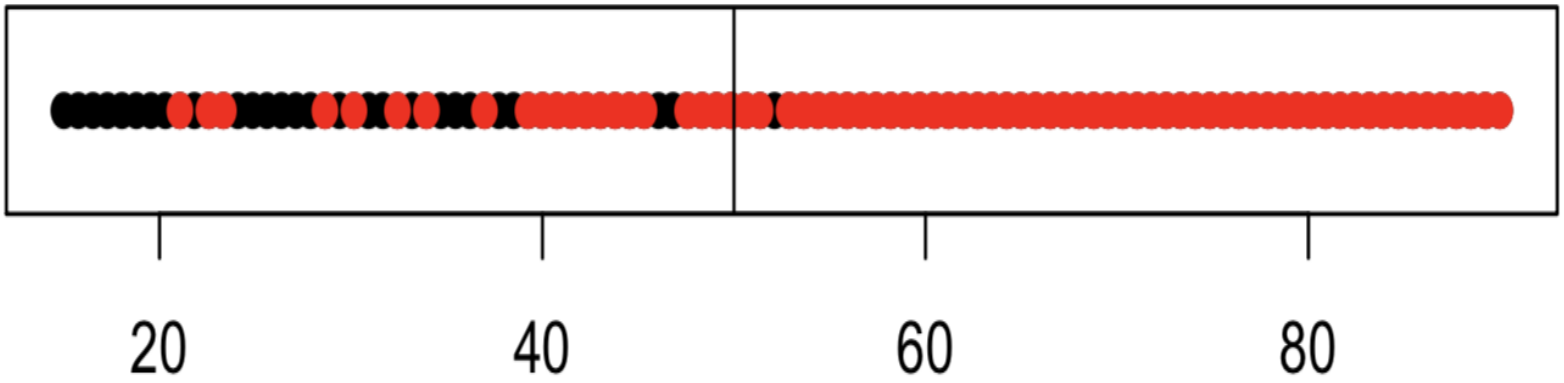
- Outcome is discrete
  - Predict weather, rainy versus sunny, based on past weather.
  - Predict whether a patient will survive after a major surgery using vitals prior to the surgery.
  - Classifying patients into high or low risk of hypertension based on their age.
- Goal: find a rule to correctly classify subjects into classes by learning from the existing data.

Classification: Motivation

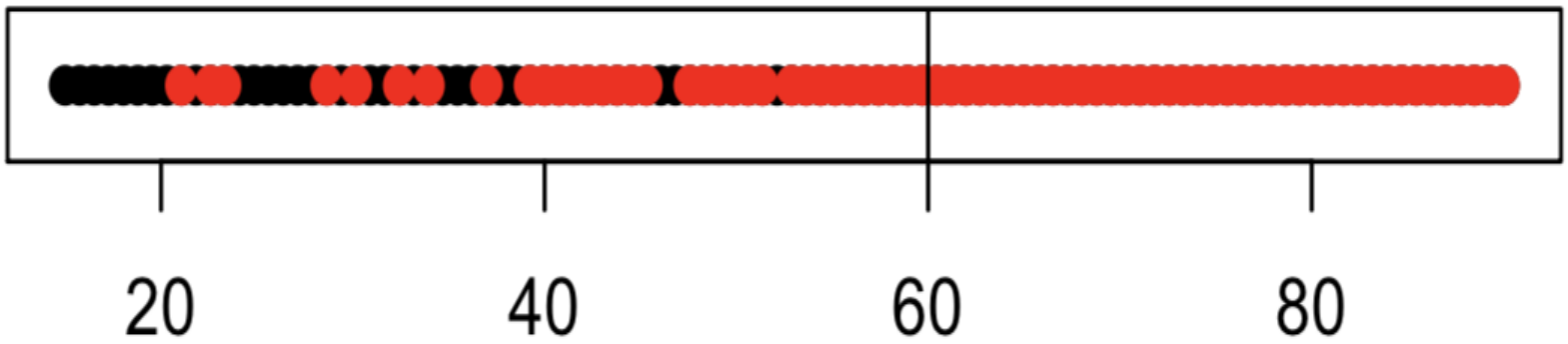


Classification: Motivation

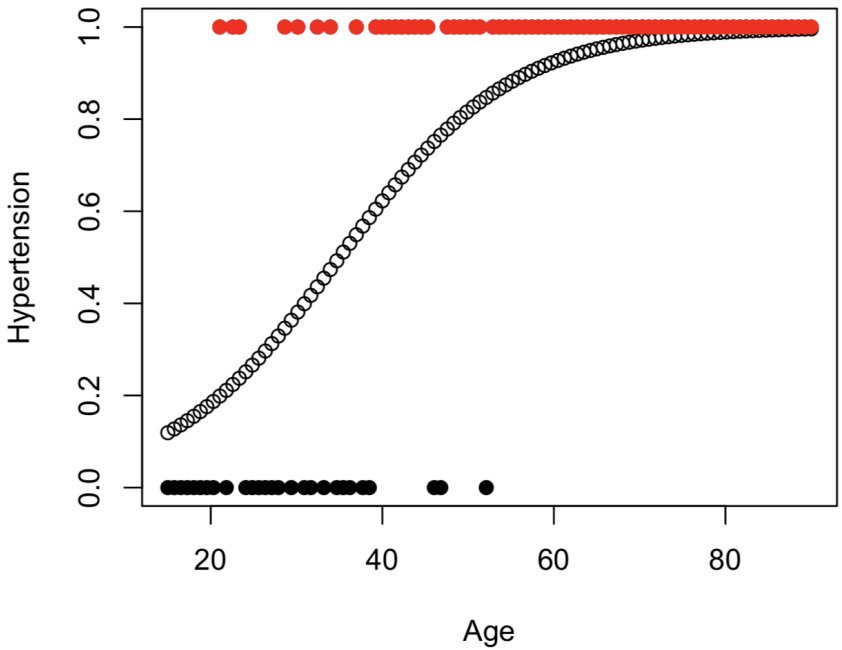
Rule 1



Rule 2



Logistic Regression



$$\log \frac{Pr(Y=1}{Pr(Y=0)} = X\beta$$

Prediction:

$$\hat{p} = \hat{Pr}(Y=1)|x_o = \frac{1}{1+\text{Exp}(-x_0\hat{\beta})}$$

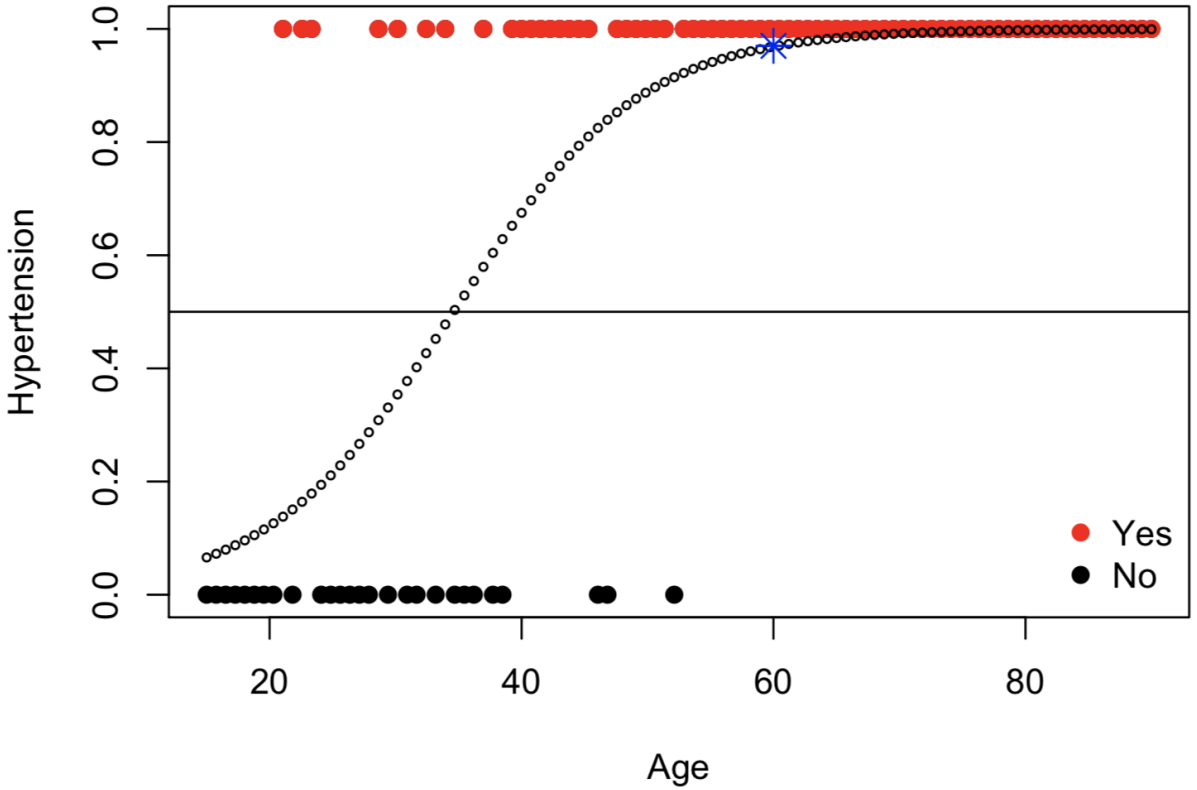
$$\hat{Y} = \mathbf{I}(\hat{p} > 0.5)$$

Decision Boundary: Linear Classifier

- A convenient property of this form is that it leads to a simple **linear** expression for classification.
- To classify any given Y we generally want to assign the value k that maximizes $Pr(Y = k|X)$, for $k = \{0, 1\}$
- Classify $Y = 1$ if

$$\begin{aligned} Pr(Y = 1|X) &> Pr(Y = 0|X) \\ &\equiv 1 > \text{Exp}(\beta X) \equiv 0 > \beta X \end{aligned}$$

- **Classification rule is linear in** X



For 2-classes case, classify $\hat{Y}|x_0 = 1$ if $\hat{Pr}(Y = 1|x_0) > 0.5$.

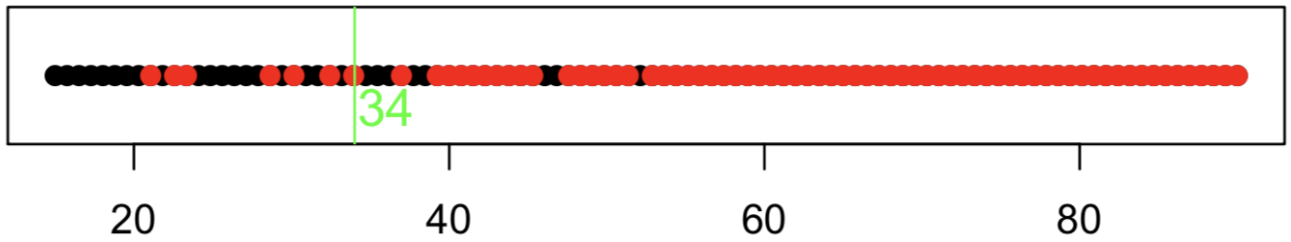


Figure: Linear Classifier

Supervised learning: Regression Models

A general regression model

$$E(Y|X) = f(X)$$

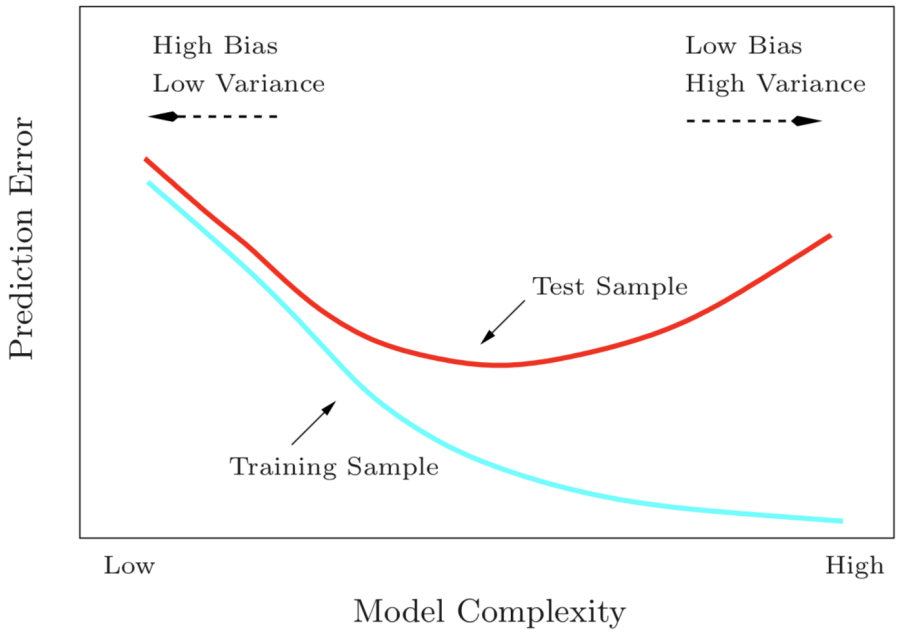
- Many other functional forms on f is possible
- f is often referred to as a **link function** in statistical learning
- Parametric versus non-parametric models: Restrict f to rely on few parameters versus using f in a bigger class of family with no restrictions
- How to decide which f to use?

General Estimation: Minimization of Expected Risk or Loss Function

$$\hat{f} = \arg\min_f E(\text{Prediction loss}(f))$$

General Estimation: Minimization of Expected Risk or Loss Function

- Optimization methods are used to do the minimization.
- Challenging if the loss function is not convex.
- Desired to build a most parsimonious model that minimizes the prediction error.



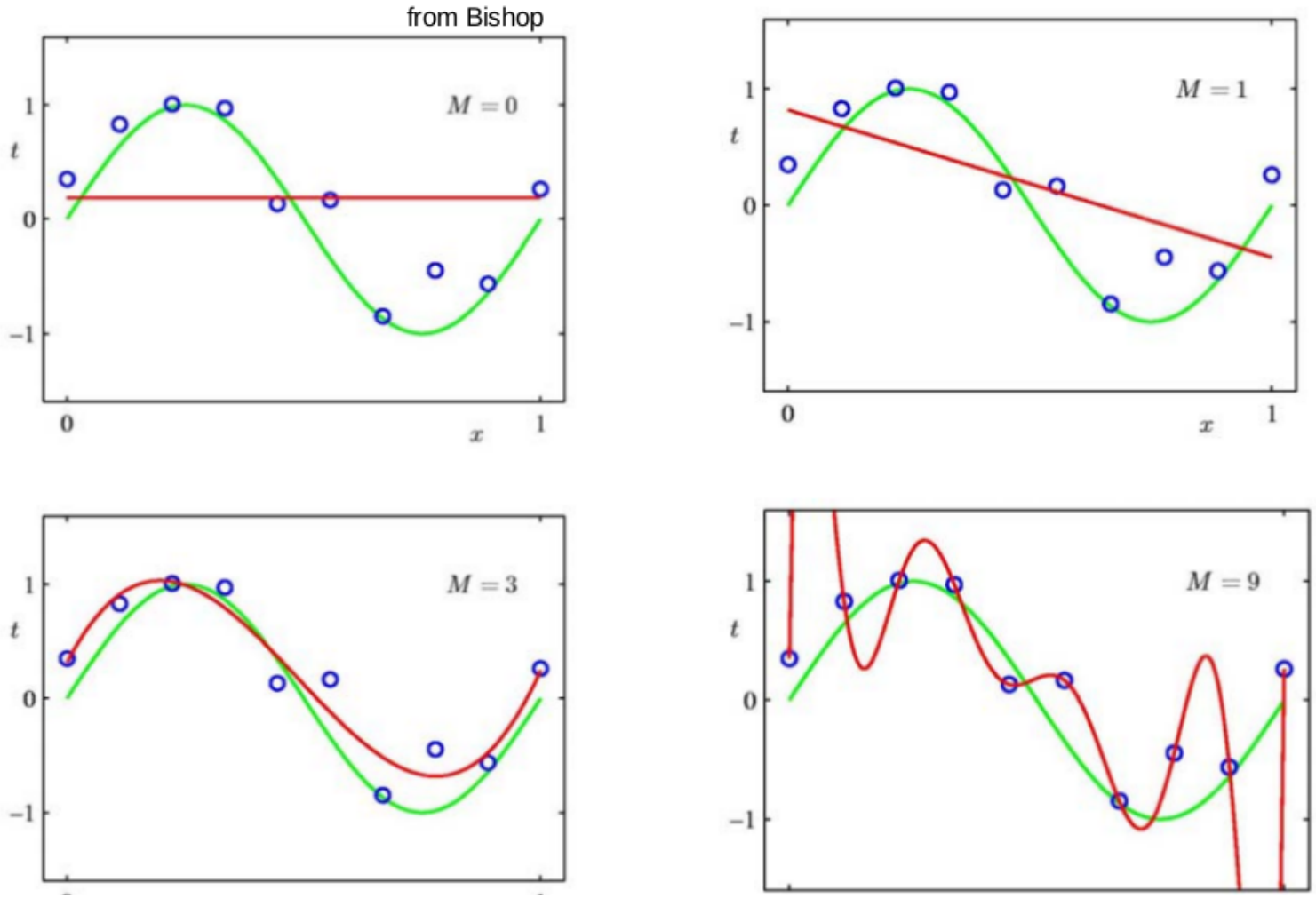
Performance metrics

Truth table (confusion matrix)

		Predicted class P	**Predicted class** N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

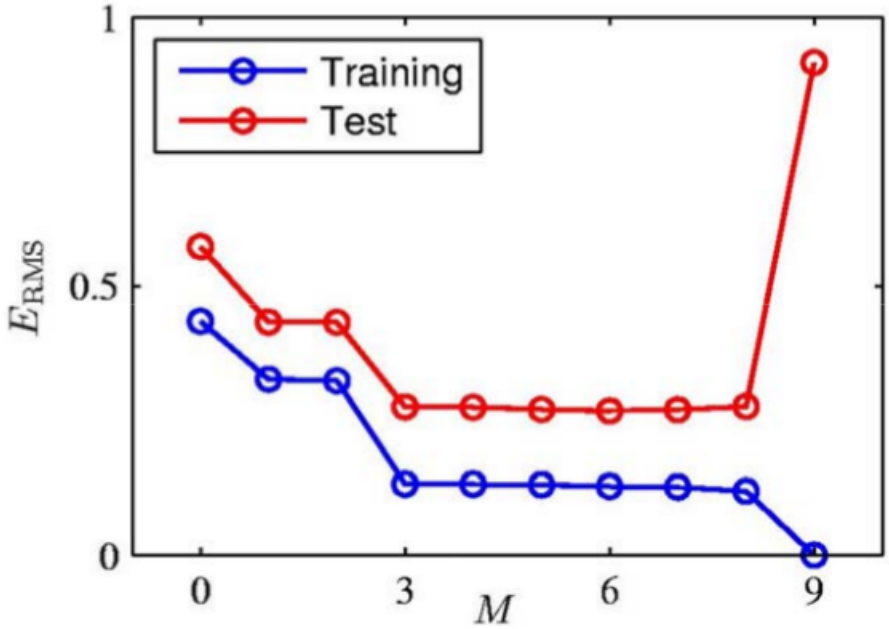
- Accuracy $= \frac{TP+TN}{P+N}$
- Sensitivity $= \frac{TP}{\text{Actual Positives}}$
- Specificity $= \frac{TN}{\text{Actual Negatives}}$
- ...

Some fits to the data: which is best?



Bias-variance trade-off

- test data: a different sample from the same true function



RootMeanSquare (RMS) Error

- training error goes to zero, but test error increases with M

$$\mathrm{RMSE} = \sqrt{(\frac{(Y_i \hat{Y}_i)^2}{n})}$$

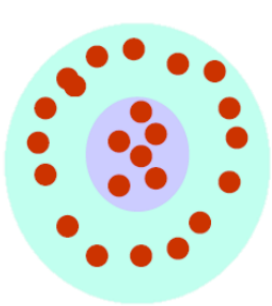
Unsupervised learning

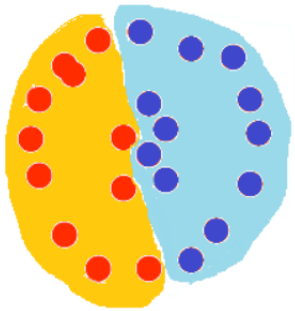
Whats is Data Clustering

- The organization of unlabeled data into similarity groups called clusters.
- A cluster is a collection of data items which are similar between them, and dissimilar to data items in other clusters.

Whats is Data Clustering

- The organization of unlabeled data into similarity groups called clusters.
- A cluster is a collection of data items which are similar between them, and dissimilar to data items in other clusters.

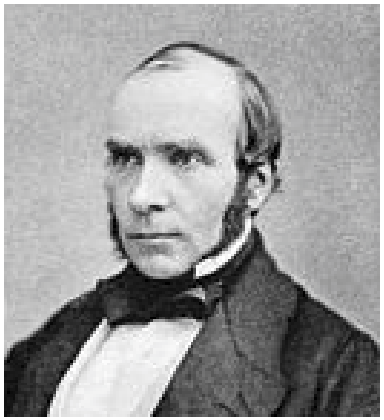




Can be done in many ways

Historic application of clustering

- John Snow, a London physician plotted the location of cholera deaths on a map during an outbreak in the 1855.
- The location indicate that cases were clustered around certain intersections where there were polluted wells – thus exposing both the problem and solution.



John Snow and famous cholera map

What do we need for clustering?

1. Proximity measure:
   - similarity measure $S(X_i, X_k)$: large if X_i, X_k are similar
   - dissimilarity (or distance) measure $D(X_i, X_k)$: small if X_i, X_k are similar

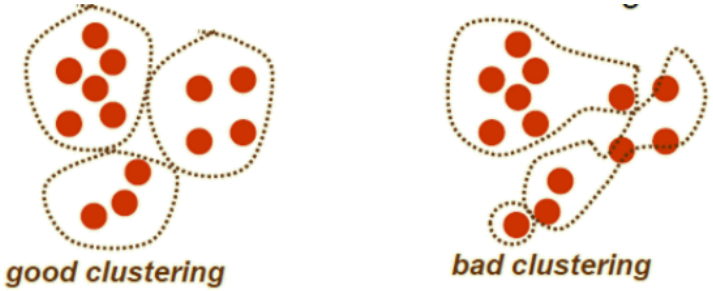
2. Criterion function to evaluate a clustering

3. Algorithm to computer clustering
   - For example, by optimizing the criterion function

What do we need for clustering?

1. Proximity measure:
   - similarity measure $S(X_i, X_k)$: large if X_i, X_k are similar
   - dissimilarity (or distance) measure $D(X_i, X_k)$: small if X_i, X_k are similar

2. Criterion function to evaluate a clustering

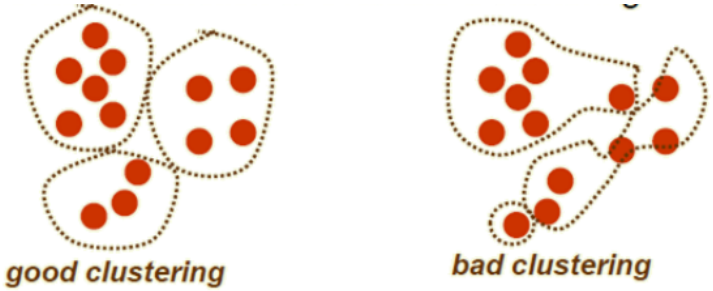


3. Algorithm to computer clustering
   - For example, by optimizing the criterion function

What do we need for clustering?

1. Proximity measure:
   - similarity measure $S(X_i, X_k)$: large if X_i, X_k are similar
   - dissimilarity (or distance) measure $D(X_i, X_k)$: small if X_i, X_k are similar

2. Criterion function to evaluate a clustering



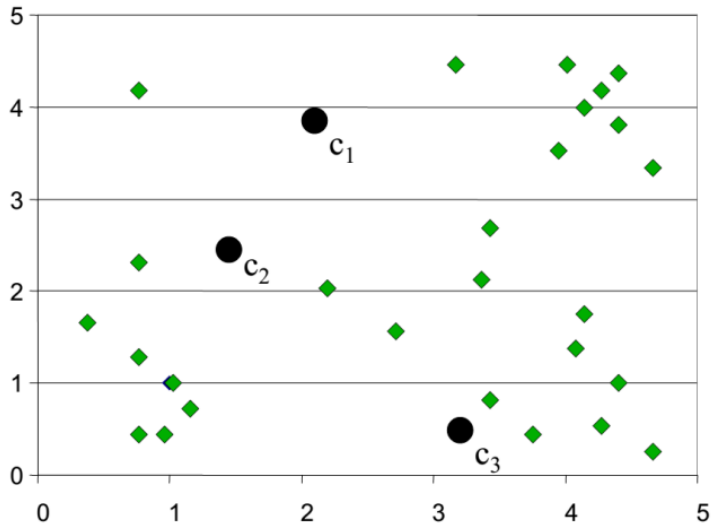
3. Algorithm to computer clustering
   - For example, by optimizing the criterion function

K-Means Clustering

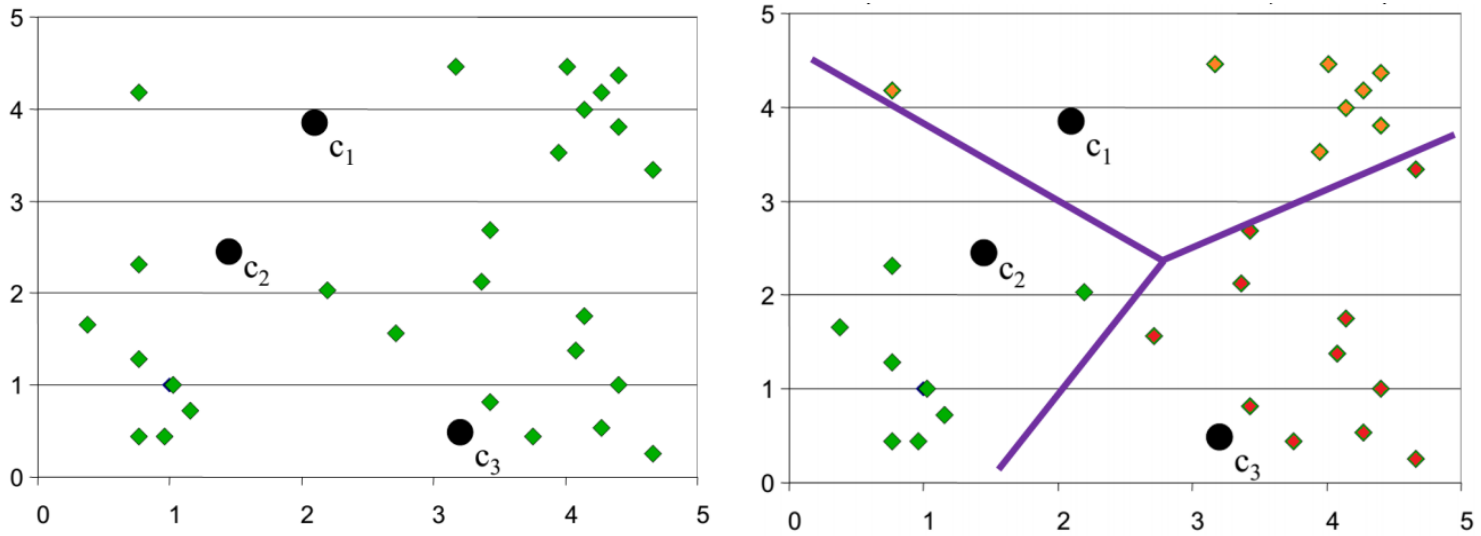
- K-means (MacQueen, 1967) is a partitional clustering algorithm
- Let the set of data points D be $x_1, x_2, ..., x_n$
  - where $x_i = (x_{i1}, x_{i2}, ..., x_{ir})$ is a vector in $X \subseteq R^n$ and n is the number of dimensions
- The k-means algorithm partitions the given data into K clusters:
  - Each cluster has a cluster center C, called centroid.
  - K is specified by the user
- convergence (stopping) criterion
  - minimum decrease in the sum of squared error (SSE)

$$SSE = \sum_{j=1}^{k} \sum_{x \in C_j} d(x, m_j)^2$$

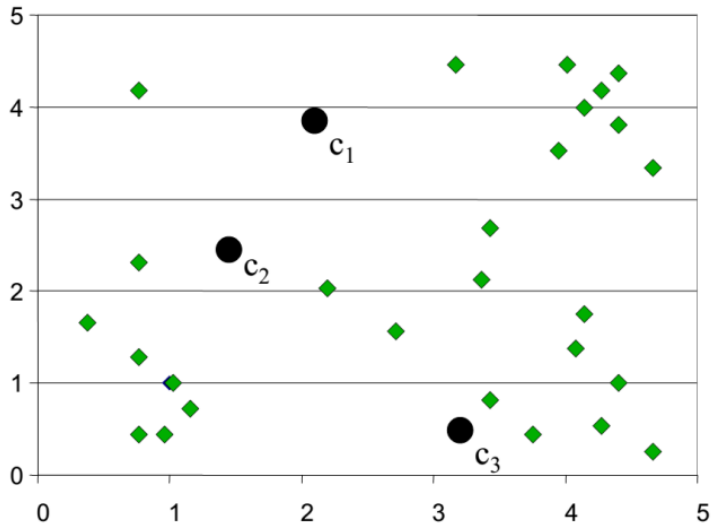
K-Means Clustering: working example

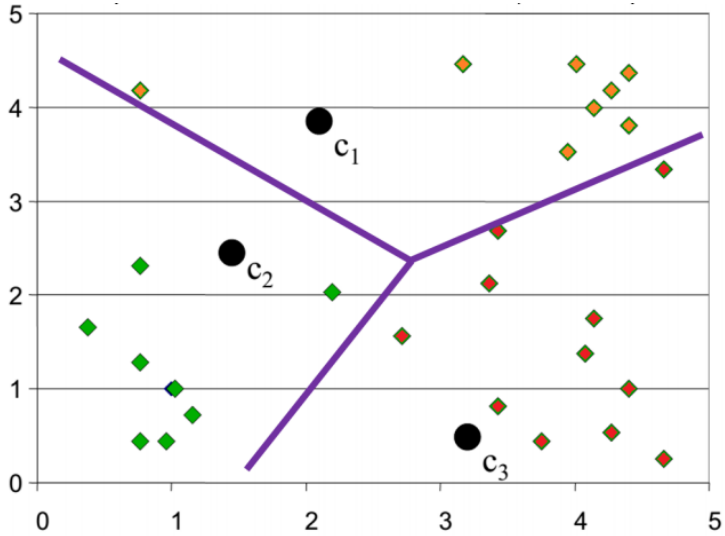


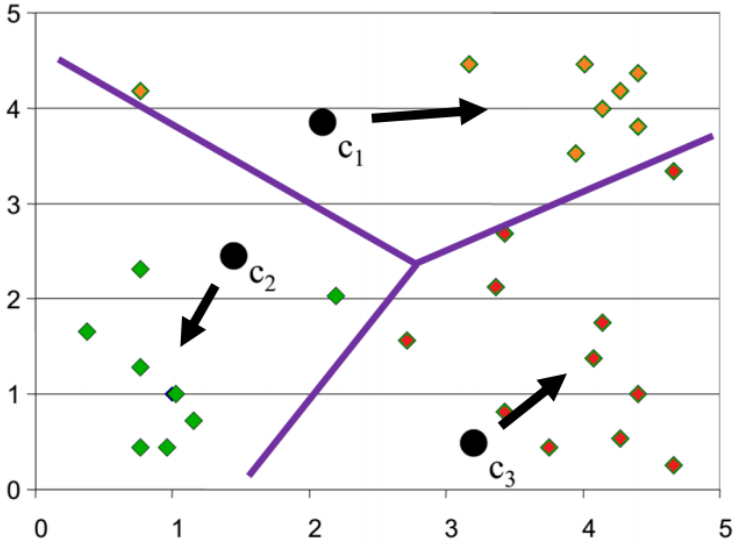
K-Means Clustering: working example



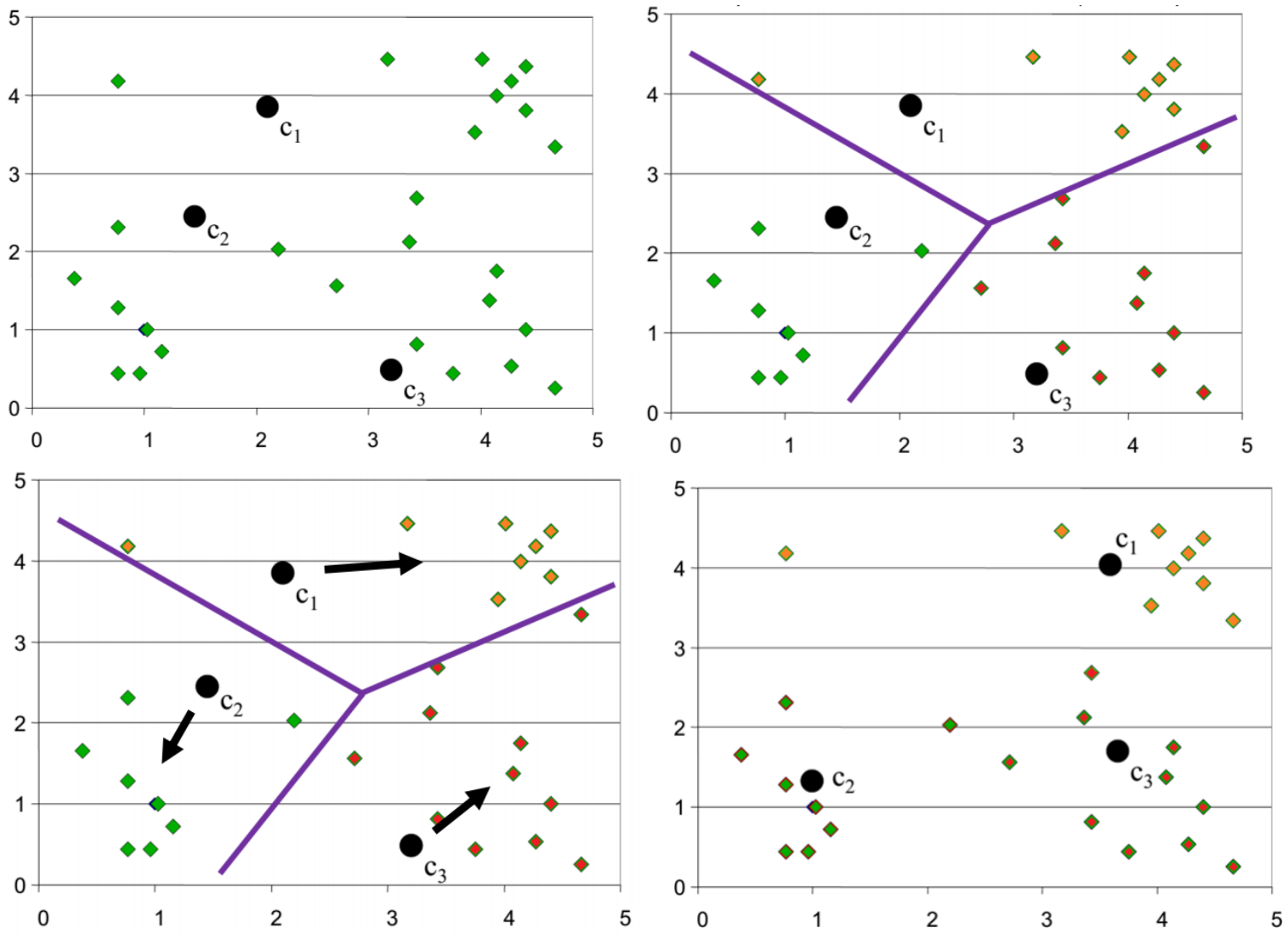
K-Means Clustering: working example





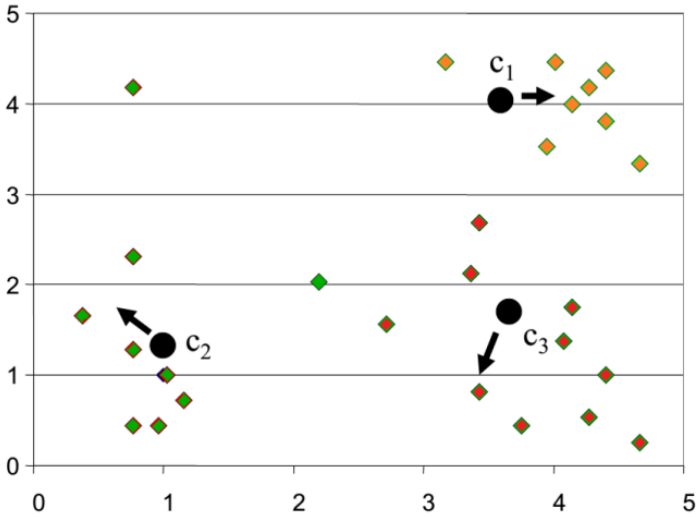


K-Means Clustering: working example

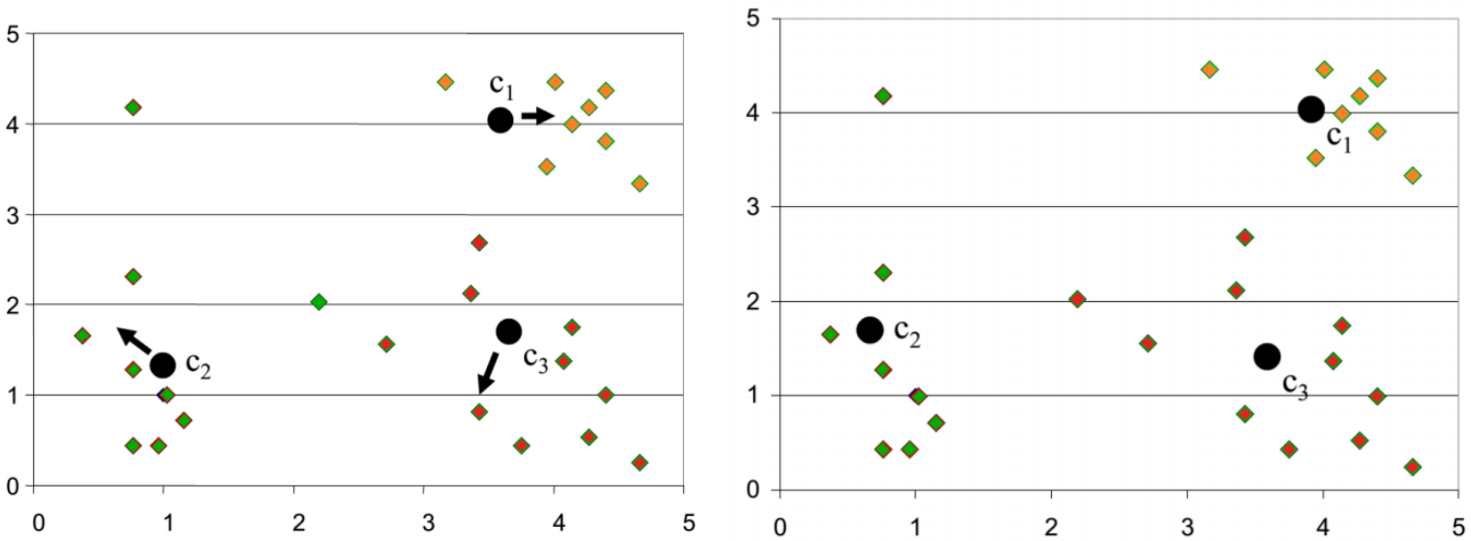


Result of first iteration

K-Means Clustering: working example



K-Means Clustering: working example



Result of second iteration

K-Means Clustering: lab working example

- Demo class work:
  `http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html`
- LAB:
  1. Data science: any working example (Samarachana)
  2. Medical imaging: brain tumor segmentation (Taman)

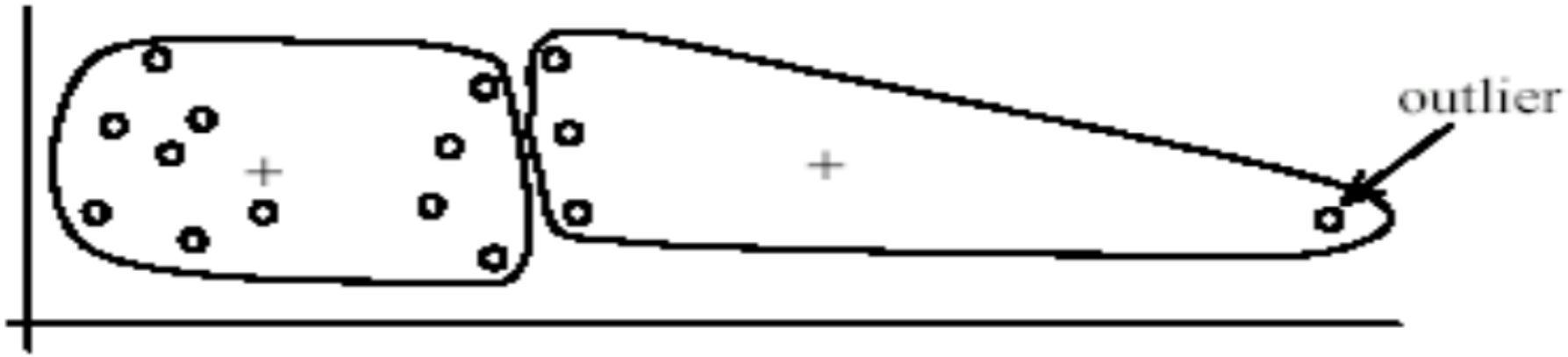
Why use K-means?

- Strengths:
  - Simple: easy to understand and to implement
  - Efficient: Time complexity: O(tkn),
    where n is the number of data points,
    k is the number of clusters, and
    t is the number of iterations.
  - Since both k and t are small. k-means is considered a linear algorithm.
- K-means is the most popular clustering algorithm.
- Note that: it terminates at a local optimum if SSE is used. The global optimum is hard to find due to complexity.

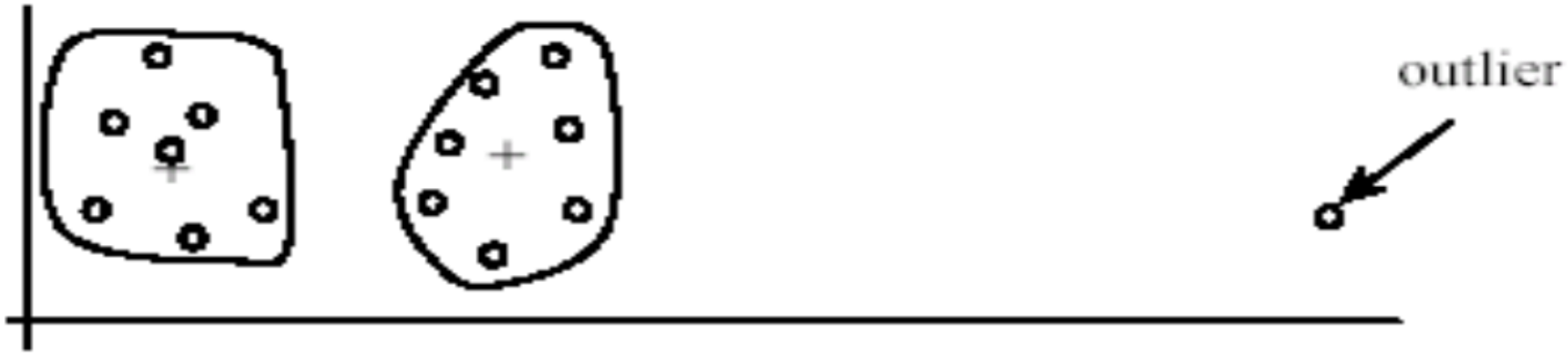
Weaknesses of K-means

- The algorithm is only applicable if the mean is defined.
  - For categorical data, k-mode - the centroid is represented by most frequent values.
- The user needs to specify k.
- The algorithm is sensitive to outliers
  - Outliers are data points that are very far away from other data points.
  - Outliers could be errors in the data recording or some special data points with very different values.

Outliers



(A): Undesirable clusters

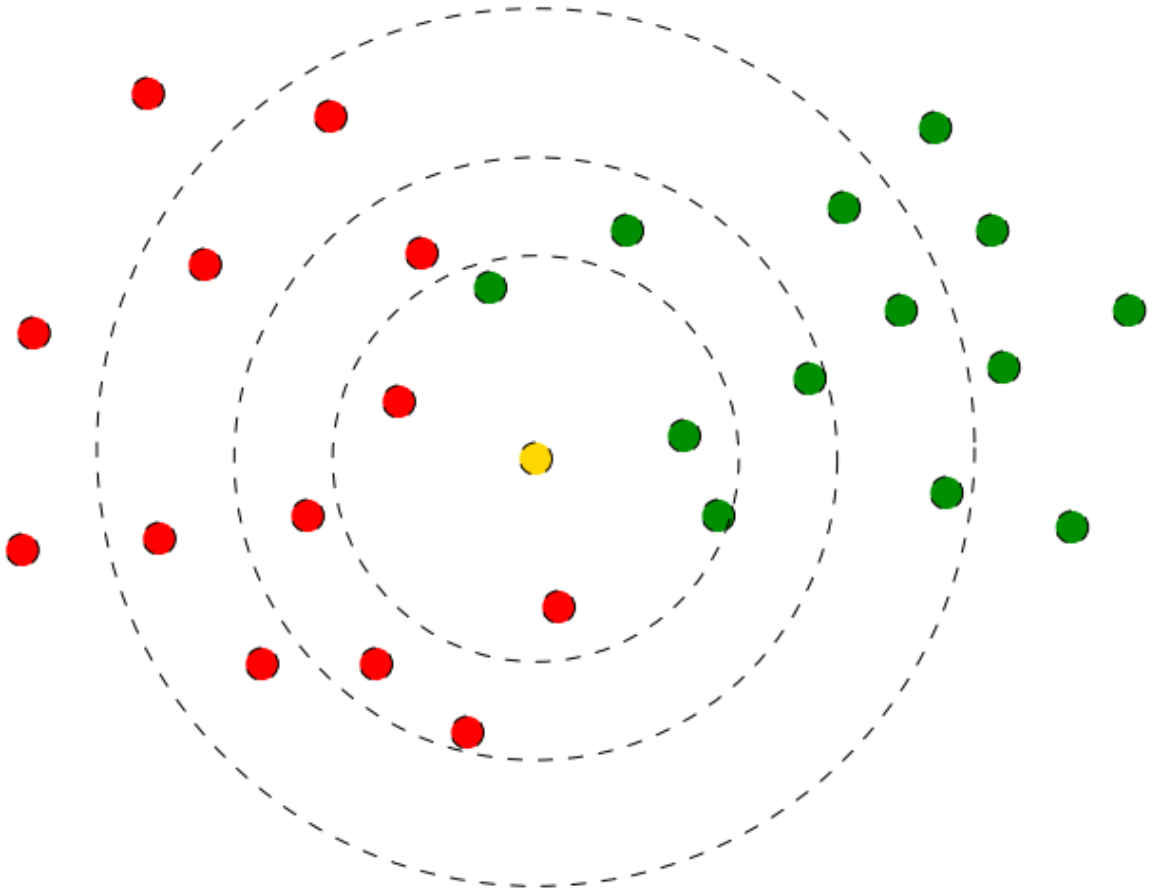


(B): Ideal clusters

K Nearest Neighbors

- **K-nearest neighbours** uses the local neighborhood to obtain a prediction
- The K memorized examples more similar to the one that is being classified are retrieved
- A distance function is needed to compare the examples similarity
  - Euclidean distance $d(x_j, x_k) = \sqrt{\sum_i (x_{j,i} - xk, i)^2}$
  - Manhattan distance $d(x_j, x_k) = \sqrt{\sum_i |x_{j,i} - xk, i)^2|}$
- This means that if we change the distance function, we change how examples are classified

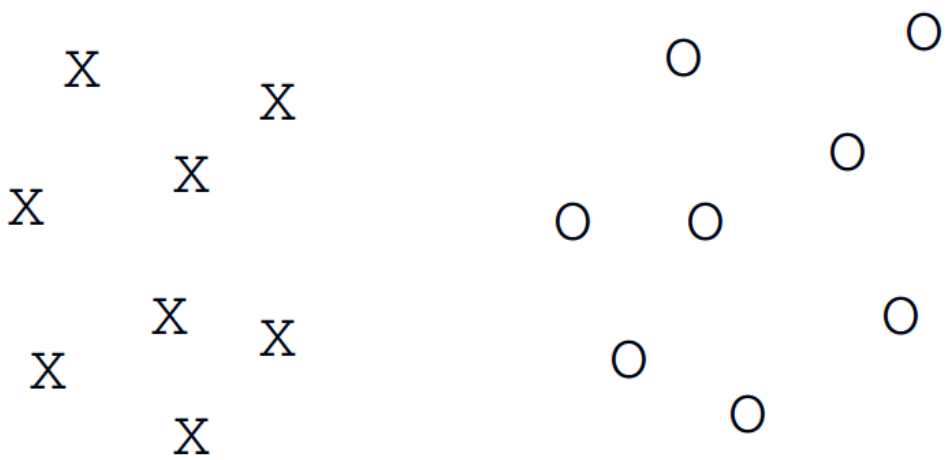
K Nearest Neighbors



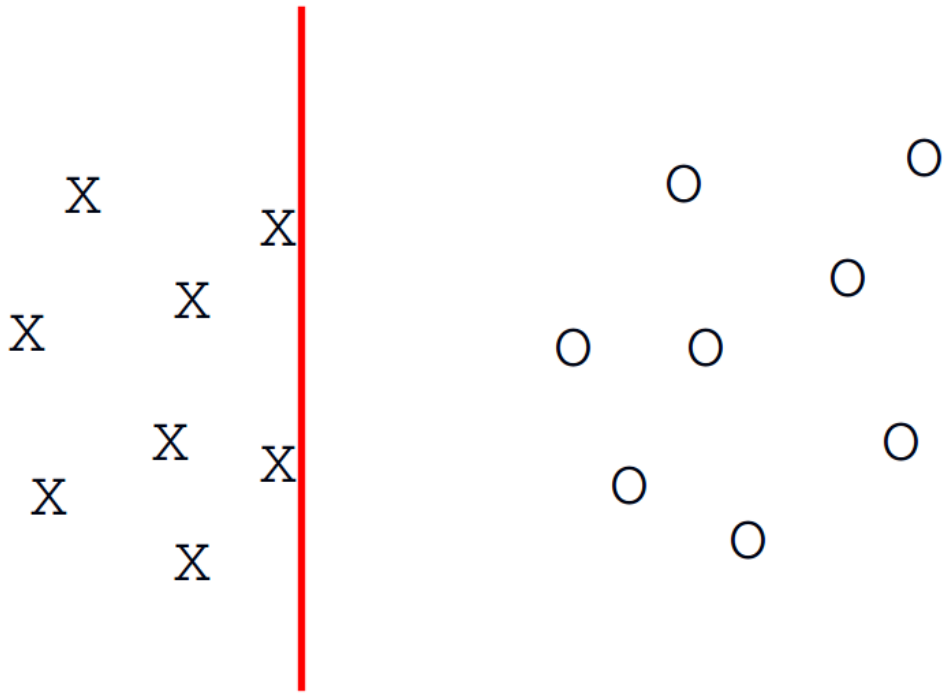
Support Vector Machine

Support Vector Machine

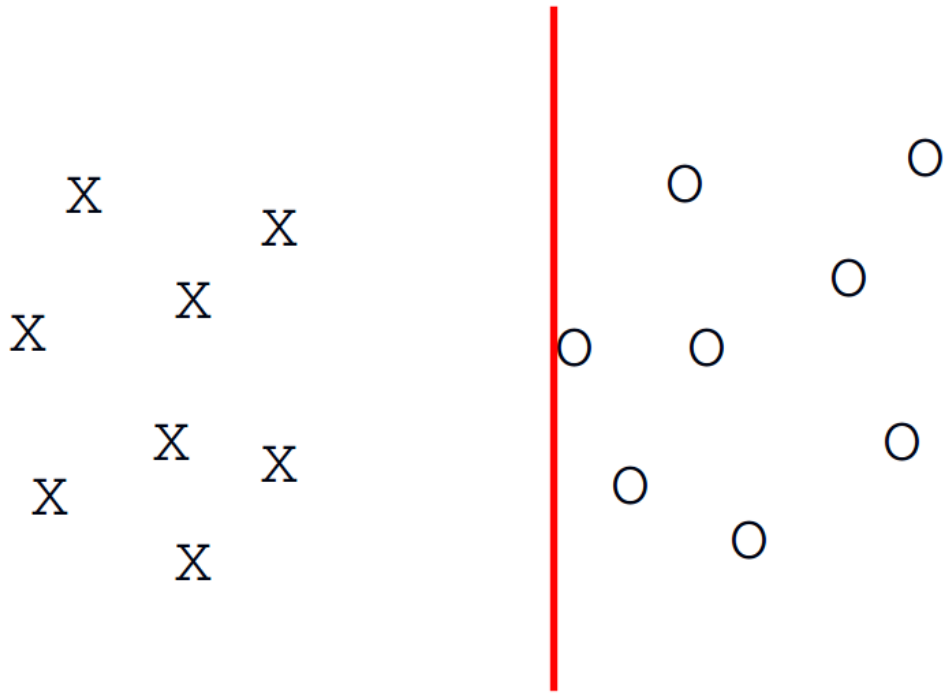
Intuitions



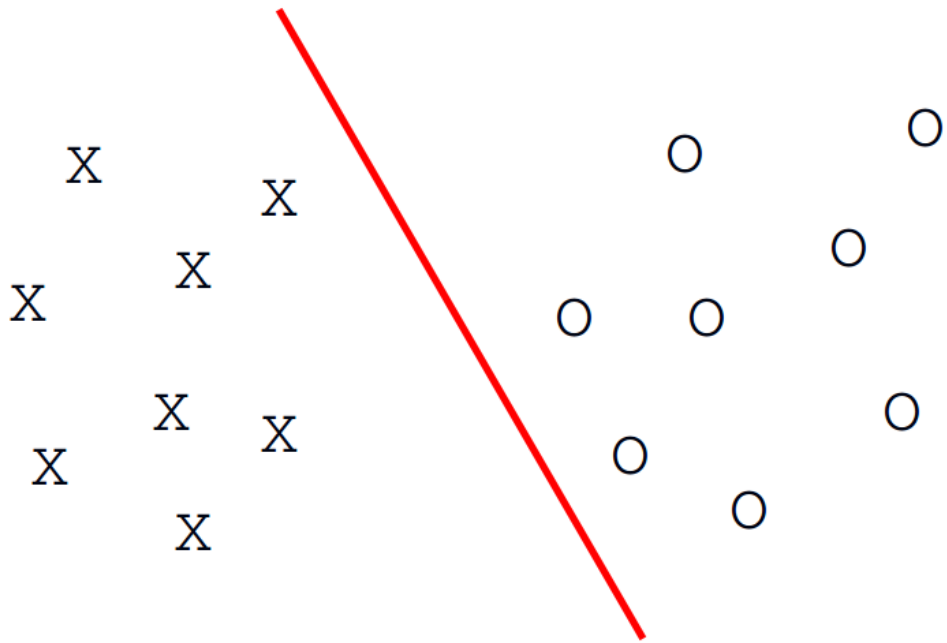
Intuitions



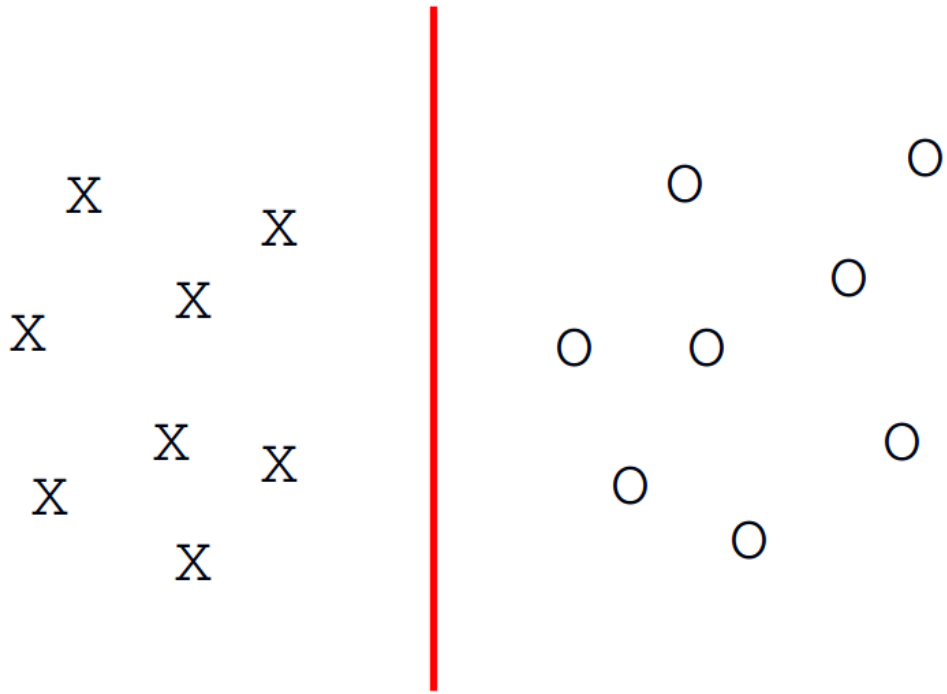
Intuitions



Intuitions

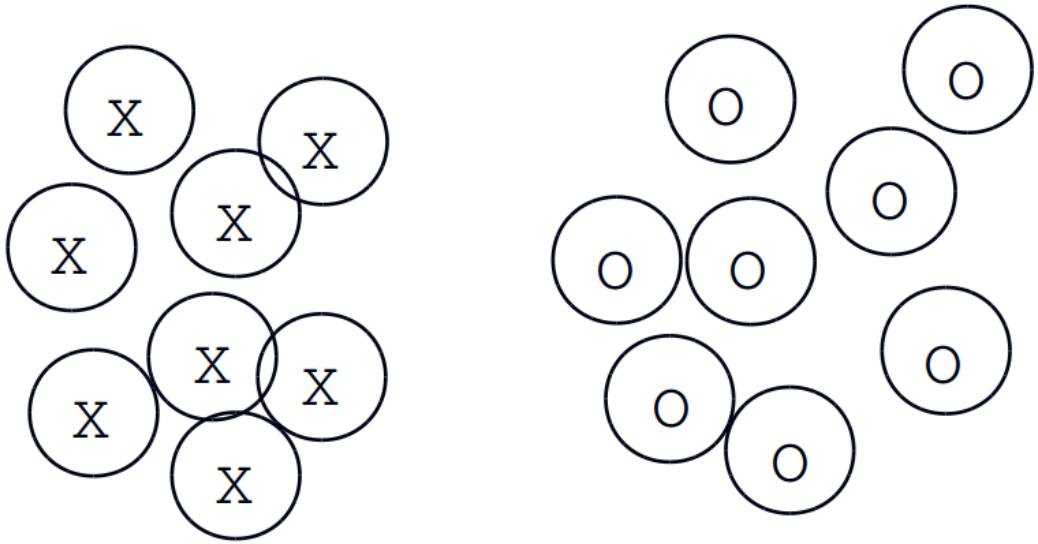


Intuitions

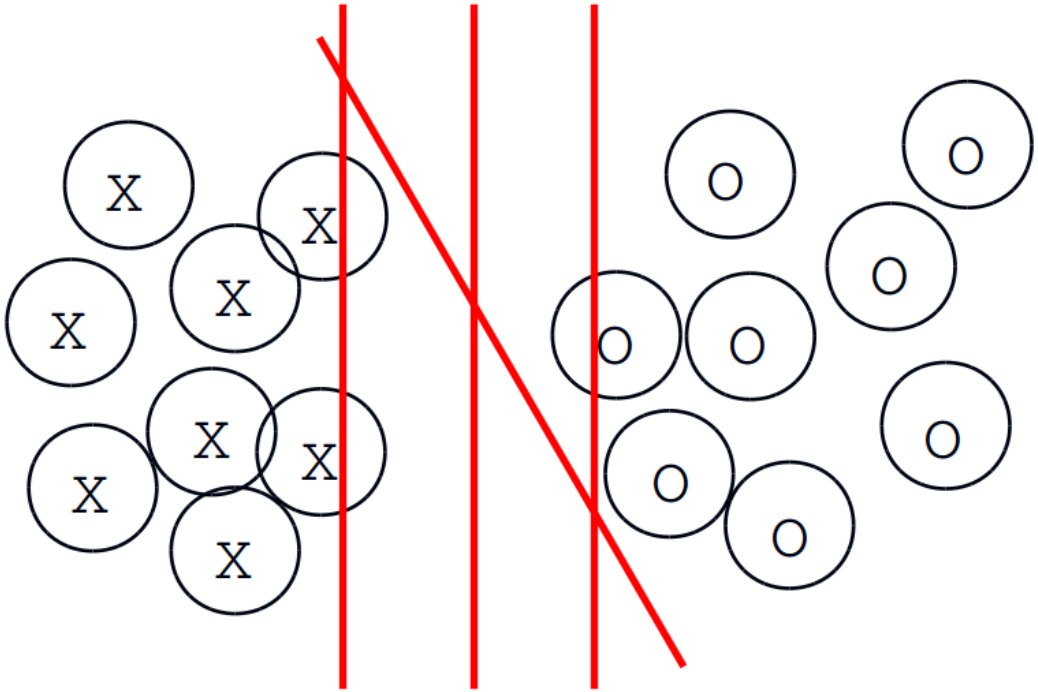


A "Good" Separator Noise in the Observations

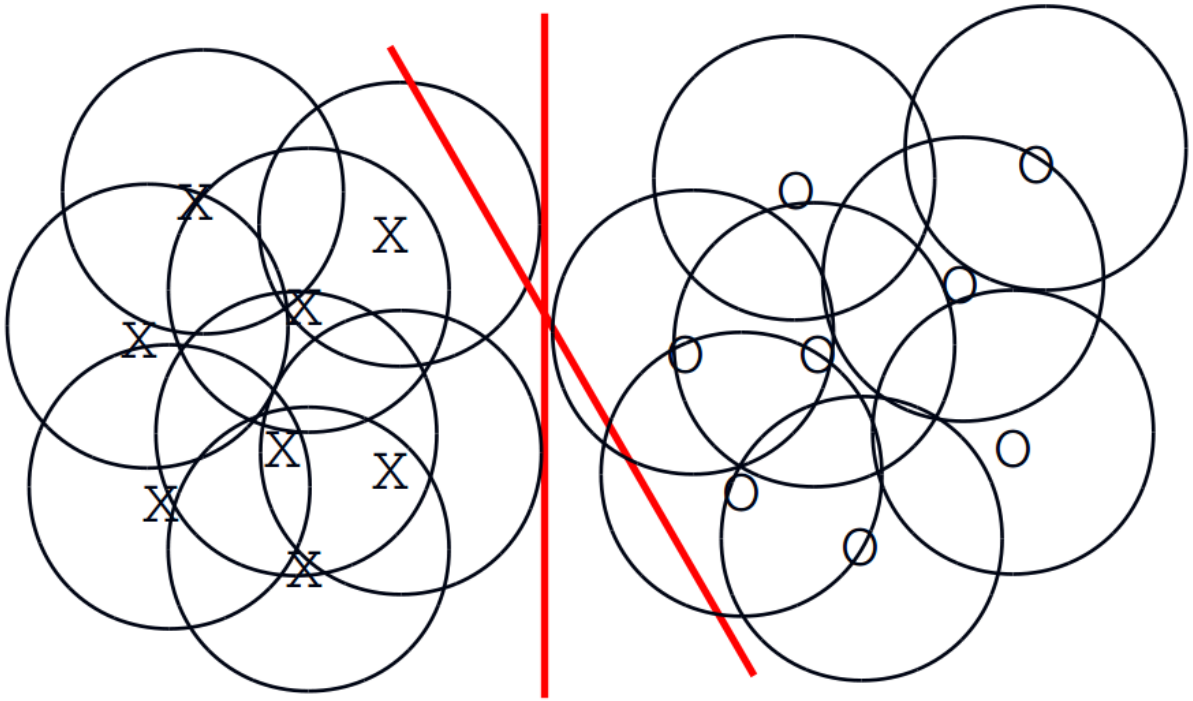
Intuitions



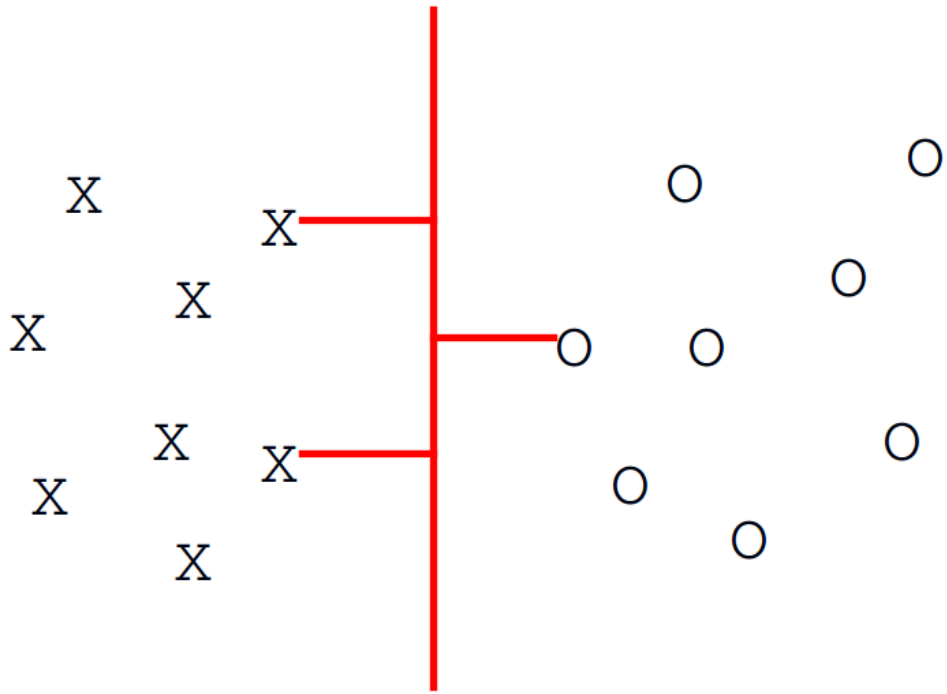
Intuitions



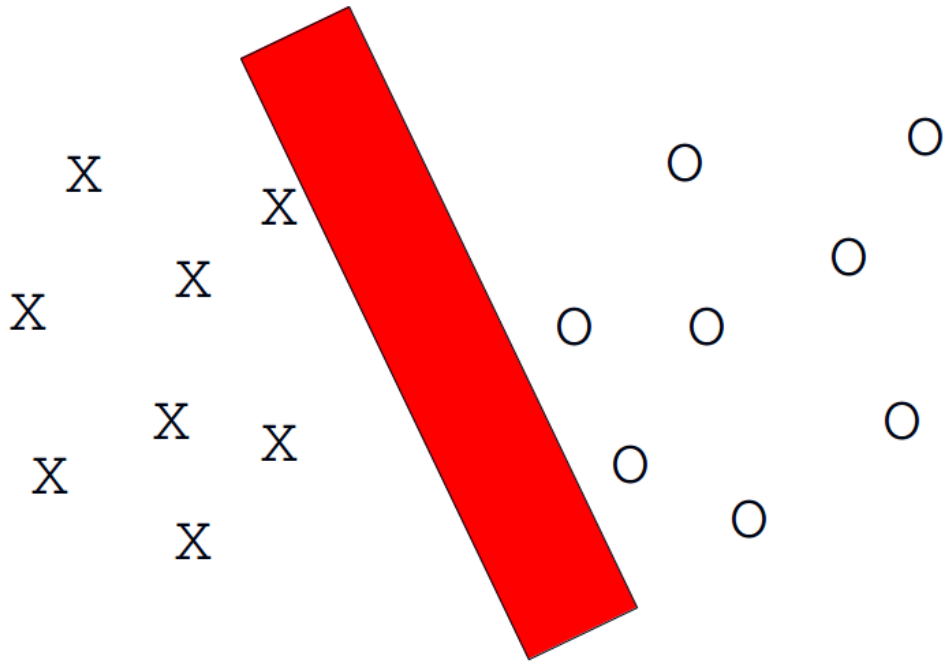
Intuitions



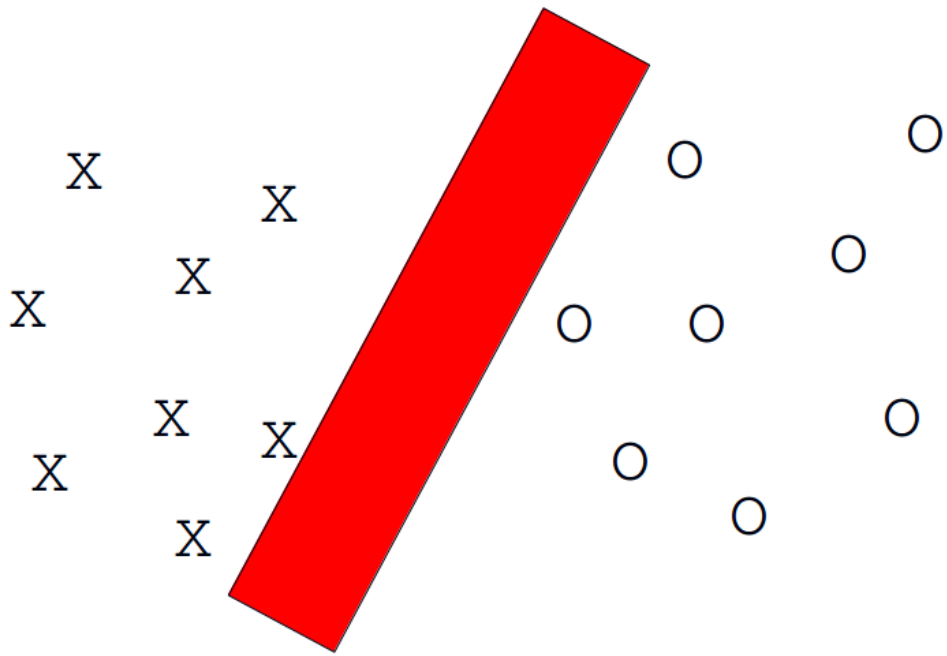
Intuitions



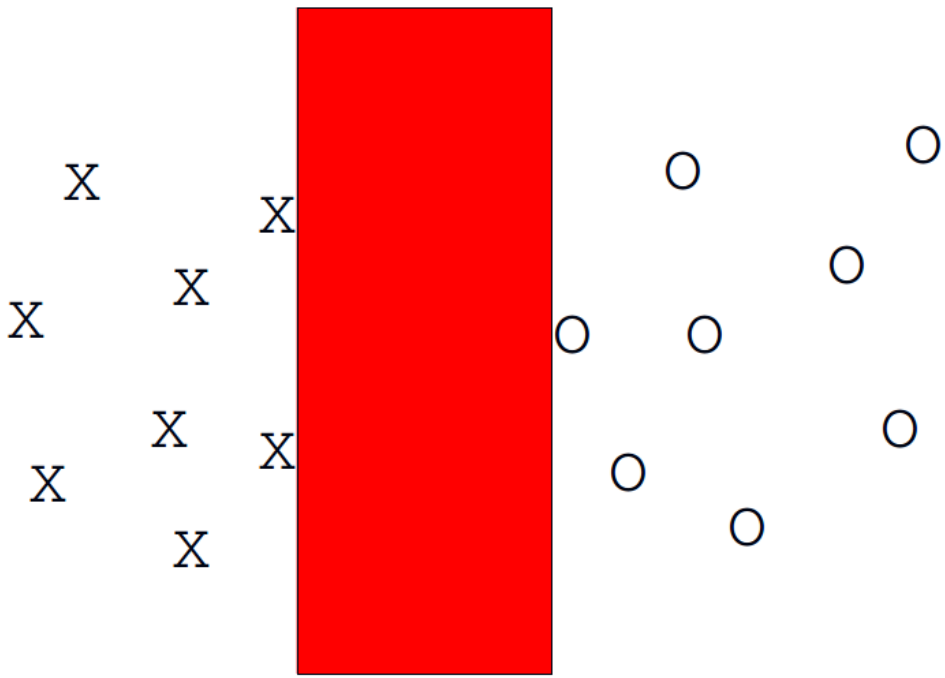
Intuitions



Intuitions



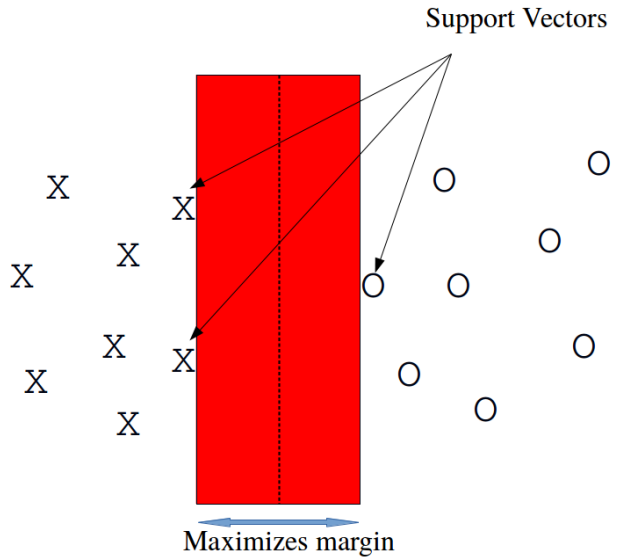
Intuitions



Idea: to select the separating hyperplane that maximizes the margin

Support Vector machine

- SVMs maximize the margin around the separating hyperplane.
  - A.ka. large margin classifiers
- The decision function is fully specified by a subset of training samples, the support vectors
- Solving SVMs is a quadratic programming problem



Why might predictions be wrong?

- True non-determinism
- Partial observability
  - hard, soft
- Representational bias
- Algorithmic bias
- Bounded resources

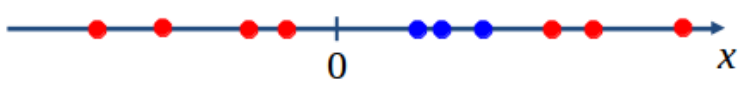
Support Vector machine

- Datasets that are linearly separable works great

Support Vector machine

- Datasets that are linearly separable works great

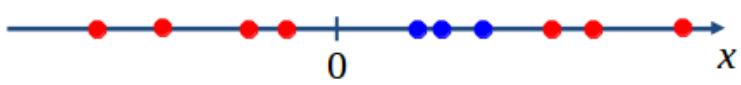
- But what are we going to do if the dataset is just too hard?



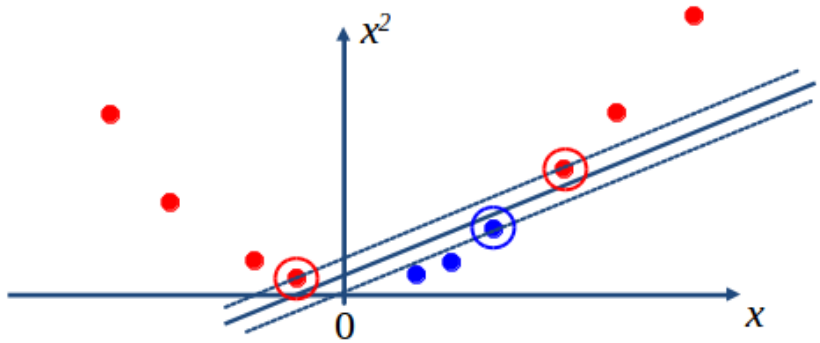
Support Vector machine

- Datasets that are linearly separable works great

- But what are we going to do if the dataset is just too hard?

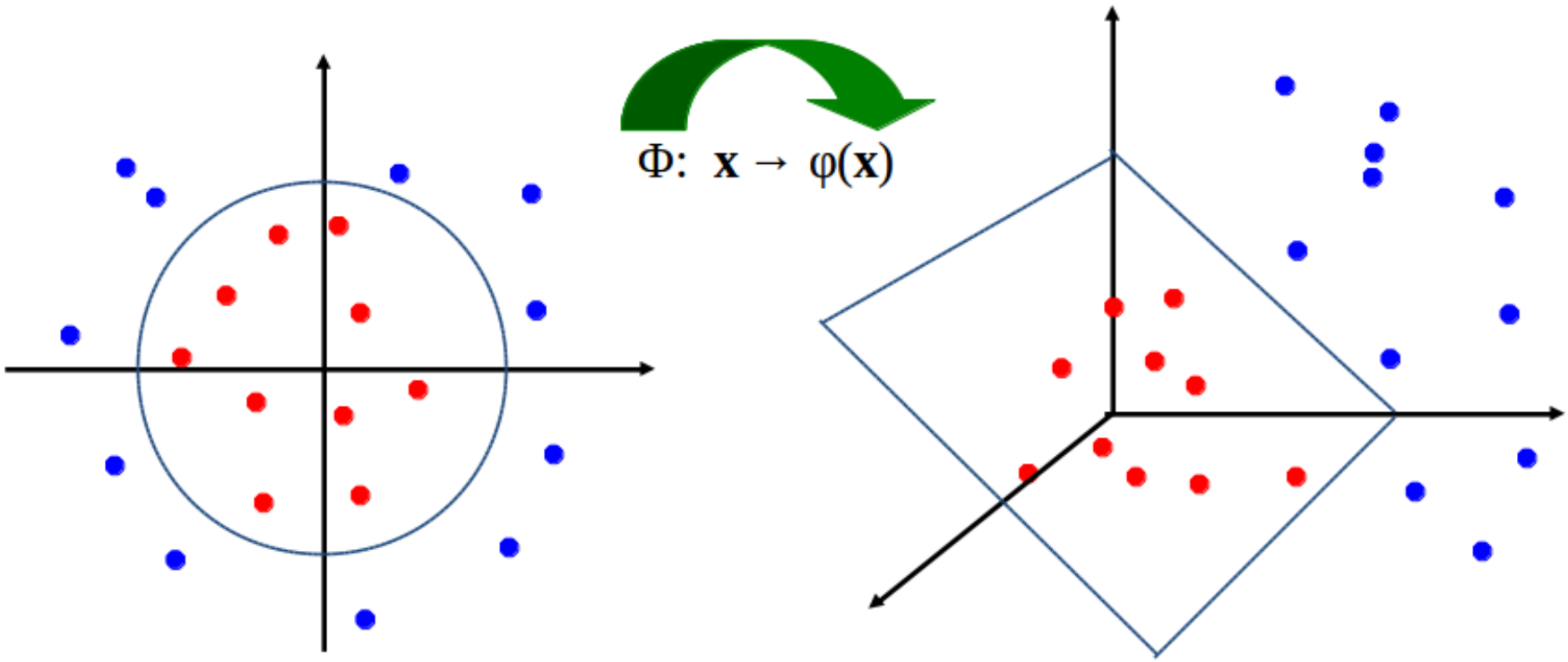


- How about mapping data to a higher-dimensional space



Non-linear SVMs: Feature spaces

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



https://www.youtube.com/watch?v=9NrALgHFwTo

The "Kernel Trick"

- The linear classifier relies on an inner product between vectors $K(x_i, x_j) = (x_i)^T(x_j)$
- If every datapoint is mapped into high-dimensional space via some transformation $\Phi : x \rightarrow \varphi(x)$, the inner product becomes: $K(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)$
- A kernel function is some function that corresponds to an inner product in some expanded feature space
  - Make non-separable problem separable.
  - Map data into better representational space
- Common Kernels
  - Linear
  - polynomial $K(x, z) = (1 + x^T z)^d$
  - Radial basis function (infinite dimensional space) $K(x_i, x_j) = e^{(||x_i - x_j||^2)/2\sigma^2}$

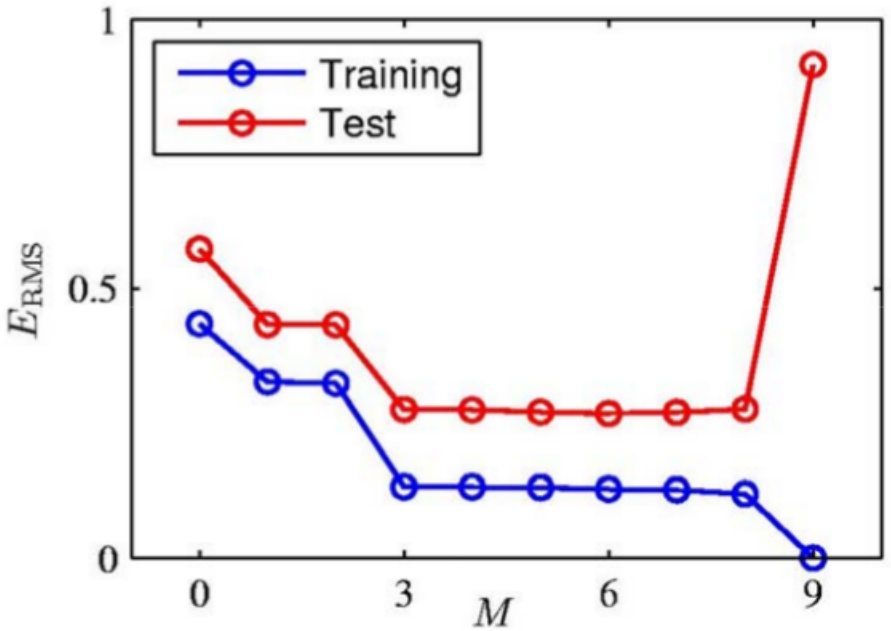
SVM "Demo class work"

https://cs.stanford.edu/~karpathy/svmjs/demo/

Regularization and Evaluation

Bias - Variance Tradeoff

$$\text{MSE} = \text{Bias}^2 + \text{Variance}$$



Regularization

In regression, when $p >> n$, non-convex optimization. Regularization is introduced as a convex relaxation of the regression model.

$$\text{argmin}_{\beta} \text{loss} + \text{penalty}(\beta)$$

Regularization - L_1 and L_2 norms

1. Lasso Penalty

$$\text{argmin}_\beta \text{loss} + \lambda|\beta|$$

2. Ridge Penalty

$$\text{argmin}_\beta \text{loss} + \lambda|\beta|_2$$

How to choose λ?

Regularization - L_1 and L_2 norms

1. Lasso Penalty

$$\text{argmin}_\beta \text{loss} + \lambda|\beta|$$

2. Ridge Penalty

$$\text{argmin}_\beta \text{loss} + \lambda|\beta|_2$$

How to choose λ?

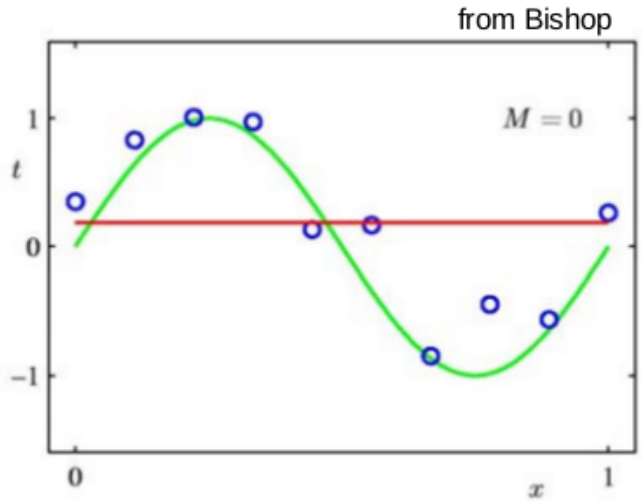
Performance metrics

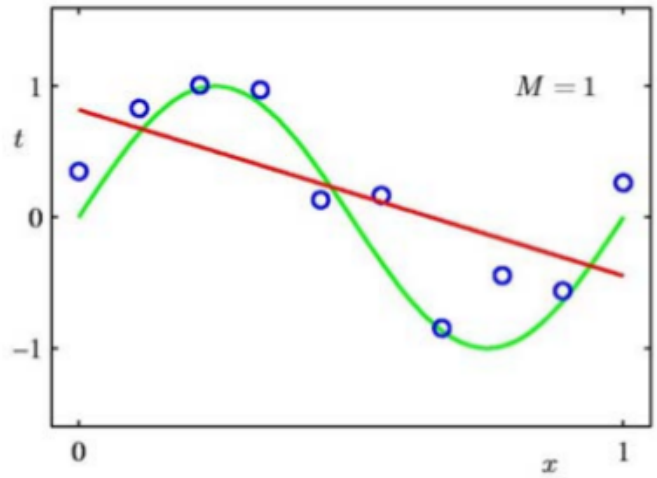
Truth table (confusion matrix)

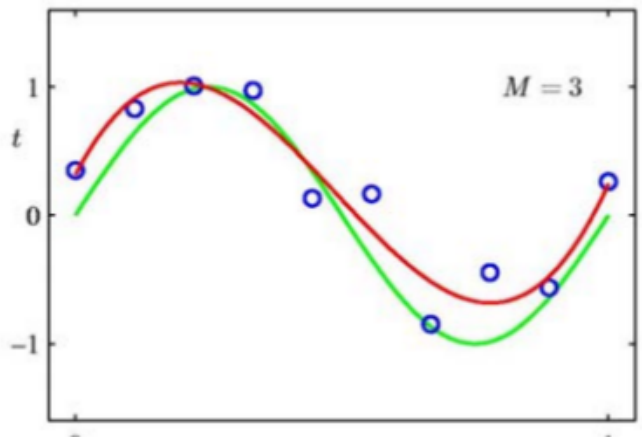
		Predicted class P	**Predicted class** N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

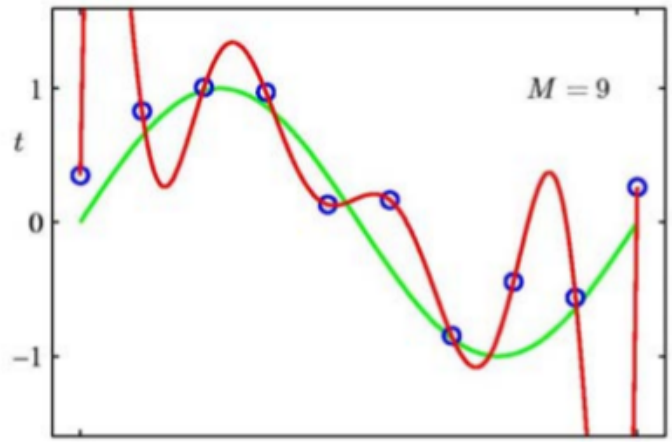
- Accuracy $= \frac{TP+TN}{P+N}$
- Sensitivity $= \frac{TP}{\text{Actual Positives}}$
- Specificity $= \frac{TN}{\text{Actual Negatives}}$
- ...

Some fits to the data: which is best?



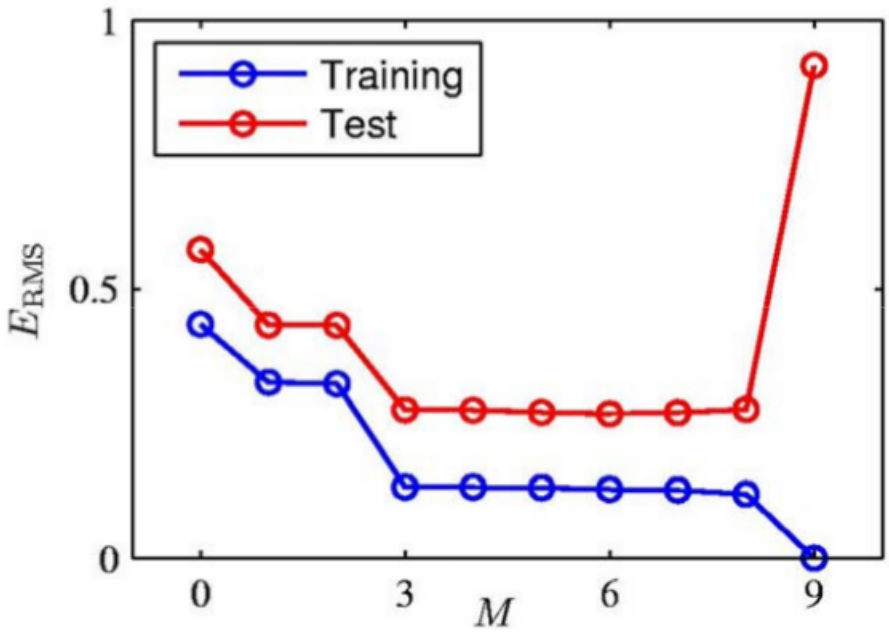






Bias-variance trade-off

- test data: a different sample from the same true function

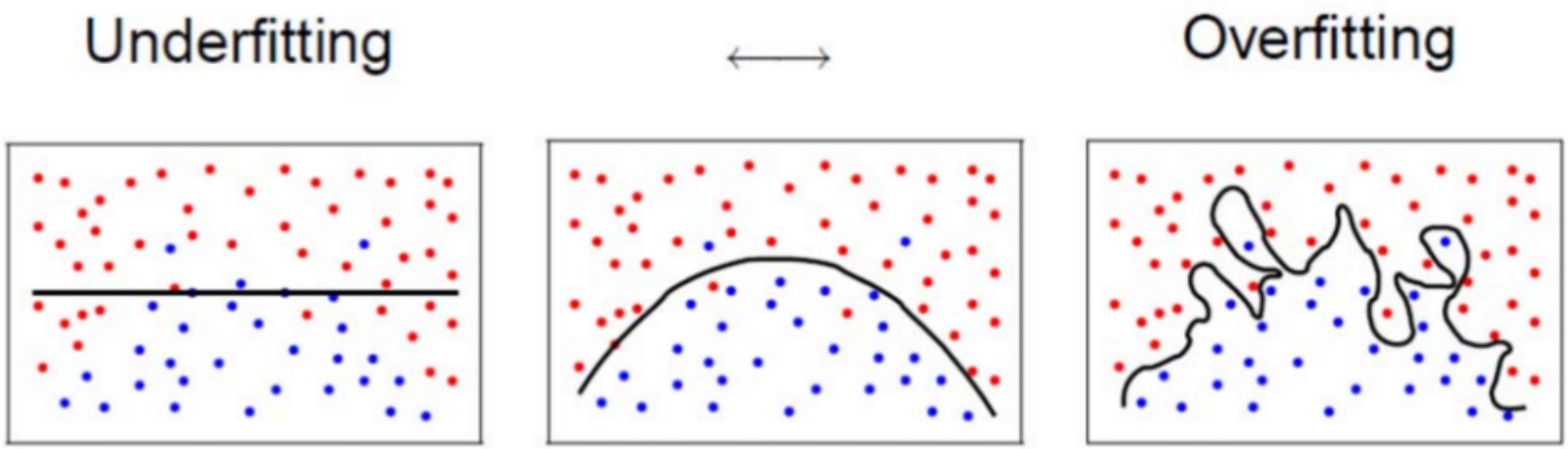


RootMeanSquare (RMS) Error

- training error goes to zero, but test error increases with M

Generalization Problem in Regression/Classification

- Best fit will be somewhere in between under-fitting and over-fitting



Cross-validation

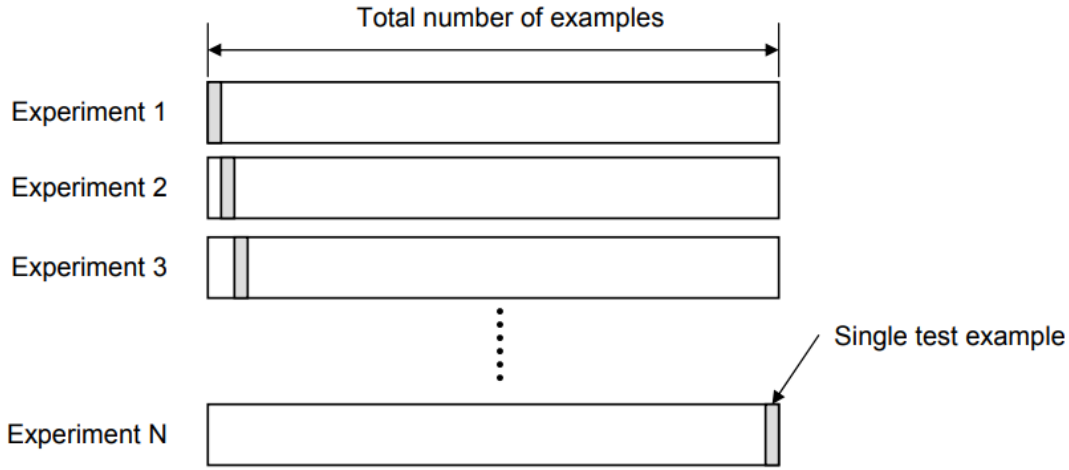


Figure: Leave-one-out cross-validation

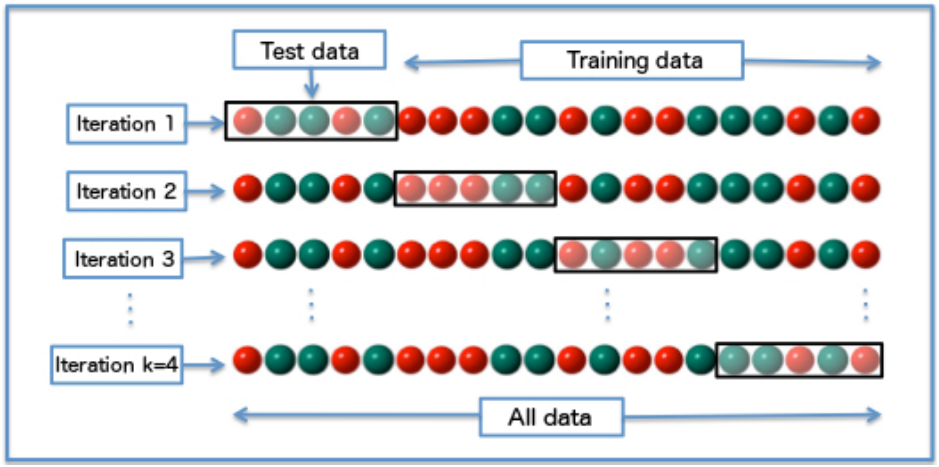


Figure: 5-fold cross-validation

Cross-validation

- 10-fold cross-validation ?

Cross-validation

- 10-fold cross-validation ?

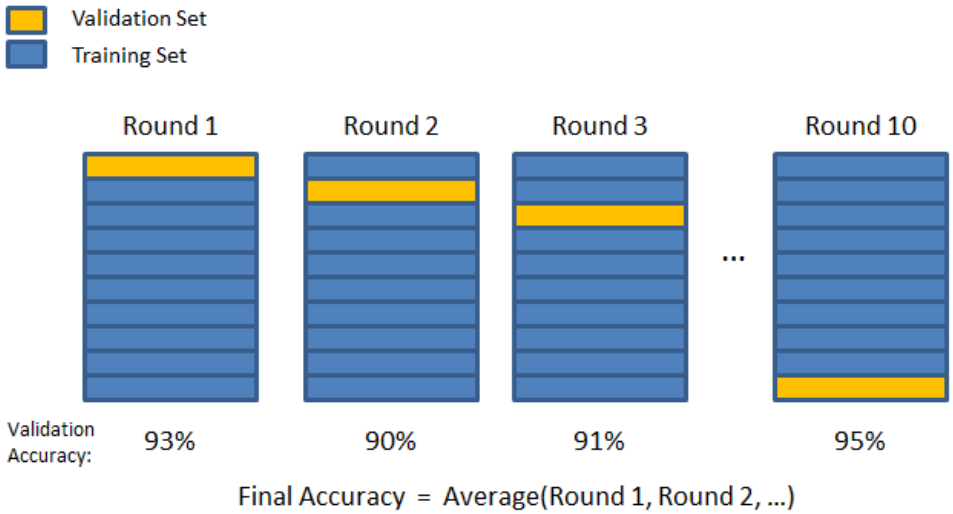


Figure: 10-fold cross-validation

Cross-validation

- 10-fold cross-validation ?

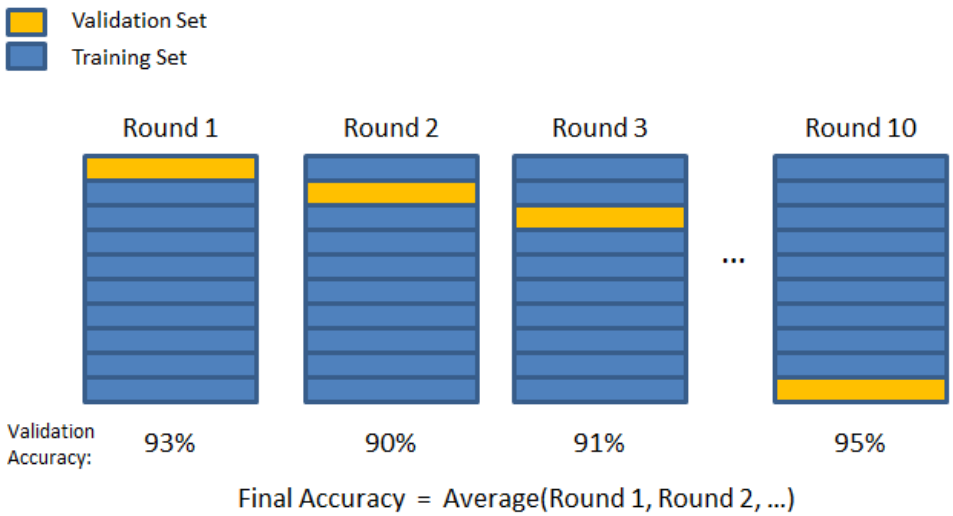
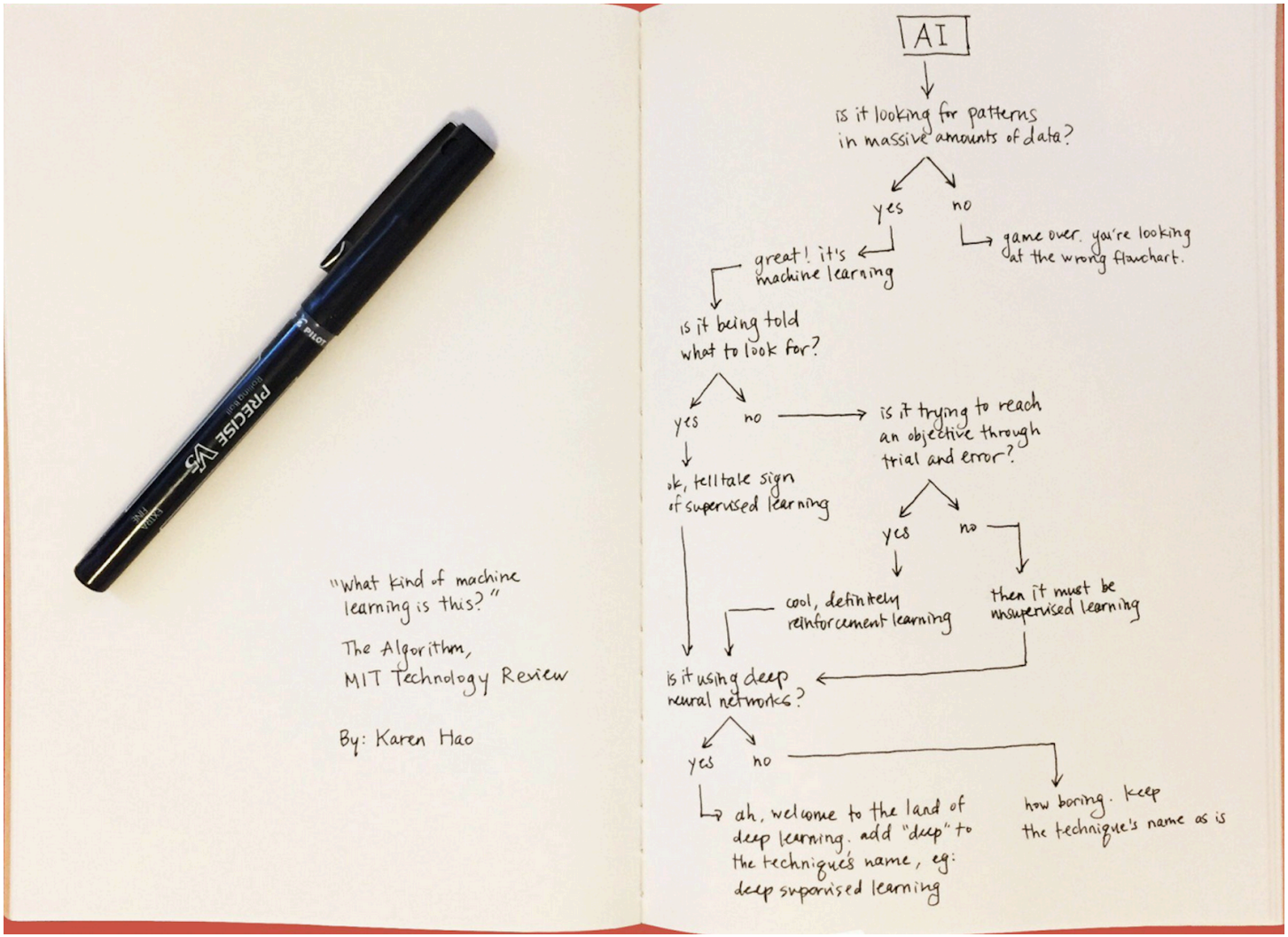


Figure: 10-fold cross-validation

- 5/10-repeated 5/10-fold cross-validation ?
  The only thing that differs is that you randomly divide the data into 5/10 folds a different way each time

Source: MIT tech review



References

COMP24111 Machine Learning
Machine Learning 10-701, Tom Mitchell
The Elements of Statistical Learning, (Friedman, Hastie and Tibshirani)

Slide credit

Pedro Domingos
E. Alpaydin
Sebastian Thruns multimedia website
Geoffrey Hinton
Tim Oates