다변량분석 Assignment 6

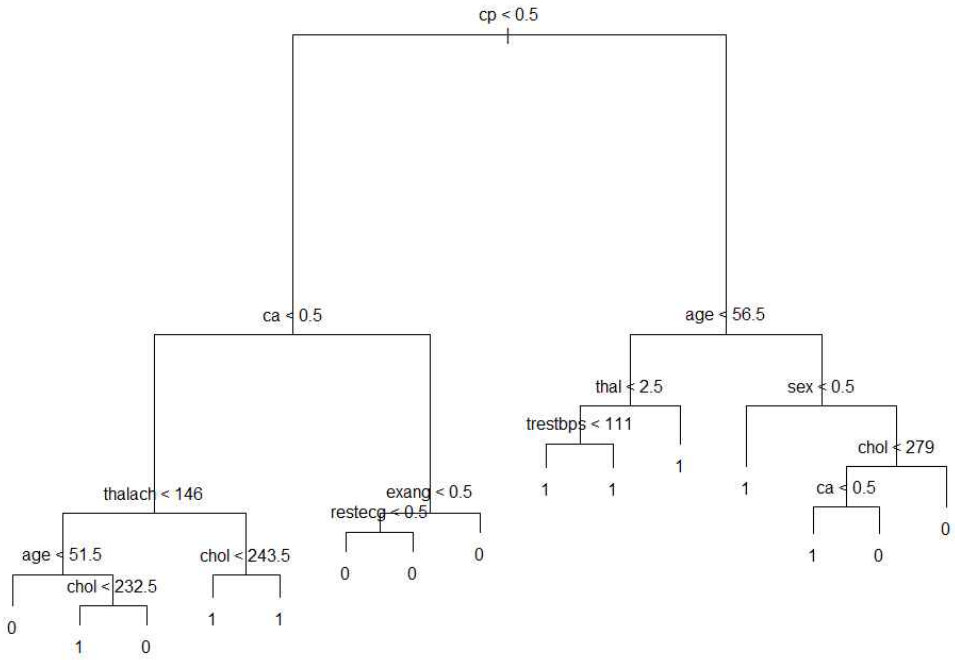
2014170824 황태민

[Q1] 'tree' package

전체 데이터 셋을 임의로 200개의 Training dataset과 103명의 validation dataset으로 구분한 뒤 'tree' package를 사용하여 학습을 진행한 결과는 다음과 같다.

```
Classification tree:
tree(formula = heartYN ~ ., data = CART.trn)
Variables actually used in tree construction:
 [1] "cp"       "ca"       "thalach"  "age"      "chol"     "exang"    "restecg"  "thal"     "trestbps" "sex"
Number of terminal nodes:  15
Residual mean deviance:  0.4438 = 82.11 / 185
Misclassification error rate: 0.09 = 18 / 200
```

학습 결과에 따르면 13개의 입력 변수중 10개의 입력변수만이 모델에 사용되었으며 terminal nodes의 개수는 15개이다. 위 모델에 따른 training dataset의 misclassification error는 0.09로 굉장히 낮은 값을 띈다.



위는 학습한 모델을 plot 한 것이다. 제일 처음 분류 기준이 되는 것은 'cp'이라는 입력 변수이다. pruning을 진행하지 않은 모델이라 꽤 많은 leaf node를 가진다.

학습한 모델을 통해 validation dataset에 대한 예측을 진행한 Confusion matrix와 Classification Performance는 다음과 같다.

	CART.prey	
	0	1
0	30	18
1	9	46

먼저 confustion matrix의 결과를 보면 실제 심장병 환자(=1) 55명 중 46명을 심장병 환자일 것이라고 예측하였다. 그리고, 실제 심장병 환자가 아닌 사람(=0) 48명 중 30명을 심장병 환자가 아닐 것이라고 예측하였다.

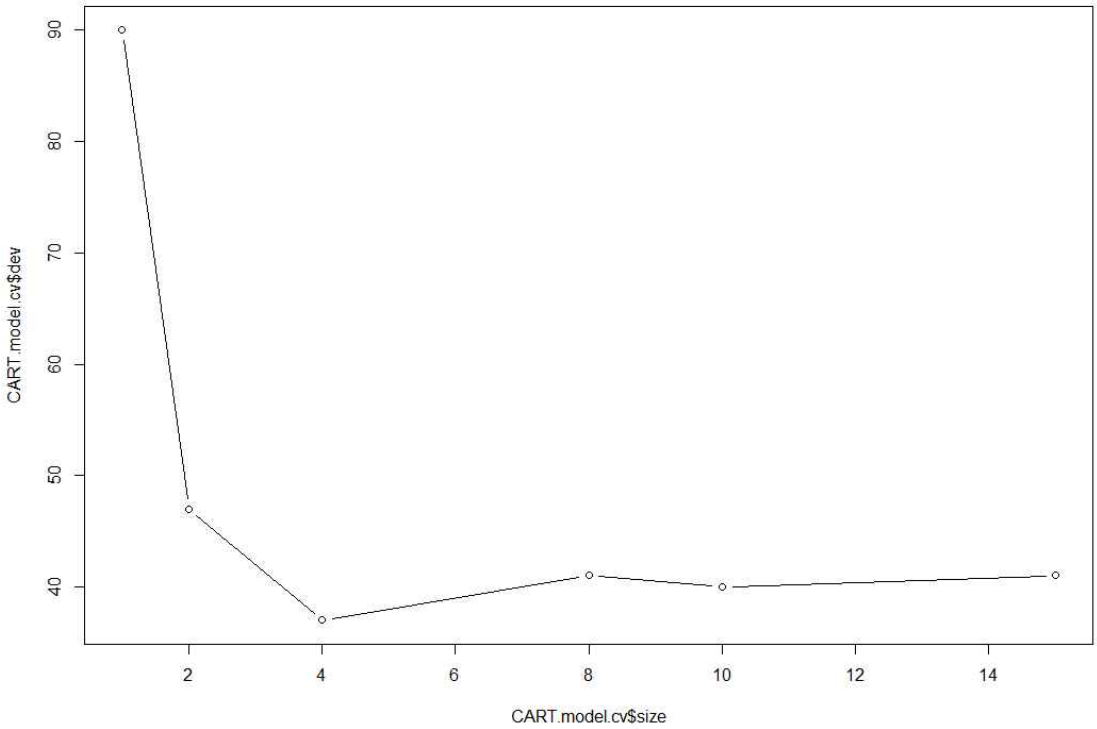
	TPR	Precision	TNR	Accuracy	BCR	F1-Measure
Tree	0.8363636	0.7187500	0.6250000	0.7378641	0.7229988	0.7731092

다음은 Classification Performance이다. 먼저 단순 정확도인 Accuracy 값은 약 0.73로 높은 정확도를 보인다. 그리고 BCR과 F1-Measure 값도 약 0.72, 0.77로 Accuracy와 비슷한 값을 보인다.

다음은 pruning을 통해 위의 모델보다 예측에 있어서 나은 성능을 보이는지를 확인해보자.

[Q2] 'tree' package pruning

먼저 'tree' 패키지의 cv.tree 함수를 이용해 최적의 split 횟수를 찾아내었다.



위의 plot을 보면 size가 4일 때 가장 낮은 deviance값을 보이는 것으로 보인다.

```
$size
[1] 15 10  8  4  2  1

$dev
[1] 41 40 41 37 47 90

$k
[1] -Inf  0.0  1.5  2.5  7.0 45.0

$method
[1] "misclass"

attr(,"class")
[1] "prune"         "tree.sequence"
```

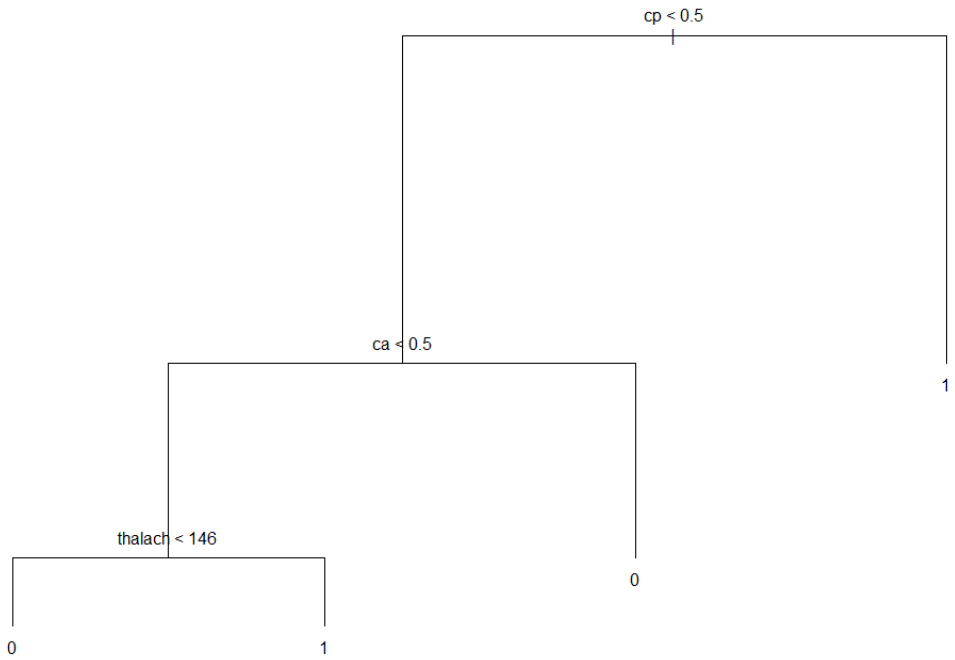
cv.tree의 결과는 위와 같다. size가 4일 때 deviance가 37로 제일 작다. 이에 따라 최적의 leaf nodes의 개수는 4개이다.

leaf nodes의 개수를 4개로 pruning한 decision tree의 결과는 다음과 같다.

```
Classification tree:
snip.tree(tree = CART.model, nodes = c(5L, 9L, 8L, 3L))
Variables actually used in tree construction:
[1] "cp"      "ca"      "thalach"
Number of terminal nodes:  4
Residual mean deviance:  0.8203 = 160.8 / 196
Misclassification error rate: 0.155 = 31 / 200
```

모델에 사용된 변수는 cp, ca thalach로 세 개의 입력 변수가 사용되었으며 terminal nodes의 개수는 설정한 대로 4개이다. 또한 misclassification error rate는 원래의 모델보다 조금 높아진 0.155이다.

다음은 위 모델의 plot이다.



원래의 모델과 마찬가지로 'cp'변수가 제일 처음 분류 기준으로 제시되었다.

학습한 모델을 통해 validation dataset에 대한 예측을 진행한 Confusion matrix와 Classification Performance는 다음과 같다.

```
CART.prey.pruned
   0  1
0 27 21
1  7 48
```

먼저 confustion matrix의 결과를 보면 실제 심장병 환자(=1) 55명 중 48명을 심장병 환자일 것이라고 예측하였다. 그리고, 실제 심장병 환자가 아닌 사람(=0) 48명 중 27명을 심장병 환자가 아닐 것이라고 예측하였다. 원래의 모델보다 심장병 환자 예측에서는 높은 예측력을

보이지만, 심장병 환자가 아닌 사람들을 잘 예측하지 못하는 모습을 보인다.

	TPR	Precision	TNR	Accuracy	BCR	F1-Measure
Tree	0.8363636	0.7187500	0.6250000	0.7378641	0.7229988	0.7731092
Tree_pruned	0.8727273	0.6956522	0.5625000	0.7281553	0.7006490	0.7741935

다음은 Classification Performance이다. 위와 같은 모습이 performance measure에서도 드러난다. TPR은 원래의 모델보다 높은 값을 보이고, Accuracy는 약 0.72로 비슷한 값을 보이지만 TNR에서 약 0.56이라는 다소 낮은 수치를 보인다. BCR은 0.02 하락하였고, F1-measure는 원래의 모델과 비슷하다.

[Q3] ‘rpart’ package

[Q3-1]

‘rpart’와 ‘rpart.plot’을 동시에 사용하였다.

[Q3-2] Options

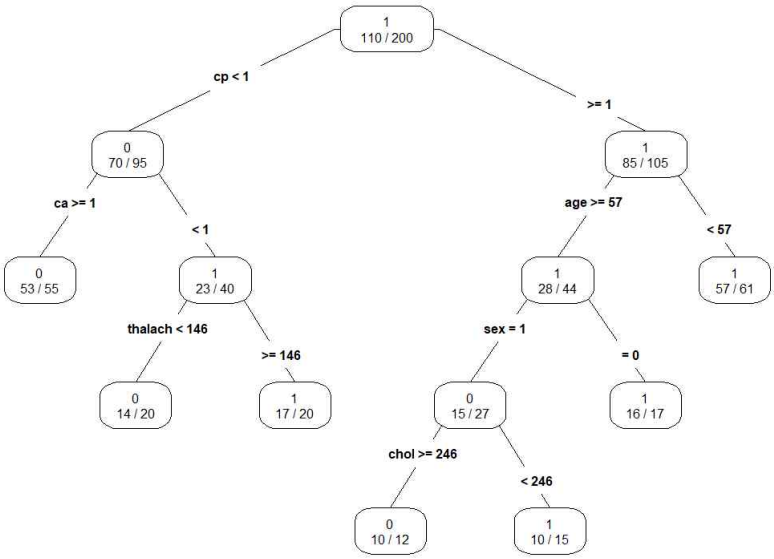
rpart 패키지에서 사용할 수 있는 옵션과 의미는 다음과 같다.

minsplit	the minimum number of observations that must exist in a node in order for a split to be attempted.
minbucket	the minimum number of observations in any terminal <leaf> node. If only one of minbucket or minsplit is specified, the code either sets minsplit to minbucket*3 or minbucket to minsplit/3, as appropriate.
cp	complexity parameter. Any split that does not decrease the overall lack of fit by a factor of cp is not attempted. For instance, with anova splitting, this means that the overall R-squared must increase by cp at each step. The main role of this parameter is to save computing time by pruning off splits that are obviously not worthwhile. Essentially,the user informs the program that any split which does not improve the fit by cp will likely be pruned off by cross-validation, and that hence the program need not pursue it.
maxcompete	the number of competitor splits retained in the output. It is useful to know not just which split was chosen, but which variable came in second, third, etc.
maxsurrogate	the number of surrogate splits retained in the output. If this is set to zero the compute time will be reduced, since approximately half of the computational time (other than setup) is used in the search for surrogate splits.
usesurrogate	how to use surrogates in the splitting process. 0 means display only; an observation with a missing value for the primary split rule is not sent further down the tree. 1 means use surrogates, in order, to split subjects missing the primary variable; if all surrogates are missing the observation is not split. For value 2 ,if all surrogates are missing, then send the observation in the majority direction. A value of 0 corresponds to the action of tree, and 2 to the recommendations of Breiman *et.al* (1984).
xval	number of cross-validations.
surrogatestyle	controls the selection of a best surrogate. If set to 0 (default) the program uses the total number of correct classification for a potential surrogate variable, if set to 1 it uses the percent correct, calculated over the non-missing values of the surrogate. The first option more severely penalizes covariates with a large number of missing values.
maxdepth	Set the maximum depth of any node of the final tree, with the root node counted as depth 0. Values greater than 30 rpart will give nonsense results on 32-bit machines.

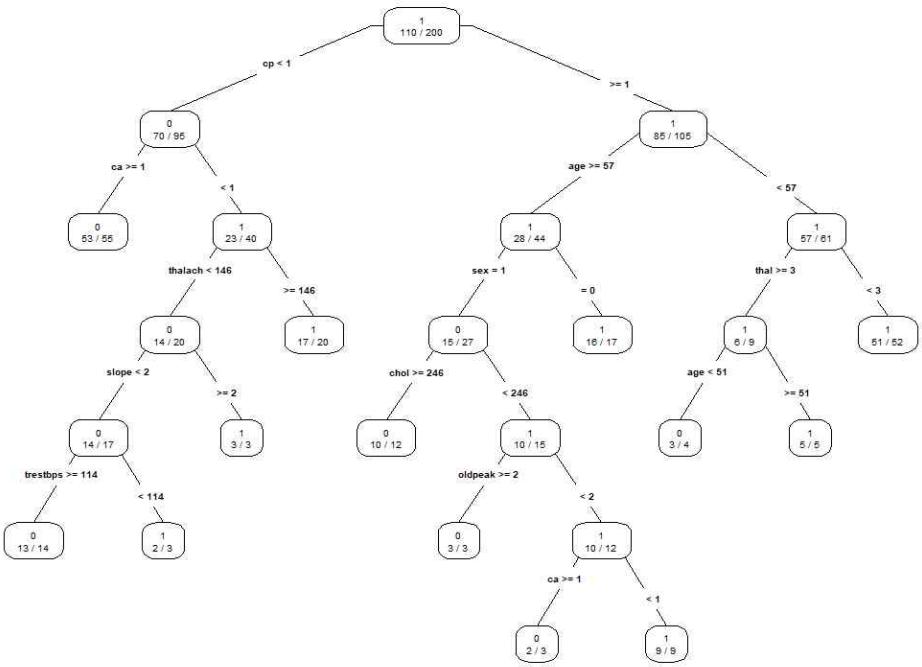
먼저 ‘surrogate’에 관한 옵션들은 missing value에 해당하는 것이므로 다루지 않도록 한다. 또한 cross validation의 횟수인 ‘xval’은 다른 모형들과 비교를 위해 10으로 설정한다. 그리고 complexity parameter 옵션을 제공하는 ‘cp’는 높은 수치로 설정하게 될 경우 제대로 학습이 이루어지지 않기 때문에 0.01로 설정하였다.

이제 minsplit, minbuckte, maxdepth를 변화시켜가며 이 옵션들이 tree를 어떻게 변화시키는지 알아보자. 또한 변화를 좀 더 알아보기 쉽게 하기 위해 [Q3-4]와도 맞닿아있는 다른 방법의 plot을 사용하여 변화를 살펴볼 것이다. 여기서는 rpart.plot의 prp 함수를 사용하였다.

[Q3-3]
먼저 minsplit은 split을 시도하기 위해 필요한 최소의 관측치 수를 설정하는 것이다. 비교를 위해 xval=10, cp=0.01만 설정한 tree의 plot은 다음과 같다.

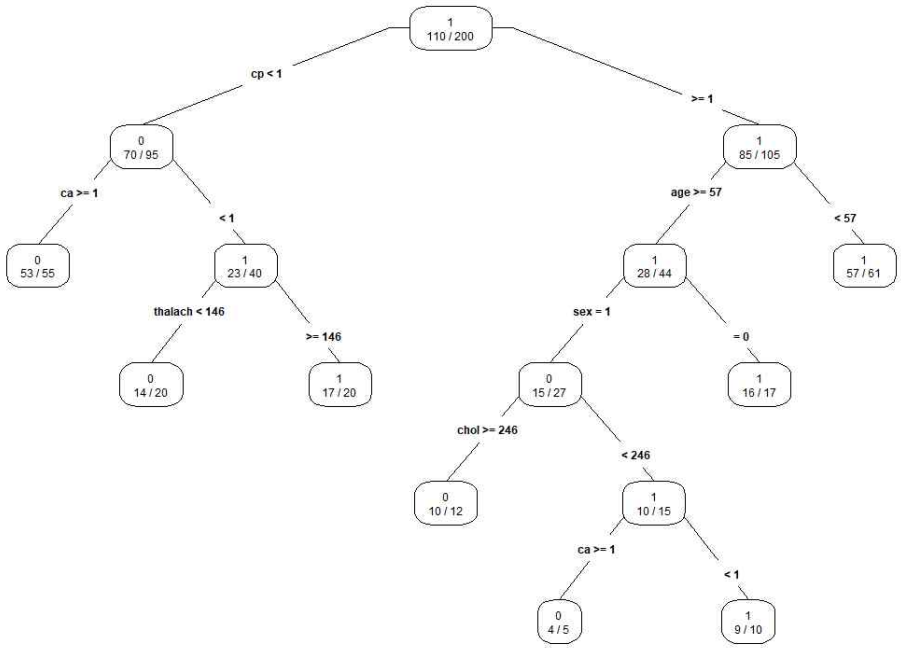


terminal nodes의 개수는 7개이고 분류도 꽤 정확하게 된 것을 볼 수 있다.
이제 minsplit을 5로 설정해보자.

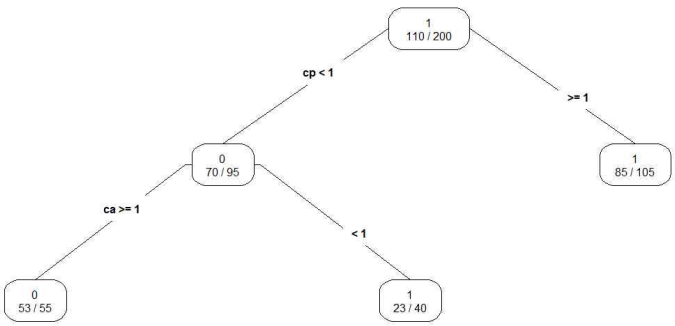


minsplit을 5로 설정한 결과 13개의 leaf node 수를 보이며 훨씬 더 많은 split이 생긴 것을 볼 수 있다. 좀 더 세분화된 decision tree를 얻고 싶을 때 minsplit의 값을 작게 설정하면 될 것이다.

다음은 minbucket을 변화시켜보자. 위의 설명에서도 볼 수 있듯이 minsplit과 minbucket은 minsplit=minbucket*3일 때 적절하다.
다음은 minbucket을 5로 설정한 결과이다.

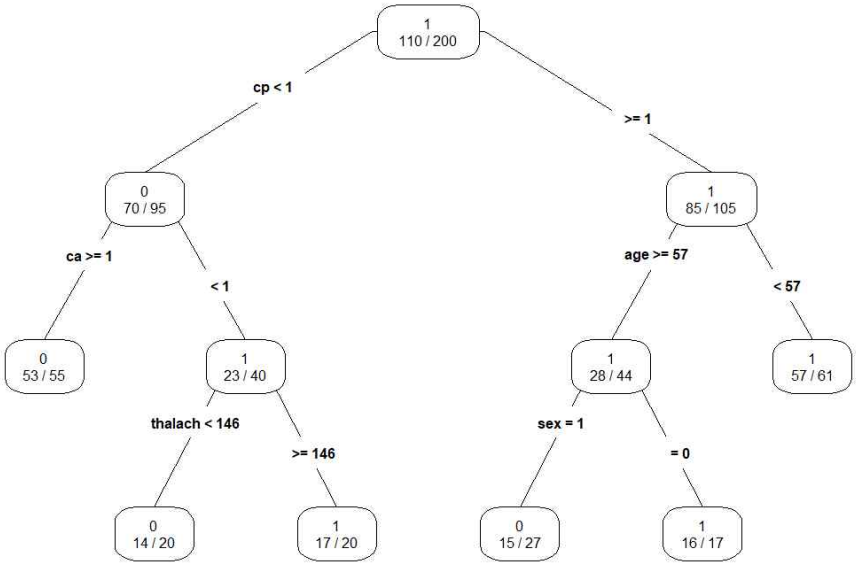


terminal node의 개수는 8개이고 분류 또한 꽤 정확하다.
다음은 minbucket을 15로 설정해보았다.

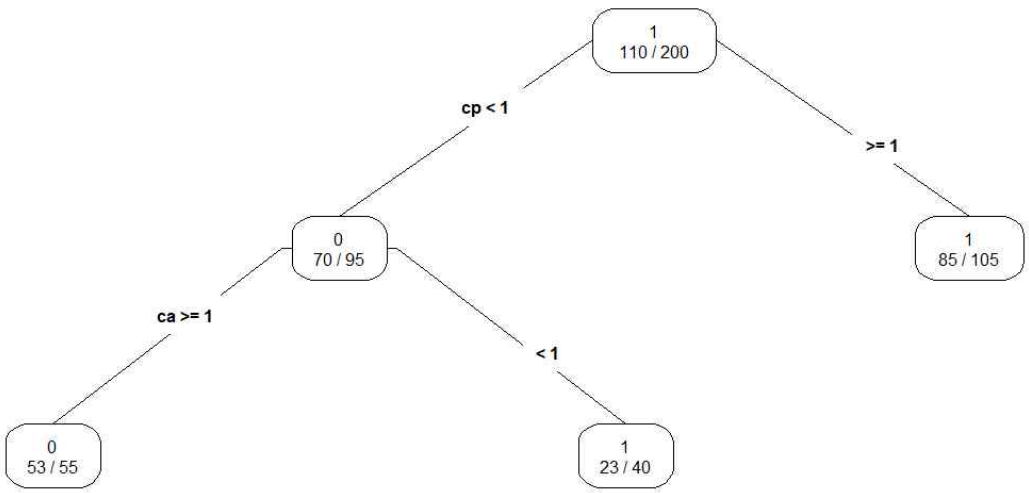


minbucket을 15로 설정하니 leaf nodes의 개수가 3개인 아주 단순한 decision tree가 만들어졌다. 위의 두 결과를 통해 minbucket은 클수록 단순한 decision tree가 만들어진다는 것을 알 수 있다.

다음은 maxdepth를 변화시켜보자. maxdepth는 말 그대로 최대 depth를 설정해주는 옵션이다. maxdepth=3으로 설정한 결과는 다음과 같다.



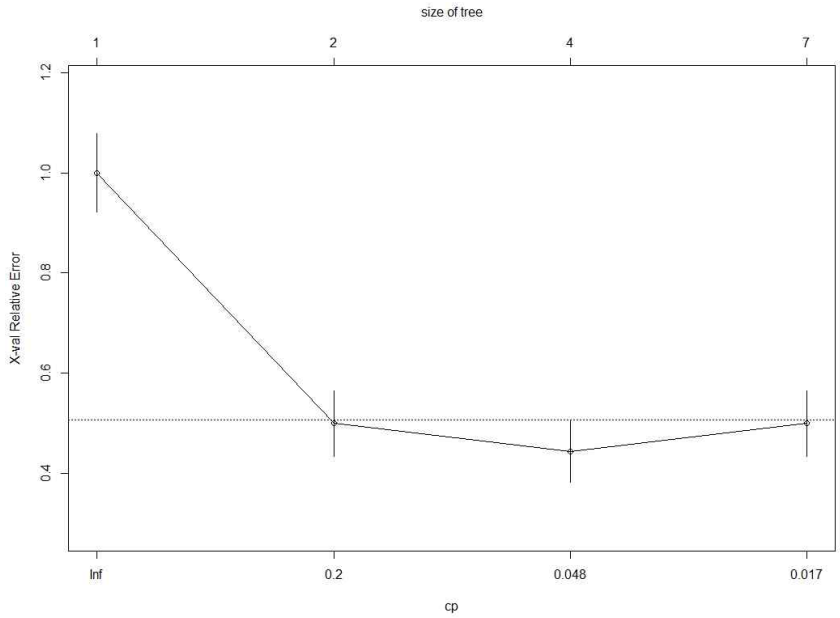
plot에서도 알 수 있듯이 depth는 3이다.
다음은 depth를 2로 설정한 결과이다.



이번에도 depth가 2인 decision tree를 보여준다.

[Q3-3] Best model 'rpart'

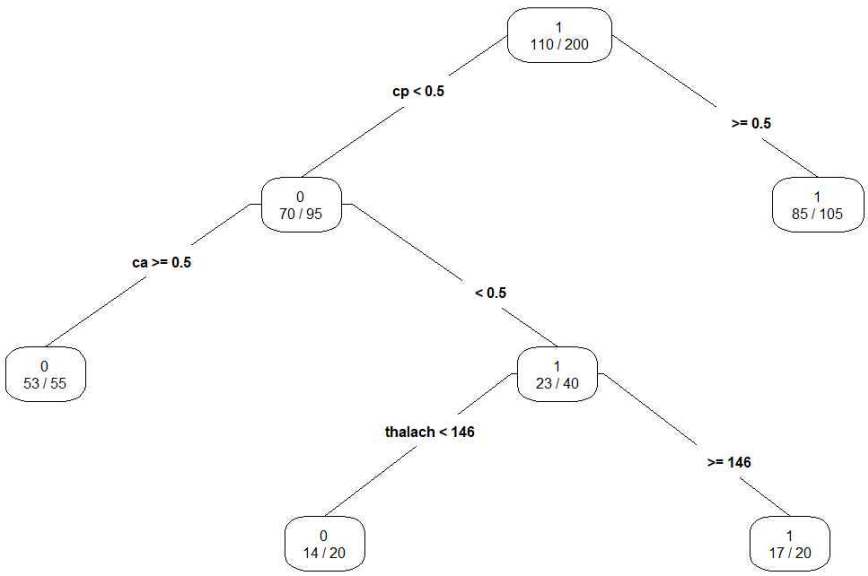
rpart의 best model을 찾기 위해 cross validation을 진행하고 가장 작은 error값을 갖는 최적의 size를 찾아내기로 하였다. 이에 따라 rpart 패키지의 printcp와 plotcp 함수의 결과는 다음과 같다.



	CP	nsplit	rel error	xerror	xstd
1	0.500000	0	1.00000	1.00000	0.078174
2	0.077778	1	0.50000	0.50000	0.065617
3	0.029630	3	0.34444	0.44444	0.062854
4	0.010000	6	0.25556	0.50000	0.065617

결과는 size가 4일 때 가장 작은 값을 갖는 것으로 보인다. 이에 따라 leaf node의 개수를 4개로 pruning을 진행하였다.

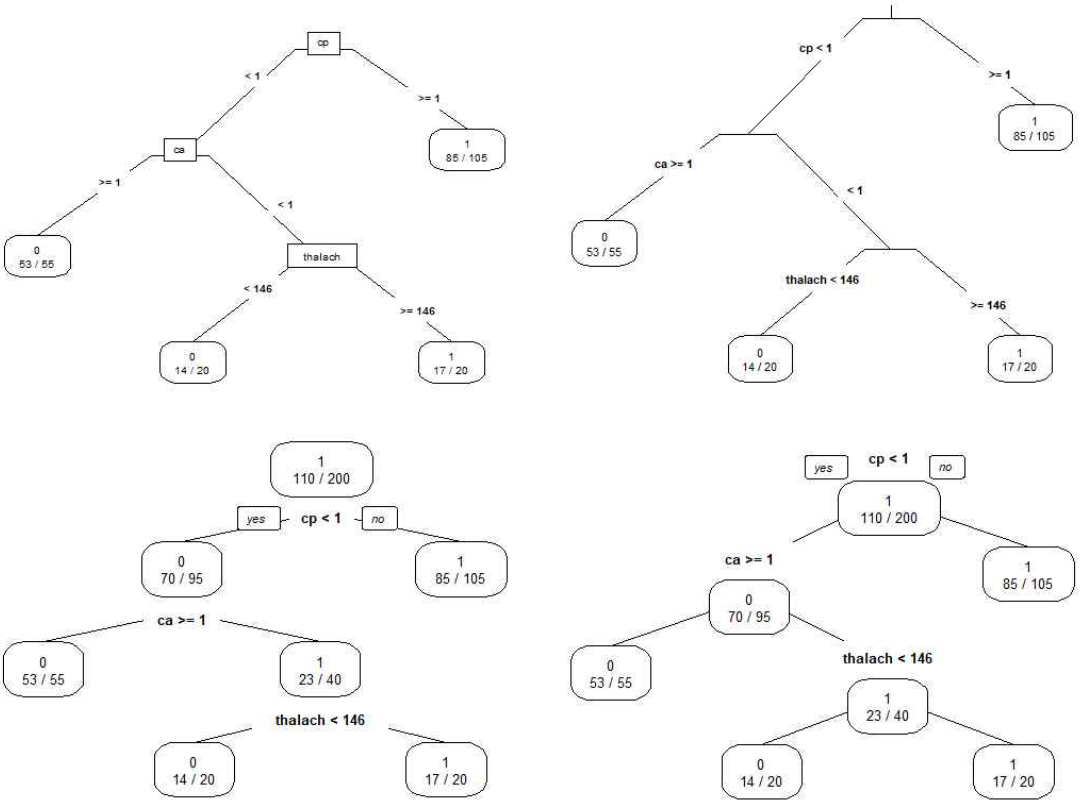
pruning된 decision tree의 모습은 다음과 같다.



leaf node의 개수는 4개이고 분류 또한 꽤 정확해 보인다.

[Q3-4] Various Plotting

rpart.plot 패키지를 이용하여 기존의 plot과는 다른 decision tree 도시가 가능한데 위의 pruned model을 보면 최종 노드의 결과 구성을 쉽게 알아볼 수 있고, arc에 기준이 명확히 표시되어 있는 점이 장점이다. 또한 다른 옵션들을 통해 조금 다른 모습의 plotting도 가능하다. 총 5개의 옵션을 제공하는데 위에 사용한 옵션은 4번 옵션이므로 나머지 4개의 옵션들의 모습은 다음과 같다.



나머지 4개의 옵션들도 원래의 plot 함수보다는 직관적이고 알아보기 쉽다.

[Q3-5] 'rpart' best model Prediction

위에서 얻은 pruned model을 통해 validation을 진행해보도록 하자.

학습한 모델을 통해 validation dataset에 대한 예측을 진행한 Confusion matrix와 Classification Performance는 다음과 같다.

	rpart.prey.pruned	
	0	1
0	27	21
1	7	48

먼저 confustion matrix의 결과를 보면 실제 심장병 환자(=1) 55명 중 48명을 심장병 환자일 것이라고 예측하였다. 그리고, 실제 심장병 환자가 아닌 사람(=0) 48명 중 27명을 심장병 환자가 아닐 것이라고 예측하였다. 이는 'tree' package를 사용하여 pruned model의 결과와 같으며 decision tree도 같다.

	TPR	Precision	TNR	Accuracy	BCR	F1-Measure
Tree	0.8363636	0.7187500	0.6250	0.7378641	0.7229988	0.7731092
Tree_pruned	0.8727273	0.6956522	0.5625	0.7281553	0.7006490	0.7741935
rpart	0.8727273	0.6956522	0.5625	0.7281553	0.7006490	0.7741935

다음은 Classification Performance 값이다. tree 패키지의 pruned model과 정확히 같은 값을 가지는 것을 볼 수 있다.

[Q3] ‘party’ package

[Q3-1]

‘party’ package를 사용하였다.

[Q3-2] Options

party 패키지에서 사용할 수 있는 옵션과 의미는 다음과 같다.

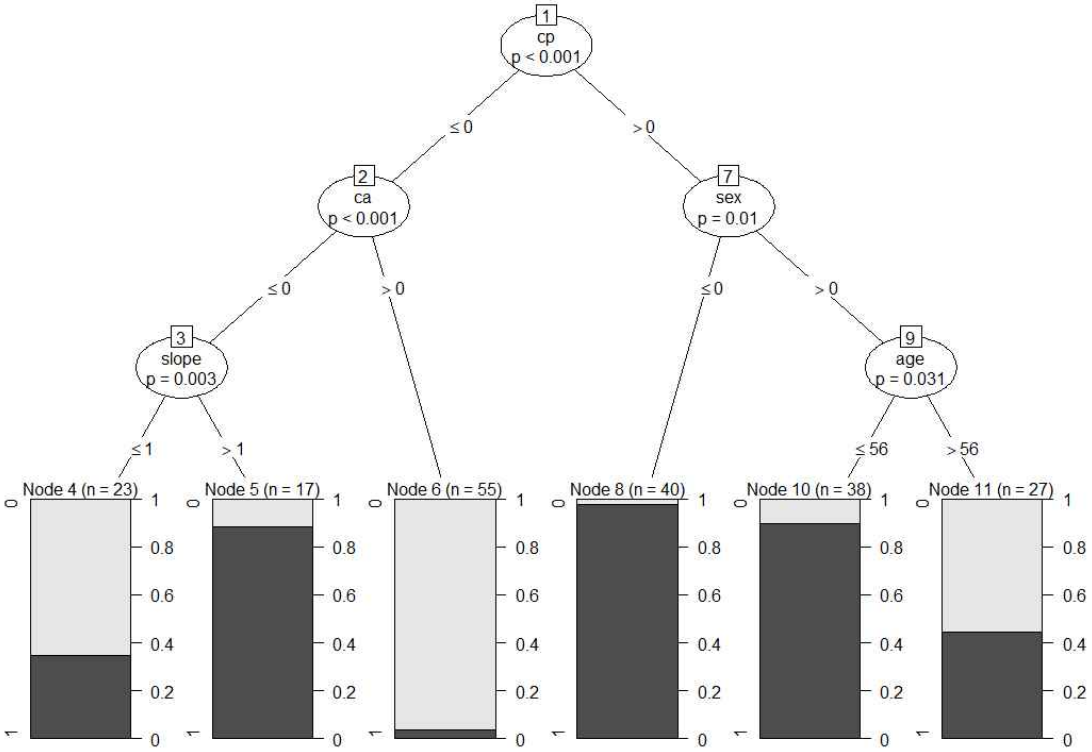
`teststat`	a character specifying the type of the test statistic to be applied.
`testtype`	a character specifying how to compute the distribution of the test statistic.
`mincriterion`	the value of the test statistic (for `testtype == "Teststatistic"`), or 1 - p-value (for other values of `testtype`) that must be exceeded in order to implement a split.
`minsplit`	the minimum sum of weights in a node in order to be considered for splitting.
`minbucket`	the minimum sum of weights in a terminal node.
`stump`	a logical determining whether a stump (a tree with three nodes only) is to be computed.
`nresample`	number of Monte-Carlo replications to use when the distribution of the test statistic is simulated.
`maxsurrogate`	number of surrogate splits to evaluate. Note the currently only surrogate splits in ordered covariables are implemented.
`mtry`	number of input variables randomly sampled as candidates at each node for random forest like algorithms. The default `mtry = 0` means that no random selection takes place.
`savesplitstats`	a logical determining if the process of standardized two-sample statistics for split point estimate is saved for each primary split.
`maxdepth`	maximum depth of the tree. The default `maxdepth = 0` means that no restrictions are applied to tree sizes.
`remove_weights`	a logical determining if weights attached to nodes shall be removed after fitting the tree.

위 옵션중 maxsurrogate는 결측치와 관련된 것이므로 제외하고 minsplit, minbucket, maxdepth는 rpart에서의 옵션과 동일하다.

[Q3-3] Best model 'party'

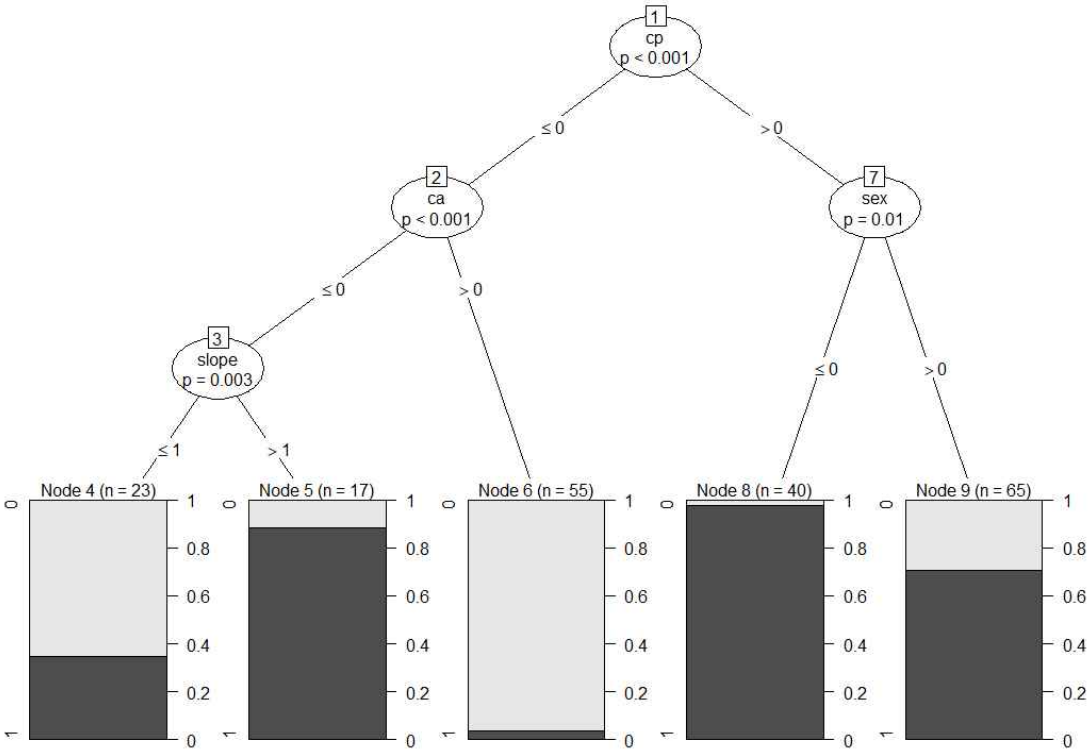
party 패키지에서 제공하는 ctree 함수는 Unbiased recursive partitioning based on permutation test 방법을 사용한다. p-test를 거친 significance를 기준으로 가지치기를 할 변수를 결정하기 때문에 별도로 가지치기를 하지 않아도 된다.

이에 따라 training dataset을 통해 학습한 모델은 다음과 같다.



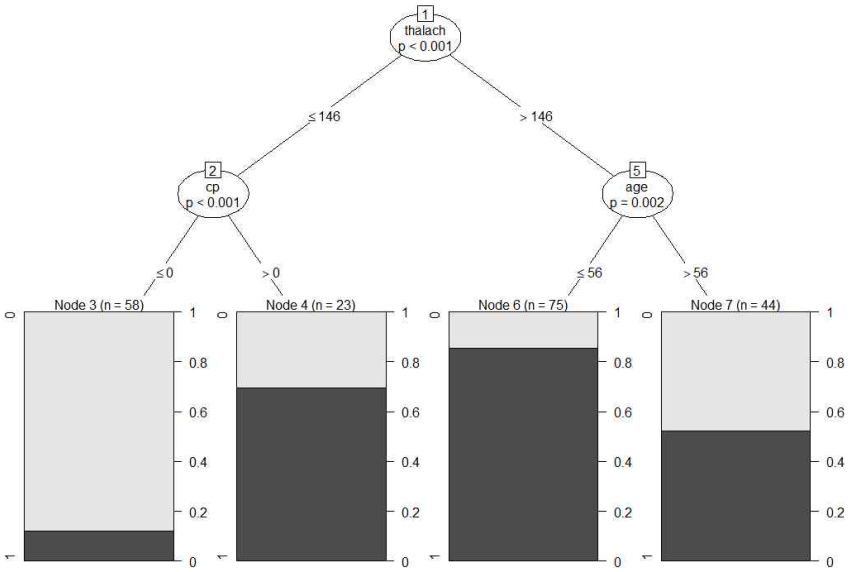
terminal node의 개수는 6개이고 분류 정확도도 꽤 높아보인다. 이 모델을 best 모델로 두고, 다른 옵션들을 바꿔가면서 tree를 만들어 보자.

먼저 mincriterion은 test statistic의 p value의 임계값을 설정하는 옵션이다. mincriterion을 0.99로 설정하면 다음과 같은 결과를 보인다.

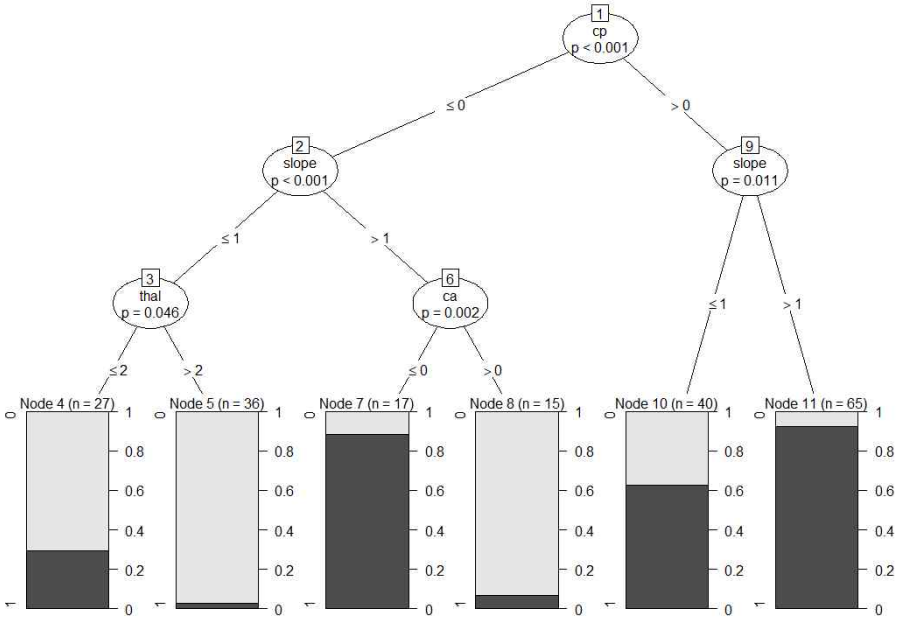


leaf node의 개수가 best model에 비해 하나가 줄은 것을 볼 수 있다.

다음은 mtry 옵션을 변화시켜보자. mtry는 입력 변수의 개수를 조절하는 변수이다. 입력변수는 random하게 선택된다.
mtry를 3으로 설정한 결과는 다음과 같다.



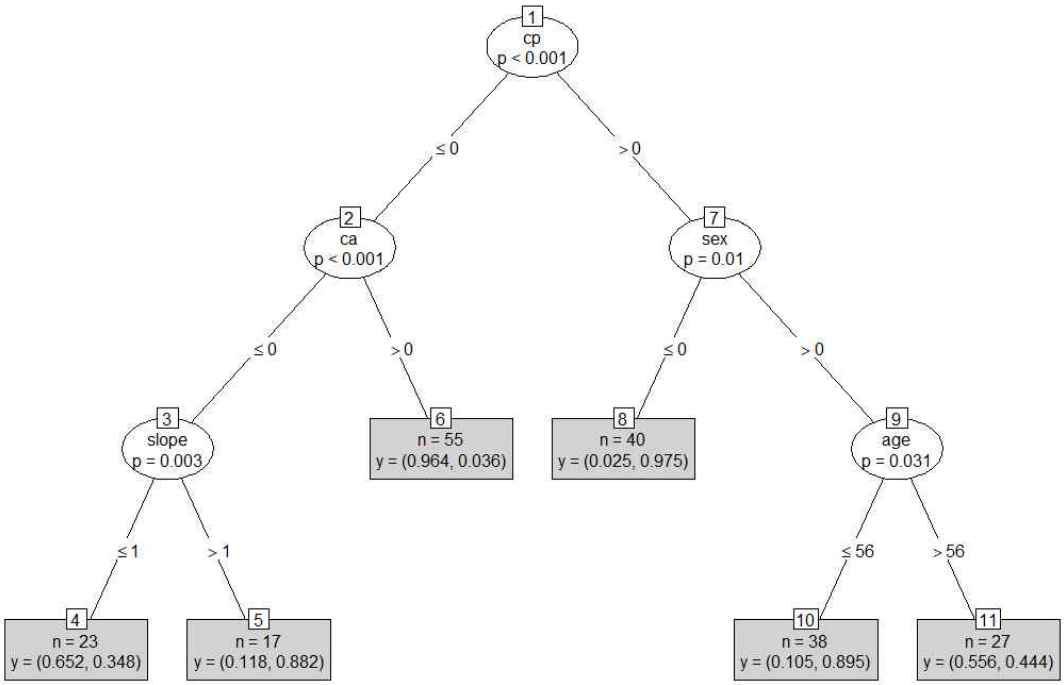
thalach, cp, age 변수가 선택되었고 leaf node의 개수는 4개 이다.
mtry를 4로 설정한 결과는 다음과 같다.



cp, slope, thal, ca 변수가 선택되었고, leaf node의 개수는 6개이다.

[Q3-4] Various Plotting

plot의 simple method를 사용하면 다음과 같은 시각화가 가능하다.



terminal node의 분포를 수치로 알 수 있다.

[Q3-5] 'party' best model Prediction

위에서 얻은 pruned model을 통해 validation을 진행해보도록 하자.

학습한 모델을 통해 validation dataset에 대한 예측을 진행한 Confusion matrix와 Classification Performance는 다음과 같다.

	party.prey	
	0	1
0	35	13
1	14	41

먼저 confustion matrix의 결과를 보면 실제 심장병 환자(=1) 55명 중 41명을 심장병 환자일 것이라고 예측하였다. 그리고, 실제 심장병 환자가 아닌 사람(=0) 48명 중 35명을 심장병 환자가 아닐 것이라고 예측하였다.

	TPR	Precision	TNR	Accuracy	BCR	F1-Measure
Tree	0.8363636	0.7187500	0.6250000	0.7378641	0.7229988	0.7731092
Tree_pruned	0.8727273	0.6956522	0.5625000	0.7281553	0.7006490	0.7741935
rpart	0.8727273	0.6956522	0.5625000	0.7281553	0.7006490	0.7741935
party	0.7454545	0.7592593	0.7291667	0.7378641	0.7372656	0.7522936

다음은 Classification Performance 값이다. tree와 rpart에 비해서 accuracy 값은 약 0.01 상승하였고, TPR, F1-Measure는 다소 하락하였으나 Precision과 TNR에서 다소 높은 값을 보여줬다. 이는 실제 심장병 환자가 아닌 사람(=0)에 대한 예측이 좋아진 결과로 해석 할 수 있다.

[Q3] ‘evtree’ package

[Q3-1]

evtree package를 사용하였다.

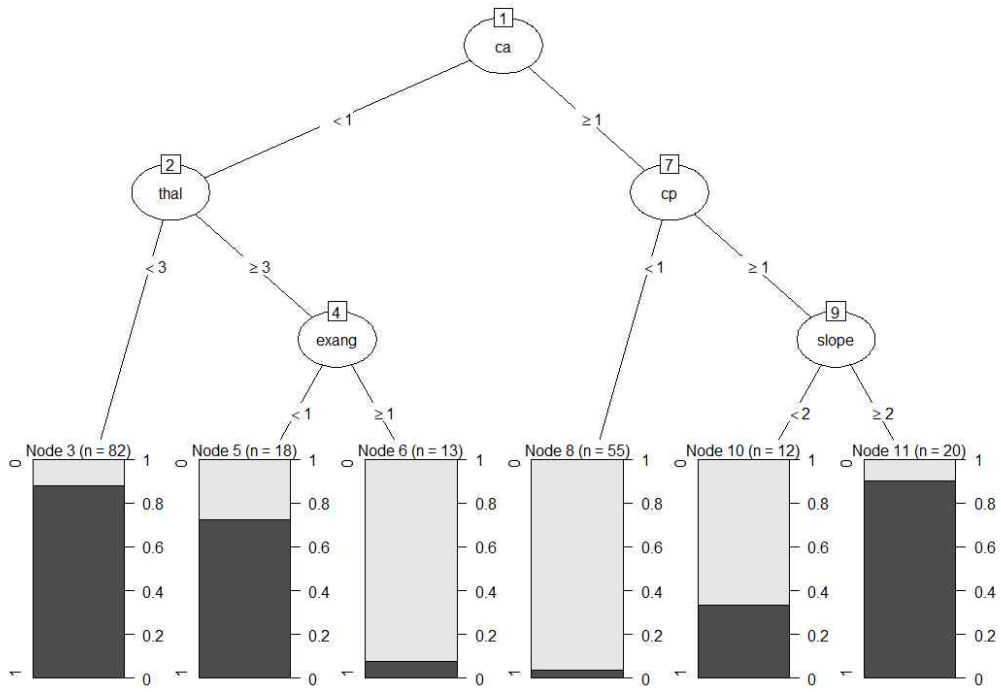
[Q3-2] Options

`minbucket`	the minimum sum of weights in a terminal node.
`minsplit`	the minimum sum of weights in a node in order to be considered for splitting.
`maxdepth`	maximum depth of the tree. Note, that the memory requirements increase by the square of the m'aximum tree depth.
`niterations`	in case the run does not converge, it terminates after a specified number of iterations defined by `niterations`.
`ntrees`	the number of trees in the population.
`alpha`	regulates the complexity part of the cost function. Increasing values of alpha encourage decreasing tree sizes.
`operatorprob`	list or vector of probabilities for the selection of variation operators. May also be specified partially in which case the default values are still used for the unspecified arguments. Always scaled to sum to 100 percent.
`seed`	an numeric seed to initialize the random number generator (for reproducibility). By default the seed is randomly drawn using `runif` in order to inherit the state of `.Random.seed`. If set to `seed = -1L`, the random number generator is initialized by the system time.

minbucket, minsplit, maxdepth는 위의 패키지들과 같은 옵션이고, evtree는 진화 알고리즘을 사용하여 decision tree를 만드는 방법이다. 이에 따라 niteration은 수렴하지 않을 때, 반복수를 설정해줄수 있고, ntrees는 tree의 개수를 설정할 수 있다. alpha는 손실함수의 parameter이고 operatorprob은 mutation rate와 cross-over rate를 설정할 수 있는 옵션이다.

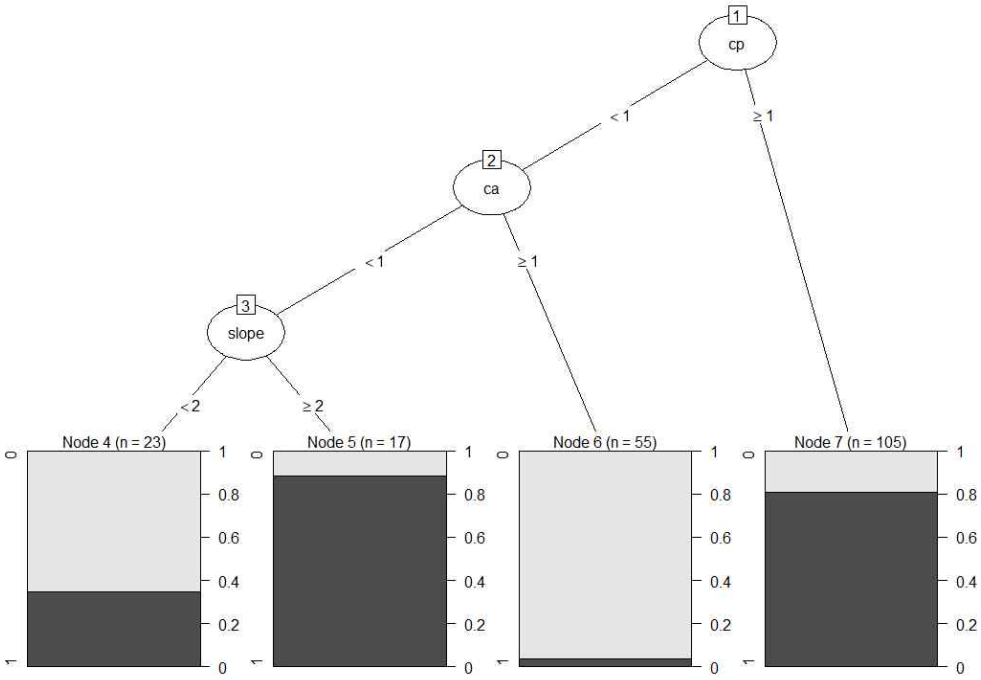
[Q3-3] Best model ‘evtree’

먼저 아무 옵션도 설정하지 않았을 때의 tree는 다음과 같다.

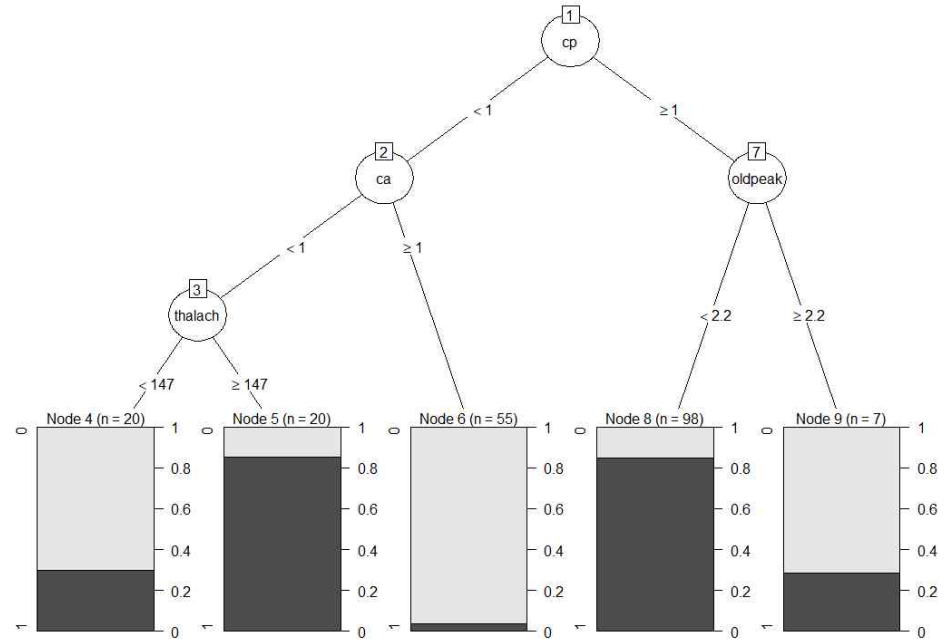


leaf nodes의 개수는 6개이고 꽤 높은 분류 정확도를 보인다.

위의 모델은 진화 알고리즘의 결과로 수렴이 된 결과이지만 극단적으로 niteration을 50으로 설정하면 수렴되지 않았다는 경고 문구와 함께 다음과 같은 결과를 보인다.

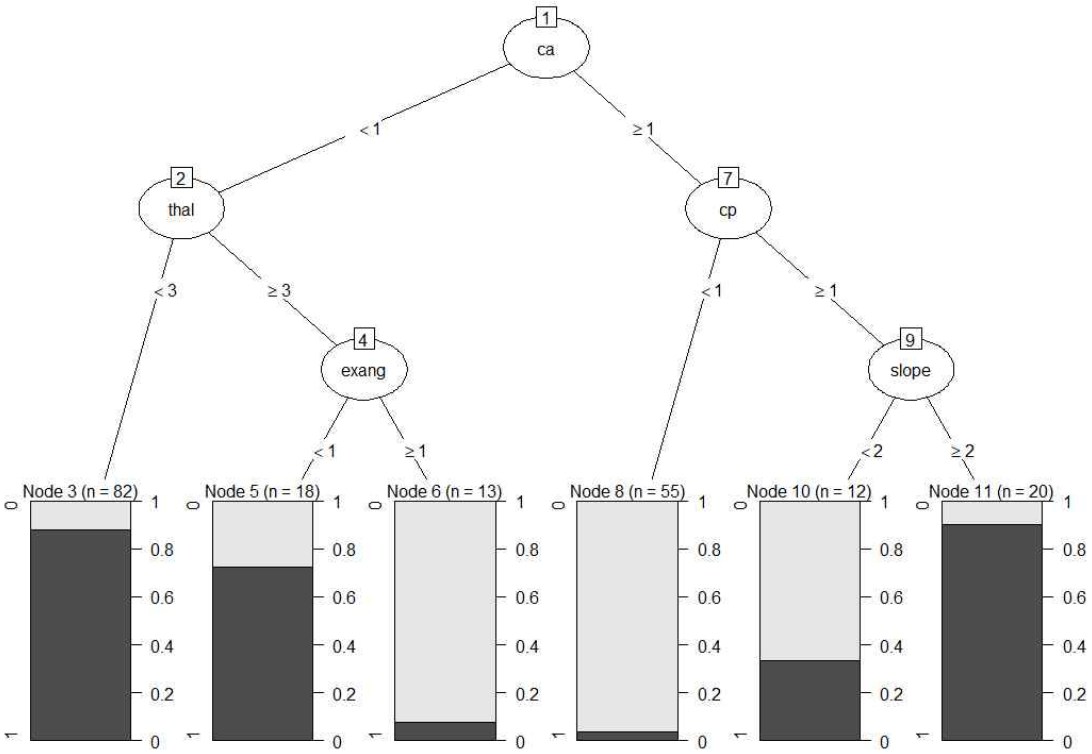


3개의 변수만 선택되었으며 leaf node의 개수는 4개이다.
다음은 ntrees를 굉장히 작은 값인 10으로 설정한 결과이다.



4개의 입력변수가 선택되었으며 leaf node의 개수는 5개이다.
niteration과 ntrees를 설정하지 않으면 default값으로 꽤 높은 값이 설정되고, 이 모델에 있어서는 최적의 모델로 수렴되는 것 같다.

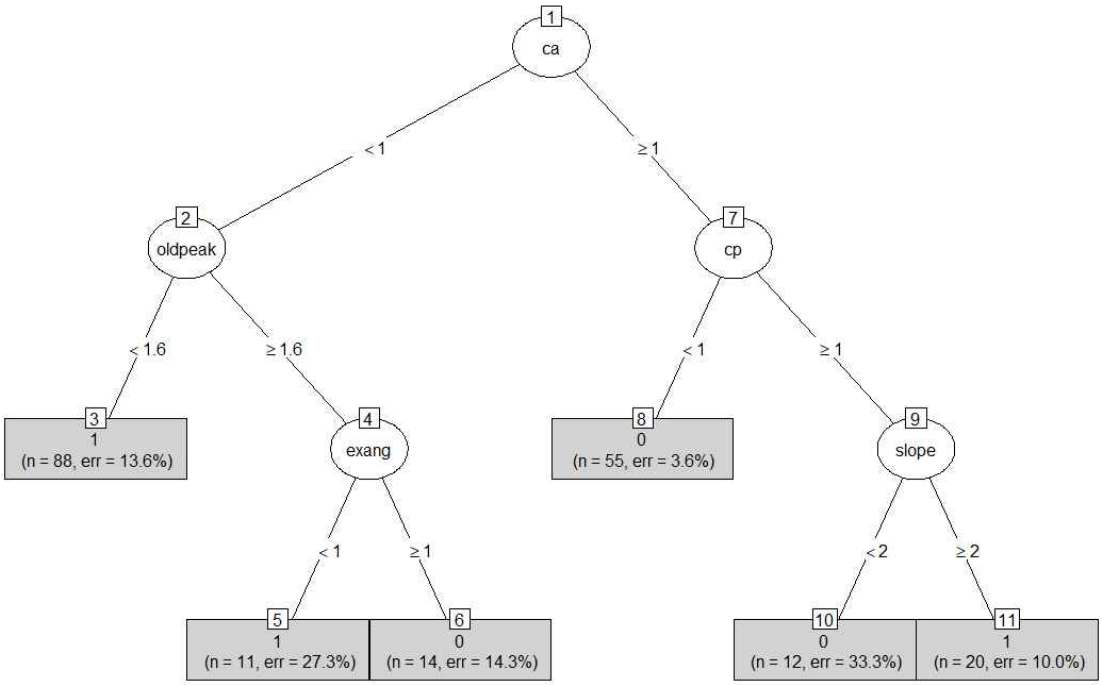
마지막으로 pmutatemajor를 0.5로 설정하고, pmutateminor를 0.2로 설정하고, pcrossover를 0.01로 설정하여 tree를 만들어 보았다.



niteration과 ntrees가 충분한 값을 가져서 그런지 default 값을 가지는 모델과 다르지 않다. 이에 따라 option들을 default로 설정한 모델을 best model로 설정하고 예측을 진행하도록 하자.

[Q3-4]

이전의 party 패키지와 마찬가지로 simple type의 plotting이 가능하다.



[Q3-5] 'evtree' best model Prediction

학습한 모델을 통해 validation dataset에 대한 예측을 진행한 Confusion matrix와 Classification Performance는 다음과 같다.

	evtree.prey	
	0	1
0	30	18
1	5	50

먼저 confustion matrix의 결과를 보면 실제 심장병 환자(=1) 55명 중 50명을 심장병 환자일 것이라고 예측하였다. 그리고, 실제 심장병 환자가 아닌 사람(=0) 48명 중 30명을 심장병 환자가 아닐 것이라고 예측하였다. 위에서 사용한 패키지들에 비해 심장병 환자 예측력이 상당히 높아졌다.

	TPR	Precision	TNR	Accuracy	BCR	F1-Measure
Tree	0.8363636	0.7187500	0.6250000	0.7378641	0.7229988	0.7731092
Tree_pruned	0.8727273	0.6956522	0.5625000	0.7281553	0.7006490	0.7741935
rpart	0.8727273	0.6956522	0.5625000	0.7281553	0.7006490	0.7741935
party	0.7454545	0.7592593	0.7291667	0.7378641	0.7372656	0.7522936
evtree	0.9090909	0.7352941	0.6250000	0.7766990	0.7537784	0.8130081

다음은 Classification Performance 값이다. 먼저 Accuracy 지표에서 약 0.77로 가장 높은 수치를 보였으며 BCR, F1-Measure도 가장 높은 값인 0.75, 0.81를 가진다. 실제 심장병 환자(=1)를 굉장히 높은 확률로 예측하기 때문에 약 0.90의 TPR 값을 가지고, 실제 심장병 환자가 아닌 사람(=0)을 예측하는 것은 party 패키지보다는 좋지 않은 예측력을 가진다.

만약 decision tree를 구축하는데에 하나의 패키지 방법론만 사용하라고 한다면 이 보고서를 작성한 저는 evtree 패키지를 사용할 것이다.