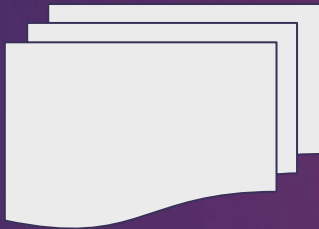


# *DAY-4* *Unsupervised* *Learning*

*Presented by*  
*ANJU LATHA NAIR S*



# CONTENTS



- ▶ Clustering
- ▶ K-means Algorithm
- ▶ Dimensionality Reduction
- ▶ Principal Component Analysis
- ▶ KNN as an unsupervised learning technique

# Unsupervised Learning

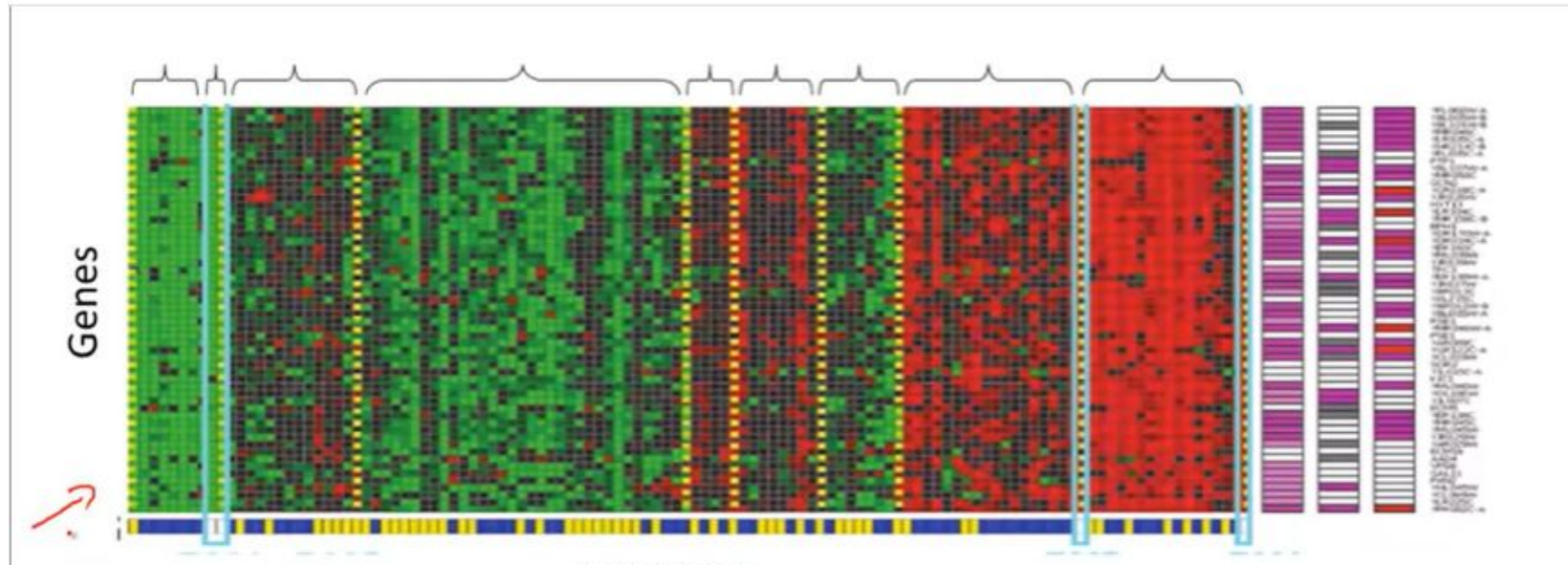
- ▶ Conceptually different from Supervised Learning
- ▶ In many circumstances, targets are difficult to obtain
- ▶ Can we perform Regression?.....  
(predict a continuous valued output)
- ▶ What about Classification???.....  
(predict a discrete value output)
- ▶ There is NO information about the correct outputs available.

## *Unsupervised Learning contd..*

- ▶ Allows us to approach problems with little or no idea what our results should look like.
- ▶ We can derive structure from data where we don't necessarily know the effect of the variables.
- ▶ We can derive this structure by **clustering** the data based on relationships among the variables in the data.
- ▶ There is no feedback based on the prediction results.

# Clustering problem- 1

- ▶ Put a group of different individuals and for each of them, you measure how much they do or do not have a certain gene.
- ▶ Technically you measure how much certain genes are expressed.
- ▶ So these colors, red, green, gray and so on, show the degree to which different individuals do or do not have a specific gene.



- ▶ I don't know what's in this data.
- ▶ I don't know who's and what type.
- ▶ I don't even know what the different types of people are,
- ▶ But can you automatically find structure in the data from the you automatically cluster the individuals into these types that I don't know in advance?
- ▶ So this is Unsupervised Learning because we're **not** telling the algorithm in advance

that these are type 1 people, those are type 2 persons, those are type 3 persons and so

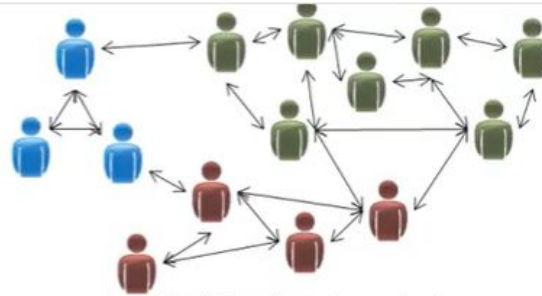
on and instead what were saying is “**yeah here's a bunch of data**”.



# Some Applications



Organize computing clusters



Social network analysis



Market segmentation



Astronomical data analysis



Machine Learning

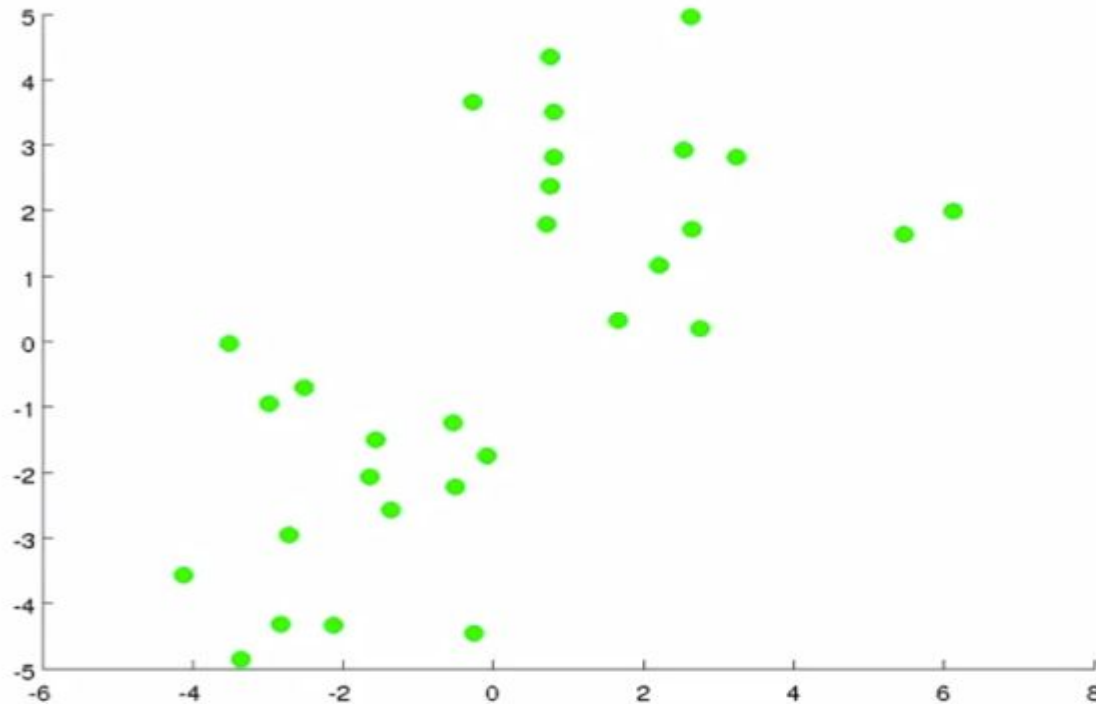
# Clustering

## K-means algorithm

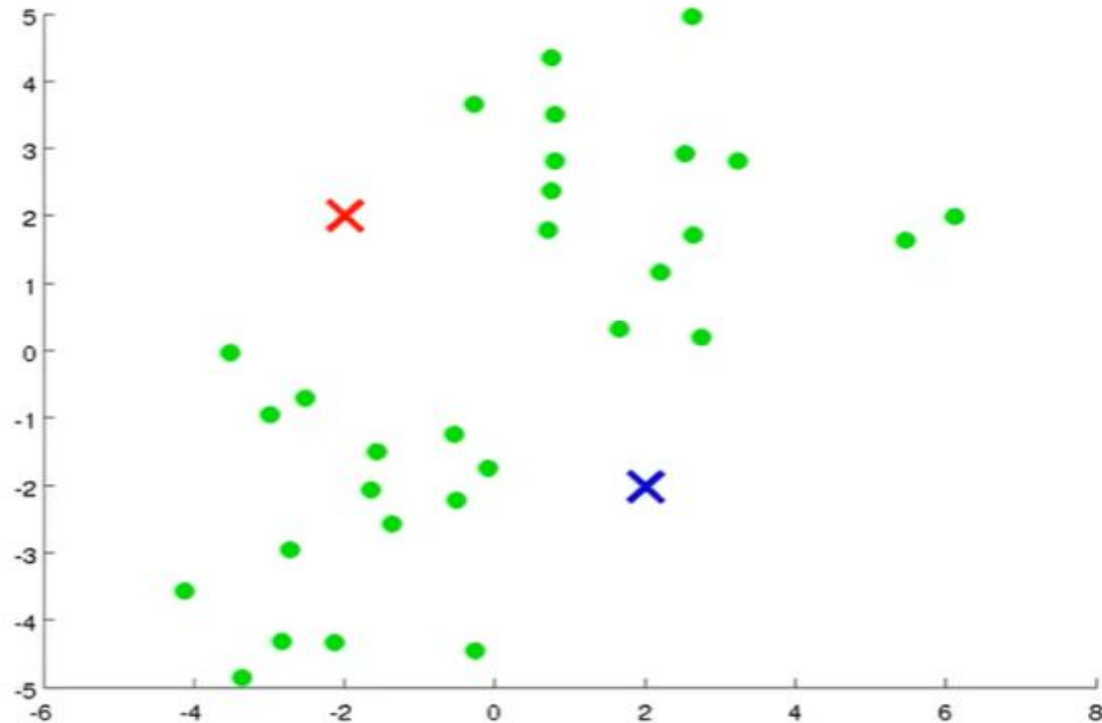


# Clustering Problem

- ▶ Given an unlabeled dataset, we would like to have an algorithm automatically group the data into coherent clusters for us.



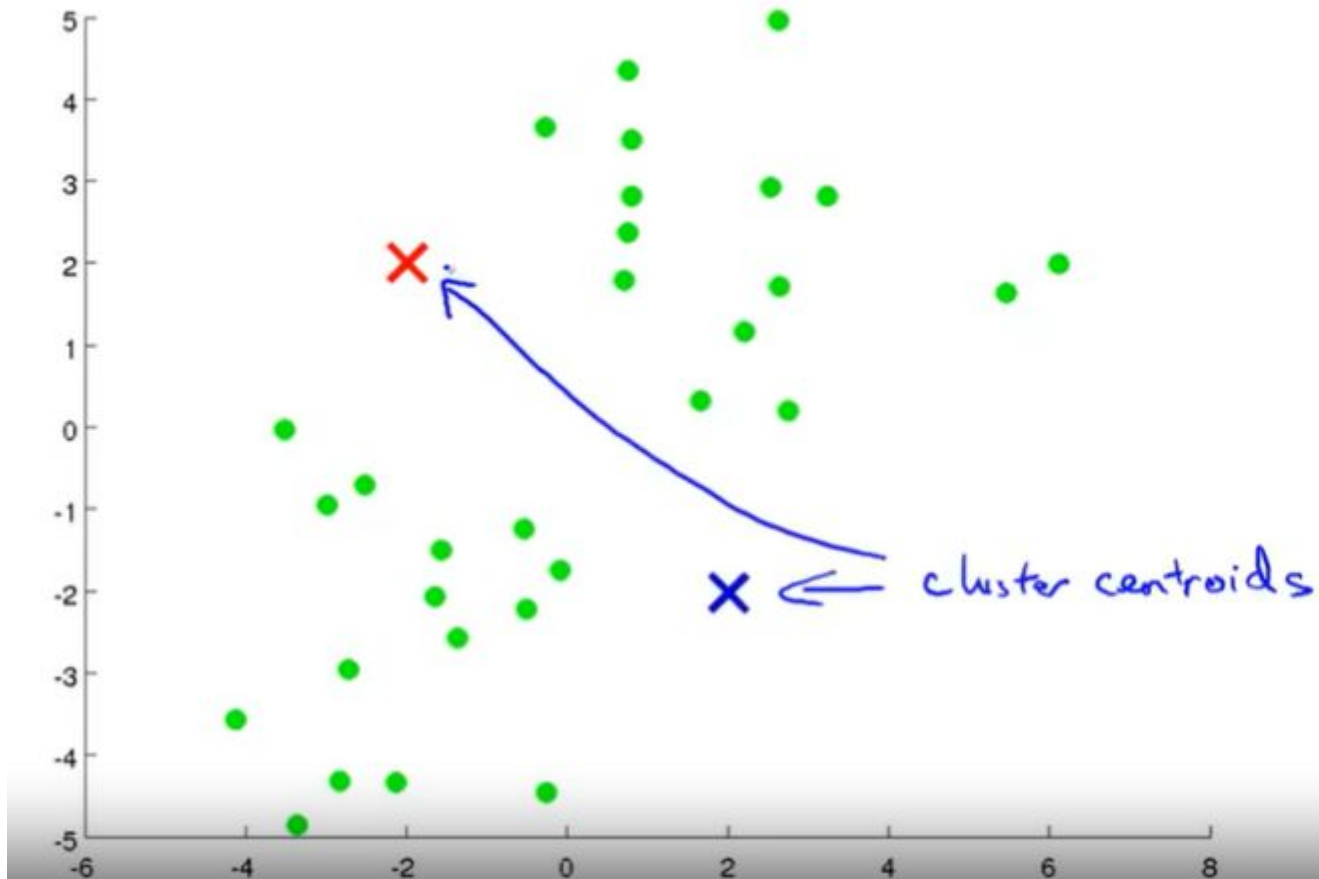
- ▶ Randomly initialize two points (cluster centroids)

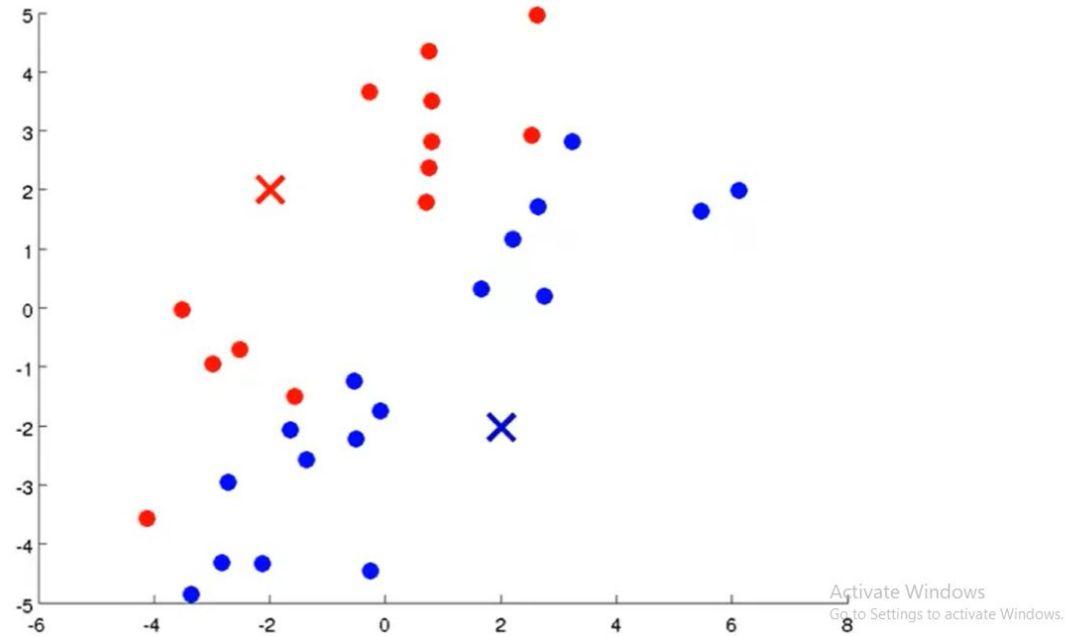


# K-Means

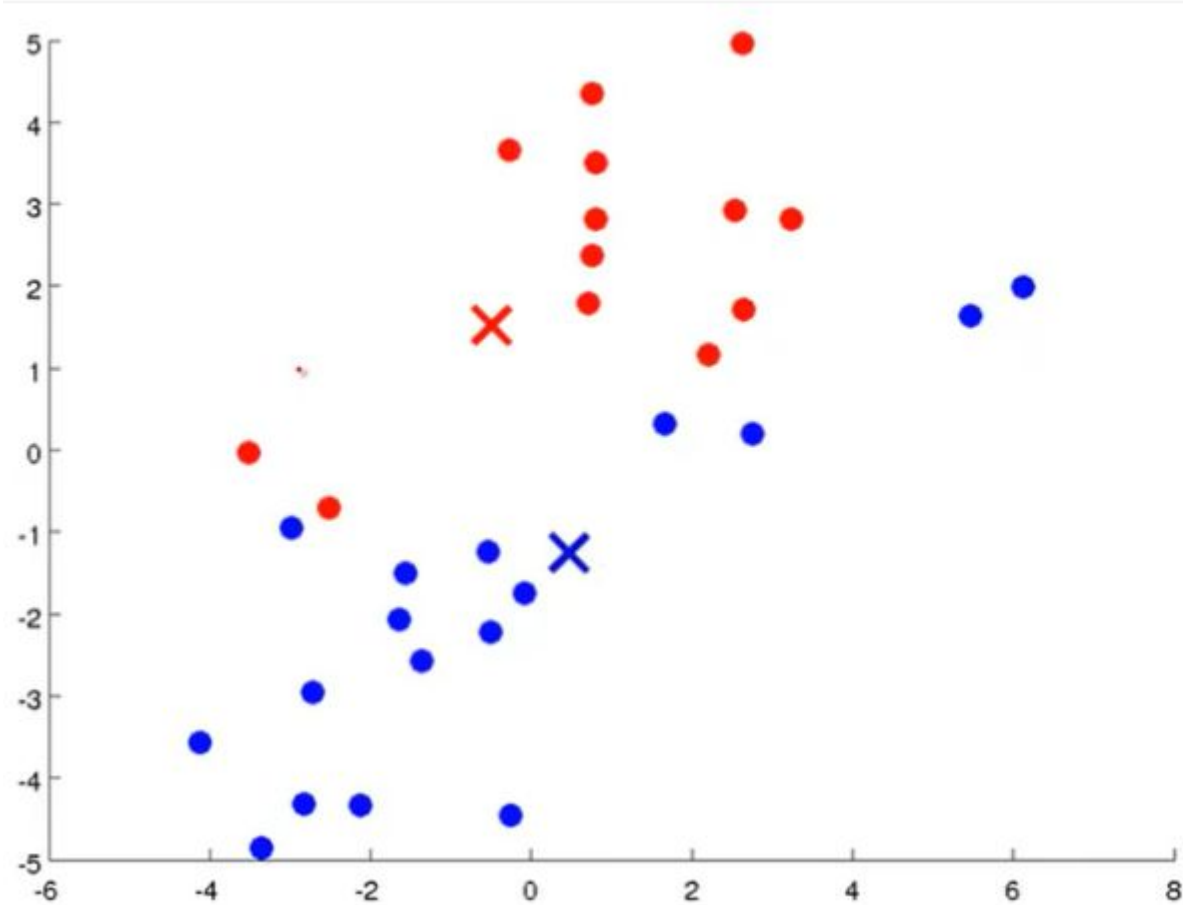
- ▶ Two steps:
  1. Cluster assignment
  2. Move Centroid

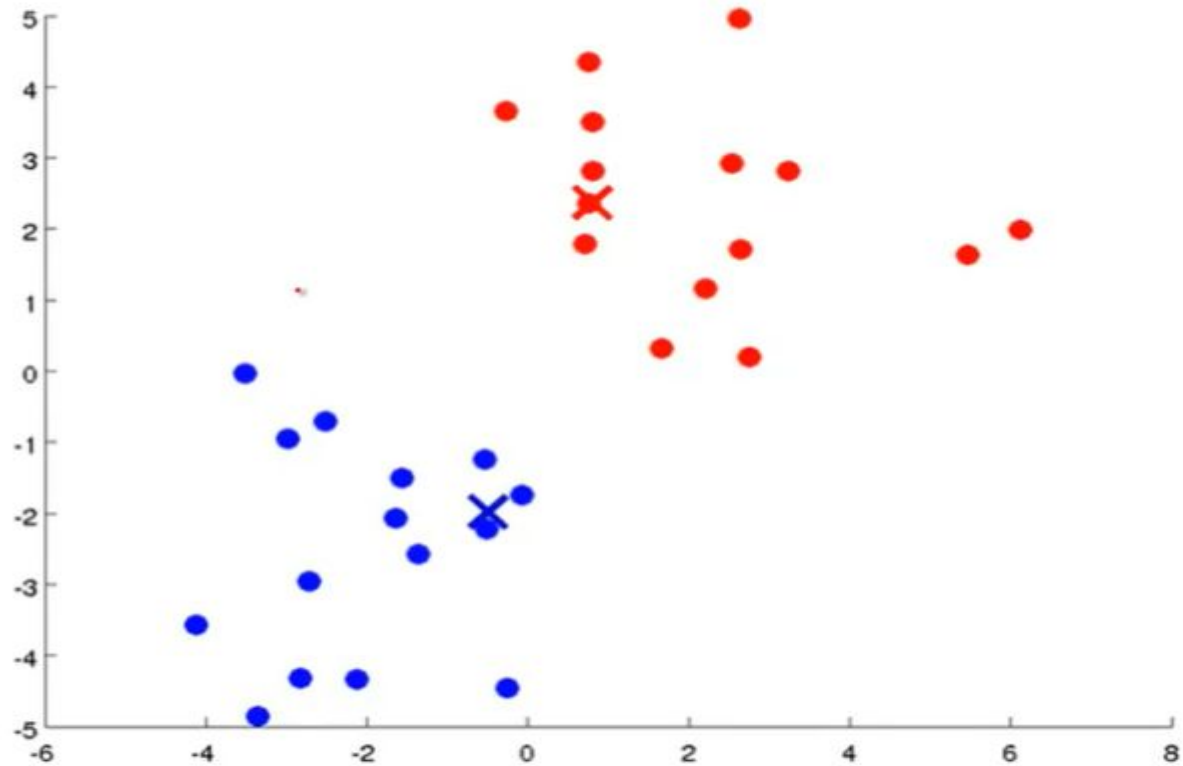
Looking at the unlabeled samples, assign red or blue to each of the points





Courtesy: Andrew N. G








# K- means summarized as .....

- ▶ Compute the average of the red points
- ▶ Compute the average of the blue points
- ▶ Do the cluster assignment step once again
- ▶ Repeat
- ▶ At some point, even if we do more number of iterations, the cluster centroids will not change any further and the colours of the points will not change any further
- ▶ i.e, K means have converged.

# K-means... in a formal way

## K-means algorithm

Input:

- $K$  (number of clusters) 
- Training set  $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$

$x^{(i)} \in \mathbb{R}^n$  (drop  $x_0 = 1$  convention)

## K-means algorithm

Randomly initialize  $K$  cluster centroids  $\mu_1, \mu_2, \dots, \mu_K \in \mathbb{R}^n$

Repeat {

  for  $i = 1$  to  $m$

$c^{(i)} :=$  index (from 1 to  $K$ ) of cluster centroid  
    closest to  $x^{(i)}$

  for  $k = 1$  to  $K$

$\mu_k :=$  average (mean) of points assigned to cluster  $k$

}

# Applications of K-Means

- ▶ Optical character Recognition
- ▶ Biometrics
- ▶ Diagnostic systems
- ▶ Military Applications
- ▶ Image Compression

# DIMENSIONALITY REDUCTION

- ▶ When looking at data and plotting results, we can never go beyond three dimensions in our data, and usually find two dimensions easier to interpret.
- ▶ The higher the number of dimensions we have, the more training data we need.
- ▶ Explicit factor for the computational cost of many algorithms
- ▶ Reducing It can reduce noise,
- ▶ Make the dataset easier to work with
- ▶ Easier to understand and interpret

# Ways to do Dimensionality Reduction

## 1. Feature Selection

Looking through the features that are available and seeing whether or not they are actually useful

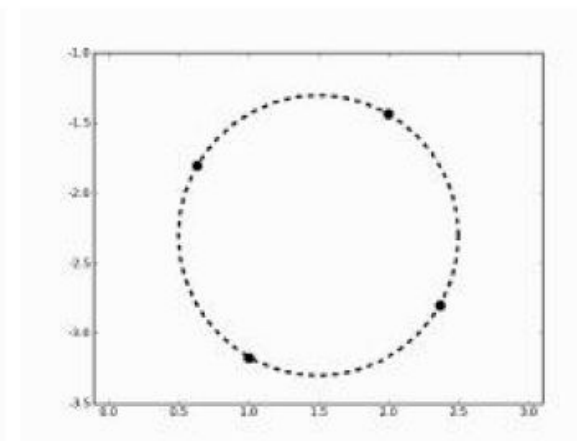
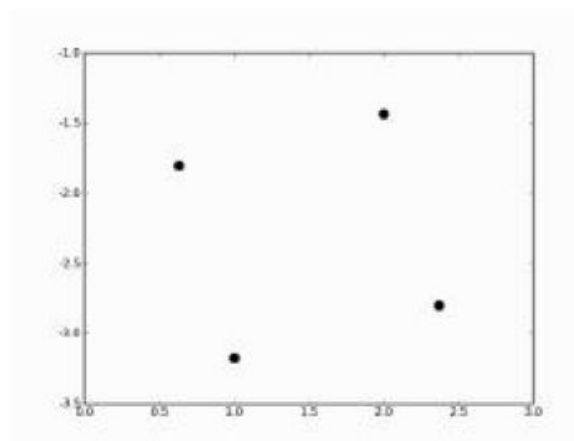
## 2. Feature Derivation

Deriving new features from the old ones, generally by applying transforms to the dataset

## 3. Clustering

Group together similar datapoints, and to see whether this allows fewer features to be used

$x$	$y$
2.00	-1.43
2.37	-2.80
1.00	-3.17
0.63	-1.80





### **Constructive method:**

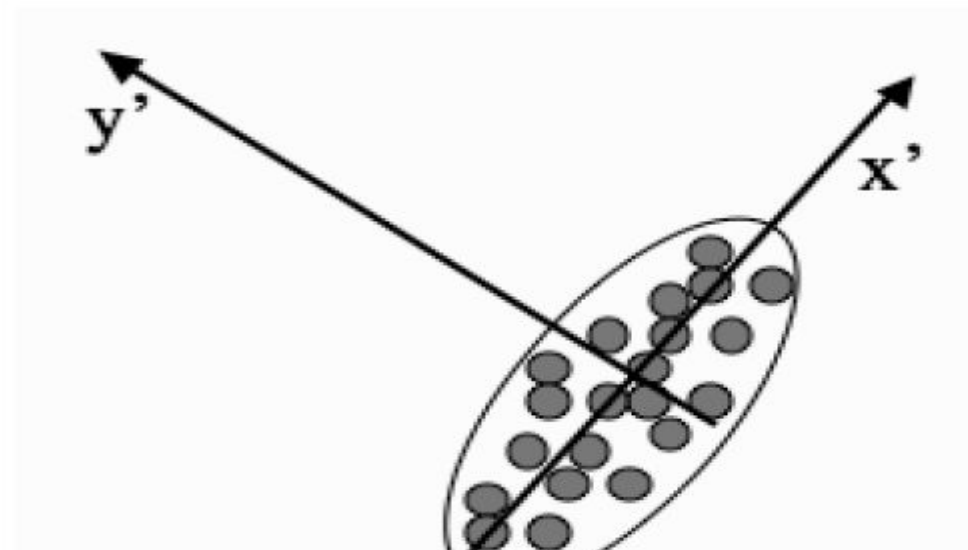
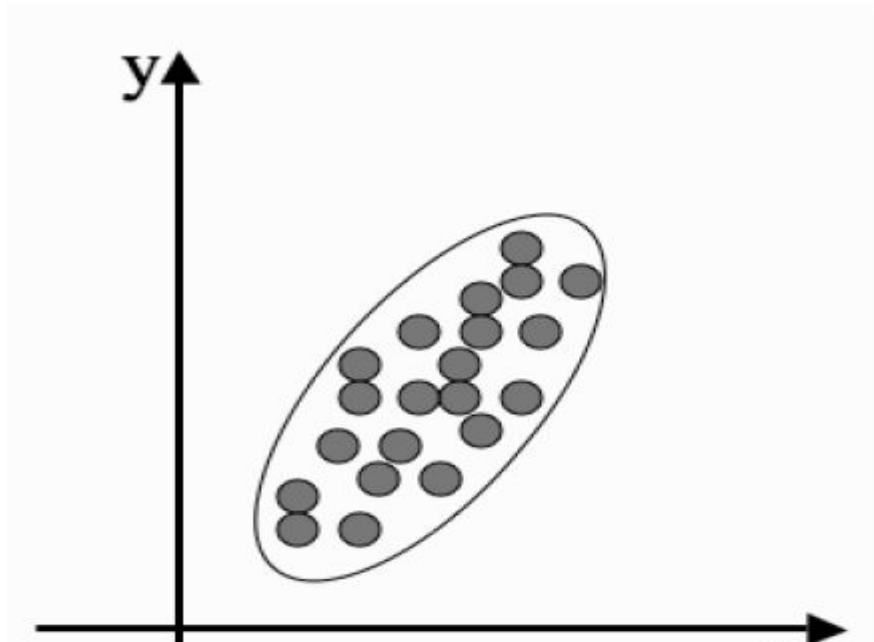
- ▶ At each stage of the algorithm it decides which feature to add next
- ▶ Start with none, and then iteratively add more, testing the error at each stage to see whether or not it changed at all when that feature was added.

### **Destructive method**

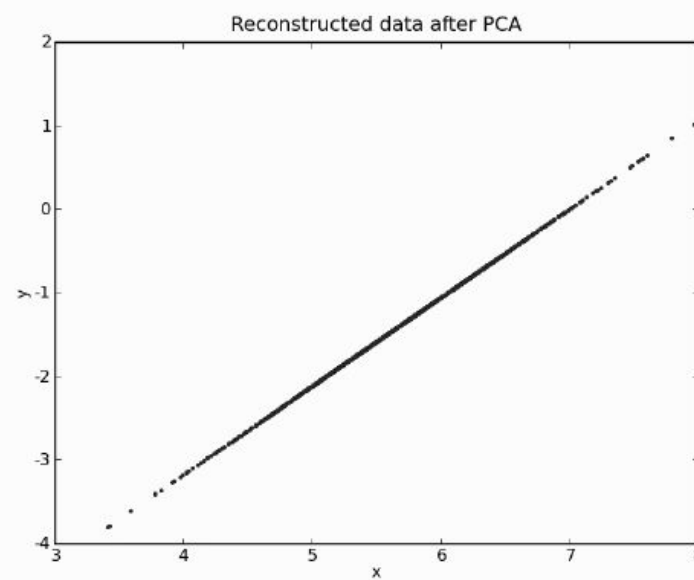
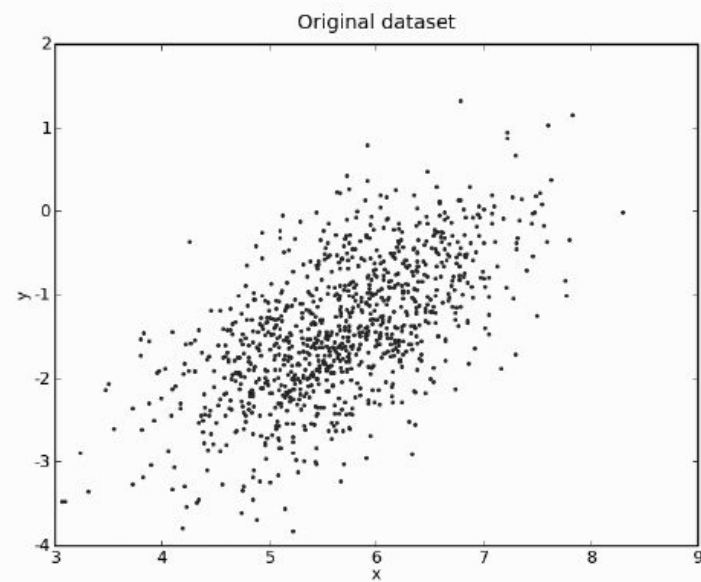
- ▶ Prune the decision tree, lopping off branches and checking whether the error changed

# Principal Component Analysis

- ▶ By finding particular sets of coordinate axes, it will become clear that some of the dimensions are not required.



- ▶ A principal component is a direction in the data with the largest variation.
- ▶ Centres the data by subtracting off the mean
- ▶ Chooses the direction with the largest variation and places an axis in that direction
- ▶ looks at the variation
- ▶ Finds another axis that is orthogonal to the first and covers as much of the remaining variation as possible.
- ▶ Iterates this until it has run out of possible axes.
- ▶ All the variation is along the axes of the coordinate set



Activate Windows

---

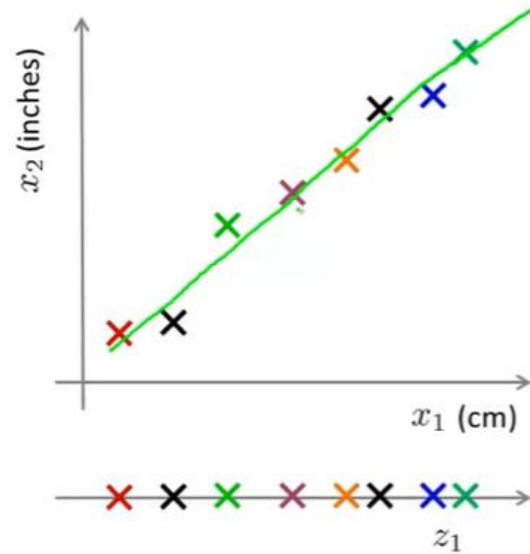
### The Principal Components Analysis Algorithm

---

- Write  $N$  datapoints  $\mathbf{x}_i = (x_{1i}, x_{2i}, \dots, x_{Mi})$  as row vectors
  - Put these vectors into a matrix  $\mathbf{X}$  (which will have size  $N \times M$ )
  - Centre the data by subtracting off the mean of each column, putting it into matrix  $\mathbf{B}$
  - Compute the covariance matrix  $\mathbf{C} = \frac{1}{N}\mathbf{B}^T\mathbf{B}$
  - Compute the eigenvalues and eigenvectors of  $\mathbf{C}$ , so  $\mathbf{V}^{-1}\mathbf{C}\mathbf{V} = \mathbf{D}$ , where  $\mathbf{V}$  holds the eigenvectors of  $\mathbf{C}$  and  $\mathbf{D}$  is the  $M \times M$  diagonal eigenvalue matrix
  - Sort the columns of  $\mathbf{D}$  into order of decreasing eigenvalues, and apply the same order to the columns of  $\mathbf{V}$
  - Reject those with eigenvalue less than some  $\eta$ , leaving  $L$  dimensions in the data
-

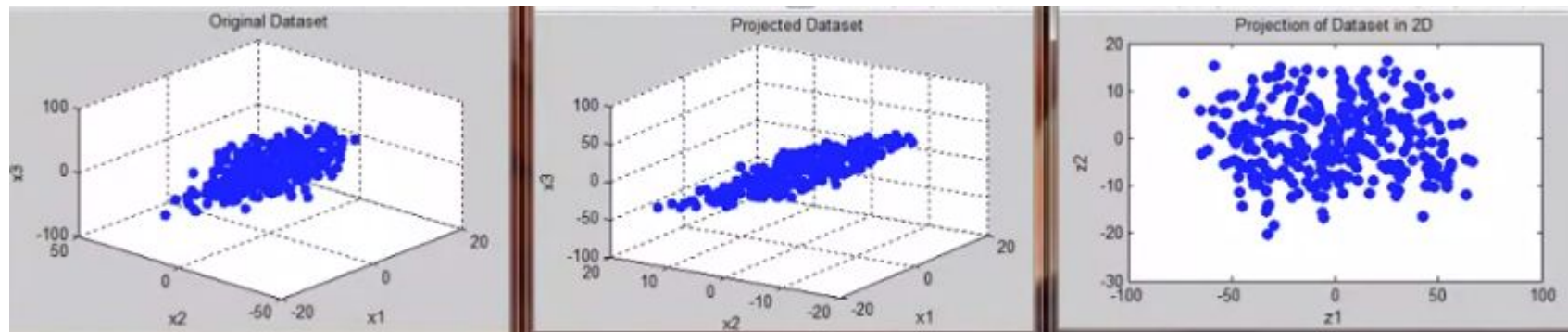
# 2D to 1D

## Data Compression



Reduce data from  
2D to 1D

# 3D to 2D





### Data Visualization

Country	GDP (trillions of US\$)	Per capita GDP (thousands of intl. \$)	Human Develop- ment Index	Life expectancy	Poverty Index (Gini as percentage)	Mean household income (thousands of US\$)	...
Canada	1.577	39.17	0.908	80.7	32.6	67.293	...
China	5.878	7.54	0.687	73	46.9	10.22	...
India	1.632	3.41	0.547	64.7	36.8	0.735	...
Russia	1.48	19.84	0.755	65.5	39.9	0.72	...
Singapore	0.223	56.69	0.866	80	42.5	67.1	...
USA	14.527	46.86	0.91	78.3	40.8	84.3	...

## Data Visualization

Country	$z_1$	$z_2$
Canada	1.6	1.2
China	1.7	0.3
India	1.6	0.2
Russia	1.4	0.5
Singapore	0.5	1.7
USA	2	1.5

# KNN (K Nearest Neighbour)

Defined as the process of finding the closest point to the input point from the given dataset.

Unlike k-means, the unsupervised k-nn does not associate a label to instances

**Does not explicitly learn the model**, but it **saves all the training data**

Requires that the **data is in metric space**.

This means that we **can define metrics for calculation distance** between data points

**Euclidean Distance:** Geometrical distance calculated as the square root of the sum of the squared differences between the two points of interest.

The kNN algorithm consists of two steps:

1. Compute and store the k nearest neighbors for each sample in the training set ("training")
2. For an unlabeled sample, retrieve the k nearest neighbors from dataset and predict label through majority vote / interpolation (or similar) among k nearest neighbors ("prediction/querying")
3. **Split our data in two different chunks**, one will be used to **train** our data and one will be use to **test** the results of our model
4. Scale the attributes
5. train the model

# Which among these are unsupervised?

Given email labeled as spam/not spam, learn a spam filter.

Given a set of news articles found on the web, group them into set of articles about the same story.

Given a database of customer data, automatically discover market segments and group customers into different market segments.

Given a dataset of patients diagnosed as either having diabetes or not, learn to classify new patients as having diabetes or not.

# Answers.....

- ▶ 1. Labelled : Supervised
- ▶ 2. Clustering : Unsupervised
- ▶ 3. Segmentation to be done automatically : Unsupervised
- ▶ 4. Classification : Supervised





Thank You!



# References

1. [https://www.tutorialspoint.com/artificial intelligence with python/artificial intelligence with python unsupervised learning clustering.htm](https://www.tutorialspoint.com/artificial_intelligence_with_python/artificial_intelligence_with_python_unsupervised_learning_clustering.htm)
2. <https://towardsdatascience.com/knn-in-python-835643e2fb53>
3. <https://datascience.stackexchange.com/questions/34073/sklearn-unsupervised-knn-vs-k-means>
4. Stephen Marsland, "MACHINE LEARNING An Algorithmic Perspective Second Edition", CRC Press