Data Science Bootcamp

Hyperiondev

An Overview of Unsupervised Learning

Welcome

Your Lecturer for this session

Sanana Mwanawina

Lecture - Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly.
- No question is daft or silly - **ask them!**
- There are Q/A sessions midway and at the end of the session, should you wish to ask any follow-up questions.
- You can also submit questions here: hyperiondev.com/sbc4-ds-questions
- For all non-academic questions, please submit a query: hyperiondev.com/support
- Report a safeguarding incident: hyperiondev.com/safeguardreporting
- We would love your feedback on lectures: https://hyperionde.wufoo.com/forms/zsgv4m40ui4i0g/

Lecture - Code Repo

Go to: github.com/HyperionDevBootcamps

Then click on the "**C4_DS_lecture_examples**" repository, do view or download the code.

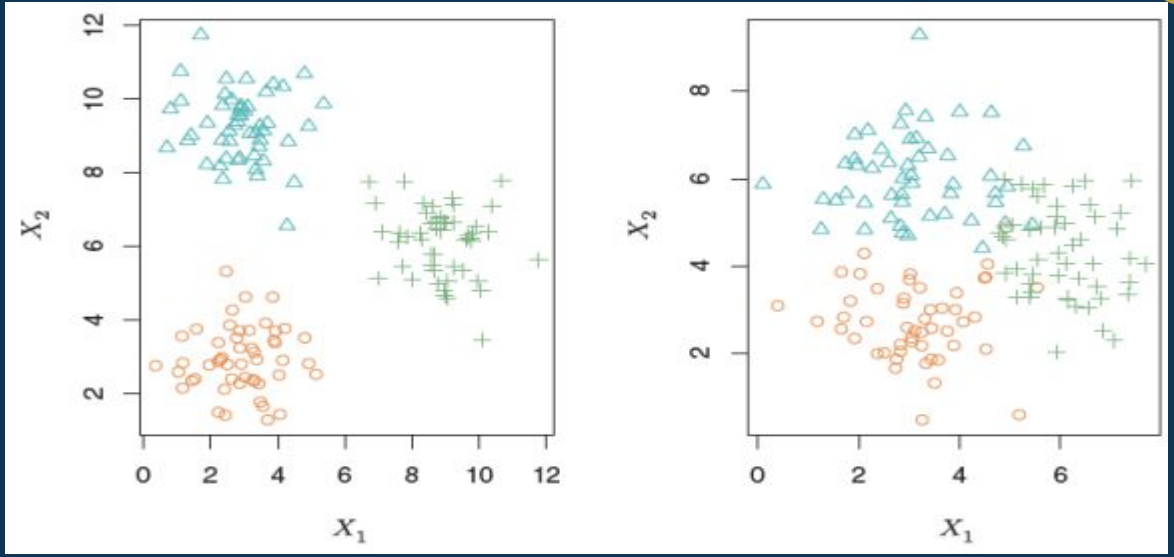
Objectives

- Learn about unsupervised learning problems, where we only observe input variables
- Introduce an unsupervised methods

Introduction to Clustering

- ★ Previously, when supervised learning was covered, it involved datasets with both input and output variables.
- ★ However, we will take a look at another class of problems, unsupervised learning problems, where only the input variables are observed.
- ★ Here we will look at the unsupervised method : clustering.

Introduction to Clustering



The graphs above show two datasets that are good candidates for applying clustering. The data on the left shows a clear grouping that a clustering algorithm could readily identify for us. The right hand side data groups with more overlap and will be harder to identify, but still suitable for a clustering approach, rather than using linear regression for example.

Introduction to Clustering

- If a clustering approach seems suitable, we can use the cluster analysis to ascertain, on the basis of our input variables whether the observations fall into relatively distinct groups by asserting that observations within a group are similar to each other, while observations in different groups are different from each other.

K-Means Clustering

- ★ K-Means clustering is the most well-known clustering algorithm. It is a *simple* and elegant approach for partitioning a dataset into K distinct clusters. To perform K-Means clustering, we first specify the desired number of clusters, K, and then assign each observation to exactly one of the K clusters.

Feature Space

- ★ There are a number of different distance metrics that are used in algorithms to decide how similar observations are.
- ★ The most common one is the Euclidean distance.

$$(x_i, y_i) \text{ and } (x_j, y_j) \text{ is } \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}.$$

- ★ To compute the mean of a number of observations, we divide the sum of the observations by the number of observations
- ★ To compute the mean of a certain number of (x,y) points, we compute the mean of all x values and the mean of all y values

The K-means Algorithm

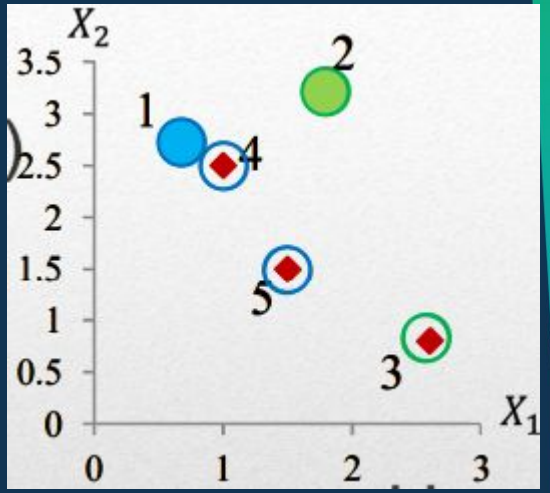
- The K-means algorithm follows the following steps :
  - Select number of clusters , K
  - Select random points from the data as starting values and initialise the mean of each cluster.
  - For *n* number of iterations :
    - Assign each point to the cluster whose mean (or “centroid”) is the nearest.
    - Re-compute the means for each cluster based on its current members.
  - Repeat steps in 3 until convergence.

K-means Algorithm

K = 2

$$X = \begin{bmatrix} 0.7 & 2.7 \\ 1.8 & 3.2 \\ 2.6 & 0.8 \\ 1.0 & 2.5 \\ 1.5 & 1.5 \end{bmatrix}$$

	Obj1	Obj2
Obj1		
Obj2	1.21	
Obj3	2.69	2.53
Obj4	0.36	1.06
Obj5	1.44	1.73

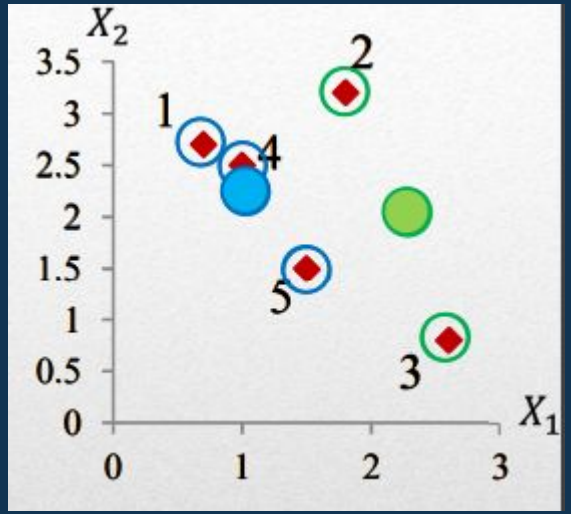


K-means Algorithm

Calculate the new centroids

$$X = \begin{bmatrix} 0.7 & 2.7 \\ 1.0 & 2.5 \\ 1.5 & 1.5 \end{bmatrix} \text{ and } X = \begin{bmatrix} 1.8 & 3.2 \\ 2.6 & 0.8 \end{bmatrix}$$

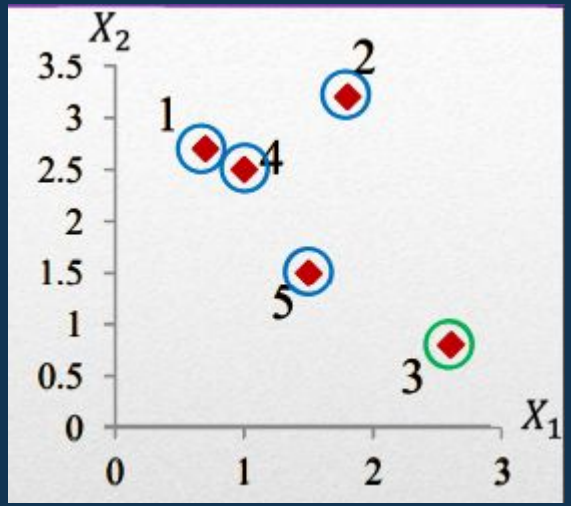
$$[1.07 \quad 2.23] \text{ and } [2.20 \quad 2.00]$$



K-means Algorithm

Find nearest centroid for each observation

	1	2	3	4	5
Blue	0.60	1.21	2.09	0.28	0.85
Green	1.66	1.26	1.26	1.30	0.86

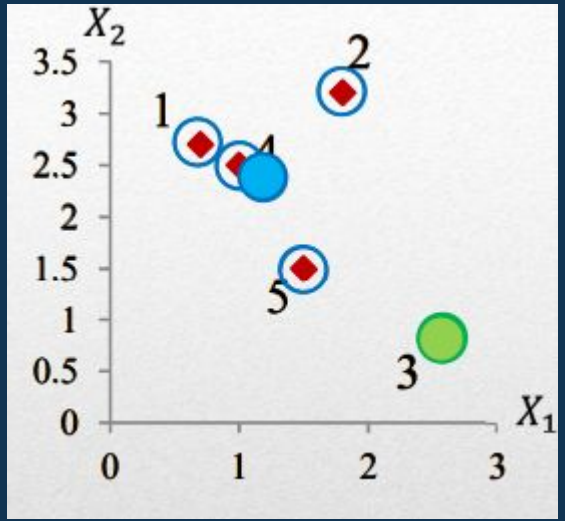


K-means Algorithm

Calculate new centroids

$$X = \begin{bmatrix} 0.7 & 2.7 \\ 1.8 & 3.2 \\ 1.0 & 2.5 \\ 1.5 & 1.5 \end{bmatrix} \text{ and } X = [2.6 \quad 0.8]$$

[1.25 2.48] and [2.6 0.8]

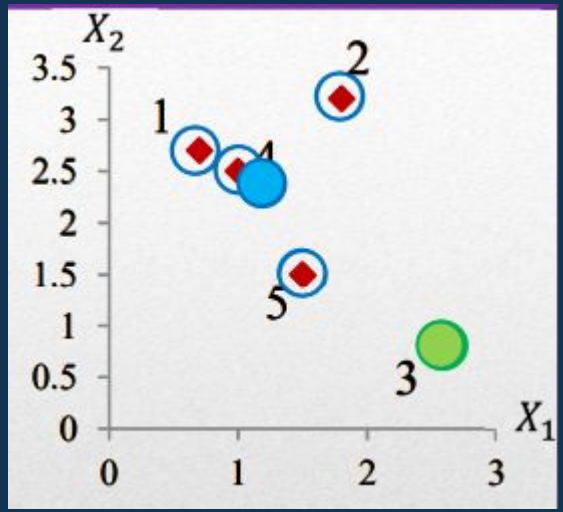


K-means Algorithm

Find distances to nearest centroid

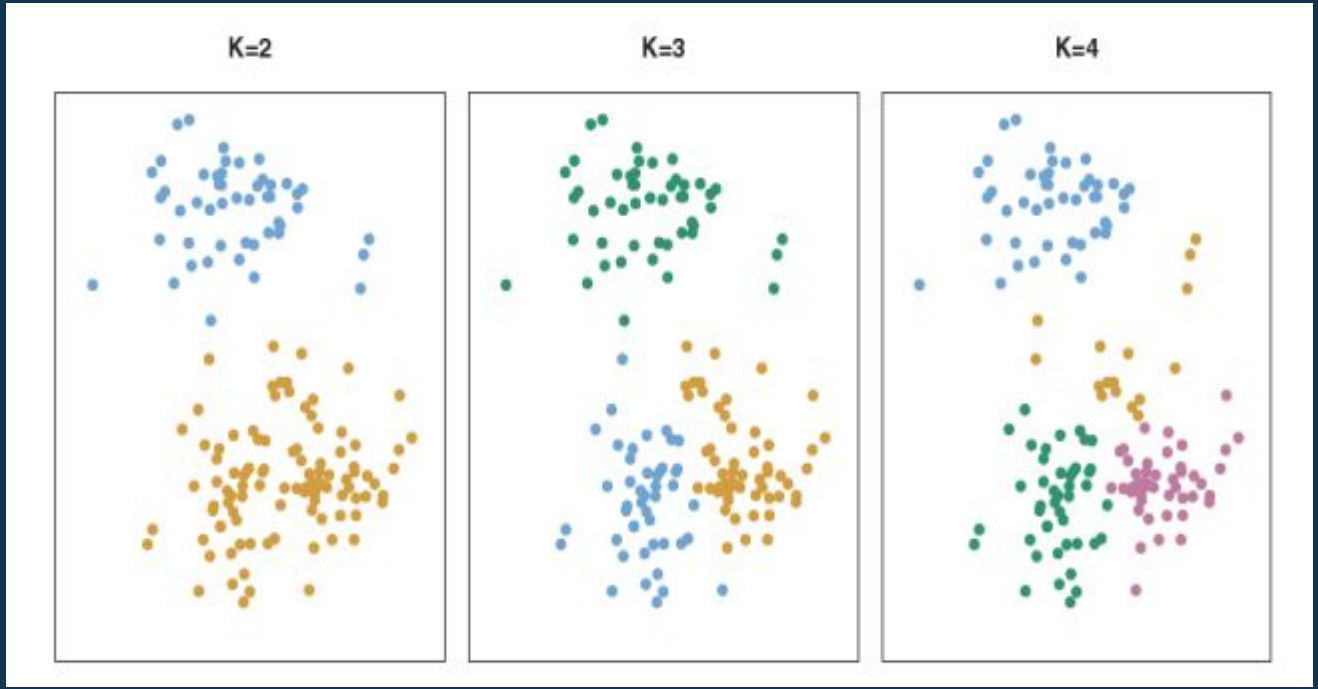
Unchanged => converged

	1	2	3	4	5
Blue	0.59	0.91	2.15	0.25	1.01
Green	2.69	2.53	0.00	2.33	1.30



Choosing K

- ★ An important thing to consider when applying K-means is that we need to choose a value for K before running the analysis.
- ★ Choosing K will greatly affect the outcome and accuracy of the clusters.
- ★ The following plots displays different outcomes of the algorithm depending on the value chosen for K.


K=2
K=3
K=4

Validating the clusters

- ★ It's possible to find clusters in any data, but it is important to determine if these clusters actually represent underlying subgroups in the data or are merely grouping with similar noise.

- ★ This is a very hard question to answer. There exist a number of techniques for assigning a significance value to a cluster in order to assess whether there is more evidence for the cluster than one would expect due to chance. However, there has been no consensus on a single best approach. The Silhouette Coefficient (sklearn.metrics.silhouette_score) is an example of an evaluation metric which indicates how similar samples within a cluster are, compared to other clusters. A higher Silhouette Coefficient score relates to a model with better-defined clusters.

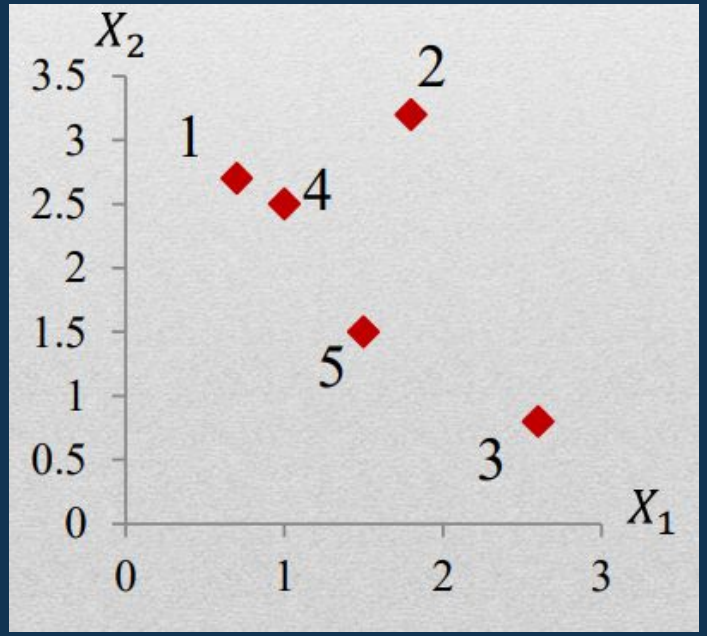
Hierarchical clustering

- ★ Agglomerative:
  1. Start with each observation in its own cluster
  2. Merge the two closest clusters
  3. Continue until all objects are in a single cluster

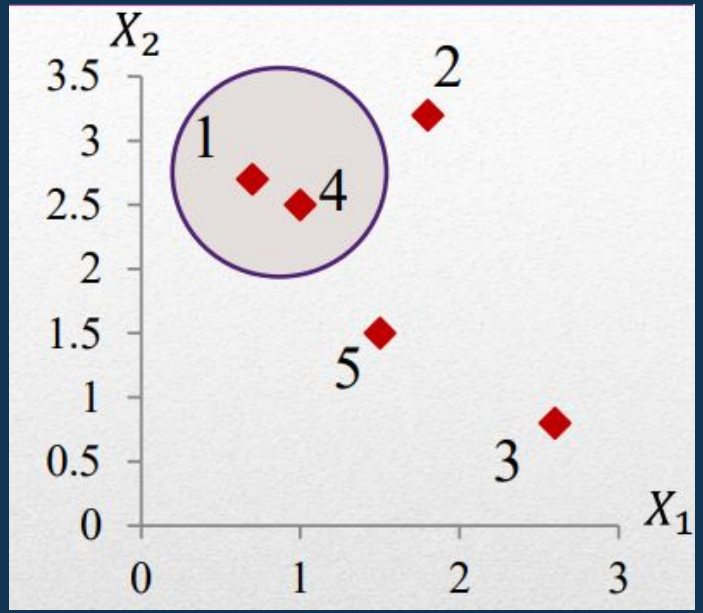
- ★ Different ways of measuring the distance between clusters
  1. Single linkage (nearest neighbour)
  2. Complete linkage (farthest neighbour)
  3. Centroid method

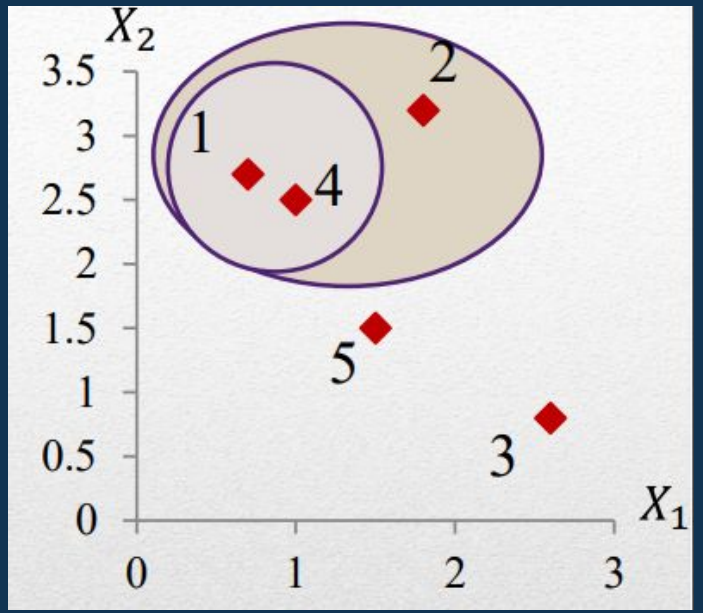
Hierarchical clustering

Create clusters using the single linkage clustering algorithm.

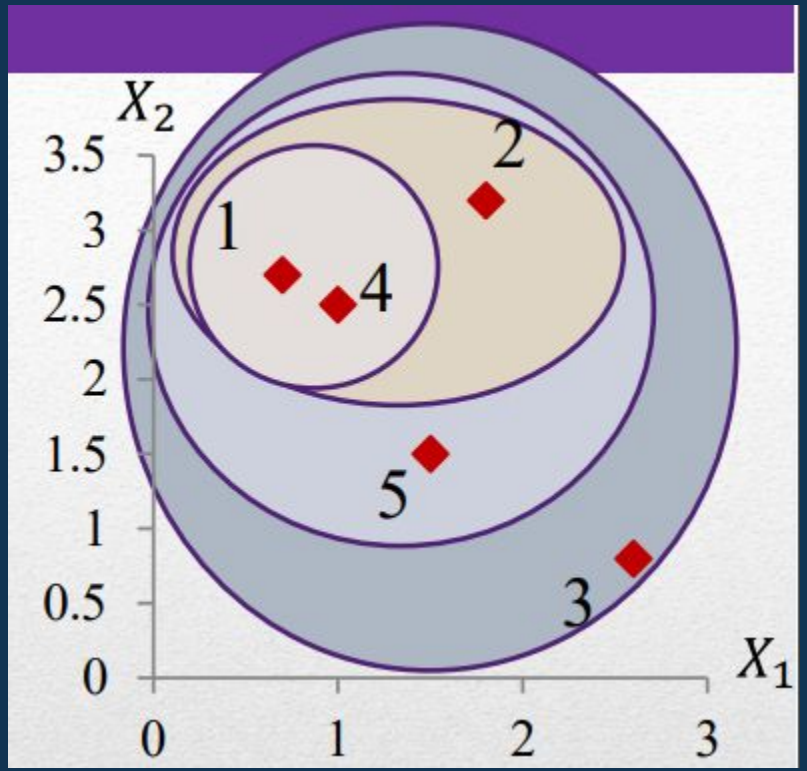


Hierarchical clustering



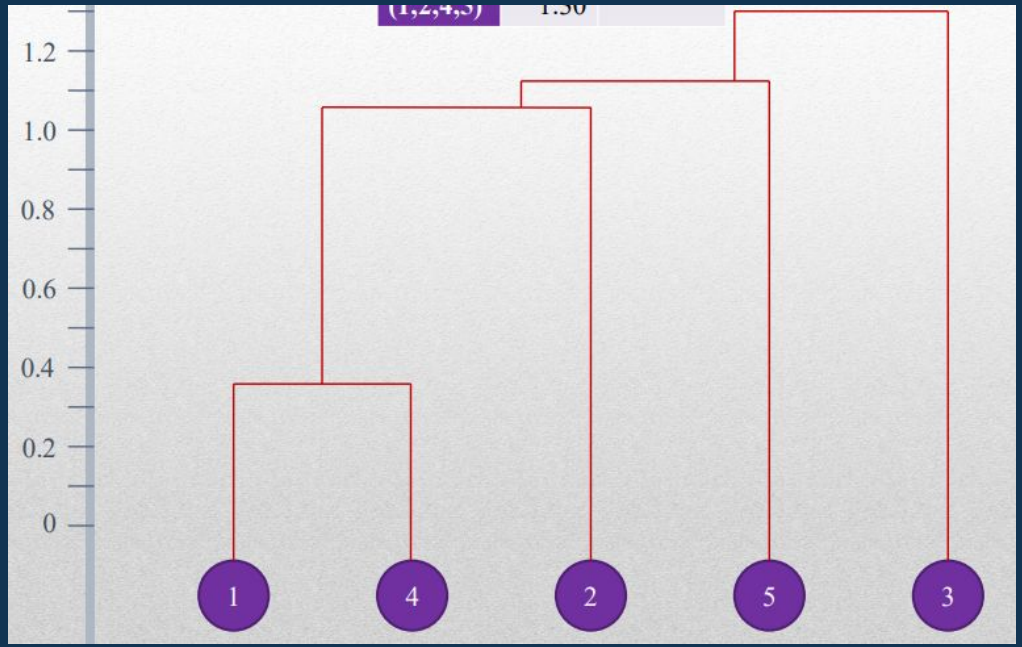


Hierarchical clustering



The dendrogram

★ We are looking for the most dissimilar clusters. The larger the distance between clusters, the more dissimilarity there is. The y axis on the dendrogram represents distance.



Principal Component Analysis

- ★ So far the problems we have looked at have had only a handful of input variables. In practice, the number of variables in a machine learning task is usually higher.

- ★ As the number of variables grows, the data becomes harder to work with. Relationships between variables become harder to see, training slows down, and the chance of overfitting increases. It is, therefore, useful to know a bit about how to reduce the number of variables while still retaining enough information about our dataset.

Principal Component Analysis

- Principal component analysis is a popular technique for analysing large datasets containing a high number of dimensions/features per observation, increasing the interpretability of data while preserving the maximum amount of information.

How it works

Using Singular Value Decomposition (svd). From the svd we can obtain the eigenvectors and eigenvalues.

- ➔ The eigenvectors are the directions in which the data is most spread out, and the eigenvalue describes how spread out the data is (how much variance there is) in that direction.
- ➔ PCA then selects the eigenvectors with the highest eigenvalues and projects the data points into a space that has only those eigenvectors as its dimensions.

Example

A company wants to assess the performance of their sales team. They have their 50 employees sit for 7 assessments. We are interested in two things:

1. Who is the overall best employee?
2. Putting employees with similar strengths in the same teams

Salesperson	Sales growth	Sales profitability	New account sales	Creativity test	Mechanical reasoning test	Abstract reasoning test	Mathematics test
1	93.0	96.0	97.8	9	12	9	20
2	88.8	91.8	96.8	7	10	10	15
3	95.0	100.3	99.0	8	12	9	26
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Mean	98.84	106.62	102.81	11.22	14.18	10.56	29.76
SD	7.34	10.12	4.71	3.95	3.38	2.14	10.54

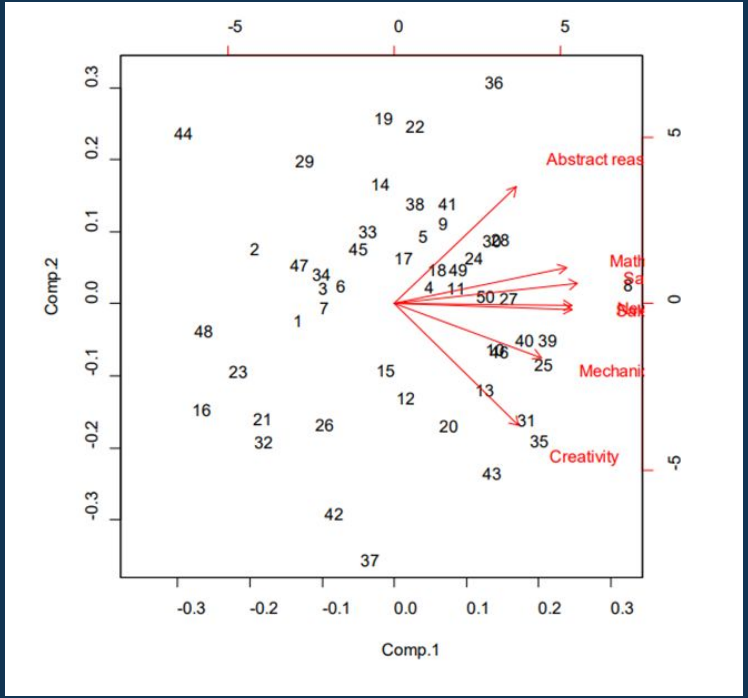
Example

Question 1 can be answered by ranking PCA scores.

	[,1]	[,2]
[1,]	-2.234315487	9
[2,]	-3.169056239	5
[3,]	-1.667240533	13
[4,]	0.671046196	28
[5,]	0.533133765	27
[6,]	-1.273143192	16
[7,]	-1.634727198	14
[8,]	5.038652862	50
[9,]	0.979901939	31
[10,]	2.012358414	41
[11,]	1.143742384	33
[12,]	0.035286745	24
[13,]	1.782278948	36
[14,]	-0.505582219	20
[15,]	-0.400990637	22
[16,]	-4.443828264	2
[17,]	-0.005341906	23
[18,]	0.733375088	29
[19,]	-0.435684210	21
[20,]	0.980282775	32
[21,]	-3.105912280	6
[22,]	0.252147719	25
[23,]	-3.631763098	4
[24,]	1.543487218	35
[25,]	3.068300900	48
[26,]	-1.732918197	12
[27,]	2.314075763	44
[28,]	2.125272796	43
[29,]	-2.168307899	10
[30,]	1.943925250	38
[31,]	2.698051620	46
[32,]	-3.069572927	7
[33,]	-0.796976666	18
[34,]	-1.802032715	11
[35,]	2.981069301	47
[36,]	1.985656789	40
[37,]	-0.753383345	19
[38,]	0.254844025	26
[39,]	3.158143577	49
[40,]	2.649495913	45
[41,]	0.967714920	30
[42,]	-1.535080175	15
[43,]	1.950114283	39
[44,]	-4.840207295	1
[45,]	-1.007674028	17
[46,]	2.091768767	42
[47,]	-2.287658850	8
[48,]	-4.387707804	3
[49,]	1.202334231	34
[50,]	1.792642978	37

Example

Question 2 can be answered using a biplot.



Hyperiondev

Why not just take the average?

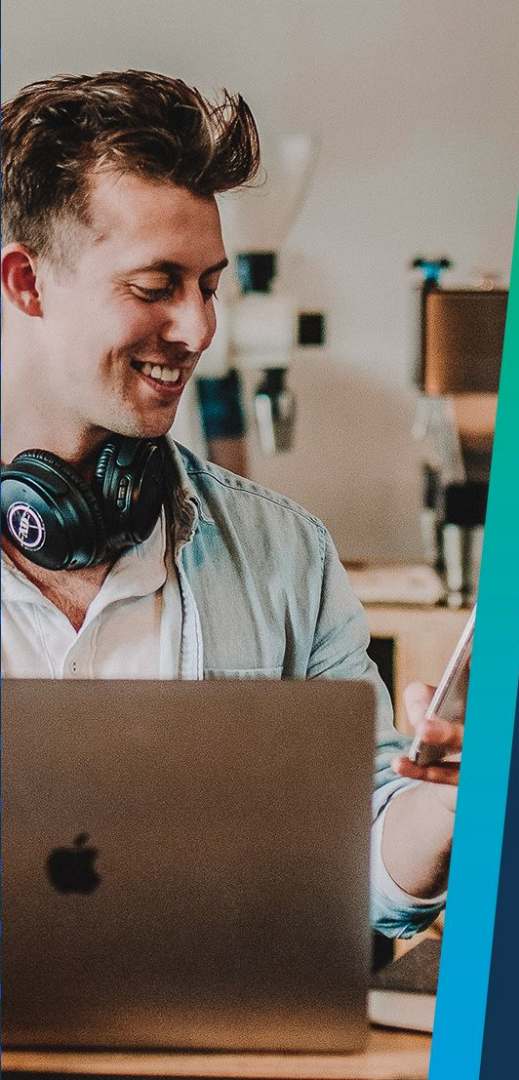
Taking the average can be useful, but PCA serves a different (and powerful) purpose:

- ★ Dimension reduction
- ★ We can capture the relationships between variables. We cannot do that when taking the average
- ★ Data visualization which we can use to identify clusters, outliers etc.

Hyperiondev

Q & A Section

Please use this time to ask any questions relating to the topic explained, should you have any



Hyperiondev

Thank you for joining us