Science Arena Publications
Specialty Journal of Electronic and Computer Sciences
ISSN: 2412-7485
Available online at www.sciarena.com
2019, Vol, 5 (3): 86-93

Presenting a Method for Predicting Content Error in Cloud Computing Using the Nearest Neighbor Method

Reza Imani

Department of Information Technologhy, Faculty of Engineering, Tabriz Branch, Islamic Azad University, Bonab, Iran.

Abstract: *Cloud computing is a new technology for managing and providing services over the Internet. Cloud computing is a large group of connected computers. These computers can be personal computers or network servers. They can also be public or private. The prediction of content error is one of the fundamental challenges in cloud computing. In this research, it was tried to increase the accuracy of prediction error of content in cloud computing using the nearest neighbor algorithm. The simulation results in MATLAB software showed that the method for the data set had a very high performance. The results were compared with the performance of three perceptron neural networks based on radial performance and nearest neighbor. Finally, the best combination among these systems resulted in the nearest neighbor error detection with nearly 74% accuracy.*

Keywords: *Cloud Computing, Nearest Neighbor, Content Error Prediction, Radial Performance.*

INTRODUCTION

The growing popularity of the Internet and Web along having access to powerful handheld, mobile and sensor tools has changed the interaction path, life management, business management, and access to or receiving services. Reducing computation and communication costs changes the attention from being personal to Data Center computing. Additionally, distributed and parallel computations have been around for several years and made comprehensive changes in the industry with its new forms as cloud computing and multicore. These issues have changed the tendencies of the industry from software development for PCs to cloud data centers of millions of users to use software simultaneously (Buyya, Vecchiola and Selvi, 2013).

Cloud computing is a model for having access to an integrated and proper network to share large volumes of customizable computing resources based on demand (like networks, servers, caches, applications, and services) that can be managed with the least need for management and can be regulated and published just by a cloud service provider (Brian et al., 2008). Cloud computing is an innovative and exciting style for programming and using computers. Cloud computing creates excellent opportunities for software developers (Magoulès, Pan and Teng, 2016). This new architecture uses bandwidth optimally and reduces communication costs to have access to popular information. This network reduces network congestion and provides error prediction, delivery, and definitive endurance (Soule, Salamatian and Taft, 2005).

With progress in cloud computing, prediction of data error has turned into an important factor in cloud computing, so that predicting cloud errors is the most significant hurdle to the speed and development of cloud computing software. Thus, error prediction is one of the key factors for cloud success (Barford et al., 2002). The development in information and communication technology, changing business environment, and the ever-increasing growth in cloud storage services have encouraged the companies and users to delegate maintaining

and managing their data to cloud-storage service providers. However, the fear of predicting errors and preserving the privacy of the uses is always a major and controversial challenge in the cloud (Lakhina, Crovella and Diot, 2004).

Overall, cloud processing faces several risks of separate but related error prediction. Besides the possibility that the stored information may be stolen by attackers or be destroyed due to system defect, the cloud service provider may abuse this information or disclose it through governmental pressure. It is clear that these error prediction risks have serious consequences (Roschke, Cheng and Meinel, 2009). Several studies have been conducted in this regard. In (Lo, Huang and Ku, 2010), to protect the cloud computing environment, a template was developed to use the intrusion detection system (IDS) to manage DDoS attacks in a centralized way. They argued that if one IDS be used per virtual machine, it would be possible to manage the attacks well and respond to attacks by sharing information between them. Reference (Roschke, Cheng and Meinel, 2009) dealt with designing IDS management architecture where IDS is distributed throughout the system, all of which is under the control of a control center and management unit. In this study, some sensors are implemented as IDS in the service delivery system and users report to the control and management center as soon as any abnormality occurs and the headquarter decides on it. Another approach designed to protect cloud environment is in (Lee et al., 2011). In this paper, the attacks were categorized based on the magnitude of the damage as well as the level of their attacks in the cloud environment. Accordingly, a risk was provided for each attack and given to them, based on which a method to deal with, it was considered.

In (Mazzariello, Bifulco and Canonico, 2010), as in (Lee et al., 2011), risk assessment method for resources has been used, and an IDS has been used in the system. Paper (Bugiel et al., 2011) addresses designing a security system for secure information exchange through authentication and encryption. Two clouds (twins) were considered in the proposed system: a trust cloud and a commodity cloud. One of the methods for assessing the success of a cloud system is K nearest neighbor. K nearest neighbor selects a group, including K records from the training set, which has the closest records to the test record, and decides on the category's superiority or label in relation to the test record category. Simply stated, this method selects the ranks in the neighborhood with the highest number of records featured to them. Thus, the rank seen among K nearest neighbor is considered as the new record rank (Manvi and Shyam, 2014). Considering the above-mentioned issues, the prediction of data content error and maintaining privacy among the most important error prediction challenges in the cloud computing environment, the study has tried to use the nearest-neighbor algorithm to predict error in cloud computing.

Methods

The nearest neighbor algorithm

One of the well-known algorithms for classification is the nearest neighbor algorithm. Although it has been several decades since its introduction, this method does not assume a model on samples for training. Moreover, this method does not need setting the input parameter by an expert. Indeed, this method does not spend time for training and waits until the request for the unlabeled sample to be labeled is sent. Thus, this method does not have the time cost for the training phase and, in contrast, requires more time in the test phase. Moreover, to determine the similarity between the samples, it is necessary to determine a distance criterion. Different distance criteria show different efficiencies in different characteristics` space (Muja and Lowe, 2014).

The first remarkable point about the nearest neighbor algorithm is how to obtain the distance between the samples. Various distance criteria can be used in this method. Moreover, in some of the past studies, specific criteria have been introduced for this method, which can be used to achieve better performance. Several useful distance criteria have been introduced here (Duda, Hart and Stork, 2012).

The Euclidean distance has been the most widely used distance criterion in various areas from the past. This criterion has also been used in the category of the traditional nearest neighbor approach. The Euclidean distance between two samples of Y and X is defined as follows:

$$d(X,Y)=\sqrt{\sum_{i=1}^{m}\left(x_i-y_i\right)^2} \quad (1)$$

Here, m is the number of features of the samples, and x_i and y_i are the i-th feature of the two examples.
It is clear that this criterion is usable and efficient when in the features space, all of the features are numerical and real values and have the same range of variations. In contrast, in the case that features have integer values, Manhattan distance criterion shows a better performance. This distance criterion is defined as follows:

$$d(X,X)=\sum_{i=1}^{m}\left|x_i-y_i\right| \quad (2)$$

Like the Euclidean distance, m is the number of features of the samples, and x_i and y_i are the values of the i-th features of the two examples.
Mahalanobis distance criterion is another widely used and famous distance criterion. In this criterion, in addition to the value of the features, propagating the samples in the feature space is considered. This propagation play affects the criterion of distance by the covariance matrix. This criterion is defined as follows:

$$d(X,X)=\sqrt{(\vec{x}-\vec{y})^T S^{-1}(\vec{x}-\vec{y})} \quad (3)$$

Here, $\vec{x}$ and $\vec{y}$ are the two feature vectors and S is the covariance matrix of the features.
In real-world problems, the values of features may not always be true. In many problems, some of the features have a nominal value. In these features, there are no larger and smaller relations and difference between two values and is not defined for two values. In these values, only the difference or similarity between the two values shows the difference between the two samples. For such a feature space that has both nominal features and true values, Wilson and Martinez have presented a criterion called Heterogeneous Value Difference Metric (HVDM). This criterion is the developed form of a measure called Value Difference Metric (VDM) that has been previously provided.
In VDM, the difference between *a* and *b* for feature f is defined as follows:

$$vdm_f(X,Y)=\sum_{c=1}^{C}\left(\frac{N_{f,a,c}}{N_{f,a}}-\frac{N_{f,b,c}}{N_{f,b}}\right)^2 \quad (4)$$

Here, $N_{f,a}$ equals to the number of iterations of a value in *f* feature, and $N_{f,a,c}$ is the same iteration in the samples with label c.
Based on the definition of VDM, HVDM criterion is defined as follows:

$$d(X,Y)=\sqrt{\sum_{f=1}^{m}d_f^2\left(x_f,y_f\right)} \quad (5)$$

Where:

$$d_f(x_f, y_f) = \begin{cases} 1 & \text{if x or y is unknown} \\ vdm_f(x, y) & \text{if f is a nominal} \\ \dfrac{|x-y|}{4\sigma_f} & \text{if f is numeric} \end{cases} \quad (6)$$

Results

1. **Perceptron Neural Network Function**

For this system, MATLAB ready-made function has been utilized.

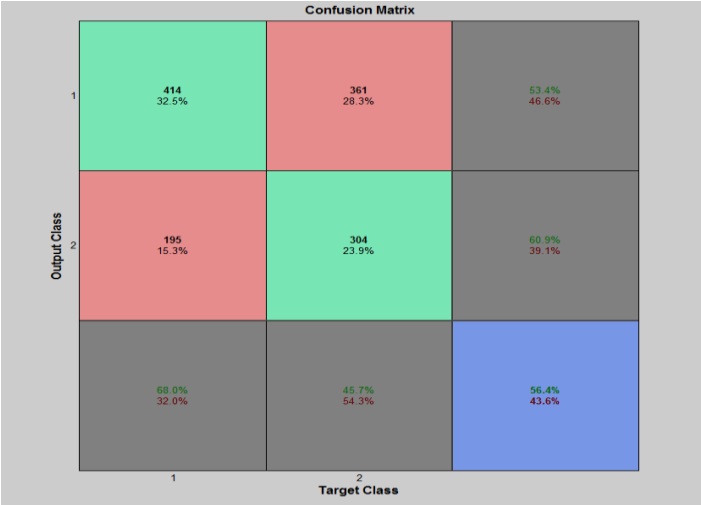

Figure 1: Confusion for the perceptron neural network

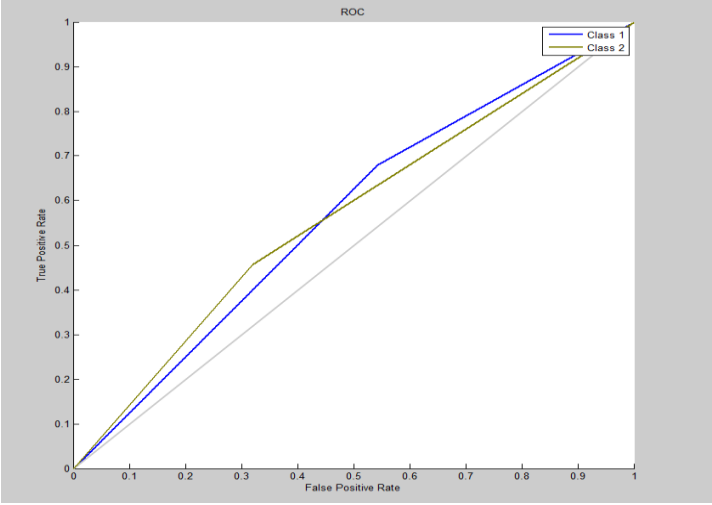

Figure 2: Receptor agent characteristic for the perceptron neural network

The results of the general comparisons are given in the table below.

Table 1: Results obtained from combining perceptron neural network

MEAN		k1	k2	k3	k4
0.204693	False classification rate	0.25	0.1315789	0.210526	0.226667
0.28712	The cost of false classification	0.35913978	0.2168459	0.342509	0.229984

0.14356	Normalized cost of false classification	0.17956989	0.1084229	0.171254	0.114992
0.786342	Sensitivity	0.73333333	0.8888889	0.829268	0.693878
0.819709	Specific rate	0.77419355	0.8387097	0.742857	0.923077
0.709125	Precision	0.71	0.69	0.72	0.7165
0.862258	Accuracy	0.825	0.8888889	0.790698	0.944444
0.786342	Recalling	0.73333333	0.8888889	0.829268	0.693878
0.818721	F combined factor	0.77647059	0.8888889	0.809524	0.8
0.455014	Compatibility	0.34623656	0.7275986	0.629268	0.116954
0.803026	AUC	0.75376344	0.8637993	0.786063	0.808477
0.793244	Balance	0.75291736	0.8615077	0.781744	0.776809
4.080331	Run time	4.20269786	4.3184428	3.591441	4.208742

As the results show, in all four runs, the stability of the perceptron neural network, and the third run provided the best result. Thus, the best precision up to this stage was 0.7091.

2. Radial basis function (RBF)

To this end, MATLAB ready-made function was used for RBF system. In this function, the radius of coverage was set to 2. The results are as follows.

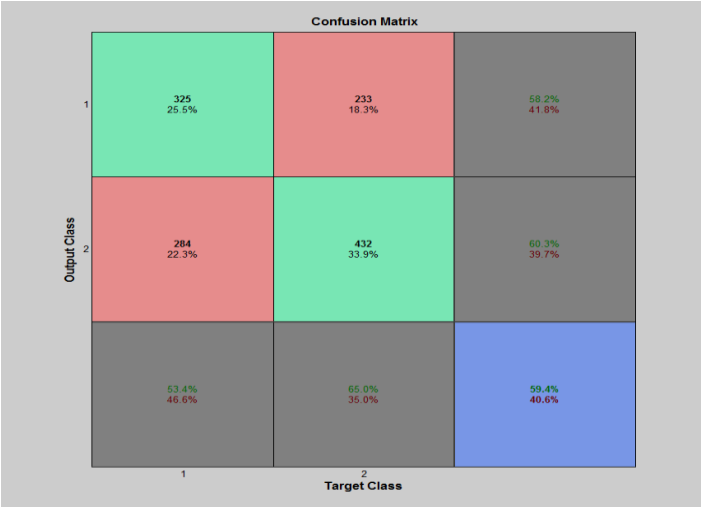

Figure 3: Confusion for RBF system

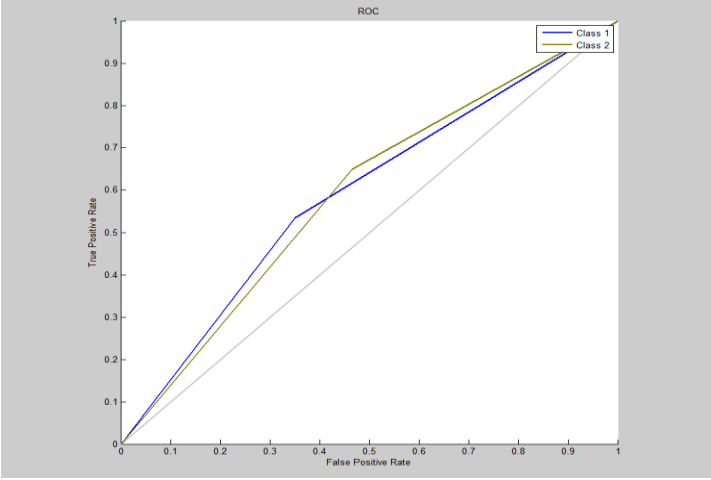

Figure 3: The receiver agent characteristic for RBF system

Table 2: Results from combining RBF system

MEAN		k1	k2	k3	k4
0.432544	False classification rate	0.43421053	0.3947368	0.407895	0.493333
0.595086	The cost of false classification	0.62131148	0.6468531	0.458333	0.653846
0.297543	Normalized cost of false classification	0.31065574	0.3234266	0.229167	0.326923
0.562101	Sensitivity	0.55737705	0.6153846	0.583333	0.492308
0.623864	Specific rate	0.6	0.5454545	0.75	0.6
0.567456	Precision	0.56578947	0.6052632	0.592105	0.506667
0.90113	Accuracy	0.85	0.8888889	0.976744	0.888889
0.562101	Recalling	0.55737705	0.6153846	0.583333	0.492308
0.69116	F combined factor	0.67326733	0.7272727	0.730435	0.633663
-3.15608	Compatibility	-1.242623	-1.6573427	-6.91667	-2.80769
0.592982	AUC	0.57868852	0.5804196	0.666667	0.546154
0.589123	Balance	0.57814986	0.5789652	0.656408	0.542971
22.21198	Run time	27.8528068	23.630439	24.05727	13.30739

Thus, up to this stage, the best attained precision was 0.605, which was a better result than the previous one.

3. The performance of K nearest neighbor system

K nearest neighbor system in this study was based on clustering K nearest neighbor. The results are as follows.

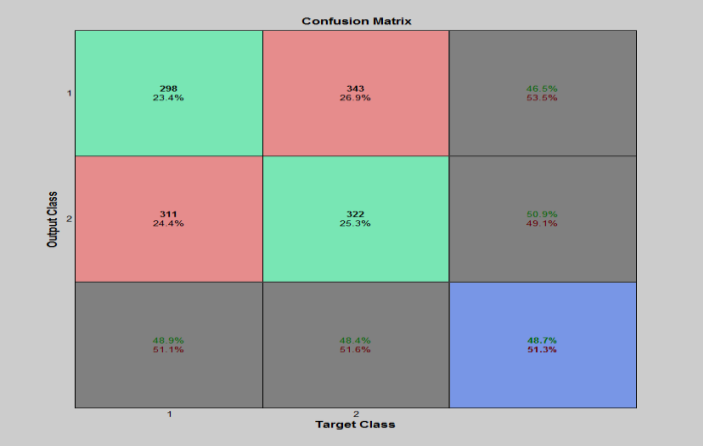

Figure 4: Confusion for K nearest neighbor

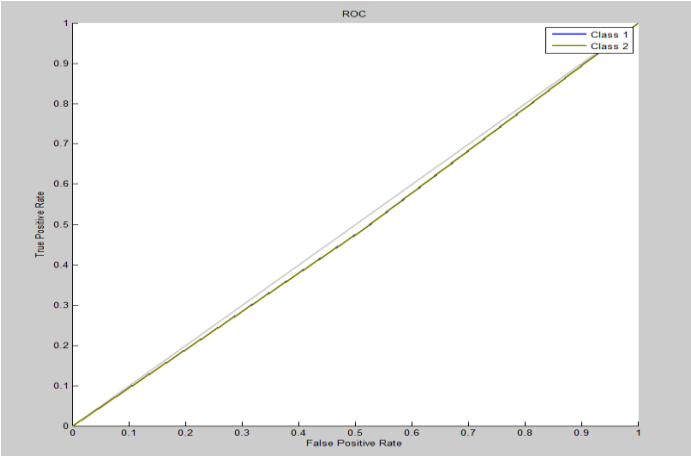

Figure 5: The receiver agent characteristic for the K nearest neighbor system

Table 3: Results obtained from combining K nearest neighbor system

MEAN		k1	k2	k3	k4
0.254167	False classification rate	0.22368421	0.1184211	0.407895	0.266667
0.314738	The cost of false classification	0.32840028	0.2121849	0.458333	0.260033
0.157369	Normalized cost of false classification	0.16420014	0.1060924	0.229167	0.130017
0.733298	Sensitivity	0.76744186	0.9285714	0.583333	0.653846
0.818613	Specific rate	0.78787879	0.8235294	0.75	0.913043
0.745833	Precision	0.77631579	0.8815789	0.592105	0.733333
0.903214	Accuracy	0.825	0.8666667	0.976744	0.944444
0.733298	Recalling	0.76744186	0.9285714	0.583333	0.653846
0.798724	F combined factor	0.79518072	0.8965517	0.730435	0.772727
-1.43517	Compatibility	0.46441156	0.8403361	-6.91667	-0.12876
0.775956	AUC	0.77766032	0.8760504	0.666667	0.783445
0.761711	Balance	0.77742563	0.8653822	0.656408	0.747627
9.773696	Run time	4.44637853	4.4135796	26.03737	4.197456

As the results show, all 4 runs improved K nearest neighbor compared to the previous one, and among the three methods, K nearest neighbor has had the best results.

Conclusion

Given the subject under study, first error detection in cloud computing and various evaluation indices were explained, and then the performance of the three perceptron neural networks based on the performance of RBF and K nearest neighbor was dealt with. Finally, the best combination between K nearest neighbor was with an accuracy of close to 74%. According to the results, it is clear that among the used prediction systems, K nearest neighbors has had the best performance.

References

1. Barford, P., Kline, J., Plonka, D., & Ron, A. (2002, November). A signal analysis of network traffic anomalies. In Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment (pp. 71-82). ACM.
2. Brian, H., Brunschwiler, T., Dill, H., Christ, H., Falsafi, B., Fischer, M., ... & Kaiserswerth, M. (2008). Cloud computing. Communications of the ACM, 51(7), 9-11.
3. Bugiel, S., Nurnberger, S., Sadeghi, A., & Schneider, T. (2011, March). Twin clouds: An architecture for secure cloud computing. In Workshop on Cryptography and Security in Clouds (WCSC 2011) (Vol. 1217889).
4. Buyya, R., Vecchiola, C., & Selvi, S. T. (2013). Mastering cloud computing: foundations and applications programming. Newnes.
5. Duda, R. O., Hart, P. E., & Stork, D. G. (2012). Pattern classification. John Wiley & Sons.
6. Lakhina, A., Crovella, M., & Diot, C. (2004, August). Diagnosing network-wide traffic anomalies. In ACM SIGCOMM computer communication review (Vol. 34, No. 4, pp. 219-230). ACM.
7. Lee, J. H., Park, M. W., Eom, J. H., & Chung, T. M. (2011, February). Multi-level intrusion detection system and log management in cloud computing. In 13th International Conference on Advanced Communication Technology (ICACT2011) (pp. 552-555). IEEE.

8. Lo, C. C., Huang, C. C., & Ku, J. (2010, September). A cooperative intrusion detection system framework for cloud computing networks. In 2010 39th International Conference on Parallel Processing Workshops (pp. 280-284). IEEE.
9. Magoulès, F., Pan, J., & Teng, F. (2016). Cloud computing: Data-intensive computing and scheduling. Chapman and Hall/CRC.
10. Manvi, S. S., & Shyam, G. K. (2014). Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey. Journal of network and computer applications, 41, 424-440.
11. Mazzariello, C., Bifulco, R., & Canonico, R. (2010, August). Integrating a network ids into an open source cloud computing environment. In 2010 Sixth International Conference on Information Assurance and Security (pp. 265-270). IEEE.
12. Muja, M., & Lowe, D. G. (2014). Scalable nearest neighbor algorithms for high dimensional data. IEEE transactions on pattern analysis and machine intelligence, 36(11), 2227-2240.
13. Roschke, S., Cheng, F., & Meinel, C. (2009, December). Intrusion detection in the cloud. In 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing (pp. 729-734). IEEE.
14. Soule, A., Salamatian, K., & Taft, N. (2005, October). Combining filtering and statistical methods for anomaly detection. In Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement (pp. 31-31). USENIX Association.