Loss Functions

CS 4277 Deep Learning

Kennesaw State University

Loss Functions

The goal of a (eager) machine learning algorithm is to find the parameters of a model that produces the best possible mapping from inputs to outputs.

- We do this using a training data set $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$
- Training uses feedback from the mismatch betweeen the model's predicted $\hat{y}$s and the ground truth ys.
- A *loss function* returns a single number that represents this mismatch.
- So finding the best possible mapping from inputs to outputs reeduces to minimizing the loss function.

KENNESAW STATE UNIVERSITY

Density Estimation

Say we have N observations of a scalar x which we denote with $\mathbf{x} = (x_1, \ldots, x_N)$.

- Estimating the distribution given the data is known as *density estimation*.
- We must assume a distribution, so we're estimating the parameters of the distribution.
- We assume data points are drawn independently and are identically-distributed.
  - This is known as the i.i.d. assumption

Example: Likelihood of the Gaussian

Since our data set is i.i.d., the probability of the data set is

$$p(\mathbf{x}|\mu, \sigma^2) = \prod_{n=1}^{N} \mathcal{N}(x_n|\mu, \sigma^2)$$

This is known as the *likelihood function* for the Gaussian.

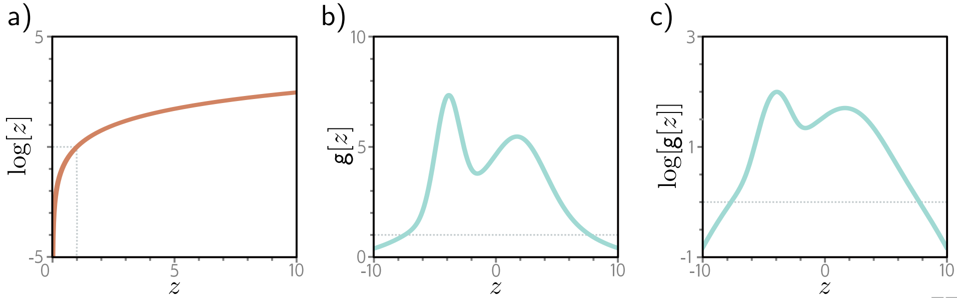
Maximum Likelihood

Finding the parameters of a distribution that maximize the probability of the observed data is known as *maximum likelihood estimation* (MLE).

In pracice, we transform likelihood functions into log likelihood functions.

Why log:

- Log of a function monotonically increasing and concave – $\operatorname{argmax} \ln(f) = \operatorname{argmax} f$
- Log easy to work with: $\ln(ab) = \ln(a) + \ln(b)$, $\ln(\frac{a}{b}) = \ln(a) - \ln(b)$, $\ln e^x = x$
- Multiplying probabilities can underflow – summing logs avoids this problem



Log Likelihood of Gaussian

For the Gaussian likelihood function we saw earlier, the log likelihood

$$
\begin{aligned}
p(\mathbf{x}|\mu,\sigma^2) &= \prod_{n=1}^{N} \mathcal{N}(x_n|\mu,\sigma^2) \\
p(\mathbf{x}|\mu,\sigma^2) &= \prod_{n=1}^{N} \frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}} \exp(-\frac{1}{2\sigma^2}(x_n-\mu)^2) \\
\ln p(\mathbf{x}|\mu,\sigma^2) &= \sum_{n=1}^{N} \ln(\frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}} \exp(-\frac{1}{2\sigma^2}(x_n-\mu)^2)) \\
\ln p(\mathbf{x}|\mu,\sigma^2) &= \sum_{n=1}^{N} \ln(1) - \ln(\sqrt{2\pi}) - \ln(\sigma) - \frac{(x_i-\mu)^2}{2\sigma^2}
\end{aligned}
$$

Maximum Likelihood of Gaussian

If we take the partial derivative of the Gaussian log likelihood function with respect to μ, set it to zero, and solve for μ we get:

$$\mu_{ML} = \frac{1}{N}\sum_{n=1}^{N} x_n$$

If we take the partial derivative with respect to σ^2 we get:

$$\sigma^2_{ML} = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu_{ML})^2$$

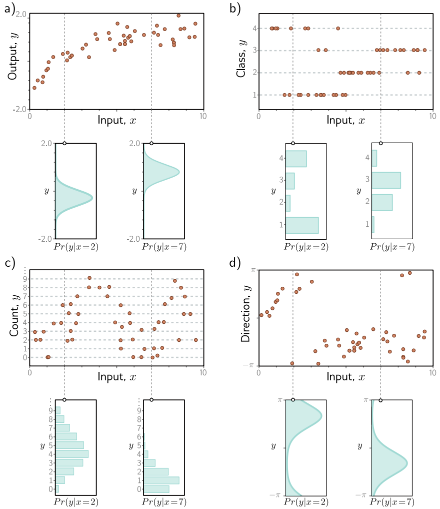
These should look familiar. They are the sample mean and sample variance of the Gaussian.

KENNESAW STATE UNIVERSITY

Loss Functions and Machine Learning Models

We seek a model $\boldsymbol{f}(\boldsymbol{x}, (\phi))$ that computes a $\hat{\boldsymbol{y}}$ given an $\boldsymbol{x}$.

- We can recast this problem as the computation of a conditional probability $p(\boldsymbol{y}_i \ \boldsymbol{x}_i)$.
- Minimizing the loss corresponds to maximizing this conditional probability.



General Maximum Likelihood Criterion

We choose a parametric distribution defined over the output domain $\boldsymbol{y}$ then train our model to compute the paramters, $\boldsymbol{\theta}$ of this distribution.

- If we choose a Gaussian distribution, then $\boldsymbol{\theta} = \{\mu, \sigma^2\}$.

We want to find the parameters of the model $\hat{\boldsymbol{\phi}}$ that maximize the conditional probability distribution $P(\boldsymbol{y}_i|\boldsymbol{\theta}_i)$ for all $\boldsymbol{y}_i$s.

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmax}}\left(\prod_{i=1}^{I} p(\boldsymbol{y}_i|\boldsymbol{x}_i)\right)$$

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmax}}\left(\prod_{i=1}^{I} p(\boldsymbol{y}_i|\boldsymbol{\theta}_i)\right)$$

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmax}}\left(\prod_{i=1}^{I} p(\boldsymbol{y}_i|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi}))\right)$$

Maximizing Log-Likelihood

Recalling that the total likelihood of the training data is:

$$P(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_I | \boldsymbol{x}_1, \ldots, \boldsymbol{x}_I) = \prod_{i=1}^{I} p(\boldsymbol{y}_i | \boldsymbol{x}_i)$$

which is impractical due to underflow, so we prefer to maximize the log-likelihood:

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmax}}(\prod_{i=1}^{I} p(\boldsymbol{y}_i | \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi}))$$

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmax}}(\log(\prod_{i=1}^{I} p(\boldsymbol{y}_i | \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi}))$$

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmax}}(\sum_{i=1}^{I} log(p(\boldsymbol{y}_i | \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi}))$$

KENNESAW STATE UNIVERSITY

Minimizing Negative Log-Likelihood

By convention we minimize the loss function. We can turn a maximization problem into a minimization problem by multiplying by -1.

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmin}}(-\sum_{i=1}^{I} log(p(\boldsymbol{y}_i|\boldsymbol{f}(\boldsymbol{x},\boldsymbol{\phi}))$$

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmin}}(L(\boldsymbol{\phi}))$$

Which is the final form of our loss function $L(\boldsymbol{\phi})$

Inference

Our network now computes a probability distribution over $\boldsymbol{y}$ instead of predicting $\hat{y}$. To get a prediction, we return the maximum of the distribution:

$$\hat{\boldsymbol{y}} = \underset{y}{\operatorname{argmax}}(p(\boldsymbol{y}|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi})))$$

This is often computed in terms of the parameters $\boldsymbol{\theta}$ predicted by the model. E.g., for Gaussian the maximum is at μ

Recipe for Constructing and Using Loss Functions

Now that we understand MLE for loss functions, we can create a recipe for constructing loss functions for training data $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{I}$ using the maximum likelihood approach:

1. Choose a suitable probability distribution $P(\boldsymbol{y}|\boldsymbol{\theta})$ defined over the predictions (output domain) $\boldsymbol{y}$ with distributoin parameters $\boldsymbol{\theta}$.
2. Set the machine learning model $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi})$ to predict one or more of these parameters, so $\boldsymbol{\theta} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi})$ and $P(\boldsymbol{y}|\boldsymbol{\theta}) = P(\boldsymbol{y}|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi}))$.
3. To train the model, find the network paramters $\hat{\boldsymbol{\phi}}$ that minimize the negative log-likelihood loss function over the training data pairs $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}$:

$$\hat{\boldsymbol{\phi}} = \underset{\boldsymbol{\phi}}{\operatorname{argmin}}(L(\boldsymbol{\phi})) = \underset{\boldsymbol{\phi}}{\operatorname{argmin}}(-\sum_{i=1}^{I} log(p(\boldsymbol{y}_i|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi}))$$

4. To perform inference for a new test example $\boldsymbol{x}$, return either the full distribution $P(\boldsymbol{y}|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\phi}))$ or the value where the distribution is maximized.