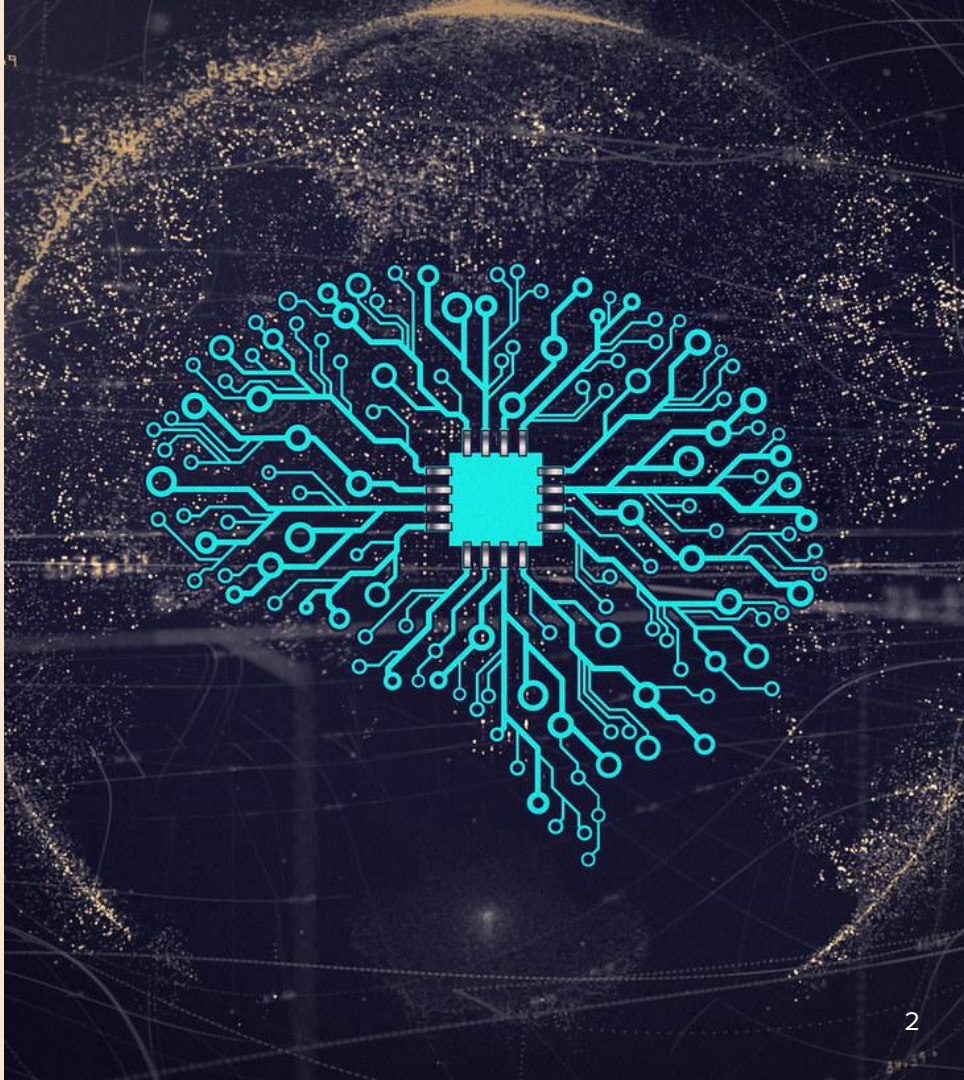
Decision Tree & K-Nearest Neighbors

Dr. Paisit Khanarsa

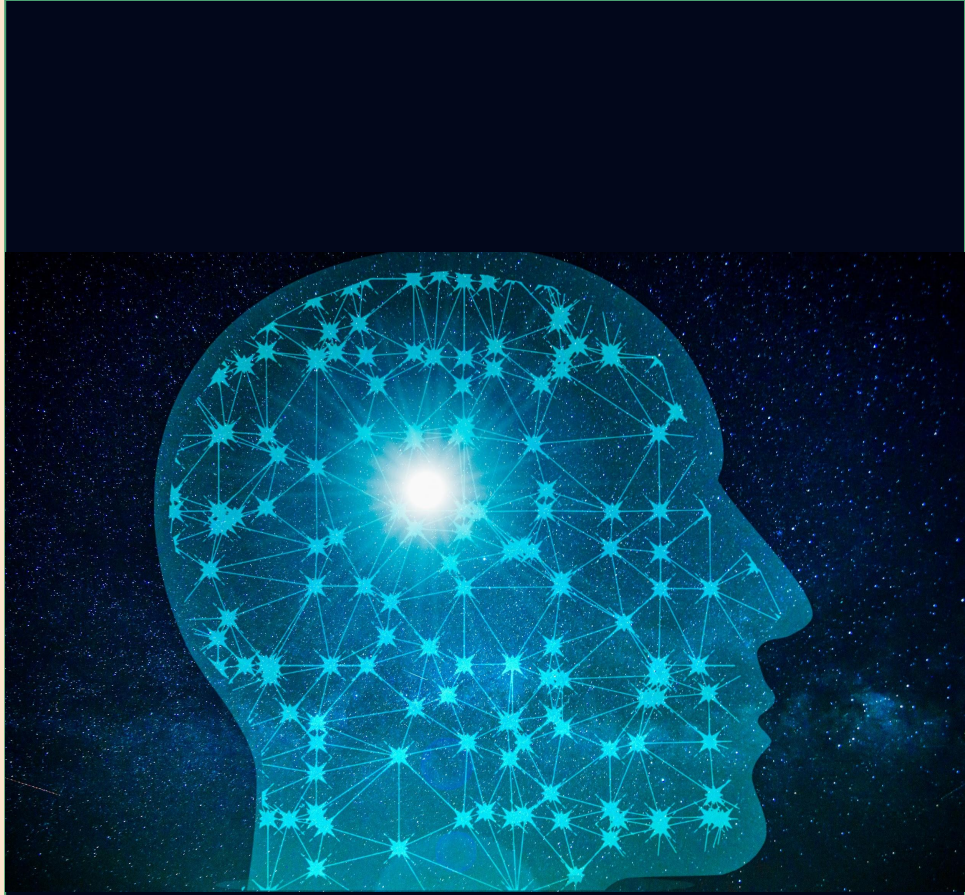
FIBO, Kmutt

Outline

- Decision tree
- Information gain
- Random forest
- Instance-based learning
- K-nearest neighbors model



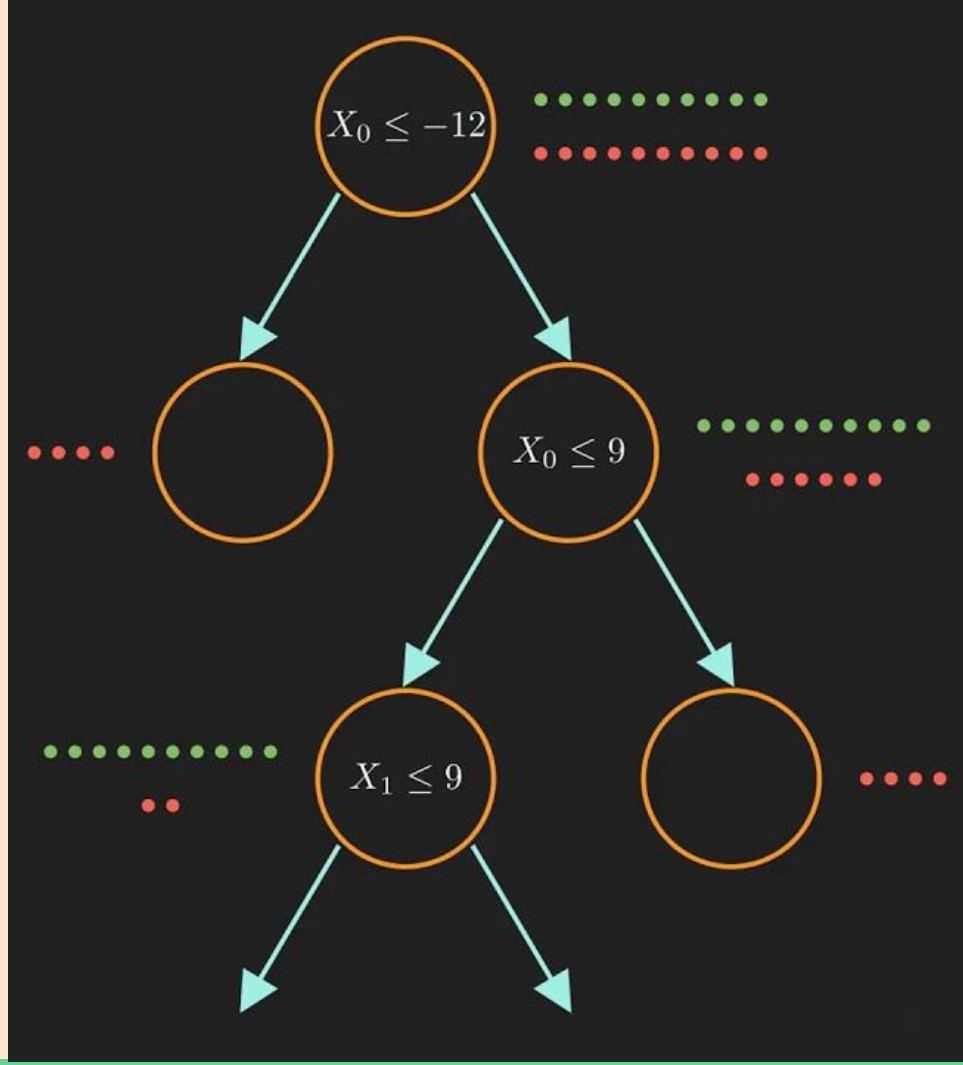
Decision Tree



Decision tree

Decision tree is a very old and simple idea. Today, it is perhaps the most popular machine learning algorithms due to following reasons.

- Easy to understand
- Easy to implement (not a lot of parameters to tweak)
- Can be used for both classification and regression models



Decision tree

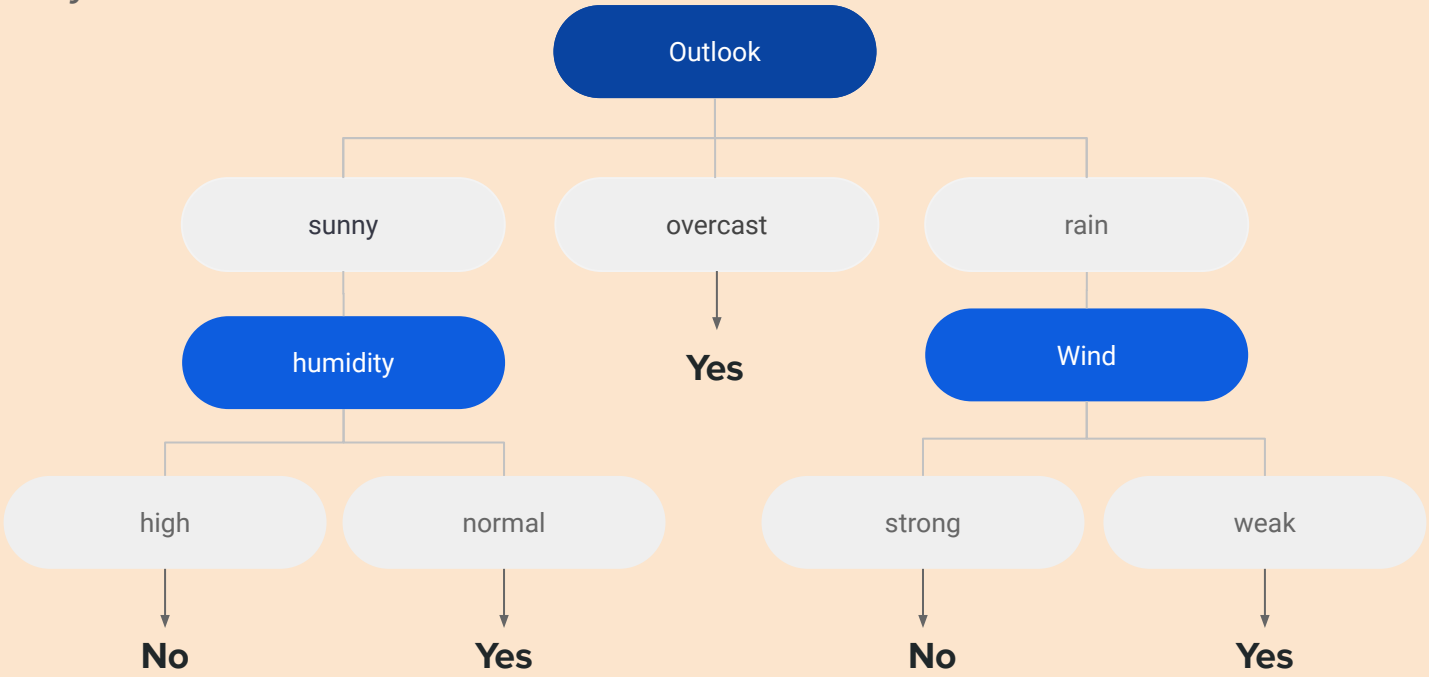
Here is an example of a decision tree for classification. Is John going to play tennis today ?

John's data

Sample	is_sunny	is_overcast	is_rain	is_humidity_high	is_wind_strong	John
1	1	0	0	1	1	no
2	0	1	0	0	1	yes
3	0	1	0	0	0	no
...	...	...	...	...	...	...

Decision tree

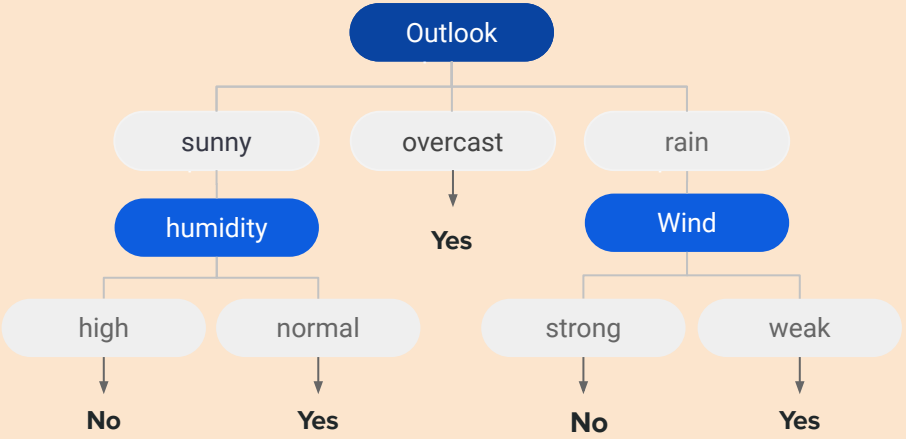
Here is an example of a decision tree for classification. Is John going to play tennis today ?



Decision tree

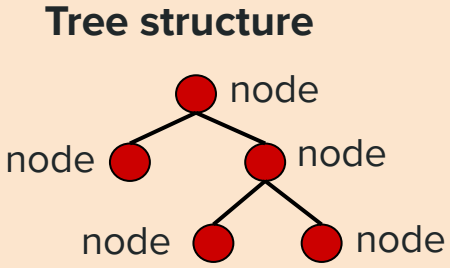
The y variable is {yes,no}, There are 4 features:

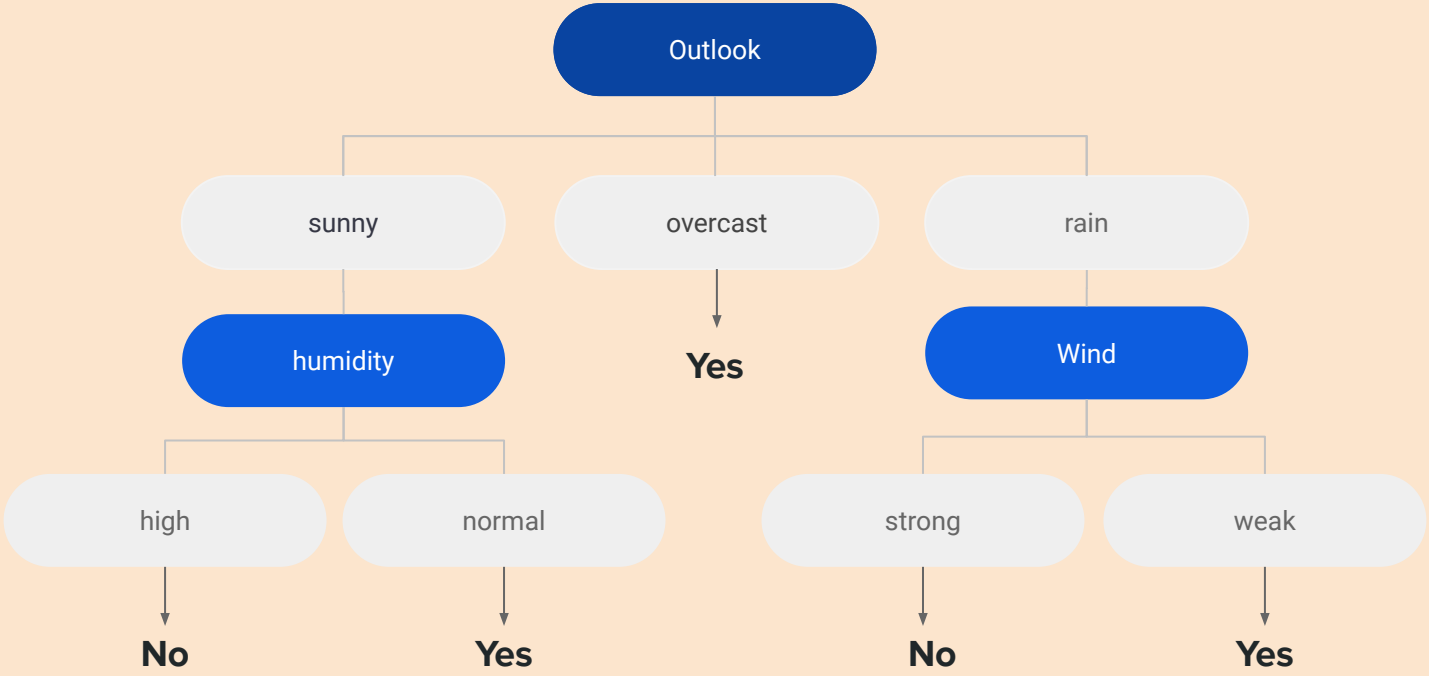
- x_1 = Outlook {sunny, overcast, rain}
- x_2 = Temperature {hot, warm, cold}
- x_3 = Humidity {high, normal}
- x_4 = Wind {strong, weak}



Decision tree

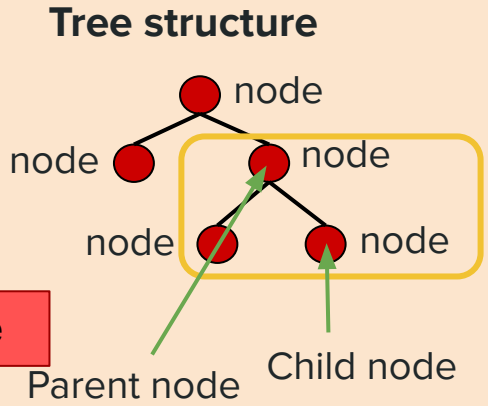
Root node, Internal node, Leaf node

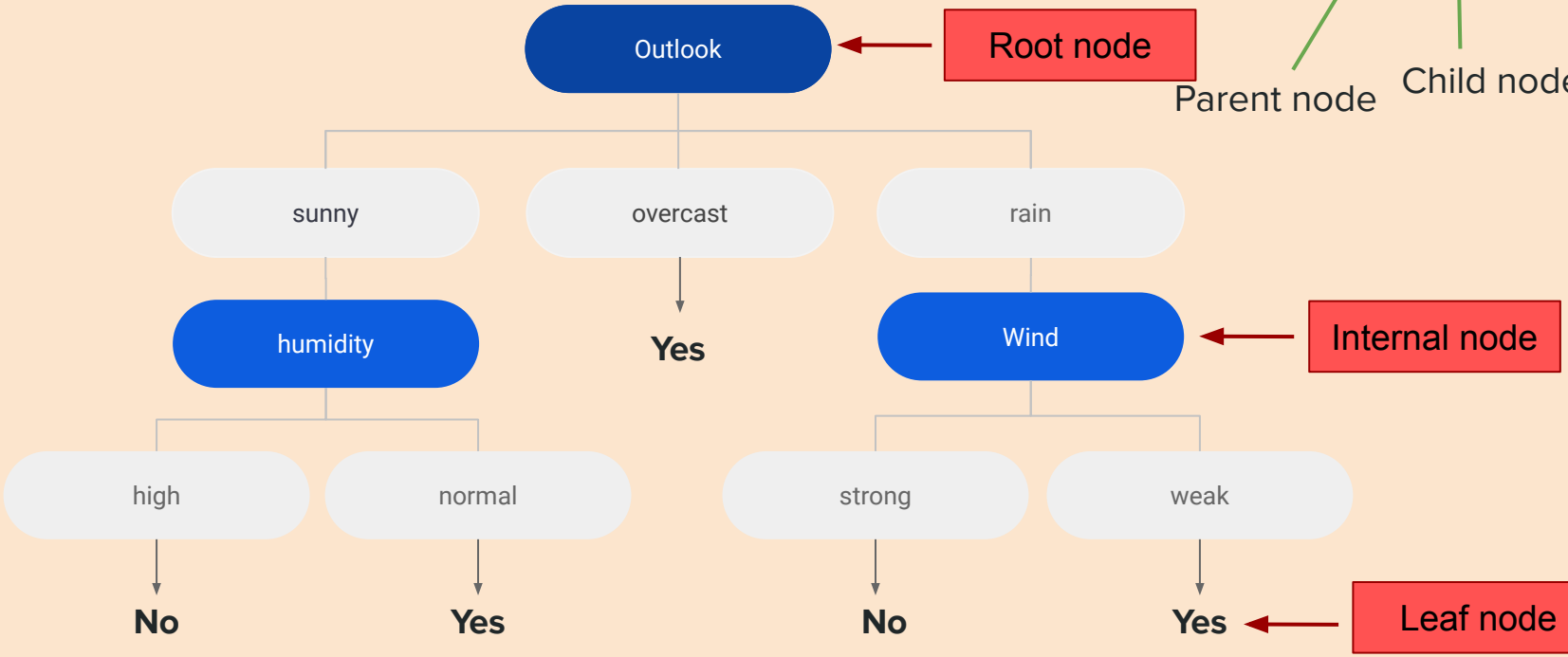




Decision tree

Root node, Internal node, Leaf node



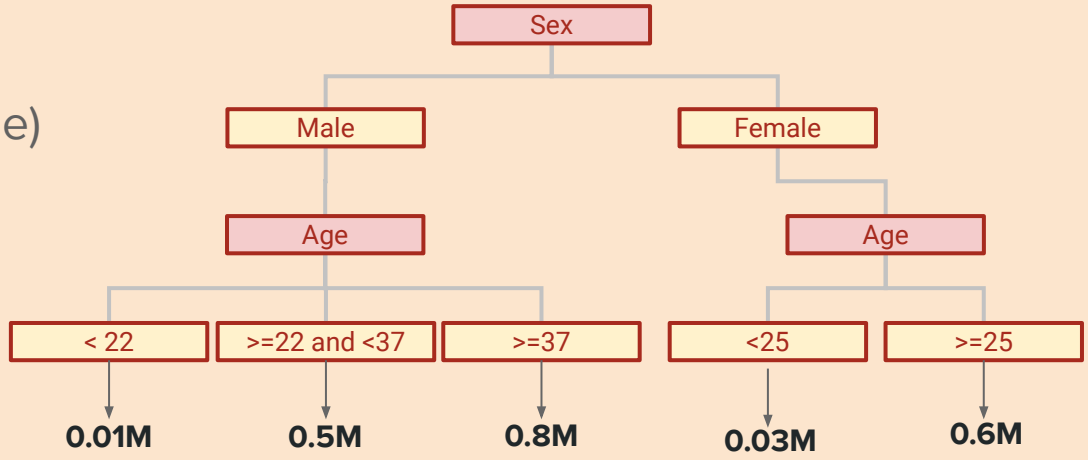


Decision tree & Continuous Variable

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)

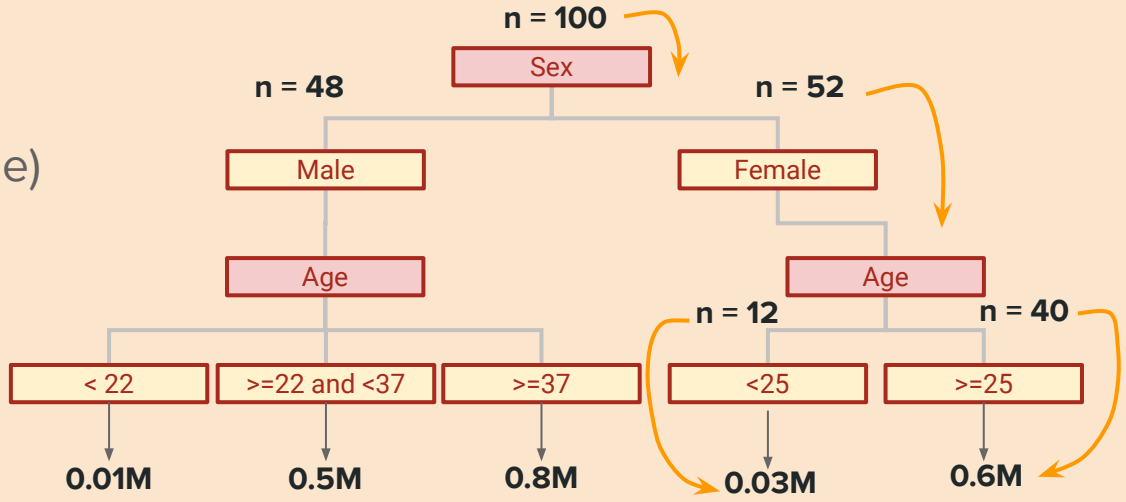


Decision tree & Continuous Variable

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

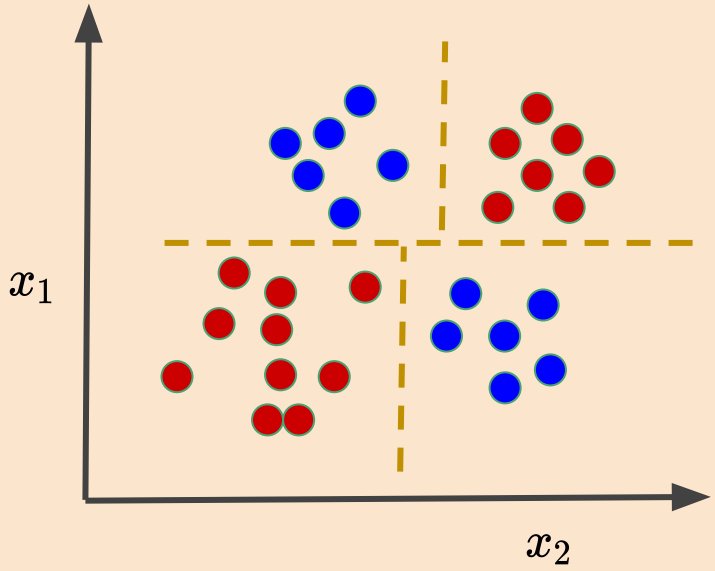
Predict amount of money in saving account of single people (y) from the following attributes:

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

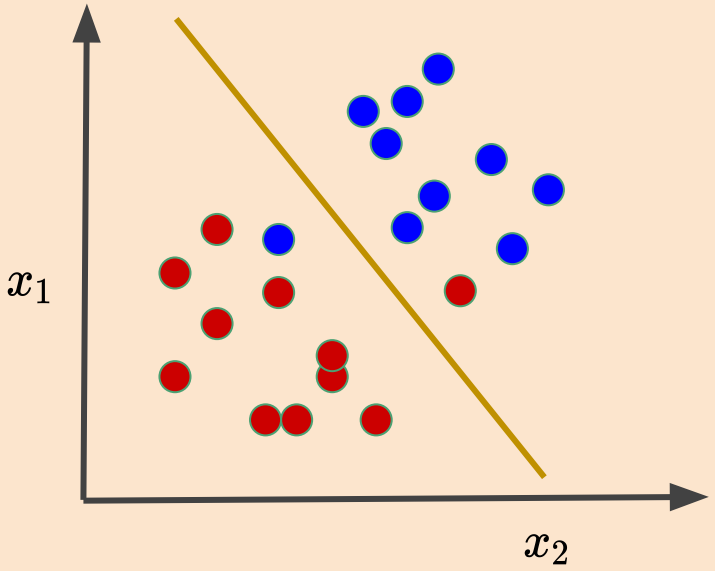


Decision boundary

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



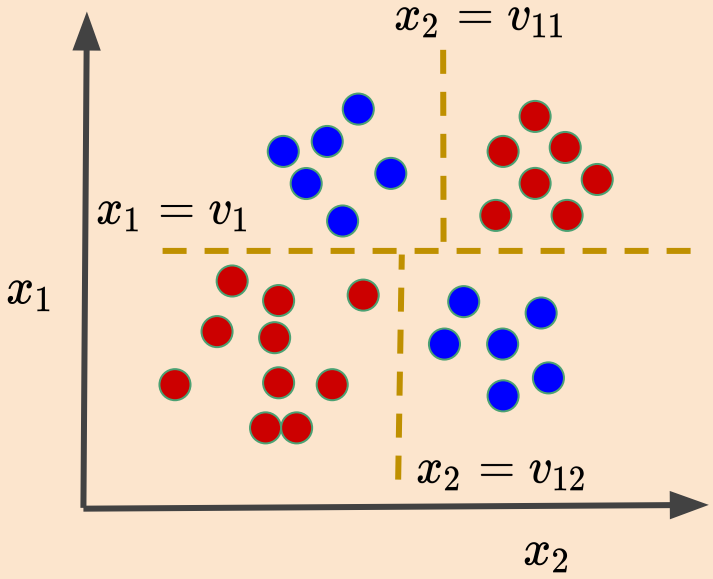
Decision tree

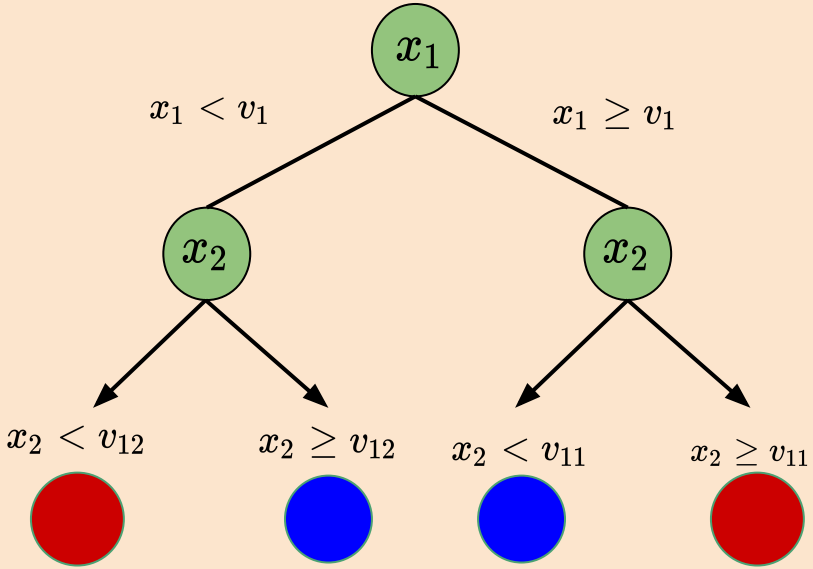


Logistic regression

Decision boundary

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



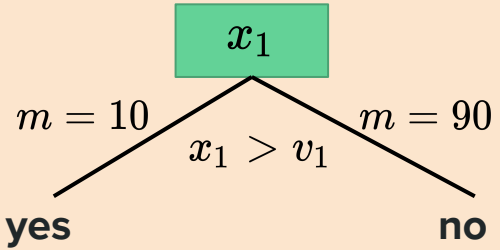


Decision tree

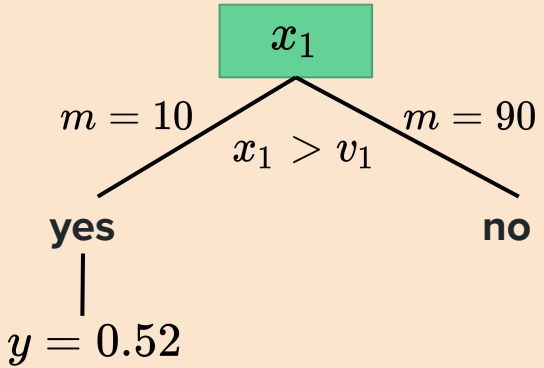
Constructing a decision tree

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



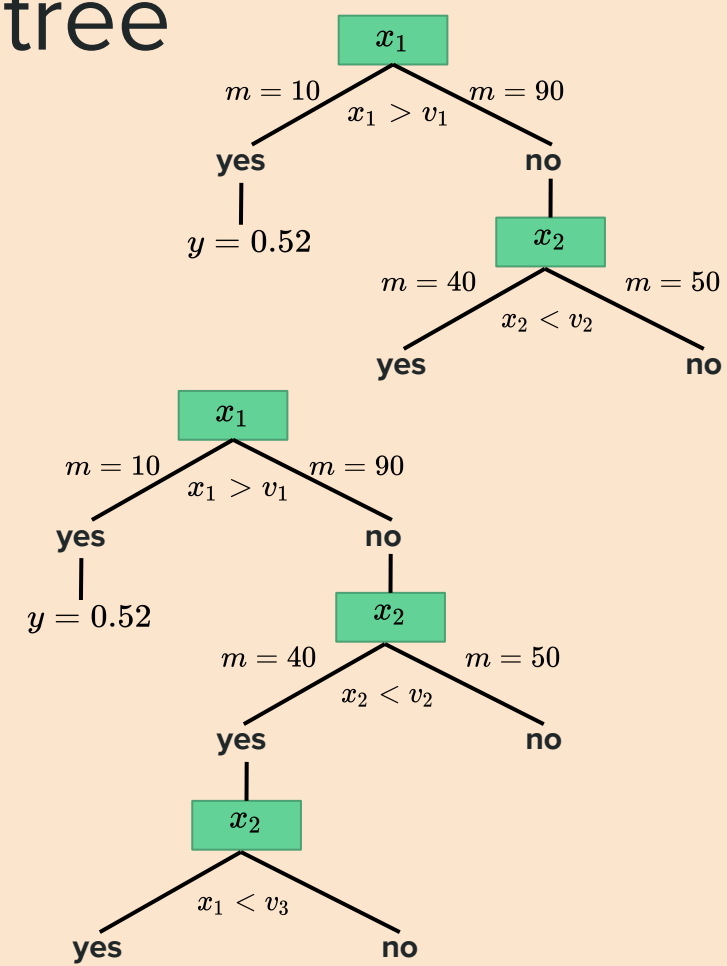
- Decide whether to add another internal node. If there is no use in creating another internal node, just add a leaf node and a prediction.



Constructing a decision tree

- On the other hand, you may decide to add another internal node which focuses on another feature

- Repeat process until we account for all samples



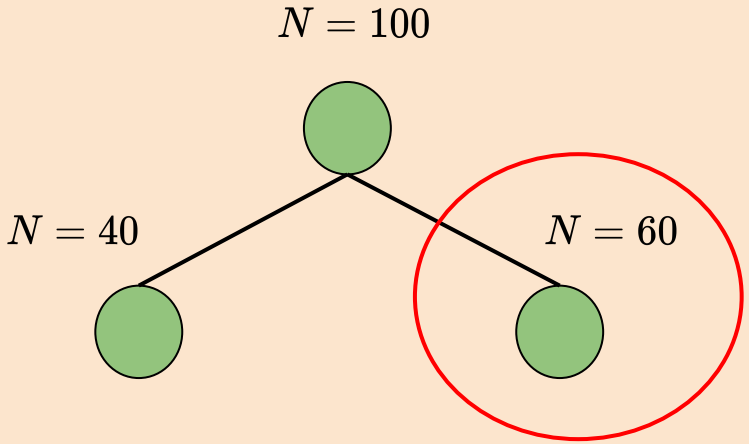
Constructing a decision tree

- Intuitively, the best split will send all the ‘yes’ samples to one side and ‘no’ samples to the other side.
    - **Choose feature:** first determine which feature give us the best split. We consider how much information about a given feature x tells us about the value of y **(information gain)**.
    - **Choose threshold:** if that feature is a continuous feature, determine the threshold to split. We consider the threshold that will maximize variance decrease.

Stopping criteria

- When to decide not to make a branch?

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

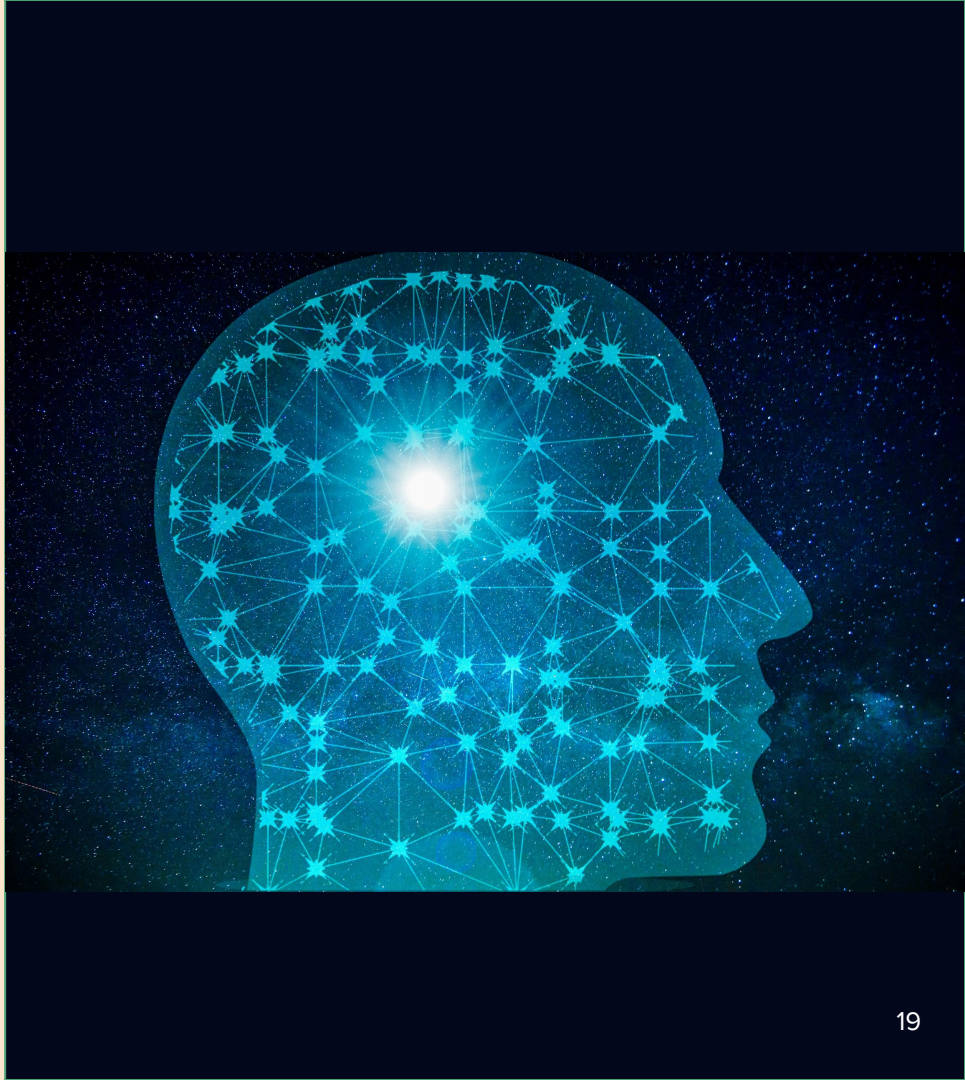


- Internal ?
- Leaf ?

Making prediction

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

Information Gain



Information theory

- Suppose we have a variable Y which could only be 0 or 1
- Y has the following probability

$$P(Y=0) = 0.2$$
$$P(Y=1) = 0.8$$

Binomial Distribution

- We may define ‘surprise’ variable S, where

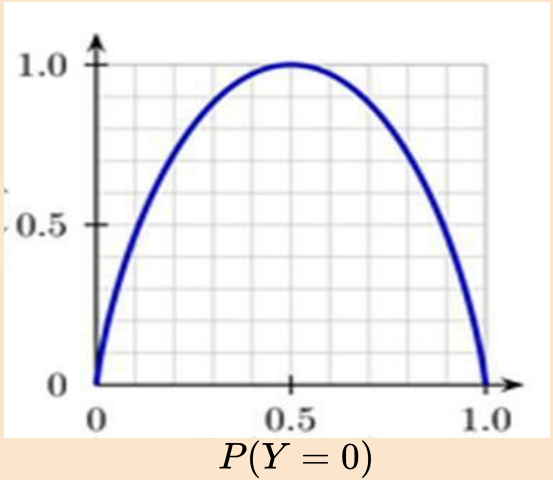
$$S(Y=c) = -log_2(P(Y=c))$$

- If Y = 0, S = 2.32 ➔ we are very surprise
- If Y = 1, S = 0.32 ➔ we are not surprise

Entropy

- Entropy is the sum of the product of the surprise and the probability of the outcome
- This is the average surprise yielded by single occurrence of Y or the uncertainty of Y.

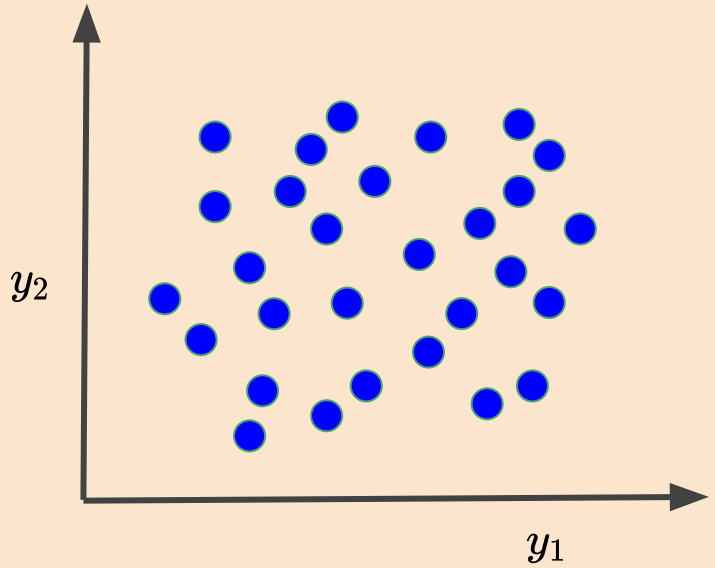
$$H(Y) = \sum_{c=0,1} [-P(Y = c) \times log(P(Y = c))]$$

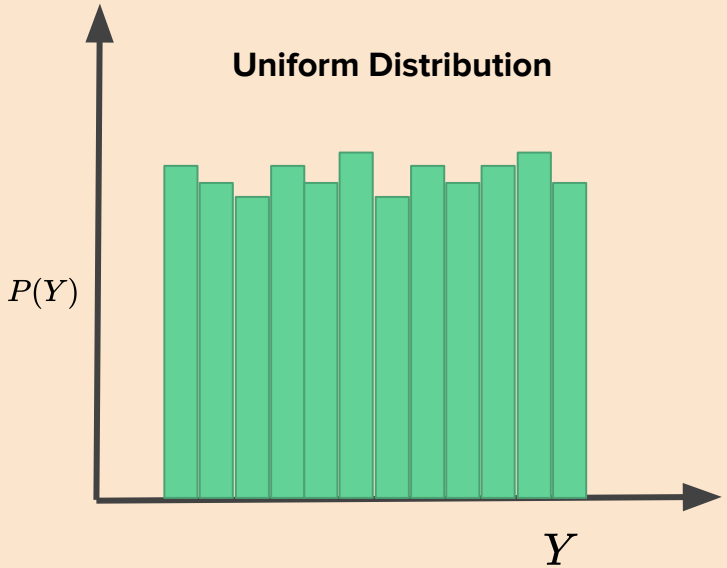


Entropy

- For continuous variable,

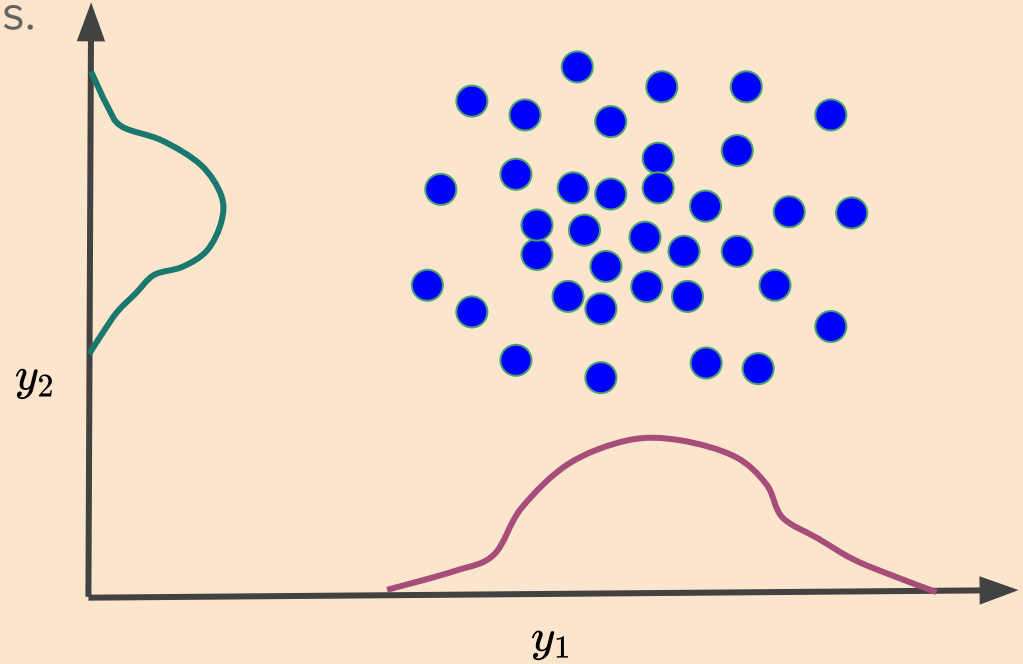
  - ‘High’ entropy: the variable has uniform (boring) distribution. It is hard to guess what the variable is.





Entropy

- For continuous variable,
  - ‘Low’ entropy: the distribution has peaks and valleys, making it easy to guess what the variable is.



Conditional Entropy

- Suppose we want to predict Y from X:
  X = college major
  Y = like ‘Titanic’

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

Conditional Entropy

- Suppose we want to predict Y from X:
  X = college major
  Y = like ‘Titanic’

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

$$P(Y = Yes) = \frac{4}{8} = 0.5$$

$$P(Y = No) = \frac{4}{8} = 0.5$$

$$H(Y) = \sum_{c=Yes,No}[-P(Y = c) \times log(P(Y = c))]$$

$$= -P(Y = Yes)log(P(Y = Yes)) - P(Y = No)log(P(Y = No))$$

$$= -0.5log_2(0.5) - 0.5log_2(0.5) = 1$$

Conditional Entropy

- Define conditional entropy at $X = v$

$H(Y|X = v)$ the entropy of Y among all recoord where $X = v$

- Example

$$H(Y|X = Math) = \sum_{c=Yes,No}[-P(Y = c) \times log(P(Y = c))]$$
$$= -0.5log_2(0.5) - 0.5log_2(0.5) = 1$$

$$H(Y|X = History) = \sum_{c=Yes,No}[-P(Y = c) \times log(P(Y = c))]$$
$$= 0 \times log_2(0) - 1 \times log_2(1) = 0$$

$$H(Y|X = CS) = \sum_{c=Yes,No}[-P(Y = c) \times log(P(Y = c))]$$
$$= 1 \times log_2(1) - 0 \times log_2(0) = 0$$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

Conditional Entropy

Conditional entropy $H(Y|X)$ is the average conditional entropy at all values of X :

$$H(Y|X) = \sum_v P(X = v) \cdot H(Y|X = v)$$

- Example

v	$P(X = v)$	$H(Y|X = v)$
Math	0.5	1
History	0.25	0
CS	0.25	0

$$H(Y|X) = 0.5 \cdot 1 + 0.25 \cdot 0 + 0.25 \cdot 0 = 0.5$$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

Information gain

Information gain is defined as:

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

The difference between the uncertainty of Y when we do not know X and when we know X.

- Example

$$H(Y) = 1$$
$$H(Y|X) = 0.5$$
$$IG(Y|X) = 0.5$$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
Math	Yes
History	No

Find the best split

- Going through all x, find $IG(Y|X)$, pick the feature with the highest IG to make 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 will choose to Gender as the feature you would like to use the split.

Find the best split

- Choose a threshold: if X is a continuous variable, we must pick the best threshold to split. This is often done by maximizing the following function.

$$m \cdot Var(D) - (m_L \cdot Var(D_L) + m_R \cdot Var(D_R))$$

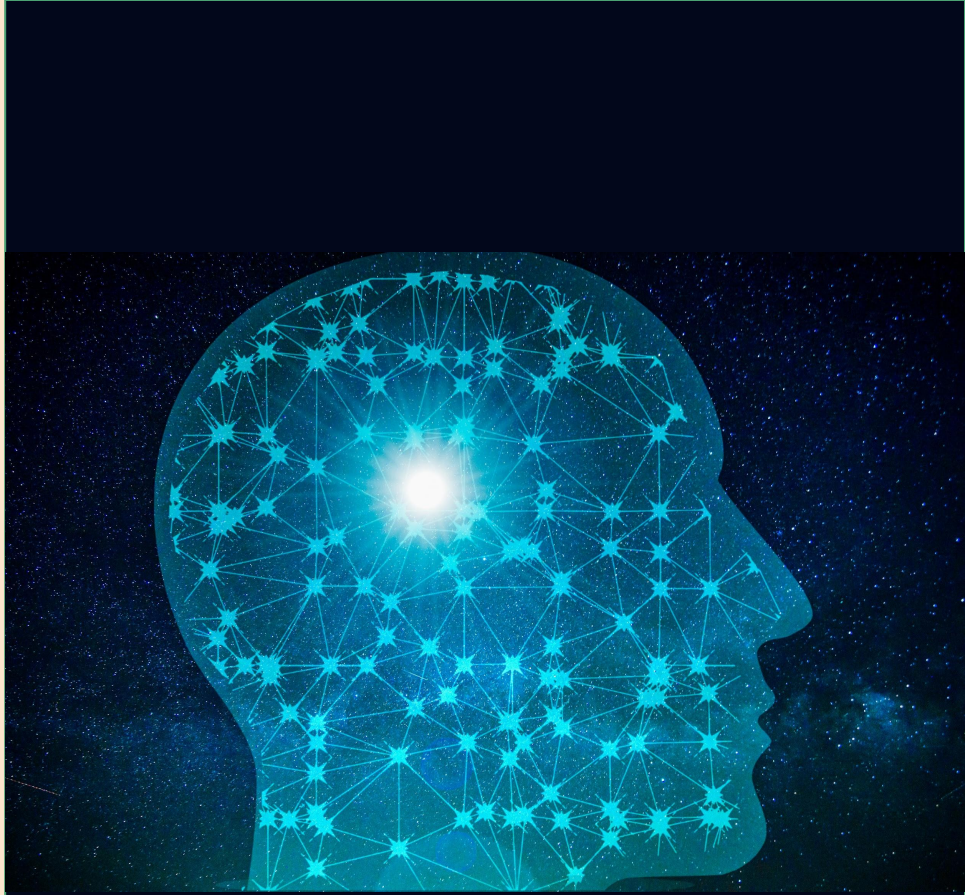
where m is # sample before split, m_L is #sample lift and m_R is#sample right

- This means the variance should decrease most substantially when we split.

Problem with decision tree

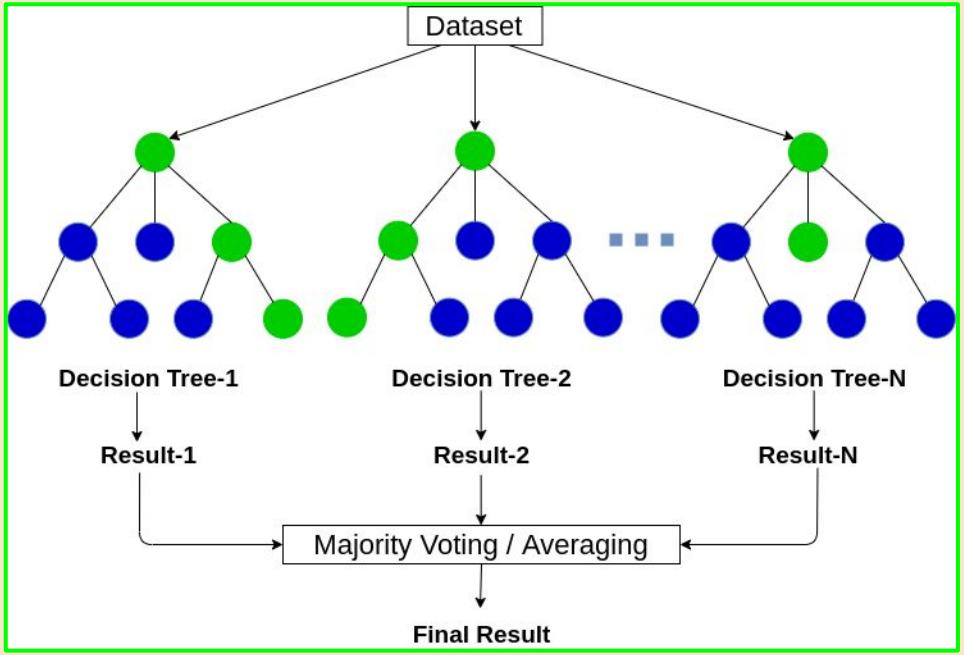
- Decision trees are prone to overfitting.
  - Use early stop criteria
  - Use ensemble method (i.e. random forest)

Random Forest



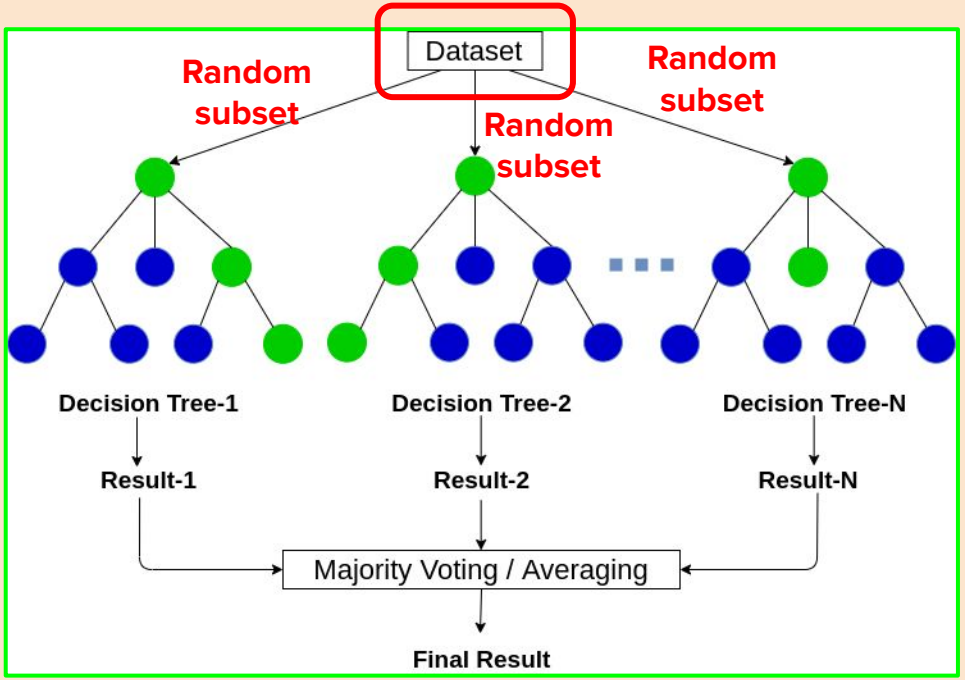
Random forest

- Random forest is an ensemble. The trees are weak learners and the random forest combines all weak learners to build a strong learner.
- Number of trees could be 10, 50, 500,..., N. The more trees, the less overfitted.



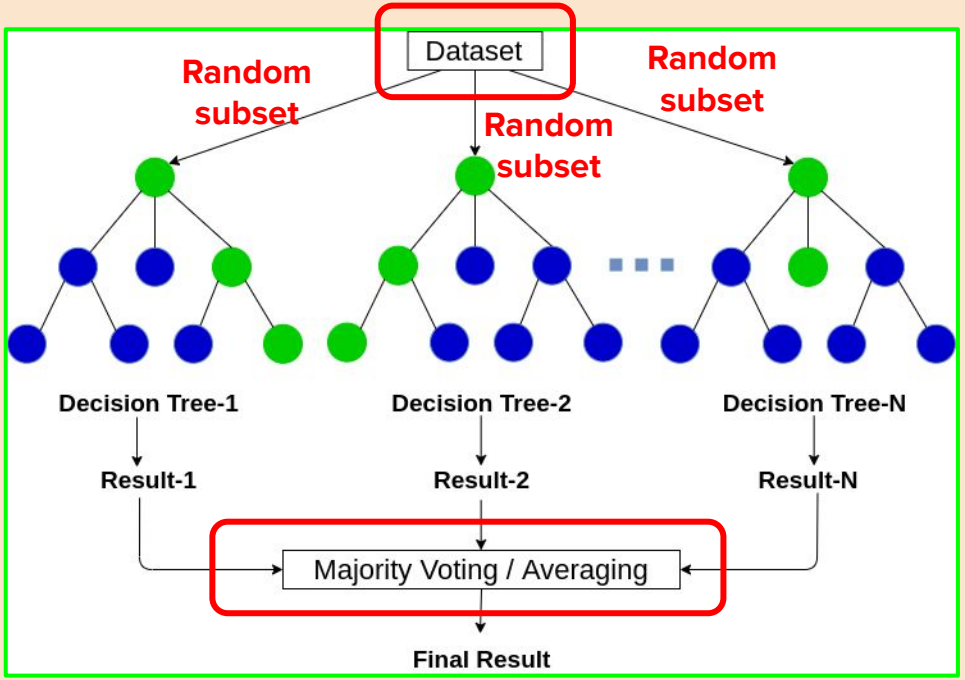
Random forest

- Each tree will receives subset of features and samples from all data. So that each tree comes up with uncorrelated model (each is equally right is setting up their rules).



Random forest

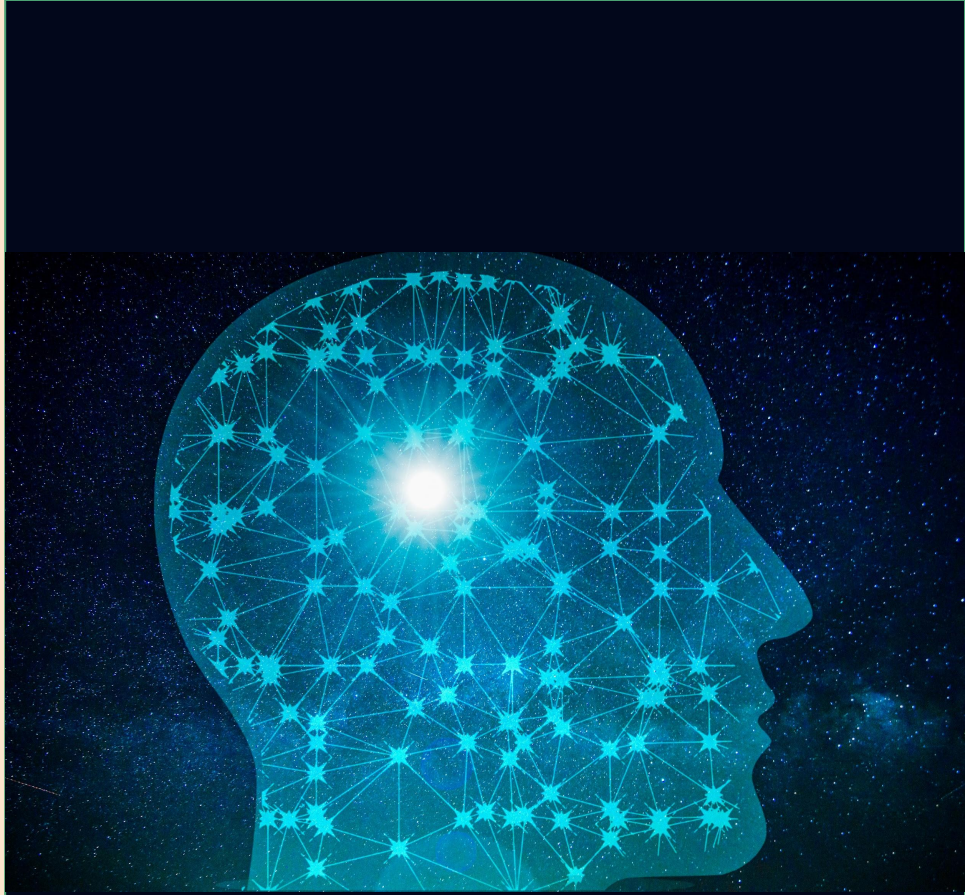
- The answer from all trees will be pooled to create the final answer. We can simply count votes for each categorical option.



Random forest

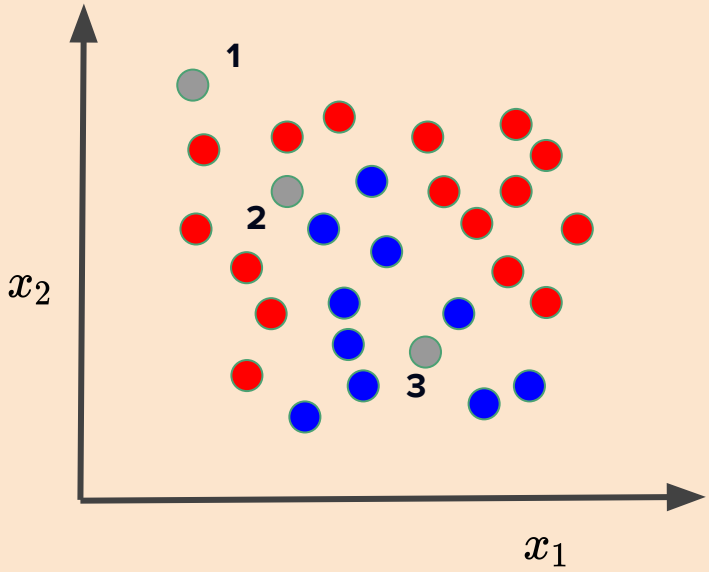
- Random forest is simple and fast to train. It has no requirement in terms of feature scaling.
- Works well with most tabular data
- Other variants include Gradient Boosted Tree, Extra Tree, which have additional features that make Random Forest fit data more accurately without overfitting

Instance-Based Learning (IBL)



Example

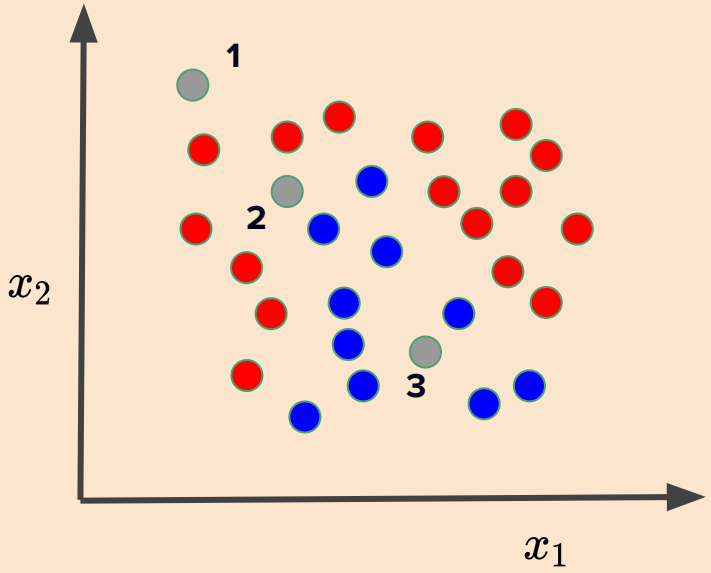
- Can you classify point 1, 2, 3 as blue or red ???



Point	Blue or Red
1	
2	
3	

Example

- Can you classify point 1, 2, 3 as blue or red ???

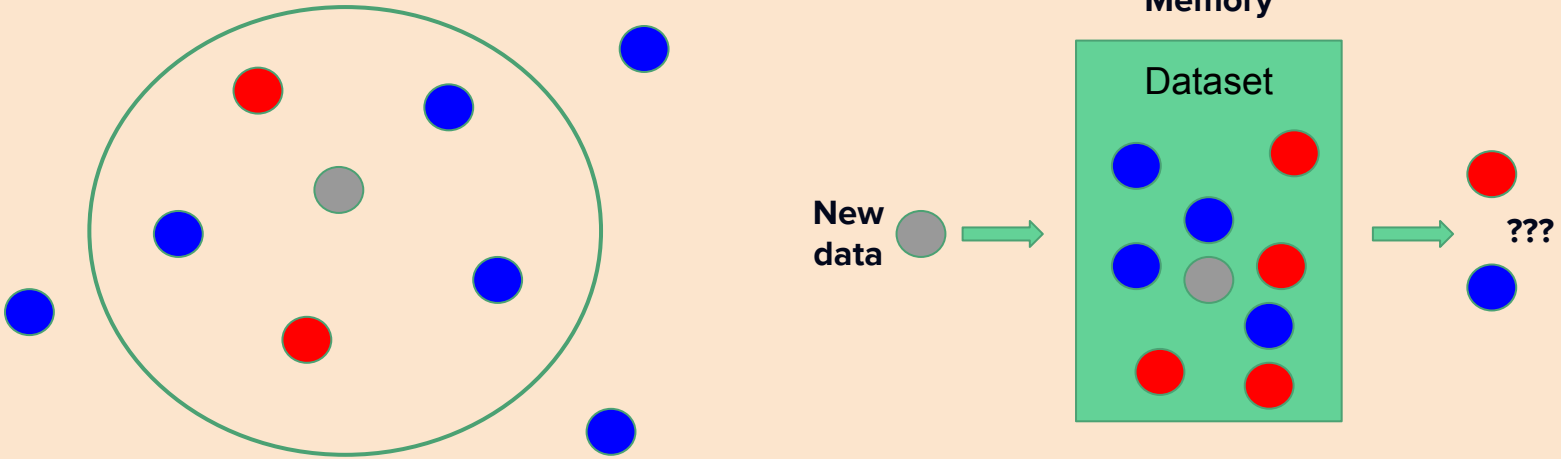


Point	Blue or Red
1	Red
2	???
3	Blue

Instance-based learning

- Your source of knowledge is the similarity between two different data points.

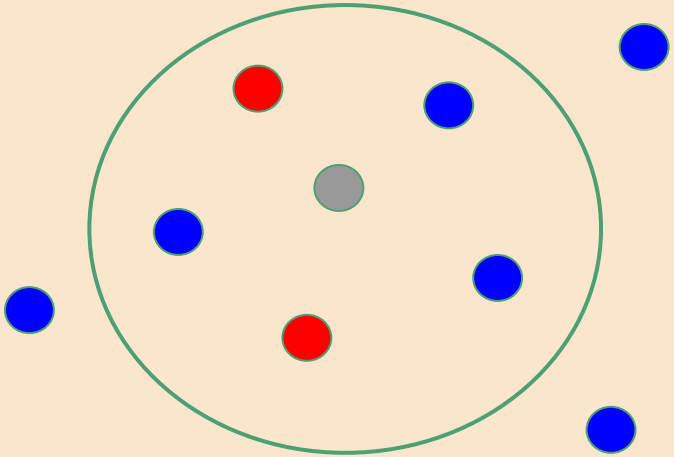
  So you use similarity to make decisions such as classification and regression
- You make decisions about one data point based in neighboring point.

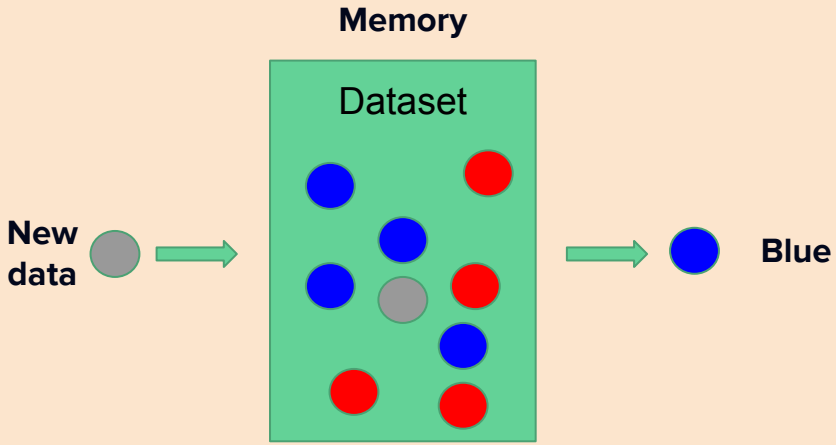


Instance-based learning

- Your source of knowledge is the similarity between two different data points.

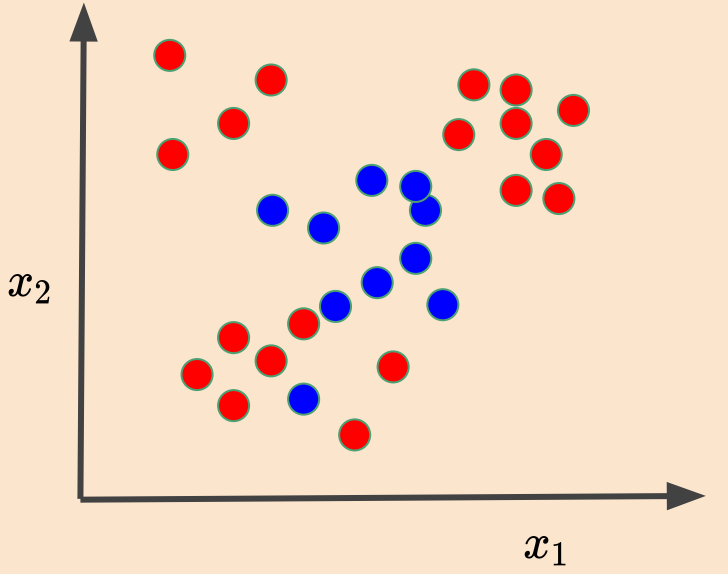
  So you use similarity to make decisions such as classification and regression
- You make decisions about one data point based in neighboring point.





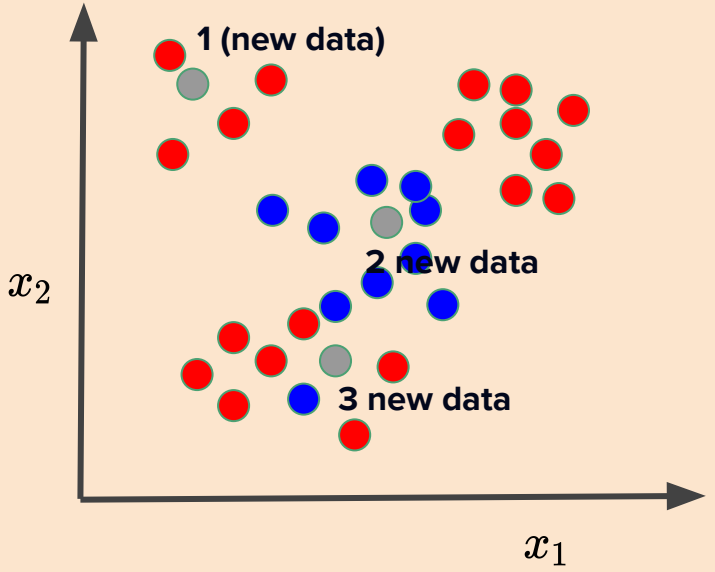
IBL-Nearest neighbor methods

- **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



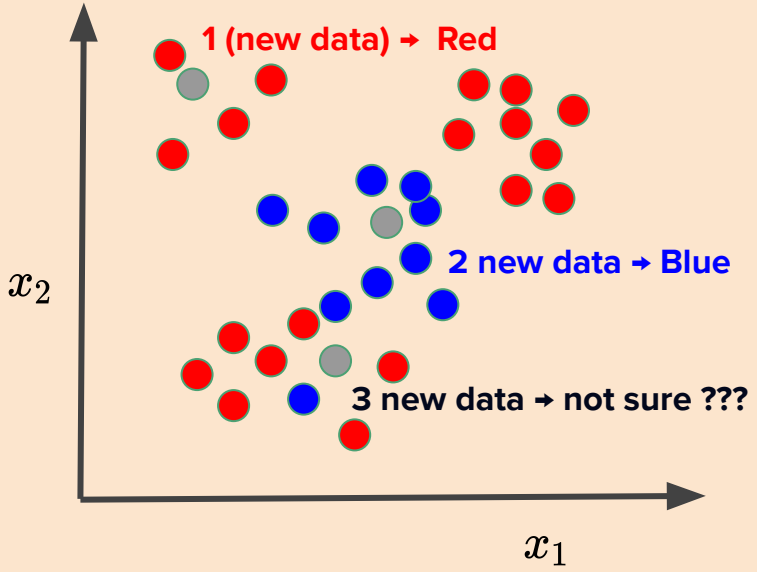
IBL-Nearest neighbor methods

- **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



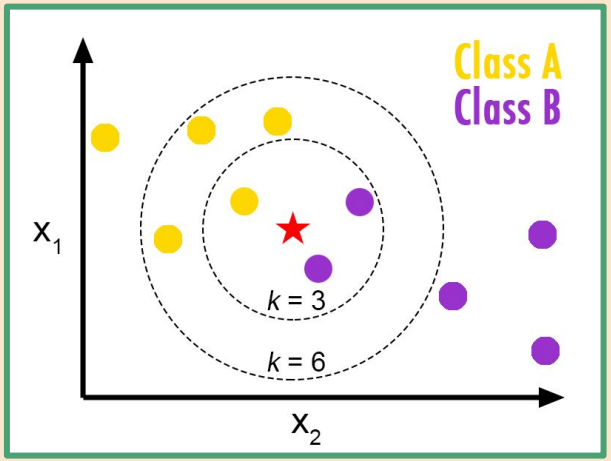
IBL-Nearest neighbor methods

- **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

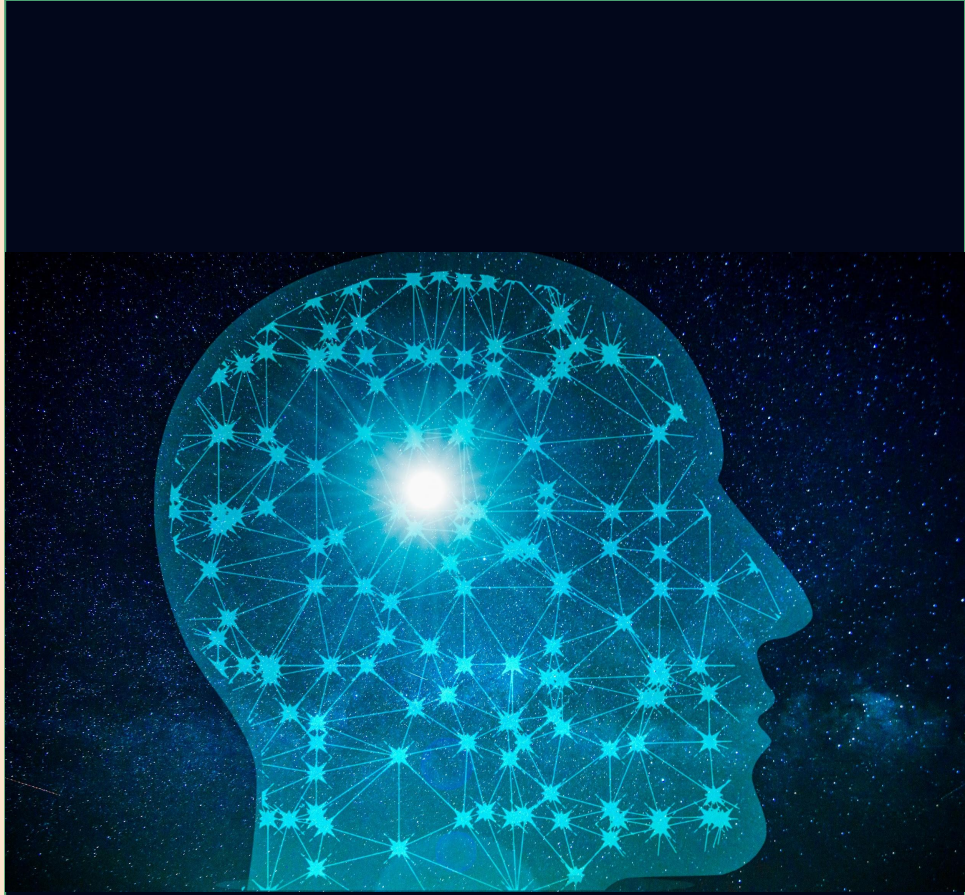


IBL-K- Nearest neighbor methods

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

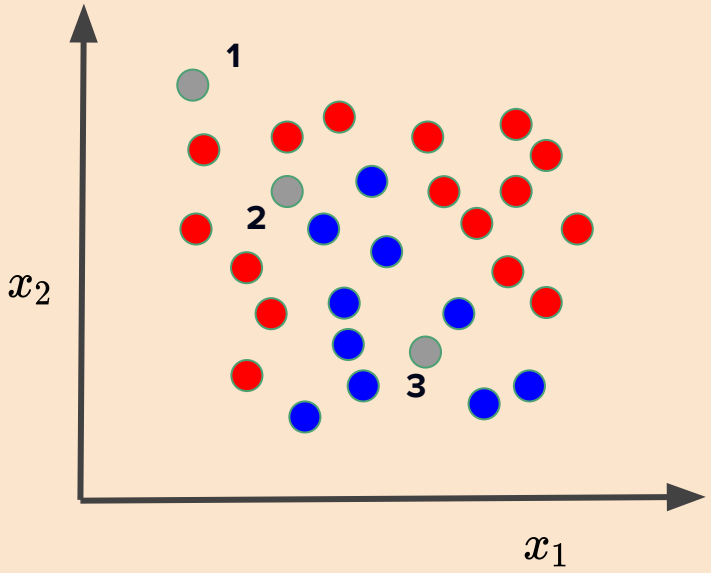


K-Nearest Neighbors Model



Example

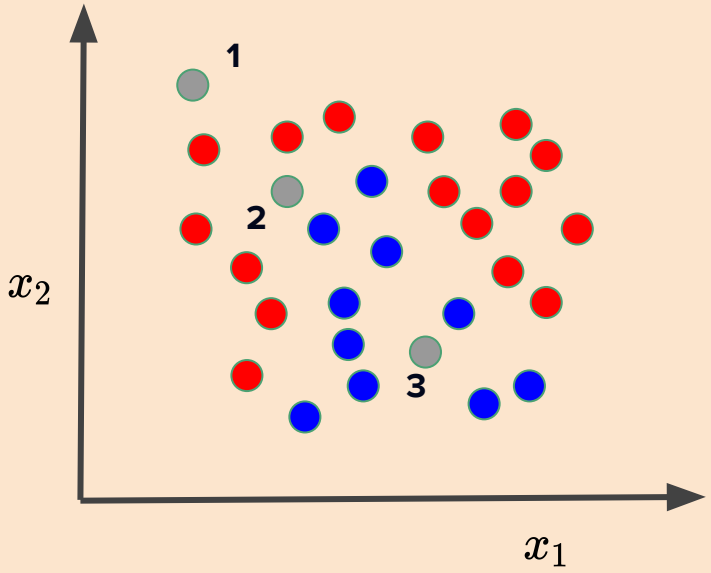
- Can you classify point 1, 2, 3 as blue or red by K-nearest neighbors model



Point	k = 1	k = 2	k = 3	k = 4
1				
2				
3				

Example

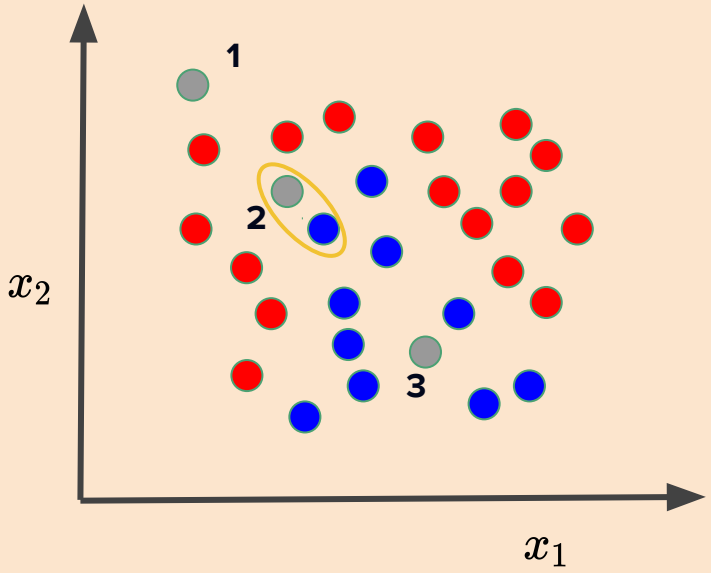
- Can you classify point 1, 2, 3 as blue or red by K-nearest neighbors model



Point	k = 1	k = 2	k = 3	k = 4
1	R	R	R	R
2				
3	B	B	B	B

Example

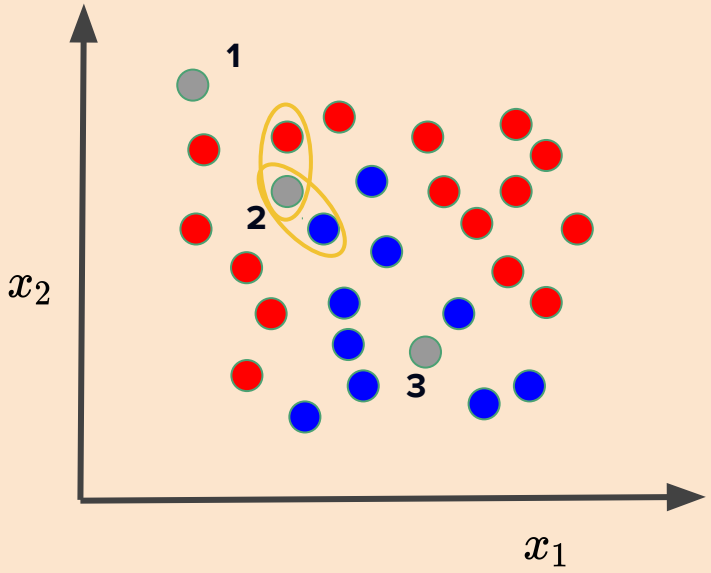
- Can you classify point 1, 2, 3 as blue or red by K-nearest neighbors model



Point	k = 1	k = 2	k = 3	k = 4
1	R	R	R	R
2	B			
3	B	B	B	B

Example

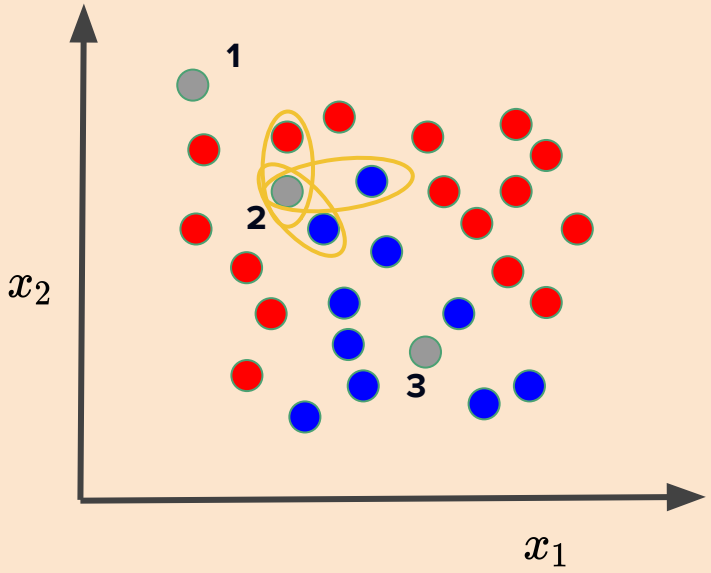
- Can you classify point 1, 2, 3 as blue or red by K-nearest neighbors model



Point	k = 1	k = 2	k = 3	k = 4
1	R	R	R	R
2	B	R/B		
3	B	B	B	B

Example

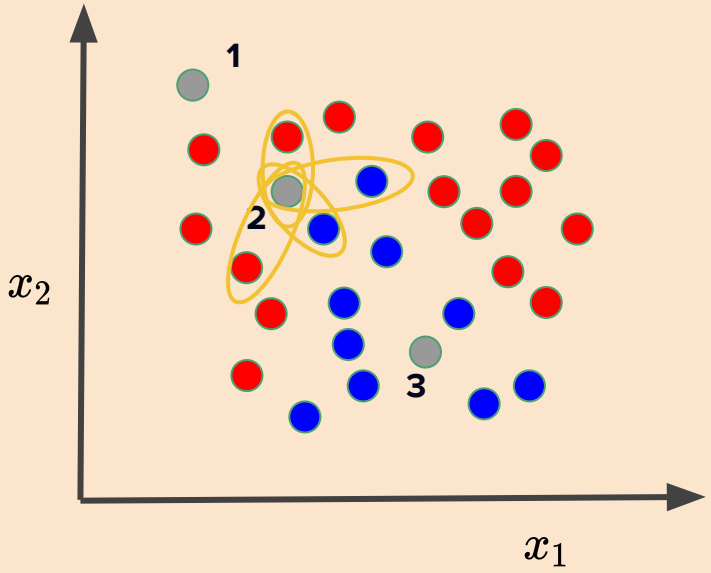
- Can you classify point 1, 2, 3 as blue or red by K-nearest neighbors model



Point	k = 1	k = 2	k = 3	k = 4
1	R	R	R	R
2	B	R/B	B	
3	B	B	B	B

Example

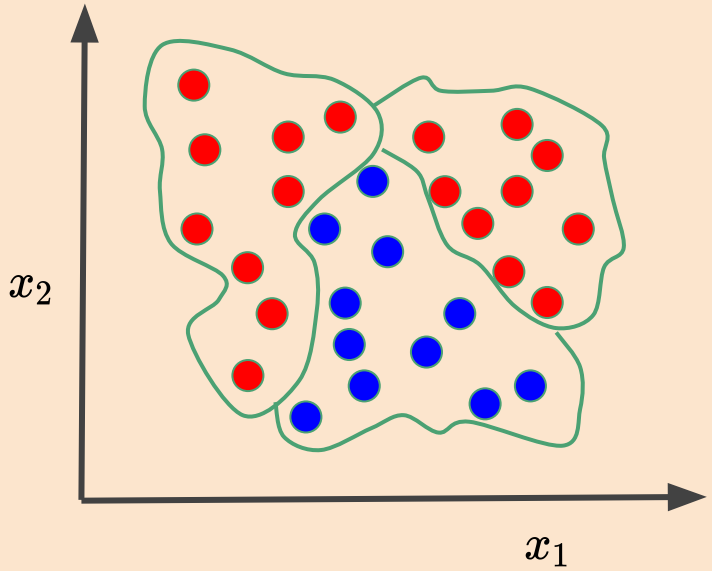
- Can you classify point 1, 2, 3 as blue or red by K-nearest neighbors model



Point	k = 1	k = 2	k = 3	k = 4
1	R	R	R	R
2	B	R/B	B	R/B
3	B	B	B	B

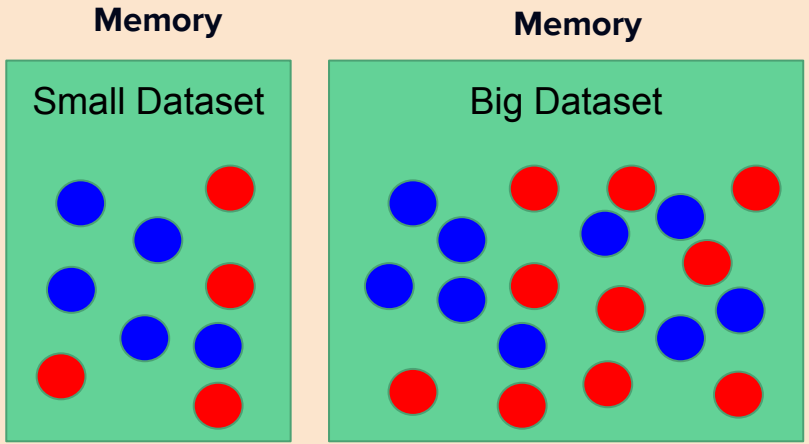
Pro and cons of K- Nearest neighbor methods

- **Pros**
- ➔ Training take no times
- ➔ Complex decision boundary is possible
- ➔ Information is not lost
- **Cons**
- ➔ Query is low
- ➔ Storage space is huge
- ➔ Easily fooled by irrelevant attributes



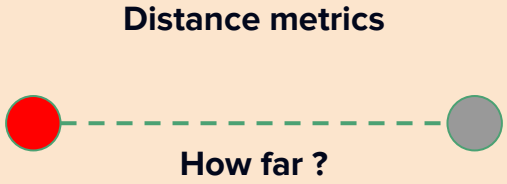
Pro and cons of K- Nearest neighbor methods

- **Pros**
- ➔ Training take no times
- ➔ Complex decision boundary is possible
- ➔ Information is not lost
- **Cons**
- ➔ Query is low
- ➔ Storage space is huge
- ➔ Easily fooled by irrelevant attributes

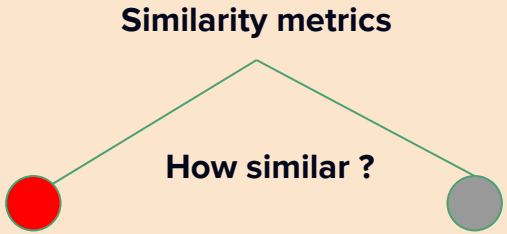


Distance and similarity Metrics

- **Distance metrics** are the numerical value that tells you that whether two points are close.



- **Similarity metrics** are the numerical value that tells you that whether two points are similar.



Distance metrics for real value features

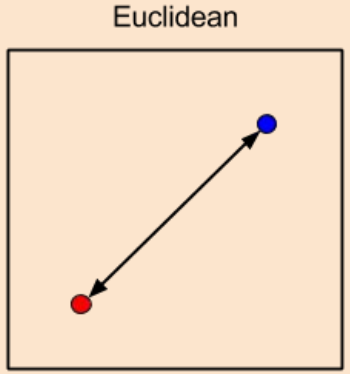
- **Euclidean Distance**

  In mathematics, the Euclidean distance between two points in Euclidean space is the length of a line segment between the two points.

$$d(p, q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

$p, q =$ two points in Eucliden n-space

$n = n - space$



Distance metrics for real value features

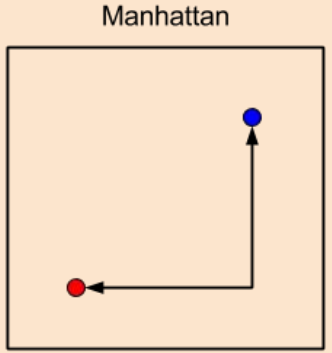
- **Manhattan Distance**

  In mathematics, the Manhattan distance between two points in real vector space is the sum of the lengths of the projections of the line segment between the points onto the coordinate axes.

$$d(p, q) = \sum_{i=1}^{n} |q_i - p_i|$$

$p, q =$ two points in real vector n-space

$n = n - space$



Distance metrics for real value features

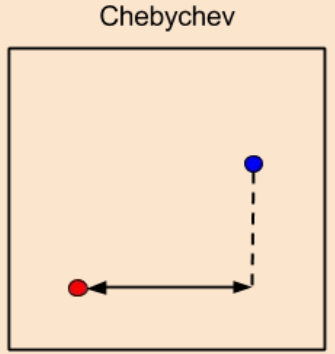
- **Chebyshev Distance**

  In mathematics, the Chebyshev distance between two points in real vector space is the maximum of the lengths of the projections of the line segment between the points onto the coordinate axes.

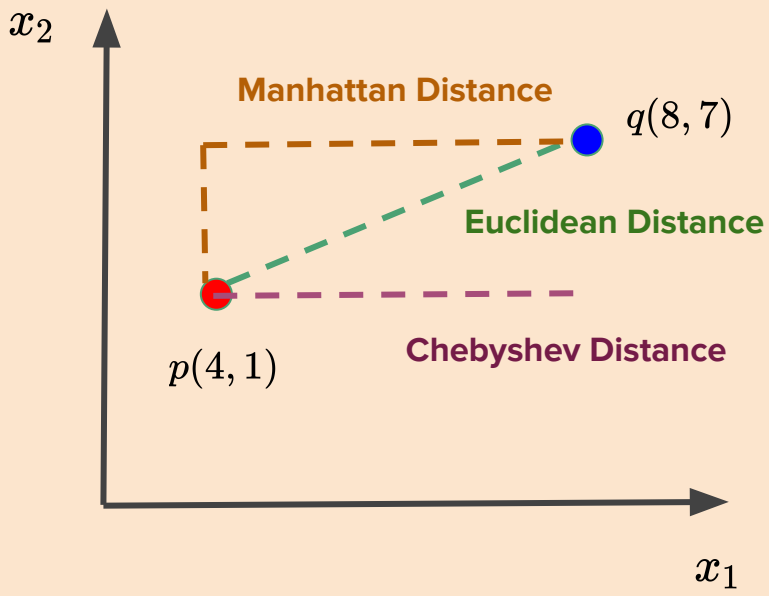
$$d(p, q) = max_i(|q_i - p_i|)$$

$p, q =$ two points in real vector n-space

$n = n - space$



Example



Euclidean Distance

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$
$$= \sqrt{(8-4)^2 + (7-1)^2} = 7.21$$

Manhattan Distance

$$d(p,q) = \sum_{i=1}^{n} |q_i - p_i|$$
$$= |8-4| + |7-1| = 10$$

Chebyshev Distance

$$d(p,q) = max_i(|q_i - p_i|)$$
$$= max(|8-4|, |7-1|) = 6$$

Distance metrics for boolean features

- **Jaccard Distance**

Feature	Me	Dad
Man barber	F	T
Toyota	T	T
MK	T	T
Water park	T	F
Temple	F	T
Bar	F	F

Distance metrics for boolean features

- **Jaccard Distance**

Feature	Me	Dad
Man barber	F	T
Toyota	T	T
MK	T	T
Water park	T	F
Temple	F	T
Bar	F	F

$$N = 6$$
$$NTT = 2$$
$$NTF = 1$$
$$NFT = 2$$
$$NFF = 1$$
$$NNEQ \text{ : Number of non-equal dimensions}$$
$$NNEQ = NTF + NFT = 1 + 2 = 3$$
$$NNZ \text{ : Number of nonzero dimensions}$$
$$NNZ = NTF + NFT + NTT = 1 + 2 + 2 = 5$$

$$JD = \frac{NNEQ}{NNZ} = \frac{3}{5} = 0.6$$

Distance metrics for boolean features

- **Jaccard Distance**

Feature	Me	Dad
Man barber	F	T
Toyota	T	T
MK	T	T
Water park	T	F
Temple	F	T
Bar	F	F

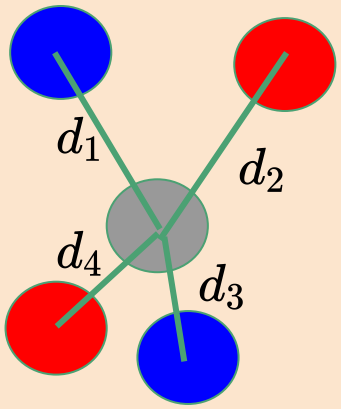
$$JD = \frac{NNEQ}{NNZ} = \frac{3}{5} = 0.6$$

$$JD \longrightarrow 1 : \text{not similar}$$

$$JD \longrightarrow 0 : \text{similar}$$

Using distance as weights

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



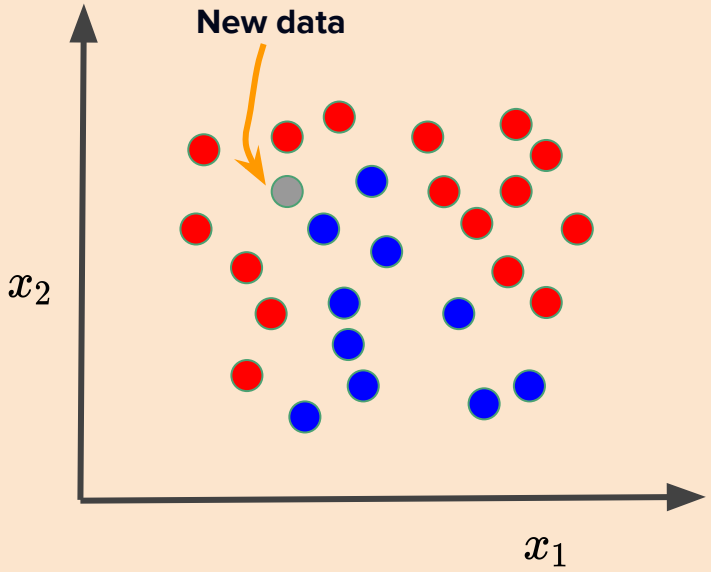
regression

$$w_i = \frac{1}{d_i}; i = 1, 2, 3, 4$$

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

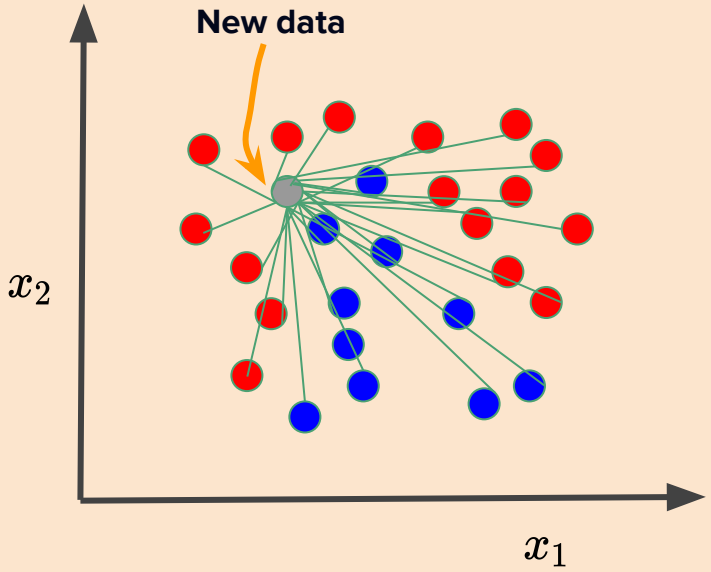
Searching for nearest neighbors

- **Brute force :** when a new sample x’ appears, calculate the distance between x’ and all other points. Consider points with lowest distance for voting.



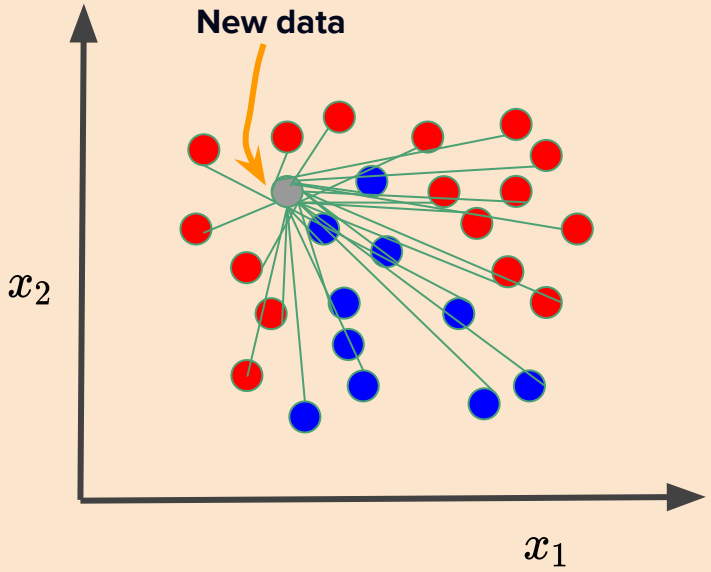
Searching for nearest neighbors

- **Brute force :** when a new sample x’ appears, calculate the distance between x’ and all other points. Consider points with lowest distance for voting.



Searching for nearest neighbors

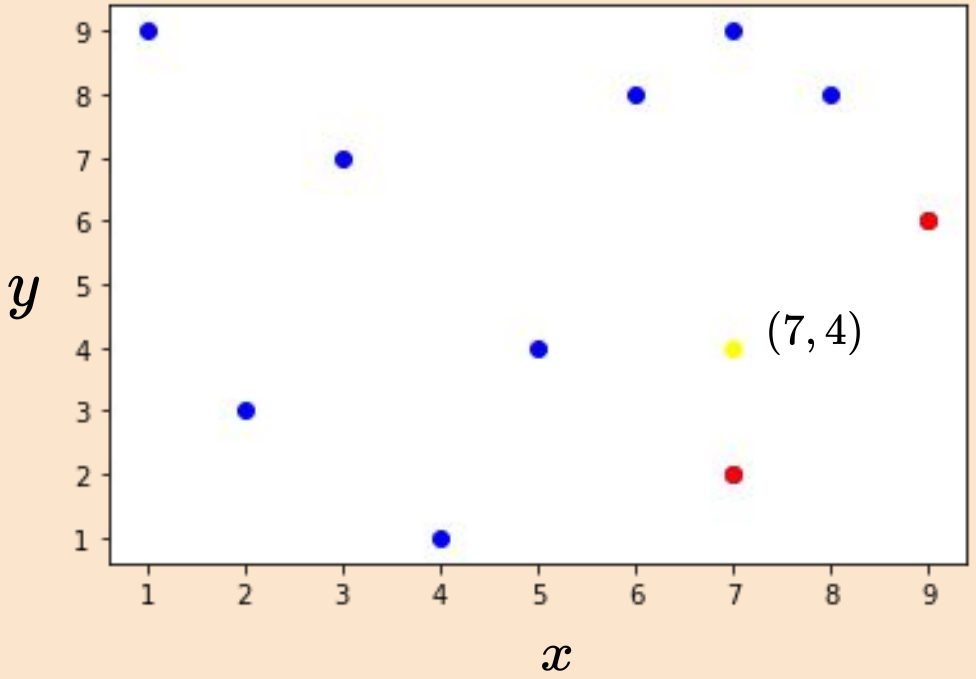
- **Brute force :** when a new sample x' appears, calculate the distance between x' and all other points. Consider points with lowest distance for voting.



- **KD tree :** To speed up the search

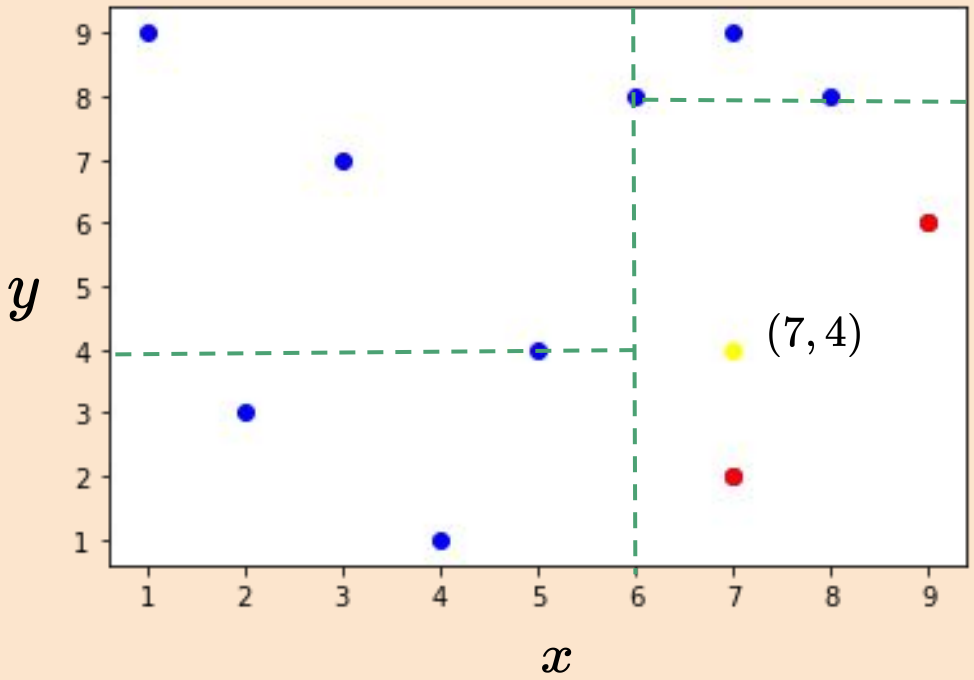
K-D Tree

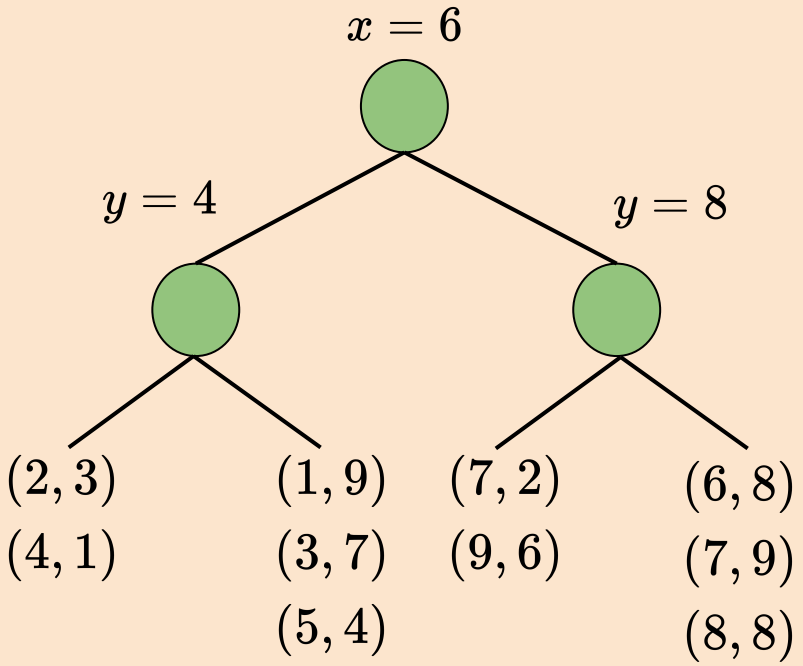
- Data = [(1,9),(2,3),(4,1),(3,7),(5,4),(6,8),(7,2),(8,8),(7,9),(9,6)]
- We would like to search for nearest neighbors of point (7,4)



K-D Tree

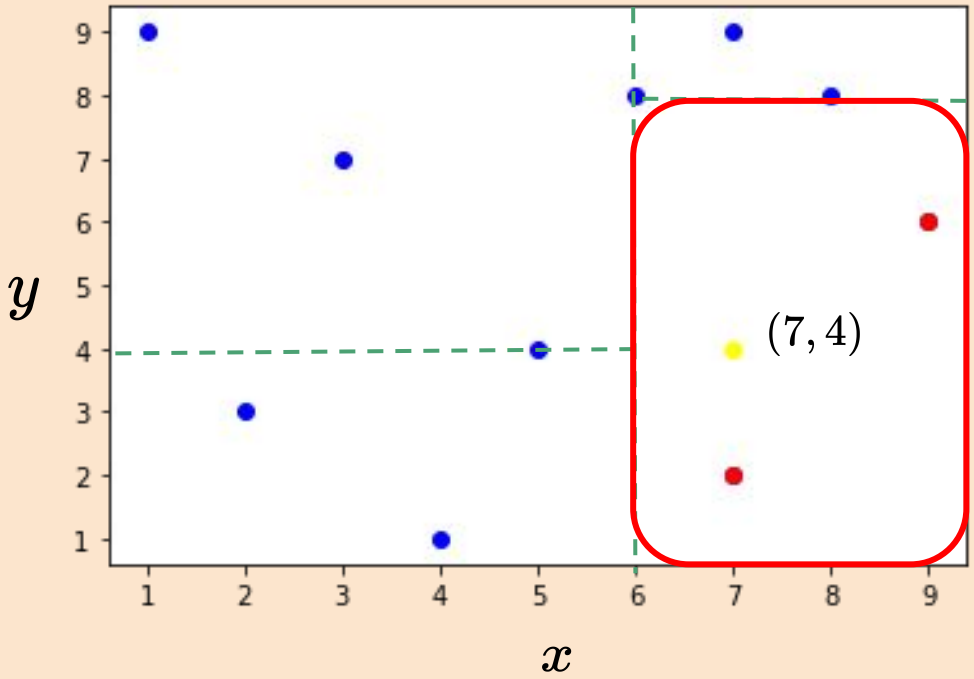
- First, pick a random dimension (assume x) find median and spilt sata. Repeat for other dimensions.

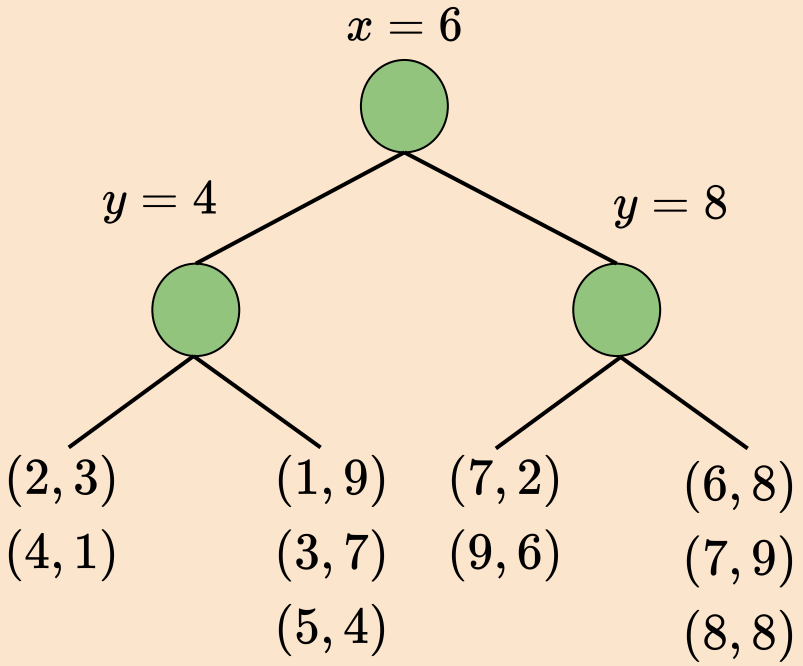




K-D Tree

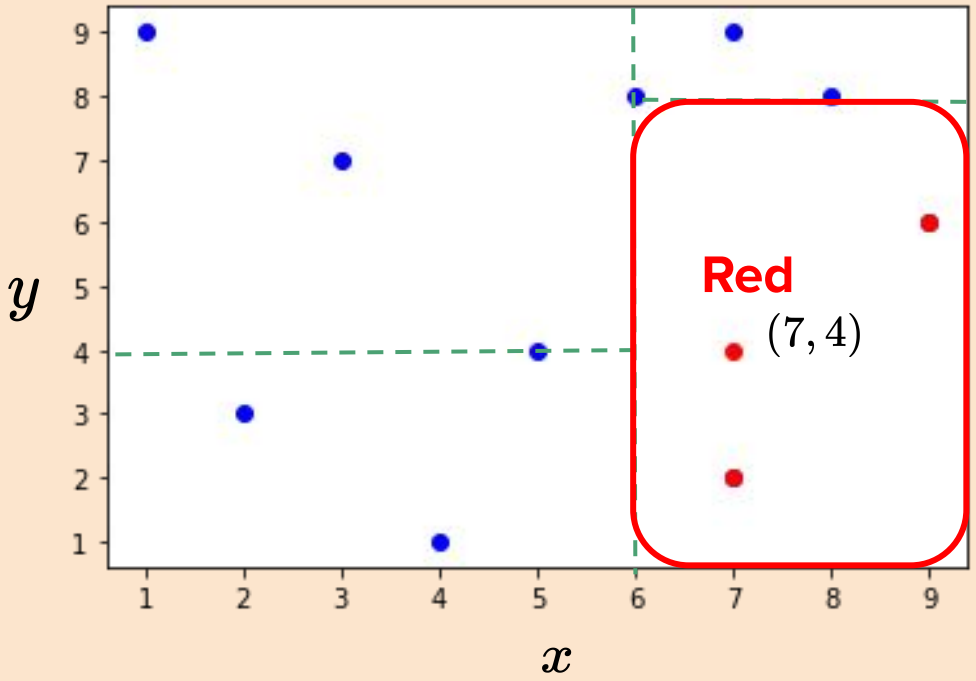
- First, pick a random dimension (assume x) find median and spilt sata. Repeat for other dimensions.

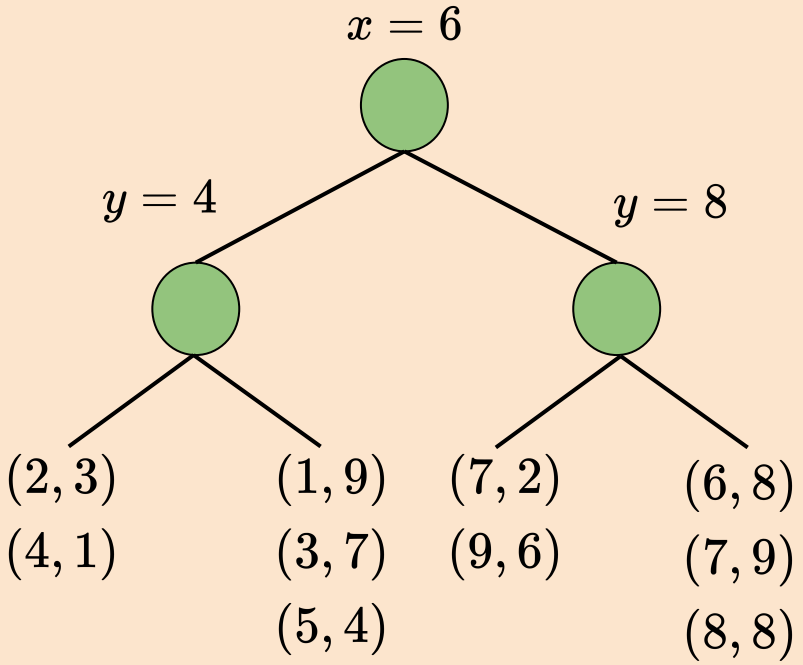




K-D Tree

- First, pick a random dimension (assume x) find median and spilt sata. Repeat for other dimensions.





Good luck 😉


Uppsala universitet 1852–1916
Det goda universitetet. Festskrift till Anders Hallberg
FOSTERA CAPRICEN
Studenten, Staden och Samingen
UPPSALA UNIVERSITE