MAXIMUM LIKELIHOOD ESTIMATION, K-MEANS

PRACTICALS MACHINE LEARNING 1, SS23

MAXIMUM LIKELIHOOD ESTIMATION

The setting

- **Our data:**

$$X = \{x_1, x_2, \ldots, x_N\}$$

$$x_n = \begin{bmatrix} x_n^1 \\ \vdots \\ x_n^d \end{bmatrix}$$

$$x_n \in \mathbb{R}^d$$

- **Our goal:** model X with probability distribution $p(x)$

Parametric/non-parametric modeling

- **Our goal:** model X with probability distribution $p(x)$,
  i.e., estimate, determine from what distribution the data at our disposal were sampled.

Deduce distribution from data

Non-parametric modeling
(kernel density estimation)

A-priori assumption about data distribution

Parametric modeling

Maximum Likelihood

Bayesian estimation
(e.g. Maximum A Posteriori)

Bayes' rule revisited

Bayes' rule

$$p(\Theta \mid X) = \frac{p(X \mid \Theta)\ p(\Theta)}{p(X)} \propto p(X \mid \Theta)\ p(\Theta)$$

Θ - model parameters

X - data

Bayes' rule revisited

Bayes' rule

$$p(\Theta \mid X) = \frac{p(X \mid \Theta)\ p(\Theta)}{p(X)} \propto p(X \mid \Theta)\ p(\Theta)$$

Posterior Likelihood Prior

Parametric modeling - Maximum Likelihood

- Assumption: $x_n \sim p(x_n \mid \Theta)$
- Parameters: Θ

- The functional form of $p(\,.\,)$ is known (we assumed it!) but not the parameters Θ.

- Samples $\{x_1, x_2, \ldots, x_N\}$ are independent and identically distributed (i.i.d.)

Likelihood function

- **The likelihood function** describes the *joint* probability of the observed data (X) as a function of parameters of the chosen model (Θ): $p(X \mid \Theta)$.

The joint probability, samples x_n i.i.d. $\Longrightarrow \prod_{n=1}^{N} p(x_n \mid \Theta)$

Maximum Likelihood "principle"

- "What was observed was *the most likely*" = observations in X are a representative sample

- For which parameter value do the observed data have the highest probability?

$$\hat{\Theta} = \arg\max_{\Theta} \prod_{n=1}^{N} p(x_n \mid \Theta)$$

Maximum Likelihood Estimation

- Given data $X = \{x_1, x_2, \ldots, x_N\}$, $x_n \in \mathbb{R}^d$ and i.i.d., and some belief about the model (e.g., probability density function):

1. Find likelihood of each single data point $p(x_n \mid \Theta)$
2. Find likelihood of the whole data set: $\prod_{n=1}^{N} p(x_n \mid \Theta)$
3. Transform likelihood into log-likelihood,

   (products become sums, $\prod \rightarrow \sum$): $\sum_{n=1}^{N} \ln(p(x_n \mid \Theta))$
4. Take $\frac{\partial}{\partial \Theta} \sum_{n=1}^{N} \ln(p(x_n \mid \Theta)) \stackrel{!}{=} 0$ to find maximum
5. Express Θ, and you are done.

Log transformation

- **Logarithm is a monotonically increasing function.**
  The log-likelihood has exactly the same maxima as the likelihood.
- **More numerically stable for small numbers.**
  Product of probabilities ($p(x_n) \in [0,1]$) would be a very small number.
  - $-\infty < \ln p(x_n) \leq 0$ ($\ln p(x_n)$ is a negative number)
- **Products become sums.**
  - Log. product rule: $\ln(p_1 \cdot p_2) = \ln p_1 + \ln p_2$

$$\ln \prod_i p_i = \sum_i \ln p_i$$

  - Easier to manipulate, especially when finding derivatives.

An example

- Assumption: 1D Gaussian $\rightarrow \Theta = \{\mu, \sigma^2\}$

1. Likelihood of each single data point: $p(x_n \mid \Theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_n - \mu)^2}{2\sigma^2}}$
2. Likelihood of the whole data set: $p(X \mid \Theta) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_n - \mu)^2}{2\sigma^2}}$
3. Transform into log-likelihood: $\ln p(X \mid \Theta) = -\frac{N}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{n=1}^{N} (x_n - \mu)^2$
4. Take the derivatives w.r.t. μ and σ^2 and calculate equations: $\frac{\partial}{\partial \mu} \ln p(X \mid \Theta) \stackrel{!}{=} 0$

   and $\frac{\partial}{\partial \sigma^2} \ln p(X \mid \Theta) \stackrel{!}{=} 0$
5. Express parameters: $\hat{\mu} = \frac{1}{N} \sum_{n=1}^{N} x_n$ and $\hat{\sigma^2} = \frac{1}{N} \sum_{n=1}^{N} (x_n - \mu)^2$

K-MEANS

K-Means

- Problem: Identifying groups (clusters) of data points in a multidimensional space
  - Given is: a data set $\{x_1, \ldots, x_N\}, x_n \in \mathbb{R}^d$
  - Goal: partition data points into K clusters (K is given or chosen)

$$\min_{\mu_1,\ldots,\mu_K,r} \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|^2$$

$$\text{s.t. } r_{nk} \in \{0,1\} \text{ and } \sum_k r_{nk} = 1,\ n = 1,\ldots,N$$

K-Means - Algorithm summary

- Goal: minimize J,

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|x_n - \mu_k\|^2$$

- Choose initial values for each μ_k (e.g., randomly)
- Repeat until convergence, or max. number of iterations reached:

  (1) Assign data points to clusters

  (2) Compute the cluster means

- Each phase reduces the value of J.
- Convergence is assured.
- But no guarantee to convergence to the global minimum of J.

K-Means - Algorithm

(1) Assign data points to clusters

How to assign the n-th data point to the closest cluster?

$$r_{nk} = \begin{cases} 1 \text{ if } k = \arg\min_j \|x_n - \mu_j\|^2 \\ 0 \text{ otherwise.} \end{cases}$$

- For each data point $x_n, n \in \{1,...,N\}$ assign a set of binary indicator variables $r_{nk} \in \{0,1\}$, where $k = 1,...,K$ describes which of K clusters the data point x_n is assigned to.
- If x_n is assigned to cluster k, then $r_{nk} = 1$, and $r_{nj} = 0$ for all $j \neq k$.
- This is known as **1-of-K** coding scheme, or **"one-hot"** encoding.

K-Means - Algorithm

(2) Compute the cluster means.

Minimize J w.r.t. μ_k (r_{nk} held fixed) gives:

$$2\sum_{n=1}^{N} r_{nk}(x_n - \mu_k) \stackrel{!}{=} 0$$

Solving for μ_k gives:

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

K-Means - Algorithm

(2) Compute the cluster means.

Minimize J w.r.t. μ_k (r_{nk} held fixed) gives:

$$2\sum_{n=1}^{N} r_{nk}(x_n - \mu_k) \stackrel{!}{=} 0$$

Solving for μ_k gives:

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

Number of points assigned to cluster k

K-Means - Algorithm

(2) Compute the cluster means.

Minimize J w.r.t. μ_k (r_{nk} held fixed) gives:

$$2\sum_{n=1}^{N} r_{nk}(x_n - \mu_k) \stackrel{!}{=} 0$$

Solving for μ_k gives:

$$\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$$

Number of points assigned to cluster k

Mean of all data points assigned to cluster k, and K clusters, hence K-Means!