CS5014 Machine learning

Maximum likelihood estimation

Lei Fang

01/02/2021

Contents

1 Maximum Likelihood Estimation

Maximum likelihood estimation is a general method we use to estimate statistical or machine learning models. It usually has the following ingredients.

- X, y: the observed dataset (y is optional for unsupervised learning)
- P: A probabilistic model, or likelihood model for $P(y|X;\theta)$, $P(X,y|\theta)$ or $P(X|\theta)$,
  - In other words, a data generating process that we have assumed for X, y
  - θ is the parameters of the likelihood function
  - For some supervised learning models (namely, discriminative models), the predictors X, are assumed fixed (we do not model them in the generating process), so we only model $P(y|X)$: the conditional distribution of y given X.
    - Examples include: linear regression, logistic regression, neural networks
  - For some other supervised learning models (generative models), we model both $P(X,y)$;
    - Examples are Naive Bayes, discriminant analysis etc.
  - For unsupervised learning, we model $P(X)$ only for obvious reasons

- θ: Parameters associatd with the model P

Note that $P(X|\theta)$, $P_\theta(X)$, $P(X;\theta)$ are used interchangeably. They all mean the same thing: conditional on the parameter θ, the probability distribution of X.

1.1 Likelihood function

The **likelihood** function is defined as $l(\theta) = P(\mathcal{D}|\theta)$: the conditional probability of observing dataset $\mathcal{D}$ given θ

- It is a function of θ: different input θ will output likelihoods of observing $\mathcal{D}$
- Depending on the problem, $\mathcal{D}$ can be $\mathcal{D} = \{y\}$ only (discriminative models), $\mathcal{D} = \{X, y\}$ (generative models) or $\mathcal{D} = \{X\}$ (unsupervised learning). Do not worry if you do not understand these different terminologies, we will cover them in detail soon. But you should know there are different ways to model the training data.

MLE is a technique that finds the "best" θ by maximising $l(\theta)$: the chance of seeing the data $\mathcal{D}$ given the input θ, which is noted as

$$\theta_{ML} = \underset{\theta}{\operatorname{argmax}}\, l(\theta) \tag{1}$$

We usually deal with the log transformed version of the likelihood function $\mathcal{L}(\theta) \equiv \log l(\theta)$ (log of products are sum of logs, which is easier to deal with), called log likelihood function, denoted with $\mathcal{L}$. Note that the maximum θ is invariant after the log transformation: because log function is a monotonically increasing function: i.e.

$$l(\theta_1) > l(\theta_2) \Leftrightarrow \mathcal{L}(\theta_1) > \mathcal{L}(\theta_2),$$

therefore,

$$\theta_{ML} = \underset{\theta}{\operatorname{argmax}}\, \mathcal{L}(\theta)$$

Example 1.1. Linear regression

We will revisit the linear regression case here. Linear regression model can be specified as following:

$$y^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} + e^{(i)},\ e^{(i)} \sim \mathcal{N}\left(0, \sigma^2\right)$$

- $i = 1, \ldots, m$: index of data samples (row index), n is the number of predictors
- $\boldsymbol{x}^{(i)} = [1, x_1^{(i)}, \ldots, x_n^{(i)}]^T$ is a $(n+1) \times 1$ vector;
  - $\boldsymbol{x}^{(i)}$ are assumed fixed values, or not random

- $e^{(i)}$, the prediction error is assumed Gaussian distributed with zero mean and variance σ^2
- $\boldsymbol{\theta}, \sigma^2$ are the model parameter.
- as $y^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)} + e^{(i)}$ is a linear function of $e^{(i)}$, therefore

  $$y^{(i)} \sim \mathcal{N}\left(\boldsymbol{\theta}^T \boldsymbol{x}^{(i)}, \sigma^2\right),$$

  which means $y^{(i)}$ is Gaussian distributed with mean $\boldsymbol{\theta}^T \boldsymbol{x}^{(i)}$ and variance σ^2. Note that a Gaussian likelihood's density function is:

  $$p(y^{(i)}|\boldsymbol{\theta}, \sigma^2, \boldsymbol{x}^{(i)}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2}{2\sigma^2}\right),$$

  the log transformed likelihood is then

  $$\log p(y^{(i)}; \boldsymbol{\theta}, \sigma^2, \boldsymbol{x}^{(i)}) = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2$$

The model parameters are $\boldsymbol{\theta}, \sigma^2$. The log likelihood function is

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}, \sigma^2) &= \log \prod_{i=1}^{m} p(y^{(i)}; \boldsymbol{\theta}, \sigma^2, \boldsymbol{x}^{(i)}) && (2)\\
&= \sum_{i=1}^{m} \log p(y^{(i)}; \boldsymbol{\theta}, \sigma^2, \boldsymbol{x}^{(i)}) && (3)\\
&= \sum_{i=1}^{m} -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2 && (4)\\
&= -\frac{m}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{m}(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2 && (5)
\end{aligned}$$

Eq (2) is true because of the definition of $\mathcal{L}$ and also we assume all observations are independent; Eq (3) is true because of log function. The third step simply plugs in the log transformed Guassian likelihood function. And the last step just applies summation to the individual entries. You are expected to verify the above steps by yourself!

It is then obvious that maximising $\mathcal{L}(\boldsymbol{\theta}, \sigma^2)$ with respect to $\boldsymbol{\theta}$ has the same effect as minimising the sum of square of error term:

$$\sum_{i=1}^{m}(y^{(i)} - \boldsymbol{\theta}^T \boldsymbol{x}^{(i)})^2,$$

which shows that $\theta_{ML} = \theta_{LS}$. So all the results we derived in Lecture 3 applies here:

$$\boldsymbol{\theta}_{ML} = \boldsymbol{\theta}_{LS} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y}.$$

For completeness, the gradient of $\mathcal{L}$ w.r.t $\sigma^2, \boldsymbol{\theta}$ are listed below:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} = -\frac{1}{\sigma^2}\sum_{i=1}^{m}(y^{(i)} - \boldsymbol{\theta}^T\boldsymbol{x}^{(i)})(\boldsymbol{x}^{(i)})^T = -\frac{1}{\sigma^2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta})^T\boldsymbol{X}; \tag{6}$$

$$\frac{\partial \mathcal{L}}{\partial \sigma^2} = -\frac{m}{2\sigma^2} + \frac{1}{2(\sigma^2)^2}\sum_{i=1}^{m}(y^{(i)} - \boldsymbol{\theta}^T\boldsymbol{x}^{(i)})^2 \tag{7}$$

I have used the chain rule to derive the gradient for $\boldsymbol{\theta}$. And note that we define gradients as row vectors, so $\nabla_{\boldsymbol{\theta}}\boldsymbol{\theta}^T\boldsymbol{x}^{(i)} = (\boldsymbol{x}^{(i)})^T$ (a row vector). You should verify the above by yourself! if you set the derivative to zero, then a closed form estimators can be found:

$$\boldsymbol{\theta}_{ML} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{8}$$

$$\sigma^2_{ML} = \frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - \boldsymbol{\theta}_{ML}^T\boldsymbol{x}^{(i)})^2 \tag{9}$$

The following exercises are optional. But if you are interested, please have a go. It should take less than 30 mins.

Exercises

The best way to understand a technique is to use it. Let's derive the MLE for Poisson distribution. If your dataset $\boldsymbol{y}$ are counting data, i.e. $y^{(i)}, i = 0, 1, 2, \ldots, m$, e.g. the number of cars in a car park, the number of visits to your website; a natural choice is Poisson distribution (or regression if you have predictors). A Poisson distribution's probability mass function (p.m.f) is

$$P(Y = k) = \frac{\theta^k e^{-\theta}}{k!},$$

where $\theta > 0$ is the model parameter, and Y can only take $k = 0, 1, 2\ldots$ Suppose you are given a dataset of m observations, $\boldsymbol{y} = \{y^{(i)}, i = 0, 1, 2, \ldots, m\}$, sampled from a Poission distribution with θ.

1) Identify the three ingredients of the estimation problem, i.e. what is the data $\mathcal{D}$, parameter θ, model P.
2) Plot the p.m.fs for some Poisson distributions for different $\theta = 1, 5, 10$.
3) Write down the log likelihood function $\mathcal{L}(\theta)$.
4) Derive the gradient of $\mathcal{L}$ w.r.t θ; and find the closed form ML estimator.

5) Use the reparameterisation trick to write a simple gradient ascent/descent algorithm to find θ_{ML} (check Lecture 5). Note that vanilla gradient descent might struggle as $\theta > 0$. You need to apply the reparameterisation trick here.

6) Try your algorithm with some artificially sampled data. Check this https://numpy.org/doc/stable/reference/random/generated/numpy.random.poisson.html for sampling from a Poisson distribution. And see whether your program can converge to your analytical solution.