Linear Discrimination

Prof. Ziping Zhao

School of Information Science and Technology
ShanghaiTech University, Shanghai, China

CS182: Introduction to Machine Learning (Fall 2021)
http://cs182.sist.shanghaitech.edu.cn

Outline

Outline

Likelihood-Based vs. Discriminant-Based Classification

- Classification based on a set of discriminant functions $g_i(x)$, $i = 1, \ldots, K$:

$$\text{Choose } C_i \text{ if } g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})$$

- Likelihood-based classification:
  - Assume a parametric, semiparametric, or nonparametric model for the class-conditional probability densities $p(\mathbf{x} \mid C_i)$.
  - Estimate the prior probabilities $P(C_i)$ and the class likelihoods $p(\mathbf{x} \mid C_i)$ from data.
  - Apply Bayes' rule to compute the posterior probabilities $P(C_i \mid \mathbf{x})$.
  - Perform optimal classification based on $P(C_i \mid \mathbf{x})$, or equivalently based on discriminant functions $g_i(\mathbf{x})$ such as $g_i(x) = \log P(C_i \mid \mathbf{x})$.
- Discriminant-based classification:
  - Assume a model directly for the discriminant functions, bypassing the estimation of $p(\mathbf{x} \mid C_i)$ or $P(C_i \mid \mathbf{x})$ from data.
  - Perform optimal classification based on the discriminant functions $g_i(\mathbf{x})$.

Likelihood-Based vs. Discriminant-Based Classification (2)

- Main difference: the likelihood-based approach makes an assumption on the form of the densities (e.g., whether they are Gaussian, or whether the inputs are correlated, etc.), but the discriminant-based approach makes an assumption on the form of the discriminants.

Discriminant Functions

- Define a model for the discriminant functions:

$$g_i(\mathbf{x} \mid \Phi_i)$$

  which are explicitly parameterized with a set of model parameters Φ_i.
- In discriminant-based approach, we make an assumption on the form of the boundaries separating classes.
- Learning is the optimization of Φ_i to maximize the quality of the separation, that is, the classification accuracy on a given labeled training set.
- Unlike the likelihood-based approach which performs density estimation separately for each class, the discriminant-based approach typically estimates Φ_i for all classes simultaneously to find the decision boundaries between classes.
- Estimating the class boundaries (discriminants) is usually easier than estimating the class densities. E.g., this is true when the discriminant can be approximated by a simple function.

Linear Discriminant Functions

- Linear discriminant functions:

$$g_i(\mathbf{x} \mid \mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T \mathbf{x} + w_{i0} = \sum_{j=1}^{d} w_{ij} x_j + w_{i0}$$

  which are linear in $\mathbf{x}$.
- Advantages:
  - Simplicity: $O(d)$ time and space complexity.
  - Understandability: final output is a weighted sum of attributes; magnitude and sign of weights have clear physical meaning.
  - Accuracy: model is quite accurate if some assumptions are satisfied, e.g., Gaussian densities for classes with shared covariance matrix.
- We should always use the linear discriminant before trying a more complicated model to make sure that the additional complexity is justified.

Generalizing the Linear Models

- When a linear model is not flexible enough, we can use the quadratic discriminant function
$$g_i(\mathbf{x} \mid \mathbf{W}_i, \mathbf{w}_i, w_{i0}) = \mathbf{x}^T \mathbf{W}_i \mathbf{x} + \mathbf{w}_i^T \mathbf{x} + w_{i0}$$
- An equivalent way is to preprocess the input by adding higher-order terms (or called product terms).
- Example: with two inputs x_1 and x_2, we define new variables
$$z_1 = x_1,\ z_2 = x_2\ z_3 = x_1^2,\ z_4 = x_2^2,\ z_5 = x_1 x_2$$
and take $\mathbf{z} = (z_1, z_2, z_3, z_4, z_5)^T$ as the new input. The linear function defined in the new $\mathbf{z}$-space corresponds to a nonlinear function in the original $\mathbf{x}$-space.
- Compared with defining a nonlinear function (discriminant or regression) in the original input space, defining a linear function in a nonlinearly transformed new space (called a generalized linear model) does not increase the number of parameters that need to be estimated significantly.

Basis Functions

- More generally, the inputs $\mathbf{x}$ are (nonlinearly) transformed into basis functions $\phi_{ij}(\mathbf{x})$ which are linearly combined to define the discriminant functions:

$$g_i(\mathbf{x}) = \sum_{j=1}^{k} w_j \phi_{ij}(\mathbf{x})$$

- Higher-order terms mentioned before are only one set of basis functions.
- Other examples of basis functions:
  - $\sin(x_1)$
  - $\exp(-(x_1 - m)^2/c)$
  - $\exp(-\|\mathbf{x} - \mathbf{m}\|^2/c)$
  - $\log(x_1)$
  - $\mathbf{1}(x_1 > c)$
  - $\mathbf{1}(ax_1 + bx_2 > c)$

Outline

Geometric View: Two Classes

- Discriminant function:

$$\begin{aligned} g(\mathbf{x}) &= g_1(\mathbf{x}) - g_2(\mathbf{x}) \\ &= (\mathbf{w}_1^T \mathbf{x} + w_{10}) - (\mathbf{w}_2^T \mathbf{x} + w_{20}) \\ &= (\mathbf{w}_1 - \mathbf{w}_2)^T \mathbf{x} + (w_{10} - w_{20}) \\ &= \mathbf{w}^T \mathbf{x} + w_0 \end{aligned}$$

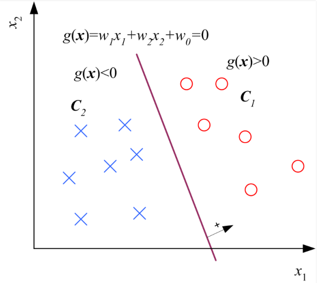
  where $\mathbf{w}$ is the weight vector and w_0 is the threshold.

- Optimal decision rule:

$$\text{Choose } \begin{cases} C_1 & \text{if } g(\mathbf{x}) > 0 \\ C_2 & \text{otherwise} \end{cases}$$

Hyperplane

- The discriminant function defines a hyperplane ($g(\mathbf{x}) = 0$) that divides the input space into 2 half-spaces:
  - Decision region $\mathcal{R}_1$ for C_1 ($g(\mathbf{x}) > 0$, i.e., positive side of the hyperplane)
  - Decision region $\mathcal{R}_2$ for C_2 ($g(\mathbf{x}) < 0$, i.e., negative side of the hyperplane)



- When $\mathbf{x} = \mathbf{0}$ (i.e., the origin), $g(\mathbf{x}) = w_0$. If $w_0 > 0$, the origin is on the positive side, and if $w_0 < 0$, the origin is on the negative side, and if $w_0 = 0$, the hyperplane passes through the origin.

Geometric Interpretation

- Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two points on the hyperplane, i.e., $g(\mathbf{x}_1) = g(\mathbf{x}_2) = 0$. So

$$\mathbf{w}^T\mathbf{x}_1 + w_0 = \mathbf{w}^T\mathbf{x}_2 + w_0$$
$$\mathbf{w}^T(\mathbf{x}_1 - \mathbf{x}_2) = 0$$

  showing that $\mathbf{w}$ is normal (orthogonal) to any vector $(\mathbf{x}_1 - \mathbf{x}_2)$ lying on the hyperplane.

- Let us express any point $\mathbf{x}$ as

$$\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$$

  where

  $\mathbf{x}_p$: normal projection of $\mathbf{x}$ onto the hyperplane

  r : distance from $\mathbf{x}$ to the hyperplane ($r > / < 0$: $\mathbf{x}$ is on the positive/negative side)

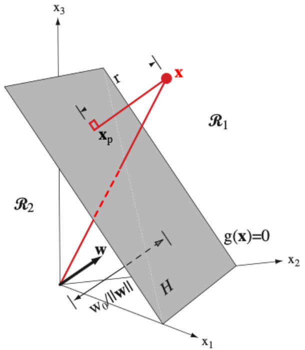
Geometric Interpretation (2)

- Calculation of r (note $g(\mathbf{x}_p) = 0$):

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = \mathbf{w}^T\mathbf{x}_p + r\frac{\mathbf{w}^T\mathbf{w}}{\|\mathbf{w}\|} + w_0 = g(\mathbf{x}_p) + r\|\mathbf{w}\| = r\|\mathbf{w}\|$$

  So we have

$$r = \frac{g(\mathbf{x})}{\|\mathbf{w}\|} \qquad \text{(sign of } r = \text{sign of } g(\mathbf{x}))$$



Geometric Interpretation (3)

- When $\mathbf{x} = \mathbf{0}$, the distance from origin to hyperplane is $\frac{g(\mathbf{0})}{\|\mathbf{w}\|} = \frac{w_0}{\|\mathbf{w}\|}$.
- Alternative view: If $\mathbf{x}$ is a point on the hyperplane, then $g(\mathbf{x}) = 0$. So

$$
\begin{aligned}
\mathbf{w}^T\mathbf{x} + w_0 &= 0 \\
\left(\frac{\mathbf{w}}{\|\mathbf{w}\|}\right)^T \mathbf{x} + \frac{w_0}{\|\mathbf{w}\|} &= 0 \\
\left|\left(\frac{\mathbf{w}}{\|\mathbf{w}\|}\right)^T \mathbf{x}\right| &= \frac{w_0}{\|\mathbf{w}\|}
\end{aligned}
$$

- The orientation of the hyperplane is determined by $\mathbf{w}$ and its distance from the origin is determined by w_0 and $\mathbf{w}$.

Geometric View: Multiple Classes

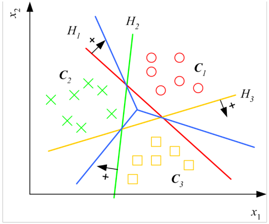
- K discriminant functions:

$$g_i(\mathbf{x} \mid \mathbf{w}_i, w_{i0}) = \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

- Linearly separable classes:

$$g_i(\mathbf{x} \mid \mathbf{w}_i, w_{i0}) = \begin{cases} > 0 & \text{if } \mathbf{x} \in C_i \\ \leq 0 & \text{otherwise} \end{cases}$$

For each class C_i, there exists a hyperplane H_i such that all $\mathbf{x} \in C_i$ lie on the positive side and all other $\mathbf{x} \in C_j$, $j \neq i$ lie on the negative side.



Linear Classifier

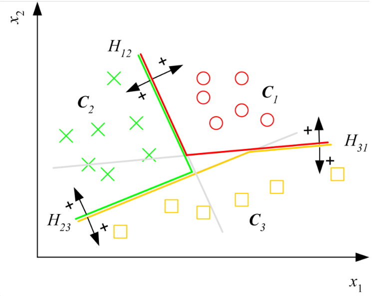
- During testing, given $\mathbf{x}$, ideally, we should have only one $g_j(\mathbf{x})$, $j = 1, \ldots, K$ greater than 0.
- However, it is possible for multiple or no $g_i(\mathbf{x})$ to be > 0. These may be taken as reject cases, but the usual approach is to assign $\mathbf{x}$ to the class having the highest discriminant.
- Decision rule for any test case $\mathbf{x}$:

$$\text{Choose } C_i \text{ if } g_i(\mathbf{x}) = \max_{j=1}^{K} g_j(\mathbf{x})$$

- Geometrically a linear classifier partitions the feature space into K convex decision regions $\mathcal{R}_i$.

Pairwise Separation

- If the classes are not linearly separable, one approach is to divide it into a set of linear problems and linear discriminants can be used to separate the classes.
- One possibility is to perform pairwise separation of classes by considering one pair of distinct classes at a time.
- $K(K-1)/2$ linear discriminants are used.
- It is easier for the classes to be pairwise linearly separable than linearly separable.



Pairwise Separation (2)

- Discriminant function for classes i and j ($i, j = 1, \ldots, K$ and $j \neq i$):

$$g_{ij}(\mathbf{x} \mid \mathbf{w}_{ij}, \mathbf{w}_{ij0}) = \mathbf{w}_{ij}^T \mathbf{x} + w_{ij0} = \begin{cases} > 0 & \text{if } \mathbf{x} \in C_i \\ \leq 0 & \text{if } \mathbf{x} \in C_j \\ \text{don't care} & \text{if } \mathbf{x} \in C_k, k \neq i, k \neq j \end{cases}$$

  - if $\mathbf{x}^{(\ell)} \in C_k$ where $k \neq i$, $k \neq j$, then $\mathbf{x}^{(\ell)}$ is not used during training of $g_{ij}(\mathbf{x})$.
- Decision rule for any test case $\mathbf{x}$:

$$\text{Choose } C_i \text{ if } \forall j \neq i, g_{ij}(\mathbf{x}) > 0$$

- Sometimes we may not be able to find such a class C_i. If we do not want to reject such cases, a relaxed decision rule can be defined based on a new set of discriminant functions:

$$g_i(\mathbf{x}) = \sum_{j \neq i} g_{ij}(\mathbf{x})$$

Outline

Linear Parametric Discrimination Revisited

- Recall that if the class-conditional densities $p(\mathbf{x} \mid C_i)$ are Gaussian sharing a common covariance matrix $\boldsymbol{\Sigma}$, the discriminant functions are linear:

$$g_i(\mathbf{x}) = \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

where

$$\mathbf{w}_i = \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i$$
$$w_{i0} = -\frac{1}{2} \boldsymbol{\mu}_i^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_i + \log P(C_i)$$

- Given a sample $\mathcal{X}$, we find the ML estimates for $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}$, denoted by $\mathbf{m}_i$ and $\mathbf{S}$, and plug them into the discriminant functions.

Two-Class Example

- Let

$$P(C_1 \mid \mathbf{x}) = y \qquad P(C_2 \mid \mathbf{x}) = 1 - y$$

- Classification rule:

$$\text{Choose } \begin{cases} C_1 & \text{if } y > 0.5 \\ C_2 & \text{otherwise} \end{cases}$$

- Equivalent tests for classification rule:

$$\frac{y}{1-y} > 1 \qquad \textit{or} \qquad \log \frac{y}{1-y} > 0$$

  where $y/(1-y)$ is called the odds (odds ratio) of y and $\log[y/(1-y)]$ is called the log odds of y or logit transformation/function of y, written as $\text{logit}(y)$.
- The $\text{logit}(\cdot)$ is a type of function that maps probability values from $(0, 1)$ to real numbers in $(-\infty, +\infty)$.

Logit Function

- In the case of two normal classes sharing a common covariance matrix, the logit function:

$$
\begin{aligned}
\text{logit}(P(C_1 \mid \mathbf{x})) &= \log \frac{P(C_1 \mid \mathbf{x})}{1 - P(C_1 \mid \mathbf{x})} = \log \frac{P(C_1 \mid \mathbf{x})}{P(C_2 \mid \mathbf{x})} \\
&= \log \frac{p(\mathbf{x} \mid C_1)}{p(\mathbf{x} \mid C_2)} + \log \frac{p(C_1)}{p(C_2)} \\
&= \log \frac{(2\pi)^{-\frac{d}{2}} |\mathbf{\Sigma}|^{-\frac{1}{2}} \exp[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_1)^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_1)]}{(2\pi)^{-\frac{d}{2}} |\mathbf{\Sigma}|^{-\frac{1}{2}} \exp[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_2)^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_2)]} + \log \frac{p(C_1)}{p(C_2)} \\
&= \mathbf{w}^T \mathbf{x} + w_0
\end{aligned}
$$

where

$$
\mathbf{w} = \mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2), \quad w_0 = -\frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)\mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) + \log \frac{p(C_1)}{p(C_2)}
$$

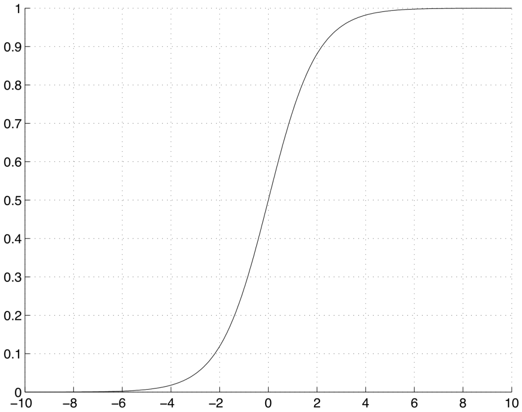
Sigmoid Function

- Sigmoid function or logistic function (inverse function of logit):

$$P(C_1 \mid \mathbf{x}) = \text{sigmoid}(\mathbf{w}^T\mathbf{x} + w_0) = \frac{1}{1 + \exp[-(\mathbf{w}^T\mathbf{x} + w_0)]}$$

  which directly computes the posterior class probability $P(C_1 \mid \mathbf{x})$.
- Training:
  - Estimate $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_1$, and $\boldsymbol{\Sigma}$ from data and plug the estimates into the discriminant functions.
- Testing:
  - Calculate $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ and choose C_1 if $g(x) > 0$, or
  - Calculate $y = \text{sigmoid}(\mathbf{w}^T\mathbf{x} + w_0)$ and choose C_1 if $y > 0.5$ (since y can be interpreted as a posterior probability and $\text{sigmoid}(0) = 0.5$).

Sigmoid Function (2)



Outline

Logistic Discrimination

- In logistic discrimination (or logistic regression), we do not model the class-conditional densities $p(x \mid C_i)$ but rather their ratio.
- Unlike the parametric classification approach (the likelihood-based approach studied before) which learns the classifier by estimating the parameters of $p(x \mid C_i)$, logistic discrimination (which is a discriminant-based approach) estimates the parameters of the discriminant directly.

Two Classes

- Let us start with two classes and assume that the log likelihood ratio is linear:

$$\log \frac{p(\mathbf{x} \mid C_1)}{p(\mathbf{x} \mid C_2)} = \mathbf{w}^T \mathbf{x} + w_0^0$$

- Using Bayes' rule, we have

$$\begin{aligned}\mathrm{logit}(P(C_1 \mid \mathbf{x})) =& \log \frac{P(C_1 \mid \mathbf{x})}{1 - P(C_1 \mid \mathbf{x})} = \log \frac{P(C_1 \mid \mathbf{x})}{P(C_2 \mid \mathbf{x})} = \log \frac{p(\mathbf{x} \mid C_1)}{p(\mathbf{x} \mid C_2)} + \log \frac{p(C_1)}{p(C_2)} \\ =& \mathbf{w}^T \mathbf{x} + w_0\end{aligned}$$

  where $w_0 = w_0^0 + \log[p(C_1)/P(C_2)]$
- Then we have

$$y = P(C_1 \mid \mathbf{x}) = \mathrm{sigmoid}(\mathbf{w}^T \mathbf{x} + w_0) = \frac{1}{1 + \exp[-(\mathbf{w}^T \mathbf{x} + w_0)]}$$

  - equivalent to the case when class-conditional densities are normal
  - logistic discrimination is more general, e.g., $\mathbf{x}$ may take discrete attributes

Parameter Learning

- Training set $\mathcal{X} = \{(\mathbf{x}^{(\ell)}, r^{(\ell)})\}_{\ell=1}^{N}$ where

$$r^{(\ell)} = \begin{cases} 1 & \text{if } \mathbf{x}^{(\ell)} \in C_1 \\ 0 & \text{if } \mathbf{x}^{(\ell)} \in C_2 \end{cases}$$

- Given an input $\mathbf{x}^{(\ell)}$, we assume that $r^{(\ell)}$ is Bernoulli with parameter $y^{(\ell)} = P(C_1 \mid \mathbf{x}^{(\ell)})$:

$$r^{(\ell)} \mid \mathbf{x}^{(\ell)} \sim \text{Bernoulli}(y^{(\ell)})$$

  Here, we see the difference from the likelihood-based methods where we modeled $p(\mathbf{x} \mid C_i)$; in the discriminant-based approach, we model directly $r^{(\ell)} \mid \mathbf{x}^{(\ell)}$.
- Likelihood:

$$L(\mathbf{w}, w_0 \mid \mathcal{X}) = \prod_{\ell} (y^{(\ell)})^{r^{(\ell)}} (1 - y^{(\ell)})^{1 - r^{(\ell)}}$$

Parameter Learning (2)

- Maximizing the likelihood function $L(\mathbf{w}, w_0 \mid \mathcal{X})$ is equivalent to minimizing an error function (cross-entropy) $E(\mathbf{w}, w_0 \mid \mathcal{X})$ defined as

$$\begin{aligned} E(\mathbf{w}, w_0 \mid \mathcal{X}) &= -\log L(\mathbf{w}, w_0 \mid \mathcal{X}) \\ &= -\sum_{\ell} \left[r^{(\ell)} \log y^{(\ell)} + (1 - r^{(\ell)}) \log(1 - y^{(\ell)}) \right] \end{aligned}$$

- No closed-form solution exists. Iterative algorithms such as gradient descent or more complicated methods can be used.
- Please keep in mind once a suitable model and an error function is defined, the (numerical) optimizationt of the model parameters to minimize the error function can be done by using one of many possible techniques.

Gradient Descent Learning

- If $y = \text{sigmoid}(a) = 1/[1 + \exp(-a)]$, its derivative is

$$\frac{dy}{da} = y(1 - y)$$

- Update equations for w_j $(j = 0, \ldots, d)$:

$$\begin{aligned}
\Delta w_j = -\eta \frac{\partial E}{\partial w_j} &= \eta \sum_{\ell} \left(\frac{r^{(\ell)}}{y^{(\ell)}} \frac{\partial y^{(\ell)}}{\partial w_j} - \frac{1 - r^{(\ell)}}{1 - y^{(\ell)}} \frac{\partial y^{(\ell)}}{\partial w_j} \right) \\
&= \eta \sum_{\ell} \left(\frac{r^{(\ell)}}{y^{(\ell)}} - \frac{1 - r^{(\ell)}}{1 - y^{(\ell)}} \right) \frac{\partial y^{(\ell)}}{\partial a^{(\ell)}} \frac{\partial a^{(\ell)}}{\partial w_j}
\end{aligned}$$

Gradient Descent Learning (2)

- Update equations for w_j $(j = 1, \ldots, d)$ and w_0:
  Since $a^{(\ell)} = \mathbf{w}^T \mathbf{x}^{(\ell)} + w_0$, we have

$$\frac{\partial a^{(\ell)}}{\partial w_j} = x_j^{(\ell)}, \quad j = 1, \ldots, d$$

  So

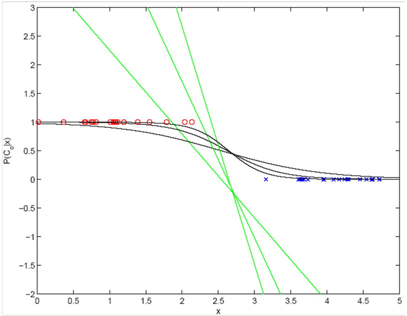
$$\begin{aligned}
\Delta w_j = -\eta \frac{\partial E}{\partial w_j} &= \eta \sum_{\ell} \left(\frac{r^{(\ell)}}{y^{(\ell)}} - \frac{1 - r^{(\ell)}}{1 - y^{(\ell)}} \right) y^{(\ell)}(1 - y^{(\ell)}) x_j^{(\ell)} \\
&= \eta \sum_{\ell} (r^{(\ell)} - y^{(\ell)}) x_j^{(\ell)}, \quad j = 1, \ldots, d \\
\Delta w_0 = -\eta \frac{\partial E}{\partial w_0} &= \eta \sum_{\ell} (r^{(\ell)} - y^{(\ell)})
\end{aligned}$$

Gradient Descent Algorithm

For $j = 0, \ldots, d$
    $w_j \leftarrow$rand(-0.01,0.01)
Repeat
    For $j = 0, \ldots, d$
        $\Delta w_j \leftarrow 0$
    For $t = 1, \ldots, N$
        $o \leftarrow 0$
        For $j = 0, \ldots, d$
            $o \leftarrow o + w_j x_j^t$
        $y \leftarrow \mathrm{sigmoid}(o)$
        $\Delta w_j \leftarrow \Delta w_j + (r^t - y) x_j^t$
    For $j = 0, \ldots, d$
        $w_j \leftarrow w_j + \eta \Delta w_j$
Until convergence

For w_0, we assume that there is an extra input x_0, which is always $+1$: $x_0^t = +1, \forall t$.

A Univariate Two-Class Example



Both $wx + w_0$ *and sigmoid*$(wx + w_0)$ *are shown as the learning develops.*

- We see that to get outputs of 0 and 1, the sigmoid hardens, which is achieved by increasing the magnitude of w, or $\|\mathbf{w}\|$ in the multivariate case.
- After training, during testing, given $\mathbf{x}^{(\ell)}$, we calculate $y^{(\ell)} = \text{sigmoid}(\mathbf{w}^T\mathbf{x}^{(\ell)} + w_0)$ and choose C_1 if $y^{(\ell)} > 0.5$ and choose C_2 otherwise.

Remarks on Parameter Learning

- To minimize # of misclassifications, we do not need to continue learning until all $y^{(\ell)}$ are 0 or 1, but only until $y^{(\ell)}$ are less than or greater than 0.5 (i.e., on the correct side of the decision boundary).
- If we do continue training beyond this point, cross-entropy will continue decreasing ($|w_j|$ will continue increasing to harden the sigmoid), but the number of misclassifications will not decrease.
- We continue training until the number of misclassifications does not decrease (which will be 0 if the classes are linearly separable).
- Actually stopping early before we have 0 training error is a form of regularization. Because we start with weights almost 0 and they move away as training continues, stopping early corresponds to a model with more weights close to 0 and effectively fewer parameters.

Multiple Classes

- One of the $K > 2$ classes, e.g., C_K, is taken as the reference class.
- Assume that

$$\log \frac{p(\mathbf{x} \mid C_i)}{p(\mathbf{x} \mid C_K)} = \mathbf{w}_i^T \mathbf{x} + w_{i0}^0, \quad i = 1, \ldots, K-1$$

So we have

$$\begin{aligned} \frac{P(C_i \mid \mathbf{x})}{P(C_K \mid \mathbf{x})} &= \frac{p(\mathbf{x} \mid C_i) p(C_i)}{p(\mathbf{x} \mid C_K) p(C_K)} \\ &= \exp(\mathbf{w}_i^T \mathbf{x} + w_{i0}^0) \cdot \exp\left(\log \frac{p(C_i)}{p(C_K)}\right) \\ &= \exp(\mathbf{w}_i^T \mathbf{x} + w_{i0}) \end{aligned} \tag{1}$$

where $w_{i0} = w_{i0}^0 + \log[p(C_i)/P(C_K)]$

Generalization of Sigmoid Function

- Summing (1) over $i = 1, \ldots, K-1$:

$$\sum_{i=1}^{K-1} \frac{P(C_i \mid \mathbf{x})}{P(C_K \mid \mathbf{x})} = \frac{1 - P(C_K \mid \mathbf{x})}{P(C_K \mid \mathbf{x})} = \sum_{i=1}^{K-1} \exp(\mathbf{w}_i^T \mathbf{x} + w_{i0})$$

we get

$$P(C_K \mid \mathbf{x}) = \frac{1}{1 + \sum_{i=1}^{K-1} \exp(\mathbf{w}_i^T \mathbf{x} + w_{i0})} \tag{2}$$

- From (1) and (2), we get

$$P(C_i \mid \mathbf{x}) = \frac{\exp(\mathbf{w}_i^T \mathbf{x} + w_{i0})}{1 + \sum_{j=1}^{K-1} \exp(\mathbf{w}_j^T \mathbf{x} + w_{j0})}, \quad i = 1, \ldots, K-1$$

Softmax Function

- If we want to treat all classes uniformly without having to choose a reference class, we can use the softmax function instead for the posterior class probabilities:

$$y_i = \hat{P}(C_i \mid \mathbf{x}) = \frac{\exp(\mathbf{w}_i^T \mathbf{x} + w_{i0})}{\sum_{j=1}^{K} \exp(\mathbf{w}_j^T \mathbf{x} + w_{j0})}, \quad i = 1, \ldots, K$$

- If $\mathbf{w}_i^T \mathbf{x} + w_{i0}$ for one class is sufficiently larger than for the others, its corresponding y_i will be close to 1 and the others will be close to 0.
- The softmax function behaves like taking a maximum, but it has the advantage of being differentiable.

Parameter Learning

- Each sample point is a multinomial trial with one draw, i.e.

$$\mathbf{r}^{(\ell)} \mid \mathbf{x}^{(\ell)} \sim \text{Mult}_K(1, \mathbf{y}^{(\ell)})$$

  where $y_i^{(\ell)} \equiv P(C_i \mid \mathbf{x}^{(\ell)})$

- Likelihood:

$$L(\{\mathbf{w}_i, w_{i0}\}_i \mid \mathcal{X}) = \prod_\ell \prod_i (y_i^{(\ell)})^{r_i^{(\ell)}}$$

- Cross-entropy error function:

$$E(\{\mathbf{w}_i, w_{i0}\}_i \mid \mathcal{X}) = -\sum_\ell \sum_i r_i^{(\ell)} \log y_i^{(\ell)}$$

Gradient Descent Learning

- If $y_i = \exp(a_i)/\sum_j \exp(a_j)$, its derivative is

$$\frac{\partial y_i}{\partial a_j} = y_i(\delta_{ij} - y_j)$$

where δ_{ij} is the Kronecker delta, which is 1 if $i = j$ and 0 if $i \neq j$.

- Update equations given $\sum_i r_i^{(\ell)} = 1$:

$$\Delta \mathbf{w}_j = \eta \sum_\ell \sum_i r_i^{(\ell)}(\delta_{ij} - y_j^{(\ell)})\mathbf{x}^{(\ell)} = \eta \sum_\ell \left[\sum_i r_i^{(\ell)}\delta_{ij} - y_j^{(\ell)} \sum_i r_i^{(\ell)}\right]\mathbf{x}^{(\ell)}$$

$$= \eta \sum_\ell (r_j^{(\ell)} - y_j^{(\ell)})\mathbf{x}^{(\ell)}, \quad j = 1, \ldots, K$$

$$\Delta w_{j0} = \eta \sum_\ell (r^{(\ell)} - y^{(\ell)})$$

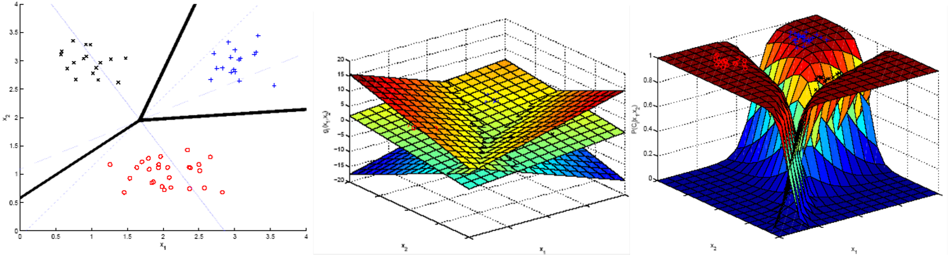
Note that because of the normalization in softmax, $\mathbf{w}_j$ and w_{j0} are affected not only by $\mathbf{x}^{(\ell)} \in C_j$ but also by $\mathbf{x}^{(\ell)} \in C_i$, $i \neq j$.

Gradient Descent Algorithm

```
For $i = 1, \ldots, K$, For $j = 0, \ldots, d$, $w_{ij} \leftarrow$ rand$(-0.01, 0.01)$
Repeat
    For $i = 1, \ldots, K$, For $j = 0, \ldots, d$, $\Delta w_{ij} \leftarrow 0$
    For $t = 1, \ldots, N$
        For $i = 1, \ldots, K$
            $o_i \leftarrow 0$
            For $j = 0, \ldots, d$
                $o_i \leftarrow o_i + w_{ij} x_j^t$
        For $i = 1, \ldots, K$
            $y_i \leftarrow \exp(o_i) / \sum_k \exp(o_k)$
        For $i = 1, \ldots, K$
            For $j = 0, \ldots, d$
                $\Delta w_{ij} \leftarrow \Delta w_{ij} + (r_i^t - y_i) x_j^t$
    For $i = 1, \ldots, K$
        For $j = 0, \ldots, d$
            $w_{ij} \leftarrow w_{ij} + \eta \Delta w_{ij}$
Until convergence
```

We take $x_0^t = 1$, $\forall t$.

A Two-dimensional Three-class Example



Remarks on Parameter Learning

- We do not need to continue training to minimize cross-entropy as much as possible; we train only until the correct class has the highest weighted sum, and therefore we can stop training earlier by checking the number of misclassifications.

Logistic Discriminant

- When data are normally distributed, the logistic discriminant has a comparable performance to the parametric, normal-based linear discriminant.
- Logistic discrimination can still be used when the class-conditional densities are nonnormal or when they are not unimodal, as long as classes are linearly separable.

Generalizing the Linear Model

- The ratio of class-conditional densities is of course not restricted to be linear.
- Quadratic discriminant:

$$\log \frac{p(\mathbf{x} \mid C_i)}{p(\mathbf{x} \mid C_K)} = \mathbf{x}^T \mathbf{W}_i \mathbf{x} + \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

  which corresponds to parametric discrimination with multivariate normal class-conditionals having different covariance matrices.
- Sum of nonlinear basis functions:

$$\log \frac{p(\mathbf{x} \mid C_i)}{p(\mathbf{x} \mid C_K)} = \mathbf{w}_i^T \mathbf{\Phi}(\mathbf{x}) + w_{i0}$$

  where $\mathbf{\Phi}(\cdot)$ are basis functions which transform the original input variables to a new set of variables.
- Basis functions are related to:
  - Hidden units like sigmoid function in neural networks (studied later)
  - Kernels in kernel methods such as support vector machines (SVM) (studied later).