One Day Workshop

Interactive Lecture & Hands on Session

Introduction to Data Science and Machine Learning

Jobin Wilson

For Students and Faculty Members of NIT Calicut

Agenda

- Introduction to Data Science and Machine Learning
- Types of Learning Algorithms
- Supervised Learning Algorithms
- Unsupervised Learning Algorithms
- Neural Networks and Deep Learning
- Large Scale Machine Learning in Practice

What is Data Science?

- Finding **answers** to **questions** that one cares about using **systems**, **processes** and **scientific methods** on data.

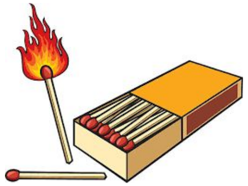
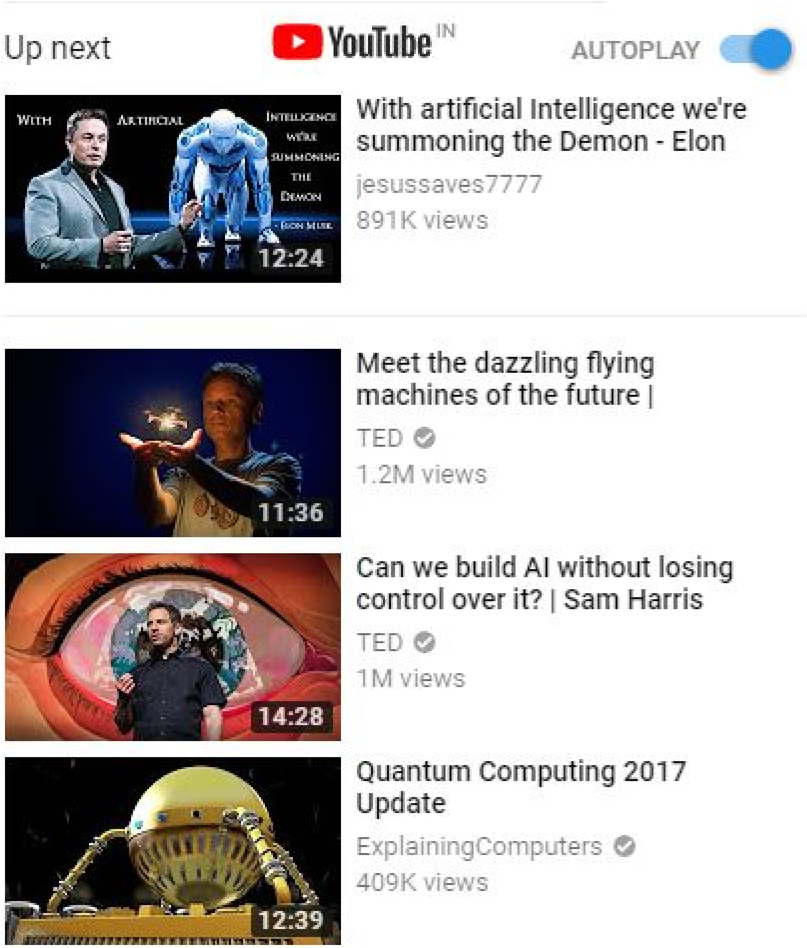


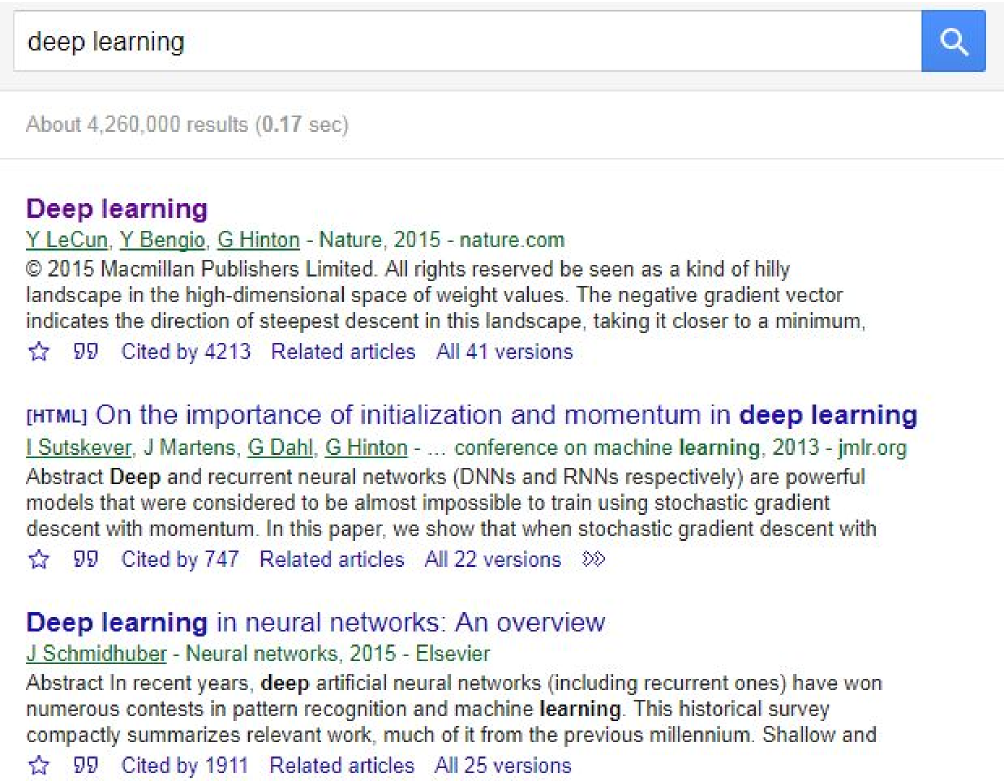
Image adapted from CS 109 (Hanspeter Pfister & Joe Blitzstein)

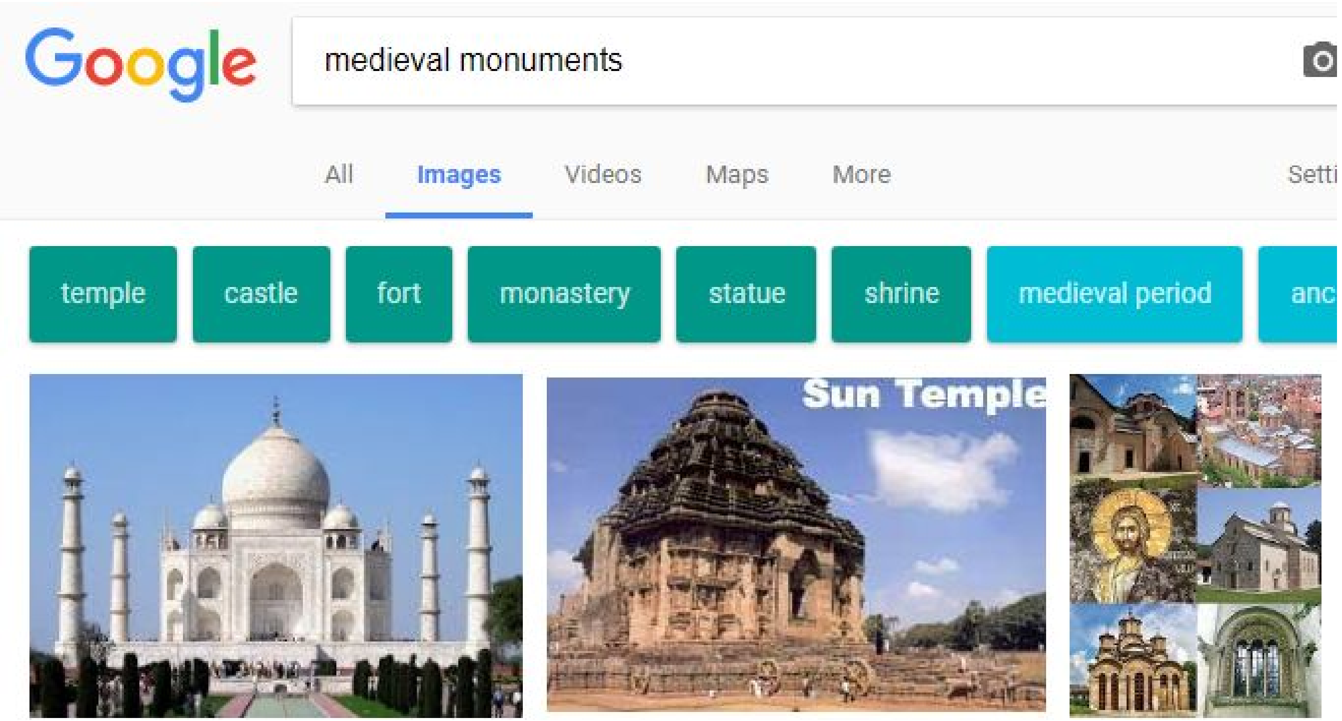
Recommender Systems



Recommending Content – "Wisdom of Crowds"

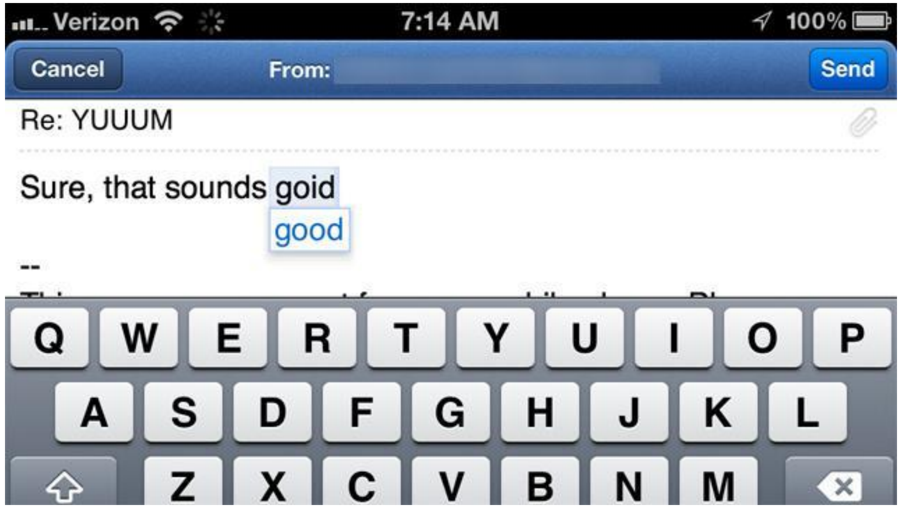
Information Retrieval

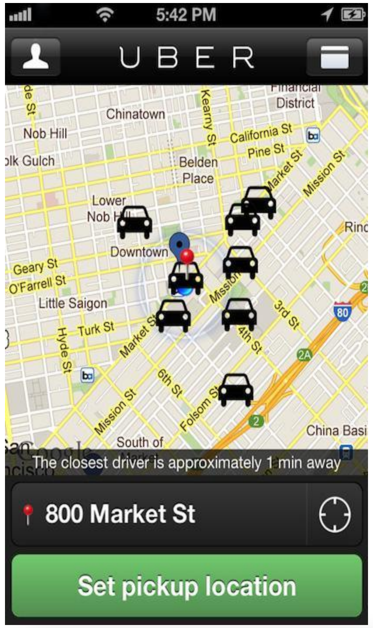


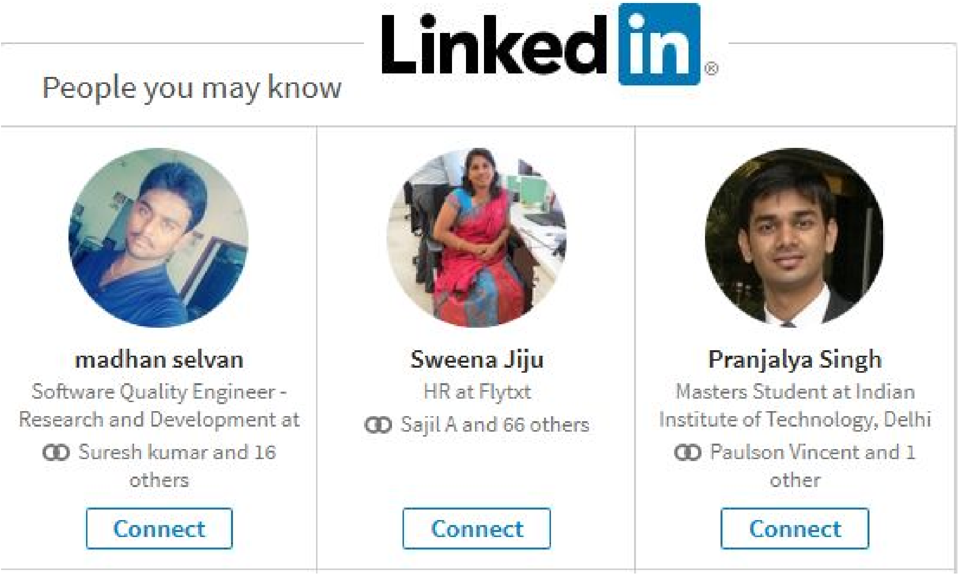


Search – "Cut through the Clutter"

Other Applications







Data Science in Media

SCIENTIFIC AMERICAN™

Sign in | Register 0

Search ScientificAmerican.com

Subscribe | News & Features | Topics | Blogs | Videos & Podcasts | Education

Mind & Brain » Mind Matters 4 :: Email

How a Computer Program Helped Reveal J. K. Rowling as Author of *Cuckoo's Calling*

Author of the *Harry Potter* books has a distinct linguistic signature

By Patrick Juola | August 20, 2013

"The man who wrote the note is a German. Do you note the peculiar construction of this sentence?" These were the words of Sherlock Holmes in "A Scandal in Bohemia," analyzing a note from a client,

This is the Brahmastra Rahul Gandhi will hurl at PM Modi

ET Online | Oct 09, 2017, 05.04 PM IST

162 Comments A+

Has the Congress found the Brahmastra for the next general elections? Perhaps it has, if one goes by reports that Congress is in touch with Big Data firm Cambridge Analytica that helped US President Donald Trump win last year.

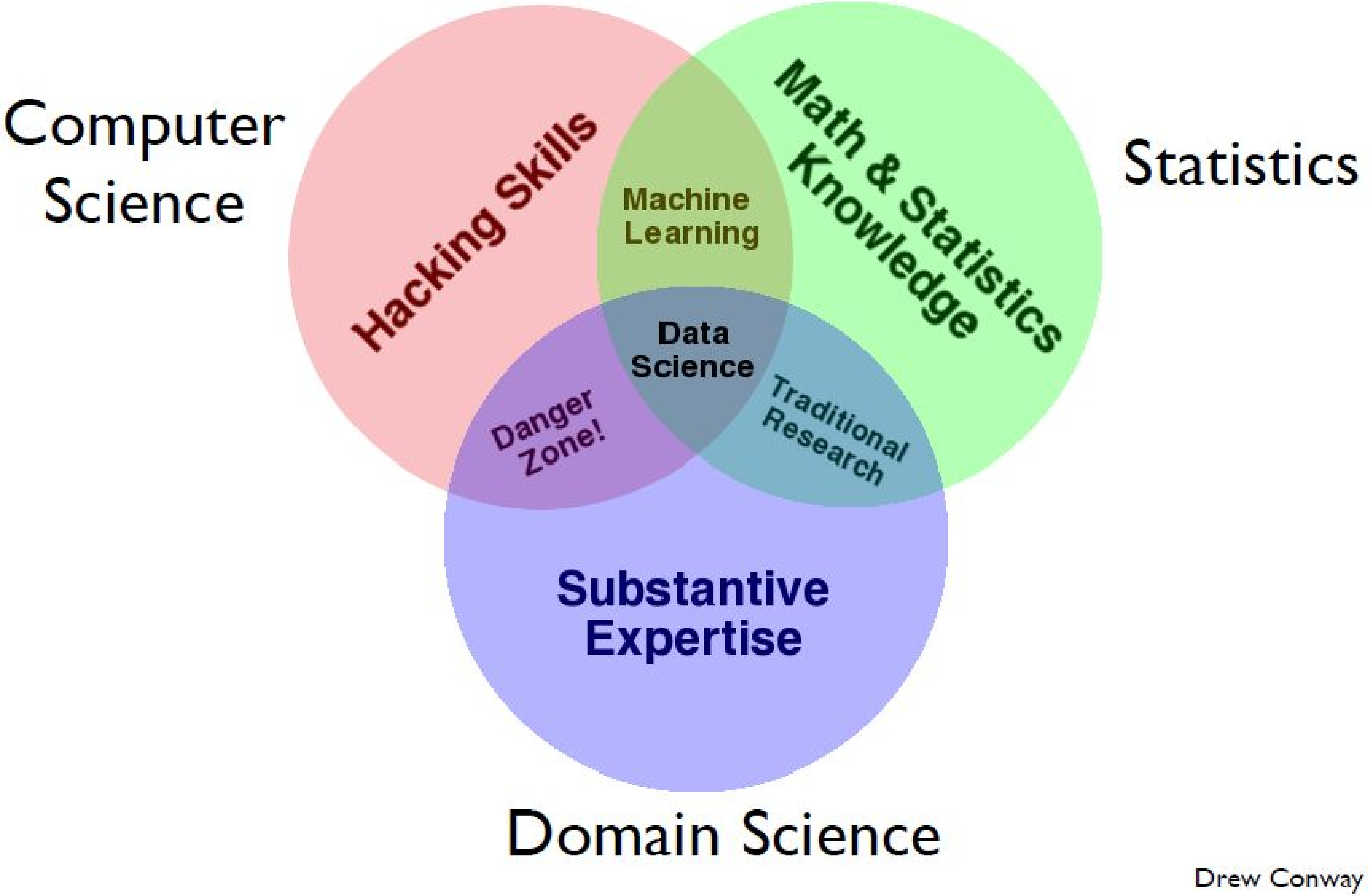
Live results | President | Senate | House | Governor | Choose your

Numbers nerd Nate Silver's forecasts prove all right on election night

FiveThirtyEight blogger predicted the outcome in all 50 states, assuming Barack Obama's Florida victory is confirmed

Luke Harding
guardian.co.uk, Wednesday 7 November 2012 10.45 EST

Data Science



Who is a Data Scientist?

- “A data scientist is someone who knows more statistics than a computer scientist and more computer science than a statistician.” - Joshua Blumenstock
- “Data Scientist = statistician + programmer + coach + storyteller + artist” - Shlomo Aragmon
- “A data scientist is that unique blend of skills that can both unlock the insights of data and tell a fantastic story via the data” — DJ Patil

Data Science - A hybrid Discipline !

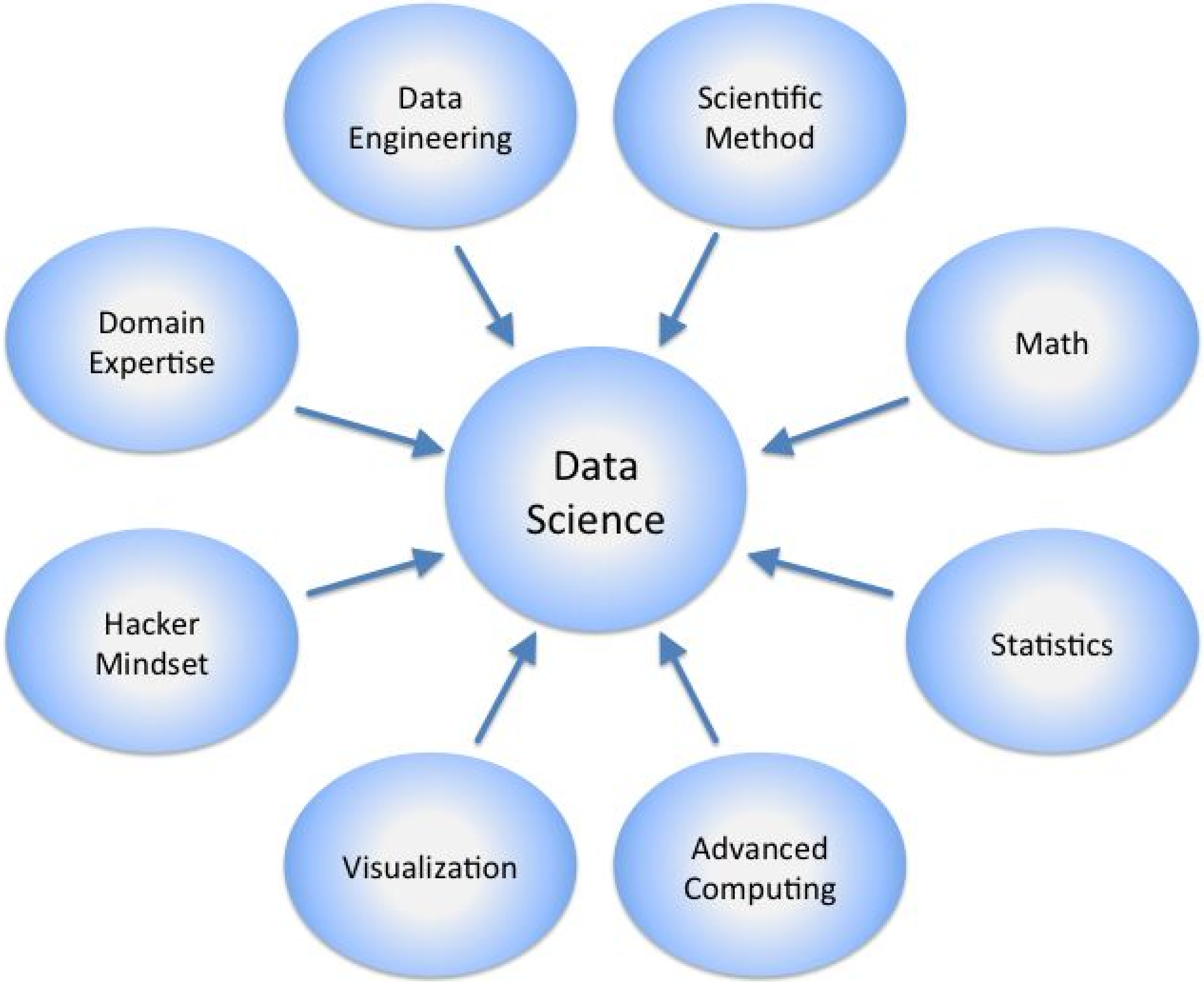
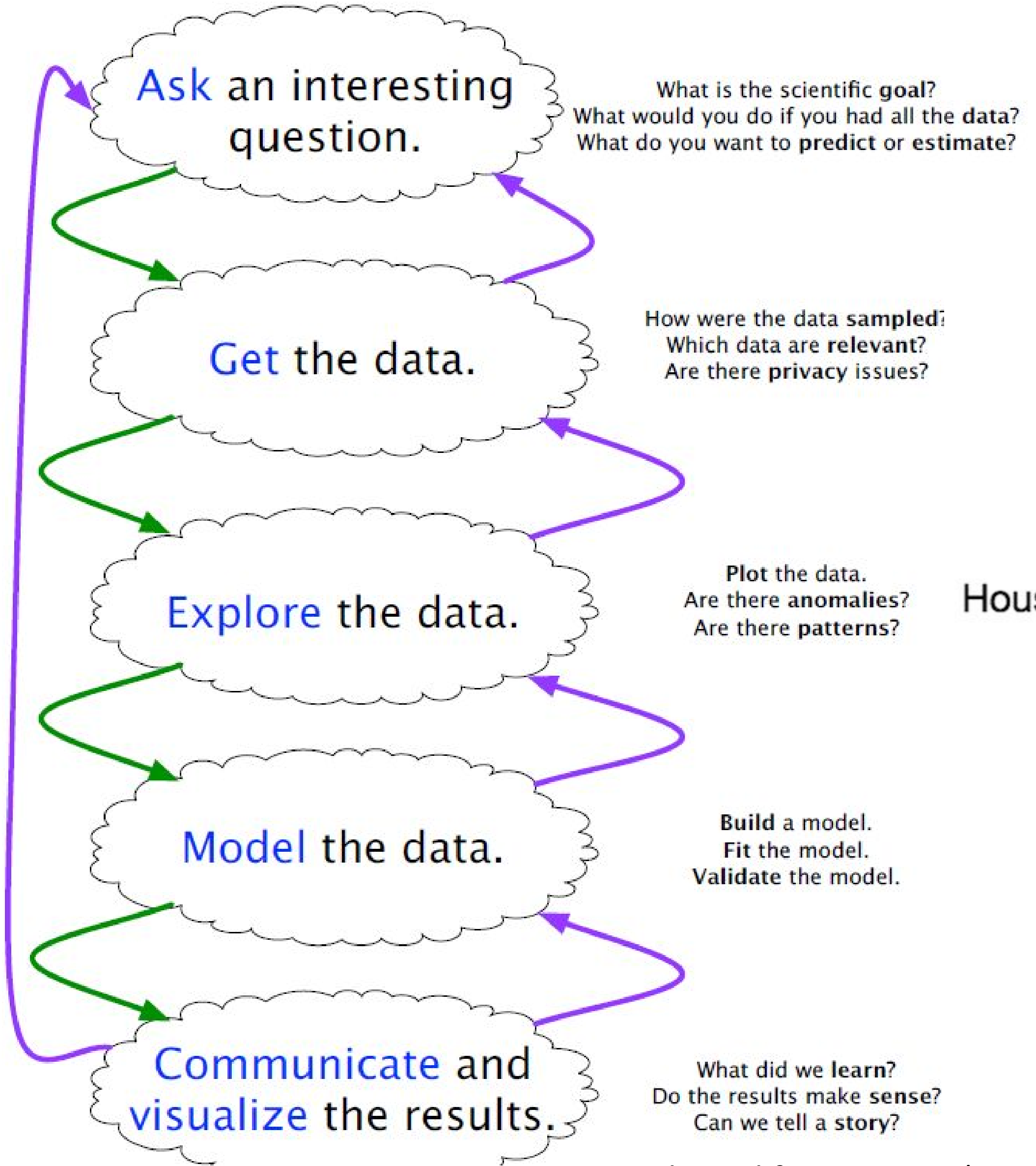


Image Source: http://en.wikibooks.org/

A Data Science Pipeline



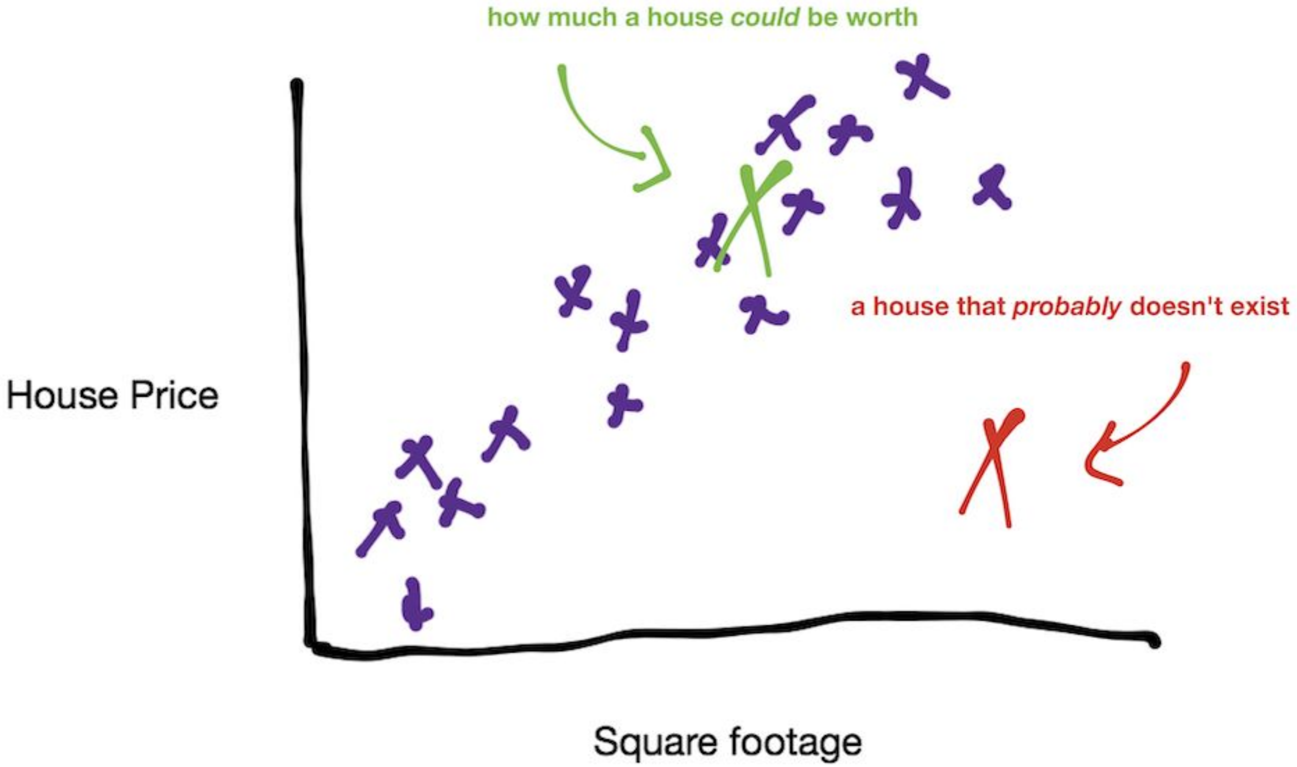
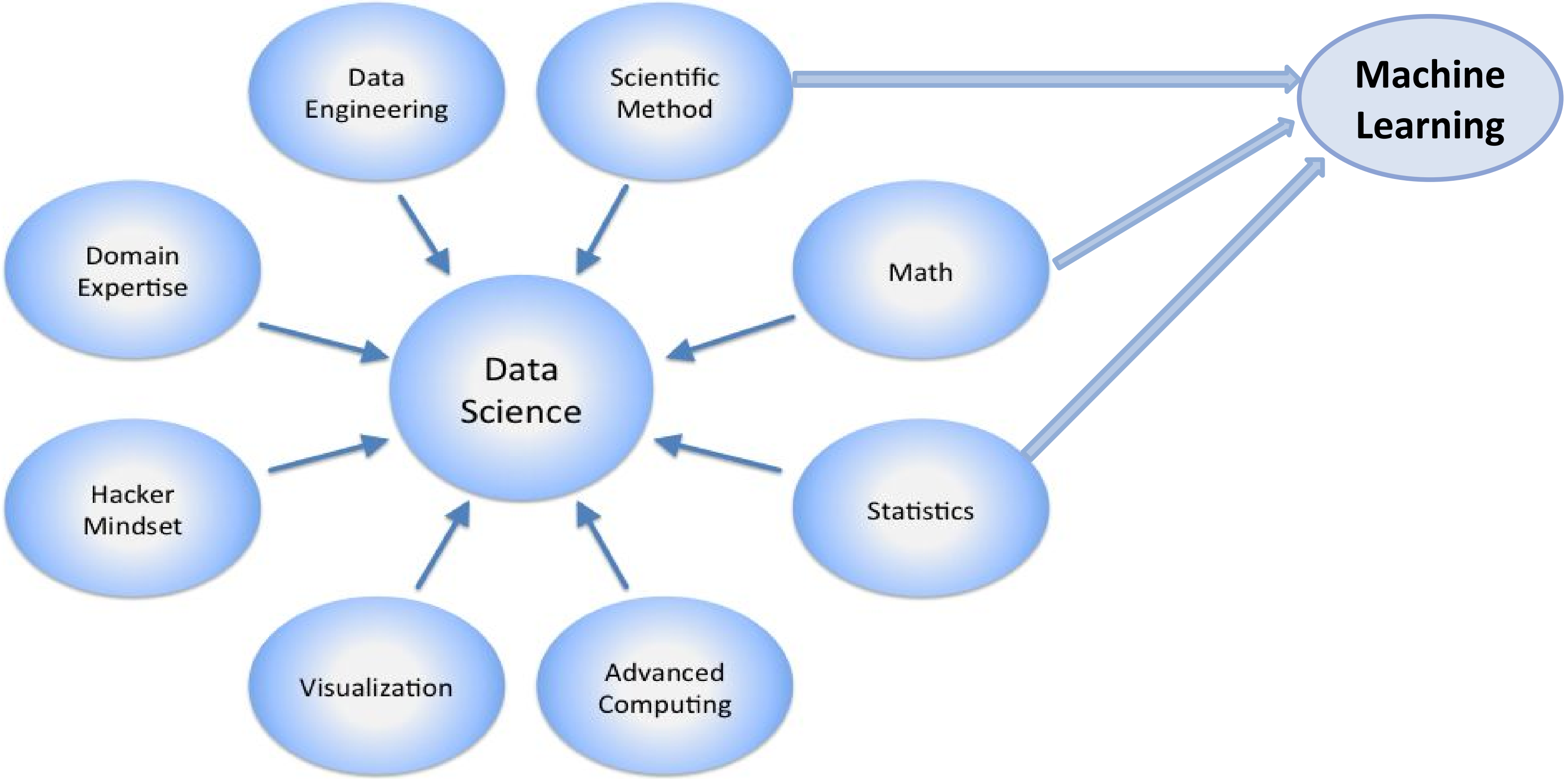


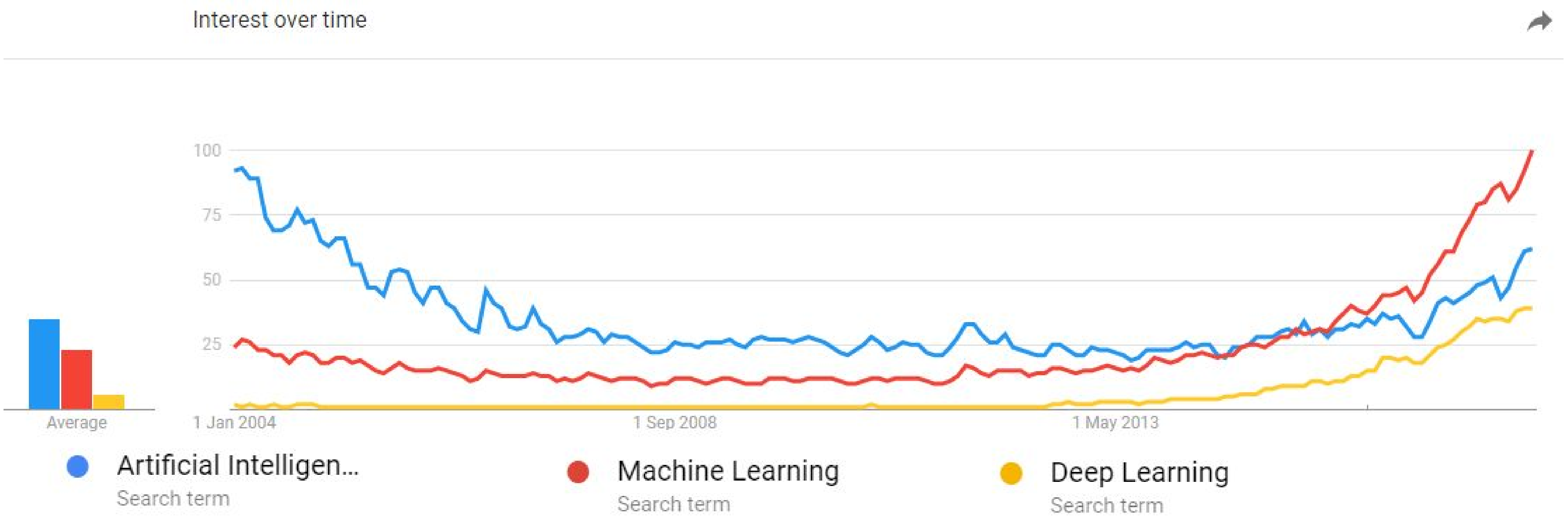
Image adapted from CS 109 (Hanspeter Pfister & Joe Blitzstein), and https://goo.gl/hqmLmq

Data Science vs. Machine Learning

Machine Learning deals with developing algorithms to automatically learn from the data and make future predictions.



AI, Machine Learning, Deep Learning over the years!



Source: Google Trends

What is Involved in Intelligence?

- Ability to interact with the world – vision, speech, motion, manipulation etc.
- Ability to model the world and to reason about it
- Ability to learn and adapt continuously

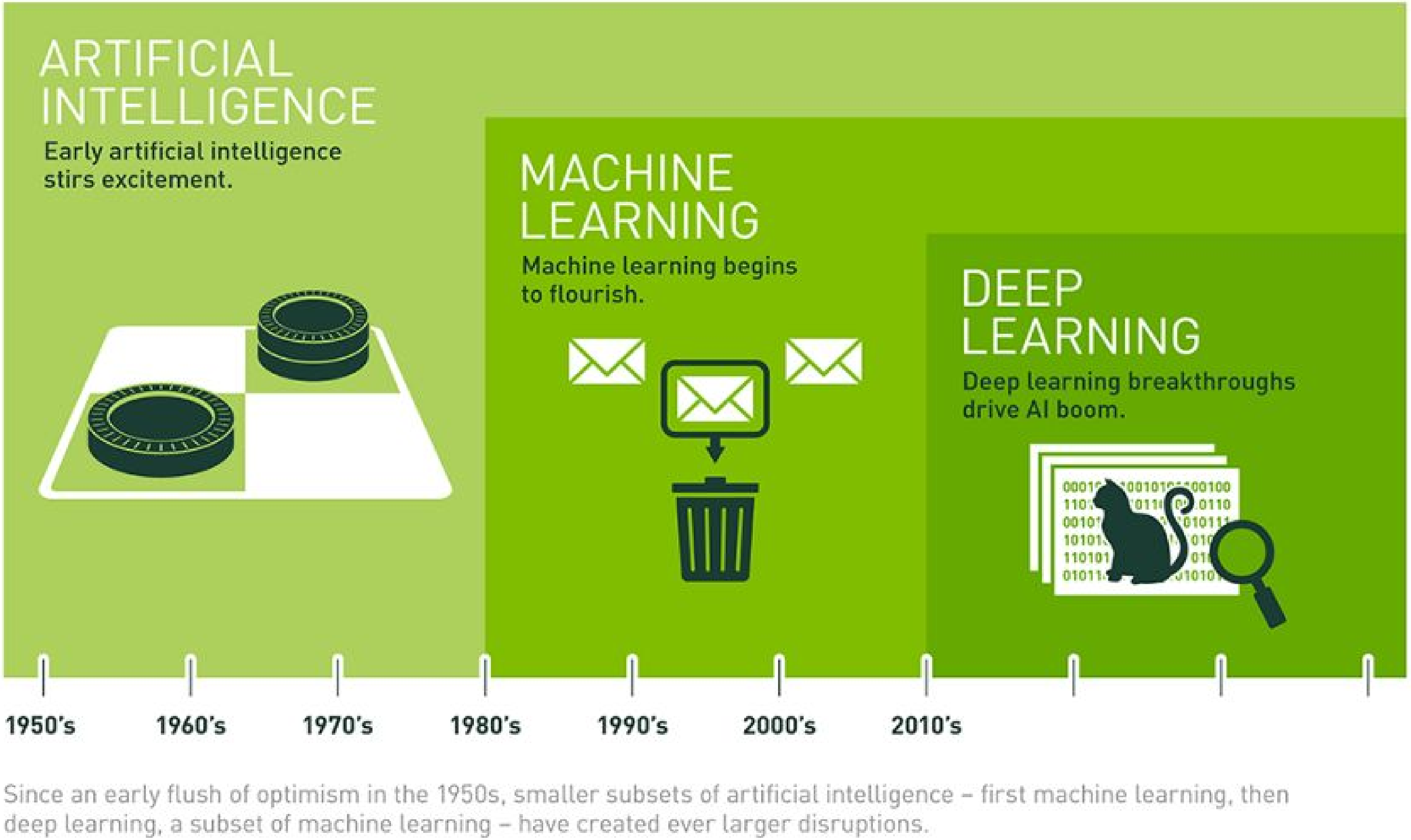
Why AI is a hard problem?








AI, Machine Learning & Deep Learning



Source : https://goo.gl/W3YUvy

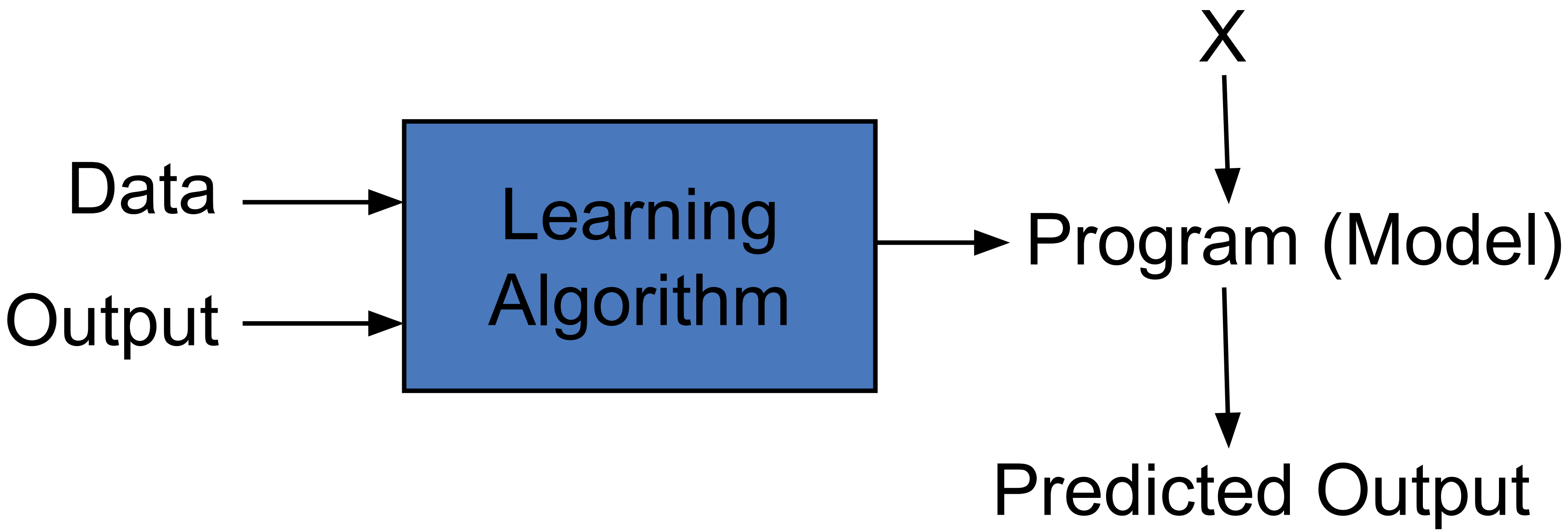
Artificial Intelligence

- Machine that mimics a "cognitive" function of human mind
- Genesis : Dartmouth workshop of 1956
- The Big Dream – “General AI” (strong AI) [The movie stuff ☺]
- The reality – “Narrow AI” (weak AI) [e.g. Apple Siri, Self driving cars etc.]
- The Giants
  - Allen Newell (CMU), Herbert Simon (CMU), John McCarthy (MIT), Marvin Minsky (MIT) , Arthur Samuel (IBM)

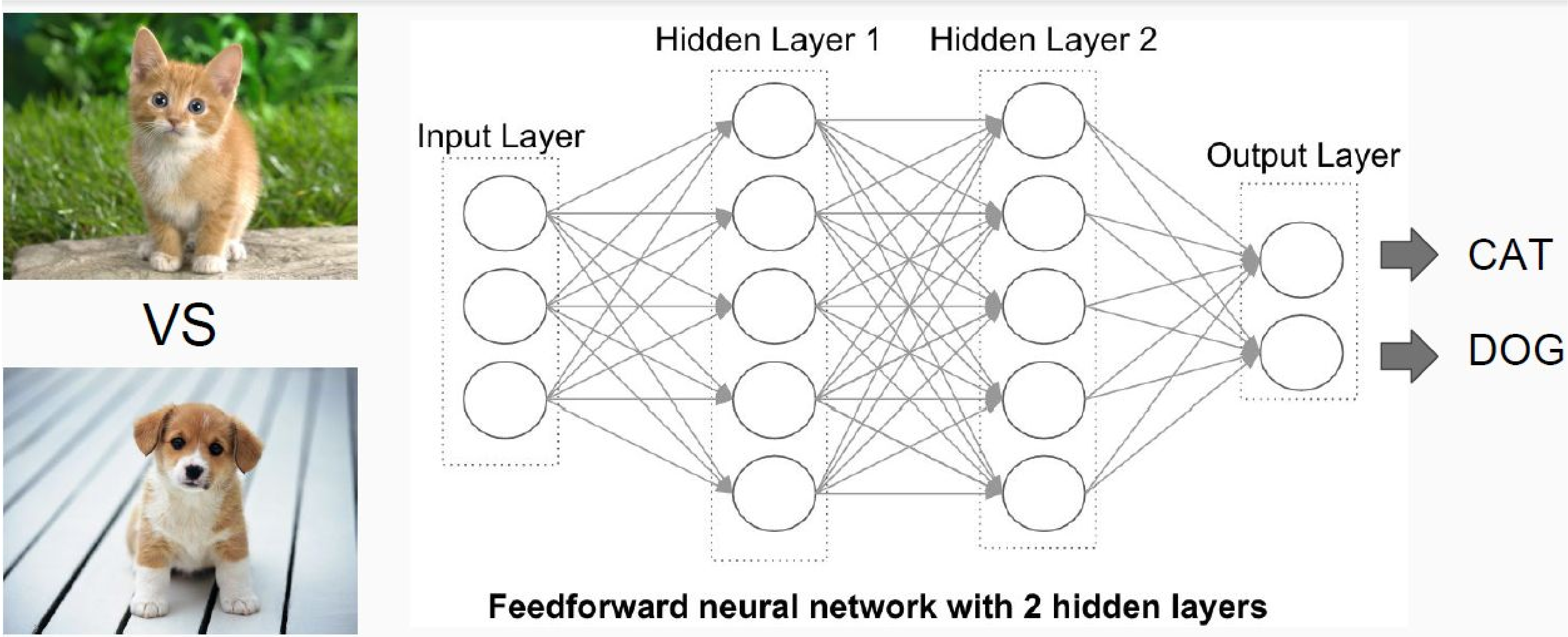
Machine Learning

- Using algorithms to analyze data, learn from it, and then make a prediction about some phenomenon of interest

- Ability to learn without being explicitly programmed



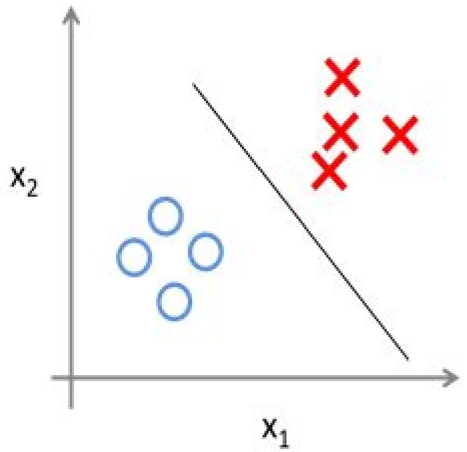
Deep Learning



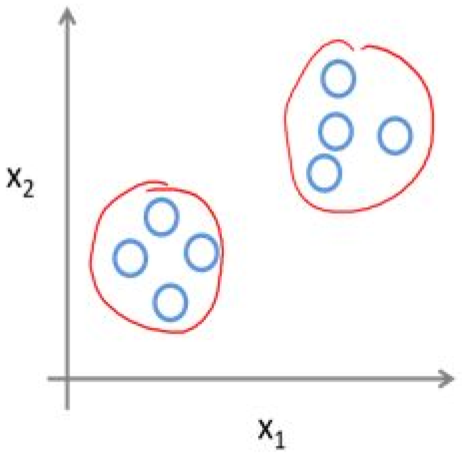
Source: Antonio Spadaro's slides from PyCon Italia 2017

Types of Learning

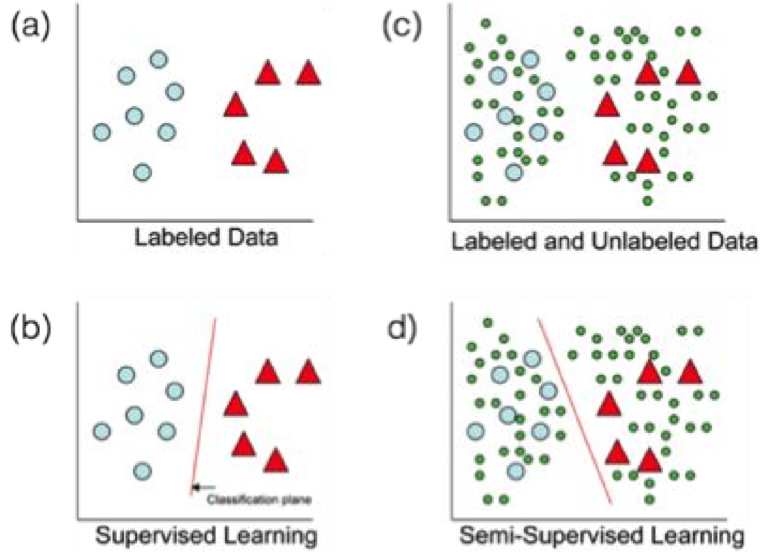
Supervised Learning



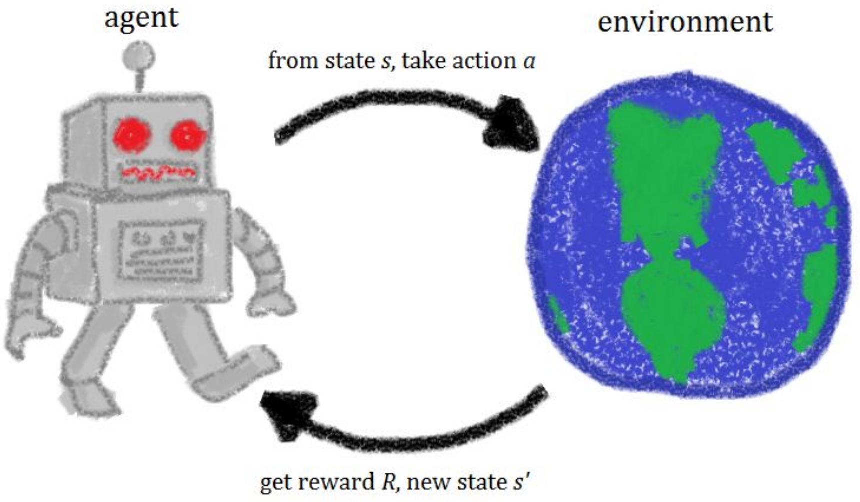
Unsupervised Learning



Semi Supervised Learning



Reinforcement Learning

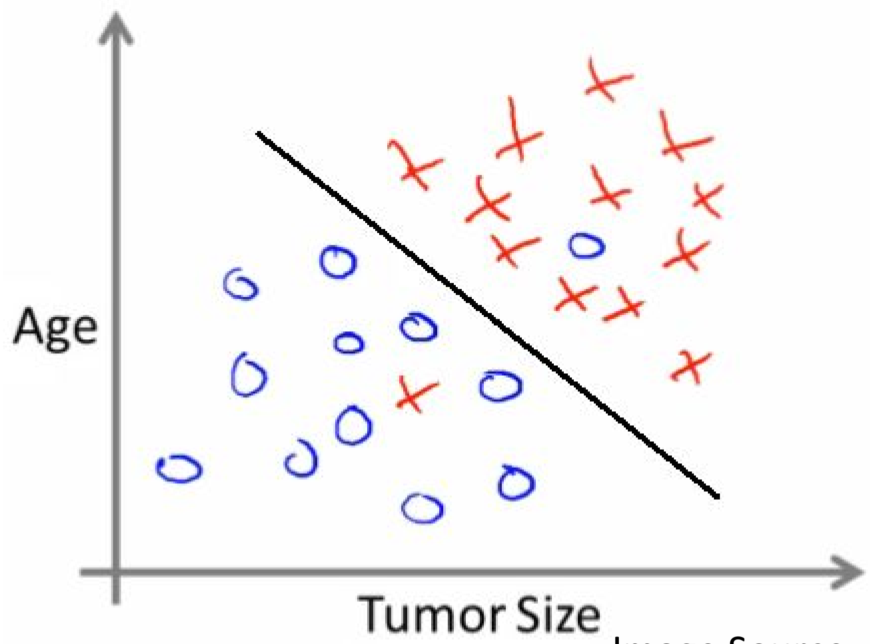


Basic Terminology

- **Instances:** Items for learning or prediction (e.g., emails, images etc.)
- **Features :** Attributes (typically numeric) to represent observations (e.g. keywords, pixel intensities etc.)
- **Labels:** class assigned to observations (e.g., spam/ham, malignant/benign etc.)
- **Training and Test Data**
  - Training data – used to training a model
  - Test data – used to evaluate the model

Supervised Learning

- **Given** several examples of a function *(X, F(X))*
- **Predict** value of *F(X)* for new examples *X*
  - Classification if F(X) is discrete
  - Regression if *F(X)* is continuous



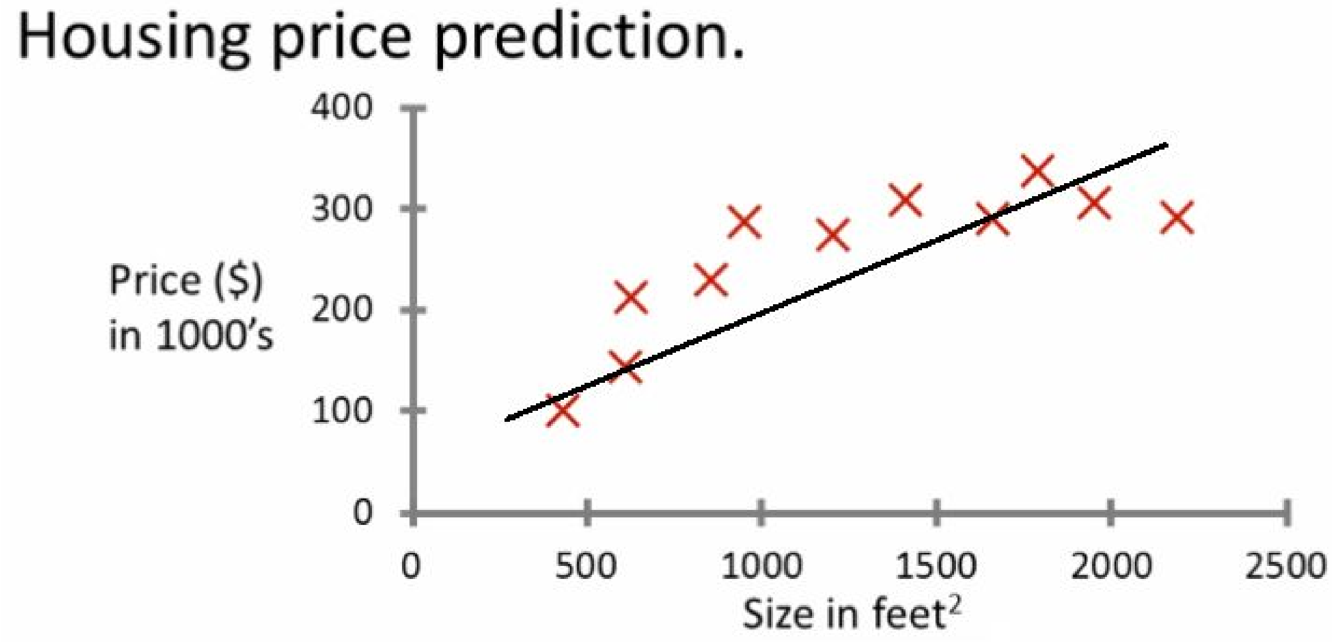
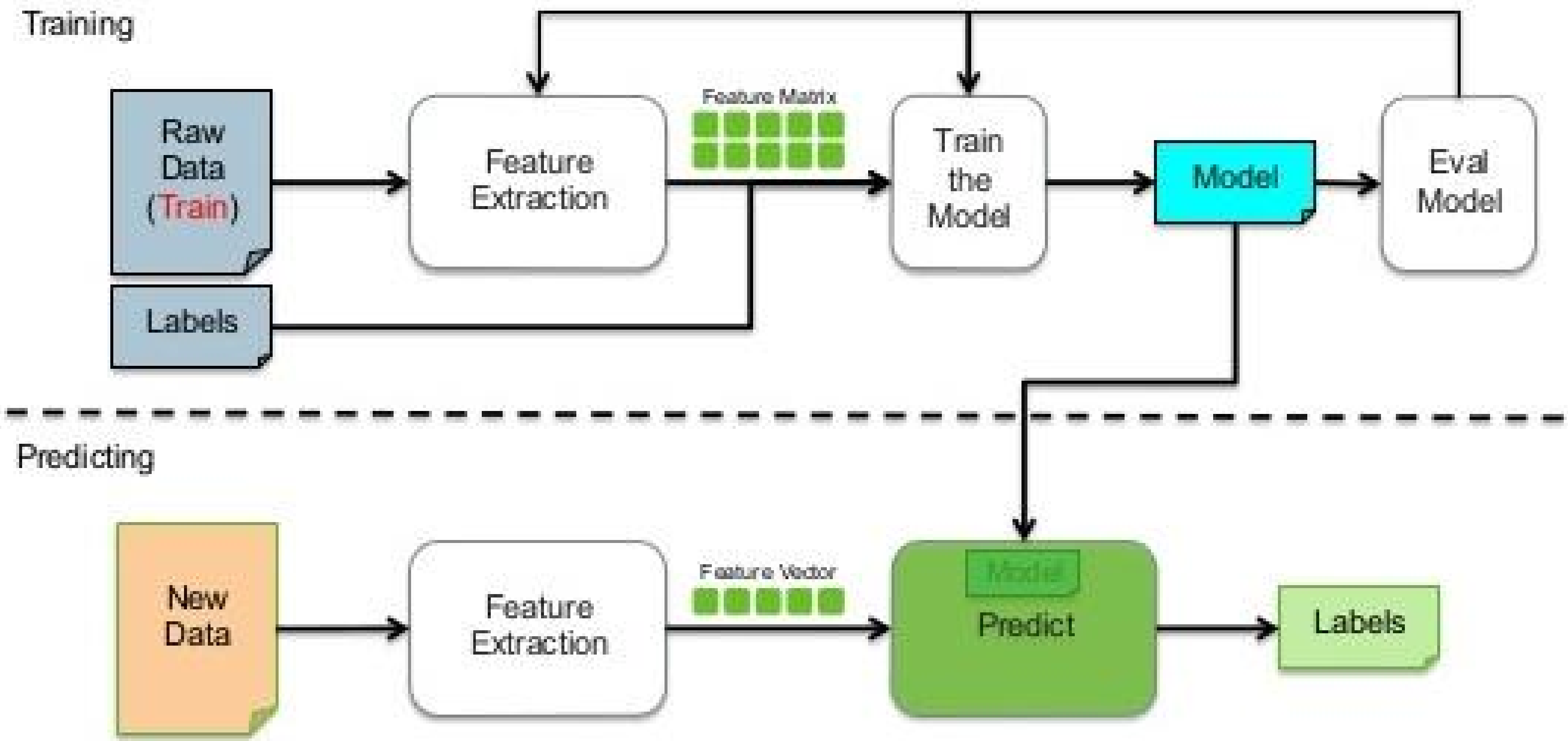


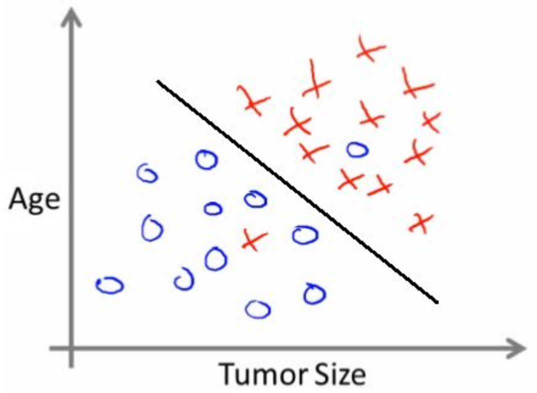
Image Source :https://www.coursera.org/learn/machine-learning

Supervised Learning Workflow

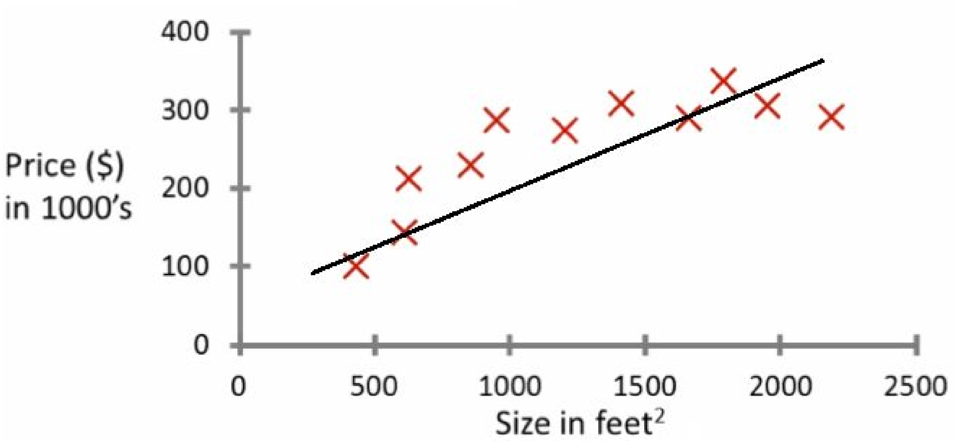


Supervised Learning Algorithms

- ***Classification***. Prediction is a discrete label; e.g. spam classification
  - Naïve Bayes
  - K-Nearest Neighbor
  - Logistic Regression
  - Decision Tree
  - Support Vector Machine



- ***Regression***. Prediction is a real value; e.g. stock prices, housing prices
  - Linear Regression
  - K-Nearest Neighbor
  - Decision Tree Regression



Naïve Bayes Classification

- Bayes' Theorem

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}.$$

- P(A) - initial degree of belief in A (Prior)
- P(B|A) - degree of belief in B, given A is observed (likelihood)
- P(B) - evidence
- P(A|B) - degree of belief in A, given B is observed (posterior)

Worked Out Example

- Assuming the following training data, is “free discount” SPAM?

Message	Class
free movie tickets	SPAM
going for movie	HAM
free watch offer	SPAM
i am free	HAM
rolex watch discount	SPAM

P(SPAM) =3/5 = 0.6

P(HAM) = 1.0 – P(SPAM) = 0.4

$$P(SPAM \mid WORD) = \frac{P(WORD \mid SPAM).\mathrm{P(SPAM)}}{P(WORD \mid SPAM).\mathrm{P(SPAM)} + P(WORD \mid HAM).\mathrm{P(HAM)}}$$

Worked Out Example..contd.

Keyword	Spam(S)	Ham (H)
tickets	1	0
for	0	1
offer	1	0
i	0	1
movie	1	1
am	0	1
watch	2	0
rolex	1	0
free	2	1
discount	1	0
going	0	1

P(free | S) = 2/9 = 0.22 and P(free | H) = 1/6 = 0.16
P(discount | S) = 1/9 = 0.11 and P(discount | H) = 0/6 = 0
Vocabulary size = 11

After Laplace Smoothing

P(free | S) = (2+1) / (9+11) = 0.15 and P(free | H) = (1+1) / (6+11) = 0.12
P(discount | S) = (1+1) / (9+11) = 0.10 and P(discount | H) = (0 + 1)/(6 +11) = 0.06

P(S | "free discount") = P(free|S). P(discount|S).P(S) / (P(free|S). P(discount|S).P(S) + P(free|H). P(discount|H).P(H))

P(S| "free discount") = 0.15*0.10*0.6 / (0.15*0.10*0.6 + 0.12*0.06*0.4)
= 0.009 / (0.009 + 0.00288) = **0.7575**

P(H| "free discount") = 0.12*0.06*0.4/ (0.12*.06*0.4+ 0.15*0.10*0.6) = **0.2424**

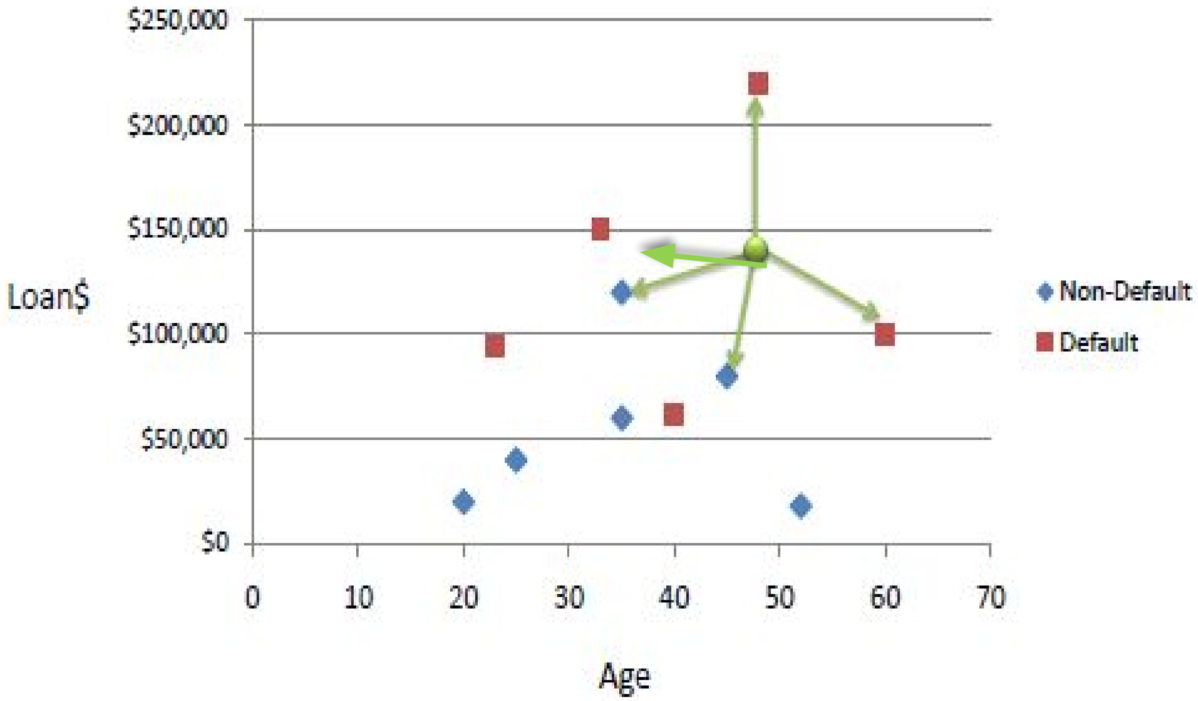
Instance Based Learning : K Nearest Neighbors

- **Task :** Given the following data, predict if a person (48, $140,000) is likely to Default?

Age	Loan Amount	Class
20	$24,000	Non-Default
23	$95,000	Default
25	$40,000	Non-Default
35	$60,000	Non-Default
32	$150,000	Default
35	$120,000	Non-Default
40	$65,000	Default
45	$72,000	Non-Default
53	$20,000	Non-Default
48	$223,000	Default
60	$100,000	Default
48	$140,000	?

KNN Classifier

- You are what you resemble!



$$distance = \sqrt{(age1 - age2)^2 + (loan1 - loan2)^2}$$

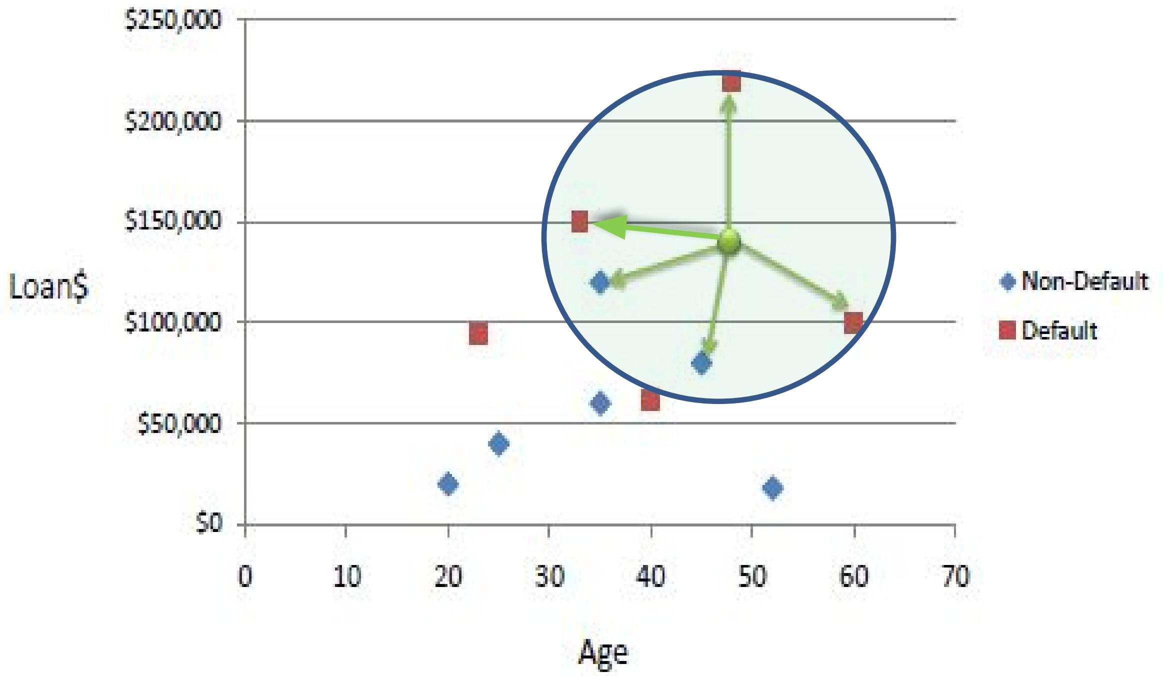
$e.g. \sqrt{(32 - 48)^2 + (150000 - 140000)^2}$ = 10000.01

age	loan_amt	label	distance
48	140000	?	0.00
32	150000	Default	10000.01
35	120000	Non-Default	20000.00
60	100000	Default	40000.00
23	95000	Default	45000.01
45	72000	Non-Default	68000.00
40	65000	Default	75000.00
35	60000	Non-Default	80000.00
48	223000	Default	83000.00
25	40000	Non-Default	100000.00
20	24000	Non-Default	116000.00
53	20000	Non-Default	120000.00

KNN Prediction

- Find the 5 nearest neighbors and take a majority vote
- Could also "weigh" the votes
- Take care of feature normalization

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$



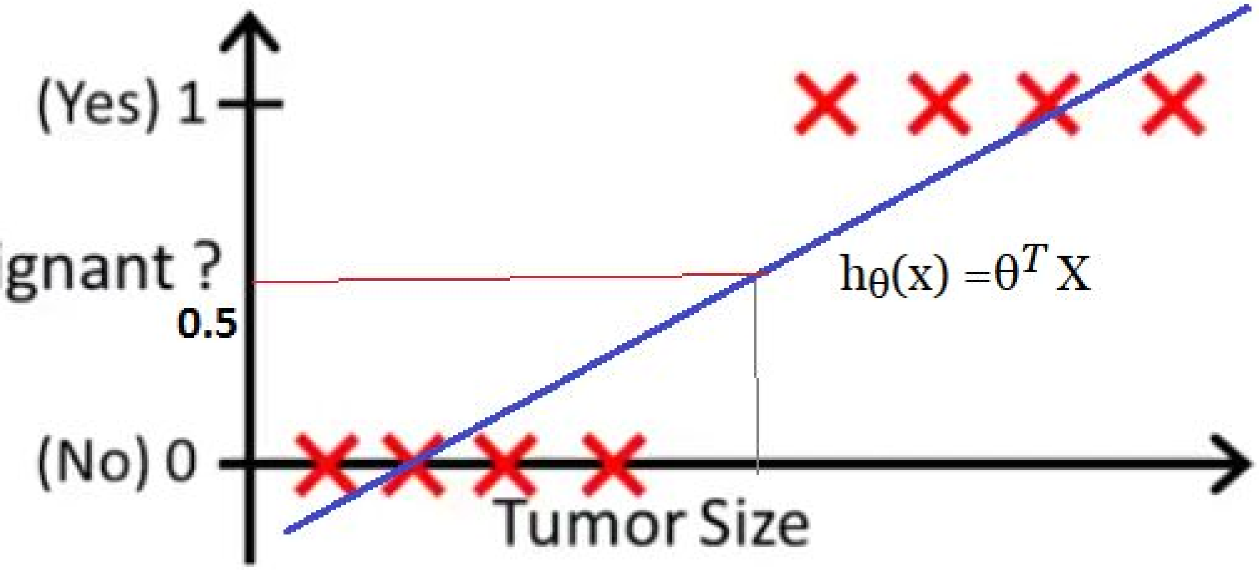
Age	Loan Amount	Class
20	$24,000	Non-Default
23	$95,000	Default
25	$40,000	Non-Default
35	$60,000	Non-Default
32	$150,000	Default
35	$120,000	Non-Default
40	$65,000	Default
45	$72,000	Non-Default
53	$20,000	Non-Default
48	$223,000	Default
60	$100,000	Default
48	$140,000	?

Default

Logistic Regression – Intuition

- Given features X ,estimate y; where $y \in \{0,1\}$
- Option 1:

Assume a line as $h_\theta(x) = \theta^T X$



Threshold classifier output $h_\theta(x)$ at 0.5:

If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

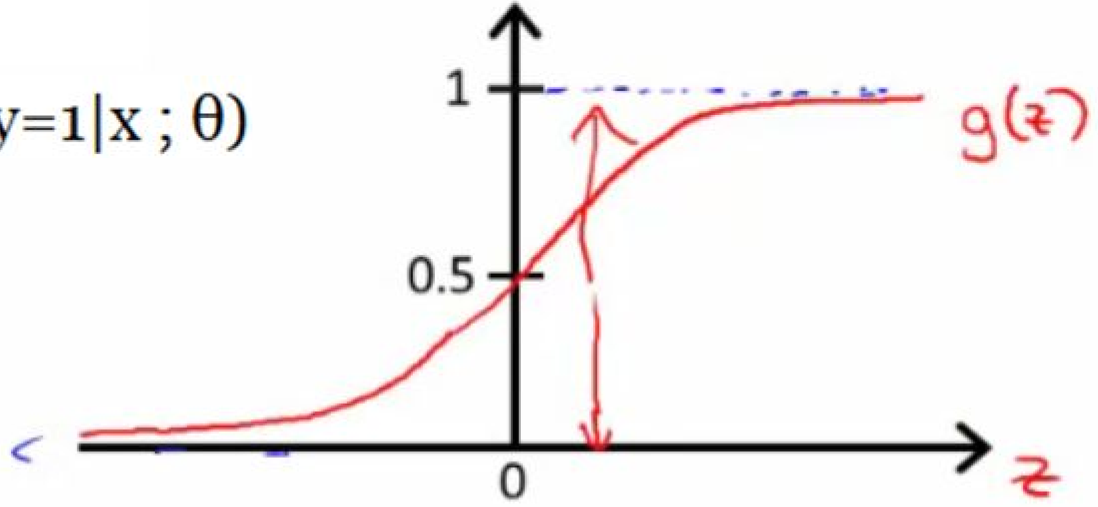
- But function not bounded between [0,1];
- Outliers could easily change decision boundary
- What to do?

Image Source :https://www.coursera.org/learn/machine-learning

Logistic Regression - Hypothesis

- Hypothesis : $h_\theta(x) = g(\theta^T x) = \dfrac{1}{1+e^{-\theta^T x}}$
- Logistic Function / Sigmoid $g(z) = \dfrac{1}{1+e^{-z}}$
- Probabilistic Interpretation : $h_\theta(x) = P(y=1|x;\theta)$

  $P(y=1|x;\theta) + P(y=0|x;\theta) = 1$
  $P(y=0|x;\theta) = 1 - P(y=1|x;\theta)$



- g(z) > 0.5 when z >= 0
- Decision boundary will be a line/hyperplane; when $z = (\theta^T x)$ > 0 then g(z) > 0.5; so predict y=1

Image Source :https://www.coursera.org/learn/machine-learning

Logistic Regression – Toy Example

- Hypothesis : $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$
- Decision boundary: y = 1 if -3 + x1 + x2 >=0
- Does this work? (2,1) => class 0, (4,2) => class 1
- Wait…how did we get the equation of the decision boundary?

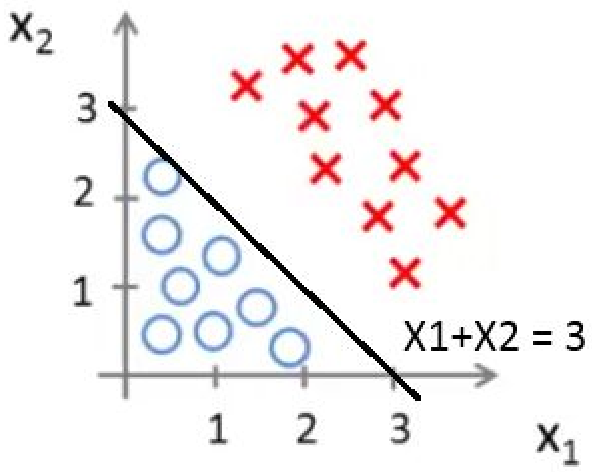


Image Source :https://www.coursera.org/learn/machine-learning

Optimizing a Cost Function

- Cost Function:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$\mathrm{Cost}(h_\theta(x), y) = -\log(h_\theta(x)) \qquad \text{if } y = 1$$

$$\mathrm{Cost}(h_\theta(x), y) = -\log(1 - h_\theta(x)) \qquad \text{if } y = 0$$

- Compact Form :

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(h_\theta(x^{(i)})) + (1 - y^{(i)})\log(1 - h_\theta(x^{(i)}))]$$

- Minimize the cost using gradient descend

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

- Update rule :

$$Repeat \ \{ \ \ \theta_j := \theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \ \ \}$$

(simultaneously update)

Adapted from :https://www.coursera.org/learn/machine-learning

Logistic Regression Algorithm

- Given: training examples $(x_i, y_i),\ i = 1 \dots m$

  let $\theta = (0,0 \dots 0) \in R^n$

  repeat until convergence {

  $$d = (0,0 \dots 0) \in R^n$$

  for $i = 1 \dots m$ {

  $$y_i' = \frac{1}{1+e^{-\theta^T x_i}}$$

  $$error = y_i - y_i'$$

  $$d = \mathrm{d} + \mathrm{error} * \mathrm{x_i}$$

  }

  $$\theta = \theta - \alpha * d$$

Classifier Evaluation

- Precision, Recall and F-Measure : widely used

  (Task: predicting tumor)

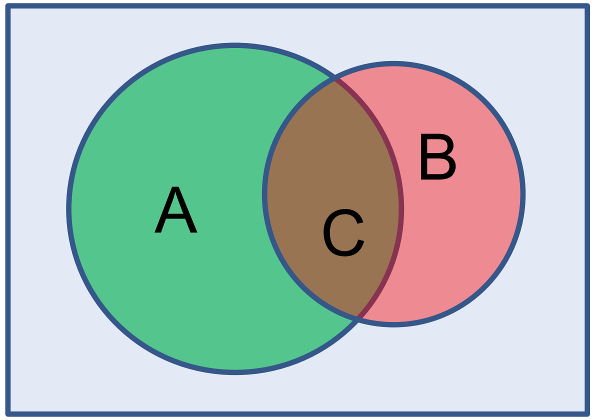
  A = Set of Predicted Tumor Patients

  B = Set of Actual Tumor Patients

  C = Set of Tumor Patients correctly Predicted

  Precision = **|C|/|A|**

  Recall = **|C|/|B|**



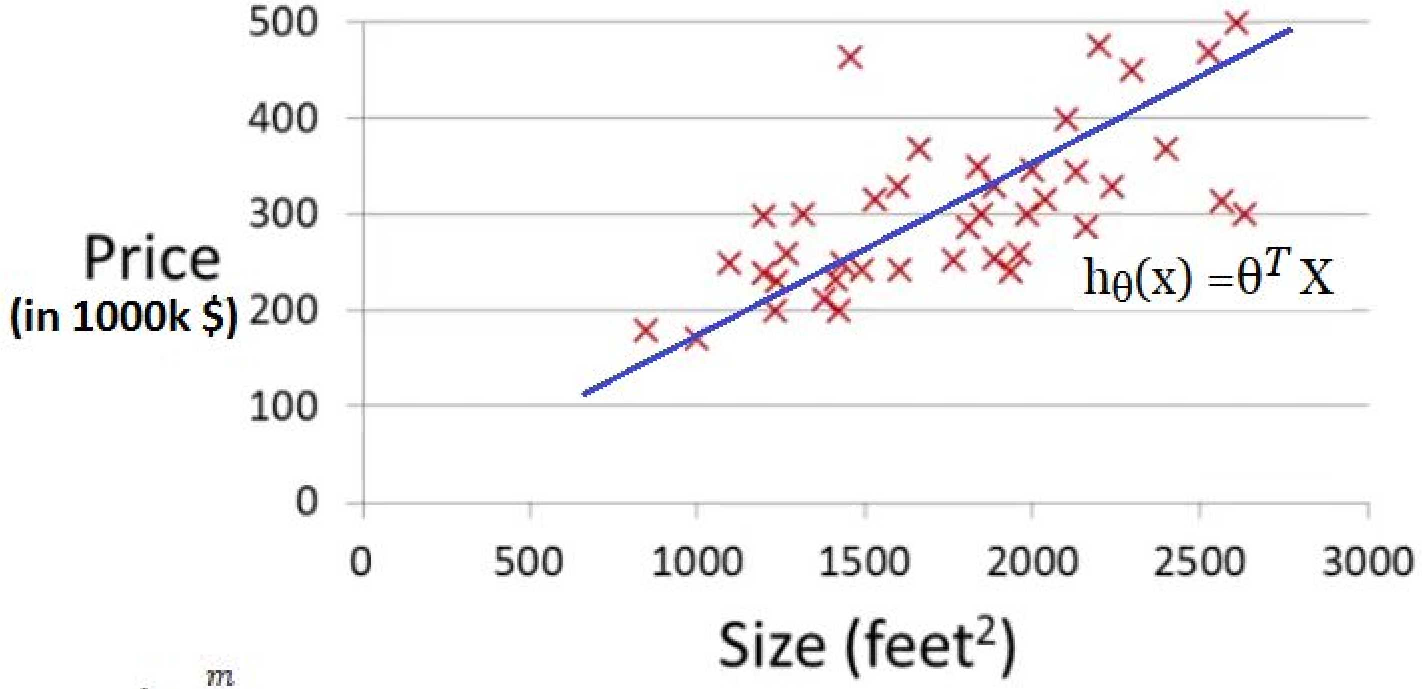
- F-Measure (or F1 Score) - harmonic mean of Precision and Recall

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

- A high F-Measure is a good indicator of model quality

Linear Regression – Intuition

- Given features X ,estimate y; where $y \in R$
- Assume a line as $h_\theta(x) = \theta^T X$
  $h_\theta(x) = \theta_0 + \theta_1 * size$
- Choose θ_0 and θ_1 such that
  $h_\theta(x_i)$ is as close to y_i



Cost function to minimize: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x_i) - y_i\right)^2$

Image Source :https://www.coursera.org/learn/machine-learning

Optimization - Intuition

Have some function $J(\theta_0, \theta_1)$

Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Outline:

- Start with some θ_0, θ_1
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$

  until we hopefully end up at a minimum

Source :https://www.coursera.org/learn/machine-learning

Gradient Descent - Intuition

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

}

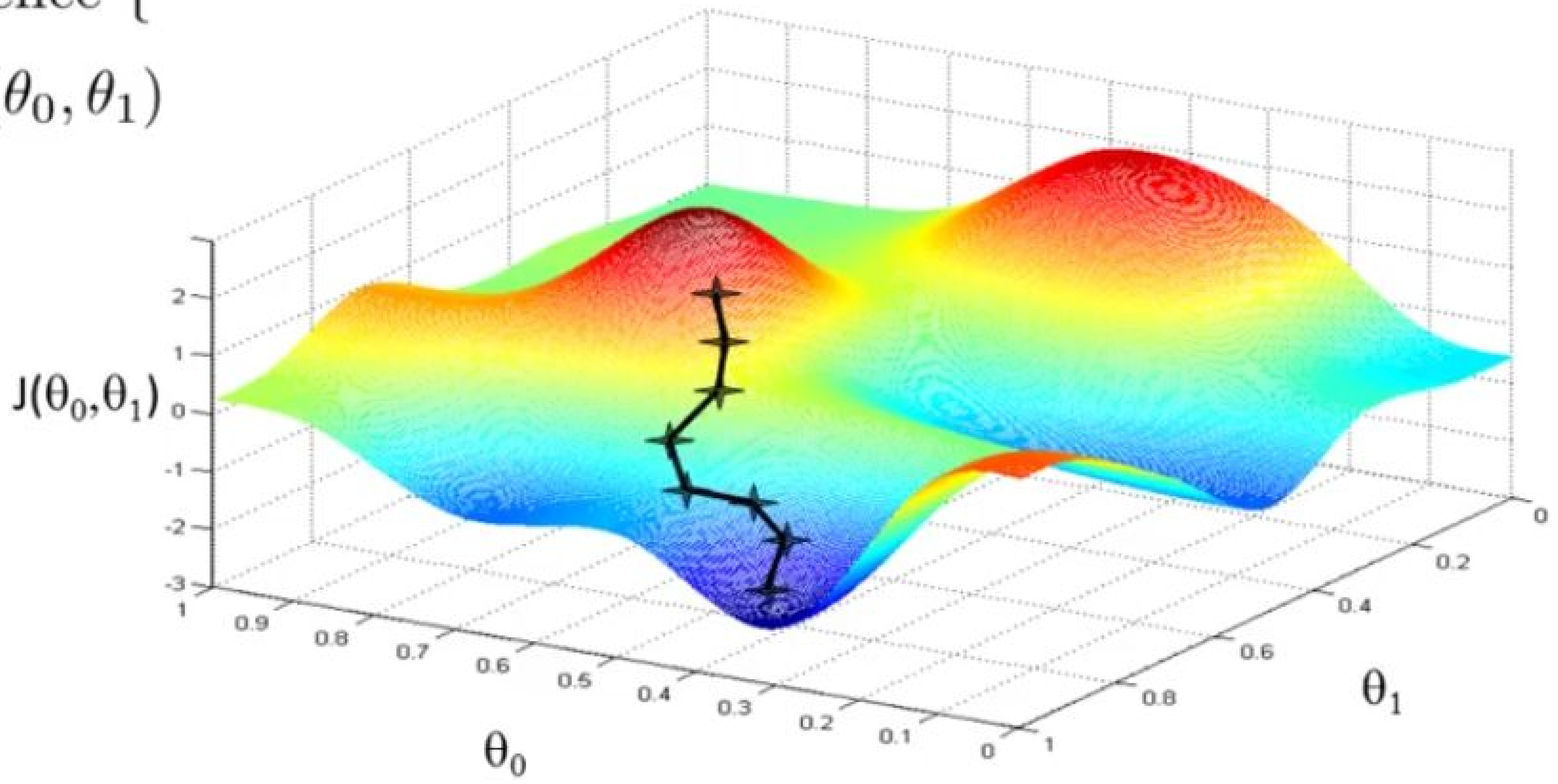


Image Source :https://www.coursera.org/learn/machine-learning

Linear Regression – Update Rule

- Update Rule

repeat until convergence: {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x_i) - y_i) x_i)$$

}

- Why?

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_\theta(x) - y)^2 \\
&= 2 \cdot \frac{1}{2} (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} (h_\theta(x) - y) \\
&= (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{n} \theta_i x_i - y \right) \\
&= (h_\theta(x) - y)\, x_j
\end{aligned}$$

Image Source :https://www.coursera.org/learn/machine-learning

Linear Regression Algorithm

- Given: training examples (x_i, y_i), $i = 1 \dots m$

  let $\theta = (0,0 \dots 0) \in R^n$

  repeat until convergence {

  $$d = (0,0 \dots 0) \in R^n$$

  for $i = 1 \dots m$ {

  $$y_i' = \theta^T x_i$$

  $$error = y_i - y_i'$$

  $$d = \mathrm{d} + \mathrm{error} * \mathrm{x_i}$$

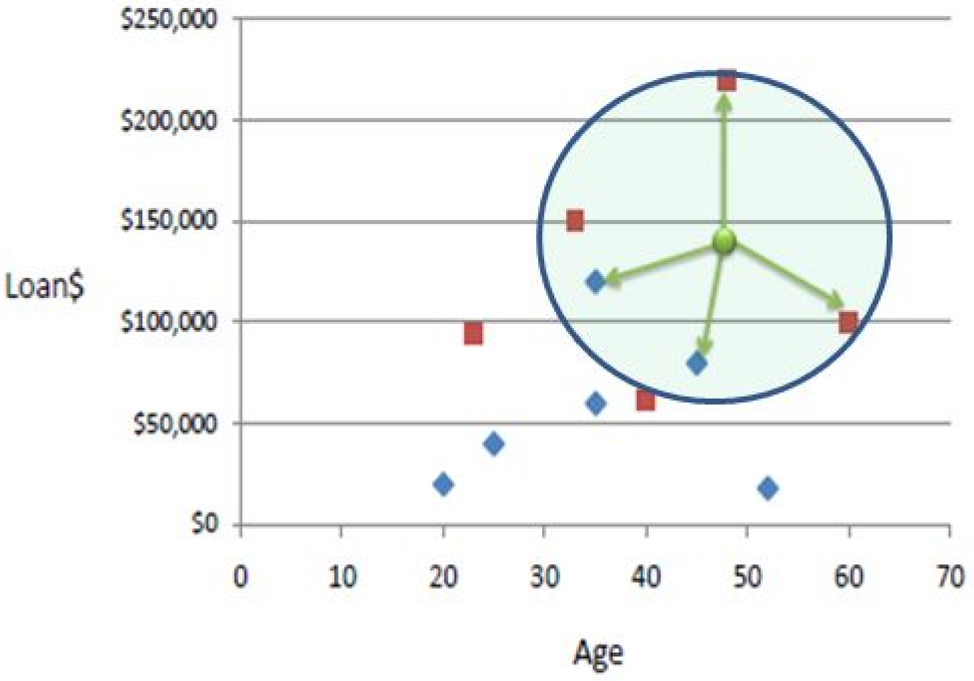
  }

  $$\theta = \theta - \alpha * d$$

KNN Regression

- Find the K nearest neighbors and take an average
- Could also take a weighted average
- Take care of feature normalization

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$



Age	Loan Amount	Annual Income
20	$24,000	$100,000
23	$95,000	$65,000
25	$40,000	$95,000
35	$60,000	$85,000
32	$150,000	$100,000
35	$120,000	$95,000
40	$65,000	$120,000
45	$72,000	$20,000
53	$20,000	$200,000
48	$223,000	$50,000
60	$100,000	$300,000
48	$140,000	???

Unsupervised Learning

- Find hidden structure / regularity in unlabeled data

- **Clustering** - grouping objects such that objects within same group are more similar than those across groups

- **Dimensionality Reduction** - given data points in ***n*** dimensions, represent them in ***d*** dimensions with minimal information loss, such that **d < n**

Clustering Example

Sports

Politics

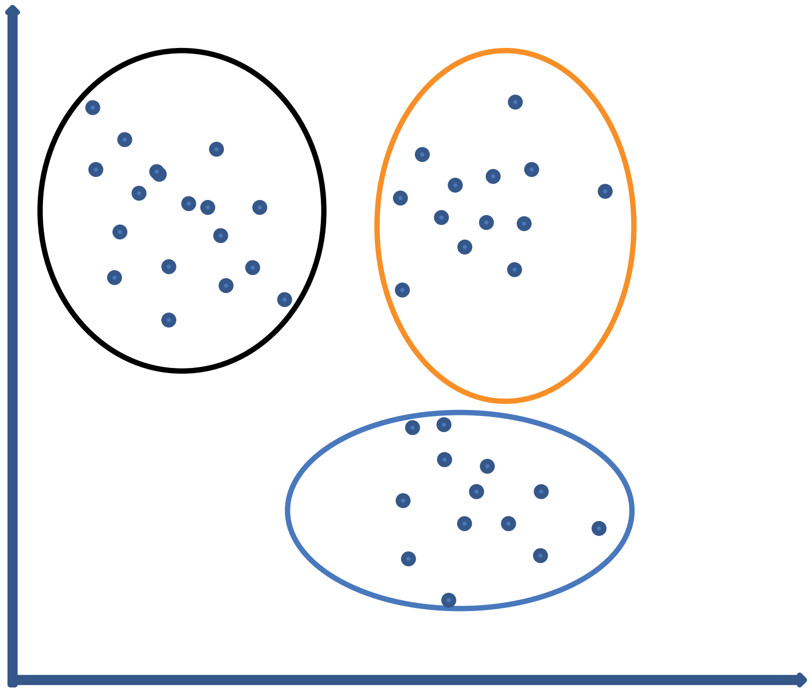
Music

Clustering News Articles

K-Means Clustering Algorithm

- Finding K natural groups in the dataset; high intra-cluster similarity & low inter-cluster similarity

- Similarity measured by a metric (e.g. Euclidian distance, cosine distance)

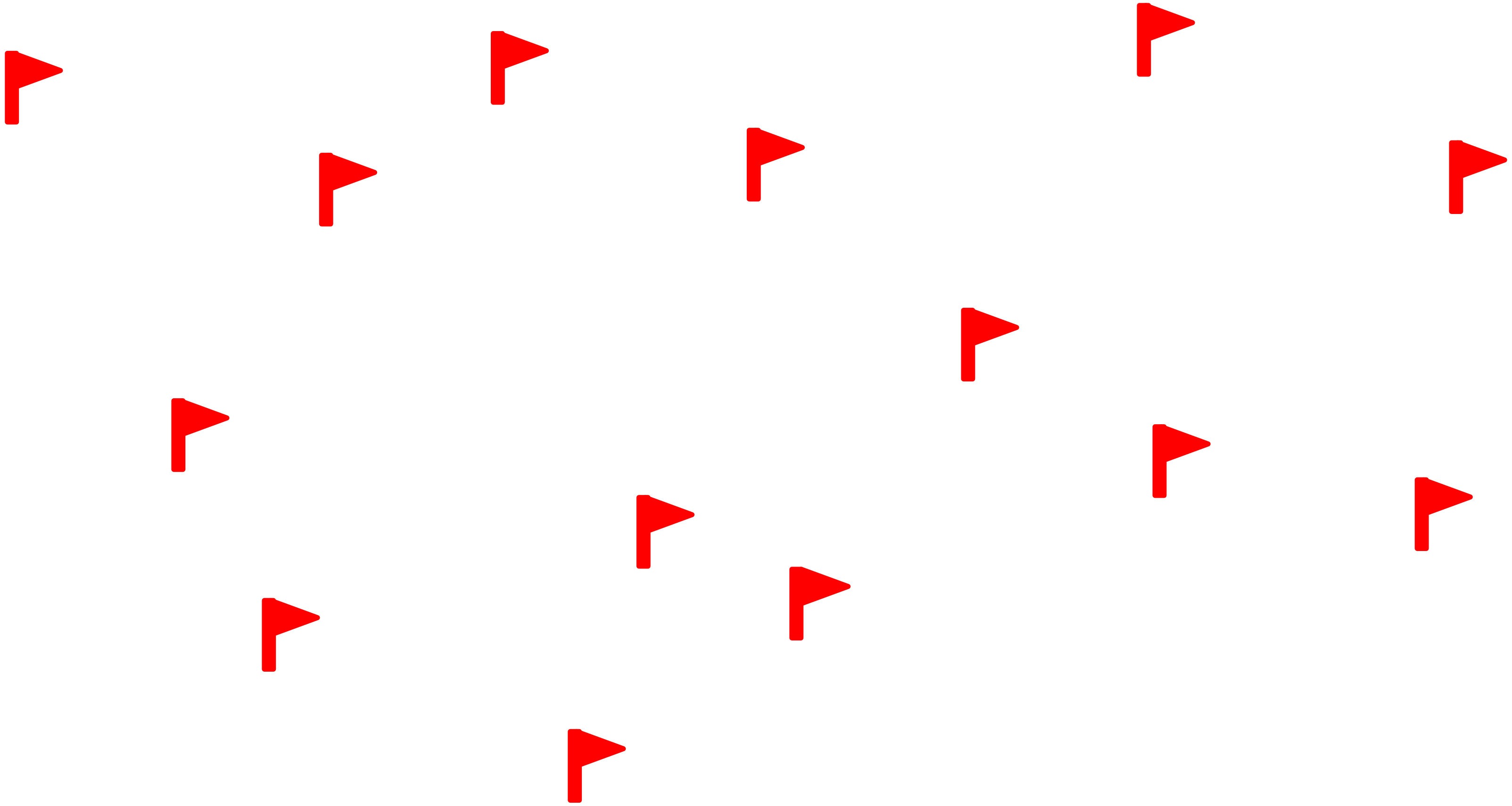


if **p** = (p_1, p_2) and **q** = (q_1, q_2) then the Euclidian distance is given by

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}.$$

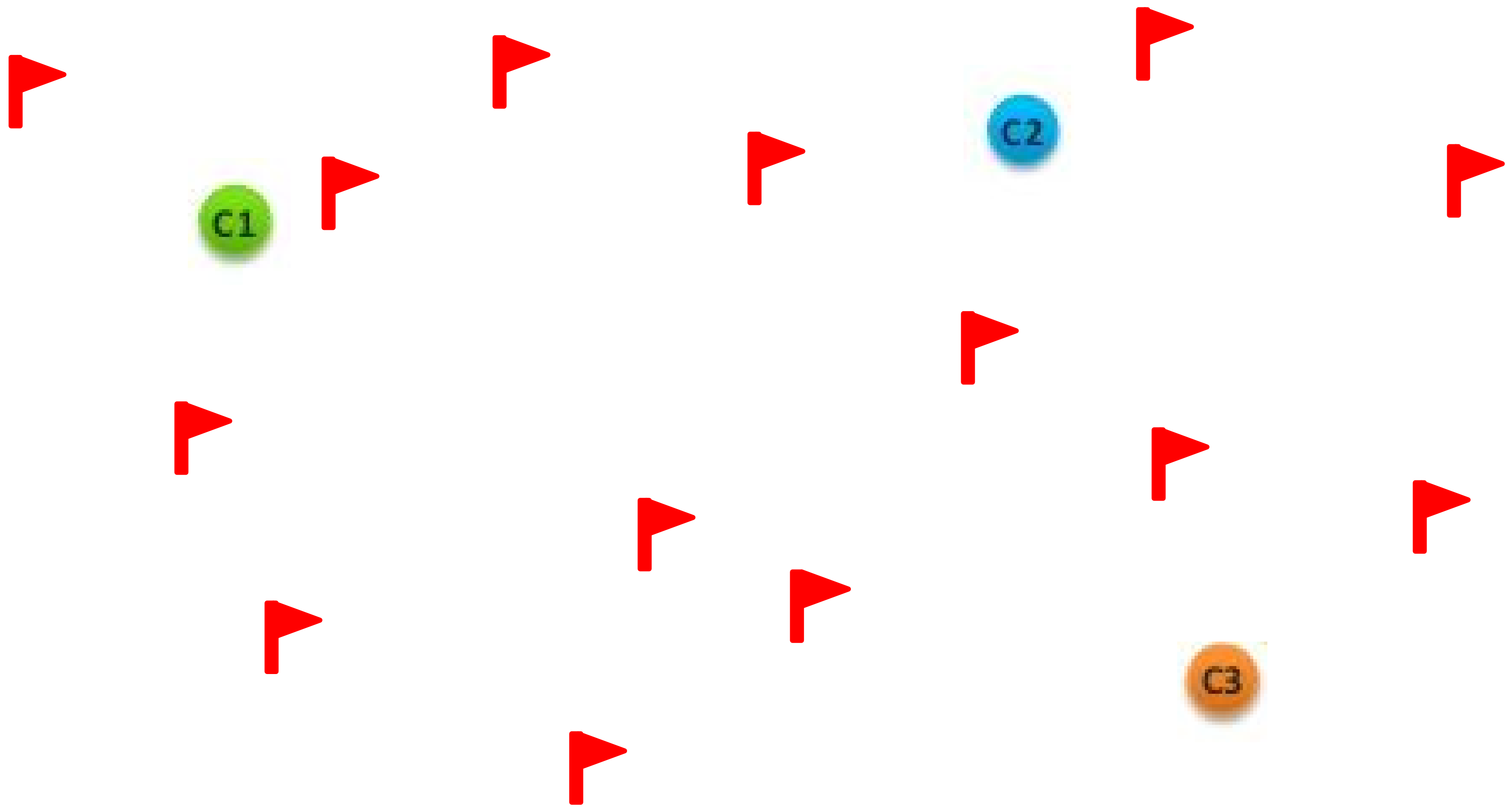
K-Means Intuition: deciding warehouse location

- Task: Set up 3 warehouses, given the following delivery points



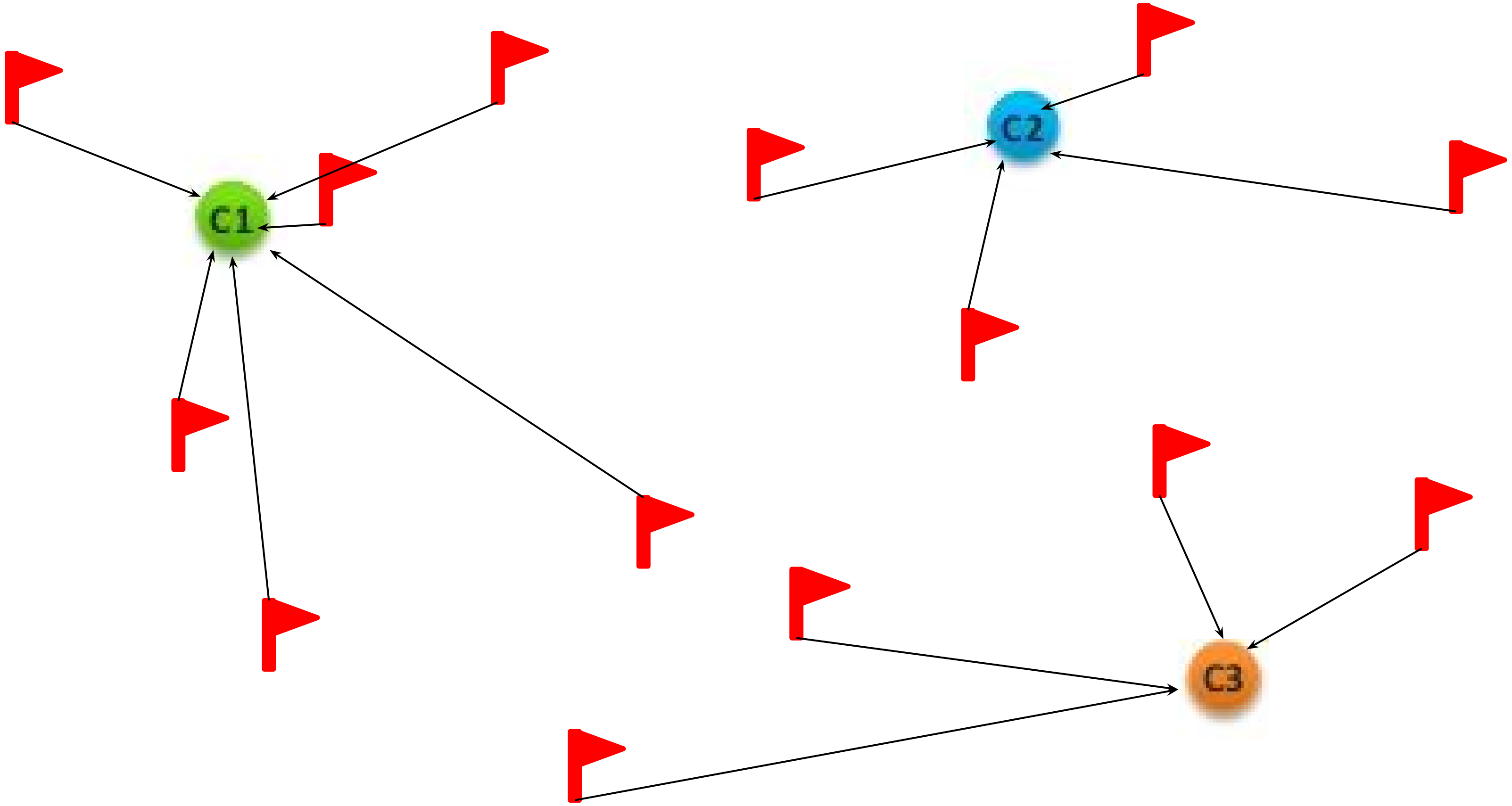
K-Means Intuition

- **Initialization:** randomly assigns cluster centers



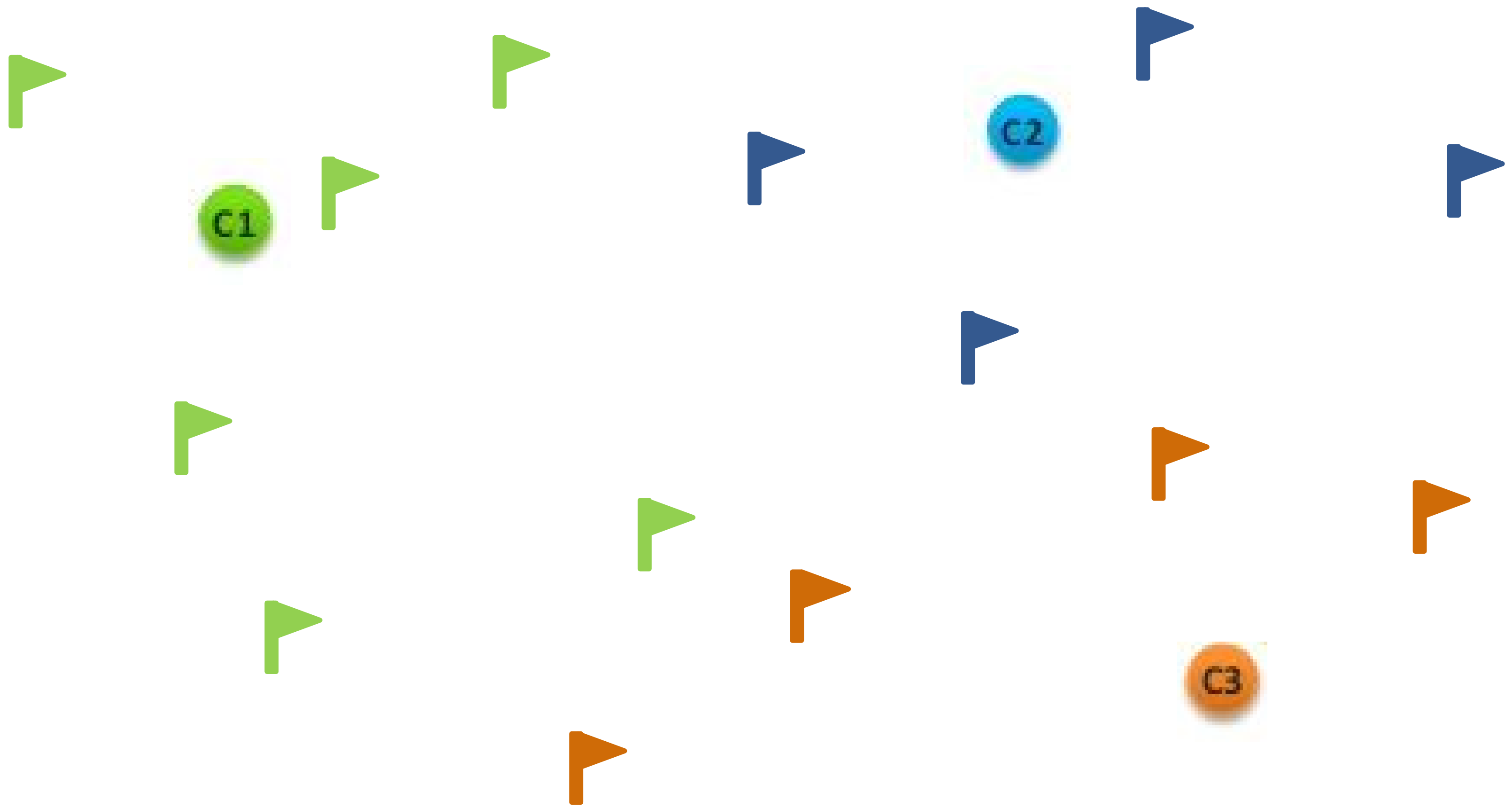
K-Means Intuition

- Assign each point to its nearest cluster center



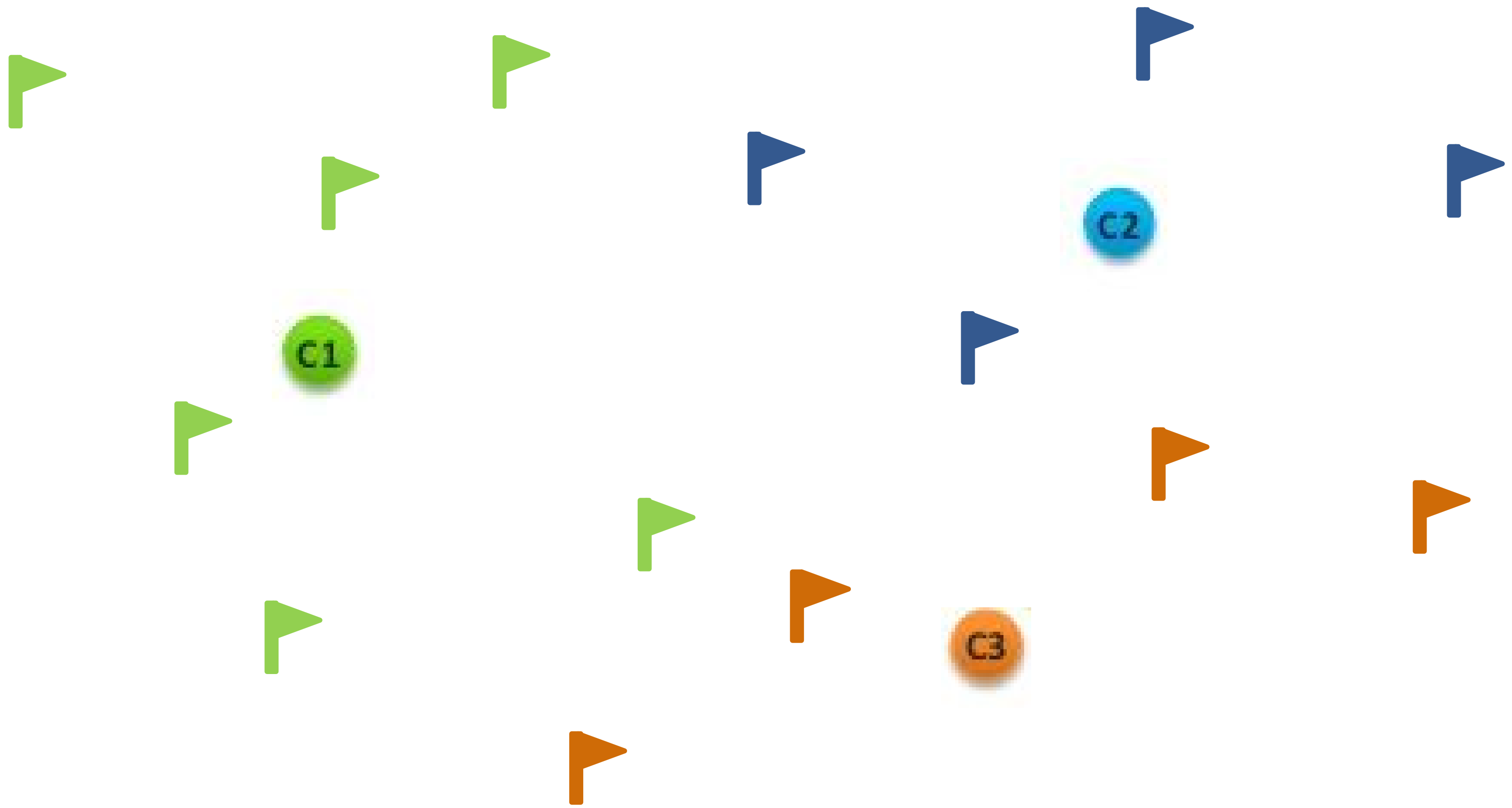
K-Means Intuition

- Assign each point to its nearest cluster center



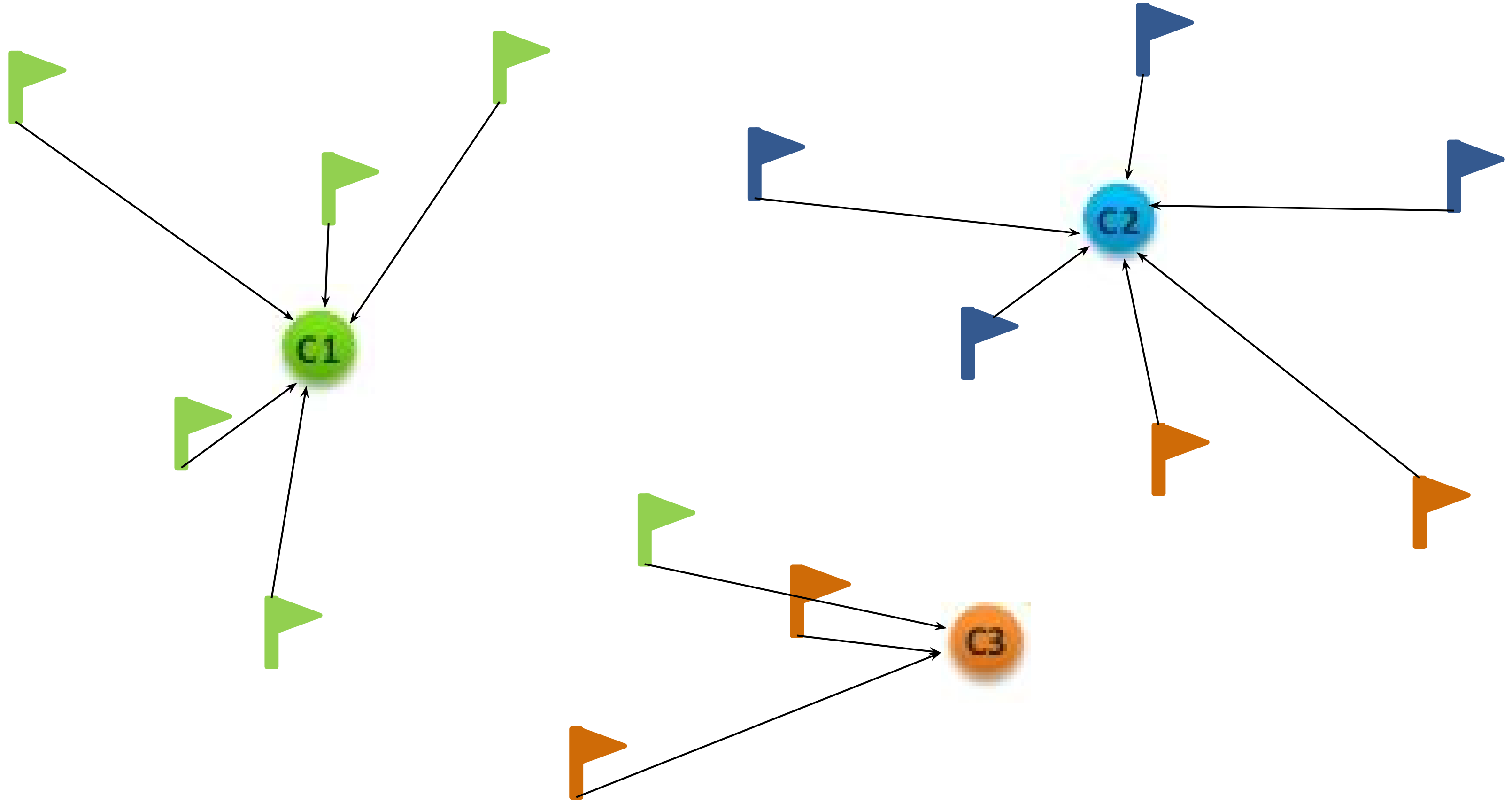
K-Means Intuition

- Calculate centroids based on current allocation; shift cluster centers to centroids



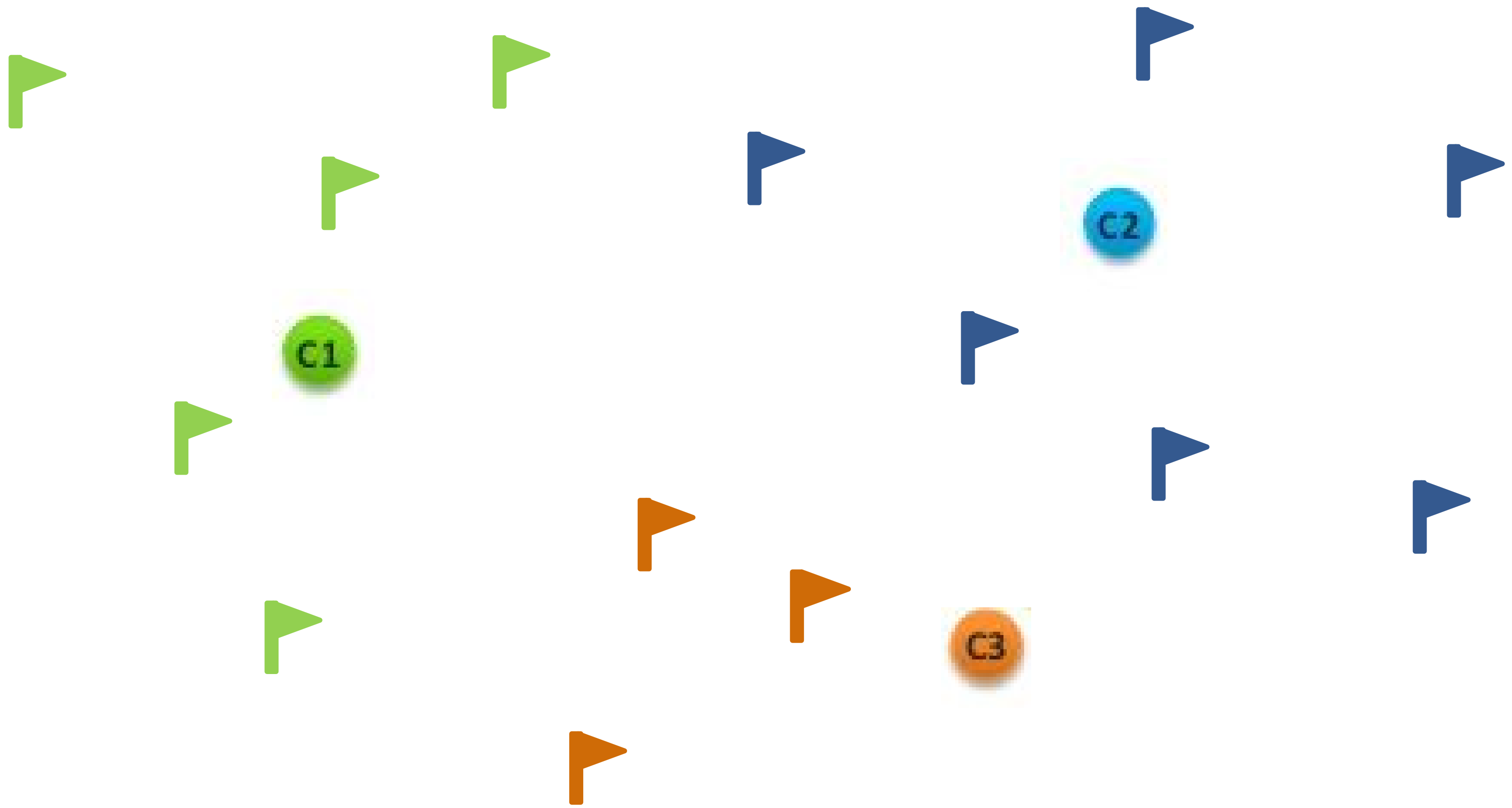
K-Means Intuition

- Re-assign each point to its nearest cluster center



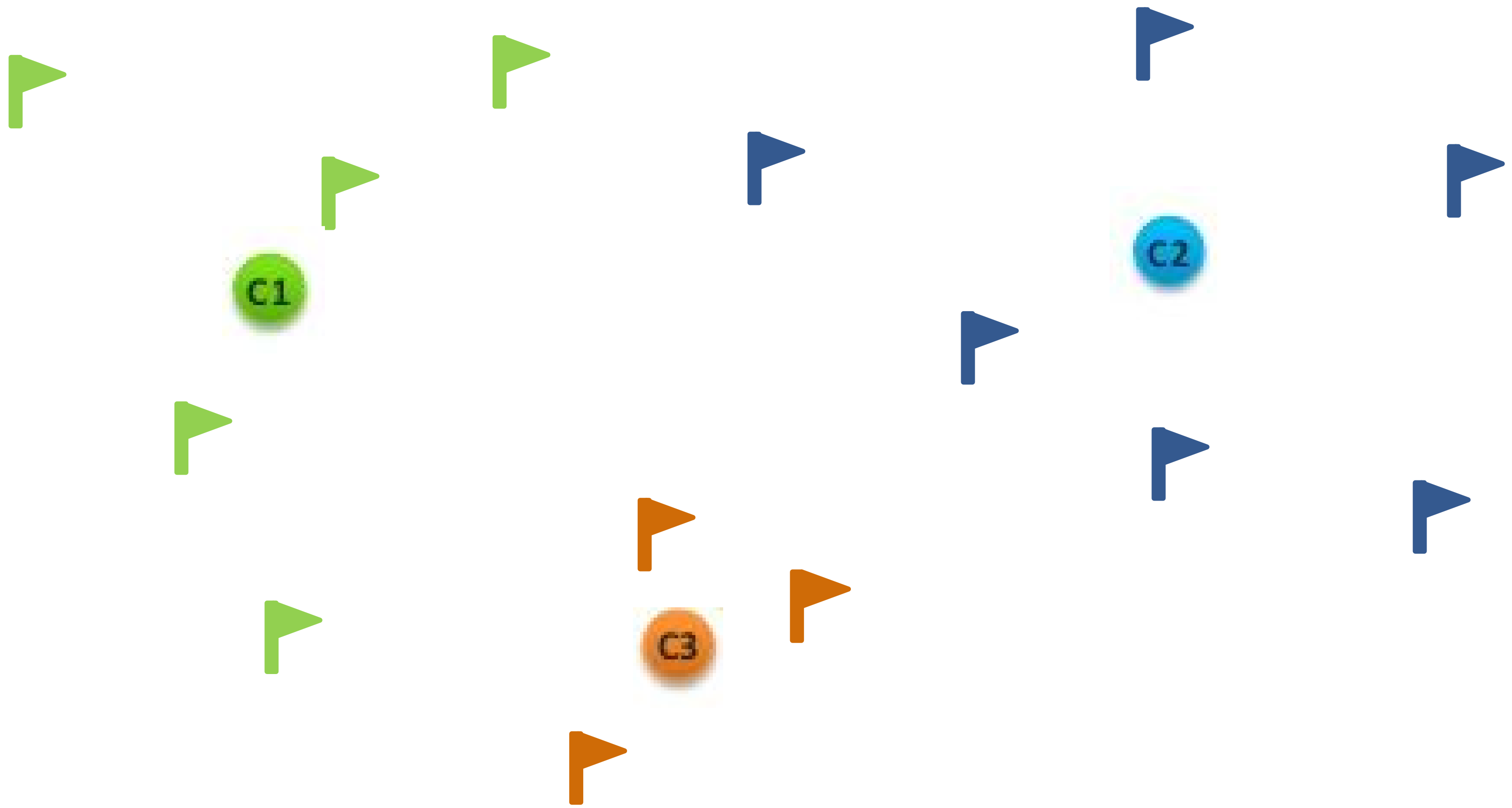
K-Means Intuition

- Form new clusters



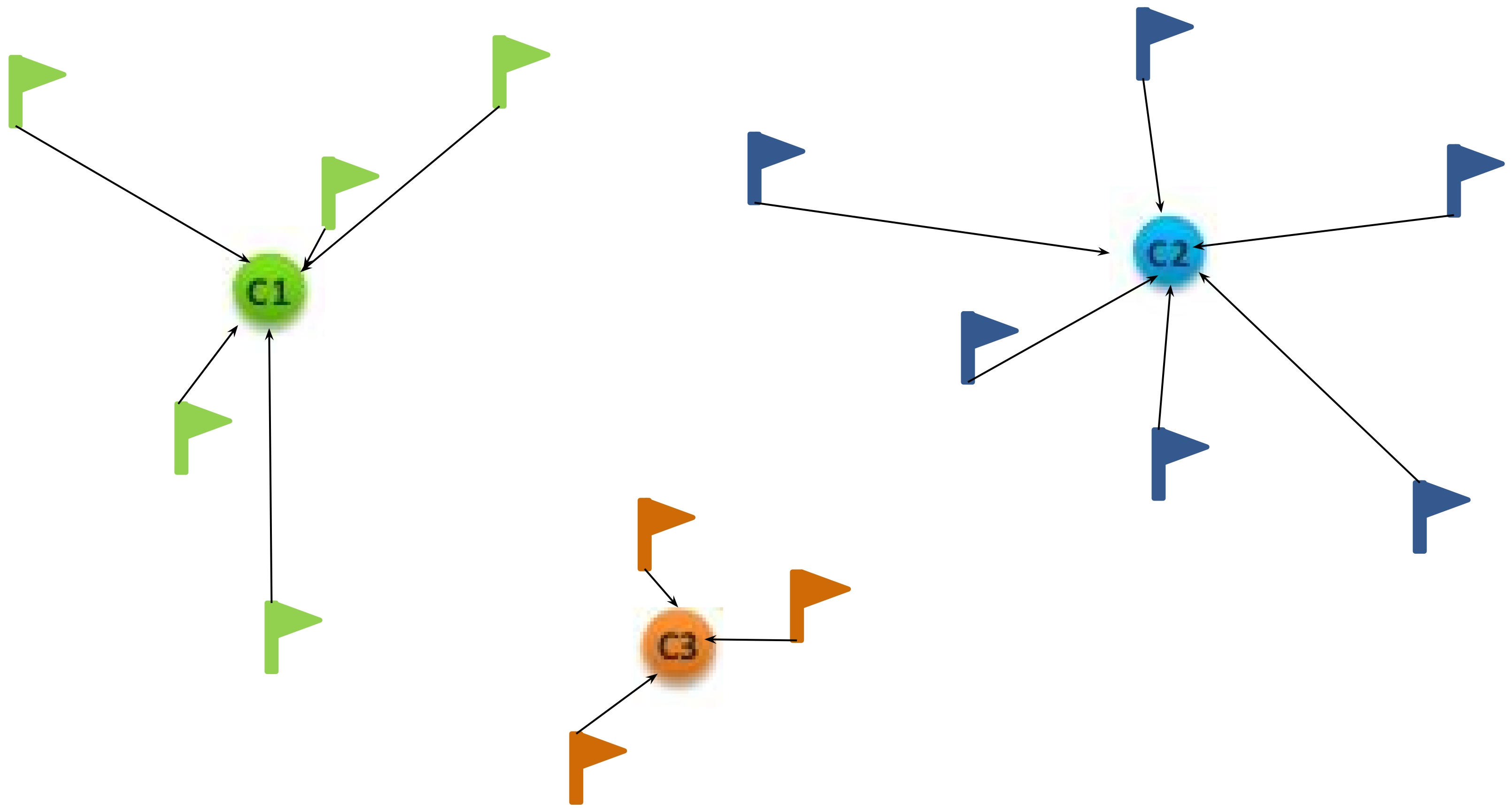
K-Means Intuition

- Re-calculate cluster centers again



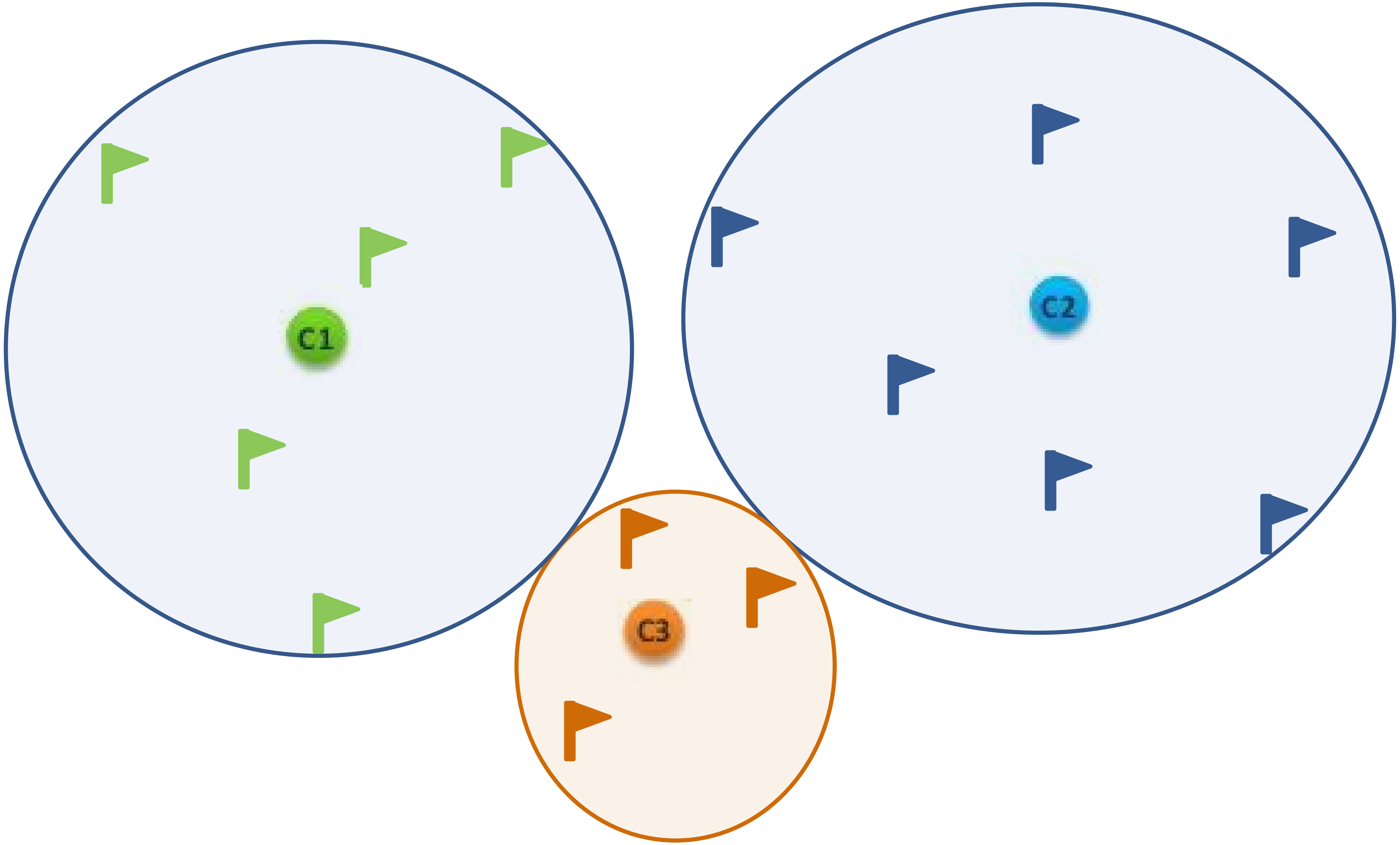
K-Means Intuition

- Re-assign each point to its nearest cluster center

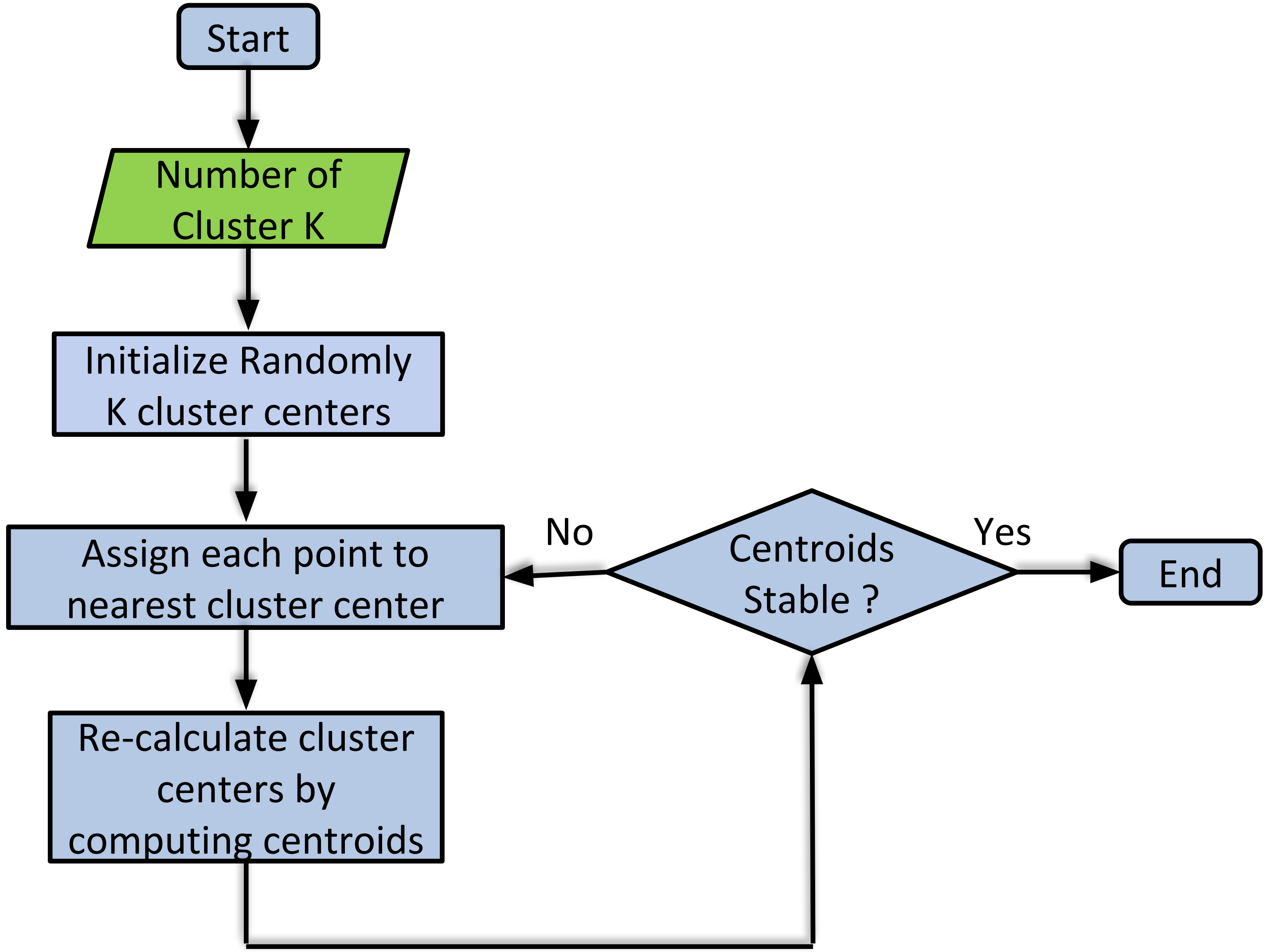


K-Means Intuition

- Convergence check - No reassignments of points => final 3 clusters



K-Means Flowchart



K-Means Algorithm

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.

2. Repeat until convergence: {

   For every i, set

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

   For each j, set

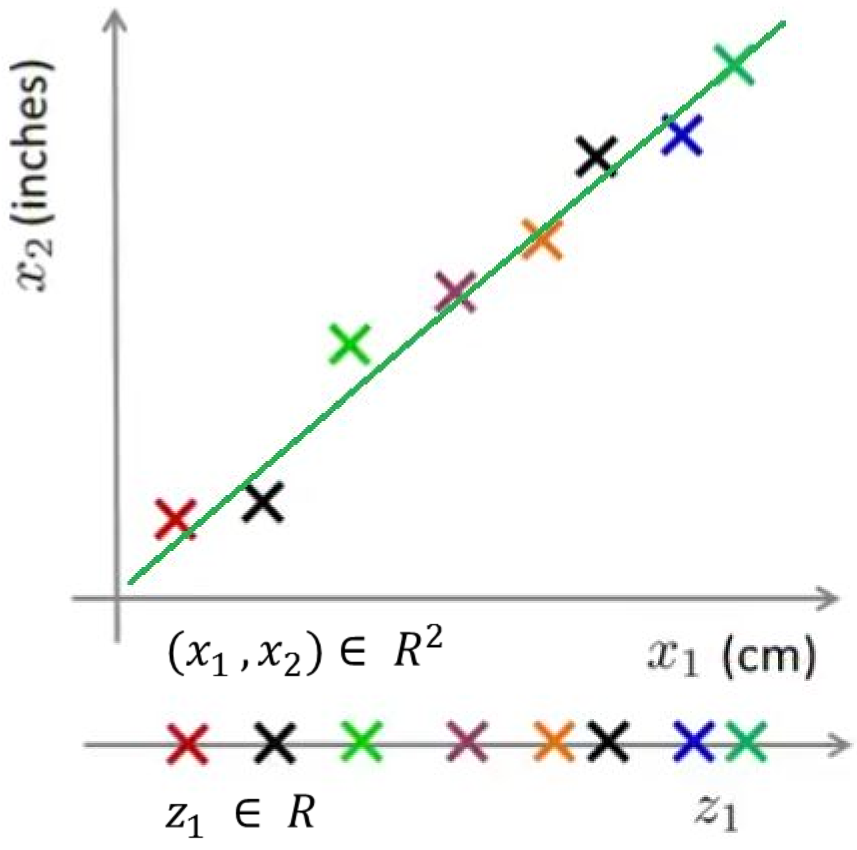
$$\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}.$$

}

Image Source : http://stanford.edu/~cpiech/cs221/handouts/kmeans.html

Dimensionality Reduction - Intuition

- Compact representation of data, minimize information loss
- Discover “intrinsic dimensionality” (e.g. text representation)



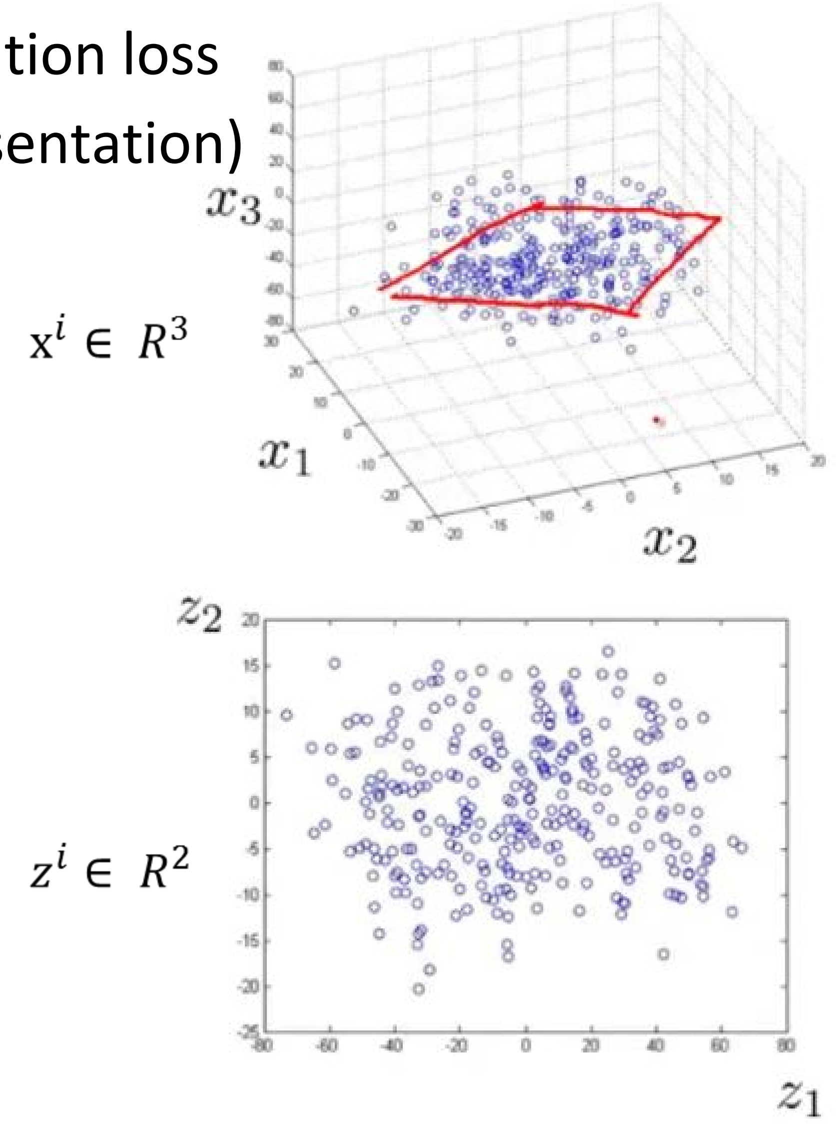


Image Source :https://www.coursera.org/learn/machine-learning

Principal Component Analysis (PCA)

- Project to lower dimension; minimize projection error
- U1 or U2 ?
- Why U1 is better?

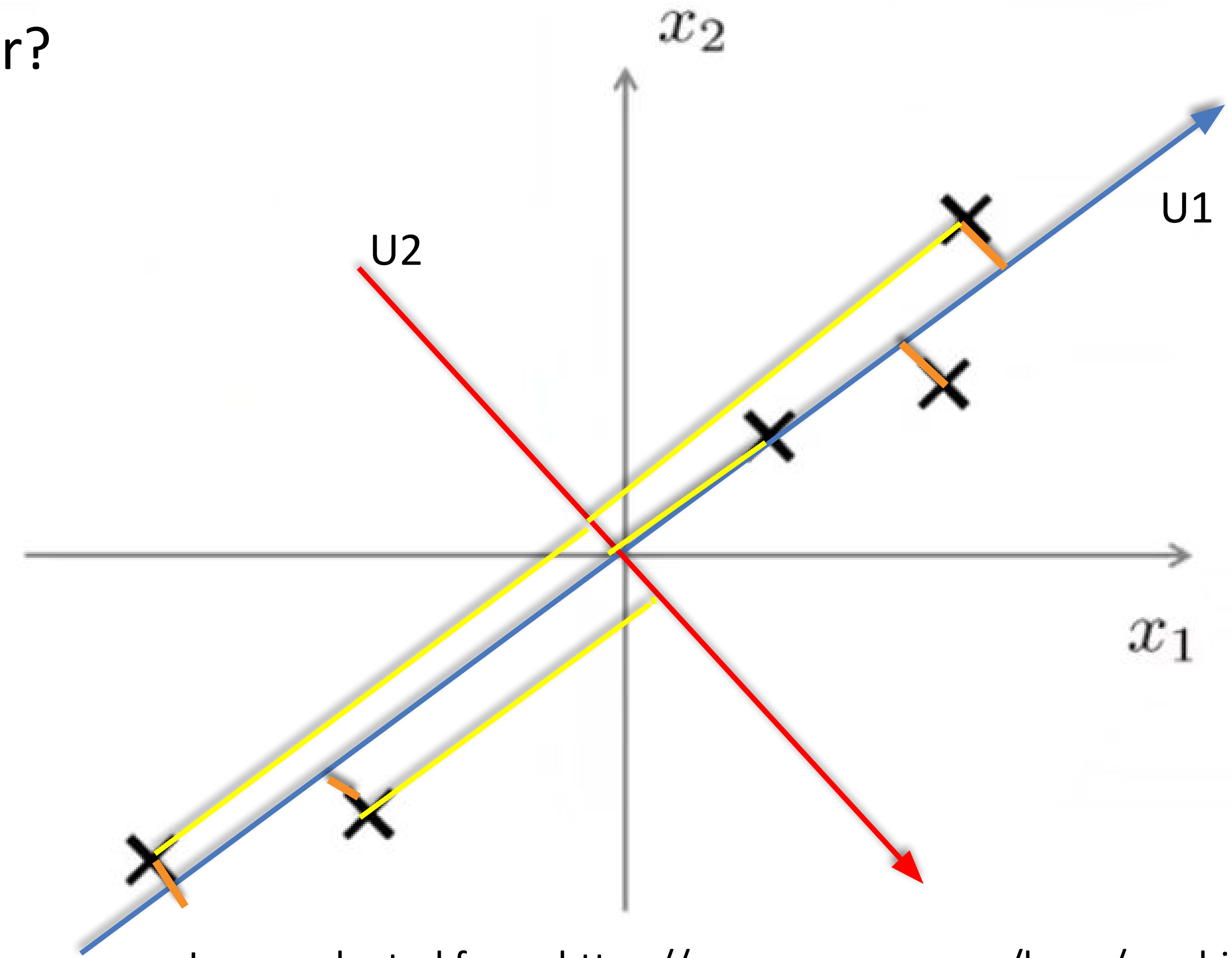


Image adapted from: https://www.coursera.org/learn/machine-learning

PCA: Finding the basis

- Mean normalization
  - From training data $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$, calculate mean $\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}$
  - Zero centering : $x_j^{(i)} = x_j - \mu_j$. (could optionally scale with max-min or std. deviation)
- Calculate covariance matrix: $\Sigma = \frac{1}{m}\sum_{i=1}^{n}(x^{(i)})(x^{(i)})^T$, vectored form = $\frac{1}{m} X^T X$
  - Note : it is an n x n matrix
- Calculate Eigen vectors : **[U,S,V] = svd(sigma)** ; U is n x n matrix

$$U = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \ldots & u^{(n)} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times n}$$

PCA: Projection & Reconstruction

- Reduce dimensionality by projecting to the new basis $U_{reduce} \in R^{n \times k}$

  $z = U_{reduce}^T \times x$, since $x \in R^{n\times 1}, z \in R^{k \times 1}$
- Reconstruction: $x_{aprox} = U_{reduce} \times z$; $x_{aprox} \in R^{n \times 1}$

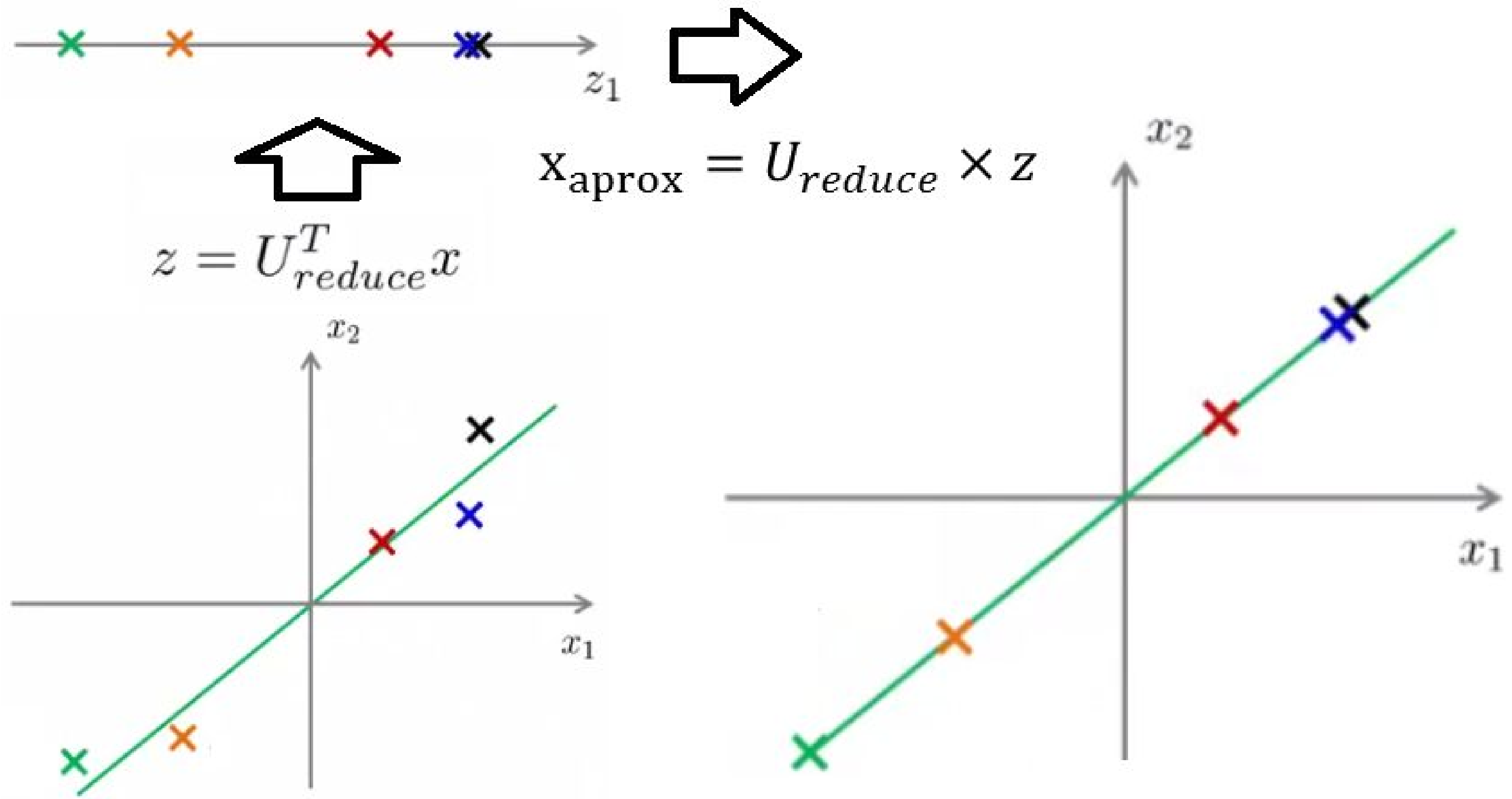


Image adapted from: https://www.coursera.org/learn/machine-learning

Choosing number of Principal Components

- Intuition:
  - PCA tries to minimize averaged squared projection error
  - Data is zero centered, so variance in data would be $\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2$

- Ratio of averaged squared projection error to total variance to be as small as possible

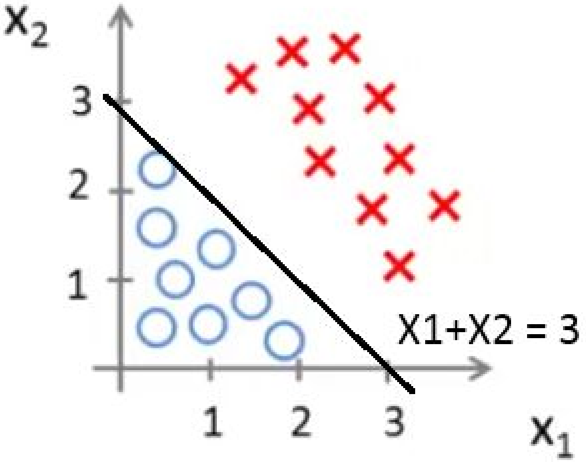
$$\frac{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x_{approx}^{(i)}\|^2}{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2} \leq 0.01 \qquad (1\%)$$

- Iteratively changing k and calculating this ratio is costly; what to do?

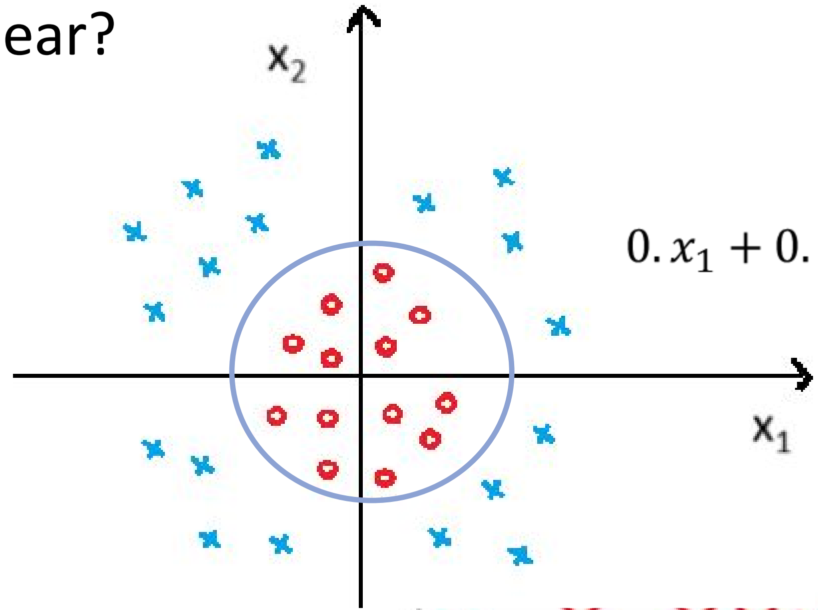
- Eigen value indicative of variance explained by corresponding eigen vector
  - Contribution from i^{th} eigen vector = $\frac{\lambda_i}{\sum_{i=1}^{n}\lambda_i}$

Neural Networks & Deep Learning : Intuition

- What if decision boundary is non-linear?



Vs.



$$0.x_1 + 0.x_2 + x_1^2 + x_2^2 = r^2$$

- A bit more complex case?

$$g(w_1.x_1 + w_2.x_2 + w_3.x_1x_2 + w_4.x_1^2x_2 + w_5.x_1x_2^2 \ldots.)$$

- Dimensionality much larger in practice
  - Which higher order terms to choose?

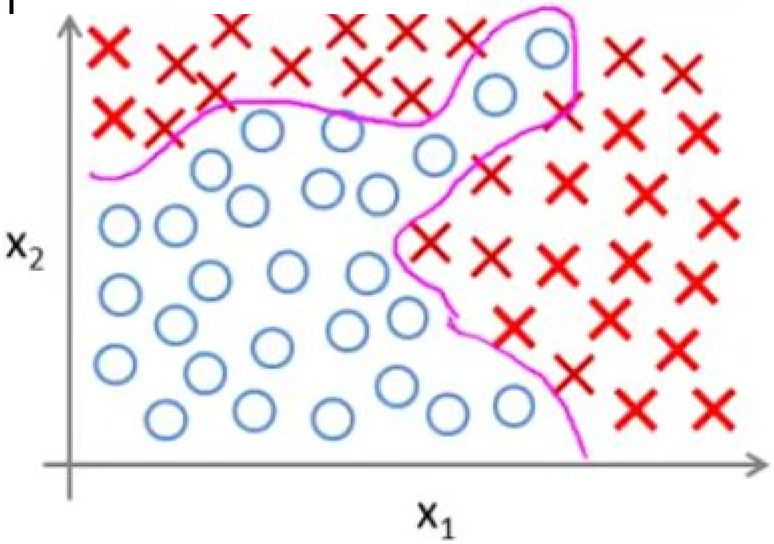


Image adapted from: https://www.coursera.org/learn/machine-learning

A practical problem: Computer Vision

- A 50x50 greyscale image => 2500 features
- A 50x50 RGB image => 7500 features
- Even 2 feature combinations are tricky
  - 2500 features = > around 3123750 features ($\approx \frac{n^2}{2}$)
  - 7500 features => around 28121250 features
- Simple linear hypothesis is not sufficient
  - Option 1: Careful feature engineering (e.g. SIFT, HOG etc.)
  - Option 2: Neural networks model complex non-linear hypothesis in high dimensions

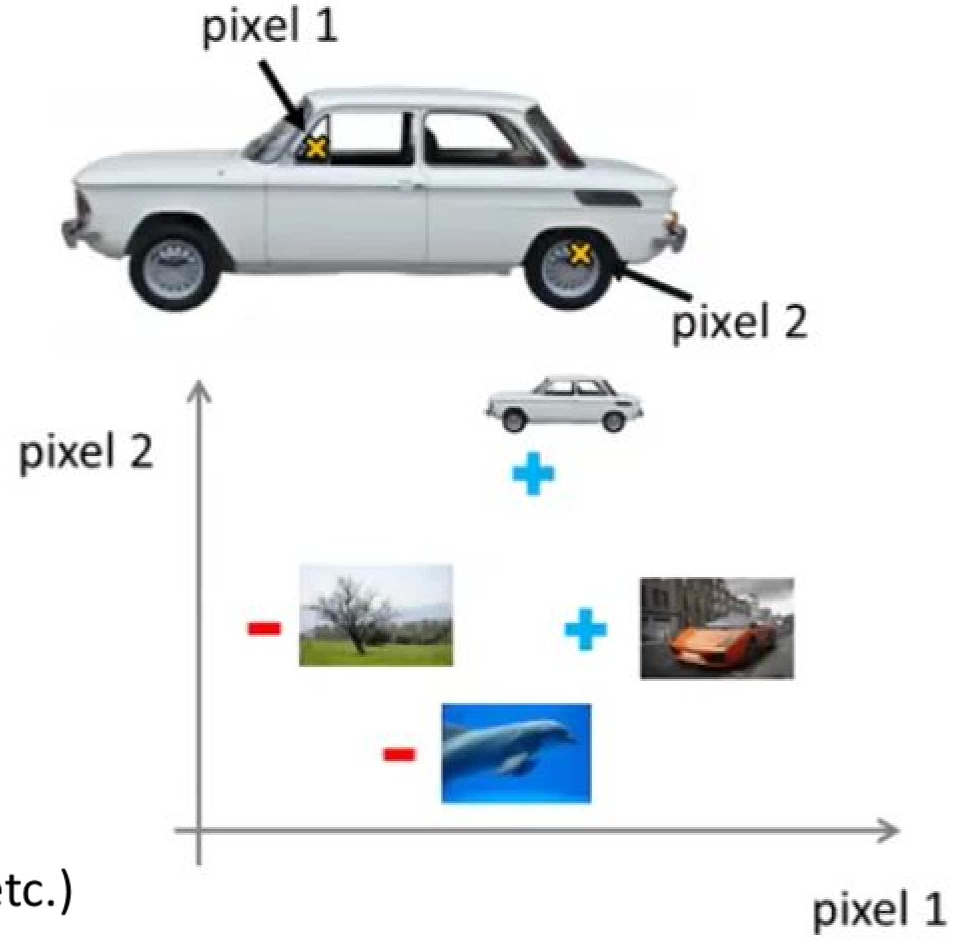
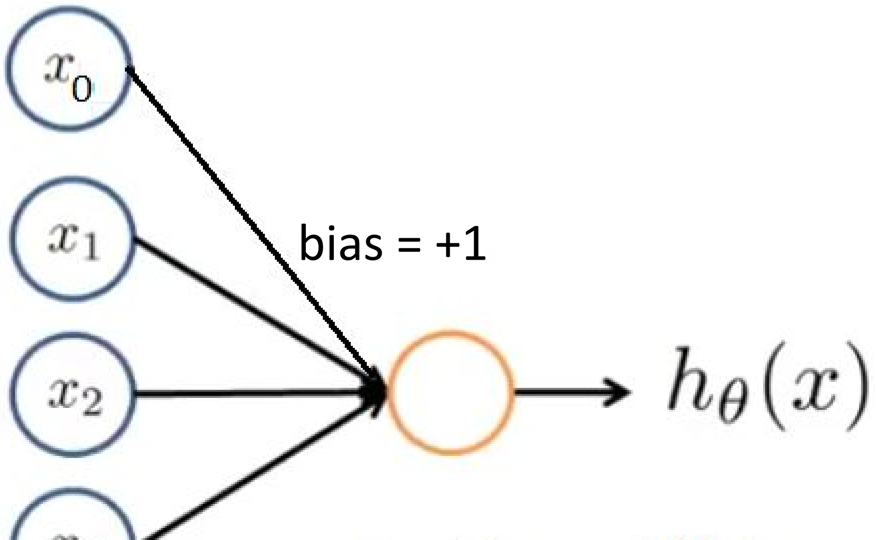


Image adapted from: https://www.coursera.org/learn/machine-learning

Logistic Unit

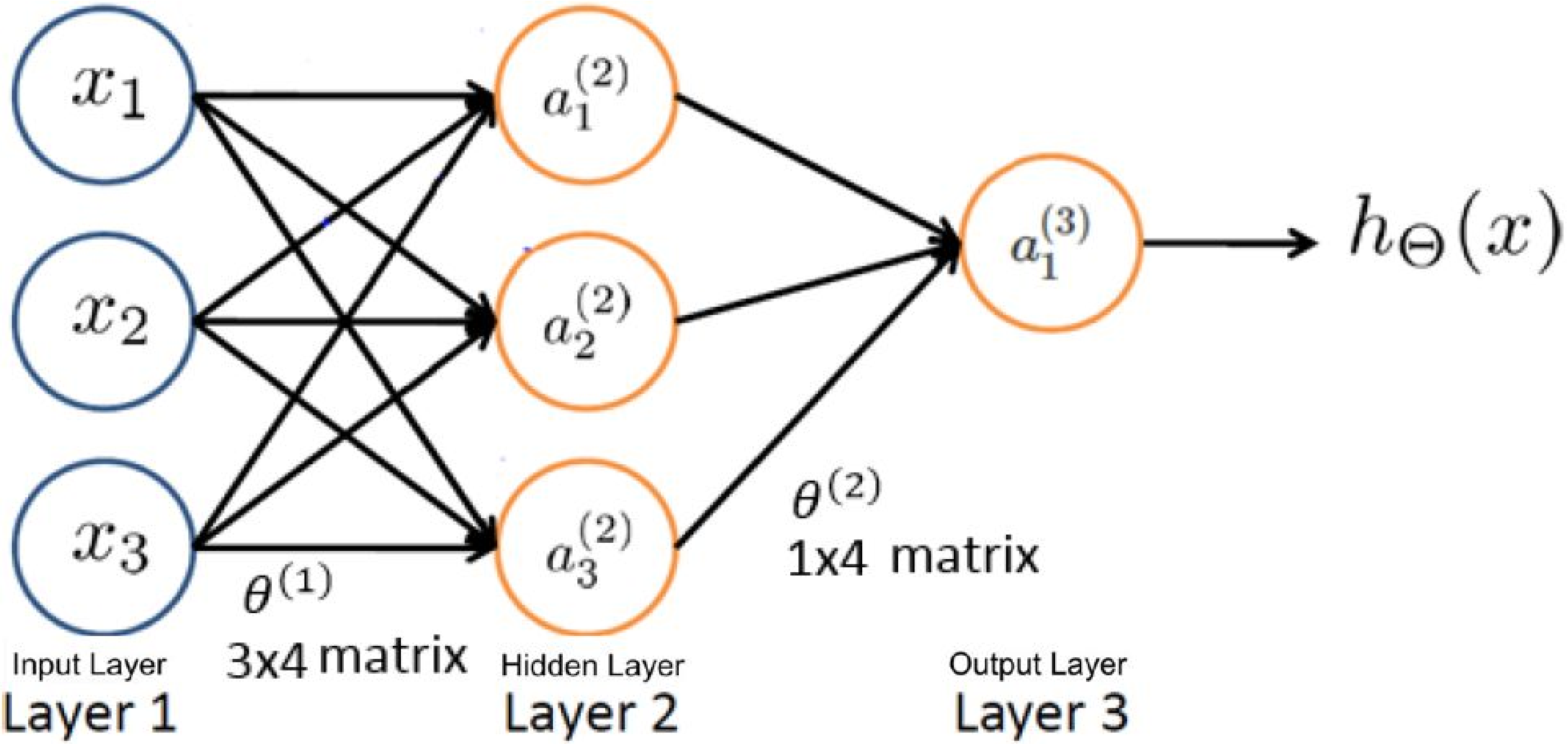
- Input is X (feature vector)
- Parameter vector/weight is theta

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$



$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

Neural Network



$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$$

Image adapted from: https://www.coursera.org/learn/machine-learning

Model Internals

- Forward propagation - from input to prediction
- Back propagation – from prediction to error, and finally to weight correction
  - Calculate final error, and back-calculate error associated with each neuron from preceding layer
- Cost function (multi-class): $h_\Theta(x) \in \mathbb{R}^K \quad (h_\Theta(x))_i = i^{th} \text{ output}$

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(h_\Theta(x^{(i)}))_k + (1-y_k^{(i)}) \log(1-(h_\Theta(x^{(i)}))_k)\right] + \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\Theta_{ji}^{(l)})^2$$

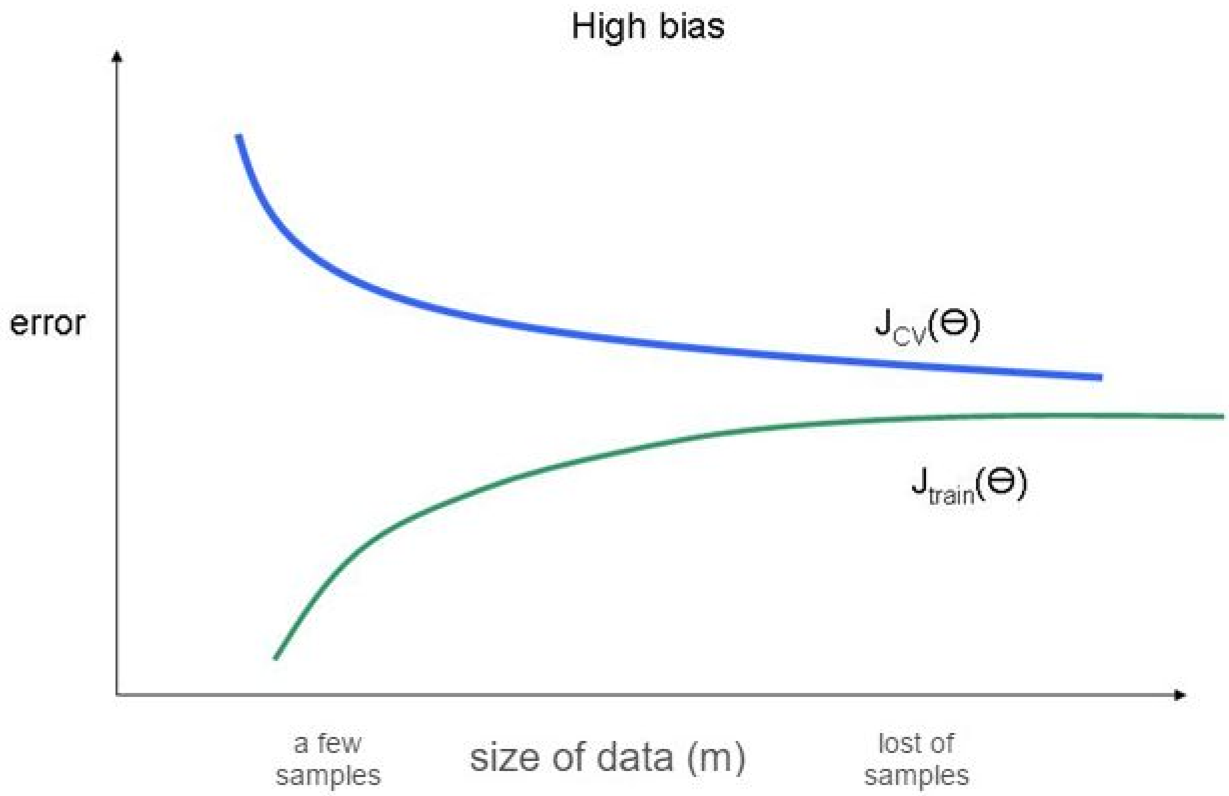
- Optimize cost function using gradient descend to obtain weights

Large scale ML != learning + parallelism

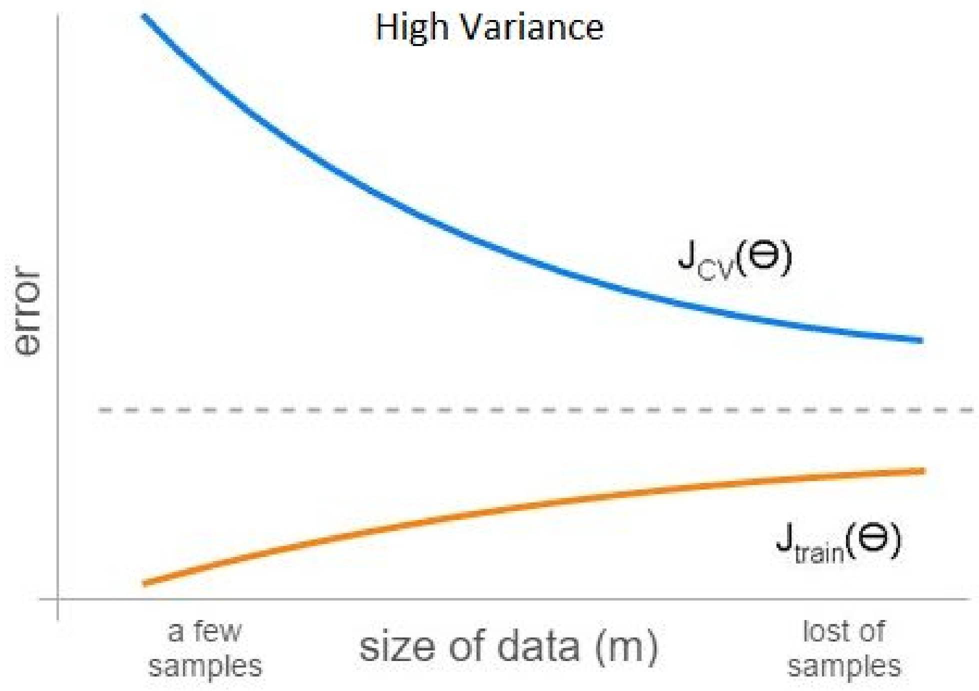
- Many millions of examples & many millions of features (e.g. web scale spam mgmt.)
- Some steps are parallelizable
  - Data pre-processing & extraction
  - Normalization
  - Model quality evaluation
- What about the learning step? It depends!
  - Naïve Bayes – only counting business, so easily parallelizable
  - Closed form matrix calculations (e.g. Normal Equations for linear regression – difficult)
  - Optimization using gradient descend (Stochastic Gradient Descend can help)
- Approximation is the key (e.g. Random projections, approx. gradient etc.)

Sampling vs. Full data

- Cant I sample a few instances (say 5000) from millions, and train?
  - Depends on the learning curve



Simply adding more data will not help
Increase features & then train on large data



Adding data would likely improve the model

Gradient Descent vs. Stochastic Gradient Descend (SGD)

- Consider optimizing linear regression cost function using gradient descend

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

(for every $j = 0, \ldots, n$)

}

Gradient calculation won't scale as m grows large

- SGD to rescue

1. Randomly shuffle (reorder) training examples
2. Repeat { // A few times (say 1..5)

   for $i := 1, \ldots, m$ {

   $$\theta_j := \theta_j - \alpha (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

   (for every $j = 0, \ldots, n$)

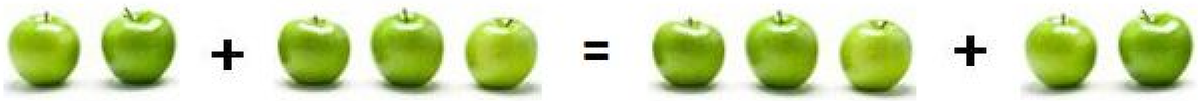
   }

   }

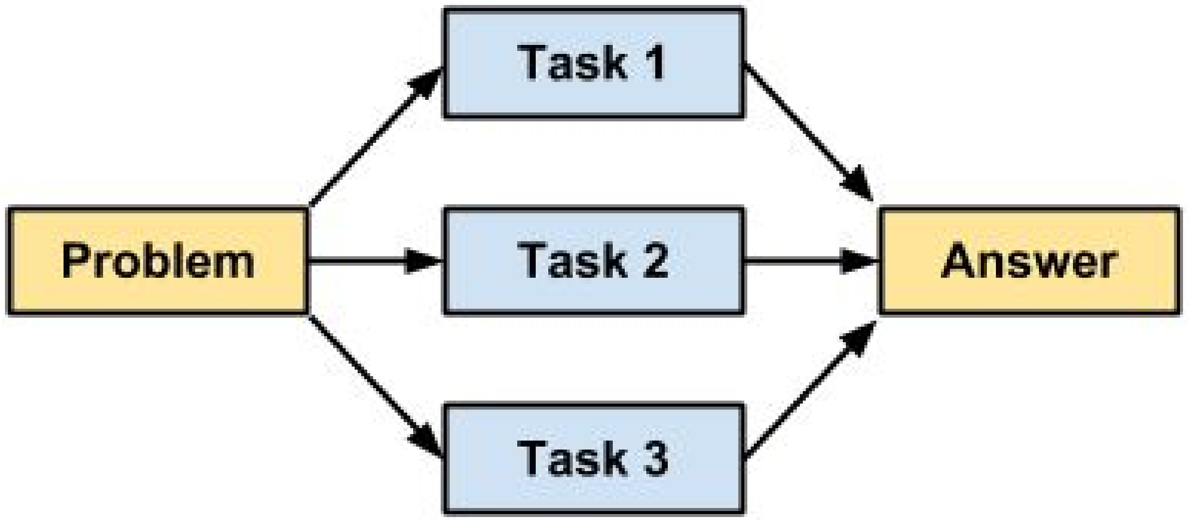
Gradient calculated per instance

Distributed Learning & Map-Reduce

- Divide & conquer + commutativity of addition => very powerful!



- Many learning algorithms can be expressed as computing sum of functions over training instances



Parallelizing Batch Gradient Descent

- Assume 400 instances, gradient calculation involves $\frac{1}{400}\sum_{i=1}^{400}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$
  - Parallelizing on 4 machines / cores to yield approx. 4X improvement

Split 1
(instances 1-100)

$$\text{Tmp1=}\sum_{ti=1}^{100} h_\theta\left(x^{(i)} - y^{(i)}\right)x_j^{(i)}$$

Split 2
(instances 101-200)

$$\text{Tmp2=}\sum_{i=101}^{200} h_\theta\left(x^{(i)} - y^{(i)}\right)x_j^{(i)}$$

Split 3
(instances 201-300)

$$\text{Tmp3=}\sum_{i=201}^{300} h_\theta\left(x^{(i)} - y^{(i)}\right)x_j^{(i)}$$

Split 4
(instances 301-400)

$$\text{Tmp4=}\sum_{i=301}^{400} h_\theta\left(x^{(i)} - y^{(i)}\right)x_j^{(i)}$$

$$g = \frac{1}{400}\sum_{i}^{4} Tmp_i$$

$$\theta_j = \theta_j - \alpha \times g$$

Key References

- CS 109 Data Science, Harvard University, http://cs109.github.io/2015/
- Machine Learning, Coursera, https://www.coursera.org/learn/machine-learning
- http://stanford.edu/~cpiech/cs221/handouts/kmeans.html
- https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/
- http://www.cs.utexas.edu/users/ear/nsc110/Mirrors/DSMirrorsArtificialIntelligence.ppt
- https://www.pycon.it/media/conference/slides/ai-e-machine-learning-cosa-bisogna-sapere.pdf

Thank You

www.flytxt.com | info@flytxt.com

DATA >>>> ECONOMIC VALUE