INT6151 Machine Learning
Lecture 3 - Classification

Ta Viet Cuong

VNU-UET

2024

Table of content

Recap: Logistic Regression - Binary Classification

Data

$$x \in \mathbb{R}^d, y \in \{0, 1\}$$

Example: image $S \times S \longrightarrow d = S^2 = 784$ (MNIST)

$$D = \{(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)\}$$

Model

$$f(x) = w^T x + w_0$$

$$Y|X = x \sim Ber(y|\sigma(f(x)))$$

Sigmoid Function

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Parameter

$$\theta = (w, w_0)$$

Recap: Logistic Regression - Binary cross entropy loss

Training

With MLE principle, calculate likelihood

$$L(w, w_0) = P(D) = \prod_{i=1}^{n} P(y_i|x_i) = \prod_{i=1}^{n} \mu_i^{y_i}(1-\mu_i)^{1-y_i}$$

where $\mu_i = \sigma(f(x_i))$

Negative-loglikelihood (NLL)

$$\ell(w, w_0) = -\log L(w, w_0) = \sum_{i=1}^{n} -y_i \log \mu_i - (1-y_i)\log(1-\mu_i)$$

Binary cross entropy loss function - BCE

Recap: Training Algorithm - Gradient Descent

TrainLR-GD$(D, \lambda) \rightarrow w, w_0$:

1. Initialize: $w = 0 \in \mathbb{R}^d$, $w_0 = 0$
2. Loop *epoch* $= 1, 2, \ldots$
   a. Calculate $\ell(w, w_0)$
   b. Calculate derivative $\nabla_w \ell, \nabla_{w_0} \ell$
   c. Update params

$$w \leftarrow w - \lambda \nabla_w \ell(w, w_0)$$
$$w_0 \leftarrow w_0 - \lambda \nabla_{w_0} \ell(w, w_0)$$

3. Stop when:
   a. Epoch is large enough
   b. The Loss function decrease negligible
   c. The derivative is small enough $\|\nabla_w \ell\|, \nabla_{w_0} \ell$

Multi-class logistics regression model

Given a label set $\mathcal{Y} = \{1, 2, \ldots, C\}$

Categorical distribution

A random variable $Y \sim \mathrm{Cat}(y|\theta_1, \theta_2, \ldots, \theta_C)$ means that

$$P(Y = c) = \theta_c, c = 1, 2, \ldots, C$$

with θ_c is the probability of the category c and $\sum_c \theta_c = 1$

For example: A six-side dice with $C = 6, \theta_c = 1/6, \forall c$

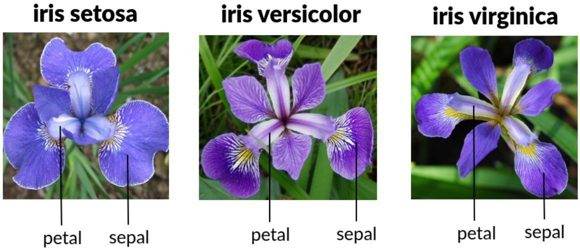
$$P(Y = y) = \prod_{c=1}^{C} \theta_c^{\mathbb{I}(c=y)}$$

where $\mathbb{I}(c = y)$ is an indicator random function denoting whether $c = y$ or not.

A dataset example with categorical labels (1)

Iris dataset:

- Number of Instances: 150
- Number of Attributes: 4 (sepal length/width in centimeters, petal length/width in centimeters)
- Number of classes: 3



A dataset example with categorical labels (2)

MNIST dataset:

- Number of Instances: 60,000 training images and 10,000 testing images.
- Number of Attributes: 28×28
- Number of classes: 10

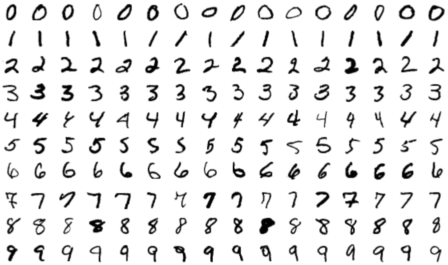


Figure: Sample images from MNIST test dataset (source: Wikipedia)

Multi-class logistic regression model

Model

Linear function with parameters $w_c \in \mathbb{R}^d, w_{c0} \in \mathbb{R},\ c = 1, 2, \ldots, C$

$$f_c(x) = w_c^T x + w_{c0}$$

with the application of Softmax function to convert a set of $(z_1, z_2, \ldots z_C)$ to probabilities

$$\mathcal{S}(z_1, z_2, \ldots z_C) = \left[\frac{e^{z_c}}{\sum_{c'=1}^{C} e^{z_{c'}}}\right]_{c=1,2,\ldots,C}$$

Probability model

$$Y|X = x \sim \mathrm{Cat}(y|\mathcal{S}(f_1(x), f_2(x), \ldots, f_C(x)))$$

where $f_c(x)$ is called as **logit**. With logistic regression, $f_c(x)$ are (simple) linear functions, more sophisticated methods make use of neural networks.

Multi-class logistic regression model

Inference

$h^{LR}(x)$: Choose c to maximize $h^{LR}(x)$,

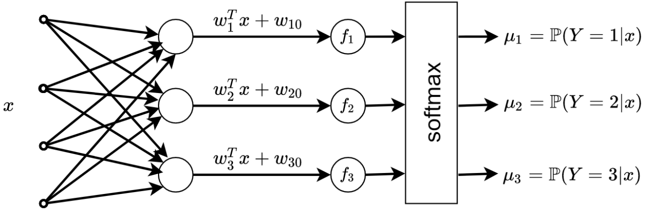


Figure: Multi-class logistic regression with Softmax

Multi-class logistic regression model

Training

The likelihood of the parameters with respect to the dataset:

$$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

$$L(\mathbf{W}) = \mathbb{P}(D) = \prod_{i=1}^{n} \mathbb{P}(Y = y_i | x_i)$$

$$= \prod_{i=1}^{n} \prod_{c=1}^{C} \mu_{ic}^{y_{ic}}$$

where μ_{ic} is computed as:

$$\mu_{ic} = \frac{e^{f_c(x_i)}}{\sum_{c'=1}^{C} e^{f_{c'}(x_i)}}$$

and the one-hot encoding of the labels

$$y_{ic} = \mathbb{I}(c = y_i)$$

Multi-class logistic regression model

Loss function: Negative log likelihood (NLL)

$$\ell(\mathbf{W}) = -\log L(\mathbf{W}) = -\sum_{i=1}^{n}\sum_{c=1}^{C} y_{ic}\log\mu_{ic}$$

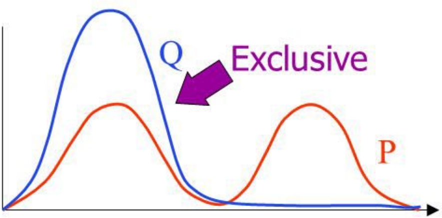
which is also called the cross-entropy loss (CE loss) function, it measures the Kullback-Leibler (KL) divergence between two distributions μ_{ic} and y_{ic}

Recall: KL divergence between two distributions P and Q

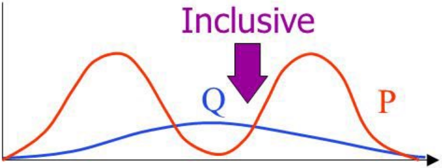
$$D_{\mathrm{KL}}(P\|Q) = \sum_{x\in\mathcal{X}} P(x)\log\frac{P(x)}{Q(x)}$$

Exclusive versus Inclusive

Minimising
$\mathrm{KL}(Q||P)$
$$= \sum_H Q(H) \ln \frac{Q(H)}{P(H|V)}$$



Minimising
$\mathrm{KL}(P||Q)$
$$= \sum_H P(H|V) \ln \frac{P(H|V)}{Q(H)}$$



Exclusive: Mode-seeking — Inclusive: Mean-seeking

Multi-class logistic regression model

Loss function: Negative log likelihood (NLL)

$$\ell(\mathbf{W}) = -\log L(\mathbf{W}) = -\sum_{i=1}^{n}\sum_{c=1}^{C} y_{ic} \log \mu_{ic}$$

which is also called the cross-entropy loss (CE loss) function, it measures the Kullback-Leibler (KL) divergence between two distributions μ_{ic} and y_{ic}

Question: What is the equivalent form of KL divergence of NLL, is it $\mathbb{E}[D_{\mathrm{KL}}(y\|\mu)]$ or $\mathbb{E}[D_{\mathrm{KL}}(\mu\|y)]$? Can we reverse the order? Why?

Multi-class logistic regression model

Gradient descend

The gradient[1] of the loss function w.r.t the loss function are

$$\nabla_{w_c}\ell(\mathbf{W}) = \sum_{i=1}^{n}(\mu_{ic} - y_{ic})x_i$$

$$\nabla_{w_{c0}}\ell(\mathbf{W}) = \sum_{i=1}^{n}(\mu_{ic} - y_{ic})$$

[1] https://deepnotes.io/softmax-crossentropy

Table of content

Multi-class classification

Problem statement

Given a dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y} = \{1, 2, \ldots, C\}$, we want to find a classifier $h(x) : \mathcal{X} \to \mathcal{Y}$

Statistical probability perspective

The dataset D is sampled from an unknown distribution $\mathcal{P}(x, y)$.
A 'good' classifier h has a low probability of making error under $\mathcal{P}$.
Given a sample $(X, Y) \sim \mathcal{P}$, the event X is misclassified by h:

$$\{h(X) \neq Y\}$$

and the *error probability* (*error rate*) of h

$$\mathrm{err}_{\mathcal{P}}(h) = \mathbb{P}_{X,Y\sim\mathcal{P}}\{h(X) \neq Y\} \tag{1}$$

Multi-class classification

Estimate the error rate

Since P is unknown, the true error (1) is not directly available. However, we have the dataset D as a realization of P. The empirical error rate on the training data D is then

$$\widehat{\mathrm{err}}_D(h) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}(h(x_i) \neq y_i) \tag{2}$$

Expectation of empirical error rate

The empirical error rate (2) is an unbiased estimator of the true error rate (1) under P since

$$\begin{aligned}\mathbb{E}_P[\widehat{\mathrm{err}}_D(h)] &= \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{x_i, y_i \sim \mathcal{P}}[\mathbb{I}(h(x_i) \neq y_i)] \\ &= \mathbb{P}\{h(X) \neq Y\} = \mathrm{err}_{\mathcal{P}}(h)\end{aligned}$$

Bounding Empirical Error Rate

Concentration bound[2]

Hoeffding inequality with empirical mean reminder: Let $X_1, X_2, \ldots X_n$ be (random) samples from a random variable X, bounded a.s in $[a, b]$, then

$$\mathbb{P}\Big\{\Big|\frac{1}{n}\sum_i^n X_i - E[X]\Big| \leq \epsilon\Big\} \geq 1 - 2\exp\frac{-2n\epsilon^2}{(b-a)^2}$$

Connection to confidence intervals: Bound the range of error ϵ with the level of significance α (change to have a wrong estimation) with the needed number of examples n

[2] https://www.stat.cmu.edu/ larry/=stat700/Lecture6.pdf

Bounding Empirical Error Rate

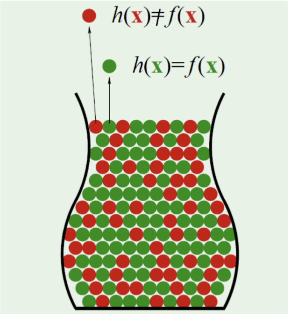
Concentration bound Apply the Hoeffding inequality to the empirical error rate, we have a concentration bound on the difference between empirical and true error rates,

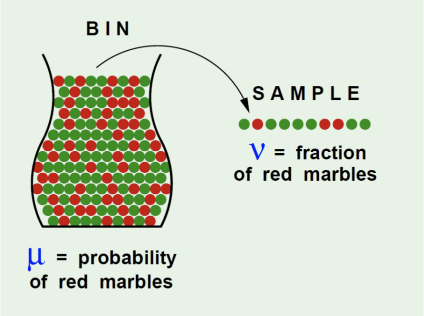
$$\mathbb{P}\left\{|\widehat{\mathrm{err}}_D(h) - \mathrm{err}_{\mathcal{P}}(h)| \leq \epsilon\right\} \geq 1 - 2e^{-2n\epsilon^2}$$

with $X_i = \mathbb{I}(h(x_i) \neq y_i)$ are indicator random variables taking values in $\{0, 1\}$.

- The inequality shows how $\widehat{\mathrm{err}}_D(h)$ closed to $\mathrm{err}_{\mathcal{P}}(h)$, given ϵ and the number of samples

Bounding Empirical Error Rate





- $X_n = [\![h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]\!]$ = one sample training error = either 0 or 1

Table of content

The optimal classifier

Fixed $X = x$, the error event

$$\mathbb{P}\{h(X) \neq Y | X = x\} = 1 - \mathbb{P}(Y = h(x) | X = x)$$

Question: If $X = x$ is fixed, what should be $h(x)$ from the numbers 1 to C to minimize the probability of error?

$$\begin{bmatrix} \mathbb{P}(Y = 1 | X = x) \\ \mathbb{P}(Y = 2 | X = x) \\ \vdots \\ \mathbb{P}(Y = C | X = x) \end{bmatrix}$$

Bayes optimal classifier: is a probabilistic model that makes the most probable classification

$$h^\star(x) = \arg\max_c \mathbb{P}(Y = c | X = x)$$

The optimal classifier

Optimal error probability

$$E^\star = \mathrm{err}_{\mathcal{P}}(h^\star) \leq \mathrm{err}_{\mathcal{P}}(h), \forall h$$

For example: A logistic regression model uses the formula of the Bayes optimal classifier, with $\mathbb{P}(Y = c|X = x)$ is the probabilities of the categorical distribution $\mathrm{Cat}(y|\mathcal{S}(f_1(x), f_2(x), \ldots, f_C(x)))$

Question: Is it possible to find the Bayesian classification function?

- Can we find the distribution P?
- Can we approximate $\mathbb{P}(Y = c|X = x)$ to approximate $h^\star$?

The optimal classifier

Two approaches in machine learning

- **Discriminative models:** learn a predictor given the observations
  - Approximate $\mathbb{P}(Y = c | X = x)$
  - Approximate the classifier $h^\star$
- **Generative models:** learn a joint distribution over all the variables.
  - Approximate $\mathbb{P}(Y = c, X = x)$

$$
\begin{aligned}
h^\star(x) &= \arg\max_c \mathbb{P}(Y = c, X = x) \\
&= \arg\max_c \mathbb{P}(X = x | Y = c)\mathbb{P}(Y = c)
\end{aligned}
$$

  - Can solve a variety of problems, not only classification

Discriminative vs generative models

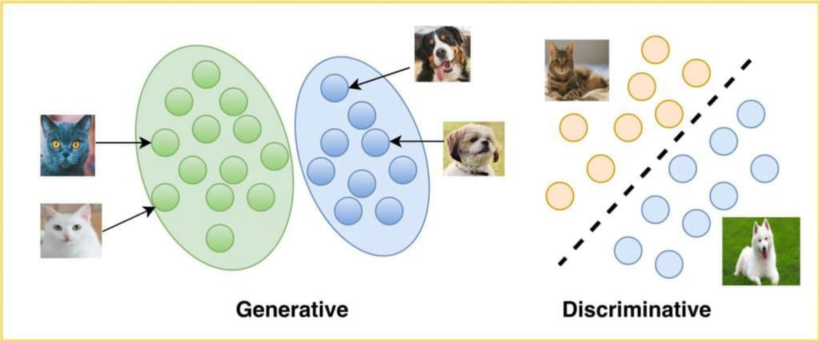


Table of content

K nearest neighbor - KNN

Estimate $\mathbb{P}(Y = c|X = x)$ in the neighborhood of x

- Find k samples from D that are closest to x
- Approximate $\mathbb{P}(Y = c|X = x)$ by the proportion of label c in the k samples.

$$\widehat{\mathbb{P}}(Y = c|X = x) = \frac{\sum_{i=1}^{n} \mathbb{I}(x_i \in V_k(x))\mathbb{I}(y_i = c)}{k}$$

where $V_k(x)$ is a neighborhood of x containing k of data samples in D. The numerator is the number of data samples in $V_k(x)$ that are labeled c.

$h^{KNN}(x)$: select the label c that occurs most in k of the data sample closest to x in D.

K nearest neighbor - KNN

Pseudocode of KNN

1. Calculate $d(x, x_i), i = 1, 2, \ldots n$, where d is a distance metric.
2. Take the first k data points whose distances are smallest among the calculated distance list, along with their labels, denoted $(x_j, y_j)_{j=1}^{k}$.
3. Return the label which has the most votes.

K nearest neighbor - KNN

Theorem of the upper bound of error rate of knn with k = 1

When the number of training samples $n \to \infty$, we have

$$R^\star \leq \mathrm{err}_{\mathcal{P}}(h^{\mathsf{KNN}}) \leq R^\star \left(2 - \frac{C}{C-1} R^\star\right).$$

where $R^\star$ is the Bayesian optimal error probability.

K nearest neighbor - KNN

Proof: Let $x_{(1)}$ as the nearest neighbor of x in D and $y_{(1)}$ as the label of this data sample and y^{true} as the label of x.
Suppose that when number of samples $n \to \infty$,

$$x_{(1)} \to x$$
$$P(y|x_{(1)}) \to P(y|x), \forall y = 1, 2, \ldots, C$$

The error probability of h^{KNN} on x occurs when $y_{(1)} \neq y^{\text{true}}$ is:

$$\begin{aligned}
\mathrm{err}(h^{\mathrm{KNN}}, x) &= P(y^{true} \neq y_{(1)}|x, x_{(1)}) \\
&= 1 - \sum_{y=1}^{C} P(y^{true} = y|x)P(y_{(1)} = y|x_{(1)}) \\
&\to 1 - \sum_{y=1}^{C} P^2(y|x) \qquad \text{(as } n \to \infty)
\end{aligned}$$

K nearest neighbor - KNN

If $y^\star = h^\star(x)$ is the output of the Bayesian optimal classifier function, then

$$P(y^\star|x) = \max_y P(y|x) = 1 - \mathrm{err}(h^\star, x)$$

$$\begin{aligned}
\sum_{y=1}^{C} P^2(y|x) &= P^2(y^\star|x) + \sum_{y \neq y^\star} P^2(y|x) \\
&\geq P^2(y^\star|x) + \frac{(\sum_{y \neq y^\star} P(y|x))^2}{C-1} \\
&\qquad \text{(Bunyakovsky inequality)} \\
&= (1 - \mathrm{err}(h^\star, x))^2 + \frac{\mathrm{err}(h^\star, x)^2}{C-1}
\end{aligned}$$

So

$$1 - \sum_{y=1}^{C} P^2(y|x) \leq 2\mathrm{err}(h^\star, x) - \frac{C}{C-1}\mathrm{err}(h^\star, x)^2 \qquad (3)$$

K nearest neighbor - KNN

Taking the expectation both sides of (3), with $R^\star = \mathbb{E}_x \mathrm{err}(h^\star, x)$, we have

$$\mathbb{E}_x \mathrm{err}(h^{\mathrm{KNN}}, x) = \mathrm{err}(h^{\mathrm{KNN}}) \leq 2R^\star - \frac{C}{C-1} \int_x \mathrm{err}(h^\star, x)^2 p(x) dx \tag{4}$$

Also,

$$\begin{aligned} 0 &\leq \int_x (\mathrm{err}(h^\star, x) - R^\star)^2 p(x) dx \\ &= \int_x \left(\mathrm{err}(h^\star, x)^2 - 2R^\star \mathrm{err}(h^\star, x) + (R^\star)^2\right) p(x) dx \\ &= \int_x \mathrm{err}(h^\star, x)^2 p(x) dx - (R^\star)^2 \end{aligned}$$

Substitute to (4)

$$\mathrm{err}(h^{\mathrm{KNN}}) \leq 2R^\star - \frac{C}{C-1}(R^\star)^2 = R^\star \left[2 - \frac{C}{C-1} R^\star\right].$$