채널 G
2016125077 최재혁
2016126049 박희재
2016125069 조세희

데이터 사이언스

- 4주차 팀 과제 (의사결정트리)

목 차

- 과제 설명

  DataSet을 바탕으로 의사결정나무 생성하기.

- 과제 목표

  주어진 DataSet을 통해 의사결정나무를 만들며, 전반적으로 의사결정 나무에 대해 이해하며 원하는 목적대로 분석할 수 있다.

- 사용 Tool

  분석용 언어 : R

목 차

A~J까지 순서대로 설명할 예정입니다

- 과제 A,B,C
- 과제 D,E
- 과제 F,G,H
- 과제 I,J

과제 A,B

- 과제 A – German Credit, 또는, Titanic Dataset나 외의 Dataset결정하기

  - Titanic Dataset으로 설정하였다.

- 과제 B – Dataset을 다운받고 토론하고, 올바른 목표변수 설정하기

  - 목표변수를 survived,즉 살아남을 수 있는지 여부로 결정하였다.

과제 C

- 과제 C – 적절한 데이터 전처리, 탐색과정을 거치기
  - 첫번째로 Titanic Dataset에서 age와 fare을 각각 범주화 하여 전처리 하였다. <사진1>참조
  - 두번째로 <사진2>처럼 6가지 속성만을 추출하며 전처리 하였고
  - 세번째로 결측치가 들어있는 행을 데이터셋에서 삭제하는 na.omit()함수를 통해 전처리 하였다.

groupedage	fare	groupedfare
adult	211.3375	200
youth	151.55	200
youth	151.55	200
adult	151.55	200
adult	151.55	200
adult	26.55	0
adult	77.9583	100
adult	0	0
adult	51.4792	100
old	49.5042	0
adult	227.525	200
youth	227.525	200
youth	69.3	100
adult	78.85	100

<사진 1>

```
#데이터 프레임 생성
df <- data.frame(df$survived,df$sex,df$sibsp,df$parch,df$groupedage,df$groupedfare)

#항목 정해줌 기존의 age&fare는 그룹화된 groupedage&fare로 대체되기에 뺐다
names(df)=c("survived","sex","sibsp","parch","groupedage","groupedfare")


#null값 있는 행 삭제
df <- na.omit(df)
```

<사진 2>

과제 D

- 과제 D – 분류 예측 정확도에 가장 영향력 있는 변수를 찾을 수 있는 방법과 찾은 변수

  - 다중선형회귀 분석을 통해 모든 변수 중에서 가장 영향력 있는 변수를 찾아낸다. <사진1>의 코드를 통해 알 수 있다.

  - P-value 값이 0.05보다 압도적으로 작은 Sex, Sibsp, Groupedfare 세 변수가 가장 영향력이 셀 것이며, 그 외의 변수들보다 훨씬 유의미한 변수들이다.

```
#모든변수에서| 다중선형회귀분석으로 영향력있는 변수 찾기
model <- lm(survived~.,data=df)
summary(model)
```

<사진 1> - Rstudio 함수

```
Residuals:
    Min      1Q  Median      3Q     Max
-0.9468 -0.1918 -0.1917  0.2882  1.0217

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     0.7118802  0.0250020  28.473  < 2e-16 ***
sexmale        -0.5201101  0.0241158 -21.567  < 2e-16 ***
sibsp          -0.0479100  0.0116914  -4.098 4.43e-05 ***
parch          -0.0108600  0.0142090  -0.764    0.445
groupedageold  -0.1195979  0.1131304  -1.057    0.291
groupedageyouth -0.0000841  0.0229695  -0.004    0.997
groupedfare     0.0014142  0.0001953   7.240 7.64e-13 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.403 on 1302 degrees of freedom
Multiple R-squared:  0.3157,    Adjusted R-squared:  0.3126
F-statistic: 100.1 on 6 and 1302 DF,  p-value: < 2.2e-16
```

<사진 2> - 사진1 실행결과

과제 E

- 과제 E – 데이터를 학습용, 시험용 데이터 셋으로 나누기 위하여 목표변수를 기준으로 층화 추출한다.
  - <사진1> 처럼 createDataPartition()을 이용하여 층화 추출한다.
  - 파라미터에 y=survived을 넣어 survived를 기준으로 층화 추출한다.
  - 파라미터에 p=0.7을 대입하여 트레이닝 셋을 70%로 추출, 나머지 30%는 테스트셋으로 분리된다.

```
#층화추출

set.seed(100) #reproducability setting
intrain<-createDataPartition(y=df$survived, p=0.7, list=FALSE)

# 트레이닝 셋 설정
train<-df[intrain, ]

# 테스트셋 설정
test<-df[-intrain, ]
train
test
```

<사진 1>

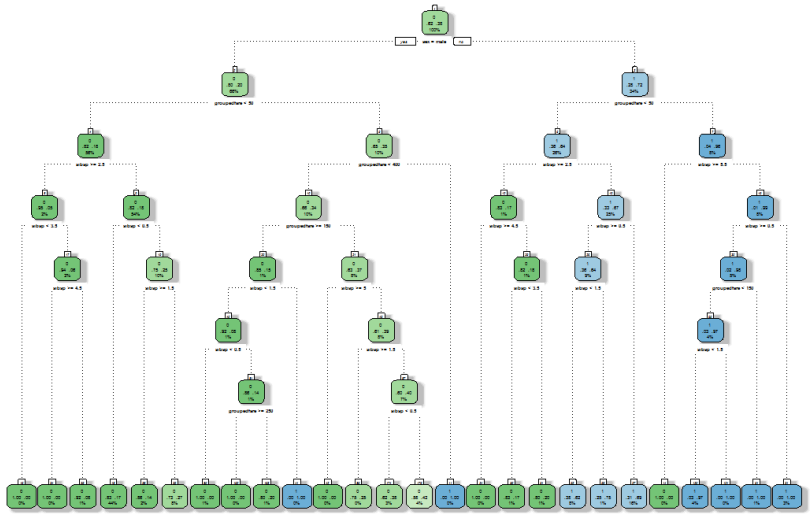
과제 F

- 과제 F – minsplit, minbucket, maxdepth, method, cp 각각 의미하는 바.

- minsplit : 상위 노드에서 추가로 spilt 할 수 있는 최소 관측치 수를 말한다.
  쉽게 말하면 spilt하기 위해 노드에 존재해야 하는 최소 관측치 수를 의미한다.
  기본값은 20이고, 상위 노드에 레코드가 20개 미만인 경우 터미널 노드라는
  레이블이 지정된다
- minbucket : minbucket로 설정한 값보다 오분류 건수가 많은 경우, depth가 추가되면서
  더 정확하게 분리한다. 그렇기에 minbucket로 설정한 값이 작을수록 트리가 잘
  게 쪼개진다.
- Maxdepth : 의사결정나무의 최대 깊이를 설정값 이상으로 자라지 못하게 제한하는
  파라미터이다.
- Method : 분류 분석 뿐만 아니라 포아송, 회귀분석 등 여러 분석 중 하나를 선택해
  분석해볼 수 있는 파라미터이다. Class를 선택했다
- CP : 정지 매개변수로 이 기준을 충족하지 않는 분할을 식별하고 너무 멀리 가기 전에
  잘라낼 수 있기 때문에 분할 검색 속도를 높이는 데 도움이 된다.

과제 G-1

- 과제 G – 적절한 예측변수 집합을 선정하고 Decision tree 모델을 생성하여 정확도를 측정.
  - 과제D에서 다중선형회귀 분석을 통해 얻었던 결과를 통해
    P-value 값이 0.05보다 압도적으로 작은 변수인 Sex, Sibsp, Groupedfare를 선정.
  - Rpart함수로 3가지 변수를 예측변수로 fit를 만들어, 의사결정 트리를 만듬.
  - as.factor()함수를 이용하여 요인으로 변환을 해야 confusionMatrix의 입력값을 맞출 수 있다.



<사진 1>- 의사결정트리

```r
# 최적 변수로 fit생성(가지치기 전)
fit <- rpart(survived~sex+sibsp+groupedfare,data=train,
             cp=-1,minsplit=2,minbucket=1,method = "class")

#트리 만들고 plot찍기
fit
plot(fit)
text(fit)
plotcp(fit)
printcp(fit)
fancyRpartPlot(fit)


#의사결정 모델을 이용한 테스트셋 예측값
rrpartpred<-predict(fit, test, type='class')
rrpartpred <- as.factor(rrpartpred)
#테스트셋의 실제값
survivpred <- as.factor(test$survived)
#테스트셋의 예측값과 실제값 비교. 성능테스트
confusionMatrix(rrpartpred,survivpred)
```

<사진 2>- Rstudio 코드

과제 G-2

- 과제 G – 적절한 예측변수 집합을 선정하고 Decision tree 모델을 생성하여 정확도를 측정.
  - 층화추출 할 때에 분리한 테스트셋으로 정확도(정분류율)를 측정했으며, 트리의 정확도는 트리의 정확도는 0.8087이며 Sensitivity는 0.8589 이다. <사진 1>참고.
  - Titanic 데이터를 전처리하지 않고, 적절한 예측변수들로 만든 트리는 정확도(정분류율)가 0.7955, 민감도가 0.8442로 전처리한 모델에 비해 성능이 떨어짐을 알 수 있다. <사진 2>참고.

```
          Reference
Prediction   0   1
         0 207  41
         1  34 110

               Accuracy : 0.8087
                 95% CI : (0.7662, 0.8464)
    No Information Rate : 0.6148
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.5925

 Mcnemar's Test P-Value : 0.4884

            Sensitivity : 0.8589
            Specificity : 0.7285
         Pos Pred Value : 0.8347
         Neg Pred Value : 0.7639
             Prevalence : 0.6148
         Detection Rate : 0.5281
   Detection Prevalence : 0.6327
      Balanced Accuracy : 0.7937
```

<사진 1>- 데이터를 전처리한 최적 예측변수로 이뤄진 트리 정확도

```
          Reference
Prediction   0   1
         0 168  33
         1  31  81

               Accuracy : 0.7955
                 95% CI : (0.7465, 0.8388)
    No Information Rate : 0.6358
    P-Value [Acc > NIR] : 6.398e-10

                  Kappa : 0.5568

 Mcnemar's Test P-Value : 0.9005

            Sensitivity : 0.8442
            Specificity : 0.7105
         Pos Pred Value : 0.8358
         Neg Pred Value : 0.7232
             Prevalence : 0.6358
         Detection Rate : 0.5367
   Detection Prevalence : 0.6422
      Balanced Accuracy : 0.7774
```

<사진 2>- 데이터를 전처리 x, 최적 예측변수로 이뤄진 트리 정확도

과제 H-1

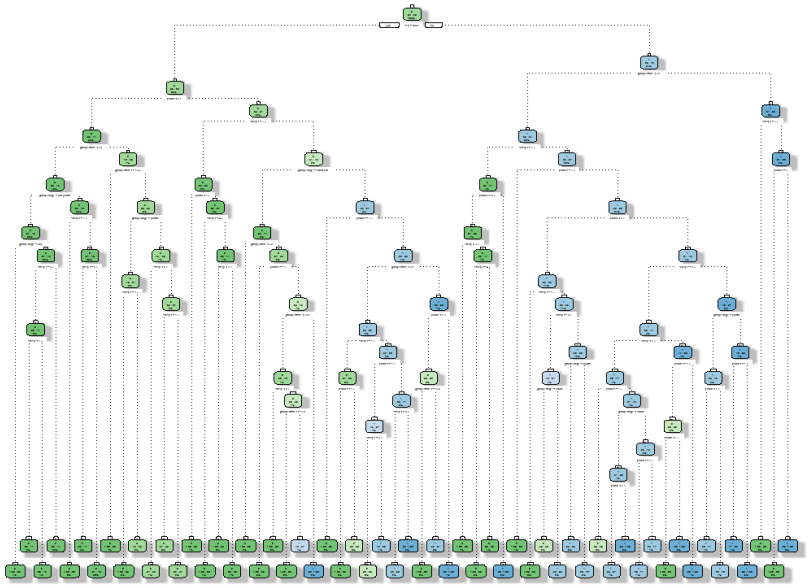
- 과제 H – 모든 변수를 포함하여 모델을 생성하고 정확도를 측정하고 위 문항에서의 결과와 비교하라

  - survived를 제외한 모든변수 sex, sibsp, parch, groupedage, groupedfare를 넣은 fit2 생성.

  - fit2로 트리를 생성. Fit로 만든 트리(최적변수만 포함한 트리)보다 훨씬 복잡 하다.
    최대 트리 깊이가 fit : 7, fit2 : 10이며, 내부노드는 fit : 24개, fit2 : 57개
    뿌리노드는 fit : 26, fit2 : 58개.. fit, fit2의 사진비교는 다음페이지에 있다. <사진2> 참고.

```
#모든 변수로 이뤄진 fit2 생성
fit2 <- rpart(survived~. ,data=train,cp=-1,minsplit=2,
              minbucket=1,method = "class") #모든 변수 포함

#트리 만들고 plot찍기
plot(fit2)
text(fit2)
plotcp(fit2)
printcp(fit2)
fancyRpartPlot(fit2)

#의사결정 모델을 이용한 테스트셋 예측값
rrpartpred2<-predict(fit2, test, type='class')

#모든변수로 만든 트리 성능테스트
rrpartpred2 <- as.factor(rrpartpred2)
survivpred <- as.factor(test$survived)
confusionMatrix(rrpartpred2,survivpred)
```

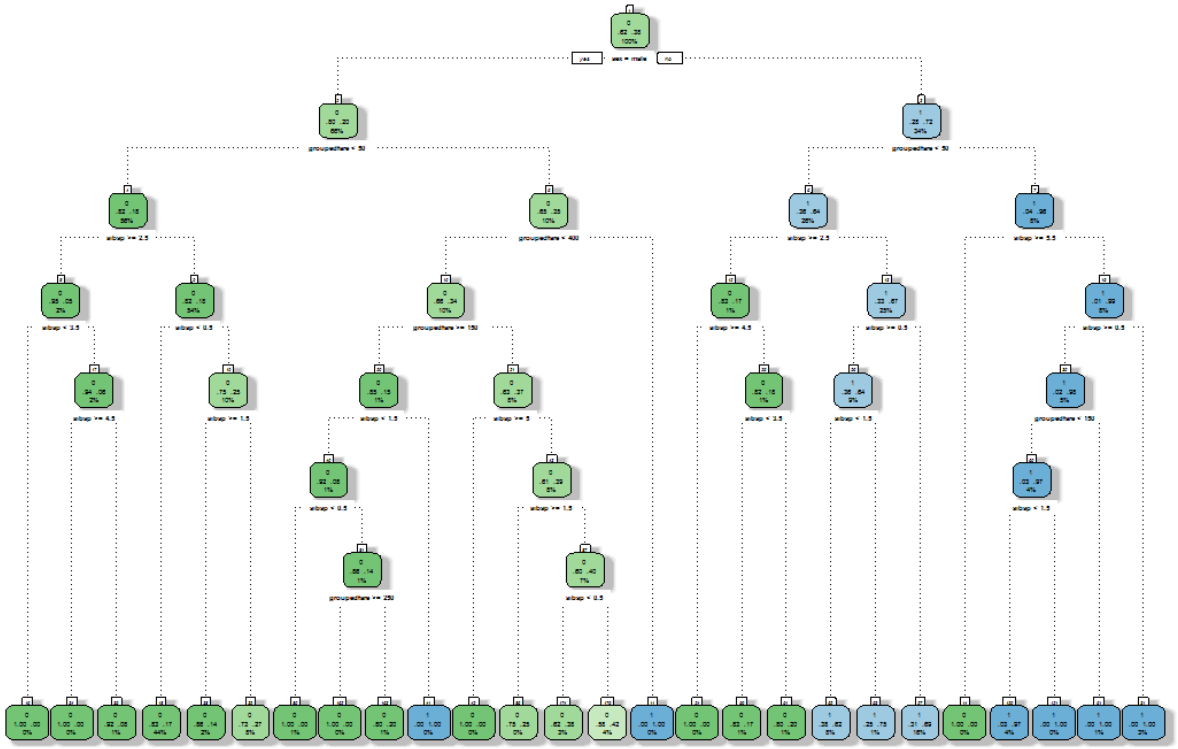
<사진 1>- 의사결정트리



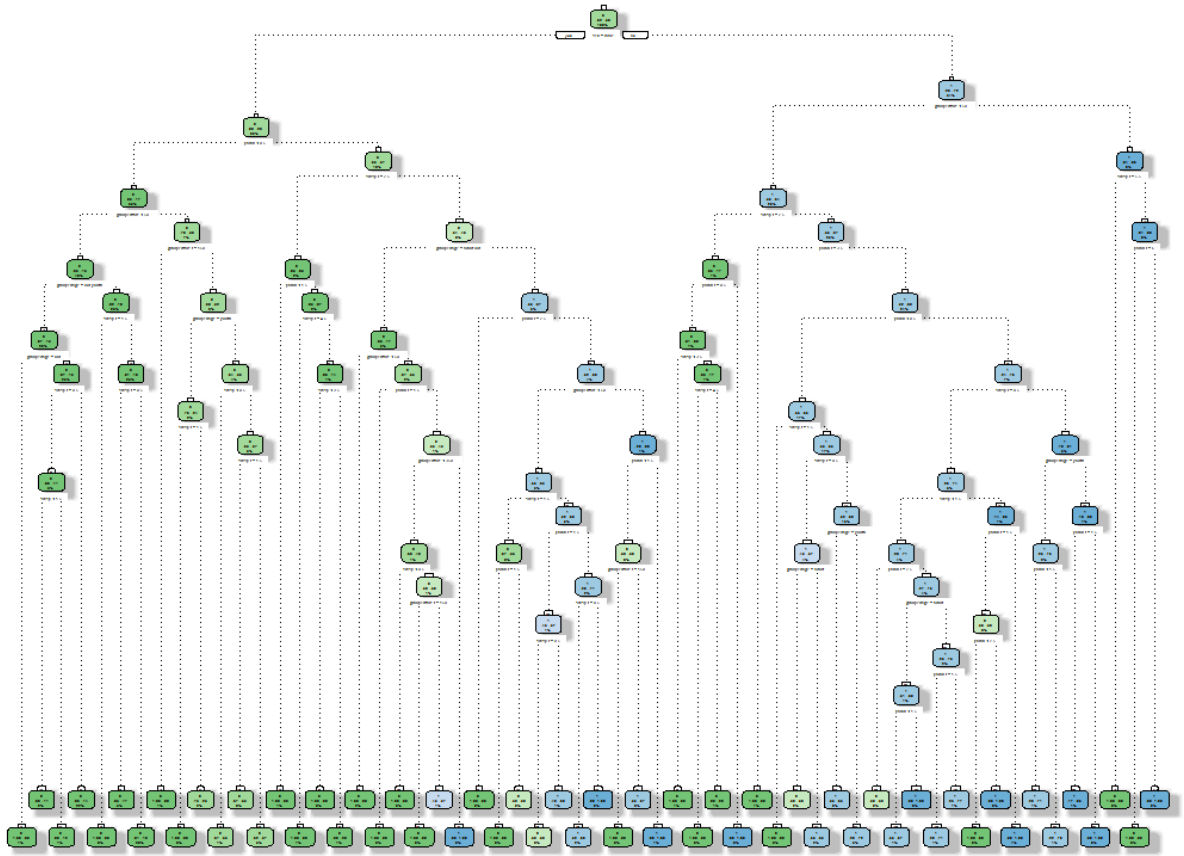
<사진 2>- 트리 모양

과제 H-2

- 과제 H – 모든 변수를 포함하여 모델을 생성하고 정확도를 측정하고 위 문항에서의 결과와 비교하라



<사진 1>- fit(최적변수로 만든 모델)트리



<사진 2>- fit2(모든 변수로 만든 모델)트리

과제 H-3

- 과제 H – 모든 변수를 포함하여 모델을 생성하고 정확도를 측정하고 위 문항에서의 결과와 비교하라

  - 모든변수로 만든 Fit2의 정확도(정분류율)는 0.8036이며 <사진1>참고, 최적변수로만 만든 fit(<사진2>)의 0.8087 보다 정확도가 떨어진다.

  - Titanic 데이터 셋에서 survived를 목표변수로 설정한 경우 최적 예측변수로 만든 트리의 정확도가 더 높으며, 이론대로 목표변수에 영향력을 많이 주는 최적 변수들로 이뤄진 모델의 성능이 더욱 뛰어나다는 것을 알 수 있다.

```
          Reference
Prediction   0   1
         0 208  44
         1  33 107

               Accuracy : 0.8036
                 95% CI : (0.7607, 0.8418)
    No Information Rate : 0.6148
    P-Value [Acc > NIR] : 6.38e-16

                  Kappa : 0.5796

 Mcnemar's Test P-Value : 0.2545

            Sensitivity : 0.8631
            Specificity : 0.7086
         Pos Pred Value : 0.8254
         Neg Pred Value : 0.7643
             Prevalence : 0.6148
         Detection Rate : 0.5306
   Detection Prevalence : 0.6429
      Balanced Accuracy : 0.7858
```

<사진 1>- 모든 변수로만 만든 fit2의 정확도

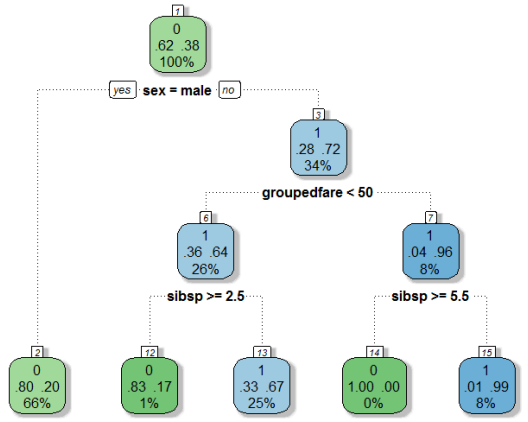
```
          Reference
Prediction   0   1
         0 207  41
         1  34 110

               Accuracy : 0.8087
                 95% CI : (0.7662, 0.8464)
    No Information Rate : 0.6148
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.5925

 Mcnemar's Test P-Value : 0.4884

            Sensitivity : 0.8589
            Specificity : 0.7285
         Pos Pred Value : 0.8347
         Neg Pred Value : 0.7639
             Prevalence : 0.6148
         Detection Rate : 0.5281
   Detection Prevalence : 0.6327
      Balanced Accuracy : 0.7937
```

<사진 2>- 최적변수로만 만든 fit의 정확도

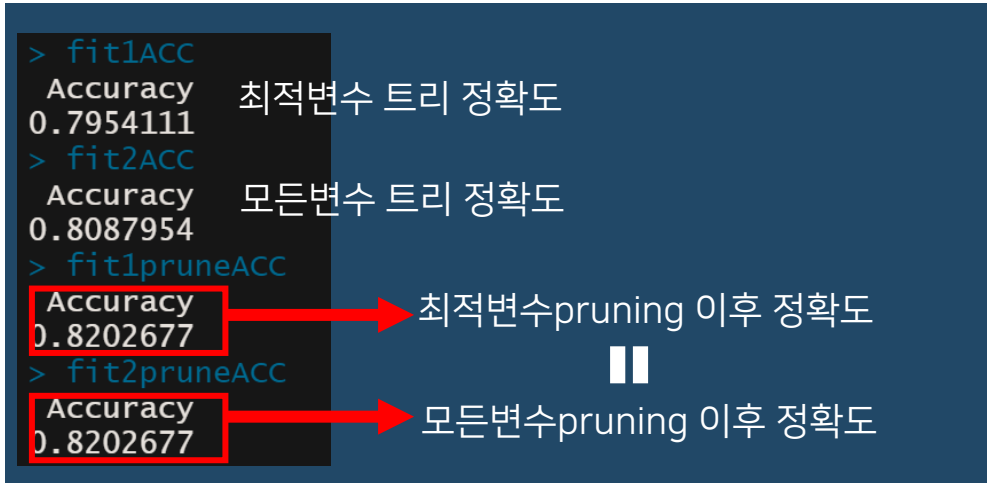
과제 I -1

- 과제 I -1 – Pruning 여부에 따라 tree 의 형태 및 정확도 변화와 정확도를 최대로 하는 최적 파라미터는 무엇인가?

  - 최적의 파라미터는 Pruning 0, 로 설정했을 때에 tree의 형태 및 정확도 변화와 정확도가 최대가 나타났다.
  - 가지치기 이전의 트리와 비교할 때 training set의 크기에 따라 정확도는 변화가 있으나 일정하게 감소하거나, 증가하는 값이 아닌 규칙성이 없이 값이 바뀌었다.
  - 모든 변수를 사용한 트리 역시 정확도를 측정해본 결과 값의 변화는 생겼지만 규칙성을 갖진 않았다. (가지치기 이후의 정확도는 최적트리와 모든변수 트리의 값이 같았다.)



<사진 1>- 가지치기 한 트리의 모양



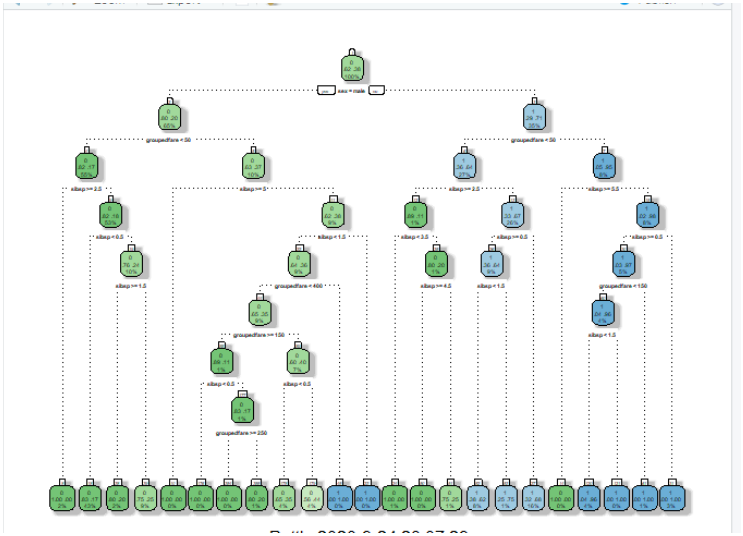
<사진 2>- 트리별 정확성 비교

과제 I -2

- 과제 I – 2 Split criterion 의 종류, 즉 Information과 gini 지수의 차이를 통한 정확도 비교
  - 기본 defaul로 들어가는 gini 지수 파라미터 값의 의사결정나무의 정확도(정분류율)은 0.8087 이었지만 아래 <사진 1> 에서 확인할 수 있듯이 params = “information”값으로 의사결정나무 생성시에 정확도(정분류율)가 0.7951로 상대적으로 떨어진 것을 확인할 수 있다.
  - 이로 써 적절한 split criterio으로는 gini가 적절하다는 것을 알 수 있다.

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 348  77
         1  57 172

               Accuracy : 0.7951
                 95% CI : (0.7621, 0.8254)
    No Information Rate : 0.6193
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.5587

 Mcnemar's Test P-Value : 0.1007

            Sensitivity : 0.8593
            Specificity : 0.6908
         Pos Pred Value : 0.8188
         Neg Pred Value : 0.7511
             Prevalence : 0.6193
         Detection Rate : 0.5321
   Detection Prevalence : 0.6498
      Balanced Accuracy : 0.7750

       'Positive' Class : 0
```

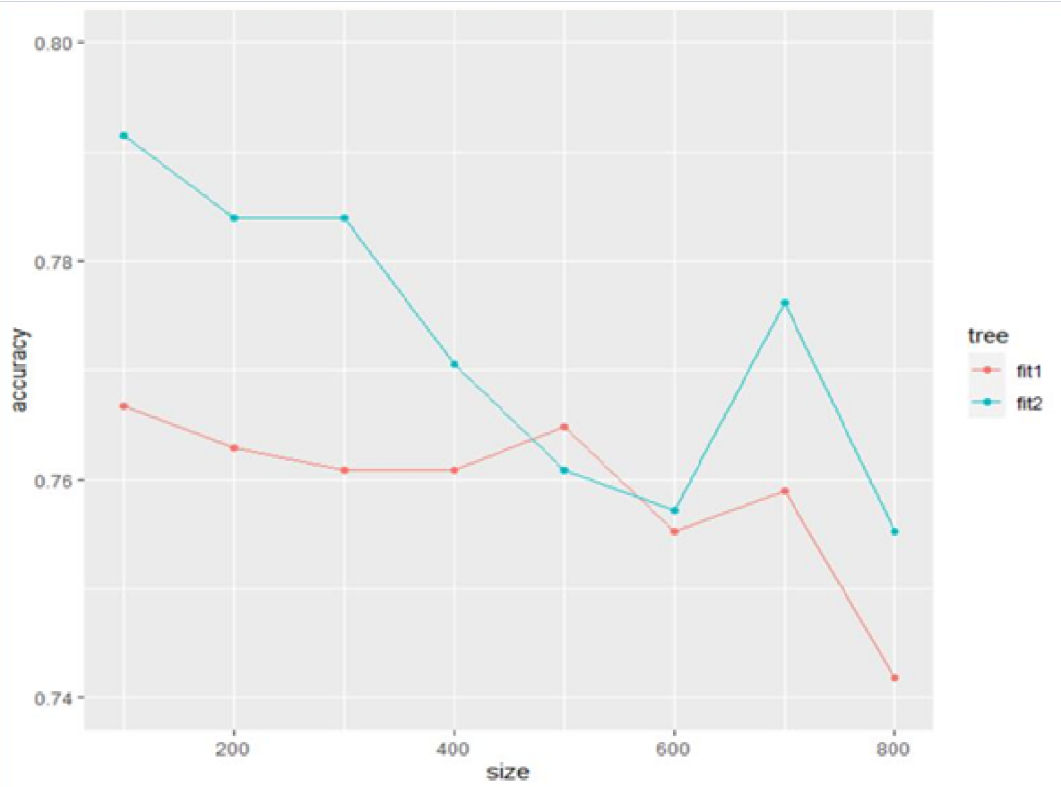
<사진 1>



<사진 1>- 트리 모양

과제 J

- 과제 J – 앞에서 찾아낸 최적 파라미터로 학습데이터 크기에 대한 학습곡선(Learning Curve)을 그려라



<사진 1>-Learning Curve

- training set의 크기100 단위로 키워 나가며, 최적변수tree(fit1)와 모든 변수 tree(fit2)의 정확도를 추출한 후 dataframe화하여 learning _Curve 그래프를 구현.

- **Graph의 모양이 이상적이지 않은 이유**

- Titanic data 자체의 목표변수와 다른 변수들 간의 연관성이 높지 않다.
- Test set의 크기를 고정하였을 시, training set 과 겹치는 경우 발생.

결론 및 고찰

- 결론 및 고찰

- 데이터를 예측 분류시키기 위해 의사결정나무를 사용하는 것은 생각보다 간단하고 예측 정확도도 꽤 나 높다는 것을 체감할 수 있었다.

- 모델의 정확도 향상을 위해서 목표변수가 독립변수들 간에 많은 연관성이 있어야 정확한 통찰을 할 수 있다는 점도 알게 되면서, 앞으로는 이 점에 주의해야할 것 같다.

- Data set의 특이값이 많을 경우 예측이 힘들다는 것을 몸소 느낄 수 있었고 이번 과제를 통해 적절한 전처리 함수를 공부할 수 있었다.

- 마지막으로 데이터 시각화는 데이터를 잘 표현해 주는 강력한 도구라는 것을 느꼈다.