# Deadlines

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Date of assignment</th>
<th>Deadline (midnight 23:59)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW1</td>
<td>Sep 6</td>
<td>Sep 19</td>
</tr>
<tr>
<td>HW2</td>
<td>Sep 20</td>
<td>Oct 3</td>
</tr>
<tr>
<td>HW3</td>
<td>Oct 4</td>
<td>Oct 17</td>
</tr>
<tr>
<td>Paper summary</td>
<td>Oct 11</td>
<td>Oct 24</td>
</tr>
<tr>
<td>HW4</td>
<td>Oct 18</td>
<td>Oct 31</td>
</tr>
<tr>
<td>HW5</td>
<td>Nov 1</td>
<td>Nov 14</td>
</tr>
<tr>
<td>HW6</td>
<td>Nov 22</td>
<td>Dec 5</td>
</tr>
<tr>
<td>Project</td>
<td>TBA</td>
<td>Dec 13 - 15</td>
</tr>
</tbody>
</table>

*All deadlines are subject to change, check out CampusWire and website for updates*
Recap
Machine Learning

Supervised Learning

Unsupervised Learning

Reinforcement Learning
Machine Learning

Reinforcement Learning

Classification

Regression

Supervised Learning

Unsupervised Learning

Reinforcement Learning

Machine Learning
Unsupervised Learning

Supervised Learning

Classification

Regression

Reinforcement Learning

Unsupervised Learning
Machine Learning

Supervised Learning
- Classification
- Regression

Reinforcement Learning

Unsupervised Learning
- Clustering
- Dimensionality reduction
High Dimensional Space

Low Dimensional Space

Dimensionality reduction

Unsupervised Learning

Torus image credit: https://mathematica.stackexchange.com/questions/39879/create-a-torus-with-a-hexagonal-mesh-for-3d-printing
What is the problem with high-dimensional things?

- **Hard to visualise**
- **Methods trained** on high-dimensional data suffer from the **curse of dimensionality**
- **Algorithms** tend to get slow
What is the **curse of dimensionality**?

In order to keep **high-dimensional** space reasonably covered you need a lot more data.

On average **55.5%** of cells will be either *empty* or *singleton*.

On average **92.5%** of cells will be either *empty* or *singleton*.
200-Dimensional data?

Are all of these dimensions equally useful?
Principle components are **not additional axes/dimensions**

**How many** PCs will be formed in **200D** space?

**First few PCs** would be **enough** to capture important information
Compute covariance matrix

\[ \text{covariance matrix} = S \]

Perform eigen decomposition

\[ S = PDP^\top \]

Compute new coordinates

\[ P^\top Z^\top = \begin{bmatrix} 2.78 & 0.81 & -0.58 & -0.81 & -2.2 \\ 0.46 & 0.58 & 0.81 & -0.58 & -0.35 \end{bmatrix} \]
Supervised Learning pipeline

1. 200D raw data
2. Normalisation (subtract mean)
3. PCA 200 PCs
4. Keep few PCs (90% variance)
5. Find the best model using CV
6. Evaluate final model on the test set

Train/test split
Safe place
Profit
Figure 6: A comparison between UMAP and t-SNE projections of a 3D woolly mammoth skeleton (50,000 points) into 2 dimensions, with various settings for parameters. Notice how much more global structure is preserved with UMAP, particularly with larger values of $n_{neighbors}$.

UMAP explained and compared to t-SNE: https://pair-code.github.io/understanding-umap/
Machine Learning

Reinforcement Learning

Classification

Regression

Supervised Learning

Unsupervised Learning

Dimensionality Reduction

Clustering
Machine Learning

- Classification
- Regression
- Supervised Learning
- Reinforcement Learning

Unsupervised Learning
- Clustering
- Dimensionality reduction
Unlabelled data

Dimensionality reduction

Unsupervised Learning

Clustering
Dimensionality reduction

Clustering

Unsupervised Learning

Unlabelled data

Clustering algorithm

Clusters

squares

circles

diamonds

triangles

Clustering methods are looking for structure in *unlabelled* data.
How come stuff does not have annotations?

(Why unsupervised learning is a thing?)
How come stuff does not have annotations?
(Why unsupervised learning is a thing?)

Labels are noisy and not trustworthy
How come stuff does not have annotations?
(Why unsupervised learning is a thing?)

Labels are noisy and not trustworthy

Labels are super expensive and laborious
How come stuff does not have annotations?
(Why unsupervised learning is a thing?)

Labels are noisy and not trustworthy

Labels are super expensive and laborious

Labels simply do not exist
How come stuff does not have annotations?
(Why unsupervised learning is a thing?)

Labels are noisy and not trustworthy

Labels simply do not exist

Labels are super expensive and laborious

We learn with very limited or no supervision
Two groups labeled by an expert.
No labels. Are there any meaningful groups?
Hierarchical clustering

At first all points are considered to be separate clusters
Hierarchical clustering

Two closest clusters are merged into one
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters
Hierarchical clustering

Two closest clusters are merged into one

These points are now belong to one cluster
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters
Now we have to **re-calculate distances** between clusters again to figure out next pair of **clusters to merge**.
Hierarchical clustering

When distance is calculated between single point cluster and multiple point cluster few options exist
Hierarchical clustering

When distance is calculated between \textbf{single point}
cluster and \textbf{multiple point cluster} few options exist

\textbf{I. Distance} between \textbf{closest points} defines the
distance between clusters (i.e. \textbf{single linkage})
Hierarchical clustering

When distance is calculated between single point cluster and multiple point cluster few options exist.

I. Distance between closest points defines the distance between clusters (i.e. single linkage).

II. Distance between furthest points defines the distance between clusters (i.e. complete linkage).
Hierarchical clustering

When distance is calculated between **single point** cluster and **multiple point cluster** few options exist.

I. Distance between closest points defines the distance between clusters (i.e. **single linkage**).

II. Distance between furthest points defines the distance between clusters (i.e. **complete linkage**).

III. Distance between averages defines the distance between clusters (i.e. **average linkage**).
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

*assume we use single linkage
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

*assume we use single linkage*
Hierarchical clustering

According to **single linkage** distance between any cluster and the new **cluster** is the distance between closest points of these clusters

*assume we use **single linkage**
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

*assume we use single linkage
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

When distances are equal, merge a pair at random

*assume we use single linkage
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

*assume we use single linkage
Hierarchical clustering

Two **closest** clusters are **merged** into one

**Euclidean** distance is calculated between all clusters

*assume we use **single linkage***
Hierarchical clustering

Two closest clusters are merged into one.

Euclidean distance is calculated between all clusters.

*assume we use single linkage
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

*assume we use single linkage
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

*assume we use single linkage
Hierarchical clustering

Two closest clusters are merged into one.

Euclidean distance is calculated between all clusters.

*assume we use single linkage*
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

*assume we use single linkage
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

*assume we use single linkage
Hierarchical clustering

Two **closest** clusters are **merged** into one.

Euclidean distance is calculated between all clusters.

*assume we use **single linkage***
Hierarchical clustering

Two closest clusters are merged into one

Euclidean distance is calculated between all clusters

*assume we use single linkage
Hierarchical clustering
Hierarchical clustering

Distance

Clusters

2 0 1 3 4 6
Hierarchical clustering

Distance

Dendrogram
Hierarchical clustering
Hierarchical clustering
Hierarchical clustering
Hierarchical clustering
Hierarchical clustering
Hierarchical clustering
Hierarchical clustering
Hierarchical clustering

Why are they on the same height?
Hierarchical clustering

Because **distances** between clusters **are the same**!

Why are they on the **same height**?

Because distances between clusters are the same!
Hierarchical clustering

Distance

Clusters

Dendrogram
Hierarchical clustering
Hierarchical clustering

These are also the same!
Hierarchical clustering

Clusters

Distance

Dendrogram
Hierarchical clustering

Distance

Clusters

Dendrogram
Hierarchical clustering

Distance

Clusters
Hierarchical clustering

What is it useful for?
Hierarchical clustering

What is it useful for?

We can **cut it**!
Hierarchical clustering

What is it useful for?

We can **cut it**!
Give me **two** clusters!
Hierarchical clustering

We can cut it!
Give me two clusters!

What is it useful for?
Hierarchical clustering

Here are your **two** clusters

We can **cut it**! Give me **two** clusters!

What is it useful for?
Hierarchical clustering

What is it useful for?

We can **cut it**!
Give me **three** clusters!

What is it useful for?
Hierarchical clustering

We can **cut it!**
Give me **three** clusters!

What is it useful for?
Hierarchical clustering

We can cut it!
Give me three clusters!

What is it useful for?
Hierarchical clustering

We can **cut it!**
Give me **three** clusters!

Here are your **three** clusters

What is it useful for?

<table>
<thead>
<tr>
<th>Distance</th>
<th>2</th>
<th>0</th>
<th>1</th>
<th>5</th>
<th>3</th>
<th>4</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>cut</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Hierarchical clustering

What is your take on?

What is it useful for?
Hierarchical clustering

Dendrogram is a way to store a structure of our data

What is it useful for?
At first **all points** are considered to be **separate clusters**. **Euclidean** distance is calculated between all clusters. Repeat as many times as there are clusters. **Two closest** clusters are **merged** into one. Until there is **one cluster** left.
At first, all points are considered to be separate clusters.

Euclidean distance is calculated between all clusters.

Agglomerative strategy:

Repeat as many times as there are clusters.

Until there is one cluster left.

Two closest clusters are merged into one.
At first all points are considered to be one cluster.

Euclidean distance is calculated between all points in clusters.

Divisive strategy

Repeat as many times as there are clusters.

Until there all points are considered separate clusters.

Two closest points in a cluster are separated apart.
Hierarchical clustering

K-means

centroids

DBSCAN

noise
Hierarchical clustering

K-means

DBSCAN

noise

centroids
K-means clustering
K-means clustering

Although they do have something in common with K-nearest neighbour, they are not the same.
All points are considered to be clusterless at first.
K-means clustering

Randomly spawn $K$ potential cluster centres
K-means clustering

Randomly spawn $K$ potential cluster centres (multiple possibilities)
K-means clustering

Randomly spawn \( K \) potential **cluster centres** (multiple possibilities)
K-means clustering

Randomly spawn \( K \) potential cluster centres (multiple possibilities)

Choose centroids randomly within dataset ranges
K-means clustering

Randomly spawn $K$ potential cluster centres (multiple possibilities)
Randomly spawn $K$ potential cluster centres (multiple possibilities)

Random cases from dataset are selected as centroids
K-means clustering

Random cases from dataset are selected as centroids

Randomly spawn \( K \) potential cluster centres (multiple possibilities)
K-means clustering

Randomly spawn $K$ potential **cluster centres** (multiple possibilities)

There are more options… (methods of initializing K-means clustering)

https://stats.stackexchange.com/questions/317493/methods-of-initializing-k-means-clustering
K-means clustering

Randomly spawn $K$ potential cluster centres (multiple possibilities)

Choose centroids randomly within dataset ranges
K-means clustering

Randomly spawn \( K \) potential cluster centres (multiple possibilities)

New centroids (cluster centres)
K-means clustering

Now as centroids were assigned we can start iterating K-means algorithm
K-means clustering

Compute distances from all points to new centroids
K-means clustering

Assign each point to the closest centroid

Compute *distances* from all points to new centroids
K-means clustering

Assign each point to the closest centroid

Compute distances from all points to new centroids
K-means clustering

Assign each point to the closest centroid

Compute distances from all points to new centroids
K-means clustering

Assign each point to the closest centroid

Compute distances from all points to new centroids
K-means clustering

Assign each point to the closest centroid

Compute distances from all points to new centroids

What about this case?
Compute distances from all points to new centroids

Assign each point to the closest centroid

Break the tie by random choice
K-means clustering

Assign each point to the closest centroid

Break the tie by random choice

Compute distances from all points to new centroids
K-means clustering

1. Compute distances from all points to new centroids.
2. Assign each point to the closest centroid.
K-means clustering

Assign each point to the closest centroid

Compute distances from all points to new centroids
K-means clustering
K-means clustering

These are our clusters so far…
Recalculate centroids for each cluster
K-means clustering

Recalculate centroids for each cluster
K-means clustering

Recalculate centroids for each cluster

\[ [2, 3, 5] \]

\[ \overline{x} = ? \]
K-means clustering

Recalculate centroids for each cluster

$[2, 3, 5]$

$\bar{x} = 3.33$
K-means clustering

\[ [2, 3, 5] \quad [5, 6, 7] \]

\[ \bar{x} = 3.33 \quad \bar{y} = ? \]

Recalculate centroids for each cluster.
K-means clustering

Recalculate centroids for each cluster

\[ \bar{x} = 3.33 \quad \bar{y} = 6 \]
K-means clustering

Recalculate centroids for each cluster

\[ \bar{x} = 3.33 \quad \bar{y} = 6 \]

New centroid for blue cluster
K-means clustering

Recalculate centroids for each cluster

\[ [3, 6, 7, 7] \]

\[ \bar{x} = 5.75 \]
K-means clustering

Recalculate centroids for each cluster

\[
\begin{align*}
\bar{x} &= 5.75 \\
\bar{y} &= 6
\end{align*}
\]
K-means clustering

Recalculate centroids for each cluster

\[ \bar{x} = 5.75 \]
\[ \bar{y} = 6 \]
K-means clustering

Recalculate centroids for each cluster

\[ [3, 6, 7, 7] \quad [4, 6, 6, 8] \]

\( \bar{x} = 5.75 \quad \bar{y} = 6 \)
Now as centroids were assigned we can start iterating K-means algorithm.
K-means clustering

Compute **distances** from all points to new centroids
K-means clustering
K-means clustering

These are our clusters so far…
K-means clustering

Recalculate **centroids** for each cluster
K-means clustering

Recalculate **centroids** for each cluster
K-means clustering

Recalculate centroids for each cluster
K-means clustering

Rinse and repeat until centroids are stable within certain radius (epsilon)
K-means clustering

Rinse and repeat until centroids are stable within a certain radius (epsilon)
K-means clustering

Rinse and repeat until centroids are stable within certain radius (epsilon)
K-means clustering

Rinse and repeat until centroids are stable within certain radius (epsilon)
K-means clustering

Rinse and repeat until centroids are stable within certain radius (epsilon)
K-means clustering

Rinse and repeat until centroids **are stable** within certain radius (epsilon)
K-means clustering

Online visualisation: https://www.naftaliharris.com/blog/visualizing-k-means-clustering/
Randomly initialise $K$ cluster centres

*K-means* algorithm

*e.g. $K = 2$*
Randomly initialise \( K \) cluster centres

Assign all points to closest cluster centres

\textbf{K-means} algorithm
Randomly initialise $K$ cluster centres

Assign all points to closest cluster centres

*K-means* algorithm

For each newly formed cluster recompute its centre
Randomly initialise $K$ cluster centres

Assign all points to closest cluster centres

*e.g. $K = 2$

Repeat until the centres are not stable within predefined radius

For each newly formed cluster recompute its centre

**K-means algorithm**
Randomly initialise $K$ cluster centres

*e.g. $K = 2$

Assign all points to closest cluster centres

Repeat until the centres are not stable within predefined radius

K-means algorithm

K-means is computationally difficult (NP-hard)*

but it has been shown to converge in finite number of steps

For each newly formed cluster recompute its centre

*https://cseweb.ucsd.edu/~avattani/papers/kmeans_hardness.pdf
In both algorithms $K$ plays an important role.
Any good way to choose $K$?
Any good way to choose $K$?

The rule of thumb is to choose $\sqrt{n/2}$ as $K$. 
Any good way to choose $K$?

The rule of thumb is to choose $\sqrt{n/2}$ as $K$.

Elbow method: increase $K$ until it does not help to describe data better.
Elbow method to predict K

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Elbow method to predict $K$

For $K = 1$:
- A single centroid is shown.

For $K = 2$:
- Two centroids are shown, indicating different clusters.

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Elbow method to predict $K$

$K = 1$

$K = 2$

$K = 3$

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Elbow method to predict \( K \)

For each case we calculate \textbf{sum of squared errors} (SSE)

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Elbow method to predict K

For each potential number \( K \) we calculate the sum of squared Euclidean distances to the closest centroid.
Elbow method to predict K

For each potential number K we calculate **sum of squared** Euclidian distances to the closest centroid.

\[
SSE = \sum_{j=1}^{k} \sum_{i=1}^{n} d(c_j, p_i)^2
\]

- Coordinates of a centroid of \( j \)th cluster
- Coordinates of a point in cluster \( j \)th
- Euclidean distance
- Number of points in \( j \)th cluster
- Number of clusters

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Elbow method to predict K

SSE: 35.42  
K = 1  
centroid

SSE: 8.16  
K = 2  
centroids

SSE: 5.33  
K = 3  
centroids

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Elbow method to predict K

SSE: 35.42

K = 1

centroid

SSE: 8.16

K = 2

centroids

SSE: 5.33

K = 3

centroids

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Elbow method to predict K

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Elbow method to predict $K$

$K = 1$

$SSE: 35.42$

$K = 2$

$SSE: 8.16$

$K = 3$

$SSE: 5.33$

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Elbow method to predict K

K = 1

K = 2

K = 3

SSE: 35.42

SSE: 8.16

SSE: 5.33

Elbow method for K-means: https://bl.ocks.org/rpgove/0060ff3b656618e9136b
Silhouette method to predict $K$
Silhouette method to predict $K$

$K = 2$
Silhouette method to predict $K$

For each point $p$ calculate: $a(p)$, $b(p)$ and $s(p)$. 

$K = 2$
Silhouette method to predict $K$

For each point $p$ calculate:
$a(p)$, $b(p)$ and $s(p)$

$a(p)$ - average distance to points that belong to the same cluster as $p$
Silhouette method to predict $K$

For each point $p$, calculate $a(p)$, $b(p)$, and $s(p)$

- $a(p)$ - average distance to points that belong to the same cluster as $p$
- $b(p)$ - minimum average distance to clusters that $p$ does not belong to
Silhouette method to predict $K$

For each point $p$ calculate:

- $a(p)$ and $b(p)$

$a(p)$ - average distance to points that belong to the same cluster as $p$

$b(p)$ - minimum average distance to clusters that $p$ does not belong to
Silhouette method to predict $K$

For each point $p$, calculate:
- $a(p)$ and $b(p)$

$a(p)$ - average distance to points that belong to the same cluster as $p$

$b(p)$ - minimum average distance to clusters that $p$ does not belong to

$K = 3$
**Silhouette** method to predict $K$

For each point $p$ calculate: $a(p)$ and $b(p)$

- $a(p)$ - average distance to points that belong to the same cluster as $p$
- $b(p)$ - minimum average distance to clusters that $p$ does not belong to

**Example:**
- Average distance from $p$ to red cluster is 0.58

$K = 3$
**Silhouette** method to predict $K$

For each point $p$ calculate: $a(p)$ and $b(p)$

- $a(p)$ - average distance to points that belong to the same cluster as $p$
- $b(p)$ - minimum average distance to clusters that $p$ does not belong to

For each point $p$

- $a(p)$ - average distance to points that belong to the same cluster as $p$
- $b(p)$ - minimum average distance to clusters that $p$ does not belong to

**Example:**
- Average distance from $p$ to **green cluster** is 0.61
- Average distance from $p$ to **red cluster** is 0.58

$K = 3$
Silhouette method to predict \( K \)

**K = 3**

For each point \( p \) calculate:  
\[ a(p) \] - average distance to points that belong to the same cluster as \( p \)

\[ b(p) \] - minimum average distance to clusters that \( p \) does not belong to

In this case \( b(p) \) is 0.58

Average distance from \( p \) to **green cluster** is 0.61

Average distance from \( p \) to **red cluster** is 0.58
Silhouette method to predict $K$

For each point $p$, calculate: $a(p)$, $b(p)$ and $s(p)$

- $a(p)$ - average distance to points that belong to the same cluster as $p$
- $b(p)$ - minimum average distance to clusters that $p$ does not belong to
Silhouette method to predict $K$

For each point $p$ calculate: $a(p)$, $b(p)$ and $s(p)$

- $a(p)$ - average distance to points that belong to the same cluster as $p$
- $b(p)$ - minimum average distance to clusters that $p$ does not belong to

$$s(p) = \frac{b(p) - a(p)}{\max\{a(p), b(p)\}}$$

Silhouette score for point $p$
Silhouette method to predict $K$

Overall silhouette score is an average of silhouette scores for all points

For each point $p$ calculate: $a(p)$, $b(p)$ and $s(p)$
**Silhouette** method to predict $K$

For each point $p$ calculate: $a(p)$, $b(p)$ and $s(p)$

Overall *silhouette* score is an average of *silhouette scores* for all points

The **higher** the total silhouette score, the **better** clustering
Any good way to choose $K$?

The rule of thumb is to choose $\sqrt{n/2}$ as $K$.

**Elbow method**: increase $K$ until it does not help to describe data better.

Overall **silhouette** score is an average of silhouette scores for all points.
Any good way to choose $K$?

No, not really.

The rule of thumb is to choose $\sqrt{n/2}$ as $K$.

Elbow method: increase $K$ until it does not help to describe data better.

Overall silhouette score is an average of silhouette scores for all points.
Hierarchical clustering

K-means

DBSCAN

Does not require K
Density-based spatial clustering of applications with noise (DBSCAN)
Two important parameters:

DBSCAN
Two important parameters:

- **$\epsilon$** (epsilon): Radius of the neighbourhood, which is searched for each point.

DBSCAN
Two important parameters:

- **epsilon**
- **minPoints**

Radius of the neighbourhood, which is searched for each point

Minimum number of points in the neighbourhood to be in cluster
DBSCAN

Pick a random point
Pick a **random** point

**Start**

**minPoints = 3**

**epsilon**

DBSCAN
DBSCAN

Pick a random point

Start

Search its $\text{epsilon}$ neighbourhood

$\text{minPoints} = 3$

$\text{epsilon}$
DBSCAN

Pick a random point

If in *neighbourhood* >= *minPoints* points, this is *cluster*

*minPoints* = 3

*epsilon*
If in neighbourhood \( \geq \text{minPoints} \) points, this is cluster (including itself)

If in neighbourhood \( \geq \text{minPoints} \) points, this is cluster (including itself)

\( \text{minPoints} = 3 \)

\( \text{epsilon} \)
DBSCAN

If in neighbourhood >= minPoints points, this is cluster (including itself)

minPoints = 3

epsilon

Start
DBSCAN

Move to the next point in the neighbourhood

minPoints = 3

epsilon

Start

X
Move to the next point in the neighbourhood
DBSCAN

Move to the next point in the neighbourhood.

minPoints = 3

epsilon
**DBSCAN**

Move to the next point in the neighbourhood

epsilon

minPoints = 3
Move to the next point in the neighbourhood

minPoints = 3

epsilon
If no points left in the neighbourhood, **pick another random point**
DBSCAN

\[
\text{minPoints} = 3
\]

\[
\text{epsilon}
\]
DBSCAN

\[
\text{minPoints} = 3
\]

\[
\text{epsilon}
\]
DBSCAN

minPoints = 3

epsilon

The figure illustrates the DBSCAN algorithm with a grid representing the dataset. Points are colored blue or red, indicating clusters or noise, respectively. The red circle (epsilon) around a point highlights the neighborhood used to define the density of points for clustering.
This point **does not have enough points**, but it is **reachable** from the point that belongs to **cluster**
This point **does not have enough points**, but it is **reachable** from the point that belongs to **cluster**.
DBSCAN

minPoints = 3

epsilon
Lonely points, are considered noise by DBSCAN

DBSCAN

epsilon

minPoints = 3

X

y
Very good visualisation: https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/
Hierarchical clustering

Parameters: **linkage type** and the **distance**

DBSCAN

Parameters: **minPoints** and **epsilon**

K-means

Parameters: **K** and the **distance**

centroids

noise
Which of these **three** we should use and when?
Hierarchical clustering

Parameters: linkage type and the distance

K-means

Parameters: K and the distance

DBSCAN

Parameters: minPoints and epsilon
Hierarchical clustering

Parameters: linkage type and the distance

Using a **dendrogram** can extract any **number** of clusters

DBSCAN

Parameters: minPoints and epsilon

K-means

Parameters: K and the distance

- centroids
- noise
Hierarchical clustering

Hierarchical clustering is very slow

Using a **dendrogram** can extract any **number** of clusters

Parameters: **linkage type** and the **distance**

---

K-means

Parameters: **K** and the **distance**

---

DBSCAN

Parameters: **minPoints** and **epsilon**
Hierarchical clustering

Hierarchical clustering is very slow \((N^3)\)

Using a **dendrogram** can extract any number of clusters

Parameters: linkage type and the distance

K-means

Parameters: \(K\) and the distance

DBSCAN

Parameters: minPoints and epsilon
Hierarchical clustering

Hierarchical clustering is very slow ($N^3$)

Using a dendrogram can extract any number of clusters

Parameters: linkage type and the distance

K-means

Parameters: $K$ and the distance

Hierarchical clustering is very slow ($N^3$)

Parameters: linkage type and the distance

DBSCAN

Parameters: $\text{minPoints}$ and $\text{epsilon}$
Hierarchical clustering

Using a **dendrogram** can extract **any** number of clusters.

- Parameters: **linkage type** and the **distance**

Hierarchical clustering is **very slow** ($N^3$)

K-means

- You will have **exactly** $K$ clusters.

- Parameters: $K$ and the **distance**

DBSCAN

- Parameters: **minPoints** and **epsilon**

- **noise**

- **centroids**

You will have exactly $K$ clusters.
Hierarchical clustering

- Parameters: linkage type and the distance
- Using a dendrogram can extract any number of clusters
- Hierarchical clustering is very slow ($N^3$)

K-means

- Parameters: K and the distance
- You need to know K
- You will have exactly K clusters

DBSCAN

- Parameters: minPoints and epsilon
- Noise is included
- Centroids are determined
Hierarchical clustering

Parameters:
- linkage type
- distance

Using a dendrogram can extract any number of clusters.

Hierarchical clustering is very slow ($N^3$).

K-means

Parameters:
- K
- the distance

You need to know K.

You will have exactly K clusters.

Although NP-hard, it works surprisingly fast.

DBSCAN

Parameters:
- minPoints
- epsilon

Can detect noise.
Hierarchical clustering is very slow ($N^3$).

Using a dendrogram can extract any number of clusters.

K-means requires you to know $K$.

You will have exactly $K$ clusters.

Although NP-hard, it works surprisingly fast.

DBSCAN does not depend on $K$.

Parameters are tedious to pick up.

Parameters: $\text{minPoints}$ and $\epsilon$. 

Parameters: $K$ and the distance.
References

• Machine Learning by **Andrew Ng** ([https://www.coursera.org/learn/machine-learning](https://www.coursera.org/learn/machine-learning))

• Curse of dimensionality: [http://cleverowl.uk/2016/02/06/curse-of-dimensionality-explained/](http://cleverowl.uk/2016/02/06/curse-of-dimensionality-explained/)


• Stanford CS class: Convolutional Neural Networks for Visual Recognition by **Andrej Karpathy** ([http://cs231n.github.io/](http://cs231n.github.io/))

• Data Mining Course by **Jaak Vilo** at University of Tartu ([https://courses.cs.ut.ee/MTAT.03.183/2017_spring/uploads/Main/DM_05_Clustering.pdf](https://courses.cs.ut.ee/MTAT.03.183/2017_spring/uploads/Main/DM_05_Clustering.pdf))
What time is it now?
That's all Folks!