PRML과 부수적인 자료들을 공부하여 간단히 정리하였습니다.

input space는 *decision regions* 로 나눠지는데 이는 *decision boundaries(decision surfaces)* 에 의해 나눠진다. 이번 챕터에서는 분류 선형모델에 대해 공부하는데 이는 decision surfaces가 **input x의 linear function** 이라는 것을 의미한다. **D차원의 input space가 D-1 차원의 hyperplane으로 나눠지는 것이다.** 크게 3가지로 나누어서 공부한다.

- Discriminant function
- generative
- Discriminative

classification에서는 discrete class labels 이나 각 class가 될 probability를 target으로 예측한다. 후자의 경우 (0,1) 사이의 값을 가질 것이다. 따라서 우리는 linear function of $\mathbf{w}$를 nonlinear function을 이용하여 transform한다.

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

machine learning에서는 f를 *activation function* 이라고 부른다. 통계학에서는 inverse of *link function* 으로 부른다. 따라서 이전에 봤던 regression model과는 다르게 더이상 parameter에 linear하지 않는 성질을 가진다.

4.1 Discriminant Functions

- discriminant : a function that takes an input vector $\mathbf{x}$ and assigns it to one of K class
  - 이번 chapter에서는 *linear discriminant* (: decision surfaces are hyperplane) 로 한정지어 공부할 것이다.

4.1.1 two classes

가장 간단한 linear discriminant function을 보면

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

- $y(\mathbf{x}) \geq 0$ 이면 class 1이고 반대면 class 2 이다. 따라서 decision boundary는 $y(\mathbf{x}) = 0$ 이고 $(D-1)$차원의 hyperplane 이다.
- decision surface 위에 두 점 $\mathbf{x}_A, \mathbf{x}_B$ 이 있다고 가정하면
  - $\mathbf{w}^T(\mathbf{x}_A - \mathbf{x}_B) = 0$ 이므로 vector $\mathbf{w}$는 decision surface에 있는 모든 점들과 orthogonal하다. 이는 $\mathbf{w}$가 decision surface의 orientation을 결정한다는 의미이다.
- 똑같이 $\mathbf{x}$가 decision surface 위의 점이라고 하고 원점과 decision surface의 거리를 계산하면 아래와 같다.
  - 여기서 $\mathbf{w}_0$는 decision surface의 위치를 결정한다.

$$\frac{\mathbf{w}^T\mathbf{x}}{||\mathbf{w}||} = -\frac{\mathbf{w}_0}{||\mathbf{w}||}$$

4.1.2 Multiple classes

K 가 2보다 큰 multiple class를 분류하는 상황을 생각해보자. linear discriminant로 분류하는 방법은 크게 두 가지로 나눌 수 있다.

- one vs the rest
- one vs one

두 방법 모두 class를 결정하는데 있어 애매한 상황이 발생한다. hyperplane이라는 제약때문에 그 어떤 class에도 속하지 못하는 지역이 발생한다. (PRML figure 4.2 에 잘 보여줌) 이를 해결하기 위해 아래와 같은 K개의 linear function을 K-class discriminant로 사용한다.

$$y_k(x) = w_k^T x + w_{k0}$$

- $y_k(\mathbf{x}) \geq y_j(\mathbf{x})$ 인 경우, $\mathbf{x}$는 k로 분류한다. 즉, 큰 값을 가지는 쪽으로!

여기서 만들어지는 decision region은 항상 singly connected and convex하다.

- decision region R_k에 들어있는 두 점 $\mathbf{x}_A, \mathbf{x}_B$
- 이 두 점을 연결한 선 위에 점 $\hat{\mathbf{x}}$이 있다고 가정하자. 이를 표현하면 ($0 \leq \lambda \leq 1$)

$$\hat{\mathbf{x}} = \lambda \mathbf{x}_A + (1-\lambda)\mathbf{x}_B$$

따라서 discriminant function은 다음을 만족한다.

$$y_k(\hat{\mathbf{x}}) = \lambda y_k(\mathbf{x}_A) + (1-\lambda) y_k(\mathbf{x}_B)$$

- $y_k(\mathbf{x}_A) > y_j(\mathbf{x}_A), y_k(\mathbf{x}_B) > y_j(\mathbf{x}_B)$ 을 만족하기에
- $y_k(\hat{\mathbf{x}}) > y_j(\hat{\mathbf{x}})$ 도 성립한다.
  - 따라서, $\hat{\mathbf{x}}$은 항상 R_k에 속한다.

이제 linear discriminant function의 parameter를 학습하는 방법에 대해 배울 것이다.

- least square
- Fisher's linear discriminant
- perceptron algorithm

4.1.3 Least squares for classification

이전의 sum of squares error function을 그대로 이용한다. target은 1-of-K binary coding하여 vector $\mathbf{t}$ 이다. (해당하는 class는 1 나머지 class는 0으로 표현)

- 각 class 마다 $y_k(\mathbf{x}) = \mathbf{w}_k^T \mathbf{x} + w_{k0}$, 이를 합쳐서 표현하면

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

- $\widetilde{\mathbf{W}}$: 각 컬럼이 $\widetilde{\mathbf{w}}_k = (\mathbf{w}_{k0}, \mathbf{w}_k^T)$
- $\widetilde{\mathbf{w}}_k^T \tilde{\mathbf{x}}$가 가장 큰 값(class)에 input $\mathbf{x}$를 할당한다.
- normal equation으로 parameter를 구하면

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

- 특징
  - exact closed-form의 solution이 나온다.
  - output이 확률의 범위 (0,1) 을 넘어가는 경우가 존재한다. (우리는 output이 확률값이길 원한다)
  - least square의 단점인 outlier에 취약하다. input data에 따라서 decision이 급변하다.

4.1.4 Fisher's linear discriminant

차원 축소의 역할로 많이 쓰이는데 classification으로도 사용가능하다. 일단은 2-class의 경우만 고려해보자.

- D 차원의 input vector $\mathbf{x}$를 1차원에 project한다고 생각하자.

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

이렇게 할 수 있다. 하지만 overlapping되니까 class seperation을 최대화하는 projection을 하는 것이다.

- 각 class의 평균을 $\mathbf{m}_1, \mathbf{m}_2$이라고 하면 아래의 값을 최대로 하는 $\mathbf{w}$를 찾아야 한다.
  - $\mathbf{m_1} = 1/N_1 \sum_{n \in C_1} \mathbf{x}_n$
  - $\mathbf{m_2} = 1/N_2 \sum_{n \in C_2} \mathbf{x}_n$

$$m_2 - m_1 = \mathbf{w}^T(\mathbf{m}_1 - \mathbf{m}_2), \quad where\ m_k = \mathbf{w}^T \mathbf{m}_k$$

$\mathbf{w}$를 계속 키우면 커지기 때문에 제약식 $\sum \mathbf{w}_i^2 = 1$을 두고 라그랑지로 풀면

$$\mathbf{w} \propto (\mathbf{m}_2 - \mathbf{m}_1)$$

의 결론을 얻는다.

- 이에 추가적으로 Fisher는 **within class의 varinace를 최소화** 하고자 했다. 반면에 **between class의 variance는 최대화** 한다.
- class C_k의 within variance는
  - $y_n = \mathbf{w}^T\mathbf{x}_n$
  - $m_k = \mathbf{w}^T\mathbf{m}_k$

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

전체 class의 within variance는 $s_1^2 + s_2^2$ 이를 통해

- Fisher criterion (ratio of the between-class variance to the within-class variance)은

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

- Fisher criterion을 다시 쓰면

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

- $\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T$: between-class covariance matrix
- $\mathbf{S}_W = \sum_{n \in C_1}(\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^T + \sum_{n \in C_2}(\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^T$: total within-class covariance matrix
- w에 대해 미분하고 위의 값을 최대화하는 값을 찾으면 $\mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$. 이 결과를 **Fisher's linear discriminant** 라고 한다. 1차원에 projection한 뒤에 특정 threshold값을 정해 classification할 수 있다.

4.1.5 Relation to least squares

Fisher criterion은 least square의 특별한 경우이다. target을 1-of-K encoding의 방법이 아닌

- class 1은 N/N_1
- class 2는 $-N/N_2$

으로 encoding 하면 된다. 이렇게 한 뒤에 least square의 방법대로 parameter를 구하면 Fisher criterion이 나온다. (과정은 생략)

4.1.6 Fisher's discriminant for multiple classes

- skip

4.1.7 The perceptron algorithm

- perceptron 특징
  - 2 class에서만 사용가능하다.
  - based on linear combination of fixed basis function
  - target을 이전에는 주로 1,0 으로 했는데 여기서는 -1, 1 로 코딩한다.
- **perceptron criterion** (error function)

$$E_p(\mathbf{w}) = -\sum_{n \in M}\mathbf{w}^T\phi_n t_n$$

M은 잘못분류한 케이스를 의미한다. 우리는 이 criterion을 최소화 하고자 한다.

- $\mathbf{w}^T\phi_n > 0$ 이면 1로 분류
- $\mathbf{w}^T\phi_n < 0$ 이면 -1로 분류
  - 따라서 분류를 잘못하면 $\mathbf{w}^T\phi_n t_n < 0$ 이고 error가 커지는 것이다.
- 위 perceptron criterion을 SGD로 iterative하게 계산하면

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_p(\mathbf{w}) = \mathbf{w}^{(\tau)} + \eta \phi_n t_n$$

(η는 learning rate) 이다. 이를 쉽게 해석하면 분류가 맞으면 놔두고 틀리면 그 ϕ_n 만큼 더하고 빼고 하는 것이다. 양변에 $-\phi_n t_n$을 곱하면 에러가 줄어듬(parameter가 converge)을 알 수 있다.

$$-\mathbf{w}^{(\tau+1)T}\phi_n t_n = -\mathbf{w}^{(\tau)T}\phi_n t_n - (\phi_n t_n)^T \phi_n t_n < -\mathbf{w}^{(\tau)T}\phi_n t_n$$

- *perceptron convergence theorem*
  - training data set is linearly separable 하면 perceptron algorithm수렴한다 (반드시 해당하는 decision boundary를 찾을 수 있다) . 아니면 수렴이 안된다.
  - 수렴하기 전까지 이게 non separable 문제인지 아니면 수렴이 천천히 되는 건지 파악하기 어렵다.

4.2 Probabilistic Generative Models

data의 분포에 대한 가정을 갖는 decision boundary에 대해 공부해보자. $p(x/C_k), p(C_k)$로 베이즈정리를 이용하여 posterior를 계산한다. (일단 binary classification의 경우)

- posterior probability for class 1 :

$$\begin{aligned} p(C_1|\mathbf{x}) &= \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_1)p(C_1) + p(\mathbf{x}|C_2)p(C_2)} \\ &= \frac{1}{1 + \frac{p(\mathbf{x}|C_2)p(C_2)}{p(\mathbf{x}|C_1)p(C_1)}} \\ &= \frac{1}{1+exp(-a)} = \sigma(a) \end{aligned}$$

$$where \;\; a = \ln \frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)}$$

- $\sigma(x) = \frac{1}{1+exp(-x)}$ 이 식은 *logistic sigmoid* function 이다.
- 이의 inverse는 $x = \ln(\frac{\sigma}{1-\sigma})$ 이고 *logit* function이라고 한다.

이번에는 일반적인 경우에 대해 살펴보자. multi class의 경우

$$p(C_k|\mathbf{x}) = \frac{p(\mathbf{x}|C_k)p(C_k)}{\sum p(\mathbf{x}|C_j)p(C_j)} = \frac{exp(a_k)}{\sum_j exp(a_j)}$$

$$where \;\; a_k = \ln p(\mathbf{x}|C_k)p(C_k)$$

이를 *normalized exponential* or *softmax function* 이라고 한다.

4.2.1 Continuous inputs

class-conditional density를 Gaussian이라고 가정하고 posterior를 살펴보자. 단 모든 class는 같은 covariance matrix를 가진다. (2-class)

$$p(\mathbf{x}|C_k) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right\}$$

이므로 이를 이용해 위에서 구한 posterior를 계산하면

- $a = \ln \frac{p(x/C_1)p(C_1)}{p(x/C_2)p(C_2)}$

$$p(C_1|\mathbf{x}) = \sigma(a) = \sigma(\mathbf{w}^T\mathbf{x} + w_0)$$

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

의 형태가 나온다. logistic sigmoid안에서 $\mathbf{x}$ 의 linear function의 형태이다.

- K-class의 경우
  - $a_k(\mathbf{x}) = \ln(p(\mathbf{x}/C_k)p(C_k)) = \mathbf{w}_k^T\mathbf{x} + w_0$
  - $\mathbf{w}_k = \Sigma^{-1}\boldsymbol{\mu}_k$
  - $w_{k0} = -\frac{1}{2}\boldsymbol{\mu}_k^T\Sigma^{-1}\boldsymbol{\mu}_k + \ln p(C_k)$
- posterior의 decision boundary는 input space에 linear하다. (공분산이 동일하다는 가정하에서)
- 공분산을 각 class마다 다르다고 가정하면 우리는 quadratic function of $\mathbf{x}$를 얻게 되고 이는 *quadratic discriminant* 이다.

이처럼 posterior probability는

$$p(\mathbf{x}|C_k) = f(linear\ of\ \mathbf{x})$$

의 형태가 되고 이를 generalized linear model이라고 한다.

4.2.2 Maximum likelihood solution

MLE를 통해 prameter들을 추정해보자. class-conditional에서 Gaussian을 가정하였는데 그에 해당하는 parameter들 이다.

- prior : $p(C_1) = \pi, p(C_2) = 1 - \pi$
- $p(x_n, C_1) = p(C_1)p(\mathbf{x}_n/C_1) = \pi N(\mathbf{x}_n/\boldsymbol{\mu}_1, \mathbf{\Sigma})$
- $p(x_n, C_2) = p(C_2)p(\mathbf{x}_n/C_2) = (1-\pi) N(\mathbf{x}_n/\boldsymbol{\mu}_2, \mathbf{\Sigma})$
- Class 1은 1, Class 2는 0 으로 target coding
- likelihood function :

$$p(\mathbf{t}|\pi, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \mathbf{\Sigma}) = \prod [\pi N(\mathbf{x}_n|\boldsymbol{\mu}_1, \mathbf{\Sigma})]^{t_n}[(1-\pi)N(\mathbf{x}_n|\boldsymbol{\mu}_2, \mathbf{\Sigma})]^{1-t_n}$$

이를 log 취하고 미분하여 MLE를 구하면 (K-class도 동일한 방법으로 구할 수 있다)

$$\pi = \frac{1}{N}\sum_{n=1}^{N} t_n = \frac{N_1}{N_1 + N_2}$$

$$\boldsymbol{\mu}_1 = \frac{1}{N_1}\sum_{n=1}^{N} t_n \mathbf{x}_n, \quad \boldsymbol{\mu}_2 = \frac{1}{N_2}\sum_{n=1}^{N} (1-t_n)\mathbf{x}_n$$

$$\mathbf{\Sigma} = \mathbf{S} = \frac{N_1}{N}\mathbf{S}_1 + \frac{N_2}{N}\mathbf{S}_2$$

4.2.3 Discrete features

각 input data feature가 2가지의 값을 갖는 discrete feature들이라고 가정해보자. 그러면 총 2^D의 경우 수가 생긴다. 이를 추정하기에는 너무 복잡하다. 따라서 naive bayes의 가정을 이용하면

$$p(\mathbf{x}|C_k) = \prod_{i=1}^{D} \mu_{ki}^{x_i}(1-\mu_{ki})^{1-x_i}$$

$$a_k(\mathbf{x}) = \ln(p(\mathbf{x}|C_k)p(C_k))$$

$$a_k(\mathbf{x}) = \sum_{i=1}^{D}\{x_i \ln \mu_{ki} + (1-x_i)\ln(1-\mu_{ki})\} + \ln p(C_k)$$

이 또한 linear한 형태이다.

4.2.4 Exponential Family

위에서 알 수 있듯이 input이 Gaussian이던 discrete이던지 우리에게 가장 중요한 posterior class probability는 generalized linear model과 sigmoid, softmax activation function에 의해 결정된다.

- 이러한 특징은 class-conditional density가 exponential family의 경우 해당한다.

$$p(\mathbf{x}\,|\lambda_k) = h(\mathbf{x})g(\lambda_k)\exp(\lambda_k^T u(\mathbf{x}))$$

여기서 제약을 위한 parameter s를 추가하고 (잘 이해못함)

$$p(\mathbf{x}\,|\lambda_k, s) = \frac{1}{s}h\left(\frac{1}{s}\mathbf{x}\right)g\left(\lambda_k\right)\exp\left(\frac{1}{s}\lambda_k^T u(\mathbf{x})\right)$$

linear function을 구할 수 있다.

$$a(\mathbf{x}) = \frac{1}{s}(\lambda_1 - \lambda_2)^T\mathbf{x} + \ln g(\lambda_1) - \ln g(\lambda_2) + \ln p(C_1) - \ln p(C_2)$$

$$a_k(\mathbf{x}) = \frac{1}{s}\lambda_k^T\mathbf{x} + \ln g(\lambda_k) + \ln p(C_k)$$

- link function과 exp fam의 관계
  - EX) Bernoulli dist

$$\begin{aligned} L(\theta) = \prod \theta^{x_i}(1-\theta)^{1-x_i} &= \exp\{\sum x_i \log\theta + \sum(1-x_i)\log(1-\theta)\} \\ &= \exp\{\sum x_i \log(\frac{\theta}{1-\theta})\}(1-\theta)^n \end{aligned}$$

- 위의 식에서 $\log(\frac{\theta}{1-\theta})$ 가 link function이다.
- $\log(\frac{\theta}{1-\theta}) = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n$ 이 이제 배울 logistic regression 이다.

4.3 Probabilistic Discriminative Models

이전과 다르게 parameter 추정을 $p(C_k/x)$에서 Maximum likelihood 를 이용하여 directly 하고자 한다. 이전에 본 generative한 방법에 비해

- parameter가 더 적다
- class-conditional density 가정이 잘못되면 성능이 좋지 않을 수 있다

4.3.1 Fixed basis functions

이제부터는 basis function $\phi(\mathbf{x})$을 사용할 것이다.

- basis function이 비선형이라 decision boudary는 original space에 linear하지 않을 것이다.
- basis function에는 $\phi(\mathbf{x}) = 1$ bias를 기본적으로 넣는다.
- original이 아닌 basis function을 사용했다고 항상 결과가 좋은 것은 아니다.

4.3.2 Logistic regression

2-class의 경우로 시작해보자. 이전에 공부했듯이 일반적인 가정하에서 posterior는 sigmoid에 linear function of ϕ (feature vector) 가 들어간 형태이다.

$$p(C_1|\phi) = y(\phi) = \sigma(\mathbf{w}^T\phi)$$

- logistic regression 의 장점
  - (2 class) M 차원이라고 가정하면 M개의 parameter가 있을 것이다. 반면에 generative한 상황을 생각하면 Gaussian class conditional density 의 경우 2M개의 평균, M(M+1) / 2개의 covariance matrix, prior 까지 총 M(M+5)/2+1 개의 parameter가 필요하다.
  - interpretable하다.

    - parameter estimation에 있어 computationally efficient 하다.
    - multiclass도 가능하다.
- 단점
    - prediction performance가 좋은 편은 아니다.

likelihood로 parameter를 추정하는 과정을 살펴보자.

- Given : $D = [(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), .., (\mathbf{x}_n, y_n)]$
- model : $t_i \sim^{iid} Bern[\sigma(\mathbf{w}^T \phi(\mathbf{x}_i))]$
- $y_n = p(C_1/\phi_n) = \sigma(\mathbf{w}^T \phi(\mathbf{x}_n))$

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

- $\mathbf{t} = (t_1, ...t_N)^T$: true target
- cross-entropy error function :

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum \{t_n \log y_n + (1 - t_n) \log(1 - y_n)\}$$

- $\mathbf{w}$에 대해 미분하면

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

- 이를 구하는 방법은 chain rule을 사용한다. 아래의 값들을 곱하면 위의 식이 나온다.
    - $\frac{\partial E}{\partial y_n} = \frac{1-t_n}{1-y_n} - \frac{t_n}{y_n} = \frac{y_n - t_n}{y_n(1-y_n)}$
    - $\frac{\partial y_n}{\partial a_n} = \frac{\partial \sigma(a_n)}{\partial a_n} = \sigma(a_n)(1 - \sigma(a_n)) = y_n(1 - y_n)$
    - $\frac{\partial a_n}{\partial \mathbf{w}} = \phi_n$
- 이전에 linear regression과는 다르게 MLE가 closed form으로 존재하지 않는다. 따라서 approximation하는 방법이 필요하다.
- 이를 Gradient descent 방법을 통해 답을 구할 수도 있다. 하지만 뒤에서는 약간 다른 방법으로 해결해본다. (전통적인 통계 방법)

4.3.3 Iterative reweighted least squares

+일단은 교재의 내용을 보기 전에 이해를 돕기 위해 추가 설명을 한다.

- 우리는 logL를 미분했을 때, 이를 0으로 만드는 MLE를 찾고싶다.
    - $g'(x)$ 는 미분가능
    - $g''(x) \neq 0$

위의 조건을 만족하는 경우 taylor expansion을 이용하여 (1차 근사)

$$0 = g'(x) \approx g'(x^t) + (x - x^t)g''(x^t)$$

이를 정리하면

$$x = x^t - \frac{g'(x^t)}{g''(x^t)}$$

이제 교재의 내용을 살펴보자.
logisitc regression은 sigmoid function의 non-linearlity 때문에 closed-form의 해를 구할 수 없다. 그래서 우리는 error function의 최소화하는 방법으로 **Newton-Raphson iterative opimization** algorithm을 사용한다.

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

- $\mathbf{H}$: hessian matrix whose elements comprise the second derivatives of $E(\mathbf{w})$ with respect to the component of $\mathbf{w}$

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

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

- $\mathbf{\Phi}^T\mathbf{R}^{1/2}\mathbf{R}^{1/2}\mathbf{\Phi} = (\mathbf{R}^{1/2}\mathbf{\Phi})^T(\mathbf{R}^{1/2}\mathbf{\Phi})$ 이기에 positive semi definite이고 이를 통해 $E(\mathbf{w})$가 convex하다는 것을 알 수 있다.
- $\mathbf{R}$: N*N diagonal matrix with elements $R_{nn} = y_n(1-y_n)$
  - y_n의 식이므로 parameter $\mathbf{w}$에 dependent하다. 따라서 $\mathbf{R}$에 대해서도 iterative하게 업데이트가 필요하다.
- 아래처럼 iterative하게 parameter를 업데이트 한다.

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

- where $\mathbf{z} = \mathbf{\Phi}\mathbf{w}^{(old)} - \mathbf{R}^{-1}(\mathbf{y}-\mathbf{t})$
- 마지막 줄을 보면 이 형태는 weighted least-square 문제에서의 normal equation의 형태이다. 하지만 $\mathbf{R}$이 상수가 아니기에 iterative하게 답을 구해야 하고 이러한 이유로 *iterative reweighted least square* 라고 부른다.
- $\mathbf{R}$의 대각성분을 variance라고 해석할 수도 있다.
  - 대각성분이 $y_n(1-y_n)$ 인데 이는 t_n의 variance이기 때문이다.

4.3.4 Multiclass logistic regression

위에서 본 binary와 똑같이 할 수 있다. multiclass에서는 softmax function을 이용한다.

$$p(C_k|\phi) = y_k(\phi) = \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

likelihood function을 구하면

$$p(\mathbf{T}|\mathbf{w}_1, ...\mathbf{w}_K) = \prod_{n=1}^{N}\prod_{k=1}^{K} p(C_k|\phi_n)^{t_{nk}} = \prod_{n=1}^{N}\prod_{k=1}^{K} y_{nk}^{t_{nk}}$$

negative log를 취하면

$$E(\mathbf{w}_1, ..., \mathbf{w}_K) = -\ln p(\mathbf{T}|\mathbf{w}_1, ..., \mathbf{w}_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\ln(y_{nk})$$

똑같이 미분을 취하고 Gradient descent나 IRLS 방법을 통해 parameter를 추정한다.

$$\nabla_{\mathbf{w}_j} E(\mathbf{w}_1, ..., \mathbf{w}_K) = \sum_{n=1}^{N}(y_{nj} - t_{nj})\phi_n$$

4.3.5 Probit regression

이전과 마찬가지로 generalized linear model의 형태

$$p(t=1|a) = f(a) = f(\mathbf{w}^T\phi)$$

를 유지하지만 조금 다른 activation function을 알아보자.

- link function으로 noisy threshold model을 생각해보면

$$\begin{cases} t_n = 1 & if\ a_n \geq \theta \\ t_n = 0 & otherwise \end{cases}$$

θ는 random variable이고 probability density가 $p(\theta)$라고 하자. 이에 따라 activation function을 CDF형태

$$f(a) = \int_{-\infty}^{a} p(\theta) d\theta$$

로 표현할 수 있다. probability density를 $N(0,1)$로 가정하면

$$\Phi(a) = \int_{-\infty}^{a} N(0,1) d\theta$$

이고 이를 *probit* function이라고 한다. (new activation function!) 이를 모델에서 사용할 때는 약간 다른 모습을 이용한다. *erf function* 은

$$erf(a) = \frac{2}{\sqrt{\pi}} \int_{0}^{a} \exp(-\theta^2) d\theta$$

이를 통해 탄생한 generalized linear model을 *probit regression* 이라고 한다.

$$\Phi(a) = \frac{1}{2} \left\{ 1 + erf\left(\frac{a}{\sqrt{2}} \right) \right\}$$

- logistic, probit regression 모두 outlier에 취약한 편이다.
  - 근데 probit은 $exp(-x^2)$이 있어서 더 취약하다.
- data가 mislabelling된 경우, 새로운 probability ϵ을 추가하여 사용할 수 있다.

$$p(t|\mathbf{x}) = (1-\epsilon)\sigma(\mathbf{x}) + \epsilon(1-\sigma(\mathbf{x})) = \epsilon + (1-2\epsilon)\sigma(\mathbf{x})$$

4.4 The Laplace Approximation

4.5장에서 logistic regression에 Bayesian 방법을 사용하고자 한다. 근데 $\mathbf{w}$의 posterior가 더 이상 Gaussian이 아니기 때문에 integrate하기가 어렵다. 따라서 특정 범위에 있는 함수를 Gaussian으로 approximation하는 방법을 이용하고자 한다. 먼저 single variable의 경우부터 살펴보자.

- Suppose the distribution $p(z)$ is defined by

$$p(z) = \frac{1}{Z} f(z), \quad Z = \int f(z) dz$$

우리의 목표는 $p(z)$의 mode를 중앙(평균)으로 갖는 Gaussian distribution을 approximation하는 것이다.

- 먼저, mode를 찾아야한다.

$$p'(z_0) = 0$$

- Taylor expansion

$$\ln f(z) \simeq \ln f(z_0) - \frac{1}{2} A (z - z_0)^2$$

$$A = - \left. \frac{d^2}{dz^2} \ln f(z) \right|_{z=z_0}$$

따라서

$$f(z) \simeq f(z_0) \exp\{ -\frac{A}{2} (z - z_0)^2 \}$$

$$q(z) = (\frac{A}{2\pi})^{1/2} \exp\{-\frac{A}{2}(z-z_0)^2\}$$

- 우리는 $p(z)$를 approximate한 Gaussian $q(z)$를 찾을 수 있다! 이 과정이 *Laplace approximation* 이다.
- Gaussian approximation에서 ($f(z)$를 두 번 미분하여 z_0를 대입) precision A는 양수이다. 따라서 z_0는 local maximum이다.

이제 다차원의 형태로 살펴보자.

- Hessian Matrix $\mathbf{A} = - \bigtriangledown \bigtriangledown \ln f(\mathbf{z}_0)$
- $f(\mathbf{z}) \simeq f(\mathbf{z}_0) \exp\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^T \mathbf{A} (\mathbf{z}-\mathbf{z}_0)\}$

$$q(\mathbf{z}) = \frac{|\mathbf{A}|^{1/2}}{(2\pi)^{M/2}} \exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^T \mathbf{A} (\mathbf{z}-\mathbf{z}_0)\right\} = N(\mathbf{z}_0, \mathbf{A}^{-1})$$

- Laplace approximation 특징
  - Mutimodal인 distribution은 다양한 Laplace approximation이 생길 수 있다.
  - CLT에 의해 Laplace approximation은 data가 많을수록 좋다.
  - 위에서 알 수 있는이 Z에 대해 알 필요가 없다.
  - Gaussian에 기반하므로 실수 변수에만 사용이 가능하다.
  - global한 특징을 잡기 어렵다.

4.4.1 Model comparison and BIC

normalization constraint Z에 대해 approximation해보자.

$$Z = \int f(\mathbf{z}) d\mathbf{z} \simeq f(\mathbf{z}_0) \int \exp\left\{\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^T \mathbf{A} (\mathbf{z}-\mathbf{z}_0)\right\} d\mathbf{z} = f(\mathbf{z}_0) \frac{(2\pi)^{M/2}}{|\mathbf{A}|^{1/2}}$$

우리는 이 결과를 통해 이전에 공부했던 Bayesian model comparison에서 model evidence를 approximation해볼 것이다.

- model evidence $p(D/M_i)$
  - M_i 생략

$$p(D) = \int p(D|\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}$$

- 아래와 같이 정의하고 우리는 model evidence를 approximation하면
  - $f(\boldsymbol{\theta}) = p(D/\boldsymbol{\theta}) p(\boldsymbol{\theta})$
  - $Z = p(D)$

$$\ln p(D) \simeq \ln p(D|\boldsymbol{\theta}_{MAP}) + \ln p(\boldsymbol{\theta}_{MAP}) + \frac{M}{2} \ln(2\pi) - \frac{1}{2} \ln |\mathbf{A}|$$

- 첫번째 term은 log likelihood evaluated using the optimized parameters
- 두번째 term부터 마지막 term까지 *Occam factor* 라고 부른다.
  - penalizes model complexity
- $\boldsymbol{\theta}_{MAP}$: mode of posterior distribution
- $\mathbf{A}$: Hessian matrix

$$\mathbf{A} = -\nabla\nabla p(D|\boldsymbol{\theta}_{MAP}) p(\boldsymbol{\theta}_{MAP}) = -\nabla\nabla \ln p(\boldsymbol{\theta}_{MAP}|D)$$

model evidence를 approximation한 식에서

- Gaussian prior가 broad하고
- Hessian이 full rank이면

우리는 해당 식을 더 간단하게 (의미없는 상수 생략)

$$\ln p(D) \simeq \ln p(D|\theta_{MAP}) - \frac{1}{2}M \ln N$$

이는 *BIC(Baysian Information Criterion)* 이다.

- M은 parameter의 갯수, N은 data의 수를 의미한다.
- AIC보다 더 간단한 모델을 추구한다.
- BIC를 쉽게 계산할 수 있지만 full rank라는 가정이 만족하기 쉽지 않아서 한계가 존재한다.

4.5 Bayesian Logistic Regression

Logistic regression에 Bayesian적으로 접근해보자.

4.5.1 Laplace approximation

일단 prior는 Gaussian으로 가정한다.

$$p(\mathbf{w}) = N(\mathbf{m}_0, \mathbf{S}_0)$$

이제 posterior를 구해보자.

$$p(\mathbf{w}|\mathbf{t}) \propto p(\mathbf{w})p(\mathbf{t}|\mathbf{w})$$

양변에 log를 취하면

$$\ln p(\mathbf{w}|\mathbf{t}) = -\frac{1}{2}(\mathbf{w}-\mathbf{m}_0)^T\mathbf{S}_0^{-1}(\mathbf{w}-\mathbf{m}_0) + \\ \sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\} + const$$

posterior에 대한 Gaussian approximation하였다고 가정하자. maximize하는 parameter를 $\mathbf{w}_{MAP}$라고 하고 covariance는

$$\mathbf{S}_N^{-1} = -\nabla\nabla \ln p(\mathbf{w}|\mathbf{t}) = \mathbf{S}_0^{-1} + \sum_{n=1}^{N} y_n(1-y_n)\phi_n\phi_n^T$$

따라서 Gaussian approximation한 posterior distribution의 form은

$$q(\mathbf{w}) = N(\mathbf{w}_{MAP}, \mathbf{S}_N)$$

이제 approximation하여 구한 posterior로 Predictive를 구해보자.

4.5.2 Predictive distribution

2-class의 경우라고 가정하자. predictive distribution은

$$p(C_1|\phi, \mathbf{t}) = \int p(C_1|\phi, \mathbf{w})p(\mathbf{w}|\mathbf{t}) \simeq \int \sigma(\mathbf{w}^T\phi)q(\mathbf{w})d\mathbf{w}$$

- Funtion $\sigma(\mathbf{w}^T\phi)$ depends on w only through tis projection onto ϕ
- (교재에 설명이 다소 빈약) 그냥 아래처럼 변형

$$\sigma(\mathbf{w}^T\phi) = \int \delta(a - \mathbf{w}^T\phi)\sigma(a)da$$

이를 predictive distribution에 대입하면

$$\int \sigma(\mathbf{w}^T\phi)q(\mathbf{w})d\mathbf{w} = \int \sigma(a)p(a)da$$

$$p(a) = \int \delta(a - \mathbf{w}^T\phi)q(\mathbf{w})d\mathbf{w}$$

- $p(a)$는 Gaussian distribution이 되는데
  - delta function (δ) imposes a linear constraint on $\mathbf{w}$이고
  - $q(\mathbf{w})$ 는 정의에 의해 Gaussian distribution
  - Gaussian의 marginal도 Gaussian

$$\mu_a = E[a] = \int p(a)ada = \int q(\mathbf{w})\mathbf{w}^T\phi d\mathbf{w} = \mathbf{w}_{MAP}^T\phi$$

$$\sigma_a^2 = var[a] = \int p(a)\{a^2 - E[a]^2\}da = \int q(\mathbf{w})\{(\mathbf{w}^T\phi)^2 - (\mathbf{m}_N^T\phi)^2\}d\mathbf{w} = \phi^T\mathbf{S}_N\phi$$

따라서 predictive distribution은

$$p(C_1|\mathbf{t}) = \int \sigma(a)p(a)da = \int \sigma(a)N(\mu_a, \sigma_a^2)da$$

sigmoid-gaussian을 analytically 구할 수 없기 때문에 이 또한 approximation을 해야한다. sigmoid와 비슷한 모양을 가지는 Probit function을 이용한다. ($\sigma(a) \approx \Phi(\lambda a), \lambda^2 = \pi/8$)

- probit function을 이용한 approximation의 장점은 Gaussian과 만나서 analytically 또 probit function으로 아래와 같은 결과가 나온다.

$$\int \Phi(\lambda a)N(\mu, \sigma^2)da = \Phi(\frac{\mu}{(\lambda^{-2} + \sigma^2)^{1/2}})$$

$$\int \sigma(a)N(\mu, \sigma^2)da \simeq \sigma(k(\sigma^2)\mu)$$

- $k(\sigma^2) = (1 + \pi\sigma^2/8)^{-1/2}$

최종 결과 approximate predictive distribution은

$$p(C_1|\phi, \mathbf{t}) = \sigma(k(\sigma_a^2)\mu_a)$$