Yonsei University, Department of Applied Statistics

Chapter 4 Classification

YBIGTA Science Team 신보현

January 30, 2019

4. Classification

대부분의 변수는 수치형 (qualitative) 보다는 범주형 (quantitative) 일 때가 많다. 해당 챕터에서는 분류 모델인 logistic resgression, linear discriminant analysis 그리고 K-nearest neighbors을 살펴본다.

4.1 An overview of Classification

regression setting과 마찬가지로 classification도 지도학습 (supervised learning) 의 한 종류이다.

4.2 Why Not Linear Regression?

어떤 환자의 증상에 따라서 환자의 의학적 상태를 예측한다고 하자. 이 간단한 예제에서 세가지 진단이 있다: stroke, drug overdose, epileptic seizure

$$Y = \begin{cases} 1 & \text{if stroke} \\ 2 & \text{if drugoverdose} \\ 3 & \text{if epileptic seizure} \end{cases}$$

예측을 하기 위해 linear regression 모델으로 위와 같이 생각하여 예측하려고 한다면 두 값의 차이가 1로 같으므로 stroke와 drug overdose의 차이점이 drug overdose와 epileptic seizure와의 차이와 같다는 것을 내포한다. 하지만 실제에서는 이런 차이가 성립하지 않을 것이다.

반면, binary qualitative 문제에 관해서는 dummy variable을 만들면 되기 때문에 상황이 조금 간단하다.

$$Y = \begin{cases} 0 & \text{if stroke} \\ 1 & \text{if drugoverdose} \end{cases}$$

이 모델을 선형 회귀 모델에 적합시켜서 $\widehat{Y} > 0.5$라면 stroke로 판단할 수 있다. 즉, fitted value인 $\widehat{Y}$를 '확률'의 개념으로 생각하여 분류를 하는데, 이 모델을 선형회귀에 적합시키면 0과 1사이의 범위를 벗어나 해석이 어려워질 수 있다.

4.3 Logistic Regression

4.3.1 The Logistic Model

Logistic Model은 기본적으로 $p(X) = \Pr(Y = 1|X)$의 관계를 이용해서 모델을 적합한다. 그렇다면 어떻게 적합할까? 4.2 section에서는 $p(X) = \beta_0 + \beta_1 X$ 를 이용해서 예측을 했지만

이것은 0 이하의 값 또는 1이상의 값이 도출되는 단점이 있었다. 이러한 문제를 피하기 위해서 우리는 0과 1사이의 값을 도출시키는 $p(X)$ 함수를 이용한다. logistic regression 에서는 이러한 함수를 logistic function이라고 부른다

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1 + e^{\beta_0+\beta_1 X}} \tag{4.2}$$

위의 모델을 적합시키기 위해서 maximum likelihood라는 방법을 사용한다. 우리는 이제 logistic function을 통해 0근처로는 가지만 0이하로 가지는 않고 1근처로는 가지만 1이상으로 가지는 않는다. logistic function은 X의 값에 상관없이 항상 0과 1사이의 값만 도출하며 S 모양을 가진다.

조금의 변형을 통해 아래의 식을 얻는다.

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0+\beta_1 X} \tag{4.3}$$

$p(X)/[1-p(X)]$는 odds라고 불리고 0과 ∞사이의 값을 가질 수 있다. 0과 ∞ 에 가까이 있는 값은 매우 낮은 확률과 매우 높은 확률을 뜻한다. $p(X)$가 1에 가까워 질수록 분모는 0에 가까워지고 $p(X)/[1-p(X)]$는 ∞에 가까워진다. $p(X)$가 0에 가까워 질수록 $p(X)/[1-p(X)]$는 0에 가까워진다.

로그를 취함으로써 다음의 결과를 얻는다.

$$log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X \tag{4.4}$$

좌변은 log-odds 또는 logit이라고 불린다. 위 식에서 볼수 있듯이, log-odds는 X에 linear 하다.

resgression model에서 β_1은 하나의 unit이 X에서 증가함에 따른 Y의 평균 변화량을 뜻한다. 하지만 logistic regression에서는 X가 하나의 unit이 증가함에 따라서 β_1의 log odds를 변화시킨다. 또는 동일한 의미로 odds를 e^{β_1} 만큼 곱한다. 하지만 $p(X)$와 X가 선형의 관계가 아니기 때문에 β_1는 X의 한단위 증가에 따른 $p(X)$의 증가를 의미하지는 않는다. 하지만 X의 값에 상관 없이 β_1이 양수라면 X의 증가가 $p(X)$의 증가와 관련 있을 것이고 반대의 경우도 마찬가지이다.

4.3.2 Estimating the Regression Coefficients

4.2에서의 β_0과 β_1은 알려지지 않으므로 training data에 의해서 estimate 되어야 한다. lest squares를 사용할 수도 있지만 더 보편적인 방법은 maximum likelihood이다. MLE의

기본적인 컨셉은 다음과 같다; 우리는 predicted probability $\widehat{p}(x_i)$가 4.2를 사용하여 얻은 개별 observed default status와 가장 가까운 값에 대응하게 해주는 β_0와 $\beta_1$estimates를 찾는다. 이러한 직관은 likelihood function이라는 식을 사용해서 표현한다.

$$l(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'})) \tag{4.5}$$

$\hat{\beta}_0$과 $\hat{\beta}_1$은 이 likelihood function을 maximize하는 값이다.

4.3.3 Making Predictions

계수들이 estimate 되었다면 확률을 계산하는 것은 X 값을 그냥 대입하면 된다. qualitative 변수를 예측하고 싶으면 dummy variable로 만들면 된다. 예를 들어 결과로 $\hat{\beta}_0 = -3.5041, \hat{\beta}_1 = 0.4049$가 나왔다고 하자. estimated coefficient을 이용하여 로지스틱 모델을 다시 써보면, 아래와 같다.

$$\widehat{p}(X) = \frac{e^{-3.5041+0.4049X}}{1 + e^{-3.5041+0.4049X}}$$

4.3.4 Multiple Logistic Regression

이제 우리는 다중 변수를 사용하여 binary response를 예측하는 문제를 생각해보자. 우리는 4.4를 사용하여 다음과 같이 일반화 할 수 있다.

$$log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X + \cdots + \beta_p X_p \tag{4.6}$$

따라서 다음과 같이 다시 바꿔서 쓸 수 있다.

$$p(X) = \frac{e^{\beta_0+\beta_1 X+\cdots+\beta_p X_p}}{1 + e^{\beta_0+\beta_1 X+\cdots+\beta_p X_p}} \tag{4.7}$$

4.3.2에서 사용했듯이 우리는 $\beta_0, \cdots, \beta_p$를 예측하기 위해 maximum likelihood method를 사용한다.

하지만 종종 하나의 변수로 회귀를 실시하는 것은 다수의 변수로 회귀를 실시하는 것과 상충되는 결과를 일으키는데 이것은 하나의 변수로 회귀를 실시할 때 영향을 주는 다른 변수들이 고려되지 않기 때문이다. 다시 말하면, 단변수 로지스틱 회귀에서는 다른 변수들의 효과를 아예 무시하는 반면에, 다변서 로지스틱 회귀에서는 다른 변수들의 효과를 고정(fixed, adjusted)한채, 각 변수들의 효과(estimated coefficient)을 살펴본다. 따라서 변수 하나만으로 로지스틱 회귀를 시행하는 것은 꽤 부정확한 결과를 가져올 수 있다(이는 회귀 분석에서도 동일하게 적용이 된다.) 이러한 현상을 confounding이라고 부른다.

4.3.5 Logistic Regression for >2 Response Classes

우리는 종종 두개 이상의 class를 가지는 반응 변수를 분류하고 싶어한다. 보통 이를 위해 이전 장에서 논의한 two-class logistic regression을 multiple-class로 확장하기도 하는데 이것보다는 discriminant analysis가 multiple-class classification에 더 보편적으로 사용된다.

4.4 Linear Discriminant Analysis

Logistic regression은 $Pr(Y = K \mid X = x)$ 을 4.7에 주어져 있는 logistic function을 이용하여 모델링 하는 것을 포함한다. 즉 X 라는 변수가 주어졌을 때, Y 라는 반응 변수의 조건부 분포를 모델링 한다. 우리는 이제 대안적이고 덜 직접적인 접근방법을 고려한다. 이러한 대안적인 접근방법에서 우리는 Y 의 각각의 class에 대한 예측 변수 X 의 분포(= $Pr(X = x \mid Y = k)$)를 따로 모델링한 후 Bayes' theroem을 이용하여 $Pr(Y = K \mid X = x)$ 을 estimate 한다. logistic regression이 있음에도 왜 다른 방법을 쓸까? class가 잘 퍼져 있을 때, logistic regession model을 위한 parameter estimates는 놀랍게 불안정하다. Linear discriminant는 이러한 문제를 해결해준다. 만약 n이 작고 각각의 class에 대한 X 의 분포가 정규분포로 근사한다면 *linear discriminant model*은 logistic regression보다 더 안정적이다.

4.4.1 Using Bayes' Theorem for Classification

우리가 어떠한 관측치를 K개의 classes로 분류한다고 생각하자. π_k 를 랜덤하게 선택된 관측치가 kth 클래스로부터 올 확률이라고 하자. 이것은 어떠한 관측치가 k번째 범주로 분류될 확률이다.

$$f_k(x) = Pr(X = x \mid Y = k)$$

위의 식을 하나의 관측치가 k번째 class에서 나올 X에 대한 density function이라고 정의하자. 다시 말해서, 높은 $f_k(x)$ 는 어떤 관측치가 k번째 class에 해당될 가능성이 높다는 것이다. $f_k(x)$ 를 Bayes' theorem에 적용해 보자. 그 이전에, Bayes' theorem에 대해서 잠시 살펴보고 가자.

어떤 통계학자가 한 분포가 어떤 모양인지에 대해서 관심이 있다고 하자. 그런데 분포는 모수 (parameter) 로 표현이 된다.(통계학 입문 책에서는 모수를 정의하기를, 분포를 잘 나타내주는 것이라고 하는데, 이는 수학적으로 엄밀한 표현은 아니다. 분포에 대한 1 to 1 대응이 모수라고 하는 것이 더 엄밀한 정의이다.) 이에 따라, 모수에 관심이 생기는 것은 자연스럽다. 그런데 통계학의 두 흐름 중 하나인 빈도론자 입장에서는 모수란, unknown but fixed한 존재이다. 하지만 베이지안 입장에서는 모수란 unknown therefore random인 존재다. 예를 들어, 회귀 분석을 시행할 때, $y = \beta_0 + \beta_1 x + \epsilon$의 모델에서 β_0, β_1 은 모수이고 unknown but fixed라고 배웠을 것이다. 이는 빈도론자의 입장이다. 베이지안은 모르니까

random이라고 생각하는 것이고 random이라는 것은 어떠한 값을 가질지 사전에 100% 예측이 불가능한 것이므로 모수를 또다른 random variable로 보는 것이다. 확률 변수이니 확률밀도함수가 있는 것이 당연하고, 베이지안은 데이터가 주어졌을 때, 모수의 분포에 아주 많은 관심이 있다.
위에서 말한, 데이터가 주어졌을 때, 모수의 분포를 사후 분포(Posterior Distribution)이라고 하고 $P(\theta \mid D)$로 표현한다. 여기서 D는 데이터를, θ는 관심있는 parameter를 의미한다. 한편, 사후 분포는 조건부 분포이기도 하다. 따라서 이를 조건부 분포의 정의에 따라서 다음과 같이 표현할 수 있다.

$$\begin{aligned} P(\theta \mid D) &= \frac{P(\theta, D)}{P(D)} \\ &= \frac{P(\theta)P(D \mid \theta)}{P(D)} \\ &= \frac{P(\theta)P(D \mid \theta)}{\int P(D, \theta) d\theta} \; for\, continuous\, variable \\ &= \frac{P(\theta)P(D \mid \theta)}{\sum_{for\, all\, possible\, \theta} P(D, \theta)} \; for\, discrete\, variable \end{aligned}$$

여기서 $P(\theta)$는 사전 분포(Prior Distribution), $P(D \mid \theta)$는 직역하면, 모수가 주어졌을 때(given) 데이터의 분포인데 이는 Likelihood 함수에 비례한다. 즉, $P(D \mid \theta) \propto P(\theta \mid D)$이다. 따라서 $P(D \mid \theta)$을 Likelihood 함수라고 부르기도 한다. 이를 그대로 적용해보자.

$$P(Y = k \mid X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)} \tag{4.10}$$

위식의 좌변에 있는 $P(Y = k \mid X = x)$은 데이터가 주어졌을 때 Y가 k일 확률이다. 이를 베이즈 정리를 이용하여 아래와 같이 표현할 수 있다.

$$\begin{aligned} P(Y = k \mid X = x) &= \frac{P(Y = k)P(X = x \mid Y = k)}{P(X = x)} \;\; \because Bayes' Thm \\ &= \frac{P(Y = k)Pr(X = x \mid Y = k)}{P(X = x \mid Y = 1) + P(X = x \mid Y = 2) + \cdots + P(X = x \mid Y = k)} \\ &= \frac{P(Y = k)P(X = x \mid Y = k)}{P(Y = 1)P(X = x \mid Y = 1) + \cdots + P(Y = k)P(X = x \mid Y = k)} \\ &= \frac{P(Y = k)P(X = x \mid Y = k)}{\sum_{l=1}^{k} P(Y = l)P(X = x \mid Y = l)} \\ &= \frac{\pi_k f_k(x)}{\sum_{l=1}^{k} \pi_l f_l(x)} \end{aligned}$$

베이즈 정리의 관점에서, $f_k(x)$가 Likelihood 함수, π_k가 prior 분포이고 $P(Y = k \mid X = x)$가 posterior 분포이다. 즉, 분류 문제는 어떤 데이터가 주어 졌을 때 y가 어떤 class에 속할

확률이 궁금한 것이고 LDA에서는 이를 모델링하기 위해 직접적으로 $P(Y = k \mid X = x)$을 추정하기 보다는, 베이즈 정리를 이용하여 한번 변형을 한 후, 간접적으로 추론하는 것이다. 우리는 앞으로 $p_k(X) = Pr(Y = k \mid X)$라고 축약할 것이다. 이것은 직접적으로 $p_k(Y)$를 계산하는 대신에 심플하게 π_k와 $f_k(x)$를 대입함으로써 얻을 수 있다. 보통 π_k에 대한 estimate은 k번째 class에 속하는 training 관측치의 비율을 계산하면 된다. 하지만 $f_k(x)$를 estimate하는 것은 함수를 추정하는 것이기에 어려운 문제이다.
우리는 ch2에서 Bayes classifier가 모든 classifiers중에서 가장 낮은 error rate를 가진다고 배웠다. 따라서 우리는 $f_k(x)$를 estimate하는 방법만 찾으면 Bayes classifier와 근사하는 classifier를 만들 수 있다. 바로 다음 section에서 자세히 살펴보자.

4.4.2 Linear Discriminant Analysis for p = 1

현재 우리는 한개의 predictor을 가지고 있다고 생각하자. 우리는 π_k를 estimate하기 위해 4.10에 대입할 수 있는 $f_k(x)$의 estimate을 얻고 싶다. 우리는 먼저 하나의 관측치를 $p_k(x)$가 최대가 되는 class에 분류할 것이다. $f_k(x)$를 estimate하기 위해서 우리는 먼저 그것의 형태에 대한 가정을 한다.

$f_k(x)$가 normal 또는 Gaussian이라고 가정하자. normal density에서는 다음과 같은 형태를 취한다.

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} exp\left(-\frac{1}{2\sigma_k^2}(x-\mu_k)^2\right) \tag{4.11}$$

여기서 μ_k와 σ_k^2는 k번째 class의 parmeter의 mean과 variance를 나타낸다. 여기서는 $\sigma_1^2 = \cdots = \sigma_K^2$라고 가정하자. 즉, k개의 클래스는 variance를 공유한다. 이것을 우리는 σ^2라고 나타낸다. 4.11를 4.10에 대입함으로써 다음과 같은 결과를 얻는다

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{1}{2\sigma^2}(x-\mu_k)^2\right)}{\sum_{l=1}^{K} \pi_l \frac{1}{2\sigma^2}\left(-\frac{1}{2\sigma^2}(x-\mu_l)^2\right)} \tag{4.12}$$

(여기서 π_k는 k번째 class에 하나의 관측치가 배정될 확률을 의미한다. 3.14159와 혼동하지 말 것)
이는 LDA의 중요한 가정이다. 각 클래스별로, 데이터에 대한 분포가 정규분포라는 점, 그리고 k개의 클래스는 공통의 분산을 공유한다는 점은 LDA를 시행하기 전에 반드시 짚고 넘어가야 하는 가정이다. 회귀 분석에서, 오차항에 대해 iid 가정을 하고, 이후 가정을 충족하는지 살펴보았던 것 처럼, LDA에서도 동일하게 확인 과정이 있어야 한다.

Bayes classifier는 하나의 관측치 $X = x$를 4.12가 최대가 되는 곳에 배정한다. 4.12에 log를 취하고 항들을 정리하면 $p_k(x)$가 최대인 클래스에 분류하는 것은 아래의 $\delta_k(x)$가 최대인

클래스에 분류하는 것과 같다.

$$\delta_k(x) = x \cdot \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + log(\pi_k) \tag{4.13}$$

예를들어 K = 2이고 $\pi_1 = \pi_2$일 때, $\delta_1(x) = x \cdot \frac{\mu_1}{\sigma^2} - \frac{\mu_1^2}{2\sigma^2} + log(\pi), \delta_2(x) = x \cdot \frac{\mu_2}{\sigma^2} - \frac{\mu_2^2}{2\sigma^2} + log(\pi)$이고 $\delta_1(x) > \delta_2(x)$, 즉 $2x(\mu_1 - \mu_2) > \mu_1^2 - \mu_2^2$라면 class 1로 배치할 것이고 그렇지 않다면 2로 배치할 것이다. 그리고 이 경우에 Bayes decision boundary는 다음과 같다.

$$x = \frac{\mu_1^2 - \mu_2^2}{2(\mu_1 + \mu_2)} = \frac{\mu_1 + \mu_2}{2} \tag{4.14}$$

현실에서는 $\mu_1, \cdots, \mu_k, \pi_1, \cdots, \pi_k, \sigma^2$를 모르기 때문에 이것을 estimate해야 하는 단점이 있다. LDA는 이러한 unkown parameter에 estimates를 대입함으로써 Bayes classifier를 근사시킨다. 아래는 사용되는 estimates이다.

$$\hat{\mu}_{\mathrm{k}} = \frac{1}{n_k} \sum_{i:y_i=k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2 \tag{4.15}$$

여기서 n은 training 관측치의 갯수이고n_k는 k번째 class의 training 관측치의 갯수이다. μ_k에 대한 estimates는 단순히 kth class의 training 관측치의 평균인 반면 $\hat{\sigma^2}$는 각각의 K class의 sample variances의 가중 평균이라고 볼 수 있다. 가끔 우리는 각 class의 확률, 즉 $\pi_1, \cdots, \pi_k$에 대한 사전 지식 이 있다면 이를 활용하면 되고 이러한 추가적인 정보가 없다면 LDA는 π_k를 training 관측치에 대한 kth class의 비율로 estimate한다. 즉, 다시 말해

$$\hat{\pi}_k = n_k/n \tag{4.16}$$

LDA는 4.15 와 4.16를 4.13에 대입 한 후 관측치 $X = x$를 4.17가 최대가 되는 class에 배정한다.

$$\hat{\delta}_k(x) = x\frac{\hat{\mu}_k}{\hat{\sigma}^2} - \frac{\hat{\mu}_k^2}{2\hat{\sigma}^2} + log(\hat{\pi}_k) \tag{4.17}$$

LDA에서 linear하다는 것은 4.17에서 $\hat{\delta}_k(x)$가 x에 대해서 선형이라는 사실에서 나온다. 종합하면 LDA는 각각의 class에 있는 관측치들이 class-specific mean과 common variance σ^2

를 가지는 normal distribution에서 나온다고 가정하고 이러한 estimates를 Bayes classifier에 대입한다. 4.4.4에서는 각 class가 다른 variance, 즉 σ_k^2를 가진다는 가정으로 바꾼다.

4.4.3 Linear Discriminant Analysis for p > 1

이제 LDA classifier를 multiple predictors의 경우로 확장해보자. 이를 위해 $X = (X_1, \cdots, X_p)$가 class-specific mean과 common covariance matrix를 가지는 multi-variate Gaussian(or multivariate normal) distribution에서 나온다고 하자. 이러한 분포에 대한 간략한 review를 먼저 한다.

multivariate Gaussian distribution은 각 개별의 predictor는 4.11와 같은 one-dimensional normal distribution을 따른다고 가정한다.

p차원의 random variable X가 multi-variate Gaussian distribution을 따르는 것을 우리는 $X \sim N(\mu, \Sigma)$라고 나타내고 $E(X) = \mu$는 X의 p개의 components를 가지는 mean이고 $Cov(X) = \Sigma$는 $p \times p$차원이 covariance matrix이다. multivariate Gaussian density는 다음과 같이 정의된다.

$$f(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right) \tag{4.18}$$

$p > 1$인 predictors의 케이스에서, LDA는 kth class의 관측치는 multivariate Gaussian distribution $N(\mu_k, \Sigma)$에서 뽑는다고 가정한다. 4.18 를 4.10에 대입한 후 약간의 계산을 하면 Bayes classifier는 하나의 관측치를 다음의 식이 최대가 되는 class에 배정을 한다.

$$\sigma_k(x) = x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + log\pi_k \tag{4.19}$$

이것은 4.13의 vector/matrix 버전이다.

다시 한번, unknown parameters $\mu_1, \cdots, \mu_k, \pi_1, \cdots, \pi_k, \Sigma$ 를 estimate해야 한다. 공식은 4.15 와 유사하다. 하나의 새로운 관측치 $X = x$를 배정하기 위해 이러한 estimates를 4.19에 대입하고 $\hat{\sigma_k}$가 최대가 되는 class에 분류를 한다. 4.19에서 σ_k는 x의 linear function임을 주목하자. 다시 말해서 LDA decision rule은 오로지 x에 linear하게 의존한다. 이것이 바로 LDA에서 linear가 들어가는 이유이다. 145쪽부터.

LDA는 posterior 확률, 즉 $p_k(X)$가 가장 높은 클래스에 데이터를 분류 한다. 이는 두 개의 클래스를 분류하는 문제에서 곧 $P(Y = class\,1 \mid X = x) > 0.5$일 때 클래스 1에 분류하는 것과 동일하다. 하지만 여기서 직접적으로 확률을 구할 수 없기 때문에 베이즈 정리를 이용한 확률에서 estimates을 대입하여 $\widehat{P}(Y = class\,1 \mid X = x)$을 구하는 것이다. 이는

베이즈 분류기(Bayes classifier)의 작동원리와 거의 유사하다. 만약 0.5라는 기준이 높다고 생각하면 이를 0.2로 낮춰 $P(Y = class\,1 \mid X = x) > 0.2$일 때, 클래스 1에 분류하게 맞추면 된다.

		True default status		
		No	Yes	Total
Predicted default status	No	9,432	138	9,570
	Yes	235	195	430
	Total	9,667	333	10,000

위의 confusion matrix는 카드사 데이터에 LDA를 적용하여 신용불량자인 사람들을 분류한 결과이다. LDA 모델이 430명의 신용 불량자를 분류했음을 확인할 수 있다. 실제 333명이 신용분량자인데 이 중 195명을 올바르게 분류했음을 알 수 있다. 여기서 주목해야할 점은, 카드 회사는 신용불량자인데 그렇지 않은 집단으로 분류하는 상황을 신용불량자가 아닌데 신용불량자 집단으로 분류하는 상황보다 더 심각하게 생각할 것이다. 따라서 threshold을 이러한 목적에 부합하도록 변형시킬 필요가 있다.

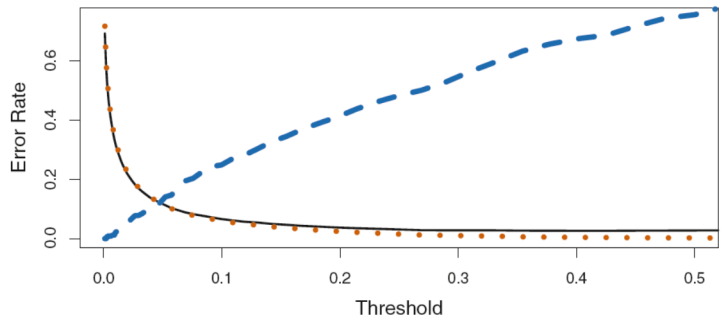

FIGURE 4.7. *For the* `Default` *data set, error rates are shown as a function of the threshold value for the posterior probability that is used to perform the assignment. The black solid line displays the overall error rate. The blue dashed line represents the fraction of defaulting customers that are incorrectly classified, and the orange dotted line indicates the fraction of errors among the non-defaulting customers.*

위 그림은 threshold의 변화에 따른 error rate의 변화를 보여준다. 검정색 선은 overall error rate($= \frac{FN+FP}{P+N}$), 파랑색 선은 잘못 분류된 신용불량자의 비율($= \frac{FN}{FN+TP}$), 오렌지색 점은 잘못분류된 비신용불량자의 비율($= \frac{FP}{FP+TN}$)을 보여준다.
threshold을 0.5로 잡는 것이 overall error rate을 최소화하는 것으로 나타나는데, 이는 베이

즈 분류기가 0.5을 threshold로 사용하고 가장 낮은 overall error rate을 가지기 때문이다. 하지만 이때, 파랑색 선의 error rate은 꽤 높은 것으로 나타난다. 그러면 최적의 threshold을 어떻게 정해야할까? 이러한 결정은 domain knowldege에 기반해야한다(뭔가 다른 기법이 있는 것 같은데 책에서는 이렇게 말하고 넘어간다)

4.4.4 Quadratic Discriminant Analysis

LDA는 각 클래스의 데이터가 정규분포를 따르고 각 클래스 정규분포의 분산이 같다고 가정한다. 하지만 k개의 클래스가 분산이 같은 경우는 그렇게 많지 않을 것이다. 그에 따라서 k개의 클래스가 분산이 다를수도 있다고 좀 더 relax하게 가정하는 것이 QDA이다. LDA는 k개의 클래스가 공통의 분산을 공유한다고 가정하기 때문에 QDA보다 estimate하는 모수의 수가 적다. QDA는 estimate하는 모수의 수가 많고 이는 모델이 더 flexible하다는 말과 동일하다. 2단원에서 배웠듯이, 모델이 flexible하다면 bias가 낮은 대신 variance가 높을 것이다. LDA는 이와는 반대로 덜 flexible하고 variance가 낮은 대신 bias가 상대적으로 높을 것이다. 이를 정리하면, *DLA*는 *training data*가 적어서 *variance*을 줄이는 것이 중요한 상황에 더 적합하고 *QDA*는 *training data*가 많아서 분류기의 *variance*가 큰 걱정이 아니거나 공통의 분산 가정이 분명하게 유지될 것 같지 않은 상황에 더 적합하다.

4.5 A Comparisonof Classfication Methods

해당 챕터에서는 로지스틱 회귀, LDA, QDA, KNN을 다양한 상황에서 적합했을 때의 결과를 비교해본다.
주목할 점은 로지스틱 회귀나 LDA가 밀접하게 연관되어 있다는 점이다. 설명변수가 한 개일때, 로지스틱 회귀에서의 lod odds는 다음과 같다.

$$log\left(\frac{p_1}{1-p_1}\right) = \beta_0 + \beta_1 x$$

LDA의 사후확률 식인 4.12을 조금만 변형해보면 다음과 같다.

$$log\left(\frac{p_1(x)}{1-p_1(x)}\right) = c_0 + c_1 x$$

여기서 c_0, c_1은 μ_1, μ_2, σ의 함수이다. 즉, 예측하는 확률의 형태가 x에 대해서 linear하다는 공통점이 있다. 하지만 LDA는 추가적인 가정이 필요하고 로지스틱 회귀는 그러한 가정이 없었다는 것을 기억하자.
KNN 기법은 비모수적인 방법으로 추가적인 가정을 필요로 하지 않고 QDA는 LDA와 KNN의 중간 지점이라고 볼 수 있다.

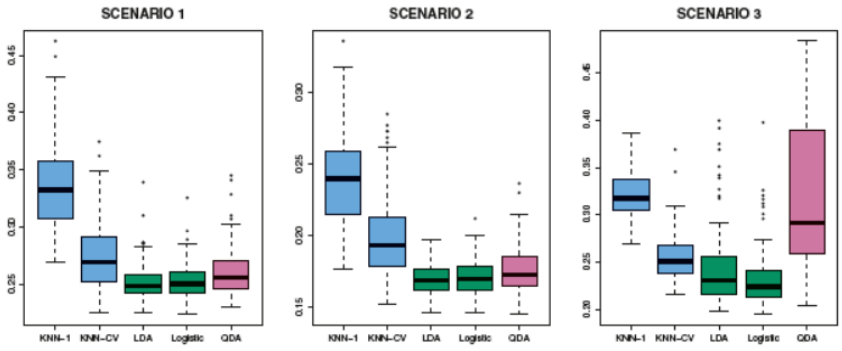

FIGURE 4.10. *Boxplots of the test error rates for each of the linear scenarios described in the main text.*

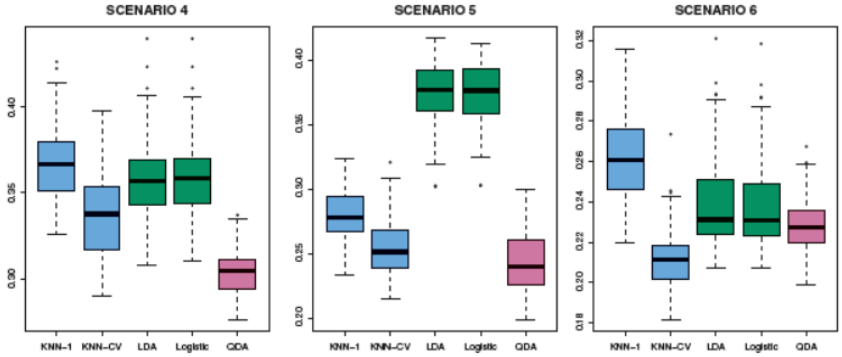

FIGURE 4.11. *Boxplots of the test error rates for each of the non-linear scenarios described in the main text.*

모든상황에서 예측변수는 2개인 데이터이다.

- 상황1: 각 클래스마다 20개의 training data가 있고 각 클래스의 관측치는 서로 다른 평균을 가지는 정규분포에서 random sample로 생성도었다. LDA의 가정이 이러한 상황과 잘 맞으므로 LDA가 가장 좋은 성능을 보였다. 로지스틱 회귀 또한 x에 대한 linear decision boundary을 가정하므로 LDA보다는 살짝 안 좋은 성능을 나타냈다.

- 상황2: 상황1과 유사하고 각 클래스마다 두 개의 예측 변수가 0.5의 상관계수를 갖는 경우이다. 다변량정규분포를 생각해보자. 각 클래스마다 0.5의 상관계수를 가진다면, 각 클래스의 정규분포의 공분산 행렬에서 off-diagonal element 또한 0.5로 동일할 것이다. 따라서 이는 여전히 LDA의 가정에 부합하므로 상황1과 거의 유사한 결과를 보였다.

- 상황3: 두 개의 예측 변수에 대한 관측치를 각 클래스마다 t 분포에서 생성했다. t 분포는 정규분포와 형태는 유사하지만 꼬리 부분이 정규분포보다 두꺼워 extreme

value를 잘 포용한다. 이는 LDA의 가정에 위반되는 상황이므로 로지스틱 회귀가 가장 좋은 성능을 보였다.

- 상황4: 첫 번째 클래스에서는 두 예측변수가 0.5의 상관계수를 가지도록, 두 번째 클래스에서는 -0.5의 상관계수를 가지도록 관측치가 생성되었다. 이는 각 클래스의 정규분포의 공분산 행렬이 다른 경우이므로, QDA의 가정에 부합하는 상황이다. 따라서 QDA가 가장 좋은 성능을 보였다.

- 상황5: 각 클래스마다 uncorrelated한 예측변수들의 다변량정규분포에서 관측치가 생성되었다. 하지만 반응변수가 $X_1^2, X_2^2, X_1 \times X_2$을 사용하여 생성되었다. 이에 따라서 quadartic decision boundary가 생성되었으며 non-linear decision boundary을 만드는 QDA 와 KNN이 linear decision boundary을 만드는 LDA, 로지스틱 회귀보다 좋은 성능을 보였다.

- 상황6: 상황5와 비슷하지만 더 복잡한 메커니즘에 따라서 반응변수가 생성되었다. 그에 따라서 KNN이 가장 좋은 성능을 보였다.

이를 통해서 모든 상황에 대해서 특정 기법이 다른 기법보다 좋다고 말할 수 없음을 확인할 수 있다. 주어진 데이터의 특성에 따라서 좋은 성능을 내는 기법이 달라지는 것이다.
마지막으로 회귀분석에서 non-linear한 term을 추가함으로써 non-linear한 관계를 잘 잡아낼 수 있었음을 상기해보자. 로지스틱 회귀나 LDA에서도 동일하게 non-linear한 term을 추가할 수 있다. 하지만 이렇게 flexibility을 추가함으로써 variance는 올라가지만 그만큼 bias가 많이 내려가면 이러한 추가는 좋은 성능을 낼 것이다. 항상 variance - bias trade off을 생각하며 모델을 설계하자.