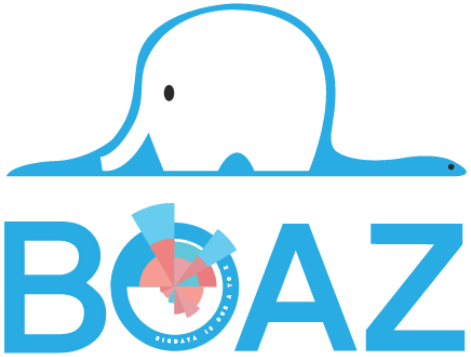


분류를 위해 확률의 원리를 활용하는 'Naive Bayes'기계학습 알고리즘

B조 - 홍예림, 이지연, 정재욱, 조대연

발표자: 정재욱, 홍예림, 조대연

2016. 11.03

오늘 함께 공부할 내용은,

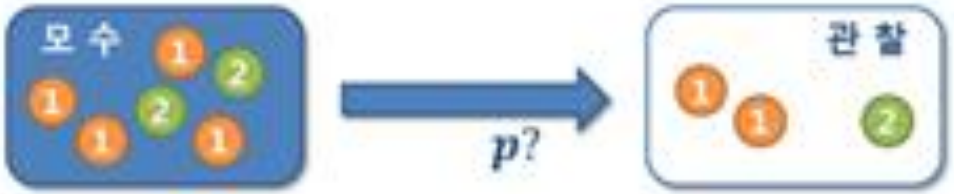
Part 1 _ 이론

- 확률 이론 기초
- 나이브 베이즈 이론 접근
- 나이브 베이즈 알고리즘 특징 및 장단점
- 라플라스 추정기

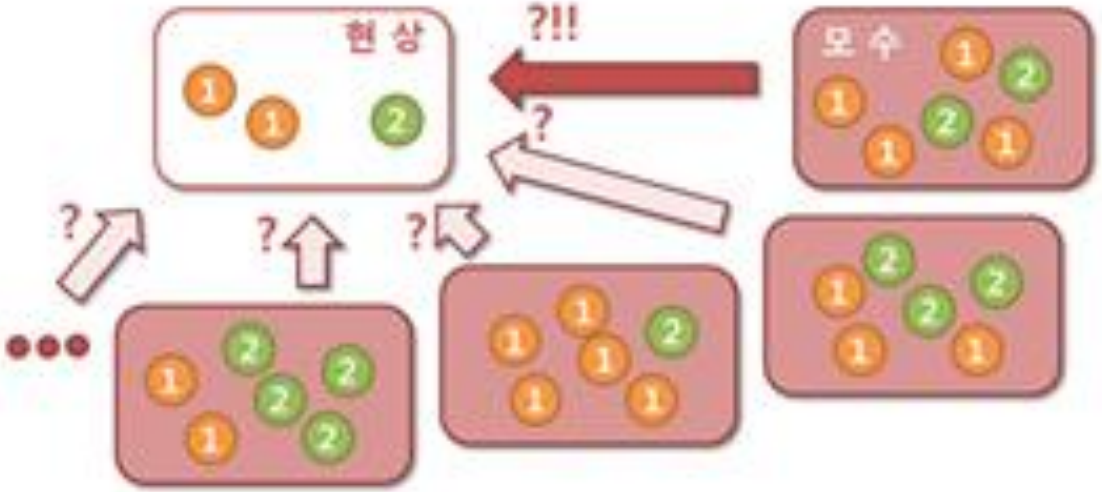
Part 2 _ 실습

- R 실습
- Python 실습

A. **확률(probability)**: 모수로부터 다음과 같이 관찰될 **확률**은?



B. **우도(likelihood)**: 현상에 대해 가장 가능성이 높은(**우도**가 높은) 모수는?



Maximum Likelihood Estimation (MLE)

random variable의 parameter를 estimate하는 방법 중 하나인데, 오직 주어진 Observation, 혹은 데이터들 만을 토대로 parameter estimation을 하는 방법

$$\mathcal{L}(\theta; x_1, x_2, \ldots, x_n) = \mathcal{L}(\theta; X) = f(X|\theta) = f(x_1, x_2, \ldots, x_n|\theta)$$

$$\hat{\theta} = \arg\max_{\theta} \mathcal{L}(\theta; X) = \arg\max_{\theta} f(X|\theta)$$

만약 관측값들이
i.i.d. (independent and identical distributed)하다면

$$f(X|\theta) = \prod_i f(x_i|\theta)$$

Maximum a Posteriori Estimation (MAP)

이 방법은 θ가 주어지고,
그 θ에 대한 데이터들의 확률을 최대화하는 것(MLE)이 아니라,
주어진 데이터에 대해 최대 확률을 가지는 θ를 찾는다

$$\hat{\theta} = \arg\max_{\theta} f(\theta|X)$$

MAP를 계산하기 위해서는 f(θ|X)가 필요하지만 우리가 관측할 수 있는 것은 오직 f(X|θ)뿐이다.
f(θ|X)를 구하기 위해서는 Bayes' Theorem 필요하다.

조건부 확률

1. 어떤 정보가 주어졌을 때 교정된 확률을 가진다.
2. 생각해야 할 표본 공간이 줄어든다.

- 조건부 확률 : **어떤 사건이 주어졌을 때 다른 사건이 발생할 확률**

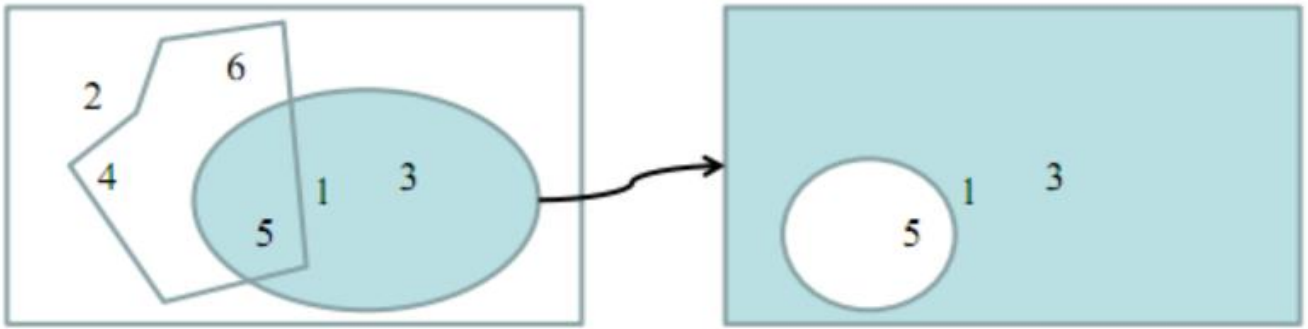
$$P(B \mid A) = \frac{P(A \cap B)}{P(A)}, \quad (\text{단}, P(A) > 0)$$

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

(예) 주사위를 한 번 던지는 시행에서

B=홀수가 나오는 사건, A=4이상의 눈이 나오는 사건

P(A) = 3/6, P(A|B) = 1/3



"B가 일어났다"는 조건 → B를 표본 공간으로 간주

(2) 확률의 분할법칙

- $P(B) = P(B \cap A) + P(B \cap A^c)$
  $= P(B|A)P(A) + P(B|A^c)P(A^c)$

(3) 전확률 공식 (분할법칙의 확장)

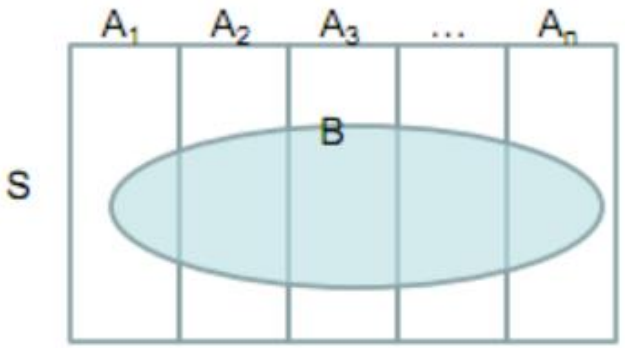
When,

$A_i \cap A_j = \varnothing \ (i \neq j)$ → ME

$A_1 \cup A_2 \cup \ldots \cup A_n = S$ → CE

$P(A_i) > 0$

→ $P(B) = P(B \cap A_1) + \ldots + P(B \cap A_n)$
$= P(B|A_1)P(A_1) + \ldots + P(B|A_n)P(A_n)$



Basic concepts of Bayesian methods

베이즈 이론 (Bayesian theory) : 주어진 사전확률과 조건부 확률을 이용하여 다른 조건부 확률을 구하고자 하는 경우

한 개의 속성인 경우,

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

$P(A)$ 는 사건 A의 사전확률, 아직 사건 B에 관한 어떠한 정보도 알지 못하는 것을 의미
$P(A|B)$ 는 B의 값이 주어진 경우에 대한 A의 사후 확률, 사건 B에 대한 정보에 의존
$P(B|A)$ 는 A의 값이 주어진 경우 B의 우도(likelihood)
$P(B)$ 는 주변우도(marginal likelihood). 확률값을 한정시키는 역할

* 이때 A는 불확실성을 계산해야 하는 대상이며, B는 관측하여 값을 알아낼 수 있는 대상으로 생각한다면,
A의 확률은 B가 관측된 후 P(A)에서 P(A|B)로 변화하며, 베이즈 정리는 이 때의 변화를 계산하는 방법을 제공한다.

다시 MLE와 MAP로 돌아가보자. Bayes' Theorem을 사용하면 MAP와 MLE의 관계를 다음과 같이 적을 수 있다.

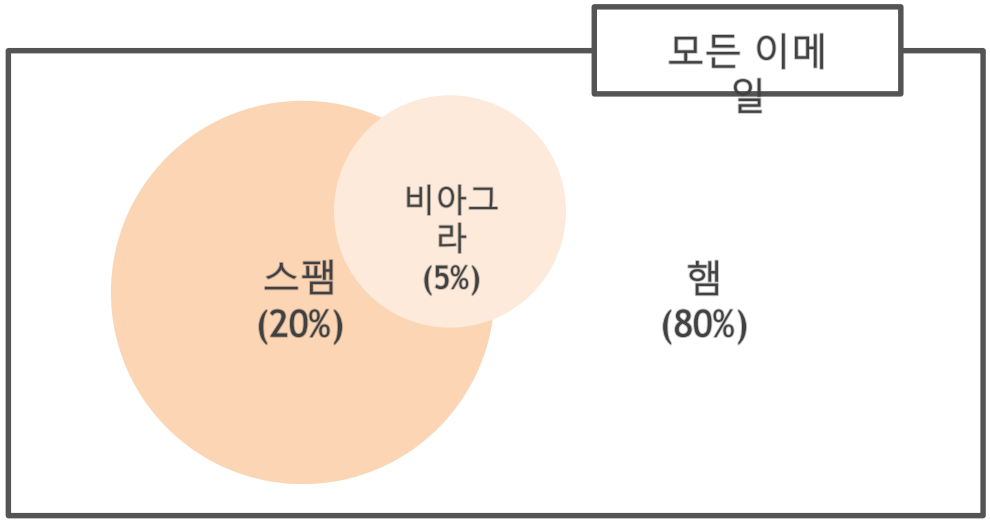
$$\hat{\theta} = \arg\max_{\theta} f(\theta|X) = \arg\max_{\theta} \frac{f(X|\theta)f(\theta)}{f(X)} = \arg\max_{\theta} \frac{\mathcal{L}(\theta;X)f(\theta)}{f(X)}$$

이때, $f(X)$ term은 θ에 영향을 받는 term이 아니기 때문에 다음과 같이 적을 수 있다.

$$\hat{\theta} = \arg\max_{\theta} \mathcal{L}(\theta;X)f(\theta)$$

Bayes'theorem - example

한 개의 속성의 경우,



받은 이메일이 스팸일 확률을 구한다고 가정하자.
어떤 추가 증거 없이 가장 합리적인 추측은 이전 메일이 스팸인지에 대한 확률(사전확률)이다.
즉 $P(spam) = 0.2$ 라는 정보를 알 수 있다.
또한 모든 메시지에 비아그라가 나타날 확률(주변우도)는 $P(Viagra) = 5/100 = 0.05$ 임을 알 수 있다.

$$P(spam|\, Viagra)$$

Bayes'theorem - example

	비아그라		
빈도	Yes	No	총합
스팸	4	16	20
햄	1	79	80
총합	5	95	100

	비아그라		
빈도	Yes	No	총합
스팸	4/20	16/20	20
햄	1/80	79/80	80
총합	5/100	95/100	100

추가 증거를 얻었다. 비아그라 단어가 이전 스팸 메시지에서 사용되었을 확률(우도),

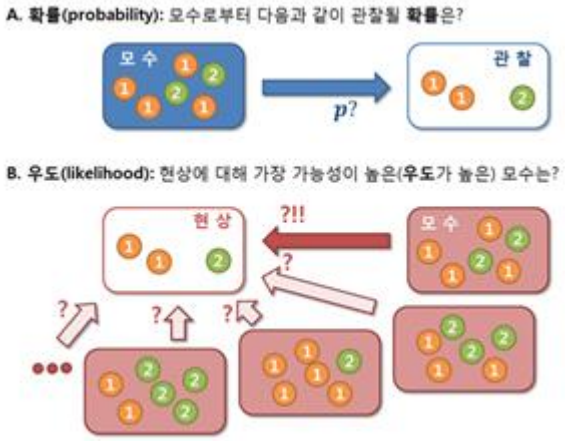
$P(Viagra \mid spam) = 4/20 = 0.2$

우도, 사전 확률

$$P(spam \mid Viagra) = \frac{P(Viagra \mid spam)P(spam)}{P(Viagra)}$$

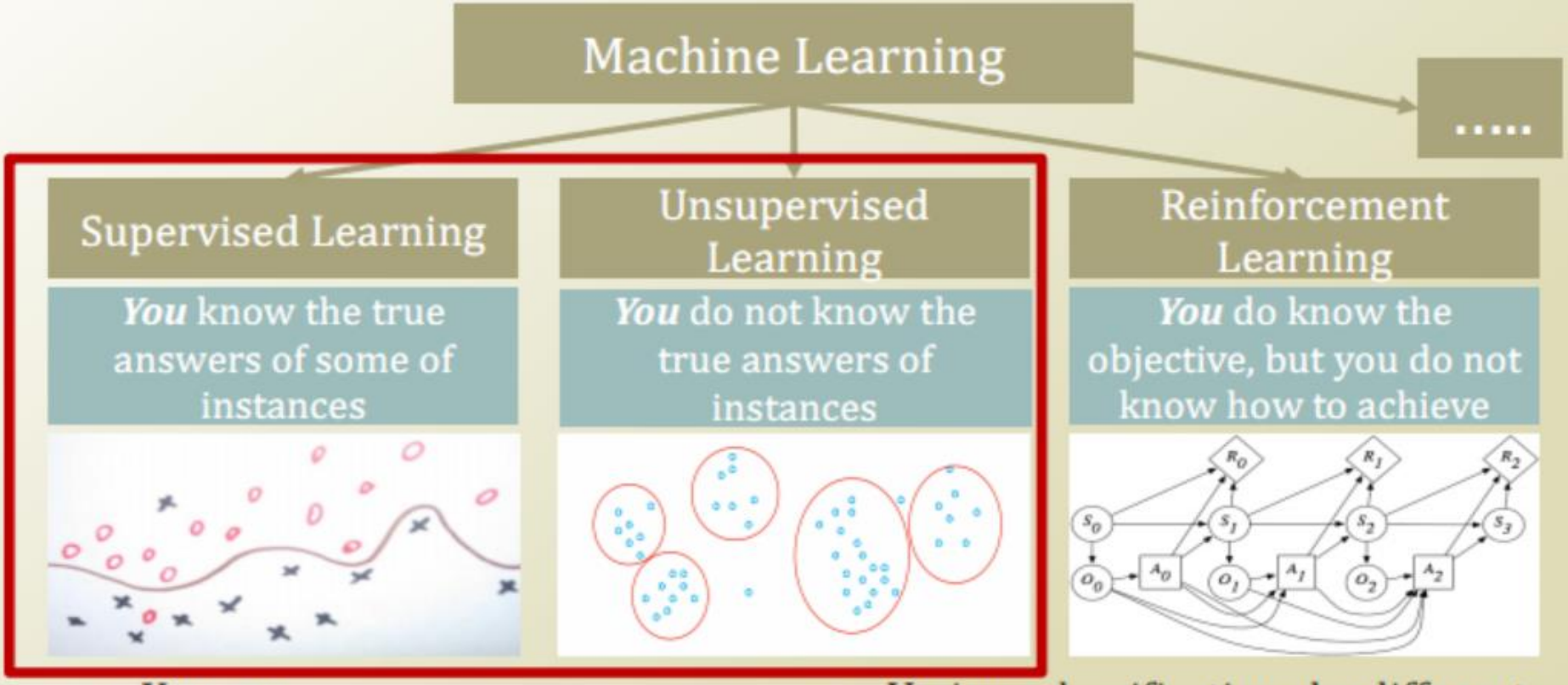
사후 확률, 주변 우도

이제 사후확률을 구해보자. $\frac{0.2*0.2}{0.05} = 0.8$



메시지에 비아그라 단어를 포함하면 메시지가 스팸일 확률은 80%이다.
따라서 이 단어가 포함된 메시지는 걸러져야한다.

Types of Machine Learning



- ***You*** can
  - Machine learning
  - Dataset provider
  - Machine learning users
  - etc

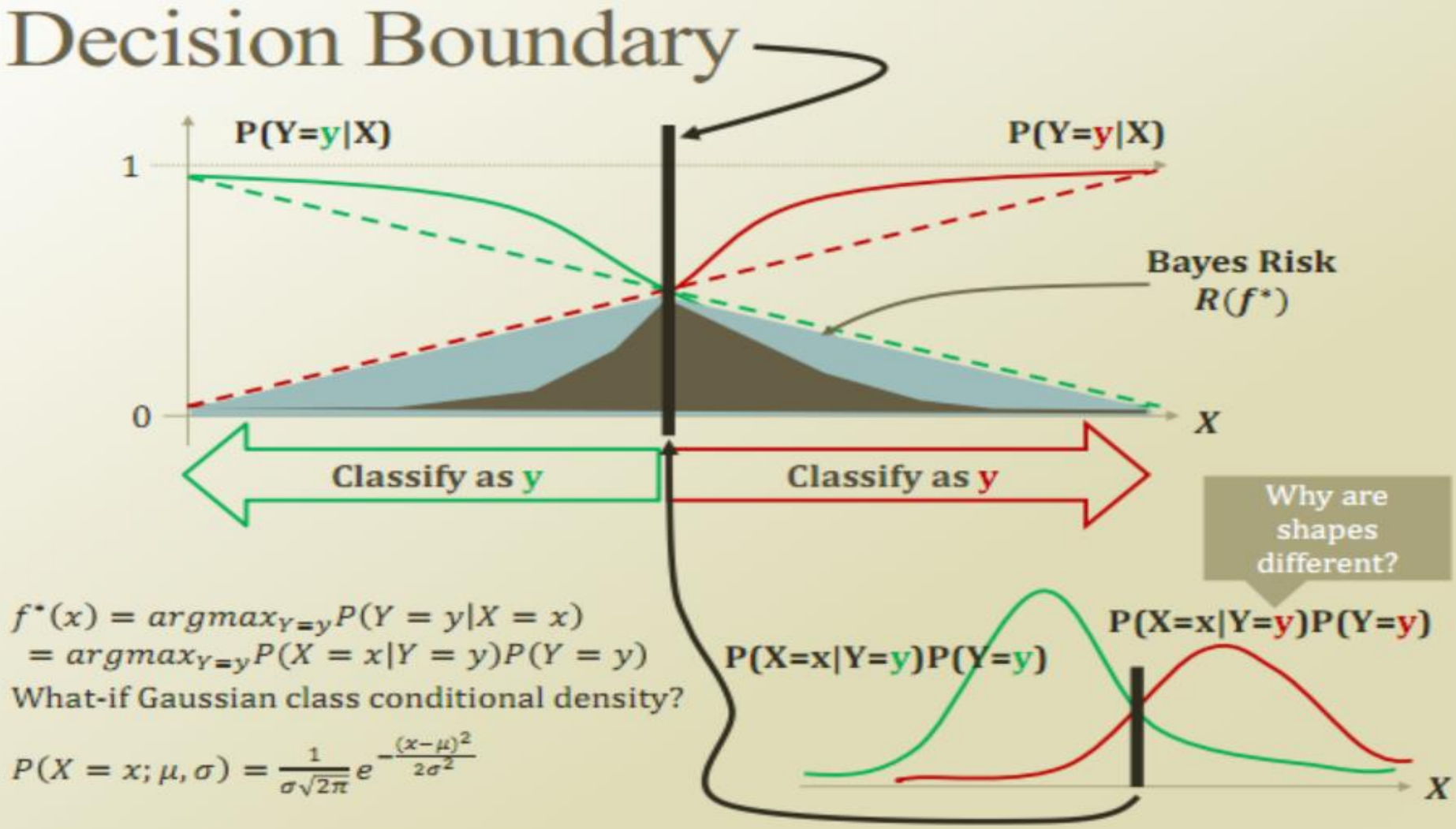
- Various classifications by different professors
  - Purpose, data types, etc
- Other learning classifications also exist

Optimal Classification

- Optimal predictor of Bayes classifier
  - $f^* = argmin_f P(f(X) \neq Y)$
  - Function approximation of error minimization
- Assuming only two classes of Y
  - $f^*(x) = argmax_{Y=y} P(Y = y|X = x)$

$$\sum_{y \in Y} P(Y = y|X = x) = ?$$

Optimal Classification



- $f^*(x) = argmax_{Y=y} P(Y = y|X = x)$
  $= argmax_{Y=y} P(X = x|Y = y)P(Y = y)$
- What-if Gaussian class conditional density?
- $P(X = x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

Dataset for Optimal Classification Learning

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

- $f^*(x) = argmax_{Y=y} P(X = x|Y = y)P(Y = y)$
  - $P(X=x|Y=y)$ $=P(x_1=\text{sunny}, x_2=\text{warm}, x_3=\text{normal}, x_4=\text{strong}, x_5=\text{warm}, x_6=\text{same}|y=\text{Yes})$
  - $P(Y=y)=(y=\text{Yes})$
- How many parameters are needed? How many observations are needed?
  - $P(X=x|Y=y)$ for all x,y — $(2^d-1)k$
  - $P(Y=y)$ for all y — $k-1$

  Often, what happens is $N >> (2^d-1)k >> |D|$
- Remember that we are not living in the perfect world!
  - Noise exists, so need to model it as a random variable with a distribution
  - Replications are needed!

Why need an additional assumption?

- $f^*(x) = argmax_{Y=y} P(X = x|Y = y)P(Y = y)$
  - To learn the above model, we need a very large dataset that is impossible to get
- The model has relaxed unrealistic assumptions, but now the model has become impossible to learn.
  - Time to add a different assumption
  - An assumption that is not so significant like the ones being relaxed
- What are the major sources of the dataset demand?
  - $P(X=x|Y=y)$ for all $x,y \rightarrow (2^d-1)k$
    - x is a vector value, and the length of the vector is d
    - d is the source of the demand
    - Then, reduce d?
    - Or, ????

Conditional Independence

- A passing-by statistician tells us
  - Hey, what if?
    - $P(X =< x_1, \dots, x_l > | Y = y) \rightarrow \prod_l P(X_l = x_l | Y = y)$
  - Your response: Is it possible?
    - Statistician: Yes! If $x_1,\dots,x_l$ are conditionally independence given y

- Conditional Independence
  - x_1 is conditionally independent of x_2 given y
  - $(\forall x_1, x_2, y) \quad P(x_1|x_2, y) = P(x_1|y)$
  - Consequently, the above asserts
    - $P(x_1, x_2|y) = P(x_1|y)P(x_2|y)$
  - Example,
    - P(Thunder|Rain, Lightning)=P(Thunder|Lightening)
    - If there is a ***lightening***, there will be a ***thunder*** with a prob. ***p*** regardless of ***raining***

Bayes'theorem

더 많은 속성의 경우,

$$
\begin{aligned}
P(C_k|x_1,\cdots,x_n) &= P(C_k)\ P(x_1,\cdots,x_n|C_k) \\
&= P(C_k)\ P(x_1|C_k)\ P(x_2,\cdots,x_n|C_k,x_1) \\
&= P(C_k)\ P(x_1|C_k)\ P(x_2|C_k,x_1)P(x_3,\cdots,x_n|C_k,x_1,x_2) \\
&= P(C_k)\ P(x_1|C_k)\ P(x_2|C_k,x_1)\cdots P(x_n|C_k,x_1,x_2,x_3,\cdots,x_{n-1})
\end{aligned}
$$

나이브 베이즈(Naive Bayes)는 **독립성**을 가정

$$
\begin{aligned}
P(C_k|x_1,\cdots,x_n) &\propto P(C_k\ ,x_1,\cdots,x_n) \\
&\propto P(C_k)\ P(x_1|C_k)P(x_2|C_k)P(x_3|C_k)\ \cdots \\
&\propto P(C_k)\prod_{i=1}^{n} P(x_i|C_k)
\end{aligned}
$$

가장 **가능성이 높은 가설을 선택**

$$\check{y} = \underset{k\in\{1,\ldots K\}}{argmax}\ P(C_k)\prod_{i=1}^{n} P(x_i|C_k)$$

즉 나이브 베이즈 분류에서는 클래스 k, 즉 C_k에 대해서 위의 식을 통해 **최대 확률을 갖는 클래스 k를 찾아낸다.**

Bayes'theorem - example

더 많은 속성의 경우

관찰한 일부 용어인 돈, 식료품, 주소 삭제 등을 추가해 스팸 필터 예제를 확장해보자.

	비아그라(W_1)		돈(W_2)		식료품(W_3)		주소 삭제(W_4)		
우도	Yes	No	Yes	No	Yes	No	Yes	No	총합
스팸	4/20	16/20	10/20	10/20	0/20	20/20	12/20	8/20	20
햄	1/80	79/80	14/80	66/80	8/80	71/80	23/80	57/80	80
총합	5/100	95/100	24/100	76/100	8/100	91/100	35/100	65/100	100

메시지가 '비아그라'와 '주소 삭제' 용어를 포함하지만 '돈'과 '식료품'을 포함하지 않는다고 가정하자. 메시지가 스팸인지 아닌지 확률을 구하는 공식은 다음과 같다.

$$P(Spam|W_1 \cap W_2 \cap W_3 \cap W_4) = \frac{P(W_1 \cap W_2 \cap W_3 \cap W_4|spam)P(spam)}{P(W_1 \cap W_2 \cap W_3 \cap W_4)}$$

조건부 독립을 가정하므로

$$P(Spam|W_1 \cap W_2 \cap W_3 \cap W_4) = \frac{P(W_1|spam)P(W_2|spam)P(W_3|spam)P(W_4|spam)P(spam)}{P(W_1)P(W_2)P(W_3)P(W_4)}$$

로 계산할 수 있다.
메시지가 스팸일 확률과 햄일 확률을 비교해야하므로

$$P(ham|W_1 \cap W_2 \cap W_3 \cap W_4) = \frac{P(W_1|ham)P(W_2|ham)P(W_3|ham)P(W_4|ham)P(ham)}{P(W_1)P(W_2)P(W_3)P(W_4)}$$

또한 구해준다.

The Laplace estimator

	비아그라(W_1)		돈(W_2)		식료품(W_3)		주소 삭제(W_4)		
우도	Yes	No	Yes	No	Yes	No	Yes	No	총합
스팸	4/20	16/20	10/20	10/20	0/20	20/20	12/20	8/20	20
햄	1/80	79/80	14/80	66/80	8/80	71/80	23/80	57/80	80
총합	5/100	95/100	24/100	76/100	8/100	91/100	35/100	65/100	100

이번에는 비아그라, 식료품, 주소 삭제 용어가 포함되어 있다고 가정하자.
같은 과정으로 계산하면 스팸은 (4/20)*(10/20)*(0/20)*(12/20)*(20/100) = 0,
햄은 (1/80)*(14/80)*(8/80)*(23/80)*(80/100)=0.0005
그러므로 스팸일 확률은 0/(0+0.0005) = 0, 햄일 확률은 0.0005/(0+0.0005) = 1이다.
이 메시지는 정상적인 메시지에서 거의 드물게 사용하는 비아그라를 포함하고 있다.
그러나 0%의 확률로 스팸이라는 결론을 내고 있다. 잘못 분류되었다고 알 수 있다.

나이브 베이즈에서 확률은 곱으로 되며, 항 중에 한 확률이 0이 되면 사후확률도 0이 되어버린다.
다른 속성에 모두 영향을 미치는 결과를 야기한다. 이 문제는 라플라스 추정기로 해결 할 수 있다.
라플라스 추정기는 각 범주의 발생 확률이 0이 되지 않게 하기 위해 각 값에 작은 수를 추가하는 방식이다.
일반적으로 1로 설정한다.
라플라스 추정기를 활용하여 다시 계산을 해보면
(4+1/20)*(10+1/20)*(0+1/20)*(12+1/20)*(20/100) = 0.0004
(1+1/80)*(14+1/80)*(8+1/80)*(23+1/80)*(80/100)=0.0001
그러므로 스팸일 확률은 0.0004/(0.0004+0.0001) = 0.8,
햄일 확률은 0.0001/(0.0004+0.0001) = 0.2이다.

나이브 베이즈 알고리즘 특징 및 장단점

- 아직 분류되지 않은 데이터를 분류할 때, 관찰된 확률을 사용한다.
- 이 관찰된 확률은 훈련 데이터를 사용해 계산된다.
- 결과의 확률을 추정하기 위해 많은 속성을 고려해야하는 문제에 가장 적합하다.
- 주로 텍스트 분류에 많이 사용됨으로써 문서를 여러 범주(스포츠, 정치, 사회 등)중 하나로 판단하는 문제에 사용된다.

장점

- 단순하고 빠르며 매우 효과적이다.
- 노이즈와 결측 데이터가 있어도 잘 수행한다.
- 상대적으로 적은 수의 훈련데이터에서도 잘 수행된다.
- 단순한 가정에도 불구하고, 많은 복잡한 실제 상황에서 잘 작동한다.

단점

- 모든 속성은 독립적이라는 결함이 있는 가정에 의존한다.
- 수치형 속성보다 명목형 속성으로 구성된 데이터셋에 적합하다.

With R

예림

With Python

대연오빠

나이브베이즈를 활용한 분류(Classification) 실습 with Python

나이브베이즈를 활용한 분류는 아시다시피 여러 분야에서 활용될 수 있어요.
ex) 이메일의 스팸 분류, 리뷰/댓글의 감정분류 등

오늘 실습에서는 그러한 분류를 위한 기초로서,
단어가방(Bag-of-words)을 만들고,

특정 문장이 주어졌을 때,
이 문장이 어느 단어가방에서 나왔을 지 분류해보도록 하겠습니다.

이 과정에서
Likelihood(우도)와 **Laplace Smoothing**의 개념을 설명할 것이며,
전체적인 파이썬 코딩 난이도는 무척 낮습니다. (제 기준으로는 어려웠습..ㅋㅋㅋㅋ)
텍스트 설명이 무척 많습니다. (지루해도 참아주세요 bb)

설명 수준 역시도 저와 같이 아예 처음 접한다고 생각하고 쉬운 기초부터 설명할게요!

기본적으로 Anaconda Python 3.5 version 이 설치되었다고 가정하고 **Jupyter Notebook** 에서 실습을 진행하겠습니다.
아직 설치 전이시라면 다운 & 설치 @ https://www.continuum.io/downloads

마지막 시각화 부분에서 KAIST elice 의 elice_utils 패키지가 필요했는데..
다짜고짜 달라고 떼 쓴 페북 메시지 한 번에 흔쾌히 소스코드를 공유해주신 elice 팀에게 감사를.. ㅎㅎㅎ
근데 소스코드를 주다가 말아서 정작 안돌아가는 건 비밀... ㅠㅠ

1. Bag-of-Words (단어 가방 만들기)

input() : 키보드 입력 및 standard input 을 받아들입니다.
.lower() : 주어진 문자열을 소문자로 치환합니다.
.split() : 주어진 문자열을 () 내의 기준으로 잘라 list 형식([x, y, …, z]) 저장하여 반환합니다.
.len() : 주어진 문자열의 길이를 반환합니다.

`re.sub(r'[^a-z]+', ' ', sentence)` : re는 정규표현식과 관련된 파이썬 내부 패키지입니다.

- sub(pattern, repl, string[, count=0])
-> string에서 pattern과 일치하는 부분에 대하여 repl로 교체하여 결과 문자열을 반환하는 함수입니다.

- r'[^a-z]+' 에서 **r'~'** 은 정규표현식을 다룬다는 것을 뜻합니다.
- [^~] == ~가 아닌 것 (a~z 이 아닌 것)
- 즉, 알파벳 소문자가 아닌 것을 한 글자 혹은 여러 글자 단위(+)로 찾아 space(공백)으로 치환합니다.

입력
: John likes to watch movies. Mary likes movies too. Let ' s go!

출력
: {'mary': 1, 'likes': 2, 'to': 1, 'go': 1, 's': 1, 'too': 1, 'john': 1, 'watch': 1, 'let': 1, 'movies': 2}

2-1. Likelihood (우도)

본 실습에서는 입력된 첫 번째 문장으로 텍스트 모델을 트레이닝 한 뒤에,
이 모델로 **입력된 두 번째 문장을 생성할 확률**을 구합니다.
확률을 구할 때, Laplace smoothing 을 적용해야 합니다.

우도 자체에 대한 설명은 앞 이론 시간에 다뤘으니 생략하겠습니다.

** 목표부터 이해하면 더 이해가 쉽습니다.

예를 들어, 어떤 텍스트 X가 모델 A에서 생성될 확률이 3.5*10^(-12) 이고 모델 B 에서 생성될 확률이 7.0 * 10^(-11) 일 경우,
X가 모델 B에서 생성될 확률이 모델 A에서 생성될 확률보다 20배 더 높으므로, **모델 B로 분류하는 것이 더 타당**합니다.

즉, **[특정 모델 = 특정 BagOfWords 그룹 = 특정 문서 카테고리]** 에 **특정 텍스트가 속할 확률**을 구하는 것
(= 여러 카테고리의 BOW들로부터 해당 텍스트가 생성될 확률을 비교하는 것)

2-1. Likelihood (우도)

본 실습에서는 **함수 calculate_doc_prob** 에서 Laplace smoothing을 이용해,
트레이닝 데이터에서 생성된 bag-of-words 모델이 테스트 데이터를 생성할 로그 확률 을 구합니다.

로그 확률을 구하는 이유는,
긴 문장의 경우 생성 확률값이 매우 작아져 Python 에서 기본으로 사용하는 floating point 형식에서 오차가 발생하기 때문입니다.
(확률이 매우 작아지면 underflow 현상으로 인해 숫자가 제대로 표현되지 않고 오차가 커지기 때문입니다)

* log(abc) = log(a) + log(b) + log©
- 자연로그값은 math.log() 함수를 이용

2-1. Likelihood (우도)

[Likelihood 상세 설명] : 발표 시 다루지 않습니다. 필요하신 경우 읽어보세요!

- Likelihood (가능도 혹은 우도) 는 일반적으로 "확률" 과 비슷한 개념입니다.
  확률 및 통계학에서, likelihood는 확률 분포의 parameter(모수 = 평균값 등 -> 특정 문서 카테고리 내 단어 분포)가 관찰된 sample(표집값 -> 특정 텍스트 내 단어 분포)과 일치하는 정도를 나타내는 parameter에 대한 함수입니다.

- 예를 들어, 동전 던지기 문제를 생각해봅시다.
  동전을 여러 번 던질 때, 각각 앞면 혹은 뒷면이 나오는 사건은 독립사건입니다.
  동전을 한 번 던졌을 때 앞면이 나올 확률을 θ 라고 합시다.
  동전을 10번 던져서 10번 다 앞이 나왔다고 할 때 likelihood 는 θ 를 열 번 곱한 L(θ|x) = θ^(10) 이 됩니다.

- 이것을 이용하면 관찰된 표집값(sample)에 가장 적합한 모델을 골라내는데에 사용할 수 있습니다.
  예를 들어, 동전을 10번 던져서 앞면이 8번 뒷면이 2번 나온 실험을 진행했고 (이것을 x 라고 합시다 **= 특정 텍스트**),
  두 동전 1 과 2가 있다고 합시다. (각각 특정 문서 카테고리 **= 특정 BOW = 특정 모델**)
  그리고 동전 1 의 앞면이 나올 확률 θ1 = 0.7, 동전 2 의 앞면이 나올 확률 θ2 = 0.4 라고 하면

L(θ1|x)=P(x|θ1) = 0.7^8 * (1-0.7)^2 ~ 0.005188
== 동전 1을 사용한 실험이었다고 가정했을 때, 0.7의 확률인 앞면이 8번 나오고 0.3의 확률인 뒷면이 2번 나올 확률
L(θ2|x)=P(x|θ2) = 0.4^8 * (1-0.4)^2 ~ 0.000236
== 동전 2을 사용한 실험이었다고 가정했을 때, 0.4의 확률인 앞면이 8번 나오고 0.6의 확률인 뒷면이 2번 나올 확률

이므로 두 동전의 likelihood를 비교하면 동전 1 의 likelihood가 훨씬 높으므로,
동전 1 이 실험 결과를 훨씬 더 잘 설명한다고 할 수 있습니다. (= 해당 실험 결과는 동전 1 모델로부터 빚어졌을 가능성이 더 높다)

2-2. Laplace Smoothing

왜 Smoothing 을 사용하는가?

학습 데이터에 대한 문제 : 상대적 확률값을 나타내는 parameter들은 학습 데이터에 overfit하는 문제가 될 수 있습니다.

- 이메일에 “minute”이라는 단어가 들어간다고 100% spam ?
- 이메일에 “seriously” 라는단어가 들어간다고 100% ham ?
- 학습데이타에 한번도 나오지않는단어는?

To generalize better: we need to smooth or regularize the estimates.

2-2. Laplace Smoothing

Laplace smoothing (== additive smoothing)은 **카테고리를 가진 데이터의 확률을 부드럽게 만들어주는 기법**입니다.
이 기법은 일어나지 않은 사건을 예측할 때 이미 관측되지 않은 카테고리에도 0이라는 확률을 주지 않게 하기 위해서 만들어졌습니다.

만약 스위치를 누를때마다 빨간색, 녹색, **파란색 사탕 세 개 중 임의로 하나**를 뽑아준다고 광고하는 기계 앞에 있다고 생각합시다.

그리고 스위치를 충분히 많이 눌렀을 때, 빨간색과 녹색은 고루 나왔지만 **파란색 사탕이 한 번도 나오지 않았다**고 합시다.
이 기계에서 파란색 사탕이 나올 확률은 정확히 0일까요?

그럴 수도 있지만, 우리는 이 기계에서 파란 사탕이 안 나온다고 확신할 수 없습니다.
스위치를 훨씬 더 많이 눌러서 빨간색과 녹색 사탕이 더 많이 나오고 파란색이 나오지 않았을지라도
파란 사탕이 나올 확률을 0이라고 단언할 수는 없습니다.

정말 정말 낮은 확률로 파란 사탕이 우연히 나오지 않은 것일수도 있기 때문입니다.
이렇기 때문에 파란 사탕이 나올 확률을 매우 작은 확률로 두게 됩니다.

2-2. Laplace Smoothing

Laplace smoothing 을 적용한 특정 사건의 발생 확률

$$\theta_i = \frac{x_i + \alpha}{N + \alpha d}$$

- θi : i ∈ {red,green,blue} 일 때 각각 빨간색, 초록색, 파란색 사탕이 나올 확률
- xi : i 색 사탕이 나온 횟수
- α(alpha) : laplace smoothing 시에 두는 파라미터로, **α가 작을수록 관측되지 않은 카테고리에 대해 확률을 작게 두게 됩니** 이를테면, α가 작을수록 아직 나오지 않은 파란 색 사탕이 나올 확률을 작게 두는 것입니다.
- N : 색깔에 관계없이 지금까지 관찰한 사탕의 개수
- D : 사탕 색깔의 개수

ex) $x_{(red)} = 6$, $x_{(green)} = 4$ 인 경우,

smoothing 의 적용이 없다면 빨간색 사탕이 6개, 초록색 사탕이 4개 나올 확률을 최대로 만드는 기계는
빨강: θ(red) = 6/10 = 0.6
초록: θ(green) = 4/10 = 0.4
파랑: θ(blue) = 0/10 = **0**
-> **만약 바로 다음 파란색 사탕을 뽑았을 경우 이 모델로 설명할 수가 없게 됩니다.**

Laplace smoothing을 적용하면 (α=0.1) 아래와 같이 각 확률이 조정됩니다.
빨강: θ(red) = 6+α/10+3α ~ 0.5922
초록: θ(green) = 4+α/10+3α ~ 0.3981
파랑: θ(blue) = 0+α/10+3α ~ **0.0097**

2-3. Likelihood & Laplace Smoothing 실습

입력

John likes to watch movies. Mary likes movies too. : 트레이닝 데이터 (BOW를 생성)
John also likes to watch football games. : 테스트 데이터 (BOW로부터 생성되었는지 확인)
0.1 : alpha for Laplace Smoothing

출력

-21.78476859340514
= 트레이닝 데이터에서 생성된 bag-of-words 모델이 테스트 데이터를 생성할 (자연)로그 확률

-9.461004789655119
= 자연로그 함수(math.log()) 대신 상용로그 함수 사용 시 결과값

일반적으로 자연어 처리에서 트레이닝된 모델이 테스트 데이터를 생성할 확률은 굉장히 낮습니다.
그렇기 때문에 확률의 절대값보다는 상대값을 많이 사용합니다.

예를 들어, 어떤 텍스트 X가 모델 A에서 생성될 확률이 3.5*10^(-12) 이고 모델 B 에서 생성될 확률이 7.0 * 10^(-11) 일 경우,
X가 모델 B에서 생성될 확률이 모델 A에서 생성될 확률보다 20배 더 높으므로, 모델 B로 분류하는 것이 더 적당합니다.

3. Naive Bayes Classifier 구현하기

여러 개의 이미 트레이닝된 모델이 있을 때,
주어진 텍스트가 어떤 모델에 더 적합한지 판정 (classify) 하는 방법론을 코딩을 통해 구현해보겠습니다.

우리에게 사탕을 뽑아주는 기계 두 개가 있다고 합시다.
이 기계들에서 각 색상별로 사탕을 뽑을 각각의 확률 θcolor 는,

첫 번째 기계 (M1) : θred = 0.7 θgreen = 0.2 θblue = 0.1
두 번째 기계 (M2) : θred = 0.3 θgreen = 0.4 θblue = 0.3

그리고 우리에게 첫 번째인지 두 번째 기계에서 뽑았을지는 모르지만, 다음과 같은 10개의 사탕들이 있다고 합시다.
- **red: 4**
- **green: 3**
- **blue: 3**

우리는 일반적으로 첫 번째 기계가 두 번째 기계보다 많이 쓰인다는 것을 알고 있다고 가정합니다.
- **p(M1)=0.7**
- **p(M2)=0.3**

우리가 계산하고 싶은 것은,
이미 우리에게 10개의 사탕이 있을 때,
이 사탕을 보고 이것들이 몇 번째 기계에서 나왔을지에 대한 확률을 구하는 것입니다.

이것은 **p(M(k)|x), k∈{1,2}** 로 나타낼 수 있습니다.

3. Naive Bayes Classifier 구현하기

이제 Bayes' Rule 을 적용하면,

p(M(k) | x) = p(M(k)) * p(x | M(k)) / p(x) 이고
여기서 p(x) 는 기계에 관계없이 같으므로 무시하겠습니다.
우리가 구하고자 하는 것은 사탕들이 이 기계에서 나올 확률의 절대값이 아니고,
1번과 2번 기계 중 어떤 기계인지, 즉 상대적인 확률을 구하고자 하는 것이기 때문입니다.

10개의 색상별 사탕들이 이미 나왔을 때(red: 4, green: 3, blue: 3),
이 사탕들이 1번or2번 기계에서 나왔을 확률의 비교는 다음과 같으며,

- p(M1 | x) ∝ **p(M1) * p(x | M1)** = 0.7 * (0.7^4 * 0.2^3 * 0.1^3) = **1.345 * 10^(-6)**
- p(M2 | x) ∝ **p(M2) * p(x | M2)** = 0.3 * (0.3^4 * 0.4^3 * 0.3^3) = **4.199 * 10^(-6)**

두 번째 기계가 첫 번째 기계보다 적게 사용된다는 사실을 고려하더라도 (p(M2) = 0.3),
본 샘플 사탕 10개를 뽑은 기계는 두 번째 기계일 확률이 훨씬 더 높은 것을 알 수 있습니다.

이제 두 확률을 normalize 하겠습니다. (두 확률의 합이 1이 되도록 조정)
(1.345 * 10^(-6) , 4.199 * 10^(-6)) -> (0.243 , 0.757)

->1.345 * 10^(-6) / { 1.345 * 10^(-6) + 4.199 * 10^(-6) } = **0.243**
-> 4.199 * 10^(-6) / { 1.345 * 10^(-6) + 4.199 * 10^(-6) } = **0.757**

그러므로 10개의 사탕 샘플을 통해 Naive Bayes Classifier를 사용했을 때,
이 샘플이 두 번째 기계에서 뽑혔을 확률이 75.7%임을 알아낼 수 있습니다.

3. Naive Bayes Classifier 구현하기

나이브 베이즈 분류를 실습해보겠습니다.
1번째 & 2번째 문장으로 BOW 모델을 만들고, 3번째 문장이 각각의 모델로부터 생성될 확률을 확인하겠습니다.

첫 번째 문장에서 생성된 모델은 위의 예제에서의 첫 번째 기계,
두 번째 문장에서 생성된 모델은 위의 예제에서의 두 번째 기계,
그리고 테스트할 문장은 위의 예제에서의 사탕들이라고 생각하면 됩니다.

입력

John likes to watch movies. Mary likes movies too. : C1 (= 사탕 문제에서 M1, 기계1)
In the machine learning, naive Bayes classifiers are a family of simple probabilistic classifiers.
: C2 (= 사탕 문제에서 M2, 기계2)
John also likes to watch football games. : x (사탕 문제에서 사탕 10개)
0.1 : α (laplace smoothing에 사용되는 숫자)
0.5 : p(C1) (= 사탕 문제에서 p(M1))
0.5 : p(C2) (= 사탕 문제에서 p(M2))

출력

(0.9999985271309568, 1.4728690432649586e-06)
= p(C1|x) 및 p(C2|x) 의 normalize된 값

- 앞서 실습한 create_BOW / log_likelihood / normalize_log_prob 함수가 모두 이미 naivebayes_utils 모듈에 완성되어 있습니다.
- 베이즈 계산 구현만 진행합니다.

4. Naive Bayes로 감정 분석하기

현재까지 감정을 컴퓨터로 분석하려는 시도는 계속 진행중입니다.
크게 **긍정적인 감정/부정적인 감정을 나누는 문제**를 **sentiment analysis (정서 분석)** 이라고 합니다.
이 외에 사람들이 느끼는 감정들,
예를 들어 **분노/기쁨/놀람/사랑/슬픔/두려움 (Parrott's Emotions) 혹은 기쁨/신뢰/두려움/놀람/슬픔/역겨움/화남/기대 (Plutchik's Wheel of Emotions) 등의 분석을 하는 문제**를 **emotion analysis (감정 분석)** 이라고 합니다.

실제 리뷰를 데이터를 통해 Naive Bayes Classifier를 트레이닝하고,
입력받을 문장이 positive일 확률 및 negative일 확률을 구하는 문제 (sentiment analysis) 를 실습하겠습니다.

[Dataset]

데이터는 Bo Pang과 Lillian Lee의 Thumbs up? Sentiment Classification using Machine Learning Techniques 논문에 쓰인것을 사용합니다.
이 데이터는 IMDB의 **영화 리뷰들 중 일부**를 정리한 것으로, **긍정적 리뷰 1000개와 부정적 리뷰 1000개**로 이루어져 있습니다.
데이터셋은 여기(http://goo.gl/gppRX) 에서 다운로드 받을 수 있습니다.

리뷰들은 txt_sentoken 디렉토리 밑의 neg 및 pos 안에 있으며, 파일 한 개가 하나의 리뷰를 의미합니다.
예를 들어, txt_sentoken/neg/cv000_29416.txt 는 부정적인 리뷰이며, 내용의 일부는 다음과 같습니다.

plot : two teen couples go to a church party , drink and then drive . they get into an accident . one of the guys dies , but his girlfriend continues to see him in her life , and has nightmares . what's the deal ?

4. Naive Bayes로 감정 분석하기

read_text_data() 함수가 구현되어 있습니다.
: 폴더 경로를 인자로 받아, 해당 폴더 내의 모든 txt 파일들의 모든 line 들을 읽어들여 all_text 에 저장하는 함수입니다.
-> positive & negative 폴더 각각을 대상으로, 각 폴더 안의 모든 txt 파일의 모든 line들을 all_text 변수에 저장해 반환합니다.
-> 우리는 리턴된 all_text 내의 단어들을 counting 하여 **긍정/부정 각각의 리뷰에 대한 BOW(단어가방) 모델을 만들 수 있습니다.**

입력 : 테스트할 문장 한 줄 (α 값은 0.1, P(pos)=0.5, P(neg)=0.5 가 사전에 코드 상에 설정되어 있습니다)
1) This movie was totally bad... haven't seen worse in my life
2) It was undeniably awesome! A must see!

출력
1) (0.05349, 0.9465) == P(pos|text) = 5.35% & P(neg|text) = 94.65%
2) (0.77613, 0.2238) == P(pos|text) = 77.61% & P(neg|text) = 22.39%

테스트할 문장을 Naive Bayes Classifier 를 이용해 분류를 수행한 뒤에,
이 문장이 긍정적인 리뷰 혹은 부정적인 리뷰인지에 대한 표준화된 확률 두 개의 튜플(() 로 감싸진 자료형)을 반환합니다.
== 표준화된 (P(neg) , P(pos))

두 확률의 합은 1이 되어야 하며 normalize_log_prob 함수에서 이미 구현되어 있습니다.

END

Thank you

BOAZ