Unsupervised Learning

Why Unsupervised Learning?

- Labeled data expensive and difficult to collect; unlabeled data cheap and abundant
- Compressed representation saves on storage and computation
- Reduce noise, irrelevant attributes in high dimensional data
- Pre-processing step for supervised learning
- Often used more in exploratory data analysis

Challenges:

- No simple goal as in supervised learning
- Validation of results is subjective

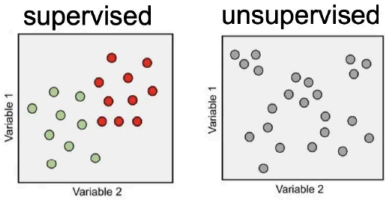
▼ ***Clustering intro and dimensional reduction***

From Supervised Learning . . .

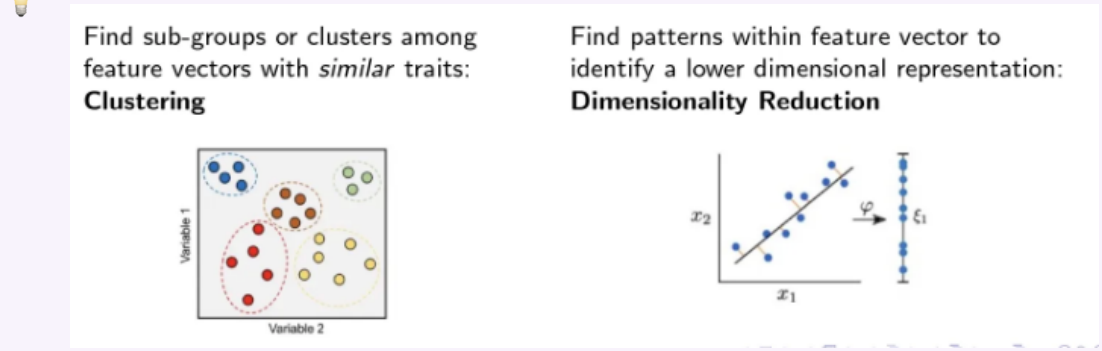
- **Labeled observations**: Each observation is a tuple $(\mathbf{x}, y)$ of feature vector $\mathbf{x}$ and output label y which are related according to an unknown function $f(\mathbf{x}) = y$.
- During training: Use the labeled observations to learn the relationship between $\mathbf{x}$ and y, i.e., find a function (or model) $h(\mathbf{x})$ that best fits the observations
- Goal: Ensure that the learned model $h(\mathbf{x})$ accurately predicts the output label of a previously unseen, test feature input (generalization)
- Labels : 'Teachers' during training, and 'validator' of results during testing

. . . to Unsupervised Learning

- **Unlabeled** data set of feature vectors

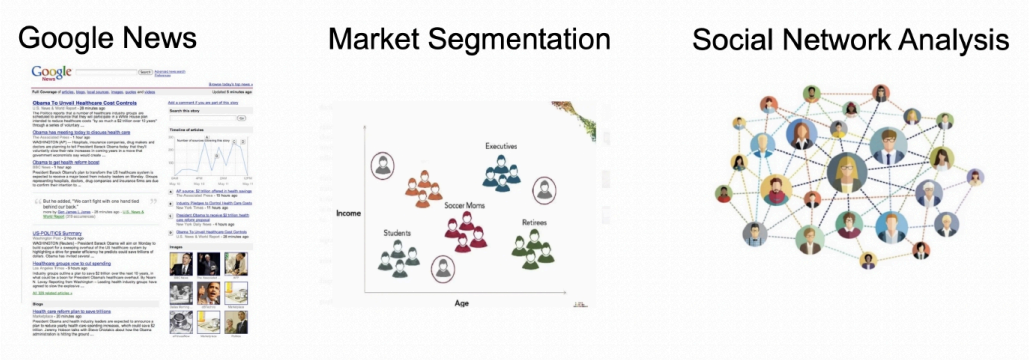


- What can we deduce?



In the dimensionality reduction picture it's showing a linear directly proportional relationship between x_1 and x_2 originally represented by a 2D graph being reduced to a 1D graph with only one label: ξ_1 with the mapping function being shown as ϕ {stylised slightly} Basically since there is a constant relationship between the two variables it can be represented as a straight increasing vertical line if the same reduction formula is applied to each point. Could be something as simple as $(5x - 1) + (5y + 3)$

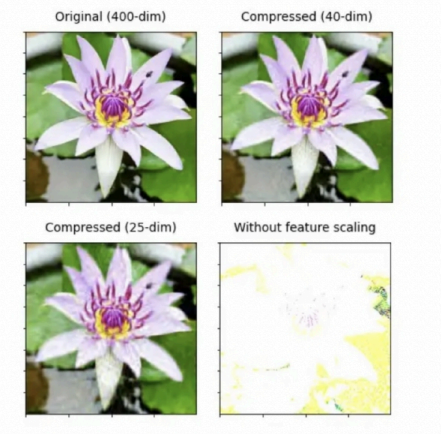
Clustering: Real World Applications



Clusters are potential 'classes'; clustering algorithms automatically find 'classes'

Dimensionality Reduction: Application

Image Compression



Techniques:

- Principal Component Analysis
- Non-negative Matrix Factorization
- Linear Discriminant Analysis

▼ ***Dimensionality Reduction Application Techniques Expanded A Little***

💡

> 💡 Dimensionality Reduction: PCA, NMF, & LDA
>
> **Why Dimensionality Reduction?**
>
> - High-dimensional data (e.g., large images) can be computationally expensive.
> - Reducing dimensions helps remove noise, reveal important patterns, and speed up learning.
> - Often used in image compression, data visualization (e.g., t-SNE, PCA plots), and preprocessing for machine learning. *(Color: Yellow)*

> 🔵 Principal Component Analysis (PCA)
>
> **Key Idea:**
>
> - Finds new axes (principal components) that capture the greatest variance in the data.
> - First principal component explains the largest amount of variance, second explains the next largest, etc., all orthogonal to each other.
>   - All at right angles to each other
>
> **Steps:**
>
> 1. Center the data S
>   a. Subtract the mean
> 2. Compute the covariance matrix
>   a. $\Sigma \frac{1}{m-1} X_{centered}^T X_{centered}$
> 3. Find eigenvalues and eigenvectors of the covariance matrix.
> 4. Choose the top *k* eigenvectors (corresponding to the largest eigenvalues).
> 5. Project the data onto these *k* components.
>
> **Applications:**
>
> - Image compression

        - Represent images with fewer principal components.
    - Noise reduction in signals.
    - Visualizing high-dimensional data in 2D or 3D
        - *Color*
            - Blues

🟢 Non-negative Matrix Factorization (NMF)

Key Idea:

- Factorizes a data matrix **X** into two matrices **W** and **H**, with all non-negative elements.
- **X ≈ W × H**, where **W** can be seen as “basis components” and **H** as “coefficients.”

Why Non-negativity?

- Makes interpretation easier
- Useful in tasks
    - Topic modeling
        - Words counts are non-negative)
        - Image decomposition
            - Pixel intensities
                - Pixel intensities can’t be negative

Applications:

- Face recognition
    - learns parts-based representations
        - Like
            - eyes
            - nose
            - mouth
- Image compression and reconstruction.
- Topic extraction from text documents.*(Color: Green)*

> 🔴 Linear Discriminant Analysis (LDA)
>
> **Key Idea:**
>
> - Supervised method that finds a linear combination of features to separate classes.
> - Maximizes the between-class variance while minimizing the within-class variance.
> - Results in lower-dimensional space that best separates different labeled classes.
>
> **How it Differs from PCA:**
>
> - PCA focuses on capturing the maximum variance in data overall (unsupervised).
> - LDA focuses on class separation (supervised).
>
> **Applications:**
>
> - Pattern recognition and classification tasks (e.g., face recognition).
> - Dimensionality reduction for supervised learning (e.g., prior to using a classifier).*(Color: Red)*

> 🟣 Key Takeaways:
>
> - **Principal Component Analysis** : Unsupervised, focuses on variance. Good for data compression and noise reduction.
> - **Non-negative Matrix Factorization** : Non-negative constraints, interpretable parts-based representation.
> - **Linear Discriminant Analysis** : Supervised, focuses on class separation.

▼ ***Fundamentals of clustering***

1. *What is clustering?*
    a. *Goal*
        i. *Find natural groupings among observations/objects/feature vectors*
    b. *Segment observations into clusters/groups such that*

a. *Objects within a cluster high similarity (high intra-cluster similarity)*

b. *Objects across clusters have low similarity (low inter-cluster similarity)*

Example 1: Clustering of Mammals

- Problem: Group mammals into three clusters (herbivores, carnivores, omnivores) based on the dental information and the weight of mammal
- The data matrix is given as

	Incisor (top)	Canine (top)	Molar (top)	Pre-molar (top)	Weight (pounds)
Badger	3	1	3	1	10
Bear	3	1	4	2	278
Cow	0	0	3	3	400
Dog	3	1	4	2	20
Fox	3	1	4	2	5

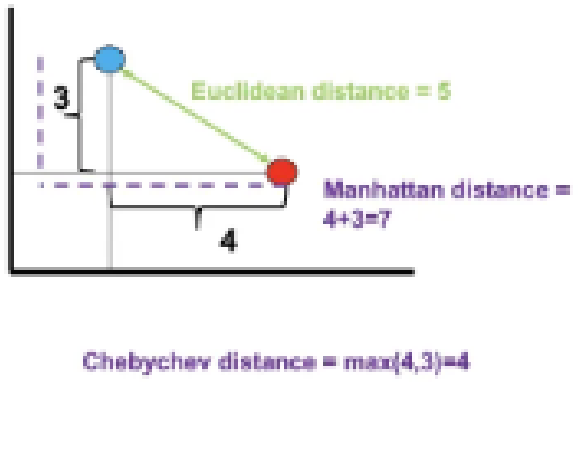
▼ How do you identify 'similar' feature vectors?

- Solution:
  - Depends on data type:
    - Continuous-valued feature attributes
      - Example
        - $x = (0.1, 11.65, 15, 1.4)$

Continuous-valued features:

- For two continuous-valued feature vectors
  - $x^{(1)} = (x_1^{(1)}, \ldots, x_m^{(1)})$ and
  - $x^{(2)} = (x_1^{(2)}, \ldots, x_m^{(2)})$,
  - the proximity index can be any of the following distances

- ***Euclidean distance***
  - $d_{Euc}(x^{(1)}, x^{(2)}) = \sqrt{(x_1^{(1)} - x_1^{(2)})^2 + \cdots + (x_m^{(1)} - x_m^{(2)})^2}$
  - Measures the straight-line distance between two points in a multi-dimensional space.
- ***Manhattan distance***
  - $d_{Man}(x^{(1)}, x^{(2)}) = |x_1^{(1)} - x_1^{(2)}| + \cdots + |x_m^{(1)} - x_m^{(2)}|$



- ***Chebyshev distance***
  - $d_{Cheb}(x^{(1)}, x^{(2)}) = \max_{j=1,\ldots,m} |x_j^{(1)} - x_j^{(2)}|$

- Nominal feature /Discrete attributes:
  - Example
    - $x = (Large, Medium, Small)$
    - Must be mapped to discrete values consistently :
      - $Large \Rightarrow 3$
      - $Medium \Rightarrow 2$
      - $Small \Rightarrow 1$

Use proximity indices to quantify the strength of relationship between any two feature vectors:

🔥 **Proximity Indices:**

Proximity indices are metrics used to quantify the similarity or dissimilarity between data points. In clustering, these indices help in grouping similar items together by measuring how "$close$" or "far apart" they are in terms of their features.

Types of Proximity Indices:

1. **Correlation Coefficient:**
   - Measures the degree to which two variables are linearly related. Example:
     - For two feature vectors X and Y, correlation coefficient r is given by:

       $r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \sum(Y_i - \bar{Y})^2}}$, where $\bar{X}$ is mean of X
2. **Cosine Similarity:**
   - Measures the cosine of the angle between two non-zero vectors, indicating the orientation rather than the magnitude.
     Example:
     - For two vectors A and B, cosine similarity is:

       $\frac{A \cdot B}{\|A\| \|B\|}$
   - Where $A \cdot B$ is the dot product and $\|A\|$ and $\|B\|$ are magnitudes of vectors A and B

▼ ***Distance Functions***

Properties of Distance Functions

All of the previous distance functions satisfy the following properties:

- Distance between two points is always non-negative, i.e.,

$$d(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) \geq 0$$

- Distance between a point to itself is zero, i.e.,

$$d(\mathbf{x}^{(1)}, \mathbf{x}^{(1)}) = 0$$

- Distance measure is symmetric i.e.,

$$d(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = d(\mathbf{x}^{(2)}, \mathbf{x}^{(1)})$$

- Distance measure satisfies a triangle inequality, i.e.,

$$d(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) \leq d(\mathbf{x}^{(1)}, \mathbf{x}^{(3)}) + d(\mathbf{x}^{(3)}, \mathbf{x}^{(2)})$$

Example 1: Revisited

- Compute the Euclidean distance between Badger and Cow.
  - Solution:

$$\begin{aligned} d_{\text{Euc}}(\text{Badger}, \text{Cow}) &= \sqrt{(3-0)^2 + (1-0)^2 + (3-3)^2 + (1-3)^2 + (10-400)^2} \\ &= \sqrt{9+1+0+4+390} = 390.017 \end{aligned}$$

- Compute the Manhattan distance between Badger and Cow.
  - Solution:

$$\begin{aligned} d_{\text{Man}}(\text{Badger}, \text{Cow}) &= |3-0| + |1-0| + |3-3| + |1-3| + |10-400| \\ &= 3+1+0+2+390 = 396 \end{aligned}$$

▼ ***Feature Scaling and Normalization***

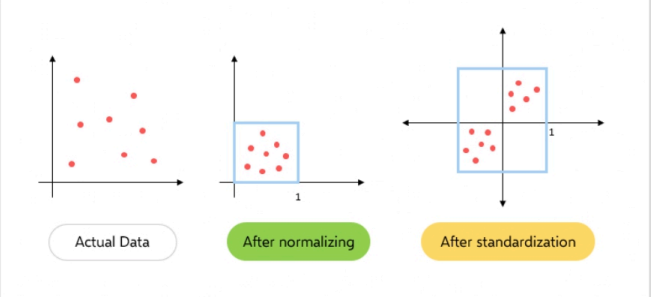
- Different choice of distance functions yields different measures of similarity.

- Distance functions implicitly assign more weighting to features with large ranges than to those with small ranges.
- Rule of thumb:
  - When no a prior domain knowledge is available
    - Clustering should follow the principle of equal weightings to each attribute
    - This necessitates feature scaling of vectors.

Challenge 2: Feature Scaling

When feature attributes have large differences in the range of values, scaling of features is essential so that each attribute contributes equally to the distance measure.

- Two well studied approaches: **min-max normalization** and **z-score standardization**



Min-Max Normalization:

In min-max normalization, all feature attributes are rescaled to lie in the $[0, 1]$ range.

Steps:

- Consider N feature vectors $\mathbf{x}^{(1)} = (x_1^{(1)}, \ldots, x_m^{(1)})$, $\mathbf{x}^{(2)} = (x_1^{(2)}, \ldots, x_m^{(2)})$, $\ldots$, $\mathbf{x}^{(N)} = (x_1^{(N)}, \ldots, x_m^{(N)})$.
- For each jth feature attribute, for $j = 1, \ldots, m$, compute
  - Maximum of jth feature over all N vectors:

$$x_{j,\max} = \max_{i=1,\ldots,N} x_j^{(i)}$$

  - Minimum of jth feature over all N vectors:

$$x_{j,\min} = \min_{i=1,\ldots,N} x_j^{(i)}$$

  - Min-max rescaling of jth attribute in each of the $i = 1, \ldots, N$ vectors:

$$x_{j,\text{new}}^{(i)} \leftarrow \frac{x_j^{(i)} - x_{j,\min}}{x_{j,\max} - x_{j,\min}}$$

Formula is:

$$\text{New value} = \frac{\textit{Current value} - \textit{Minimum value across all attributes of that order } \{\textit{Like the minimum 1st value across all the N attributes}\}}{\textit{Maximum value across all} - \textit{Minimum value across all}}$$

Example 2 Revisited: Normalization

Problem: Normalize the data matrix of Example 2.

Badger	3	1	3	1	10
Bear	3	1	4	2	278
Cow	0	0	3	3	400
Dog	3	1	4	2	20
Fox	3	1	4	2	5

- For first feature, $x_{1,\max} = \max\{3,3,0,3,3\} = 3$ and $x_{1,\min} = \min\{3,3,0,3,3\} = 0$. Using this, transform the first column of the data matrix as

$$\begin{bmatrix} 3 \\ 3 \\ 0 \\ 3 \\ 3 \end{bmatrix} \rightarrow \begin{bmatrix} (3-0)/3 \\ (3-0)/3 \\ (0-0)/3 \\ (3-0)/3 \\ (3-0)/3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \quad (1)$$

- Continue this for other feature columns.

- For the final feature, we have $x_{5,\max} = \max\{10, 278, 400, 20, 5\} = 400$ and $x_{1,\min} = \min\{10, 278, 400, 20, 5\} = 5$ with $x_{5,\max} - x_{5,\min} = 395$. Using this, transform the last column of the data matrix as

$$\begin{bmatrix} 10 \\ 278 \\ 400 \\ 20 \\ 5 \end{bmatrix} \rightarrow \begin{bmatrix} (10-5)/395 \\ (278-5)/395 \\ 395/395 \\ (20-5)/395 \\ (5-5)/395 \end{bmatrix} = \begin{bmatrix} 5/395 \\ 273/395 \\ 1 \\ 5/395 \\ 0 \end{bmatrix} \tag{2}$$

- The min-max normalized data matrix is then obtained as:

Badger	1	1	0	0	5/395
Bear	1	1	1	0.5	273/395
Cow	0	0	0	1	1
Dog	1	1	1	0.5	5/395
Fox	1	1	1	0.5	0