Machine Learning Algorithms

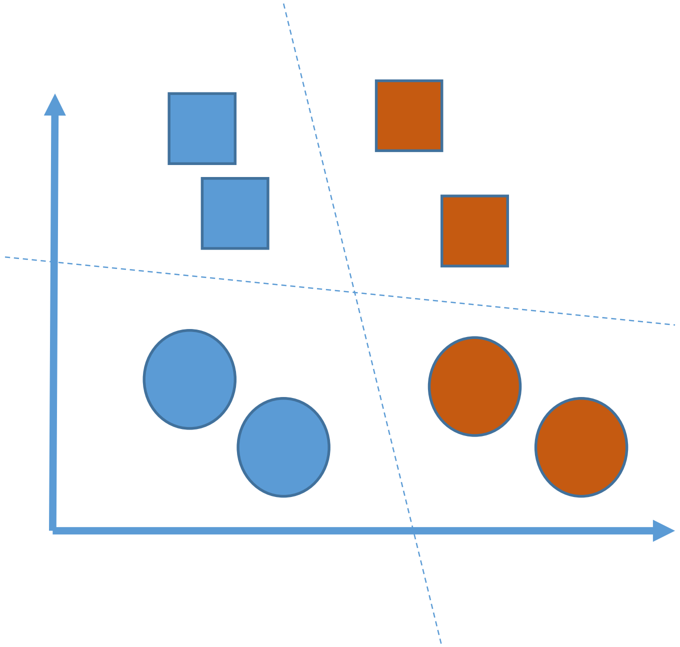
Finding Multiple Categories
(Unsupervised Learning)

Markus Rupp

23.9.2020

Options

- Find multiple hyperplanes that separate sets into subsets with distinct features
  - Then combine them (and/or) to define new subsets with particular features



Two hyperplanes destinguish between shape (circle,square) and color (blue, brown) allowing to distinguish between four sets

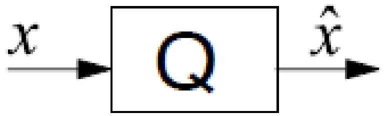
Options

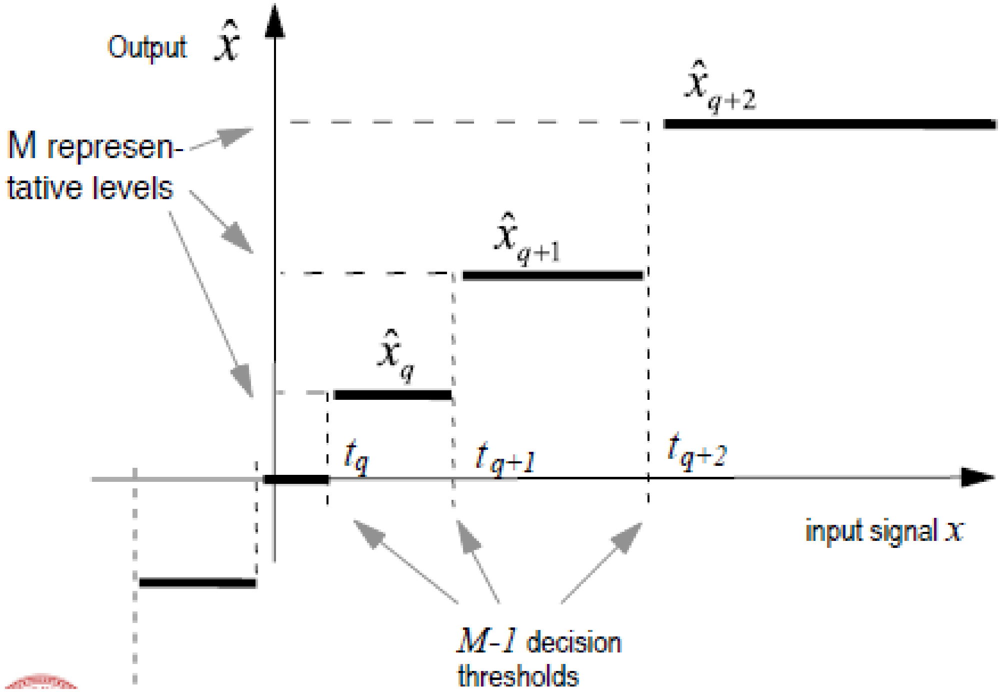
- This method works best if the sets are nicely visible as such and their number is known

- If not, another method with less supervision may result in better performance: k-means clustering algorithm

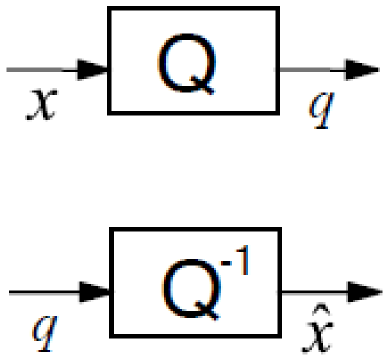
- This is a particular form of Lloyd‘s algorithm:
  - 1. Lloyd‘s algorithm
  - 2. k-means clustering algorithm
  - 3. k-nearest neighbors algorithm

Consider Scalar Quantization





Sometimes, this convention is used:



Lloyd‘s Algorithm (1957,1982)

- Problem: we want to map quantization levels $\hat{x}_q$ being valid in the interval $\left[t_q, t_{q+1}\right]$
  Let us assume M+1 quantization levels: q=0..M
  →M thresholds $t_1,\ldots,t_M$

Lloyd‘s Algorithm (1957,1982)

- Minimize
  Mean Squared Error
  (MSE):

$$
\begin{aligned}
MSE = &\int_{-\infty}^{t_1} \left(x-\hat{x}_0\right)^2 f_{\mathrm{x}}(x)dx \\
&+\int_{t_1}^{t_2} \left(x-\hat{x}_1\right)^2 f_{\mathrm{x}}(x)dx \\
&+\ldots \\
&+\int_{t_{M-1}}^{t_M} \left(x-\hat{x}_{M-1}\right)^2 f_{\mathrm{x}}(x)dx \\
&+\int_{t_M}^{\infty} \left(x-\hat{x}_M\right)^2 f_{\mathrm{x}}(x)dx
\end{aligned}
$$

Lloyd‘s Algorithm (1957,1982)

- Minimize
  Mean Squared Error

  (MSE): $\min_{\{t_1,\ldots,t_N,\hat{x}_0,\ldots,\hat{x}_N\}} MSE = ?$

- Build derivative with respect to $\hat{x}_q$

$$\frac{\partial MSE}{\partial \hat{x}_q} = \frac{\partial}{\partial \hat{x}_q} \int_{t_q}^{t_{q+1}} \left(x - \hat{x}_q\right)^2 f_{\mathbf{x}}(x)dx = 0$$

$$\int_{t_q}^{t_{q+1}} \left(x - \hat{x}_q\right) f_{\mathbf{x}}(x)dx = 0$$

$$\hat{x}_q = \frac{\int_{t_q}^{t_{q+1}} x f_{\mathbf{x}}(x)dx}{\int_{t_q}^{t_{q+1}} f_{\mathbf{x}}(x)dx} = \frac{m_q}{p_q}$$

Lloyd‘s Algorithm

- (Un)fortunately, this is not sufficient and only **<u>necessary</u>** to find the thresholds.
- Lloyd found a second condition:

$$t_q = \frac{\hat{x}_{q-1} + \hat{x}_q}{2}$$

- (Un)fortunately ,this is still not sufficient and only **<u>necessary</u>** to find the thresholds.

Lloyd‘s Algorithm

- 1. Guess initial set of representative levels $\hat{x}_q$ and corresponding probabilities p_q and MSE.
- 2. Calculate M decision thresholds
$$t_q = \frac{\hat{x}_{q-1} + \hat{x}_q}{2}$$
- 3. Calculate M+1 new representative levels $\hat{x}_q$ and probabilities p_q.
$$\hat{x}_q = \frac{m_q}{p_q}$$
- 4. Compute new MSE.
- 5. Repeat 2 and 3 until MSE is no longer getting smaller .

Lloyd‘s Algorithm

- Example: Consider symmetric pdf (uniform distribution between -1 and 1 and M=2 that is three quantization levels $\hat{x}_0, \hat{x}_1, \hat{x}_2$.
  Clearly, $\hat{x}_0 = -\hat{x}_2$
  For the two thresholds we find: $t_1 = -t_2$
- Exercise: define the MSE=f($\hat{x}_0, \hat{x}_1, t_1$)

differentiate w.r.t. the three parameters .

What is the Minimum (MSE),
what are the best values?

Example

- MSE

$$MSE = \frac{(t_1 - x_0)^3}{3} + \frac{(1 + x_0)^3}{3} - \frac{t_1(t_1^2 + 3x_1^2)}{3}$$

$$\frac{\partial}{\partial x_1} = 0 \rightarrow x_1 = 0$$

$$MSE = \frac{(t_1 - x_0)^3}{3} + \frac{(1 + x_0)^3}{3} - \frac{t_1^3}{3}$$

$$\frac{\partial}{\partial t_1} \rightarrow x_0 = 2t_1$$

$$\frac{\partial}{\partial x_0} = 0 \rightarrow t_1 = -\frac{1}{3} \rightarrow x_0 = -\frac{2}{3} \rightarrow MSE = \frac{1}{81}$$

K-Means Clustering Algorithm

- The term "*k*-means" was first used by James MacQueen in 1967, though the idea goes back to Hugo Steinhaus in 1957.
- The standard algorithm was first proposed by Stuart Lloyd in 1957 as a technique for pulse-code modulation, though it wasn't published outside of Bell Labs until 1982.
- In 1965, E. W. Forgy published essentially the same method, which is why it is sometimes referred to as Lloyd-Forgy

K-Means Clustering Algorithm

- The problem is to find k clusters, defined by their means (centroids), given a large amount of N data points (feature vectors).
- In that way, the entire set S of data points is split into k disjoint subsets, with not necessarily identical amounts of elements.
- Although the algorithm converges fast, it is not necessarily finding the optimal solution.
- Its solution depends on the starting values
- A good starting set is to select k points with relatively large distance from each other.

K-Means Clustering Algorithm

- Find subsets S_i with means $\underline{\overline{x}}_i$; i=1,2,…,k in set S

$$\arg\min_S \sum_{i=1,2,\ldots k} \sum_{\underline{x}_j \in S_i} \left\| \underline{x}_j - \underline{\overline{x}}_i \right\|^2 = \arg\min_S \sum_{i=1}^{k} |S_i| \operatorname{var}(S_i)$$

- Simple search algorithm by testing all data points $\underline{x}_j$; j=1,2,….,N
- Test for all clusters i=1,2,…,k and all data points $\underline{x}_j$; j=1,2,….,N
  - $\underline{x}_j$ belongs to S_i if

$$\left\| \underline{x}_j - \underline{\overline{x}}_i \right\|^2 < \left\| \underline{x}_j - \underline{\overline{x}}_m \right\|^2 \; m = 1, 2., , , , k, m \neq i$$

  - Update mean values according to new assignments
  - Continue doing so

Variations

- *k*-medians clustering uses the median in each dimension instead of the mean, and this way minimizes the L_1 norm

- A different algorithm though is the **k nearest neighbors** algorithm
  - Here, clusters are already defined and new data points arrive.
  - By a simple majority vote (k=2m-1) of the k neighbors, one can assign the new data point.
  - Example: ask k=5 neighbors, m=3 belong to a particular cluster→decision