데이터마이닝(DataMining)

Chapter 2.1. 데이터 전처리

- 데이터마이닝 분석에서 고려해야 할 데이터의 전처리 과정과 구축된 모형에 대한 평가 방법

- 데이터의 전처리(pre-processing)는 모형을 구축하기 전 단계에서 수행되며, 모형의 성능을 제고하는 데 중요

- 데이터 정제(cleaning), 정규화, 변환 및 변수 추출과 선택 등이 포함

- 구축된 모형에 대한 평가 방법과 평가를 위한 여러 가지 척도를 소개

데이터 전처리

- 영- 과 영근처-분산 예측변수의 처리
- 상관된 예측변수 식별: 중복 변수 제거
- 예측변수의 변환
- 기타 전처리 방법

- 영- 과 영근처-분산 예측변수의 처리
  - 일부 상황에서 데이터 생성 메커니즘은 한 개의 값만을 취하는(영-분산) 예측변수를 생성할 수 있음
  - 많은 모형(트리 기반 모델 제외)에서, 영-분산 예측변수는 모형을 망가뜨리거나 불안정한 적합 의 원인
  - 마찬가지로 예측변수는 매우 낮은 빈도로 발생하는 몇 개의 값만을 취할 수도 있음
  - mdrr{caret} 자료

```
> data(mdrr)    # 다중약물내성 자료(528개 관측치, 342개 변수)
> data.frame(table(mdrrDescr$nR11))
  Var1 Freq
1    0  501
2    1    4
3    2   23
```

- 주의할 점은 이들 예측변수들이 교차타당성/붓스트랩 하위 샘플로 분할될 때 영-분산 예측변수가 되거나, 일부 샘플이 모형에 과도한 영향을 미치게 되는 경우

- 이러한 영근처-분산 예측변수는 모형화 이전에 식별되고 제거되어야 함

- 이러한 유형의 예측변수를 식별하기 위한 다음의 두 가지 측도

  - 빈도 비율(frequency ratio): (일 순위 빈발값의 빈도)를 (차 순위 빈발값의 빈도)로 나눈 값. 정상적인 예측변수에서는 1에 가까운 값을 가지고, 매우 불균형적인 데이터에 대해서는 매우 큰 값을 가짐

  - 유일 값들(unique values)의 비율: 유일한 값들의 수(종류)를 전체 표본의 수로 나눈 값(100)이다. 데이 터의 세분화가 증가함에 따라 0에 가까워짐

- 빈도 비율이 미리 지정된 임계값보다 크고 유일한 값들의 비율이 임계값 보다 작으면 예측변수 가 영-분산에 가깝다고 간주할 수 있음

- mdrr 자료에서 nearZeroVar() 함수를 사용하여 영근처-분산의 변수를 식별 할 수 있음

- nearZeroVar() 함수의 호출시 saveMetrics=TRUE 옵션을 사용하면 각 예측값에 대한 빈도비 율(freqRatio)과 유일 값들의 비율(percentUnique)을 얻을 수 있으며, 동시에 각 변수가 영-분 산 또는 영-근처 분산을 가지는지를 알려줌

- 디폴트로, 표본에서 유일 값의 비율이 10% 이하이고 빈도비율이 19(95/5)보다 큰 예측변수는 영-근처 분산으로 분류

```
> nzv <- nearZeroVar(mdrrDescr, saveMetrics = TRUE)
> str(nzv); nzv[nzv$nzv, ][1:10,]
 'data.frame': 342 obs. of 4 variables:
  $ freqRatio    : num 1.25 1.12 1 1.25 1.25 ...
  $ percentUnique: num 90 42.6 83 84.3 82.8 ...
  $ zeroVar      : logi FALSE FALSE FALSE FALSE FALSE FALSE ...
  $ nzv          : logi FALSE FALSE FALSE FALSE FALSE FALSE ...

      freqRatio percentUnique zeroVar  nzv
nTB     23.00000     0.3787879   FALSE TRUE
nBR    131.00000     0.3787879   FALSE TRUE
nI     527.00000     0.3787879   FALSE TRUE
nR03   527.00000     0.3787879   FALSE TRUE
nR08   527.00000     0.3787879   FALSE TRUE
nR11    21.78261     0.5681818   FALSE TRUE  # (501/23=21.78, 3/528=0.0568)
nR12    57.66667     0.3787879   FALSE TRUE
```

- 기본적으로 nearZeroVar() 함수는 문제가 되는 변수의 위치를 반환

```
D.Dr03 527.00000     0.3787879   FALSE TRUE
D.Dr07 123.50000     5.8712121   FALSE TRUE
D.Dr08 527.00000     0.3787879   FALSE TRUE
```

```
> dim(mdrrDescr)
 [1] 528 342
```

```
> nzv <- nearZeroVar(mdrrDescr)
> nzv    # 영근처-분산을 가지는 변수
 [1] 22 31 32 34 38 41 42 259 262 ...
 [41] 338 339 340 341 342
> filteredDescr <- mdrrDescr[, -nzv]
> dim(filteredDescr)
 [1] 528 297    # 297개 변수(영근처-분산 제거 후)
```

- 상관된 예측변수의 식별: 중복 변수 제거
  - 상관관계가 있는 예측변수에 대해서도 잘 작동하는 일부 모형이 있지만(예: PLS 회귀), 다른 모형들은 예측변수들 간의 상관관계의 수준을 줄이는 것이 좋음

  - 주어진 상관 행렬에 대해 findCorrelation() 함수는 다음 절차를 통해 제거해야 할 예측변수를 제공

```
> descrCor <- cor(filteredDescr)
> # 아래 sum() 함수: 논리형 자료의 합
> (highCorr <- sum(abs(descrCor[upper.tri(descrCor)]) > .999))
 [1] 65     # 상관계수가 0.999 이상인 경우가 65개임( 297C2=43,956개 중)
```

```
> summary(descrCor[upper.tri(descrCor)])
    Min.  1st Qu.  Median    Mean 3rd Qu.    Max.
-0.99610 -0.05373 0.25010 0.26080 0.65530 1.00000
```

- 아래의 코드는, 이전 mdrr 자료에서, 0.75 이상의 절대 상관계수를 갖는 예측변수를 제거하는 과정과 그 효과를 보여줌

```
> highlyCorDescr <- findCorrelation(descrCor, cutoff = 0.75)
> filteredDescr <- filteredDescr[, -highlyCorDescr]
> descrCor2 <- cor(filteredDescr)
> summary(descrCor2[upper.tri(descrCor2)])
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-0.70730 -0.05378  0.04418  0.06692  0.18860  0.74460
```

- 예측변수의 변환
  - 중심화와 척도화
    - preProcess() 함수는 중심화(centering)와 척도화(scaling)를 포함하여 예측변수에 대해 많은 연산을 제공
    - preProcess() 함수는 각 연산에 필요한 매개 변수를 추정하고, predict.preProcess() 함수는 특정 데이터 집합에 이를 적용하는 데 사용
    - 이 함수는 train() 함수를 호출 할 때 인터 페이스가 될 수도 있음
    - preProcess() 함수는 특정 데이터셋(예를 들어, 훈련용 자료)으로부터 요구하는 것을 추정한 다음, 이 값을 재계산하지 않고 임의의 데이터 세트에 이들 변환을 적용

- 이 예제에서 mdrr{caret} 자료의 절반은 예측변수의 위치와 척도를 추정하는데 사용
- preProcess() 함수는 실제로 데이터를 전처리하지 않음
- predict.preProcess() 함수는 이 데이터셋(훈련용)과 다른 데이터셋(검증용)을 전처리하는데 사용

```
> set.seed(200)
> inTrain <- sample(seq(along = mdrrClass), length(mdrrClass)/2)
> training <- filteredDescr[inTrain, ]
> test <- filteredDescr[-inTrain, ]
> trainMDRR <- mdrrClass[inTrain]
> testMDRR <- mdrrClass[-inTrain]
```

```
> # 중심화와 척도화 수행
> preProcValues <- preProcess(training, method = c("center",
"scale"))
> trainTransformed <- predict(preProcValues, training)
> testTransformed <- predict(preProcValues, test)
```

- preProcess() 함수에서 method= "ranges" 옵션은 0과 1 사이의 값으로 데이터를 변환

- 예측변수의 변환
  - 박스-콕스 변환
    - preprocess() 함수에서 method="BoxCox" 옵션은 예측변수에 대한 박스-콕스 변환의 차수를 추정 (데이터가 영보다 큰 경우에 한함)

```
> # 박스-콕스 변환 수행
> preProcValues2 <- preProcess(training, method = "BoxCox")
> trainBC <- predict(preProcValues2, training)
> testBC <- predict(preProcValues2, test)
> preProcValues2
 Created from 264 samples and 31 variables

 Pre-processing:
     - Box-Cox transformation (31)
     - ignored (0)

 Lambda estimates for Box-Cox transformation:
   Min. 1st Qu. Median   Mean 3rd Qu.   Max.
-2.0000 -0.2000 0.4000 0.4323  1.5500 2.0000
```

- NA 값은 변환될 수 없는 예측 인자에 해당
- 이 변환은 데이터가 0보다 커야함
- 두 가지 유사한 변형으로, Yeo-Johnson과 Manly(1976)의 지수 변환이 preProcess() 함수에 사용될 수 있음

- 기타 전처리 방법
  - 더미변수 생성
    - {caret}의 dummyVars() 함수는 하나 이상의 요인(factor)으로부터 완전한 더미 변수 집합을 생성

```
> library(earth)
> data(etitanic)
> str(etitanic)
 'data.frame': 1046 obs. of 6 variables:
 $ pclass      : Factor w/ 3 levels "1st","2nd","3rd": 1 1 1 1 1 1
1 1 1 1 ...
 $ survived    : int 1 1 0 0 0 1 1 0 1 0 ...
 $ sex         : Factor w/ 2 levels "female","male": 1 2 1 2 1 2
1 2 1 2 ...
 $ age         : num 29 0.917 2 30 25 ...
 $ sibsp       : int 0 1 1 1 1 0 1 0 2 0 ...
 $ parch       : int 0 2 2 2 2 0 0 0 0 0 ...
```

```
> # model.matrix{stats} 함수 이용
> head(model.matrix(survived ~ ., data = etitanic))
  (Intercept) pclass2nd pclass3rd sexmale     age sibsp parch
1           1         0         0       0 29.0000     0     0
2           1         0         0       1  0.9167     1     2
3           1         0         0       0  2.0000     1     2
4           1         0         0       1 30.0000     1     2
5           1         0         0       0 25.0000     1     2
6           1         0         0       1 48.0000     0     0
> # dummyVar() 함수 이용
> dummy.1 <- dummyVars(survived ~ ., data = etitanic)
> head(predict(dummy.1, newdata = etitanic))
  pclass.1st pclass.2nd pclass.3rd sex.female sex.male     age sibsp parch
1          1          0          0          1        0 29.0000     0     0
2          1          0          0          0        1  0.9167     1     2
3          1          0          0          1        0  2.0000     1     2
4          1          0          0          0        1 30.0000     1     2
5          1          0          0          1        0 25.0000     1     2
6          1          0          0          0        1 48.0000     0     0
```

- 절편은 없으며 각 요인은 각 수준에 대해 더미변수를 가지므로, 이러한 더미 변수는 lm()을 비롯한 일부 모형 함수에 유용하지 않을 수 있음에 유의

- 기타 전처리 방법
  - 선형 종속성
    - {caret}의 findLinearCombos() 함수는 행렬의 QR 분해를 사용하여 선형결합의 집합을 열거 (존재하는 경우)

```
> # 자료 생성
> ltfrDesign <- matrix(0, nrow = 6, ncol = 6)
> ltfrDesign[, 1] <- c(1, 1, 1, 1, 1, 1)
> ltfrDesign[, 2] <- c(1, 1, 1, 0, 0, 0)
> ltfrDesign[, 3] <- c(0, 0, 0, 1, 1, 1)
> ltfrDesign[, 4] <- c(1, 0, 0, 1, 0, 0)
> ltfrDesign[, 5] <- c(0, 1, 0, 0, 1, 0)
> ltfrDesign[, 6] <- c(0, 0, 1, 0, 0, 1)
```

    - 이 자료는 열들 간에 선형독립성이 성립하지 않음

- findLinearCombos() 함수는 이러한 종속성을 열거
- findLinearCombos() 함수는 선형종속성을 없애기 위해 제거할 열 위치의 벡터를 제공

```
> comboInfo <- findLinearCombos(ltfrDesign)
> comboInfo
$linearCombos
$linearCombos[[1]]
[1] 3 1 2

$linearCombos[[2]]
[1] 6 1 4 5

$remove
[1] 3 6
```

```
> ltfrDesign[, -comboInfo$remove]
     [,1] [,2] [,3] [,4]
 [1,]    1    1    1    0
 [2,]    1    1    0    1
 [3,]    1    1    0    0
 [4,]    1    0    1    0
 [5,]    1    0    0    1
 [6,]    1    0    0    0
```

- 기타 전처리 방법
  - 결측값 대치
    - preProcess() 함수는 훈련용 자료에서의 정보만을 기반으로 데이터셋의 결측값을 대치(impute) 하는 데 사용

    - 이를 수행하는 한 가지 방법은 k-근접 이웃을 이용하는 것. 임의의 하나의 표본에 대해, k 개 의 가장 가까운 이웃을 훈련용 자료에서 발견하고, 이들 값(예를 들어, 평균)을 이용하여 예측 변 수의 결측값을 대치

    - 이 접근법을 사용하면 method= 인자가 무엇이든 관계없이 자동적으로 preProcess()가 데이터 의 중심화와 척도화를 수행

- 대안으로, 베깅(bagged) 트리 모형이 대치에 사용
- 데이터의 각 예측변수에 대해, 훈 련용 자료의 다른 모든 예측변수를 사용하여 베깅 트리가 만들어짐
- 새로운 표본이 결측 예측 값을 가질 때, 이 값을 예측하는데 베깅 모형이 사용
- 이론적으로 이것은 더 강력한 대치 방법이지만 계산 비용은 근접 이웃 기법보다 훨씬 높음

- airquality 자료의 결측값을 k-인접 이웃 방법으로 대치한다. preProcess() 함수가 값을 돌려 주지는 않음

```
> library(caret)
> data(airquality); summary(airquality)    # summary(): 결측값의 개수
를 확인
> imp.1 <- preProcess(airquality, method=c("knnImpute"))
> library(RANN)
> imp.2 <- predict(imp.1, airquality); summary(airquality)
```

- 기타 전처리 방법
  - 군집 거리 계산
    - {caret} 패키지에는 군집 중심까지의 거리를 기반으로 새로운 예측 변수를 생성하는 함수가 포함 (선형판별분석 방법과 유사)
    - 요인 변수의 각 수준에 대해 군집의 중심과 공분산 행렬이 계산
    - 새로운 표본에 대해, 각 군 집 중심까지의 마할라노비스 거리가 계산되고 이 값은 추가 예측변수로 사용 (결정 경 계의 참값이 실제 선형일 때 비선형 모형에 대해 유용)
    - 표본보다 군집 내에 예측 변수가 더 많은 경우, classDist() 함수는 pca= 와 keep 인자를 통해, 각 군집 내에서 주성분분석이 가능하도록 하여, 특이(singular) 공분산행렬 문제를 해결

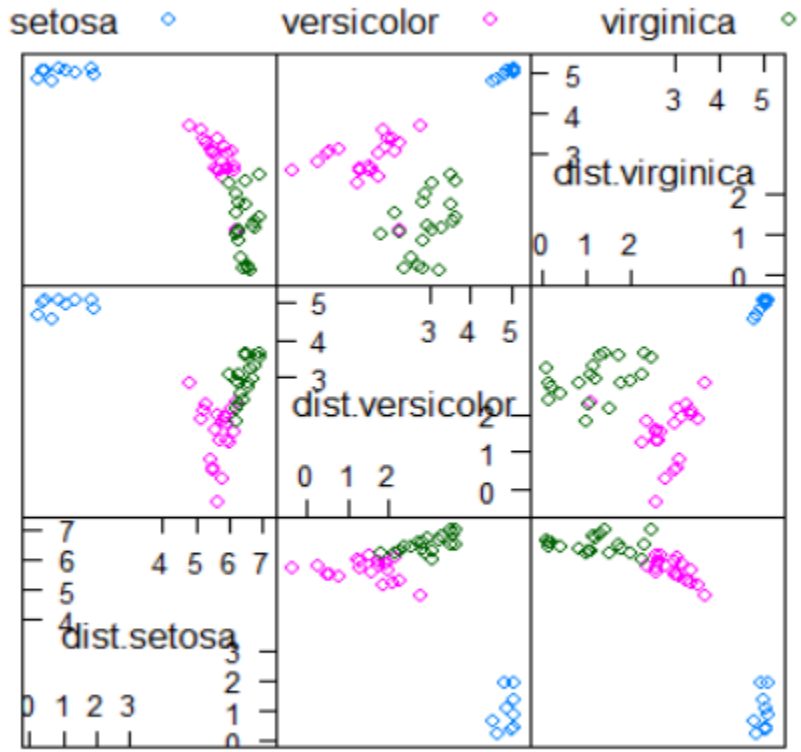
- predict.classDist() 함수는 군집 거리를 생성하는 데 사용

```
> # 훈련용 자료로부터 군집 중심과 공분산 행렬 계산
> trainSet <- sample(1:150, 100)
> distData <- classDist(iris[trainSet, 1:4], iris$Species[trainSet])
> distData$values
 $setosa
 $setosa$means
 Sepal.Length    Sepal.Width    Petal.Length    Petal.Width
     4.9682927      3.4121951       1.4512195     0.2463415

 $setosa$A
              Sepal.Length  Sepal.Width Petal.Length Petal.Width
 Sepal.Length   19.200711 -11.86012929  -1.59842385  -7.4263596
 Sepal.Width   -11.860129  13.94635609   0.01871781   0.5196262
 Petal.Length   -1.598424   0.01871781  38.55336083 -18.8187399
 Petal.Width    -7.426360   0.51962616 -18.81873988  95.5437185
```

```
> # 제외된(검증용) 자료에 대해 군집 중심까지의 마할라노비스 거리 계산
> newDist <- predict(distData, iris[-trainSet, 1:4])
> newDist
     dist.setosa dist.versicolor dist.virginica
 6     1.3503799       5.0292664     5.02013231
 11    0.8691851       5.0645744     5.11572643
 19    1.8969447       5.0655339     5.10526027
 ...
 148   6.3827146       2.5365787     0.44342511
 150   6.2645486       2.2601476     1.05286720
```

```
> # 제외된 자료에 대한 군집 거리의 산점도 행렬
> splom(newDist, groups = iris$Species[-trainSet],
        auto.key=list(columns=3))
```



산점도 행렬(scatter plot matrix)