기타감독학습

Jinseog Kim
Dongguk University
jskim1986@gmail.com

2018-05-02

Contents

1 기타 감독합습 모형

1.1 Naive Bayes 모형

 입력변수의 조건부 분포가 서로 독립 = 단순 베이즈 가정(naive Bayes assumption)

$$
\begin{aligned}
P(Y = k|X_1 = x_1, \ldots, X_p = x_p) &\propto P(X_1 = x_1, \ldots, X_p = x_p|Y = k)P(Y = k) \\
&\propto P(Y = k) \prod_{j=1}^{p} P(X_j = x_j|Y = k)
\end{aligned}
$$

1.2 k-근방 분류

- k-NN은 통계적 모형을 적합하지 않는 메모리 기반의 방법
- x점이 주어지면 x와 가장 거리가 가까운 k개의 훈련자료점의 y값들을 비교
- $N(x)$를 x와 거리가 가장 가까운 k개의 훈련자료들의 집합인 k-근방(k-nearest neighborhood)이라 하면

$$\hat{y}(x) = \arg \max_{l \in \{-1,+1\}} \sum_{x_j \in N(x)} I(y_j = l)$$

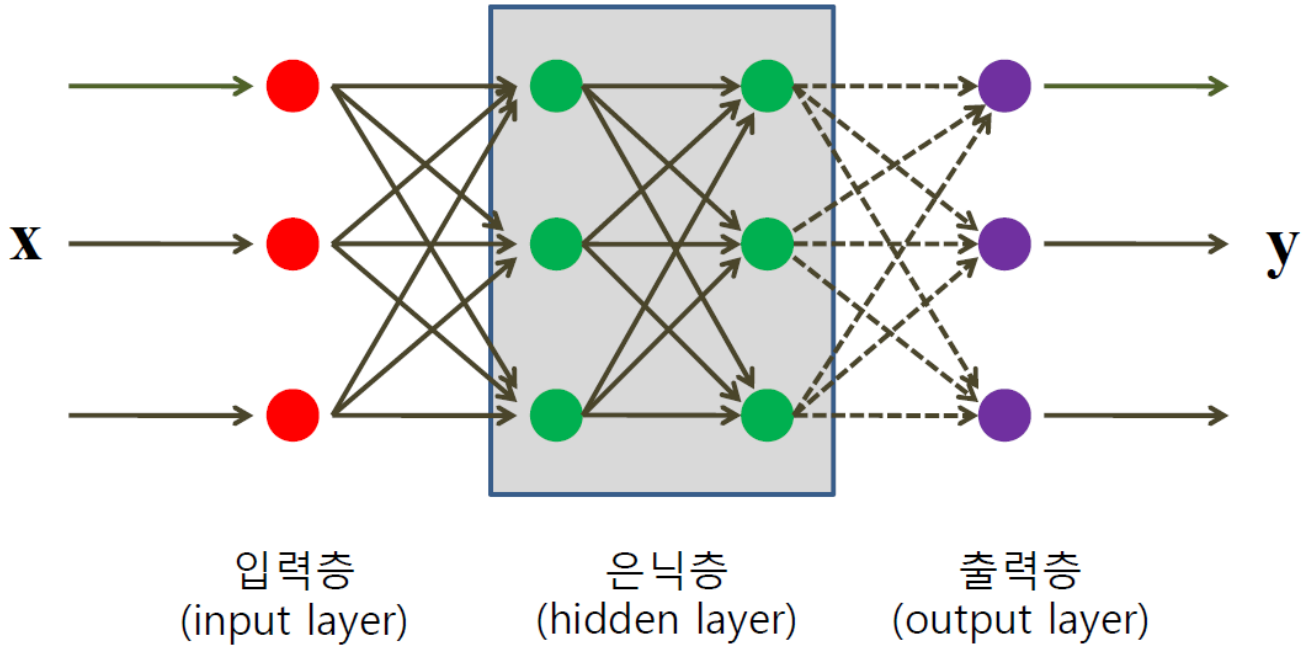
- k가 증가하면 편의는 커지고 분산은 감소한다.
- 점근적으로 1-NN의 오분류율은 최적 Bayes 오분류율의 2배를 넘지 않는다고 알려져 있다.

1.3 신경망 (Neural networks)

- 다층신경망

1. 입력층(input layer):
   - 입력변수에 대응되는 노드로 구성, 노드의 수는 입력변수의 개수
2. 은닉층(hidden layer):
   - 입력층으로부터 전달되는 변수값들의 선형결합을 비선형함수로 처리
   - 출력층 또는 다른 은닉층에 전달하는 역할
3. 출력층(output layer)
   - 출력변수에 대응되는 노드
   - 분류모형에서는 클래스의 수 만큼의 출력노드가 생성

 다층신경망 모형의 구조



1.4 SVM

- Hinge loss: $(1 - y_i f(x_i))_+ \triangleq \max(1 - y_i f(x_i), 0)$
- $\lambda > 0$: tunning parameter
- SVM은 아래의 penalized hinge loss를 최소화하는 방법

$$\min_{f \in \mathcal{H}} \left(\frac{1}{n} \sum_{i=1}^{n} (1 - y_i f(x_i))_+ + \lambda \|f\|^2 \right).$$

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x_i, x)$$

- f는 n개의 훈련자료점에서 계산된 커널의 선형결합

$$\min_{\beta, \beta_0} \left(\frac{1}{n} \sum_{i=1}^{n} (1 - y_i(\beta_0 + \Phi(x_i)^T \beta))_+ + \lambda \|\beta\|^2 \right)$$

2 R 예제

2.1 R 패키지 및 함수

R패키지 및 함수명	옵션	함수/옵션 설명
e1071::naiveBayes	laplace	Laplace smoothing(default=0)
class::knn	k	근방 관측치 수
nnet::nnet	size	hidden node의 수
	linout	switch for linear output units
	entropy	switch for entropy
	softmax	switch for softmax
	decay	parameter for weight decay(default=0)
	maxit	역전파 알고리즘의 반복 횟수를 지정하며 기본값은 100임
neuralnet::neuralnet	hidden	
	err.fct	error function(“sse”|“ce”=cross entroy)
	act.fct	activation function(“logistic”|“tanh”)
	linear.ouout	switch for linear output units
	algorithm	learning algorithm(“backprop”|rprop+|…)
e1071::svm	kernel	“linear”|“polynomial”|“radial basis”
	gamma	parameter for kernel function
	type	“C-classification”|“one-classification”

R패키지 및 함수명	옵션	함수/옵션 설명
		“nu-classification”
	cost	C constraint(regularized param: defaut=1)
	cross	K-fold CV(default=0)
dataset::iris		Edgar Anderson’s Iris Data
ElemStatLearn::spam		spam mail data

2.2 Naive Bayes 예제

1. 모형적합

```
library(e1071)
iris_cat <- iris
# 입력변수를 범주화
iris_cat$Sepal.Length <- cut(iris$Sepal.Length, c(-Inf, median(iris$Sepal.Length), Inf))
iris_cat$Sepal.Width <- cut(iris$Sepal.Width, c(-Inf, median(iris$Sepal.Width), Inf))
iris_cat$Petal.Length <- cut(iris$Petal.Length, c(-Inf, median(iris$Petal.Length), Inf))
iris_cat$Petal.Width <- cut(iris$Petal.Width, c(-Inf, median(iris$Petal.Width), Inf))
model <- naiveBayes(Species ~ ., data = iris_cat)
model
```

```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
```

```
    setosa versicolor  virginica
 0.3333333  0.3333333  0.3333333

Conditional probabilities:
            Sepal.Length
Y            (-Inf,5.8] (5.8, Inf]
  setosa           1.00       0.00
  versicolor       0.48       0.52
  virginica        0.12       0.88

            Sepal.Width
Y            (-Inf,3] (3, Inf]
  setosa         0.16     0.84
  versicolor     0.84     0.16
  virginica      0.66     0.34

            Petal.Length
Y            (-Inf,4.35] (4.35, Inf]
  setosa             1.0         0.0
  versicolor         0.5         0.5
  virginica          0.0         1.0
```

```
             Petal.Width
Y             (-Inf,1.3] (1.3, Inf]
  setosa            1.00       0.00
  versicolor        0.56       0.44
  virginica         0.00       1.00
```

2. 예측

```r
predict(model, iris_cat[1:5, -5], type = "raw") # class probabilities
```

```
        setosa versicolor    virginica
[1,] 0.9750390  0.0249610 4.735903e-08
[2,] 0.5863038  0.4136959 2.902204e-07
[3,] 0.9750390  0.0249610 4.735903e-08
[4,] 0.9750390  0.0249610 4.735903e-08
[5,] 0.9750390  0.0249610 4.735903e-08
```

```r
pred <- predict(model, iris_cat[,-5])
```

3. 모형평가

```r
# Confusion Matrix
table(pred, iris$Species)
```

```
pred         setosa versicolor virginica
  setosa         50         19         0
  versicolor      0         13         6
  virginica       0         18        44
```

```r
# using laplace smoothing:
model2 <- naiveBayes(Species ~ ., data = iris, laplace = 3)
pred <- predict(model, iris[,-5])
table(pred, iris$Species)
```

```
pred         setosa versicolor virginica
  setosa         50         50        50
  versicolor      0          0         0
  virginica       0          0         0
```

```r
library(e1071)
# 훈련자료. 검증자료로 data 분할
set.seed(12345) # 랜덤 seed
# 훈련자료의 인덱스 추출(비복원 추출), 추출되지 않은 나머지는 검증자료의 인덱스
tr.idx <- sample(nrow(iris), nrow(iris)*0.5, replace=F)
head(tr.idx)
```

```
[1] 109 131 113 149  67  25
```

```r
iris2 <- iris
cut_values <- apply(iris2[, -5], 2, quantile)
# 입력변수를 범주형 변수로 변환 (여기서는 4분위수 이용)
for(i in 1:4) iris2[, i] <- cut(iris2[, i], cut_values[,i])
head(iris2)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1    (4.3,5.1]   (3.3,4.4]      (1,1.6]   (0.1,0.3]  setosa
2    (4.3,5.1]     (2.8,3]      (1,1.6]   (0.1,0.3]  setosa
3    (4.3,5.1]     (3,3.3]      (1,1.6]   (0.1,0.3]  setosa
4    (4.3,5.1]     (3,3.3]      (1,1.6]   (0.1,0.3]  setosa
5    (4.3,5.1]   (3.3,4.4]      (1,1.6]   (0.1,0.3]  setosa
```

```
6    (5.1,5.8]   (3.3,4.4]   (1.6,4.35]   (0.3,1.3]  setosa
```

 Naive Bayes modelling

```
model <- naiveBayes(Species ~ ., data = iris2[tr.idx, ])
```

 predict

```
# predict
# type = "raw"이면 사후확률 추정, 아니면 클래스범주 출력
predict(model, iris2[1:5, -5], type = "raw")
```

```
     setosa   versicolor    virginica
[1,]      1 2.556303e-12 4.445744e-10
[2,]      1 5.668323e-09 8.502484e-09
[3,]      1 1.079681e-09 1.619521e-09
[4,]      1 1.079681e-09 1.619521e-09
[5,]      1 2.556303e-12 4.445744e-10
```

```
# 검증자료에 대해 예측
pred <- predict(model, iris2[-tr.idx,-5])
```

```r
# Confusion Matrix
o <- table(pred, iris2$Species[-tr.idx])
o
```

```
pred         setosa versicolor virginica
  setosa         23          4         0
  versicolor      0         19         5
  virginica       0          4        20
```

```r
# Misclasification Error rate
1- sum(diag(o))/sum(o)
```

```
[1] 0.1733333
```

```r
# using laplace smoothing:
model2 <- naiveBayes(Species ~ ., data = iris2[tr.idx, ], laplace = 3)
pred2 <- predict(model2, iris2[-tr.idx, -5])
table(pred2, iris2$Species[-tr.idx])
```

```
pred2        setosa versicolor virginica
  setosa         23          1         0
  versicolor      0         22         8
  virginica       0          4        17
```

2.3 k NN 예제

1. 모형적합

```
library(class)
set.seed(1)
y <- iris[,5]
# Partition indice into training data & test data
tr.idx <- sample(length(y), 75)
```

2. 예측

```
pred <- knn(train=iris[tr.idx, -5], test=iris[-tr.idx, -5], cl=y[tr.idx], k = 3)
```

3. 모형 평가

```
table(pred, y[-tr.idx])
```

```
pred         setosa versicolor virginica
  setosa         24          0         0
  versicolor      0         22         3
  virginica       0          0        26
```

2.4 neural network 예제: nnet

1. fit

```
library(nnet)
ir1 <- nnet(Species~., data=iris[tr.idx,], size = 2, decay = 5e-4)
```

```
# weights:  19
initial  value 98.070522
iter  10 value 49.984837
iter  20 value 33.254723
iter  30 value 4.667817
iter  40 value 3.787903
iter  50 value 3.701295
iter  60 value 3.672516
iter  70 value 3.670662
iter  80 value 3.666205
iter  90 value 3.664552
iter 100 value 3.662954
final  value 3.662954
stopped after 100 iterations
```

```
summary(ir1)
```

```
a 4-2-3 network with 19 weights
options were - softmax modelling  decay=5e-04
 b->h1 i1->h1 i2->h1 i3->h1 i4->h1
  4.82   0.17   1.73  -1.43  -2.47
 b->h2 i1->h2 i2->h2 i3->h2 i4->h2
  0.42   0.68   1.79  -3.12  -1.42
 b->o1 h1->o1 h2->o1
 -4.75   1.80   9.45
 b->o2 h1->o2 h2->o2
 -1.43   8.42  -8.72
 b->o3 h1->o3 h2->o3
  6.12 -10.13  -0.80
```

2. predict

```
pred <- predict(ir1, iris[-tr.idx, -5], type = "class")
```

3. 평가 : confusion matrix

```r
table(iris$Species[-tr.idx], pred)
```

```
            pred
             setosa versicolor virginica
  setosa         24          0         0
  versicolor      0         20         2
  virginica       0          1        28
```

2.5 Support Vector Machine : Spam Data

1. data

```
#install.packages("ElemStatLearn")
data(spam, package = "ElemStatLearn")
```

```
str(spam)
### 'data.frame':   4601 obs. of  58 variables:
###  $ A.1 : num  0 0.21 0.06 0 0 0 0 0 0.15 0.06 ...
###  $ A.2 : num  0.64 0.28 0 0 0 0 0 0 0 0.12 ...
### ...
###  $ A.57: int  278 1028 2259 191 191 54 112 49 1257 749 ...
###  $ spam: Factor w/ 2 levels "email","spam": 2 2 2 2 2 2 2 2 2 2 ...
```

2. parmeter tunning & fitting

```r
# sampling index of train data
tr.idx <- sample(nrow(spam), 0.7*nrow(spam))
# tunning parameters
# tuning
obj <- tune(svm, spam ~ ., data = spam[tr.idx,],
            ranges = list(cost = c(0.25, 0.35)))
summary(obj)
```

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 cost
 0.35

- best performance: 0.07888199

- Detailed performance results:
```

```
  cost      error dispersion
1 0.25 0.08385093 0.01603720
2 0.35 0.07888199 0.01598365
```

```r
# fit svm model
model <- svm(spam ~., data = spam[tr.idx,], cost=obj$best.parameters$cost)
summary(model) # coefs
```

```

Call:
svm(formula = spam ~ ., data = spam[tr.idx, ], cost = obj$best.parameters$cost)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  0.35
      gamma:  0.01754386

Number of Support Vectors:  1146

 ( 550 596 )
```

```
Number of Classes:  2

Levels:
 email spam
```

4. prediction & performance evaluation

```r
pred <- predict(model, newdata=spam[-tr.idx,])
table(spam$spam[-tr.idx], pred)
```

```
       pred
        email spam
  email   808   37
  spam     69  467
```