UTILIZAÇÃO DE TÉCNICAS DE APRENDIZADO DE MÁQUINA PARA CLASSIFICAR COMENTÁRIOS DE FILMES.

[1] Daniel Winter Santos ROCHA, [1] Júlio César Machado ÁLVARES, [1] Marcus Vinícius Rodrigues CAMPOS, [2] Laerte Mateus RODRIGUES

[1] Alunos de Graduação em Engenharia de Computação pelo IFMG - *campus* Bambuí. [2] Professor do IFMG - *campus* Bambuí

RESUMO

O objetivo do trabalho é a utilização de quatro algoritmos de *machine learning* para a classificação de uma base de dados de comentários de filmes retirados do IMDB. Os algoritmos utilizados foram: *K-Nearest Neighbors* (KNN), *Neural Network Perceptron*, *Support Vector Machine* (SVM) e *Naive Bayes*. Para o desenvolvimento do trabalho foram implementados dois algoritmos, sendo eles KNN e *Perceptron*, utilizando a linguagem de programação *Python* e as bibliotecas *scikit-learn* e *numpy* para tratamento dos dados e para utilização dos outros algoritmos.

Palavras-chave: *Machine Learning*, Classificação de Texto, KNN, Perceptron, Naive Bayes, SVM.

1 INTRODUÇÃO

Os algoritmos de *machine learning* são muito utilizados para automatização de tarefas, classificação de dados de forma regressiva. Uma das técnicas de aprendizado de *machine learn* adotadas pelos algoritmos utilizados foi a de aprendizado supervisionado.

Este tipo de técnica define a observação de alguns pares de entradas e saídas de forma a modelar que, dado uma entrada X resultará em uma ação/classificação Y baseadas em sua fase de treinamento. Y é definido aproximando uma função $y = h'$ de uma função h não conhecida, sendo essa a função real, tendo base os pontos de treinamento.

Alguns problemas apresentados pelos algoritmos de aprendizado supervisionado são: *overfitting* e *underfitting*. O *overfitting* consiste em uma aproximação da função h' muito específica, enviesando o modelo de classificação, ou seja, deixando-o muito específico e sensível à novas observações. O *underfitting* por sua vez, consiste em uma aproximação da função h' muito fraca, deixando o modelo muito propenso a falhas, ou seja, o modelo é pouco generalizado.

1.1 *K-Nearest Neighbors*

O KNN é um algoritmo de aprendizado supervisionado que procura classificar um determinado elemento com base nas proximidades dos seus vizinhos. Dado um determinado elemento ele será classificado com base nos seguintes fatores.

Primeiro é calculada a distância do objeto em questão até os demais. Em seguida, armazena-se os K objetos com a menor distância e realiza-se uma votação majoritária entre eles, assim, a classe vencedora é rotulada como a classe do novo objeto.

1.2 *Perceptron*

A Perceptron é um algoritmo de aprendizado supervisionado baseado no funcionamento do cérebro humano, simulando os neurônios e suas sinapses. O algoritmo, em sua forma mais simples, consiste de N neurônios de entrada, sendo N o conjunto de dados de cada amostra, e 1 neurônio de saída.

O funcionamento do algoritmo consiste em épocas de aprendizado. Para cada época é feito um reajuste nos pesos, sendo estes a aproximação da função objetivo para a separação das classes de dados.

1.3 *Support Vector Machines* (SVM)

O SVM é um algoritmo de aprendizado supervisionado cujo princípio de classificação é estabelecer um hiperplano ótimo entre as classes na sua base de dados para definir a qual classe pertence determinada amostra. O SVM sofre de dois problemas: O primeiro deles, *outlier,* acontece quando um determinado indivíduo apresenta características incondizentes com seu padrão de características, sendo ele um ponto fora do padrão.

1.4 *Naive Bayes*

O Naive Bayes é um algoritmo de aprendizado supervisionado cuja principal característica é o motivo de ter recebido o nome "*Naive*"(ingênuo), onde desconsidera a correlação entre as variáveis . Por exemplo.: se um carro é azul, mede 3 metros e é quadrado, ele desconsidera a correlação entre eles e trata cada atributo individualmente. O nome "*Bayes"* vem do teorema probabilístico de Bayes. O teorema adota uma probabilidade de um determinado evento com base nos conhecimentos prévios adquiridos, relacionar-se a um evento futuro que possam estar ligado ou semelhante ao evento prévio. No escopo do trabalho, o algoritmo categoriza os comentários baseado na frequência das palavras utilizadas para identificar se o comentário é bom ou ruim que também pode ser utilizado para filtragem de spam em e-mails.

Por ser um algoritmo relativamente simples e rápido possui um desempenho relativamente maior que os outros .Além desta vantagem, ele necessita de uma pequena base de dados de teste para uma boa classificação e para uma boa precisão.

$$P(A|B) = P(B|A) * P(A) / P(B),$$

sendo P(A|B) a probabilidade posterior, P(B|A) a probabilidade conhecida, P(A) a primeira classe e P(B) a segunda classe.

1.5 Base dos Dados

A base de dados a ser utilizada nos testes consiste de 25.000 comentários sobre filmes, retirados do IMDB, sendo esses com classificação 1 ou 10. A divergência entre a classificação dos comentários foi feita com intuito de demonstrar o funcionamento dos algoritmos, sendo que as características entre eles é discrepante e facilitaria a classificação.

2 DESENVOLVIMENTO

Para o desenvolvimento do trabalho, foi utilizado a linguagem de programação *Python* versão 3.6.3, as bibliotecas *numpy, scipy e scikit-learn* além do ambiente de desenvolvimento *Sublime-Text 3* e um terminal para execução dos algoritmos.

A primeira parte do trabalho proposto consiste em tratar os comentários do IMDB. Para o tratamento dos dados, foi utilizado a técnica *TFIDF*, sendo essa uma técnica para parametrizar a importância de um termo em um documento em relação a uma coleção de documentos. O valor *TFIDF* é diretamente proporcional ao aumento do número de ocorrências do termo no documento, esse valor é equilibrado pela frequência da palavra na coleção. (TFIDF, 2017)

Em seguida, os algoritmo KNN e *Neural Network Perceptron* foram desenvolvidos sem o auxílio de bibliotecas. A estruturação do algoritmo KNN foi feita em uma função principal e uma auxiliar, sendo a principal o julgamento dos K-vizinhos mais próximos ao ponto utilizado e a função secundária para calcular a distância euclidiana entre os pontos. A estruturação da *perceptron* foi feita em uma função principal de treino, sendo essa a que ajusta o vetor de pesos, adequando a função esperada à função desejada. Os resultados foram gerados utilizando a metodologia de validação cruzada com 90% dos dados para treino e 10% para testes.

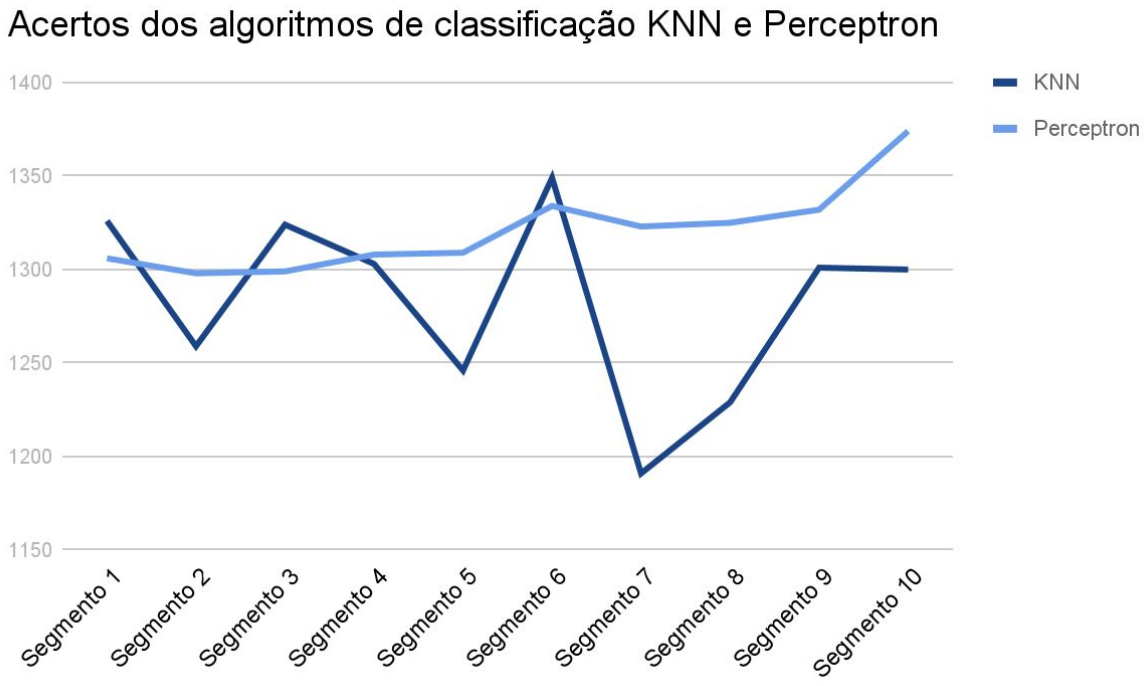
Para os algoritmos SVM e *Naive Bayes*, foi utilizado a biblioteca *scikit-learn*. Para a SVM, foi utilizado o módulo *support vector classification* e aplicado a validação cruzada. O mesmo, foi feito para o algoritmo Naive Bayes, utilizando o módulo *Gaussian Naive Bayes* e aplicada a validação cruzada.

2.1 Resultados

O algoritmo KNN apresentou uma média de acertos de 51.4%, com 1285 acertos. O menor resultado apresentado foi de 1191 acertos, somando 47.64% e o maior resultado foi de 1349 acertos, somando 53.96%, como mostra a Figura 1 e a Tabela 1.

O algoritmo *Perceptron* apresentou média de acertos de 52.8%, com 1320 acertos. O menor resultado apresentado foi de 1298 acertos, somando 51.04% e o maior resultado foi de 1374 acertos, somando 54.96%, como mostra a Figura 1 e a Tabela 2.

Figura 1.



Fonte: (OS AUTORES, 2017).

Os algoritmos SVM e *Naive Bayes* apresentaram resultados inesperados, sendo que ambos acertaram 100% dos testes, como mostra a Tabela 1.

Tabela 1.

	KNN	***Perceptron***	***Naive Bayes***	**SVM**
Intervalo 1 (acertos)	1326	1306	2500	2500
Intervalo 2 (acertos)	1259	1298	2500	2500
Intervalo 3 (acertos)	1324	1299	2500	2500
Intervalo 4 (acertos)	1303	1308	2500	2500

Intervalo 5 (acertos)	1246	1309	2500	2500
Intervalo 6 (acertos)	1349	1334	2500	2500
Intervalo 7 (acertos)	1191	1323	2500	2500
Intervalo 8 (acertos)	1229	1325	2500	2500
Intervalo 9 (acertos)	1301	1332	2500	2500
Intervalo 10 (acertos)	1322	1374	2500	2500

Fonte: (OS AUTORES, 2017).

3.2 Discussão dos Resultados

Os algoritmos KNN e *Perceptron* apresentaram resultados não muito satisfatórios. Devem-se a isso, fatores como a falta de pré-processamento da base de dados, que foi tratada *in natura* e submetida ao processo de *TFIDF*, sem exclusão de palavras insignificantes. Isso faz com que o algoritmo leve em conta palavras que não são necessárias ao modelo, dando outros resultados à classificação.

Os algoritmos *Naive Bayes* e SVM apresentaram *overfitting* completo quanto a classificação dos dados, sendo que ambos os algoritmos apresentaram 100% de acertos. Tal *overfitting* deve-se a uma possível presença de um "super indivíduo" entre as amostras, ou seja, um indivíduo muito bom ou muito ruim. Tal indivíduo enviesou a classificação dos comentários, deixando o modelo muito específico.

4 CONCLUSÃO

Os objetivos foram satisfeitos. Alguns resultados, todavia, foram inesperados. O SVM e o *Naive Bayes* apresentaram *overfitting* já o *Perceptron* e o KNN apresentaram comportamento dentro dos padrões. Levando em conta os motivos para determinados comportamentos, consideramos o objetivo do trabalho concluído dentro do escopo da disciplina e os motivos pelos quais tais devem-se tais comportamentos, entendidos.

ABSTRACT

The work aims the utilization of four machine learning algorithms to classificate a database of movies comments retrieved from IMDB. The used algorithms were: K-Nearest Neighbors (KNN), Neural Network Perceptron, Support Vector Machine (SVM) and Naive Bayes. For the development of the work were implemented two algorithms, being them KNN and Perceptron, using the programming language Python and the libraries scikit-learn and numpy to the treatment of the data and for the utilization of the other algorithms.

Keywords: *Machine Learning, text classification, KNN, Perceptron, Naive Bayes, SVM.*

REFERÊNCIAS BIBLIOGRÁFICAS

IMDB. Disponível em: <http://www.imdb.com/>, Acessado em 18 dez. 2017.

Scikit-learn. Disponível em: <http://scikit-learn.org>, Acessado em 18 dez. 2017.

Numpy and Scipy Documentation. Disponível em: <https://docs.scipy.org/doc/>, Acessado em 18 dez. 2017.

TFIDF. Disponível em: <https://pt.wikipedia.org/wiki/Tf%E2%80%93idf>, Acessado em 18 dez. 2017.

NAIVE BAYES classifier-MATLAB. Disponível em: <https://www.mathworks.com/help/stats/naivebayes-class.html?requestedDomain=www.mathworks.com>, Acessado em 18 dez.2017.

PERCEPTRON. Disponível em: <http://wiki.icmc.usp.br/images/7/7b/Perceptron.pdf>, Acessado em 18 dez. 2017.

Introduction to KNN, K-Nearest Neighborns: Simplified. Disponível em: <https://www.analyticsvidhya.com/blog/2014/10/introduction-k-neighbours-algorithm-clustering/>, Acessado em 18 dez.2017.