24

LINEAR MODELS FOR CLASSIFICATION

24.1 Logistic regression

24.1.1 Logistic regression model

Logistic regression model is one of the most important models for classification tasks. Given the observation X of the features, logistic regression models the log-odd of the outcome label Y taking 0 or 1 via following linear relationship

$$\log \frac{P(Y=1|X=x)}{p(Y=0|X=x)} = \log\left(\frac{P(Y=1|X=x)}{1-p(Y=1|X=x)}\right) = \beta_0 + \beta^T x$$

where $\beta_0 \in \mathbb{R}$ and $\beta \in \mathbb{R}^p$ are model parameters. Explicitly, let $p(x) = P(Y=1|X=x)$, we have

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta^T x = \beta_1 x1 + \ldots + \beta_p x_p$$

The interpretation of coefficient β_i is that: if we increase X_j by one unit, and keeping all other features fixed, the change of the log odd is given by β_j.

The conditional probability can be explicitly written as

$$P(Y=1|X=x) = \frac{\exp(\beta_0 + x^T\beta)}{1+\exp(\beta_0 + x^T\beta)},$$

and

$$P(Y=0|X=x) = 1 - P(Y=1|X) = \frac{1}{1+\exp(\beta_0 + x^T\beta)}.$$

Suppose that we have a logistic regression model with estimated $\hat{\beta}_0, \hat{\beta}$, given a new input $x \in \mathbb{R}^p$, we can predict its classification probability by

$$\hat{p}(x) = \frac{\exp\left(\beta_0 + \hat{\beta}^T x\right)}{1+\exp\left(\beta_0 + \hat{\beta}^T x\right)}$$

and its label can be decided via

$$\hat{y} = \begin{cases} 0 & \hat{p}(x) \leq T \\ 1 & \hat{p}(x) > T \end{cases}$$

where $T \in [0,1]$ is the threshold usually taking $T = 0.5$.

Notably, the geometry of the decision is any x lying on the half-space

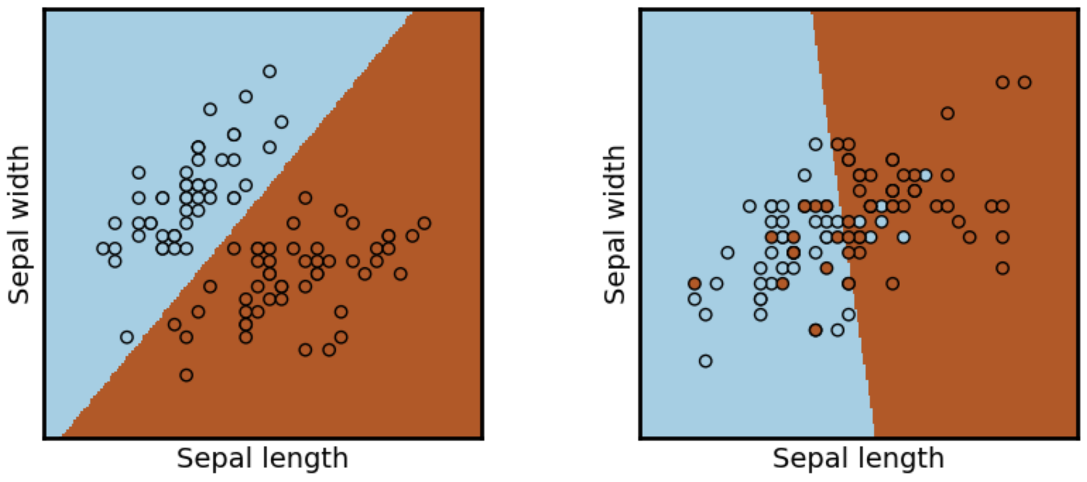
$$H^+ = \{x \in \mathbb{R}^p : \hat{\beta}_0 + \hat{\beta}_1 x_1 + \ldots + \hat{\beta}_p x_p \geq 0\}$$

will be classified with label 1. Similarly, any x lying on the half-space $H^- = \{x \in \mathbb{R}^p : \hat{\beta}_0 + \hat{\beta}x < 0\}$ will be classified with label 0. The hyperplane

$$\hat{\beta}_0 + \hat{\beta}_1 x_1 + \ldots + \hat{\beta}_p x_p = 0$$

is known as the **decision boundary**. Clearly, the decision boundary has a linear geometry.

In Figure 24.2.1, we demonstrated the **linear decision boundary** when performing binary classification using the Iris dataset.



(a) Decision boundary of a two-class (setosa and versicolor) classification problem using the Iris data set.

(b) Decision boundary of a two-class (versicolor and virginica) classification problem using the Iris data set.

Figure 24.1.1: Logistic regression for classification on the Iris data set.

24.1.2 Parameter estimation via maximum likelihood estimation

The estimation of β is via maximum likelihood estimation (MLE). Given N observations $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$, with $x_i \in \mathbb{R}^p, y_i \in \{0, 1\}$. Denote $p_i = P(Y_i = 1 | X = x_i)$. The log-likelihood function is given by

$$L(\beta) = \log \frac{i=1}{N} p_i^{y_i}(1-p_i)^{(1-y_i)} = \sum_{i=1}^{N}(y_i \log p_i + (1-y_i)\log(1-p_i)).$$

In general, we do not have closed-form solution of $\hat{\beta}$ that maximizes $L(\beta)$ and seek solutions via iterative gradient descent. In the follow Lemma, we establish gradient and Hessian.

Also note that maximizing the log-likelihood function is also equivalent to minimizing the binary cross entropy [Definition 22.3.2].

Lemma 24.1.1 (likelihood function, gradient and Hessian for binary logistic regression). *The log-likelihood function for* ***binary logistic regression*** *with N observations is given by*

$$L(\beta) = \sum_{i=1}^{N} (y_i \beta^T x_i - \log(1 + \exp(\beta^T x_i))),$$

where $y_i \in \{0, 1\}, x_i \in \mathbb{R}^p, \beta \in \mathbb{R}^p$.

The ***gradient*** *and* ***Hessian*** *of the log-likelihood function are given by the following*

$$g(\beta) = \frac{dL(\beta)}{d\beta} = \sum_{i=1}^{N} x_i(y_i - p_i(x_i;\beta)) = X^T(y - p), y \in \mathbb{R}^n, p \in \mathbb{R}^n, X \in \mathbb{R}^{n\times p}.$$

$$H(\beta) = \frac{dg^T(\beta)}{d\beta} = -\sum_{i=1}^{N} x_i x_i^T p(x_i;\beta)(1 - p(x_i;\beta)) = -X^T W X$$

where $p_i(x_i;\beta)$ is the probability of being 1 for input x_i and parameter β, W is an $N \times N$ diagonal matrix of weights with

$$W_{ii} = p(x_i;\beta)(1 - p(x_i;\beta)), i = 1, 2, ..., N.$$

Proof. (1)

$$\begin{aligned} L(\beta) &= \sum_{i=1}^{N} (y_i \log p_i + (1 - y_i)\log(1 - p_i)) \\ &= \sum_{i=1}^{N} (y_i \beta^T x_i - y_i \log(1 + \exp(\beta^T x_i) + (1 - y_i)\log 1 - (1 - y_i)\log(1 + \exp(\beta^T x)) \\ &= \sum_{i=1}^{N} (y_i \beta^T x_i - \log(1 + \exp(\beta^T x_i))) \end{aligned}$$

(2)

$$\begin{aligned}
g(\beta) &= \frac{dL(\beta)}{d\beta} \\
&= \frac{d}{d\beta}(\sum_{i=1}^{N}(y_i\beta^T x_i - \log(1+\exp(\beta^T x_i)))) \\
&= \sum_{i=1}^{N}(y_i x_i - \frac{x_i \exp(\beta^T x_i)}{1+\exp(\beta^T x_i)}) \\
&= \sum_{i=1}^{N}(y_i x_i - x_i p_i) \\
&= \sum_{i=1}^{N} x_i(y_i - p_i(x_i;\beta)) \\
&= X^T(y-p)
\end{aligned}$$

and

$$\begin{aligned}
H(\beta) &= \frac{dg^T(\beta)}{\mathrm{d}\beta} \\
&= \frac{d}{d\beta}\sum_{i=1}^{N} x_i(y_i - p_i(x_i;\beta)) \\
&= \sum_{i=1}^{N} -x_i(\frac{d}{d\beta}p_i(x_i;\beta)) \\
&= \sum_{i=1}^{N} -x_i(x_i^T p_i - x_i^T p_i^2) \\
&= \sum_{i=1}^{N} -x_i x_i^T p_i(1-p_i) \\
&= -X^T W X
\end{aligned}$$

□

Note that the Hessian is negative semi-definite (i.e., W is a diagonal matrix with non-negative entries). Therefore we can design gradient descent with Newton step. In the following Lemma, we show that each iteration can also be solved in the generalized linear regression framework [Theorem 15.1.12]. Further, because L is concave, the iterative β will finally converge to global optimum. The complete algorithm is given by algorithm 33.

Lemma 24.1.2 (Iteratively reweighted least squares (IRLS) for logistic regression). *[1, p. 250] Let β_{old} be the current iterate of β in maximizing $L(\beta)$ and W be the weighting matrix associated with β_{old}. Then the Newton-Raphson step is given by*

$$\begin{aligned}\beta^{new} &= \beta^{old} + H^{-1}g \\ &= (X^TWX)^{-1}X^TWz\end{aligned}$$

which is equivalent to the following optimization problem given by

$$\beta^{new} = \arg\min_{\beta}(z - X\beta)^T W(z - X\beta).$$

where $z \triangleq (X\beta^{old} + W^{-1}(y-p))$.

Proof.

$$\begin{aligned}\beta^{new} &= \beta^{old} + (X^TWX)^{-1}X^T(y-p) \\ &= (X^TWX)^{-1}X^TW(X\beta^{old} + W^{-1}(y-p)) \\ &= (X^TWX)^{-1}X^TWz\end{aligned}$$

where $z \triangleq (X\beta^{old} + W^{-1}(y-p))$. This is also the solution to a generalized linear regression [Theorem 15.1.12]. □

Algorithm 33: Iteratively reweighted least squares for logistic regression

```
  Input: Data set consists of $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$
1 Set $\beta = 0$, and compute $p$ by setting its components to
      $$p(x_i;\beta) = \frac{\exp(\beta^T x_i)}{1+\exp(\beta^T x_i)}, i = 1, 2, ..., N.$$
2 Compute the diagonal matrix $W$ with diagonal element being
      $$W_{ii} = p(x_i;\beta)(1 - p(x_i;\beta)), i = 1, 2, ..., N.$$
3 repeat
4     $z = X\beta + W^{-1}(y - p)$
5     $\beta = (X^TWX)^{-1}X^TWz.$
6 until stopping criteria is met;
  Output: the optimalized value $\beta$.
```

24.1.3 Logistic regression with regularization

In an effort to prevent overfitting, we can add L_2 penalty on the model parameters β. The model parameter gradient with L_2 penalty is given below. We can pass these modified gradients into any gradient-based optimizer to find the optimizer.

Corollary 24.1.0.1 (MLE with L^2 regularization). *The maximum likelihood problem for the parameter β under L^2 regularization is given by* [a]

$$\max_{\beta} L_r(\beta, \lambda) \triangleq L(\beta) - \lambda \beta^T \beta$$

$$\max_{\beta} \sum_{i=1}^{N} (y_i \beta^T x_i - \log(1 + \exp(\beta^T x_i))) - \lambda \beta^T \beta$$

and the gradient and Hessian are given by

$$\begin{aligned} g_r(\beta) &= g(\beta) + \lambda \beta \\ H_r(\beta) &= H(\beta) + \lambda \boldsymbol{I} \end{aligned}$$

where $L(\beta), g(\beta)$, and $H(\beta)$ are the original likelihood function, gradient and Hessian [Lemma 24.1.1).

[a] β_0 is not penalized.

Proof. Directly use linearity of differentiation. □

Similarly, we can use L_1 penalty to select sparse features.

Corollary 24.1.0.2 (MLE with L^1 regularization). *The maximum likelihood problem for the parameter β under L^1 regularization is given by* [a]

$$\max_{\beta} L_r(\beta, \lambda) \triangleq L(\beta) - \lambda \sum_{j=1}^{p} \left|\beta_j\right|$$

$$\max_{\beta} \sum_{i=1}^{N} (y_i \beta^T x_i - \log(1 + \exp(\beta^T x_i))) - \lambda \sum_{j=1}^{p} \left|\beta_j\right|$$

and the gradient and Hessian are given by

$$\begin{aligned} g_r(\beta) &= g(\beta) + \lambda \beta \\ H_r(\beta) &= H(\beta) + \lambda \boldsymbol{I} \end{aligned}$$

where $L(\beta), g(\beta)$, and $H(\beta)$ are the original likelihood function, gradient and Hessian [Lemma 24.1.1].

a β_0 is not penalized.

24.1.4 Feature augmentation strategies

Logistic regression is a scalable machine learning algorithm that has benefit of efficient computation and the ability to cope with large-scale features. In modeling complex relations, a linear regression model with simple and small-scale features can suffer from underfitting. A general strategy to overcome underfitting is to introduce additional features, usually generated from existing features via non-linear mapping (e.g., polynomial terms) and discretization of continuous features. Thanks to the scalability of logistic regression, we can apply these feature augmentation techniques to increase model complexity in the logistic regression framework without concerning the computational cost. For example, discretizing continuous feature into multiple discrete features amounts to introduce learnable nonlinear features. These discrete features can be further combined to introduce cross terms.

A more formal way to introduce cross-terms feature while avoiding the exploding number of model parameters is via factorization machine [2]. The quadratic factorization machine model on log-odd is defined as

$$\hat{y}(x) := w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} W_{ij} x_i x_j$$

where the model parameters are:

$$w_0 \in \mathbb{R}, \quad w_1, ... w_n \in \mathbb{R}^n, \quad W \in \mathbb{R}^{n\times n}.$$

To facilitate robust estimation of quadratic term coefficients W, we introduce low rank approximation to W via

$$W \approx v_1^T v_1 + \cdots + v_n^T v_n, = V^T V, V \in \mathbb{R}^{k\times n}, k \ll n.$$

Then we can write the model as

$$\hat{y}(x) := w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} v_i^T v_j x_i x_j.$$

The parameter estimation of w, v can be obtained via following optimization problem [also see subsection 24.5.5]

$$\begin{aligned} l(w,v) =& \frac{1}{m}\sum_{h=1}^{m} \ln\left(1+e^{-y^{(h)}\left(w_0+\sum_{i=1}^{n} w_i x_i+\sum_{i=1}^{n}\sum_{j=i+1}^{n} v_i^T v_j x_i x_j\right)}\right) \\ &+ \frac{\lambda_0}{m}w_0^2+\frac{\lambda_1}{m}\sum_{i=1}^{n} w_i^2+\frac{\lambda_2}{m}\sum_{i=1}^{n} v_i^T v_i, y^{(h)} \in \{-1,1\} \end{aligned}$$

24.1.5 Multinomial logistic regression

The binary logistic regression model can be generalize to multiple class regression, known as **multinomial logistic regression**. It is also called a **maximum entropy classifier**, which has the following form

Definition 24.1.1 (logistic regression model for K classies). *Let the training data consists of N pairs $(x_1,y_1),(x_2,y_2),...,(x_N,y_N)$, with $x_i \in \mathbb{R}^p, y_i \in \{1,2,...,K\}$. Then the* ***multinomial logistic model*** *models the probabilities as*

$$p(y=c|x,W) = \frac{\exp(w_c^T x)}{\sum_{c=1}^{C}\exp(w_c^T x)} \tag{9}$$

where $w_i \in \mathbb{R}^p$ and $W = \{w_1,...,w_K\}$ are model parameters.

Usually, we use MLE to estimate the coefficient W. Let $y_i = (\mathbb{I}(y_i = 1),\mathbb{I}(y_i = 1),\cdots,\mathbb{I}(y_i = K))$, $\mu_i = (p(y=1|x_i,W),p(y=2|x_i,W),\cdots,p(y=K|x_i,W))$, then the log-likelihood function can be written as

$$\begin{aligned} \ell(W) &= \log\prod_{i=1}^{N}\prod_{c=1}^{K}\mu_{ic}^{y_{ic}} = \sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic}\log\mu_{ic} \\ &= \sum_{i=1}^{N}\left[\left(\sum_{c=1}^{K} y_{ic} w_c^T x_i\right) - \log\left(\sum_{c=1}^{K}\exp(w_c^T x_i)\right)\right] \end{aligned}$$

Define $A \otimes B$ be the **kronecker product** of matrices A and B.If A is an $m \times n$ matrix and B is a $p \times q$ matrix, then $A \otimes B$ is the $mp \times nq$ block matrix

$$A \otimes B \triangleq \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \vdots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix}$$

The gradient and Hessian are given by

$$g(W) = \sum_{i=1}^{N} (\mu - y_i) \otimes x_i$$
$$H(W) = \sum_{i=1}^{N} (\mathrm{diag}(\mu_i) - \mu_i \mu_i^T) \otimes (x_i x_i^T)$$

where $y_i = (\mathbb{I}(y_i = 1), \mathbb{I}(y_i = 1), \cdots, \mathbb{I}(y_i = K))$ and $\mu_i = (p(y = 1|x_i, W), p(y = 2|x_i, W), \cdots, p(y = K|x_i, W))$ are column vectors of length K.

We can use a gradient-based optimizer to perform MLE.

Remark 24.1.1 (deriving gradient the Hessian)**.**

- Note that
$$l = \sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \ln \mu_{nk},$$
then
$$\begin{aligned}\frac{\partial l}{\partial w_j} &= -\sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \frac{1}{\mu_{nk}} \frac{\partial \mu_{nk}}{w_j} \\ &= -\sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \frac{1}{\mu_{nk}} \frac{\exp(w_k^T x_n)}{(\exp(w_k^T x_n))^2} (-\exp(w_k^T x_n) x_n + x_n \delta_{ij}) \\ &= -\sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \frac{1}{\mu_{nk}} \mu_{nk} \mu_{nj} x_n + x_n \delta_{jk} \mu_{nk} \\ &= \sum_{n=1}^{N} \sum_{k=1}^{K} (y_{nk} \mu_{nj} x_n + y_{nk} \frac{1}{\mu_{nk}} x_n \delta_{jk}) \\ &= \sum_{n=1}^{N} x_n (\mu_{nj} - y_{nj})\end{aligned}$$

24.1.6 Application examples

24.1.6.1 *South Africa heart disease*

Here we follow the South Africa heart disease example consider in [3, p. 122]. The features are sbp (Systolic blood pressure), tobaaco, obseity, etc. and the label is binary variable representing the presence or absence of heart disease at the time of the survey.

The pair plot analysis [Figure 24.1.2] shows that features alcohol and obesity barely depend on the label as they are similar conditional distributions (conditioned on the label). The coefficients in the logistic regression [Table 24.1.1] also show that these two features have low z-score. Further analysis on the L1 regularzation path [Figure 24.1.3] also indicates similar feature importance.

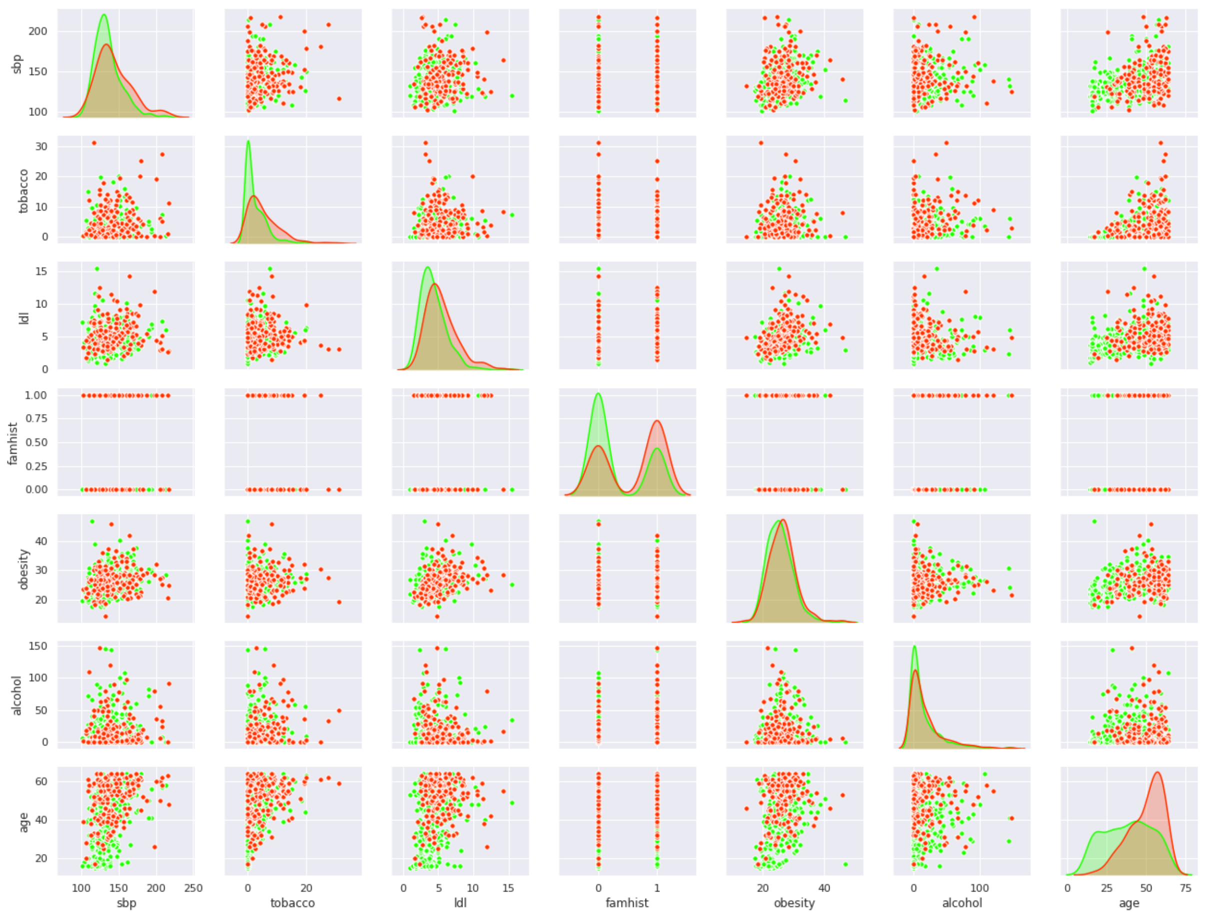


Figure 24.1.2: Pair plot analysis results on South Africa heart disease problem.

	Coefficient	Std. Error	Z-Score
(Intercept)	-4.13	0.964	-4.283
sbp	0.006	0.006	1.023
tobacco	0.08	0.026	3.034
ldl	0.185	0.057	3.218
famhist	0.939	0.225	4.177
obesity	-0.035	0.029	-1.187
alcohol	0.001	0.004	0.136
age	0.043	0.01	4.181

Table 24.1.1: Logistic regression results on South Africa heart disease problem.

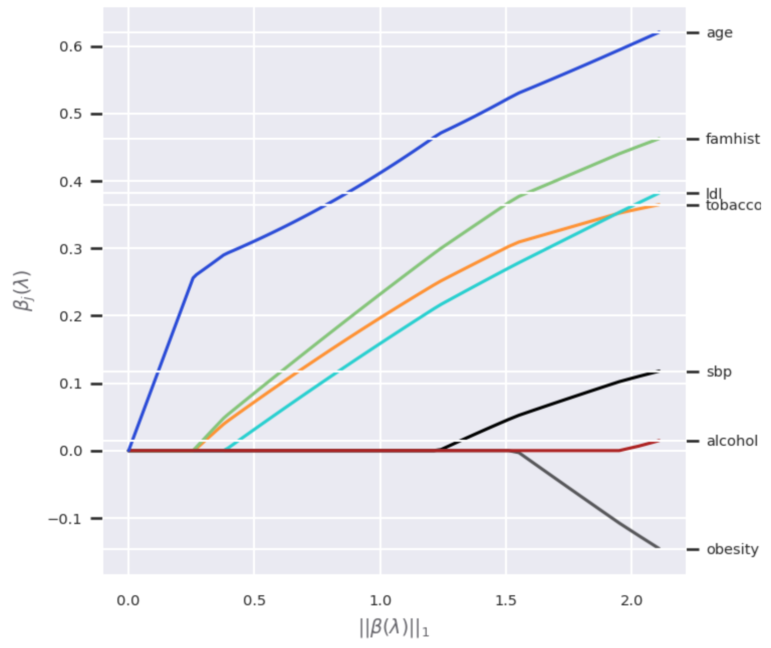


Figure 24.1.3: L1 regularization path for South Africa heart disease problem.

24.1.6.2 *Credit card fraud detection*

Now we apply the logistic regression method to fraud detection problem. We have transaction data that contains a number of features and are labeled as **fraud** or **genuine**. The feature s includes transition amount, time, and other confidential features. In the first stage feature screening, we plot the histogram of each continuous feature with respect to label. Note that because the training examples are highly imbalanced (genuine transactions are dominant), we need to use class weight to balance the sample in the gradient descent process. Specifically, each sample is associated with its class weight.

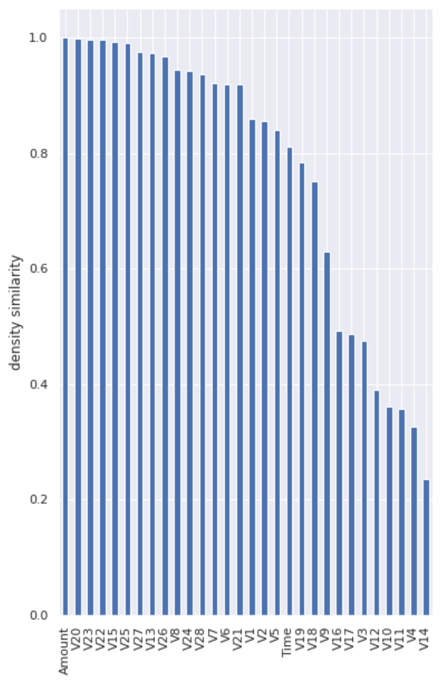
It is clear that there are some features' distribution/histogram conditioned on the label are similar; thus they will highly not contribute to classification simultaneously. What is more, including these feature will make the model are complicated and likely lead to overfitting.

We characterize the similarity in the conditional distribution via a quantity, density similarity, defined as

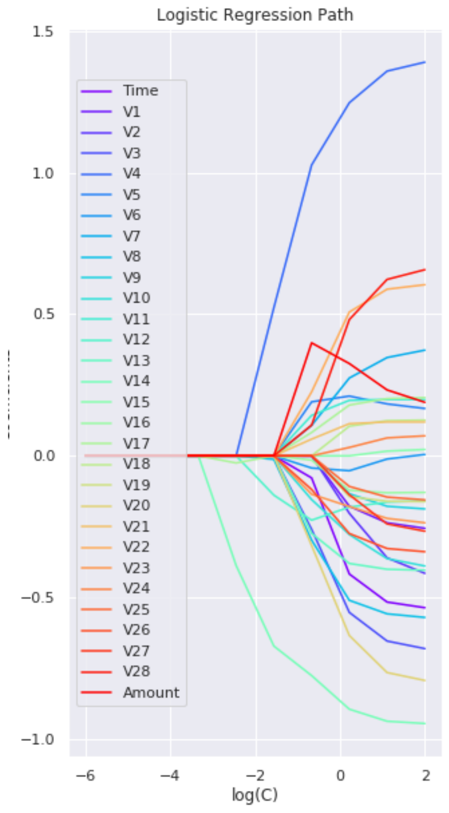
$$s(d_i) = d_i^{fraud} \cdot d_j^{genuine},$$

where $d_i^{fraud}, d_i^{genuine}$ is the conditional probability vector for feature i with label **fraud** and **genuine**, respectively. The results are showed in Figure 24.1.4 (a). We can see that some features can have density similarity as large as 1.

We perform feature selection by adding L1 regularization [subsection 24.1.3]. The coefficient regularization path is showed in Figure 24.1.4 (b). We can see that coefficients get shrunken rapidly. The classification performance on the testing data set, characterized by balance-accuracy [subsubsection 22.3.2.4], is showed in Figure 24.1.5. The classifier has better performance at proper regularization levels.



(a) Conditional pdf similarity.



(b) L1 regularization path (inverse regularization strength) in credit card fraud detection problem.

Figure 24.1.4: Logistic regression result with L1 penalty.

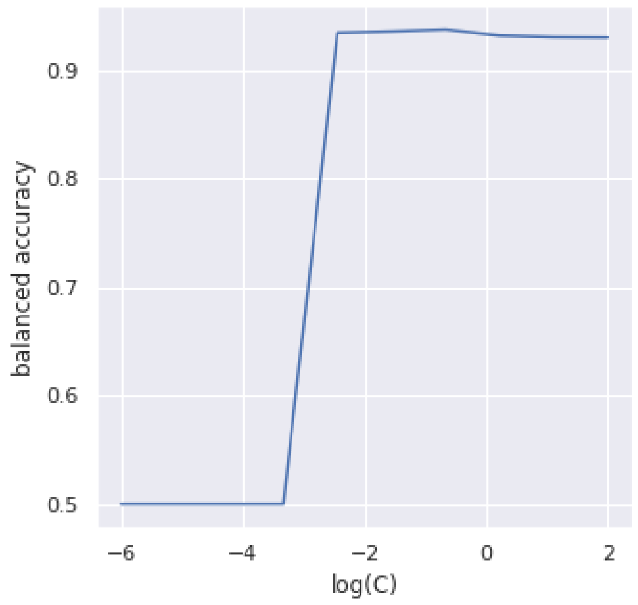


Figure 24.1.5: Balanced accuracy score vs. inverse regularization strength in credit card fraud detection problem.

24.1.6.3 *MNIST*

Here we consider the problem of classifying MNIST data set using Logistic regression one over the rest scheme. Because the likelihood for one class is proportional to $\beta_0 + \beta_1 x$, β_1 illustrates the pixels in the image that are contributes to classification.

In Figure 24.1.6, we plot the β (reshaped as squared images) coefficient for each class/digit, which clearly shows the most salient features that distinguish the digit.

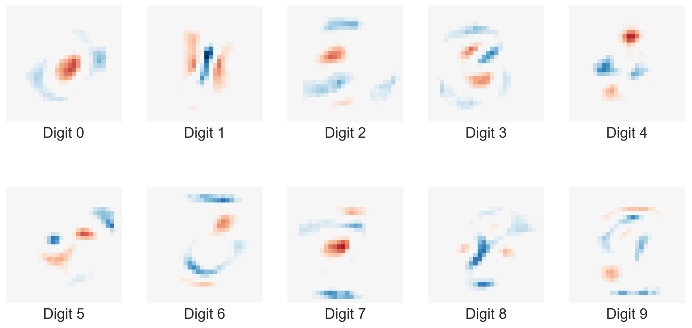


Figure 24.1.6: Logistic regression coefficients corresponding to each class.

24.2 Gaussian discriminate analysis

24.2.1 Linear Gaussian discriminant model

24.2.1.1 *The model*

Definition 24.2.1 (linear discriminant model). *Let the training data consists of N pairs $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$, with $x_i \in \mathbb{R}^p, y_i \in \{1, ..., K\}$. Then the **linear Gaussian discriminant model** models the pdf as*

$$\begin{aligned} f(X|Y=1) &\sim MN(\mu_1, \Sigma), \\ f(X|Y=2) &\sim MN(\mu_2, \Sigma), \\ &\cdots \\ f(X|Y=K) &\sim MN(\mu_K, \Sigma). \end{aligned}$$

where $\mu_1, ..., \mu_K \in \mathbb{R}^p, \Sigma \in \mathbb{R}^{p\times p}$ and prior probabilities of classes $\pi_1, ..., \pi_K$ are model parameters. Based on Bayes rule, we have the posterior probability

$$Pr(Y=k|X=x) = \frac{f(X=x|Y=k)\pi_k}{\sum_{k=1}^K f(X=x|Y=k)\pi_k}.$$

Under the linear discriminate model framework, we can compare outcome probability given x in the following way. Denote $f_k(x) = f(X|Y=k)$. Consider we want to compare $P(Y=1|x) > P(Y=2|x)$ via

$$\begin{aligned} \ln\frac{P(Y=1|x)}{P(Y=2|x)} &= \ln\frac{f_1(x)\pi_1}{f_2(x)\pi_2} \\ &= \ln\frac{\pi_1}{\pi_2} + \frac{1}{2}((x-\mu_2)^T\Sigma^{-1}(x-\mu_2) - (x-\mu_1)^T\Sigma^{-1}(x-\mu_1)) \\ &= \ln\frac{\pi_1}{\pi_2} - x^T\Sigma^{-1}(\mu_2-\mu_1) + \frac{1}{2}\mu_2^T\Sigma^{-1}\mu_2 - \frac{1}{2}\mu_1^T\Sigma^{-1}\mu_1) \end{aligned}$$

Note that quadratic terms of x are canceled out. A step further and we can summarize the following linear discriminate function for classification application.

Methodology 24.2.1 (linear discriminate functions for classification). *Suppose model parameters μ, Σ, π are given. Given a new input x, we can use following way to classify x.*

- *Define **linear discriminate functions** for each class K as*

$$\delta_k(x) = x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \log \pi_k.$$

- *x belongs to class C that has the largest $\delta_k(x)$.*

24.2.1.2 *Model parameter estimation*

From the maximum likelihood estimation (MLE) theory covered in Lemma 14.1.6, we have the following:

Lemma 24.2.1 (maximum likelihood estimator).

- $$\hat{\pi}_k = \frac{N_k}{N}, N_k = \sum_{i=1}^{N} \mathbf{1}(y_i = k).$$
- $$\hat{\mu}_k = \frac{\sum_{i:y_i=k} x_i}{N_k}.$$
- $$\hat{\Sigma} = \sum_{k=1}^{K} \sum_{i:y_i=k} \frac{(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T}{N}.$$

Corollary 24.2.0.1 (special case for one dimensional input space). *For the case $p = 1$, we can estimate $\hat{\mu}, \hat{\sigma}^2$ using following procedures:*

- $$\hat{\mu}_k = \frac{1}{N_k} \sum_{i:y_i=k} x_i.$$
- $$\hat{\sigma}^2 = \frac{1}{N} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2.$$

24.2.1.3 *Geometry of decision boundary*

The parameter setup in the model can lead to different decision boundaries [Figure 24.2.1].

For two-class classification problem, the decision boundary determined by $\delta_1(x) = \delta_2(x)$ is a hyperplane given by

$$x^T \Sigma^{-1}(\mu_2 - \mu_1) + \ln \frac{\pi_1}{\pi_2} + \frac{1}{2}\mu_2^T \Sigma^{-1} \mu_2 - \frac{1}{2}\mu_1^T \Sigma^{-1} \mu_1) = 0.$$

This hyperplane has normal vector $\Sigma^{-1}(\mu_2 - \mu_1)$ and passing $\frac{\mu_1+\mu_2}{2}$ when $\pi_1 = \pi_2$ since

$$\delta_1(\frac{\mu_1 + \mu_2}{2}) = \delta_2(\frac{\mu_1 + \mu_2}{2}).$$

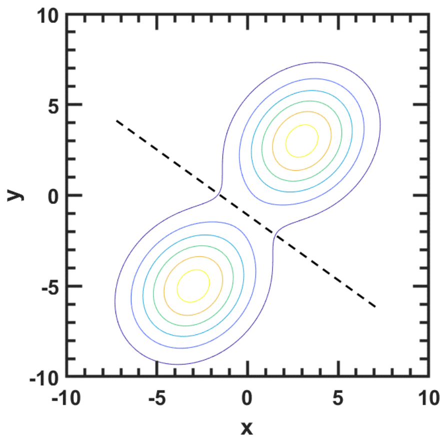
Similarly, for multiple-class classification problem, the decision boundary between any pair of Gaussian distributions are hyperplane; all hyperplanes will intersect at the same point (or line, plane, depending on dimensionality of input space.)

For a three-class classification problem with 2D input space. Three decision boundaries will intersect at the same point; it can be showed: Suppose x_0 satisfies

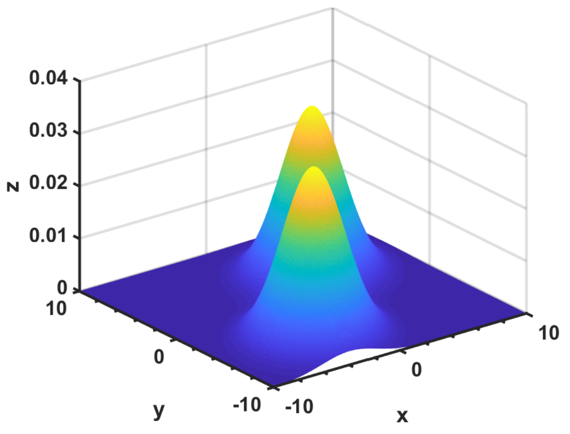
$$\delta_1(x_0) = \delta_2(x_0), \delta_1(x_0) = \delta_3(x_0),$$

then x_0 also satisfies

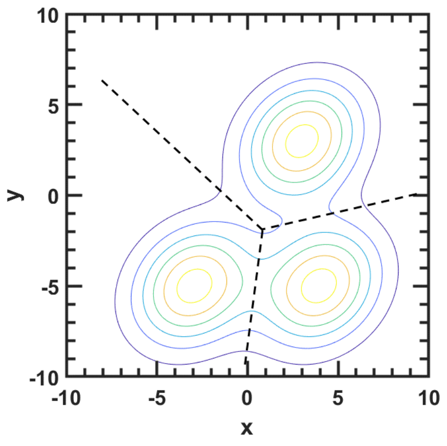
$$\delta_2(x_0) = \delta_3(x_0).$$



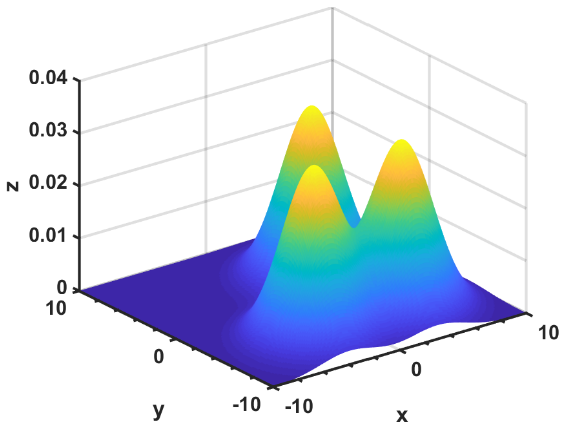
(a) Decision boundary of a two-class classification problem. The conditional distributions are represented by the contours of the Gaussian distribution.



(b) A 3D view of the conditional distributions.



(c) Decision boundary of a three-class classification problem. The conditional distributions are represented by the contours of the Gaussian distribution.



(d) A 3D view of the conditional distributions.

Figure 24.2.1: Geometry of decision boundary in linear Gaussian discriminant model.

24.2.2 Quadratic Gaussian discriminant model

24.2.2.1 *The model*

In linear Gaussian discriminant model, we assume all classes share the same covariance matrix. Now we extend the expressive power by assuming each class also has its own covariance matrix as model parameters. Such models are called quadratic Gaussian discriminant model.

Definition 24.2.2 (quadratic discriminant model). *Let the training data consists of N pairs $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$, with $x_i \in \mathbb{R}^p, y_i \in \{1, ..., K\}$. Then the Gaussian Quadratic discriminant model models the pdf as*

$$\begin{aligned} f(X|Y=1) &\sim MN(\mu_1, \Sigma_1), \\ f(X|Y=2) &\sim MN(\mu_2, \Sigma_2), \\ &\cdots \\ f(X|Y=K) &\sim MN(\mu_2, \Sigma_K) \end{aligned}$$

where $\mu_1, ..., \mu_K \in \mathbb{R}^p, \Sigma_1, \Sigma_2, ..., \Sigma_K \in \mathbb{R}^{p\times p}$ and prior probabilities of classes $\pi_1, ..., \pi_K$ are model parameters are model parameters. Based on Bayes rule, we have

$$P(Y=k|X=x) = \frac{\pi_k f_k(x)}{\sum_{i=1}^{k} \pi_k f_k(x)},$$

which can be use to predict y label given x. Here $\pi_1, ..., \pi_K$ are the prior probabilities on Y.

Similarly, we can compare outcome probability given x in the following way. Denote $f_k(x) = f(X|Y=k)$. Consider we want to compare $P(Y=1|x) > P(Y=2|x)$ via

$$\ln \frac{P(Y=1|x)}{P(Y=2|x)} = \delta_1(x) - \delta_2(x),$$

where

$$\delta_k(x) = -\frac{1}{2}\ln|\Sigma_k| - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \ln \pi_k,$$

is called quadratic discriminate function.

Methodology 24.2.2 (classification via discriminate function). *Suppose model parameters μ, Σ, π are given. Given a new input x, we can use following way to classify x.*

- *Define **linear discriminate functions** for each class K as*

$$\delta_k(x) = -\frac{1}{2}\ln|\Sigma_k| - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + \ln \pi_k.$$

- *x belongs to class C that has the largest $\delta_k(x)$.*

24.2.2.2 *Model parameter estimation*

From the maximum likelihood estimation theory covered in Lemma 14.1.6, we have the following:

Lemma 24.2.2 (maximum likelihood estimator).

- $$\hat{\pi}_k = \frac{N_k}{N}, N_k = \sum_{i=1}^{N} \mathbf{1}(y_i = k).$$
- $$\hat{\mu}_k = \frac{\sum_{i:y_i=k} x_i}{N_k}.$$
- $$\hat{\Sigma}_k = \sum_{k=1}^{K} \sum_{i:y_i=k} \frac{(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T}{N - K}.$$

Corollary 24.2.0.2 (Estimation of parameters). *For the case $p = 1$, we can estimate $\hat{\mu}, \hat{\sigma}^2$ using following procedures:*

- $$\hat{\mu}_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i.$$
- $$\hat{\sigma}_k^2 = \frac{1}{n - K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2.$$

24.2.3 Application examples

24.2.3.1 *A toy example*

In the following example [Figure 24.2.2], we consider a toy example involving binary classification on a 2D feature space. Linear Gaussian discriminant analysis (Gaussian LDA) gives linear decision boundary and encounters difficulty in separating the two classes in the 2D feature space. Gaussian discriminant analysis (Gaussian QDA) gives nonlinear boundary that results in better classification. On the other hand, by introduce cross and quadratic terms in the LDA, we can yield similar classification boundary and performance.

The nonlinear boundary in 2D feature space can be understood by the discriminate decision function. For a Gaussian LDA, its discriminate function is given by [Methodology 24.2.1],

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k.$$

which is a plane on the 3D space. The decision boundary in binary classification is the intersection of two discriminate functions, and will be a line.

On the other hand, the Gaussian QDA has ts discriminate function is given by [Methodology 24.2.2],

$$\delta_k(x) = -\frac{1}{2} \ln|\Sigma_k| - \frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) + \ln \pi_k.$$

which is a 3D quadratic surface. And the intersection of two quadratic surface will be a curve.

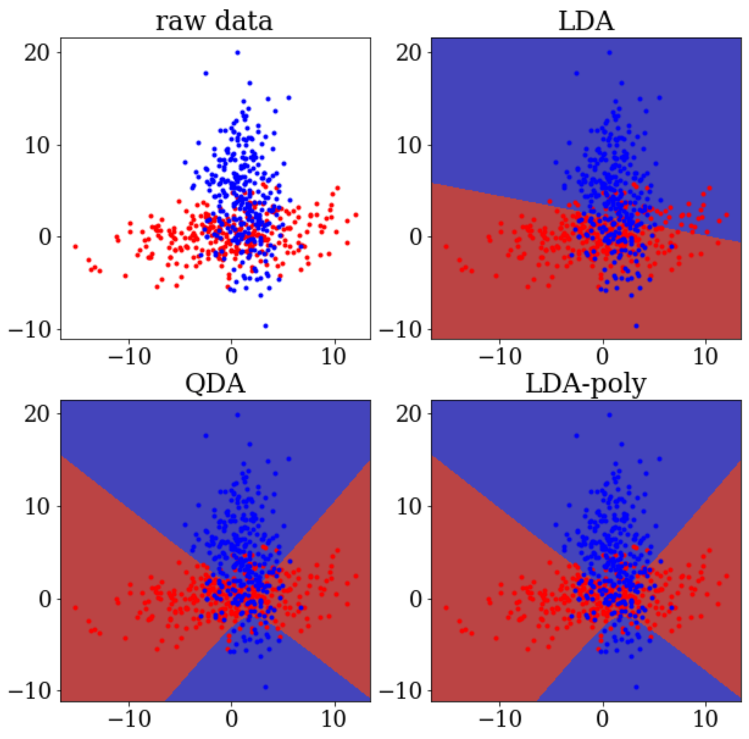


Figure 24.2.2: Comparison of Gaussian LDA and Gaussian QDA on binary classification. Decision boundary of LDA is simply a line in 2D input space and unable to discriminate difficult cases. Gaussian discrimination can have richer decision boundary geometry. LDA using polynomial features is a special case of GDA.

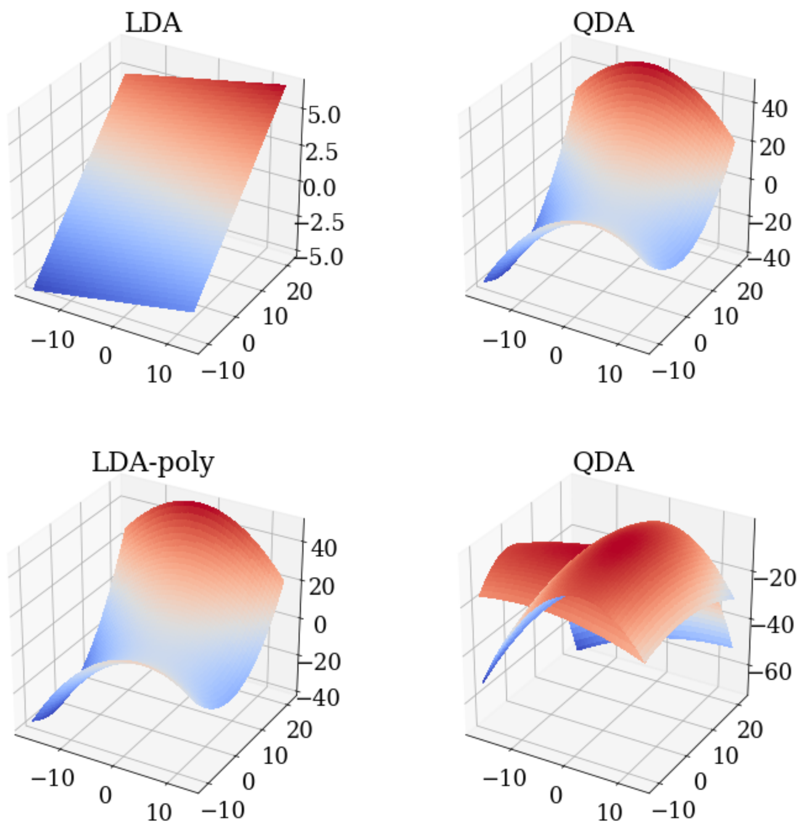


Figure 24.2.3: The decision boundary geometry of LDA and GDA can be understood via their decision functions

24.3 Fisher Linear discriminate analysis (Fisher LDA)

24.3.1 One dimensional linear discriminant

24.3.1.1 *Basics*

Fisher linear discriminate analysis (Fisher LDA) aims to find a low dimensional representation of the origin data that maximizes the class separability. Classification rules can then be constructed more naturally on the basis of the low dimension representation instead of the original data.

More specifically, Fisher LDA is to seek discriminant vectors/subspace such that the class separation are maximized when features are projected onto the discriminant subspace [Figure 24.3.1], which can be summarized as follows.

Definition 24.3.1 (Fisher linear discriminant analysis problem). *Consider a set of input data $x_1, ..., x_N \in \mathbb{R}^D$ being classified as two classes such that $y_i \in \{1, 2\}$ and the following definitions*

- *Between-class scatter matrix*

$$S_B = (\mu_1 - \mu)(\mu_2 - \mu)^T, \mu_i = \frac{1}{N_i} \sum_{j:y_j=i} x_j.$$

- *Within-class scatter matrix*

$$S_W = \underbrace{\sum_{i:y_i=1} (x_i - \mu_1)(x_i - \mu_1)^T}_{S_1, \text{class 1 scatter matrix}} + \underbrace{\sum_{i:y_i=2} (x_i - \mu_2)(x_i - \mu_2)^T}_{S_2, \text{ class 2 scatter matrix}}.$$

The goal is to seek a vector $w \in \mathbb{R}^D$ such that the projection of x onto w maximize the class separability of the projections $w^T x_1, w^T x_2, ..., w^T x_N$. The vector w is known as ***Fisher linear discriminant****.*

Or equivalently, the goal is to maximize projected mean distance over total within-class ***projected scattering*** *via maximizing the following* ***Fisher criterion function****:*

$$J(w) = \frac{\left|w^T(\mu_2 - \mu_1)\right|^2}{w^T(S_1 + S_2)w} = \frac{w^T S_B w}{w^T S_w w}.$$

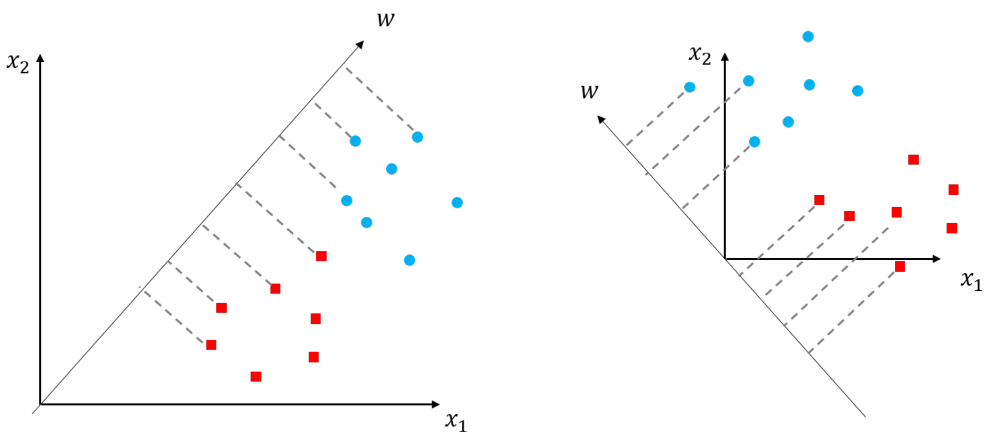


Figure 24.3.1: The linear discriminant w that maximizes the separability for 2D sample points belonging to two classes.

Lemma 24.3.1 (optimality condition for Fisher criterion function). *[4, p. 186] The optimal w^*, known as Fisher linear discriminant, that maximizes*

$$J(w) = \frac{w^T S_B w}{w^T S_W w}$$

has following properties (assuming S_W is invertible)

- *The first-order optimality condition gives $S_w^{-1} S_B w^* = J w^*$*
- *The direction of w^* is given by*[a]

$$w^* \propto S_W^{-1}(\mu_2 - \mu_1).$$

- *Let v be the eigenvector of $S_w^{-1} S_B$ and λ be the associated eigenvalue*[b]*, then*

$$w^* = v, J^* = \lambda.$$

a note that here we mean $w^* = cS_W^{-1}(\mu_2 - \mu_1)$; c is chosen such that $S_w^{-1} S_B w^* = J w^*$. However, the value J is scale invariant.

b $S_w^{-1} S_B$ is of rank 1; therefore it only has one eigen-pair.

Proof. (1) Take the derivative of $J(w)$ with respect to w, we have

$$\frac{\partial J}{\partial w} = -\frac{2 S_W w (w^T S_B w)}{(w^T S_W w)^2} + \frac{2 S_B w}{(w^T S_W w)} = 0,$$

which gives

$$\begin{aligned} &2S_w w(w^T S_B w) - 2S_B w(w^T S_W w) = 0 \\ \implies &S_B w = J S_W w \\ \implies &S_w^{-1} S_B w = Jw \end{aligned}$$

(2) Note that

$$\begin{aligned} Jw &= S_W^{-1} S_B w \\ &= S_W^{-1}(m_2 - m)\underbrace{(\mu_2 - \mu)^T w}_{c \in \mathbb{R}} \\ \implies w &= S_W^{-1}(\mu_2 - \mu)c/J \propto S_W^{-1}(\mu_2 - \mu_1). \end{aligned}$$

(3) Based on the eigenvector/eigenvalue definition, we have

$$S_W^{-1} S_B w^* = \lambda w^* \implies S_B w^* = \lambda S_W w^*.$$

Then,

$$J^* = \frac{[w^*]^T S_B w^*}{[w^*]^T S_W w^*} = \frac{[w^*]^T \lambda S_W w^*}{[w^*]^T S_W w^*} = \lambda.$$

All items can also be proved using generalized Rayleigh quotients [Corollary 4.8.4.1]. □

Remark 24.3.1 (interpretation).

- The Fisher linear discriminant is a direction that has the maximum between-class variance over within-class variance. The idea is similar in calculating top principal eigenvectors in PCA [subsection 29.1.2].
- Given an optimal w, the discriminant function is

$$y = w^T x + w_0,$$

where $w_0 \in \mathbb{R}$ is such a value that minimizes the classification error.

24.3.1.2 *Application in classification*

Lemma 24.3.2 (distance metric in the projected coordinates). *Let w be the Fisher linear discriminant and $x \in \mathbb{R}^D$ be a new input. It follows that*

- *$w = S_W^{-1/2} e$, e is the unit eigenvector of $S_W^{-1/2} S_B S_W^{-1/2}$.*
- *The distance to mean in the projected coordinates is*

$$d_1(x) = \left| w^T(x - \mu_1) \right|^2 = (x - \mu_1)^T S_W^{-1} (x - \mu_1)^T,$$

$$d_2(x) = \left|w^T(x-\mu_2)\right|^2 = (x-\mu_2)^T S_W^{-1}(x-\mu_2)^T.$$

Proof. (1) From Lemma 24.3.1, we know that w is the eigenvector of $S_W^{-1}S_B$ such that

$$\begin{aligned} S_W^{-1}S_B w &= \lambda w \\ S_W^{-1}S_B S_W^{-1/2}e &= \lambda S_W^{-1/2}e \\ S_W^{-1/2}S_B S_W^{-1/2}e &= \lambda e \end{aligned}$$

(2)

$$\begin{aligned} d_1 &= \left|w^T(x-\mu_1)\right|^2 \\ &= (x-\mu_1)^T S_W^{-1/2} e^T e S_W^{-1/2}(x-\mu_1)^T \\ &= (x-\mu_1)^T S_W^{-1}(x-\mu_1)^T \end{aligned}$$

□

Methodology 24.3.1 (Binary classification using Fisher linear discriminant). *Consider a set of input data $x_1,...,x_N \in \mathbb{R}^D$ being classified as two classes $\mathcal{C}_1$ and $\mathcal{C}_2$. Let w be the Fisher linear discriminant. The classification rule is to assign x to the class i if the projected distance $w^T(x-\mu_i)$ is the smallest.*

More formally,

- *Define the distance to mean in the projected coordinates as*

$$d_1(x) = \left|w^T(x-\mu_1)\right|^2 = (x-\mu_1)^T S_W^{-1}(x-\mu_1)^T,$$

$$d_2(x) = \left|w^T(x-\mu_2)\right|^2 = (x-\mu_2)^T S_W^{-1}(x-\mu_2)^T.$$

- *The classification rule is: if $d_1(x) \leq d_2(x)$, then x belongs to class 1; otherwise class 2.*

Remark 24.3.2 (connection to Gaussian discriminant method). Consider the situation that the individual within-class scattering matrix $S_1 = S_2 = \Sigma$. In the Gaussian discriminant model, the **linear discriminate functions** [Methodology 24.2.1] for each class k as

$$\delta_k(x) = x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \log\pi_k,$$

or equivalently,

$$\begin{aligned}\delta_k(x) - \frac{1}{2}x^T\Sigma^{-1}x &= -\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k) + \log\pi_k \\ &= -\frac{1}{2}(x-\mu_k)^T(\frac{1}{2}S_W)^{-1}(x-\mu_k) + \log\pi_k \\ &= -d_k(x) + \log\pi_k\end{aligned}$$

Therefore, when all the priors are equal; that is, $\pi_k = 1/2$, the linear discriminate function and the distance function give the same classification result.

24.3.1.3 *Possible issues*

Remark 24.3.3 (Complex data and rank deficiency for high-dimensional input)**.**

- Similar to other linear methods, applying LDA to complex data can yield poor performance [Figure 24.3.2].
- Another complication in applying LDA to real data occurs when the number features exceeds the number of samples in each class. Since we are required to calculate $S_w^{-1}S_B, S_w^{-1}$ in this case, the covariance estimates do not have full rank, and so cannot be inverted.

There are a number of ways to deal with this. One is to use a pseudo inverse instead of the usual matrix inverse in the above formula. However, better numeric stability may be achieved by first projecting the problem onto the subspace spanned by Σ_b. Another strategy to deal with small sample size is to use a shrinkage estimator of the covariance matrix, which can be expressed mathematically as

$$\Sigma = (1-\lambda)\Sigma + \lambda I\Sigma = (1-\lambda)\Sigma + \lambda I,$$

where I is the identity matrix, and λ is the shrinkage intensity or regularisation parameter. This leads to the framework of regularized discriminant analysis or shrinkage discriminant analysis

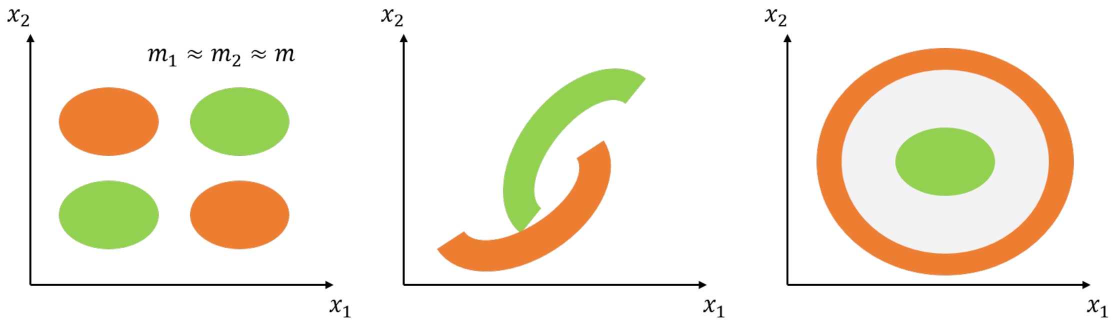


Figure 24.3.2: Fisher linear discriminate will fail to achieve class separability for complex data structures.

24.3.2 Multi-dimensional linear discriminate

24.3.2.1 *Basics*

Definition 24.3.2 (multi-dimensional linear discriminate). *[5, p. 654] Consider a set of input data $x_1, ..., x_N \in \mathbb{R}^D$ being classified as two classes such that $y_i \in \{1,2\}$ and the following definition*

- *Within-class scatter matrix*

$$S_W = \sum_{i=1}^{C} \sum_{j:y_j=i} (x_i - \mu_i)(x_i - \mu_i)^T,$$

where $\mu_i = \frac{1}{N_i} \sum_{j:y_j=i} x_i$.

- *Between-class scatter matrix*

$$S_B = \sum_{i=1}^{C} N_i(\mu_i - \mu)(\mu_i - \mu)^T,$$

where $\mu = \frac{1}{N} \sum_{i=1} x_i = \frac{1}{N} \sum_{i=i}^{C} N_i \mu_i$.

*The goal is to seek $w_1, ..., w_s \in \mathbb{R}^D$ such that the projection of x onto the subspace spanned by $W = [w_1, ..., w_s]$ maximize the separability of the projections $W^T x_1, W^T x_2, ..., W_T x_N$. The vectors $w_1, ..., w_s$ are known as **linear discriminants**.*

*Or equivalently, the goal is to maximize projected mean distance over total within-class **projected scattering** via maximizing the following **Fisher criterion functions**:*

- $$J_1(w_1) = \frac{w_1^T S_B w_1}{w_1^T S_W w_1},$$
- $$J_2(w_2) = \frac{w_2^T S_B w_2}{w_2^T S_W w_2}, \text{ s.t. } Cov(w_1^T X, w_2^T X) = 0,$$
  where X is the data matrix consisting of $x_1, ..., x_N$.
- $$\begin{aligned} J_k(w_k) &= \frac{w_k^T S_B w_k}{w_k^T S_W w_k}, k = 3, ..., s, \\ s.t.\ &Cov(w_1^T X, w_k^T X) = 0, \\ &... \\ &Cov(w_{k-1}^T X, w_k^T X) = 0. \end{aligned}$$

Theorem 24.3.1 (discriminants for multi-dimensional LDA). *The discriminants $w_1, w_2, ..., w_s$ are given by*

$$S_W^{-1/2} u_1, S_W^{-1/2} u_2, ..., S_W^{-1/2} u_s$$

, where $u_1, u_2, ..., u_s$ are the eigenvectors of the matrix $S_W^{-1/2} S_B S_W^{-1/2}$ associated with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_s$.

Proof. (1) Note that

$$Cov(w_1^T X, w_k^T X) = w_1^T Cov(X, X) w_k = w_1 S_W w_k,$$

therefore, the set of maximization problems are the direct result general quadratic form optimization problem in Corollary 4.12.4.1. □

24.3.2.2 *Application in classification*

Lemma 24.3.3 (distance metric in the projected coordinates and variance preserving). *Let $w_1, w_2, ..., w_p$ be the* ***complete set*** *of Fisher linear discriminant and $x \in \mathbb{R}^D$ be a new input such that $w_i = S_W^{-1/2} e_i$, where e_i is the unit eigenvector of $S_W^{-1/2} S_B S_W^{-1/2}$. It follows that*

- *The distance of a sample x to its class mean m_i in the projected coordinates is* [a]

$$d_i(x) = \left\| y - m_i^{(y)} \right\|^2 = (x - m_1)^T S_W^{-1} (x - m_i)^T, i = 1, 2, ..., K.$$

where $y = W^T x, m_i^{(y)} = W^T m_i$.
- *Let $\mathcal{V}$ be the subspace spanned by $V = [w_1, ..., w_s]$. Then is the total class-variance in projected coordinates $V^T x_1, ..., V^T x_N$ is given by*

$$\Delta_V^2 = \sum_{i=1}^{K} (V^T m_i - V^T m)^T (V^T m_i - V^T m) = \lambda_1 + ... + \lambda_s.$$

- *Total class variance is related by*

$$\Delta^2 = \sum_{i=1}^{K} (m_i - m)^T (m_i - m) = \lambda_1 + ... + \lambda_p.$$

[a] Note that the distance to mean via projected coordinates preserves the within-class-scattering weighted distance.

Proof. (1) Consider class 1.

$$\begin{aligned} d_1 &= \left\| W^T (x - m_1) \right\|^2 \\ &= \sum_{i=1}^{p} \left| w_i^T (x - m_1) \right|^2 \\ &= \sum_{i=1}^{p} (x - m_1)^T S_W^{-1/2} e_i e_i^T S_W^{-1/2} (x - m_1) \\ &= (x - m_1)^T S_W^{-1/2} \sum_{i=1}^{p} (e_i e_i^T) S_W^{-1/2} (x - m_1) \\ &= (x - m_1)^T S_W^{-1/2} E^T E S_W^{-1/2} (x - m_1) \\ &= (x - m_1)^T S_W^{-1/2} S_W^{-1/2} (x - m_1) \\ &= S_W^{-1} (x - m_1) \end{aligned}$$

(2)(3)

$$\begin{aligned}
\Delta_V^2 &= \sum_{i=1}^{K}(V^T m_i - V^T m)^T (V^T m_i - V^T m) \\
&= \sum_{i=1}^{K}(m_i - m)^T V V^T (m_i - m) \\
&= Tr(\sum_{i=1}^{K}(m_i - m)^T V V^T (m_i - m)) \\
&= Tr(V V^T \sum_{i=1}^{K}(m_i - m)(m_i - m)^T) \\
&= Tr(V V^T S_B) \\
&= \lambda_1 + ... + \lambda_s
\end{aligned}$$

Note that a similar proof is in Theorem 14.2.2. □

Remark 24.3.4 (implications and dimensional reduction). Using top s discriminants, we can use low-dimensional representation $y_i = V^T x_i$, which preserves most of the class-variance.

24.3.3 Supervised dimensional reduction via Fisher LDA

Methodology 24.3.2. *The dimension reduction via LDA has the following steps:*

- *Compute the mean vectors for each classes*

$$m_i = \frac{1}{N_i} \sum_{j:y_j=i} x_j.$$

- *Compute the within-class scatter matrix*

$$S_W = \sum_{i=1}^{K} \sum_{j:y_j=i} (x_j - m_i)(x_j - m_i)^T.$$

- *Compute the between-class scatter matrix*

$$S_B = \sum_{i=1}^{K} N_i (m_i - m)(m_i - m)^T,$$

where m is the overall mean.
- *Solve the eigenvalue problem for the matrix $S_W^{-1}S_B$ such that*

$$S_W^{-1}S_B$$

- *Construct the new low dimensional feature space.*

Remark 24.3.5 (eigenvalue properties of the matrix $S_W^{-1}S_B$). Let S_B be the between-class scatter matrix of K classes. Then the matrix $S_W^{-1}S_B$ has at most $K-1$ non-zero eigenvalues.

This is because:

- S_B is the sum of K rank 1 matrices like $(\mu_i - \mu)(\mu_i - \mu)^T$; therefore S_B has at most K ranks. And from Lemma 4.4.1,

$$rank(S_W^{-1}S_B) \leq \min(rank(S_W^{-1}), rank(S_B)) = \min(D, K)$$

- S_B is a matrix that has been demeaned via projection, i.e.,

$$S_B = U(I - \frac{1}{K}J)U^T, U = [\mu_1, ..., \mu_K]$$

where UU^T has rank at most K. Therefore, S_B has rank at most $K-1$ (use Lemma 4.4.3).
- In conclusion,

$$rank(S_W^{-1}, S_B) = \min(D, K-1).$$

Remark 24.3.6 (PCA vs. LDA).

- PCA is not optimal for classification task since class label information is not used in searching principal components.
- PCA keeps dimensions with largest variance but might throw out dimensions with discriminant information (i.e., class label information).
- LDA tries to keep dimensions that contains the most variance between classes using class label information.

24.4 Separating hyperplane and Perceptron learning algorithm

24.4.1 Basic geometry of hyperplanes

Definition 24.4.1 (hyperplane and halfspace). *Let $a \in \mathbb{R}^d, \delta \in \mathbb{R}$, then the set $H = \{x \in \mathbb{R}^d : \langle a, x \rangle = \delta\}$ is called hyperplane. The set $H^+ = \{x \in \mathbb{R}^d : \langle a, x \rangle \geq \delta\}$ and $H^- = \{x \in \mathbb{R}^d : \langle a, x \rangle \leq \delta\}$ are halfspaces.*

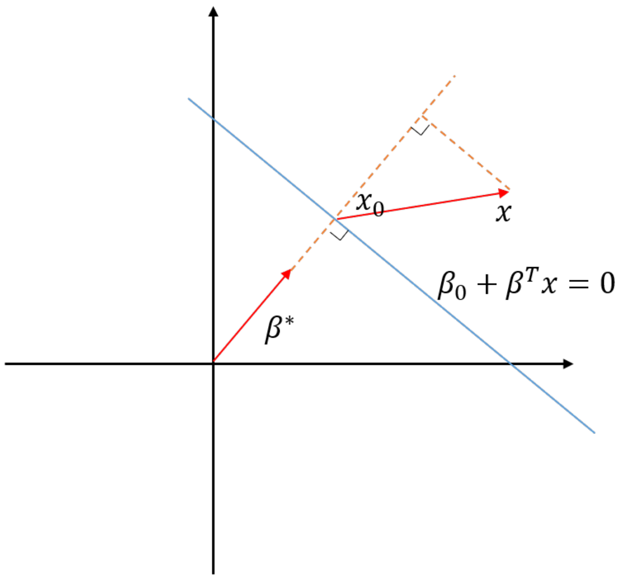


Figure 24.4.1: Scheme of a hyperplane.

Lemma 24.4.1 (signed distance to hyperplane). *Let $f(x) = \{x \in \mathbb{R}^d | w^T x + b = 0, w \in \mathbb{R}^d, b \in \mathbb{R}\}$ be a hyperplane in $\mathbb{R}^d$, given a point $x_1 \in \mathbb{R}^d$, its signed distance is given as*

$$(wx_1 + b)/\|w\|$$

Proof. Let $x_0 \in \mathbb{R}^d$ be a point on the hyperplane,i.e. $w^T x_0 + b = 0$, then the signed distance is given by projection formula as $w^T(x_1 - x_0)/\|w\| = (w^T x_1 - w^T x_0)/\|w\| = (wx_1 + b)/\|w\|$. □

24.4.2 The Perceptron learning algorithm

The Perceptron learning algorithm aims to minimize distance of misclassified examples to the hyperplane. For a misclassified example that has $y_i = 1, \beta^T x_i + \beta_0 < 0, \hat{y}_i = -1$, its distance is the hyperplane is proportional to $-y_i f(x_i) = -y_i(\beta^T x_i + \beta_0)$. The algorithm will stop if all examples are correctly classified (in the linearly separable case) or reach some criterion.

We summarize these key aspect in the following Lemma.

Lemma 24.4.2. *Consider a binary classification problem with examples* $D = \{(x_1, y_1), ..., (x_N, y_N)\}, x_i \in \mathbb{R}^P, y_i \in \{-1, 1\}$. *The loss function associated with the Perceptron Learning algorithm is given by*

$$L(\beta, \beta_0) = \sum_{i \in \mathcal{M}} -y_i(\beta^T x_i + \beta_0),$$

where $\mathcal{M}$ *is the index set of misclassified examples.*

The gradients are

$$\partial \frac{L(\beta, \beta_0)}{\partial \beta} = -\sum_{i \in \mathcal{M}} y_i x_i$$
$$\partial \frac{L(\beta, \beta_0)}{\partial \beta_0} = -\sum_{i \in \mathcal{M}} y_i$$

Algorithm 34: Perceptron learning algorithm

```
Input: training data set $D = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}, y_i \in \{-1, 1\}$,
       threshold $\epsilon$, learning rate $\alpha$ .
1 Initialize coefficients $\beta, \beta_0$.
2 repeat
3     Predict all labels via $\hat{y}_i = I(f(x_i) > 0)$.
4     foreach $i \in \mathcal{M}$ do
5         $\beta = \beta + \alpha y_i x_i, \beta_0 = \beta_0 - \alpha y_i$.
6     end
7 until loss reduction < $\epsilon$;
Output: the hyperplane coefficients $\beta, \beta_0$
```

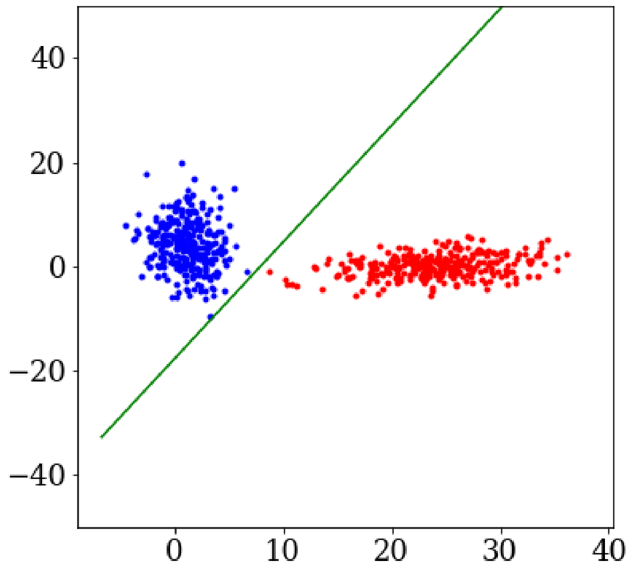


Figure 24.4.2: Binary classification using the Perceptron learning algorithm. The hyperplane learned separates the two clusters.

We consider a toy binary classification example [Figure 24.4.2]. The Perceptron learning algorithm learns a hyperplane that separates the two clusters. Compared with SVM that we will cover in section 24.5, the hyperplane is not fully optimized since it is quite close to some data points and thus can have large error when used to classify new examples.

24.5 Support vector machine classifier

24.5.1 Motivation and formulation

Let the binary classification training data consist of N pairs $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$, with $x_i \in \mathbb{R}^p, y_i \in \{1, -1\}$. Assuming training data are linearly separable.[1]

The SVM classification problem consist of the following two steps:

- Define a hyperplane by $x^T\beta + \beta_0 = 0$, which gives a classification rule of
$$y = sign(x^T\beta + \beta_0)$$
for training and future data.
- Optimize hyperplane parameter such that it has biggest distance from the closest x to the hyperplane, as showed in Figure 24.5.1.

Note that the hyperplane defined by $x^T\beta + \beta_0 = 0$ is invariant to the scaling of β, β_0, i.e., $kx^T\beta + k\beta_0 = 0$ represents the same hyperplane. We set the hyperplane sitting near the label $y = 1$ with the function $x^T\beta + \beta_0 = 1$ and hyperplane sitting near the label $y = -1$ with the function $x^T\beta + \beta_0 = -1$. The distance of the two margin hyperplanes are $\frac{2}{\|\beta\|}$.

The optimization problem for SVM is formulated as

$$\min_{\beta,\beta_0} \|\beta\|^2$$

subject to
$$y_i(x_i^T\beta + \beta_0) \geq 1, i = 1, 2, ..., N.$$

We have following interpretation on the optimization formulation.

- The constraint requires that every point x must be on the correct side (from the multiplier y_i).
- Maximizing the margin distance $\frac{2}{\|\beta\|^2}$ is achieved by minimizing $\|\beta\|^2$.

1 That is, there exist a hyperplane that can correctly classify all training sample.

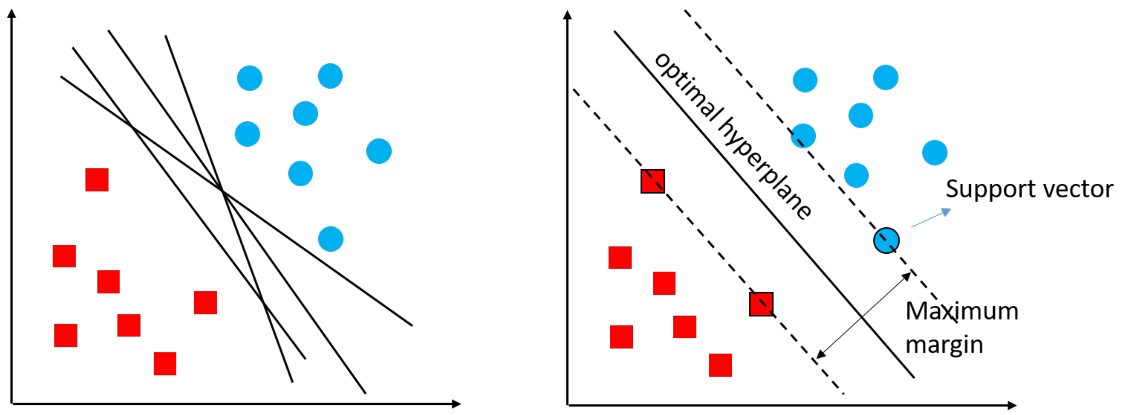


Figure 24.5.1: Left: existence of multiple separating hyperplanes in 2D binary classification problem. Right: hyperplanes with maximum margin.

24.5.2 Optimality condition and dual form

Based on the KKT theory developed in previous chapter [Theorem 7.3.4], the optimality condition for this quadratic optimization is given by the following theorem.

Theorem 24.5.1 (KKT optimality condition for primal form). *The Lagrangian function is given as*

$$L(w,b,\lambda) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N}\lambda_i(y_i(x_i^T w + b) - 1),$$

with KKT optimality condition as:

$$\lambda_i \geq 0, i = 1, ..., N$$

$$\frac{\partial L}{\partial b} = 0 \implies 0 = \sum_{i=1}^{N}\lambda_i y_i$$

$$\frac{\partial L}{\partial w} = 0 \implies w = \sum_{i=1}^{N}\lambda_i y_i x_i \text{ (\textbf{primal-dual relationship})}$$

Note 24.5.1 (conditions for existence of solutions). Note that only when data are linearly separable, there are feasible solutions.

Because of the optimization problem is a convex optimization, we can also formulate its dual optimization form, where optimize over Lagrange multipliers as dual variables. Dual form SVM have multiple advantages.

- In general, solving the primal optimization (optimizing over β, β_0) has a computational complexity of $O(p^3)$. If $p \gg N$, solving dual optimization can reduce the cost to $O(N^3)$. However, if $N \gg p$, solving primal optimization is preferred. Recently, a popular method is using stochastic gradient decent used on an alternative loss function [Lemma 24.5.2], rather than directly seeking solutions satisfying KKT.
- Even when $p < N$, dual form SVM can be extended to kernel SVM [subsection 24.5.4].

Theorem 24.5.2 (dual form optimization problem). *The dual form of the optimization problem is*

$$\max_{\lambda} \tilde{f}(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i\lambda_j y_i y_j x_i \cdot x_j$$

with constraints:

$$\lambda_i \geq 0, i = 1, ..., N$$

$$0 = \sum_{i=1}^{N} \lambda_i y_i$$

Proof. Based on the dual problem definition [Definition 9.4.3], we have

$$\tilde{f}(\lambda) = \inf_{\beta,\beta_0} L(\beta, \beta_0, \lambda).$$

Set first derivatives to zeros, we have

$$\frac{\partial L}{\partial \beta_0} = 0 \implies 0 = \sum_{i=1}^{N} \lambda_i y_i$$

$$\frac{\partial L}{\partial \beta} = 0 \implies \beta = \sum_{i=1}^{N} \lambda_i y_i x_i$$

Then

$$\begin{aligned}\tilde{f}(\lambda) &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i\lambda_j y_i y_j x_i \cdot x_j - \sum_{i=1}^{N} \lambda_i y_i (x_i^T w) + \sum_{i=1}^{N} \lambda_i y_i b + \sum_{i=1}^{N} \lambda_i \\ &= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i\lambda_j y_i y_j x_i \cdot x_j + \sum_{i=1}^{N} \lambda_i\end{aligned}$$

The dual problem constraints are the constraints on the Lagrangian multipliers. □

Remark 24.5.1 (prediction and support vectors).

- If we replace plug $\beta = \sum_{i=1}^N \lambda_i y_i x_i$ into $y = \beta^T x + \beta_0$, we get

$$y = \sum_{i=1}^N \lambda_i y_i x_i^T x + \beta_0.$$

- **Support vectors are x_i with $\lambda_i > 0$.** Since our support vectors are used in the prediction, we only need to store support vectors for real-world applications to save memory.

Remark 24.5.2 (Solving dual optimization problem). Although dual optimization methods can be solved via general quadratic optimization algorithms, a specialized sequential minimal optimization method has been developed to accelerate the solution process[6].

24.5.3 Soft margin SVM

24.5.3.1 *Basics*

The SVM, also known as **hard margin SVM**, we introduced so far has several limitations:

- It only applies to linearly separable data; otherwise, the optimization problem has no feasible solution.
- Even for linearly separable data, the separation margin can move significantly if some support vector moves (i.e., not robust to outliers).

To cope with these limitation, in this section, we introduce soft margin SVM. The core idea is to introduce a penalty to the data points sitting on the wrong of the margin. This modification extends SVM to data that is not linearly data and enhance its robustness to outliers.

Definition 24.5.1 (soft margin SVM optimization formulation). *The **soft margin SVM classification** is formulated as minimization as*

$$\min_{\beta,\beta_0,\eta} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^n \eta_i$$

under the constraints:

$$\begin{gathered} y_i(x_i^T\beta + \beta_0) \geq 1 - \eta_i, i = 1, 2, ..., N \\ \eta_i \geq 0, i = 1, ..., N \end{gathered}$$

where the C is the regulation parameter.

We can interpret the formulation in the following way.

- When $C \to \infty$, the soft margin optimization problem will converge to the hard margin SVM.
- C is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the training error. Small C tends to emphasize the margin and ignore the outliers in the training data, while large C may tend to overfit the training data[Figure 24.5.2].

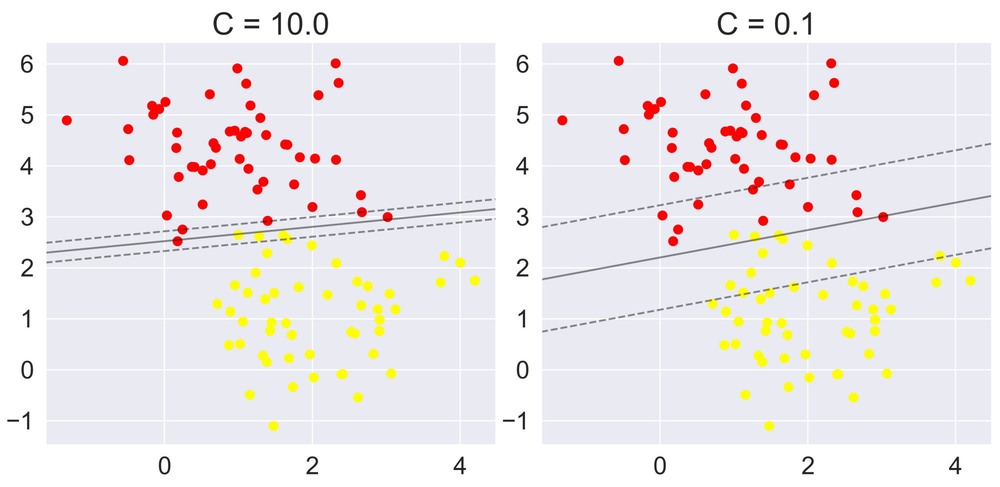


Figure 24.5.2: SVM classification with different regularization strength. Small C tends to emphasize the margin and ignore the outliers in the training data, while large C may tend to overfit the training data.

24.5.3.2 *Optimality condition for soft margin SVM*

Similarly, we can use the KKT theory developed in previous chapter [Theorem 7.3.4] to develop the optimality condition for this quadratic optimization.

Lemma 24.5.1 (KKT condition for primal form). *The Lagrangian function is given as*

$$L(\beta, \beta_0, \eta, \lambda, \mu) = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\eta_i - \sum_{i=1}^{N}\lambda_i(y_i(x_i^T\beta + \beta_0) - 1) - \sum_{i=1}^{N}\mu_i\eta_i,$$

with KKT condition as:

$$\begin{aligned}
\lambda_i &\geq 0, i = 1, ..., N \ (dual\ feasible) \\
\mu_i &\geq 0, i = 1, ..., N \ (dual\ feasible) \\
y_i(x_i^T \beta + \beta_0) &\geq 1 - \eta_i, i = 1, 2, ..., N \ (primal\ feasible) \\
\eta_i &\geq 0, i = 1, ..., N \ (primal\ feasible) \\
\lambda_i(y_i(x_i^T \beta + \beta_0) - 1 + \eta_i) &= 0, i = 1, 2, ..., N \ (complementrary\ slackness) \\
\frac{\partial L}{\partial \beta_0} = 0 &\implies 0 = \sum_{i=1}^N \lambda_i y_i \\
\frac{\partial L}{\partial \beta} = 0 &\implies \beta = \sum_{i=1}^N \lambda_i y_i x_i \\
\frac{\partial L}{\partial \eta_i} = 0 &\implies 0 = C - \lambda_i - \mu_i, i = 1, 2, ..., N.
\end{aligned}$$

We can also extend this formulation to the dual form SVM. Recall that dual form has the following advantages,

- The dual form involves optimizing over N dual variables, whereas the primal form involves optimization over p variables; the dual form is beneficial when $p \gg N$. However, if $N \gg p$, solving primal optimization is preferred. Recently, a popular method is using stochastic gradient decent used on an alternative loss function [Lemma 24.5.2], rather than directly seeking solutions satisfying KKT.
- The dual form only requires the inner product between x_i and x_j, which can be extended to other kernels using kernel trick.

Theorem 24.5.3 (dual form soft margin SVM optimization problem). *The dual form of the optimization problem is*

$$\max_{\lambda} \tilde{L}(\lambda) = \sum_{i=1}^N \lambda_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j y_i y_j x_i^T x_j$$

with constraints:

$$\begin{aligned}
C \geq \lambda_i &\geq 0, i = 1, ..., N \\
0 &= \sum_{i=1}^N \lambda_i y_i
\end{aligned}$$

Proof. Based on the dual problem definition [Definition 9.4.3], we have

$$\tilde{f}(\lambda, \mu) = \inf_{\beta, \beta_0} L(\beta, \beta_0, \eta, \lambda, \mu).$$

Set first derivatives to zeros, we have

$$\begin{aligned}
\frac{\partial L}{\partial \beta_0} &= 0 \implies 0 = \sum_{i=1}^{N} \lambda_i y_i \\
\frac{\partial L}{\partial \beta} &= 0 \implies w = \sum_{i=1}^{N} \lambda_i y_i x_i \\
\frac{\partial L}{\partial \eta_i} &= 0 \implies 0 = C - \lambda_i - \mu_i, i = 1, 2, ..., N.
\end{aligned}$$

Then

$$\begin{aligned}
\tilde{L}(\lambda, \mu) &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i\lambda_j y_i y_j x_i \cdot x_j - \sum_{i=1}^{N} \lambda_i y_i (x_i^T w) + \sum_{i=1}^{N} \lambda_i y_i b + \sum_{i=1}^{N} \lambda_i + \sum_{i=1}^{N} \eta_i (C - \lambda_i - \mu_i) \\
&= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i\lambda_j y_i y_j x_i \cdot x_j + \sum_{i=1}^{N} \lambda_i
\end{aligned}$$

The dual problem constraints are the constraints on the Lagrangian multipliers. In summary, we have

$$\tilde{L}(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i\lambda_j y_i y_j x_i^T x_j$$

with constraints:

$$\begin{aligned}
\lambda_i &\geq 0, i = 1, ..., N \\
\mu_i &\geq 0, i = 1, ..., N \\
C &= \lambda_i + \mu_i, i = 1, ..., N \\
0 &= \sum_{i=1}^{N} \lambda_i y_i.
\end{aligned}$$

Note that the first three conditions can be simplified to

$$0 \leq \lambda_i \leq C, i = 1, 2..., N.$$

□

24.5.3.3 *Algorithm*

In the following, we summarize the soft margin SVM, which consists of two major steps:

- Solve the dual form optimization problem.

- Construct the classifier based on dual solutions.

Note that data normalization or re-scaling data is critical for SVM since SVM involves minimizing distance. We need to make sure that for each dimension of the feature, the values are scaled to lie roughly the same range.

Algorithm 35: Soft margin SVM algorithm

```
Input: Training data $(x_1, y_1), ..., (x_N, y_N)$ where $x_i \in X$, $y_i \in Y = \{-1, 1\}$
1 Initialize regularizer parameter $C > 0$ and construct the dual form convex
    optimization problem:
      $$\min_{\lambda} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{N}\lambda_i$$
      $$s.t. \sum_{i=1}^{N}\lambda_i y_i = 0$$
      $$0 \leq \lambda_i \leq C, i = 1, 2, ..., N$$
2 Solve the optimization problem $\lambda^* = (\lambda_1^*, ..., \lambda_N^*)$.
3 Compute $w^* = \sum_{i=1}^{N}\lambda_i^* y_i x_i$ based on KKT condition.
4 Select any $\lambda_i^*$ satisfying $0 < \lambda_i^* < C$, and compute
      $$b^* = y_i - \sum_{i=1}^{N} y_i \lambda_i^* (x_i \cdot x_j).$$
5 Compute the separating hyperplane via
      $$x^T\beta^* + \beta_0^* = 0.$$
6 Get the decision function
      $$f(x) = sign(x^T\beta^* + \beta_0^*)$$
Output: $f(x)$
```

Remark 24.5.3 (the value of β^*). From the KKT condition that if λ_i is not binding to the constraints 0 and C, i.e., $0 < \lambda_i < C$, then the constraint $y_i(x_i^T\beta + \beta_0^*) = 1$ is satisfied.

Therefore, we have

$$\begin{aligned}
y_i(x_i^T\beta + \beta_0) &= 1\\
\beta_0^* &= \frac{1}{y_i} - x_i^T\beta^*\\
\beta_0^* &= y_i - x_i^T\beta^*\\
\beta_0^* = y_i - \sum_{i=1}^{N} y_i\lambda_i^*(x_i \cdot x_j)
\end{aligned}$$

Note that β_0^* is not necessarily unique. In reality, we can take the average value of all qualified β_0^*.

24.5.4 SVM with kernels

If in the original feature space $\mathcal{X}$, there are significant overlap between classes, we might like to perform nonlinear transformation $\phi(x)$ on the original feature space. One way to perform such implicit nonlinear transformation (i.e., without explicitly knowing $\phi(x)$) is to use kernels [section 22.6]. If we are given a kernel function k which satisfies $k\left(x_i, x_j\right) = \phi\left(x_i\right) \cdot \phi\left(x_j\right)$, then we can use kernel SVM as we introduce in the following.

The dual form of soft margin SVM [Theorem 24.5.3] only involves the inner product $x_i \cdot x_j$. Therefore, by replacing the inner product $x_i \cdot x_j$ by the kernel $k(x_i, x_j)$, the dual form of the optimization problem with kernel is given by

$$\tilde{L}(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i\lambda_j y_i y_j k(x_i, x_j)$$

with constraints:

$$\begin{aligned}
C \geq \lambda_i \geq 0, i = 1, ..., N\\
0 = \sum_{i=1}^{N} \lambda_i y_i
\end{aligned}$$

Once we solve the model coefficient β, β_0, the classifier decision function is given by

$$f(x) = \sum_{i=1}^{N} \lambda_i y_i k(x_i, x) + \beta_0.$$

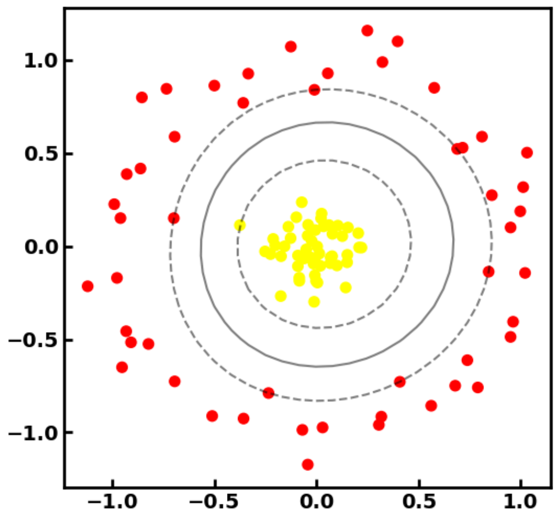


Figure 24.5.3: SVM classification using Gaussian kernel. The original problem cannot be separated by linear kernel.

24.5.5 A unified perspective from loss functions

In this section, we show that Logistic regression, Perceptron learning, and SVM for binary classification problem are unified under the same optimization problem given by

$$\min_{\beta_0,\beta} \sum_{i=1}^{N} L(y_i, f(x)) + \lambda \beta^T \beta,$$

where L is the loss function taking different forms, respectively, and $f(x) = \beta^T x + \beta_0$.

The formulation of SVM into this framework relies on the following Lemma.

Lemma 24.5.2 (equivalent form of soft margin SVM). *Let the binary classification training data consist of N pairs* $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$, *with* $x_i \in \mathbb{R}^p, y_i \in \{1, -1\}$. *The* ***soft margin SVM classification*** *optimization*

$$\min_{\beta,\beta_0,\eta} \|\beta\|^2 + C \sum_{i=1}^{N} \eta_i$$

under the constraints:

$$y_i(x_i^T\beta+\beta_0)\geq 1-\eta_i, i=1,2,...,N$$
$$\eta_i\geq 0, i=1,...,N$$

where the C is the regulation parameter, is equivalent to

$$\min_{\beta_0,\beta}\sum_{i=1}^N L(y_i,f(x_i))+\lambda\|\beta\|^2,$$

where

$$L(y_i,f(x_i))=\max(0,1-y_if(x_i)).$$

Proof. Because

$$y_i(x_i^T\beta+\beta_0)\geq 1-\eta_i,\eta_i\geq 0,$$

we have

$$\eta_i\geq 1-y_i(x_i^T\beta+\beta_0),\eta_i\geq 0,$$

or equivalently

$$\eta_i=\max(0,1-y_if(x_i)).$$

Further let $C=1/\lambda$, we will have the final result. □

In the original logistic regression model, we assume the target labels are $\{0,1\}$ in the binary cross-entropy loss formulation. We need to re-formulate the cross-entropy loss with labels in $\{-1,1\}$. Denote $\sigma(x)$ as the logit function given by

$$\sigma(x)=\frac{1}{1+\exp(-x)}.$$

Then

$$P(y=1|x)=\sigma(f(x)),P(y=0|x)=1-\sigma(f(x))=\sigma(-f(x)).$$

The cross-entropy loss function of logistic regression can be written by

$$\begin{aligned}L_{LR}&=-I(y=1)\log P(y=1|x)-I(y=-1)\log P(y=0|x)\\&=-I(y=1)\log\sigma(f(x))-I(y=-1)\log\sigma(-f(x))\\&=-\log\sigma(yf(x))\\&=\log(1+\exp(-yf(x)))\end{aligned}$$

As a comparison, the loss functions in Perceptron learning and SVM are

$$L_P=\max(0,-yf(x)),$$

and

$$L_{Hinge} = \max(0, 1 - yf(x)).$$

These loss functions are plotted in Figure 24.5.4.

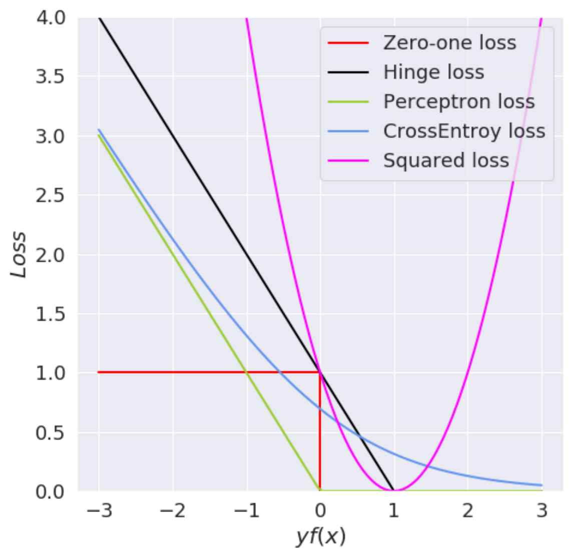


Figure 24.5.4: Comparison of classification loss functions.

Remark 24.5.4 (Comparison between logistic regression and SVM)**.**

- SVM try to maximize the margin between the closest support vectors while logistic regression maximize the posterior class probability.
- Logistic regression is more sensitive to outliers than SVM because its cost function diverges faster than the Hinge loss of SVM. Also more data points will contribute to logistic regression results while data lying on the correct margin side will not contribute to the model estimation in SVM.
- Logistic regression produces probabilistic values while SVM produces 1 or 0.
- SVM requires data normalization while logistic regression can work with original data.
- The loss function in SVM [Lemma 24.5.2] contains regularization, while logistic loss does not.

24.6 Note on bibliography

Linear classification model has been covered on text books, for example [7][1][4][8]. For an excellent review on Gaussian LDA and QDA, see [9].

BIBLIOGRAPHY

1. Murphy, K. P. *Machine learning: a probabilistic perspective* (MIT press, 2012).
2. Rendle, S. *Factorization machines* in *2010 IEEE International Conference on Data Mining* (2010), 995–1000.
3. Friedman, J., Hastie, T. & Tibshirani, R. *The elements of statistical learning (2017 corrected version)* (Springer series in statistics Springer, Berlin, 2007).
4. Bishop, C. M. Pattern Recognition and Machine Learning (2006).
5. Johnson, R. & Wichern, D. *Applied Multivariate Statistical Analysis* ISBN: 9780131877153 (Pearson Prentice Hall, 2007).
6. Platt, J. Sequential minimal optimization: A fast algorithm for training support vector machines (1998).
7. Li, H. *Statistical Learning Methods* (Tsinghua University, 2011).
8. Zhuo, Z. *Machine Learning* (Tsinghua University, 2016).
9. Ghojogh, B. & Crowley, M. Linear and Quadratic Discriminant Analysis: Tutorial. *arXiv preprint arXiv:1906.02590* (2019).