

Algoritmos de Classificação Naïve Bayes

Leonardo Camilo Ribeiro, João Carlos Pandolfi Santana

Instituto Federal do Espírito Santo – IFES

Serra - ES - Brasil

{leo.camilo.ribeiro, joaopandolfi} @gmail.com

Abstract. *This paper presents applications and general explanation of the Naïve Bayes algorithm. Concepts of the algorithm, a small application to text classification written in two different languages, and knowledge extraction from a data set, and conclusions about the difficulty of implementation and efficiency based on the results of both applications.*

Resumo. *Este artigo apresenta aplicações e explicação geral do algoritmo de Naïve Bayes. São apresentados conceitos do algoritmo, uma pequena aplicação para classificação textual escrita em duas linguagens diferentes, e para extração de conhecimento de uma massa de dados, e são apresentadas conclusões em relação à dificuldade de implementação e eficiência com base nos resultados obtidos de ambas as aplicações.*

1- Introdução

O algoritmo leva este nome pelo fato de ser chamado de “ingênuo” (Naïve), pois o método no qual o mesmo se baseia é na análise independente dos termos, isso faz com que ao verificar entidades separadas perde-se um pouco o conceito geral.

O mesmo tem como objetivo calcular a probabilidade no qual uma amostra desconhecida possa pertencer a alguma das possíveis classes pré-definidas, ou seja, estabelecer a priori há qual classe ele provavelmente pertence. Este tipo de predição é chamada de classificação estatística, pois é completamente baseada em probabilidades.

O que o faz uma ferramenta funcional, para análise de dados, é o fato de que ao aumentarmos substancialmente a quantidade de informação a ser processada, o resultado da mesma converge para um valor mais confiável, logo o algoritmo contém certo grau de “aprendizado”, porém é necessário um lastro de informação inicial pré-determinado e confiável para o funcionamento do mesmo.

A classificação baseia-se em verificar uma entidade, no qual está inserida em um determinado contexto, é uma das técnicas de mineração de dados amplamente utilizada.

“A classificação consiste no processo de encontrar, através de aprendizado de máquina, um modelo ou função que descreva diferentes classes de dados” [Han e Kamber 2006].

Uma das aplicações do método Bayesiano é a categorização textual, que consiste na organização de documentos em classes preestabelecidas. “Esta categorização tem diversas aplicações na área de Recuperação de Informação, tais como detecção de SPAM, organização automática de e-mails, identificação de páginas com conteúdo adulto e detecção de expressões multpalavras” [Manning et al. 2008].

Neste trabalho desenvolvemos um classificador de texto baseado no teorema bayesiano onde dado um texto qualquer ele é processado pelo programa e classificado como sendo de alguma classe pré-existente no programa.

2- O algoritmo de Naïve Bayes

Baseia-se em métodos probabilísticos para análise e classificação da informação, dado um conjunto $(f_1, f_2, f_3, \dots, f_n) \in F$ a probabilidade do conjunto F pertencer a uma determinada classe C é obtida do seguinte modo, calcula-se a probabilidade a priori, ou seja, do conjunto como um todo pertencer a uma determinada classe e de cada elemento do conjunto F pertencer a mesma, obtendo a estimativa final.

Uma representação matemática do algoritmo descrito acima é demonstrado logo abaixo.

$$probPosteriori = p(C) * \prod_{i=1}^n p(f_i/C)$$

Calcula-se este fator para cada classe existente, ao final o conjunto F é classificado como pertencente à classe C, essa classificação é feita de acordo com o maior valor em questão, pelo fato de o algoritmo ser probabilístico o resultado final não é uma determinação, o mesmo deve ser analisado de acordo com a aplicação em questão.

Computacionalmente falando, este algoritmo é muito custoso, pois caso haja muitos elementos em uma classe ou muitas classes, ele se torna muito longo, e por se tratar de um produto de probabilidades, é quase certo o *under flow*, para evitar isso é utilizado o logaritmo.

$$\log(C|F) = \log\left(p(C) * \prod_{i=1}^n p(f_i|C)\right) = \log(p(C)) + \sum_{i=1}^n \log(p(f_i|C))$$

Este é um algoritmo de probabilidade condicional onde as probabilidades a priori, ou seja, que você já conhece, modelam as prioridades a posteriori, por isso é dito que é um algoritmo de aprendizagem de máquina, pois a probabilidade condicional $p = (a|b)$ se baseia na chances de ocorrer o evento a dado o evento b , se é sabido quem em 100 testes realizados 93 resultaram em $p = x$ é correto dizer que x possui 93% de chances de ocorrer em $p = (a|b)$.

3- Implementação

Para aplicação do teorema de Naïve Bayes foi implementado um classificador de documentos de texto na linguagem C, a aplicação contém um TAD (tipo abstrato de dados) para aplicar o teorema na classificação de texto e outro para tratar do texto.

A princípio o programa abre um arquivo de índice onde estão listados todos os documentos que deverão ser abertos, e as respectivas classes ao qual são pertencentes,

```
arqIndex = fopen("index.txt", "r");
```

Logo após isso cada documento é transformado em uma estrutura de dados do tipo lista encadeada de palavras através da função `convertToList(text)`; e salvo em uma estrutura de dados, essa em questão é uma lista de listas, no caso, uma lista de documentos.

Ambos os TAD's possuem funções quem auxiliam na obtenção das variáveis necessárias para o calculo, por fim é calculada a probabilidade de determinado texto pertencer a alguma das classes existentes, para isso é usada a função:

```
prob(documentList, textList, class, probPriori);
```

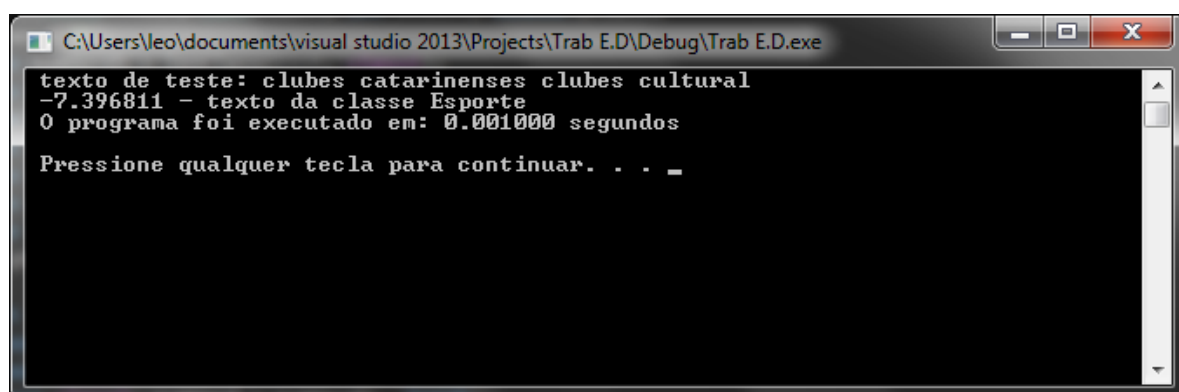
Esta por sua vez retorna o valor estimado do texto “`textList`” baseado nas classes existentes na base de dados “`documentList`” pertencer a classe “`class`”, e a probabilidade que possuir o maior valor é a classe à qual o texto pertence.

Também foi implementado o mesmo algoritmo em python, a fim de se avaliar alguns aspectos, como eficiência, tempo de execução e dificuldade de implementação.

A tabela abaixo mostra os documentos existentes e suas respectivas classes.

Cultura	brinquedos criativos estimulam encantam criancas pais summer balneario
	prefeitura camboriu abre inscricoes concurso rainha princesas camboriu festa ru
	concurso cultural vai batizar filhotes zoo beto carrero world
Esporte	desafio estrelas pilotos famosos estaraos em penha
	apos entrega laudos clubes catarinenses esperam liberacao estadios
	presidente confederacao brasileira de tenis acreditamos nesta nova geracao
Policia	pais chamam policia encontrar drogas escondidas quarto filho anos timbo
	prefeita luzia recebe visita comandante batalhão policia militar

O texto teste utilizado foi: “clubes catarinenses clubes cultural”, conforme mostra os resultados abaixo.



programa escrito em C

```

Python 3.3.3 (v3.3.3:c3896275c0f6, Nov 18 2013, 21:19:30) [MSC v.1600 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
Probabilidade da classe 1 : 5.703189737386985e-20
Probabilidade da classe 3 : 2.0474591803494141e-13
Probabilidade da classe 2 : 9.838596339240109e-20
Portanto Pertence a classe: 2
Tempo de execução: 0.013000965118408203
>>> |

```

programa escrito em Python (valor diferente pois não foi calculado o logaritmo)

O algoritmo funciona da seguinte maneira:

- primeiramente é calculada a probabilidade a priori:

$$p(Esporte) = \frac{3}{8}$$

- Após isso é calculado as prioridades de cada elemento do conjunto:
 - T_i Elemento i da classe Texto
 - C Conjunto C
 - oC numero de ocorrência no conjunto C
 - oG numero de ocorrência em todos os conjuntos
 - nC numero de palavras no conjunto
 - nG numero de palavras em todos os conjuntos

$$p(T_i|C) = \frac{oC + oG}{nC + nG}$$

$$p(clubes|esporte) = \frac{1 + 1}{24 + 69} = \frac{2}{93}$$

$$p(catarinenses|esporte) = \frac{1 + 1}{24 + 69} = \frac{2}{93}$$

$$p(cultural|esporte) = \frac{0 + 1}{24 + 69} = \frac{1}{93}$$

- Por fim é realizada a soma dos logaritmos de todas as probabilidades

$$p(texto|Esporte) = \log\left(\frac{3}{8}\right) + 3 * \log\left(\frac{2}{93}\right) + \log\left(\frac{1}{93}\right) \cong -7,3968$$

Com uma maior base de teste o algoritmo fica mais preciso, e se cada vez que um novo texto que for classificado for salvo na base de testes, o algoritmo se torna “autodidata” e aprende com o tempo, ele se mostrou eficiente em classificar os textos em seus gêneros, porém a massa de testes

usada era pequena, ao colocar uma grande quantidade de informações provavelmente ele se mostraria muito mais lento, porém bastante eficiente. A execução do algoritmo em python foi em torno de 0.013 seg enquanto o em C rodou em 0.001 seg, 13 vezes mais rápido, e essa diferença se torna mais notável com o aumento dos textos e do número de classes pré-existentes, porém é compreensível visto que python é interpretada e com um nível de abstração muito alto.

4- Conclusão

O algoritmo de classificação Naïve Bayes, mostrou-se eficiente em vista dos testes propostos, podemos apontar problemas como alto custo computacional e alto índice de fidelidade nos corpos bases para treinamento do mesmo, mas estas desvantagens não são um grande fator impeditivo ao analisarmos a dificuldade de implementação, obtivemos mais trabalho ao utilizarmos a linguagem C do que quando utilizamos a linguagem Python (código não anexado a este artigo), mas mesmo assim tivemos relativa facilidade ao codificá-lo, o que acaba se tornando um fator positivo.

Uma das vantagens da classificação bayesiana é o fato de que com poucas palavras em comum entre os textos analisados, podemos ter uma classificação aproximada do mesmo, fato que quanto menor essa proximidade de palavras menor é o fator de aproximação da classe.

Ao rodarmos os códigos em Python e em C, notamos uma diferença substancial de tempo (X segundos), no qual o escrito em C foi mais rápido, isto se dá pela estrutura de funcionamento de cada linguagem e pela arquitetura escolhida diferencialmente para cada uma devido a suas particularidades.

Podemos aplicar também o algoritmo de Bayes na extração de informação de uma massa de dados, facilitando assim tomadas de decisão e possíveis previsões baseadas no passado.

Referências

Han, J. and Kamber, M. (2006). Data Mining: Concepts and Techniques. Morgan

Kaufmann Publishers, 2nd ed.

Manning, C. D.; Raghavan, P. and Schütze H. (2008). Introduction to Information

Retrieval. Cambridge University Press.

SIMÕES, Priscyla Waleska Targino de Azevedo; NASSAR, Silvia Modesto; PIRES, Maria Marlene de Souza. Sistema de Apoio na Avaliação da Falência do Crescimento Infantil. In: CONGRESSO BRASILEIRO DE COMPUTAÇÃO, Workshop de Informática Aplicada à Saúde, 2001.