Classification

Jiayu Zhou

[1]Department of Computer Science and Engineering
Michigan State University
East Lansing, MI USA

February 18, 2016

Table of contents

Problem Statement

Given an input vector $\mathbf{x}$ and a set of training patterns $\mathbf{x}_1, \ldots, \mathbf{x}_N$, the goal is to assign it to one of K discrete classes $\mathcal{C}_k$ where $k = 1, \ldots, K$

Different Approaches to Classification

- Construct a **discriminant function** which assigns each vector $\mathbf{x}$ to a specific class
- Model the conditional probability distribution $p(\mathcal{C}_k|\mathbf{x})$ in an inference stage, and use this distribution to make optimal decisions
- Two methods to model $p(\mathcal{C}_k|\mathbf{x})$
  - Discriminant Model
    Example: Representing $p(\mathcal{C}_k|\mathbf{x})$ as parametric models and then optimizing the parameters using a training set
  - Generative method
    Model the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ and prior probabilities $p(\mathcal{C}_k)$, and compute $p(\mathcal{C}_k|\mathbf{x})$ using Bayes theorem

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}$$

Basic Definitions

- Discriminant
  A discriminant is a function that takes an input vector $\mathbf{x}$ and assign it to one of K classes, denoted as $\mathcal{C}_k$
- Linear discriminant
  In linear discriminant, the decision boundaries (or decision surfaces) are hyperplanes in the input space
- Linear separable
  Data sets whose classes can be separated exactly by linear decision surfaces are said to be **linear separable**
- Generalized linear models

$$y(\mathbf{x}) = f(\mathbf{w}^T\mathbf{x} + w_0)$$

$f(.)$ is called **activation function**, and it may be **nonlinear**.

Discriminant FunctionsTwo Classes

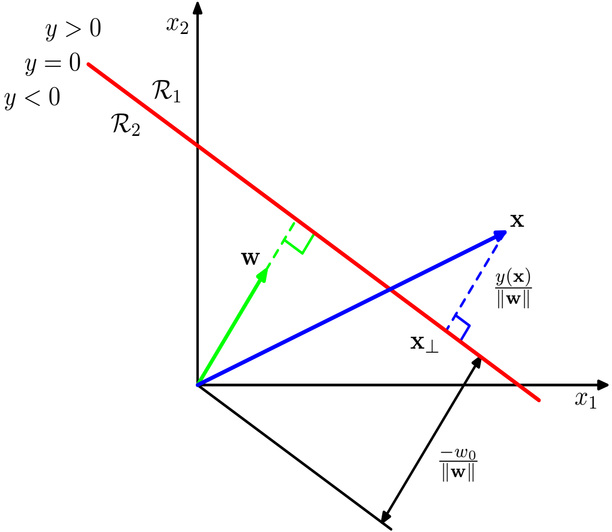
- Formulation

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$$

  - $\mathbf{w}$: weight vector
  - w_0: bias
  - $-w_0$: threshold
- Decision Rule
  An input vector $\mathbf{x}$ is assigned to class $\mathcal{C}_1$ if $y(\mathbf{x}) \geq 0$ and to class $\mathcal{C}_2$ otherwise
- The geometric property
  - $\mathbf{w}$: the direction of decision surface
  - w_0: the location of decision surface
  - The signed distance r of point $\mathbf{x}$ from the decision surface

$$r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}$$

An Example of Geometry



Discriminant FunctionsMultiple Classes

- Possible Methods
  1. One-versus-the-rest
     Use $K-1$ two-class classifiers
  2. One-versus-one
     Use $K(K-1)/2$ two-class classifiers
- Define a single K-class discriminant comprising K linear functions of the form

$$y_k(\mathbf{x}) = \mathbf{w}_k^T\mathbf{x} + w_{k0}$$

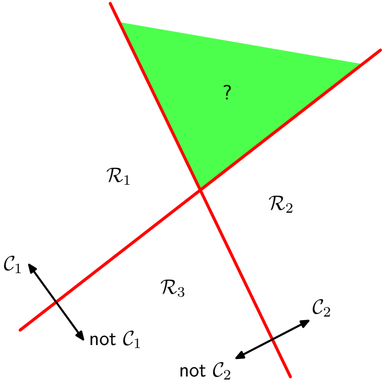
- Decision Rule
  Assigning a point $\mathbf{x}$ to class $\mathcal{C}_k$ if $y_k(\mathbf{x}) > y_j(\mathbf{x})$ for all $j \neq k$
- The decision boundary between class $\mathcal{C}_k$ and class $\mathcal{C}_j$

$$y_k(\mathbf{x}) = y_j(\mathbf{x})$$

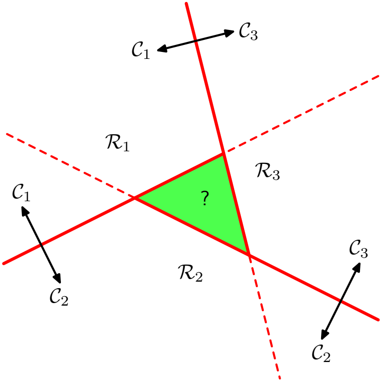
  This boundary is $(D-1)$-dimensional hyperplane defined by

$$(\mathbf{w}_k - \mathbf{w}_j)^T\mathbf{x} + (w_{k0} - w_{j0}) = 0$$

The Difficulty of One-versus-the-rest and One-versus-one



The dilemma of One-versus-the-rest



The dilemma of One-versus-one

Properties of Multi-class Classifier

The decision regions of the discriminant given by $y_k(\mathbf{x}) = \mathbf{w}_k^T\mathbf{x} + w_{k0}$ are convex.

Proof

Any point $\hat{\mathbf{x}}$ that lies on the line connecting $\mathbf{x}_A$ and $\mathbf{x}_B$ can be expressed in the form

$$\hat{\mathbf{x}} = \lambda\mathbf{x}_A + (1\lambda)\mathbf{x}_B, \text{where} 0 \leq \lambda \leq 1$$

If $\mathbf{x}_A$ and $\mathbf{x}_B$ lies in $\mathcal{R}_k$, then

$$y_k(\mathbf{x}_A) > y_j(\mathbf{x}_A) \text{ for all } j \neq k$$

$$y_k(\mathbf{x}_B) > y_j(\mathbf{x}_B) \text{ for all } j \neq k$$

From the linearity of the discriminant functions, it follows that

$$y_k(\hat{\mathbf{x}}) > y_j(\hat{\mathbf{x}}) \text{ for all } j \neq k$$

It means that $\hat{\mathbf{x}} \in \mathcal{R}_k$.

- Classification and Decision Surface
- Multiple Classes

2 Non-probabilistic Methods

- Least Squares
- Fisher's Linear Discriminant
- Generative Model Vs. Discriminative Mode
- Logistic Regression
- Implementations of Logistic Regression

Least Squares for Classification

- Each class $\mathcal{C}_k$ is described by its own linear model so that

$$y_k(\mathbf{x}) = \mathbf{w}_k^T \mathbf{x} + w_{k0}$$

  where $k = 1, \ldots, K$

- Or equivalently

$$\mathbf{y} = \tilde{\mathbf{W}}^T \tilde{\mathbf{x}}$$

  where $\tilde{\mathbf{W}} = [\tilde{\mathbf{w}}_1, \ldots, \tilde{\mathbf{w}}_K], \tilde{\mathbf{w}}_k = (w_{k0}, \mathbf{w}_k^T)^T, \tilde{\mathbf{x}} = (1, \mathbf{x}^T)^T$

- By defining the target matrix T, the sum-of-squares error function is

$$\begin{aligned} E_D(\tilde{\mathbf{W}}) &= \frac{1}{2} \|\tilde{\mathbf{X}}\tilde{\mathbf{W}} - \mathbf{T}\|_F^2 \\ &= \frac{1}{2} \mathrm{Tr}\{(\tilde{\mathbf{X}}\tilde{\mathbf{W}} - \mathbf{T})^T (\tilde{\mathbf{X}}\tilde{\mathbf{W}} - \mathbf{T})\} \end{aligned}$$

Least Squares for Classification (cont.)

- The solution is

$$\tilde{\mathbf{W}} = (\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^1\tilde{\mathbf{X}}^T\mathbf{T} = \tilde{\mathbf{X}}^\dagger\mathbf{T}$$

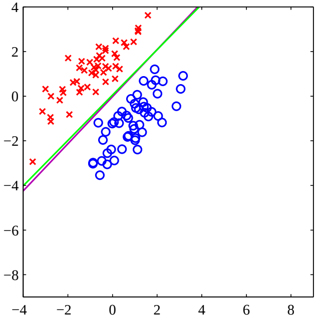
- Then the discriminant is

$$\mathbf{y}(\mathbf{x}) = \tilde{\mathbf{W}}^T\tilde{\mathbf{x}} = \mathbf{T}^T(\tilde{\mathbf{X}}^\dagger)^T\tilde{\mathbf{x}}$$

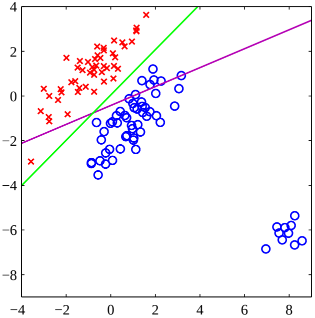
Drawbacks of Least Square

- Least-squares solutions **lack robustness** to outliers
  The sum-of-squares error function penalizes predictions that are "too correct" in that they lie a long way on the correct side of the decision boundary
- Sometimes least squares have **poor performance**
  Least squares corresponds to maximum likelihood under the assumption of a Gaussian conditional distribution, whereas binary target vectors clearly have a distribution that is far from Gaussian

Least-squares Solutions Lack Robustness to Outliers

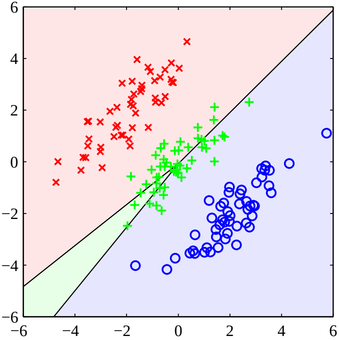


Original Boundary. Magenta curve is least squares and green curve is logistic regression

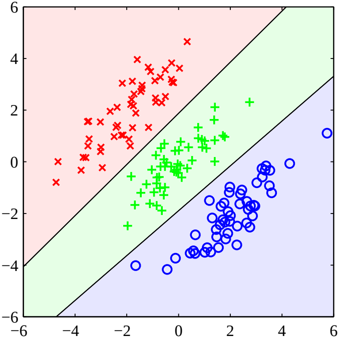


New Boundary

Poor Performance of Least-squares



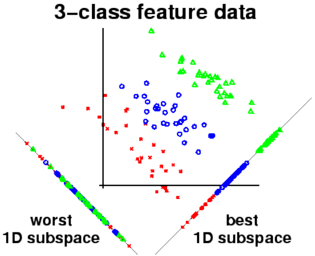
Least Squares



Logistic Regression

Fisher's Linear Discriminant

- Basic Idea
  Project the data in the original D-dimensional space into lower space. In the projection, we expect to maximize the between-class distance and minimize within-class distance



- For simplicity, we First consider the projection to 1-dimensional space for two-class problem.

Formal Formulation

- N_1: number of points in class $\mathcal{C}_1$
  N_2: number of points in class $\mathcal{C}_2$
- The mean vectors of each class in original space

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in \mathcal{C}_1} \mathbf{x}_n \qquad (1)$$

$$\mathbf{m}_2 = \frac{1}{N_2} \sum_{n \in \mathcal{C}_2} \mathbf{x}_n \qquad (2)$$

- The linear projection is

$$y = \mathbf{w}^T \mathbf{x}$$

Fisher's Linear Discriminant

- The distance between classes is measured by the distance of means in the projected 1-dimensional space

$$m_2 - m_1 = \mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)$$

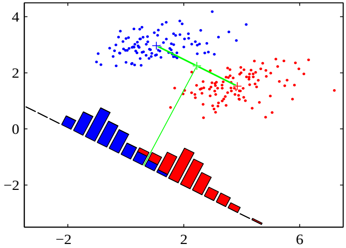
- The measure of within-class distance is measured by the variance within each class in the projected 1-dimensional space

$$s_k^2 = \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2$$

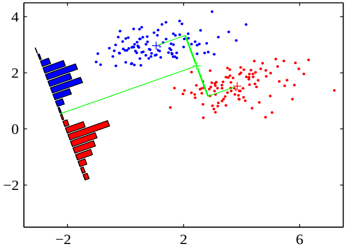
- Fisher criterion

$$\max J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

Example of Linear Projection



Projection onto the line jointing class means



LDA Projection

Fishers Linear Discriminant

- Reformulation

$$\max J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

  - Between-class covariance matrix $\mathbf{S}_B$

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T$$

  - Within-class covariance matrix $\mathbf{S}_W$

$$\mathbf{S}_W = \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \mathbf{x}_1)(\mathbf{x}_n - \mathbf{x}_1)^T + \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \mathbf{x}_2)(\mathbf{x}_n - \mathbf{x}_2)^T$$

- Set gradient to zero

$$\frac{\partial J}{\partial \mathbf{w}} = 0 \Leftrightarrow (\mathbf{w}^T \mathbf{S}_B \mathbf{w}) \mathbf{S}_W \mathbf{w} = (\mathbf{w}^T \mathbf{S}_W \mathbf{w}) \mathbf{S}_B \mathbf{w} \Rightarrow \mathbf{w} \propto \mathbf{S}_W^{-1} (\mathbf{m}_2 - \mathbf{m}_1)$$

Least Squares Versus Fishers Linear Discriminant

- For two-class problems, Fishers Linear Discriminant can be considered as a special case of least squares.
- For multi-class problems, Fishers Linear Discriminant can also be considered a special case of least squares by constructing a special indicator matrix (Ye, ICML 2007).

Two-class Problem I

- Key Idea: define a special target variable for different classes in least squares
- Consider a special least squares with the following target variable

$$t_n = \begin{cases} \frac{N}{N_1} & \text{if } n \in \mathcal{C}_1 \\ -\frac{N}{N_2} & \text{if } n \in \mathcal{C}_2 \end{cases}$$

- Properties of the target variable:

$$\sum_{n=1}^{N} t_n = 0$$

$$\sum_{n=1}^{N} t_n x_n = N(\mathbf{m}_1 - \mathbf{m}_2)$$

Two-class Problem II

- The corresponding sum-of-squares error function for the target variable is

$$\min E = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n + w_0 - t_n)^2$$

- Setting the derivatives of E with respect to w_0 and $\mathbf{w}$ to 0, we have

$$\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n + w_0 - t_n) = 0 \Leftrightarrow w_0 = -\mathbf{w}^T\mathbf{m}$$

$$\text{where } \mathbf{m} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n = \frac{1}{N}(N_1\mathbf{m}_1 + N_2\mathbf{m}_2)$$

$$\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n + w_0 - t_n)\mathbf{x}_n = 0 \Leftrightarrow (\mathbf{S}_W + \frac{N_1N_2}{N}\mathbf{S}_B)\mathbf{w} = N(\mathbf{m}_1 - \mathbf{m}_2)$$

Two-class Problem III

$$(\mathbf{S}_W + \frac{N_1 N_2}{N}\mathbf{S}_B)\mathbf{w} = N(\mathbf{m}_1 - \mathbf{m}_2) \Leftrightarrow \mathbf{S}_W\mathbf{w} = N(\mathbf{m}_1 - \mathbf{m}_2) - \frac{N_1 N_2}{N}\mathbf{S}_B\mathbf{w}$$

Note that

$$\mathbf{S}_B\mathbf{w} = (\mathbf{m}_2 - \mathbf{m}_1)\left((\mathbf{m}_2 - \mathbf{m}_1)^T\mathbf{w}\right) = s(\mathbf{m}_2 - \mathbf{m}_1), \text{ where } s \in \mathbb{R}$$

Therefore

$$\begin{aligned} \mathbf{S}_W\mathbf{w} &= s'(\mathbf{m}_2 - \mathbf{m}_1), \text{ where } s' \in \mathbb{R} \\ &\Rightarrow \mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1) \end{aligned}$$

This result is the same as Fishers Linear Discriminant

Multi-class LDA I

- Suppose the class number is K, we consider project the data in the original D-dimensional space $(D > K)$ data space into D'-dimensional space, where $D' > 1$

$$\mathbf{y} = \mathbf{W}^T\mathbf{x}, \text{ where } \mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{D'}]$$

- The within class covariance $\mathbf{S}_W$

$$\mathbf{S}_W = \sum_{k=1}^{K} \mathbf{S}_k$$

where

$$\mathbf{S}_k = \sum_{n \in \mathcal{C}_k} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^T$$

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{x}_n$$

Multi-class LDA II

- The within class covariance $\mathbf{S}_T$

$$\mathbf{S}_T = \sum_{n=1}^{N} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^T$$

$$\mathbf{m} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n$$

- The between class covariance $\mathbf{S}_B$

$$\mathbf{S}_B = \sum_{k=1}^{K} N_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T$$

- From these definitions we can show that

$$\mathbf{S}_T = \mathbf{S}_B + \mathbf{S}_W$$

Multi-class LDA III

- In the projected space, we can define similar structures

$$\mathbf{s}_W = \sum_{k=1}^{K} \sum_{n \in \mathcal{C}_k} (\mathbf{y}_n - \boldsymbol{\mu}_k)(\mathbf{y}_n - \boldsymbol{\mu}_k)^T$$

$$\mathbf{s}_B = \sum_{k=1}^{K} N_k (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^T$$

where

$$\boldsymbol{\mu} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{y}_n$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{y}_n$$

Multi-class LDA IV

- Many objective functions can be chosen in the lower space.
- One common choice is

$$\max J(\mathbf{W}) = \mathrm{Tr}\{\mathbf{s}_W^{-1}\mathbf{s}_B\} \Leftrightarrow \max J(\mathbf{W}) = \mathrm{Tr}\{(\mathbf{W}\mathbf{S}_W\mathbf{W}^T)^{-1}(\mathbf{W}\mathbf{S}_B\mathbf{W}^T)\}$$

- In fact, $\mathbf{W}$ is given by the D' eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$ corresponding to the D' largest eigenvalues.
- $\mathbf{S}_B$ is composed of the sum of K matrices, each of which is an outer product of two vectors and therefore of rank 1. In addition, only $(K-1)$ of these matrices are independent. Thus, $\mathbf{S}_B$ has rank at most equal to $(K-1)$ and so there are at most $(K-1)$ nonzero eigenvalues. In practice, we commonly set $D' = K-1$.

Outline

Generative Model Vs. Discriminative Mode

- Probabilistic Generative Model
  Model the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ and prior probabilities $p(\mathcal{C}_k)$, and compute $p(\mathcal{C}_k|\mathbf{x})$ using Bayes' theorem

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}$$

- Probabilistic Discriminative Model
  Maximize a likelihood function defined through the conditional distribution $p(\mathcal{C}_k|\mathbf{x})$
- Advantages of Discriminative Models
  - Fewer adaptive parameters need to be determined.
  - Performance will be improved, especially when the class-conditional density assumption gives a poor approximation to the true distributions.

Fixed Basis Function

- The original data $\mathbf{x}$ are mapped into $\phi(\mathbf{x})$ using a vector of basis functions $\phi(\mathbf{x})$.
- The resulting model is linear in the feature space ϕ, but may not be linear in the original $\mathbf{x}$ space.
- All of the algorithms are equally applicable if we first make a fixed nonlinear transformation of the inputs using a vector of basis functions $\phi(\mathbf{x})$.
- The following discussions are based on the ϕ space.

Logistic Regression - I

- In two-class problem, the posterior probability of class $\mathcal{C}_1$ can be written as a logistic sigmoid acting on a linear function of the feature vector ϕ so that

$$p(\mathcal{C}_1|\phi) = y(\phi) = \sigma(\mathbf{w}^T\phi).$$

- $\sigma(.)$ is the logistic sigmoid function.
- $p(\mathcal{C}_2|\phi) = 1 - p(\mathcal{C}_1|\phi)$
- In statistics, the model is known as **logistic regression**.
- The model is for **classification**, not for regression.
- In logistic regression, we estimate the parameter $\mathbf{w}$ directly.

Logistic Regression - II

- Comparison of logistic regression and generative model in M-dimensional space
  - In logistic regression, only M parameters (components of $\mathbf{w}$)
  - In generative model, suppose Gaussian class-conditional densities and maximum likelihood method are used, the number of parameters is $M(M+5)/2+1$
    - Means: $2M$ parameters
    - Shared covariance: $(M+1)M/2$ parameters
    - Prior $p(\mathcal{C}_1)$: 1 parameter
- **Maximum Likelihood** method is used to determine the parameters of the logistic regression model.
- Maximum likelihood can exhibit severe over-fitting.
- This can be overcome by inclusion of a prior and finding a MAP solution for $\mathbf{w}$, or equivalently by adding a regularization term to the error function.

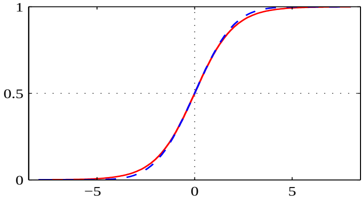
Logistic Regression - III - Logistic Sigmoid function

- Definition of **Logistic Sigmoid** function $\sigma(a)$:

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

- Properties of Logistic Sigmoid function $\sigma(a)$:

$$\sigma(-a) = 1 - \sigma(a)$$
$$\frac{d\sigma}{da} = \sigma(1 - \sigma)$$



Plot of the logistic sigmoid function $\sigma(a)$ (Red Line)

Logistic Regression - IV - Estimate w

- For a training data set $\{\phi_n, t_n\}$ where $t_n \in \{0, 1\}$ and $n = 1, 2, , N$, the likelihood function is

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n}(1 - y_n)^{1-t_n} \tag{3}$$

- Definitions of $t_n, \mathbf{t}$ and y_n

$$t_n = \begin{cases} 1 \text{ if } n \in \mathcal{C}_1 \\ 0 \text{ if } n \in \mathcal{C}_2 \end{cases}$$

$$\mathbf{t} = (t_1, t_2, \ldots, t_N)^T$$

$$y_n = p(\mathcal{C}_1|\phi_n) = \sigma(\mathbf{w}^T \phi_n)$$

Logistic Regression - V

- The error function is the negative logarithm of the likelihood, namely, **Cross-entropy** error function:

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = \sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

The gradient of cross entropy function with respect to **w** is

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\phi_n$$

Implementations of Logistic Regression - I

- Two different algorithms:
    - Batch version
    - Online version

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \eta \left(t_n - \sigma(\mathbf{w}^{(\tau)T} \phi_n) \right) \phi_n$$

- The cross-entropy error function is convex, and a unique minimum is ensured.
- **Newton-Raphson Algorithm**
  It uses a local quadratic approximation to the cross-entropy error function to update $\mathbf{w}$ iteratively

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - H^{-1} \nabla E(\mathbf{w})$$

Implementations of Logistic Regression - II

- For linear regression error $E = \frac{1}{2}\sum_{i=1}^{N}\{t_n - \mathbf{w}^T\phi_n\}^2$, the gradient and Hessian matrix are given by

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N}\{\mathbf{w}^T\phi_n - t_n\}\phi_n = \mathbf{\Phi}^T\mathbf{\Phi}\mathbf{w} - \mathbf{\Phi}^T\mathbf{t}$$

$$H = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N}\phi_n\phi_n^T = \mathbf{\Phi}^T\mathbf{\Phi}$$

  where n-th row of $\mathbf{\Phi}$ is ϕ_n^T

- The Newton-Raphson update for this function is

$$\begin{aligned}\mathbf{w}^{(\text{new})} &= \mathbf{w}^{(\text{old})} - (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\{\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{w}^{(\text{old})} - \mathbf{\Phi}^T\mathbf{t}\} \\ &= (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}\end{aligned}$$

- It shows that if the error function is quadratic, the Newton-Raphson method gives the exact solution in one step.

Implementations of Logistic Regression - III

- For cross-entropy error function, we can compute gradient and Hessian matrix as follows:

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\phi_n = \mathbf{\Phi}^T(\mathbf{y} - \mathbf{t})$$

$$H = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} y_n(1 - y_n)\phi_n\phi_n^T = \mathbf{\Phi}^T\mathbf{R}\mathbf{\Phi}$$

  where $\mathbf{R}$ is an $N \times N$ diagonal matrix, and $R_{nn} = y_n(1 - y_n)$
- Note that $0 < y_n < 1$, then $\mathbf{R}$ is positive definite, and H is also positive definite
- Since H is positive definite, the cross-entropy error function is convex, and it has a unique minimum

Implementations of Logistic Regression - IV

- The Newton-Raphson update for cross-entropy error function is

$$\begin{aligned}\mathbf{w}^{(\text{new})} &= \mathbf{w}^{(\text{old})} - (\mathbf{\Phi}^T\mathbf{R}\mathbf{\Phi})^{-1}\mathbf{\Phi}^T(\mathbf{y}-\mathbf{t}) \\ &= (\mathbf{\Phi}^T\mathbf{R}\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{R}\mathbf{z}\end{aligned}$$

  where $\mathbf{z} = \mathbf{\Phi}\mathbf{w}^{(old)} - \mathbf{R}^{-1}(\mathbf{y}-\mathbf{t})$
- The solution takes the form of a set of normal equations for a weighted least-squares problem, and diagonal matrix $\mathbf{R}$ is the weight in each iteration.
- Newton-Raphson algorithm is also known as **iterative reweighted least squares**.