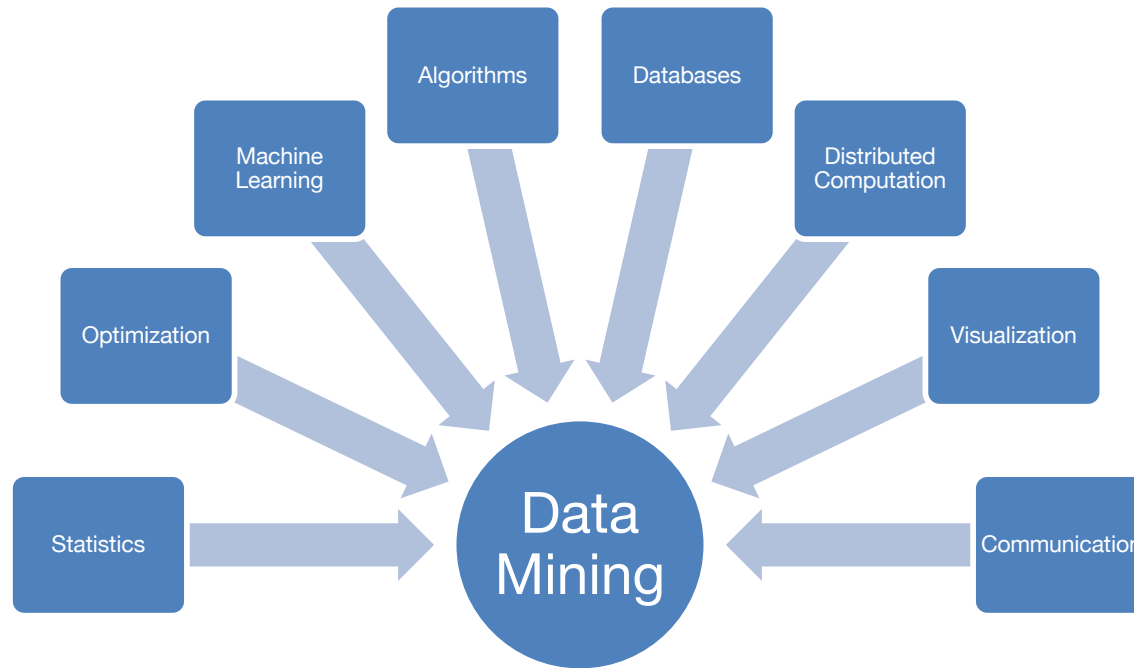


# What is Data Mining?

## Lecture 1



# Intersecting Disciplines



“A data scientist is someone who knows more statistics than a computer scientist and more computer science than a statistician.”

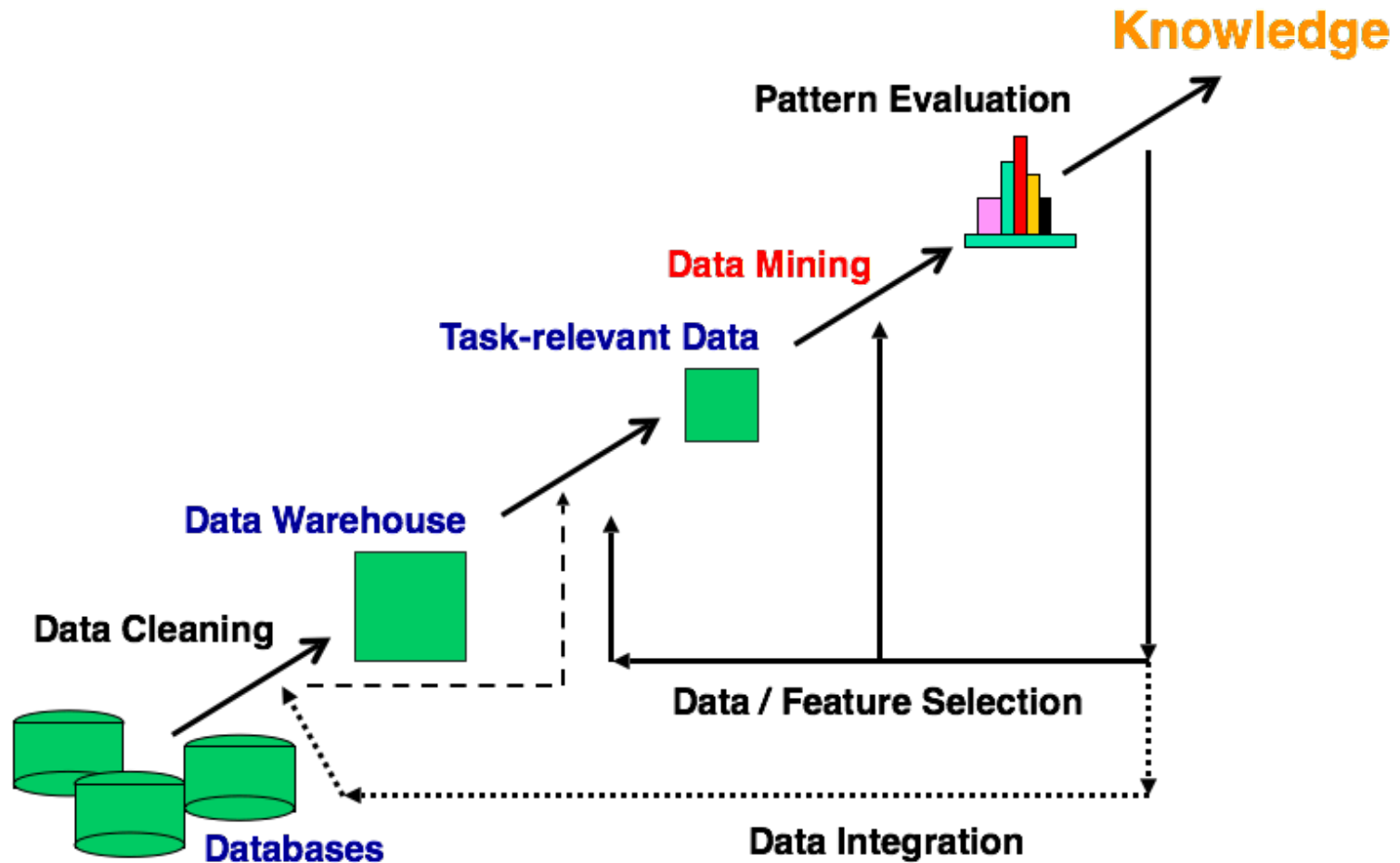
– Josh Blumenstock (UW)

“Data Scientist = statistician + programmer + coach + storyteller + artist”

– Shlomo Aragon (Ill. Inst. of Tech)



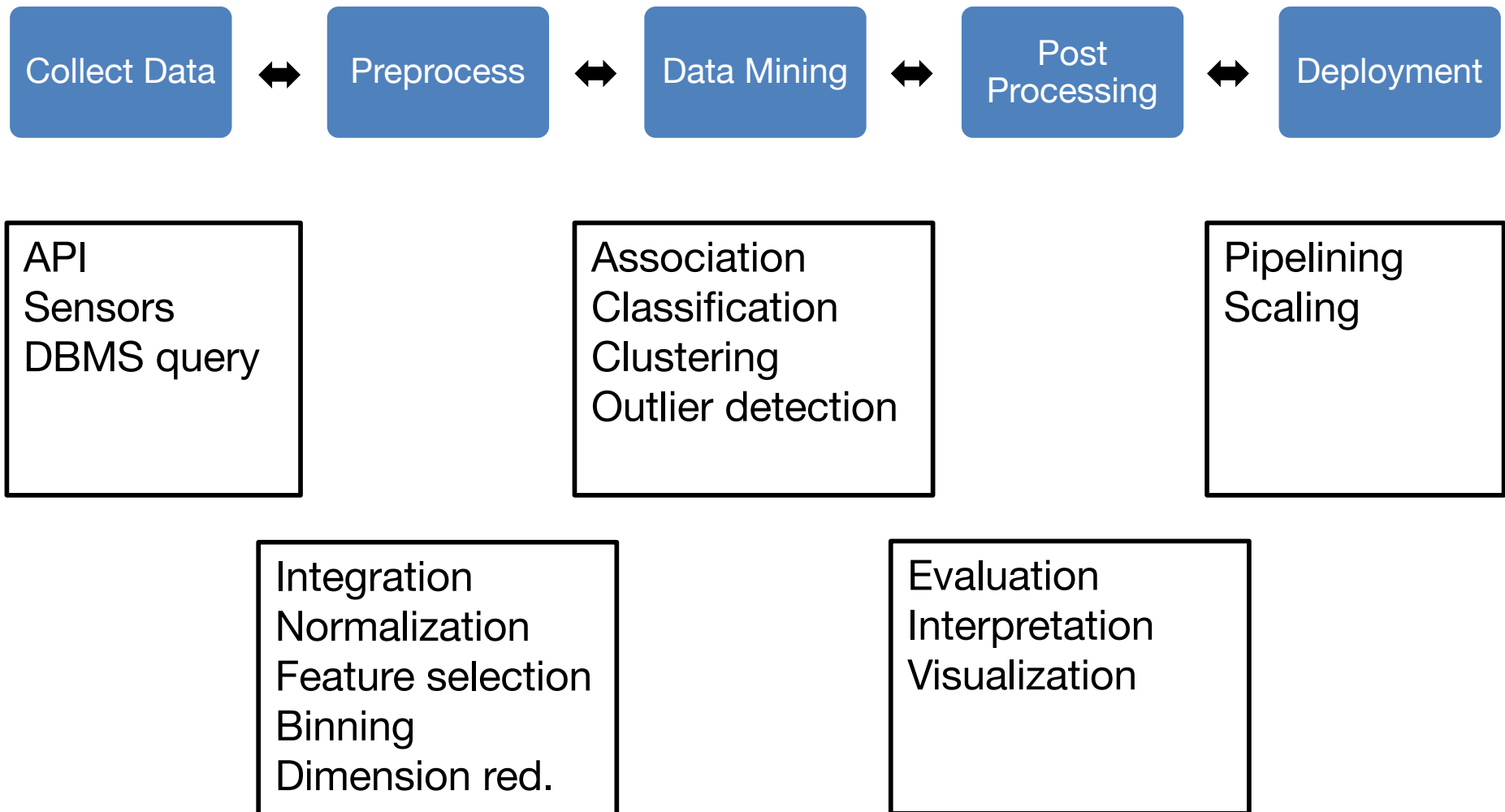
# Database Perspective



(slide adapted from Han et al. Data Mining Concepts and Techniques)



# Machine Learning Perspective



# What is Machine Learning?

- The study/construction of algorithms that can learn from data
- The study of algorithms that improve their performance **P** at some task **T** with experience **E**
  - Tom Mitchell (CMU)
- Fusion of algorithms, artificial intelligence, statistics, optimization theory, visualization, ...
- "Software Is Eating the World, but AI Is Going to Eat Software" -Jensen Huang (CEO, NVIDIA)
  - NLP, vision, games, robotics, medicine, ads, ...



# Jobs!

Position	Salary*
Data Scientist Machine Learning Engineer Data Mining Scientist	<b>\$123,732</b>
Software Engineer	\$102,640
Data Mining Engineer	\$95,582
Data Mining Analyst	\$73,616

\*glassdoor.com, National Avg as of September 4, 2017

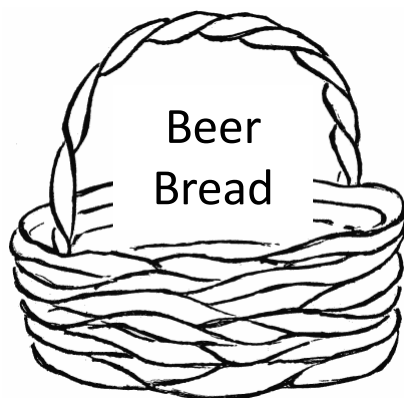
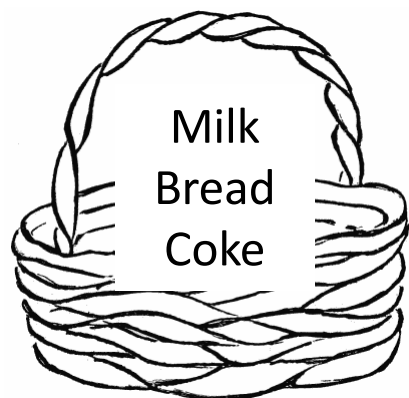


# Three Aspects of Data Mining

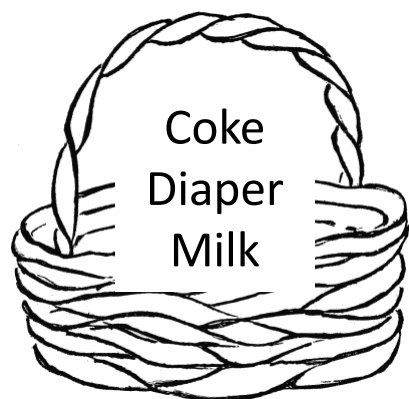
- Data Types/Representations
  - Sets, tables, matrices, graphs, time series, sequences, ...
- Tasks -> Methods/Algorithms
  - Supervised: kNN, linear regression, NB
  - **Unsupervised: this class :)**
  - Reinforcement: TD-learning
- Applications
  - Recommender systems, community detection, market basket analysis, ...



# Sets -> Association Rules

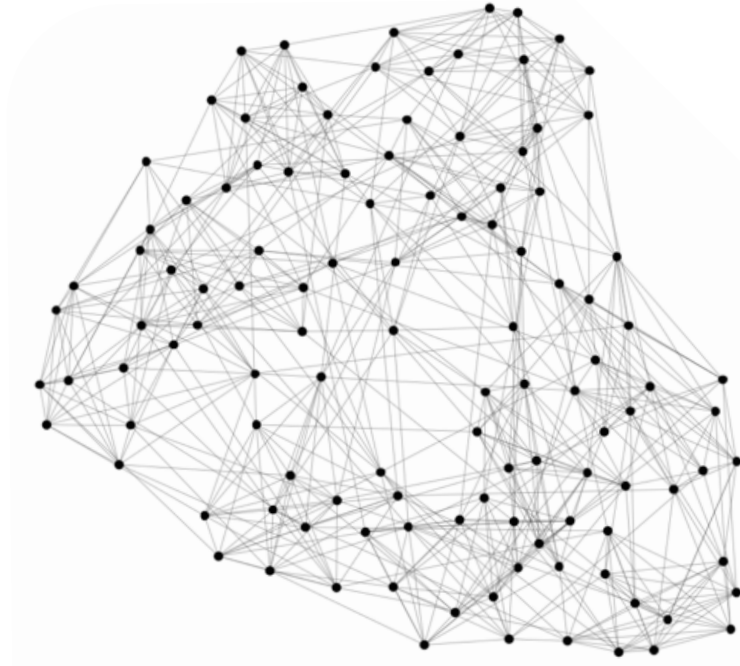


- Basket of items
  - What items are frequently purchased together
- {Milk} -> {Coke}





# Graphs -> Community Detection

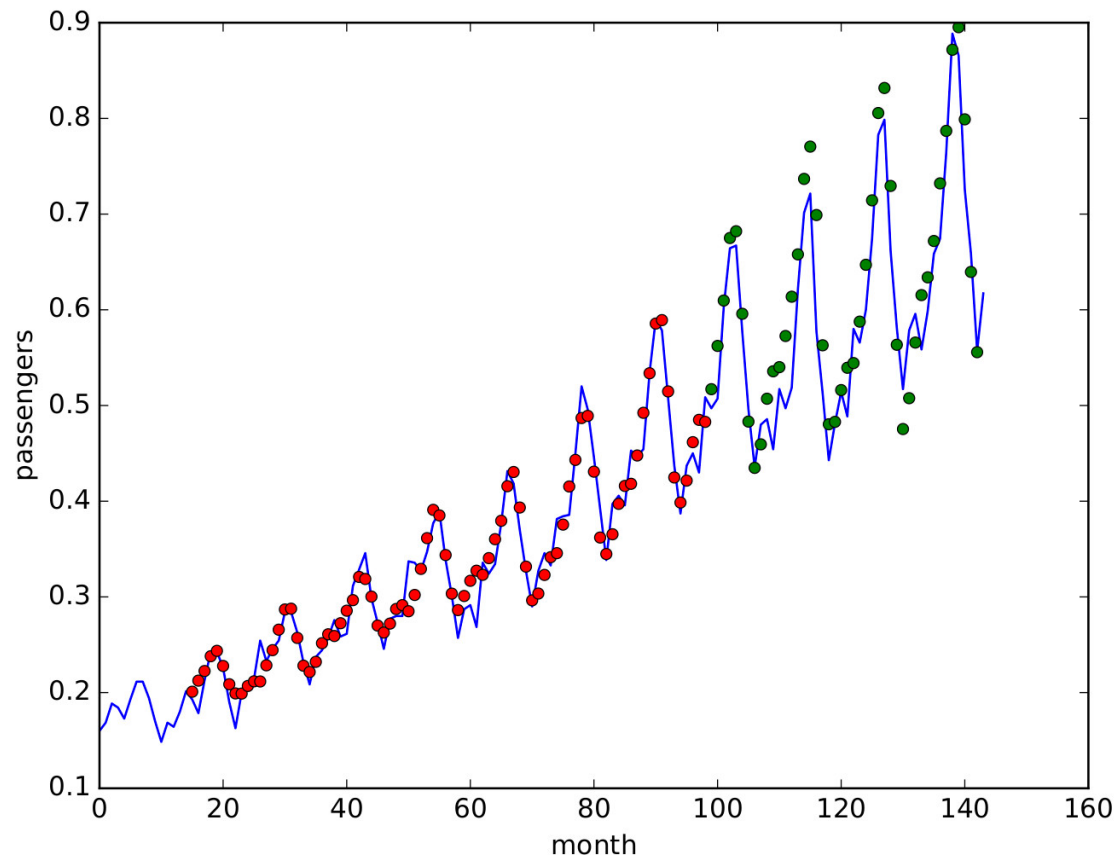


Can we identify groups of tightly connected nodes?

(Adapted from: Mining of Massive Datasets, <http://www.mmds.org>)



# Time Series Data -> Predictions



Can we predict future data points from a trend?

(Adapted from: <https://am241.wordpress.com/tag/time-series/>)



# Matrix -> Recommender Systems

	Item 1	Item 2	Item 3	Item 4
Properties				
User A				
User B				
User C				
User D		???		

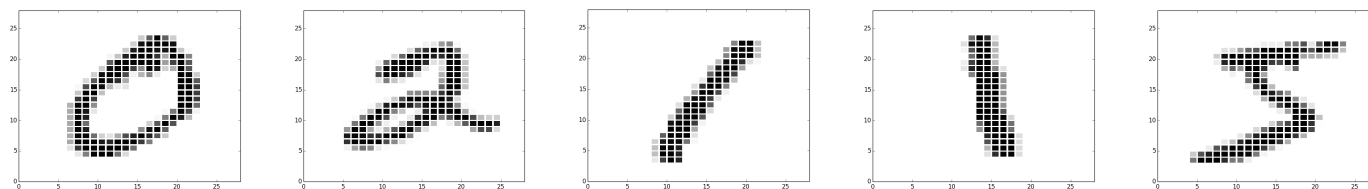
Can we predict user preferences based upon similarities between items and/or other user preferences?



# Images -> Classification

Let's consider a task: handwritten digit recognition

Given as input...



Have the computer correctly identify...

0

2

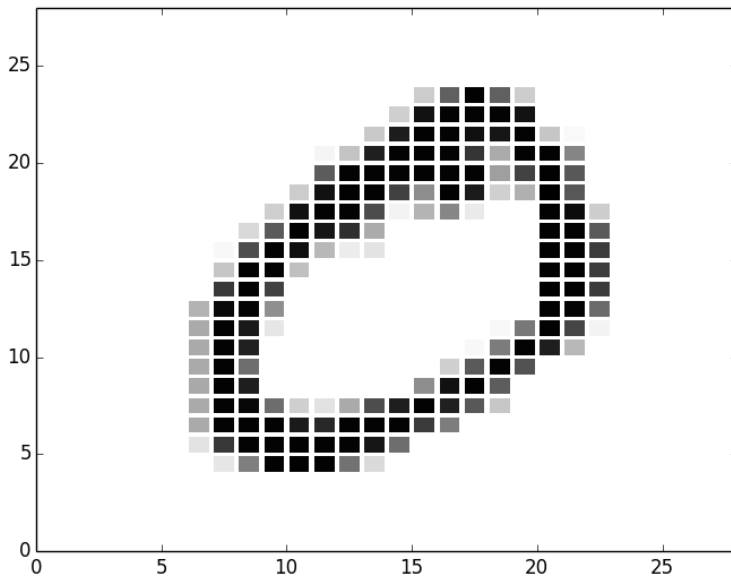
1

1

5



# Table-Based Data

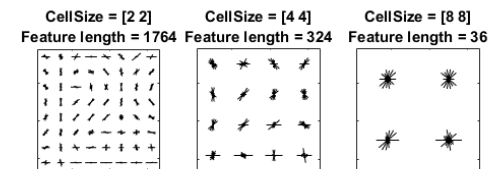


*example, instance*

Unit of input

Composed of *features*  
(or *attributes*)

- In this case, we could represent each digit via raw pixels:  
28x28=784-pixel **vector** of greyscale values [0-255]
  - **Dimensionality**: number of features per instance (|vector|)
- But other **data representations** are possible, and might be advantageous



- In general, the problem of **feature selection** is challenging



# Instances/Features = Table

Features

Outlook	Temperature	Humidity	Windy	Play
sunny	85	85	false	no
sunny	80	90	true	no
overcast	83	86	false	yes
rainy	70	96	false	yes
rainy	68	80	false	yes
rainy	65	70	true	no
overcast	64	65	true	yes
sunny	72	95	false	no
sunny	69	70	false	yes
rainy	75	80	false	yes
sunny	75	70	true	yes
overcast	72	90	true	yes
overcast	81	75	false	yes
rainy	71	91	true	no

**Instance**



# “Target” Feature

When trying to predict a particular feature given the others

***target, label, class, concept, dependent***

Outlook	Temperature	Humidity	Windy	Play
sunny	85	85	false	no
sunny	80	90	true	no
overcast	83	86	false	yes
rainy	70	96	false	yes
rainy	68	80	false	yes
rainy	65	70	true	no
overcast	64	65	true	yes
sunny	72	95	false	no
sunny	69	70	false	yes
rainy	75	80	false	yes
sunny	75	70	true	yes
overcast	72	90	true	yes
overcast	81	75	false	yes
rainy	71	91	true	no



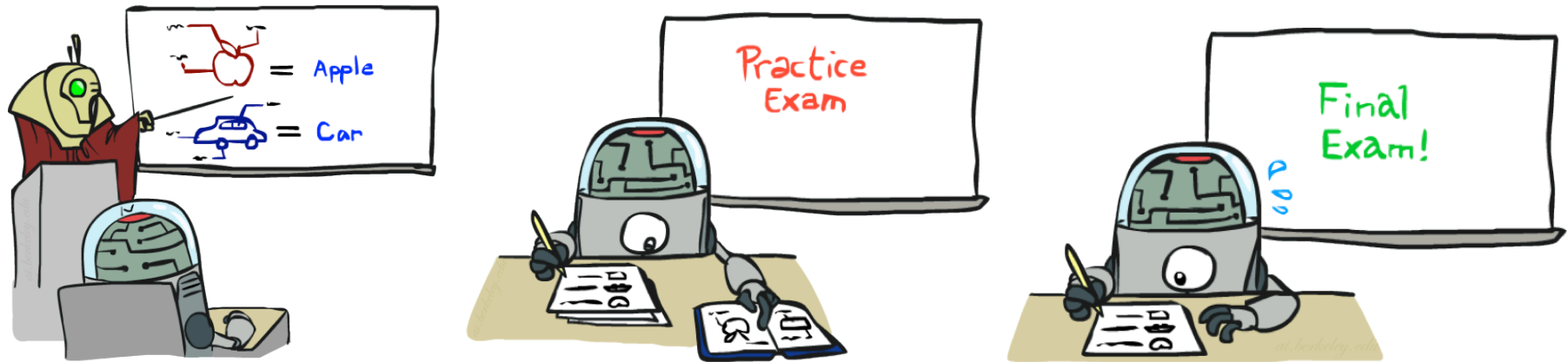
# Machine Learning Tasks

- ***Supervised***
  - Given a ***training set*** and a target variable, ***generalize***; measured over a ***testing set***
- ***Unsupervised***
  - Given a dataset, find patterns/parameters
- ***Reinforcement***
  - Learn an optional action ***policy*** over time; given an environment that provides states, affords actions, and provides feedback as numerical ***reward***, maximize the ***expected*** future reward

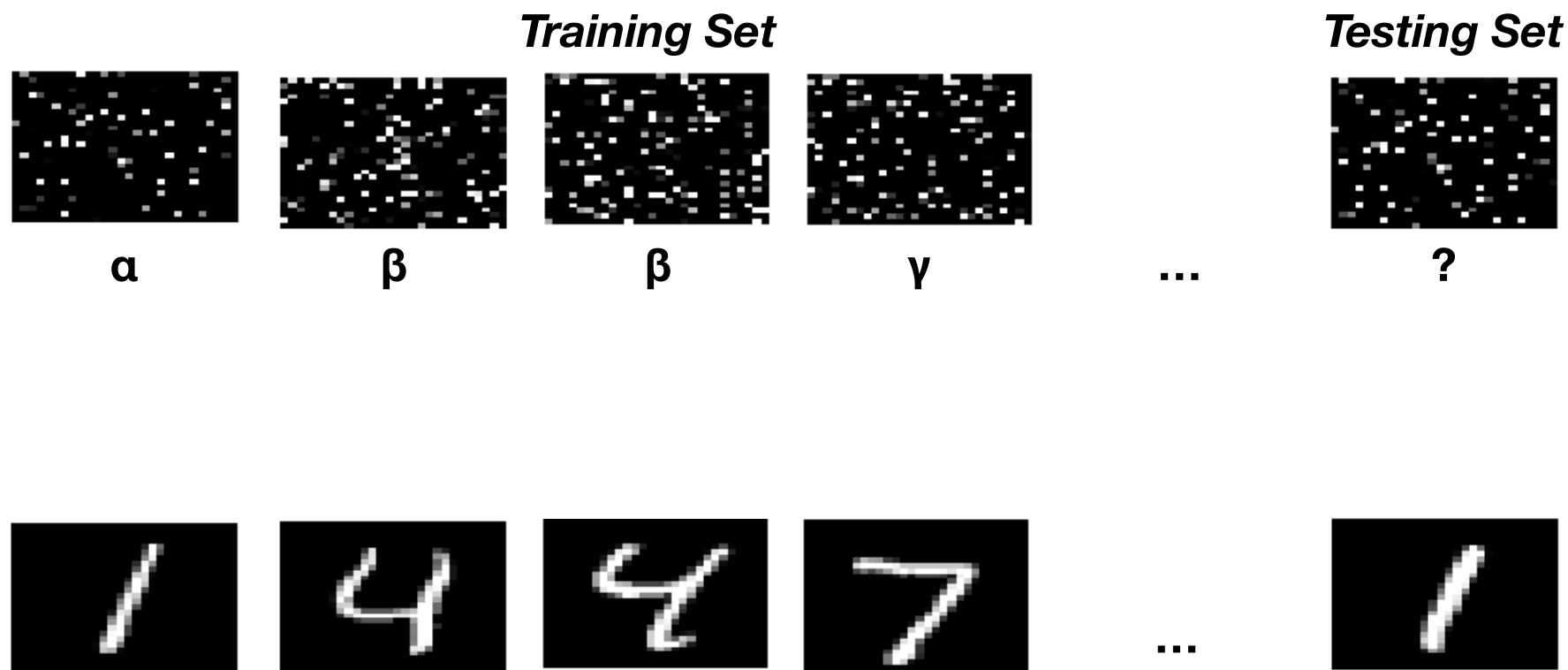




# Supervised Learning (1)



# Supervised Learning (2)



Goal: *generalization*



# Supervised Tasks (1)

## Classification

- Discrete target
- Binary vs. multi-class



SepalLength	SepalWidth	PetalLength	PetalWidth	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa



# Supervised Tasks (2)

## Regression

- Continuous target

mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
18	8	307	130	3504	12	70	1	chevrolet chevelle malibu
15	8	350	165	3693	11.5	70	1	buick skylark 320
18	8	318	150	3436	11	70	1	plymouth satellite
16	8	304	150	3433	12	70	1	amc rebel sst
17	8	302	140	3449	10.5	70	1	ford torino
15	8	429	198	4341	10	70	1	ford galaxie 500
14	8	454	220	4354	9	70	1	chevrolet impala
14	8	440	215	4312	8.5	70	1	plymouth fury iii
14	8	455	225	4425	10	70	1	pontiac catalina
15	8	390	190	3850	8.5	70	1	amc ambassador dpl
15	8	383	170	3563	10	70	1	dodge challenger se
14	8	340	160	3609	8	70	1	plymouth 'cuda 340
15	8	400	150	3761	9.5	70	1	chevrolet monte carlo
14	8	455	225	3086	10	70	1	buick estate wagon (sw)
24	4	113	95	2372	15	70	3	toyota corona mark ii
22	6	198	95	2833	15.5	70	1	plymouth duster
18	6	199	97	2774	15.5	70	1	amc hornet
21	6	200	85	2587	16	70	1	ford maverick
27	4	97	88	2130	14.5	70	3	datsun pl510
26	4	97	46	1835	20.5	70	2	volkswagen 1131 deluxe sedan
25	4	110	87	2672	17.5	70	2	peugeot 504
24	4	107	90	2430	14.5	70	2	audi 100 ls
25	4	104	95	2375	17.5	70	2	saab 99e
26	4	121	113	2234	12.5	70	2	bmw 2002



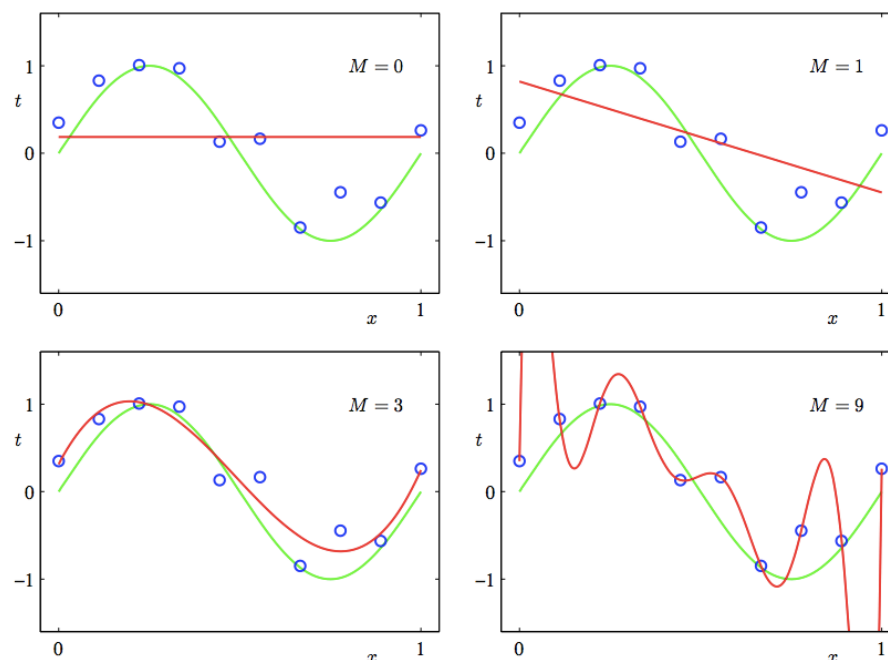
# Under/Over-fitting

**Underfitting:** the model does not capture the important relationship(s)

**Overfitting:** the model describes noise instead of the underlying relationship

## Approaches

- **Regularization**
- Robust evaluation
  - Cross validation



# Validation Set

- One approach in an ML-application pipeline is to use a ***validation*** dataset (could be a ***holdout*** from the training set)
- Each model is built using just training; the validation dataset is then used to compare performance and/or select model parameters
- But still, the final performance is only measured via an independent test set



# More Training Data = Better

- In general, the greater the amount of training data, the better we expect the learning algorithm to perform
  - But we also want reasonable amounts of validation/testing data!
- So how do we not delude ourselves, achieve high performance, *and* a reasonable expectation of future performance?



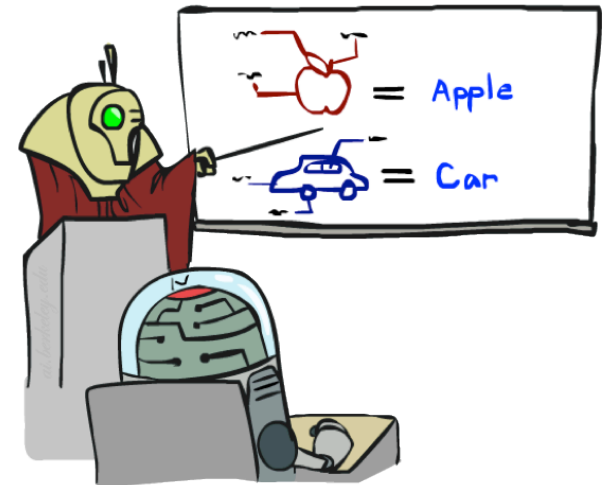
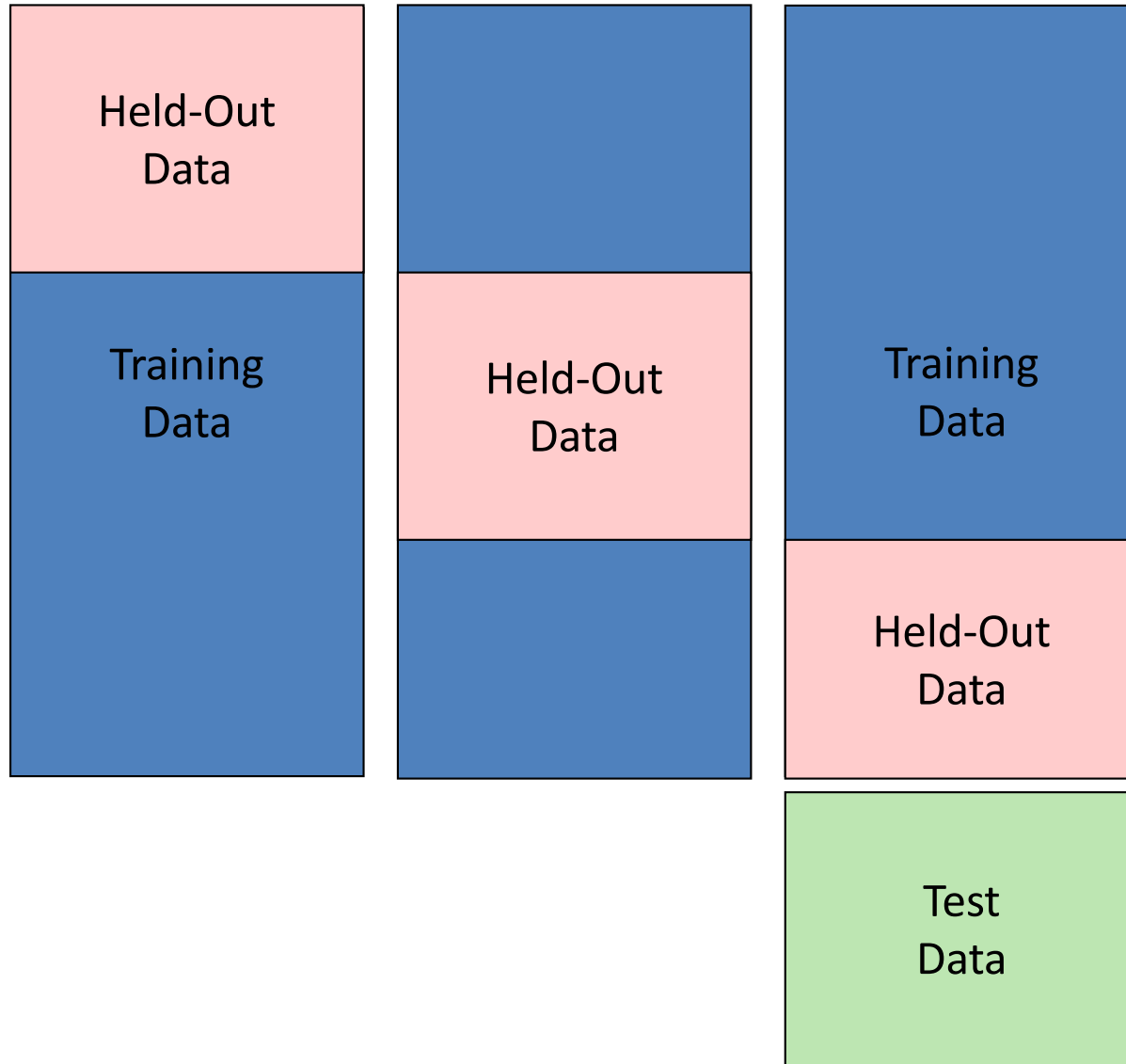
# *k*-Fold Cross-Validation

- Basic approach
  - Divide the data into  $k$  randomly selected partitions (typically 10)
  - For each, use the fold as test data, the remainder as training data (i.e. repeat the train/test process  $k$  times)
  - Average results
- To control for unfortunate outcomes in random selection, consider repeating (e.g. 10 x 10-fold cross validation = 100 train/test)
  - Expensive!





# k-Fold Cross Validation Visualized



# Common Algorithms

- Instance-based
  - **Nearest Neighbor (kNN)**
- Tree-based
  - ID3, C4.5, Random Forests
- Optimization-based
  - **Linear regression**, logistic regression, support vector machines (SVM)
- Probabilistic
  - **Naïve Bayes**
- Artificial Neural Networks
  - Backpropagation
  - Deep learning



# kNN

## Training

- Store all examples

## Testing

- Find the nearest  $k$  neighbors to target
  - Via distance function
- Vote on class

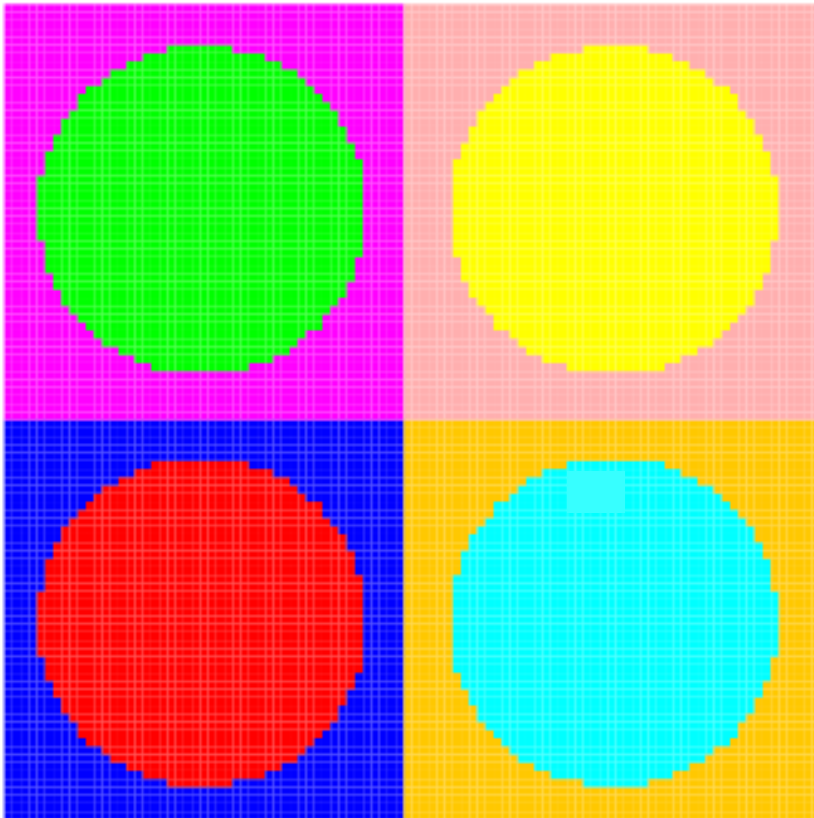


**Non-parametric** algorithm (i.e. grows with |examples|!)

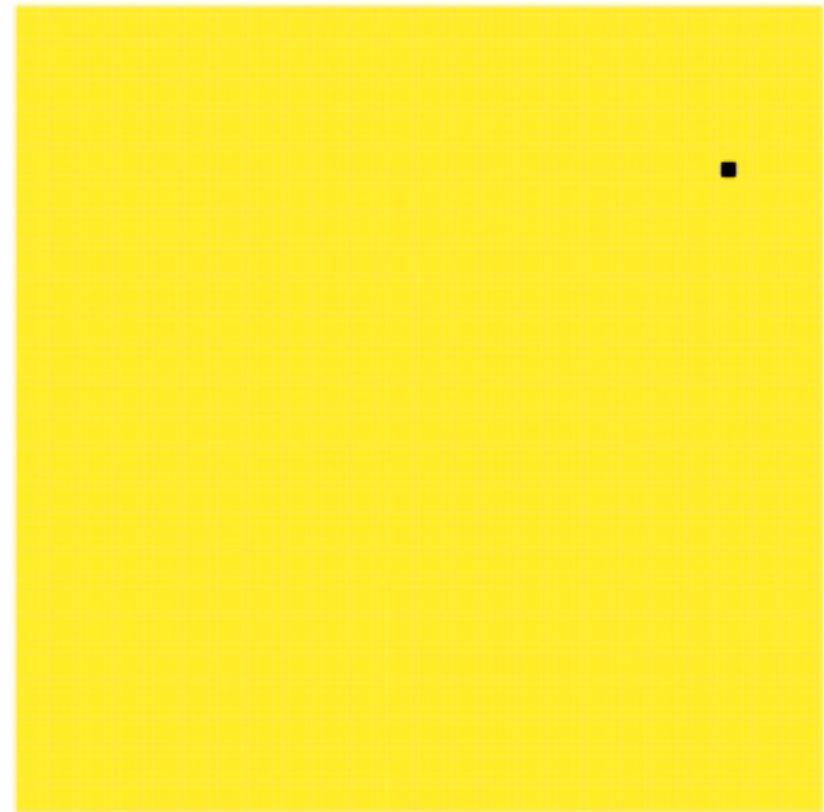


# 2D Multiclass Classification

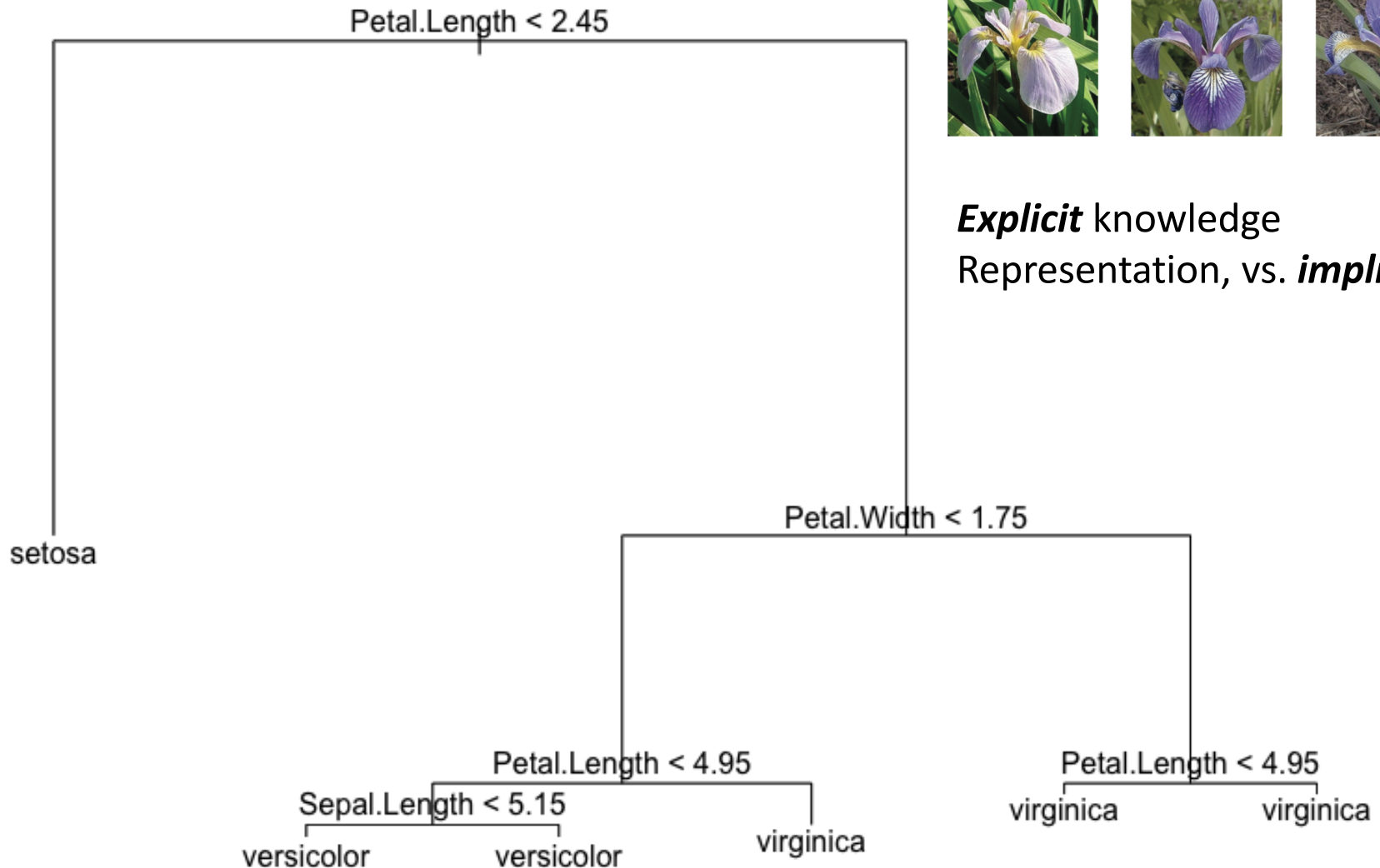
## Boundary Tree



## 1-NN via Linear Scan



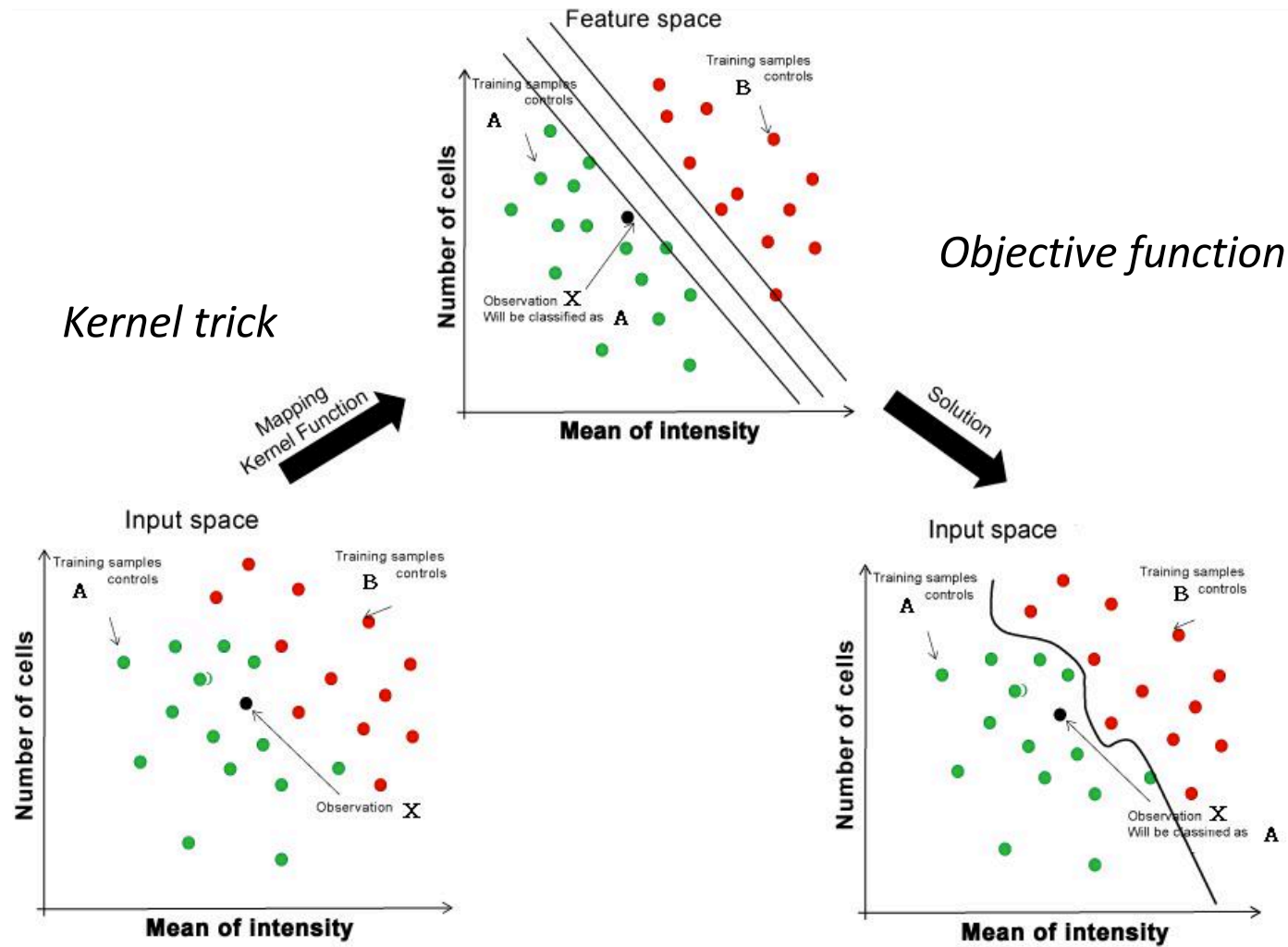
# Decision Trees/Forests



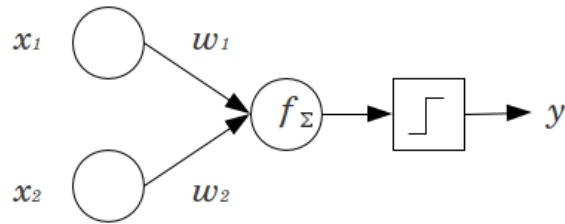
**Explicit** knowledge  
Representation, vs. **implicit**



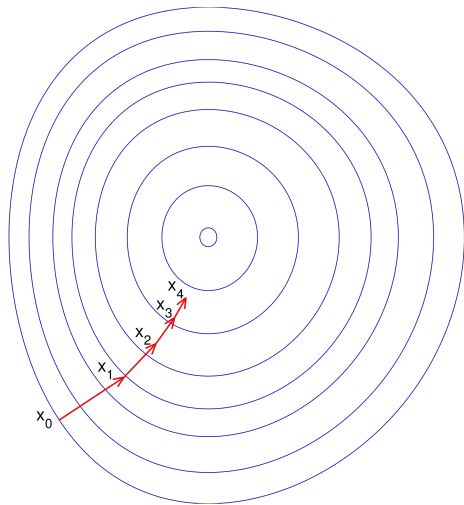
# Support Vector Machine (SVM)



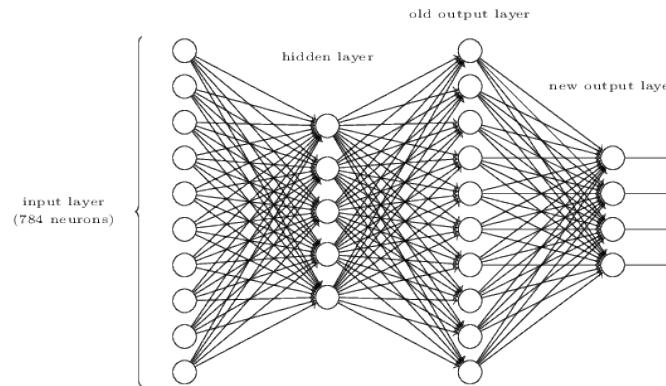
# Artificial Neural Networks (ANN)



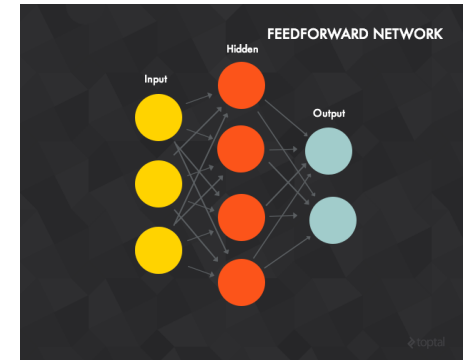
Perceptron  
*Linear classifier*



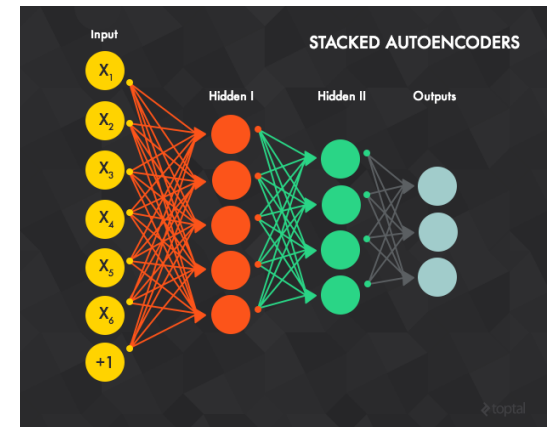
Gradient descent



Backpropagation



Feedforward vs.  
Recurrent



Deep Architectures  
*Vanishing Gradient*

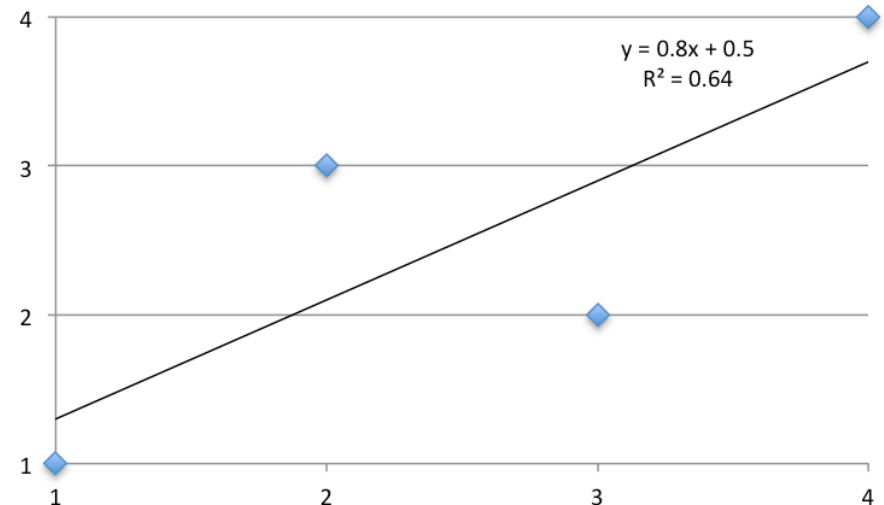


# Linear Regression

## Input

x	y
1	1
2	3
3	2
4	4

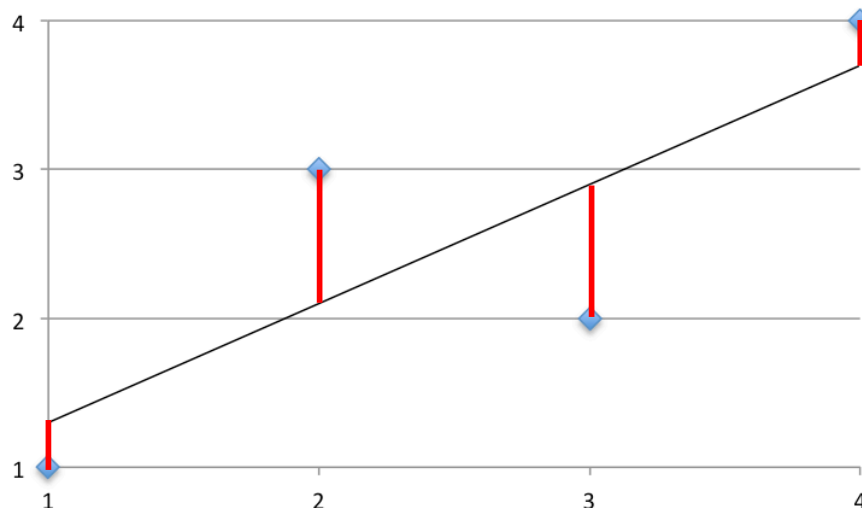
## Output





# Linear Regression as Optimization

- Why this line?
  - Minimizer **error**
- In 2D, the algorithm tries to find a slope and intercept that yields the smallest sum of the square of the error (SSE)



$$\arg \min_{m,b} \sum_{i=1}^N e_i^2 = (y_i - (mx_i + b))^2$$

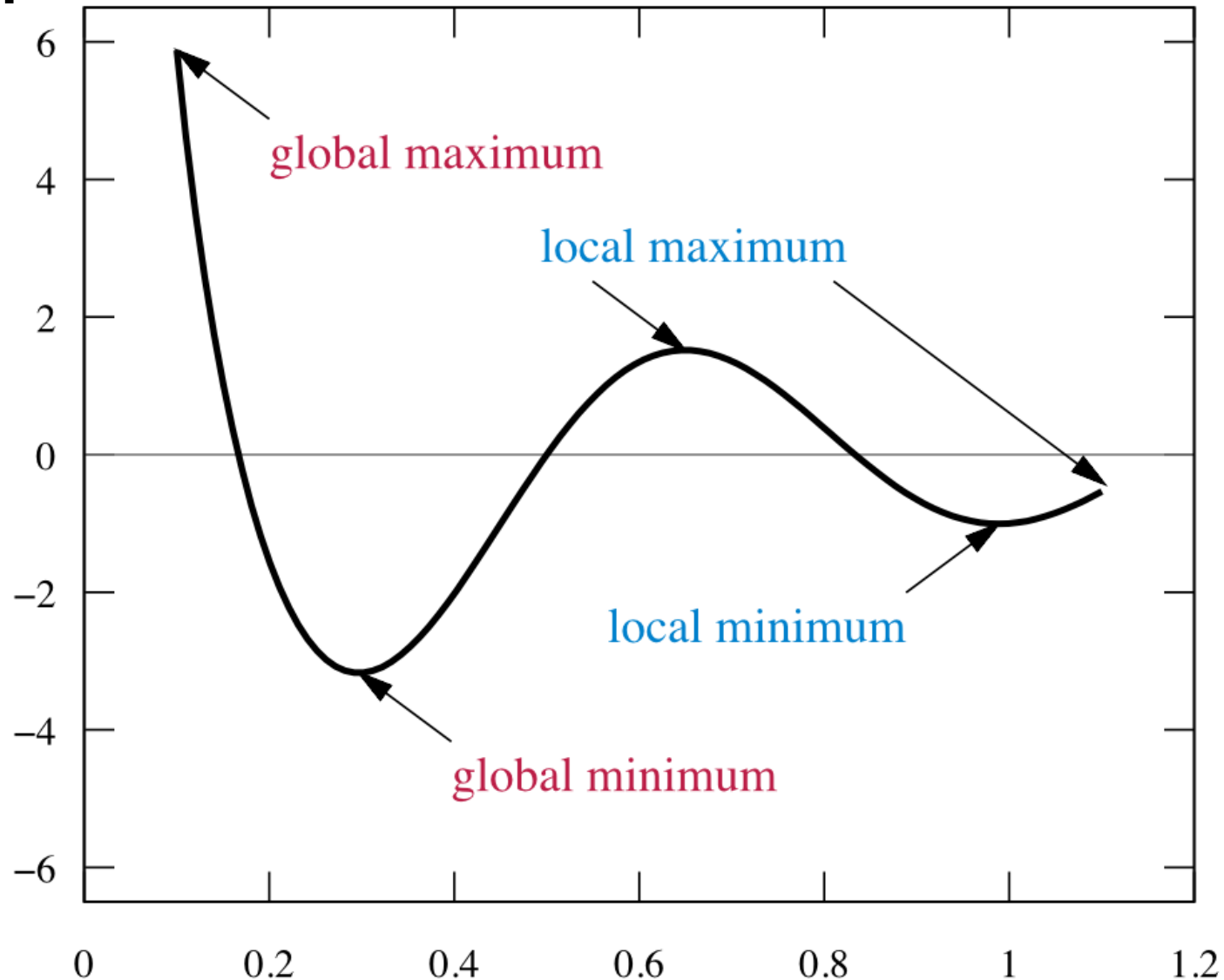


# Machine Learning via Optimization

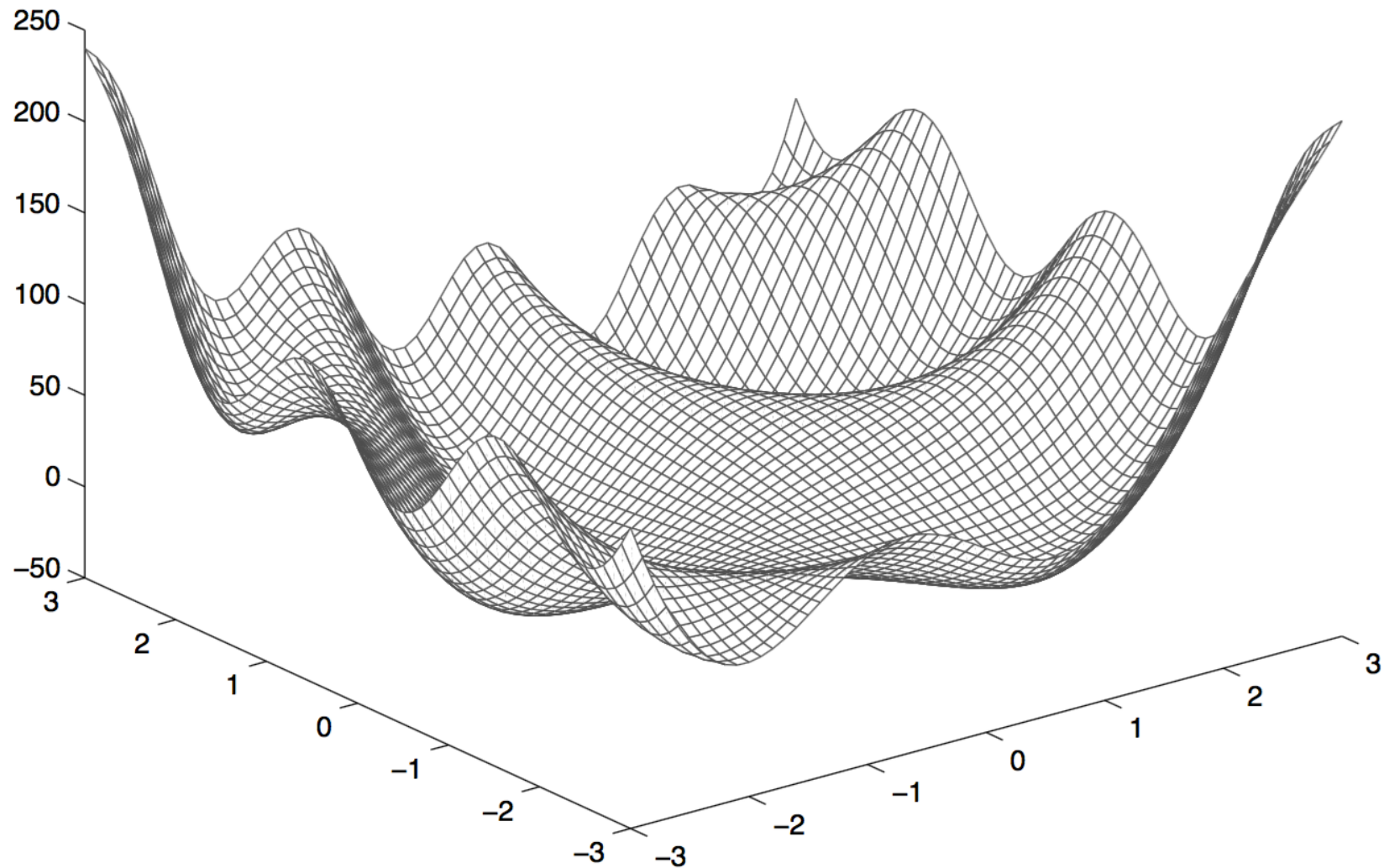
1. Define an error function
2. Find model parameters that minimize the error function given the data
  - Sometimes closed-form solution (e.g. linear)
  - Sometimes [iterative] solution [with guarantees] (e.g. convex)
  - Most of the time will require approximation
    - Iteration (limited by number, delta)
    - Softening constraints
    - Post-processing
  - ...



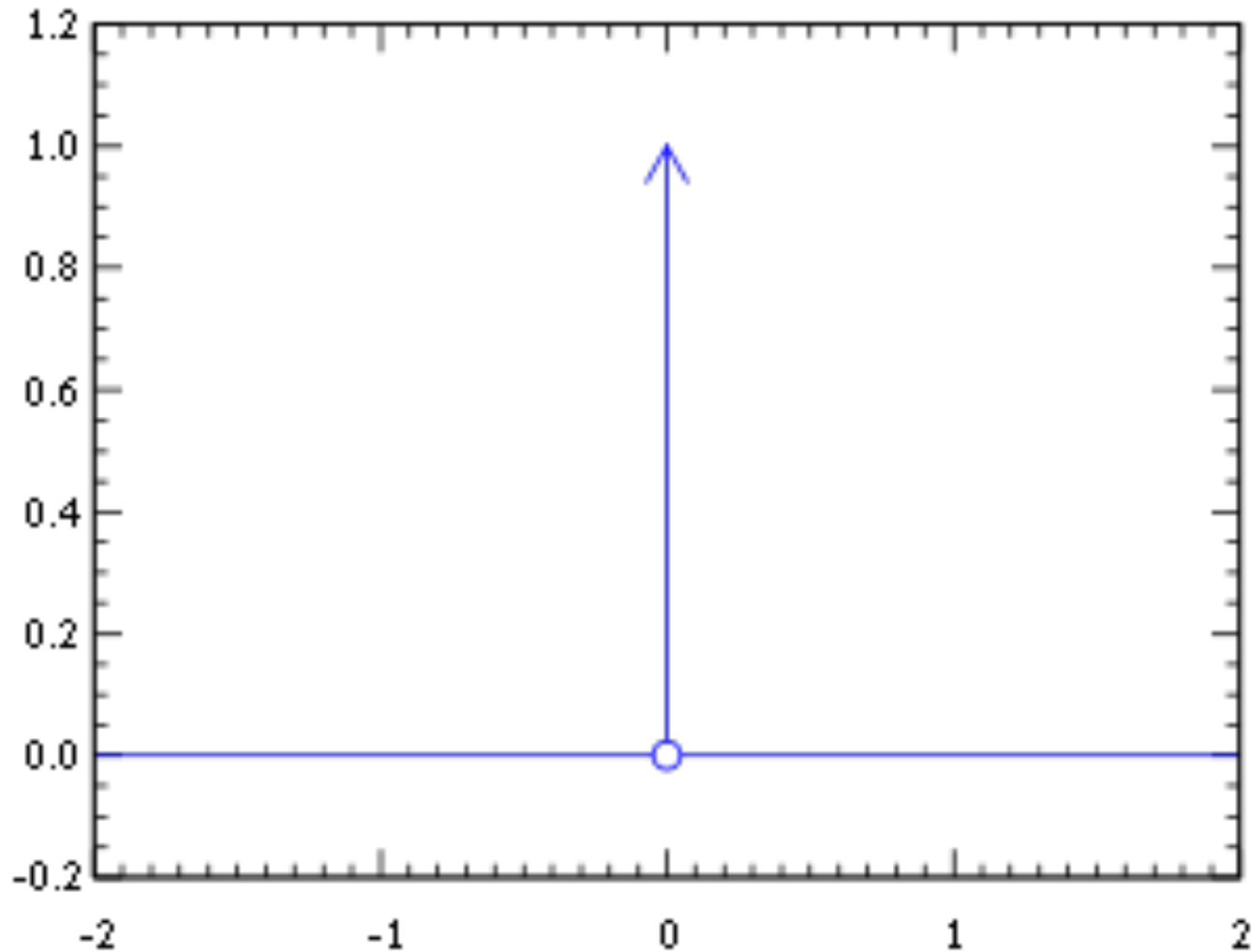
# Optimization is Hard in General



# Consider Many [Cursed] Dimensions



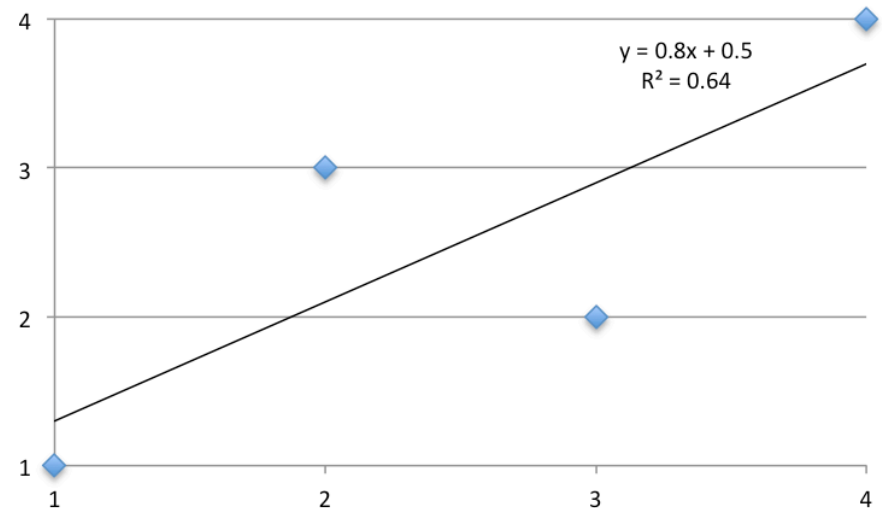
# Consider Discontinuities



# Linear Regression

## Recipe

1. Define error function
2. Find parameter values that minimize error given the data

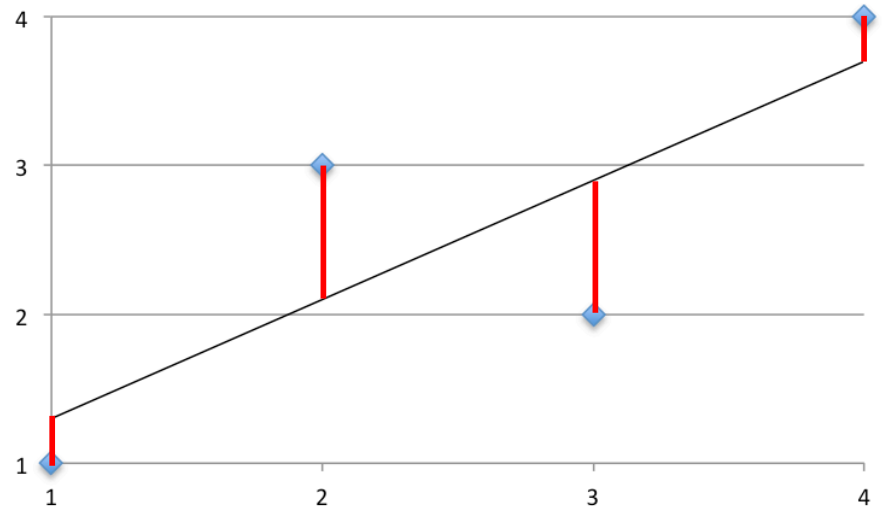


# Error Function

Sum of Squared Error (SSE)

*aka Residual Sum of Squares (RSS)*

$$\text{SSE}_{\text{line}} = \sum_{i=1}^N (y_i - f(x_i))^2 = (y_i - (mx_i + b))^2$$



# Algebra (1)

$$\begin{aligned}\text{SSE}_{\text{line}} &= \sum_{i=1}^N (y_i - (mx_i + b))^2 \\ &= \sum_{i=1}^N y_i^2 - 2y_i(mx_i + b) + (mx_i + b)^2 \\ &= \sum_{i=1}^N y_i^2 - 2mx_iy_i - 2by_i + m^2x_i^2 + 2mbx_i + b^2\end{aligned}$$





# Algebra (2)

$$\sum_{i=1}^N a_i = N\bar{a}$$

**SO...**

$$\begin{aligned} \text{SSE}_{\text{line}} &= \sum_{i=1}^N y_i^2 - 2mx_i y_i - 2by_i + m^2 x_i^2 + 2mbx_i + b^2 \\ &= N\bar{y}^2 - 2Nm\bar{x}\bar{y} - 2Nb\bar{y} + Nm^2\bar{x}^2 + 2Nmb\bar{x} + Nb^2 \end{aligned}$$



# Recall: Critical Points

- For a differentiable function of several variables, a critical point is a value in its domain where all partial derivatives are zero
- So to find the point at which error is minimized, we take partial derivatives of the error function w.r.t. the parameters, set these equal to 0, solve



# Calculus (1)

$$\text{SSE}_{\text{line}} = N\bar{y}^2 - 2Nm\bar{x}\bar{y} - 2Nb\bar{y} + Nm^2\bar{x}^2 + 2Nmb\bar{x} + Nb^2$$

$$\frac{\partial \text{SSE}_{\text{line}}}{\partial m} = -2N\bar{x}\bar{y} + 2Nm\bar{x}^2 + 2Nb\bar{x} = 0$$

$$\frac{\partial \text{SSE}_{\text{line}}}{\partial b} = -2N\bar{y} + 2Nm\bar{x} + 2Nb = 0$$



# Algebra (3)

$$\frac{\partial \text{SSE}_{\text{line}}}{\partial b} = -2N\bar{y} + 2Nm\bar{x} + 2Nb = 0$$

$$0 = -2N\bar{y} + 2Nm\bar{x} + 2Nb$$

$$0 = -\bar{y} + m\bar{x} + b$$

$$\bar{y} = m\bar{x} + b \quad (\bar{x}, \bar{y})$$



# Algebra (4)

$$\frac{\partial \text{SSE}_{\text{line}}}{\partial m} = -2N\overline{xy} + 2Nm\overline{x^2} + 2Nb\overline{x} = 0$$

$$0 = -2N\overline{xy} + 2Nm\overline{x^2} + 2Nb\overline{x}$$

$$0 = -\overline{xy} + m\overline{x^2} + b\overline{x}$$

$$\overline{xy} = m\overline{x^2} + b\overline{x}$$

$$\frac{\overline{xy}}{\overline{x}} = m\frac{\overline{x^2}}{\overline{x}} + b$$

$$\left(\frac{\overline{x^2}}{\overline{x}}, \frac{\overline{xy}}{\overline{x}}\right)$$



# And Finally...

$$(\bar{x}, \bar{y})$$

$$\left(\frac{\overline{x^2}}{\bar{x}}, \frac{\overline{xy}}{\bar{x}}\right)$$

$$m = \frac{\frac{\overline{xy}}{\bar{x}} - \bar{y}}{\frac{\overline{x^2}}{\bar{x}} - \bar{x}}$$

$$= \frac{\overline{xy} - \bar{x}\bar{y}}{\overline{x^2} - \bar{x}^2}$$

$$\bar{y} = m\bar{x} + b$$

$$b = \bar{y} - m\bar{x}$$



# A Quick Aside: Meaning of Slope (1)

- **Covariance.** A measure of how much two random variables change together

$$\text{Cov}(X, Y) = \sigma(X, Y) = \text{E}[(X - \text{E}[X])(Y - \text{E}[Y])]$$

- If both  $X/Y$  increase relative to their means, positive; else negative

$$\text{Cov}(X, X) = \text{Var}(X)$$



# Meaning of Slope (2)

$$\begin{aligned}\text{Cov}(X, Y) = \sigma(X, Y) &= \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])] \\ &= \mathbb{E}[XY - X\mathbb{E}[Y] - Y\mathbb{E}[X] + \mathbb{E}[X]\mathbb{E}[Y]] \\ &= \mathbb{E}[XY] - \mathbb{E}[X\mathbb{E}[Y]] - \mathbb{E}[Y\mathbb{E}[X]] + \mathbb{E}[\mathbb{E}[X]\mathbb{E}[Y]] \\ &= \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y] - \mathbb{E}[X]\mathbb{E}[Y] + \mathbb{E}[X]\mathbb{E}[Y] \\ &= \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]\end{aligned}$$





# Meaning of Slope (3)

Given data, we can approximate the expected value of a random variable by the sample mean

$$E[A] \approx \bar{A}$$



## And Finally...

$$\begin{aligned}\text{Cov}(X, Y) &= \text{E}[XY] - \text{E}[X]\text{E}[Y] \\ &= \overline{XY} - \bar{X}\bar{Y}\end{aligned}$$

But remember...

$$\begin{aligned}m &= \frac{\overline{xy} - \bar{x}\bar{y}}{\overline{x^2} - \bar{x}^2} \\ &= \frac{\overline{xy} - \bar{x}\bar{y}}{\overline{xx} - \bar{x}\bar{x}}\end{aligned}$$

$$\begin{aligned}m &= \frac{\text{Cov}(X, Y)}{\text{Cov}(X, X)} \\ &= \frac{\text{Cov}(X, Y)}{\text{Var}(X)}\end{aligned}$$



# Evaluating Linear Regression

The natural question to ask: to what extent is the line capturing the variation in  $y$  as a result of the variation in  $x$

- To quantify: look at the ratio of the error of the line and the error of  $y$

$$R^2 = 1 - \frac{SSE_{\text{line}}}{SSE_{\bar{Y}}}$$

$$SSE_{\bar{Y}} = (y_1 - \bar{Y})^2 + (y_2 - \bar{Y})^2 + \dots$$



# The Multi-Dimensional Case

- The preceding discussion assumed a single independent variable ( $x$ ), and thus derived a single slope ( $m$ ) and intercept to linearly approximate the dependent variable ( $y$ )
- We now consider the multi-dimensional case, where each independent variable ( $x_i$ ) is associated with a slope/intercept



# Problem Setup

We begin with an analogous representation

$$Y = XB + e$$

where...

- **Y** is  $N \times 1$
- **X** is  $N \times (k+1)$ ; extra 1 to multiply intercept
- **B** is  $(k+1) \times 1$ ; first intercept, then coefficients
- **e** is  $N \times 1$



# k-Dimensional Linear Regression

$$\begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_N \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdot & \cdot & \cdot & x_{1k} \\ 1 & x_{21} & x_{22} & \cdot & \cdot & \cdot & x_{2k} \\ \cdot & \cdot & & & & & \\ \cdot & \cdot & & & & & \\ \cdot & \cdot & & & & & \\ 1 & x_{N1} & x_{N2} & \cdot & \cdot & \cdot & x_{Nk} \end{bmatrix} \begin{bmatrix} b \\ m_1 \\ m_2 \\ \cdot \\ \cdot \\ \cdot \\ m_k \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \cdot \\ \cdot \\ \cdot \\ e_N \end{bmatrix}$$



# Step 1: Error Function

- We will use the same error method as last time, which is SSE (i.e. square the difference between  $Y$  and  $XB$ )

$$\begin{aligned} \text{SSE} &= e^T e \\ &= (Y - XB)^T (Y - XB) \end{aligned}$$



# Matrix Algebra (1)

$$\begin{aligned}\text{SSE} &= (Y - XB)^\top (Y - XB) \\ &= (Y^\top - B^\top X^\top)(Y - XB) \\ &= Y^\top Y - Y^\top XB - B^\top X^\top Y + B^\top X^\top XB \\ &= Y^\top Y - 2Y^\top XB + B^\top X^\top XB\end{aligned}$$





# Matrix Calculus (1)

$$\text{SSE} = Y^T Y - 2Y^T X B + B^T X^T X B$$

$$\frac{\partial \text{SSE}}{\partial B} = -2X^T Y + 2X^T X B$$



# Matrix Algebra (2)

$$0 = -2X^T Y + 2X^T X B$$

$$-2X^T X B = -2X^T Y$$

$$X^T X B = X^T Y$$

$$B = (X^T X)^{-1} X^T Y$$



Look familiar?



# Naïve Bayes Classification

- Choose classification that maximizes **posterior** probability given the data (i.e. a Bayesian classifier)
- Simplifying assumption: assume all features of an instance contribute *independently* to the **likelihood** of observing the data



# Axiom of Conditional Probability

Conditional Probability



$$P(A, B) = P(A|B) \cdot P(B)$$
$$= P(B|A) \cdot P(A)$$



Joint Probability

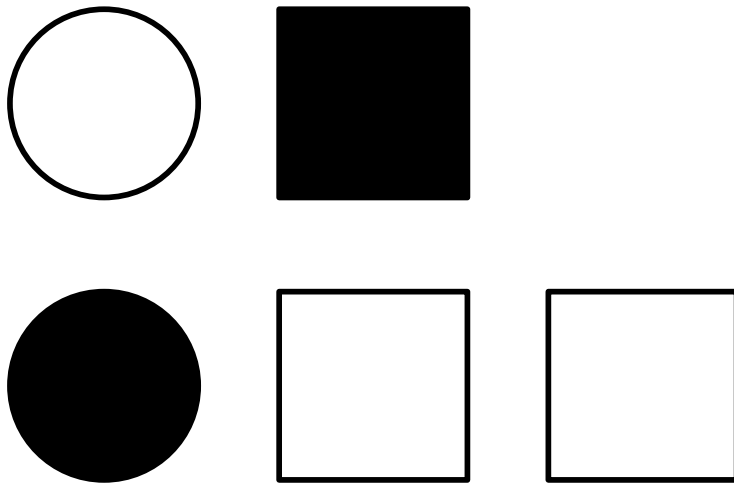


# Simple Example

- A = filled
- B = shape is square

$$P(A) = \frac{2}{5} \quad P(B) = \frac{3}{5}$$

$$P(A|B) = \frac{1}{3} \quad P(B|A) = \frac{1}{2}$$



$$P(A, B) =$$

$$=$$

$$= \frac{1}{5}$$



# Bayes' Rule

$$P(A, B) = P(B, A)$$

$$P(A|B) \cdot P(B) = P(B|A) \cdot P(A)$$

$$P(A|B) = \frac{\overset{\text{Likelihood}}{P(B|A)} \cdot \overset{\text{Prior}}{P(A)}}{\underset{\text{Evidence/Support}}{P(B)}}$$



# Why Does Bayes' Rule Matter?

Often we know/can estimate likelihood and prior information easier than the posterior

$$P(\text{Hypothesis}|\text{Data}) = \frac{P(\text{Data}|\text{Hypothesis}) \cdot P(\text{Hypothesis})}{P(\text{Data})}$$

## Clinical example

- A: person has cancer
- B: person smokes

Easy from historical data

- $P(A) = 10\%$
- $P(B) = 40\%$
- $P(B|A) = 80\%$

$$P(A|B) = 20\%$$



# Learning via Probability Estimates

- Consider the posterior probability distribution over a discrete set of classes ( $C$ ) and fixed set of features ( $\mathbf{x}$ ; each continuous or discrete)

$$P(C_k | \mathbf{x}) = \frac{P(C_k) \cdot P(\mathbf{x} | C_k)}{P(\mathbf{x})}$$

- The maximum a posteriori (MAP) decision rule says to select the class that maximizes the posterior, thus...

$$\hat{y} = \arg \max_{k \in \{1 \dots K\}} \frac{P(C_k) \cdot P(\mathbf{x} | C_k)}{P(\mathbf{x})}$$





# Note

- The evidence term is only dependent on the data, and applies a normalizing constant (i.e. p's sum to 1)

$$\begin{aligned} P(\mathbf{x}) &= \sum_k P(\mathbf{x}, C_k) \\ &= \sum_k P(\mathbf{x}|C_k) \cdot P(C_k) \end{aligned}$$

- For classification we care only about selecting the maximum value, and so we can maximize the numerator and ignore the denominator

$$\hat{y} = \arg \max_{k \in \{1 \dots K\}} P(C_k) P(\mathbf{x}|C_k)$$



# How Much Data is Necessary?

$$\hat{y} = \arg \max_{k \in \{1 \dots K\}} P(C_k) P(\mathbf{x} | C_k)$$

- We can reasonably estimate the class prior via data (e.g. 2 classes ~ 100 points)
- However, likelihood is exponential
  - $P(\{0,0,0\dots,0\} | 0) \times 100$
  - $P(\{0,0,0\dots,0\} | 1) \times 100$
  - $P(\{0,0,0\dots,1\} | 0) \times 100$
  - $P(\{0,0,0\dots,1\} | 1) \times 100$
  - ...



# Feasibility via Conditional Independence

- The term **naïve** refers to the algorithmic assumption that each feature is conditionally independent of *every other* feature
  - This has the effect of reducing the necessary estimation data from exponential to linear
- In practice, while the independence assumption typically may not hold, Naïve Bayes works surprisingly well and is efficient for very large data sets with many features



# Conditional Independence

$X$  is conditionally independent of  $Y$  given  $Z$ , if and only if the probability distribution governing  $X$  is independent of the value of  $Y$  given  $Z$

$$(\forall i, j, k) P(X = x_i | Y = y_j, Z = z_k) = P(X = x_i | Z = z_k)$$



# Deriving Naïve Bayes

Consider the two-feature example:

$$P(X_1, X_2|Y) = P(X_1|X_2, Y) \cdot P(X_2|Y)$$

Now apply the conditional independence assumption...

$$= P(X_1|Y) \cdot P(X_2|Y)$$



# More Generally...

$$\begin{aligned}P(X_1, \dots, X_n | C_k) &= P(X_1 | C_k) \cdot P(X_2, \dots, X_n | C_k, X_1) \\ &= P(X_1 | C_k) \cdot P(X_2 | C_k, X_1) \cdot P(X_3, \dots, X_n | C_k, X_1, X_2) \\ &= \dots\end{aligned}$$

where...

$$P(X_i | C_k, X_j) = P(X_i | C_k)$$

$$P(X_i | C_k, X_j, X_q) = P(X_i | C_k)$$

$$P(X_i | C_k, X_j, X_q, \dots) = P(X_i | C_k)$$



And so...

$$\hat{y} = \arg \max_{k \in \{1 \dots K\}} P(C_k) \cdot P(\mathbf{x} | C_k)$$

$$= \arg \max_{k \in \{1 \dots K\}} P(C_k) \cdot \prod_{i=1}^n P(x_i | C_k)$$



# Parameter Estimation – Prior

- Default approach
  - $(\# \text{ examples of class}) / (\# \text{ examples})$
- Could also assume equiprobable
  - $1/(\# \text{ distinct classes})$





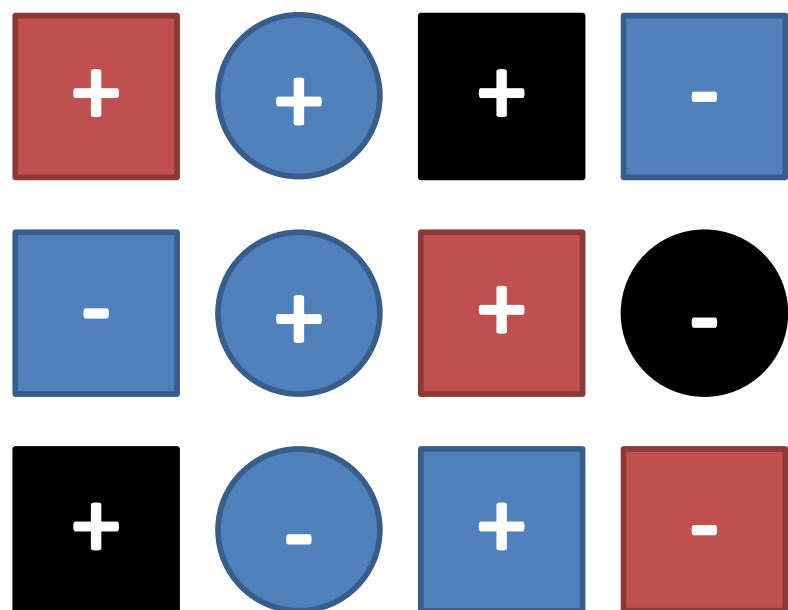
# Parameter Estimation - Likelihood

- For discrete feature values, can assume a **multinomial distribution** and use the maximum likelihood estimate (MLE)
- For continuous values, a common assumption is that for each discrete class label the distribution of each continuous feature is **Gaussian**



# Example

## Dataset



Color = {Red, Blue, Black, Orange}

Shape = {Square, Circle}

## Input



$$P(+)=\frac{7}{12}$$

$$P(-)=\frac{5}{12}$$

$$P(\text{Blue}|+)=\frac{3}{7}$$

$$P(\text{Blue}|-) = \frac{3}{5}$$

$$P(\text{Square}|+)=\frac{5}{7}$$

$$P(\text{Square}|-) = \frac{3}{5}$$

$$P(x|+) = \frac{3}{7} \cdot \frac{5}{7} \sim 0.31$$


$$P(x|-) = \frac{3}{5} \cdot \frac{3}{5} = 0.36$$

$$P(+|\text{Blue, Square}) = \frac{7}{12} \cdot \frac{3}{7} \cdot \frac{5}{7} \sim 0.18 \quad \checkmark$$

$$P(-|\text{Blue, Square}) = \frac{5}{12} \cdot \frac{3}{5} \cdot \frac{3}{5} = 0.15$$



# Additive Smoothing

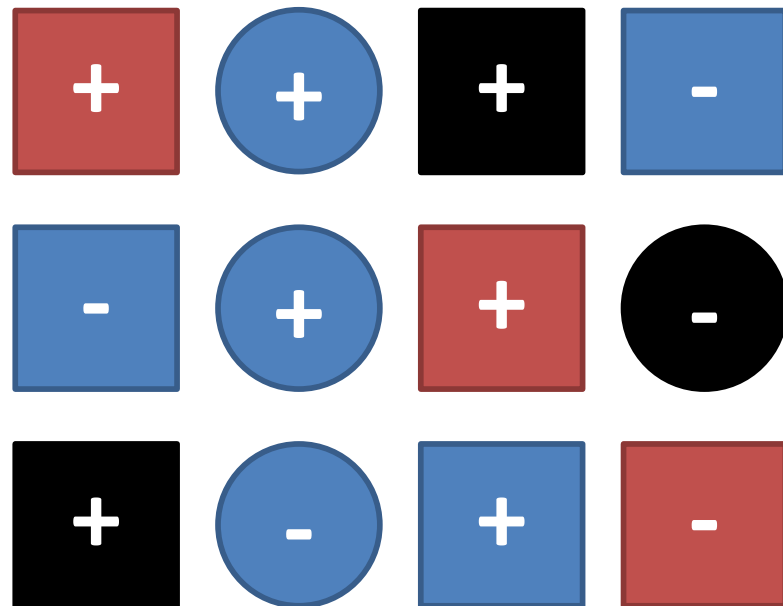
- An issue that arises in the calculation is what to do when evaluating a feature value you haven't seen (e.g. )
- To accommodate, use additive smoothing
  - $d$  = feature dimensionality
  - $\alpha$  = smoothing parameter/strength ( $\geq 0$ )
    - $0$  = no smoothing
    - $< 1$  = Lidstone smoothing
    - $\geq 1$  = Laplace smoothing

$$\frac{x + \alpha}{N + \alpha d}$$



# Example, Laplace Smoothing

**Dataset**



**Input**



$$P(+)=\frac{7}{12} \qquad P(-)=\frac{5}{12}$$

$$P(\text{Orange}|+)=\frac{0+1}{7+4}=\frac{1}{11} \qquad P(\text{Orange}|-)=\frac{0+1}{5+4}=\frac{1}{9}$$

$$P(\text{Square}|+)=\frac{5}{7} \qquad P(\text{Square}|-)=\frac{3}{5}$$

$$P(\mathbf{x}|+)=\frac{1}{11} \cdot \frac{5}{7} \sim 0.06$$

$$P(\mathbf{x}|-)=\frac{1}{9} \cdot \frac{3}{5} \sim 0.07$$

Color = {Red, Blue, Black, Orange}

Shape = {Square, Circle}

$$P(+|\text{Orange, Square})=\frac{7}{12} \cdot \frac{1}{11} \cdot \frac{5}{7} \sim 0.04 \quad \checkmark$$

$$P(-|\text{Orange, Square})=\frac{5}{12} \cdot \frac{1}{9} \cdot \frac{3}{5} \sim 0.03$$



# Gaussian MLE Estimate

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i$$

		Humidity	Mean	Std. Dev.
Play Golf	yes	86 96 80 65 70 80 70 90 75	79.1	10.2
	no	85 90 70 95 91	86.2	9.7

$$\sigma = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \mu)^2}$$

Humidity = 74

$$P(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$P(\text{humidity} = 74 | \text{play} = \text{yes}) = \frac{1}{\sqrt{2\pi}(10.2)} e^{-\frac{(74-79.1)^2}{2(10.2)^2}} = 0.0344$$

$$P(\text{humidity} = 74 | \text{play} = \text{no}) = \frac{1}{\sqrt{2\pi}(9.7)} e^{-\frac{(74-86.2)^2}{2(9.7)^2}} = 0.0187$$



# Practical Issues

- When multiplying many small fractions together you may suffer from **underflow**, resulting in the computer rounding to 0
- To account for this, it is common to take the [natural] log of probabilities and sum them:  $\log(a*b) = \log(a) + \log(b)$ 
  - Remember: all we care about is the argmax for classification



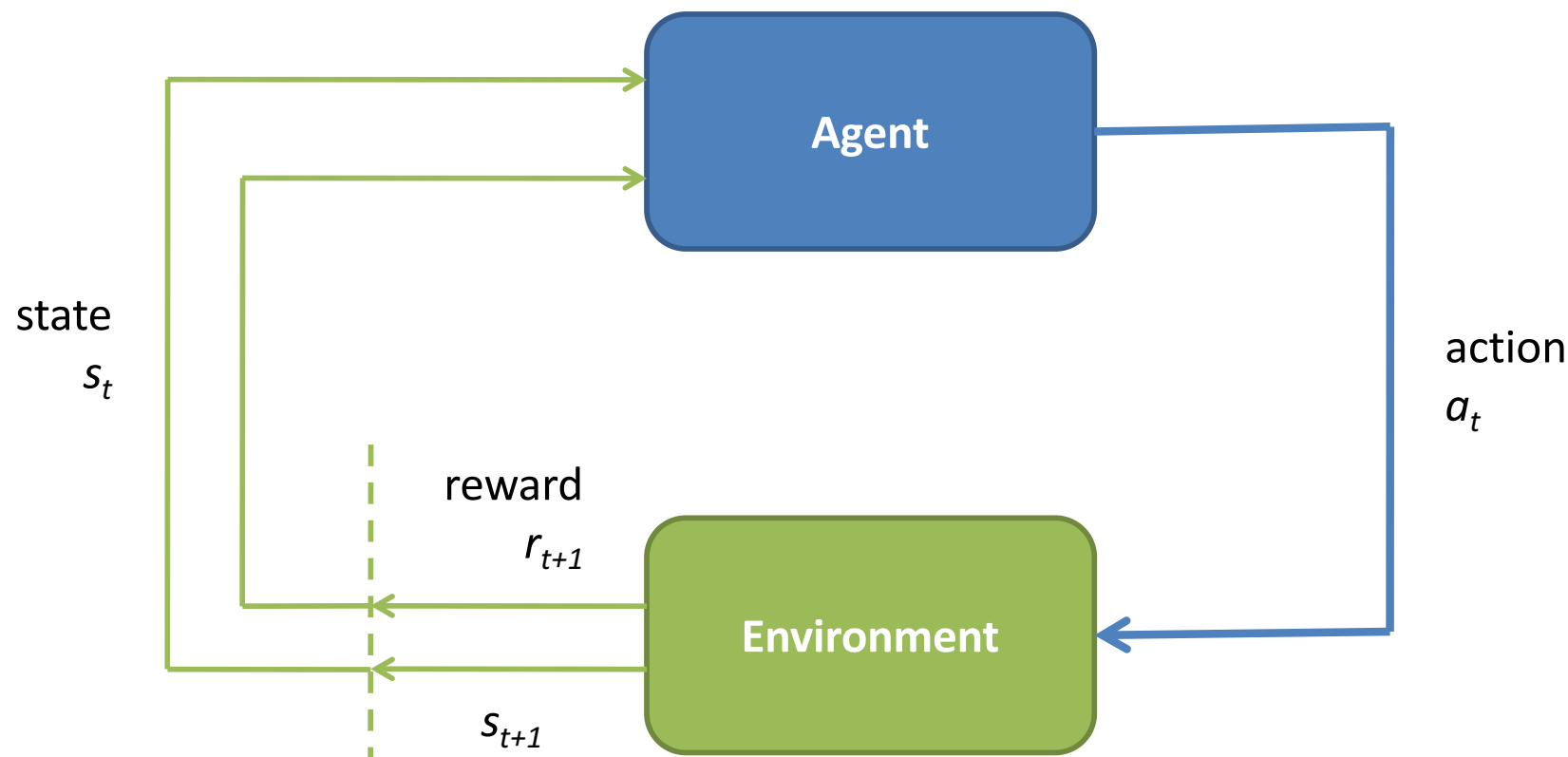
# Reinforcement Learning (RL)

Choose actions to maximize future reward



# The RL Cycle

**Issues.** credit assignment, exploration vs. exploitation, reward function, ...





# Temporal Difference (TD) Learning

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

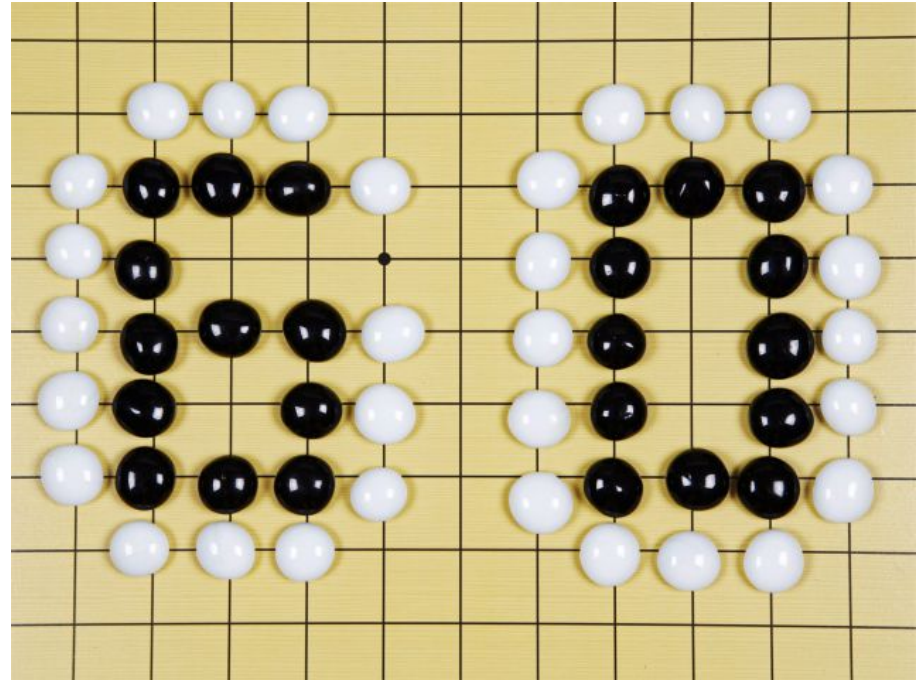


- Evidence that some neurons (dopamine) operate similarly
- Lead to world-class play via TD-Gammon (neural network trained via TD-learning)

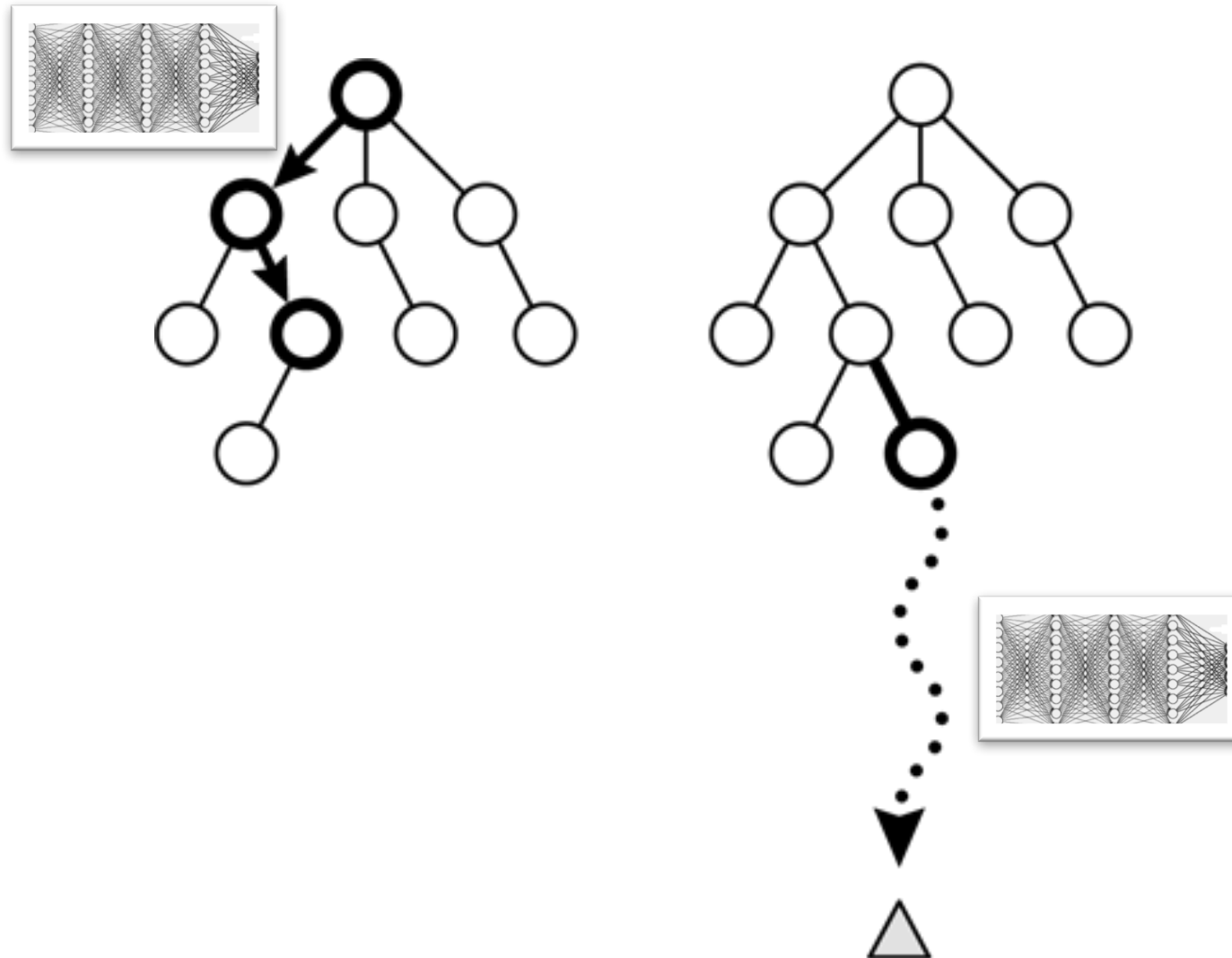


# AI + ML = AlphaGo

- Until recently, AI was not competitive at champion level
  - 2015: beat Fan Hui, European champion (2-dan; 5-0)
  - 2016: beat Lee Sedol, one of the best players in the world (9-dan; 4-1)
  - 2017: beat Ke Jie, #1 in the world (9-dan; 3-0)
- MCTS + ANNs for **policy** (what to do) and **evaluation** (how good is a board state)



# MCTS + 2xANNs



# Unsupervised Learning

Find structure or patterns in data

## Tasks

- Clustering
- Dimensionality reduction
- Density estimation
- Discovering graph structure
- Matrix completion

...



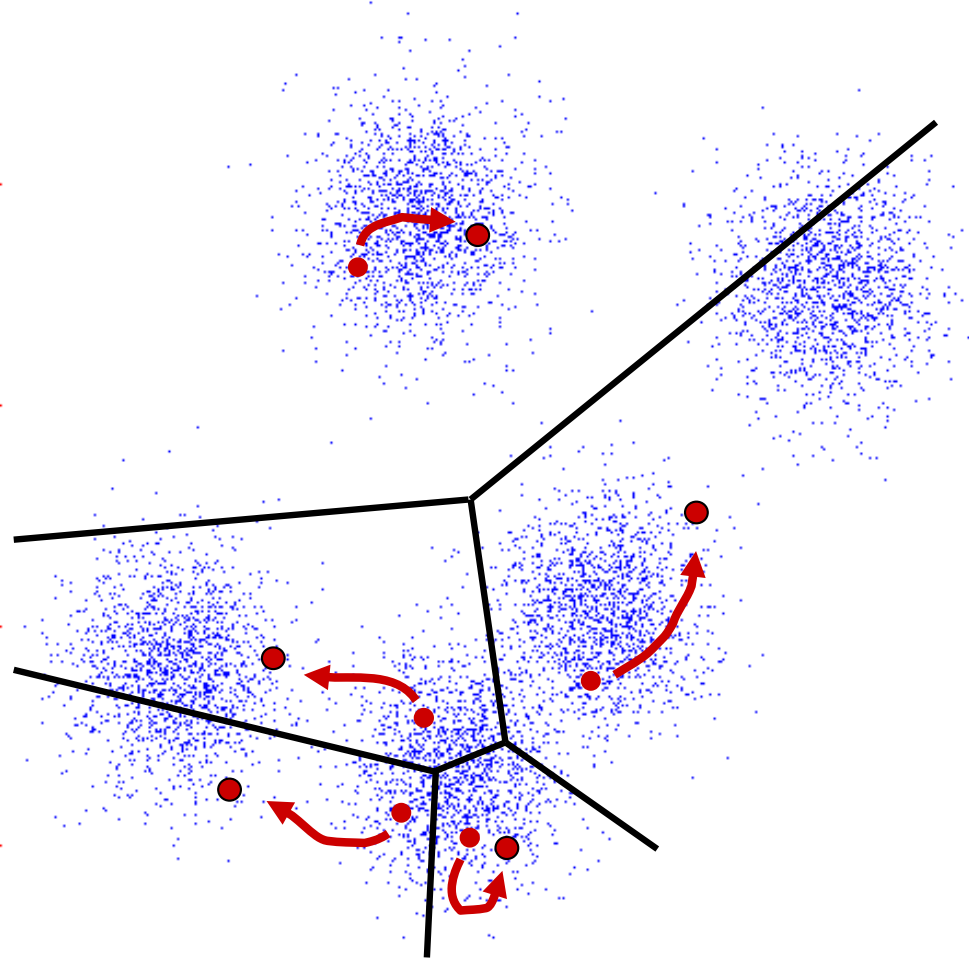
# Common Algorithms

- k-Means Clustering
- Collaborative Filtering
- Principle Component Analysis (PCA)
- Expectation Maximization (EM)

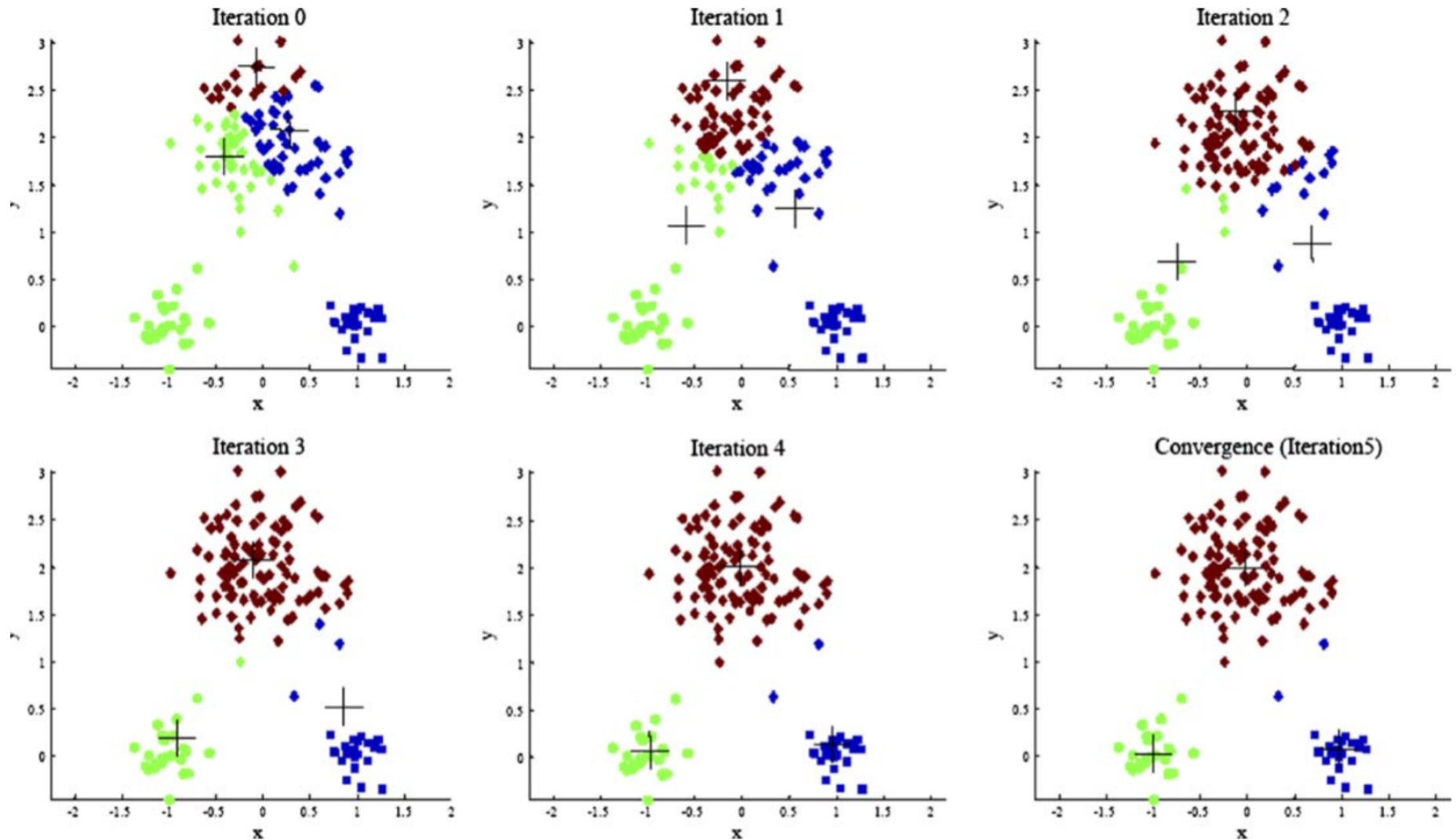


# $k$ -Means Clustering (1)

- Pick  $K$  random points as cluster centers (means)
- Alternate:
  - Assign data instances to closest mean
  - Assign each mean to the average of its assigned points
- Stop when no points' assignments change



# $k$ -Means Clustering (2)



# Checkup

1. Build an Atari system that learns game-winning techniques via actually playing and adjusting actions based upon score changes
  2. Given a dataset of past credit-card transactions (known to be fraudulent or not), build a system to identify future fraud
  3. If we assume incoming CS1 students are bi-modal, but normally distributed, find the average grades of the two groups
1. RL
  2. Supervised
    - Classification
  3. Unsupervised
    - Parameter estimation



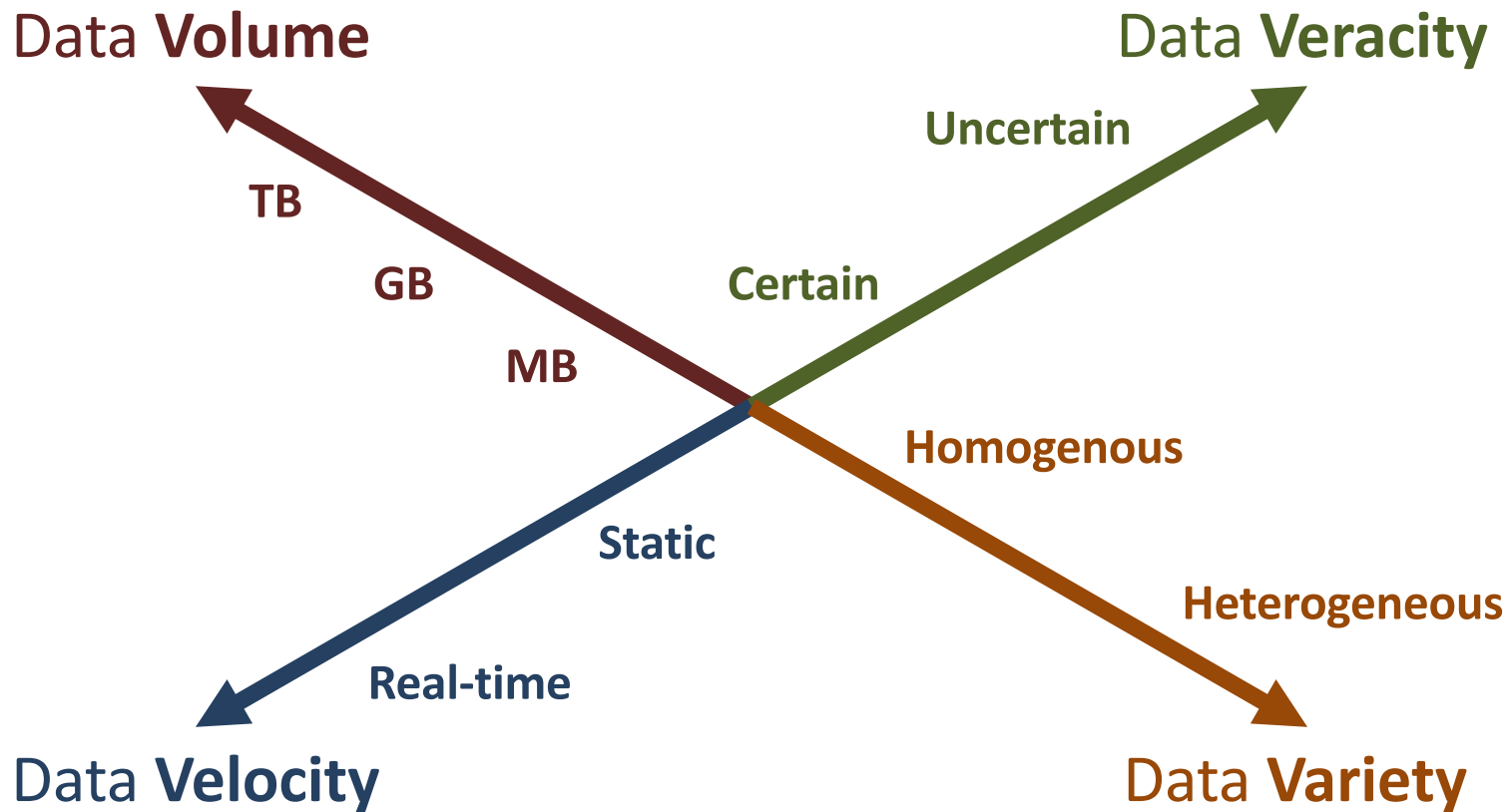


# Issues/Challenges

- Big Data
- Curse of Dimensionality
- No Free Lunch



# Big Data – The Four V's



**Parametric** algorithm: model does not grow with data size



# The Curse of Dimensionality

“Various phenomena that arise when analyzing and organizing data in high-dimensional spaces (often with hundreds or thousands of dimensions) that do not occur in low-dimensional settings such as the three-dimensional physical space of everyday experience.” – Wikipedia

- Memory requirement increases
- Required sampling increases
- Distance functions become less useful

...



# No Free Lunch

- There is no universally best model – a set of assumptions that works well in one domain may work poorly in another
- We need many different models, and algorithms that have different speed-accuracy-complexity tradeoffs

