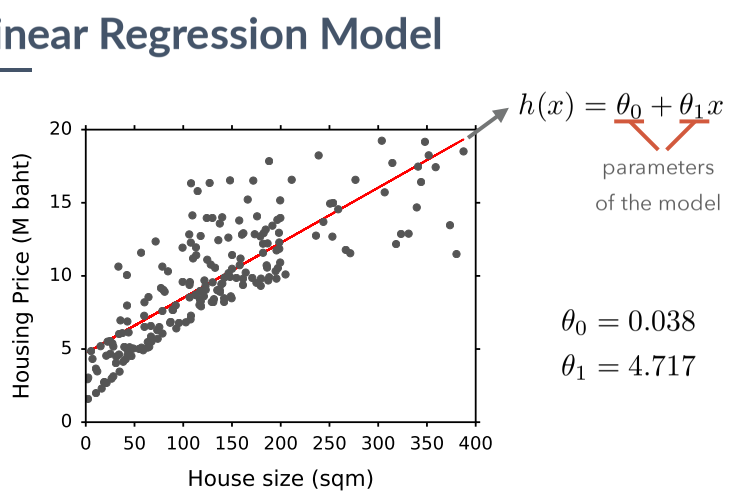
- Why Feature Selection
- How to do feature selection

LINEAR & LOGISTIC REGRESSION

Regression, Classification & Clustering

Instructor: Warasinee Chaisangmongkon, PhD

Linear Regression Model



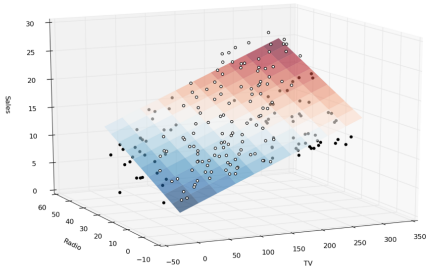
Linear Regression Model

$$h(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + ... + \theta_n x_n$$

$$h(x) = \Sigma_i^n \theta_i x_i$$

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_n \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \cdots \\ \theta_n \end{bmatrix}$$

$$\mathrm{h(x)} = \begin{bmatrix} \theta_0 & \theta_1 & \theta_2 & \dots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix} = \theta^T \mathbf{x}$$

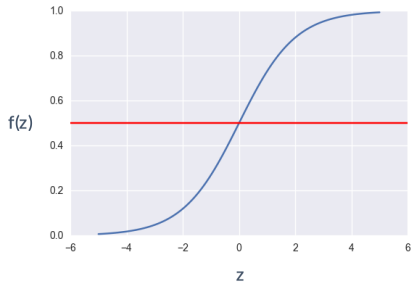


source: https://goo.gl/qFBRts

Logistic Regression Model

The Model

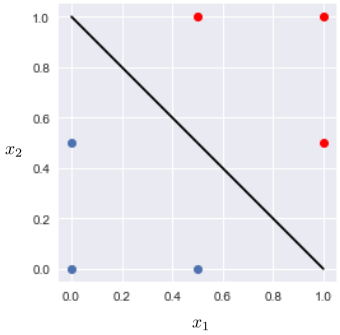
$$h(x) = f(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + ...)$$



$$f(z) = \frac{1}{1 + e^{-z}}$$

y' = 1, if f(z) > 0.5 or z > 0
y' = 0, if f(z) < 0.5 or z < 0

Logistic Regression Model



model: $h(x) = f(\theta_0 + \theta_1 x_1 + \theta_2 x_2) = f(z)$

parameter: $\theta_0 = -1$
$\theta_1 = 1$
$\theta_2 = 1$

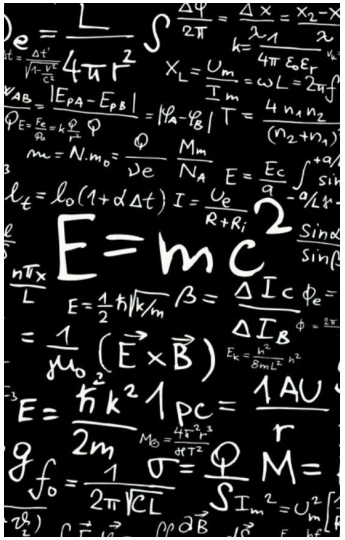
x1	x2	z	y'
1	1		
0	0		
0.5	0		
0	0.5		

Regression Performance

- R-squared: how close the data are to the fitted regression line.
- R-squared = the percentage of the response variable variation that is explained by a linear model.

R-squared = Explained variation / Total variation

- R-squared = 0 (model explains none of the variability of data).
- R-squared = 1 (model explains all of the variability of data).

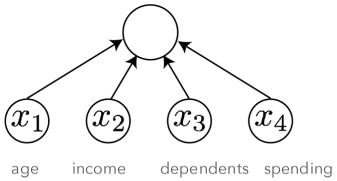


LOGISTIC REGRESSION QUIZ

A simple example

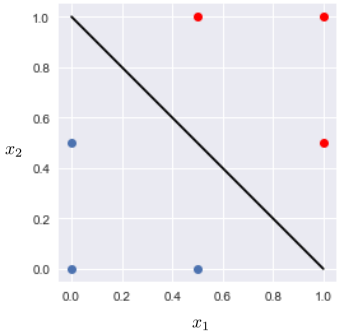
Let's look at a simple logistic regression procedure.

$$h = f(\Sigma_j w_j x_j)$$



x1	x2	x3	x4	history
40	50	0	30	1
25	40	2	35	1
18	10	0	12	0
34	22	1	10	1

Logistic Regression Model



model: $h(x) = f(\theta_0 + \theta_1 x_1 + \theta_2 x_2) = f(z)$

parameter: $\theta_0 = -1$

$\theta_1 = 1$

$\theta_2 = 1$

Logistic regression model draws a boundary at z=0

$$1 * x_1 + 1 * x_2 - 1 = 0$$

$$x_2 = -x_1 + 1$$

A simple example

We first need to do preprocessing, such as normalization and standardization.

x1	x2	x3	x4	history
40	50	0	30	1
25	40	2	35	1
18	10	0	12	0
34	22	1	10	1

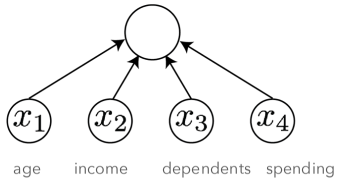
x1	x2	x3	x4	history
0.44	0.63	0	0.6	1
0.28	0.50	0.5	0.7	1
0.20	0.13	0	0.24	0
0.38	0.28	0.25	0.2	1

A simple example

Then fit the logistic regression to the data.

Suppose after fitting, here are the weight numbers.

$$h = f(\Sigma_j w_j x_j)$$



w1	0.7
w2	0.6
w3	-0.1
w4	-0.2

What the output means

14

Classification results

	X	W	X*W
age	0.44	0.7	0.31
income	0.63	0.6	0.38
dependent	0.00	-0.1	0.00
spending	0.60	-0.2	-0.12
		sum:	0.57
		h:	0.64

h indicates the probability of customer being good

h = 0.64

64% chance that he will be good

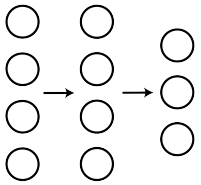
36% chance that he will be bad

Classification Cost Function

Classification problem with 2 classes : Binomial Log Loss

$$E(W) = -\frac{1}{m}\sum_{i=1}^{m}[y^i log(h(x^i)) + (1-y^i) log(1-h(x^i))]$$

Classification problem with more than 2 classes : Multinomial Log Loss



$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

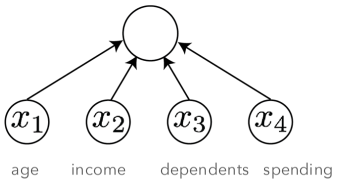
$$E(W) = -\frac{1}{m}\sum_{c=1}^{k}\sum_{i=1}^{m}[y_c^i log(h(x^i)_c) + (1-y_c^i) log(1-h(x^i)_c)]$$

A simple example

13

Let us make prediction for a single customer...

$$h = f(\Sigma_j w_j x_j)$$



	X	W	X*W
age	0.44	0.7	0.31
income	0.63	0.6	0.38
dependent	0.00	-0.1	0.00
spending	0.60	-0.2	-0.12
		sum:	0.57
		h:	0.64

Regression Cost Function

Regression

For single output unit:

$$E(W) = \frac{1}{2m}\sum_{i=1}^{m}[y^i - h(x^i)]^2$$

For multiple output units:

$$E(W) = \frac{1}{2m}\sum_{c=1}^{k}\sum_{i=1}^{m}[y_c^i - h(x^i)_c]^2$$

- What is decision tree
- How it is constructed

DECISION TREE

Classification Performance

- Log Loss
- Accuracy
- Precision
- Recall
- F1-Measure

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

$$PRE = \frac{TP}{TP + FP}$$

$$REC = TPR = \frac{TP}{P} = \frac{TP}{FN + TP}$$

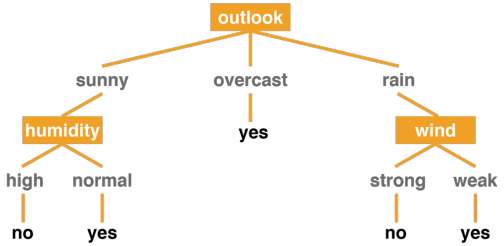
$$F_1 = 2 \cdot \frac{PRE \cdot REC}{PRE + REC}$$

Decision Tree

- Decision tree is a very old and simple idea. Today, it is perhaps the most popular machine learning algorithm due to the following reasons.
  - 1. Easy to understand
  - 2. Easy to implement (not a lot of parameters to tweak)
  - 3. Can be used for both classification and regression problems

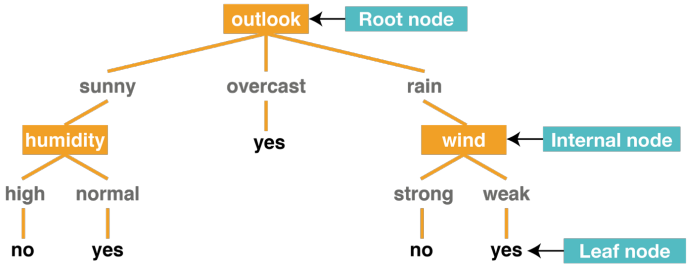
Decision Tree

- Here's an example of a decision tree for classification. Is Josh going to play tennis today?



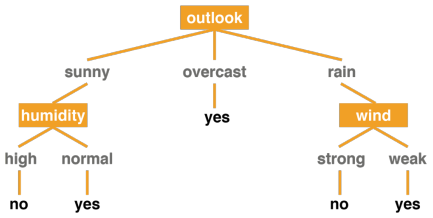
Decision Tree

Root Node, Internal Node, Leaf Node



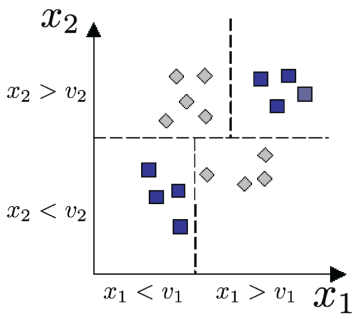
Decision Tree

- The y variable is {yes, no} There are 4 features:
  - x1 = Outlook, {sunny, overcast, rain}
  - x2 = Temperature, {hot, warm, cold}
  - x3 = Humidity, {high, normal} x4 = Wind, {strong, weak}



Decision Boundary

Decision trees divide the space into axis-parallel rectangles and label each rectangle with class membership.

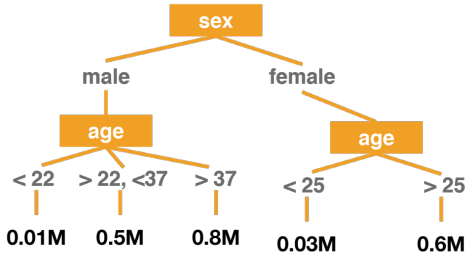


Decision Tree & Continuous Variable

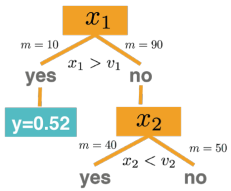
Both x (features) and y (predicted value) can be continuous variables, for example...

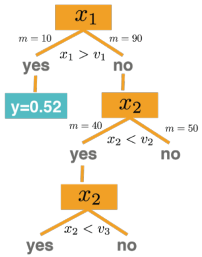
Predict amount of money in saving account of single people (Y) from the following attributes:

- x1 = age (15-65 years old)
- x2 = gender (male or female)



Constructing Decision Tree

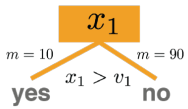


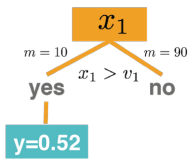


- On the other hand, you may decide to add another internal node which focuses on another feature.
- Repeat the process until we account for all samples.

Constructing Decision Tree

One can construct a decision tree by a constructive search. Suppose we have a dataset with m = 100 samples.





- Decide on the feature at the root node and a boundary to create branches.
- Decide whether to add another internal node. If there's no use in creating another internal node, just add a leaf node and make a prediction.

Constructing Decision Tree

- At a given node, you always have to make a decision. Do I continue adding a node to the tree?
  - if yes, which feature do I use to split the tree, at what threshold?
  - if no, how do I make a prediction?

Pseudo-Algorithm

Function: Build subtree
Require: node n, data at the node D

(n_L, n_R, D_L, D_R) = FindBestSplit(D)

if StoppingCriteria(D_L) **then**
    FindPrediction(D_L)
else
    BuildSubtree(n_L, D_L)
end if

if StoppingCriteria(D_R) **then**
    FindPrediction(D_R)
else
    BuildSubtree(n_R, D_R)
end if

Information Gain

- Information gain is defined as:

  $IG(Y|X) = H(Y) - H(Y|X)$

- The difference between the uncertainty of Y when we don't know X and when we know X.
- This is the information about Y that we gained from having known X.

Find the best split

- Intuitively, the best split will send all the 'yes' samples to one side and 'no' samples to the other side.
- To determine the best split, we consider how much information about a given feature x tells us about the value of y.
- The best split involves the feature and associated threshold that has the most 'information gain'.

Stopping Criteria

- When to decide not to make a branch?

1. When the leaf is pure, i.e. the variance of Y is small
2. When the number of samples in the leaf is too small

Information Gain

- Going through feature xi, find IG(Y |xi), pick the feature with the highest IG to make a split!
- For example, to predict who will live the longest, we compute the following IG
  - IG(LongLife | HairColor) = 0.0001
  - IG(LongLife | Gender) = 0.25
  - IG(LongLife | Smoke) = 0.1
  - IG(LongLife | NSiblings) = 0.005
- Then you'll choose to Gender as the feature you'd like to use for the split.

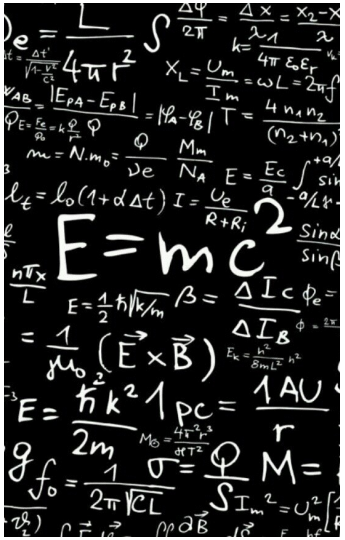
- Instance-based Learning
- K-Nearest Neighbor

INSTANCE-BASED LEARNING

Making Prediction

- For a classification problem:
- Predict most common y of the examples in the leaf.

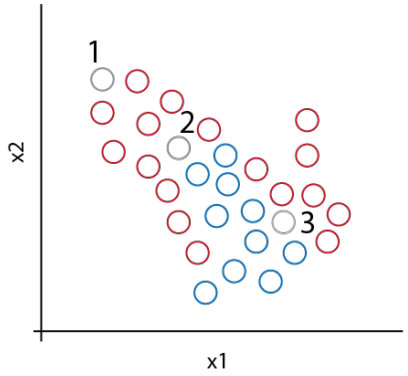
- For a regression problem:
- Predict the average y of the examples in the leaf or build a linear regression model on the examples in the leaf.



INTUITION QUIZ

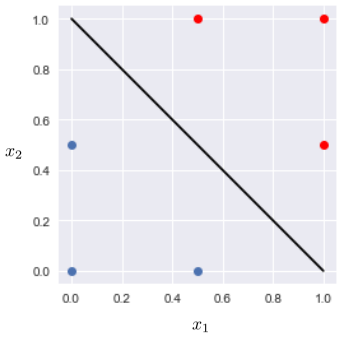
Intuition Quiz

Would you classify point 1, 2, 3 as blue or red. Fill in the table.



Pt	BLUE or RED
1	
2	
3	

Logistic Regression Model



model: $h(x) = f(\theta_0 + \theta_1 x_1 + \theta_2 x_2) = f(z)$

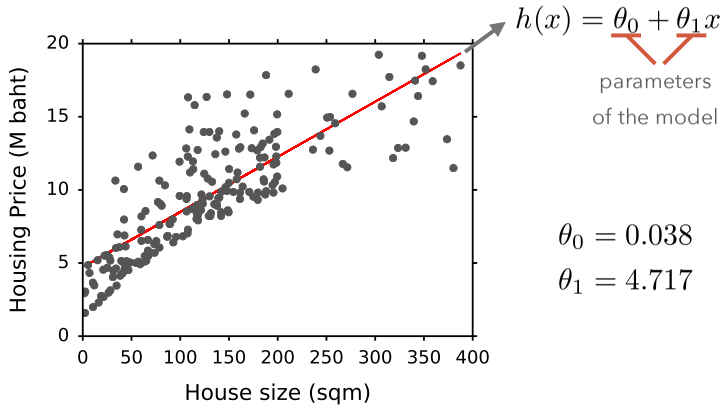
parameter: $\theta_0 = -1$

$\theta_1 = 1$

$\theta_2 = 1$

x1	x2	z	y'
1	1		
0	0		
0.5	0		
0	0.5		

Linear Regression Model



$h(x) = \theta_0 + \theta_1 x$

parameters of the model

$\theta_0 = 0.038$

$\theta_1 = 4.717$

Instance-based Learning

- Lazy algorithm: when you see your training set, you do nothing, just store them in the memory.
- When new sample comes you compare the new sample with the existing samples in the memory.
- Examples of algorithms in this family: nearest neighbor, kernel machines.

IBL: How Decision is Made

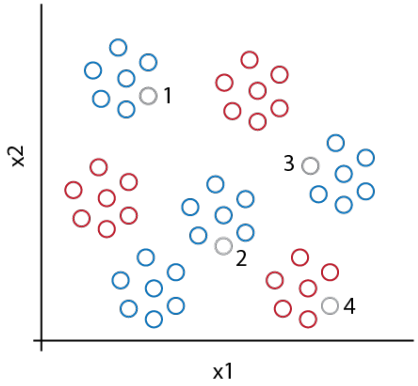
- Your source of knowledge is the similarity between two different data points. So you use similarity to make decisions such as classification and regression.
- You make decisions about one data point based on neighboring points.

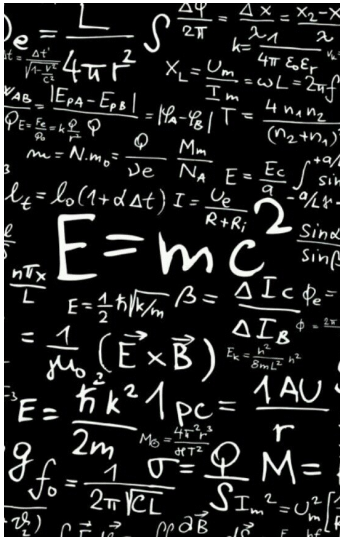
IBL - K-Nearest Neighbor Methods

- **K-Nearest neighbor:** locate k nearest neighbors around x′.
  - For classification problem, let k neighbors vote for the right label of x′.
  - For regression problem, average the y values of all neighbors and predict that y as the label of x′.

IBL - Nearest Neighbor Methods

- **Nearest neighbor:** when you see a new data point (x′), locate the nearest data point (x) and predict the label of x′ to be the same as label of x.

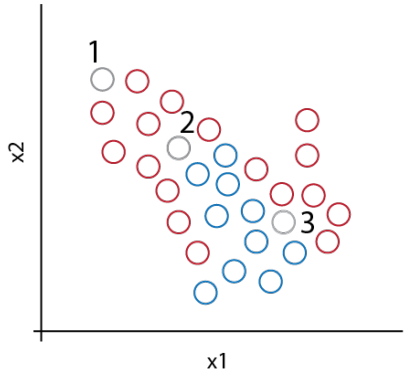




K-NEAREST NEIGHBOR QUIZ

IBL - K-Nearest Neighbor Quiz

Use K-Nearest Neighbor Rule to classify point 1, 2, and 3 with different values of k.



Pt	k=1	k=2	k=3	k=4
1				
2				
3				

Distance and Similarity Metrics

- To determine whether two points are close, we use distance metrics.
- **Distance metrics** are the numerical value that tells you whether two points are close (low value) or far apart (high value).
- There are several ways to define distance metrics, such as euclidean distance, minkowski distance.
- **Similarity metrics** are the numerical value that tells you whether two points are close (very similar - high value) or far apart (very dissimilar - low value).
- Distance and similarity metrics are important in many ML models such as 'Support Vector Machine', 'K-Nearest Neighbor', 'K-Mean Clustering'

Pros and Cons

- Pros:
  - Training takes no time
  - Complex decision boundary is possible
  - Information is not lost
- Cons:
  - Query is slow (the more data the slower)
  - Storage space is huge
  - Easily fooled by irrelevant attributes

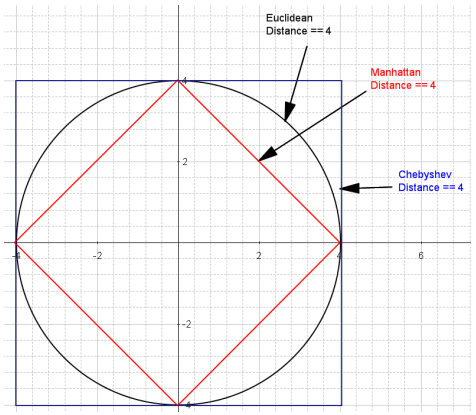
Distance Metrics for Real Value Features

- Euclidean Distance

  $\text{sqrt}(\text{sum}((x - x')^2))$

- Manhattan Distance

  $\text{sum}(|x - x'|)$



Distance Metrics for Boolean Features

- Jaccard Distance

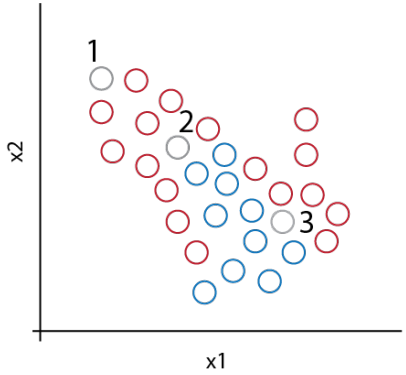
Feature	Me	My Dad
Man Barber	F	T
Toyota	T	T
MK	T	T
Water Park	T	F
Temple	F	T
Bar	F	F

N=6
NTT=2
NTF=1
NFT=2
NFF=1
NNEQ : number of non-equal dimensions
NNEQ = NTF + NFT = 3
NNZ : number of nonzero dimensions
NNZ = NTF + NFT + NTT = 5

NNEQ / NNZ = 3/5 = 0.6

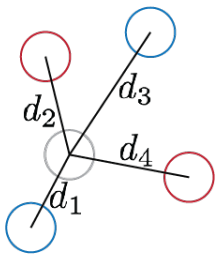
How to Avoid Overfitting

- Use k as an overfitting control.
  - If k is one, you are very susceptible to noise (overfitting).
  - If k is high, you are averaging over really large regions, you lose resolution (under fitting).



Using Distances as Weights

- **Neighbors who are closer to the target data point should get more say in the voting process.**



$$y' = \frac{w_1 y_1 + w_2 y_2 + w_3 y_3 + w_4 y_4}{w_1 + w_2 + w_3 + w_4}$$

$$w_i = \frac{1}{d_i}$$

- Unsupervised Learning
- K-Means Algorithm
- Clustering Performance

K-MEANS CLUSTERING

How to Avoid Overfitting

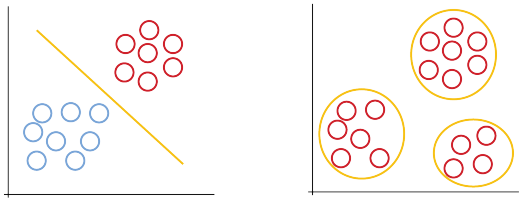
- Remove noisy instances prior to using nearest neighbor algorithm. Remove x if all nearest neighbors of x are in the opposite class.
- Form prototypes. If you observe lots and lots of very similar samples, lump them into a prototype by finding an average over all dimensions.

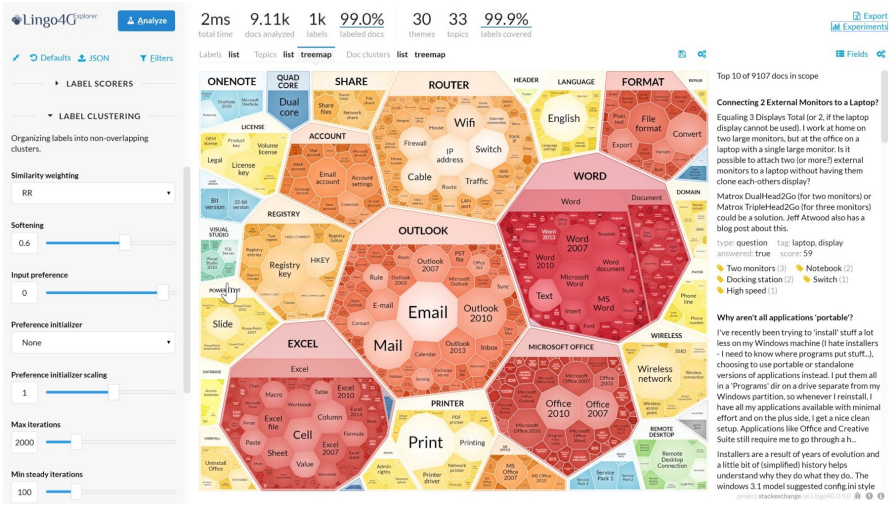
Unsupervised Learning

54

In **supervised learning,** your data come with labels indicating what class corresponds to each sample.

Sometimes, data do not come with categorical labels, but you can tell that there is a grouping structure.

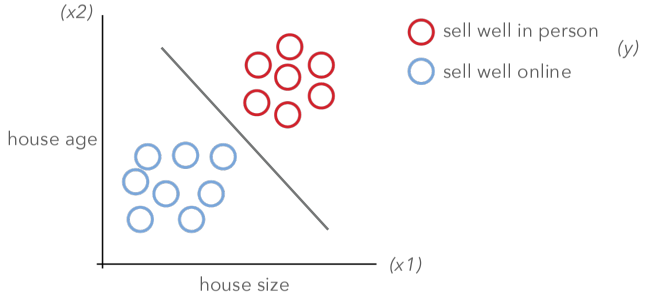




Supervised vs. Unsupervised Learning

53

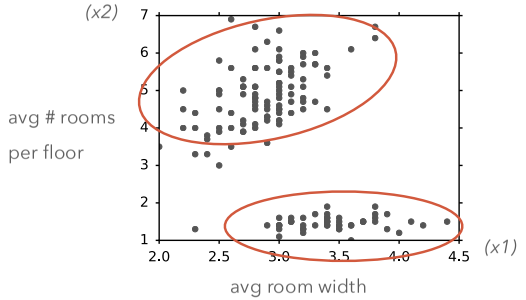
- **Classification:** predicting **discrete** output from input
  - Example: predicting whether a house would sell well in person or online



Unsupervised Learning

55

- **Clustering:** grouping unlabeled input by similarities
  - Example: grouping houses in the database



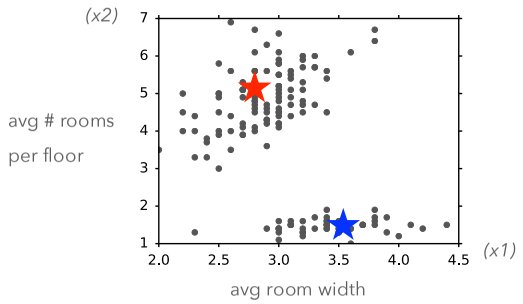
Supervised vs. Unsupervised Learning

- Supervised learning problem: "given all history of feature data points (X) and their labels (Y), find a model that computes label (y') for a given sample data point (x'). The prediction must minimize a cost function (minimize errors)."
- Clustering problem: "given all feature data points (X), find their labels (Y). The label Y must minimize a given objective function."

	X	Y	x'	y'	Predict	Cost Function
Supervised	Yes	Yes	Yes	No	y'	Error
Unsupervised	Yes	No	Yes	No	Y, y'	Objective F.

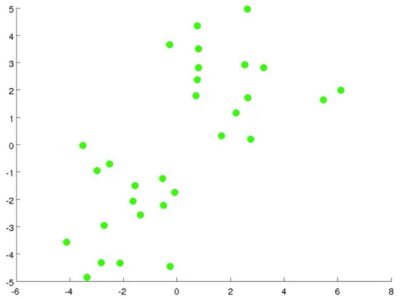
K-Means Clustering

- K-means clustering is the most popular clustering algorithm.
- K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.



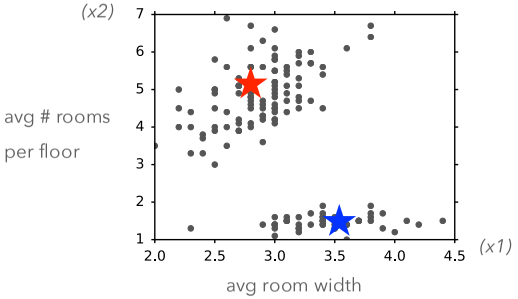
K-Means Algorithm

First, we have a bunch of unlabeled data points. We decide that we are going to find two clusters in this data.



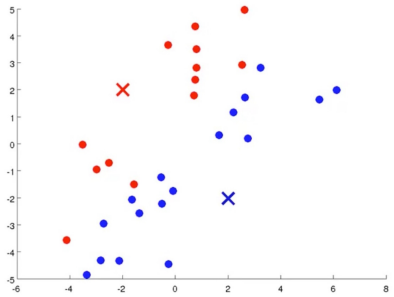
K-Means Clustering

- The red and blue stars are called "centroids".
- The red star is the centroid of top cluster.
- The blue star is the centroid of the bottom cluster.
- The location of the centroid is at the mean of all points in the cluster.



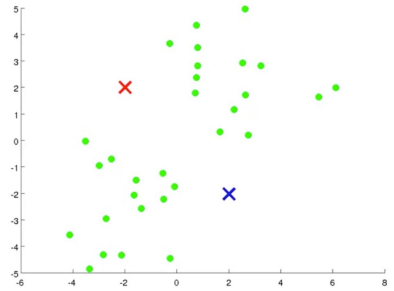
K-Means Algorithm

Then, we do cluster assignment step, where we go through every sample and determine whether each dot is closer to red or blue centroid. Label the sample to red/blue accordingly.



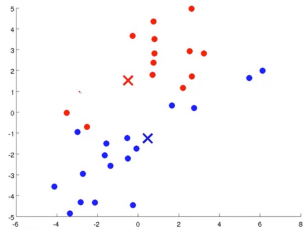
K-Means Algorithm

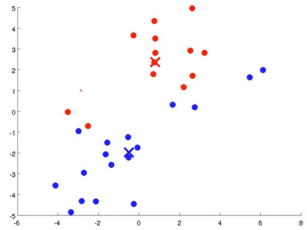
The first step is to pick two random locations to be our cluster centroids.



K-Means Algorithm

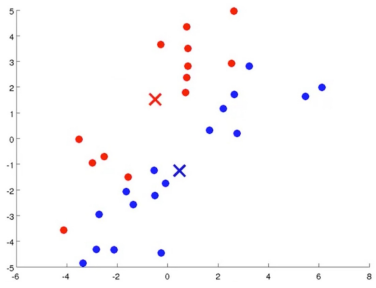
Repeat the cluster assignment step and centroid movement step, alternately.





K-Means Algorithm

After the assignment step, we do centroid movement step where we move the red and blue centroids to the means of our clusters.

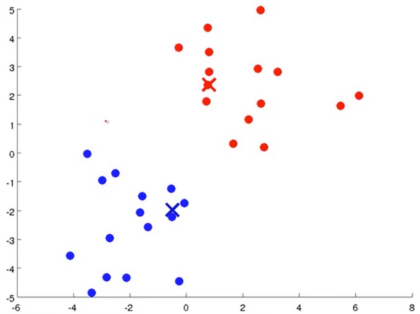


K-Means Algorithm

- To summarize, k-means algorithm has two steps:
  - In cluster assignment step, we fixed the centroids and label each data point to belong to the nearest cluster centroid.
  - In centroid movement step, we fixed data point labels and move each centroid to the mean of its data points.
- Iterating over these two steps and you will achieve the best clusters.

K-Means Algorithm

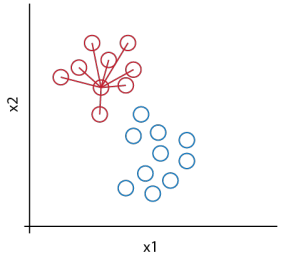
The algorithm converges to the solution when cluster centroids are not changed anymore.



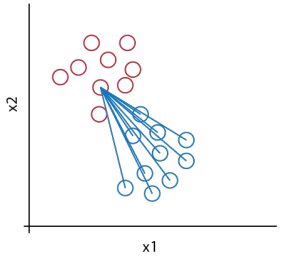
Clustering Performance

- How good is your clustering results? It is not as simple as counting right and wrong predictions. Performance of clustering algorithm is often judged based on how well you algorithm separates out the data into several clusters.

Cohesion

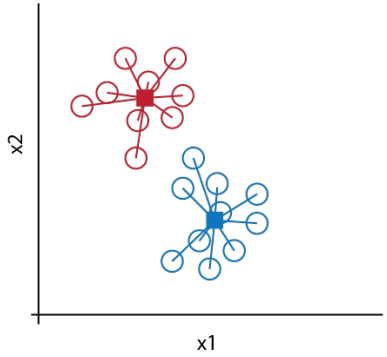


Separation



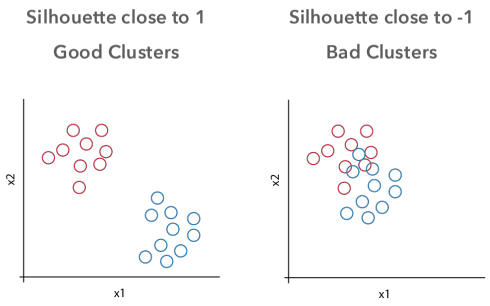
K-Means Algorithm

- What K-Means algorithm is doing?
- The algorithm minimize the total distances between all data points and the centroids of the clusters they belong to.



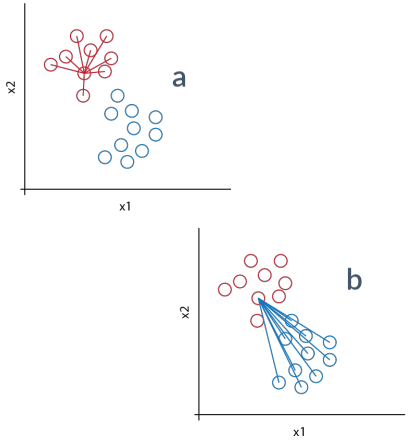
Clustering Performance

- Silhouette Coefficient = 1- (a/b)
- If b>>a, SC is close to 1.
- If a<<b, SC is close to -1.



Clustering Performance

- Silhouette Coefficient is one such measure. It's defined by two separate scores:
  - a: mean distance between a sample and all other points in the same class.
  - b: mean distance between a sample and all other points in the next nearest cluster.
- Silhouette Coefficient
  = (b-a)/max(b,a)
  = 1- (a/b) - if b>a



Clustering Performance

- Other methods of clustering performance include Calinski-Harabaz Index (dispersion from centroid), percentage of explained variance (between-group variance / total variance), and many others.
- We can calculate these performance metrics using any distance metrics we learned about (Euclidean, Manhattan, Jaccard).
- Typically, it would be best if we could find some ground truth to validate our clusters. For example, if we cluster 1M news articles, we could get people to label 5000 news articles to establish ground truth for validating our clusters.

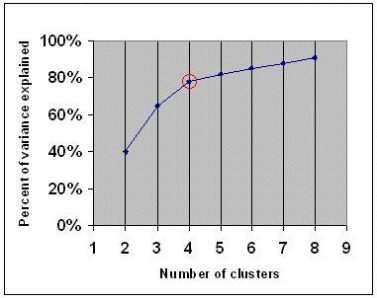
Clustering Performance

- When ground-truth labels are provided, people talk about clustering performance in terms of homogeneity, completeness, and v-measure
  - Homogeneity: each cluster contains only members of a single class.
  - Completeness: all members of a given class are assigned to the same cluster.
  - Both homogeneity and completeness are bounded below by 0.0 and above by 1.0 (higher is better)
  - V-Measure Score: the harmonic mean between homogeneity and completeness.

v = 2 * (homogeneity * completeness) / (homogeneity + completeness)

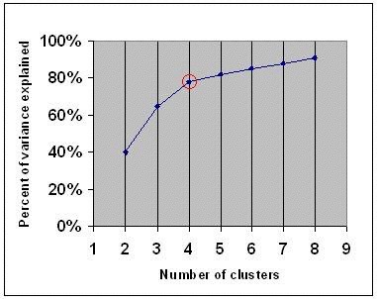
Choosing the Right K

- Pick the k at the elbow point. At this point, more clusters do not necessarily mean higher performance.
- Similarly we use metrics like Information Criterion (the famous one being Bayesian - BIC, and Akaike - AIC). These are measures that find a good balance between the variance explained and the complexity of the model.



Choosing the Right K

- How to determine the number of clusters in the dataset?
- The most common way is called elbow methods.
- You compute performance metrics such as Silhouette Coefficient or Percentage of Variance (in the figure), while varying k.



Choosing the Right K

- Pick the k at the elbow point. At this point, more clusters do not necessarily mean higher performance.
- Similarly we use metrics like Information Criterion (the famous one being Bayesian - BIC, and Akaike - AIC). These are measures that find a good balance between the variance explained and the complexity of the model.