**데이터 사이언스 기말고사 보고서**

데이터 사이언티스트

21000348 서안드레

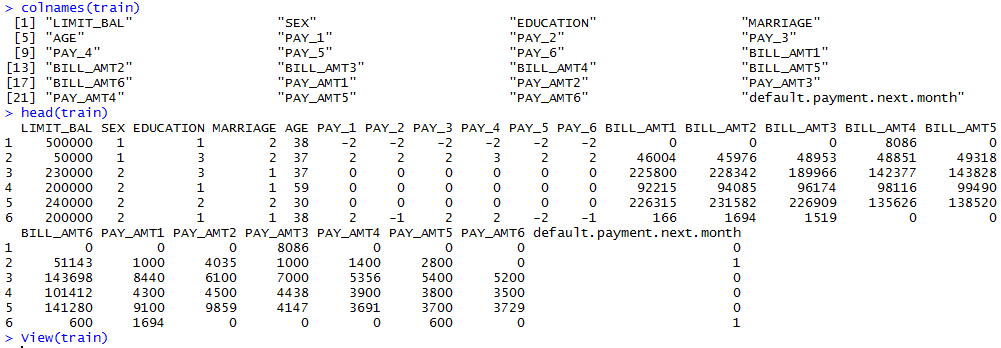
**수행 목표**

train dataset으로부터 채무불이행 여부를 예측하는 모델을 만들어서 test dataset의 고객들에 대해 채무불이행 여부를 예측하는 것.

#주어진 데이터를 읽어와 각각 train과 test라는 이름으로 저장하였다.



#데이터를 탐색하였다.



- 예측할 변수(target variable)는 data set의 마지막 column인 train$default.next.month이며 level로 0과 1의 값을 가지며 각각 채무이행과 채무불이행을 뜻한다.

#의사결정나무(Decision tree)를 사용하여 예측 모델을 만들기로 결정하였다. 그 동안 배웠던 여러 가지 예측 알고리즘들 중 Decision tree를 선택한 핵심적인 이유는 다음과 같다.

- Decision trees는 데이터 ‘분류’에 활용되는 대표적인 모델이다. 이번 분석의 목적 역시 ‘분류’이기 때문에 decision tree 알고리즘이 가장 적합하다고 생각했다. ‘분류’라 함은 고객들에 대한 채무불이행 여부를 TRUE 아니면 FALSE로 예측하여 나눠내는 것이다. 고객들의 나이, 상환금액 등을 예측하는 작업이었으면 적합하지 않았을 것이다.

- Decision trees는 데이터 값들을 형태의 변경 없이 기록된 그대로 분석할 수 있다. 75분이라는 짧은 시간 동안 모델 구축과 평가, 성능 개선까지 이뤄져야 하므로, 최소의 시간, 최소의 노력, 그리고 데이터 분석가 입문자인 본인의 능력을 고려했을 때, decision trees알고리즘이 가장 적합하다고 생각했다. (다른 알고리즘을 사용 못하는 것은 아님)

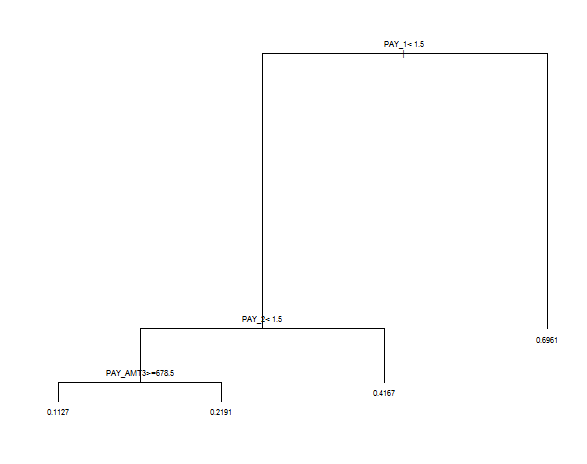
#rpart package를 사용하여 Decision Tree 모델을 만들었다.



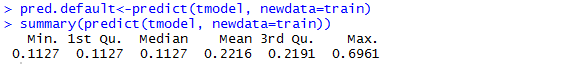
- 예측하기 원하는 변수를 target으로 적었다.

- 사용할 변수(input variable)로 타겟변수[24번째열]를 제외한 모든 변수를 넣었다.

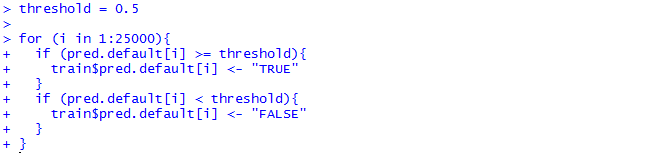
- 아래처럼 그려지는 decision tree를 얻었다.



#위에서 만든 Decision Tree모델로 trainset 고객들에 대한 채무불이행 확률을 예측하였다.



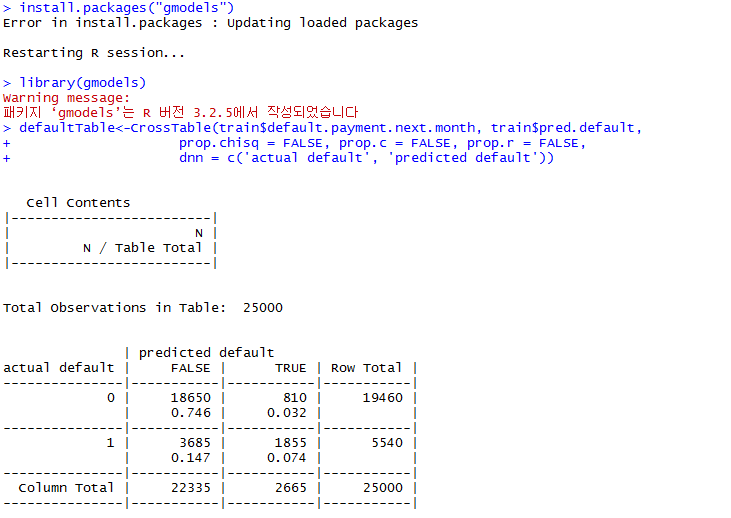
#threshold를 설정하고, 25000명의 고객에 대해 채무불이행 확률이 threshold보다 더 높으면 TRUE(채무불이행)를, 더 낮으면 FALSE(채무이행)를 Input하였다.



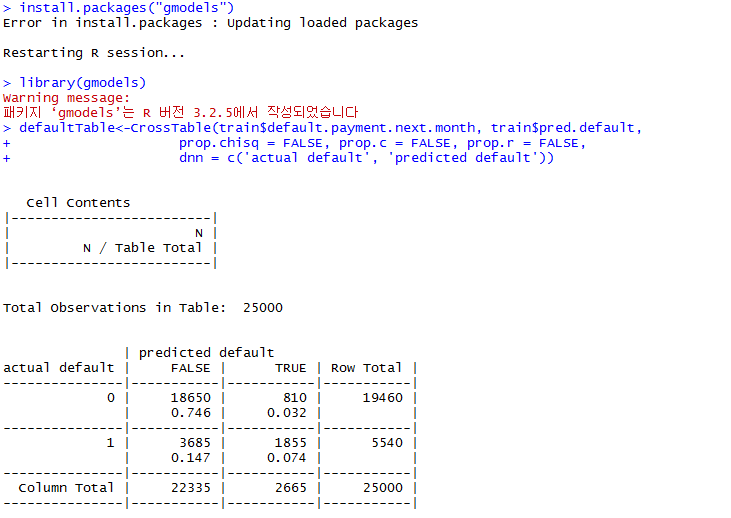
- threshold는 우선 절반 확률로 하여 0.5로 설정하였다.

- T/F에 대한 Input은 train dataset 내에 새로 만든 train$pred.default column에 추가하였다.

#F1 Score로 성능을 평가하기 위하여 Crosstable함수로 defaultTable이라는 테이블을 만들었다.



- F1 Score는 Precision과 Recall 값을 기반으로 하며, 만들어진 테이블을 활용하여 값을 구할 수 있다.

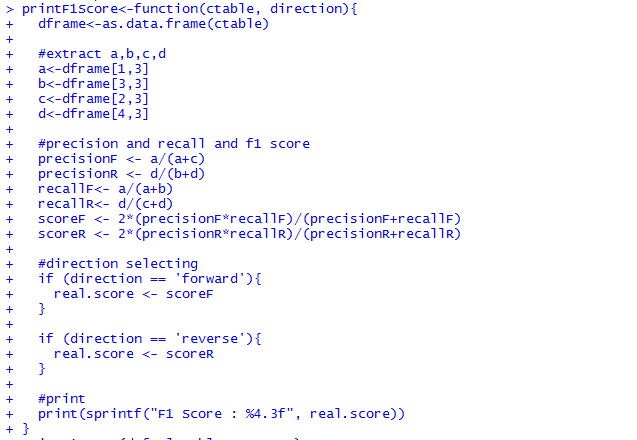


#위에서 만든 테이블을 기반으로 F1 Score 성능을 계산하여 print 하였다.



- 초기 성능 0.452가 나왔다. 성능 향상이 필요하다.

# F1 Score를 계산하여 출력해주는 함수



- Crosstable로 만든 테이블을 data.frame으로 바꾸고, 필요한 데이터를 추출하여 a,b,c,d로 설정한다. (a = 0으로예측, 실제0 / b = 1로예측, 실제0 / c = 0으로예측, 실제1 / d = 1로예측, 실제1)

- 추출한 각 a,b,c,d를 기반으로, Precision, Recall, F1 Score를 계산한다.

- direction(방향)은 CrossTable에서 a,b,c,d를 추출하는 2가지 방향에 대한 것이다. 방향이 다르면 의미가 반대가 되 때문에 인위적으로 설정하였고, 본 테이블에서는 역방향으로 계산했다.

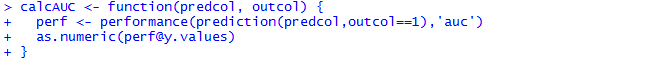
- 계산한 F1 Score를 프린트한다.

#AUC로 성능을 평가하여 출력하였다.



- 0.7366567 이 나왔다. 성능 개선이 필요하다.

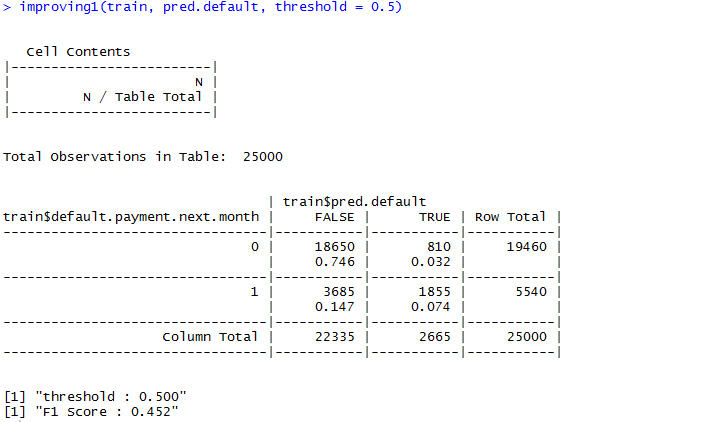
- AUC 측정 함수



#모델을 완성하기 전에, 여러 가지 방법으로 모델의 성능을 올릴 것을 계획 및 수행하였다.

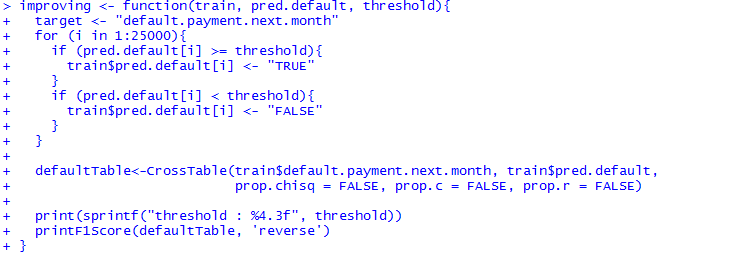
#첫 번째 방법인 “threshold 값 조정”을 사용하기 위해, 전용 함수를 만들어 threshold가 바뀔 때마다 성능이 어떻게 바뀌는지 파악하여, 가장 좋은 성능이 나오는 threshold를 찾았다.

- threshold = 0.5일 때



F1 Score = 0.452

- 사용한 함수



- threshold = 0.6일 때



F1 Score = 0.452

- threshold = 0.4일 때



F1 Score = 0.523

- threshold = 0.3일 때



F1 Score = 0.523

- threshold = 0.2일 때

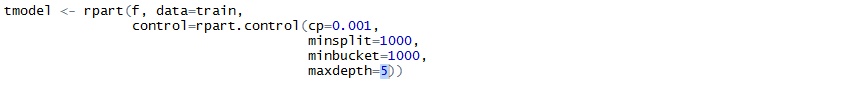


F1 Score = 0.493

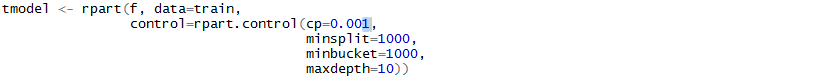
threshold 값을 조정해보면서 얻은 성능이 제일 높은 threshold는 0.3과 0.4였으며, 그 때의 F1 Score는 0.523였다. 이 모델의 threshold에 0.3을 사용하기로 결정하였다.

#두 번째 방법으로 rpart control 인자를 활용하여 트리의 복잡도를 제한하면서 성능 향상을 시도했다.

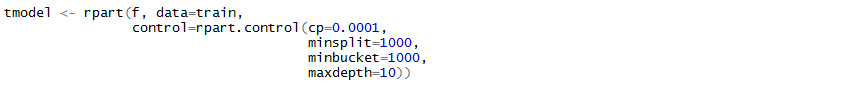
- 디폴트 값



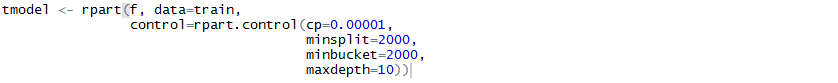
- 인자 조작1



- 인자 조작2



- 인자 조작3



- threshold를 0.3으로 두었을 때, 세 번의 인자 조작으로 성능을 개선하려고 했지만 F1 Score는 계속 0.523으로 동일하였다. 더 조작해보고 싶었으나 시간의 제약으로 그러지 못하였다.



- 다른 성능 향상 방법으로, 변수를 골라내고 선별한 변수를 사용하여 모델을 구성하는 방법을 생각했었다. 하지만, 역시 시간의 제약 때문에 시도하지 못하였고, 현재의 모델로부터 이미 목표치로 제시된 성능을 뛰어 넘는 성능을 얻어내었으므로 시간 내에 만들 수 있는 최적의 모델이라고 판단하였고, 따라서 testset 고객에 대하여 이 모델을 사용하기로 결정하였다. 마지막으로 최종 모델에 대해 정리한 후, 다음 단계로 넘어갔다.

===============================================

# 최종 모델 명세

- 기법 : Decision Tree with Rpart Package

- 사용 변수 : dataset의 모든 변수(Target Variable을 제외)

- 채무불이행 확률에 대한 threshold : 0.3

- training set에서의 성능 : F1 Score = 0.523 / AUC = 0.7366567

===============================================

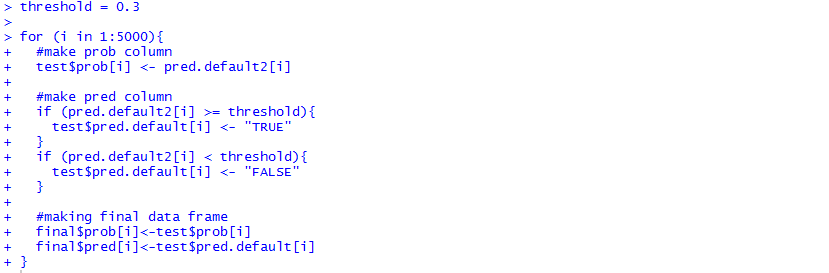
#최종 모델로 testset 고객들의 채무불이행 여부에 대해 예측하기 전에, 제출 파일(.csv file)의 기반이 되는 final 데이터 프레임(2, 5000)을 만들어 두었다.



#testset 고객들의 채무불이행 확률을 예측하였다.



#5000명의 고객에 대해 채무불이행 확률이 threshold보다 더 높으면 TRUE를, 더 낮으면 FALSE를 Input하였다. 마지막 두 줄에서는 prob(확률)와 pred(T/F)를 final 데이터 프레임에 Input 하였다.



#최종적으로 만들어진 final 데이터 프레임을 CSV 파일로 write하였고, 파일이 잘 만들어졌는지 확인하는 것으로 마무리 하였다.

