채널 G
2016125077 최재혁
2015125043 어성준
2015125080 표정진

데이터 사이언스

- 8주차 팀 과제 (인공신경망과 svm을 통한 심장 질환자 분석)

목 차

데이터 셋 선택, 전처리

- 데이터 셋 결정

  지난 주에 사용했던 heart data set의 독립 변수간에 독립성이 비교적 높게 나타나, 분류모델을 만드는데 괜찮을 것으로 보여 선택함

- 데이터 전처리
  - na.omit() 함수를 이용하여 컬럼에 존재하는 결측치 제거
  - 명목형 변수인 경우, factor()함수를 통하여 순서형 변수로 변환
  - 정수 값은, numeric()함수를 통하여 num type으로 변환

```
> ht <- na.omit(ht[,-1])
```

<사진 1> 결측 값 제거

```
> sex <- as.numeric(ht$Sex)
> cp <- factor(ht$ChestPain, levels = c("typical", "nontypical" ,
+                                       "asymptomatic", "nonanginal"))
> ca <- as.numeric(ht$Ca)
> thal <- factor(ht$Thal, levels = c("fixed", "normal", "reversable"))
> AHD <- factor(ht$AHD, levels = c("No", "Yes"))
```

<사진 3> 형 변환

#age - 나이
#sex - (1 = 남성; 0 = 여성)
#cp - 가슴 통증 유형(0, 1, 2, 3, 4)
#trestbps - 안정 혈압(병원 입원시 mm Hg)
#chol - 혈청 콜레스테롤(mg/dl)
#fbs - (공복 혈당 > 120 mg/dl)(1 = true; 0 = false)
#restecg - 안정 심전도 결과(0, 1, 2)
#thalach - 최대 심박동수
#exang - 협심증 유발 운동(1 = yes; 0 = no)
#oldpeak - 비교적 안정되기까지 운동으로 유발되는 ST depression
#slope - 최대 운동 ST segment의 기울기
#ca - 형광 투시된 주요 혈관의 수(0-3)
#thal - (3 = 보통; 6 = 해결된 결함; 7 = 해결가능한 결함)
#target - 심장병 진단(1 = true; 0 = false)

<사진 2> 각 속성 값에 대한 설명

유효한 독립변수 선택

- 유의한 변수 추출
  - 다변량 회귀 분석을 통하여 (lm 함수 사용) 모델을 생성하고 확인한 결과, p-value 값이 2.2e-16보다 작기 때문에 회귀 계수들은 유의미하며, Pr(>|t|) t통계량의 값이 0.05 보다 작은 4가지 변수를 선택
  - 이를 가지고 새로운 데이터 셋을 구성

```
> #회귀분석을 통해서 유의한 변수 추출
> model0 <- lm(target~.,data = heart2)
> summary(model0)

Call:
lm(formula = target ~ ., data = heart2)

Residuals:
     Min       1Q   Median       3Q      Max
-0.94748 -0.21270  0.06608  0.25022  0.93509

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.8288987  0.2929344   2.830 0.004987 **
age         -0.0008204  0.0026962  -0.304 0.761129
sex         -0.1959956  0.0471429  -4.157 4.24e-05 ***
cp           0.1127034  0.0223816   5.036 8.40e-07 ***
trestbps    -0.0019910  0.0012573  -1.583 0.114407
chol        -0.0003535  0.0004217  -0.838 0.402545
fbs          0.0173736  0.0596669   0.291 0.771125
restecg      0.0498480  0.0399228   1.249 0.212819
thalach      0.0030193  0.0011304   2.671 0.007988 **
exang       -0.1440459  0.0513689  -2.804 0.005387 **
oldpeak     -0.0587887  0.0229269  -2.564 0.010847 *
slope        0.0789788  0.0423896   1.863 0.063453 .
ca          -0.1006022  0.0218565  -4.603 6.25e-06 ***
thal        -0.1190392  0.0356550  -3.339 0.000952 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3542 on 289 degrees of freedom
Multiple R-squared:  0.5175,    Adjusted R-squared:  0.4958
F-statistic: 23.85 on 13 and 289 DF,  p-value: < 2.2e-16
```

<사진 1> 회귀 분석의 결과

```
> str(ht)
'data.frame':   297 obs. of  5 variables:
 $ sex : num  1 1 1 1 0 1 0 0 1 1 ...
 $ cp  : Factor w/ 4 levels "typical","nontypical",..: 1 3 3 4 2 2 3 3 3 3
 $ ca  : num  0 3 2 0 0 0 2 0 1 0 ...
 $ thal: Factor w/ 3 levels "fixed","normal",..: 1 2 3 2 2 2 2 2 3 3 ...
 $ AHD : Factor w/ 2 levels "No","Yes": 1 2 2 1 1 1 2 1 2 2 ...
```

<사진 2> 전처리 후 Data set 의 특성

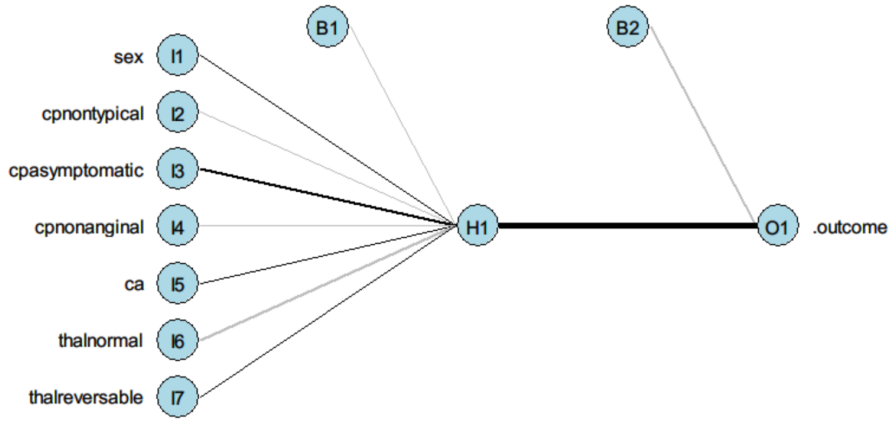
인공 신경망 모델

- 인공 신경망 모델의 파라미터 최적화
  - 10fold CV 2회 반복, nnet 모델 100회 반복하도록 설정. 반복수를 늘려도 일정 치 이상(약 60회) 수렴됨을 확인. Best decay는 0.6

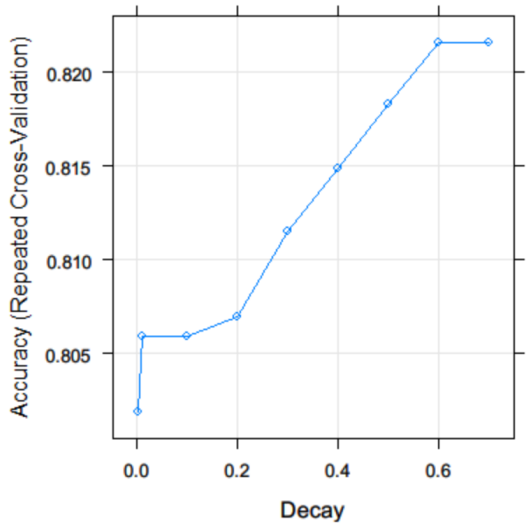
```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 2,
                           classProbs = TRUE)
nnet <- train(AHD~.,
              data = train,
              method = "nnet",
              metric = "Accuracy",
              maxit = 50,
              linout = FALSE,
              trControl = fitControl,
              trace = TRUE,
              tuneGrid = expand.grid(size=1, decay=c(0.001, 0.01, 0.1, 0.2,
                                                     0.3, 0.4, 0.5, 0.6, 0.7)))
```

<사진 1> nnet 모델 파라미터

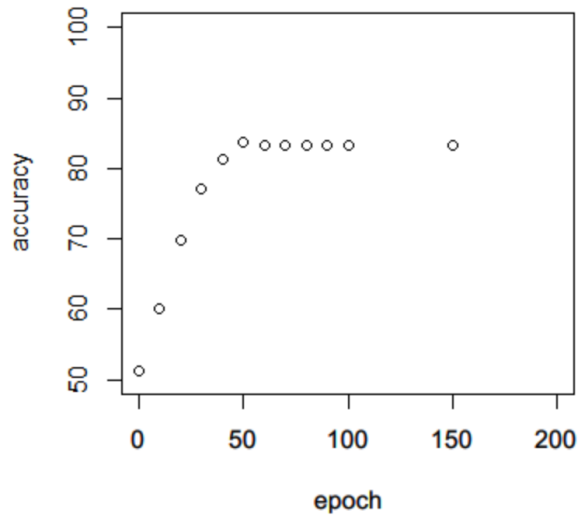
- 모델 생성, 신경망 plot



<사진 2> nnet 모델



<사진 3> Decay vs Accuracy 곡선



<사진 4> epoch vs Accuracy 곡선

SVM 모델

- SVM 모델의 다양한 Method

```
> names(getModelInfo("svm"))
 [1] "lssvmLinear"          "lssvmPoly"            "lssvmRadial"
 [4] "ORFsvm"               "svmBoundrangeString"  "svmExpoString"
 [7] "svmLinear"            "svmLinear2"           "svmLinear3"
[10] "svmLinearWeights"     "svmLinearWeights2"    "svmPoly"
[13] "svmRadial"            "svmRadialCost"        "svmRadialSigma"
[16] "svmRadialWeights"     "svmSpectrumString"
```

<사진 1> svm methods

- SVM 모델 생성 (by svmLinear2)
  - SVM에 있는 method 중 대표적인 svmLinear2를 사용하였으며, metric에는 “ROC”함수를 이용하였다.
  - trControl은 신경망 모델과 동일

※ metric 옵션은 "accuracy", "Kappa", "RMSE", "Rsquared“ 들이 존재, 이 옵션은 최종모형의 선택에 사용되는 목적함수를 지정

```
> svm <- train(AHD~.,
+              data = train,
+              method = "svmLinear2",
+              metric = "ROC",
+              preProcess = c("center","scale"),
+              trControl = fitControl
+              #tuneGrid = expand.grid(C = seq(0, 2, length = 20))
+              )
```

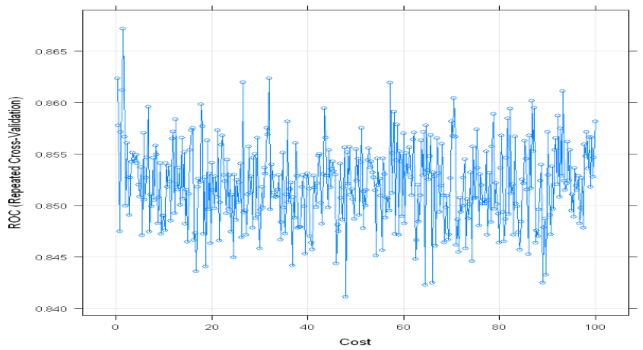
<사진 2> svm 모델

SVM 모델

- ROC 그래프와 모델 튜닝
  - 모델을 plot함수를 이용하여 그래프로 표현한 형태
  - bestTune옵션을 주어 cost 파라미터 확인

```
> plot(svm)
> svm$bestTune
  cost
```

<사진 1> svm methods



<사진 2> 비용 vs ROC

- SVM 최적화 모델 생성
  - Cost 변수를 추가하여 모델성능 향상 확인

```
svm <- train(AHD~.,
             data = train,
             method = "svmLinear2",
             metric = "ROC",
             preProcess = c("center","scale"),
             trControl = fitControl,
             tuneGrid = expand.grid(cost = 1.5))
```

<사진 3> 최적화 모델 생성

```
          Reference
Prediction No Yes
       No  32   8
       Yes  3  31

               Accuracy : 0.8514
                 95% CI : (0.7496, 0.9234)
    No Information Rate : 0.527
    P-Value [Acc > NIR] : 4.072e-09

                  Kappa : 0.704

 Mcnemar's Test P-Value : 0.2278

            Sensitivity : 0.7949
            Specificity : 0.9143
         Pos Pred Value : 0.9118
         Neg Pred Value : 0.8000
             Prevalence : 0.5270
         Detection Rate : 0.4189
   Detection Prevalence : 0.4595
      Balanced Accuracy : 0.8546

       'Positive' Class : Yes
```

→

```
          Reference
Prediction No Yes
       No  32   3
       Yes  8  31

               Accuracy : 0.8514
                 95% CI : (0.7496, 0.9234)
    No Information Rate : 0.5405
    P-Value [Acc > NIR] : 1.477e-08

                  Kappa : 0.704

 Mcnemar's Test P-Value : 0.2278

            Sensitivity : 0.8000
            Specificity : 0.9118
         Pos Pred Value : 0.9143
         Neg Pred Value : 0.7949
             Prevalence : 0.5405
         Detection Rate : 0.4324
   Detection Prevalence : 0.4730
      Balanced Accuracy : 0.8559

       'Positive' Class : No
```

<사진 4> 기존 모델(왼)과의 비교

SVM 모델

- 다른 함수들과 비교
  - 다른 함수들 왼(“svmRadial”), 오(“svmPoly”)와 성능 비교를 했을 때, svmLinear2 메서드가 더 성능이 좋은 것으로 판단
  - 또한 각자 최적화된 파라미터(svmRadial -> (sigma, C), svmPoly -> (degree,Scale,C))로 모델을 만들어 비교하여도 svmLinear2가 더 적합

```
          Reference
Prediction No Yes
       No  32   4
       Yes  8  30

               Accuracy : 0.8378
                 95% CI : (0.7339, 0.9133)
    No Information Rate : 0.5405
    P-Value [Acc > NIR] : 6.769e-08

                  Kappa : 0.6764

 Mcnemar's Test P-Value : 0.3865

            Sensitivity : 0.8000
            Specificity : 0.8824
         Pos Pred Value : 0.8889
         Neg Pred Value : 0.7895
             Prevalence : 0.5405
         Detection Rate : 0.4324
   Detection Prevalence : 0.4865
      Balanced Accuracy : 0.8412

       'Positive' Class : No
```

<사진 1> “svmRadial”을 사용한 모델

```
          Reference
Prediction No Yes
       No  28   2
       Yes 12  32

               Accuracy : 0.8108
                 95% CI : (0.703, 0.8925)
    No Information Rate : 0.5405
    P-Value [Acc > NIR] : 1.077e-06

                  Kappa : 0.6273

 Mcnemar's Test P-Value : 0.01616

            Sensitivity : 0.7000
            Specificity : 0.9412
         Pos Pred Value : 0.9333
         Neg Pred Value : 0.7273
             Prevalence : 0.5405
         Detection Rate : 0.3784
   Detection Prevalence : 0.4054
      Balanced Accuracy : 0.8206

       'Positive' Class : No
```

<사진 1> “svmPoly”를 사용한 모델

모델의 정확도 비교

- 의사 결정나무 vs 인공신경망 vs SVM 성능 비교

```
          Reference
Prediction No Yes
       No  34   6
       Yes  3  31

               Accuracy : 0.8784
                 95% CI : (0.7816, 0.9429)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : 6.755e-12

                  Kappa : 0.7568

 Mcnemar's Test P-Value : 0.505

            Sensitivity : 0.9189
            Specificity : 0.8378
         Pos Pred Value : 0.8500
         Neg Pred Value : 0.9118
             Prevalence : 0.5000
         Detection Rate : 0.4595
   Detection Prevalence : 0.5405
      Balanced Accuracy : 0.8784

       'Positive' Class : No
```

```
          Reference
Prediction No Yes
       No  33   7
       Yes  2  32

               Accuracy : 0.8784
                 95% CI : (0.7816, 0.9429)
    No Information Rate : 0.527
    P-Value [Acc > NIR] : 1.277e-10

                  Kappa : 0.7578

 Mcnemar's Test P-Value : 0.1824

            Sensitivity : 0.9429
            Specificity : 0.8205
         Pos Pred Value : 0.8250
         Neg Pred Value : 0.9412
             Prevalence : 0.4730
         Detection Rate : 0.4459
   Detection Prevalence : 0.5405
      Balanced Accuracy : 0.8817

       'Positive' Class : No
```

```
          Reference
Prediction No Yes
       No  32   8
       Yes  3  31

               Accuracy : 0.8514
                 95% CI : (0.7496, 0.9234)
    No Information Rate : 0.527
    P-Value [Acc > NIR] : 4.072e-09

                  Kappa : 0.704

 Mcnemar's Test P-Value : 0.2278

            Sensitivity : 0.9143
            Specificity : 0.7949
         Pos Pred Value : 0.8000
         Neg Pred Value : 0.9118
             Prevalence : 0.4730
         Detection Rate : 0.4324
   Detection Prevalence : 0.5405
      Balanced Accuracy : 0.8546

       'Positive' Class : No
```

- 정확도 : 인공신경망 = 의사결정나무 > SVM
- 민감도 : 인공신경망 > 의사결정나무 ≒ SVM
- 특이도 : 의사결정나무 > 인공신경망 > SVM
- 맥니마 검정 지수 : 인공신경망 > SVM > 의사결정나무
- 균형 정확도 : 인공신경망 > 의사결정나무 > SVM

다방면에서 인공신경망이 제일 정확히 분류하였다고 볼 수 있음.

결론 및 고찰

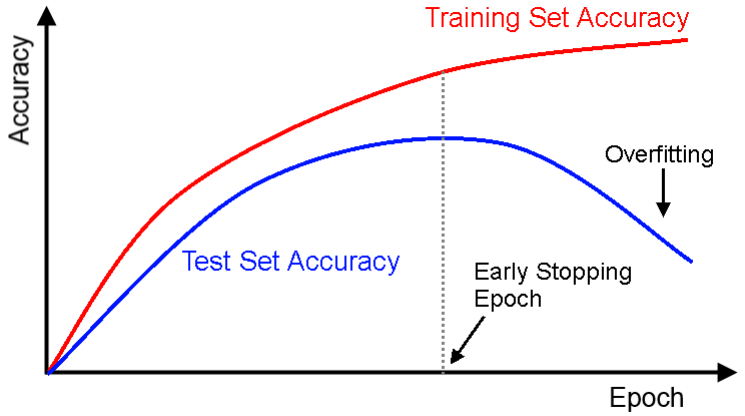
- 결론 및 고찰

  - 1. epoch값에 정확도가 정비례하지 않는다

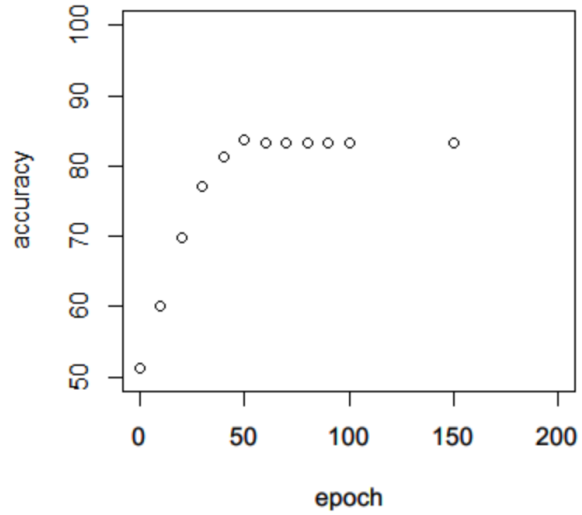
  - 2. svmLinear의 cost와 그에 따른 정확도

결론 및 고찰

- maxit값에 비례해 정확도가 증가하지 않는 이유
  - train() 함수는 maxit(최대 반복수) 파라미터로 데이터 셋 반복 학습 수를 결정하고 그중 가장 좋은 모델을 반환함.
  - epoch가 커질수록 보통 학습 데이터 셋에 대한 정확도는 늘어나지만 테스트 데이터에 대한 정확도는 떨어지는 overfitting이 발생.
  - 그러므로 일정 값 이상으로는 정확도가 수렴.



<사진 1> Epoch vs Accuracy

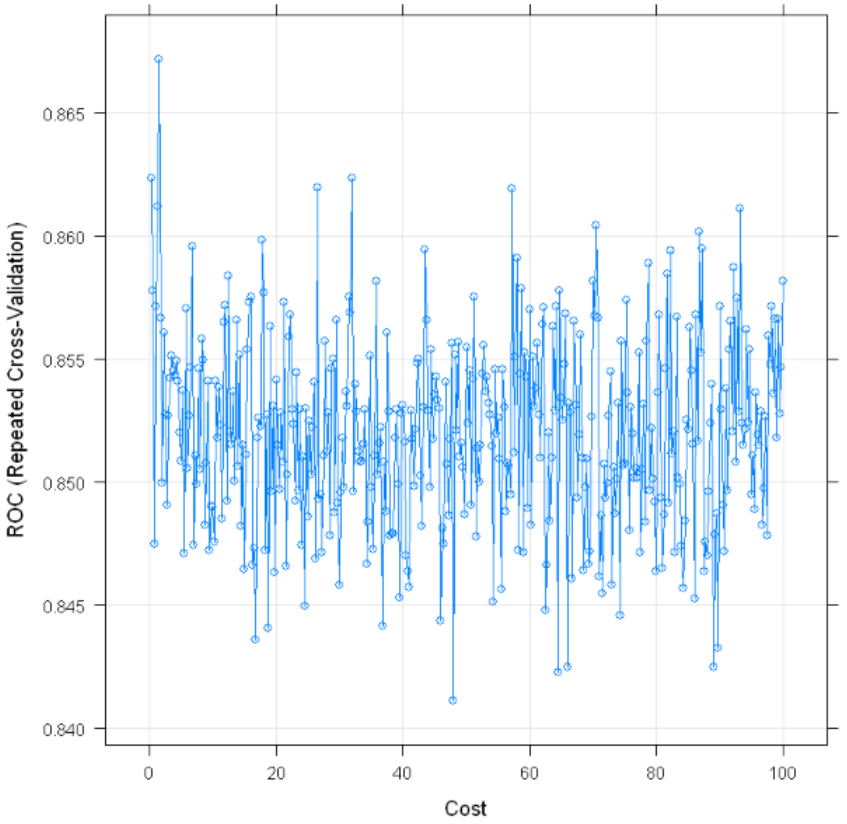


<사진 2> Epoch(maxit) vs Accuracy

결론 및 고찰

- svm의 cost와 그에 따른 정확도

  - 마진이 클수록 좋은 결정 경계.
  - cost : 오류(오염점)의 허용 정도, 마진을 침범할 수준.
  - Heart 데이터 셋의 크기가 작고 ROC값의 편차가 작아 svm$bestTune에 의한 가장 좋은 cost 값과 가장 좋지 않은 cost 값에 따른 정확도의 차이가 생기지 않음.



<사진 1> Cost vs ROC