Análise de algoritmos de reconhecimento de padrões

Antônio Adelino da S. Neto[a], Armstrong Lohãns de M. G. Quintino[b]

Garanhuns, Brasil

[a]*antonio.asn03@gmail.com*
[b]*lohansdemelo1108@gmail.com*

Abstract

O objetivo principal desse trabalho foi a elaboração e o estudo de dois algoritmos para sistemas de aprendizado de máquina, os quais trabalhariam na classificação de textos em linguagem natural. Tais programas foram o algoritmo da Árvore de Decisão e o algoritmo de Naive Bayes. Inicialmente são mostrados os conceitos básicos sobre cada algoritmo de decisão acima citado. Depois haverá uma apresentação das técnicas usadas para a análise e as devidas conclusões.

Keywords: Reconhecimento de Padrões, Árvore de Decisão, Naive Bayes

1. Introdução

Os seres humanos e alguns outros animais possuem, entre outras habilidades, a aptidão no reconhecimento de padrões. O ser humano, especificamente, possui essa capacidade muito bem desenvolvida e tem uma enorme facilidade no reconhecimento formas, dando a elas significado e valor. Dentre elas pode-se citar a fisionomia de outros seres humanos, formas animais e vegetais, características pessoais e afins.

Essa habilidade sempre foi muito importante, pois foi por meio dela que a espécie humana conseguiu desenvolver-se com mais facilidade ao longo do tempo, uma vez que ela permite a assimilação e inferência de características em formas aparentemente semelhantes. Partindo dessa premissa, é possível notar a relevância dessa aptidão em reconhecimento para o ser humano, em especial o reconhecimento de padrões, visto que é por meio dela que consegue-se inferir em formas desconhecidas julgamentos prévios a partir de conhecimentos anteriores.

Com isso, afirma-se então que toda e qualquer forma de reconhecimento de padrões, por indivíduos, dá-se a partir de uma experiência passada. Dessa maneira é possível perceber que a destreza, ou não, no reconhecimento de padrões está

diretamente vinculada aos estímulos que cada indivíduo foi submetido ao longo de sua vida [1].

Partindo dessas afirmativas, o presente artigo expõe um estudo que busca a análise comparativa de dois algoritmos. O algoritmo da Árvore de Decisão e o algoritmo de Naive Bayes, ambos voltados a classificação de dados baseando-se nos princípios de aprendizagem de máquina.

A análise desses algoritmos dá-se por meio da classificação de textos em linguagem natural, onde os classificadores recebem os textos e inferem a eles o sentido do que está escrito de acordo com classes anteriormente estabelecidas.

2. Referencial Teórico

2.1. Algoritmo da Árvore de Decisão

As Árvores de Decisão são técnicas muito populares de aprendizado de máquina, são aplicadas às tarefas de classificação e regressão. Esta técnica é caracterizada pelo seu modelo resultante, o qual é codificado como uma estrutura em árvore [2].

As árvores de decisão são algoritmos que buscam a classificação dos dados a partir da estruturação em árvore. O algoritmo divide um conjunto de dados em subconjuntos menores. Sabendo que o código estrutura-se em árvore, cada nó folha representa uma decisão.

Para chegar em uma decisão, o algoritmo comporta-se da seguinte maneira, com base nos valores dos recursos das instâncias, as árvores de decisão classificam os dados. Cada nó representa um recurso em uma instância da árvore de decisão que deve ser classificada, e cada ramo representa um valor [3].

Sabendo disso, percebe-se que cada dado, para ser classificado, passa por um conjunto finito de nós, tal conjunto é definido como as regras de classificação, pois a partir desse conjunto é possível saber o passo a passo do algoritmo, mostrando assim todas as regras que levaram a classificação daquela única instância. A principal vantagem do uso das árvores de decisão está justamente na capacidade do retorno dos passos para a decisão e não unicamente no resultado da classificação.

2.2. Algoritmo de Naive Bayes

Além das árvores de decisão, pode-se também fazer uso de outros tipos de classificadores, entre eles destaca-se o Naive Bayes, o qual possui uma análise dos dados a partir de conceitos probabilísticos, diferenciando-se das árvores de decisão.

O algoritmo de Naive Bayes é um classificador probabilísticos simples (baseado no Teorema de Bayes), tem como base em uma suposição comum de que todos os recursos são independentes um do outro [4]. A partir disso, ele desconsidera completamente a correlação entre todas as variáveis, tratando cada variável de forma independente, esse algoritmo é frequentemente aplicado em processamento de linguagem natural.

Uma das principais vantagens do classificador Naive Bayes é que ele requer apenas uma pequena quantidade de dados iniciais de treinamento para poder estimar as médias e variações das variáveis necessárias para classificação [5].

3. Ferramentas usadas

3.1. Linguagem Python

Antes de iniciar a implementação optamos por usar a linguagem de programação Python. Essa linguagem, além de ser uma das mais populares do mundo, é vastamente usada na produção de softwares e algoritmos voltados aos conceitos de aprendizagem de máquina, tanto no meio acadêmico quanto na indústria. Pode-se afirmar também que a linguagem de programação Python é uma linguagem de fácil manipulação e de grande eficiência produtiva o que facilita todo o processo de codificação.

3.2. Módulo Sickit-Learn

Partindo dessa escolha inicial, foi possível utilizar o módulo Sickit-Learn para Python. Tal módulo integra em si uma grande quantidade de algoritmos de aprendizagem de máquina que são voltados para problemas supervisionados e não supervisionados [6].

Além disso, essa biblioteca Python tem como característica a simplificação do uso de algoritmos de aprendizagem de máquina buscando a sua popularização. Essa difusão dá-se por meio da grande facilidade de uso, do bom desempenho e da sua documentação detalhada. Esse módulo ainda procura incentivar o seu uso, por meio de dependências mínimas e licença simplificada, em ambientes acadêmicos e comerciais de todo o mundo [6].

Diante dessas características encontradas na linguagem de programação Python e na biblioteca Sickit-Learn a implementação dos algoritmos, que servem de experimento para o presente artigo, tornou-se mais rápida e mais objetiva.

3.3. Wisdom Quotes (Base de dados)

Para a criação da base de dados inicial dos classificadores foi necessário a utilização do site `http://wisdomquotes.com` a fim de escolher textos aleatórios que fossem divididos por classificadores. O Wisdom Quotes conta com uma grade variedade de citações separadas por temas, permitindo a relação intuitiva citação-tema, o que vem a facilitar a criação e manipulação da base de dados dos classificadores.

4. Algoritmo e testes

4.1. Criação da base de dados

Antes de iniciar a implementação dos algoritmos brutos de aprendizado de máquina (Árvore de Decisão e Naive Bayers) tivemos que coletar as citações do site Wisdom Quotes. Para isso implementamos um script simples em Python (Crawler) que tinha como finalidade apenas a coleta das citações do endereço URL que passamos e colocar em arquivos de texto. Com o Crawler o processo de coleta de dados tornou-se muito mais rápido e eficaz. A coleta foi feita em três páginas do mesmo site, cada página continha citações de uma única categoria, as três categorias foram escolhidas randomicamente e são elas:

1. Peace (do ingês, "Paz")
2. Success (do ingês, "Sucesso")
3. Silence (do ingês, "Silêncio")

Com as citações salvas em arquivos de texto, tivemos que escolher aleatoriamente cento e cinquenta de cada classificador, a fim de deixar a base de dados uniforme, totalizando quatrocentos e cinquenta frases.

A partir disso, armazenamos essas frases em dois vetores, o vetor de treino e o vetor de teste. Essas listas continham respectivamente sessenta porcento e quarenta por cento das citações, escolhidas aleatoriamente entre si. Essas porcentagens foram estabelecidas visando um melhor desempenho dos classificadores, uma vez que a quantidade de dados para treino é um pouco maior quando comparada a quantidade de dados para testes.

4.2. Árvore de Decisão

Com a base de dados inicial pronta, implementamos a arvore de decisão. Para usar a funçãopara criação de uma árvore de decisão no Sickit-Learn é necessário inicialmente vetorizar as frases de teste, a função (também do Sickit-Learn)

```
x = CountVectorizer()
treinoFrase = x.fit_transform(treinoFrase).toarray()
```

Figura 1: Vetorizando frases de teste

.fit_transform() do *CountVextorizer()* vetoriza as frases em forma de matriz, além de contar a repetição de cada palavra na frase, trecho ilustrado na Figura 1.

Com as frases devidamente vetorizadas, criamos a árvore de decisão a partir da função *tree.DecisionTreeClassifier()* e a treinamos com o comando *.fit(parm1, parm2)*, como mostrado na Figura 2, passando como parâmetro as frases vetorizadas e a lista de classificações. Cada frase vetorizada está localizada na posição de índice correspondente ao seu classificador que está na lista de classificadores.

```
arvore = tree.DecisionTreeClassifier()
arvore.fit(treinoFrase, treinoClass)
```

Figura 2: Instanciando e treinando árvore de decisão

Após a criação da árvore, colocamos a base de testes para validar as classificações, a função *.score(parm1, parm2)* retorna o percentual de acerto dos testes passados no segundo parâmetro, no primeiro parâmetro está o conjunto contendo todas as palavras conhecidas pela árvore (*Bag of Words*), veja na Figura 3.

```
arvore.score(bagOfWords, testeClass)
```

Figura 3: Função que retorna a porcentagem de acertos da estrutura

Com a árvore de decisão montada também é possível obter as regras de classificação de uma determinada frase passada, para isso usamos a função, também do do Sickit-learn, *.decision_path()*, conforme mostra a Figura 4.

Com isso, terminamos a implementação do algoritmo da árvore de decisão e suas funcionalidades principais, dando destaque a possibilidade de listar o caminho de decisão de uma citação.

4.3. Naive Bayers

- Bullet point one
- Bullet point two

```
arvore.decision_path(frase)
```

Figura 4: Função que retorna a as regras de decisão da árvore

1. Numbered list item one
2. Numbered list item two

4.4. Subsection One

Quisque elit ipsum, porttitor et imperdiet in, facilisis ac diam. Nunc facilisis interdum felis eget tincidunt. In condimentum fermentum leo, non consequat leo imperdiet pharetra. Fusce ac massa ipsum, vel convallis diam. Quisque eget turpis felis. Curabitur posuere, risus eu placerat porttitor, magna metus mollis ipsum, eu volutpat nisl erat ac justo. Nullam semper, mi at iaculis viverra, nunc velit iaculis nunc, eu tempor ligula eros in nulla. Aenean dapibus eleifend convallis. Cras ut libero tellus. Integer mollis eros eget risus malesuada fringilla mattis leo facilisis. Etiam interdum turpis eget odio ultricies sed convallis magna accumsan. Morbi in leo a mauris sollicitudin molestie at non nisl.

Treatments	**Response 1**	**Response 2**
Treatment 1	0.0003262	0.562
Treatment 2	0.0015681	0.910
Treatment 3	0.0009271	0.296

Tabela 1: Table caption

4.5. Subsection Two

Donec eget ligula venenatis est posuere eleifend in sit amet diam. Vestibulum sollicitudin mauris ac augue blandit ultricies. Nulla facilisi. Etiam ut turpis nunc. Praesent leo orci, tincidunt vitae feugiat eu, feugiat a massa. Duis mauris ipsum, tempor vel condimentum nec, suscipit non mi. Fusce quis urna dictum felis posuere sagittis ac sit amet erat. In in ultrices lectus. Nulla vitae ipsum lectus, a gravida erat. Etiam quam nisl, blandit ut porta in, accumsan a nibh. Phasellus sodales euismod dolor sit amet elementum. Phasellus varius placerat erat, nec gravida libero pellentesque id. Fusce nisi ante, euismod nec cursus at, suscipit a enim. Nulla facilisi.

Integer risus dui, condimentum et gravida vitae, adipiscing et enim. Aliquam erat volutpat. Pellentesque diam sapien, egestas eget gravida ut, tempor eu nulla.

Vestibulum mollis pretium lacus eget venenatis. Fusce gravida nisl quis est molestie eu luctus ipsum pretium. Maecenas non eros lorem, vel adipiscing odio. Etiam dolor risus, mattis in pellentesque id, pellentesque eu nibh. Mauris nec ante at orci ultricies placerat ac non massa. Aenean imperdiet, ante eu sollicitudin vestibulum, dolor felis dapibus arcu, sit amet fermentum urna nibh sit amet mauris. Suspendisse adipiscing mollis dolor quis lobortis.

$$e = mc^2 \tag{1}$$

5. The Second Section

Reference to Section 1. Etiam congue sollicitudin diam non porttitor. Etiam turpis nulla, auctor a pretium non, luctus quis ipsum. Fusce pretium gravida libero non accumsan. Donec eget augue ut nulla placerat hendrerit ac ut mi. Phasellus euismod ornare mollis. Proin tempus fringilla ultricies. Donec pretium feugiat libero quis convallis. Nam interdum ante sed magna congue eu semper tellus sagittis. Curabitur eu augue elit.

Aenean eleifend purus et massa consequat facilisis. Etiam volutpat placerat dignissim. Ut nec nibh nulla. Aliquam erat volutpat. Nam at massa velit, eu malesuada augue. Maecenas sit amet nunc mauris. Maecenas eu ligula quis turpis molestie elementum nec at est. Sed adipiscing neque ac sapien viverra sit amet vestibulum arcu rhoncus.

Vivamus pharetra nibh in orci euismod congue. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Quisque lacus diam, congue vel laoreet id, iaculis eu sapien. In id risus ac leo pellentesque pellentesque et in dui. Etiam tincidunt quam ut ante vestibulum ultricies. Nam at rutrum lectus. Aenean non justo tortor, nec mattis justo. Aliquam erat volutpat. Nullam ac viverra augue. In tempus venenatis nibh quis semper. Maecenas ac nisl eu ligula dictum lobortis. Sed lacus ante, tempor eu dictum eu, accumsan in velit. Integer accumsan convallis porttitor. Maecenas pretium tincidunt metus sit amet gravida. Maecenas pretium blandit felis, ac interdum ante semper sed.

In auctor ultrices elit, vel feugiat ligula aliquam sed. Curabitur aliquam elit sed dui rhoncus consectetur. Cras elit ipsum, lobortis a tempor at, viverra vitae mi. Cras sed urna sed eros bibendum faucibus. Morbi vel leo orci, vel faucibus orci. Vivamus urna nisl, sodales vitae posuere in, tempus vel tellus. Donec magna est, luctus non commodo sit amet, placerat et enim.

Referências

[1] P. Prado, A. Monteiro, Pattern recognition algorithms 5 (2008).

[2] G. Nuti, L. A. J. Rugama, A.-I. Cross, A bayesian decision tree algorithm, stat 1050 (2019) 11.

[3] R. Pandya, J. Pandya, C5. 0 algorithm to improved decision tree with feature selection and reduced error pruning, International Journal of Computer Applications 117 (2015) 18–21.

[4] S. Xu, Bayesian naïve bayes classifiers to text classification, Journal of Information Science 44 (2018) 48–59.

[5] S. Vijayarani, S. Dhayanand, Liver disease prediction using svm and naïve bayes algorithms, International Journal of Science, Engineering and Technology Research (IJSETR) 4 (2015) 816–820.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.