기계학습

목차

Naive Bayes

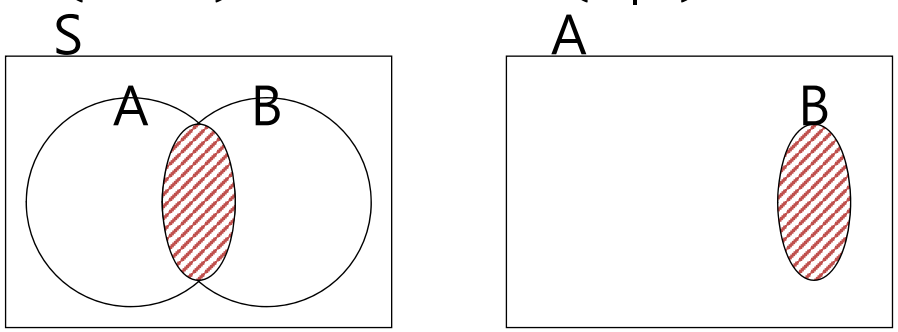
- 확률에 기반한 classification 알고리즘
- 사용 분야
  - 정크 이메일 필터링과 같은 문서 분류, 저자 식별, 주제 식별
  - 네트워크 침입 탐지, 네트워크 이상 행동 탐지
  - 관찰된 증상을 고려한 질병 진찰

- 결과의 확률을 추정하기 위해 많은 속성을 고려해야 하는 문제에 적합
  - 많은 알고리즘이 약한 효과를 내는 속성을 무시하는 반면, 베이즈 기법은 예측을 예민하게 변경하는 모든 증거(속성)을 분류에 활용
  - 다수의 속성이 상대적으로 매우 미비한 효과를 가진다 하더라도 그 영향을 합하면 꽤 큰 효과를 기대할 수 있음

베이즈 정리

- **조건부 확률**
  - 어떤 한 사상 A가 이 발생하였다는 조건에서 다른 사상 B가 발생할 확률

$P(A \cap B)$ $P(B|A)$



$$P(B|A) = \frac{P(A \cap B)/n(S)}{n(A)/n(S)} = \frac{P(A \cap B)}{P(A)}$$

- **조건부 확률의 성질**

$$P(\emptyset|A) = 0$$

$$P(B^c|A) = 1 - P(B|A)$$

$$P[(A \cup B)|C] = P(A|C) + P(B|C) - P[(A \cap B)|C]$$

예제 2.1

- 어느 고등학교 학생 100명을 대상으로 헌혈을 한 경험이 있는지 조사하였더니 다음 표와 같았다. 이 중에서 여학생 한 명을 선택했을 때, 이 학생이 헌혈을 한 적이 있는 학생일 확률은?
  - A : 선택된 학생이 여학생 사상
  - B : 선택된 학생이 헌혈한 적이 있을 사상
  - $A \cap B$: 선택된 학생이 여학생이고 헌혈한 적이 있을 사상

성별 \ 헌혈	헌혈을 한 적이 있다	헌혈을 한 적이 없다	합계
남학생	32	20	52
여학생	25	23	48
합계	57	43	100

$$P(A) = \frac{n(A)}{n(S)} = \frac{48}{100} = 0.48$$

$$P(A \cap B) = \frac{n(A \cap B)}{n(S)} = \frac{25}{100} = 0.25$$

$$P(B|A) = \frac{P(A \cap B)}{P(A)} = \frac{0.25}{0.48} = 0.52$$

- **곱셈법칙**

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

$$P(A \cap B) = P(A)P(B|A)$$

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

$$P(A \cap B) = P(B)P(A|B)$$

$$P(A \cap B) = P(A)P(B|A) = P(B)P(A|B)$$

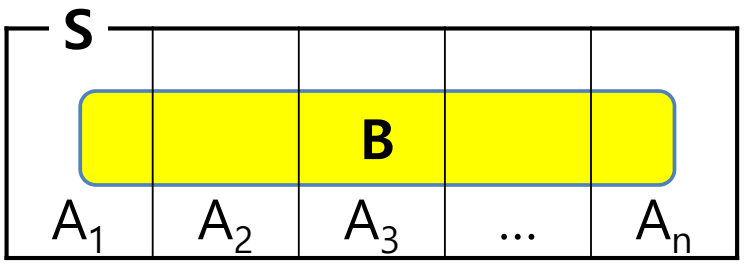
- **영국 철학자 Thomas Bayes의 이름에서 유래된 정리로 사건 A의 확률을 새로운 정보 B에 의해 Update하는 방법**
  - **즉, Bayes Theorem은 Pr(A|B)을 구하는데 유용한 정리**
  - **Update하기 전 사건 A 확률[Pr(A)]을 사전적 확률(Prior Probability)**
  - **Update 한 후에 구해진 확률[Pr(A|B)]을 사후적 확률(Posterior Probability)**

- 표본공간 S에 대해서 다음 조건을 만족시키는 사상 A_1, A_2,… ,A_n을 S의 분할

$$A_i \neq \phi$$

$$S = \bigcup_{i=1}^{n} A_i$$

$$A_i \cap A_j = \emptyset, i \neq j$$



- $P(B|A_k)$뿐만 아니라 $P(A_k|B)$과 같은 조건부 확률에 관심

$$P(A_k|B) = \frac{P(A_k \cap B)}{P(B)}$$

  - 여기에서 $P(B)$는 $P(A_1 \cap B)$, $P(A_2 \cap B)$, …, $P(A_n \cap B)$의 합사상

$$P(B) = P(A_1 \cap B) + P(A_2 \cap B) + \cdots + P(A_n \cap B)$$

  - 또한 $P(A_i \cap B)$는 조건부 확률의 곱셈법칙을 이용

$$P(A_i \cap B) = P(A_i)P(B|A_i)$$

- **베이즈 정리**

$$P(A_k|B) = \frac{P(A_k \cap B)}{P(B)} = \frac{P(A_k)P(B|A_k)}{\sum_{i=1}^{n} P(A_i)P(B|A_i)}$$

- **전확률의 법칙**

$$P(B) = \sum_{i=1}^{n} P(A_i)P(B|A_i)$$

예제 2.2

- **두 개의 항아리가 있다.**
  - **항아리1 : 노란 구슬 2개와 빨간 구슬 4개**
  - **항아리2 : 노란 구슬 3개와 빨간 구슬 5개**
- **항아리2에서 하나의 구슬을 꺼내어 항아리1에 넣고 항아리1에서 하나의 구슬을 꺼낼 때 노란 구슬이 나올 확률은?**
  - **Y : 항아리1에서 꺼낸 구슬이 노란색일 사상**
  - **A : 항아리2에서 꺼낸 구슬이 노란색일 사상**
  - **B : 항아리2에서 꺼낸 구슬이 빨간색일 사상**

$$P(A) = \frac{3}{8} \quad P(B) = \frac{5}{8}$$

- 노란 구슬을 첫 번째 항아리에 넣고 첫 번째 항아리에서 노란 구슬이 나올 확률

$$P(Y|A) = \frac{3}{7}$$

- 빨간 구슬을 첫 번째 항아리에 넣고 첫 번째 항아리에서 노란 구슬이 나올 확률

$$P(Y|B) = \frac{2}{7}$$

- 따라서 전확률의 법칙에 의해 첫 번째 항아리에서 노랑 구슬을 뽑을 확률

$$P(Y) = P(A)P(Y|A) + P(B)P(Y|B)$$
$$= \frac{3}{8} \times \frac{3}{7} + \frac{5}{8} \times \frac{2}{7} = \frac{19}{56}$$

나이브 베이즈안 분류법

- 나이브 베이지안(Naive Bayesian) 분류법이란 속성변수들과 범주변수가 확률분포를 따른다고 간주하여 베이즈 정리와 조건부 독립성을 활용한 분류기법
- 속성변수들이 범주형일 때 주로 사용되나, 연속형인 경우에도 확률분포의 형태를 가정하여 사용 가능

- 어떤 객체 x에 대한 범주를 Y
  - x가 주어질 때 Y의 조건부 확률분포는 베이즈 정리에 의해($\mathbf{x}=(x_1,x_2,\cdots,x_p)^T$)

$$f(y|\mathbf{x}) \propto f(y)f(\mathbf{x}|y) \quad , \quad y = 1,2,\cdots,J$$

    - 여기서, f(y)는 Y의 사전분포(prior), f(y|x)는 x의 관측 후 사후분포(posterior)
- 나이브 베이지안 분류법은 이 사후분포를 산출하여 가장 큰 값을 갖는 범주를 객체 x에 부여하는 것
- 사후분포를 정확한 계산을 위해서 비례상수가 필요하나 범주를 부여하기 위해서는 오른쪽의 두 항을 산출하면 가능
  (모든 범주 Y에 대해 P(x)는 공통)

- f(x|y)는 속성변수들의 조건부 결합확률분포
  - 나이브 베이지안에서는 조건부 독립성을 이용하여 이를 평가
  - 확률변수 X_1, X_2, Y에 대하여 다음이 성립하면 Y가 주어질 때 X_1과 X_2 가 조건부 독립

$$P(X_1 = x_1|X_2, Y) = P(X_1 = x_1|Y)$$

- 따라서 속성변수들의 범주가 주어질 때(동일한 범주 내에서) 서로 독립이라 가정하면

$$f(y|\mathbf{x}) \propto f(y)\prod_{i=1}^{p} f_{X_i}(x_i|y)$$

  - 사후확률(또는 비례항)을 평가하기 위해 학습표본으로부터 Y의 사전확률 및 각 속성변수별 분포를 추정

- Y의 사전확률 및 각 속성변수별 분포
  - f(y)=범주 y의 상대도수, y=1,2,…,J
  - $f_{x_i}(x_i|y)$=범주 y인 객체 중에서 $X_i=x_i$인 객체들의 비율
- 사후확률이 산출된 후에는 가장 큰 값을 갖는 범주를 선택하여 객체 x에 부여

예제 2.3

- 9명의 고객에 대한 학습표본이 있다.
  나이브 베이지안 분류법을 적용하여 분류하시오.

고객	1	2	3	4	5	6	7	8	9
성별 (X_1)	남	남	남	남	여	여	여	여	여
나이 (X_2)	20대	20대	30대	40대	10대	20대	20대	30대	40대
범주 (Y)	구매	비구매	구매	구매	구매	비구매	구매	비구매	비구매

- Y의 사전분포

$$f(\text{구매}) = \frac{5}{9} \quad , \quad f(\text{비구매}) = \frac{4}{9}$$

- **변수별 조건부 확률분포**

$$f(\text{남자}|\text{구매}) = \frac{3}{5} \quad , \quad f(\text{여자}|\text{구매}) = \frac{2}{5}$$

$$f(\text{남자}|\text{비구매}) = \frac{1}{4} \quad , \quad f(\text{여자}|\text{비구매}) = \frac{3}{4}$$

$$f(10\text{대}|\text{구매}) = \frac{1}{5} \quad , \quad f(20\text{대}|\text{구매}) = \frac{2}{5}$$

$$f(30\text{대}|\text{구매}) = \frac{1}{5} \quad , \quad f(40\text{대}|\text{구매}) = \frac{1}{5}$$

$$f(10\text{대}|\text{비구매}) = 0 \quad , \quad f(20\text{대}|\text{비구매}) = \frac{2}{4}$$

$$f(30\text{대}|\text{비구매}) = \frac{1}{4} \quad , \quad f(40\text{대}|\text{비구매}) = \frac{1}{4}$$

- **고객 1의 범주별 사후확률**

$$f(\text{구매}|X_1 = \text{남자}, X_2 = 20\text{대})$$
$$= f(\text{구매})f(\text{남자}|\text{구매})f(20\text{대}|\text{구매})$$
$$= \frac{5}{9} \times \frac{3}{5} \times \frac{2}{5} = \frac{2}{15}$$

$$f(\text{비구매}|X_1 = \text{남자}, X_2 = 20\text{대})$$
$$= f(\text{비구매})f(\text{남자}|\text{비구매})f(20\text{대}|\text{비구매})$$
$$= \frac{4}{9} \times \frac{1}{4} \times \frac{2}{4} = \frac{1}{18}$$

  - **구매에 대한 사후확률이 더 크므로 고객 1의 범주는 구매**

- **결과**

고객	1	2	3	4	5	6	7	8	9
성별 (X_1)	남	남	남	남	여	여	여	여	여
나이 (X_2)	20대	20대	30대	40대	10대	20대	20대	30대	40대
범주 (Y)	구매	비구매	구매	구매	구매	비구매	구매	비구매	비구매
f(구매|X_1,X_2)	2/15	2/15	1/15	1/15	2/45	4/45	4/45	2/45	2/45
f(비구매|X_1,X_2)	1/18	1/18	1/36	1/36	0	1/6	1/6	1/12	1/12
결과	구매	구매	구매	구매	구매	비구매	비구매	비구매	비구매

- **남자이고 10대인 새로운 고객 범주**

$$\mathrm{f}(\text{구매}|\mathrm{X}_1 = \text{남자}, \mathrm{X}_2 = 10\text{대})$$
$$= \mathrm{f}(\text{구매})\mathrm{f}(\text{남자}|\text{구매})\mathrm{f}(10\text{대}|\text{구매})$$
$$= \frac{5}{9} \times \frac{3}{5} \times \frac{1}{5} = \frac{1}{15}$$

$$\mathrm{f}(\text{비구매}|\mathrm{X}_1 = \text{남자}, \mathrm{X}_2 = 20\text{대})$$
$$= \mathrm{f}(\text{비구매})\mathrm{f}(\text{남자}|\text{비구매})\mathrm{f}(20\text{대}|\text{비구매})$$
$$= \frac{4}{9} \times \frac{1}{4} \times 0 = 0$$

  - **구매에 대한 사후확률이 더 크므로 새고객의 범주는 구매**

- 장점
  - 단순하고 빠르며 효과적
  - 노이즈와 결측 데이터가 있어도 잘 수행
  - 훈련에 대해 상대적으로 적은 예제가 필요(많은 예제에서도 잘 수행)
  - 예측에 대한 추정된 확률을 얻기 쉽움
- 단점
  - 모든 속성은 동등하게 중요하고 독립적이라는 알려진 결함(open-faulty assumption)
  - 연속형 수치 속성으로 구성된 데이터셋에 대해 이상적이지 않음(discretization 필요)
  - 추정된 확률은 예측된 범주 보다 덜 신뢰적

- 라플라스 추정값(Laplas estimator)

$$\mathbf{x} = (x_1, x_2, \cdots, x_{p-1}, \text{None})^T$$

  - 위의 경우와 같이 훈련 데이터에 없는 경우가 테스트 케이스에 포함 되어 있다면

$$f(\text{구매}|X_1 = \text{남자}, X_2 = 50\text{대})$$
$$= f(\text{구매})f(\text{남자}|\text{구매})f(50\text{대}|\text{구매})$$
$$= \frac{5}{9} \times \frac{3}{5} \times 0 = 0$$

$$f(\text{비구매}|X_1 = \text{남자}, X_2 = 50\text{대})$$
$$= f(\text{비구매})f(\text{남자}|\text{비구매})f(50\text{대}|\text{비구매})$$
$$= \frac{4}{9} \times \frac{1}{4} \times 0 = 0$$

- 나이를 제외한 나머지 정보를 가지고도 분류가 가능한데 0이 곱해져서 다른 정보도 쓸모없게 됨

– 이를 방지하기 위해 확률이 0 이 되지 않도록 임의의 값(대부분 1 을 사용)을 부여하여 계산
– 라플라스 추정값은 각 속성별로 서로 다른 값을 부여할 수 도 있음
– 속성의 중요도에 따라 서로 다른 값을 부여할 수 도 있음
– 훈련 데이터셋이 충분히 크다면 라플라스 추정값은 고려하지 않아도 됨

- 대부분의 분류 알고리즘들은 모델의 성능에 영향을 미치는 parameter를 가지고 있다.
  – KNN의 k
  – Naive Bayes의 경우는 Laplas estimator
- 이와 같은 parameter를 조율 모수(tuning parameter)
- 학습 모델을 수립시 최적의 조율 모수를 찾는 것이 과제 중 하나

- **나이브 베이즈안 분류(e1071 패키지)**

```
> naiveBayes(formula, data, laplace=0)
> naiveBayes(x, y, laplace=0)
```

  - **formula : 모형식**
    - **class ~ $x_1+x_2+\cdots$**
  - **data : 분석 데이터(data frame)**
  - **x : 학습 데이터(data frame)**
  - **y : 집단 Label 벡터(factor type)**
  - **laplace : Laplace 평활화**

예제 2.4

- **다음 붓꽃(iris.csv)의 종(Species)을 나이브 베이지안 방법을 이용하여 분류하시오.**
  - **종 : setosa, versicolor, virginica**

```
> library(e1071)
> iris.data<-read.csv("iris.csv")
> head(iris.data)
  Sepal.Length Sepal.Width Petal.Length
Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> # 나이브 베이지안 분류
> model<-naiveBayes(iris[,1:4], iris$Species)
> model<-naiveBayes(Species~., data=iris.data)
```

```
> # 분류 예측
> pr<-predict(model, newdata=iris[,1:4])
>
> table(pr, iris$Species) # 분류표

pr             setosa versicolor virginica
  setosa           50          0         0
  versicolor        0         47         3
  virginica         0          3        47
> acc<-mean(pr==iris$Species)   # 분류율
> acc
[1] 0.96
```

```
> # 적합 모형을 이용한 새로운 데이터 분류
> x1<-sample(iris$Sepal.Length, size=10)
> x2<-sample(iris$Sepal.Width, size=10)
> x3<-sample(iris$Petal.Length, size=10)
> x4<-sample(iris$Petal.Width, size=10)
> n.data<-data.frame(Sepal.Length=x1,
+                    Sepal.Width=x2, Petal.Length=x3,
+                    Petal.Width=x4)
> # 새로운 자료의 분류 예측
> new.pr<-predict(model, newdata=n.data)
> new.pr
 [1] versicolor setosa virginica  versicolor versicolor
 [6] versicolor setosa versicolor versicolor virginica
Levels: setosa versicolor virginica
```

예제 2.5

- **다음은 스팸메일을 분류하기 위한 자료 이다. 나이브 베이지안 분류를 통하여 분류하시오.**

	A	B
1	type	text
2	ham	Hope you are having a good week. Just checking in
3	ham	K..give back my thanks.
4	ham	Am also doing in cbe only. But have to pay.
5	spam	complimentary 4 STAR Ibiza Holiday or FR10,000 cash needs your URGENT collection. 09066364349 NOW from Landline
6	spam	okmail: Dear Dave this is your final notice to collect your 4* Tenerife Holiday or #5000 CASH award! Call 09061743806 fr
7	ham	Aiya we discuss later lar... Pick u up at 4 is it?
8	ham	Are you this much buzy
9	ham	Please ask mummy to call father
10	spam	Marvel Mobile Play the official Ultimate Spider-man game (FR4.50) on ur mobile right now. Text SPIDER to 83338 for the
11	ham	fyi I'm at usf now, swing by the room whenever
12	ham	Sure thing big man. i have hockey elections at 6, shouldn't go on longer than an hour though
13	ham	I anything lor...
14	ham	By march ending, i should be ready. But will call you for sure. The problem is that my capital never complete. How far wi
15	ham	Hmm well, night night
16	ham	K I'll be sure to get up before noon and see what's what
17	ham	Ha ha cool cool chikku chikku:-):-DB-)
18	ham	Darren was saying dat if u meeting da ge den we dun meet 4 dinner. Cos later u leave xy will feel awkward. Den u meet
19	ham	He dint tell anything. He is angry on me that why you told to abi.

```
> sms_raw<-read.csv("sms_spam.csv",
+                                 stringsAsFactors=FALSE)
> head(sms_raw)
  type
1  ham
2  ham
3  ham
4 spam
5 spam
6  ham

text
1  Hope you are having a good week. Just checking in
2  K..give back my thanks.
:        :
```

```
> # sms 데이터 구조
> dim(sms_raw)
[1] 5559    2
> str(sms_raw)
'data.frame':        5559 obs. of  2 variables:
 $ type: chr  "ham" "ham" "ham" "spam" ...
 $ text: chr  "Hope you are having a good week. Just
checking in" "K..give back my thanks." "Am also doing in
cbe only. But have to pay." "complimentary 4 STAR Ibiza
Holiday or 짙10,000 cash needs your URGENT collection.
09066364349 NOW from Landline"| __truncated__ ...
> # factor 유형으로 spam/ham으로 변환
> sms_raw$type<-factor(sms_raw$type)
```

```
> # 변수형 확인
> str(sms_raw$type)
 Factor w/ 2 levels "ham","spam": 1 1 1 2 2 1 1 1 2 1 ...
> table(sms_raw$type)

 ham spam
4812  747
> # 텍스트 마이닝(tm) 패키지가 처리할 수 있는
> # 말뭉치(corpus) 생성
> # corpus는 tm에서 문서를 관리하는 기본 구조(텍스트 문서들의 집합)
> library(tm)
Loading required package: NLP
Warning message:
package 'tm' was built under R version 3.5.1
```

```
> sms_corpus<-Corpus(VectorSource(sms_raw$text))
> sms_corpus
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 5559> # sms 말뭉치(corpus) 확인
> inspect(sms_corpus)
:        :
> sms_corpus[[1]]$content
[1] "Hope you are having a good week. Just checking in"
> sms_corpus[[2]]$content
[1] "K..give back my thanks."
>
> # tm_map() 사용하여 말뭉치 정리
> # 대문자는 소문자로
> corpus_clean<-tm_map(sms_corpus, tolower)
```

```
> # 숫자 제거
> corpus_clean<-tm_map(corpus_clean,
+                                 removeNumbers)
>
> # 기본 불용어 및 추가 불용어(and, but, not)제거
> sword2<-c(stopwords(),"and","but","not")
> corpus_clean<-tm_map(corpus_clean,
+                                 removeWords, sword2)
> # 마침표, 콤마, 세미콜론, 콜론 등 제거
> corpus_clean<-tm_map(corpus_clean,
+                                 removePunctuation)
>
> # 여러 개의 공백을 하나의 공백으로 변환
> corpus_clean<-tm_map(corpus_clean,
+                                 stripWhitespace)
```

```
> # 말뭉치 정리 확인
> inspect(sms_corpus[1:3])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 3

[1] Hope you are having a good week. Just checking in
K..give back my thanks.
[3] Am also doing in cbe only. But have to pay.
> inspect(corpus_clean[1:3])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 3

[1] hope good week just checking  kgive back thanks
also cbe pay
```

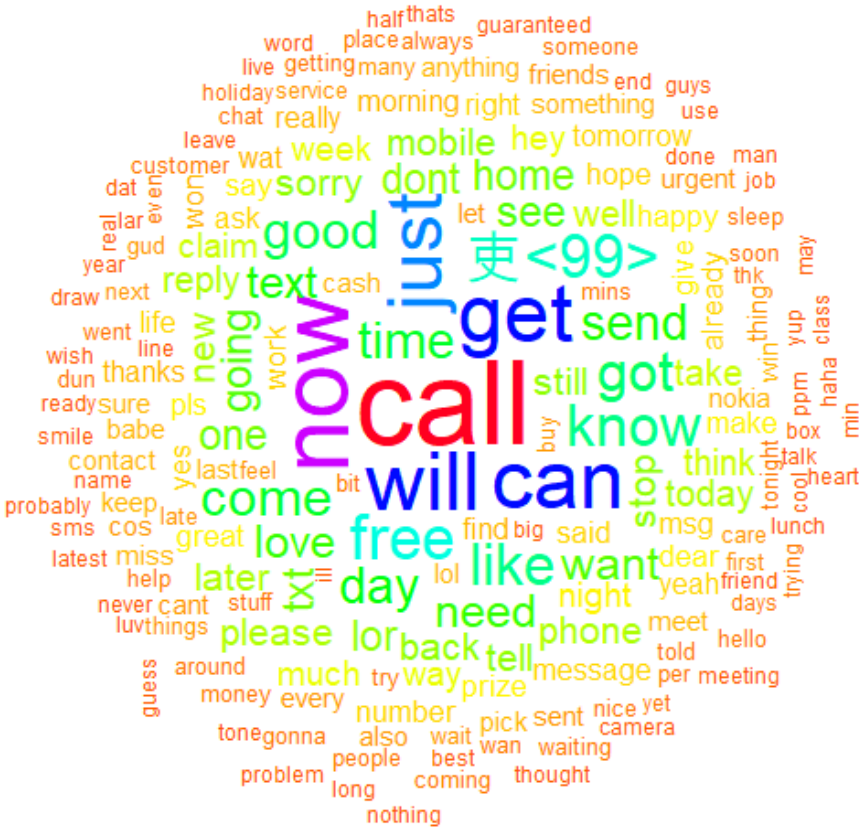
```
> sms_corpus[[1]]$content
[1] "Hope you are having a good week. Just checking
in"
> corpus_clean[[1]]$content
[1] "hope good week just checking "
> sms_corpus[[2]]$content
[1] "K..give back my thanks."
> corpus_clean[[2]]$content
[1] "kgive back thanks"
> sms_corpus[[3]]$content
[1] "Am also doing in cbe only. But have to pay."
> corpus_clean[[3]]$content
[1] " also cbe pay"
```

```
> # tm 패키지가 분석할 수 있는
> # Term-Document 형식의 행렬로 변환
> # 문서-용어 희소 매트릭스 생성
> sms_dtm<-DocumentTermMatrix(corpus_clean)
> sms_dtm
<<DocumentTermMatrix (documents: 5559, terms:
7931)>>
Non-/sparse entries: 42881/44045548
Sparsity           : 100%
Maximal term length: 40
Weighting          : term frequency (tf)
```

```
> # Term-Document 형식의 행렬을
> #위해서는 행렬로 변환
> as.matrix(sms_dtm)[1:10,15:20]
 Terms
Docs complimentary holiday ibiza landline lose needs
  1             0       0     0        0    0     0
  2             0       0     0        0    0     0
  3             0       0     0        0    0     0
  4             1       1     1        1    1     1
  5             0       1     0        1    0     0
  6             0       0     0        0    0     0
  7             0       0     0        0    0     0
  8             0       0     0        0    0     0
  9             0       0     0        0    0     0
  10            0       0     0        0    0     0
```

```
> # 훈련과 테스트 데이터셋 생성
> sms_raw_train<-sms_raw[1:4169, ]
> sms_raw_test<-sms_raw[4170:5559, ]
> sms_dtm_train<-sms_dtm[1:4169, ]
> sms_dtm_test<-sms_dtm[4170:5559, ]
> sms_corpus_train<-corpus_clean[1:4169]
> sms_corpus_test<-corpus_clean[4170:5559]
> # 스팸 비율 확인
> prop.table(table(sms_raw_train$type))

      ham      spam
0.8647158 0.1352842
> prop.table(table(sms_raw_test$type))

      ham      spam
0.8683453 0.1316547
```

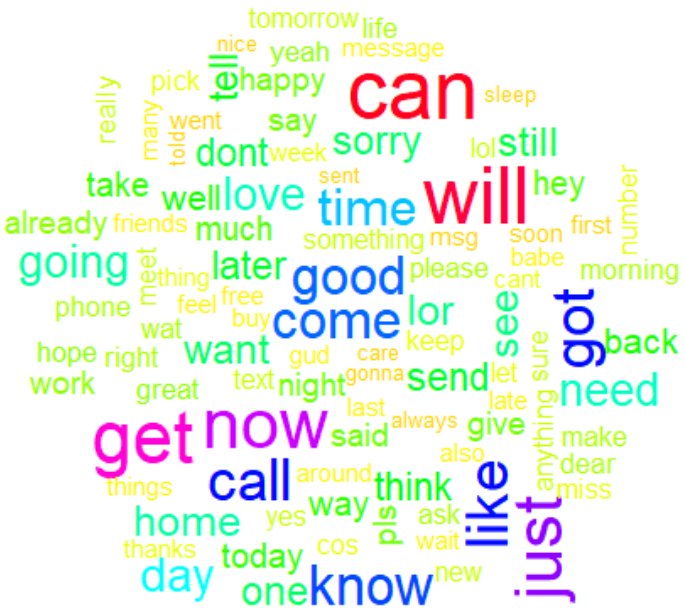
```
> # 워드클라우드 시각화
> library(wordcloud)
> # 훈련데이터 워드클라우드
> wordcloud(sms_corpus_train, min.freq=30,
+                       colors=rainbow(50),
+                       random.order=FALSE)
> # 학습자료 단어 빈도표
> sms_dtm_train<-
+               DocumentTermMatrix(sms_corpus_train)
> m_train<-as.matrix(sms_dtm_train)
> freq.train<-sort(colSums(m_train),decreasing=T)
> freq.train[1:10]
call  now  will  get  can  just  free 吏<99> like  know
453  367  305 299  299  266  219  210  193  192
```



```
> # 훈련 데이터를 스팸과 햄으로 구분
> spam<-subset(sms_corpus_train,
+                         sms_raw_train$type=="spam")
> ham<-subset(sms_corpus_train,
+                         sms_raw_train$type=="ham")
>
> # 스팸에서 나오는 단어
> wordcloud(spam, max.words=100,
+                   scale=c(3, 0.5), colors=rainbow(50))
> title(main="SPAM")
> # 햄에서 나오는 단어
> wordcloud(ham, max.words=100, scale=c(3, 0.5),
+                   colors=rainbow(50))
> title(main="HAM")
```

SPAM

HAM



```
> # 출현빈도 5회 이상 단어
> findFreqTerms(sms_dtm_train, 5)
  [1] "checking"       "good"           "hope"           "just"           "week"           "back"
  [7] "thanks"         "also"           "pay"            "cash"           "collection"     "complimentary"
 [13] "holiday"        "landline"       "lose"           "needs"          "now"            "urgent"
 [19] "更<99>"         "award"          "box"            "call"           "collect"        "dear"
 [25] "final"          "notice"         "ppm"            "sae"            "tcs"            "tenerife"
 [31] "lar"            "later"          "pick"           "much"           "ask"            "father"
 [37] "please"         "free"           "game"           "mobile"         "official"       "play"
 [43] "right"          "send"           "text"           "room"           "swing"          "usf"
 [49] "whenever"       "big"            "hour"           "longer"         "man"            "sure"
 [55] "thing"          "though"         "anything"       "lor"            "ending"         "far"
 [61] "march"          "never"          "problem"        "ready"          "will"           "work"
 [67] "hmm"            "night"          "well"           "get"            "noon"           "see"
 [73] "chikku"         "cool"           "cos"            "darren"         "dat"            "den"
 [79] "dinner"         "dun"            "feel"           "leave"          "lunch"          "meet"
 [85] "meeting"        "saying"         "angry"          "tell"           "told"           "come"
 [91] "din"            "wan"            "can"            "draw"           "every"          "gift"
 [97] "music"          "starting"       "tscs"           "txt"            "vouchers"       "win"
[103] "word"           "coming"         "goodnight"      "gym"            "birthday"       "today"
[109] "wish"           "gud"            "reading"        "contact"        "cost"           "joke"
[115] "less"           "one"            "ones"           "school"         "sent"           "think"
  :
[985] "water"          "tonight"        "freephone"      "drug"           "regards"        "recently"
[991] "south"          "hard"           "workin"         "vikky"          "lmao"           "bother"
[997] "medical"        "track"          "glad"           "exam"
[ reached getOption("max.print") -- omitted 217 entries ]
```

```
> # 출현빈도 5회 이상 단어
> sms_dict<-findFreqTerms(sms_dtm_train, 5)
> # 훈련자료 출현빈도 5회 이상 단어
> sms_train<-DocumentTermMatrix(sms_corpus_train,
+                               list(dictionary=sms_dict))
> # 검정자료 출현빈도 5회 이상 단어
> sms_test<-DocumentTermMatrix(sms_corpus_test,
+                              list(dictionary=sms_dict))
>
> # 단어 출현 빈도를 factor로 변환
> convert_counts <- function(x) {
+   x<-ifelse(x>0, 1, 0)
+   x<-factor(x, levels=c(0, 1), labels=c("No", "Yes"))
+ }
```

```
> # apply() convert_counts()를 사용한
> # 훈련/테스트 데이터 추출
> sms_train<-apply(sms_train, MARGIN=2,
+                                 convert_counts)
> sms_test<-apply(sms_test, MARGIN=2,
+                                 convert_counts)
> sms_train<-data.frame(sms_train)
> sms_test<-data.frame(sms_test)
```

```
> sms_train[1:10,1:10]
 checking good hope just week back thanks also pay cash
1     Yes    Yes   Yes  Yes  Yes   No     No   No   No   No
2      No     No    No   No   No  Yes    Yes   No   No   No
3      No     No    No   No   No   No     No  Yes  Yes   No
4      No     No    No   No   No   No     No   No   No  Yes
5      No     No    No   No   No   No     No   No   No  Yes
6      No     No    No   No   No   No     No   No   No   No
7      No     No    No   No   No   No     No   No   No   No
8      No     No    No   No   No   No     No   No   No   No
9      No     No    No   No   No   No     No   No   No   No
10     No     No    No   No   No   No     No   No   No   No
```

```
> # 나이브 베이지안 분류(훈련자료)
> library(e1071)
> sms_model<-naiveBayes(sms_train,
+                                    sms_raw_train$type)
> sms_model
:       :
> # 검정자료 분류
> sms_test_pred<-predict(sms_model, sms_test)
> # 분류표
> library(gmodels)
> CrossTable(sms_test_pred, sms_raw_test$type,
+                    prop.chisq=FALSE, prop.t=FALSE,
+                    prop.r=FALSE,
+                    dnn=c('predicted', 'actual'))
```

```
Cell Contents
|-------------------------|
|                       N |
|           N / Col Total |
|-------------------------|

Total Observations in Table:  1390

             | actual
   predicted |        ham |       spam | Row Total |
-------------|------------|------------|-----------|
         ham |       1203 |         32 |      1235 |
             |      0.997 |      0.175 |           |
-------------|------------|------------|-----------|
        spam |          4 |        151 |       155 |
             |      0.003 |      0.825 |           |
-------------|------------|------------|-----------|
Column Total |       1207 |        183 |      1390 |
             |      0.868 |      0.132 |           |
-------------|------------|------------|-----------|
```

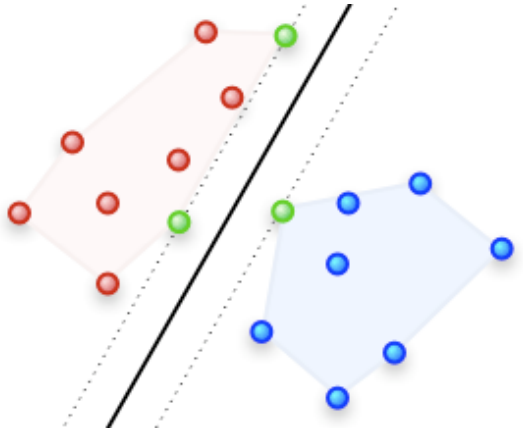
```
> # 분류율
> acc<-mean(sms_test_pred==sms_raw_test$type)
> acc
[1] 0.9741007
>
> # 나이브 베이지안 분류 모형 성능 향상
> sms_model2<-naiveBayes(sms_train,
+                        sms_raw_train$type, laplace=1)
> sms_test_pred2<-predict(sms_model2, sms_test)
> # 분류표
> CrossTable(sms_test_pred2, sms_raw_test$type,
+                        prop.chisq=FALSE, prop.t=FALSE,
+                        prop.r=FALSE,
+                        dnn=c('predicted', 'actual'))
```

```
  Cell Contents
|-------------------------|
|                       N |
|           N / Col Total |
|-------------------------|

Total Observations in Table:  1390

             | actual
   predicted |        ham |       spam | Row Total |
-------------|------------|------------|-----------|
         ham |       1203 |         31 |      1234 |
             |      0.997 |      0.169 |           |
-------------|------------|------------|-----------|
        spam |          4 |        152 |       156 |
             |      0.003 |      0.831 |           |
-------------|------------|------------|-----------|
Column Total |       1207 |        183 |      1390 |
             |      0.868 |      0.132 |           |
-------------|------------|------------|-----------|
```

```
> # 분류율
> acc<-mean(sms_test_pred2==sms_raw_test$type)
> acc
[1] 0.9748201
> detach("package:tm", unload=TRUE)
> detach("package:wordcloud", unload=TRUE)
> detach("package:e1071", unload=TRUE)
> detach("package:gmodels", unload=TRUE)
```

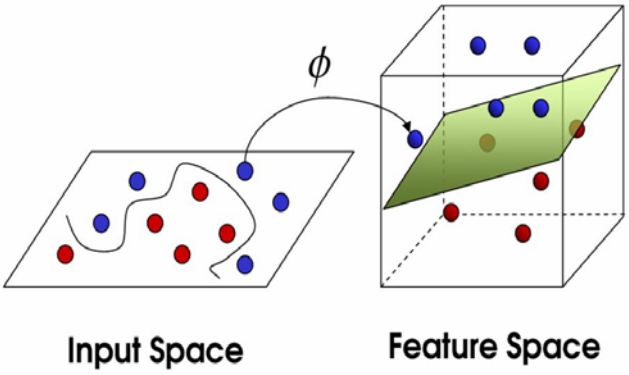
SVM(Support Vector Machine)

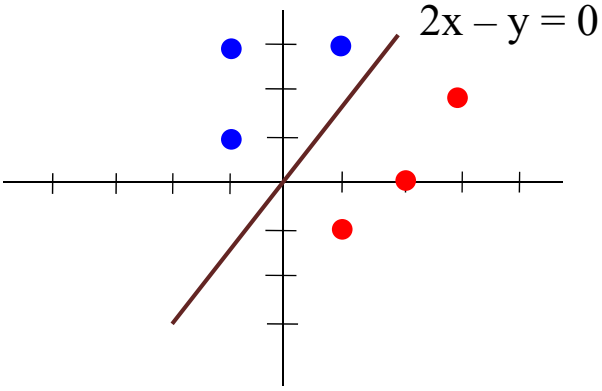
- **Introduction**
  - 분류 알고리즘 중에 하나
  - 원조 SVM은 두 범주 분류
    - 다중 범주로 확장
  - SVM은 기본적으로 선형 분류 방법
  - 두가지 아이디어
    - 최대-최소 하이퍼플레인(hyperplane) 찾기
    - 입력 데이터를 더 높은 차원의 공간으로 이동 (커널 방법)



연두색 점 : support vector

Principle of Support Vector Machines (SVM)





(-1,1) → -3
(-1,3) → -5 (-)
(1,3) → -1

(1,-1) → 3
(2,0) → 4 (+)
(3,2) → 4

- p차원 공간에서의 선형함수를 하이퍼플레인
  - 적절한 하이퍼플레인을 찾으면 오분류를 회피
- 두 범주를 구분하는 하이퍼플레인은 무수히 많은데 이 중에서 가장 적절한 것을 선택
  - 이러한 개념은 새로운 객체들을 분류하거나 추후에 다룰 오분류가 불가피한 경우를 다루기 위해 서 유용
- p차원 변수 벡터 x에 대한 하이퍼플레인 H

$$H: \mathbf{w}^T\mathbf{x} + b = 0$$

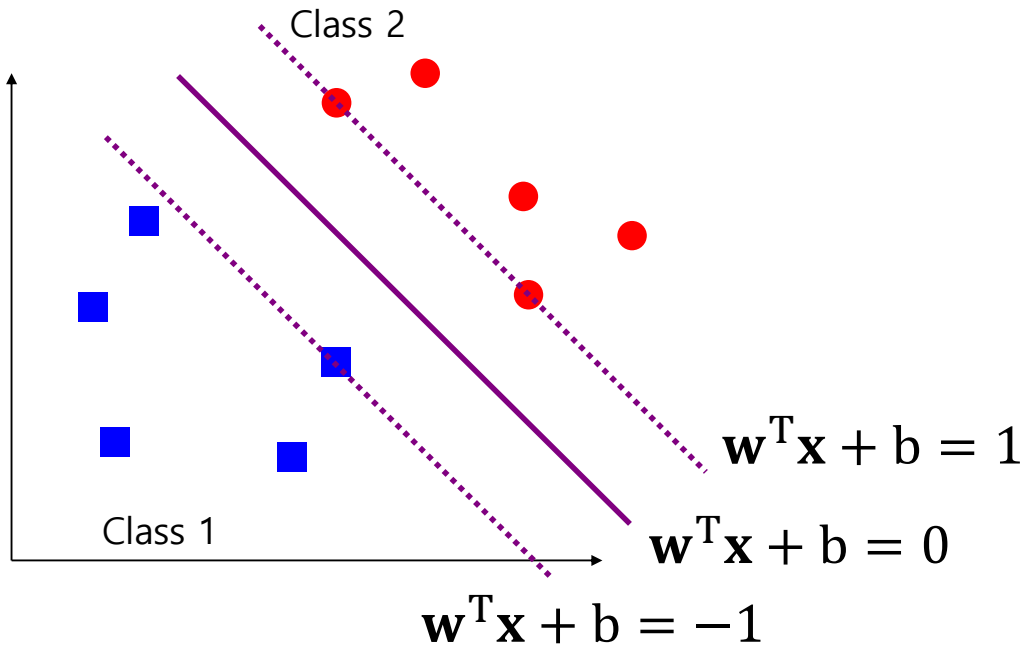
  - 여기에서, w는 하이퍼플레인의 계수벡터, b는 상수
  - 원점에서 하이퍼플레인과의 수직거리

$$|b|/\|\mathbf{w}\|$$

- 하이퍼플레인(H)와 평행인 두 하이퍼플레인

$$H_1: \mathbf{w}^T\mathbf{x} + b = 1$$
$$H_2: \mathbf{w}^T\mathbf{x} + b = -1$$

  - 단, H_1과 H_2 사이에는 객체가 없어야 함
  - H_1은 범주 1의 객체 중 분리 하이퍼플레인 H에 가장 가까운 객체를 지나며, H_2는 범주 -1의 객체 중 분리 하이퍼플레인 H에 가장 가까운 객체를 지남



- H_1과 H_2 사이의 거리

$$2/\|\mathbf{w}\|$$

- 선형 SVM은 이 거리를 최대로 하는 분리 하이퍼플레인을 찾는 문제로 귀착

- N개의 객체로 이루어진 학습자료의 i번째 객체를 p개의 변수로 이루어진 벡터 x_i로 표기하고, 이에 대응하는 분류된 범주를 y_i로 표기
- 이때 분리 하이퍼플레인을 찾는 문제는 다음과 같은 최적화 문제

$$\text{Max } \frac{2}{\mathbf{w}^T\mathbf{w}}$$

$$\text{Subject to}$$

$$\mathbf{w}^T\mathbf{x}_i + b \geq 1 \text{ for } y_i = 1 \quad , \quad i = 1,2,\cdots,N$$

$$\mathbf{w}^T\mathbf{x}_i + b \leq -1 \text{ for } y_i = -1 \quad , \quad i = 1,2,\cdots,N$$

  - 위 제약식은 다음과 같은 하나의 식으로 표현

$$y_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1$$

- 따라서 최적화 문제는 다음과 같아 표현

$$\text{Min } \frac{\mathbf{w}^T\mathbf{w}}{2}$$

$$\text{Subject to}$$

$$y_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1 \quad , \quad i = 1,2,\cdots,N$$

  - 위 제약조건에 비음의 라그랑지 계수 (Lagrange Multiplier) α_i를 도입하여 라그랑지 함수를 유도하면 다음과 같은 최적화 문제(원문제 : primal problem)

$$\text{Min } L_P = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i\left[y_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) - 1\right]$$

$$\alpha_i \geq 0 \quad , \quad i = 1,2,\cdots,N$$

- 원문제에 KKT(Karush-Kuhn-Tucker) 조건을 적용시키면

$$\frac{\partial L_P}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = 0$$

$$\frac{\partial L_P}{\partial b} = \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\alpha_i \left[y_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) - 1 \right] = 0 \quad , \quad i = 1,2,\cdots,N$$

$$y_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) - 1 \geq 0 \quad , \quad i = 1,2,\cdots,N$$

$$\alpha_i \geq 0 \quad , \quad i = 1,2,\cdots,N$$

  - KKT 조건은 원문제 해에 대한 필요충분조건을 제공

- 앞의 식에 따라서

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

  - 따라서 α_i가 결정되면 분리 하이퍼플레인의 계수 w를 산출 가능
  - α_i 값을 구하기 위해서는 울프쌍대문제를 고려
- $\alpha_i > 0$인 임의의 객체 i에 대해서 다음과 같이 b를 산출

$$b = \frac{1 - y_i \mathbf{w}^T \mathbf{x}_i}{y_i} \quad , \quad \alpha_i > 0$$

  - 객체별로 다른 b값이 도출될 수 있는데, 이 경우는 평균값을 취하여 최종 b값을 정하는 것이 안정적

- 분리 하이퍼플레인을 도출하는 것은 SVM을 학습시키는 것
- α_i 값을 얻기 위한 원문제에 대한 울프쌍대문제 (wolf dual problem)

$$\text{Min } L_P = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \left[y_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) - 1 \right]$$

Subject to

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \quad , \quad i = 1,2,\cdots,N$$

- 앞선 식 $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$ 를 L_P 의 목적식에 대입하면 쌍대문제는 다음과 동일

$$\text{Min } L_P = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

Subject to

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \quad , \quad i = 1,2,\cdots,N$$

  - 위 문제는 목적함수가 α_i 에 대해 이차이고 제약조건이 선형이므로 이차계획문제 (quadratic problem)

- 따라서 이차계획문제(quadratic problem)에서 α_i를 결정한 후 $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$와 $b = \frac{1 - y_i \mathbf{w}^T \mathbf{x}_i}{y_i}$를 이용하여 하이퍼플레인의 계수들을 결정
- 최적해에서 $\alpha_i > 0$인 객체를 서포트 벡터라 하며 이는 하이퍼플레인 H_1 또는 H_2 상에 위치
- 그리고 H_1 및 H_2 상에 놓이지 않는 객체는 $\alpha_i = 0$
- SVM을 학습시킨 후 새로운 객체 x에 대한 분류 규칙

$y_i = \mathbf{w}^T \mathbf{x}_i + b > 0$ 이면, 범주 1로 분류

$y_i = \mathbf{w}^T \mathbf{x}_i + b \leq 0$ 이면, 범주 -1로 분류

- 학습표본이 선형 하이퍼플레인에 의해 두 개의 범주로 분류되지 않는 경우 최적화문제와 이차계획문제는 가능해가 존재 하지 않음
- 이때 오분류를 허용하는 것이 불가피하며, 이를 위한 최적화문제는 여유변수(slack variable) ξ_i가 추가

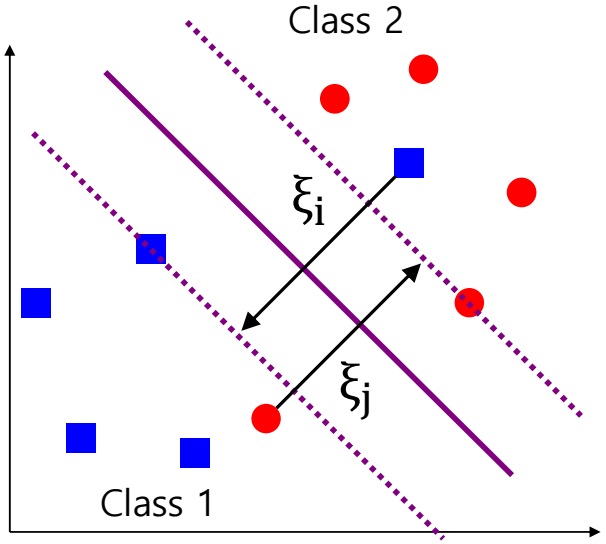
$$\text{Min} \ \frac{\mathbf{w}^T \mathbf{w}}{2} + C \sum_{i=1}^{N} \xi_i$$

Subject to

$$y_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) \geq 1 - \xi_i \quad , \quad i = 1, 2, \cdots, N$$

$$\xi_i \geq 0 \ \forall i$$

- 두 개의 평행한 하이퍼플레인의 마진 밖에 다른 범주의 객체가 놓이는 것을 허용하되 다른 범주 영역에 속하는 경우 하이퍼플레인에서 떨어진 거리에 비례하여 패널티를 주는 것
  - C값이 패널티 단가에 해당
  - C값은 입력해야 하는 상수



- 따라서 울프쌍대문제는 라그랑지 변수를 도입할 때 다음과 같이 표현

$$\text{Min } L_P = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{Subject to}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

$$0 \le \alpha_i \le C \quad , \quad i = 1,2,\cdots,N$$

  - 이는 앞선 경우와 비교하면 α_i의 상한값이 C

- 이 문제에 대한 최적해에서 각 객체는 다음과 같이 위치
  - $\alpha_i = C$일 때, x_i는 마진에 위치
  - $0 \le \alpha_i \le C$일 때 , x_i는 H_1 또는 H_2에 위치
  - $\alpha_i = 0$일 때, x_i는 그 밖에 위치
    - $\alpha_i > 0$인 객체를 서포트 벡터
- 계수 w는 $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$ 또는 서포트 벡터만을 고려하여 산출

$$\mathbf{w} = \sum_{i=1}^{N_S} \alpha_i y_i \mathbf{x}_i$$

- 상수 b를 구하기 위해서는 여유변수가 추가된 원문제에 KKT 조건을 적용시킨 제약식 검토가 필요
  - $0 \le \alpha_i \le C$ 일 때 $\xi_i = 0$이 되므로 이 조건을 만족시키는 임의의 객체에 대하여 다음과 같은 식으로 b를 산출

$$b = \frac{1 - y_i \mathbf{w}^T \mathbf{x}_i}{y_i} \quad , \quad \alpha_i > 0$$

    - 여기서도 계산의 안정성을 위해서 이 조건을 만족시키는 모든 객체에 대한 b값을 구한 후 평균값을 사용

- **C값에 따라 하이퍼플레인이 다르므로 분류기준 또한 변하지만 크게 민감하지 않는 것으로 알려짐**
- **적절한 C값을 찾기 위해서는 cross-validation) 을 활용**
- **SVM을 학습시킨 후 새로운 객체 x에 대한 분류 규칙**

$y_i = \mathbf{w}^T\mathbf{x}_i + b > 0$ 이면, 범주 1로 분류

$y_i = \mathbf{w}^T\mathbf{x}_i + b \leq 0$ 이면, 범주 -1로 분류

- **SVM(e1071 패키지)**

```
> svm(tr, cl, scale=TRUE, kernel="radial",
+         degree=3, coef0=0, cost=1,
+         gamma=if (is.vector(x)) 1 else 1 / ncol(x))
```

  - **tr : 학습자료(matrix)**
  - **cl : 학습자료의 범주 변수(factor)**
  - **scale : 척도를 지정할 변수**
  - **kernal : 학습과 예측에 사용되는 커널 유형**
    - **"linear", "polynomial", "radial", "sigmoid"**
  - **degree: polynomial 커널에 필요한 모수**

- coef0 : 다항 또는 시그모이드 커널에 요구되는 모수
- cost : 제약 위배의 비용
- gamma : 선형을 제외한 모든 커널에 필요한 매개 변수

▪ SVM(e1071 패키지)

```
> svm(formula, data=NULL,
+         type="C-classification", ...)
```

- formula : 모형식
  - class ~ $x_1+x_2+\cdots$
- data : 분석 데이터(data frame)
- type : 수행방법(분류 또는 회귀)
  - C-classification, eps-regression, nu-classification, nu-regression

- **초모수 조율(e1071 패키지)**

```
> tune.svm(formula, data=NULL, ...)
```

  - **formula : 모형식**
    - **class ~ $x_1+x_2+\cdots$**
  - **data : 분석 데이터(data frame)**

예제 2.6

- **다음 붓꽃(iris.csv)의 종(Species)을 SVM을 이용하여 분류하시오.**
  - **종 : setosa, versicolor, virginica**

```
> iris.data<-read.csv("iris.csv")
> head(iris.data)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> # SVM 학습
> library(e1071)
> model<-svm(Species~., data=iris.data)
> detach("package:e1071", unload=TRUE)
```

```
> model

Call:
svm(formula = Species ~ ., data = iris.data)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.25

Number of Support Vectors:  51
```

```
> # 예측
> pred<-predict(model, newdata=iris.data[,-5])
>
> # 분류표
> library(gmodels)
> CrossTable(pred, iris.data$Species,
+                    prop.chisq=FALSE, prop.t=FALSE,
+                    prop.r=FALSE,
+                    dnn=c('predicted', 'actual'))
> detach("package:gmodels", unload=TRUE)
>
> acc<-mean(pred==iris.data$Species) # 분류율
> acc
[1] 0.9733333
```

```
Cell Contents
|-------------------------|
|                       N |
|           N / Col Total |
|-------------------------|

Total Observations in Table:  150

             | actual
   predicted |     setosa | versicolor |  virginica |  Row Total |
-------------|------------|------------|------------|------------|
      setosa |         50 |          0 |          0 |         50 |
             |      1.000 |      0.000 |      0.000 |            |
-------------|------------|------------|------------|------------|
  versicolor |          0 |         48 |          2 |         50 |
             |      0.000 |      0.960 |      0.040 |            |
-------------|------------|------------|------------|------------|
   virginica |          0 |          2 |         48 |         50 |
             |      0.000 |      0.040 |      0.960 |            |
-------------|------------|------------|------------|------------|
Column Total |         50 |         50 |         50 |        150 |
             |      0.333 |      0.333 |      0.333 |            |
-------------|------------|------------|------------|------------|
```

```
> library(e1071)
Warning message:
package 'e1071' was built under R version 3.5.1
> # 모형 수정
> model<-svm(Species~., data=iris.data, cost=5,
+                     gamma=0.5)
> model

Call:
svm(formula = Species ~ ., data = iris.data, cost = 5, gamma = 0.5)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  5
      gamma:  0.5

Number of Support Vectors:  48

> detach("package:e1071", unload=TRUE)
```

```
> # 예측
> pred<-predict(model, newdata=iris.data[,-5])
>
> # 정확도
> library(gmodels)
Warning message:
package 'gmodels' was built under R version 3.5.1
> # 분류표
> CrossTable(pred, iris.data$Species,
+                     prop.chisq=FALSE, prop.t=FALSE,
+                     prop.r=FALSE,
+                     dnn=c('predicted', 'actual'))
> detach("package:gmodels", unload=TRUE)
> acc<-mean(pred==iris.data$Species) # 분류율
> acc
[1] 0.9866667
```

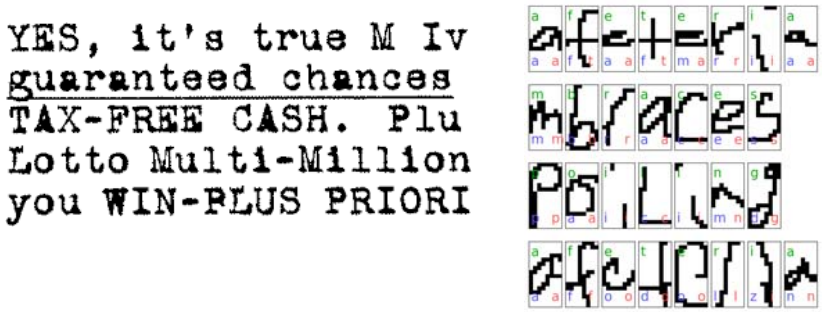
```
Cell Contents
|-------------------------|
|                       N |
|           N / Col Total |
|-------------------------|

Total Observations in Table:  150

             | actual
   predicted |     setosa | versicolor |  virginica |  Row Total |
-------------|------------|------------|------------|------------|
      setosa |         50 |          0 |          0 |         50 |
             |      1.000 |      0.000 |      0.000 |            |
-------------|------------|------------|------------|------------|
  versicolor |          0 |         48 |          0 |         48 |
             |      0.000 |      0.960 |      0.000 |            |
-------------|------------|------------|------------|------------|
   virginica |          0 |          2 |         50 |         52 |
             |      0.000 |      0.040 |      1.000 |            |
-------------|------------|------------|------------|------------|
Column Total |         50 |         50 |         50 |        150 |
             |      0.333 |      0.333 |      0.333 |            |
-------------|------------|------------|------------|------------|
```

```
> library(e1071)
> tuned<-tune.svm(Species~., data=iris.data,
+                         degree=1:5, coef0=0:1,
+                         cost=10^(0:2), gamma=10^(-6:-1))
> # model<-svm(iris.data[,1:4], iris.data$Species)
> tuned

Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 degree gamma coef0 cost
      1   0.1     0    1

- best performance: 0.04666667

> detach("package:e1071", unload=TRUE)
```

예제 2.7

- **다음 영문자 자료(letterdata.csv)를 SVM을 이용하여 영문자(letter)를 분류하시오.**

	A	B	C	D	E	F	G	H	I	J	
1	letter	xbox	ybox	width	height	onpix	xbar	ybar	x2bar	y2bar	xyl
2	T	2	8	3	5	1	8	13	0	6	
3	I	5	12	3	7	2	10	5	5	4	
4	D	4	11	6	8	6	10	6	2	6	
5	N	7	11	6	6	3	5	9	4	6	
6	G	2	1	3	1	1	8	6	6	6	
7	S	4	11	5	8	3	8	8	6	9	
8	B	4	2	5	4	4	8	7	6	6	
9	A	1	1	3	2	1	8	2	2	2	
10	J	2	2	4	4	2	10	6	2	6	
11	M	11	15	13	9	7	13	2	6	2	
12	X	3	9	5	7	4	8	7	3	8	




```
> library(e1071)   # for svm
> library(gmodels) # for CrossTable
>
> # Read dataset
> letters.data<-read.csv("letterdata.csv")
> head(letters.data)
 letter xbox ybox width height onpix xbar ybar x2bar y2bar xybar x2ybar xy2bar xedge xedgey yedge yedgex
1      T    2    8     3      5     1    8   13     0     6     6     10      8     0      8     0      8
2      I    5   12     3      7     2   10    5     5     4    13      3      9     2      8     4     10
3      D    4   11     6      8     6   10    6     2     6    10      3      7     3      7     3      9
4      N    7   11     6      6     3    5    9     4     6     4      4     10     6     10     2      8
5      G    2    1     3      1     1    8    6     6     6     6      5      9     1      7     5     10
6      S    4   11     5      8     3    8    8     6     9     5      6      6     0      8     9      7
>
> # 학습자료와 검정자료
> n<-dim(letters.data)[1]  # 전체 데이터 수
> # 학습자료 표본 번호
> m<-sort(sample(1:n, size=16000))
```

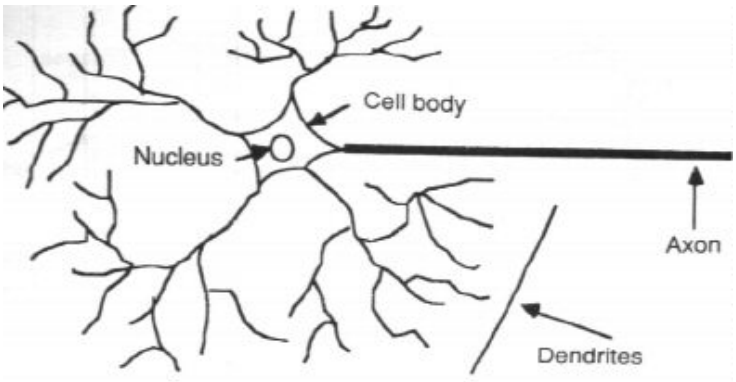
```
> letters_train<-letters.data[m, ]
> letters_test<-letters.data[-m, ]
> # 모형의 학습
> model<-svm(letter~., data=letters_train)
> # 적합된 모형의 검정
> result<-predict(model, newdata=letters_test[,-1])
> # 분류표
> CrossTable(result, letters_test$letter,
+                    prop.chisq=FALSE, prop.t=FALSE,
+                    prop.r=FALSE,
+                    dnn=c('predicted', 'actual'))
> acc<-mean(result==letters_test[,1]) # 분류율
> acc
[1] 0.94825
```

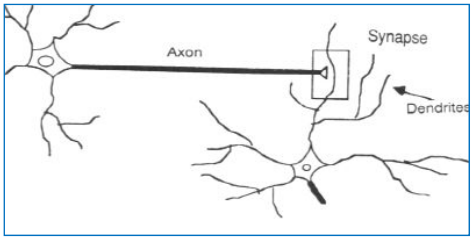
```
> detach("package:gmodels", unload=TRUE)
> detach("package:e1071", unload=TRUE)
```

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Col Total |
|-------------------------|


Total Observations in Table:  4000


             | actual
   predicted |         A |         B |         C |         D |         E |         F |         G |         H |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
           A |       163 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |
             |     1.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
           B |         0 |       141 |         0 |         1 |         3 |         1 |         0 |         2 |
             |     0.000 |     0.940 |     0.000 |     0.006 |     0.023 |     0.006 |     0.000 |     0.013 |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
           C |         0 |         0 |       139 |         0 |         1 |         0 |         0 |         0 |
             |     0.000 |     0.000 |     0.946 |     0.000 |     0.008 |     0.000 |     0.000 |     0.000 |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
           D |         0 |         4 |         0 |       158 |         0 |         0 |         3 |         6 |
             |     0.000 |     0.027 |     0.000 |     0.969 |     0.000 |     0.000 |     0.019 |     0.040 |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
           E |         0 |         1 |         1 |         0 |       126 |         0 |         0 |         0 |
             |     0.000 |     0.007 |     0.007 |     0.000 |     0.955 |     0.000 |     0.000 |     0.000 |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
           F |         0 |         0 |         0 |         0 |         0 |       160 |         0 |         0 |
             |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.970 |     0.000 |     0.000 |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
           G |         0 |         1 |         5 |         0 |         1 |         0 |       146 |         1 |
             |     0.000 |     0.007 |     0.034 |     0.000 |     0.008 |     0.000 |     0.948 |     0.007 |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
           H |         0 |         0 |         0 |         0 |         0 |         0 |         0 |       124 |
             |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.827 |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
           I |         0 |         0 |         0 |         0 |         0 |         1 |         0 |         0 |
             |     0.000 |     0.000 |     0.000 |     0.000 |     0.000 |     0.006 |     0.000 |     0.000 |
-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
```

신경망

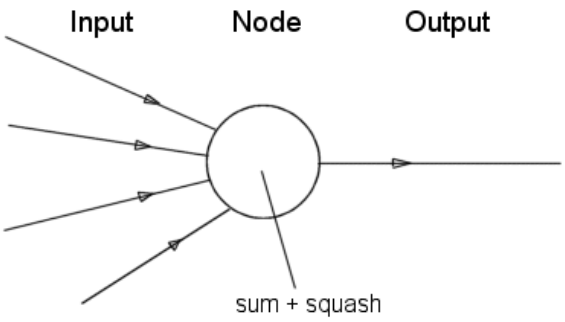
- 뉴런이라는 신경세포체가 신경물질을 분비함으로써 축색말단을 통하여 다른 뉴런에 자극을 전달하는 모습

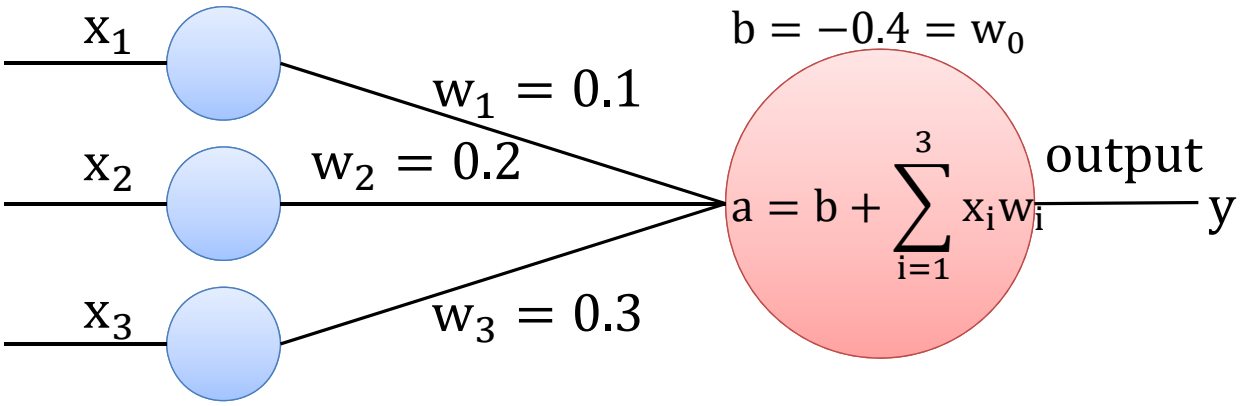




- 우리 뇌신경세포는 기본적으로 전기자극으로 작동되며 뉴런에서 뉴런으로 신호를 전파

- 뉴런에 전달되는 어떤 자극의 크기가 어느 정도 이상이면 이런 전기신호가 전달
  - 이런 수많은 신경전달들이 학습이 되고 기억
- 이런 과정을 단순화시켜서 모형을 만들어 관측된 입력값을 주면 원하는 출력값을 만드는 뉴런의 구조를 설계





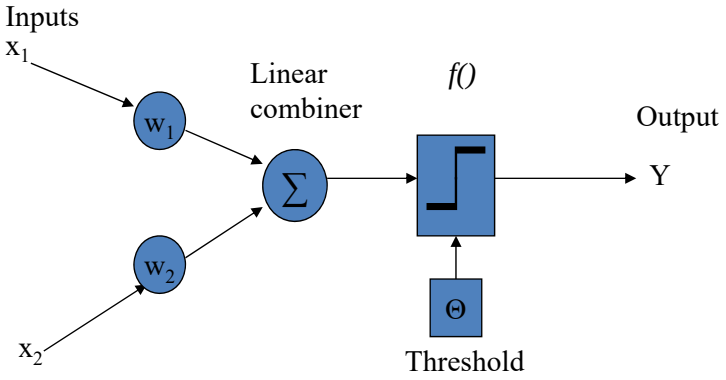
- 위 모형에서 x_1, x_2, x_3에 0 또는 1의 값 입력

$$\text{output} = \begin{cases} 0 & , \quad a < \text{threshold} \\ 1 & , \quad a \geq \text{threshold} \end{cases}$$

- 여기에서 노드에 해당하는 부분이 뉴런
  - 입력값이 전달되는 층이 입력층(input layer)
  - 출력값이 나오는 부분이 출력층(output layer)

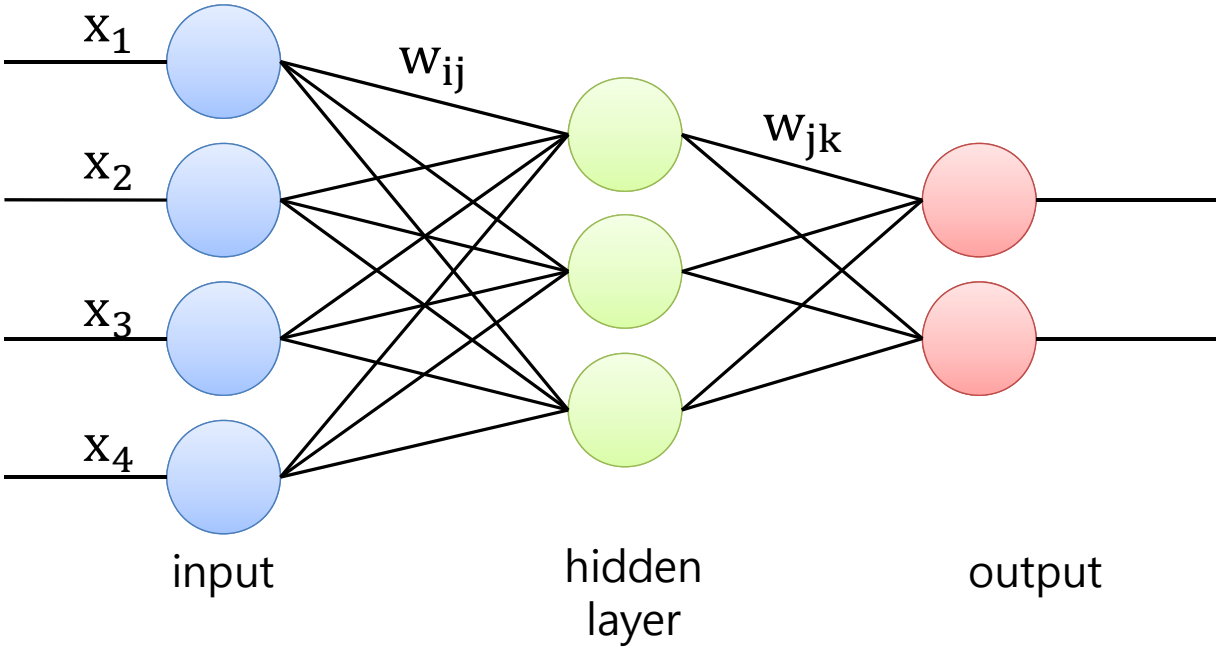
- 출력층을 제외한 전체 층이 1개 밖에 없으므로 단층신경망(single layer neural network) 또는 퍼셉트론(perceptron)
  - 여기에서 변수는 입력값 X, 출력값 y, 가중치 w, 노드에 더하는 값(bias) b
  - 출력 노드에 마지막으로 붙이는 함수를 활성함수(activation function)
- 네트워크를 설계를 한다는 것
  - 입력층의 노드를 수 결정
  - 총 layer를 수 결정
  - 가중치와 bias의 값 등을 결정하는 것
- 구조가 정해지고 입력값과 출력값이 정해지면 거기에 가장 적당한 가중치와 bias를 학습

- **Perceptron 모형은 1975년 코넬항공연구소의 Rosenbalatt0| 제기**



  - **선형합성함수로 비선형문제는 풀지 못하는 문제**

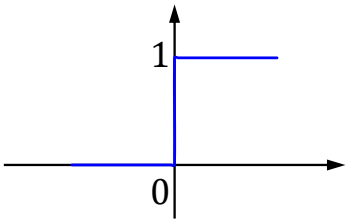
- **1969년 MIT 교수 minsky와 papert0| 중간에 은닉층(hidden layer)을 도입하는 다층신경망 (multi-layer neural network) 모형을 발전시켜 정확도를 크게 개선**



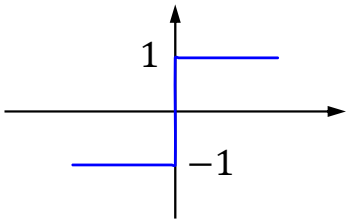
- 입력값은 은닉층에 도달
- 은닉층의 노드는 주어진 입력에 따라 활성화
- 활성화된 은닉 노드는 출력값을 계산하고 그 결과를 출력층에 전달
- 출력층의 노드는 입력에 따라 활성화되고, 활성화된 노드들이 최종 출력값을 계산

- 이론적으로 여러 층의 은닉층 가능
- 입력층을 제외하고 모든 노드는 bias b를 가지고 있으며 노드와 노드의 연결선은 가중치 w
- 활성함수 f를 이용하여 출력층 값이 결정되면 출력값을 결정

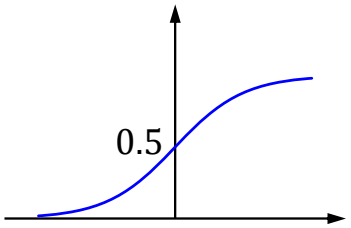
- Output node의 출력값을 계산하기 위하여 사용되는 활성함수



$$\text{step}(x) = \begin{cases} 1 & , \quad x > t \\ 0 & , \quad \text{other} \end{cases}$$



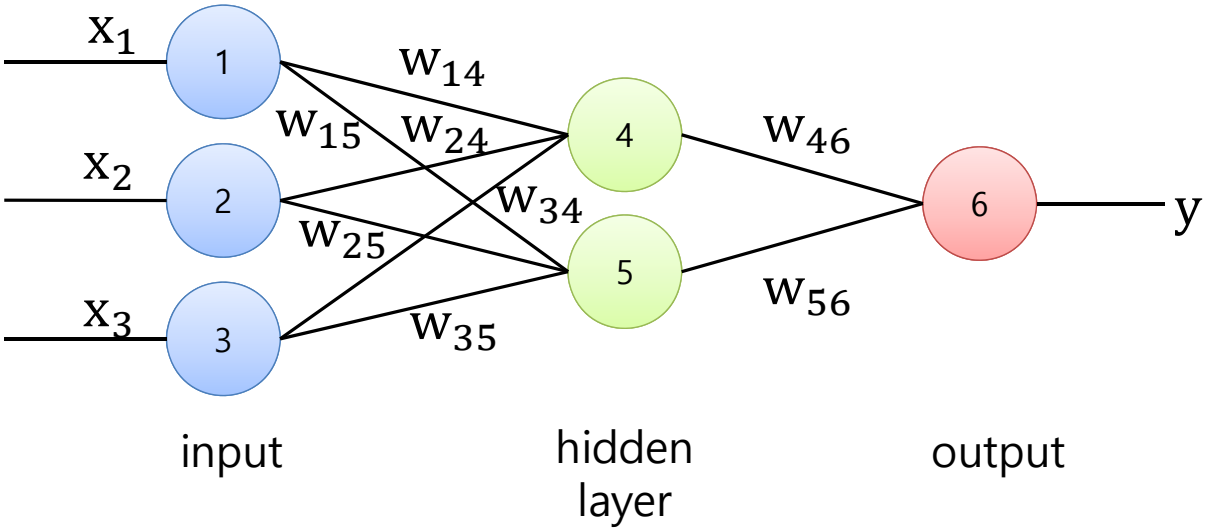
$$\text{sign}(x) = \begin{cases} 1 & , \quad x \geq 0 \\ -1 & , \quad \text{other} \end{cases}$$



$$\text{sigmoide}(x) = \frac{1}{1 + e^{-x}}$$

- 신경망 학습은 가중치 w의 조절 작업
  - 입력값 x가 원하는 출력값 y가 있을 때 입력값을 입력층에 준 다음 모델의 출력값이 원하는 출력값과 같은지 확인
  - 값이 같지 않으면 가중치 w를 적절히 조절

- 가중계수의 추정
  - 기울기 하강법(gradient descent method)을 사용



  - 모든 n개의 데이터의 관측값을 y_i라하고 신경망의 추정치를 $\hat{y}_l$라고 하면 오차함수

$$E(w) = \sum_{i=1}^{n_i} (y_i - \hat{y}_l)^2$$

    - 여기서 E(w)를 최소로 하는 가중계수 w를 찾는 것이 목표
  - j노드에서 k노드로 가는 가중치를 w_{jk}라 하고 이것으로 E(w)를 편미분

$$\frac{\partial E(w)}{\partial w_{jk}} = -2 \sum_{i=1}^{n_i} (y_i - \hat{y}_l) \frac{\partial \hat{y}_l}{\partial w_{jk}}$$

    - $\frac{\partial \hat{y}_l}{\partial w_{jk}}$는 $\hat{y}_l(1 - \hat{y}_l)$에 비례

- E(w)가 감소하는 방향으로 w_{jk}를 움직이고 싶다면 다음과 같이 만들 수 있을 것

$$w_{jk} \leftarrow w_{jk} - \lambda \frac{\partial E(w)}{\partial w_{jk}}$$

  - λ는 상수로 학습률
- 그러면

$$\frac{\partial \hat{y}_l}{\partial w_{jk}} = \frac{\partial \left(\sigma\left(w_{jk} O_j + b_k\right)\right)}{\partial w_{jk}} = O_k(1 - O_k)O_j$$

  가 되고 $O_k(1 - O_k)(y_i - \hat{y}_l)$를 E_k라 하면 E_k오차=오차변화율×추정오차
- Bias b는 $b \leftarrow b - \lambda E_k$ 같이 추정
  - 결국 E_k를 구하는 것이 관건

- 역전파 알고리즘
  - 가장 먼저 해야 할 일은 λ를 0~1사이의 적당한 값으로 선택하고 가중계수 w와 bias b에 초기값으로 임의(random)의 값을 설정(-1~1 사이의 값)
  - 그 다음 맨 마지막인 output node인 6번부터 앞의 알고리즘을 수행해 보는 것
  - 먼저 O_1~O_6까지의 값을 구할 수 있는데 O_1~O_3은 입력값이고 O_4~O_6은 임의의 초기값
  - 가장 마지막 노드인 6번의 E_6을 계산하는 것으로 알고리즘 시작

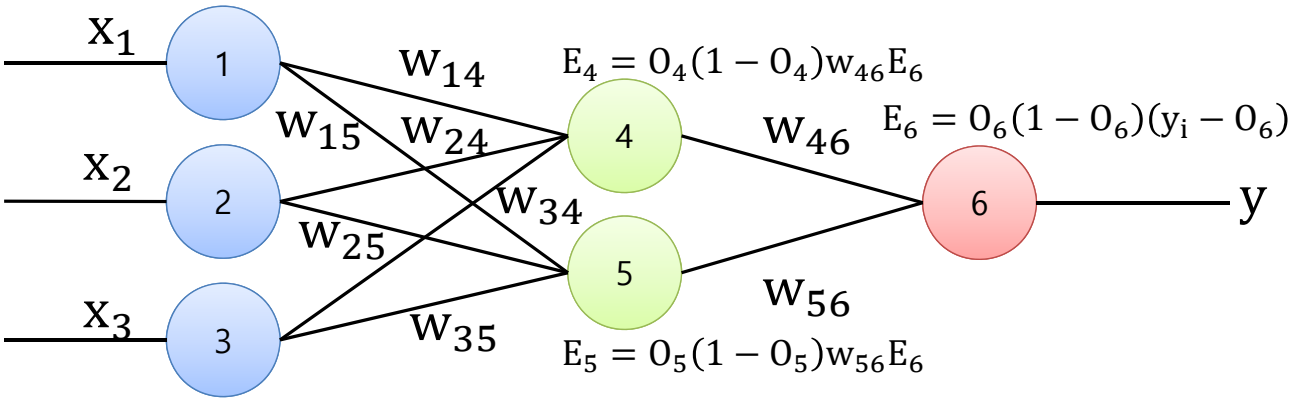
$$E_6 = O_6(1 - O_6)(y_i - O_6)$$

오차변화율 추정오차

– E_5는 5번 노드와 연결된 모든 노드의 오차 E_k와 가중계수의 곱을 모두 더한 값이 추정오차
  - E_4도 동일한 방법으로 계산

$$E_5 = O_5(1 - O_5)\sum_k w_{5k}E_k$$

$$E_4 = O_4(1 - O_4)\sum_k w_{4k}E_k$$



– 이렇게 하여 가중계수와 편향을 다음과 같이 학습

$$w_{46} \leftarrow w_{46} + \lambda E_6 O_4$$
$$w_{56} \leftarrow w_{56} + \lambda E_6 O_5$$
$$w_{14} \leftarrow w_{14} + \lambda E_4 x_1$$
$$w_{15} \leftarrow w_{15} + \lambda E_5 x_1$$
$$w_{24} \leftarrow w_{24} + \lambda E_4 x_2$$
$$w_{25} \leftarrow w_{25} + \lambda E_5 x_2$$
$$w_{34} \leftarrow w_{34} + \lambda E_4 x_3$$
$$w_{35} \leftarrow w_{35} + \lambda E_5 x_3$$

  - 이런 식으로 오차 E_k가 뒤에서 앞으로 계산되면서 가중계수와 편향이 학습

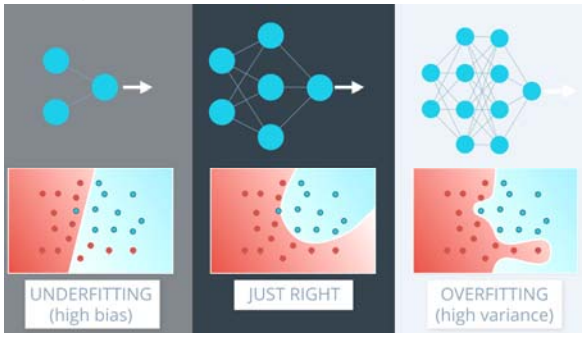
- 모델의 출력값이 원하는 출력값과 얼마나 같은지 비교하는 기준
  - 오차제곱합(sum of square error)
    - 실제값 y, 예측값 $\hat{y}_i$

$$\sum (y_i - \hat{y}_i)^2$$

  - 엔트로피

$$-\sum y_i \ln(\hat{y}_i)$$

- 장점
  - 신경망은 분류나 수치 예측 문제에 적용 가능
  - 가장 정확한 모델링 방법중의 하나
- 단점
  - 모형이 복잡해질 수 있고 훈련시간이 길어짐
  - 훈련데이터에 overfitting 되거나 underfitting 되는 문제



  - 모델의 해석이 불가능하지는 않지만 어려움

▪ 신경망(nnet 패키지)

```
> nnet(formula, data, ...)
> nnet(tr, target, size, rang=0.7, decay=0,
+         linout=FALSE, entropy=FALSE, maxit=100)
```

- formula : 모형식
  - class ~ $x_1+x_2+\cdots$
- tr : 학습자료(matrix or vector)
- target : 종속변수(class 정보)
- size : 은닉층 노드 수
- rang : 초기 무작위 가중치[-rang, rang]
- decay : 가중치 감소를 위한 모수 (default: 0)

- linout : 선형활성함수
  - TRUE(선형 함수), FALSE(시그모이드 함수)
- entropy : 모형 학습시 모형의 출력과 원하는 값을 비교할 때 사용할 함수
  - TRUE(엔트로피), FALSE(SSE)
- maxit : 최대 반복 수(default: 100)

▪ 지시 행렬(nnet 패키지)

```
> class.ind(cl)
```

- cl : 범주(factor)형 벡터

▪ 신경망(neuralnet 패키지)

```
> neuralnet(formula, data, hidden=1, threshold=0.01,
+           stepmax=1e+05, err.fct="sse", act.fct="logistic")
```

- formula, data : 모형식과 데이터 프레임
  - class ~ $x_1+x_2+\cdots$
- hidden : 은닉층 노드 수
  - 다층인 경우 c(k,n,m,…)로 지정
- threshold : 모형의 중지 임계값
- stepmax : 신경망 훈련의 최대 단계
- err.fct : 오차 함수
  - 오차제곱합("SSE"), 교차엔트로피("CE")
- act.fct : 활성함수
  - 로지스틱("logistic"), 쌍곡함수("tanh")

▪ 신경망(neuralnet 패키지)

```
> compute(object, covariate)
```

- object : 신경망 모형(neuralnet) 개체
- covariate : 독립변수 데이터프레임

예제 2.8

- 다음 붓꽃(iris.csv)의 종(Species)을 나이브 신경망 방법을 이용하여 분류하시오.
  - 종 : setosa, versicolor, virginica

```
> library(nnet)
> iris.data<-read.csv("iris.csv")
> head(iris.data)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> # 범주형 자료의 지시 행렬
> targets<-class.ind(iris$Species)
```

```
> head(targets)
     setosa versicolor virginica
[1,]      1          0         0
[2,]      1          0         0
[3,]      1          0         0
[4,]      1          0         0
[5,]      1          0         0
[6,]      1          0         0
> # 신경망
> set.seed(12345)
> model<-nnet(Species~., data=iris.data, size=2,
+                     rang=0.1, decay=5e-4, maxit=200)
```

```
# weights:  19
initial  value 164.936374
iter  10 value 73.620336
iter  20 value 69.839776
iter  30 value 69.539390
iter  40 value 69.504825
iter  50 value 68.216490
iter  60 value 23.210824
iter  70 value 8.592605
iter  80 value 8.532545
iter  90 value 8.412819
iter 100 value 7.989675
iter 110 value 7.272828
iter 120 value 6.573304
iter 130 value 6.381460
iter 140 value 6.321648
iter 150 value 6.306650
iter 160 value 6.300459
final  value 6.299108
converged
```

```
> # 예측
> pred<-predict(model, newdata=iris.data[,-5])
> head(pred)
      setosa    versicolor     virginica
1 0.9998297 0.0001454478 2.481006e-05
2 0.9998009 0.0001717016 2.735733e-05
3 0.9998207 0.0001536327 2.562327e-05
4 0.9997850 0.0001862689 2.870126e-05
5 0.9998315 0.0001438540 2.464957e-05
6 0.9998253 0.0001494552 2.521127e-05
> predClass<-max.col(pred)
> predClass<-factor(predClass, levels=1:ncol(pred),
+                                 labels=colnames(pred))
> detach("package:nnet", unload=TRUE)
```

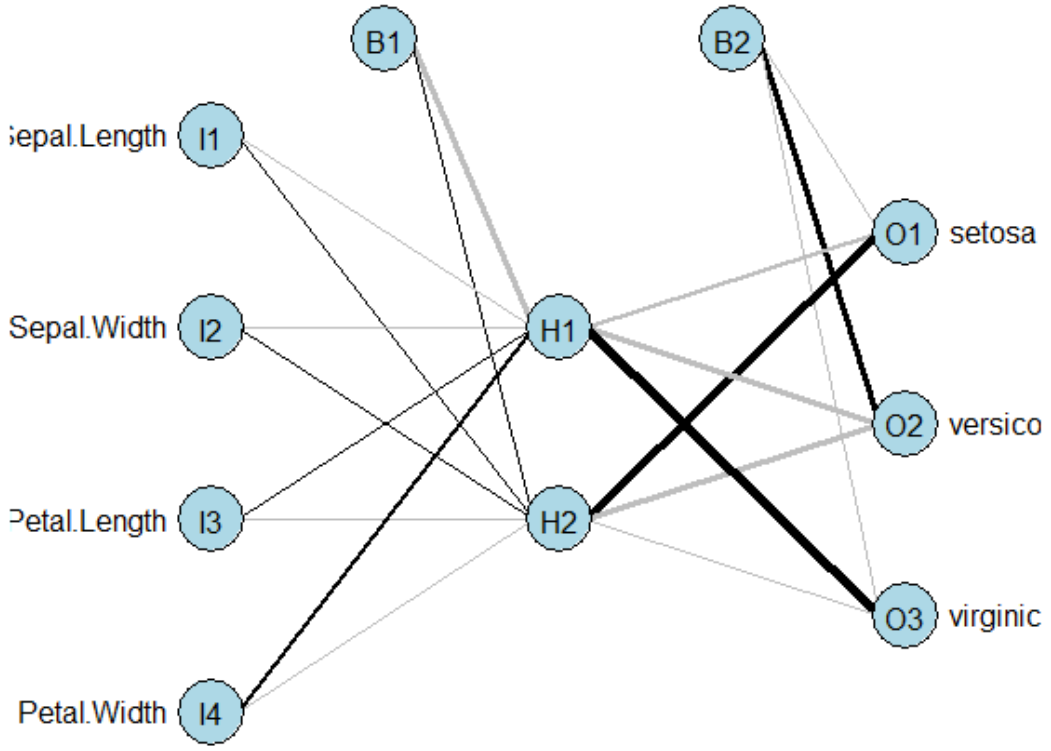
```
> # 정확도
> # 분류표
> library(gmodels)
> CrossTable(predClass, iris.data$Species,
+                       prop.chisq=FALSE, prop.t=FALSE,
+                       prop.r=FALSE,
+                       dnn=c('predicted', 'actual'))
> detach("package:gmodels", unload=TRUE)
> # 분류율
> acc<-mean(predClass==iris.data$Species)
> acc
[1] 0.9866667
```

Cell Contents
|-------------------------|
| N |
N / Col Total

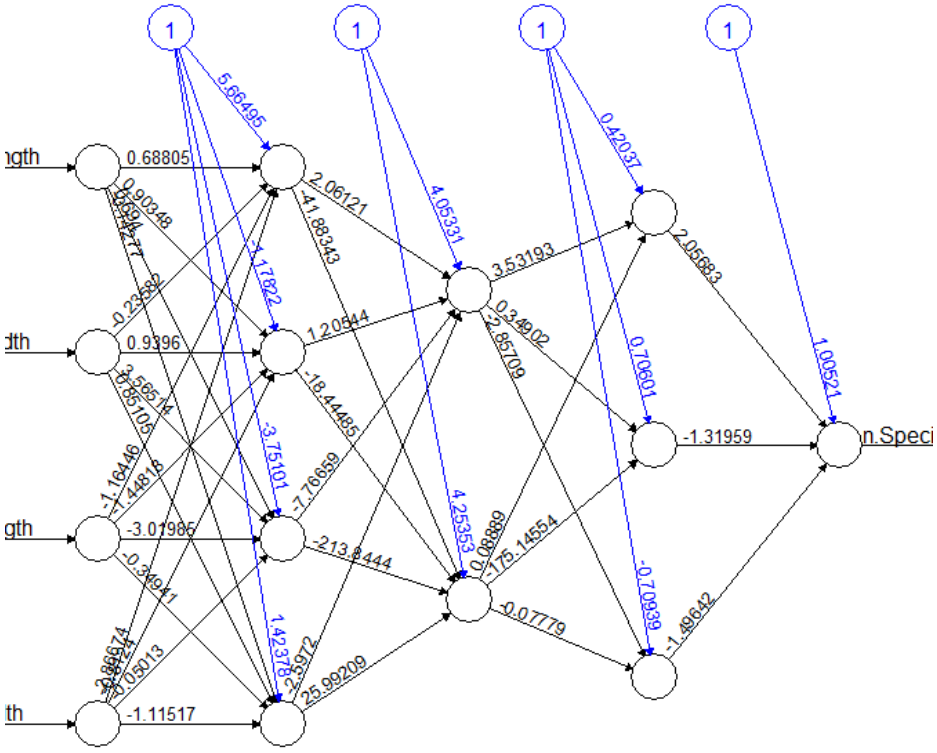
Total Observations in Table: 150

	actual			
predicted	setosa	versicolor	virginica	Row Total
setosa	50	0	0	50
	1.000	0.000	0.000	
versicolor	0	49	1	50
	0.000	0.980	0.020	
virginica	0	1	49	50
	0.000	0.020	0.980	
Column Total	50	50	50	150
	0.333	0.333	0.333	

```
> library(devtools)
> source_url('https://gist.githubusercontent.com/Peque/
+ 41a9e20d6687f2f3108d/raw/
+ 85e14f3a292e126f1454864427e3a189c2fe33f3/
+ nnet_plot_update.r')
SHA-1 hash of file is
bf3c7b8ac910823b729e3ce73bb6ab5e6955ad3d
> windows()
> plot.nnet(model)
Loading required package: scales
Loading required package: reshape
Loading required package: reshape
       :                    :
Loading required package: reshape
There were 19 warnings (use warnings() to see them)
> detach("package:devtools", unload=TRUE)
```




```
> library(neuralnet)
Warning message:
package 'neuralnet' was built under R version 3.5.1
> n.iris.data<-cbind(iris.data,
n.Species=as.numeric(iris.data$Species))
> formula<-paste0("n.Species~",
+             paste(names(n.iris.data[,-c(5,6)]), collapse="+"))
> set.seed(12345)
> model<-neuralnet(formula, data=n.iris.data,
+                                 hidden=c(4,2,3))
> # 신경망 모형
> plot(model)
```




```
> # 예측
> pred<-round(model$net.result[[1]],0)
> pred<-factor(pred, levels=1:3,
+              labels=c("setosa", "versicolor", "virginica"))
> # 정확도
> library(gmodels)
Warning message:
package 'gmodels' was built under R version 3.5.1
> CrossTable(pred, n.iris.data$Species, prop.chisq=FALSE,
+            prop.t=FALSE, prop.r=FALSE,
+            dnn=c('predicted', 'actual'))
> detach("package:gmodels", unload=TRUE)
> acc<-mean(pred==n.iris.data$Species) # 분류율
> acc
[1] 0.9933333333
> detach("package:neuralnet", unload=TRUE)
```

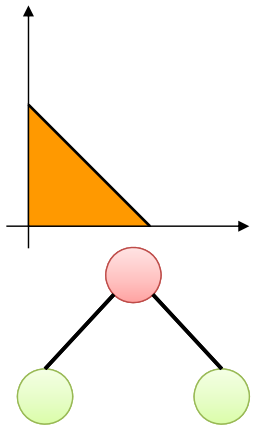
Cell Contents

N
N / Col Total

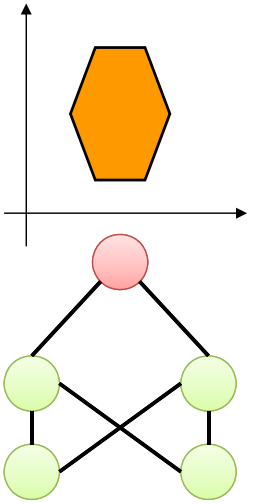
Total Observations in Table: 150

predicted \ actual	setosa	versicolor	virginica	Row Total
setosa	50 1.000	0 0.000	0 0.000	50
versicolor	0 0.000	49 0.980	0 0.000	49
virginica	0 0.000	1 0.020	50 1.000	51
Column Total	50 0.333	50 0.333	50 0.333	150

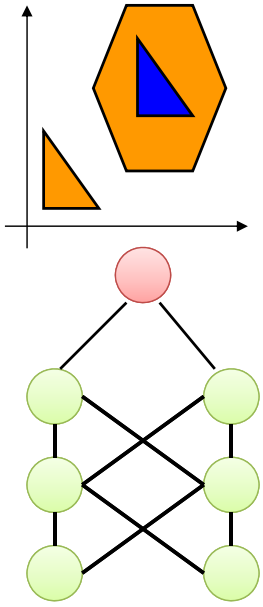
- **신경망 모형은 여러 개의 은닉층을 가질 수 있음**
  - **네트워크 구조와 의사결정계**
    - **분계점(threshold) 활성함수**



1st layer draws linear boundaries



2nd layer combines the boundaries

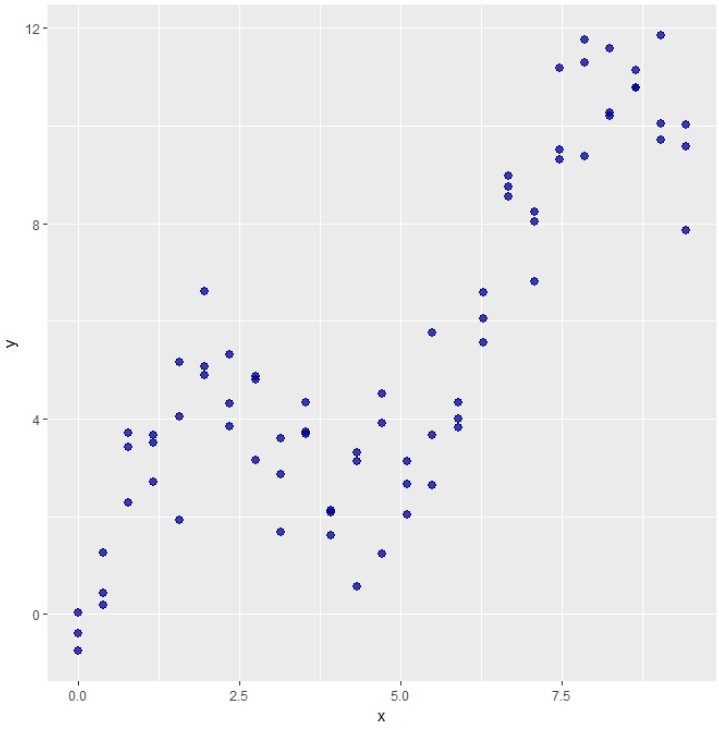


3rd layer can generate arbitrarily complex boundaries

- 은닉층의 수는 의사결정경계를 정하는데 중요
  - 은닉층의 수결정
    - 다층신경망은 단층신경망에 비해 훈련 어려움
    - 시그모이드 활성함수를 가지는 2개 층의 네트워크(1개 은닉층)는 임의의 의사결정경계를 모형화 가능
  - 각 층의 노드 수 결정
    - 출력층 노드 수는 출력 범주의 수
    - 입력층 노드 수는 입력 차원의 수
    - 은닉층 노드 수는 너무 적으면 복합한 의사결정계를 만들 수 없고, 너무 많으면 일반화가 어려움

예제 2.9

- 독립변수(x)와 종속변수(y)의 관계를 선형회귀모형과 SVM 모형, 신경망 모형에 적합시켜 보시오.



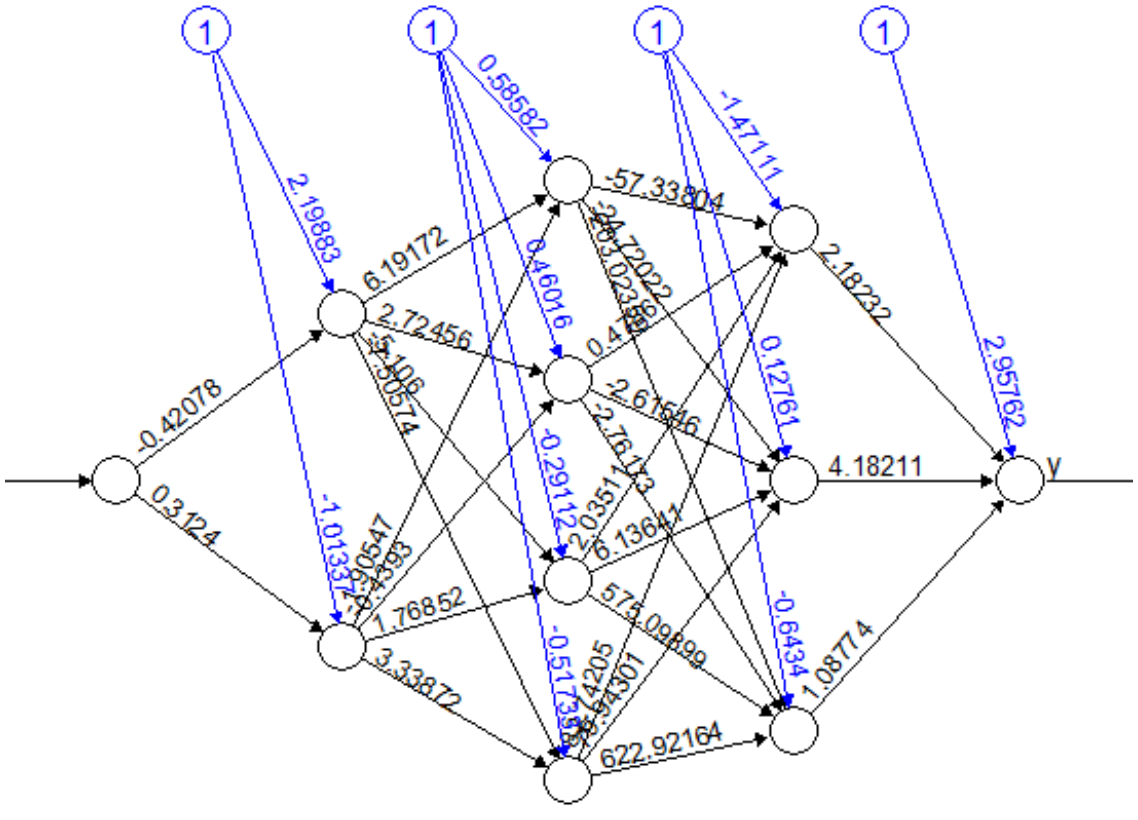
```
> # 데이터 불러오기
> reg.exam<-read.csv("reg_exam.csv", header=T)
>
> # 산점도
> library(ggplot2)
Warning message:
패키지 'ggplot2'는 R 버전 3.4.3에서 작성되었습니다
> windows()
> scap<-ggplot(data=reg.exam, aes(x=x, y=y))
> scap+geom_point(alpha=0.75, size=2.5,
+                                 colour="darkblue")
> detach("package:ggplot2", unload=TRUE)
```

```
> # 선형회귀모형
> lm.model<-lm(y~x, data=reg.exam)
> lm.model

Call:
lm(formula = y ~ x, data = reg.exam)

Coefficients:
(Intercept)            x
     0.6056       1.0094

> pred1<-predict(lm.model)
>
> # MSE
> mean((reg.exam$y-pred1)^2)
[1] 3.769396
```

```
> # SVM 모형
> library(e1071)
Warning message:
패키지 'e1071'는 R 버전 3.4.4에서 작성되었습니다
> tuned<-tune(svm, y~x, data=reg.exam,
+                     type="eps-regression",
+       ranges=list(gamma=10^seq(-1,-0.001,length=11),
+       epsilon=0.1^seq(1.5,1.1,length=11))) # 모수조정
> tuned
Parameter tuning of 'svm':
- sampling method: 10-fold cross validation
- best parameters:
     gamma     epsilon
 0.9977001 0.06606934
- best performance: 1.175244
```

```
> hp<-as.numeric(tuned$best.parameters) # 최적모수
> model<-svm(y~x, data=reg.exam, gamma=hp[1],
+                     epsilon=hp[2])  # 최종모형
> model
Call:
svm(formula = y ~ x, data = reg.exam, gamma = hp[1],
epsilon = hp[2])

Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.9977001
    epsilon:  0.06606934

Number of Support Vectors:  67
```

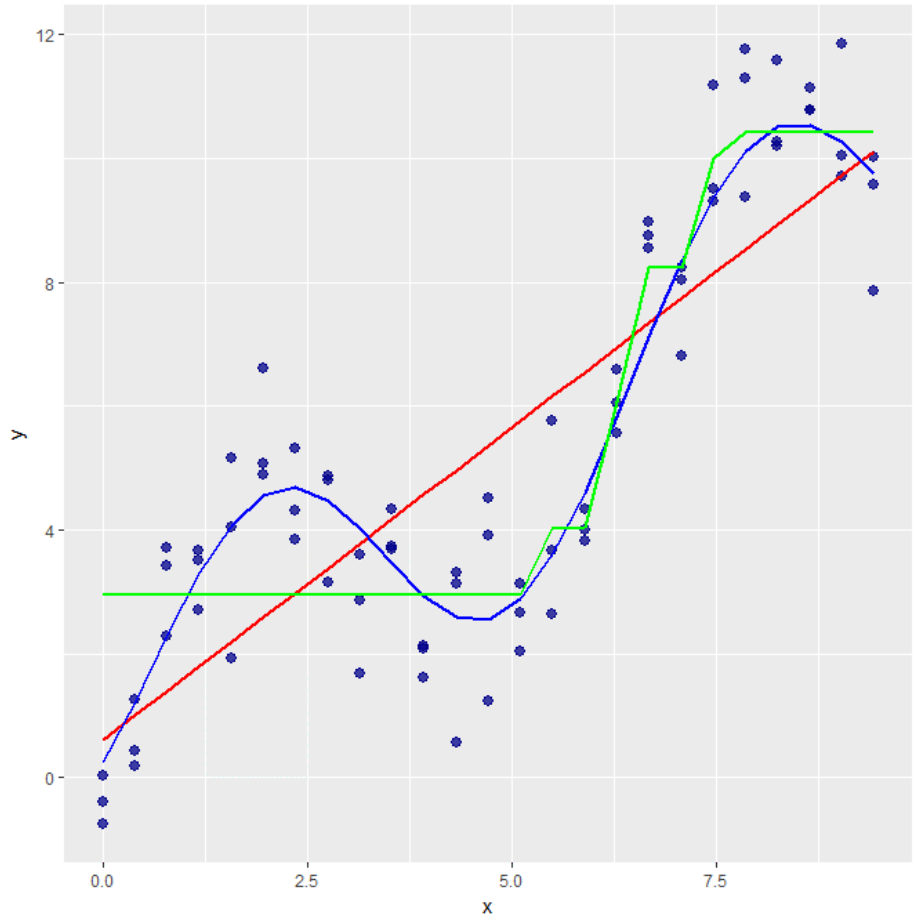
```
> pred2<-predict(model)  # 예측
>
> # MSE
> mean((reg.exam$y-pred2)^2)
[1] 1.055606
> detach("package:e1071", unload=TRUE)
>
> # 신경망 모형
> library(neuralnet)
Warning message:
패키지 'neuralnet'는 R 버전 3.4.4에서 작성되었습니다
> set.seed(23456)
> n.model<-neuralnet(y~x, data=reg.exam,
+                    hidden=c(2,4,3), stepmax=1.5e+5)
> # 신경망 모형 시각화
> plot(n.model)
```



Error: 72.411897 Steps: 32440

```
> # 예측
> pred3<-n.model$net.result[[1]]
> detach("package:neuralnet", unload=TRUE)
>
> # MSE
> mean((reg.exam$y-pred3)^2)
[1] 1.930983925
```

```
> # 산점도 및 추정직선
> library(ggplot2)
Want to understand how all the pieces fit together? Buy the ggplot2
book: http://ggplot2.org/book/
Warning message:
패키지 'ggplot2'는 R 버전 3.4.3에서 작성되었습니다
> windows()
> scap<-ggplot(data=reg.exam) # 데이터와 변수 할당
> scap+geom_point(aes(x=x, y=y), alpha=0.75, size=2.5,colour="darkblue")+
+    geom_line(aes(x=x, y=pred1), colour="red", cex=1)+
+    geom_line(aes(x=x, y=pred2), colour="blue", cex=1)+
+    geom_line(aes(x=x, y=pred3), colour="green", cex=1)
> detach("package:ggplot2", unload=TRUE)
```


12
8
4
0
y
0.0
2.5
5.0
7.5
x