Machine Learning

Unsupervised Learning

Unsupervised Learning

- In supervised learning the training data provides desired target output for learning
- In unsupervised learning the training data does not contain a target value
  - Training data does not specify what to learn
  - Learning algorithm has to specify what to learn
    - Find potential hidden categories
    - Find patterns in the data (data mining)
    - Learn feature representations for the data
  - Performance function has to be defined entirely in terms of the input data

Clustering

- Clustering is the task of dividing the data into a set of groups (clusters)
- Clustering can be thought of as dividing data into hidden/unknown "classes" (clusters)
    - Akin to clustering but without knowledge of any class labels or any definitions of what "classes" are
    - To allow this, the clustering algorithm has to define internally what makes good clusters
        - "Similarity" of instances is the most common criterion

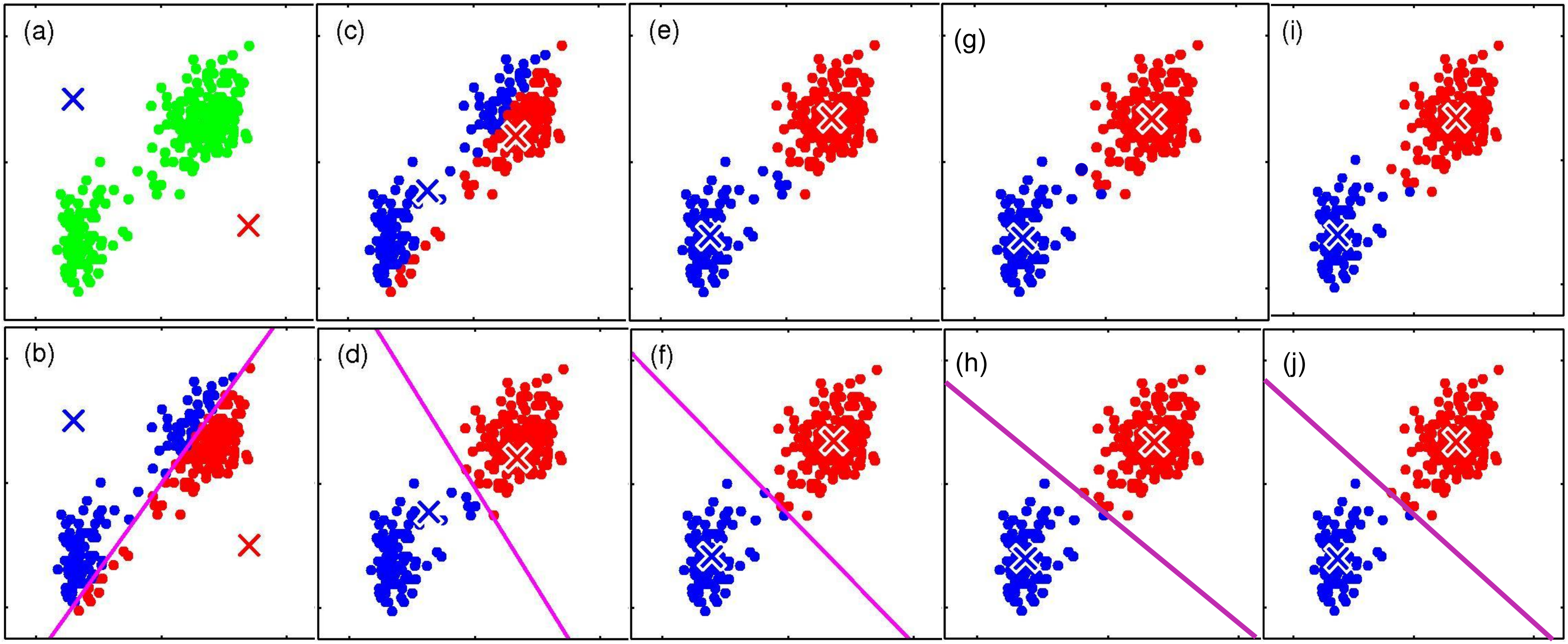
Clustering

- "Classes" are not given so the number of clusters has to be determined differently
  - Fixed number of clusters set beforehand
  - Maximum "dissimilarity" in a class
  - Ratio of intra-cluster and inter-cluster variance
- Clustering is based on a measure of similarity
  - Euclidian distance of feature vector
  - Graph similarity (traversal stats, modification distance)
  - Dynamic Time Warping (DTW) distance
  - Distribution similarity (KL divergence)

K-Means Clustering

- K-Means clustering is one of the simplest clustering algorithms
  - Randomly place k points in the feature space
    - These are the first cluster centers
  - For each cluster center form a set of data points by assigning each data point to the set with the closest center
    - Euclidian distance in feature space
  - Compute new cluster centers as mean of points in each set and repeat until point sets do not change

K-Means Example



K-Means Clustering

- K-Means clustering requires use of particular representation and performance functions
  - Similarity function has to be Cartesian distance
  - Performance function is average squared distance from cluster mean
    - If these assumptions are violated then the mean of the data set for the cluster might no longer be the right pick for the new cluster center
- K-Means performs iterative local optimization on the average squared (Cartesian) distance

K-Means Clustering

- If the similarity metric is not Cartesian distance the algorithm has to be adapted
  - Mean calculation for new cluster center is no longer valid
    - Often it is not even defined (e.g. for certain types of graphs or metrics such as DTW)
  - Use the central element of the set as a cluster prototype (instead of mean)
    - Maintains the property of optimizing the average Cartesian distance in each cluster
    - But: is no longer K-Means clustering !

K-Means Clustering

- K-Means and related techniques are very simple and frequently used clustering techniques
  - Sensitive to the starting point
  - High complexity due to the iterative optimization
  - Prone to local extrema
  - Requires to pre-determine number of clusters
    - Can repeat it with different cluster numbers and see whether the performance increases significantly

Hierarchical Clustering

- Hierarchical clustering forms a hierarchy of clusters (i.e. different number of clusters at each level of the hierarchy).
  - Divisive clustering starts with a single cluster and divides clusters recursively according to a metric
    - Generally exponential complexity to find best cluster split
  - Agglomerative clustering starts with one cluster per data point and recursively merges clusters according to a cluster similarity measure
    - Complexity varies based on similarity metric used

Agglomerative Clustering

- Agglomerative clustering incrementally merges clusters based on cluster similarity
  - Start with one cluster for each data point
  - Find the two most "similar" clusters and merge them and repeat until either only one cluster is left or until the clusters become too dissimilar
- Need to define which two clusters are most similar to each other
  - Similarity of two data sets (not data points)
    - Often called linkage

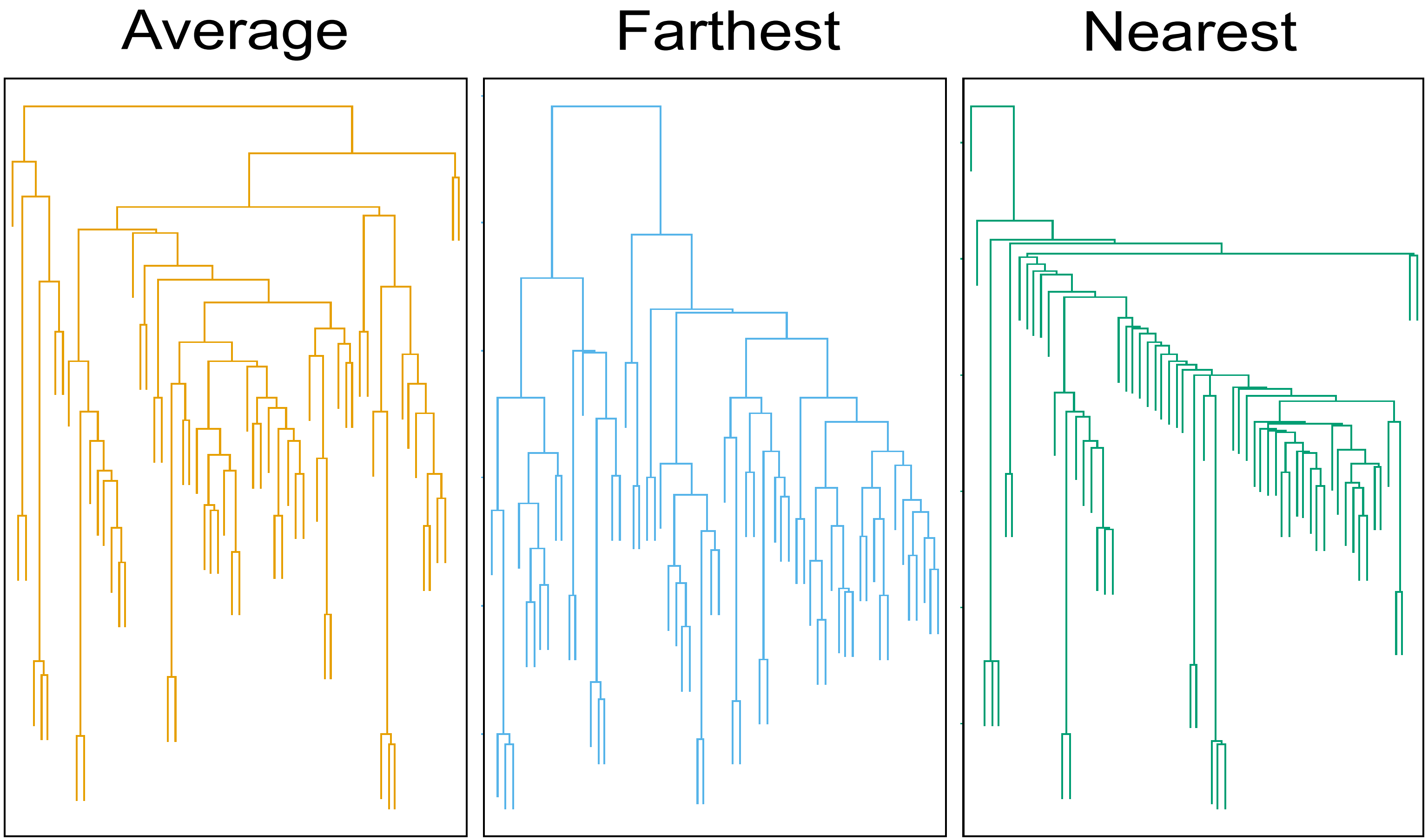
Linkage Criteria

- Agglomerative clustering can use different linkage (cluster similarity) measures
  - Average (mean) linkage: average similarity between any two items from the two clusters
  - Single (minimum) linkage: similarity of the two most similar items in the two clusters
  - Complete (maximum) linkage: similarity of the two most dissimilar items in the two clusters
- Linkage influences complexity and clusters

Linkage Criteria

- Average linkage forms most "uniform" clusters
  - Average linkage requires recomputation of the linkage after every merge using $O(n^2)$ time
    - Clustering is $O(n^3)$
- Complete linkage forms most "compact" clusters
  - Complete linkage recomputation uses MAX ($O(n)$)
    - Clustering is $O(n^2)$
- Single linkage forms most "connected" clusters
  - Single linkage recomputation uses MIN ($O(n)$)
    - Clustering is $O(n^2)$

Example



Hastie

Hierarchical Clustering

- Hierarchical clustering (agglomerative or divisive) form a hierarchy of clusters
  - Need to decide which clusters to use
    - Change to intra-cluster variances
    - Maximum linkage for merging
    - Ratio of inter-cluster and intra-cluster variances
- Hierarchical clustering has a fixed maximum run time that depends on size of data set
  - Agglomerative linkage criteria influence run time
    - Complexity of computing linkage is higher for average
    - Average number of merge operations is different

Probabilistic (Soft) Clustering

- So far a data point had to belong to one cluster
  - For fixed cluster sizes this can result in cluster boundaries that go through dense data
    - Does not look like clusters
  - This makes clusters sensitive in noisy situations
  - Does not allow overlapping clusters
- Probabilistic (Soft) clustering removes this assumption by using the probability that a point belongs to a cluster, $P(z/x^{(i)})$

Probabilistic Clustering

- Like in Bayesian (probabilistic) classification, probabilistic clustering takes a generative view

  $$P(z \mid x) = \frac{p(x \mid z)P(z)}{p(x)}$$

  - But, cluster labels are not given so they need to be predicted based on a cluster "purity" criterion
    - Clusters should have a particular shape and as cleanly as possible reflect the data
      - Maximize marginal likelihood given a particular type of cluster distribution

$$\hat{\theta} = \arg\max_{\theta} \prod_{i} p(x^{(i)}) = \operatorname{argmax}_{\theta} \prod_{i} \sum_{c_j} p(x^{(i)}, z = c_j)$$

Gaussian Mixture Models

- In Gaussian Mixture Models each cluster is represented by a multivariate Gaussian distribution

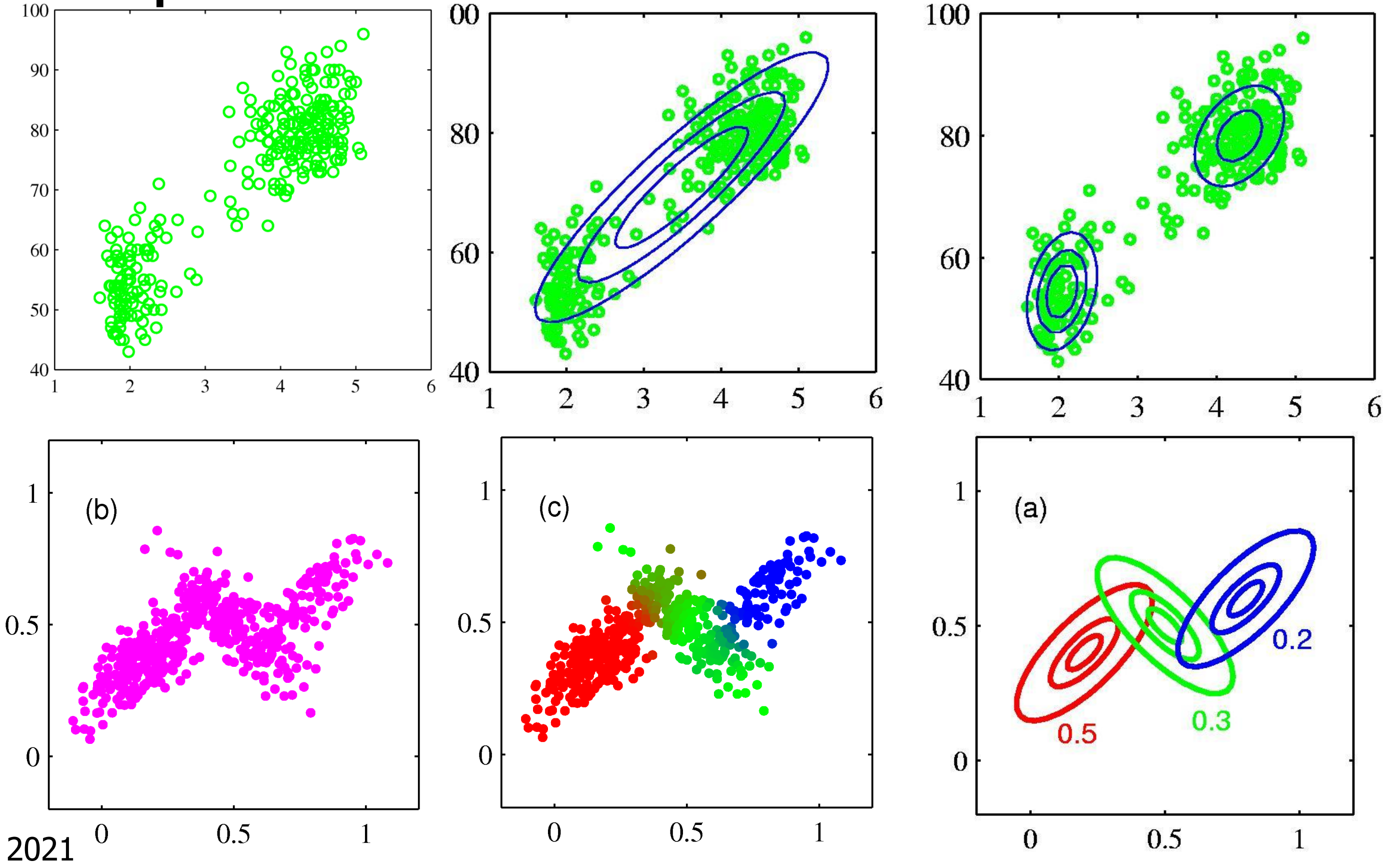
$$p(x \mid z_i) = \frac{1}{(2\pi)^{\frac{m}{2}} \left\|\Sigma_i\right\|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i (x-\mu_i)}$$

  - Cluster assignments are then done probabilistically

$$P(z \mid x) = \frac{p(x \mid z)P(z)}{p(x)}$$

Gaussian Mixture Models

- Examples



Gaussian Mixture Model

- Learning is performed by maximizing the marginal likelihood

$$\hat{\theta} = \text{argmax}_\theta \prod_i p(x^{(i)}) = \text{argmax}_\theta \prod_i \sum_{c_j} p(x^{(i)}, z = c_j)$$

$$= \text{argmax}_\theta \prod_i \sum_{c_j} p(x^{(i)} \mid z = c_j) P(z = c_j)$$

$$= \text{argmax}_\theta \prod_i \sum_{c_j} \frac{1}{(2\pi)^{\frac{m}{2}} \left\|\Sigma_j\right\|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(x^{(i)} - \mu_j\right)^T \Sigma_j \left(x^{(i)} - \mu_j\right)} P(z = c_j)$$

  - Requires learning Gaussian parameters for *p(x|z)* as well as prior cluster probabilities *P(z)*
    - Difficult to solve and with many local extrema

Expectation Maximization

- Expectation Maximization (EM) is a general algorithm to maximize marginal likelihood

$$\hat{\theta} = \operatorname{argmax}_\theta \prod_i \sum_{c_j} p(x^{(i)}, z = c_j \mid \theta) = \operatorname{argmax}_\theta \sum_i \log\left(\sum_{c_j} p(x^{(i)}, z = c_j \mid \theta)\right)$$

  - Performs optimization by iterating two steps
    - Expectation step: computes the expectations for the hidden/missing variables given the current parameters
    - Maximization step: optimizes the parameters with a weighted estimate (optimizing lower bound)
  - Similar to coordinate ascent
    - In the GMM case it is similar to the way K-Means does its optimization

Expectation Maximization

- Expectation step:
  - Computer expected distribution for hidden variables

$$P(z_j \mid x^{(i)}, \theta_t) = \alpha p(x^{(i)} \mid z_j, \theta_t) P(z_j \mid \theta_t)$$

- Maximization step:
  - Recompute the optimal parameters for the expected log likelihood (a lower bound on the likelihood)

$$\theta_{t+1} = \mathrm{argmax}_\theta \sum_i \sum_j P(z_j \mid x^{(i)}, \theta_t) \log p(z_j, x^{(i)} \mid \theta)$$

$$= \mathrm{argmax}_\theta \sum_i E_z\left[\log p(z_j, x^{(i)} \mid \theta)\right]$$

Optimise lower bound (bo

Expectation Maximization

- Expectation maximization effectively conducts coordinate ascent on the lower bound of the log likelihood
  - Still maximizes the original likelihood
    - Expectation step tightens the bound
      - Ensures that bound eventually has same extremum as function
  - Significantly simpler than original optimization
    - Often both steps are analytically solvable
  - Guaranteed convergence
    - But: Only to local optimum, making start point important

EM for GMM

- Applying EM to GMM:
  - Expectation step computes the expected cluster values for the data points

    $$P(z_j \mid x^{(i)}, \theta_t) = \alpha p(x^{(i)} \mid z_j, \theta_t) \tilde{P}(z_j \mid \theta_t)$$

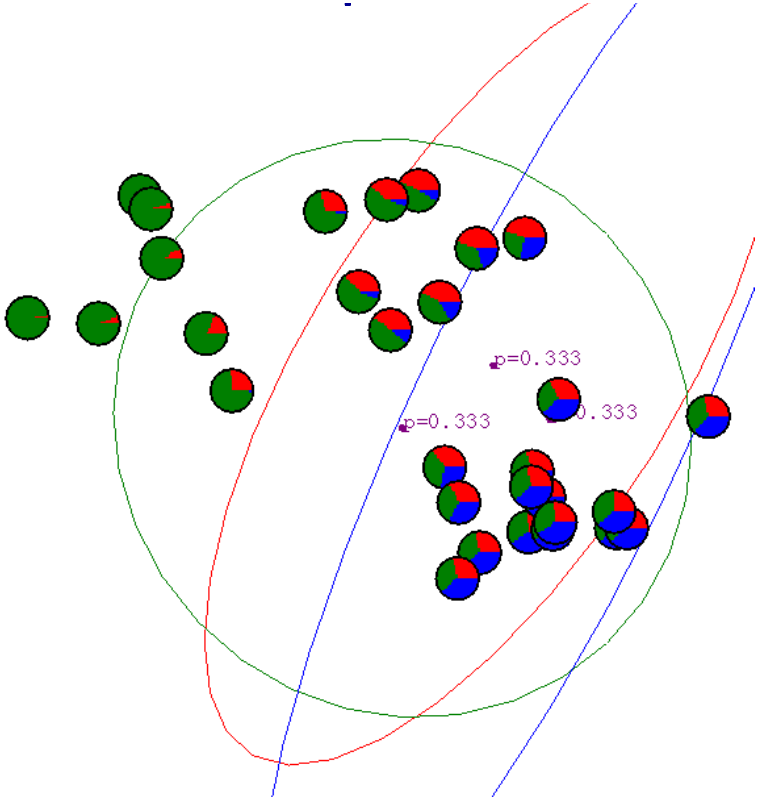
  - Maximization step re-computes the means and variances of the Gaussian distributions for each cluster as well as the cluster priors

    $$\tilde{P}(z_j \mid \theta_{t+1}) = \frac{\sum_i P(z_j \mid x^{(i)}, \theta_t)}{n} \quad , \quad \mu_j(t+1) = \frac{\sum_i P(z_j \mid x^{(i)}, \theta_t) x^{(i)}}{\sum_i P(z_j \mid x^{(i)}, \theta_t)}$$

    $$\Sigma_j(t+1) = \frac{\sum_i P(z_j \mid x^{(i)}, \theta_t)\left(x^{(i)} - \mu_j(t+1)\right)\left(x^{(i)} - \mu_j(t+1)\right)^T}{\sum_i P(z_j \mid x^{(i)}, \theta_t)}$$

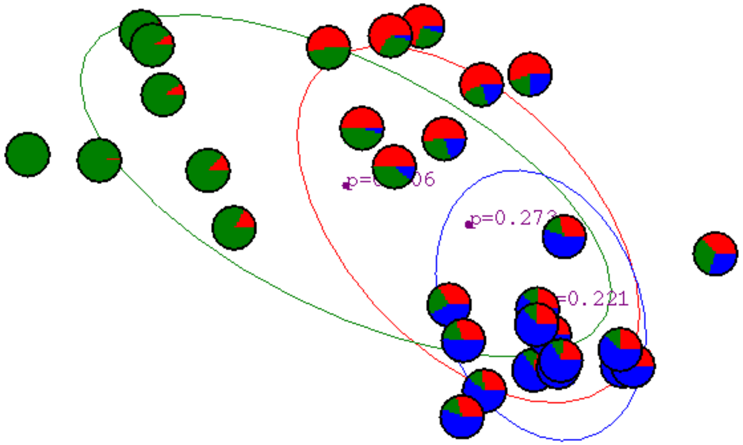
GMM Example

- Start with 3 clusters



GMM Example

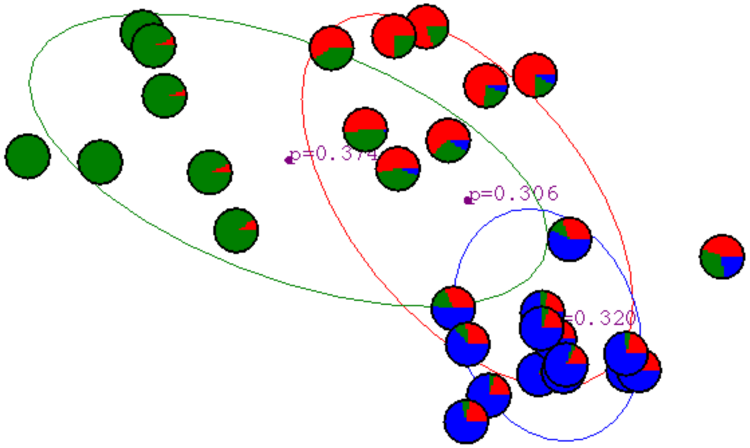
- First Iteration



© Manfred Huber 2021

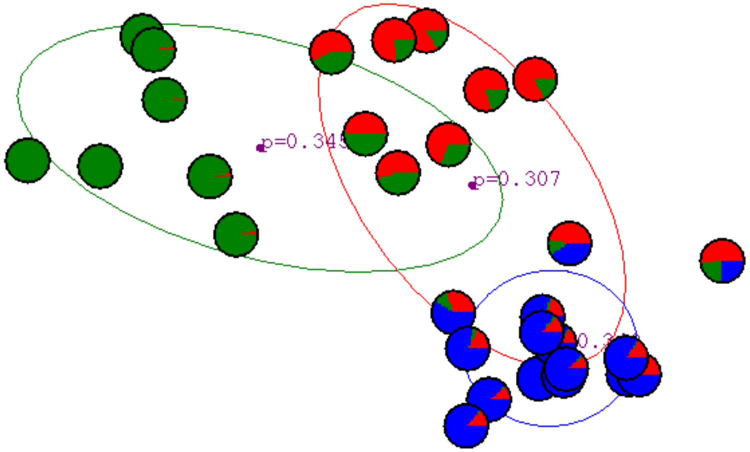
GMM Example

- Second Iteration



GMM Example

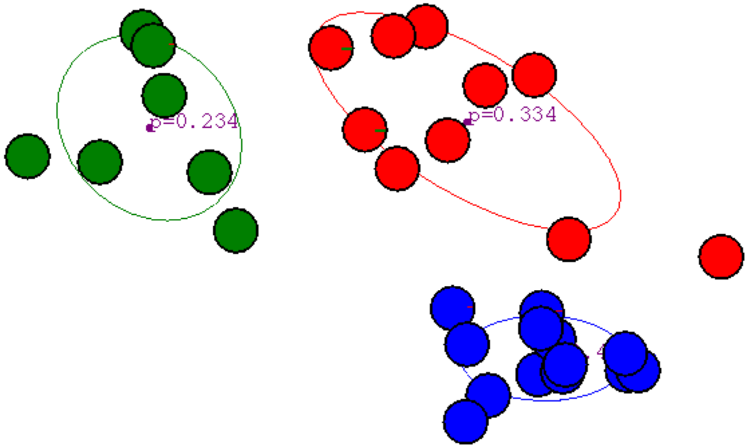
- Third Iteration



© Manfred Huber 2021

GMM Example

- 20th Iteration



Clustering

- There are a wide range of other clustering methods
  - Clustering methods for other similarity metrics
    - E.g.: Spectral clustering – graph similarity measure
  - Clustering using neural networks
    - E.g.: Self-organizing maps for topological clustering
  - Clustering using sampling methods
    - E.g.: Ant colony optimization-based clustering

Clustering

- Clustering is an unsupervised learning problem aimed at dividing data into groups
    - Like classification but without known classes
    - What makes good clusters has to be designed into the algorithm in the form of a similarity measure
- Deterministic clustering assigns each data point to exactly one cluster
    - K-Means clustering uses a fixed number of clusters
    - Hierarchical clustering builds hierarchy of clusters

Clustering

- Clustering can be used for a number of purposes
  - Identify different "types" within the data
    - "Type" can be used as a feature for subsequent tasks
    - In probabilistic clustering the probability vector over "types" can be used as a continuous feature vector
  - Identify different "causes" for the data
    - Can be used to identify whether the data generation process was uniform
  - Approximately compress the data set