

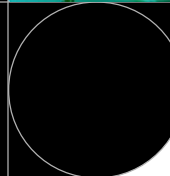
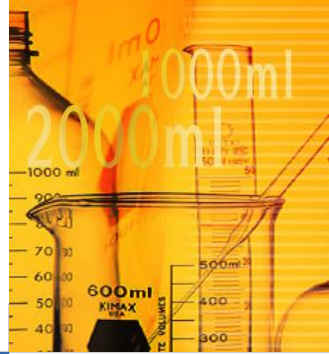
Machine learning

Chapter 4

Naive Bayes

Sejong Oh

Bio Information Technology Lab.



- Naïve 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)를 분류에 활용한다.
 - 다수의 속성이 상대적으로 매우 미비한 효과를 가진다 하더라도 그 영향을 합하면 꽤 큰 효과를 기대할 수 있다

- 조건부 확률

표본공간 S 의 두 사건 A, B 에 대하여 확률이 0이 아닌 사건 A 가 일어났을 때 사건 B 가 일어날 확률을 사건 A 가 일어났을 때의 사건 B 의 **조건부확률**이라 하고, 기호로 $P(B|A)$ 와 같이 나타낸다.

표본공간 S 에서 사건 A 가 일어났을 때의 사건 B 의 조건부확률은 $P(B|A) = \frac{n(A \cap B)}{n(A)}$ 이다. 이 식의 우변의 분자와 분모를 각각 $n(S)$ 로 나누면 다음이 성립한다.

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

- 조건부 확률

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 확률을 구하여 보자.

$P(A)$: 선택된 학생이 여학생일 확률

$P(B)$: 선택된 학생이 헌혈한 적이 있을 확률

$P(B|A)$: 선택된 학생이 여학생 일 때, 헌혈한 적이 있을 확률

Bayes theory

- 조건부 확률

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

$$P(A) = 0.48$$

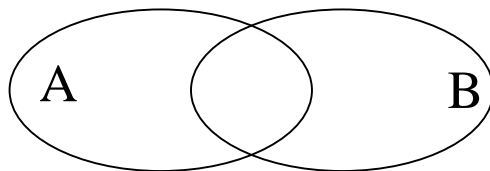
$$P(B) = 0.57$$

$$P(B|A) = \frac{n(A \cap B)}{n(B)} = \frac{25}{48} = 0.52$$

Bayes theory

- 베이즈 정리는 1740년대의 영국의 목사인 토머스 베이즈(Thomas Bayes)가 정립한, 조건부 확률에 대한 수학적 정리
- 사건 A가 있고 사건 B가 있을 때 사건 B가 일어난 것을 전제로 한 사건 A의 조건부 확률($P(A|B)$)을 구하고 싶다. 그런데 지금 알고 있는 것은 사건 A가 일어난 것을 전제로 한 사건 B의 조건부 확률($P(B|A)$), A의 확률($P(A)$), B의 확률($P(B)$)뿐이다. 그럴 때 다음과 같이 구할 수가 있다.

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)} = \frac{P(A)P(B|A)}{P(A)P(B|A) + P(A^c)P(B|A^c)}$$



Bayes theory

- 베이지 정리 예제

어느 대학에 합격한 신입생(B) 중 남학생(A_1)의 확률을 구하고자 함.

알려진 정보 :

- 응시학생 중 남녀 학생의 비율 7:3 ,
- 응시한 여학생의 합격률 0.4,
- 응시한 남학생의 합격률 0.2

$$\begin{aligned} P(A_1|B) &= \frac{P(A_1)P(B|A_1)}{P(B)} = \frac{P(A_1)P(B|A_1)}{P(A_1)P(B|A_1) + P(A_2)P(B|A_2)} \\ &= \frac{0.7 \times 0.2}{0.7 \times 0.2 + 0.3 \times 0.4} = 0.538 \end{aligned}$$

$$P(A_2|B) = \frac{P(A_2)P(B|A_2)}{P(B)} = \frac{P(A_2)P(B|A_2)}{P(A_1)P(B|A_1) + P(A_2)P(B|A_2)}$$

- 베이즈 정리의 일반화

$$P(A_i|B) = \frac{P(A_i)P(B|A_i)}{P(A_1)P(B|A_1) + P(A_2)P(B|A_2) + \dots + P(A_n)P(B|A_n)}$$

Naïve Bayes classifier

- Class 를 알 수 없는 입력 벡터 $D = e_1, e_2, e_3, \dots, e_n$
- D 가 클래스 C_i 일 확률 : $P(C_i | D) = P(C_i | e_1, e_2, e_3, \dots, e_n)$
- Bayes 정리를 적용하면

$$P(C_i | D) = \frac{P(C_i)P(D | C_i)}{P(D)}$$

이 값이 가장 큰 C_i 가 D 가 속하는 클래스라고 결정

- 분류 문제에서 모든 C_i 에 대해 $P(D)$ 는 공통이므로 실제로는 계산하지 않는다.
- $P(C_i) = n(C_i)/N$: 쉽게 계산
- $P(D | C_i)$: 계산이 복잡하고 매우 많은 양의 계산이 필요.

Naïve Bayes classifier

- Naïve Bayes에서는 입력 벡터를 구성하는 각각의 요소가 입력 벡터에 나타날 확률은 서로 독립적이라고 가정 (사실은 독립이 아님)

$$D = e_1, e_2, e_3, ..e_n$$

$$P(C_i|D) = \frac{P(C_i)P(D|C_i)}{P(D)} \Rightarrow P(C_i) \times P(e_1|C_i) \times P(e_2|C_i) \times .. \times P(e_n|C_i)$$

Example

	Predictors				Response
	Outlook	Temperature	Humidity	Wind	Class Play=Yes Play=No
Day1	Sunny	Hot	High	Weak	No
Day2	Sunny	Hot	High	Strong	No
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	No
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	No
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	No

D =

sunny	Cool	High	Strong
-------	------	------	--------

 ??

Example

$$\begin{aligned}P(\text{Yes}|\text{D}) &= P(\text{Yes}) \times P(\text{Sunny}|\text{Yes}) \times P(\text{Cool}|\text{Yes}) \times P(\text{High}|\text{Yes}) \times P(\text{Strong}|\text{Yes}) \\&= 9/14 \times 2/9 \times 3/9 \times 3/9 \times 3/9 \\&= 0.0053\end{aligned}$$

$$\begin{aligned}P(\text{No}|\text{D}) &= P(\text{No}) \times P(\text{Sunny}|\text{No}) \times P(\text{Cool}|\text{No}) \times P(\text{High}|\text{No}) \times P(\text{Strong}|\text{No}) \\&= 5/14 \times 3/5 \times 1/5 \times 4/5 \times 3/5 \\&= 0.00205\end{aligned}$$

\therefore D is classified into “Yes”

- 장점

- 단순하고 빠르며 효과적이다
- 노이즈와 결측 데이터가 있어도 잘 수행한다
- 훈련에 대해 상대적으로 적은 예제가 필요 (많은 예제에서도 잘 수행)
- 예측에 대한 추정된 확률을 얻기 쉽다

- 단점

- 모든 속성은 동등하게 중요하고 독립적이라는 알려진 결함 (open-faulty assumption) 을 가짐
- 연속형 수치 속성으로 구성된 데이터셋에 대해 이상적이지 않다. (discretization 필요)
- 추정된 확률은 예측된 범주 보다 덜 신뢰적이다

라플라스 추정값 (Laplas estimator)

D =

sunny	Cool	High	None
-------	------	------	------

 ??

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

$$\begin{aligned} P(\text{Yes}|D) &= P(\text{Yes}) \times P(\text{Sunny}|\text{Yes}) \times P(\text{Cool}|\text{Yes}) \times P(\text{High}|\text{Yes}) \times P(\text{None}|\text{Yes}) \\ &= 9/14 \times 2/9 \times 3/9 \times 3/9 \times 0/9 \\ &= 0 \end{aligned}$$

$$\begin{aligned} P(\text{No}|D) &= P(\text{No}) \times P(\text{Sunny}|\text{No}) \times P(\text{Cool}|\text{No}) \times P(\text{High}|\text{No}) \times P(\text{None}|\text{No}) \\ &= 5/14 \times 3/5 \times 1/5 \times 4/5 \times 0/5 \\ &= 0 \end{aligned}$$

- Wind 를 제외한 나머지 정보를 가지고도 분류가 가능한데 0 이 곱해져서 다른 정보도 쓸모 없게 만든다.
- 이를 방지하기 위해 확률이 0 이 되지 않도록 임의의 값 (대부분 1 을 사용) 을 부여하여 계산

라플라스 추정값 (Laplas estimator)

- 라플라스 추정값은 각 속성별로 서로 다른 값을 부여할 수 도 있다.
- 속성의 중요도에 따라 서로 다른 값을 부여할 수 도 있다.
- 훈련 데이터셋이 충분히 크다면 라플라스 추정값은 고려하지 않아도 된다

대부분의 분류 알고리즘들은 모델의 성능에 영향을 미치는 parameter 를 가지고 있다. (KNN의 k, Naïve Bayes 의 경우는 Laplas estimator 가 대표적). 이와 같은 parameter 를 조율 모수(tuning parameter) 라고 한다. 학습 모델을 수립시 최적의 조율 모수를 찾는 것이 과제 중의 하나이다.

Naïve Bayes in R

- Package : e1071

```
naiveBayes(train, cl, laplace = 0)
```

- train : training dataset (data frame)
- cl : class label vector (factor type)

```
library(e1071)
train.ds = iris[,1:4]
train.cl = iris[,5]
test.ds = iris[,1:4]
test.cl = iris[,5]

model = naiveBayes(train.ds, train.cl)
pr = predict(model, test.ds)

acc = mean(pr==test.cl)    # accuracy
print(acc)
```


Practice : 스팸 메일 분류

- Step 1. 데이터 준비

- Spam dataset : <http://www.dt.fee.unicamp.br/~tiago/smsspamcollection/>
- 강의자료 : [mlr-ko/chapter 4/sms_spam2.csv](#)

	A	B	C	D	E	F	G	H	I	J	K	L
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 wit										
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 l										
19	ham	He dint tell anything. He is angry on me that why you told to abi.										

Practice : 스팸메일 분류

- Step 2. 데이터 읽기 및 탐색

```
sms_raw <- read.csv("sms_spam.csv", stringsAsFactors  
= FALSE)  
sms_raw[,1] <- iconv(sms_raw[,1], "WINDOWS-  
1252", "UTF-8") # multibyte character conversion  
  
# sms 데이터 구조  
dim(sms_raw)  
str(sms_raw)
```

Practice : 스팸메일 분류

```
# 팩터로 spam/ham으로 변환
sms_raw$type <- factor(sms_raw$type)

# 변수형 확인
str(sms_raw$type)
table(sms_raw$type)

# 텍스트 마이닝 (tm) 패키지를 사용하여 말뭉치 (corpus) 생성
library(tm)
sms_corpus <- Corpus(VectorSource(sms_raw$text))

# sms 말뭉치 확인
print(sms_corpus)
inspect(sms_corpus[1:3])
sms_corpus[[1]]$content
sms_corpus[[2]]$content
```

안됨

Practice : 스팸메일 분류

- 이메일 또는 SMS 메시지는 단어, 공백, 숫자, 마침표 등으로 구성된 문자열
- 이런 복잡한 종류의 비정형 데이터 처리에는 많은 노력이 필요
- 스팸을 걸러내는데 불필요한 문자들은 제거 해야함. 또한 문장을 단어 단위로 분리하는 것이 필요
- 텍스트 데이터 처리의 첫단계는 말뭉치(copus: 텍스트 문서의 모음)을 만드는 일
- tm package
 - 텍스트 마이닝을 지원 하는 패키지

Practice : 스팸메일 분류

```
## tm_map() 사용하여 말뭉치 정리 -----  
# 대문자는 소문자로  
corpus_clean <- tm_map(sms_corpus,  
                        content_transformer(tolower))  
# 숫자 제거  
corpus_clean <- tm_map(corpus_clean, removeNumbers)  
# 불용어(to, and, but,...) 제거  
corpus_clean <- tm_map(corpus_clean, removeWords,  
                        stopwords())  
# 마침표 제거  
corpus_clean <- tm_map(corpus_clean,  
                        removePunctuation)  
# 보이지 않는 공백 제거  
corpus_clean <- tm_map(corpus_clean, stripWhitespace)
```

Practice : 스팸메일 분류

말뭉치 정리 확인

inspect(sms_corpus[1:3]) # 안됨

inspect(corpus_clean[1:3]) # 안됨

sms_corpus[[1]]\$content

corpus_clean[[1]]\$content

sms_corpus[[2]]\$content

corpus_clean[[2]]\$content

sms_corpus[[3]]\$content

corpus_clean[[3]]\$content

```
> 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"
> |
```

Practice : 스팸메일 분류

```
# 문서-용어 희소 매트릭스 생성
```

```
sms_dtm <- DocumentTermMatrix(corpus_clean)
sms_dtm
```

- DocumentTermMatrix() : 말뭉치를 입력 받아 토큰화 한다. 그 결과 희소 행렬 생성

```
> sms_dtm
<<DocumentTermMatrix (documents: 5559, terms: 7905)>>
Non-/sparse entries: 42623/43901272
Sparsity           : 100%
Maximal term length: 40
Weighting           : term frequency (tf)
> |
```

Practice : 스팸메일 분류

```
inspect(sms_dtm[1:10,20:30])
```

```
> inspect(sms_dtm[1:10,20:30])
<<DocumentTermMatrix (documents: 10, terms: 11)>>
Non-/sparse entries: 0/110
Sparsity           : 100%
Maximal term length: 10
Weighting          : term frequency (tf)

      Terms
Docs abelu aberdeen abi ability abiola abj able abnormally aboutas abroad absence
  1      0          0  0          0      0  0  0          0          0          0
  2      0          0  0          0      0  0  0          0          0          0
  3      0          0  0          0      0  0  0          0          0          0
  4      0          0  0          0      0  0  0          0          0          0
  5      0          0  0          0      0  0  0          0          0          0
  6      0          0  0          0      0  0  0          0          0          0
  7      0          0  0          0      0  0  0          0          0          0
  8      0          0  0          0      0  0  0          0          0          0
  9      0          0  0          0      0  0  0          0          0          0
 10      0          0  0          0      0  0  0          0          0          0
> |
```


Practice : 스팸메일 분류

- Step 3. 훈련/테스트 데이터셋 생성

```
# 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$type))  
prop.table(table(sms_raw_test$type))
```

```
> 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
```

Practice : 스팸메일 분류

```
# 단어 클라우드 시각화
library(wordcloud)

wordcloud(sms_corpus_train, min.freq = 30,
random.order = FALSE)
```


Practice : 스팸메일 분류

```
# 훈련 데이터를 스팸과 햄으로 구분
```

```
spam <- subset(sms_raw_train, type == "spam")
```

```
ham <- subset(sms_raw_train, type == "ham")
```

```
wordcloud(spam$text, max.words = 40, scale = c(3,  
0.5))
```

```
wordcloud(ham$text, max.words = 40, scale = c(3,  
0.5))
```


- Step 4. 빈도 단어에 대한 지표 속성 생성
 - 희소 메트릭스를 Naïve Bayes 분류기를 훈련하기 위한 구조로 변환하는 과정 필요
 - 즉, 출현 빈도가 낮은 단어(속성)은 데이터셋에서 제거 한다
 - 단어의 빈도수를 찾는 `tm` 함수 : `findFreqTerms()`

```
findFreqTerms(sms_dtm_train, 5) # 출현빈도 5회 이상 단어

#sms_dict <- Dictionary(findFreqTerms(sms_dtm_train,
5)) ## Error
sms_dict <- findFreqTerms(sms_dtm_train, 5)
sms_train <- DocumentTermMatrix(sms_corpus_train,
                                list(dictionary = sms_dict))
sms_test  <- DocumentTermMatrix(sms_corpus_test,
                                list(dictionary = sms_dict))
```

Practice : 스팸메일 분류

개수를 팩터로 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)
```

Practice : 스팸 메일 분류

```
> sms_train[1:10, 11:20]
      added address admirer advance aft afternoon age ago aha ahead
1      No      No      No      No  No           No  No  No  No  No
2      No      No      No      No  No           No  No  No  No  No
3      No      No      No      No  No           No  No  No  No  No
4      No      No      No      No  No           No  No  No  No  No
5      No      No      No      No  No           No  No  No  No  No
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
> |
```


Practice : 스팸메일 분류

- Step 5. 데이터로 모델 훈련

```
library(e1071)
sms_model <- naiveBayes(sms_train,
                        factor(sms_raw_train$type))
sms_model
```

```

              still
sms_raw_train$type      No      Yes
      ham 0.970873786 0.029126214
      spam 0.992907801 0.007092199

              stockport
sms_raw_train$type      No      Yes
      ham 1.0000000 0.0000000
      spam 0.9893617 0.0106383

              stop
sms_raw_train$type      No      Yes
      ham 0.993065187 0.006934813
      spam 0.877659574 0.122340426

              store
sms_raw_train$type      No      Yes
      ham 0.9997226075 0.0002773925
      spam 0.9893617021 0.0106382979
```

- Step 6. 모델 성능 평가

```
sms_test_pred <- predict(sms_model, sms_test)

library(gmodels)
CrossTable(sms_test_pred, factor(sms_raw_test$type),
           prop.chisq = FALSE, prop.t = FALSE,
           prop.r = FALSE,
           dnn = c('predicted', 'actual'))

acc <- mean(sms_test_pred==factor(sms_raw_test$type))
acc
```

Practice : 스팸메일 분류

Total Observations in Table: 1390

predicted	actual		Row Total
	ham	spam	
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	

```
> acc  
[1] 0.9741007  
> |
```

- Step 7. 모델 성능 향상

```
sms_model2 <- naiveBayes(sms_train,  
  factor(sms_raw_train$type), laplace = 1)  
sms_test_pred2 <- predict(sms_model2, sms_test)  
CrossTable(sms_test_pred2, factor(sms_raw_test$type),  
  prop.chisq = FALSE, prop.t = FALSE, prop.r  
= FALSE, dnn = c('predicted', 'actual'))  
  
acc <- mean(sms_test_pred2==factor(sms_raw_test$type))  
acc
```

Practice : 스팸메일 분류

Total Observations in Table: 1390

predicted	actual		Row Total
	ham	spam	
ham	1204	31	1235
	0.998	0.169	
spam	3	152	155
	0.002	0.831	
Column Total	1207	183	1390
	0.868	0.132	

```
>
> acc <- mean(sms_test_pred2==factor(sms_raw_test$type))
> acc
[1] 0.9755396
> |
```

Exercise 1

- Naïve Bayes 를 구현 하시오
- iris 데이터셋을 가지고 구현한 알고리즘을 테스트 하시오

Exercise 2

- 다음의 데이터셋을 다운 받아 Naïve Bayes classifier 를 테스트 하시오
- Spam dataset : <http://www.csmining.org/index.php/enron-spam-datasets.html>

Navigation

[Enron-Spam datasets](#)

[Ling-Spam datasets](#)

[PU1 and PU123A datasets](#)

[Spam-Assassin datasets](#)

[20 Newsgroups](#)


Enron-Spam datasets


This directory contains the Enron-Spam datasets, as described in the paper:

V. Metsis, I. Androutsopoulos and G. Paliouras, "Spam Filtering with Naive Bayes - Which Naive Bayes?". Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS 2006), Mountain View, CA, USA, 2006.

Download

Preprocessed

 enron1.tar.tar (1.8 MB)

 enron2.tar.tar (2.9 MB)