##### **과제04-Naive Bayes Classifier**

김의찬

국민대학교 전자공학부

**요 약**

강의 11, 12, 13 내용을 기반으로 조건부 확률과 베이즈 정리를 이해하고 Naive Bayes Classifier를 구현합니다.

1. **배경 이론**
   1. **확률 기본 단어** 사건 : 특정 조건에 따라 일어날 수 있는 각각의 경우를 뜻함.  
        
       확률 변수 : 확률 결과를 변수로 묘사는 개념. 사건에 수치가 부여된 것이라고 생각 할 수 있음
   2. **조건부확률** 사건 B가 일어났을 때 사건 A가 일어날 확률을 합니다. 구하는 식은 다음과 같습니다.
   3. **베이즈 정리**A = 특정 단어가 메일에 있는 사건   
      B = 메일을 관련 있다고 분류하는 사건  
       메일에 특정 단어가 있을 때 관련 있다고 분류될 확률은(사후 확률, P(B|A)), 분류한 메일에서 그 단어가 나올 확률과(증거 추가, P(A|B)), 원래 메일을 관련 있다고 분류할 확률을 곱한 것과 같다(사전 확률, P(B)). P(A)는 정규화를 위한 상수처럼 작용합니다.
2. **과제 수행 내용** 과제를 하면서 용어와 사건들을 다음과 같이 나타나겠습니다.  
     
   **data doc** : 사전에 가지고있던 주어진 문장  
   **판별 해야 할 doc** : 특정 사건으로 우리가 새로 판별 해야 할 문장  
     
   A -> doc을 관련 있음으로 분류하는 사건  
   B -> doc을 관련 없음으로 분류하는 사건  
     
   X(“like”) -> doc 에서 “like” 라는 단어가 나오는 사건
   1. **P(A), P(B) 계산** 임의의 **판별해야 할 doc**이 출현했을 때 두가지 사건으로 분류될 수 있습니다.  
        
      1. 관련 있음으로 분류된다. A  
      2. 관련 없음으로 분류된다. B  
        
       각각의 확률은 ‘**data doc**들이 어떤 분포를 가지고 있었는지’로 그 확률을 추정할 수 있습니다. 즉 P(A) = data doc중 A로 판별된 수 / 전체 data doc 수라고 할 수 있겠습니다. 나머지 클래스에 대해서도 마찬가지입니다.  
        
      ****  
      확률은 1보다 작아 계속 곱하면 0으로 수렴해버리기 때문에 나중에 크기를 비교하기 힘들 수도 있습니다.
   2. **P(X(“단어”)|A), P(X(“단어”)|B)의 계산** 이 부분을 설명하기 전에 몇 가지 가정을 해야 합니다.  
        
       1. 단어들이 나오는 사건들은 서로 순서적으로 독립된 사건이라 가정합니다. 예를 들어 He likes her, she likes him 이런식으로 likes 앞뒤에 단어가 바뀌어도 likes가 출연할 확률은 달라지지 않는다고 가정합니다.  
        
       2. 중복을 허용된다고 가정했습니다. Help, help, help 가 나올 경우 help에 대한 계산을 3번 한다고 가정합니다.  
        
       p(X(“단어”)|A)의 의미를 풀어서 쓰자면 내가 어떤 doc을 관련이라고 찍었을 때 그 doc에서 “단어” 가 나올 확률입니다.  
        
       강의13번에서 구하는 방법과 제가 생각하는 방법 2가지로 구현하였습니다.  
        
      **방법 1)** p(X(“help”)|A) = 해당 클래스에서의 “help” 빈도수/전체 클래스에서 “help” 빈도수  
      **방법 2)** p(X(“help”)|A) = 해당 클래스에서의 “help” 빈도수/해당 클래스에서의 전체 단어 수  
        
      방법 1은 수업에 나온 방법이고 방법2는 제가 생각한 방법입니다. 저는 p(X(“단어”)|A)에서 B사건에서 일어난 결과가 반영되면 안된다고 생각하기 때문에 방법2번이 맞는 방법이라 생각했습니다.  
        
       문제되는 점은 확률이 둘 다 0이 되 버리는 경우가 생깁니다. 이러한 경우는 데이터의 양을 늘리면 해결됩니다.  
        
      **텍스트이(가) 표시된 사진

      자동 생성된 설명**
   3. **Score 구현**함수의 입력은 문장과 어떤 식으로 주장할 것인지를 받습니다. 결과는 그 주장에 대한 신뢰도입니다. Score (“help me”, “relevant”) 로 입력했을 경우 help me 는 관련 있는 doc이다라는 주장에 대한 신뢰도가 얼마다 라는 결과가 나온다는 것 입니다. 여기서는 베이즈 정리가 사용됩니다. 먼저 어느 doc이 도착했을 때 이 문서는 관련 있는 문서다 라는 주장에 대해서는 P(A) 가 됩니다.  
        
       문장을 이루는 단어 하나하나들은 증거가 됩니다. 우리는 이러한 증거들을 바탕으로 신뢰도를 업데이트 할 것입니다. 베이즈 정리에 따라 다음과 같이 나타낼 수 있습니다.
   4. **Predict 구현**입력은 분류할 doc이고, 출력은 해당 doc의 class입니다. Predict(“help me”)는 help me라는 doc에 대해서 두가지 신뢰도를 구합니다. 만약 다음과 같은 결과가 나왔다고 한다면  
        
      Help me는 관련된 문서다 -> 신뢰도 80%  
      Help me는 관련 없는 문서다 -> 신뢰도 60%  
      우리는 Help me를 관련된 클래스로 분류할 수 있는 것입니다.
   5. **Evaluate 구현** 분류 결과와 GT값을 가지고 precision, recall, f1 score을 구합니다.
   6. **Train stop words** 일반 train 코드를 그대로 복사하고, doc 을 단어 단위로 변환할 때 차집합을 구하여 구했습니다.  
        
      **텍스트, 시계, 어두운이(가) 표시된 사진

      자동 생성된 설명**
3. **실험 결과 및 결과 분석**방법1)  
   텍스트이(가) 표시된 사진

   자동 생성된 설명  
   방법2)  
   **텍스트이(가) 표시된 사진

   자동 생성된 설명** 방법1에서보다 방법2에서 10% 더 높은 f1 score가 나옵니다. 방법 1의 문제는test에서 class 별 데이터의 수가 많이 차이날 때 가중이 너무 심하게 일어나는 경우가 생길 수 있습니다. (이미 가중은 2.1을 계산할 때 반영됩니다.) 어느정도 정확도가 맞는 이유는 두 분포가 비슷하기 때문이라고 생각됩니다. (넓은 의미로)  
     
   f1 score 0.88로 어느정도 분류를 해낼 수 있다는 것을 알 수 있습니다.
4. **결론** 나이브 베이즈 분류기는 관련 있는 내용과 없는 내용을 어느 정도 분류했습니다. 또한 훈련부터 분류까지 빠른 속도를 보여줬습니다. 의미있는 분류기라고 생각 할 수 있습니다  
     
    이를 구현하면서 기본적인 확률 개념들과, 베이즈 정리를 다시 공부하고 개념을 확실히 이해 할 수 있었습니다.