Artificial Intelligence

Lab 8

Machine Learning Algorithms
ID3
DBscan

Agenda

Decision tree.

- ID3

Clustering

- DBSCAN Algorithm.

Decision Trees

- The idea is to partition input space into a disjoint set of regions and to use a very simple predictor for each region.
- For classification simply predict the most frequent class in the region

Play tennis training data

- Hard to guess.
- Divide & Conquer:
  - split into subsets
  - are they are pure? (all yes or all no)
  - if yes: stop.
  - If no: repeat.
- See which subset new data falls into

Training examples: **9 yes / 5 no**

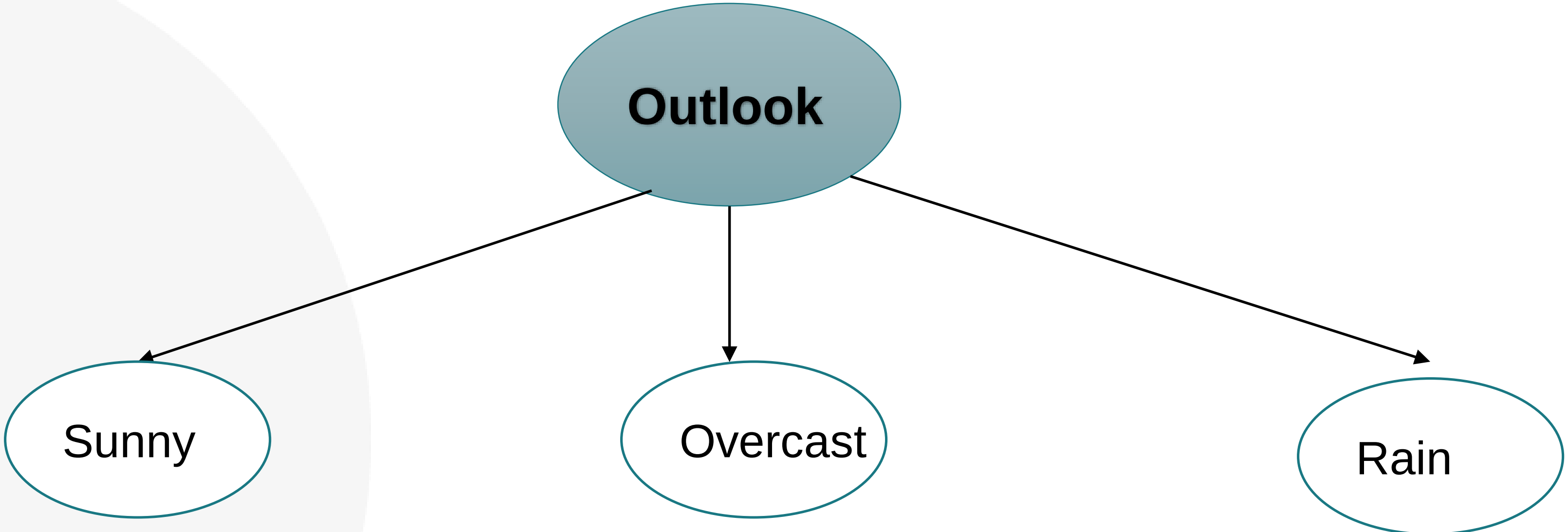
Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	Overcast	High	Weak	Yes
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No

New Data

D15	Rain	High	weak	?

Decision Tree Representation

- Each internal node tests an attribute.
- Each branch corresponds to attribute value.
- Each leaf node make a prediction.



Day	Outlook	Humid	Wind
D1	Sunny	High	Weak
D2	Sunny	High	Strong
D8	Sunny	High	Weak
D9	Sunny	Normal	Weak
D11	Sunny	Normal	Strong

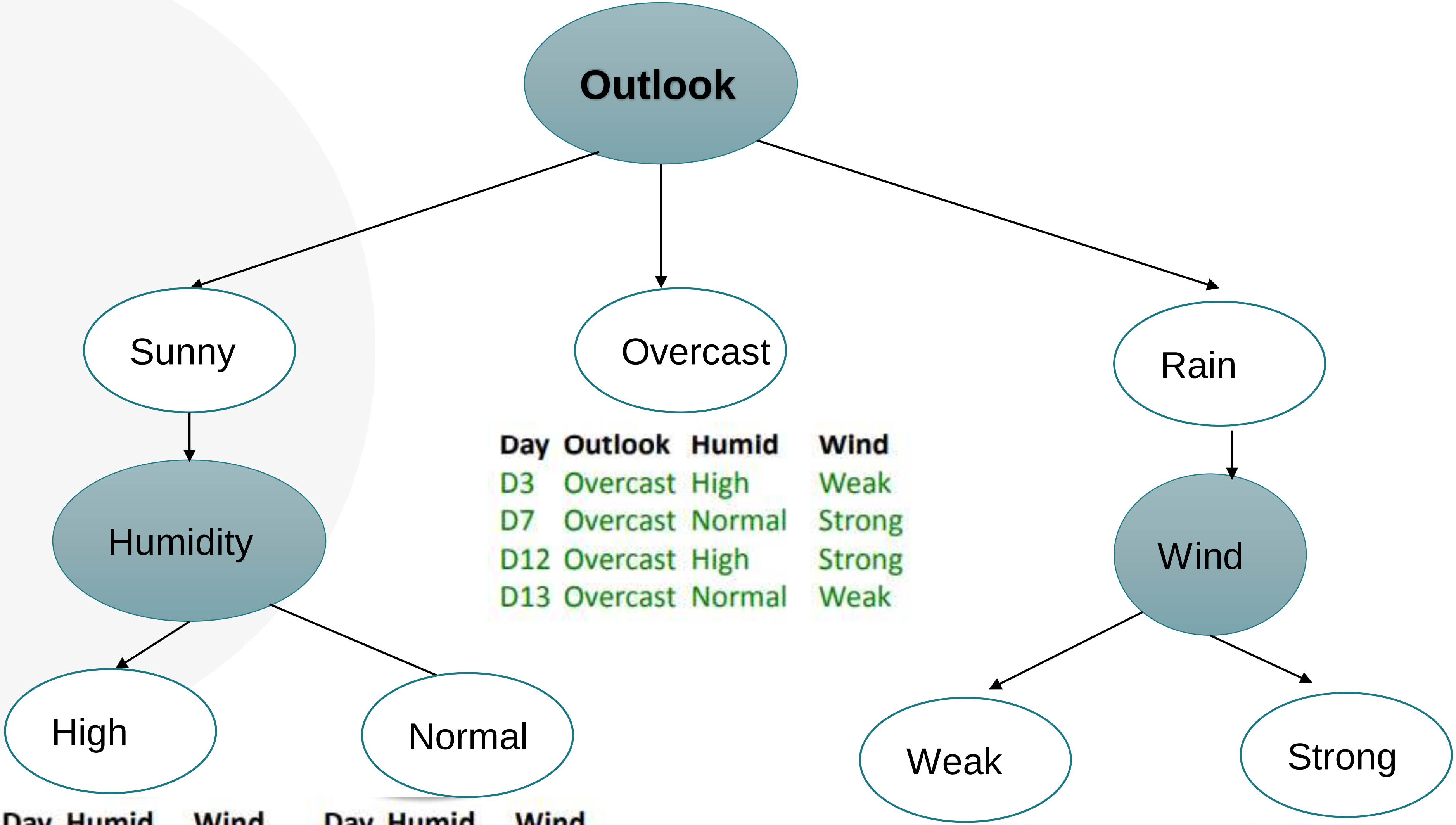
2 yes / 3 no
split further

Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

4 yes / 0 no
pure subset

Day	Outlook	Humid	Wind
D4	Rain	High	Weak
D5	Rain	Normal	Weak
D6	Rain	Normal	Strong
D10	Rain	Normal	Weak
D14	Rain	High	Strong

3 yes / 2 no
split further



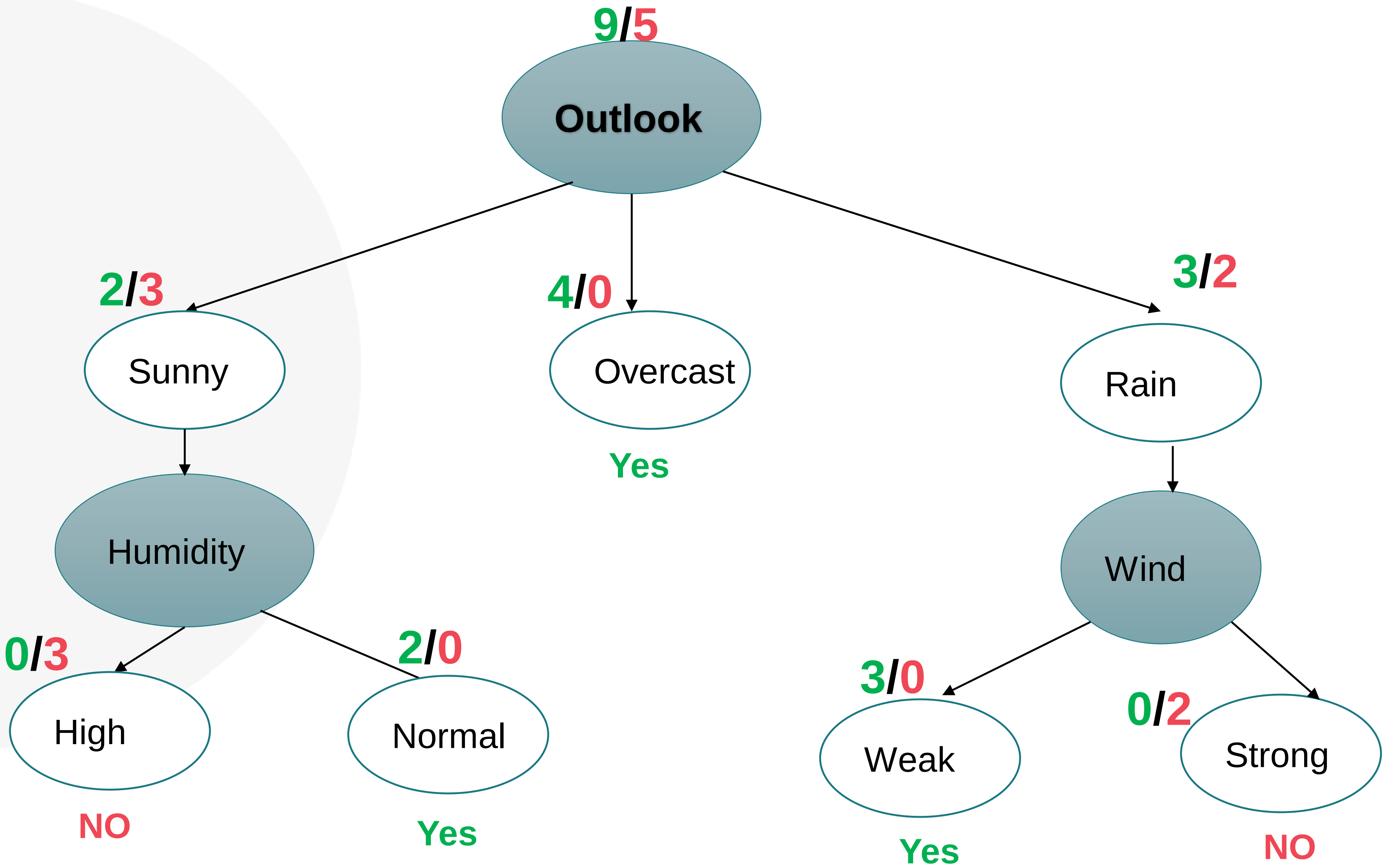
Day	Outlook	Humid	Wind
D3	Overcast	High	Weak
D7	Overcast	Normal	Strong
D12	Overcast	High	Strong
D13	Overcast	Normal	Weak

Day	Humid	Wind
D1	High	Weak
D2	High	Strong
D8	High	Weak

Day	Humid	Wind
D9	Normal	Weak
D11	Normal	Strong

Day	Humid	Wind
D4	High	Weak
D5	Normal	Weak
D10	Normal	Weak

Day	Humid	Wind
D6	Normal	Strong
D14	High	Strong


9/5
Outlook
2/3
Sunny
4/0
Overcast
Yes
3/2
Rain
Humidity
Wind
0/3
High
NO
2/0
Normal
Yes
3/0
Weak
Yes
0/2
Strong
NO

Which attribute to split on

- Want to measure "purity" of the split
  - more certain about Yes/No after the split
    - pure set (4 yes / 0 no) => completely certain (100%)
    - impure (3 yes / 3 no) => completely uncertain (50%)
  - can't use P("yes" | set):
    - must be symmetric: 4 yes / 0 no as pure as 0 yes / 4 no

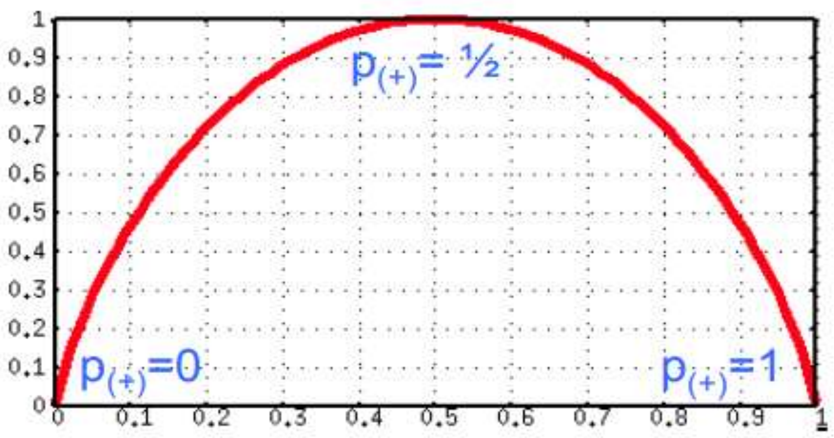
Entropy

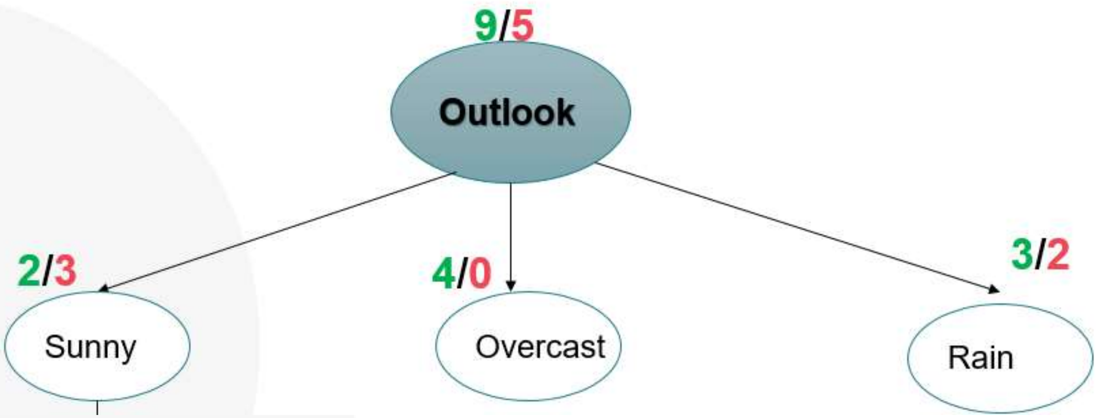
- Entropy: $H(S) = - p_{(+)} \log_2 p_{(+)} - p_{(-)} \log_2 p_{(-)}$ bits
  - S … subset of training examples
  - $p_{(+)}$ / $p_{(-)}$ … % of positive / negative examples in S
- Interpretation: assume item X belongs to S
  - how many bits need to tell if X positive or negative
- impure (3 yes / 3 no):

$$H(S) = -\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6} = 1 \text{ bits}$$

- pure set (4 yes / 0 no):

$$H(S) = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0 \text{ bits}$$





$$H(S) = -p_{(+)} \log_2 p_{(+)} - p_{(-)} \log_2 p_{(-)}$$

- H(Outlook) = $-\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14}$
- H(Sunny) = $-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}$
- H(Overcast) = $-\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4}$
- H(Rain) = $-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}$

Information Gain

V ... possible values of A
S ... set of examples {X}
S_V ... subset where $X_A = V$

Want many items in pure sets.

Expected drop in entropy after split:

$$Gain(S,A) = H(S) - \sum_{V \in Values(A)} \frac{|S_V|}{|S|} H(S_V)$$

Gain (S, Wind)

$= H(S) - {}^{8}/_{14}\, H(S_{weak}) - {}^{6}/_{14}\, H(S_{strong})$

$= 0.94 - {}^{8}/_{14} * 0.81 - {}^{6}/_{14} * 1.0$

$= 0.049$

Wind Example

$-\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14}$ 9 yes / 5 no

$H(S) = 0.94$

6 yes / 2 no

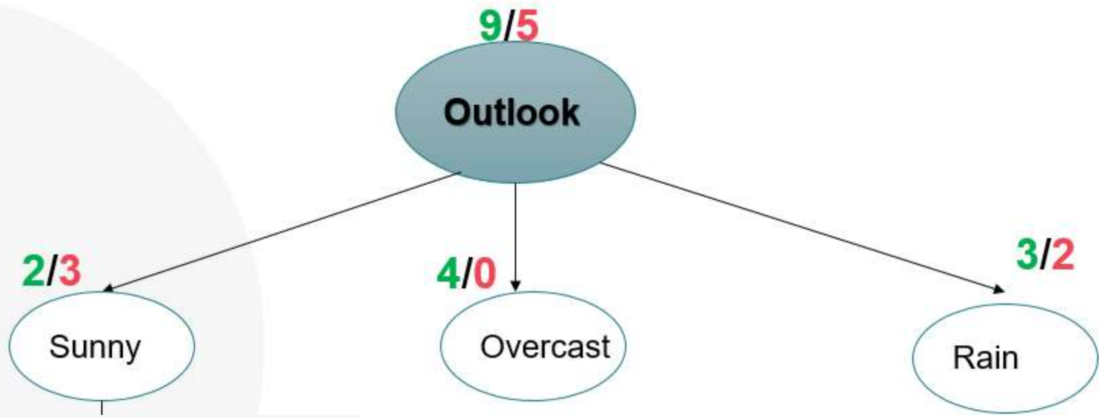
$-\frac{6}{8}\log_2\frac{6}{8} - \frac{2}{8}\log_2\frac{2}{8}$

$H(S_{weak}) = 0.81$

3 yes / 3 no

$-\frac{3}{6}\log_2\frac{3}{6} - \frac{3}{6}\log_2\frac{3}{6}$

$H(S_{strong}) = 1.0$



$$Gain(S,A) = H(S) - \sum_{V \in Values(A)} \frac{|S_V|}{|S|} H(S_V)$$

- H(Outlook) = $-\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14}$
- Gain(Outlook) = H(Outlook) $- \sum_{v \in Outlook} \frac{S_v}{S} H(Sv)$
- Gain(Outlook) = H(Outlook) - ($\frac{5}{14}$ H(Sunny) + $\frac{4}{14}$ H(Overcast) + $\frac{5}{14}$ H(Rain))

Similarly,

Note: Highest gain is always selected.

Gain(Humidity)=0.151

Gain(Outlook)=0.246

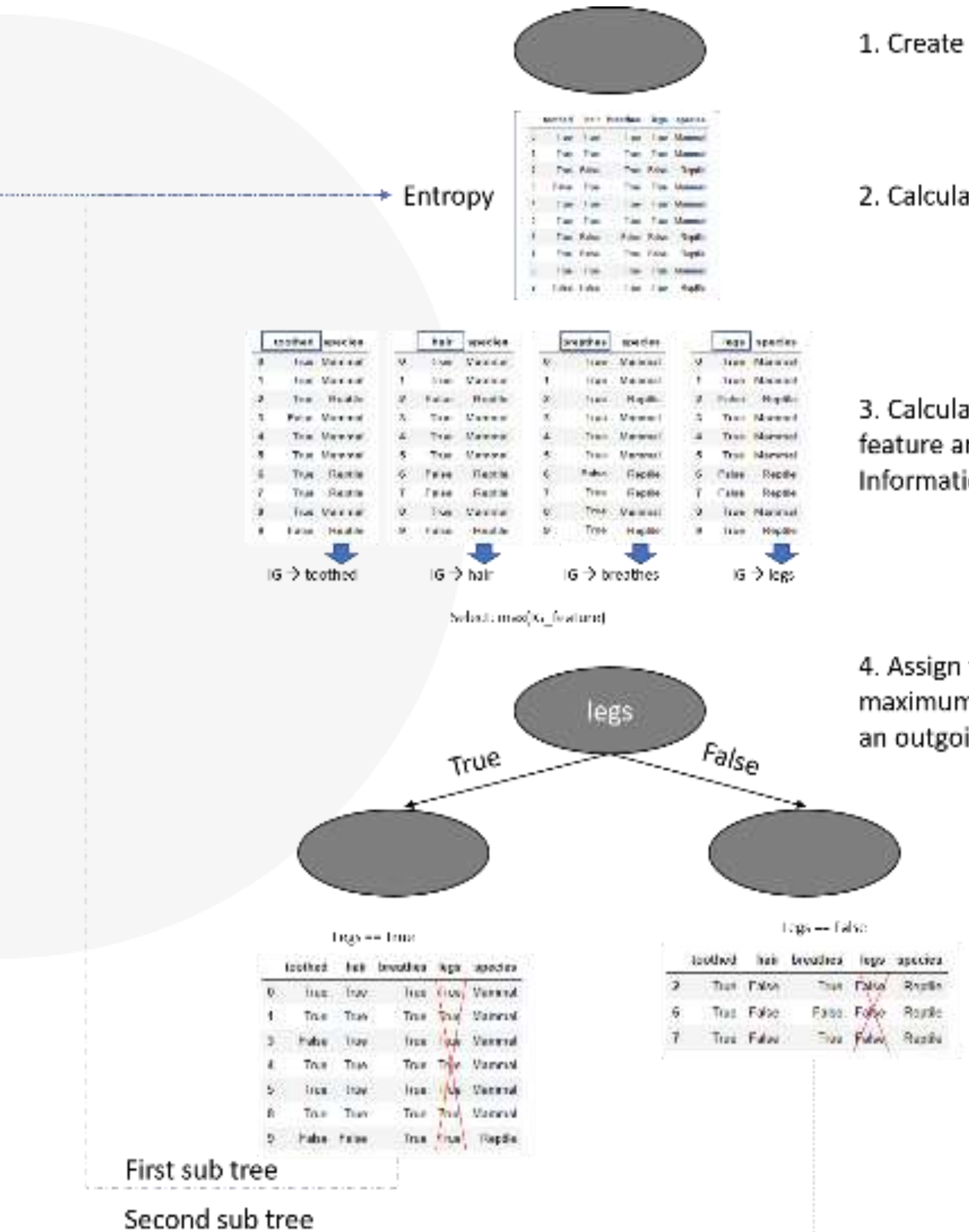
Gain(Wind)=0.048

Choose the highest to split on

ID3 Algorithm

- Split (node, {examples}):
    1. A ← the best attribute for splitting the {examples}
    2. Decision attribute for this node ← A
    3. For each value of A, create new child node
    4. Split training {examples} to child nodes
    5. If examples perfectly classified: STOP
       else: iterate over new child nodes

       Split (child_node, {subset of examples})



1. Create a root node

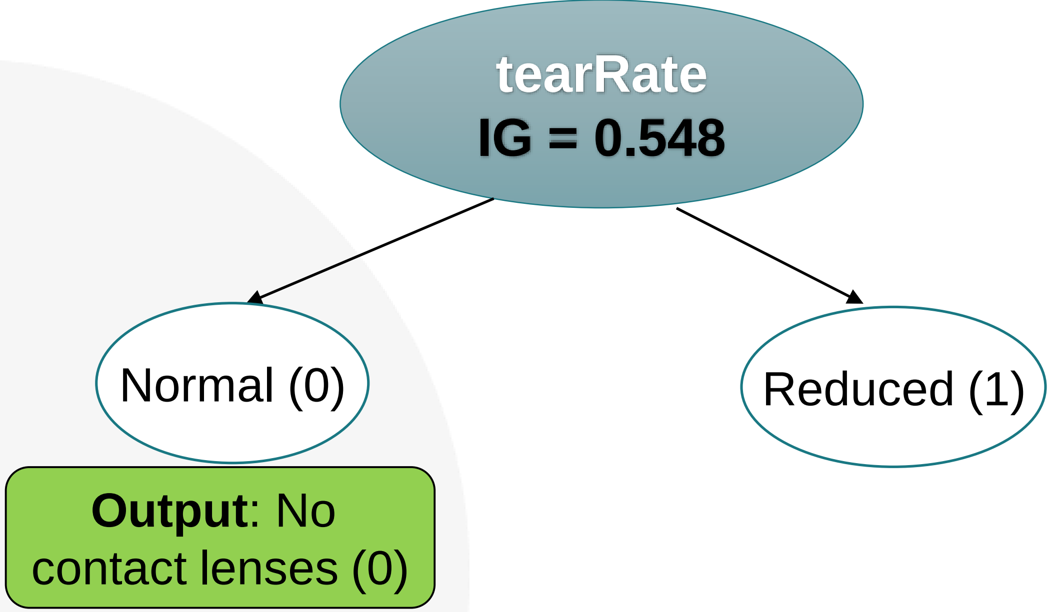
2. Calculate the entropy of the whole (sub) dataset

3. Calculate the Information gain of each single feature and pick that feature with the larges Information gain

4. Assign the (root) node the label of the feature with the maximum information gain. Grow for each feature value an outgoing branch and add unlabelled nodes at the end

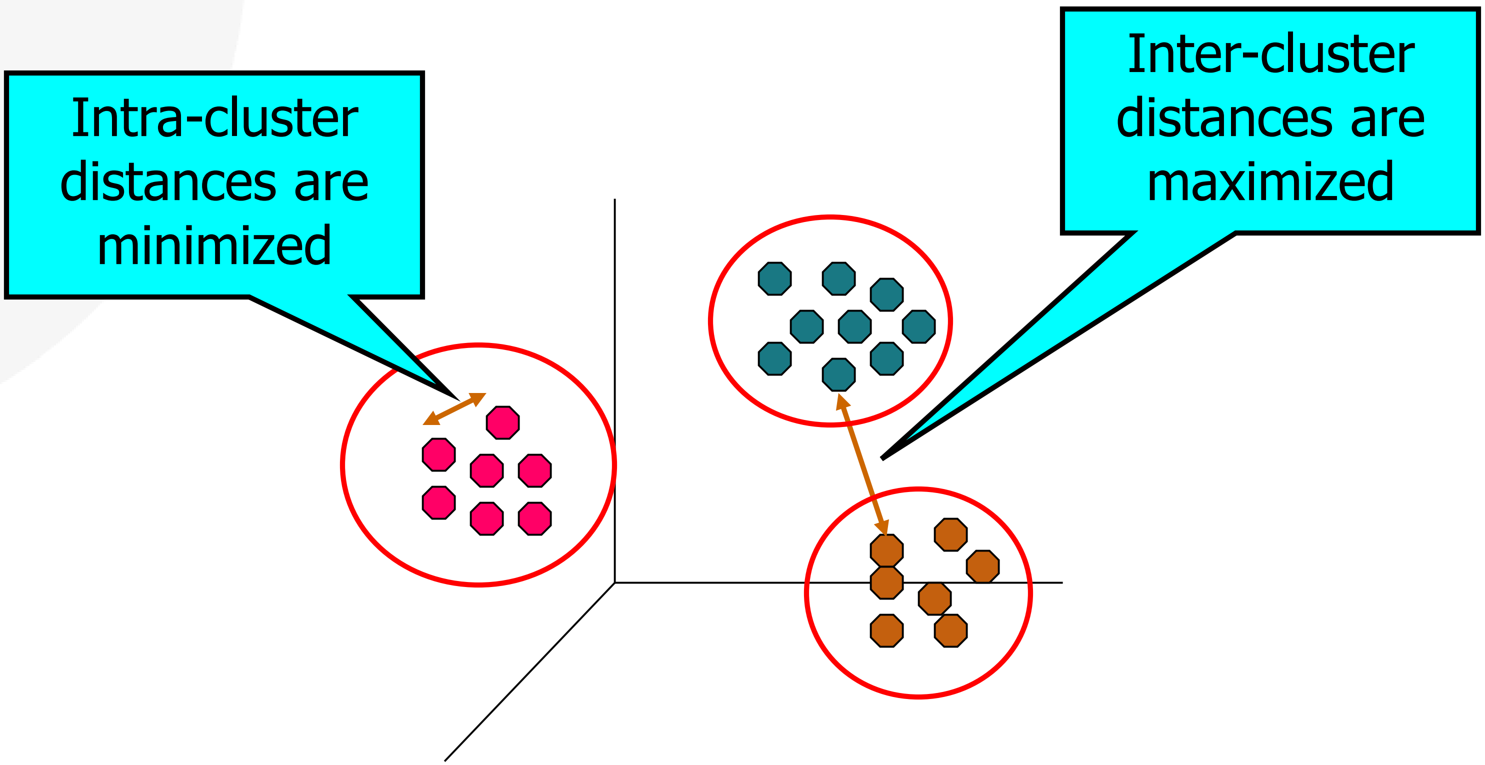
5. Split the dataset along the values of the maximum information gain feature and remove this feature from the dataset

6. For each of the sub_datasets, repeat steps 2 to 5 until a stopping criteria is satisfied →Here the recursion kicks in


tearRate
IG = 0.548
Normal (0)
Reduced (1)
Output: No contact lenses (0)

What is a Clustering?

In general a grouping of objects such that the objects in a group (cluster) are similar (or related) to one another and different from (or unrelated to) the objects in other groups



DBSCAN: Density-Based Clustering

DBSCAN is a Density-Based Clustering algorithm

Reminder: In density based clustering we partition points into dense regions separated by not-so-dense regions.

Important Questions:

- How do we measure density?
- What is a dense region?

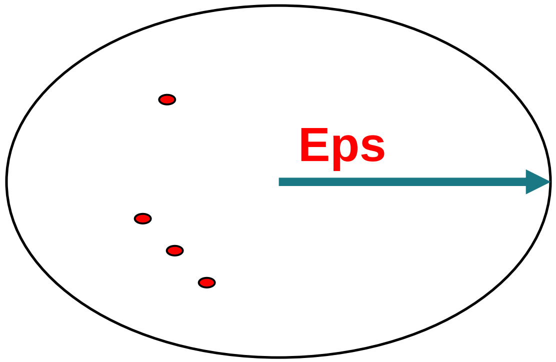
DBSCAN:

- Density at point p: number of points within a circle of radius Eps
- Dense Region: A circle of radius Eps that contains at least MinPts points

Dbscan model parameters

Eps : defines the **radius of neighborhood around a point x**. It's called the epsilon-neighborhood of x.

The parameter **MinPts** is the minimum number of neighbors within "eps" radius.



MinPts =4

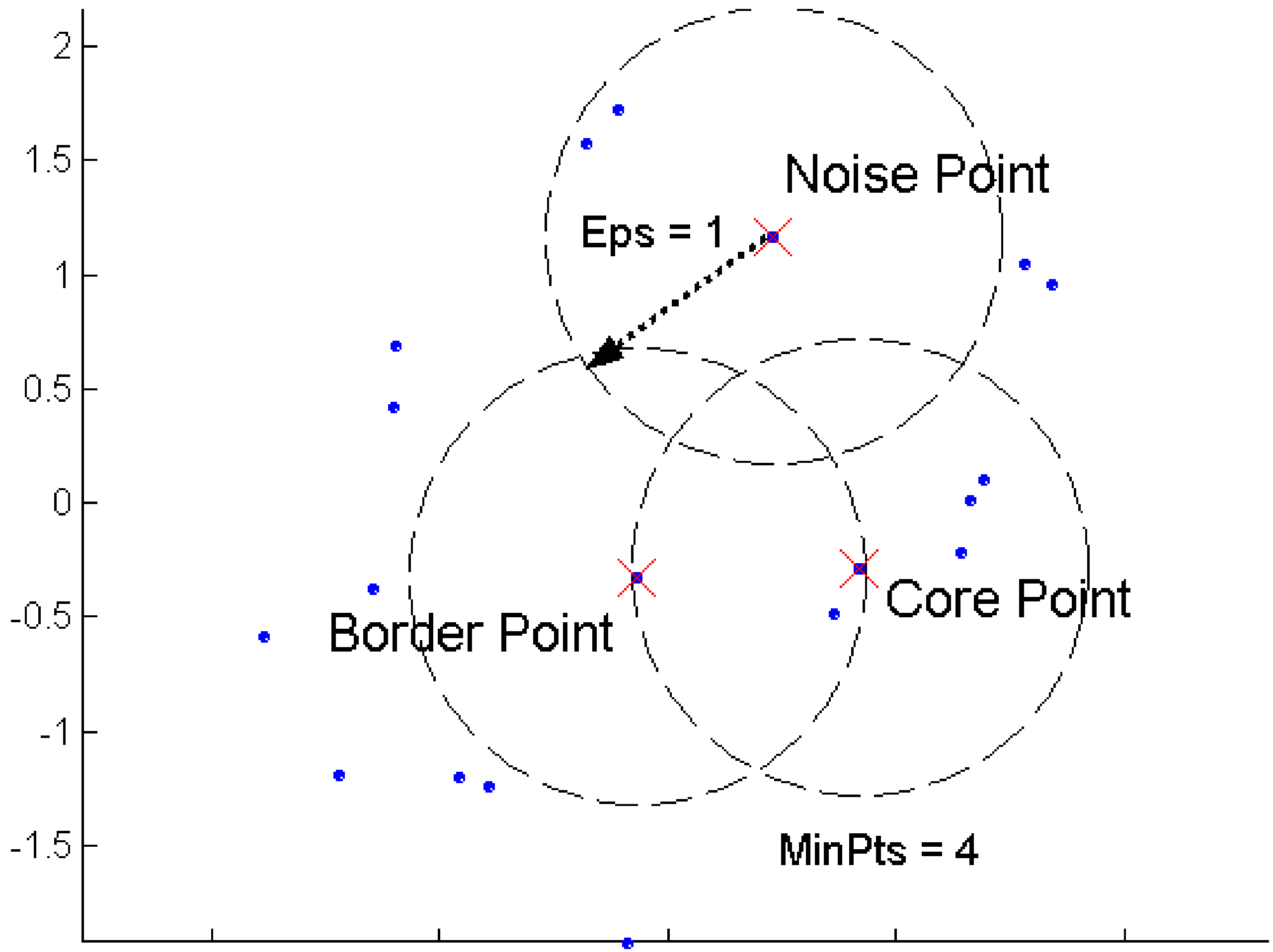
DBSCAN

Characterization of points

Density=number of points within a specified radius r (Eps)

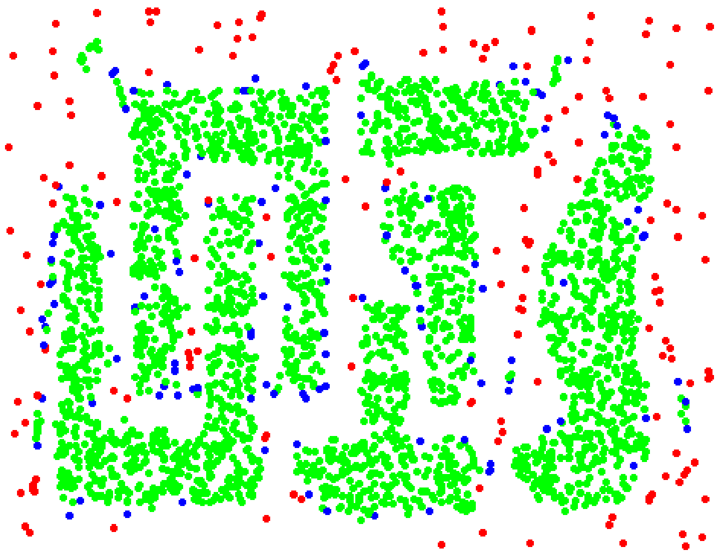
- A point is a core point if it has more than a specified number of points (MinPts) within Eps
  - These points belong in a dense region and are at the interior of a cluster
- A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point.
- A noise point is any point that is not a core point or a border point.

DBSCAN: Core, Border, and Noise Points



DBSCAN: Core, Border and Noise Points

Original Points



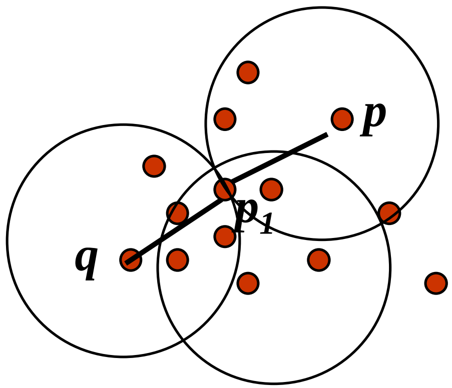
Point types: core, border and noise

Eps = 10, MinPts = 4

Density-Connected points

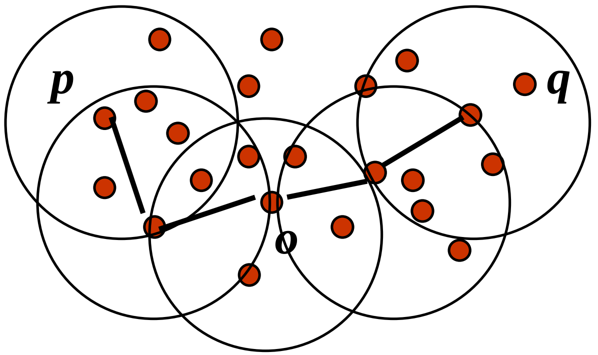
Density edge

- We place an edge between two core points q and p if they are within distance Eps.



Density-connected

- A point p is density-connected to a point q if there is a path of edges from p to q



DBSCAN Algorithm

Label points as core, border and noise

Eliminate noise points

For every core point p that has not been assigned to a cluster

- Create a new cluster with the point p and all the points that are density-connected to p.

Assign border points to the cluster of the closest core point.

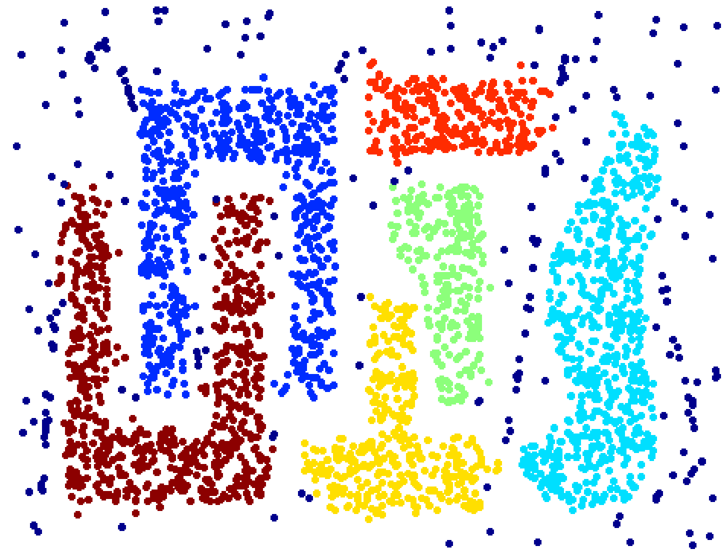
```
DBSCAN(D, epsilon, min_points):
    C = 0
    for each unvisited point P in dataset
        mark P as visited
        sphere_points = regionQuery(P, epsilon)
        if sizeof(sphere_points) < min_points
            ignore P
        else
            C = next cluster
            expandCluster(P, sphere_points, C, epsilon, min_points)

expandCluster(P, sphere_points, C, epsilon, min_points):
    add P to cluster C
    for each point P' in sphere_points
        if P' is not visited
            mark P' as visited
            sphere_points' = regionQuery(P', epsilon)
            if sizeof(sphere_points') >= min_points
                sphere_points = sphere_points joined with sphere_points'
            if P' is not yet member of any cluster
                add P' to cluster C

regionQuery(P, epsilon):
    return all points within the n-dimensional sphere centered at P with radius epsilon (including P)
```

When DBSCAN Works Well

Original Points



Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

Advantages & Disadvantages of DBSCAN

Advantages:

- Unlike K-means, DBSCAN not required to specify number of clusters to be generated.
- Find any shape of clusters
- Can identify the outliers

Disadvantages:

- Does not work well with high dimensional datasets
- Parameters selections are tricky

Hands on

Open Dbscan algorithm template and complete the DBSCAN & Expand functions

Questions?