AI Course

Dr. Mürsel Taşgın

Machine Learning, Classification, Clustering

Learning

- *"Learning is any process by which a system improves performance from experience"*

  Herbert Simon

- *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."*

  Tom Mitchell

Experience

Learning

PERFORMANCE

Learning

- Learning is essential for unknown environments

  i.e., when designer lacks information

- Learning is useful as a system construction method

  i.e., agent will learn from real environment rather than writing/defining it explicitly

- Learning modifies the agent's decision mechanism to improve performance

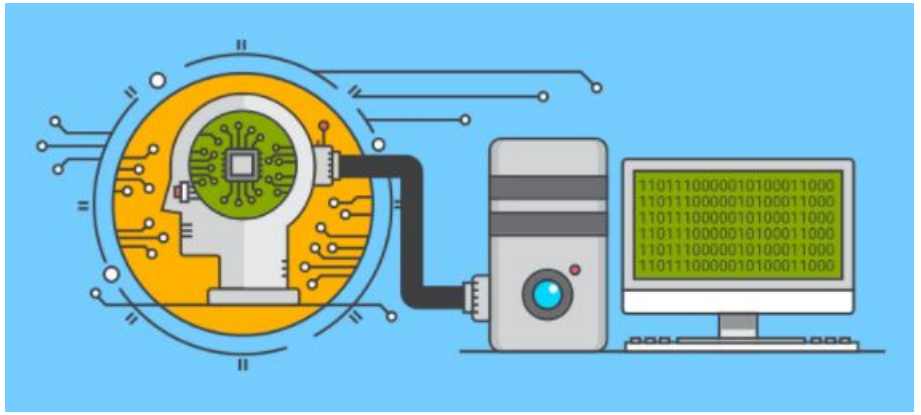
- Learning is "a *process* that leads to *change*, which occurs as a result of *experience* and increases the potential for improved performance and future learning" – *Ambrose et al.*

Learning

Machine Learning

- *Machine learning* is the study of computer algorithms that can improve automatically through experience and by the use of data.

- Machine learning algorithms build a *model* based on sample data, known as training data, in order to make predictions or decisions *without being explicitly programmed to do so*



Machine Learning

- Machine learning: how to acquire a model on the basis of data / experience
  - Learning parameters *(i.e., probabilities)*
  - Learning structure *(i.e., BN graphs, physical systems)*
  - Learning hidden concepts *(i.e., clustering, complex rules)*

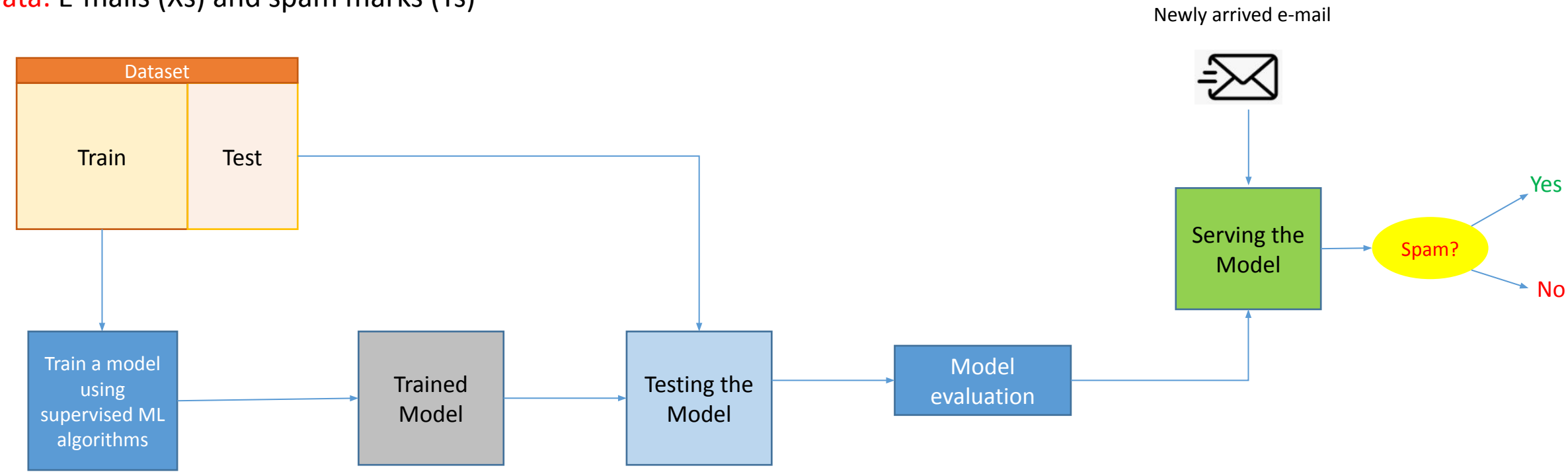
Machine Learning – *Types of learning*

- **Supervised Learning:** Data and corresponding labels are given *(Xs & Ys)*
- **Unsupervised Learning:** Only data is given, no labels provided *(Xs but no Ys)*
- **Semi-supervised Learning:** Some (if not all) labels are present *(Xs and some of Ys)*
- **Reinforcement Learning:** An agent interacting with the world makes observations, takes actions, and is rewarded or punished; it should learn to choose actions in such a way as to obtain a lot of reward (Xs and Reward(Xs))

Machine Learning – *A Supervised Learning example*

- Spam e-mail filtering

- Data: E-mails (Xs) and spam marks (Ys)



Machine Learning

The resulting *model* is also called the *hypothesis*.

Given a model space and an optimality criterion, a *model* satisfying this criterion is sought.

Some criteria:

- Maximizing the prediction accuracy
- Minimizing the hypothesis' size
- Maximizing the hypothesis fitness to the input data
- Maximizing the hypothesis interpretability
- Minimizing the time complexity of prediction

Classification

Machine Learning – *Classification*

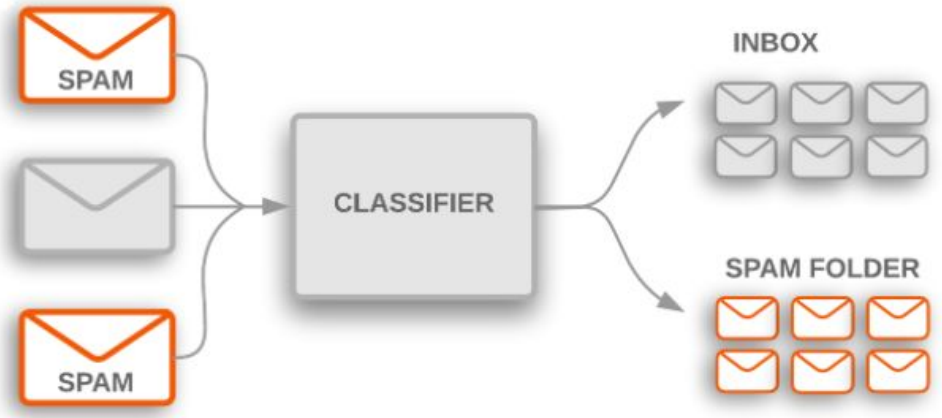
Classification: Systematic arrangement in groups or categories according to established criteria

Classification in machine learning

- Having training data and its labels (target), trying to identify label's of new data according the training and learned model

Example classification problems:

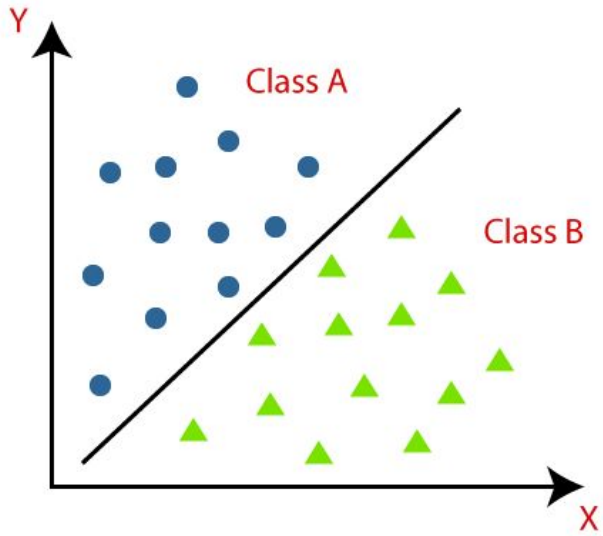
- Hand-written digit recognition *(MNIST)*
- Fraud detection
- Identifying flowers *(IRIS dataset)*
- Spam filtering
- Cancer prediction (Health data)



Machine Learning – *Classification*

Classification: Systematic arrangement in groups or categories according to established criteria


MNIST



Machine Learning – *Classification*

Some terminology

- **Classifier** – It is an algorithm that is used to map the input data to a specific category.
- **Classification Model** – The model predicts or draws a conclusion to the input data given for training, it will predict the class or category for the data.
- **Feature** – A feature is an individual measurable property of the phenomenon being observed.
- **Binary Classification** – It is a type of classification with two outcomes, i.e., either true or false.
- **Multi-Class Classification** – The classification with more than two classes, in multi-class classification each sample is assigned to one and only one label or target.
- **Multi-label Classification** – This is a type of classification where each sample is assigned to a set of labels or targets.
- **Initialize** – It is to assign the classifier to be used for the
- **Train the Classifier** – Each classifier in sci-kit learn uses the fit(X, y) method to fit the model for training the train X and train label y.
- **Predict the Target** – For an unlabeled observation X, the predict(X) method returns predicted label y.
- **Evaluate** – This basically means the evaluation of the model i.e classification report, accuracy score, etc.

Machine Learning – *Classification*

Some methods for classification

- **Linear Models**
  - Linear Regression
  - Support Vector Machines

- **Non-linear Models**
  - K-Nearest Neighbours
  - Kernel SVM
  - Naïve Bayes
  - Decision Tree Classification
  - Random Forest Classification

Machine Learning – *Classification*

- You trained a model to predict cancer from image data using a state of the art machine learning methods

Your model has an accuracy of 99.9%

Machine Learning – *Classification*

By looking at the confusion matrix you realize that the model does not detect any of the positive examples.

Machine Learning – *Classification Errors*

Classifiers try to reduce the overall error so they can be biased towards the majority class.

```
# Negatives = 998
# Positives = 2
```

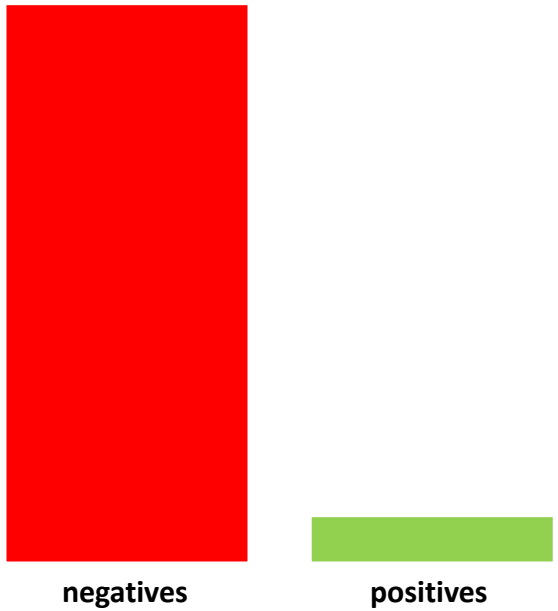
If our algorithm predicts every case as negative class, then the accuracy of the algorithm will be 99.8% !! □ 998/1000 *(Not what we want!)*

This is due to dataset imbalance! (class imbalance within the dataset).

- Different classes within training data is imbalanced (i.e., number of positives, negatives)

Machine Learning – *Classification Errors*

After plotting your class distribution you see that you have thousands of negative examples but just a couple of positives.



For cancer prediction *(and similar critical cases)*, even a single case is critical!

Machine Learning – *Classification*

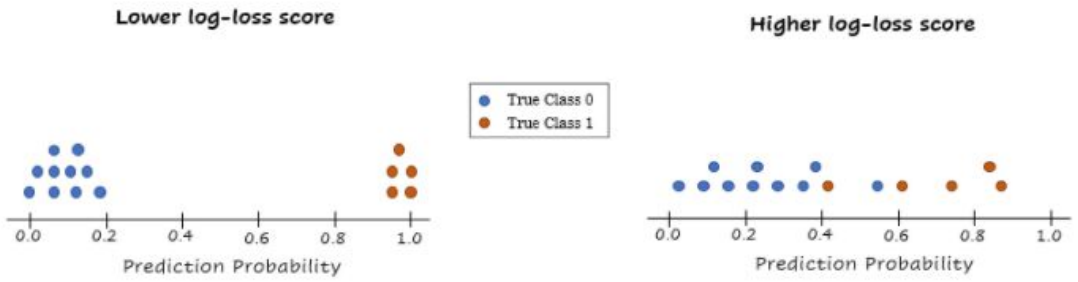
Evaluating the performance

- **Log loss function**

  Log-loss is indicative of how close the prediction probability is to the corresponding actual/true value (0 or 1 in case of binary classification). The more the predicted probability diverges from the actual value, the higher is the log-loss value.

$$Logloss_i = -[y_i \, ln \, p_i + (1 - y_i) \, ln(1 - p_i)]$$

$$Logloss = -\frac{1}{N}\sum_{i=1}^{N}[y_i \, ln \, p_i + (1 - y_i) \, ln(1 - p_i)]$$



Email Seq	Actual Class (y)	Prediction Probability (p)	y * ln(p) (1)	(1-y) * ln(1-p) (2)	Logloss -(1+2)
876	1	0.90	-0.105	0.000	0.105
343	0	0.20	0.000	-0.223	0.223

Machine Learning – *Classification*

Evaluating the performance

Cross-entropy

- Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1.
- Cross-entropy loss increases as the predicted probability diverges from the actual label.
- So predicting a probability of .012 when the actual observation label is 1 would be bad and result in a high loss value.
- *A perfect model would have a log loss of 0.*

$$-(y\log(p)+(1-y)\log(1-p))$$

Binary cross-entropy loss

$$-\sum_{c=1}^{M} y_{o,c}\log(p_{o,c})$$

Multi-class cross-entropy loss

Machine Learning – *Classification*

Cross-entropy

In information theory, we like to describe the "*surprise*" of an event. An event is more surprising the less likely it is, meaning it contains more information.

- **Low Probability Event** (*surprising*): More information.
- **Higher Probability Event** (*unsurprising*): Less information.

Information $h(x) = -\log(P(x))$

Entropy is the number of bits required to transmit a randomly selected event from a probability distribution. A skewed distribution has a low entropy, whereas a distribution where events have equal probability has a larger entropy.

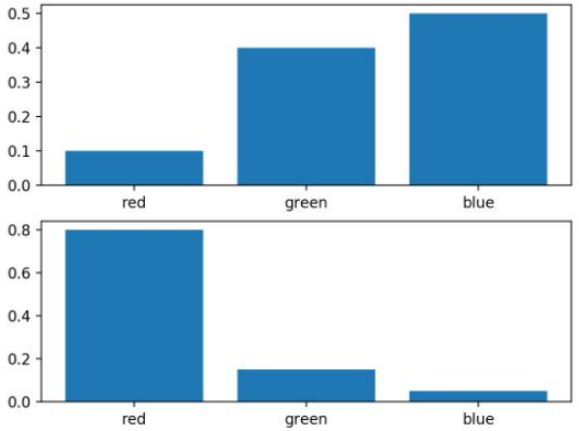
A skewed probability distribution has less "surprise" and in turn a low entropy because likely events dominate. Balanced distribution are more surprising and turn have higher entropy because events are equally likely.

- **Skewed Probability Distribution** (*unsurprising*): Low entropy.
- **Balanced Probability Distribution** (*surprising*): High entropy.

Entropy $H(X) = -\sum_{x \in X} P(x) * \log(P(x))$

Cross-entropy can be calculated using the probabilities of the events from P and Q, as follows:

Cross-entropy $H(P, Q) = -\sum_{x \in X} P(x) * \log(Q(x))$

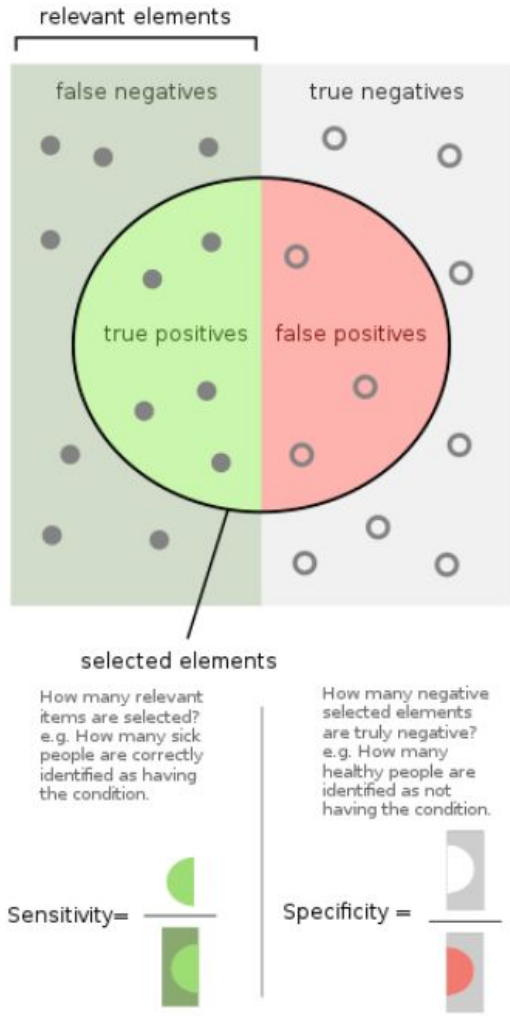


Machine Learning – *Confusion Matrix*

Evaluating the performance

Confusion Matrix

		Predicted condition	
	Total population = P + N	**Positive (PP)**	**Negative (PN)**
Actual condition	**Positive (P)**	**True positive (TP)**, hit	**False negative (FN)**, type II error, miss, underestimation
	Negative (N)	**False positive (FP)**, type I error, false alarm, overestimation	**True negative (TN)**, correct rejection



Machine Learning – *Confusion Matrix*

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP), hit	False negative (FN), type II error, miss, underestimation
	Negative (N)	False positive (FP), type I error, false alarm, overestimation	True negative (TN), correct rejection

Machine Learning – *Confusion Matrix*

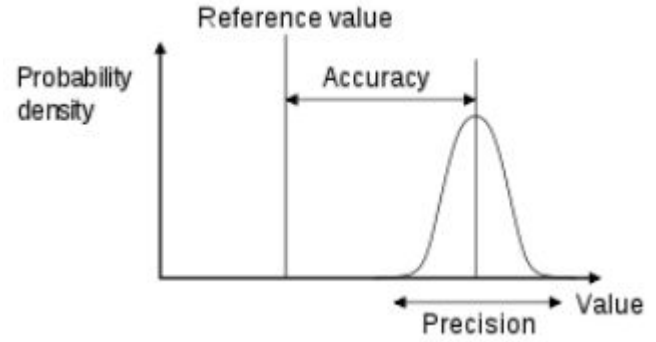
A: accurate and precise

B: precise, but not accurate

C: neither accurate nor precise

D: accurate, but not precise

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP), hit	False negative (FN), type II error, miss, underestimation
	Negative (N)	False positive (FP), type I error, false alarm, overestimation	True negative (TN), correct rejection



Machine Learning – *Classification Example*

We built a model for cancer prediction. Following table shows the classification predictions and true labels

Patient ID	Prediction *(Cancer:1, Not-Cancer:0)*	True Label *(Cancer:1, Not-Cancer:0)*
ID-1	1	1
ID-2	0	0
ID-3	0	1
ID-4	1	0
ID-5	0	0
ID-6	1	1
ID-7	0	0
ID-8	0	0
ID-9	0	1
ID-10	1	0

Accuracy ?
Precision ?
Recall ?
F1-score?
Sensitivity?
Specifity?

Machine Learning – *Classification Example*

We built a model for cancer prediction. Following table shows the classification predictions and true labels

Patient ID	Prediction *(Cancer:1, Not-Cancer:0)*	True Label *(Cancer:1, Not-Cancer:0)*
ID-1	1	1
ID-2	0	0
ID-3	0	1
ID-4	1	0
ID-5	0	0
ID-6	1	1
ID-7	0	0
ID-8	0	0
ID-9	0	1
ID-10	1	0

		Prediction Condition			
		Positive		Negative	
Actual Condition	Positive	TP	2	FN	2
	Negative	FP	2	TN	4

Accuracy	0,60
Precision	0,50
Recall	0,50
F1-score	0,50
Sensitivity	0,5
Specifity	0,67

Machine Learning – *Classification*

Problems in classification

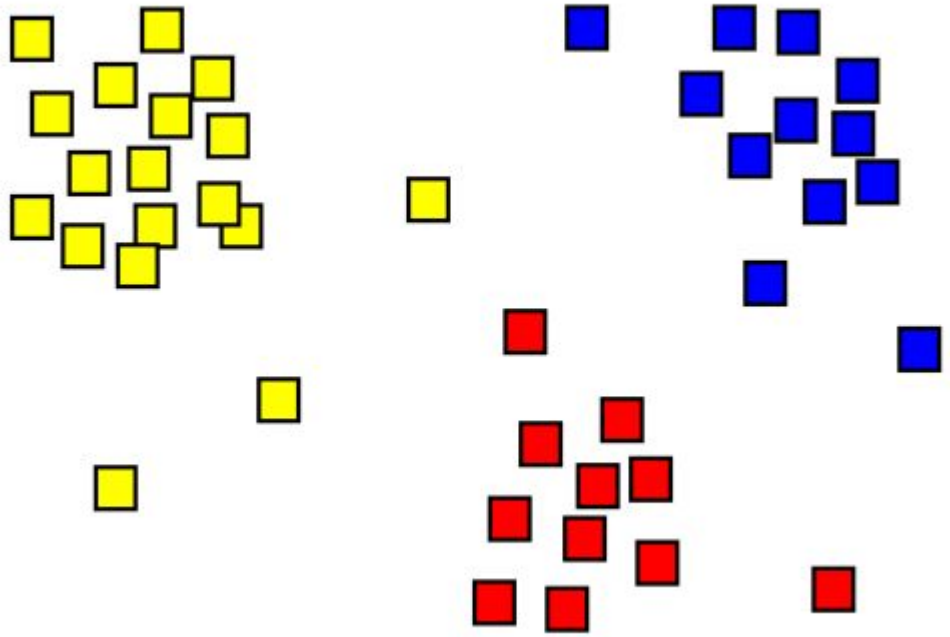
- Dataset imbalance
  - SMOTE oversampling
  - Random undersampling

- Lack of labeled data
  - Data generation
  - Data labeling
  - Self-supervised learning, semi-supervised learning

Clustering

Machine Learning – *Clustering*

Clustering is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense) to each other than to those in other groups (clusters).



Machine Learning – *Clustering*

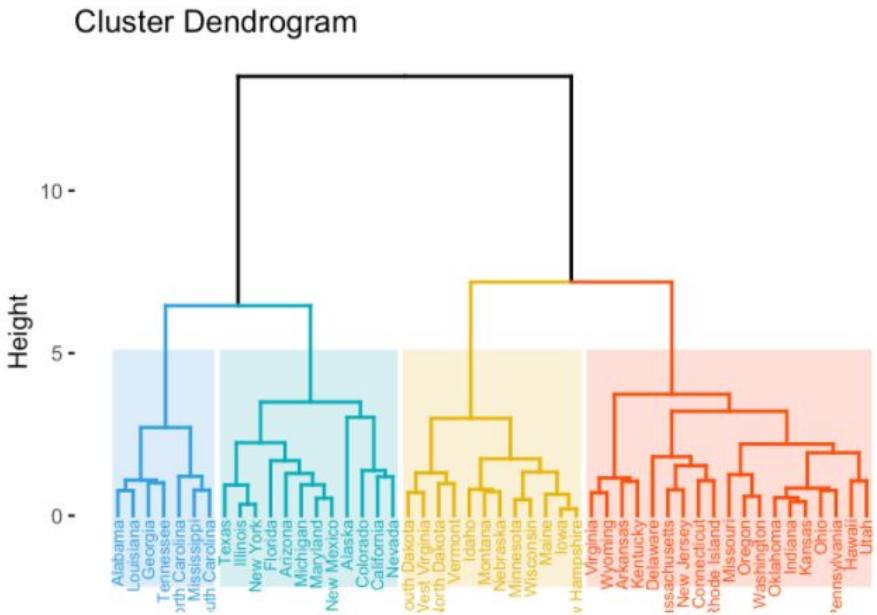
A **cluster** cannot be defined precisely *There are so many clustering algorithms!*

Clustering Algorithm Types:

- *Connectivity models (Hierarchical clustering, distance based)*
- *Centroid models (k-means)*
- *Distribution models (based on statistical distributions)*
- *Density models (DBSCAN)*
- *Group models*
- *Graph-based models (cliques, communities)*
- *Neural models (Self-organizing maps)*

Machine Learning – *Clustering*

Hierarchical Algorithms



Divisive

- *Initially all data points in a single cluster*
- *Based on a heuristic, divide data into clusters iteratively*
- *Stop when further division is not possible or do not make improvement*

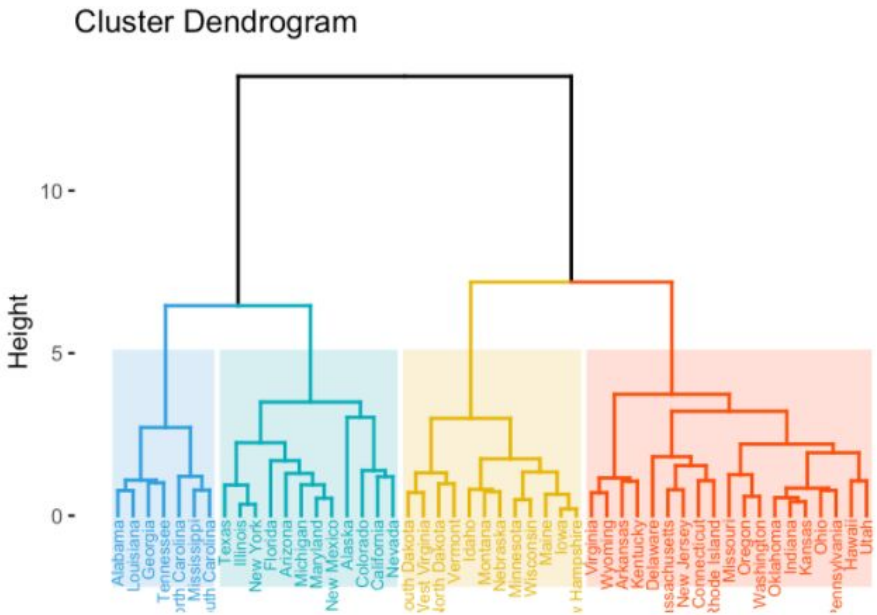
Agglomerative

- *Initially all data points form a separate cluster (of size 1)*
- *Based on a heuristic, merge data points into clusters iteratively*
- *Stop when no more merging is possible or do not make further improvement*

Machine Learning – *Clustering*

Hierarchical Algorithms

- *Using a heuristic either divide or merge operation is used*
- *A fitness function, cost function or any metric should be used to decide on best next move, i.e.,* `merge_next(a,b), divide(c)`
- *Build a dendrogram representing the hierarchical grouping*
- *Cut the dendrogram at some point to identify clusters*

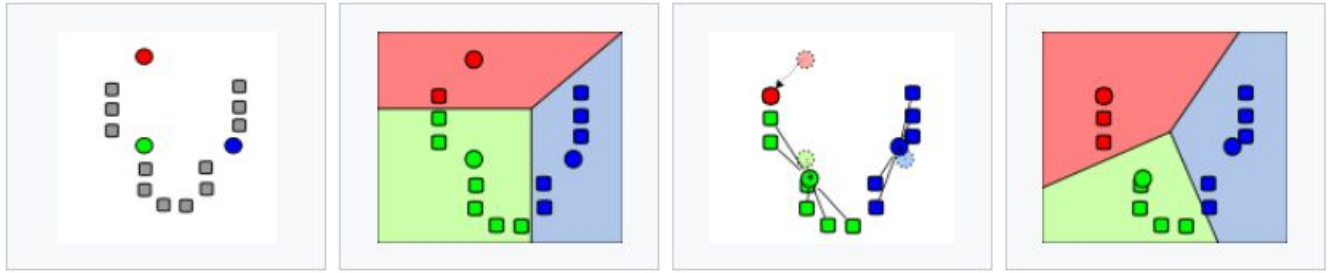


Machine Learning – *Clustering*

Centroid models

k-means algorithm

- *N* observations into *k* clusters
- Randomly choose k observations as *initial means*
- Assign *k* points as means of data
- Each observation belongs to *nearest mean* (cluster center, centroid)
- Calculate *centroids* of each k clusters *new means for next iteration*
- Repeat iteration until convergence (no change occurs)



1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).

2. *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

3. The centroid of each of the *k* clusters becomes the new mean.

4. Steps 2 and 3 are repeated until convergence has been reached.

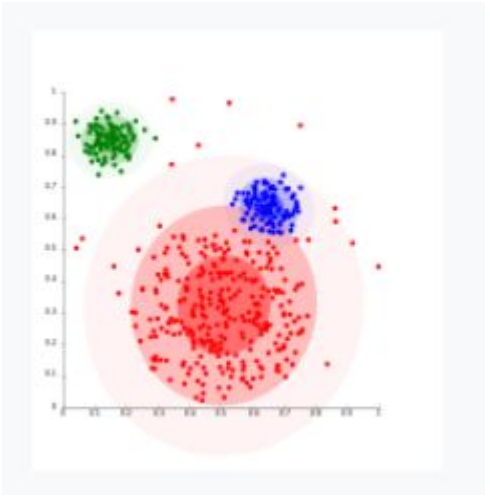
Machine Learning – *Clustering*

Distribution-based clustering

- *Statistics based*
- *Clusters can be defined as objects belonging to same distribution*
- *Sampling objects from a distribution*

Gaussian mixture models (Expectation – maximization)

- Data set is usually modeled with a fixed number of Gaussian distributions that are initialized randomly
- Parameters of distributions are iteratively optimized to better fit the data set
- This will converge to a local optimum, so multiple runs may produce different results.
- Objects are often then assigned to the Gaussian distribution they most likely belong to



Machine Learning – *Clustering*

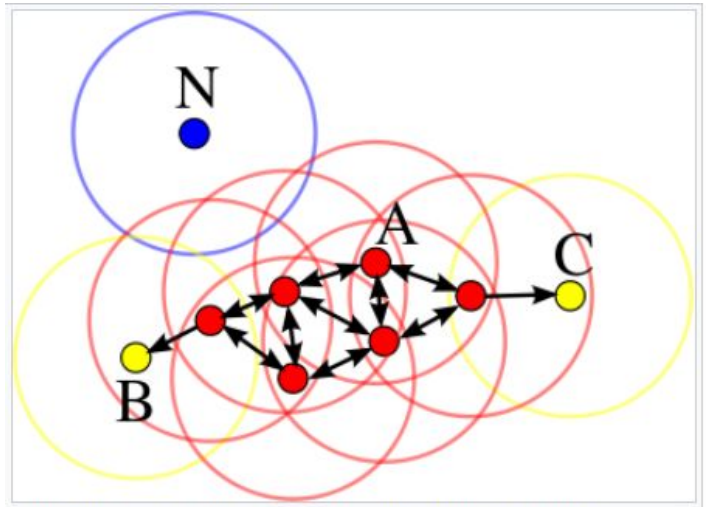
Density-based clustering

In density-based clustering, clusters are defined as areas of higher density than the remainder of the data set. Objects in sparse areas – that are required to separate clusters – are usually considered to be noise and border points.

DBSCAN

- Non-parametric model
- Connects points within certain distance thresholds
- Only the points that satisfy a density criterion are connected

- A point p is a *core point* if at least *minPts* points are within distance ε of it (including p).
- A point q is *directly reachable* from p if point q is within distance ε from core point p. Points are only said to be directly reachable from core points.
- A point q is *reachable* from p if there is a path $p_1, \ldots, p_n$ with $p_1 = p$ and $p_n = q$, where each p_{i+1} is directly reachable from p_i. Note that this implies that the initial point and all points on the path must be core points, with the possible exception of q.
- All points not reachable from any other point are *outliers* or *noise points*.



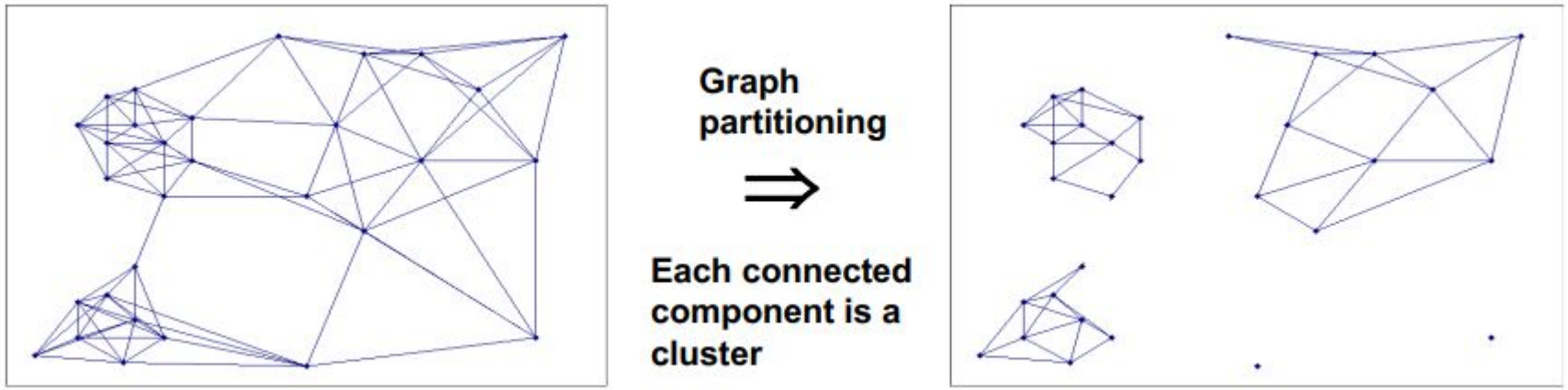
In this diagram, minPts = 4. Point A and the other red points are core points, because the area surrounding these points in an ε radius contain at least 4 points (including the point itself). Because they are all reachable from one another, they form a single cluster. Points B and C are not core points, but are reachable from A (via other core points) and thus belong to the cluster as well. Point N is a noise point that is neither a core point nor directly-reachable.

Machine Learning – *Clustering*

Graph-based clustering

Transform the data into a graph representation

- Vertices are the data points to be clustered
- Edges are weighted based on similarity between data points
- Apply a clustering method on graph
- Find the components, communities or clusters in the graph



Machine Learning – *Clustering*

Graph-based clustering

Objective Function

- An objective function to determine what would be the best «cut» or grouping in the graph

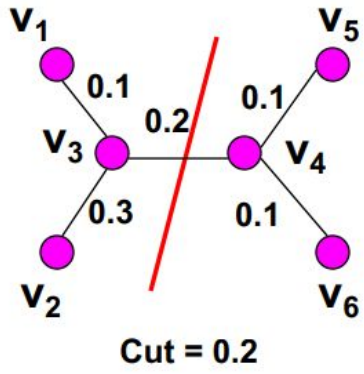
Algorithm

- Find the optimal partition (according to objective function)

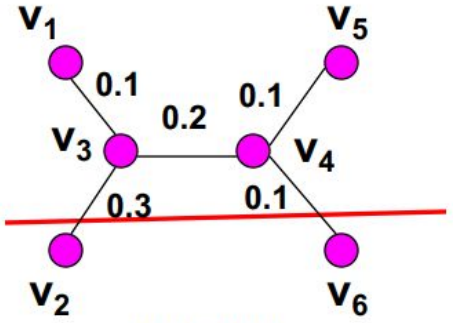
i.e., Minimize cut

$$\mathrm{Cut}(V_1, V_2) = \sum_{\substack{i \in V_1, \\ j \in V_2}} w_{ij}$$

w_{ij} is weight of the edge between nodes i and j



Cut = 0.2



Cut = 0.4

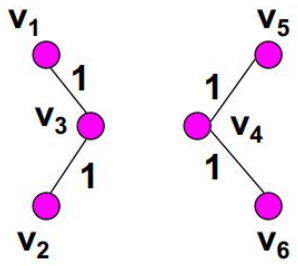
Limitation: The optimal solution may split up a single node from the rest of the graph!

Machine Learning – *Clustering*

Graph-based clustering

Spectral clustering

- Spectral properties of a graph
  - Spectral properties: eigenvalues/eigenvectors of the adjacency matrix can be used to represent a graph
  - Graph Laplacian (using adjacency matrix)
  - There exists a relationship between spectral properties of a graph and the graph partitioning problem



$$W = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Two block matrices

$$\text{Laplacian, } L = D - W$$

$$L = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & -1 & -1 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}$$

Laplacian also has a block structure

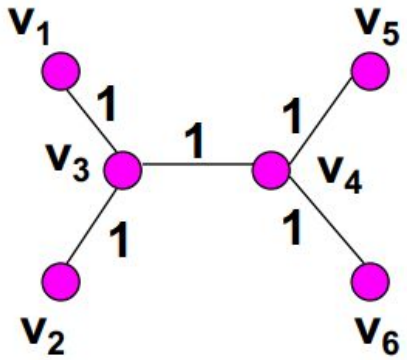
Consider a data set with N data points

1. Construct an N × N similarity matrix, W
2. Compute the N × N Laplacian matrix, L = D – W
3. Compute the k "smallest" eigenvectors of L
   a) Each eigenvector v_i is an N × 1 column vector
   b) Create a matrix V containing eigenvectors v_1, v_2, .., v_k as columns (you may exclude the first eigenvector)
4. Cluster the rows in V using k-means or other clustering algorithms into K clusters

Machine Learning – *Clustering*

Graph-based clustering

Spectral clustering



$$L = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 3 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}$$

Eigenvalues of L:

$$\Lambda = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.44 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4.56 \end{bmatrix}$$

Eigenvectors of L:

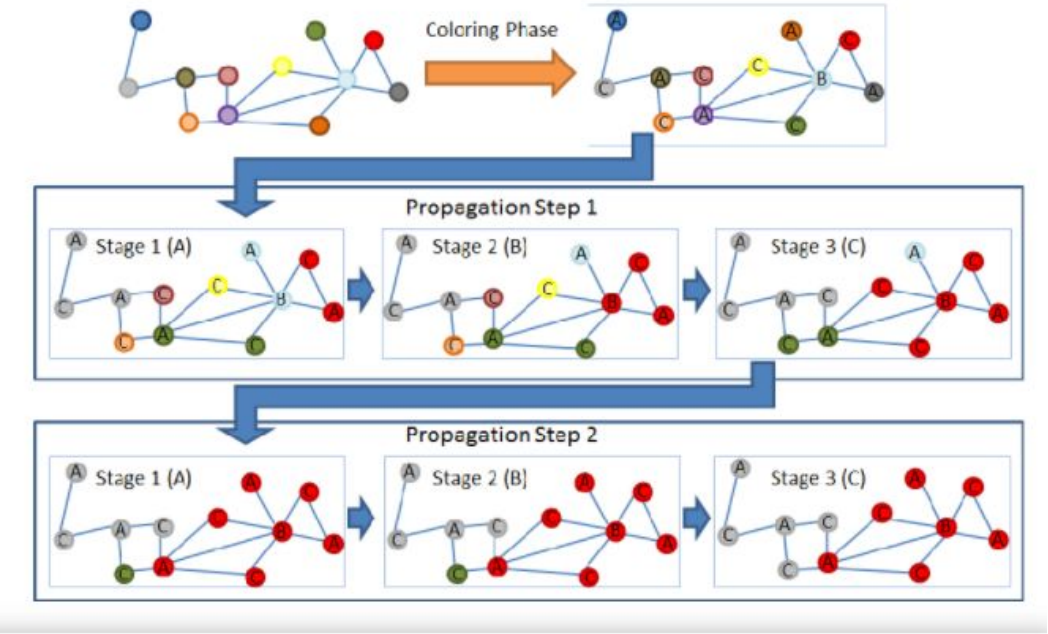
$$V = \begin{bmatrix} 0.41 & 0.46 & 0.65 & -0.28 & 0.29 & -0.18 \\ 0.41 & 0.46 & -0.65 & 0.28 & 0.29 & -0.18 \\ 0.41 & 0.26 & 0 & 0 & -0.58 & 0.66 \\ 0.41 & -0.26 & 0 & 0 & -0.58 & -0.66 \\ 0.41 & -0.46 & 0.28 & 0.65 & 0.29 & 0.18 \\ 0.41 & -0.46 & -0.28 & -0.65 & 0.29 & 0.18 \end{bmatrix}$$

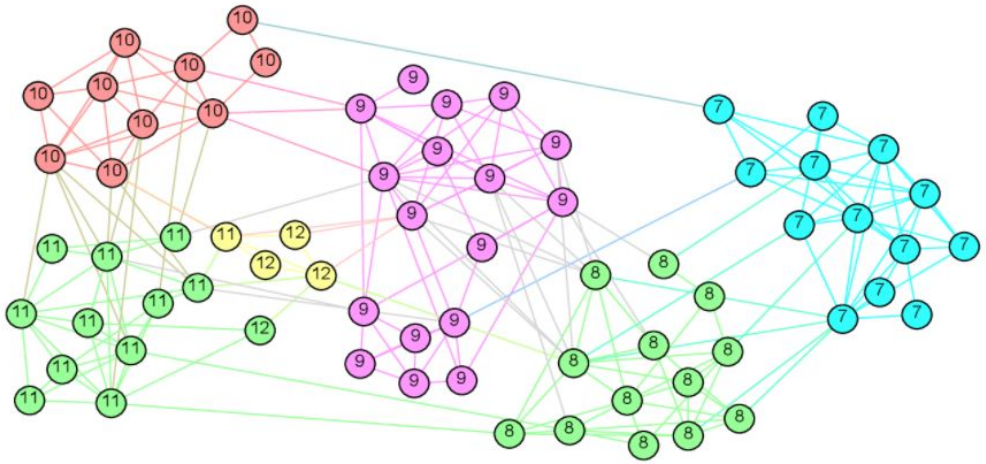
Machine Learning – *Clustering*

Graph-based clustering

Label propagation

- *Labels represent cluster identity for each node*
- *Initially all nodes have their own labels (clusters of size 1)*
- *Iteratively, at each round, nodes get the most popular label among its neighbors*
- *Stop when no label update is possible (no change of labels)*





Machine Learning – *Clustering*

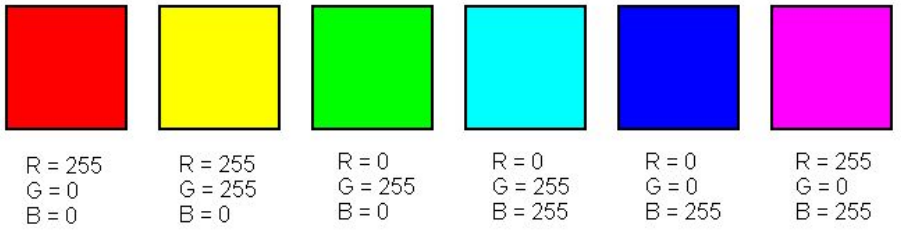
Neural models

Self-organizing maps

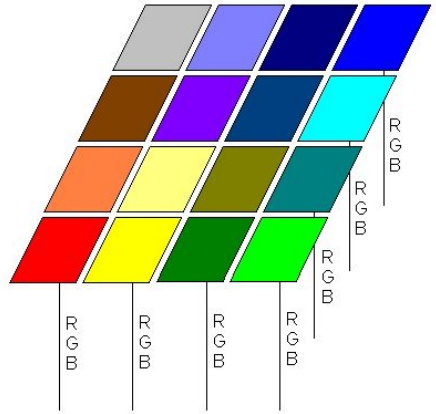
- An unsupervised machine learning technique used to produce a *low-dimensional representation* of a higher dimensional data set while preserving the topological structure of the data.
- Competitive Learning instead of loss minimization

Example: RGB colors

- Input Data: 3 dimensional color data (Red-Green-Blue)



- Neural Network: 2 dimensional layout for neurons. Each neuron contains a weight vector representing its RGB values and a geometric location in the grid.



Machine Learning – *Clustering*

Neural models

Self-organizing maps

The SOM learning algorithm:

- Initialize the weights
- Iterate over the input data, finding the "winning" neuron for each input (best matching unit – BMU)
- Adjust weights based on the location of that "winning" neuron

```
Initialize weights
      For 0 to X number of training epochs
            Select a sample from the input data set
            Find the "winning" neuron for the sample input
            Adjust the weights of nearby neurons

      End for loop
```

Finding winning neuron

$$\sqrt{(n_r - v_r)^2 + (n_g - v_g)^2 + (n_b - v_b)^2}$$

Neighbor Update

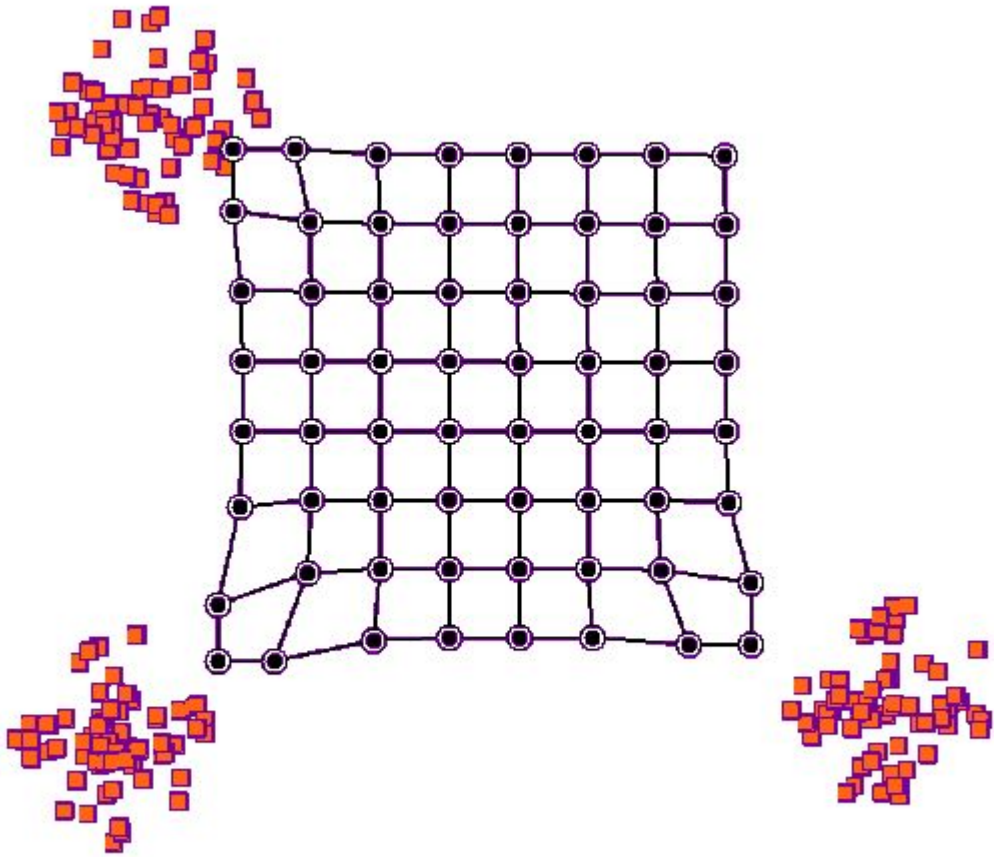
$$n_i(t+1) = n_i(t) + h * [v(t) - n_i(t)]$$

$n_i(t)$ = weight vector of neuron i at regression step t
$v(t)$ = input vector at regression step t
h = neighborhood function

Machine Learning – *Clustering*

Neural models

Self-organizing maps



Machine Learning – *Clustering Use Cases*

Where to use clustering?

- Fraud detection (clusters of fraudsters)
- Identification of customer groups/interest groups
- Anomaly detection
- Drug discovery, cancer research
- Genetics research
- Recommender systems
- Identifying crime locations
- Delivery optimization
- Ride-share data analysis
- Document classification
- Spam filtering