

# 비즈니스를 위한 데이터마이닝

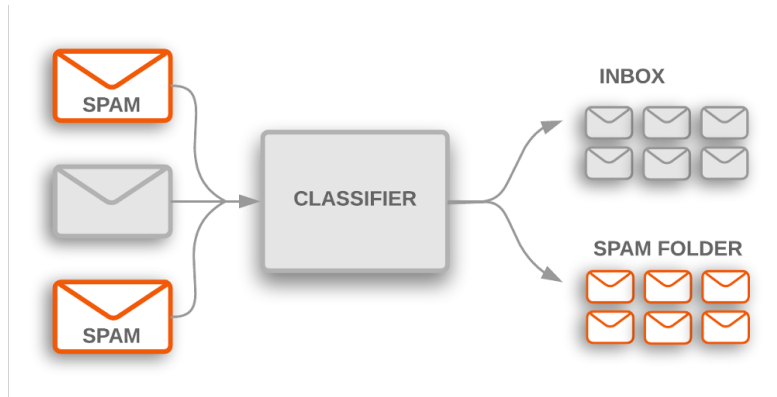
가톨릭대학교 경영학과

이홍주

# 나이브 베이즈 Naïve Bayes

가장 쉬운 분류 모델

- 스팸 메일을 분류하는 모델을 만든다고 가정해 보자.



(출처: <https://medium.com/@naveen.kumar.k/naive-bayes-spam-detection-7d087cc96d9d>)

# 나이브 베이즈 Naïve Bayes

- 총 10개의 메일
  - 3개가 스팸 메일
  - Free 라는 단어가 포함된 메일은 4개
  - 스팸으로 분류된 메일 중 Free 라는 단어를 포함한 메일은 2개
  - Free 라는 단어를 포함한 메일이 스팸으로 분류될 확률은?  $\frac{1}{2} (\frac{2}{4})$



# 나이브 베이즈 Naïve Bayes

- 총 10개의 메일

- 3개가 스팸 메일  $P(\text{SPAM}) = \frac{3}{10}$

- Free 라는 단어가 포함된 메일은 4개  $P(\text{Free}) = \frac{4}{10}$

- 스팸으로 분류된 메일 중 Free 라는 단어를 포함한 메일은 2개  $P(A \cap B)$

$$P(\text{Free} | \text{SPAM}) = \frac{2}{3}$$

- Free 라는 단어를 포함한 메일이 스팸으로 분류될 확률은?



$$P(\text{SPAM} | \text{Free}) = ?$$

조건부 확률

# 나이브 베이즈 Naïve Bayes

- 베이즈 정리
- 이미 알고 있는 확률 값을 통해서 Free라는 단어가 포함된 메일이 스팸일 확률을 구할 수 있음
- A 라는 사건이 발생할 확률을  $P(A)$ 라고 하면 A 사건과 B 사건이 동시에 일어나는 확률은

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

- 두 사건이 독립이 아니라면 B라는 사건이 발생하였을 때 A 사건이 발생할 확률은 다음과 같음

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

- $P(A)$ ,  $P(B)$ ,  $P(A | B)$ 를 안다면 A라는 사건이 발생하였을 때 B 사건이 발생할 확률  $P(B | A)$ 를 구할 수 있음

# 나이브 베이즈 Naïve Bayes

- 베이즈 정리

- $P(A | B) = \frac{P(A \cap B)}{P(B)}$ , 양변에  $P(B)$ 를 곱하면 다음과 같음

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

- 마찬가지로,  $P(B | A) = \frac{P(B \cap A)}{P(A)}$  양변에  $P(A)$ 를 곱하면 다음과 같음

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

- $P(B \cap A)$  와  $P(A \cap B)$ 는 동일하므로,

- $P(A | B) \times P(B) = P(B | A) \times P(A)$  이며,  $P(B | A) = \frac{P(A | B) \times P(B)}{P(A)}$

# 나이브 베이즈 Naïve Bayes

- 총 10개의 메일

- 3개가 스팸 메일  $P(\text{SPAM}) = \frac{3}{10}$

- Free 라는 단어가 포함된 메일은 4개  $P(\text{Free}) = \frac{4}{10}$

- 스팸으로 분류된 메일 중 Free 라는 단어를 포함한 메일은 2개

$$P(\text{Free} | \text{SPAM}) = \frac{2}{3}$$

- Free 라는 단어를 포함한 메일이 스팸으로 분류될 확률은?

$$P(\text{SPAM} | \text{Free}) = \frac{P(\text{Free} | \text{SPAM}) \times P(\text{SPAM})}{P(\text{Free})} = \frac{2/3 \times 3/10}{4/10} = \frac{2}{4}$$

# 나이프 베이즈 Naïve Bayes

- 모델 중심이 아닌 데이터 중심
- 데이터에 대한 가정을 하지 않음
- 16세기 중반 영국 통계학자이자 장로교 목사인 토마스 베이즈 Thomas Bayes(1702~1761)의 이름을 따서 명명 됨





# 나이브 베이즈 Naïve Bayes

- 기본 아이디어
  1. 입력 변수 값들이 동일한 다른 모든 레코드를 찾음
  2. 그 레코드들이 어떤 클래스에 속하고 어떤 클래스가 가장 일반적인지 결정
  3. 그 클래스를 새로운 레코드의 답

# 정확한 베이즈 Exact Bayes

즉, 7장 베이즈 네트워크 알고리즘과 비슷

- 입력 변수의 값이 정확히 동일한 다른 데이터 파악
- 해당 입력변수 값일때 어떤 클래스에 속할 확률 계산

데이터가 있을 경우

정확히 동일한 데이터

찾거나 계산 쉽지 않

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

$$P(C_i | x_1, \dots, x_p) = \frac{P(x_1, \dots, x_p | C_i) P(C_i)}{P(x_1, \dots, x_p | C_1) P(C_1) + \dots + P(x_1, \dots, x_p | C_m) P(C_m)} \quad (8.2)$$

- 입력변수가 많아지면 정확히 일치하는 데이터를 찾기 어려움

# 나이브 베이즈 Naïve Bayes

· 단순함

- 해결책 - 입력변수들간의 독립 가정
  1. C1 클래스에 대해 각 예측 변수의 개별 조건부 확률  $P(x_j|C_1)$  추정
  2. 이 확률을 서로 곱한 후 C1 클래스에 속하는 레코드들의 비율을 곱함
- 모든 클래스에 대해서 단계 ①과 ②를 반복한다.
- 클래스  $C_i$ 에 대해 단계 ②에서 계산한 값을 모든 클래스의 값들의 합으로 나눠서 클래스  $C_i$ 의 확률을 추정한다.

$$P_{nb}(C_1 | x_1, \dots, x_p) = \frac{P(C_1)[P(x_1|C_1)P(x_2|C_1) \cdots P(x_p|C_1)]}{P(C_1)[P(x_1|C_1)P(x_2|C_1) \cdots P(x_p|C_1)] + \cdots + P(C_m)[P(x_1|C_m)P(x_2|C_m) \cdots P(x_p|C_m)]} \quad (8.3)$$

# 예제: 사기 재무 보고 (Financial Fraud)

- 목표 변수: 사기 보고 여부(truthful / fraud)
- 입력 변수
  - 이전 법적 문제 여부 (y / n)
  - 회사 크기 (small / large)

Charges?	Size	Outcome
y	small	truthful
n	small	truthful
n	large	truthful
n	large	truthful
n	small	truthful
n	small	truthful
y	small	fraud
y	large	fraud
n	large	fraud
y	large	fraud

# 예제: 사기 재무 보고 (Financial Fraud)

- 정확한 베이즈

- small / y 인 경우 예측 *값이 동일*

- 2번 등장. truthful 1번, fraud 1번





- $P(\text{fraud} \mid \text{charges}=y, \text{size}=\text{small})$   
 $= \frac{1}{2} = 0.50$

Charges?	Size	Outcome
y	small	truthful
n	small	truthful
n	large	truthful
n	large	truthful
n	small	truthful
n	small	truthful
y	small	fraud
y	large	fraud
n	large	fraud
y	large	fraud

# 예제: 사기 재무 보고 (Financial Fraud)

- 나이브 베이즈
  - fraud 중  $y$  비율 \* fraud 중 small 비율 \* fraud 비율  
 $= 3/4 * 1/4 * 4/10 = 0.075$
  - truthful 중  $y$  비율 \* truthful 중 small 비율 \* truthful 비율  
 $= 1/6 * 4/6 * 6/10 = 0.067$
  - $P(\text{fraud} \mid \text{charges, small}) = 0.075 / (0.075 + 0.067) = 0.528$
- 정확한 베이즈와 큰 확률의 차이는 없음
- 입력변수가 모두 동일한 데이터뿐만 아니라 모든 데이터가 활용됨

# 나이브 베이즈 Naïve Bayes

알고리즘	주요 데이터 타입	분포 가정	예시	비고
<b>Gaussian</b> 	연속형 숫자	 정규분포	키, 온도, 나이	수치형
<b>Multinomial</b> 	 이산형(카운트) 횟수	다항분포	단어 수	음수 불가
<b>Bernoulli</b>	이진형 0, 1만 존재	베르누이분포	단어 존재 여부	0/1
<b>Categorical</b>	명목형(범주형) 카테고리	범주 분포	색상, 지역코드	정수 인코딩 필요

△ 섞인 것들도 있어서 나이브 베이즈 처리가 어려울수도 0  
→ 타 알고리즘 적용해도 괜찮은 됨..

# 예제: 연착 항공편 예측

원하는 카테고리 input 등 세미-항상

- 항공기 정시 출발 / 지연 출발 예측
- Flight Status: ontime/delayed

표 8-3 항공편 연착 예제에 대한 변수 설명

Day of week	1=월요일, 2=화요일, ..., 7=일요일
Sch. dep. time	오전 6:00와 오후 10:00 사이를 18개 구간으로 나눈 출발 시간
Origin	3개의 출발 공항 코드: DCA(레이건 국제공항), IAD(덜러스 국제공항), BWI(볼티모어-워싱턴 국제공항)
Destination	3개의 도착 공항 코드: JFK(케네디 국제공항), LGA(라구아디아 공항), EWR(뉴어크 국제공항)
Carrier	8개의 항공사 코드: CO(컨티넨탈 항공), DH(아틀란틱 코스트 항공), DL(델타 항공), MQ(아메리카 이글 항공), OH(컴에어 항공), RU(컨티넨탈 익스프레스 항공), UA(유나이티드 항공), US(US에어웨이 항공)



# 예제: 연착 항공편 예측

```
delays_df = pd.read_csv('FlightDelays.csv')

# convert to categorical
delays_df.DAY_WEEK = delays_df.DAY_WEEK.astype('category')
delays_df['Flight Status'] = delays_df['Flight
Status'].astype('category')

# create hourly bins departure time
delays_df.CRS_DEP_TIME = [round(t / 100) for t in
delays_df.CRS_DEP_TIME]
delays_df.CRS_DEP_TIME =
delays_df.CRS_DEP_TIME.astype('category')
predictors = ['DAY_WEEK', 'CRS_DEP_TIME', 'ORIGIN', 'DEST',
'CARRIER']
outcome = 'Flight Status'
```

# 예제: 연착 항공편 예측

**# Dummies**

```
X = pd.get_dummies(delays_df[predictors])  
y = delays_df['Flight Status']  
classes = list(y.cat.categories)
```

**# split into training and validation**

```
X_train, X_valid, y_train, y_valid = train_test_split(X, y,  
    test_size=0.40, random_state=1)
```

Default alpha = 1

**# run naive Bayes**

```
delays_nb = MultinomialNB(alpha=0.01)  
delays_nb.fit(X_train, y_train)
```

Smoothing: 학습 시에 없었던

속성에 대해서 대처

**# predict probabilities**

```
predProb_train = delays_nb.predict_proba(X_train)  
predProb_valid = delays_nb.predict_proba(X_valid)
```

**# predict class membership**

```
y_valid_pred = delays_nb.predict(X_valid)
```

# 예제: 연착 항공편 예측

$$\begin{aligned}\hat{P}(\text{delayed} | \text{Carrier} = \text{DL}, \text{Day\_Week} = 7, \text{Dep\_Time} = 10, \text{Dest} = \text{LGA}, \text{Origin} = \text{DCA}) \\ \propto (0.0958)(0.1609)(0.0307)(0.4215)(0.5211)(0.2) = 0.000021\end{aligned}$$

$$\begin{aligned}\hat{P}(\text{ontime} | \text{Carrier} = \text{DL}, \text{Day\_Week} = 7, \text{Dep\_Time} = 10, \text{Dest} = \text{LGA}, \text{Origin} = \text{DCA}) \\ \propto (0.2040)(0.1048)(0.0519)(0.5779)(0.6478)(0.8) = 0.00033\end{aligned}$$

$$\begin{aligned}\hat{P}(\text{delayed} | \text{Carrier} = \text{DL}, \text{Day\_Week} = 7, \text{Dep\_Time} = 10, \text{Dest} = \text{LGA}, \text{Origin} = \text{DCA}) \\ = \frac{0.000021}{0.000021 + 0.00033} = 0.058\end{aligned}$$

$$\begin{aligned}\hat{P}(\text{on time} | \text{Carrier} = \text{DL}, \text{Day\_Week} = 7, \text{Dep\_Time} = 10, \text{Dest} = \text{LGA}, \text{Origin} = \text{DCA}) \\ = \frac{0.00033}{0.000021 + 0.00033} = 0.942\end{aligned}$$

# 예제: 연착 항공편 예측

표 8-6 예제 항공기의 평가 점수(확률과 클래스)



나이프 베이즈를 사용해 데이터의 점수를 구하는 파이썬 코드

```
# classify a specific flight by searching in the dataset
# for a flight with the same predictor values
df = pd.concat([pd.DataFrame({'actual': y_valid, 'predicted': y_valid_pred}),
                pd.DataFrame(predProb_valid, index=y_valid.index)], axis=1)
mask = ((X_valid.CARRIER_DL == 1) & (X_valid.DAY_WEEK_7 == 1) &
        (X_valid.CRS_DEP_TIME_10 == 1) & (X_valid.DEST_LGA == 1) & (X_valid.ORIGIN_DCA == 1))

df[mask]
```

실행 결과

	actual	predicted	0	1
1225	ontime	ontime	0.057989	<u>0.942011</u>

# 예제: 연착 항공편 예측

성적이 좋긴 X  
표 8-7 나이브 베이즈 분류기를 사용한 항공편 연착에 대한 혼동 행렬



혼동 행렬을 위한 파이썬 코드

```
# training
classificationSummary(y_train, y_train_pred, class_names=classes)

# validation
classificationSummary(y_valid, y_valid_pred, class_names=classes)
```

## 실행 결과

Confusion Matrix (Accuracy 0.7955)     79%, ... 정확도

Actual	Prediction	
	delayed	ontime
delayed	52	209
ontime	61	998

Confusion Matrix (Accuracy 0.7821)     78%, ... 정확도

Actual	Prediction	
	delayed	ontime
delayed	26	141
ontime	51	663