Análise de sentimentos utilizando a implementação do algoritmo Naive Bayes da biblioteca NLTK

Projeto de Processamento de Linguagem Natural
2° quadrimestre de 2018

Universidade Federal do ABC

Discente:
11043512 - Lucas Zanferrari Caraça

Santo André, 07 de maio de 2018

Resumo

Este artigo visa apresentar um exemplo de caso de uso do algoritmo de Naive Bayes (do inglês Bayes "Ingênuo") para a solução de um problema de classificação do campo de Processamento de Linguagem Natural. O objetivo central é implementar, treinar e validar um modelo de análise de sentimentos baseado na base de dados SAR14. Esta base de dados foi escolhida por ser grande o suficiente e significativamente relevante na comunidade de Processamento de Linguagem Natural para treinar modelos de análise de sentimento.

Para este fim, a linguagem de programação Python (versão 3.6) será utilizada, em conjunto com a biblioteca NLTK (Natural Language ToolKit) e outras bibliotecas frequentemente utilizadas em computação científica: scikit-learn, numpy, pandas e wordcloud.

O idioma em que o modelo será treinado será o inglês. Isto se deve exclusivamente ao fato de que a expressiva maioria das bases de dados já classificadas disponíveis online - de forma gratuita - são de conteúdo escrito nesta língua.

Palavras-chave: NLTK, Naive Bayes, Sentiment Analysis, Natural Language Processing, Conditional Probability.

1 Introdução

O algoritmo de Naive Bayes (Jurafsky & Martin, 2000) consiste em um algoritmo de classificação. Ou seja, o seu objetivo é, após o treinamento, gerar um modelo capaz de atribuir uma classe escolhida dentre um grupo definido de classes, a um vetor de atributos previamente desconhecido. Ele se baseia no teorema de Bayes, que é explicado em mais detalhes na seção de Fundamentos, para fazer estas atribuições.

Além disso, Naive Bayes também é um algoritmo de aprendizado supervisionado, o que significa que ele deriva uma função de classificação capaz de atribuir classes a vetores de atributos desconhecidos analisando uma base de dados anteriormente classificada. Note que isto significa que é necessário possuir uma base de dados (ou corpus) já classificada para que o treinamento do modelo seja possível.

Note que a precisão da função de classificação derivada depende de diversos fatores, mas em especial do volume, da exatidão das classificações no que tange a representação da realidade e da diversidade da distribuição de atributos presente nos registros da base de dados de treinamento.

A base SAR14 (Nguyen *et al.* , 2014) consiste em aproximadamente 234 mil avaliações de filmes provenientes de usuários do IMDb. Cada linha da matriz representada no arquivo contém o texto da avaliação do filme (primeira coluna) e a nota dada pelo usuário (segunda coluna) em uma escala de 1 a 10. Avaliações de filmes tendem a gerar boas bases de dados para treinar modelos de análise de sentimentos (representado aqui como subproblema do problema de classificação) pois costumam ser altamente polarizadas e tão volumosas quanto necessário. Para este artigo, apenas uma parte desta base foi utilizada - cerca de 1750 avaliações - dado que devido as dimensões da matriz de atributos da base de treinamento (186880 x 296936) e a complexidade do classificador de Naive Bayes ($\mathcal{O}(Nd)$ com N sendo o número de registros na base de dados e d, o número de atributos), o treinamento do modelo poderia levar mais de um ano para ser concluído no equipamento descrito na seção de Conclusões.

2 Metodologia

Neste estudo, a heurística de conversão de notas em sentimentos foi a seguinte: notas abaixo de 6 representam uma opinião negativa sobre o filme, enquanto notas maiores ou iguais a 6 representam uma opinião positiva. Este número é chamado de Limiar Positivo.

Testes estatísticos (cálculo de média da acurácia) foram efetuados para que este valor fosse definido. O modelo foi treinado repetidas vezes, variando o limiar positivo de 5 a 8, e sua precisão foi medida a cada vez. Com precisões de aproximadamente oitenta e cinco porcento, o modelo com limiar positivo igual a 6 foi o escolhido. Sabe-se que o critério de escolha é ainda vago, mas não cabe no escopo deste experimento definir o melhor limiar positivo possível. Estratégias para tal são discutidas na seção de Próximos Passos.

O primeiro passo foi dividir a base de dados em base de treinamento e base de validação. 20% da base foi destinado à validação (ou testes) e 80%, à treinamento. Esta distribuição é comum em aplicações de aprendizado de máquina. Após isso, foram apresentadas, em forma de nuvem de palavras, as palavras mais frequentemente associadas a avaliações positivas e negativas. Ambas as nuvens podem ser visualizadas nas figuras 1 e 2.

Figura 1: *Nuvem de palavras das avaliações positivas*

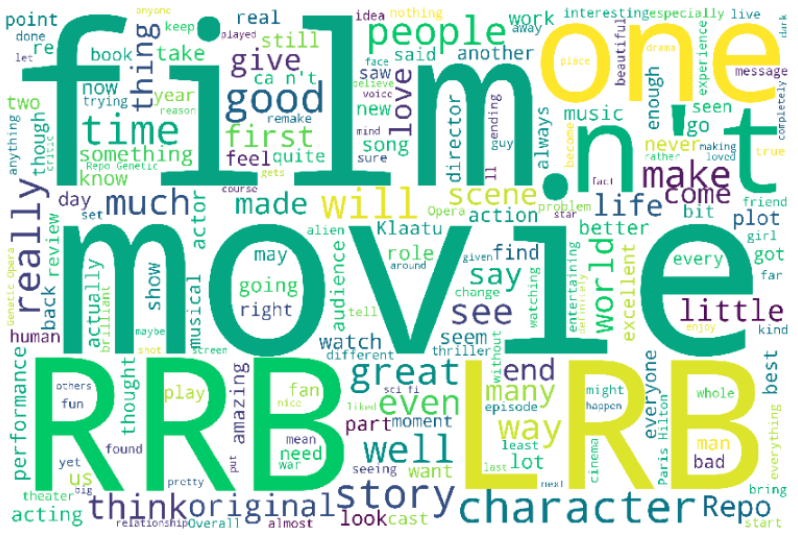
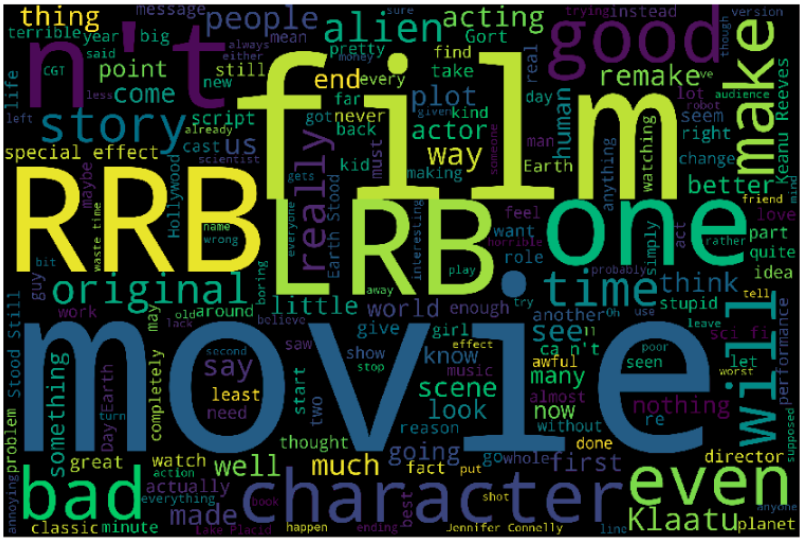


Figura 2: *Nuvem de palavras das avaliações negativas*

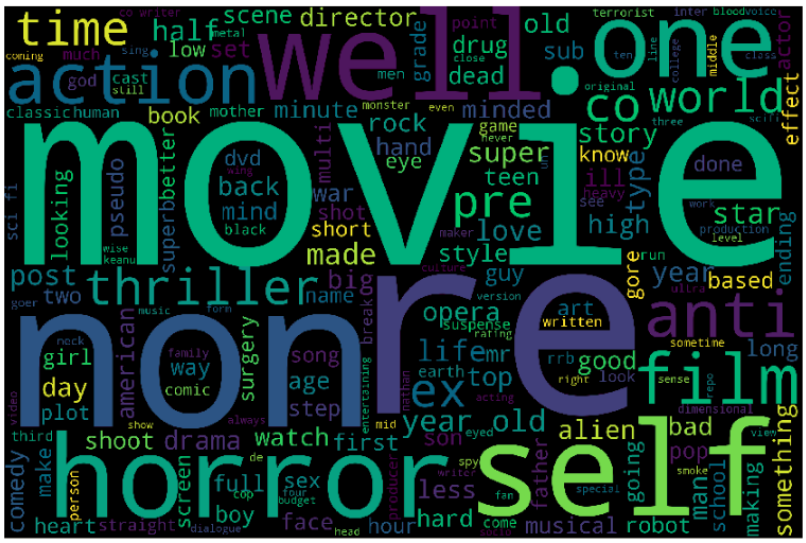


A seguir, as avaliações foram tokenizadas, os tokens considerados inúteis foram removidos (stopwords, sinais de pontuação e qualquer palavra com menos de 3 caracteres) e a heurística de conversão das notas das avaliação em sentimentos, cujo código segue abaixo, foi aplicada. Note que o parâmetro pos_threshold refere-se ao limiar positivo.

```
def convert_grade_to_sentiment(grade, pos_threshold):
    return "positive" if grade >= pos_threshold else "negative"
```

Funções para a extração da do vocabulário, da bag-of-words e dos atributos textuais que compõem a matriz de atributos no formato apresentado na seção de Fundamentos foram definidas. A nuvem de palavas da base de dados completa foi mostrada. Ela segue abaixo, na figura 3.

Figura 3: *Nuvem de palavras de todas as avaliações*



É possível observar que a maioria possui um tom nêutro ao se referir ao universo de produções cinematrográficas, ou seja, podem ser utilizadas tanto em contextos positivos quanto negativos.

A implementação do algoritmo de Naive Bayes contida na biblioteca NLTK (Loper & Bird, 2002) foi utilizada. O modelo foi treinado sobre a matriz de atributos textuais da base de treinamento.

Por fim, um teste de precisão do modelo foi efetuado. Utilizou-se o modelo para predizer a classe de cada vetor na base de validação e logo após, a predição foi comparada com o valor real. A precisão observada foi de aproximadamente 85%, tanto para avaliações positivas quanto negativas.

Os scripts usados neste experimento foram inspirados no trabalho de Peter Nagy, contido em https://www.kaggle.com/ngyptr/python-nltk-sentiment-analysis

3 Conclusões

O algoritmo de Naive Bayes costuma funcionar suficientemente bem para o problema de análise de sentimentos, com exceção de quando avaliam textos contendo ironia e/ou sarcasmo. É esperado que, principalmente quando analisando de textos de tom negativo (pois é neste contexto que ironia e sarcasmo são mais frequentemente utilizados), ocorram resultados falsos, mas devido ao volume da base de dados utilizada e a sua natureza - avaliações de filmes tendem a possuir um caráter mais sério do que interações entre usuários de redes sociais, por exemplo - esta expectativa não prejudicou o treinamento do modelo de forma significativa.

4 Próximos Passos

Como a escolha de avaliações de filmes para a base de treinamento e para a de testes ocorre de forma aleatória, sabe-se que a escolha do limiar positivo igual a 6, apesar de intuitiva, não possui evidências conclusivas que a suporte. Implementar Validação Cruzada para a escolha do limiar positivo faria com que estas evidências fossem obtidas.

Outro ponto que poderia ser melhorado seria fazer com que o procedimento de extração da matriz de atributos gerasse uma matriz termo-documento, pois da forma como a implementação foi efetuada, a frequência de palavras em cada documento é desprezada.

Além disso, o treinamento do modelo poderia levar mais de um ano para ser concluído em um computador com Windows 10, 8GB de memória RAM e processador Core i7 de quarta geração com 2.5GHz por núcleo. Este tempo pode ser reduzido se o algoritmo de Naive Bayes for refatorado para ser processado paralelamente por dois ou mais núcleos presentes no processador.

O tempo de treinamento também pode ser reduzido através da aplicação de técnicas de stemming ou SVD na matriz de atributos extraída da base de dados, pois assim o número de dimensões na matriz seria reduzido. Ou ainda applicando feature hashing na matriz, pois assim as consultas às celulas da matriz seriam agilizadas.

5 Fundamentos

Os conceitos brevemente explorados abaixo resumem a base teórica sobre a qual este trabalho será desenvolvido.

5.1 Terminologia

- Tokenização: divisão do texto em um vetor de palavras.
- Bag-of-words: distribuição de frequências de palavras pelo corpus.

5.2 Representação de texto em aprendizado de máquina

Algoritmos de aprendizado de máquina são capazes de processar matrizes, portanto, para que texto seja fornecido, é preciso formatá-lo em uma matriz.

As seguintes sentenças podem ser transformadas em uma matriz binária da seguinte forma:

- "este filme é ruim."
- "adorei este figurino."
- "este filme é bom."

este	filme	é	ruim	adorei	figurino	bom
1	1	1	1	0	0	0
1	0	0	0	1	1	0
1	1	1	0	0	0	1

Onde cada coluna é uma palavra existente no grupo de sentenças e, se o valor na linha que representa um sentença for 1, isso significa que esta palavra está presente nela. Analogamente, se for 0, significa que a palavra está ausente.

O intuito do programa a ser desenvolvido é, uma vez treinado o algoritmo de Naive Bayes, ser capaz de classificar a matriz da seguinte maneira.

este	filme	é	ruim	adorei	figurino	bom	classe
1	1	1	1	0	0	0	negativo
1	0	0	0	1	1	0	positivo
1	1	1	0	0	0	1	positivo

Ou com as classes que convenham, uma vez escolhido o assunto.

5.3 Teorema de Bayes

O teorema de Bayes visa descrever a probabilidade de um evento c ocorrer, dado que outro evento D já ocorreu.

$$P(c \mid D) = \frac{P(D \mid c)\, P(c)}{P(D)}$$

No contexto do classificador de Naive Bayes aplicado a processamento de linguagem natural, D é o vetor de palavras de um corpo textual (documento), cada uma representada como um atributo da entrada, e $c \in C$ é uma classe em que D pode ser enquadrado (positivo ou negativo, por exemplo). C é o conjunto de classes possíveis.

Como o objetivo do classificador de Naive Bayes é computar a probabilidade de um determinado vetor de palavras pertencer a uma classe e, após isso, atribuir este vetor de palavras à classe com a probabilidade máxima (ou seja, comparar probabilidades computadas com $P(D)$ constante entre elas), não é preciso computar $P(D)$.

$P(D \mid c)$ pode ser escrito como a probabilidade condicional das palavras $w_1, w_2, ..., w_n \in D$, dada um classe $c \in C$.

$$P(D \mid c) = P(w_1, w_2, ..., w_n \mid c) = P(w_1 \mid c) * P(w_2 \mid c) * ... * P(w_n \mid c)$$

Note que, caso existam $P(D \mid c) = 0$ ou $P(c) = 0$ provenientes da base de dados de treinamento, a fórmula de Bayes é zerada. Além disso, é possível observar que um número muito grande de palavras em D faz com que $P(D \mid c) \to 0$, pois é o produto de diversas probabilidades cujos valores são restritos entre 0 e 1, e isso prejudica a avaliação das grandezas pelo algoritmo, portanto, na prática, são utilizados os logaritmos dos produtos destes componentes.

$$\hat{c} = max_{c \in C} \left(log(P(c)) + \sum_{i}^{n} log(P(w_i \mid c)) \right)$$

Onde $\hat{c}$ é classe predita para D.

6 Apêndice

O código desenvolvido para este experimento acompanha este artigo com o nome de min_SAR14_naive_bayes.ipynb, no formato de um Jupyter Notebook, assim como o arquivo da base de dados usada (minSAR14.csv).

Referências

Jurafsky, Daniel, & Martin, James H. 2000. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.* 1st edn. Upper Saddle River, NJ, USA: Prentice Hall PTR. Chap. Naive Bayes Classification and Sentiment.

Loper, Edward, & Bird, Steven. 2002. Nltk: The natural language toolkit. *Pages 63–70 of: Proceedings of the acl-02 workshop on effective tools and methodologies for teaching natural language processing and computational linguistics - volume 1.* ETMTNLP '02. Stroudsburg, PA, USA: Association for Computational Linguistics.

Nguyen, Dai Quoc, Vu, Thanh, & Pham, Son Bao. 2014. Sentiment classification on polarity reviews: An empirical study using rating-based features. *Pages 128–135 of: Proceedings of the 5th workshop on computational approaches to subjectivity, sentiment and social media analysis.* Baltimore, Maryland: Association for Computational Linguistics.