6강: Support Vector Machines(SVM)

2. Naive Bayes

2.1 Multi-variate Bernoulli event model (skipped in Spring 2019 offering)

(1) Motivation

지금까지 GDA에서는 input feature x가 continuous, real-valued vector인 경우를 다뤘다. 이번에는 feature x가 discrete-value인 경우를 생각해보자. 그 예시로는 **text classification** 문제가 있다.

(2) Text classification problem

우리는 여러개의 단어로만 이루어진 이메일이 스팸 메일인지 아닌지를 분류해야 한다. 따라서 ouput y는 스팸메일이면 1, 아니면 0인 binary값을 갖는다.

- **feature vector x**

먼저 우리는 email을 **feature vector** x로 mapping 해주어야 한다.

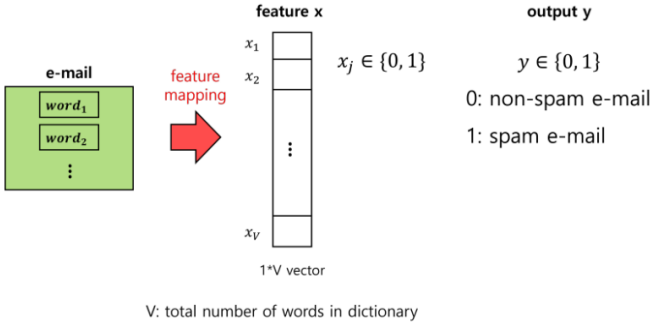
여기서 input x는 d * 1 vector이고, d는 dictionary에 있는 모든 단어의 개수이다.

x는 0, 1로 구성되며 이는 단어의 포함여부를 나타낸다. 만약 email에 dictionary의 j번째 단어를 포함한다면 $x_j = 1$, 포함하지 않는다면 $x_j = 0$이다.

$$x = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \text{a} \\ \text{aardvark} \\ \text{aardwolf} \\ \vdots \\ \text{buy} \\ \vdots \\ \text{zygmurgy} \end{matrix}$$

위의 vector x는 email을 나타내는데 “a”와 “buy” 단어는 포함하지만 “aardvark”, “aardwolf”, ”zygmurgy” 단어는 포함하지 않는다.

위의 과정을 한번에 그림으로 정리해보자



(3) Probabilistic Assumption

따라서 만약 사전에 50000개의 단어가 있다면 x는 50000*1 vector이고 2^{50000}개의 output을 가질 수 있는 multinomial distribution 모델이 나올 것이다. 여기에서는 $(2^{50000}-1)$차원의 parameter vector가 존재한다.

이때 문제는 parameter가 너무 많다는 것이다. 따라서 $p(x|y)$ 모델을 세울 때, strong assumption을 추가하게 된다. **given y일 때, x_i끼리는 conditionally independent하다는 것이다.** 이 가정을 **Naive Bayes assumption**이라고 하고, 이를 바탕으로 하는 알고리즘을 **Naive Bayes Classifier**라고 한다.

이 가정을 말로 설명하면 다음과 같다. y=1(a particular piece of email is spam)이라고 주어졌을 때, email에 “buy”라는 단어(x_{2087})가 나타나는지에 대한 정보는 “price”라는 단어(x_{39831})가 나타나는지에 영향을 미치지 않는다는 것이다.

◘

💥주의: x_{2087}와 x_{39831}가 independent하다는 것과는 다르다. 이것은 $p(x_{2087}) = p(x_{2087}|x_{39831})$ 로 나타낸다.

-

이제 가정을 적용하여 다음과 같이 나타낼 수 있다.

$$
\begin{aligned}
& p(x_1, \ldots, x_{50000}|y) \\
&= p(x_1|y)p(x_2|y,x_1)p(x_3|y,x_1,x_2)\cdots p(x_{50000}|y,x_1,\ldots,x_{49999}) \\
&= p(x_1|y)p(x_2|y)p(x_3|y)\cdots p(x_{50000}|y) \\
&= \prod_{j=1}^{d} p(x_j|y)
\end{aligned}
$$

$$
\begin{aligned}
& p(x_1, \ldots, x_d|y) \\
&= p(x_1|y)p(x_2|y,x_1)p(x_3|y,x_1,x_2)\cdots p(x_d|y,x_1,\ldots,x_d)) \\
&= p(x_1|y)p(x_2|y)p(x_3|y)\cdots p(x_d|y) \\
&= \prod_{j=1}^{d} p(x_j|y)
\end{aligned}
$$

참고로 Naive Bayes assumption이 stong assumption임에도 불구하고 이를 바탕으로 한 알고리즘은 다양한 문제에서 잘 작동한다고 한다.

- parameters

y가 given일 때, x_j 단어가 포함되는지 여부는 Bernoulli distribution으로 모델링할 수 있다.

y가 가질 수 있는 값이 {0, 1}의 두 가지이므로, 각각의 경우일 때 x_j의 Bernoulli distribution에 대한 parameter, 그리고 y도 Bernoulli distribution으로 모델링되므로 ϕ_y가 있다.

또한 우리는 $p(x_j|y)$는 모든 j값에 대해 같다고 가정했다.

(4) MLE

- 주어진 데이터 셋에 대한 joint likelihood function:

$$\begin{aligned}\mathcal{L}(\phi_y, \phi_{k|y=0}, \phi_{k|y=1}) &= \prod_{i=1}^{n} p(x^{(i)}, y^{(i)}) \\ &= \prod_{i=1}^{n} \left(\prod_{j=1}^{d_i} p(x_j^{(i)}|y; \phi_{k|y=0}, \phi_{k|y=1}) \right) p(y^{(i)}; \phi_y).\end{aligned}$$

ϕ_y, $\phi_{j|y=1}$, $\phi_{j|y=0}$에 대하여 위의 function을 최대화하면 다음의 MLE 값을 얻는다.

- MLE를 통한 parameter값들:

$$\begin{aligned}\phi_{j|y=1} &= \frac{\sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\}}{\sum_{i=1}^{n} 1\{y^{(i)} = 1\}} \\ \phi_{j|y=0} &= \frac{\sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\}}{\sum_{i=1}^{n} 1\{y^{(i)} = 0\}} \\ \phi_y &= \frac{\sum_{i=1}^{n} 1\{y^{(i)} = 1\}}{n}\end{aligned}$$

$\wedge$ 는 and를 의미한다.

이 파라미터들은 직관적인 해석이 가능하다. 예를들어 $\phi_j|y = 1$는 y=1인 메일 중에서 해당 단어(word j)가 실제로 포함되었던 비율이다.

- prediction

새로운 example인 feature x에 대해 어떻게 예측할 수 있을까?

$$\begin{aligned}p(y = 1|x) &= \frac{p(x|y = 1)p(y = 1)}{p(x)} \\ &= \frac{\left(\prod_{j=1}^{d} p(x_j|y = 1)\right) p(y = 1)}{\left(\prod_{j=1}^{d} p(x_j|y = 1)\right) p(y = 1) + \left(\prod_{j=1}^{d} p(x_j|y = 0)\right) p(y = 0)},\end{aligned}$$

fit된 파라미터에 대해서 위와 같은 계산을 하면 된다. 그리고 가장 높은 사후확률을 가지는 class를 선택한다.

2.2 Laplace smoothing

그러나 위의 방법은 문제가 존재하는데,, 지금까지 한번도 관측되지 않은 단어에 대해 생각해보자. 이 단어는 “neurips”이고 사전의 35000번째 단어이다. 너의 naive bayes spam filter는 $\phi_{35000|y}$라는 파라미터의 MLE를 다음과 같이 얻는다.

$$\begin{aligned}\phi_{35000|y=1} &= \frac{\sum_{i=1}^{n} 1\{x_{35000}^{(i)} = 1 \wedge y^{(i)} = 1\}}{\sum_{i=1}^{n} 1\{y^{(i)} = 1\}} = 0 \\ \phi_{35000|y=0} &= \frac{\sum_{i=1}^{n} 1\{x_{35000}^{(i)} = 1 \wedge y^{(i)} = 0\}}{\sum_{i=1}^{n} 1\{y^{(i)} = 0\}} = 0\end{aligned}$$

즉, “neurips” 단어는 이전에 spam , non-spam training examples에서 본적이 없기 때문에 확률이 0으로 나온 것이다.

그러므로, “neurips”단어가 포함된 메시지중 하나가 spam인지 결정할 때, class의 사후 확률을 계산하면 다음과 같이 0/0의 결과를 얻는다.

$$\begin{aligned}p(y = 1|x) &= \frac{\prod_{j=1}^{d} p(x_j|y = 1)p(y = 1)}{\prod_{j=1}^{d} p(x_j|y = 1)p(y = 1) + \prod_{j=1}^{d} p(x_j|y = 0)p(y = 0)} \\ &= \frac{0}{0}.\end{aligned}$$

이러한 단점을 보완하는 방법이 **Laplace smoothing** 이다. 이 아이디어는 MLE를 구할 때, 총 K개의 값을 가질 수 있는 x의 parameter ϕ_j에 대한 estimation에서 분모에는 k, 분자에는 1을 더해주는 것이다.

$$\phi_j = \frac{\sum_{i=1}^{n} 1\{z^{(i)} = j\}}{n}.$$

$$\phi_j = \frac{1 + \sum_{i=1}^{n} 1\{z^{(i)} = j\}}{n + k}.$$

이를 우리의 naive bayes classifier에 적용하면, x가 가질 수 있는 값은 {0,1}로 2가지 이므로,

$$\begin{aligned}\phi_{j|y=1} &= \frac{1 + \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\}}{2 + \sum_{i=1}^{n} 1\{y^{(i)} = 1\}} \\ \phi_{j|y=0} &= \frac{1 + \sum_{i=1}^{n} 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\}}{2 + \sum_{i=1}^{n} 1\{y^{(i)} = 0\}}\end{aligned}$$

이다.

여기서 ϕ_y에 대해서는 laplace smoothing 을 적용하든 말든 상관없다. 왜냐면 일반적으로 위의 예시에서 spam메일의 분포가 어느 한 쪽으로 치우치는 극단적인 분포는 아닐 것이기 때문이다.

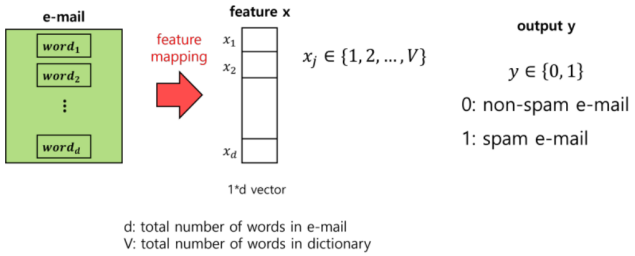
2.3 Multinomial event model for text classification

(1) Multinomial event model

앞에서의 bernoulli event model에서 feature vector x의 차원은 항상 고정되어 있었고, 그 차원은 사전에 포함된 총 단어의 수와 같았다. 그리고 각각의 단어 x_j는 bernoulli r.v이고, 이들은 y가 given일 때 contionally indepedent하다.

그러나 multinomial event model은

- feature vector x의 차원은 하나의 메일에 포함된 단어의 수이고,
- 각각의 element(=메일에 포함된 하나의 단어)는 dictionary에서의 index를 값을 갖는다.
- 각 element는 mutinomial r.v이고 1부터 V까지 중 하나의 값을 갖는다. (V= 사전에 포함된 모든 단어의 수)



(2) Assumption

💥+ naive bayes assumption

확률 모델의 가정은 먼저 y가 bernoulli distribution 확률에 의해 정해지고, y가 given 상태에서 각각의 x_j들은 서로 indepedent하고 identical한 mutinomial distribution확률에 의해 정해진다는 것이다.

$$y \sim Bernoulli(\phi_y)$$

$$x_j|y \sim multinomial(\phi_{1|y}, \ldots, \phi_{V|y})$$

(2) MLE

- probability

$$\prod_{i=1}^{m} p\left(x^{(i)}, y^{(i)}\right)$$

$$= \prod_{i=1}^{m} p\left(x^{(i)} | y^{(i)}\right) \cdot p\left(y^{(i)}\right)$$

$$= \prod_{i=1}^{m} \prod_{j=1}^{d} p\left(x_j^{(i)} | y^{(i)}\right) \cdot p\left(y^{(i)}\right)$$

- parameters

◗

$$\phi_{k|y=1} = p(x_j = k|y = 1)$$

$$\phi_{k|y=0} = p(x_j = k|y = 0)$$

- MLE

파라미터를 MLE로 구한 결과는 다음과 같다.

$$\varphi_{k|y=0} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{d_i} 1\left\{y^{(i)}=0 \text{ and } x_j^{(i)}=k\right\}}{\sum_{i=1}^{m} 1\left\{y^{(i)}=0\right\}\cdot d_i}$$

$$\varphi_{k|y=1} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{d_i} 1\left\{y^{(i)}=1 \text{ and } x_j^{(i)}=k\right\}}{\sum_{i=1}^{m} 1\left\{y^{(i)}=1\right\}\cdot d_i}$$

$$\varphi_{y} = \frac{\sum_{i=1}^{m} 1\left\{y^{(i)}=1\right\}}{m}$$

2.3.1 Laplace smoothing: a self-contained introduction

여기에 Laplace smoothing을 적용하면, x_j가 가질 수 있는 값이 V개 이므로, 분모에는 V, 분자에는 1을 더하면 다음과 같다.

$$\varphi_{k|y=0} = \frac{1+\sum_{i=1}^{m}\sum_{j=1}^{d_i} 1\left\{y^{(i)}=0 \text{ and } x_j^{(i)}=k\right\}}{V+\sum_{i=1}^{m} 1\left\{y^{(i)}=0\right\}\cdot d_i}$$

$$\varphi_{k|y=1} = \frac{1+\sum_{i=1}^{m}\sum_{j=1}^{d_i} 1\left\{y^{(i)}=1 \text{ and } x_j^{(i)}=k\right\}}{V+\sum_{i=1}^{m} 1\left\{y^{(i)}=1\right\}\cdot d_i}$$

- prediction

- 마무리

지금까지 naive bayes classifier의 두가지 모델인 Bernoulli event model과 multinomial event model을 살펴보았다. naive bayes classifier는 best classification algorithm은 아니지만, simplicity & ease of implementation라는 점에서는 “first thing to try”이다…..!

Support Vector Machine

<이번강의>

1. optimal margin classifier (seperable case)

<다음강의>

2. kernels

3. inseperable case

(1) concept overview

-logistic regression에서의 hypothesis :

$h_\theta(x) = g(\theta^T x)$
$g(z) = \frac{1}{1+e^{-z}}$

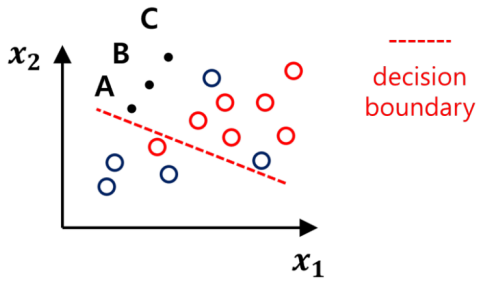
-새로운 input x가 주어졌을 때, g(z)가 0.5이상이면 1, 미만이면 0으로 predict한다.

$$\begin{matrix} 1 & if\ \theta^T x \geq 0 \\ 0 & if\ \theta^T x < 0 \end{matrix}$$

이것을 다르게 표현하면, 다음과 같다.

$if\ y^{(i)} = 1,\ want\ \theta^T x \gg 0$
$if\ y^{(i)} = 0,\ want\ \theta^T x \ll 0$

svm의 중요한 개념인 **margin**은 이러한 관점에서 직관적으로 이해할 수 있다. label 1데이터에 대해서는 가능한 θ, x의 내적이 0보다 클수록, label 0데이터에 대해서는 가능한 θ, x의 내적이 0보다 작을 수록 더 좋은 모델이다. 더 작고, 더 클수록 예측 정확도는 높아진다.



위와 같은 binary classification에서 decision boundary보다 가장 먼, 즉 θ, x의 내적이 0보다 가장 큰 C를 label 1라고 확실하게 예측할 것이다.

(2) notation

margin 개념을 더 수학적으로 정의하기 앞서 svm에서의 notation을 먼저 정리해보고자 한다.

- label y

문제 상황은 기본적으로 input feature vector x에 대해 label을 둘중 하나로 예측하는 binary classification 문제이다. 이때 label을 {0,1}이 아닌 {-1, 1}로 나타낸다

$label\ y \in -1,\ 1$

- hypothesis $h(x)$

logistic regression의 경우

$h_\theta(x) = g(\theta^T x) \qquad \theta^T x\ \in\ R^{n+1},\ x_0 = 1$

svm의 경우 파라미터는 weight term w, bias term b로 나누어 다음과 같이 나타낸다.

$h_{w,b}(x) = g(w^T x + b) \qquad w^T x \ \in \ R^n, \ b \in R$

- output

또한 logistic regression에서 sigmoid fuction을 사용하여 y=1일 확률을 나타낸 뒤 0.5보다 크거나 같으면 1, 작으면 0으로 predict했던 것 대신에, directly 1 또는 -1의 값을 output로 하는 g(z)로 바뀌었다.

$have \ h \ ouput \ value \ in \ \{-1, \ 1\}$

$g(z) = 1 \qquad z \geq 0$
$\quad\ -1 \qquad z < 0$

Functional Margin& Geometric Margin

(1) Functional Margin

하나의 training data $(x^{(i)}, y^{(i)})$에 대한 functional margin은 다음과 같이 정의한다.

$\hat{\gamma}^{(i)} = y^{(i)}(w^T x^{(i)} + b)$

위에서 논의했던 것처럼 좋은 모델은 기본적으로 functional margin을 가장 크게 갖는 모델이다.

따라서

if $y^{(i)} = 1$, want $w^T x^{(i)} + b \gg 0$

if $y^{(i)} = -1$, want $w^T x^{(i)} + b \ll 0$

we hope $\hat{\gamma} \gg 0$

를 만족해야 하고

$\hat{\gamma} > 0$ 은 곧 $h(x^{(i)}) = y^{(i)}$를 의미한다.

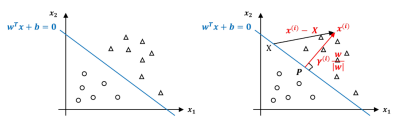
이제 주어진 training 데이터 셋에 대한 **functional margin**은 각 training 데이터의 functional margin중 최솟값으로 정의한다.

$\hat{\gamma}^{(i)} = min_{i=1,...m} \hat{\gamma}^{(i)}$

그런데 이 functional margin의 하나의 문제점은 w, b의 scale에 대해 freedom이 존재한다는 것이다. 만약 w,b에 상수 k만큼을 곱한다고 하더라도 여전히 decision boundary는 같다. 하지만 functional margin의 값은 k만큼 증가하므로, 이 경우 functional margin이 모델의 confidence를 정확하게 나타낸다고 볼 수 없다. 따라서 normalization condition을 추가적으로 두어야 한다.

$$(w, \ \ b) \rightarrow \left(\frac{w}{|w|}, \ \ \frac{b}{|w|} \right)$$

(2) geometric margin



이는 x가 2차원 벡터일 때 decision bounary $w^T x + b = 0$를 나타낸 것이다. training example 중 하나인 $x^{(i)}$에서의 geometric margin은 $x^{(i)}$에서 hyperplane $w^T x + b = 0$까지의 거리이다.

수식 유도를 생략하면 하나의 training example에 대한 geometric margin은 다음과 같다.

,,,,선대 내용 추후에 적어놓을게요!

$$\gamma^{(i)} = \frac{w^T x^{(i)} + b}{|w|}$$

주어진 전체 training set에 대한 geometric margin은 모든 example 중 minimum으로 정의된다.

$$\gamma = \min_{i=1, \dots, m} \gamma^{(i)}$$

(3) optimization problem

위에서 정의한 geometric margin개념을 이용하여 optimal classifier를 찾는다는 것은 geometric margin을 maximize 시키는 decision boundary를 찾는 것이다.

우선 given training set이 lineary seperable하다고 가정하자. 이제 optimization problem은 다음과 같이 정리된다.

$$\max_{\gamma, w, b} \gamma$$

$$s.t \quad \frac{y^{(i)}\left(w^T x^{(i)} + b\right)}{|w|} \geq \gamma \quad for \quad i = 1 \quad to \quad m$$

이를 minimization 문제로 바꾸면,

,,, 유도 생략,,, 지윤이 짱,,,

$$\min_{w, b} \frac{1}{2}|w|^2$$

$$s.t \quad y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1 \quad for \quad i = 1 \quad to \quad m$$

와 같은 optimization 문제로 바뀐다. 이렇게 수식을 유도한 이유는 이것이 “”convex optimization problem””이기 때문이며 이를 푸는 tool은 뒤에서 설명할 예정이다…!