데이터마이닝 기말과제

2018312824 류채은

Multiple Linear Regression

6.1.

a. 데이터를 학습 세트와 검증 세트로 나누는 이유는 아직 목격하지 않은(unseen) 데이터에 대해 튼튼한 모델을 만드는 것이 목적이기에 과적합(overfitting)을 방지하고 편향을 제거한 데이터로 모델 성능을 평가하기 위해서이다. 학습 세트는 모델을 데이터에 적합하게 만드는데(build) 사용하고, 검증 세트는 모델을 평가하는데 활용된다.

```
> summary(reg)#summary of multiple linear regression model

Call:
lm(formula = MEDV ~ CRIM + CHAS + RM, data = train.df)

Residuals:
   Min     1Q Median     3Q    Max
-23.89  -2.64  -0.21   2.46  38.97

Coefficients:
            Estimate Std. Error t value             Pr(>|t|)
(Intercept) -30.1500     3.1901   -9.45 < 0.0000000000000002 ***
CRIM         -0.2118     0.0351   -6.03          0.0000000048 ***
CHAS          2.0838     1.2987    1.60                  0.11
RM            8.4951     0.5003   16.98 < 0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.06 on 296 degrees of freedom
Multiple R-squared:  0.576,     Adjusted R-squared:  0.571
F-statistic:  134 on 3 and 296 DF,  p-value: <0.0000000000000002
```

그림 1 주택 가격 중앙값에 대한 다중선형회귀모델

b.

$$\hat{MEDV} = -30.1500 - 0.2118 * CRIM + 2.0838 * CHAS + 8.4951 * RM$$

수식 1 중앙 값 예측 수식

그림 1을 통해 **수식 1**를 도출할 수 있다.

c.

예측 값(predicted value)은 20.7996960527438, 실제 값(actual value)은 33이고 예측 오차는 검증 데이터에서 계산하며 예측 값에서 실제 값을 뺀 결과이기에 12.3003039472562가 된다.

d.

i.) 어떠한 예측 변수들이 동일한 것을 측정할 것인지를 확인하기 위해선 다중공선성(Multi Collinearity)를 측정해야 한다. 다중 공선성이란, 독립변수들 간의 강한 상관관계를 뜻한다. 따라서, 다중 공선성을 탐색하기 위해선 변수간 상관관계를 확인해야 하고, 이 때 정확한 비교를 위해 단위 통일을 위한 표준화(scaling)를 했다.

	INDUS	NOX	TAX
INDUS	1.0000000	0.7555741	0.6865849
NOX	0.7555741	1.0000000	0.6573858
TAX	0.6865849	0.6573858	1.0000000

표 1 상관계수(INDUS,NOX,TAX)

행렬의 대각성분을 제외하고 **표 1**를 해석해보면 변수 INDUS와 변수 NOX 사이의 상관계수가 높은 것으로 보아 다중공선성을 의심해 볼 수 있다. 하지만, 이보다 더 정확한 다중공선성 여부를 파악하기 위해선 다중공선성에 대한 진단의 척도인 분산 팽창 인자(Variance Inflation Factor, VIF)를 검사해보는 것이 바람직하다. VIF 값을 계산하기 위해선 car 패키지 설치 및 로드가 필요하다.

train.df$INDUS	train.df$NOX	train.df$TAX
2.73512939	2.54610914	2.06688008

표 2 Variance Inflation Factor for INDUS, NOX, TAX

VIF 값들이 10보다 클 때 다중공선성이 존재한다고 파악하는데 **표 2**를 통해 INDUS, NOX, TAX 사이엔 다중 공선성이 존재하지 않지만 INDUS와 NOX의 VIF 값이 가장 높고 따라서 이 두 변수가 동일한 것을 측정할 확률이 가장 높음을 알 수 있다.

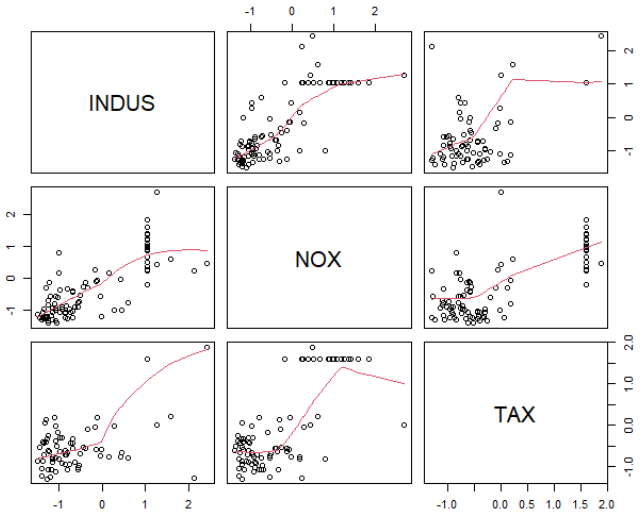

그림 2 변수 간 scatter plot

마지막으로, **그림 2**를 통해 정말로 예측 변수들이 동일한 것을 측정하는지 시각적으로 나타낼 수 있다. **그림 2**에서 데이터의 경향성을 알려주는 빨간 선의 기울기가 양수인 것으로 보아 어느 정도 변수 간 양의 상관관계를 맺고 있는 것을 알 수 있다.

하지만, 다중 공선성이 강하지 않기에 어떠한 예측 변수들이 동일한 것을 측정할 것 같은지를 **그림 2**를 통해 시각적으로 파악하는 건 무리가 있어 보인다.

ii.)

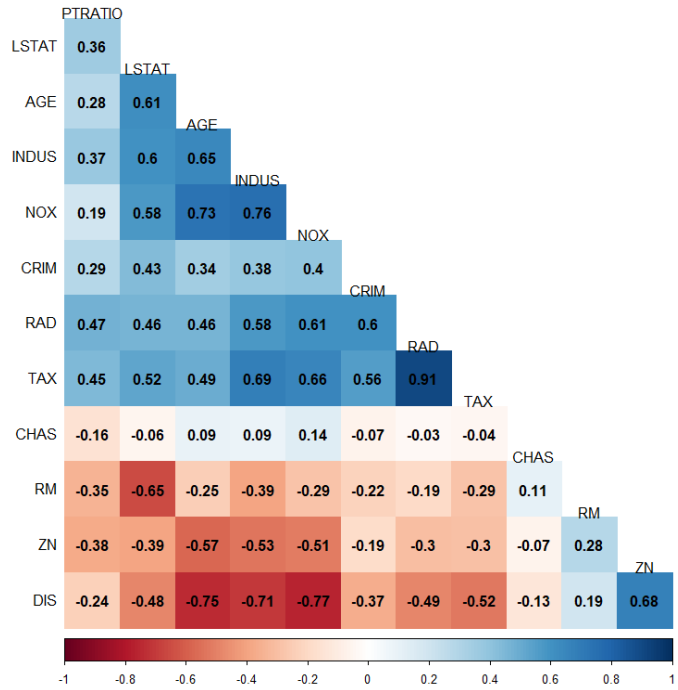

그림 3 수치형 예측 변수 간 상관관계

그림 3은 수치형 예측 변수 12개에 대한 상관관계를 쉽게 파악할 수 있도록 시각화 한 것이다. 짙은 남색일 수록 강한 양의 상관관계를, 짙은 붉은 색일수록 강한 음의 상관관계를 나타낸다. 이를 통해 쉽게 RAD와 TAX 사이의 상관관계(0.91)가 가장 높음을 알 수 있다.

```
     CRIM         ZN      INDUS       CHAS        NOX
1.68916608 2.30700404 3.63012506 1.10689345 4.28263844
       RM        AGE        DIS        RAD        TAX
2.14712620 3.23502187 4.11662435 7.01559998 7.85083196
  PTRATIO      LSTAT
1.76232711 3.20511156
```

표 3 12개 수치형 예측 변수들에 대한 VIF 수치

뿐만 아니라, VIF 값을 계산한 **표 3**을 확인해보았을 때 RAD(7.01)과 TAX(7.85) 값이 가장 높은 것으로 보아 제거하고자 하면 RAX와 TAX를 제거하는 것이 바람직하다. 하지만, 통상적으로 다중 회귀에 있어 VIF 값이 10 보다 클 때만 변수를 제거한다고 한다.

iii)

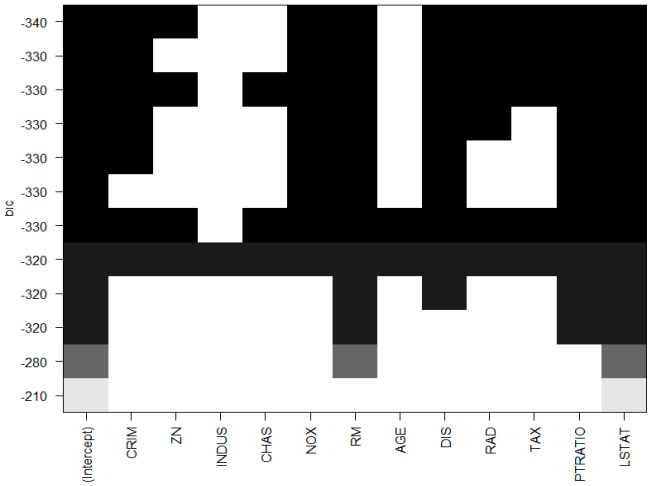

그림 4 최상위 모델 3개를 고를 때 활용한 BIC 지표 시각화

전역 탐색 방법(exhaustive search)를 진행한 후, 선택된 모델의 성능을 높이기 위해 문제에서 추후 성능 비교에 쓰이지 않고 값이 낮을수록 좋은 Bayesian Information Criterion(BIC) 평가 지표를 기준으로 가장 좋은 상위 3개 모델을 선택했다. 이때, **그림 4**는 모델 선택 시 사용한 시각화 자료로, 변수 조합에 따른 BIC 지표를 시각화한 것이다. Y축이 BIC 지표를 나타내기에 변수 명을 기록한 x축을 보고 y축과 비교하여 쉽게 모델을 고를 수 있다.

	first	second	third
	<dbl>	<dbl>	<dbl>
ME	-0.07086869	-0.06336056	-0.01406006
RMSE	6.00728030	5.46753898	5.18980734
MAE	4.31872949	4.07272125	3.70541036
MPE	-6.24990069	-6.96854275	-5.54453426
MAPE	19.76593296	21.01134966	18.93556004

표 4 세 모델간 ME, RMSE 등 평가 지표 비교

이후, 예측 정확도를 비교한 결과는 위에 있는 **표 4**와 같고, 이 표를 통해 세 모델 간 평균오차(Mean Error, ME)와 RMSE(Root Mean Square Error)가 매우 근소한 차이를 보임을 알 수 있다. 그럼에도, ME와 RMSE 모두 가장 낮은 값을 기록한 세 번째 모델이 가장 좋은 모델이다.

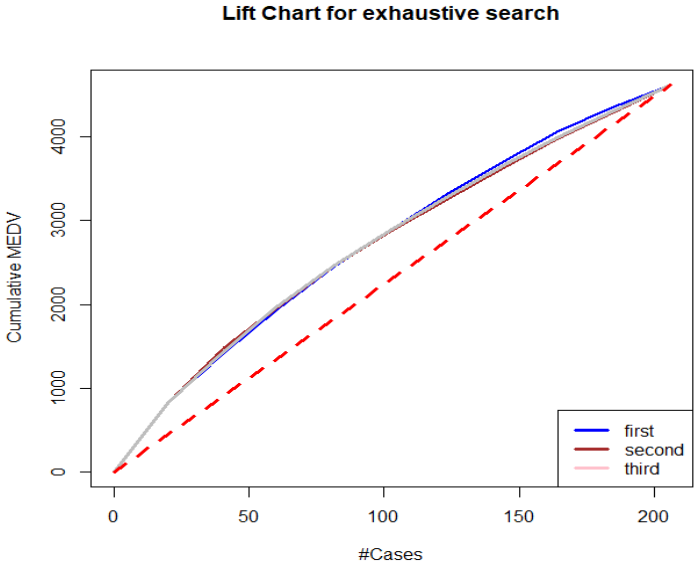

그림 5 리프트 도표를 통한 세 모델의 비교

표 4에서 모든 평가 지표(ME, RMSE, MAE, MPE, MAPE)에 대해 세 모델의 성능 차이가 거의 없었기에 세 모델의 리프트 도표 (**그림 5**) 역시 매우 비슷하여 거의 겹치는 것을 알 수 있다. 그럼에도 세 모델 중 가장 좋은 모델은 모든 손실 지표에서 낮은 값을 기록한 세 번째 모델이다.

K-Nearest Neighbors

7.2

a.

1) ID와 우편번호를 제외한 모든 예측 변수 사용:

```
#read data
bank.df <- read_csv("UniversalBank.csv")
colnames(bank.df)
bank.df <- bank.df[,-c(1,5)]#remove ID and ZIp Code variable
```

그림 6 ID와 우편번호 변수 제외한 모든 변수 사용

2) 분류를 통해 예측할 값 정의

```
#예측할 값 (find)
# :이전 캠페인에서 고객에게 제안된 개인 대출을 받아들인 고객들 480명
find <- whole.norm %>%
  filter(`Personal Loan`==1)
colnames(train.norm)
```

그림 9 분류할 값: 이전 캠페인을 받아들인 고객 480명

3) k = 1일 때 K-최근접 이웃 분류 수행

```
#train a knn classifier with threshold=0.5, (success:1, fail:0)
nn <- knn(train = train.norm[,predictors],
          test = find[,predictors],
          cl = train.norm$`Personal Loan`,k=1)
```

그림 7 KNN분류기 학습(k = 1)

4) 고객들 분류 결과

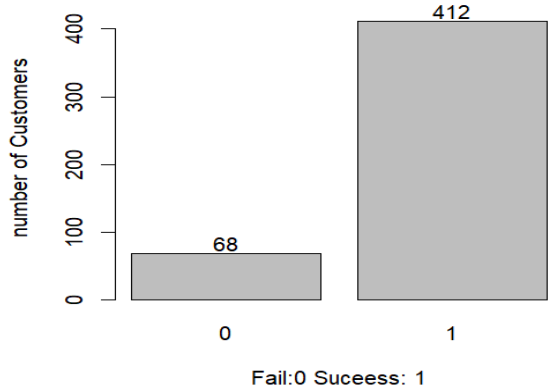

그림 8 KNN 분류기를 통한 분류 결과

그림 9를 통해 이전 캠페인에서 고객에게 제안된 개인 대출을 받아들인 480명 중 68명이 대출을 받아들이지 않을 것이고 412명이 대출을 수락할 것으로 분류되었다.

b.

Confusion Matrix를 활용해 k 값을 1부터 16까지 정확도, Sensitivity, Specifity, Precision, F1-Score를 평가 지표로 평가해보았다.

```
> accuracy.df
    k accuracy sensitivity specificity precision   F1
1   1     0.96        0.66        0.99      0.91 0.76
2   2     0.94        0.43        1.00      0.98 0.60
3   3     0.95        0.57        1.00      0.94 0.71
4   4     0.94        0.45        1.00      0.98 0.62
5   5     0.95        0.55        1.00      0.94 0.69
6   6     0.94        0.42        1.00      0.97 0.59
7   7     0.95        0.49        1.00      0.95 0.65
8   8     0.94        0.40        1.00      0.96 0.56
9   9     0.94        0.44        1.00      0.96 0.60
10 10     0.94        0.39        1.00      0.96 0.55
11 11     0.94        0.43        1.00      0.96 0.59
12 12     0.94        0.37        1.00      0.96 0.54
13 13     0.94        0.39        1.00      0.94 0.56
14 14     0.93        0.35        1.00      0.95 0.51
15 15     0.94        0.40        1.00      0.94 0.56
16 16     0.93        0.34        1.00      0.93 0.50
```

표 5 K값에 따른 평가 지표 수치

표 5를 통해 k = 1일 때 KNN 분류기가 대부분의 평가 지표에서 가장 좋은 성능을 보이는 것을 알 수 있었는데 k가 1이면 분류하고자 하는 데이터와 가장 가까운 데이터 한 개만 활용하기에 과적합이 발생할 확률이 매우 높다.

또한, 고객 유치가 매우 중요한 신생 은행의 입장에서 생각했을 때, 대출을 수락하지 않을 고객을 수락하는 것으로 분류하는 것(False Positive)보다 대출을 수락할 고객을 수락하지 않을 고객(False Negative)으로 분류하는 것을 줄이는 것이 더 중요하다. 즉, 이는 제시된 은행 상황에서 False Negative를 최소화하는 것이 False Positive를 고려하는 것 보다 중요하기에 평가 지표 중 False Negative를 최소화하고자 할 때 쓰이는 Sensitivity를 최우선으로 고려하는 것이 적합하다. 따라서, 올바른 성능 평가를 위해 accuracy 대신 Sensitivity 값을 중심으로 살펴보면 k = 3일 때 Sensitivity 값이 0.57, k = 5일 때 Sensitivity 값이 0.55인 순서로 Sensitivity 값이 가장 높으므로 균형을 잡아주는 k의 값은 3이라고 할 수 있다.

c.

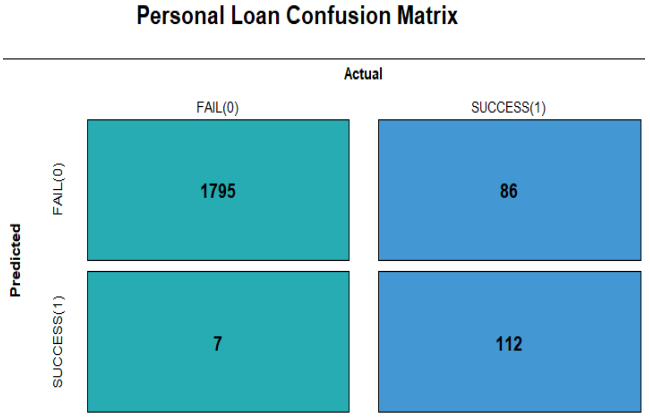

그림 10 최적의 K값(K=3)에 대한 정오행렬표

그림 10은 최적의 값 K=3를 사용하여 검증세트에 대해 정오행렬표를 시각화한 것이다. 거부는 0, 수락은 1임을 나타냈다. 의도했던 대로 Sensitivity(Recall) 수치가 높은

것으로 보아 검증 데이터셋에 대해 일반화가 잘 되었다고 평가할 수 있다.

d.

```
# 최적의 K = 3를 사용하여 고객들(find)을 분류
best.knn <- knn(train = train.norm[,predictors]
          test = find[,predictors],
          cl = train.norm$`Personal Loan`,k=3)
```

그림 11 K = 3인 KNN 분류기로 분류

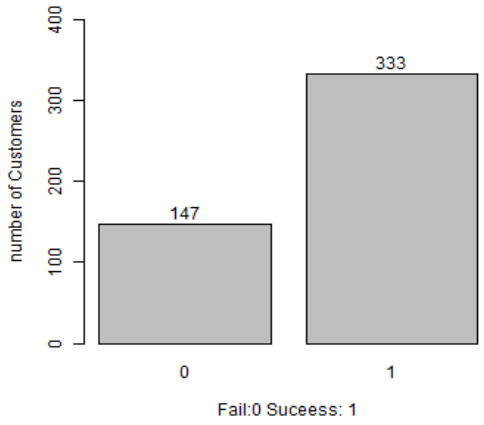

그림 12 k = 3인 KNN 분류기로 고객들을 분류한 결과

그림 11은 K = 3 인 KNN 분류기로 분류하는 코드를 나타내고 **그림 12**는 K = 3인 KNN 분류기로 이전 캠페인에서 제안된 개인 대출을 받아들인 고객들을 분류한 결과를 시각화한 자료이다. K = 3일 때 KNN 분류기는 총 고객 480명을 대출을 수락하지 않을 147명과 대출을 수락할 333명으로 분류하였다. 이를 통해 **a.**에서 K = 1인 KNN 분류기로 분류한 결과(대출 거절: 68명, 대출 수락: 412명)와 비교했을 때, K = 3 인 KNN 분류기가 대출을 안 받아들일 고객이 더 많을 것이라고 예측함을 알 수 있다.

e.

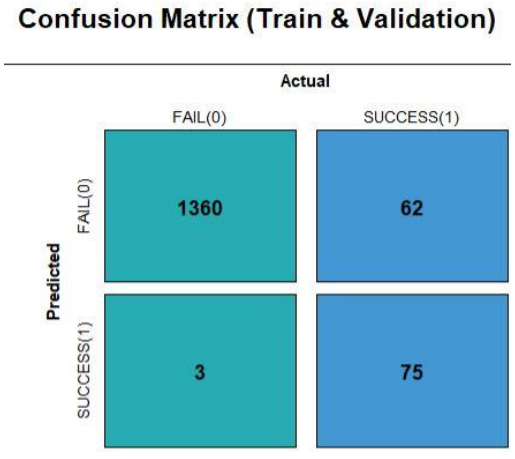

그림 13 학습 및 검증 세트의 정오행렬표

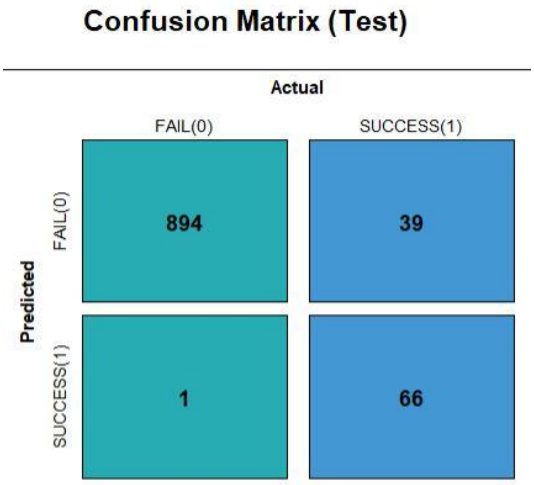

그림 14 평가 세트에 대한 정오행렬표

검증 세트는 1500개 데이터, 평가 세트는 1000개의 데이터를 가지고 있어 **그림 13**과 **그림 14**의 두 정오행렬표에 적힌 숫자의 절대적 크기를 비교하지 않고 비율을 비교하는 것이 정당하다.

이 둘을 비교할 때, **b.**에서 제시했듯 Sensitivity를 기준으로 평가하는 것이 적합하다. Sensitivity 공식은 TP/(TP+FN)이므로 학습 및 검증 세트의 정오 행렬표에선 0.54(=75/(75+62)), 평가 세트의 정오 행렬표에선 0.62(=66/(66+39)이다. 따라서, 평가 세트의 정오행렬표 Sensitivity 값이 학습 및 검증 세트의 정오행렬표 Sentivity값보다 높다. 대부분의 데이터마이닝 알고리즘에선 학습 및 검증 세트에서의 성능이 평가 세트의 성능보다 더 높아 모델이 과적합되기 쉽다. 하지만, 이 문제의 경우 학습 및 검증 세트에서의 모델 성능보다 테스트 세트에서의 모델 성능이 더 높았다. 그 이유는 학습 및 검증 세트의 경우 학습 데이터의 데이터 샘플 분포를 파악해 검증 세트 데이터에 대해 거리를 구해 분류를 했다면, 평가 세트의 경우엔 학습 및 검증 세트를 합친 데이터 샘플 분포를 파악한 후 테스트 세트 데이터에 대한 거리를 구해 분류했기 때문이다. 즉, 학습 및 검증 세트에 대한 분류를 진행할 때보다 평가 세트에 대한 분류를 진행할 때에 있어서 KNN분류기로 예측할 때 참고할 수 있는 데이터가 더 많았기 때문에 평가 세트의 Sensitivity값이 더 높을 수 밖에 없다. 이는 훈련과 예측의 함수가 독립된 형태로 나눠져 있지 않은 knn() 함수의 특징 때문이기도 하다. 따라서, 학습 및 검증 세트에 대해 교차 검증(Cross Validation)을 실행할 경우 테스트 세트에서 보다 성능이 더 높게 나올 수도 있을 것 같다는 예측을 했다.

The Naïve Bayes Classifier

8.2

```
#가변수 생성(MAX_SEV_IR이 1이나 2면 Yes, 그렇지 않으면 0)
accidents.df$INJURY <- ifelse(accidents.df$MAX_SEV_IR>0,"Yes","No
```

그림 15 가변수(INJURY) 생성

a.

	No	Yes	Total
Frequencies:	20721	21462	42183
Proportions:	0.491	0.509	1.000

표 6 INJURY 변수의 데이터 값(YES,NO)별 비율 및 개수

Naïve Rule 에 의거해 추가 적인 정보가 없다면 예측은 INJURY=Yes 가 될 것이다. 그 이유는 위에 있는 **표 6** 을 통해 전체 데이터세트에서 INJURY 변수 데이터의 비율과 개수를 확인했을 때, INJURY 변수 값이 No 인 데이터가 전체에서 차지하는 비중이 0.491, 값이 Yes 인 데이터가 전체에서 차지하는 비중이 0.509 로 사고가 동반될 확률(Yes)이 더 높은 것을 확인할 수 있기 때문이다.

b.

i)

```
, , INJURY = No

          TRAF_CON_R
WEATHER_R 0 1 2
        1 1 1 1
        2 5 1 0

, , INJURY = Yes

          TRAF_CON_R
WEATHER_R 0 1 2
        1 2 0 0
        2 1 0 0
```

표 7 INJURY, TRAF_CON_R, WEATHER_R에 대한 피벗 테이블

ii)

명칭의 간략화를 위해 WEATHER_R은 W, TRAF_CON_R은 T, INJURY = I로 표기했으며 소수점 세자리 수까지 계산했다.

확률	계산 과정	값
$P(I=1\|W=1,T=0)$	$\frac{P(W=1,T=0\|I=1)P(I=1)}{P(W=1,T=0)}$	0.667
$P(I=1\|W=1,T=1)$	$\frac{P(W=1,T=1\|I=1)P(I=1)}{P(W=1,T=1)}$	0
$P(I=1\|W=1,T=2)$	$\frac{P(W=1,T=2\|I=1)P(I=1)}{P(W=1,T=2)}$	0
$P(I=1\|W=2,T=0)$	$\frac{P(W=0,T=0\|I=1)P(I=1)}{P(W=2,T=0)}$	0.167
$P(I=1\|W=2,T=1)$	$\frac{P(W=0,T=1\|I=1)P(I=1)}{P(W=2,T=1)}$	0
$P(I=1\|W=2,T=2)$	$\frac{P(W=0,T=2\|I=1)P(I=1)}{P(W=2,T=2)}$	Nan

표 8 6개 조합에 대한 베이즈 조건부 확률

P(I=1|W=2,T=2)값이 Nan인 이유는 W=2이고 T=2인 순간이 없어서 분모가 0이기에 계산할 수 없기 때문이다.

iii)

```
> use.df
   WEATHER_R TRAF_CON_R INJURY prob.injury estimated
1          1          0    Yes       0.667       Yes
2          2          0     No       0.167        No
3          2          1     No       0.000        No
4          1          1     No       0.000        No
5          1          0     No       0.667       Yes
6          2          0    Yes       0.167        No
7          2          0     No       0.167        No
8          1          0    Yes       0.667       Yes
9          2          0     No       0.167        No
10         2          0     No       0.167        No
11         2          0     No       0.167        No
12         1          2     No       0.000        No
```

표 9 12개의 사고 분류 결과(estimated)

iv)

P(INJURY='Yes'|WEATHER_R=1,TRAF_CON_R=1)
=((3/12)x((2/3)x(0/3)))/
(((3/12)x((2/3)x(0/3)))+((9/12)x((3/9)x(2/9))))
= 0

v)

	actual	nb.class	nb.prob.No	nb.prob.Yes
1	Yes	Yes	0.257	0.74311927
2	No	No	0.847	0.15311909
3	No	No	1.000	0.00000127
4	No	No	1.000	0.00002025
5	No	Yes	0.257	0.74311927
6	Yes	No	0.847	0.15311909
7	No	No	0.847	0.15311909
8	Yes	Yes	0.257	0.74311927
9	No	No	0.847	0.15311909
10	No	No	0.847	0.15311909
11	No	No	0.847	0.15311909
12	No	No	1.000	0.00004628

표 10 나이브 베이즈 분류기 분류 결과

표 10은 나이브 베이즈 분류기로 분류한 결과이다. 표에서 actual은 실제 값, nb.class는 나이브베이즈 분류기의 분류 결과, nb.prob.No는 No라고 분류할 확률, nb.prob.Yes는 Yes라고 분류할 확률을 뜻한다. 나이브 베이즈 분류기를 훈련할 때에 있어서 categorical 변수로 WEATHER_R과 TRAF_CON_R 변수를 바꿔도 나이브 베이즈 분류기 성능이 수기로 분류한 결과보다 좋지 않자 WEATHER_R과 TRAF_CON_R 변수에 대해 one-hot encoding을 진행했다. 또한, 모델 성능을 최대한 높이기 위해서 k-fold validation을 진행했고 k값을 여러 값으로 실험해 본 결과 k=11일 때 가장 좋은 결과를 얻을 수 있어 k=11로 고정했다.

```
> nb.tbl#naive bayes classification result
accuracy.nb
  Correct Incorrect
       10         2
> manual.tbl#manual classification result
accuracy.manual
  Correct Incorrect
       10         2
```

표 11 나이브 베이즈 분류기의 분류와 직접 계산한 결과 비교

표 11을 통해 나이브베이즈 분류기와 수기로 계산했던 정확한 베이즈 분류 결과를 비교해볼 수 있다. 두 방법 모두 정확하게 분류한 데이터가 10개, 오분류 데이터가 2개로 같은 성능을 보이고 있다.

c.

i) 포함시킬 수 있는 변수:

HOUR_I_R,ALIGN_I, WRK_ZONE, WKDY_I_R, INT_HWY, LGT CON_I_R, PROFIL_I_R, SPD_LIM, SUR_CON, TRAF_CON_R, TRAF_WAY, WEATHER_R

ii)

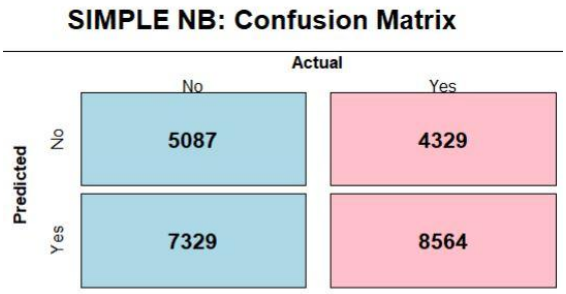

그림 16 모든 데이터를 활용해 계산한 정오행렬

그림 16은 훈련데이터에 대해 나이브 베이즈 분류기를 한 번 훈련했을 때 훈련 데이터에 대한 분류 결과를 정오행렬로 표현한 것이다. 이 나이브베이즈 분류기의 accuracy 값은 0.539로 매우 낮은 훈련 정확도 값을 기록한다.

iii)

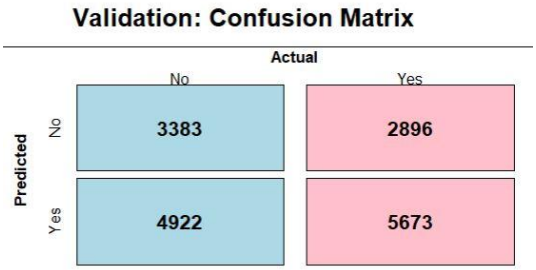

그림 17 검증 데이터셋에 대한 정오행렬

그림 17은 검증 데이터셋에 대한 정오행렬이며 검증 데이터셋에 대해 분류를 했을 때의 정확도는 0.537로 이전 훈련 데이터셋에 대한 분류를 진행했을 때보다 더 낮다. 이는 나이브베이즈 분류기를 훈련시킬 때 검증 데이터가 포함되지 않았기에 당연한 결과다. 하지만, 훈련 데이터셋에 대한 정확도인 0.539와 검증데이터셋에 대한 정확도인 0.537은 매우 근소한 차이를 보이며 정확도 자체가 낮기에 과적합 되었다고 보긴 어렵다. 전체 분류 오차는 이 정오행렬 결과에 따라 1에서 정확도(0.537)을 뺀 값인 0.463이며, 확률로는 46%이다.

iv)

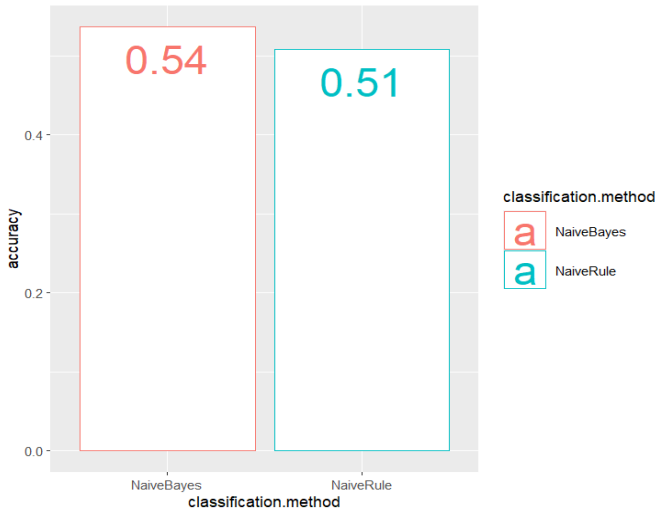

그림 18 나이브 규칙(하늘색)과 비교

그림 18에서 나와있듯 검증 데이터를 사용하여 나이브 베이즈 분류기의 성능(분홍색, 0.54)과 나이브 규칙의 성능(하늘색, 0.51)을 비교했을 때 성능향상 정도는 약 0.3 정도가 되며 매우 근소하다.

v)

P(INJURY=NO|SPD_LIM=5)

= P(INJURY=No, SPD_LIM=5)/P(SPD_LIM=5)이다.

```
> head(freq.tbl)
  INJURY SPD_LIM Freq          p
1     No       5    1 0.0000593
2    Yes       5    2 0.0001185
3     No      10    4 0.0002371
4    Yes      10    1 0.0000593
5     No      15   37 0.0021927
6    Yes      15   36 0.0021335
```

표 12 INJURY값과 SPD_LIM 값이 동시에 나타날 빈도수

이때, **표 12**를 통해 변수 INJURY 값과 SPD_LIM의 값에 따른 빈도수를 피벗 테이블을 통해 확인해 보면, INJURY가 No이고 SPD_LIM이 5인 경우의 빈도수는 1로 매우 작고 이러한 사건이 발생할 확률(p) 또한 0에 가까움을 알 수 있다. 즉, P(INJURY=No, SPD_LIM=5)부분이 0에 매우 가깝게 되기에 P(INJURY=NO|SPD_LIM=5)의 확률 값이 0으로 계산 된다.

Trees

9.1

a.

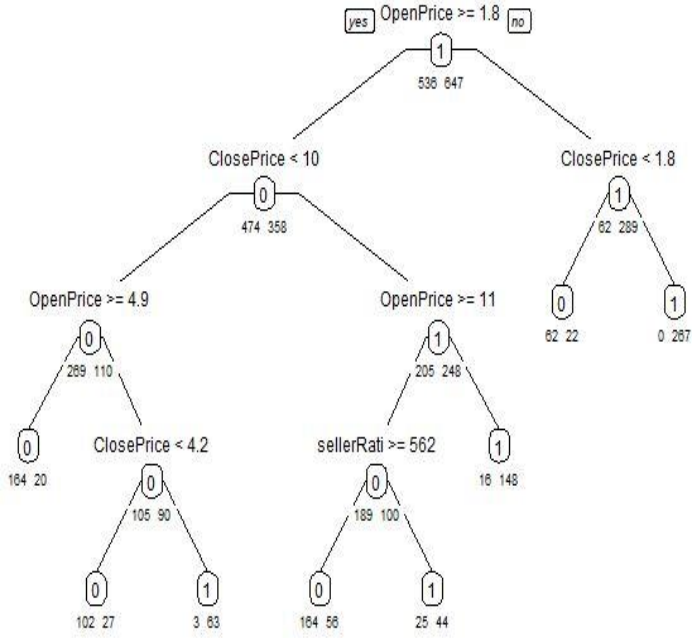

그림 19 트리 분류기의 분류 규칙 시각화

그림 19는 트리 분류기가 어떤 데이터의 경매가 경쟁적일지(1), 아닐지(0)를 판단하는 규칙을 시각화한 것이다.

<규칙>

조건	결과
If (OpenPrice>=1.8)&(ClosePrice<10)&(OpenPrice>=4.9)&(ClosePrice<4.2)	비경쟁적 경매(0)
If (OpenPrice<1.8)&(ClosePrice<10)&(OpenPrice<4.9)&(ClosePrice<4.2)	비경쟁적 경매(0)
If (OpenPrice<1.8)&(ClosePrice<10)&(OpenPrice<4.9)&(ClosePrice>=4.2)	경쟁적 경매(1)
If (OpenPrice>=1.8)&(ClosePrice>=10)&(OpenPrice>=11)&(sellerRating>=562)	비경갱적 경매(0)
If (OpenPrice>=1.8)&(ClosePrice>=10)&(OpenPrice>=11)&(sellerRating<562)	경쟁적 경매(1)
If (OpenPrice>=1.8)&(ClosePrice>=10)&(OpenPrice<11)	경쟁적 경매(1)
If (OpenPrice<1.8)&(ClosePrice<1.8)	비경쟁적 경매(0)
If (OpenPrice<1.8)&(ClosePrice>=1.8)	경쟁적 경매(1)

표 13 분류 레이블을 결정짓는 변수 규칙

b.

```
            Reference
Prediction   0   1
         0 333  92
         1  37 327

               Accuracy : 0.837
```

표 14 검증 데이터에 대한 트리 분류기의 분류 결과

표 13은 검증 데이터에 대해 트리 분류기로 예측을 한 후 정오 행렬로 나타낸 결과이다. 정확도가 0.837인 것을 보아 목격하지 않은 경매 데이터에 대한 일반화 능력이 양호한 것으로 보인다. 하지만, **그림 19**를 보면 경매 마감가격을 뜻하는 변수인 ClosePrice가 분류 결과에 큰 영향을 끼치는 것을 알 수 있는데 만약 새로운 경매가 아직 끝나지 않은 경매이기에 경매 마감 가격이 정해지지 않았다면 트리 분류기의 성능은 매우 떨어질 것이다. 따라서, 이 모델은 실제 사용할 수 없다.

c.

흥미로운 부분은 **표 13**의 규칙 중 네 번째 규칙과 다섯 번째 규칙을 확인해보면 OpenPrice(시작 가격)와 ClosePrice(마감 가격)가 가장 클 때 결국 경쟁적인 경매인지의 여부는 sellerRating(판매자의 등급)에 의해 결정된다는 것이다. 개인적으로 예측하기론 판매자의 등급이 높을 수록 판매자에 대한 신뢰감이 높게 생겨 입찰이 많을 거라고 생각했는데 내 예측과 반대로 오히려 판매자의 등급이 낮을 경우 경매가 경쟁적이었고 판매자의 등급이 높을 때 경매가 비경쟁적이었다.

반면, 흥미롭지 못한 정보는 쉽게 예측할 수 있는 부분에 대한 것이다. 시작 가격보다 마감 가격이 낮은 경우를 **표 13**에선 첫 번째 규칙과 두 번째 규칙에서 찾을 수 있는데, 이땐 당연히 판매 상품에 대한 가격이 시작 시점에 비해 마감하는 시점에서 낮기 때문에 사람들이 경매 기간 동안 입찰을 하려 하지 않은 경우로 비경쟁적 경매로 결론이 난다. 또한, 시작 가격에 비해 마감가격이 높은 경우(**표 13** 마지막 규칙)엔 반대의 상황으로 당연히 경쟁적 경매가 된다.

d.

ClosePrice 변수를 훈련 데이터와 검증 데이터에서 제거했다. 이후, 트리 분류기의 성능을 높이기 위해 rpart() 함수에 내장되어있는 교차 검증(cross validation)의 fold 개수를 나타내는 파라미터인 xval을 5로 했다. 즉, 훈련 데이터셋에 한해 5-fold cross validation을 진행했다.

또한, 가지치기를 할 때 필요한 파라미터인 복잡도 정도를 나타내는 cp값은 *new.tree$cptable[which.min(new.tree$cptable[,"xerror"]),"CP"]* 코드를 통해 손실 값을 최소화하는 cp 값이 자동으로 설정되도록 코드를 구성했다. 이를 통해, 자동적으로 가지치기가 최적화된다.

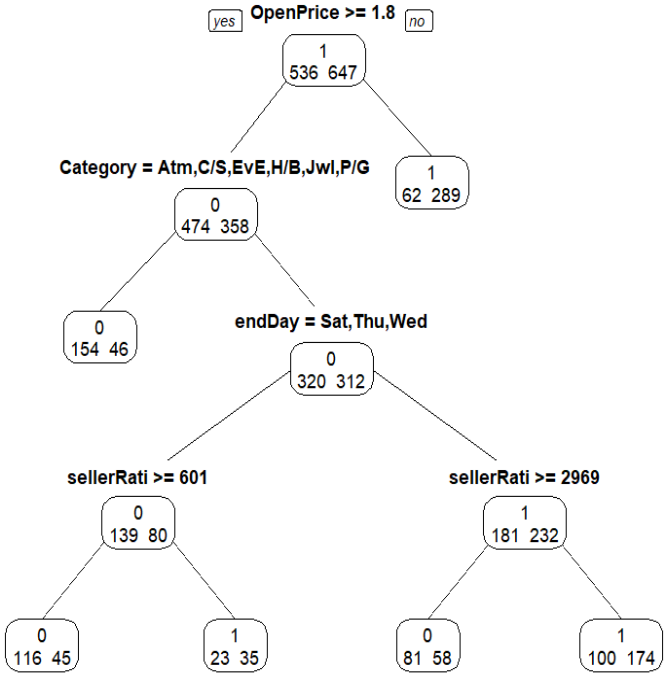

그림 20 최적으로 가지치기한 트리 분류기

<규칙>

조건	결과
If (OpenPrice ≥1.8)&(Category=Atm,C/s,EvE,H/B,Jwl,P/G)	비경쟁적(0)
If (OpenPrice ≥1.8)&(Category≠Atm,C/s,EvE,H/B,Jwl,P/G)&(endDay=Sat,Thu,Wed)&(sellerRating≥601)	비경쟁적(0)
If (OpenPrice ≥1.8)&(Category≠Atm,C/s,EvE,H/B,Jwl,P/G)&(endDay=Sat,Thu,Wed)&(sellerRating<601)	경쟁적(1)
If (OpenPrice ≥1.8)&(Category≠Atm,C/s,EvE,H/B,Jwl,P/G)&(endDay≠Sat,Thu,Wed)&(sellerRating≥2969)	비경쟁적(0)
If (OpenPrice ≥1.8)&(Category≠Atm,C/s,EvE,H/B,Jwl,P/G)&(endDay≠Sat,Thu,Wed)&(sellerRating<2969)	경쟁적(1)
If (OpenPrice<1.8)	경쟁적(1)

표 15 두 번째 분류나무 모델의 규칙

e.

사용하기에 적합하지 않은 ClosePrice를 제외하고 OpenPrice와 Category가 가장 좋은 두 개의 예측 변수임을 **그림 20**을 통해 알 수 있다. 전체 데이터에 대해 트리 분류기로 레이블을 예측한 후 가장 좋은 두 개의 예측 변수인 OpenPrice와 Category에 대해 산점도를 그린 결과는 **그림 21**과 같다. 하지만, Category 변수가 범주형(categorical) 변수이기에 시각화 자료를 통해 분할하기 어려웠으므로 첫 번째로 좋은 변수 OpenPrice와 연속형(continuous) 변수인 세 번째로 좋은 변수 sellerRating을 사용하여 다시 산점도를 **그림 22**로 시각화하였다. 또한, 전체 데이터를 활용할 경우 평균과 많이 떨어져있는 이상치(outliers)값들이 많아 **그림 22**에선 더 나은 시각화를 위해 전체 데이터 대신 검증 데이터를 활용했다. 소수의 이상치가 시각화를 저해하지 않도록 x축과 y축의 범위를 제한도 했다. 마지막으로, 검증 데이터셋에 있는 데이터 789개를 연결하는 것은 시각화 결과를 복잡하게 하기에 **그림 20**에 보이듯 트리 분류기의 세 구분선 (sellerRating==601,sellerRating==2969,OpenPrice==1.8)을 시각화해 분류 결과에 대한 이해를 높였다.

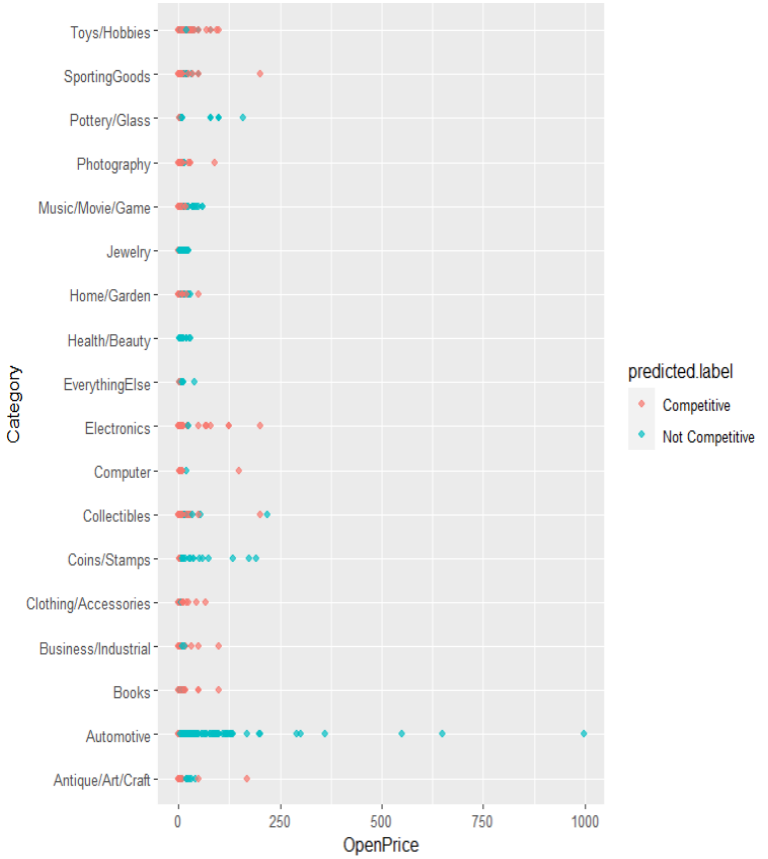

그림 21 두 축(Category, OpenPrice)으로 시각화 한 예측 결과

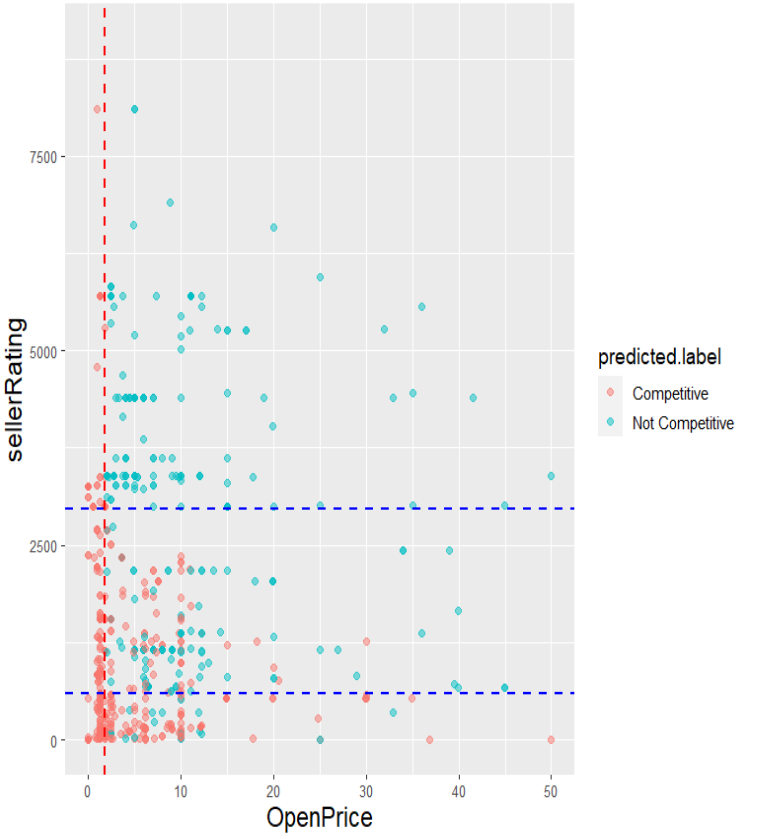

그림 22 최종 트리 모델 예측 결과 시각화

그림 22의 구분 선에 따라 분할된 공간에 하나의 색깔이

대부분 압도하는 것을 보아 **그림 22**만 봐선 예측 결과가 꽤 합리적이라고 해석할 수 있다. 하지만, **그림 22**를 통해 해당 데이터를 분류할 때에 있어서 수직 구간 혹은 수평 구간을 나누어 분류하는 것이 최적의 방법이 아님 역시 알 수 있다. **그림 22**를 보면 대다수의 파란색 점(비경쟁적 경매)들이 오른쪽 윗 부분에 분포되어있고 대부분의 빨간색 점(경쟁적 경매)들이 왼쪽 아래 부분에 분포되어있는 것으로 보아 트리 분류기를 통해 수직적 혹은 수평적 선으로 분류하기 보다는 오른쪽 위와 왼쪽 아래를 가를 수 있는 음수의 기울기와 양수의 y절편을 갖는 구분선을 활용하는 것이 더 좋을 것이다. 즉, 수직 혹은 수평 분류선을 갖는 Tree Classifier 보다 기울어진 구분 선을 활용할 수 있는 Multiple Linear Regression과 Support Vector Machine을 활용해 분류할 때 성능이 더 향상될 것으로 예측한다.

f.

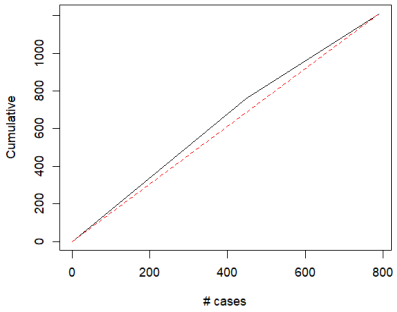

그림 23 향상차트

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 236 106
         1 134 313

               Accuracy : 0.696
```

표 16 분류표

향상 차트와 분류표를 통해 예측 성능이 좋지 않음을 알 수 있다. 향상 차트를 보면, Naïve Rule인 빨간 점선에 비슷하게 트리 분류기의 성능이 그려지는 것을 볼 수 있는데 이는 예측 성능이 좋지 않음을 의미한다. 또한, 분류표를 통해 정확도 69%정도인 것을 알 수 있어 이 또한 모델의 성능이 양호하지 않음을 알려준다.

g.

	pruned.tree$variable.importance
OpenPrice	78.8
sellerRating	26.0
Category	25.4
endDay	12.4
Duration	5.0
currency	4.1

표 17 트리 모델로 평가한 변수 별 중요성

표 17은 경쟁적 경매의 여부를 예측할 때에 있어 트리 모델이 평가한 변수 별 중요성이다. 이를 통해 판매자 경매 조건에선 시작 가격(OpenPrice), 경매 마감일(endDay), 경매 기간(Duration), 통화(currency) 순서로 중요한 것을 알 수 있다. 따라서, 판매자에게 경쟁 경매가 이루어질 수 있도록 조언을 한다면 **그림 20**을 함께 참고하여 시작 가격 변수의 값이 1.8보다 작도록 하는 것이 최우선 전략임을 알려줄 수 있다.

Neural Nets

11.3

a.

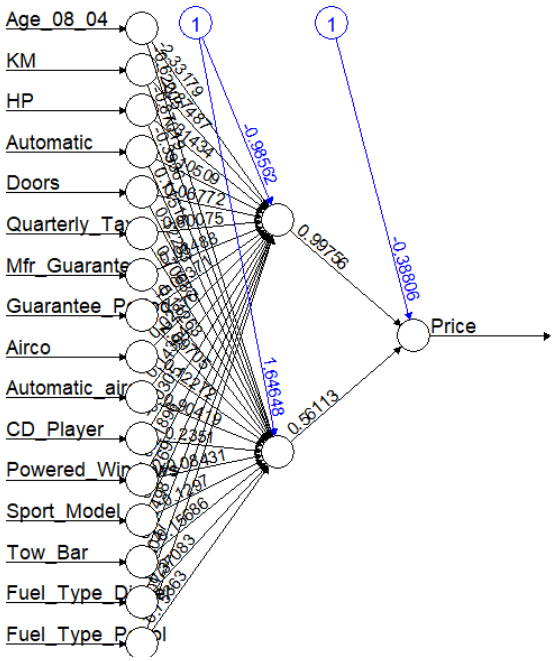

그림 24 학습 데이터에 적합한 2개의 노드를 갖는 신경망 모델

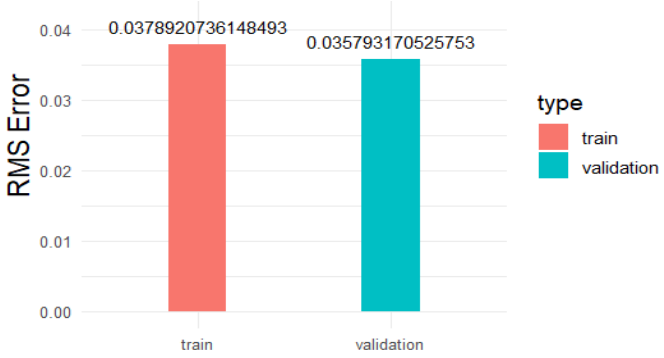

그림 25 RMS 오차 비교(훈련 데이터와 검증 데이터)

그림 24는 2개의 노드를 갖는 신경망 모델을 학습 데이터에 훈련시킨 후 갱신된 가중치(weights, bias)와 함께 나타낸 것이다. 또한, **그림 25**는 학습을 마친 후 두 개의 노드를 가진 이 모델을 각 학습 데이터와 검증 데이터를 예측하여 RMS 오차를 계산한 결과를 막대 그래프로 나타낸 것이다. 이 모델의 경우 검증 데이터에 대한 오차(0.036)가 학습 데이터에 대한 오차(0.038)보다 더 낮기에 언더피팅(Underfitting)이 되었다고 볼 수 있다. 또한, 언더피팅이 일어난 이유로는 적은 노드 수와 적은 은닉층의 수로 예측할 수 있다.

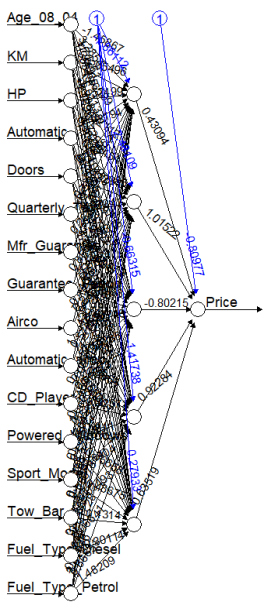

그림 26 NN1

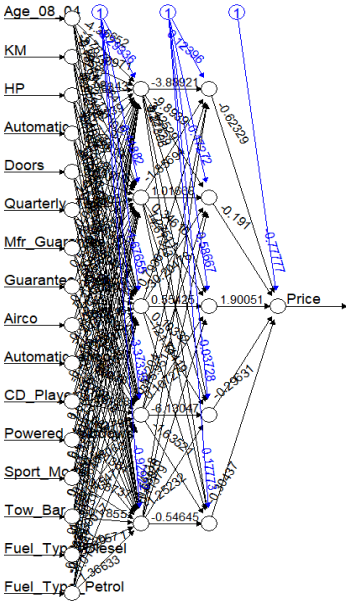

그림 27 NN2

그림 26은 5개의 노드를 갖는 하나의 은닉층을 가진 신경망(NN1)을, **그림 27**은 각 층에 5개의 노드를 갖는 두 개의 은닉층을 갖는 신경망(NN2)을 학습된 가중치와 함께 시각화한 것이다.

model	a.nn	nn1	nn2
hidden layer	1.000	1.000	2.000
number of nodes	2.000	5.000	5.000
train error	0.038	0.034	0.034
validation error	0.036	0.039	0.038

표 18 세 신경망 요약

표 18은 지금까지의 세 모델의 구조와 오차를 요약한 결과이다.

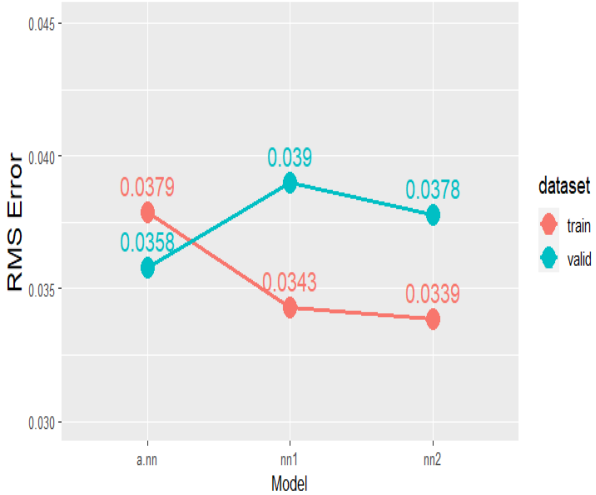

그림 28 세 가지 모델 비교

그림 28은 세 가지 신경망 모델을 비교한 내용으로, 분홍색 선은 학습 데이터에 대한 RMS 오차를 나타내고 하늘색 곡선은 검증 데이터에 대한 RMS 오차를 나타낸다. 또한, x축에 놓인 모델은 차례로 2개의 노드를 갖고 하나의 은닉층을 사용한 모델(a.nn), 5개 노드를 갖는 하나의 은닉층을 가진 모델(nn1), 그리고 각 층에 5개 노드를 갖는 두 개의 은닉층을 가진 모델(nn2)를 나타낸다.

i) 그림 28을 통해 x축의 오른쪽 방향으로 진행할수록 학습 데이터에 대한 RMS 오차를 나타내는 분홍색 선이 감소하는 것을 보며 층과 노드의 수를 증가시킴에 따라 학습 데이터에 대한 RMS 오차가 감소하는 것을 알 수 있다.

ii) 훈련 데이터의 경우와는 달리 검증 데이터에 대한 RMS 오차는 모델의 복잡도와 반비례하진 않는다. 이는 **그림 28**를 통해 알 수 있는데, 검증 데이터에 대한 RMS 오차가 가장 낮을 때는 가장 간단한 구조를 가진 2개의 노드와 하나의 은닉층을 사용한 모델(nn1)을 활용했을 때다.

iii) 층과 노드의 적절한 수를 찾기 위해 추가적 실험을 진행하여 밑에 있는 **그림 29**로 시각화하였다.

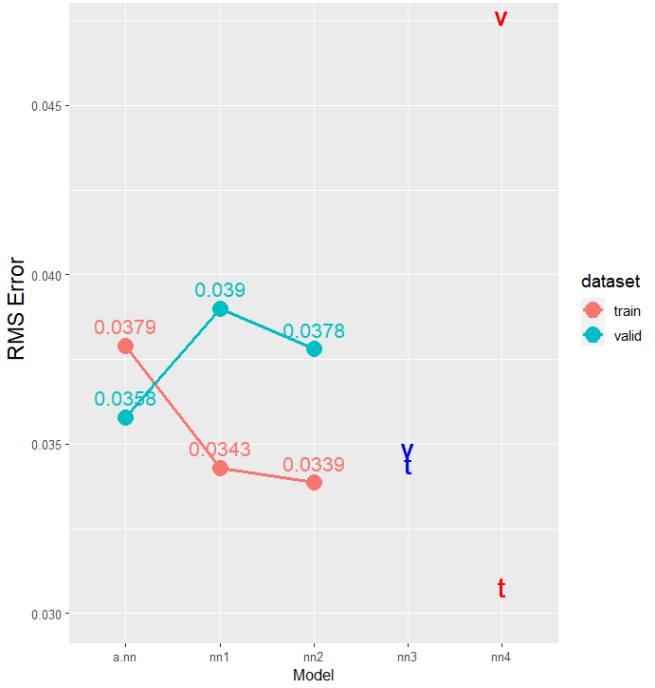

그림 29 다섯 가지 모델 비교

이 그림은 **그림 28**에 데이터를 추가한 것으로 t는 학습 데이터에 대한 RMS 오차, v는 검증 데이터에 대한 RMS 오차를 나타낸다. 또한, 파란색 **t**와 **v**는 층별 노드 수가 5개인 은닉층 3개를 지닌 모델 nn3의 오차이며 빨간색 **t**와 **v**는 층별 노드 수가 10개인 은닉층 3개를 지닌 모델 nn4의 오차이다. 해당 그림은 x축의 오른쪽 방향으로 진행될 수록 모델의 구조가 복잡해짐에도 불구하고 성능이 좋아지지 않는 것을 보여준다. 특히, 은닉층의 수와 노드 개수가 가장 많은 마지막 모델 nn4의 경우 훈련 오차와 검증 오차의 간극이 가장 커 심각한 과적합이 발생했음을 알 수 있다. 결론적으로 지금까지 실험한 모델 다섯 가지 중에서는 훈련 오차와 검증 오차의 차이가 가장 적어 과적합이 발생하지 않고 낮은 검증 오차로 인해 일반화 성능이 가장 좋은 모델 a.nn이 가장 좋은 모델이다. 즉, 적절한 층과 노드의 수는 a.nn의 구조처럼 은닉층 한 개와 노드 두 개다..