

Classification: Logistic Regression



Xiang Zhou

School of Data Science
Department of Mathematics
City University of Hong Kong

Statistical Decision Theory for Classification

- k -NN method
- Bayes classifier
- LDA and QDA

Review of Bayes Classifier

0-1 loss

We use 0-1 loss to evaluate the performance of classifiers.

The zero-one (0-1) loss function h for the labelled class y and the predicted class \hat{y} is defined ¹,

$$\ell_{01}(y, \hat{y}) = \mathbf{1}(y \neq \hat{y}) \triangleq \begin{cases} 1 & \text{if } y \neq \hat{y} \\ 0 & \text{if } y = \hat{y} \end{cases} \quad (1)$$

where the misclassifications are charged by a single positive unit.

Remark (other equivalent forms)

If the binary outcome is encoded as $\{-1, +1\}$, then

$\ell_{01}(y, \hat{y}) = 1 - \text{heaviside}(y\hat{y}) = (1 - \text{sign}(y\hat{y}))/2$ where

$\text{heaviside}(t) = \begin{cases} 1 & \text{if } t \geq 0 \\ 0 & \text{if } t < 0 \end{cases}$ and $\text{sign}(t) = \begin{cases} 1 & \text{if } t \geq 0 \\ -1 & \text{if } t < 0 \end{cases}$. So 0-1 loss is equivalent to $-\text{heaviside}(y\hat{y})$ and $-\text{sign}(y\hat{y})$.

¹In logistic regression, it is h , the pdf, not the class label \hat{y} , that shows in the loss function
Xiang Zhou

For any classifier $G : \mathcal{X} \rightarrow \{1, \dots, K\}$, its 0-1 loss overall test error rate is

$$\begin{aligned}\mathbb{E}_{X,Y} [\ell_{01}(Y, G(X))] &= \mathbb{E}_X \left(\sum_{k=1}^K \ell_{01}(k, G(X)) \times \mathbb{P}(Y = k|X) \right) \\ &= 1 - \mathbb{E}_X \left[\mathbb{P}(Y = G(X)|X) \right] \quad \because \ell_{01} \text{ is 0-1 loss}\end{aligned}\tag{2}$$

So,

$$\inf_G \mathbb{E}_{X,Y} [\ell_{01}(Y, G(X))]$$

is equivalent to

$$\sup_G \mathbb{E}_X \left[\mathbb{P}(Y = G(X)|X) \right]$$

which has the following optimal solution defined point-wisely for x :

$$G^*(x) = \operatorname{argmax}_{k \in \{1, \dots, K\}} \mathbb{P}(Y = k|X = x)$$

Definition (Bayes classifier)

$$G_{bayes}^*(x) = \operatorname{argmax}_{k \in \{1, \dots, K\}} h_k(x)$$

where

$$h_k(x) := \mathbb{P}(Y = k | X = x)$$

Definition (Gibbs classifier)

Given an input x , the predicted class is a random sample from $\{1, \dots, K\}$ according to the prob mass fun $\{h_k(x), 1 \leq k \leq K\}$

This also works well.

💡 Bayes classifier is optimal for misclassification error

Recall for regression, the conditional probability $f(x) = \mathbb{E}(Y|X = x)$ minimizes the squared error loss $\mathbb{E}[|Y - f(X)|^2]$ and the conditional median $f(x) = \text{median}(Y|X = x)$ minimizes the L_1 error loss $\mathbb{E}|Y - f(X)|$. For classification, we have the analogy for the Bayes classifier.

Theorem

Bayes classifier minimizes the expected 0-1 loss.

The 0-1 loss of Bayes classifier is called **Bayes error rate**:

$$1 - \mathbb{E}_X \left[\max_k \mathbb{P}(Y = k|X) \right] = 1 - \mathbb{E}_X \left[\max_k h_k(X) \right].$$

Bayes Theorem for Bayes Classifier

- The **Bayes theorem** is to view this as the *posterior* distribution of Y with the given observation $X = x$:

$$\begin{aligned}h_k(x) &= \mathbb{P}(Y = k|X = x) \\&= \frac{\mathbb{P}(X = x|Y = k)\mathbb{P}(Y = k)}{\mathbb{P}(X = x)} \\&=: \boxed{\frac{\rho_k(x)\pi_k}{\sum_{l=1}^K \rho_l(x)\pi_l}}\end{aligned}\tag{3}$$

- $\rho_k(x)$: the **class-conditional pdf** of X in class $Y = k$;
- π_k : the (prior) distribution of the class Y ;
- For any given x , the conditional pmf of Y is

$$h_k(x) \propto \rho_k(x)\pi_k.$$

- One might estimate $\rho_k(x)$ (“density estimation”) and π_k (the fraction of training examples belong to class k) directly from the data.

Bayesian classifier

- **Bayesian classifier** assigns each observation to the most likely class, given its predictor value x , i.e., classifies into the maximal posterior prob.

$$G^*(x) = \operatorname{argmax}_{1 \leq k \leq K} \mathbb{P}(Y = k | X = x) = \operatorname{argmax}_{1 \leq k \leq K} [f_k(x)\pi_k] \quad (4)$$

This is called Brute Force MAP (*maximum a posterior*) Learner in Computer Science .

- *Bayes decision boundary* is the decision boundary determined by this Bayes classifier.
- The joint distribution of (X, Y) is $\mathbb{P}(X = x, Y = y) = \sum_{k=1}^K \rho_k(x)\pi_k$. Then this model is the typical mixture models: convex combination of K distributions of f_k :
connection to missing data problem, EM algorithm.

mixed Gaussian: $\rho_k \sim \mathcal{N}(\mu_k, \sigma_k^2)$

Recall the posterior distribution in (3) $\mathbb{P}(Y = k|X = x) = \frac{\rho_k(x)\pi_k}{\sum_{l=1}^K \rho_l(x)\pi_l}$
where $\rho_k(x) \triangleq \mathbb{P}(X = x|Y = k)$ is the distribution of X conditioned on $Y = k$. Bayes classifier maximize this over k .

Exercise

Assume that $X \in \mathbb{R}^1$ and $\rho_k \sim \mathcal{N}(\mu_k, \sigma_k^2)$, $1 \leq k \leq K$.

Then the Bayes classifier corresponds to the maximizer k^ of the following discriminant function*

$$\delta_k(x) = -\frac{x^2}{2\sigma_k^2} + x \cdot \frac{\mu_k}{\sigma_k^2} - \frac{\mu_k^2}{2\sigma_k^2} + \log \pi_k. \quad (5)$$

When $K = 2$, find the point corresponding to the Bayes decision boundary.

Naive Bayesian Classifier:

If \mathcal{X} has a dimensionality $d \gg 1$, then the class-conditional pdf $\rho_k(x)$ is a high dim fun of x . The “naive ” idea in Naive Bayesian classifier is to **ASSUME** that each component is independent !

$$\rho_k(x) = \rho_k(x_1, \dots, x_d) = \prod_{j=1}^d \rho_{kj}(x_j)$$

BENEFIT: decompose a high dim problem (intractable in density estimation) to low dim problems.

JUSTIFICATION: works surprisingly well in practice for **certain problems**.

“Along with decision trees, neural networks, k-nearest neighbours, the Naive Bayes Classifier is one of the most practical learning methods.”

Bayes learning: Bayesian Belief Network

Bayesian Belief Network assumes the k class-conditional pdf dependency in the form of a network representing the conditional information (causal knowledge).

$$\rho_k(x_1, \dots, x_d) = \mathbb{P}(X = (x_1, \dots, x_d) | Y = k)$$

Linear/Quadratic Discriminant Analysis (LDA/QDA)

Recall the mixed Gaussian model (5) above but in d dimension, Assuming $X|Y = k \sim \mathcal{N}_d(\mu_k, \Sigma_k)$, the d -dim Gaussian distribution, then

$$\delta_k(X) = \log(\pi_k) - (1/2) \log |\Sigma_k| - (1/2)(X - \mu_k)^T \Sigma_k^{-1} (X - \mu_k).$$

Classify x to class k with the largest $\delta_k(x)$.

- $\hat{\pi}_k = n_k/n$: the ratio of samples belonging to class k in totally n population;
- μ_k is estimated by the centroid in each class k
- Σ_k is estimated by sample covariance matrix with each class k
- **Assuming** that all Σ_k are equal (estimated by pooled sample variance matrix $\hat{\Sigma}$), we can reduce δ_k to a **linear** function in x ;
- The **QDA** method use the original **quadratic** function δ_k , but QDA need estimate the in-class variance σ_k or the covariance matrix $\hat{\Sigma}_k$ in high dim \mathbb{R}^d , $d > 1$. This requires more data than the LDA for better estimation.
- The decision boundary of QDA is (quadratically) curved.

Exercise

In LDA, consider the eigen-decompositon of the sample covariance matrix $\hat{\Sigma}$ and derive the reduced rank LDA.



k -NN (nearest neighboring) methods

a non-parametric approach to regression and classification

k -NN method ¹: directly estimate the conditional expectation/probability from the data $D = \{(x_i, y_i) : 1 \leq i \leq n\}$.




For regression: $\mathbb{E}(Y|X = x) \approx \text{AVE}(y_i | x_i \in N_k(x)) := \frac{1}{k} \sum_{i: x \in N_k(x)} y_i$.



For classification: $\mathbb{P}(Y = \text{class} | X = x) \approx \frac{1}{k} \sum_{i \in N_k(x)} \mathbf{1}(y_i = \text{class})$

where $N_k(x)$ is the collection of k points in $\{x_i\}$ *closest* to x .

¹  k is the number of points in constructing a neighbourhood. Not to be confused with index previously used for the labels of K classes.

Logistic, LDA, QDA, k-NN ?

model selection – to be addressed later

[p154, [ISL]] “When the true decision boundaries are linear, then the LDA and logistic regression approaches will tend to perform well. When the boundaries are moderately non-linear, QDA may give better results. Finally, for much more complicated decision boundaries, a non-parametric approach such as KNN¹ can be superior. ”

¹with correct choice of k such as by the cross-validation
Xiang Zhou CityU

Logistic Regression

- Today, the logistic regression model is one of the most widely used binary models in the analysis of categorical data where $\mathcal{Y} = \{1, \dots, K\}$.

- Logistic regression, an extension of the linear regression for classification, is based on modeling the *odds* of an outcome:

$$\mathbb{P}(Y = k|X = x),$$

in contrast to the outcome $Y = k$ itself.

- Before that, Fisher proposed *linear discriminant analysis* (**LDA**) in 1936. There are other methods based on the use of some discriminant function, which may not be $\mathbb{P}(Y = k|X = x)$.

We focus on the following four questions which we shall always follow for any machine learning problems

- ① How to represent the “odds” ?
- ② How to model the the cost/loss functions ?
- ③ How to minimize the cost function?
- ④ How to evaluate the performance of the trained model ?

Logistic regression model for binary classification

Now $\mathcal{Y} = \{0, 1\}$. Recall that in Bayes classifier, we introduced a function $h(x)$ as the **conditional probability** of $y = 1$ for a given input x :

$$h(x) = \mathbb{P}(Y = 1|X = x), \quad 1 - h(x) = \mathbb{P}(Y = 0|X = x)$$

- Assume that the logarithm of this probability, as a function of x , is a linear function: $\log h(x; \theta) = f(x, \theta) = \theta \cdot x$, we would have $h = e^{\theta \cdot x}$ which is always positive but has no upper bounds.
- The modification is to use the “0” class probability (i.e. $1 - h$) as a *reference value*. Then the logistic regression model is to assume that

$$\log h(x; \theta) - \log(1 - h(x; \theta)) = f(x; \theta)$$

or

$$h = \frac{1}{1 + e^{-f}}, \quad 1 - h = \frac{e^{-f}}{1 + e^{-f}}$$

Now, f can be any \mathbb{R} -valued continuous function on \mathcal{X} . You can propose any hypothesis space ($\subset \mathcal{X}$) you want to search the best f in this space.

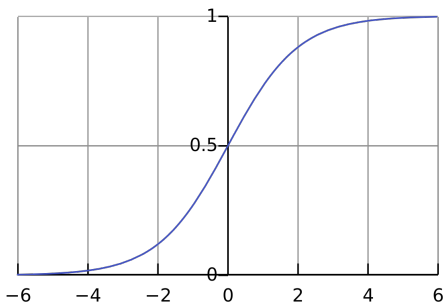
Definition

The **logit** function: $(0, 1) \rightarrow \mathbb{R}^1$ is

$$h \rightarrow z = \log \frac{h}{1-h} =: \text{logit}(h)$$

The inverse of logit function is the **sigmoid(logistic)** function: $\mathbb{R}^1 \rightarrow (0, 1)$

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



activation function family

Why σ this form ?

- What kind of *activation* function mapping \mathbb{R}^1 onto $(0, 1)$?
 - ▶ Heaviside function $\sigma(x) = I_{\{x>0\}}$;
 - ▶ capped linear function $\sigma(x) = \max\{\min(kx + c, 1), 0\}$ with $k > 0$, $c \in \mathbb{R}$;
 - ▶ $\sigma(x) = \frac{1}{\pi} \arctan(x) + \frac{1}{2}$
 - ▶ $\sigma(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-x^2} dx$
- The considerations of choosing the feature variable and the activation function:
 - ▶ simple
 - ▶ computational concerns in minimizing the loss
 - ▶ Inverse function
 - ▶ some certain stat/prob. interpretation (*log-odds* by G. A. Barnard, 1949)

Sigmoid logistic function

- The logistic function was invented for the purpose of describing the population growth ([history](#)). Logistic map: $x_{n+1} = rx_n(1 - x_n)$. Logistic function was given its name by a Belgian mathematician, P.F. Verhulst (1838). So, the logistic function is used in areas far beyond the classification.
- The logistic function is an offset and scaled **hyperbolic tangent function**: $\sigma(x) = \frac{1}{2} + \frac{1}{2} \tanh(z)$ because $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$.
- $\sigma(x)$ is smooth and symmetric in the sense

$$\sigma(x) + \sigma(-x) = 1$$

Exercise

Show that the sigmoid function satisfies the logistic equation

$$\sigma'(z) = \sigma(z)(1 - \sigma(z))$$

and show that if $h(x; \theta) = \sigma(z(\theta, x))$ for a general bi-variate function $z(\cdot, \cdot)$ of θ and x , then the gradient $\nabla_{\theta} h(x; \theta) = \sigma'(z) \nabla_{\theta} z$, and the Hessian matrix $\nabla_{\theta}^2 h(x; \theta) = \sigma''(z) \nabla_{\theta} z \nabla_{\theta} z^T + \sigma'(z) \nabla_{\theta}^2 z$

Exercise (softplus function)

Show that the derivative of the so called **softplus** function

$$\text{softplus}(x) = \ln(1 + e^x)$$

is the sigmoid function $\sigma(x) = 1/(1 + e^{-x})$. Equivalent $\int_{-\infty}^x \sigma(x') dx' = \text{softplus}(x)$. In addition, show that

$$-\log(\sigma(z)) = \text{softplus}(-z) \quad \text{and} \quad -\log(1 - \sigma(z)) = \text{softplus}(z)$$

So, softplus function is connected to the negative log likelihood function.

Softplus function is a smooth function close to the RELU:

$$\text{RELU}(x) = \max(0, x)$$

We now have a general framework for classification problem by working on the log-odd, z :

$$x \in \mathcal{X} \xrightarrow{f(x;\theta)} z \in \mathbb{R}^1 \xrightarrow{\sigma(\cdot)} h \in (0, 1) \xrightarrow[h > 0.5]{h < 0.5} y \in \mathcal{Y} = \{0, 1\}$$

The decision boundary $\{x : h(x) = 0.5\}$ becomes the set where $z = f(x; \theta) = 0$ since $\sigma(0) = 0.5$

- $z = f(x) > 0 \iff h(x) > 0.5$: classifies x as “1”.
- $z = f(x) < 0 \iff h(x) < 0.5$: classifies x as “0”.
- $z = f(x) = 0$ gives the decision boundary.

In summary, the classifier given by f is to assigne x to the class $= \text{heaviside}(f(x)) = \text{heaviside}(h(x) - 0.5)$

What remains is to select a model class (hypothesis space \mathcal{H}) for representing

$$x \in \mathcal{X} \xrightarrow{f(x;\theta)} z \in \mathbb{R}^1$$

Logistic regression model for $K > 2$ classes

Softmax regression (multinomial logistic regression)

Generalize to the K -class where the labels $y \in \{1, \dots, K\}$ by assuming that the conditional prob. takes the form

$$\mathbb{P}(Y = k|X = x) = h_k(x; \theta) \quad (6)$$

But we have the constraint

$$\sum_k h_k(x; \theta) = \sum_k \mathbb{P}(Y = k|X = x) = 1.$$

WLOG, we use the last one $h_K(x; \theta)$. Define z_k , the **logit**,

$$z_k := \log h_k(x; \theta) - \log h_K(x; \theta), \quad k = 1, 2, \dots, K-1.$$

Then $h_k = h_K e^{z_k}$ and $z_K = 0$. The constraint $\sum_k h_k = 1$ leads to

$$h_K = \left(\sum_{k=1}^K e^{z_k} \right)^{-1} \text{ and}$$

$$\boxed{h_k(x; \theta) = \frac{e^{z_k}}{\sum_{k=1}^K e^{z_k}}, \quad k = 1, 2, \dots, K} \quad (7)$$

Softmax function

Definition

The **softmax** function is the following nonlinear mapping $\mathbb{R}^K \rightarrow (0, 1)^K$:

$$z = (z_1, \dots, z_K) \mapsto h = (h_1, \dots, h_K) = \text{softmax}(z)$$

where

$$h_k = \frac{e^{z_k}}{\mathcal{Z}}, \quad \text{where } \mathcal{Z} := \sum_{i=1}^K e^{z_i}$$

i.e., $\log h_k = z_k - c$ where c is a constant such that $\sum_{k=1}^K h_k = 1$.

Remark

The name “softmax” comes from the fact

$\lim_{\delta \rightarrow 0} \text{softmax}(z/\delta) = (0, \dots, 0, 1, 0, \dots, 0)$ where the position of 1 entry corresponds to $\arg\max_k \{z_k\}$.

It takes a vector of arbitrary real-valued scores and squashes the vector to a new vector with values between 0 and 1 and with zero sum.

Exercise

- If $z_1 < z_2$, then $h_1 < h_2$. So h keeps the order of z (and magnifies the difference among the values $\{z_k\}$)
- $\text{softmax}(z + c) = \text{softmax}(z)$ for any scalar c . If choose $c = -\max\{z_1, \dots, z_K\}$, then every elements in the vector $z + c$ is not positive. The calculation of $\text{softmax}(z + c)$ is more stable than that directly on $\text{softmax}(z)$. ($\exp(1000)$ gives you NaN on computers.)
- When $z_k = \theta_k \cdot x$, show that the shift $\theta \rightarrow \theta - c$ does not change the value of $h(x; \theta)$. So the softmax regression's K parameters are redundant. In learning θ , we can simply set $\theta_K = 0$ or adding the linear constraint $\sum \theta_k = 0$.

Linear model assumption for z_k

- With the aid of softmax, the representation of the function $\{h_k(x)\}$ becomes the representation of \mathbb{R}^K -value functions $z_k(x) = f_k(x; \theta)$.
- The softmax (logistic) regression assume the linear form $z_k = f_k(x; \theta_k) = \theta_k \cdot x$, with K parameters $\theta_k \in \mathbb{R}^d$ ¹. For convenience, we still use $\theta = \{\theta_1, \dots, \theta_K\}$ to denote all the parameters of our model .
- Like in nonlinear regression, all same techniques can be applied here to represent the function $x \rightarrow z_k$. (sparse, kernel, spline ,.....)
- Recently, the DNN (deep neural network) models the function $x \rightarrow z_k$ by neural network. The result is a huge success.

¹Effectively, $K - 1$ parameters, since $z_K = 0$.

Decision rule of softmax regression

$$x \in \mathcal{X} \xrightarrow[k=1, \dots, K]{f_k(x; \theta)} z_k \in \mathbb{R}^1 \xrightarrow{\text{softmax}} h_k \in (0, 1) \xrightarrow[y=k^*]{\max_k h_k(x)} y \in \mathcal{Y} = \{0, 1\}$$

$h(x; \theta) = \text{softmax}(z)$ i.e., $h_k(x; \theta) = \frac{e^{z_k}}{\sum_{k=1}^K e^{z_k}}, k = 1, 2, \dots, K$ and $z_k = f_k(x; \theta) = \theta_k \cdot x$. Then for an input x , we assign it to the class which is

$$\begin{aligned} k^*(x) &= \operatorname{argmax}_{1 \leq k \leq K} h_k(x; \theta) = \operatorname{argmax}_k e^{z_k} \\ &= \operatorname{argmax}_k z_k = \operatorname{argmax}_k f_k(x) \\ &= \operatorname{argmax}_k (\theta_k \cdot x) \end{aligned}$$

The last step shows that it is a linear classifier since z_k is linear in x .

Exercise: For example, $K = 3$, and $d = 2$, $\theta_1 = (1, 0), \theta_2 = (1, 1), \theta_3 = (0, 0)$. draw the three domains where x are classified as 1, 2, 3, respectively.

How to choose loss function

- A criterion must be set to define the loss function ℓ in order to find the optimal parameter θ in $h_k(x; \theta)$.
- We first recall that the 0-1 loss is defined for the classification outcome: $\mathcal{Y} \times \mathcal{Y} \rightarrow \{0, 1\}$.

- The predicted class $\hat{y}(x) = \operatorname{argmax}_k h_k(x; \theta)$, then $\ell_{01}(y, \hat{y}) = I(y = \operatorname{argmax}_k h_k(x; \theta))$
- The empirical risk from the data $D = \{(x_i, y_i)\}$ is

$$\sum_{i=1}^N I(y_i = \operatorname{argmax}_k h_k(x_i; \theta)) = \sum_{k=1}^K \sum_{(x_i, y_i=k)} I(k = \operatorname{argmax}_{k'} h_{k'}(x_i; \theta))$$

- For the binary case where $\mathcal{Y} = \{0, 1\}$, with $h(x; \theta) = h_1(x; \theta)$,

$$\begin{aligned} & \sum_{i=1}^N I(y_i = \operatorname{argmax}_k h_k(x_i; \theta)) \\ &= \sum_{(x_i, y_i=1)} I(h(x_i; \theta) > 0.5) + \sum_{(x_i, y_i=0)} (1 - I(h(x_i; \theta) > 0.5)). \end{aligned}$$

- The above 0-1 loss only used the sign of $h(x) - 0.5$; the prob. meaning of $h(x)$ is not used. In addition, the minimization for the above empirical 0-1 loss is hard.
- The key difference of logistic regression from most machine learning methods based on linear separating hyperplane (SVM) is that logistic regression attempt to model and estimate $\mathbb{P}(Y = k|X = x)$ for each k directly .

Two Main Principles to Build Loss functions



Statistical Learning Approach:

- ▶ Maximize Likelihood
- ▶ Bayesian = Prior \times Likelihood



Information Theory Approach: Minimize the “distance” between prob. measures.

Loss function of logistic regression = negative log-likelihood

[coursera video](#).

- In linear regression, the squared error $l(y, \hat{y}) = (y - \hat{y})^2$, has the interpretation of negative log likelihood for Gaussian-type residuals. The same idea of taking negative log likelihood as the loss for classification is as follows.
- For a given data point (x, y) , in binary case, the probability $\mathbb{P}(Y = y|X = x) = h(x; \theta)$ if $y = 1$ and $\mathbb{P}(Y = y|X = x) = 1 - h(x; \theta)$ if $y = 0$, which can be unified in one expression for the likelihood function:

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

Then the [negative log-likelihood](#) is

$$-y \log h - (1 - y) \log(1 - h)$$

which is will be defined as ℓ . Thus, MLE is equivalent to minimizing ℓ .

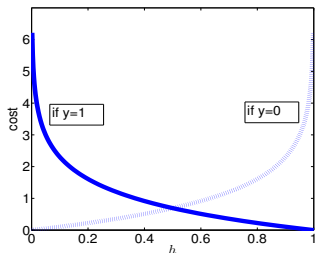
Loss function for Binary classification

Definition

The loss function for the binary logistic regression is the function $(0, 1) \times \{0, 1\} \rightarrow \mathbb{R}^+$ in the form

$$\ell(h, y) = -y \log h - (1 - y) \log(1 - h) = \begin{cases} -\log h & \text{if } y = 1 \\ -\log(1 - h) & \text{if } y = 0 \end{cases} \quad (8)$$

Note that this form of this loss function is unlike the regression loss where the two input argument are both \mathcal{Y} -valued.



Homework ℓ is convex in h .

discussion: Why this cost function makes sense from the viewpoint of minimizers at different values of y ?
Derivative $\partial \ell / \partial h$

Then the objective function ¹ on the training dataset $D = \{(X^{(i)}, Y^{(i)})\}$ is the sum of loss from all individual examples

$$\begin{aligned} J(\theta) &= \frac{1}{n} \sum_{i=1}^n \ell(h(X^{(i)}; \theta), Y^{(i)}) \\ &= \frac{1}{n} \left(\sum_{i:Y^{(i)}=1} -\log h(X^{(i)}; \theta) + \sum_{i:Y^{(i)}=0} -\log(1 - h(X^{(i)}; \theta)) \right) \\ &= \frac{1}{n} \left(\sum_{i:Y^{(i)}=1} \log \left(1 + e^{-f(X^{(i)}; \theta)} \right) \right. \\ &\quad \left. + \sum_{i:Y^{(i)}=0} \log \left(1 + e^{+f(X^{(i)}; \theta)} \right) \right). \end{aligned} \tag{9}$$

In logistic regression, $f(x; \theta) = \theta \cdot x$.

Exercise

Show that J is convex in θ for logistic regression.

¹sometimes, we drop the $\frac{1}{n}$ factor in J since it does not affect the minimizers.

The following exercise shows that the binomial deviance loss is just the loss in logistic regression, written in terms of f rather than of h .

Exercise

Recall the relation that the odd $h = \sigma(z)$, σ is the sigmoid function, and $z = f(x; \theta)$. Then the logistic loss ℓ in (8) can be written in terms of f ,^a

$$\ell(f, y) = \begin{cases} -\log h(x; \theta) = \log(1 + e^{-f}) = \text{softplus}(-f) & \text{if } y = 1 \\ -\log(1 - h(x; \theta)) = \log(1 + e^f) = \text{softplus}(f) & \text{if } y = 0 \end{cases} \quad (10)$$

Change the binary coding of \mathcal{Y} to $\{\pm 1\}$ (i.e, “0” class is named as “-1” class now), then $\ell(f, y)$ has a convenient expression:

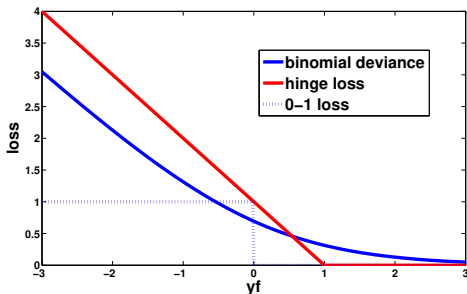
$$\ell_{bd}(f, y) := \text{softplus}(-yf(x)), \quad y \in \mathcal{Y} = \{-1, 1\}, f : \mathcal{X} \rightarrow \mathbb{R}. \quad (11)$$

which is the product of y and f . This loss $\ell_{bd}(f, y)$ has the name **binomial deviance loss**, which arises from deviance statistics for binormal distribution.^b

^aWe here abused the use of ℓ . The definition domain of ℓ function here is $\mathbb{R} \times \{0, 1\}$, not $(0, 1) \times \{0, 1\}$.

In this page, we use $\{\pm 1\}$ -encoded \mathcal{Y} and compare the 0-1 loss and the binomial deviance loss.

- Note that the classifier with a given f is equal to $\text{sign}(f(x))$. Then the 0-1 loss can be rewritten as $\ell_{01}(y, f(x)) = \text{heaviside}(-yf(x))$.
- Logistic regression: $\ell_{bd}(y, f(x)) = \text{softplus}(-yf(x))$.
- The third loss: hinge loss $\ell_{\text{hinge}}(y, f) = (1 - yf)_+$ is used in support vector classifier (to be discussed later)



Exercise

Recall in Topic 1, we have \mathcal{E} defined as the expected loss (the generalized error):

$$\mathcal{E}_{bd}(f) = \mathbb{E} \ell_{bd}(Y, f(X)) = \mathbb{E}_{X,Y} [\text{softplus}(-Y f(X))]$$

Denote $\pi_{\pm} = \mathbb{P}(Y = \pm 1)$ the distribution of Y and $\rho_{\pm} = p_{X|Y=\pm}(x)$, the conditional distribution of X given Y . Solve the variational problem

$$\inf_f \mathcal{E}_{bd}(f)$$

Show that the optimal f_* is

$$\sigma(f_*(x)) = \frac{\pi_+ \rho_+(x)}{\pi_+ \rho_+(x) + \pi_- \rho_-(x)} = \frac{\mathbb{P}(X = x, Y = +)}{p_X(x)} = \mathbb{P}(Y = + | X = x)$$

This expression of f^* is consistent to the fact that $h(x) = \sigma(z) = \sigma(f(x))$.

Loss function of K -classification

- The loss function as the negative log likelihood for K -classification on an input-output (x, y) is

$$\ell(\mathbf{h}, y) = \begin{cases} -\log h_1(x; \theta) & \text{if } y = 1 \\ -\log h_2(x; \theta) & \text{if } y = 2 \\ \dots\dots\dots \\ -\log h_K(x; \theta) & \text{if } y = K \end{cases} = -\log h_y(x; \theta) \quad (12)$$

where $\mathbf{h} = (h_1, \dots, h_K) = \text{softmax}(z)$ with $z_k = f_k(x) = x \cdot \theta_k$.

- Then we have the objective function

$$J(\theta) := \frac{1}{n} \sum_{i=1}^n -\log h_{Y^{(i)}}(X^{(i)}; \theta) = \frac{1}{n} \sum_{k=1}^K \left(\sum_{\substack{i \in \{1, \dots, n\} \\ Y^{(i)} = k}} -\log h_k(X^{(i)}; \theta) \right) \quad (13)$$

- cross-entropy

Definition

- The (Shannon) **entropy** of a prob. distribution p is

$$H(p) = H(p, p) = -\mathbb{E}_{Y \sim p}[\log p(Y)] = -\sum_y p(y) \log p(y).$$

- The **cross-entropy** between a distribution p and another distribution q is defined as:

$$H(p, q) \triangleq -\sum_y p(y) \log q(y) = -\mathbb{E}_{Y \sim p}[\log q(Y)]$$

- The **Kullback-Leibler divergence** is defined as

$$D_{\text{KL}}(p \| q) \triangleq -\sum_x p(x) \log \frac{q(x)}{p(x)} = H(p, q) - H(p)$$

- $D_{\text{KL}}(p\|q)$ is non-negative and is the measurement of how far from q to p . Note that $D_{\text{KL}}(p\|q) \neq D_{\text{KL}}(q\|p)$ in general. But $D_{\text{KL}}(p\|q) = 0$ iff $p = q$.
- For fixed p , minimizing $D_{\text{KL}}(p\|q)$ over q is equivalent to minimizing $H(p, q)$.

How to choose p and q for classification problem ?

- In the above logistic regression for the K classification, *given* x , q is a Bernoulli distribution $q(k) = \mathbb{P}(Y = k|X = x) = h_k(x; \theta), 1 \leq k \leq K$.
- p is from one given sample $(x, y) \in \mathcal{X} \times \{1, \dots, K\}$, it is the delta distribution (**one hot distribution**): $p(k) = 1$ if $k = y$ and $p(k) = 0$ if $k \neq y$, i.e., $p(k) = \delta_{k,y}$
- So, $H(p, q) = -\sum_k p(k) \log q(k) = -\log h_y(x; \theta)$, which is identical to the loss (12)

This is why (12) called the **cross-entropy loss** or *log loss*.

Numerical Optimization of Logistic Regression Models

gradient calculation

Recall that $\ell(h(x; \theta), y) = -y \log h(x; \theta) - (1 - y) \log(1 - h(x; \theta))$ and $h(x; \theta) = \sigma(z)$ where $z = f(x; \theta) = \theta \cdot x$.

$$\begin{aligned}\nabla_{\theta} \ell &= -y/\sigma(z) \cdot \sigma'(z) \nabla_{\theta} z + (1 - y)/(1 - \sigma(z)) \cdot \sigma'(z) \nabla_{\theta} z \\ &= -y(1 - \sigma(z)) \nabla_{\theta} z + (1 - y)\sigma(z) \nabla_{\theta} z \\ &= (\sigma(z) - y) \nabla_{\theta} z = (h(x; \theta) - y)x\end{aligned}$$

Exercise

Show that the Hessian matrix of ℓ is

$$\nabla_{\theta}^2 \ell = (h(x; \theta) - y) \nabla_{\theta}^2 z + \sigma'(z) (\nabla_{\theta} z) (\nabla_{\theta} z)^T = h(1 - h) x x^T.$$

Show that this matrix has the rank 1 and is positive semi-definite.

$$J(\theta) = \sum_{i=1}^n \ell(h(x^{(i)}; \theta), y^{(i)})$$

$$\nabla_{\theta} J = \sum_{i=1}^n (h(x^{(i)}; \theta) - y^{(i)}) x^{(i)} = \mathbf{X}(\sigma(\mathbf{z}) - \mathbf{y})$$

$$\text{where } \mathbf{z} = \mathbf{X}\theta$$

Here, the matrix $\mathbf{X} \triangleq [x^{(1)}, x^{(2)}, \dots, x^{(n)}] \in \mathbb{R}^{d \times n}$ whose n columns corresponding to n data points. θ and $\mathbf{y} = (y^{(1)}, y^{(2)}, \dots, y^{(n)})^T$ are column vectors. σ function acts on the vector in the element-wise sense. Then the gradient descent is

$$\theta^{new} = \theta^{old} + \text{learning rate} \times \frac{1}{n} \sum_{i=1}^n (y^{(i)} - h(x^{(i)}; \theta^{old})) x^{(i)}$$

Now consider the displacement $\Delta\theta := \theta^{new} - \theta^{old}$ on the projection of $x^{(i)}$, then $\Delta z^{(i)} = \Delta\theta \cdot x^{(i)} = \eta(y^{(i)} - h(x^{(i)}; \theta^{old})) \|x^{(i)}\|^2$. So, $y^{(i)} = 1$ means $\Delta z^{(i)} > 0$ and $y^{(i)} = 0$ means $\Delta z^{(i)} < 0$. Recall the decision boundary in \mathcal{Z} space.