Análise de Métodos para Classificação de Web Spam

Alan C. Silva
Universidade Federal de São Carlos
UFSCar
Sorocaba, Brasil
alan.silva@gmail.com

Cássia C. Monteiro
Universidade Federal de São Carlos
UFSCar
Sorocaba, Brasil
kssia.cm@gmail.com

Lucas Renan A. Nunes
Universidade Federal de São Carlos
UFSCar
Sorocaba, Brasil
contato@lucasrenan.com

Thamires C. Luz
Universidade Federal de São Carlos
Ufscar
Sorocaba, Brasil
thamiluz@gmail.com

***Resumo*—Currently, with the internet rising and the constant use of search engines, bad people creating some techniques that consists in made some manipulations in search engines with the intention of getting a better rank in the search results that are presented to the user. This type of technique is known as Web spam.**

The aim of this paper is to attack the problem of web spam classification, applying three machine learning methods, more specifically, Naive Bayes, Logistic Regression and Neural Networks (Multilayer Perceptron) in a database that has been collected for analysis of this type classification.

In this article, we will present some experiments from tests and comparisons between the three methods in order to find one that showing a better result in our web spam classification problem.

I. Resumo

Atualmente, com o crescimento da internet e a constante utilização de sites de busca, surgiram algumas técnicas que realizam manipulações nos sites de busca com a intenção de se obter uma melhor posição na classificação dos resultados que são apresentados ao usuário. Esse tipo de técnica é conhecida como Web Spam.

O objetivo desse trabalho é analisar o problema de classificação de web spam, aplicando três métodos de aprendizado de máquina, mais especificamente, *Naive Bayes, Regressão Logística e Redes Neurais Artificiais (Multilayer Perceptron)* em uma base de dados que foi coletada para análise nesse tipo de classificação.

Nesse artigo, serão apresentados alguns experimentos a partir de testes e comparações entre os três métodos com a finalidade de se encontrar o que apresenta um melhor resultado na classificação de Web Spam.

***Keywords*-web spam; classificador; comparação de métodos; regressão logística; naive bayes; redes neurais artificiais;**

II. Introdução

Hoje em dia, é comum a utilização de sites de busca com a finalidade de se encontrar resultados como páginas, arquivos, imagens, músicas, etc . Os resultados exibidos são classificados através de *search engines* que geram um *ranking* contendo os melhores resultados de acordo com o nível de relevância do conteúdo da página. O conceito de *search engine* se originou aproximadamente em 1994, com o *World Wide Web Worm* que foi considerada a primeira *search engine* desenvolvida como um trabalho na Universidade do Colorado por *Olivier McBryan* e tinha por volta de 110.000 páginas web associadas. [1]

Com o crescimento do uso de sites de busca surgiu o *web spam* que segundo *Molina et. Al.* em [2] é definido pela ação de alterar páginas da internet com a intenção de enganar as search engines, de maneira que o conteúdo inserido para classificar a página no rank não tenha relação com a página em questão.

Araujo et. Al. em [3] relatam que atualmente o *web spam* é um dos maiores problemas das *search engines*, isso porque acaba sendo um problema que degrada a qualidade dos resultados oferecidos, deixando muitas pessoas frustadas ao acessarem páginas que não tem relação com o termo buscado. O artigo também relata que o *web spam* causa um impacto também na área econômica, devido ao fato de que uma boa colocação no *ranking* de busca acaba garantindo um aumento no retorno financeiro para as empresas que se utilizam dos serviços de propaganda nas *search engines*.

Com esse problema detectado e em constante crescimento, acaba se tornando viável uma série de estudos com a intenção de detectar a utilização de métodos fraudulentos por uma página que queira alterar o *ranking* da *search engine*, gerando benefícios para a mesma. Utilizando técnicas de aprendizado de máquina, é possível classificar essas páginas como sendo legítimas ou spam, nesse trabalho serão aplicados três métodos que tem como intuito a classificação de *web spam*.

Na sessão III, é realizada a descrição da base de dados que foi utilizada nesse trabalho e são citados os métodos

definidos. A sessão IV contém uma explicação mais detalhada dos métodos utilizados e modelo de comparação entre os algoritmos, já as escolhas de parâmetros são relatadas na sessão V. Os resultados são apresentados na sessão VI, seguido da conclusão na sessão VII.

III. Descrição do trabalho

Para esse trabalho foi disponibilizado uma base de dados com 3.596 amostras, ou seja, páginas com 137 atributos além do atributo classe. Os atributos são dados contínuos e descrevem características para páginas que são ou não são *spam*. A base está balanceada tendo metade das amostras classificadas como *spam* e a outra metade classificada como *não spam*.

Tendo essa base como objeto principal de aplicação e comparação dos três métodos de aprendizagem de máquina, que são, Naive Bayes, Regressão Logística e Redes Neurais Artificiais, que serão descritos com detalhe na próxima seção, podemos efetuar a classificação de *web spam*.

Como ferramenta para aplicação, implementação e demonstração dos métodos propostos e descritos, foi utilizado o software livre Octave.

IV. Métodos

A. Naive Bayes

De acordo com *Lowd et. Al* em [4], o classificador bayesiano é uma técnica probabilística baseada no teorema de *Thomas Bayes* e denominado *Naive Bayes*, o qual é considerado um algoritmo "ingênuo", pois assume que todos os atributos possuem relações independentes entre si, o algoritmo de *Naive Bayes* pode ser descrito como o produtório das probabilidades de cada atributo pelas classes existentes, sendo que a amostra é classificada para determinada classe, caso o valor calculado de produtório seja superior ao das outras classes.

Comparativos entre algoritmos demonstram que o *Naive Bayes* obteve resultados compatíveis com os métodos de árvore de decisão e redes neurais e devido a sua simplicidade e seu alto poder preditivo, acaba sendo um dos algoritmos mais utilizados [5].

B. Regressão Logística

A regressão logística é um classificador que através da função sigmoidal classifica um grupo de amostras. *Pohar et. Al* em [6], retrata que o método é robusto, flexível e de fácil utilização, com o objetivo de encontrar o melhor modelo que descreva os atributos em uma determina classe.

C. Redes Neurais Artificiais

Redes Neurais Artificiais é um método que pode ser usado como classificador e foi inicialmente desenvolvido para funcionar de forma semelhante ao sistema neural humano, desde então, uma grande variedade de modelos de redes foram desenvolvidos, sendo que, a escolha do modelo da rede depende do problema a ser resolvido, segundo *Sarle* em [7] o modelo mais utilizado é o de *Multilayer perceptrons* ou *MLP*. A *MLP* é constituída de nós fontes que formam a camada de entrada da rede *(input layer)*, uma ou mais camadas intermediárias *(hidden layers)* e uma camada de saída *(output layer)*. De acordo com *Keller et. Al* em [8], as redes neurais são consideradas como uma importante ferramenta que pode ser aplicada na mineração de dados.

V. Metodologia

Os algoritmos foram criados seguindo os critérios de otimização e eficiência. As escolhas dos parâmetros, dificuldades e soluções são descritas nas sub-sessões abaixo.

A. Naive Bayes

A principal dificuldade na implementação do *Naive Bayes* foi otimizar o algoritmo para obter resultados satisfatórios. Para uma base de dados com atributos contínuos e grande dispersão de dados, o algoritmo se mostrou lento no cálculo de probabilidade dos elementos.

Quando constatado que o *Naive Bayes* não tem um bom resultado com dados contínuos, para resolver esse problema, foi escolhida a solução de cestas, que é uma forma conhecida por discretizar os dados contínuos e agrupá-los em cestas de equivalência, uma descrição mais detalhada pode ser encontrada em [9].

Para determinar a escala dos atributos foi utilizado o quartil das amostras de treino, sendo utilizado essa mesma escala para as amostras de teste. Após definida a escala, as amostras foram agrupadas em cinco cestas de valores múltiplos com relação a escala, onde o valor cinco foi escolhido a partir de testes realizados através da análise do comparativo entre desempenho e acurácia obtida.

B. Regressão Logística

Ao testar o algoritmo implementado de regressão logística foi constatado que com os 137 atributos da base de dados o resultado apresentado foi bom, porém o desempenho computacional foi alto. Para melhorar o desempenho, foi decidido utilizar a técnica de análise de componentes principais *(PCA)* com o intuíto de se reduzir a dimensionalidade dos atributos [10].

Com o uso do PCA foi calculado o parâmetro k com a matriz de autovalores S retornada da função SVD, de modo que a soma dos valores até k calculado sobre o total de atributos, resultasse em uma variância de 99%. Com a matriz de atributos reduzidos a $k = 87$, o método apresentou melhor desempenho computacional e não teve interferência na acurácia apresentada.

C. Redes Neurais Artificiais

Na implementação de Redes Neurais Artificiais foram encontradas algumas dificuldades para obter um valor de acurácia aceitável para o método, sendo que, a principal

dificuldade foi em relação a atualização dos pesos, que foram denominados nesse trabalho como *theta*, onde após algumas tentativas o *theta* inicial foi parametrizado para ser um valor próximo de zero.

Além da escolha do *theta* inicial, para obtermos um melhor resultado foi necessário aplicar a normalização dos dados, para que ficassem com média 0 e desvio padrão 1.

Para obter um bom desempenho computacional, definimos o número de neurônios na camada intermediária utilizando as *rules of thumb*, consulte [11], onde após alguns testes foi definido como sendo 20 neurônios na camada intermediária, obtendo ,dessa forma, a menor taxa de erro alinhada ao desempenho computacional.

Como o resultado de acurácia varia de acordo com a escolha do *theta* inicial sendo de forma randômica, para otimizarmos o resultado, foi gerado um arquivo contendo uma otimização dos melhores valores para os *thetas* que foram apresentados através de testes, sendo possível classificar novas amostras com o uso desses *thetas* pré-definidos ou utilizar o modo randômico, sendo que, para maiores detalhes de implementação é sugerida a consulta ao apêndice no final do artigo.

D. Medida de desempenho

Para validar os resultados obtidos de todos os algoritmos implementados utilizamos a validação cruzada ou *cross-validation*, que consiste em separar amostras classificadas da base de dados em k partes, onde após essa separação fazemos n repetições de maneira que cada parte seja utilizada como teste de classificação, e as outras partes sejam utilizadas para o treinamento do algoritmo. De acordo com *Williams et. Al* em [12], a escolha de k é geralmente igual a dez.

VI. Resultados

Para cada método descrito acima foi calculado a acurácia, F-medida, revocação e precisão. Os resultados dos testes são demonstrados na Tabela I de comparação, em negrito estão os melhores resultados encontrados.

	Naive Bayes	Regressão Logística	Redes Neurais
Acurácia	80.358948	**90.166835**	79.221436
Revocação	84.731058	**93.528817**	78.928929
Precisão	74.064712	**87.636191**	79.726997
F-Medida	79.039655	**90.486672**	79.325956

Tabela I
Comparativo entre métodos

De acordo com a tabela I, o método que teve um resultado melhor foi a *Regressão Logística*, obtendo 90% de acurácia. O desempenho computacional dos métodos *Naive Bayes* e *Regressão Logística* foram equivalentes, levando menos de um minuto para computar os resultados. Já o método de *Redes Neurais (MLP)* teve um desempenho computacional maior, devido a sua otimização.

Em termos gerais, os algoritmos obtiveram um bom resultado, ficando a acurácia de todos acima ou próximos de 80% e a menor F-Medida igual a 79%.

VII. Conclusão

Nosso trabalho demonstra de forma clara e objetiva a implementação de três métodos de aprendizado de máquina. Com a comparação dos resultados de testes verificamos que a simplicidade e eficiência da *Regressão Logística* apresentou um melhor desempenho e maior acurácia para uma base de dados balanceada e com dados contínuos. As validações cruzadas utilizadas confirmam que todos os métodos são eficientes com dados apresentados de maneira distintas.

Dessa forma, podemos concluir que a classificação de páginas webspam ou legítimas pode ser feita por meio de qualquer um dos três métodos utilizados nesse trabalho.

Referências

[1] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," 1998.

[2] Z. Gyongyi, H. G. Molina, and J. Pedersen, "Combating web spam with trustrank," vol. 30, pp. 576–587, 2004.

[3] L. Araujo and J. Martinez-Romo, "Web spam detection: New classification features based on qualified link analysis and language models," *IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY*, vol. 5, 2010.

[4] D. Lowd and P. Domingos, "Naive bayes models for probability estimation," *ICML '05 Proceedings of the 22nd international conference on Machine learning*, pp. 143–161, 2005.

[5] H. Zhang, "The optimality of naive bayes."

[6] M. Pohar, M. Blas, and S. Turk, "Comparison of logistic regression and linear discriminant analysis: A simulation study," pp. 529 – 536, 2004.

[7] W. S. Sarle, "Neural networks and statistical models," 1994.

[8] O. Intrator and N. Intrator, "Interpreting neural-network results: A simulation study," 2001.

[9] R. R. Bouckaert, "Naive bayes classifiers that perform well with continuous variables," pp. 1089–1094, 2005.

[10] H. Abdi and L. J. Williams, "Principal component analysis," 2010.

[11] K. L. Priddy and P. E. Keller, "Artificial neural networks: An introduction," pp. 42 – 47, 2005.

[12] C. Elkan, "Evaluating classifiers," 2012.

VIII. Apêndice

Nessa sessão apresentamos um descritivo da implementação, para que os testes possam ser reproduzidos.

A. Naive Bayes

1) A base de dados deve estar na raiz da pasta IMPLEMENTACAO.
2) Para rodar o método, entre na pasta "Naive_Bayes", chame o arquivo "nb.m".
3) Para dados contínuos, é necessário escalar a base de treinamento chamando "nb_escalarAtributos", a função irá retornar a base escalada, e o range utilizado. Como parâmetro de "qtdeBags", utilizamos 5.
4) Para calcular as probabilidades dos elementos, chame a função "nb_calcProbabilidade", onde deve ser passado como uma matriz os elementos únicos da matriz escalada como parâmetro.
5) Para classificar uma nova amostra que não estava na base de treino, é necessário chamar novamente a "nb_escalarAtributos", informando o range retornado no passo anterior.
6) Chame então a função "nb_classificarAtributos", que retornará a classificação da nova amostra.

B. Regressão Logística

1) A base de dados deve estar na raiz da pasta IMPLEMENTACAO.
2) Para rodar o método, entre na pasta "Regressao_Logistica", e chame o arquivo "rl.m".
3) Para melhor desempenho computacional usamos a função "reduzirAtributos", que reduz para k amostras, de maneira que a variância seja de 99%.
4) Setar o *theta* inicial com uns.
5) Chamar a função "fminunc", passando função "rl_funcaoCusto".
6) Com o *theta* ótimo calculado, chamar a "rl_predicao", que irá classificar a nova amostra de dados..

C. Redes Neurais Articiais - MultiLayer Perceptron

1) A base de dados deve estar na raiz da pasta IMPLEMENTACAO.
2) Para rodar o método, entre na pasta "RN_MLP", e chame o arquivo "rn_mlp.m".
3) Defina o item "tam_cam_inter", essa variável determina a quantidade de neurônios na camada intermediária. Em nossos testes utilizamos 20.
4) Para melhor acurácia, chamamos o "normalizarAtributos".
5) Para o resultado acima informado, carregamos o *theta* pré-definido no arquivo "theta_ini". Caso deseje um outro valor de *theta* inicial, deve-se descomentar as linhas que geram os *thetas* aleatórios.
6) Chamar a função "fminunc", passando a função "rn_JDeltha".
7) Com o *theta* ótimo calculado, utilizar a função "reshape", que deve separar o *theta1* e *theta2*.
8) Chamar a "rn_predicao", que irá classificar a nova amostra de dados. Deve-se passar como parâmetro os *thetas* do item anterior.