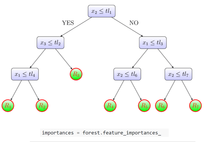
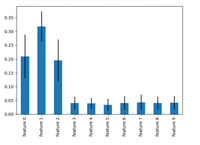
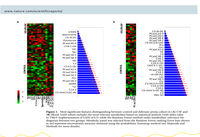






Classification loss

$H = -\sum P(x) \log P(x)$ Entropy

Logistic regression (in binary classification)

Sigmoid activation function

$$J(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}log\left(\hat{p}^{(i)}\right) + \left(1-y^{(i)}\right)log\left(1-\hat{p}^{(i)}\right)\right]$$

Logistic Regression loss (binary classification)

more than two classes

$$J(\boldsymbol{\Theta}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log\left(\hat{p}_k^{(i)}\right)$$

in network

belong to class k

Softmax activation function

[1 0 0 0 0]

[15 9 15 5 20]

[0.0 0.03 0.07 0.93 0.03]

$\Sigma = 1$

$$\begin{aligned} \mathrm{KL}(p\|q) &= -\int p(\mathbf{x}) \ln q(\mathbf{x})\, \mathrm{d}\mathbf{x} - \left(-\int p(\mathbf{x}) \ln p(\mathbf{x})\, \mathrm{d}\mathbf{x}\right) \\ &= -\int p(\mathbf{x}) \ln \left\{\frac{q(\mathbf{x})}{p(\mathbf{x})}\right\} \mathrm{d}\mathbf{x}. \end{aligned} \tag{1.113}$$

(from earlier)

This is known as the *relative entropy* or *Kullback-Leibler divergence*, or *KL divergence* (Kullback and Leibler, 1951), between the distributions $p(\mathbf{x})$ and $q(\mathbf{x})$. Note that it is not a symmetrical quantity, that is to say $\mathrm{KL}(p\|q) \neq \mathrm{KL}(q\|p)$.

Clustring methods

$X_{m \times n}$

$x_1 \quad x_2 \quad \cdots \quad x_n$

p_1
p_2
$\vdots$
p_m

← training set

n neurons

$C = [x_1 \quad x_2 \quad \cdots \quad x_n]$

if C is a vector with 8 components then by a metric we can compare C_i to the 1000 points in traing set

if we have another point C', then we can find that a point (e.g. P_{200}) is closest to C or C'. Then we can assign it to C or C'.

Suppose 600 are close to C and 400 are close to C'

Then we have

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \xrightarrow[\text{for } P_{200}]{\text{belong to}} C$$

$$\begin{bmatrix} \bar{x}_1 & \bar{x}_2 & \cdots & \bar{x}_n \end{bmatrix}$$

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \xrightarrow[\text{for } P_{400}]{\text{belong to}} C'$$

$$\begin{bmatrix} \bar{x}_1 & \bar{x}_2 & \cdots & \bar{x}_n \end{bmatrix}$$

Update (new centers)

$C \rightarrow C = \left[\bar{x}_1 \quad \bar{x}_2 \quad \cdots \quad \bar{x}_n\right]$

$C' \rightarrow C' = \left[\bar{x}_1 \quad \bar{x}_2 \quad \cdots \quad \bar{x}_n\right]$

k = 2

2 centers

K-Means

This is the most popular clustering algorithm

Lloyd Algorithm

Lloyd algorithm is the most popular way of implementing k-means.

Algorithm

- First we choose k - the number of clusters we want to get
- Then we randomly initialize k cluster centers (cluster centroids)

This is an iterative algorithm, and on each iteration it does 2 things:

- cluster assignment step
- move centroids step

Cluster Assignment Step:

- go through each example and choose the closest centroids
- and assign the example to it

Move Centroids Step:

- Calculate the average for each group
- and move the centroids there

Repeat this until converges

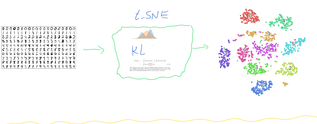
https://www.naftaliharris.com › blog › visualizing-k-me...

Visualizing K-Means Clustering - Naftali Harris

Jan 15, 2014 — The **k-means** algorithm captures the insight that each point in a **cluster** should be near to the center of that **cluster**. It works like this: first ...

t-distributed stochastic neighbor embedding (**t-SNE**) is a statistical method for visualizing high-dimensional data by giving each datapoint a location in a two or three-dimensional map. It is based on Stochastic Neighbor Embedding originally developed by Sam Roweis and Geoffrey Hinton,[1] where Laurens van der Maaten proposed the t-distributed variant.[2] It is a nonlinear dimensionality reduction technique well-suited for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions. Specifically, it models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points with high probability.

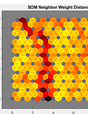
The t-SNE algorithm comprises two main stages. First, t-SNE constructs a probability distribution over pairs of high-dimensional objects in such a way that similar objects are assigned a higher probability while dissimilar points are assigned a lower probability. Second, t-SNE defines a similar probability distribution over the points in the low-dimensional map, and it minimizes the Kullback–Leibler divergence (KL divergence) between the two distributions with respect to the locations of the points in the map. While the original algorithm uses the Euclidean distance between objects as the base of its similarity metric, this can be changed as appropriate.

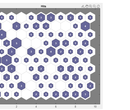


SOM (w,int)

A **self-organizing map** (**SOM**) or **self-organizing feature map** (**SOFM**) is an unsupervised machine learning technique used to produce a low-dimensional (typically two-dimensional) representation of a higher dimensional data set while preserving the topological structure of the data. For example, a data set with p variables measured in n observations could be represented as clusters of observations with similar values for the variables. These clusters then could be visualized as a two-dimensional "map" such that observations in proximal clusters have more similar values than observations in distal clusters. This can make high-dimensional data easier to visualize and analyze.

An SOM is a type of artificial neural network but is trained using competitive learning rather than the error-correction learning (i.e., backpropagation with gradient descent) used by other artificial neural networks. The SOM was introduced by the Finnish professor Teuvo Kohonen in the 1980s and therefore is sometimes called a **Kohonen map** or **Kohonen network**.[1][2] The Kohonen map or network is a computationally convenient abstraction building on biological models of neural systems from the 1970s[3] and morphogenesis models dating back to Alan Turing in the 1950s.[4]





three similar images, for example