International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2015): 6.391

Data Mining for Web Spam Detection Analysis of Techniques

Mugdha Kolhe[1], Disha Bhukte[2]

UG, CE Student, Department of Computer Engineering, Pune Institute of Computer Technology, Pune, India

UG, CE Student, Department of Computer Engineering, Pune Institute of Computer Technology, Pune, India

Abstract: ***World Wide Web is a source of vast information system, which is used by users by using search engine in order to look for required information from the high volume of data available. The sites appearing at the top of result page more frequently visited sites. These sites generally have more visitors since they have more relevant content. Thus, in order to increase the significance of a site, search engine optimization is used by various sites to improve their page rank legally. To improve page ranks illegally, web spam techniques are used. Page ranking algorithms of search engines are tricked into giving more weightage to a site that has no real merit. Which consequently result in users losing trust in search engines results. In this paper, we firstly explain web spamming techniques and then, present a comparison of data mining techniques for detecting web spam.***

Keywords: Data Mining, Page Rank, Search Engine, Web spam, Search Engine Optimization

1. Introduction

The Internet is the global system of interconnected computer networks that link devices worldwide. It is a *network of networks* that consists of private, public, academic, business, and government networks of local to global scope, linked by a broad array of electronic, wireless, and optical networking technologies. The Internet carries an extensive range of information resources and services, such as the inter-linked hypertext documents and applications of the World Wide Web (WWW), electronic mail, newsgroups, voice over IP telephony, and peer-to-peer networks for file sharing.

A web search engine is a software system that is designed to search for information on the World Wide Web. The search results are generally presented in a line of results often referred to as search engine results pages (SERPs). The information may be a mix of web pages, images, and other types of files. Some search engines also mine data available in databases or open directories. Unlike web directories, which are maintained only by human editors, search engines also maintain real-time information by running an algorithm on a web crawler.

Search engine optimization (SEO) is the process of affecting the visibility of a website or a web page in a web search engine's unpaid results—often referred to as "natural", "organic", or "earned" results. In general, the earlier (or higher ranked on the search results page), and more frequently a site appears in the search results list, the more visitors it will receive from the search engine's users, and these visitors can be converted into customers. SEO may target different kinds of search, including image search, local search, video search, academic search, news search and industry-specific vertical search engines.

Web spam (also referred to search spam) is a phrase used to describe webpages that are designed to "spam Google search results" using SEO tactics that are against Google publishers guidelines. Pages that use web spam to improve search engine results page (SERP) rankings typically use black hat SEO tactics such as keyword stuffing or cloaking, the latter of which involves employing misleading redirects and/or doorway pages of websites.

2. Types of Web Spam

Web spam is the result of using unethical methods to manipulate search results. Perkins has defined web spam as follows: “The attempt to deceive algorithms related to search engines” [9]. Researcher have detected and identified various type of web spam, and they have been divided into three categories:

- **Content based spam**

The content of a page is altered to obtain a higher rank. Most of content spamming techniques target ranking algorithms based on TF-IDF. Content in title, body, Meta tags and anchor text of page are modified to increase rank of page by including unrelated terms and repeating one or more special terms. Content based spamming is also known as keyword stuffing or term spamming.

- **Link based spam**

Link-based web spam is manipulation of link structure to obtain high rank. Spammers add link farms, which are collections of links that are connected to each other increasing page rank.

- **Page hiding spam**

Page hiding-based web spam presents a different content to search engines to obtain high rank. This includes cloaking and redirection techniques.

3. Classification Techniques

Data mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both.

The technique of web spam page detection comes under supervised classification problem of the data mining. In the supervised classification, formerly classified pages train a set of classifier to decide whether the page is spam or not. There are quite a few web spam classification techniques which has been presented in this section. Following are the classification techniques used for classifying a page into spam or non-spam.

a) *ADTree*

An alternating decision tree (ADTree) is combination of simple decision trees and boosting algorithm. An alternating decision tree consists of decision nodes and prediction nodes. Decision nodes specify a predicate condition. Prediction nodes contain a single number. ADTrees always have prediction nodes as both root and leaves. An instance is classified by an ADTree by following all paths for which all decision nodes are true and summing any prediction nodes that are traversed.

1. Initialize the weights of all instances with 1
2. Calculate the prediction value of root node

$$a = 0.5 * \ln \frac{W_{+}(C)}{W_{-}(C)} \quad (1)$$

3. Update the weights

$$W_{+}(C) = \sum W_{j,0} , j \in class1$$

$$W_{-}(C) = \sum W_{k,0} , k \in class2$$

4. Find the splitter node which has minimum Z_t , where

$Z_t = \sqrt{}(W+(c1,c2) * W-(c1,c2)) + \sqrt{}(W+(c1,\sim c2) * W-(c1,\sim c2)) + W(\sim c2)$ (2)

5. Find the prediction values of splitter nodes

$$a = 0.5 * \ln \frac{W_{+}(c1 \wedge c2)}{W_{-}(c1 \wedge c2)} \quad (3)$$

$$b = 0.5 * \ln \frac{W_{+}(c1 \wedge \sim c2)}{W_{-}(c1 \wedge \sim c2)} \quad (4)$$

6. Update the weights
7. Repeat the steps until tree is completely generated

b) *JRIP*

JRIP implements a propositional rule learner, Repeated Incremental Pruning to Produce Error Reduction (RIPPER), which was proposed by William W. Cohen as an optimized version of IREP. Ripper builds a ruleset by repeatedly adding rules to an empty ruleset until all positive examples are covered. Rules are formed by greedily adding conditions to the antecedent of a rule, starting with empty antecedent, until no negative examples are covered. After a ruleset is constructed, an optimization post pass massages the ruleset so as to reduce its size and improve its fit to the training data. A combination of cross-validation and minimum-description length techniques is used to prevent overfitting. The algorithm is briefly described as follows [2]:

Initialize RS = {}, and for each class from the less prevalent one to the more frequent one, DO:

1. Building stage:

Repeat 1.1 and 1.2 until the description length (DL) of the rule set and examples is 64 bits greater than the smallest DL met so far, or there are no positive examples, or the error rate >= 50%.

1.1. Grow phase

Grow one rule by greedily adding antecedents (or conditions) to the rule until the rule is perfect (i.e. 100% accurate). The procedure tries every possible value of each attribute and selects the condition with highest information gain: p(log(p/t)-log(P/T)).

1.2. Prune phase

Incrementally prune each rule and allow the pruning of any final sequences of the antecedents; The pruning metric is (p-n)/(p+n).

2. Optimization stage

After generating the initial rule set {Ri}, generate and prune two variants of each rule Ri from randomized data using procedure 1.1 and 1.2. But one variant is generated from an empty rule while the other is generated by greedily adding antecedents to the original rule. Then the smallest possible DL for each variant and the original rule is computed. The variant with the minimal DL is selected as the final representative of Ri in the rule set. After all the rules in {Ri} have been examined and if there are still residual positives, more rules are generated based on the residual positives using Building Stage again.

3. Delete the rules from the rule set that would increase the DL of the whole rule set if it were in it and add resultant rule set to RS.

c) *C4.5*

C4.5 is an extension of Quinlan's earlier ID3 algorithm developed by Ross Quinlan. C4.5 is often referred to as a statistical classifier [5]. J48 is the Java implementation of C4.5 in Weka Tool. In C4.5, a decision tree is formed by splitting the dataset according to splitting criteria.

1) The input is set of attributes and the data partition D, which is a set of training tuples.
2) The normalized information gain is calculated for each attribute using Information gain and Gain ratio.
3) The attribute having the highest gain ratio becomes split point.
4) The tuples in the data partition are divided according to the split point forming the branches for the split point.
5) The above steps are repeated till data partition is not empty.
6) In case of continuous values, we sort tuples in ascending order according to values of attribute A (which has continuous values). If there are v values for attribute A then we calculate normalized information gain for v-1 values and determine the split point. Otherwise, the midpoint is taken as the split point.
7) Pruning of the trees after their creation is carried out for better results. Pruning involves removal of branches that do not help and replacing them with the leaf nodes.

Information gain:-
The expected information needed to classify a tuple in D is given by

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) \quad (5)$$

Where, p_i is the non-zero probability that an arbitrary tuple in D belongs to class C_i and is estimated by $|C_i,D|/|D|$.

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j) \quad (6)$$

The term $|D_j|/|D|$ acts as the weight of the j^{th} partition. Info A (D) is the expected information required to classify a tuple from D based on the partitioning by A.

$$Gain(A) = Info(D) - Info_A(D) \quad (7)$$

Gain Ratio:-

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right) \quad (8)$$

The above formula represents the potential information generated by splitting the training dataset, D, into v partitions, corresponding to the v outcomes of a test on attribute A. The gain ratio is given by:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)} \quad (9)$$

d) *Random Forest*

Random forest (or random forests) is an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees. The term came from random decision forests that was first proposed by Tin Kam Ho of Bell Labs in 1995. The method combines Breiman's[3] "bagging" idea and the random selection of features.

Each tree is constructed using the following algorithm:

Let the number of training cases be *N*, and the number of variables in the classifier be *M*.

1) We are told the number *m* of input variables to be used to determine the decision at a node of the tree; *m* should be much less than *M*.
2) Choose a training set for this tree by choosing *n* times with replacement from all *N* available training cases (i.e. take a bootstrap sample). Use the rest of the cases to estimate the error of the tree, by predicting their classes.
3) For each node of the tree, randomly choose *m* variables on which to base the decision at that node. Calculate the best split based on these *m* variables in the training set.
4) Each tree is fully grown and not pruned (as may be done in constructing a normal tree classifier).

For prediction a new sample is pushed down the tree. It is assigned the label of the training sample in the terminal node it ends up in. This procedure is iterated over all trees in the ensemble, and the average vote of all trees is reported as random forest prediction.

4. Results and Conclusion

For experimentation of aforementioned algorithms were carried out on dataset named "WEBSPAM-UK2011" [7]. This dataset consists of around 3,700 web pages. This dataset consists only of content based features. This dataset has features including amount of anchor text, the size of the words, average length of words, length of the titles, total number of words in the web page, maximum and minimum length of the words etc. this is also known as keyword stuffing. There are overall eleven features included in it.

The below shown results were performed using 10 cross validation on Weka tool for both training and testing. The learning algorithms for classification purpose that has been considered are: AD Tree, JRIP, C4.5, and Random Forest.

Table 1: Performance Measures of algorithms

Performance Measures	RANDOM FOREST	C4.5	JRIP	AD TREE
Precision	0.911	0.881	0.853	0.716
Build Time	0.53	0.31	1.89	0.42
Recall	0.911	0.81	0.853	0.716
F-Measure	0.911	0.881	0.853	0.716
Efficiency	91.07	88.13	85.26	71.58

Table 2: Accuracy of Classification

Algorithms	**Instances**	**TP**	**TN**
RANDOM FOREST	3766	1862	1568
C4.5	3766	1785	1534
JRIP	3766	1716	1495
AD TREE	3766	1475	1221

The results of the table 1 clearly show that for content based features of Web Spam UK-2011, Random Forest classification technique gives the best results as Precision and efficiency are highest for it. The results of table 2 confirm that Random Forest and C4.5 gives the highest True Positive rate. On the other hand AD Tree gives minimum true positive rate among all the four techniques so we can come to the conclusion that for content based features, AD tree classification is the least efficient by showing the TP rate, FP Rate and the precision value, and random forest is the best.

References

[1] Amudha.J, Soman.K.P,c"Feature Selection in Top-Down Visual Attention Model using WEKA", International Journal of Computer Applications, Volume 24– No.4, June 2011.
[2] Hiren Gadhvi and Madhu Shukla, "Comparative Study of Classification Algorithms for Web Spam Detection", International Journal of Engineering Research & Technology (IJERT), Vol. 2 Issue 12, December - 2013
[3] Leo Breiman, "RANDOM FORESTS", Springer, 2001. K. Elissa, "Title of paper if known," unpublished.
[4] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly, "Detecting spam web pages through content analysis," WWW'06, 2006, pp. 83–92.
[5] J. Ross Quinlan, Book Review: C4.55: "Programs for Machine Learning", Morgan Kaufmann Publishers, 1993

[6] Mahdieh Danandeh Oskuie and Seyed Naser Razavi, "A Survey of Web Spam Detection Techniques", International Journal of Computer Applications Technology and Research Volume 3– Issue 3.
[7] Wahsheh H., Abu Doush I., Al-Kabi M., Alsmadi I. and Al-Shawakfa E. (2012), Using Machine Learning Algorithms to Detect Content-based Arabic Web Spam, International Journal of Information Assurance and Security (JIAS)