A Unified View of Binary Classification Algorithms

Sentao Chen*

1 Classification with Independent and Identically Distributed (IID) Data

1.1 Binary Classification

In a binary classification problem, let $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ be the input features and $y \in \mathcal{Y} = \{-1, +1\}$ be the class label. Given a training dataset $\{(\boldsymbol{x}_i^{tr}, y_i^{tr})\}_{i=1}^{m_{tr}}$, where $(\boldsymbol{x}_i^{tr}, y_i^{tr}) \sim P^{tr}(\boldsymbol{x}, y)$, the goal is to learn a classification function $h^*(\boldsymbol{x})$ from the training dataset, such that $h^*(\boldsymbol{x})$ well predicts the class labels for the testing dataset $\{(\boldsymbol{x}_i^{te}, y_i^{te})\}_{i=1}^{m_{te}}$, where $(\boldsymbol{x}_i^{te}, y_i^{te}) \sim P^{te}(\boldsymbol{x}, y) = P^{tr}(\boldsymbol{x}, y)$[1] and the true labels y_i^{te} are unknown before prediction. Namely, we should solve the following optimization problem

$$
\begin{aligned}
h^*(\boldsymbol{x}) &= \operatorname*{argmin}_h \frac{1}{m_{te}} \sum_{i=1}^{m_{te}} L_{0/1}(h(\boldsymbol{x}_i^{te}), y_i^{te}) && (1)\\
&\approx \operatorname*{argmin}_h \int_{\mathcal{X}\times\mathcal{Y}} L_{0/1}(h(\boldsymbol{x}), y) P^{te}(\boldsymbol{x}, y) d\boldsymbol{x} dy, \quad \text{(law of large numbers)} && (2)\\
&= \operatorname*{argmin}_h \int_{\mathcal{X}\times\mathcal{Y}} L_{0/1}(h(\boldsymbol{x}), y) P^{tr}(\boldsymbol{x}, y) d\boldsymbol{x} dy, \quad (P^{te}(\boldsymbol{x}, y) = P^{tr}(\boldsymbol{x}, y)) && (3)\\
&\approx \operatorname*{argmin}_h \frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} L_{0/1}(h(\boldsymbol{x}_i^{tr}), y_i^{tr}). \quad \text{(law of large numbers)} && (4)
\end{aligned}
$$

Here, $L_{0/1}(h(\boldsymbol{x}), y)$ is the 0-1 loss function which evaluates 1 if $yh(\boldsymbol{x}) < 0$ and 0 otherwise. The resultant optimization problem is

$$
\min_h \frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} L_{0/1}(h(\boldsymbol{x}_i^{tr}), y_i^{tr}). \tag{5}
$$

1.2 Hypothesis Space

The classification function can be a linear function from a linear function space

$$
\mathcal{H}_{\text{lin}} = \{h(\boldsymbol{x}; \boldsymbol{\theta}) | h(\boldsymbol{x}; \boldsymbol{\theta}) = \boldsymbol{\theta}^\top \boldsymbol{x} = \theta_1 x_1 + ... + \theta_d x_d\}
$$

or a nonlinear function from a kernel function space

$$
\mathcal{H}_{\text{ker}} = \{h(\boldsymbol{x}; \boldsymbol{\theta}) | h(\boldsymbol{x}; \boldsymbol{\theta}) = \boldsymbol{\theta}^\top \boldsymbol{k}(\boldsymbol{x}) = \theta_1 k(\boldsymbol{x}, \boldsymbol{x}_1^{tr}) + ... + \theta_{m_{tr}} k(\boldsymbol{x}, \boldsymbol{x}_{m_{tr}}^{tr})\}.
$$

Here, $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_d)^\top \in \mathbb{R}^d$ or $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_{m_{tr}})^\top \in \mathbb{R}^{m_{tr}}$, and $\boldsymbol{k}(\boldsymbol{x}) = (k(\boldsymbol{x}, \boldsymbol{x}_1^{tr}), \cdots, k(\boldsymbol{x}, \boldsymbol{x}_{m_{tr}}^{tr}))^\top \in \mathbb{R}^{m_{tr}}$. $k(\boldsymbol{x}, \boldsymbol{y})$ is a Gaussian kernel function $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(\frac{-\|\boldsymbol{x}-\boldsymbol{y}\|^2}{\sigma})$ with kernel width $\sigma > 0$ or a polynomial kernel function $k(\boldsymbol{x}, \boldsymbol{y}) = (1 + \boldsymbol{x}^\top \boldsymbol{y})^c$ with degree $c > 0$.

*Sentao Chen is with Department of Computer Science, Shantou University

[1] Since the training data and testing data are both independently sampled from the training joint distribution $P^{tr}(\boldsymbol{x}, y)$ and the testing joint distribution $P^{te}(\boldsymbol{x}, y)$ and since $P^{tr}(\boldsymbol{x}, y)$ and $P^{te}(\boldsymbol{x}, y)$ are identical, we say that the training data and testing data are Independent and Identically Distributed (IID).

1.3 Loss Function and Optimization Problem

Because the 0-1 loss function is nonconvex and discrete, the optimization problem in (5) is difficult to solve. Therefore, we can use the surrogate loss functions (*e.g.*, square loss, hinge loss, exponential loss, logistic loss) to replace the 0-1 loss. These surrogate loss functions are convex, continuous, and upper bounds the 0-1 loss. See Figure 1. Then, we can instead solve one of the following optimization problems

- square loss.

$$L_{0/1}(h(\boldsymbol{x};\boldsymbol{\theta}),y) \leq L_{\text{squ}}(h(\boldsymbol{x};\boldsymbol{\theta}),y) = (h(\boldsymbol{x};\boldsymbol{\theta})-y)^2 = (1-yh(\boldsymbol{x};\boldsymbol{\theta}))^2 \tag{6}$$

$$\Rightarrow \frac{1}{m_{tr}}\sum_{i=1}^{m_{tr}} L_{0/1}(h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta}),y_i^{tr}) \leq \frac{1}{m_{tr}}\sum_{i=1}^{m_{tr}} L_{\text{squ}}(h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta}),y_i^{tr}) \tag{7}$$

$$\Rightarrow \min_{\boldsymbol{\theta}} \frac{1}{m_{tr}}\sum_{i=1}^{m_{tr}} \left(1-y_i^{tr}h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta})\right)^2 + \lambda\|\boldsymbol{\theta}\|^2 \tag{8}$$

  This is the optimization problem of least squares classification algorithm.

- hinge loss.

$$L_{0/1}(h(\boldsymbol{x};\boldsymbol{\theta}),y) \leq L_{\text{hin}}(h(\boldsymbol{x};\boldsymbol{\theta}),y) = \max\{0, 1-yh(\boldsymbol{x};\boldsymbol{\theta})\} \tag{9}$$

$$\Rightarrow \frac{1}{m_{tr}}\sum_{i=1}^{m_{tr}} L_{0/1}(h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta}),y_i^{tr}) \leq \frac{1}{m_{tr}}\sum_{i=1}^{m_{tr}} L_{\text{hin}}(h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta}),y_i^{tr}) \tag{10}$$

$$\Rightarrow \min_{\boldsymbol{\theta}} \frac{1}{m_{tr}}\sum_{i=1}^{m_{tr}} \max\left\{0, 1-y_i^{tr}h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta})\right\} + \lambda\|\boldsymbol{\theta}\|^2 \tag{11}$$

  This is the optimization problem of Support Vector Machine (SVM) algorithm.

- logistic loss.

$$L_{0/1}(h(\boldsymbol{x};\boldsymbol{\theta}),y) \leq \frac{1}{\log 2} L_{\log}(h(\boldsymbol{x};\boldsymbol{\theta}),y) = \frac{1}{\log 2}\log\left(1+\exp(-yh(\boldsymbol{x};\boldsymbol{\theta}))\right) \tag{12}$$

$$\Rightarrow \frac{1}{m_{tr}}\sum_{i=1}^{m_{tr}} L_{0/1}(h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta}),y_i^{tr}) \leq \frac{1}{m_{tr}}\sum_{i=1}^{m_{tr}} \frac{1}{\log 2} L_{\log}(h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta}),y_i^{tr}) \tag{13}$$

$$\Rightarrow \min_{\boldsymbol{\theta}} \frac{1}{m_{tr}}\sum_{i=1}^{m_{tr}} \log\left(1+\exp(-y_i^{tr}h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta}))\right) + \lambda\|\boldsymbol{\theta}\|^2 \tag{14}$$

  This is the optimization problem of logistic regression algorithm. It is equivalent to the softmax regression algorithm when the number of classes is 2.

In the above optimization problems, a regularization term $\lambda\|\boldsymbol{\theta}\|^2$ with regularization parameter $\lambda > 0$ is added to avoid overfitting to the training dataset.

1.4 Optimization Algorithm and Prediction

We exploit the first-order (*e.g.*, gradient descent) or second-order (*e.g.*, Newtons's method, L-BFGS) optimization algorithms to learn the optimal parameters $\widehat{\boldsymbol{\theta}}$ of the function h. Then, prediction is performed via the equation $y = \text{sign}(h(\boldsymbol{x};\widehat{\boldsymbol{\theta}}))$.

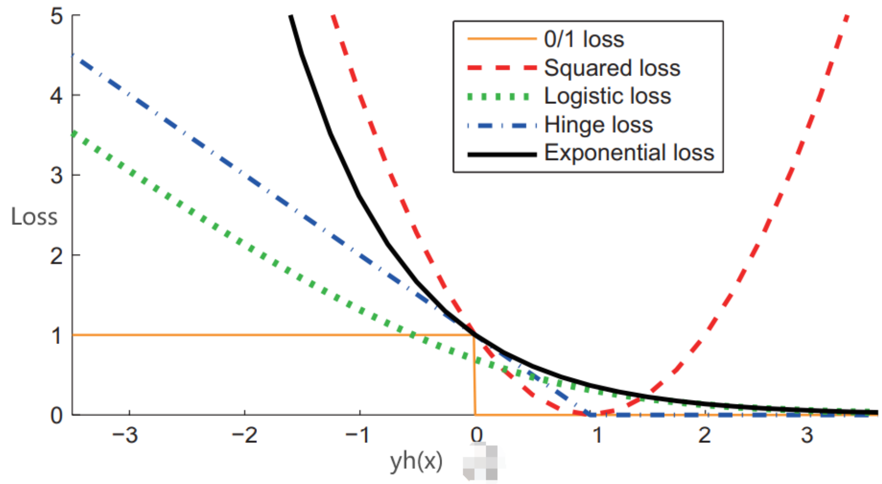

Figure 1: Illustration of various loss functions.

2 Classification with Non-IID Data

2.1 Weighted Binary Classification

Given a training dataset $\{(\boldsymbol{x}_i^{tr}, y_i^{tr})\}_{i=1}^{m_{tr}}$ and an unlabeled testing dataset $\{\boldsymbol{x}_i^{te}\}_{i=1}^{m_{te}}$, where $(\boldsymbol{x}_i^{tr}, y_i^{tr}) \sim P^{tr}(\boldsymbol{x}, y)$ and $\boldsymbol{x}_i^{te} \sim P^{te}(\boldsymbol{x}) = \int_{\mathcal{Y}} P^{te}(\boldsymbol{x}, y)dy$, the goal is to learn a classification function $h^*(\boldsymbol{x})$ from the training dataset and unlabeled testing dataset, such that $h^*(\boldsymbol{x})$ well predicts the class labels for the unlabeled testing data. Here, the training joint distribution $P^{tr}(\boldsymbol{x}, y)$ and the testing joint distribution $P^{te}(\boldsymbol{x}, y)$ are different, *i.e.*, $P^{tr}(\boldsymbol{x}, y) \neq P^{te}(\boldsymbol{x}, y)$. If we make the assumption that the joint distribution difference is originated from the marginal distribution difference, *i.e.*, $P^{tr}(\boldsymbol{x}) \neq P^{te}(\boldsymbol{x})$, then we can solve the following optimization problem to learn the classification function

$$\begin{aligned}
h^*(\boldsymbol{x}) &= \underset{h}{\operatorname{argmin}} \frac{1}{m_{te}} \sum_{i=1}^{m_{te}} L_{0/1}(h(\boldsymbol{x}_i^{te}), y_i^{te}) && (15)\\
&\approx \underset{h}{\operatorname{argmin}} \int_{\mathcal{X}\times\mathcal{Y}} L_{0/1}(h(\boldsymbol{x}), y) P^{te}(\boldsymbol{x}, y) d\boldsymbol{x}dy, \quad \text{(law of large numbers)} && (16)\\
&= \underset{h}{\operatorname{argmin}} \int_{\mathcal{X}\times\mathcal{Y}} L_{0/1}(h(\boldsymbol{x}), y) P^{te}(\boldsymbol{x}) P(y|\boldsymbol{x}) d\boldsymbol{x}dy, \quad (P^{te}(\boldsymbol{x}, y) = P^{te}(\boldsymbol{x})P(y|\boldsymbol{x})) && (17)\\
&= \underset{h}{\operatorname{argmin}} \int_{\mathcal{X}\times\mathcal{Y}} L_{0/1}(h(\boldsymbol{x}), y) \frac{P^{te}(\boldsymbol{x})}{P^{tr}(\boldsymbol{x})} P^{tr}(\boldsymbol{x}) P(y|\boldsymbol{x}) d\boldsymbol{x}dy, \quad (P^{te}(\boldsymbol{x}) = \frac{P^{te}(\boldsymbol{x})}{P^{tr}(\boldsymbol{x})} P^{tr}(\boldsymbol{x})) && (18)\\
&= \underset{h}{\operatorname{argmin}} \int_{\mathcal{X}\times\mathcal{Y}} L_{0/1}(h(\boldsymbol{x}), y) \frac{P^{te}(\boldsymbol{x})}{P^{tr}(\boldsymbol{x})} P^{tr}(\boldsymbol{x}, y) d\boldsymbol{x}dy, \quad (P^{tr}(\boldsymbol{x}, y) = P^{tr}(\boldsymbol{x})P(y|\boldsymbol{x})) && (19)\\
&\approx \underset{h}{\operatorname{argmin}} \frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} \frac{P^{te}(\boldsymbol{x}_i^{tr})}{P^{tr}(\boldsymbol{x}_i^{tr})} L_{0/1}(h(\boldsymbol{x}_i^{tr}), y_i^{tr}). \quad \text{(law of large numbers)} && (20)
\end{aligned}$$

The resultant optimization problem is

$$\min_h \frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} \frac{P^{te}(\boldsymbol{x}_i^{tr})}{P^{tr}(\boldsymbol{x}_i^{tr})} L_{0/1}(h(\boldsymbol{x}_i^{tr}), y_i^{tr}). \tag{21}$$

2.2 Distribution Ratio Function Estimation

Since the distribution ratio function $\frac{P^{te}(\boldsymbol{x})}{P^{tr}(\boldsymbol{x})}$ is unknown, we should estimate it from the unlabeled training dataset $\{\boldsymbol{x}_i^{tr}\}_{i=1}^{m_{tr}}$ and unlabeled testing dataset $\{\boldsymbol{x}_i^{te}\}_{i=1}^{m_{te}}$, where $\boldsymbol{x}_i^{tr} \sim P^{tr}(\boldsymbol{x})$ and $\boldsymbol{x}_i^{te} \sim P^{te}(\boldsymbol{x})$. We use a function $r(\boldsymbol{x})$ to approximate the unknown distribution ratio function $\frac{P^{te}(\boldsymbol{x})}{P^{tr}(\boldsymbol{x})}$, exploit the square loss $L_{\text{squ}}(r(\boldsymbol{x}), \frac{P^{te}(\boldsymbol{x})}{P^{tr}(\boldsymbol{x})}) = \left(r(\boldsymbol{x}) - \frac{P^{te}(\boldsymbol{x})}{P^{tr}(\boldsymbol{x})}\right)^2$ as the loss function, and solve the following optimization problem to learn the optimal function $r^*(\boldsymbol{x})$

$$
\begin{aligned}
r^*(\boldsymbol{x}) &= \underset{r}{\operatorname{argmin}} \int_{\mathcal{X}} \left(r(\boldsymbol{x}) - \frac{P^{te}(\boldsymbol{x})}{P^{tr}(\boldsymbol{x})}\right)^2 P^{tr}(\boldsymbol{x}) d\boldsymbol{x} && (22)\\
&= \underset{r}{\operatorname{argmin}} \Big(\int_{\mathcal{X}} r(\boldsymbol{x})^2 P^{tr}(\boldsymbol{x}) d\boldsymbol{x} - 2\int_{\mathcal{X}} r(\boldsymbol{x}) P^{te}(\boldsymbol{x}) d\boldsymbol{x} + \int_{\mathcal{X}} \Big(\frac{P^{te}(\boldsymbol{x})}{P^{tr}(\boldsymbol{x})}\Big)^2 P^{tr}(\boldsymbol{x}) d\boldsymbol{x} \Big) && (23)\\
&= \underset{r}{\operatorname{argmin}} \Big(\int_{\mathcal{X}} r(\boldsymbol{x})^2 P^{tr}(\boldsymbol{x}) d\boldsymbol{x} - 2\int_{\mathcal{X}} r(\boldsymbol{x}) P^{te}(\boldsymbol{x}) d\boldsymbol{x} \Big), && (24)\\
&\approx \underset{r}{\operatorname{argmin}} \Big(\frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} r(\boldsymbol{x}_i^{tr})^2 - \frac{2}{m_{te}} \sum_{i=1}^{m_{te}} r(\boldsymbol{x}_i^{te}) \Big). \quad \text{(law of large numbers)} && (25)
\end{aligned}
$$

Let the function $r(\boldsymbol{x})$ be a nonlinear function from the kernel function space $\mathcal{H}_{\text{ker}} = \{r(\boldsymbol{x};\boldsymbol{\alpha}) | r(\boldsymbol{x};\boldsymbol{\alpha}) = \boldsymbol{\alpha}^\top \boldsymbol{k}(\boldsymbol{x}) = \alpha_1 k(\boldsymbol{x}, \boldsymbol{x}_1^{te}) + ... + \alpha_{m_{te}} k(\boldsymbol{x}, \boldsymbol{x}_{m_{te}}^{te})\}$. Then the optimal parameters $\widehat{\boldsymbol{\alpha}}$ of the function r can be learned by solving the following optimization problem

$$
\widehat{\boldsymbol{\alpha}} = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \Big(\frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} r(\boldsymbol{x}_i^{tr};\boldsymbol{\alpha})^2 - \frac{2}{m_{te}} \sum_{i=1}^{m_{te}} r(\boldsymbol{x}_i^{te};\boldsymbol{\alpha}) \Big) + \gamma \|\boldsymbol{\alpha}\|^2, \tag{26}
$$

where a regularization term $\gamma\|\boldsymbol{\alpha}\|^2$ with regularization parameter $\gamma > 0$ is added to avoid overfitting.

2.3 Optimization Problem and Prediction

We use the surrogate loss functions (*e.g.*, square loss, hinge loss, exponential loss, logistic loss) to replace the 0-1 loss and solve one of the following optimization problems to learn the optimal parameters $\widehat{\boldsymbol{\theta}}$ of the function h

- square loss.

$$
\min_{\boldsymbol{\theta}} \frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} r(\boldsymbol{x}_i^{tr};\widehat{\boldsymbol{\alpha}}) \big(1 - y_i^{tr} h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta})\big)^2 + \lambda\|\boldsymbol{\theta}\|^2 \tag{27}
$$

  This is the optimization problem of importance weighted least squares classification algorithm.

- hinge loss.

$$
\min_{\boldsymbol{\theta}} \frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} r(\boldsymbol{x}_i^{tr};\widehat{\boldsymbol{\alpha}}) \max\big\{0, 1 - y_i^{tr} h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta})\big\} + \lambda\|\boldsymbol{\theta}\|^2 \tag{28}
$$

  This is the optimization problem of importance weighted Support Vector Machine (SVM) algorithm.

- logistic loss.

$$
\min_{\boldsymbol{\theta}} \frac{1}{m_{tr}} \sum_{i=1}^{m_{tr}} r(\boldsymbol{x}_i^{tr};\widehat{\boldsymbol{\alpha}}) \log\big(1 + \exp(-y_i^{tr} h(\boldsymbol{x}_i^{tr};\boldsymbol{\theta}))\big) + \lambda\|\boldsymbol{\theta}\|^2 \tag{29}
$$

  This is the optimization problem of importance weighted logistic regression algorithm.

Finally, prediction is performed via the equation $y = \text{sign}(h(\boldsymbol{x};\widehat{\boldsymbol{\theta}}))$.