

졸업논문

제 목 : 여러가지 방법론을 적용한 고객의 신용도 평가

지도교수 : 이 석 호

2014 년 4 월 30 일

한 국 외 국 어 대 학 교 자 연 과 학 대 학 통 계 학 과

학 번 : 2 0 0 8 0 1 5 3 7 성 명 : 변 희 재 ㉠

학 번 : 2 0 0 9 0 3 8 7 7 성 명 : 황 성 윤 ㉠

여러가지 방법론을 적용한
고객의 신용도 평가

by

Hee-jae Byun

Sung-yoon Hwang

Under the direction

of

Professor Seok-ho Lee

2014. 4. 30

A thesis submitted and approved by the committee of the Department of
Statistics of Hankuk University of Foreign Studies in partial fulfillment of
the requirements for the degree of Bachelor

Thesis Committee : _____

DEPARTMENT OF STATISTICS

< 목 차 >

제1장 서 론

1.1 연구의 배경 및 목적	4
1.2 분석에 사용한 데이터	4

제2장 분석방법에 대한 개요

2.1 K-nearest Neighbor Classification (K 최근접 이웃기법)	6
2.2 Support Vector Machine (서포트 벡터 머신)	7
2.3 Decision Tree (의사결정나무)	8
2.4 Logistic Regression Model (로지스틱 회귀모형)	10
2.5 Linear Discriminant Analysis (선형판별분석)	12

제3장 각 방법론에 따른 최적의 방법 선택

3.1 방법	14
3.2 K-nearest Neighbor Classification (K 최근접 이웃기법)	14
3.3 Support Vector Machine (서포트 벡터 머신)	15
3.4 Decision Tree (의사결정나무)	16
3.5 Logistic Regression Model (로지스틱 회귀모형)	18

제4장 최적의 방법을 사용한 모의실험

4.1 개요	22
4.2 결과	22
4.2-1 K-nearest Neighbor Classification (K 최근접 이웃기법)	23
4.2-2 Support Vector Machine (서포트 벡터 머신)	23
4.2-3 Decision Tree (의사결정나무)	24
4.2-4 Logistic Regression Model (로지스틱 회귀모형)	24
4.2-5 Linear Discriminant Analysis (선형판별분석)	25

제5장 결론

5.1 결론	26
※ 참고문헌	27
※ R code	28

1. 서론

1.1 연구의 배경 및 목적

최근 들어서 인터넷과 스마트폰 등 통신매체의 발달에 의하여 이전에는 없었던 신종사기 범죄가 증가하고 있다. 이에 따라 우리나라의 기업을 포함한 세계의 여러 기업들이 이러한 범죄에 따른 손실을 막기 위해 노력하고 있으며 고객의 신용도를 범죄율을 예측하는 데 있어서 필요한 지표들 중 하나로 사용하고 있다. 신용도를 제대로 예측하게 되면 범죄를 미리 예방하는 것에 있어서 많은 도움을 얻을 수 있고 자연스럽게 범죄율은 낮아질 것이다. 이에 따라 기업은 잠재적인 미래의 고객들에게 긍정적인 평가를 받게 될 것이며 더 발전할 수 있는 발판을 마련하게 될 것이다.

기업에게 있어 신용은 일종의 자산과 마찬가지로이다. 신용이 없으면 기업 활동은 사실상 불가능해진다. 신용등급이 높은 회사에는 투자 자본이 밀려들어오게 되지만 신용등급이 하락하게 되면 투자자들로부터 철저히 외면당할 수밖에 없게 된다. 기업 입장에서는 좋은 신용등급을 보유하는 것이 자금조달을 유리하게 할 수 있는 방법이 되기 때문에 그만큼 신용도는 중요한 척도이다.

따라서 본 논문에서는 고객의 신용도와 관련된 유명한 데이터들 중 하나인 ‘German Credit Data’를 가지고 신용도를 예측하는 데 있어 바람직한 방법은 어떤 것인지 알아보고자 한다.

1.2 분석에 사용한 데이터

UCI Machine Learning Repository (<http://archive.ics.uci.edu/ml/>) 사이트를 방문하게 되면 ‘German Credit Data’를 csv 파일 형태로 손쉽게 얻을 수 있다. 이 데이터 안에는 총 1000명의 고객에 대한 정보가 들어있으며 변수는 ‘Default’라는 명칭의 신용도를 나타내는 반응변수와 나머지 20개의 설명변수들로 이루어져 있다. 각각의 변수들에 대하여 간략하게 설명해보면 다음과 같다.

Default -> 1(신용도 나쁨), 0(신용도 좋음)

1 checkingstatus -> 당좌 예금의 유무

2 duration -> 신용도의 지속기간(month)

3 history -> 거래내역

4 purpose -> 거래목적

5 amount -> 거래양

6 savings -> 저축 예금의 유무

7 employ -> 고용 여부

8 installment -> 가처분 소득의 백분율에 의한 할부율

9 status -> 개인 신상과 성별

10 others -> 채무자 또는 보증인의 유무

11 residence -> 거주기간

12 property -> 재산

13 age -> 나이

14 otherplans -> 할부계획

15 housing -> 집소유 유무

16 cards -> 기존 신용도의 수

17 job -> 일자리의 상태

18 liable -> 보수를 책임져야 할 사람의 수

19 tele -> 전화등록 유무

20 foreign -> 외국인노동자 유무

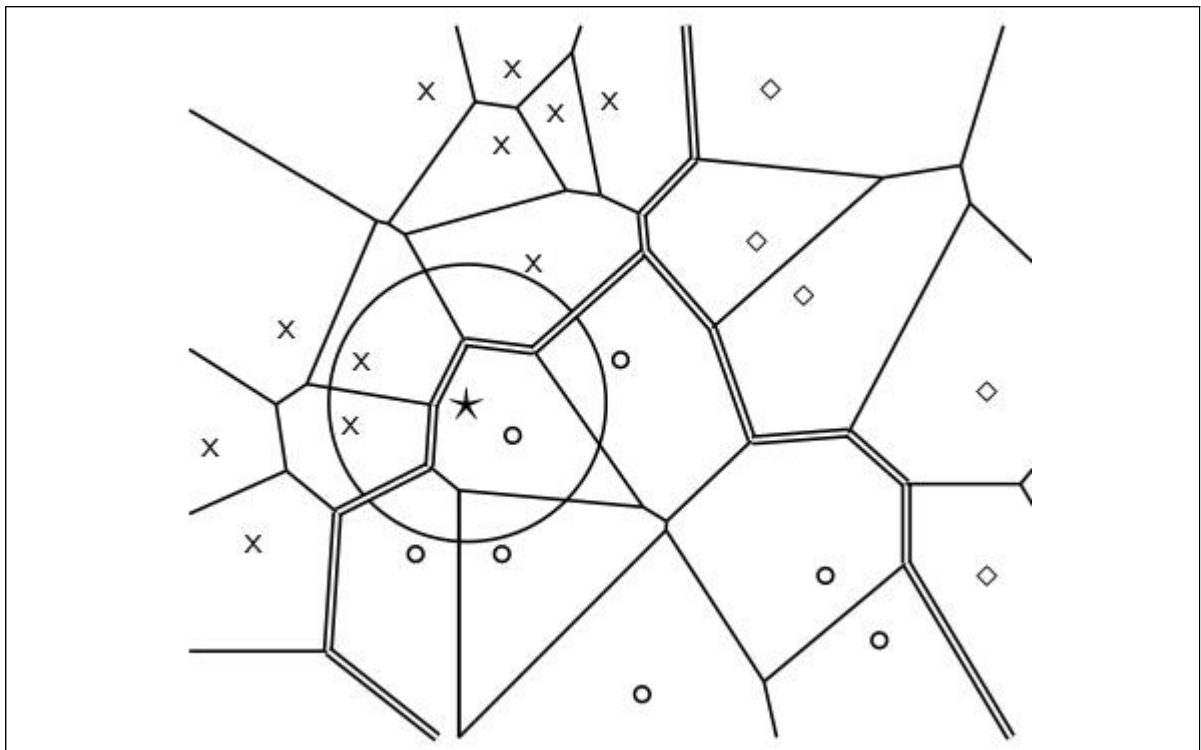
이 변수들 중 duration, amount 그리고 age 만 연속형 변수이고 나머지 변수들은 모두 이산형이다. 그렇기에 반응변수 Default 를 제외한 설명변수들 중 이산형 변수들에 한하여 dummy variable 을 생성하여 분석 시 용이하게 하였음을 밝혀둔다. 반응변수 Default 의 값이 1인 경우가 신용도가 좋지 않은 고객이다.

2. 분석방법에 대한 개요

분석을 진행하기에 앞서 사용하게 될 분석방법에 대하여 간략하게 설명을 하겠다.

2.1 K-nearest Neighbor Classification (K 최근접 이웃기법)

K-Nearest Neighbor (K 최근접 이웃기법)은 실험 문헌에 가장 근접한 K개의 문헌을 선택하여, 이들 범주 중 가장 많은 범주를 선택하는 분류법이다. 이 분류방법은 본 논문에서 소개할 5가지 분석방법들 중 가장 단순한 방법이고 직관적으로 이해가 가능한 방법이다.



(그림 2.1) $k=3$ 인 경우의 KNN의 원리

이 방법은 데이터를 분류할 때 가장 가까운 하나의 이웃에 의존 하는 것이 아닌 가장 가까운 K개의 데이터에 의존한다. 따라서 K의 선택에 민감한 영향을 받으므로 어떤 k값을 선택하느냐가 중요한 문제이다. 일반적으로 작은 값의 K를 선택하게 되면 편향이 작아지는 대신에 분산이 커지는 문제점이 있고, 반대로 큰 값의 K를 선택하게 되면 분산이 작아지는 대신 편향이 커지는 문제점이 있게 된다. 그러므로 본 연구에서는 각 K의 값에 따른

error의 값을 계산해서 가장 수치가 낮은 K값을 모의실험에 적용할 것이다. 일반적으로 K의 값은 경험적인 방법으로서 보통 3 또는 5 등의 홀수로 결정하게 된다. 또는 멀리 있는 data에는 가중치를 적게 주고 가까이 있는 data에는 가중치를 많이 주는 KK-NN (거리가중투표) 방식을 적용하기도 한다. 다음의 (식 2.1)은 KK-NN 방법의 원리를 나타낸 식이다.

KK-NN 거리가중투표 (식 2.1)

$$y' = \underset{v}{\operatorname{argmax}} \sum_{(X_i, y_i) \in D_z} w_i \times (v = y_i)$$

2.2 Support Vector Machine (서포트 벡터 머신)

SVM(Support Vector Machine)은 주어진 데이터를 두 개의 클래스로 구분 할 수 있는 기준이 되는 경계를 구하는 것이 목적이다. 이 경계선을 ‘초평면’ 이라고 하며 초평면은 무수히 많이 존재 하게 된다. 그 중 분류 오류가 가장 작은 선분을 골라 최적의 초평면으로 정하는 것이 바로 SVM 방법이다. 커널 매핑 개념과 최적화 기술을 통계적 학습의 원리에 통합한 알고리즘으로 초평면을 찾은 후에는 입력된 벡터 값의 좌표의 위치에 따라 해당 클래스를 반환하게 된다. 본 연구에서는 kernel type 중 Radial basis를 적용하여 분석할 것이며 그 수식은 (식 2.2)와 같다.

(식 2.2) *Radial basis* : $\exp(-\gamma \cdot |u - v|^2)$

그러므로 여러 가지 γ 의 값들과 범위의 한계를 결정지어주는 cost의 값들의 조합들 중 error의 수치가 가장 낮은 조합의 결과를 이용하여 모의실험을 할 것이다.

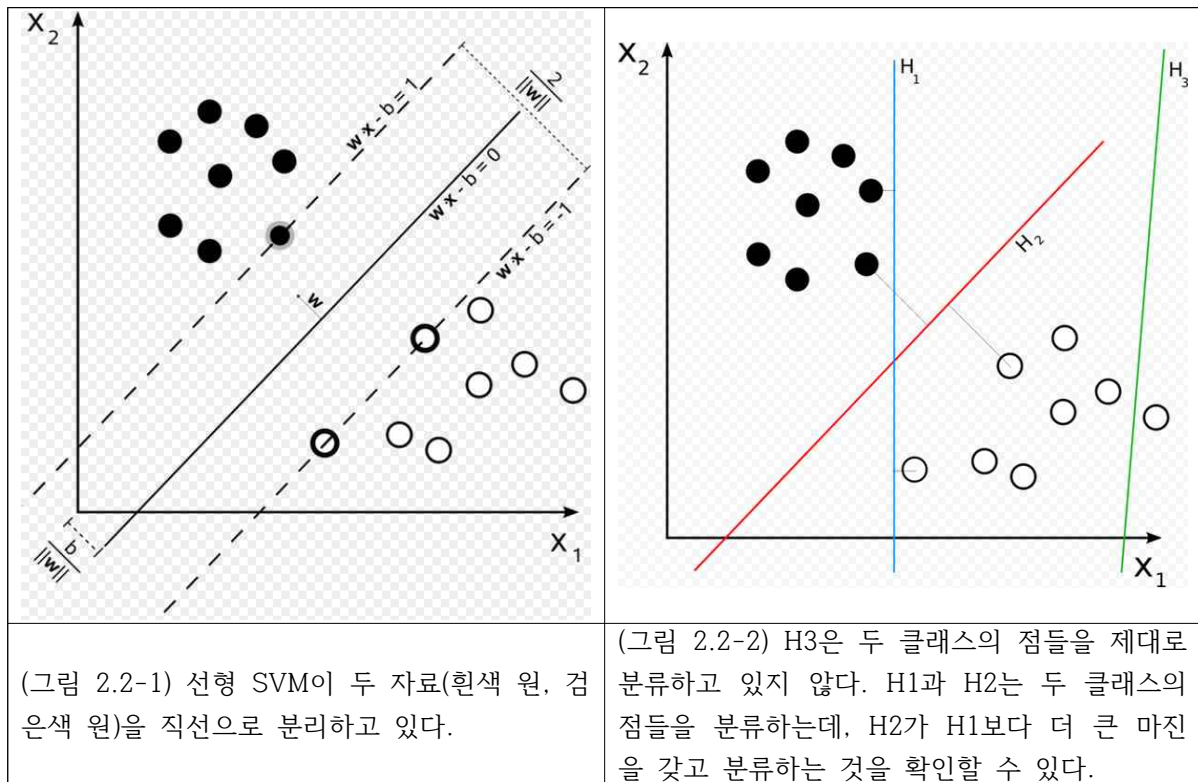
예를 들어 다음과 같은 학습데이터집합 D가 주어졌다고 생각해보자.

$$D = \{ (x_i, c_i) | x_i \in R^p, c_i \in \{-1, 1\} \}, (i = 1, 2, \dots, n)$$

c_i 는 1이나 -1의 값을 갖는 변수로 x_i 가 속한 class이며, x_i 는 p차원 실수벡터이다.

신경망을 포함하여 많은 학습 알고리즘들은, 이러한 학습데이터가 주어졌을 때 $c_i = 1$ 인 점들과 $c_i = -1$ 인 점들을 분리하는 hyperplane 을 찾아내는 것이 공통의 목표인데,

SVM이 다른 알고리즘과 차별화되는 특징은 단지 점들을 분리하는 hyperplane을 찾는 것으로 끝나는 것이 아니라, 점들을 분리할 수 있는 수많은 후보평면들 가운데 마진이 최대가 되는(maximum-margin) hyperplane을 찾는다는 것이다. 여기서 margin이란 hyperplane으로부터 각 점들에 이르는 거리의 최소값을 말하는데, 이 margin을 최대로 하면서 점들을 두 클래스로 분류하려면, 결국 class 1에 속하는 점들과의 거리 중 최소값과 class -1에 속하는 점들과의 거리 중 최소값이 같도록 hyperplane이 위치해야 하며, 이러한 hyperplane을 maximum-margin hyperplane이라고 한다. 결론적으로 SVM은 두 클래스에 속해있는 점들을 분류하는 수많은 hyperplane들 중, 최대한 두 클래스의 점들과 거리를 유지하는 것을 찾아내는 알고리즘이라 할 수 있다.

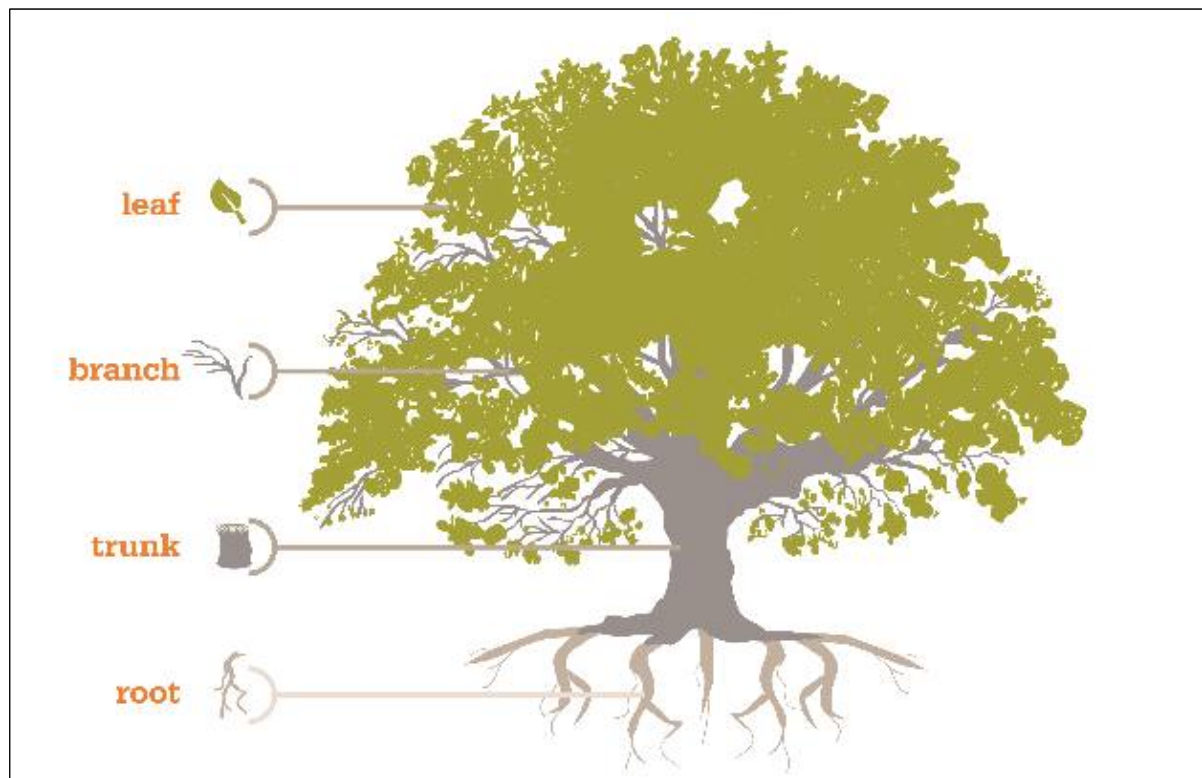


2.3 Decision Tree (의사결정나무)

의사결정나무(Decision Tree)는 의사결정규칙(Decision Rule)을 나무구조로 도표화하여 분류와 예측을 수행하는 분석 방법이다. 이 방법은 분류 또는 예측의 과정이 나무구조에

의한 추론규칙에 의해서 표현되기 때문에 다른 방법들(신경망, 판별분석, 회귀분석 등)에 비하여 연구자가 그 과정을 쉽게 이해하고 설명할 수 있다는 장점을 가지고 있다. 데이터마ining에서의 의사결정나무는 탐색과 모형화라는 두 가지 특성을 모두 가지고 있다고 할 수 있다. 즉, 의사결정나무는 판별분석 또는 회귀분석 등과 같은 모수적 모형을 분석하기 위하여 사전에 이상치들을 검색하고나 분석에 필요한 변수 또는 모형에 포함되어야 할 교호효과를 찾아내기 위하여 사용될 수도 있고, 그 자체가 분류 또는 예측 모형으로 사용될 수도 있다.

의사결정나무는 하나의 나무구조를 이루고 있으며, 마디(node)라고 불리는 구성요소들로 이루어져 있다. 마디는 그 기능에 따라서 다음과 같이 여러가지로 분류할 수 있다.



(그림 2.3-1) Node of tree

뿌리마디(Root Node) : 나무구조가 시작되는 마디로써 전체 자료로 이루어져 있다.

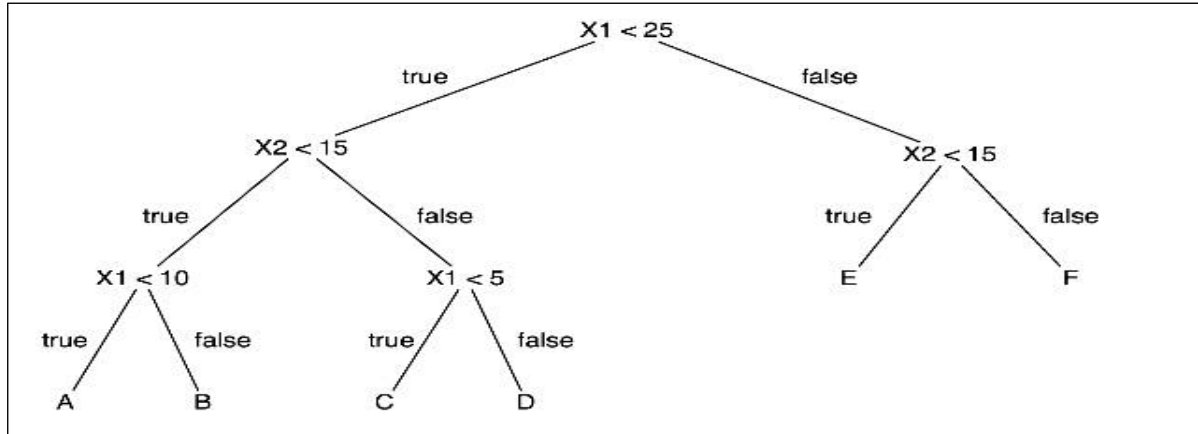
자식마디(Child Node) : 하나의 마디로부터 분리되어 나간 2개 이상의 마디들을 의미한다.

부모마디(Parent Node) : 자식마디의 상위마디를 의미한다.

끝마디(Terminal Node) 또는 잎(Leaf) : 각 나무줄기의 끝에 위치하고 있는 마디를 의미한다.

중간마디(Internal Node) : 나무구조의 중간에 있는 끝마디가 아닌 마디들을 의미한다.

가지(branch) : 하나의 마디로부터 끝마디까지 연결된 일련의 마디들을 의미하며, 이 때 가지를 이루고 있는 마디의 개수를 깊이(Depth)라고 한다.



(그림 2.3-2) Example of Decision tree

의사결정나무분석은 일반적으로 다음과 같은 단계를 거치게 된다.

의사결정나무의 형성 : 분석의 목적과 자료구조에 따라서 적절한 분리기준(Split Criterion)과 정지규칙(Stopping Rule)을 지정하여 의사결정나무를 얻는다.

가지치기 : 분류오류(Classification Error)를 크게 할 위험(Risk)이 높거나 부적절한 추론규칙(Induction Rule)을 가지고 있는 가지(Branch)를 제거한다.

타당성 평가 : 이익도표(Gains Chart)나 위험도표(Risk Chart) 또는 검증용 자료(Test Data)에 의한 교차타당성(Cross Validation) 등을 이용하여 의사결정나무를 평가한다.

해석 및 예측 : 의사결정나무를 해석하고 예측모형을 설정한다.

위와 같은 과정에서 분리기준, 정지규칙, 평가기준 등을 어떻게 지정하느냐에 따라서 서로 다른 의사결정나무가 형성된다.

본 연구에서는 나무모형을 가지치기할 때 Relation error 의 값이 가장 작을 경우의 cp 통계량을 찾아서 이를 적용하여 모의실험을 진행할 것이다.

2.4 Logistic Regression Model (로지스틱 회귀모형)

로지스틱 회귀모형은 종속변수가 범주형인 경우 사용되는데 통상 2개 범주를 다루는 모형을 지칭하나 3개 이상의 범주를 다루는 경우에도 사용된다. 다시 말하면 어느 자료가 특정한 집단에 속할 확률을 예측하는 모형을 세우고 어느 설명변수가 유의한지를 찾아내는 방법이다. 본 연구에서 다루는 데이터는 2개의 클래스로 나누어진 경우이므로 이에 대해서만 생각해 보겠다.

일반적인 회귀모형은 (식 2.4-1)와 같다.

$$E(y|X) = f(x_1, \dots, x_p) = \beta_0 + \sum_{j=1}^p \beta_j x_j \quad (\text{식 2.4-1})$$

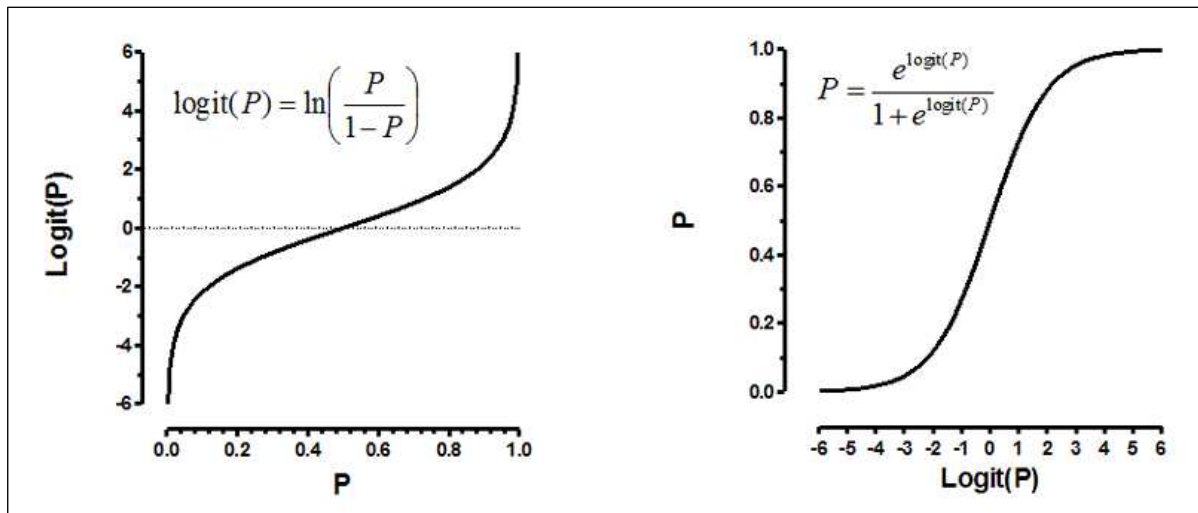
(식 2.4-1)에서 반응변수의 값이 0과 1을 갖는다면 (식 2.4-2)의 식을 얻을 수 있다.

$$p(y=1|X) = \frac{\exp(\beta_0 + \sum_{j=1}^p \beta_j x_j)}{\exp(\beta_0 + \sum_{j=1}^p \beta_j x_j) + 1} \quad (\text{식 2.4-2})$$

따라서 최종적으로 (식 2.4-3)의 로지스틱 회귀모형을 얻을 수 있으며,

$$\log(P(y=1|X)) = \log\left(\frac{P(y=1|X)}{1-P(y=1|X)}\right) = f(x_1, \dots, x_p) = \beta_0 + \sum_{j=1}^p \beta_j x_j \quad (\text{식 2.4-3})$$

이에 따라 (그림 2.4-1)과 같이 확률 p와 logit(odds의 값에 log를 취한 값)과의 관계를 나타내는 S자 형태의 곡선이 일반적으로 나오게 된다.



(그림 2.4-1)

모형 전체의 유의성은 $-2 \cdot \log(\text{Likelihood})$, AIC 또는 SBC 통계량을 이용하고 회귀계수의 유의성 검정은 최대가능도추정량(Maximum Likelihood Estimator(MLE))의 정규근사성질을 적용한 Wald's Chi-square test를 이용하게 된다. 본 연구에서는 Wald's Chi-square test, 또는 Z-test를 통하여 신용도에 영향을 미치는 설명변수에는 어떠한 것들이 있는지 살펴보고 이에 따른 최종모형을 적용하여 모의실험을 진행할 것이다. 참고적으로 Wald's test의 이론적인 배경이 되는 MLE의 정규근사성에 대해 간략하게 설명해보겠다.

Asymptotic normality of MLE

Suppose that $f(x|\theta)$ is the probability density(or mass) function of random variable X, then Fisher's information number $I(\theta)$ is defined by,

$$I(\theta) = E\left[\left(\frac{d}{d\theta} \log(f(x|\theta))\right)^2\right] = -E\left[\frac{d^2}{d\theta^2} \log(f(x|\theta))\right] .$$

And if $\hat{\theta}$ is the MLE of parameter θ , then,

$$\sqrt{n} \cdot (\hat{\theta} - \theta_0) \xrightarrow{d} N\left(0, \frac{1}{I(\theta_0)}\right) \text{ as } n \rightarrow \infty$$

2.5 Linear Discriminant Analysis (선형판별분석)

판별분석은 어떠한 객체가 가지는 특성을 바탕으로 미리 정의된 그룹중의 하나에 그 객체를 분류하는 것이다. 어떠한 관심있는 모집단이 K개의 서로 다른 그룹 G_1, G_2, \dots, G_k 로 나누어지고, 이 모집단에서 각 그룹 G_i 의 구성비율은 π_i 로 가정한다. 여기서

$\pi_i \geq 0$ ($i = 1, 2, \dots, k$) 이고 $\sum_{i=1}^k \pi_i = 1$ 이다. π_i 는 객체가 그룹 G_i 에서 나올 확률을 나타낸

다. 이 모집단에서 추출된 객체는 단지 하나의 그룹에 속하는 것으로 가정한다. 객체가 가지는 p개의 특성은 p차원 확률벡터 x 로 나타낸다. 여기에서는 그룹의 수가 2개인 경우에 한하여 설명하겠다. 분포는 보통 다변량 정규분포를 가정한다.

한 객체를 두 개의 그룹 중의 하나에 분류하는 문제에서 첫 번째 그룹 G_1 의 관측치 x_{G1} 과 그룹 G_2 의 관측치 x_{G2} 에 대하여 선형판별분석에서는 다음을 가정한다.

$$x_{G1} \sim N_p(\mu_1, \Sigma_1) , x_{G2} \sim N_p(\mu_2, \Sigma_2) \quad (\Sigma_1 = \Sigma_2 = \Sigma)$$

$$\hat{\Sigma} = \frac{(n_1 - 1) \cdot S_1 + (n_2 - 1) \cdot S_2}{(n_1 + n_2 - 2)} ,$$

$$S_1 = \frac{1}{n_1 - 1} \sum_{i=1}^{n_1} (x_{G1i} - \overline{x_{G1}})(x_{G1i} - \overline{x_{G1}})^T , S_2 = \frac{1}{n_2 - 1} \sum_{i=1}^{n_2} (x_{G2i} - \overline{x_{G2}})(x_{G2i} - \overline{x_{G2}})^T$$

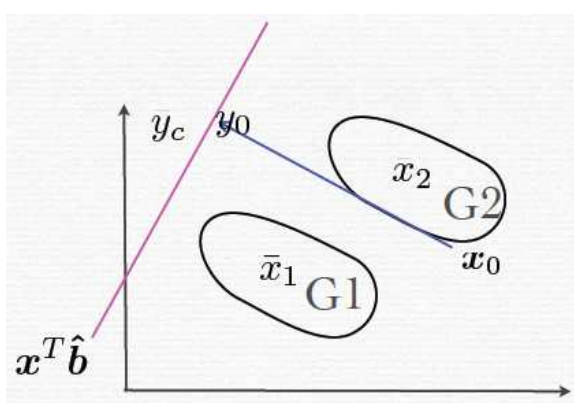
(식 2.5-1)

R.A.Fisher 는 다음과 같은 방법으로 선형판별함수를 제안하였다. 즉, 판별함수를 설명변수의 선형결합으로써 $y = x^T \cdot b$ 로 표현하였다. 여기에서 벡터 b 는 다음과 같은 방법을 통해서 추정하게 된다.

$E(x_{G1}^T \cdot b) = \mu_1^T \cdot b, E(x_{G2}^T \cdot b) = \mu_2^T \cdot b$ $Var(y) = Var(x^T \cdot b) = Var(b^T \cdot x) = b^T \cdot \Sigma \cdot b$ <p>이므로 다음의 식을 만족하는 벡터 b를 찾게 된다.</p> $\max_b \frac{(\mu_1^T \cdot b - \mu_2^T \cdot b)^2}{b^T \cdot \Sigma \cdot b}$ <p>이에 대한 결과는 코쉬-슈바르츠 부등식을 이용하게 되면 $b = \Sigma^{-1} \cdot (\mu_1 - \mu_2)$ 로써 얻어지게 되며 주어진 자료를 통하여 벡터 b에 대한 추정치를 다음 식을 통하여 얻게 된다.</p> $\hat{b} = \widehat{\Sigma}^{-1} \cdot (\overline{x_{G1}} - \overline{x_{G2}})$

(식 2.5-2)

판별의 기준(Cut-Off Value)과 임의의 자료에 대한 그룹판별방법은 (식 2.5-3)와 같다.

	$\overline{y_c} = \frac{n_1 y_1 + n_2 y_2}{n_1 + n_2}$ <p>(y_1과 y_2의 가중평균(Weighted average)) ($y_1 = (\overline{x_{G1}})^T \cdot \hat{b}$, $y_2 = (\overline{x_{G2}})^T \cdot \hat{b}$)</p> <p>여기에서 임의의 자료 x_0의 그룹을 판별할 때 만약 $y_1 < y_2$ 이면 다음과 같이 판별한다.</p> $y_0 = x_0^T \cdot \hat{b} > \overline{y_c} \Rightarrow$ $x_0 \in G_2,$ $y_0 = x_0^T \cdot \hat{b} < \overline{y_c} \Rightarrow$ $x_0 \in G_1$
(그림 2.5) LDA의 원리	(식 2.5-3)

3. 각 방법론에 따른 최적의 방법 선택

3.1 방법

이제부터는 모의실험을 진행하기에 앞서 각 방법론들을 적용할 경우 해당하는 모수의 값들을 어떻게 설정해야 최적의 방법을 얻을 수 있는지 살펴보려고 한다. KNN 분류의 경우는 인접한 자료의 개수 K 의 값, SVM은 radial basis에서의 γ 값과 범위를 결정해주는 cost 의 값, Decision Tree에서는 cp 통계량의 값, 그리고 Logistic Regression Model에서는 신용도에 유의하게 영향을 미치는 변수들을 Wald's Chi-square test를 통하여 선정할 것이다. KNN, SVM, 그리고 Decision Tree의 경우에는 전체 데이터의 절반 크기인 Train data를 생성해서 진행할 것이며 sampling하는 방법은 'bootstrapping'으로 통일시켰다. 그리고 Logistic Regression Model의 경우는 전체 데이터를 이용하여 최적의 모형을 찾을 것이다. 마지막으로 Linear Discriminant Analysis의 경우는 별도의 선택 과정 없이 모든 설명변수들을 이용하여 모의실험을 진행할 것이다. 최적의 방법 선택 및 모의실험 모두 패키지 'R 3.1.0'을 사용한다.

3.2 K-nearest Neighbor Classification (K 최근접 이웃기법)

KNN 분류에서는 인접한 자료의 개수 k 가 분석결과에 결정적인 역할을 하게 된다. 따라서 적절한 k 의 값을 선정해야 하는데 k 의 값이 커지면 분산이 작아지는 대신 편향은 커지고 반대로 k 의 값이 작아지면 편향이 작아지는 대신 분산이 커지게 된다. 그러므로 각각의 k 값에 대하여 error 값을 계산하고 그 수치가 가장 작은 k 값을 모의실험에 적용시키게 된다. 데이터의 절반크기인 train data를 얻은 다음 1부터 120까지의 k 값에 대하여 시행한 결과 다음과 같은 결과가 나왔다.

Parameter tuning of 'knn.wrapper':

- sampling method: bootstrapping

- best parameters:

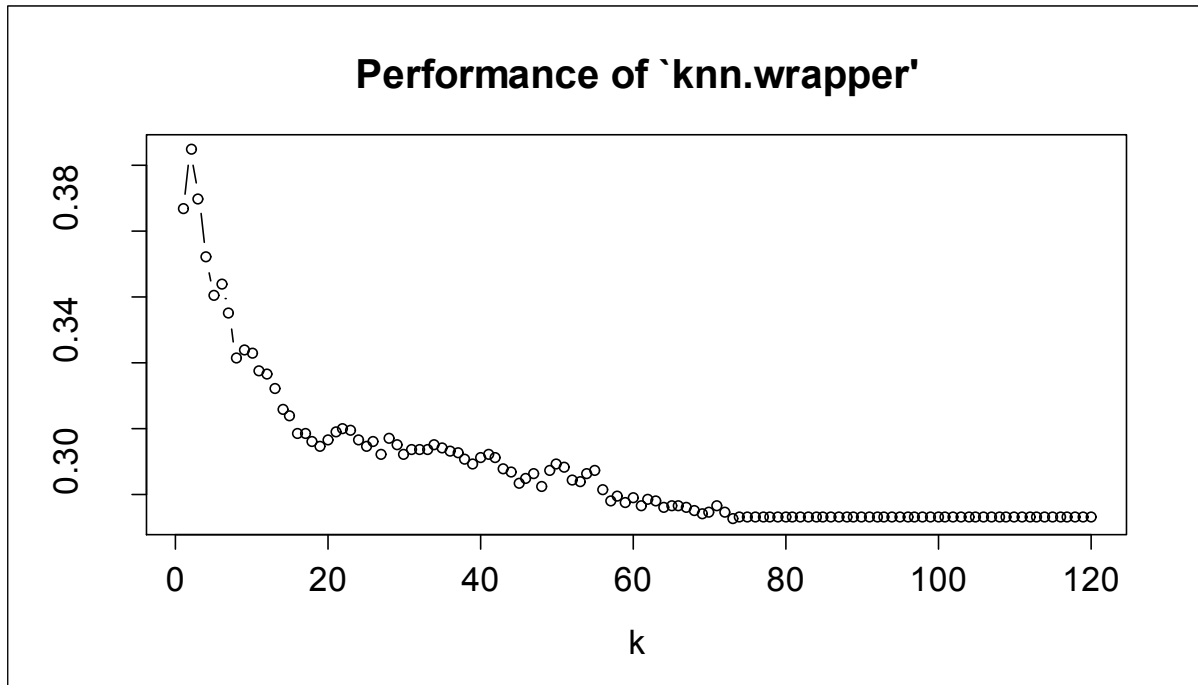
 - k

 - 73

- best performance: 0.2924368

$k=73$ 인 경우에 error의 값이 0.2924368 로서 가장 작은 수치를 나타냈다. 이 결과를 그

림으로 나타내보면 (그림 3.2-1)과 같다.



(그림 3.2-1)

가로축은 k의 값이고 세로축은 error의 값이다. k=73인 경우에 error 값이 가장 작으므로 이 값을 모의실험에 적용하도록 한다.

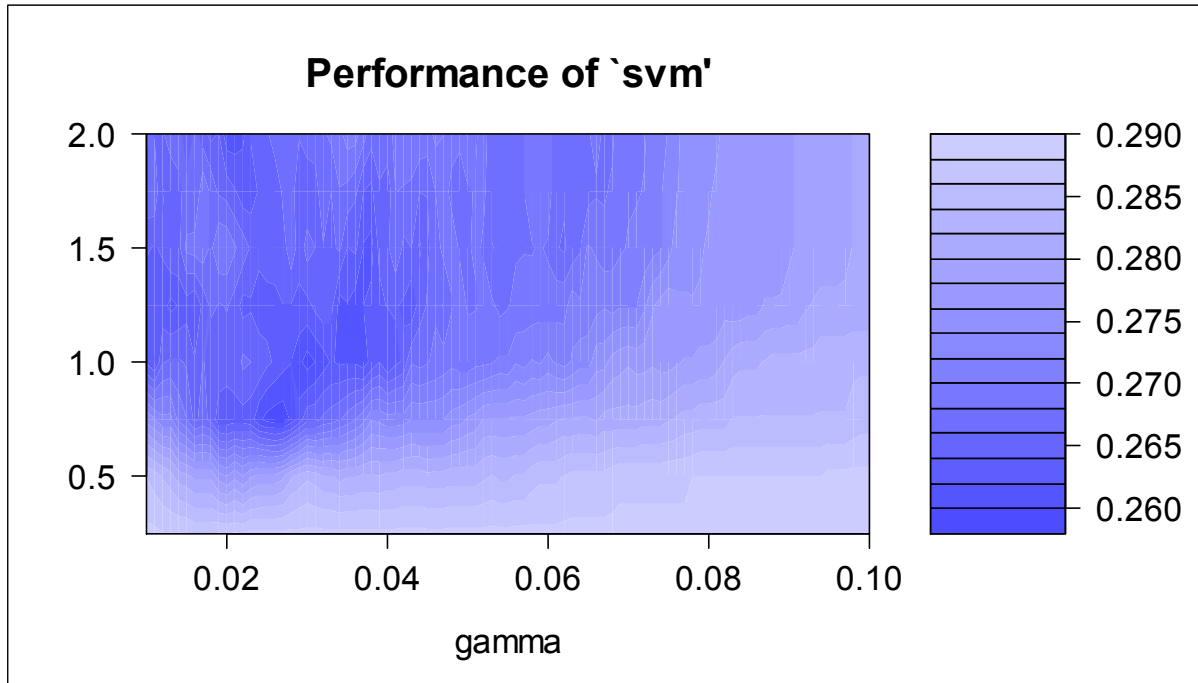
3.3 Support Vector Machine (서포트 벡터 머신)

SVM 분류에서는 radial basis에서의 gamma 값과 범위를 결정해주는 cost의 값, 이 두 가지 수치들의 조합에 의해 분석의 민감도가 결정된다. KNN 분류에서와 마찬가지로 이 방법론에서도 해당 모수의 값에 따른 error 값들을 산출해보고 가장 작은 수치를 나타낼 때의 모수의 값을 모의실험에 적용할 수 있다. 데이터의 절반크기인 train data를 얻은 다음 0.01부터 0.1까지의 0.001 간격의 gamma 값들과 0.25부터 2까지의 0.25 간격의 cost 값들의 조합에 따른 error 값들을 계산한 결과 다음과 같은 결과를 얻을 수 있었다.

```
Parameter tuning of 'svm':
- sampling method: bootstrapping
- best parameters:
  gamma cost
  0.026 0.75
```

- best performance: 0.2587884

결과적으로 gamma의 값이 0.026이고 cost의 값이 0.75일 때 error의 값이 0.2587884로서 최소의 수치를 보였음을 알 수 있고 이 결과를 보기 쉽게 그림으로 나타내면 (그림 3.3-1)과 같다.



(그림 3.3-1)

그림에서 색이 진할수록 error 값이 작은 것이다. gamma=0.026, cost=0.75 일 때 최소의 error 값이 나왔으므로 이 결과를 모의실험에 적용하도록 한다.

3.4 Decision Tree (의사결정나무)

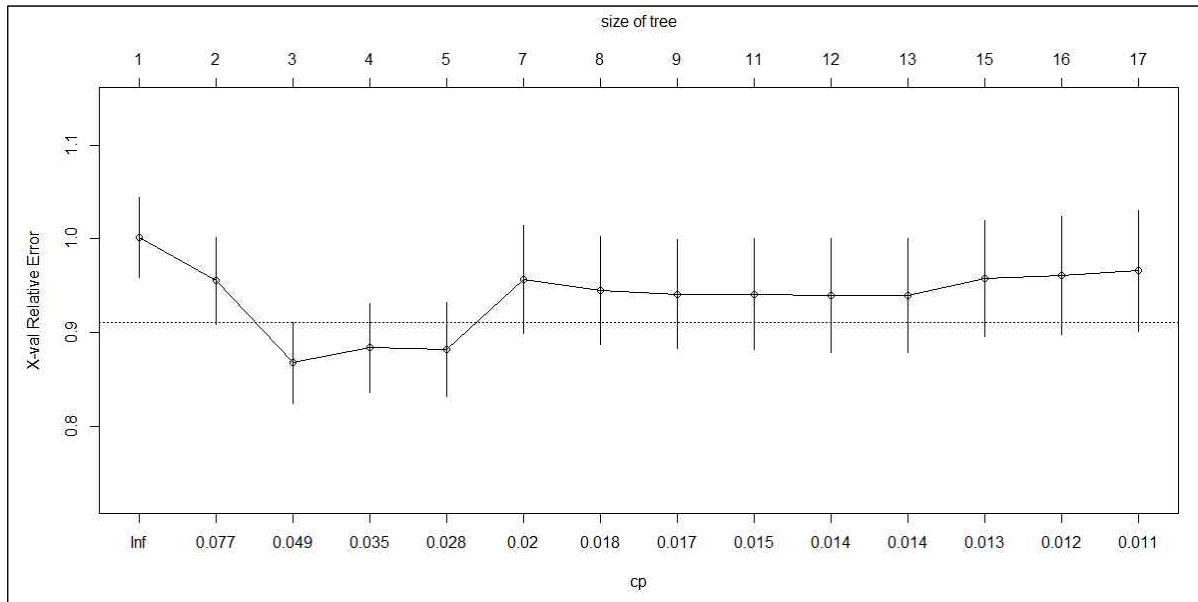
의사결정나무에 의한 분류에서는 가지치기(pruning)을 실시할 때 Relation error의 값이 가장 작은 cp 통계량의 값을 적용하는 것이 바람직하다. 이를 위하여 먼저 데이터의 절반 크기인 train data를 얻고 이에 대하여 cp 통계량의 값에 따른 Relation error의 값을 산출한 결과 다음과 같았다.

Variables actually used in tree construction:

[1] amount	checkingstatus1	checkingstatus2	duration
[5] employ3	housing1	housing2	others1
[9] property1	purpose4	purpose9	tele

Root node error: 101.67/500 = 0.20334

	CP	nsplit	rel error	xerror	xstd
1	0.086069	0	1.00000	1.00164	0.042917
2	0.068477	1	0.91393	0.95534	0.046167
3	0.035648	2	0.84545	0.86761	0.043361
4	0.035318	3	0.80981	0.88379	0.047407
5	0.022778	4	0.77449	0.88226	0.050270
6	0.017704	6	0.72893	0.95665	0.057180
7	0.017556	7	0.71123	0.94470	0.057863
8	0.016400	8	0.69367	0.94085	0.058122
9	0.014531	10	0.66087	0.94118	0.059678
10	0.014186	11	0.64634	0.93958	0.060540
11	0.013934	12	0.63215	0.93946	0.060660
12	0.012969	14	0.60429	0.95771	0.061872
13	0.011334	15	0.59132	0.96140	0.062947
14	0.010000	16	0.57998	0.96609	0.064725



(그림 3.4-1)

위 그림 중 첫 번째 그림은 가지치기를 하기 전의 결과이다. 결과적으로 $cp=0.049$ 일 때 Relative error의 값이 가장 작음을 확인할 수 있고 이 결과를 모의실험에 적용하도록 한다.

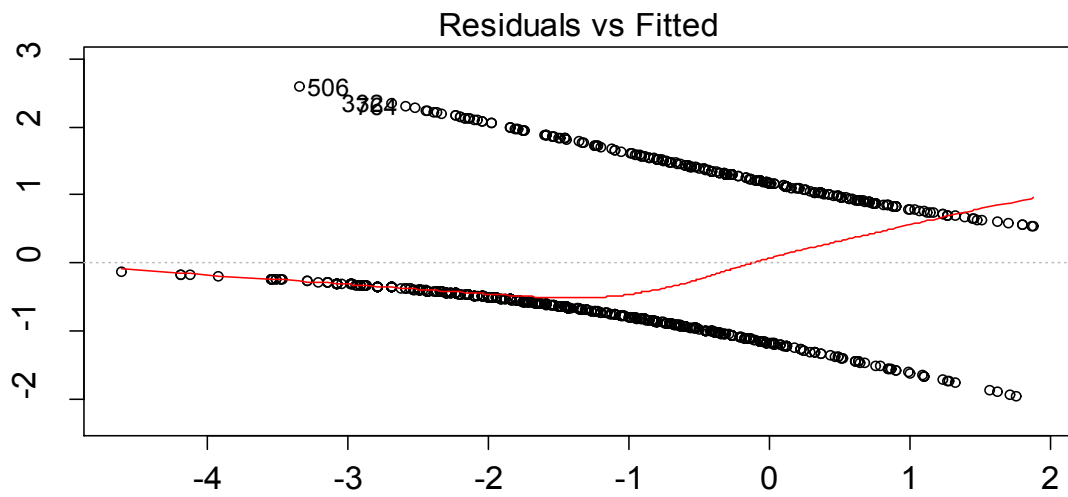
3.5 Logistic Regression Model (로지스틱 회귀모형)

일반적인 회귀분석의 경우와 마찬가지로 로지스틱 회귀모형을 이용한 분석에서는 반응변수에 유의한 영향을 미치는 설명변수들을 적절하게 선정하는 것이 중요하다고 할 수 있겠다. 일반적인 회귀분석의 경우에는 잔차가 정규성, 등분산성, 독립성, 선형성의 가정을 만족한다는 결과를 얻을 경우 t-test를 통하여 회귀계수의 유의성을 검정하여 설명변수를 선택한다. 로지스틱 회귀모형의 경우는 최대가능도추정량(Maximum Likelihood Estimator)의 정규근사 성질을 이용한 Wald's Chi-square test, 또는 Z-test를 사용하여 회귀계수의 유의성을 검정하여 적절한 설명변수들을 찾게 된다. 데이터에 총 4번에 걸친 모형적합을 통해서 적절한 설명변수는 checkingstatus, duration, history, others, otherplans, 그리고 foreign 임을 확인하였으며 그 결과는 다음과 같았다. 이 결과를 토대로 모의실험을 진행해보도록 하겠다.

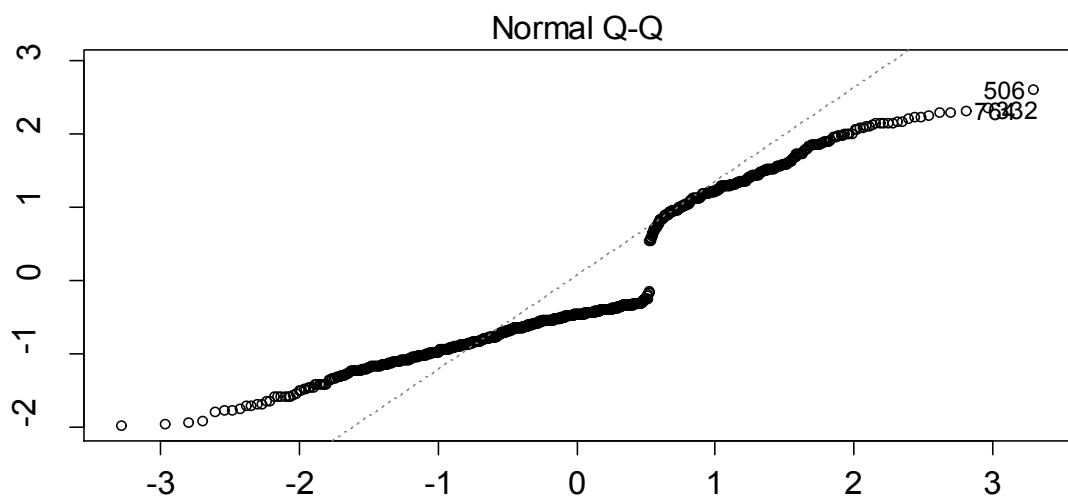
```

Call:
glm(formula = Default ~ checkingstatus1 + checkingstatus2 + checkingstatus3 +
     duration + history1 + history2 + history3 + history4 + others1 +
     others2 + otherplans1 + otherplans2 + foreign, family = "binomial",
     data = german)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9586 -0.7951 -0.4507  0.9188  2.5972
Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)   -5.552233   0.728531  -7.621 2.51e-14 ***
checkingstatus1  1.976201   0.209338   9.440 < 2e-16 ***
checkingstatus2  1.440962   0.211121   6.825 8.78e-12 ***
checkingstatus3  0.806632   0.353989   2.279 0.022685 *
duration        0.032414   0.006404   5.061 4.16e-07 ***
history1        1.542602   0.398488   3.871 0.000108 ***
history2        1.259669   0.371248   3.393 0.000691 ***
history3        0.645282   0.196067   3.291 0.000998 ***
history4        0.516589   0.307164   1.682 0.092608 .
others1         1.037122   0.391970   2.646 0.008147 **
others2         1.619915   0.527498   3.071 0.002134 **
otherplans1     0.489889   0.221536   2.211 0.027013 *
otherplans2     0.358511   0.351065   1.021 0.307155
foreign         1.244905   0.591975   2.103 0.035469 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 1221.7  on 999  degrees of freedom
Residual deviance: 1001.1  on 986  degrees of freedom
AIC: 1029.1
Number of Fisher Scoring iterations: 5

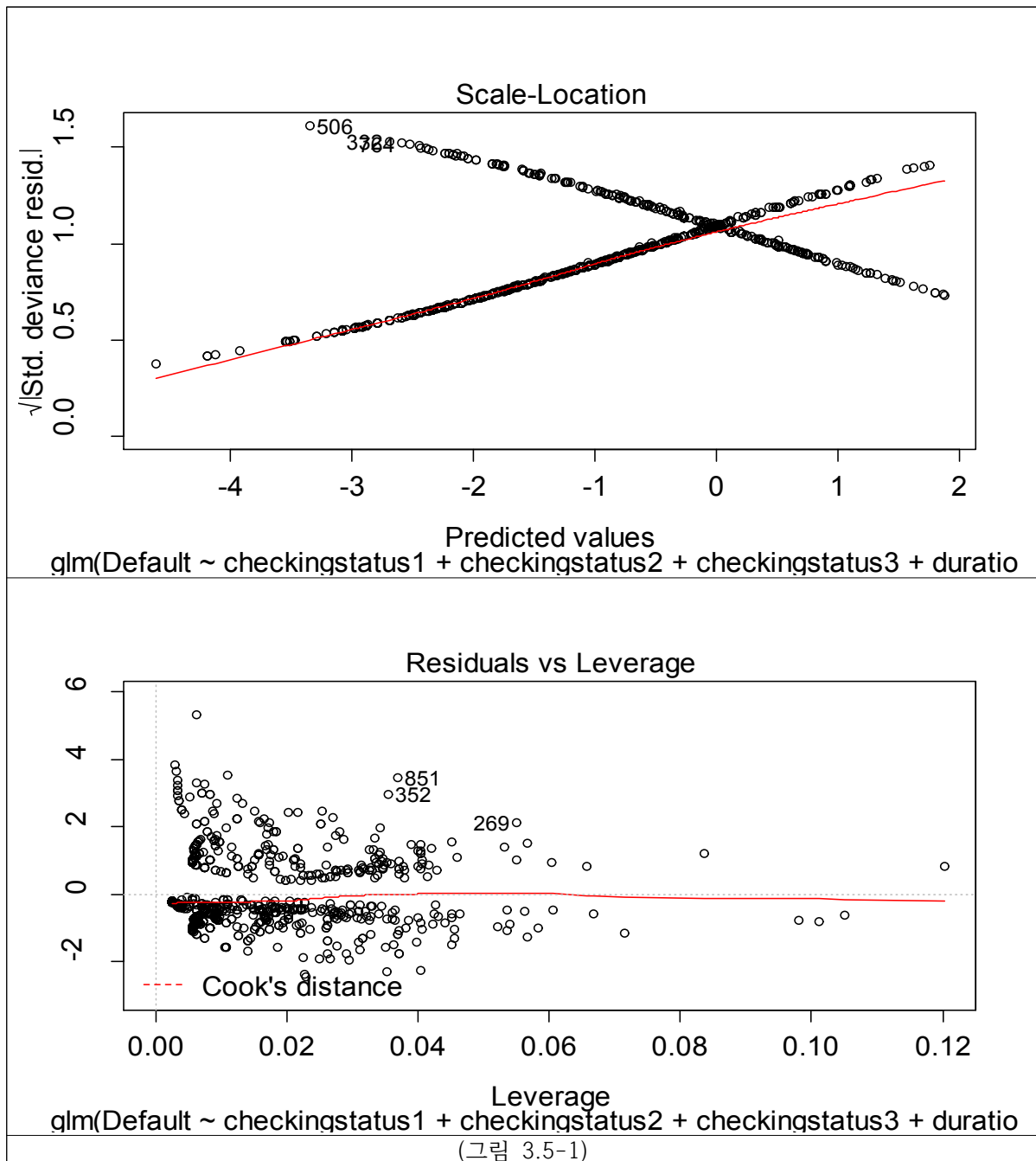
```



Predicted values
`glm(Default ~ checkingstatus1 + checkingstatus2 + checkingstatus3 + duratio`



Theoretical Quantiles
`glm(Default ~ checkingstatus1 + checkingstatus2 + checkingstatus3 + duratio`



(그림 3.5-1)에 있는 4가지 그림은 최종적인 모형에 대한 검진과 관련된 그래프이다. 특히 2번째 그림인 잔차에 대한 Normal Q-Q Plot을 보면 점들이 두 그룹으로 나누어져 있고 대부분 직선과 일치하지 않는 것으로 보아 정규성과는 거리가 있다는 것을 알 수 있고 ‘German credit data’를 분석할 때 일반적인 회귀분석을 적용하는 것은 잘못된 방법이라는 것을 확인할 수 있겠다. 반응변수인 Default가 0 또는 1의 값을 취하는 이분법적인 변수이다 보니 잔차와 관련된 그래프가 크게 두 그룹으로 나누어져 있는 것도 확인할 수 있겠다.

4. 최적의 방법을 이용한 모의실험 (Calculate Misclassification Rate)

4.1 개요

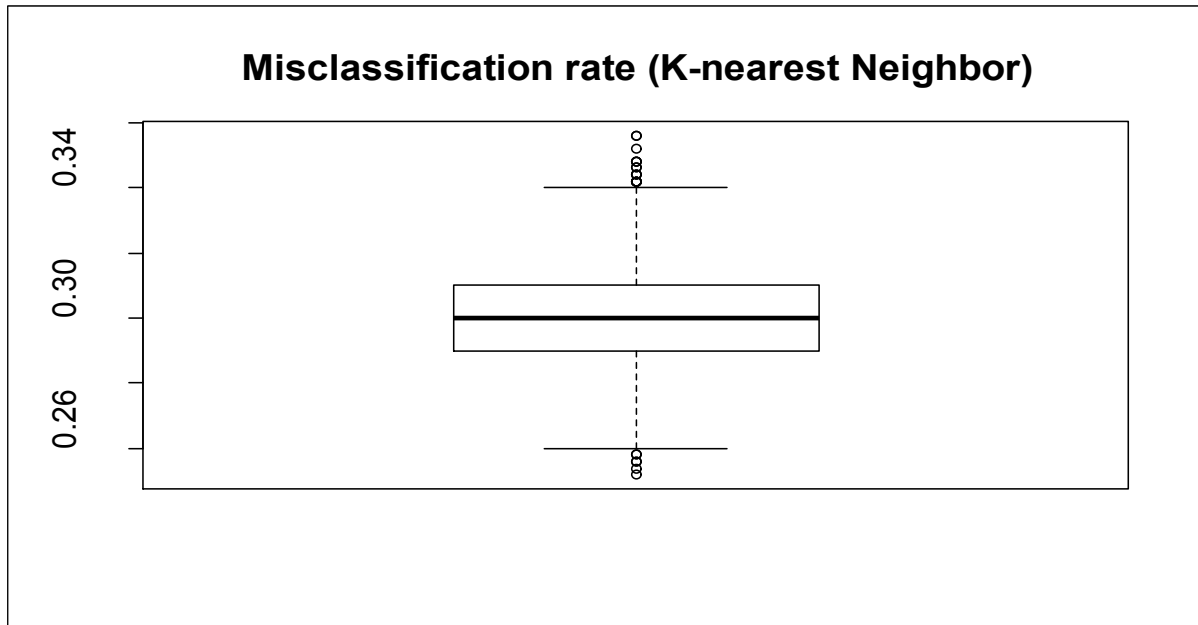
앞에서 찾은 각 방법론에 따른 최적의 방법을 이용한 모의실험을 진행하여 오분류율이 가장 작게 나오는 이상적인 방법론이 어떤 것인지 살펴보도록 한다. 데이터의 크기는 Train : Test = 3:2 로 하였으며 (즉, Train data의 크기는 600, Test data의 크기는 400으로 함.), 각 방법론마다 'German credit data'를 Train data와 Test data로 나누는 작업을 5000번 반복하여 시행에 따른 오분류율을 계산하고 최종적으로 평균을 내서 어느 방법론이 최소의 오분류율을 산출해내는지 알아봤다. 단, KNN 분류의 경우는 Train data와 Test data의 size가 반드시 같아야 함수가 작동하므로 예외적으로 두 데이터의 크기를 같게 설정했다. 그리고 LDA(선형판별분석)의 경우에는 따로 최적화된 모형을 선정하는 기준이 없으므로 모든 설명변수들을 적용할 것이며 Decision Tree와 Logistic Regression Model의 경우는 $\text{Pr}(\text{Default}=1) \geq 0.5$ 인 상황을 신용도가 좋지 않은 사람이라고 간주하고 모의실험을 진행하였다. 그 결과는 다음과 같이 주어졌다.

4.2 결과

결과에 따른 각각의 boxplot은 시행을 5000번 반복해서 얻어진 오분류율에 대하여 그린 것이고, Final answer에 나와있는 수치는 각 방법론에 따라 5000번 반복했을 때 나온 오분류율의 평균값이라는 것을 미리 밝혀둔다.

4.2-1 K-nearest Neighbor Classification (K 최근접 이웃기법)

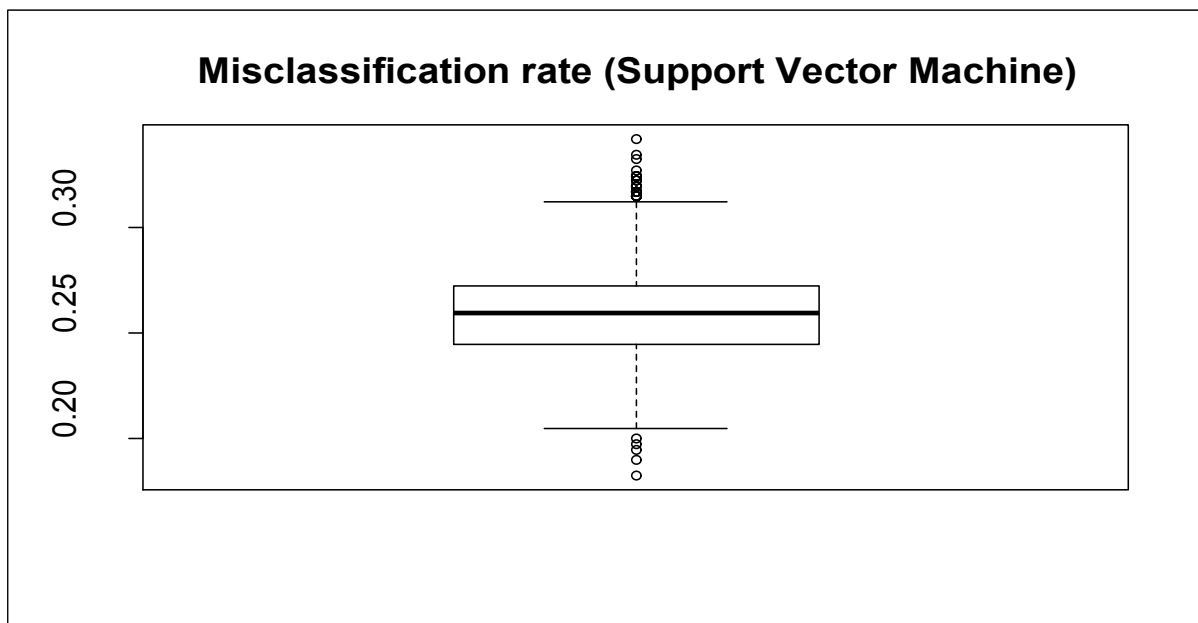
Final answer : 0.2999408



(그림 4.2-1)

4.2-2 Support Vector Machine (서포트 벡터 머신)

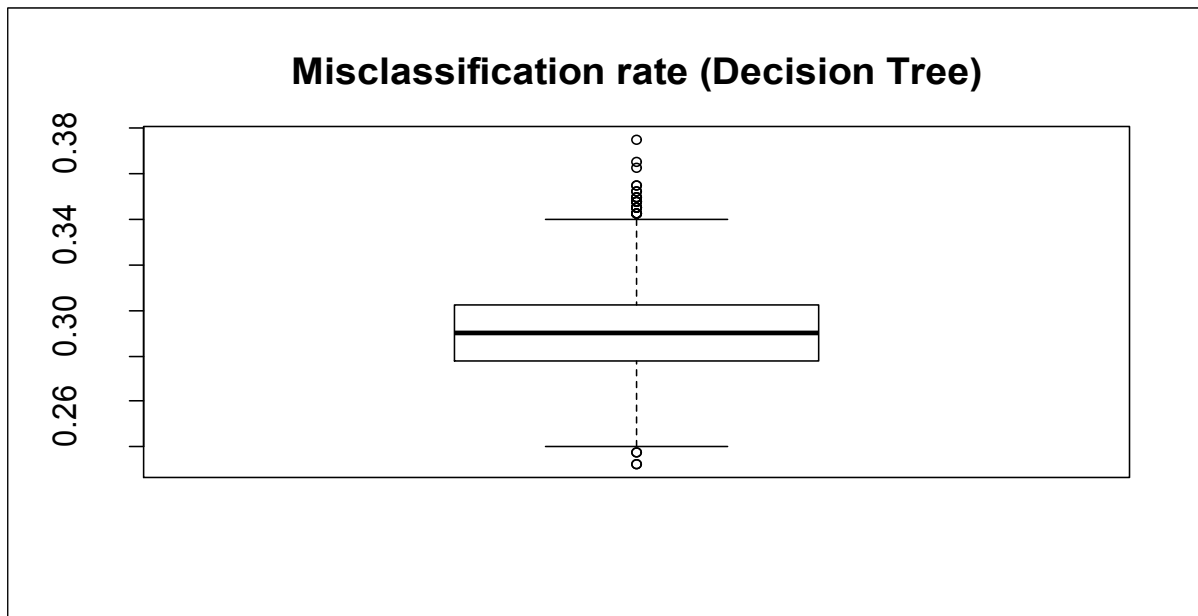
Final answer : 0.259802



(그림 4.2-2)

4.2-3 결과 : Decision Tree (의사결정나무)

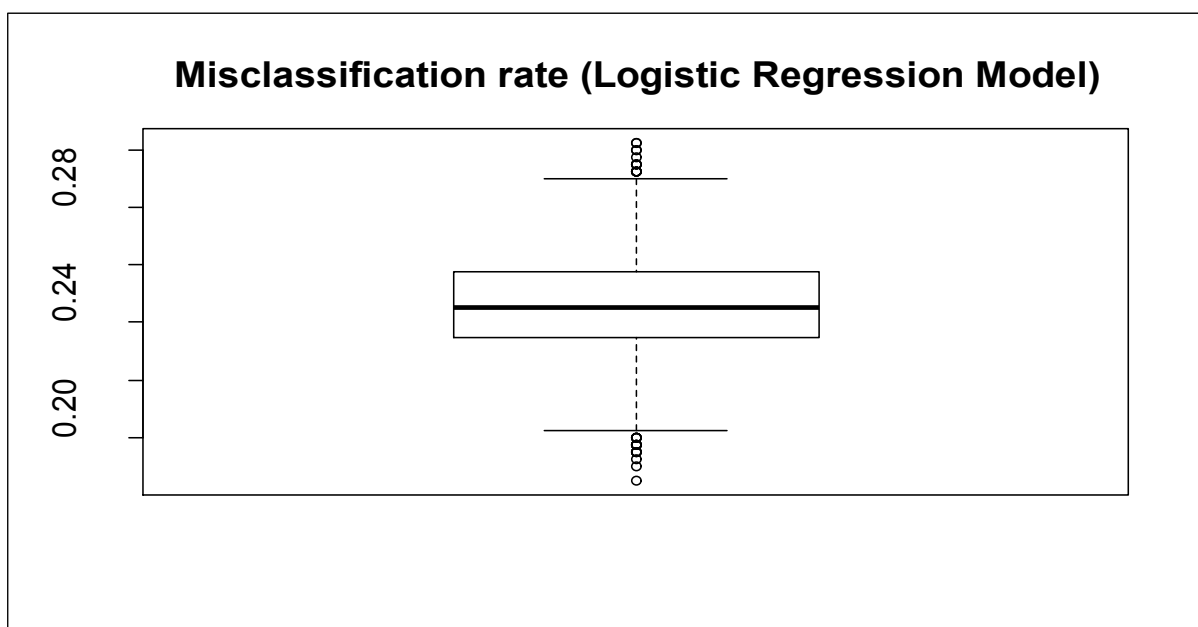
Final answer : 0.310574



(그림 4.2-3)

4.2-4 결과 : Logistic Regression Model (로지스틱 회귀모형)

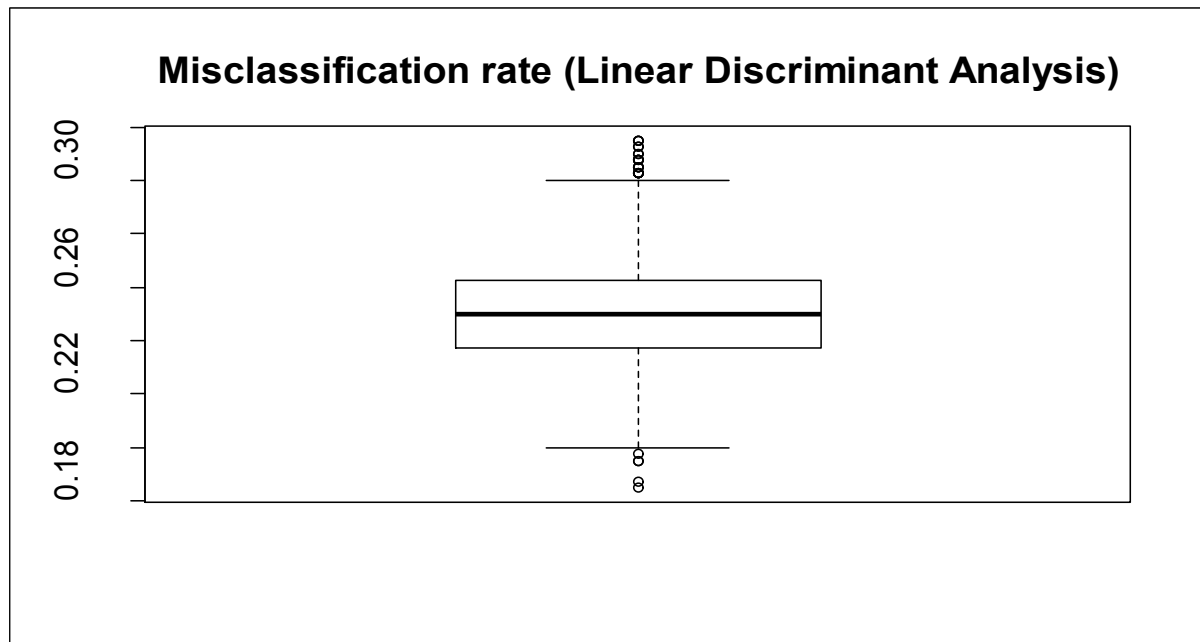
Final answer : 0.245382



(그림 4.2-4)

4.2-5 결과 : Linear Discriminant Analysis (선형판별분석)

Final answer : 0.2508126



(그림 4.2-5)

5. 결론

5.1 결론

5가지의 방법론들을 통한 모의실험을 통해서 얻은 오분류율의 평균값들을 정리해보면 다음과 같다.

K-nearest Neighbor : 0.2999408
Support Vector Machine : 0.259802
Decision Tree : 0.310574
Logistic Regression Model : 0.245382
Linear Discriminant Analysis : 0.2508126

결과적으로 Logistic Regression Model을 적용했을 경우 오분류율이 가장 작게 나왔으며, 반면에 Decision Tree의 경우는 오분류율이 가장 크게 나왔다. 이 결과만을 가지고 해석한다면 Logistic Regression Model이 가장 바람직한 방법이라고 할 수 있을 것이다. 하지만 위의 5가지의 방법론들이 임의로 주어진 데이터가 속해있는 Group을 판별하는 데 사용할 수 있는 방법론의 전부는 아니다. 그 외에 Quadratic Discriminant Analysis(이차형판별분석), Semiparametric Logistic Regression Model(준모수적 로지스틱 회귀모형), Bagging, 그리고 Boosting 등 다양한 방법론들이 존재한다는 것이 알려져 있다. 이러한 방법론들을 토대로 오분류율을 계산할 경우 본 논문에 적용된 방법론들의 결과보다 더 좋은 결과를 얻을 수도 있다. 이는 향후 연구에 남기도록 하겠다.

※ 참고문헌

An Introduction to Statistical Learning

- Springer Texts in Statistics

(Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani)

UCI Machine Learning Repository

([http://archive.ics.uci.edu/ml/datasets/Statlog+\(German+Credit+Data\)](http://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data)))

로지스틱 회귀모형 (wolfpark.hnu.ac.kr/lecture/Regression/ch9_logistic.pdf)

의사결정나무 (openlife.tistory.com/264 , datawaffle.com/f_DataMining/8798)

Support Vector Machine

(http://ko.wikipedia.org/wiki/%EC%84%9C%ED%8F%AC%ED%8A%B8_%EB%B2%A1%ED%84%B0_%EB%A8%B8%EC%8B%A0)

선형판별분석

(jun.hansung.ac.kr/PR/12%20LDA.ppt?)

KNN algorithm

(http://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm)

벡터 공간 분류 (mbm.konkuk.ac.kr/2013-1-정보검색/lecture14.pdf?)

SVM을 이용한 스미싱 탐지 기법

(http://jse.or.kr/insiter.php?design_file=74309.php&article_num=72)

R 기반 다변량 분석 (교우사, 정강모, 김명근 저)

수리통계학개론 (Introduction to Mathematical Statistics) 제6판

(Robert V.Hogg , Joseph W. Mc Kean , Allen Craig)

※ R code

```
german <- read.csv("D:/수업자료/대학/(4)Senior/졸업논문/
SAS, R code/germancredit1.csv",sep="," ,header=T)
germanknn <- read.csv("D:/수업자료/대학/(4)Senior/졸업논문/
SAS, R code/germanknn1.csv",sep="," ,header=T)
germansvm <- read.csv("D:/수업자료/대학/(4)Senior/졸업논문/
SAS, R code/germansvm1.csv",sep="," ,header=T)
head(german)

#### Optimal method
-----

### 1. Tuning (KNN, SVM, Decision Tree 를 사용할 때 가장 최적의 방법을 찾는다.)

install.packages("MASS") # LDA에 필요
library(MASS)
install.packages("class") # KNN에 필요
library(class)
install.packages("e1071") # SVM과 Tuning에 필요
library(e1071)
install.packages("rpart") # Decision Tree에 필요
library(rpart)

# 데이터의 절반 크기인 train data 생성
# (이산형인 설명변수들은 모두 dummy variable로 변환시킴.)
# sampling 방법은 'boot'을 적용함.

## 1. K-nearest Neighbor에서 적절한 K값은?

n <- dim(germanknn)[1]
train.size <- round(n/2)
train <- sample(1:n, size=train.size, replace=F)
german.train <- germanknn[train,-1]
german.test <- germanknn[train,1]

obj1 <- tune.knn(german.train,german.test,k=1:120,
tunecontrol=tune.control(sampling ="boot"))
summary(obj1) # k=73
attributes(obj1)
plot(obj1)
```

2. Support vector machine에서 적절한 gamma와 cost의 값은?

```
n <- dim(germansvm)[1]
test.size <- round(n/2)
test <- sample(1:n, size=test.size, replace=F)
german.test <- germansvm[test,-1]
german.train <- germansvm[-test,]

obj2 <- tune(svm,Default~.,data=german.train,ranges=list(gamma=seq(0.01,0.1,by=0.001),
cost=seq(0.25,2,by=0.25)),tunecontrol=tune.control(sampling ="boot"))
summary(obj2) # gamma=0.026, cost=0.75
attributes(obj2)
plot(obj2)
```

3. Decision Tree를 적용할 시 적절한 해결책은(Pruning)?

```
n <- dim(german)[1]
test.size <- round(n/2)
test <- sample(1:n, size=test.size, replace=F)
german.test <- german[test,-1]
german.train <- german[-test,]

obj3 <- rpart(Default~.,data=german.train)
install.packages("partykit")
library(partykit)
plot(as.party(obj3))
printcp(obj3)
plotcp(obj3) # Size=3, cp=0.049
```

2. 최적의 Logistic Regression Model은?

```
glm.german <- glm(Default~.,data=german,family="binomial")
summary(glm.german)
# Unavailable explanatory variables :
# purpose, savings, employ, status, residence, property,
# age, housing, cards, liable, tele

glm.german <- glm(Default~checkingstatus1+checkingstatus2+checkingstatus3+duration
+history1+history2+history3+history4+amount+installment1+installment2
```

```

+installment3+others1+others2+otherplans1+otherplans2
+foreign,data=german,family="binomial")
summary(glm.german)
# Unavailable explanatory variables : amount

glm.german <- glm(Default~checkingstatus1+checkingstatus2+checkingstatus3+duration
+history1+history2+history3+history4+installment1+installment2+installment3+others1
+others2+otherplans1+otherplans2+foreign,data=german,family="binomial")
summary(glm.german)
# Unavailable explanatory variables : installment

glm.german <- glm(Default~checkingstatus1+checkingstatus2+checkingstatus3+duration
+history1+history2+history3+history4+others1+others2+otherplans1+otherplans2
+foreign,data=german,family="binomial")
summary(glm.german) # Best model~~!!
plot(glm.german)

anova(glm.german,test="Cp")
anova(glm.german,test="LRT")
anova(glm.german,test="Rao")

#### Calculate Misclassification Rate
-----

### 1. K-nearest Neighbor

## KNN의 경우는 train data와 test data의 size가
## 반드시 같아야 함수가 작동하므로 유의해야 한다.

par(mfrow=c(1,1))
n <- dim(germanknn)[1]
test.size <- round(n/2)
error.rate.knn <- NULL
for ( i in 1:5000) {
  test <- sample(1:n, size=test.size, replace=F)
  german.test <- germanknn[test,-1]; actual <- germanknn[test,1]
  german.train <- germanknn[-test,-1]
  pred.knn <- knn(german.train,german.test,actual,k=73)
  error.rate.knn <- c(error.rate.knn, mean(actual != pred.knn))
}

```

```

boxplot(error.rate.knn,main="Misclassification rate (K-nearest Neighbor)")
mean(error.rate.knn) # 0.2999408

### 2. Support Vector Machine

par(mfrow=c(1,1))
n <- dim(germansvm)[1]
test.size <- round(2*n/5)
error.rate.svm <- NULL
for ( i in 1:5000) {
  test <- sample(1:n, size=test.size, replace=F)
  german.test <- germansvm[test,-1]; actual <- germansvm[test,1]
  german.train <- germansvm[-test,]
  svm.german <- svm(Default~.,data=german.train,gamma=0.026,cost=0.75)
  pred.svm <- predict(svm.german,german.test)
  error.rate.svm <- c(error.rate.svm, mean(actual != pred.svm))
}
boxplot(error.rate.svm,main="Misclassification rate (Support Vector Machine)")
mean(error.rate.svm) # 0.259802

### 3. Decision Tree

par(mfrow=c(1,1))
n <- dim(german)[1]
test.size <- round(2*n/5)
error.rate.tr <- NULL
for ( i in 1:5000) {
  test <- sample(1:n, size=test.size, replace=F)
  german.test <- german[test,-1]; actual <- german[test,1]
  german.train <- german[-test,]
  tr.german1 <- rpart(Default~.,data=german.train)
  tr.german2 <- prune(tr.german1,cp=0.049)
  pred.tr <- as.numeric(predict(tr.german2,german.test) >= 0.5)
  error.rate.tr <- c(error.rate.tr, mean(actual != pred.tr))
}
boxplot(error.rate.tr,main="Misclassification rate (Decision Tree)")
mean(error.rate.tr) # 0.310574

### 4. Logistic Regression Model

```

```

par(mfrow=c(1,1))
n <- dim(german)[1]
test.size <- round(2*n/5)
error.rate.glm <- NULL
for ( k in 1:5000) {
  test <- sample(1:n, size=test.size, replace=F)
  german.test <- german[test,-1]; actual <- german[test,1]
  german.train <- german[-test,]
  glm.german <- glm(Default~checkingstatus1+checkingstatus2+checkingstatus3
+duration+history1+history2+history3+history4+others1+others2+otherplans1
+otherplans2+foreign,data=german,family="binomial")
  pred.glm <- as.numeric(predict(glm.german, german.test, type="response") >= 0.5)
  error.rate.glm <- c(error.rate.glm, mean(actual != pred.glm))
}
boxplot(error.rate.glm,main="Misclassification rate (Logistic Regression Model)")
mean(error.rate.glm) # 0.245382

```

5. Linear Discriminant Analysis

```

par(mfrow=c(1,1))
n <- dim(german)[1]
test.size <- round(2*n/5)
error.rate.lda <- NULL
for ( k in 1:5000) {
  test <- sample(1:n, size=test.size, replace=F)
  german.test <- german[test,-1]; actual <- german[test,1]
  german.train <- german[-test,]
  lda.german <- lda(Default~.,data=german.train)
  pred.lda <- predict(lda.german, german.test)$class
  error.rate.lda <- c(error.rate.lda, mean(actual != pred.lda))
}
boxplot(error.rate.lda,main="Misclassification rate (Linear Discriminant Analysis)")
mean(error.rate.lda) # 0.2508126

```

Final Result

```

# K-nearest Neighbor : 0.2999408
# Support Vector Machine : 0.259802
# Decision Tree : 0.310574

```


Logistic Regression Model : 0.245382 # Linear Discriminant Analysis : 0.2508126
--