Crawled Data Analysis on Baidu API Website for Improving SaaS Platform (Short Paper)

Lei Yu[1(✉)], Shanshan Liang[1], Shiping Chen[2], and Yaoyao Wen[1]

[1] Department of Computer Science, Inner Mongolia University, Hohhot, China
yuleiimu@sohu.com
[2] Commonwealth Scientific and Industrial Research Organization, Canberra, Australia

Abstract. SaaS (Software-as-a-Service) is a cloud computing model, which is sometimes referred to as "on-demand software". Existing SaaS platforms are investigated before building new distributed SaaS platform. The service data mining and evaluation on existing SaaS platforms improve our new SaaS platform. For SaaS that provide various APIs, we analysis their website data in this paper by our data mining method and related software. We wrote a crawler program to obtain data from these websites. The websites include Baidu API and ProgrammableWeb API. After ETL (Extract-Transform-Load), the obtained and processed data is ready to be analyzed. Statistical methods including non-linear regression and outlier detection are used to evaluate the websites performance, and give suggestions to improve the design and development of our API website. All figures and tables in this paper are generated from IBM SPSS statistical software. The work helps us improve our own API website by comprehensively analyzing other successful API websites.

Keywords: SaaS (Software-as-a-Service) · Baidu API · Data analysis · Regression · Micro service

1 Introduction

There are three cloud computing models: IaaS (Infrastructure-as-a-service), PaaS (Platform-as-a-service) and SaaS (Software-as-a-service). Researchers have been studying service data mining in various platforms for boosting research on service recommendations and service compositions.

However, researchers in service computing domain lack of a cloud SaaS platform hosting real-world services. Based on three models of cloud computing, we aim at building a public cloud for service data mining. The advantage of using the vertical architecture is that we can monitor SaaS applications in detail, such as their network bandwidth, CPU and memory usage, user locations and amount of sessions, etc.

H. Gao et al. (Eds.): CollaborateCom 2018, LNICST 268, pp. 708–716, 2019.
https://doi.org/10.1007/978-3-030-12981-1_49

2 Related Works

Researches on commercial and non-profit service orchestration platforms are discussed in this section, and then related service data mining are investigated. OpenStack Heat, Windows Azure AppFabric/MarketPlace (including AppMarket), Amazon AWS Lambda and Google App Engine are discussed below.

The combination of Windows Server and AppFabric [2] provides an easy-to-manage platform for developing, deploying, and reliably hosting middle-tier WCF/WF services. Malawski [3] have developed prototype workflow executor functions using AWS Lambda. Villamizar [4] presents a cost comparison of a web application developed and deployed using the same scalable scenarios by AWS Lambda. Abrahao [5] proposed a platform-independent monitoring middleware for cloud services. This middleware was implemented in both Microsoft Azure and Google App Engine to monitor the quality of cloud services. Using Google App Engine as a platform, Nishida [6] proposes a modeling and simulation based framework to predict the cloud performance. Basu [7] presents a performance case-study on implementing the building blocks of a privacy preserving collaborative filtering scheme in Java on the Google App Engine (GAE/J) cloud platform. Prodan [8] employs the Google App Engine (GAE) for high-performance parallel computing. Prodan [9] designed a generic master-slave framework that enables implementation and integration of new algorithms by instantiating one interface and two abstract classes.

Using frequent association mining techniques to a large-scale data set that contains descriptively narrated texts, Park [10] analyzes the association between words. Huang [11] proposes a novel dimensionality reduction method called locality-regularized linear regression discriminant analysis for feature extraction. Bravi [12] focuses on nonlinear regression problems by assuming that the process underlying the generated data can be approximated by a continuous function. They present a new feature ranking method based on the solution of instances of the global optimization problem.

3 Data Analysis on Baidu API Website

In the section, statistical methods are used to study whether the statistical variables are related. We discover what relations among these variables are. All figures and tables in this section are generated by IBM SPSS statistical software.

3.1 Nonlinear Regression

The nonlinear regression process is used to establish a nonlinear relationship between the dependent variable and a set of independent variables.

A nonlinear regression model can generally be expressed as:

$$Y = f(x, \theta) + e_i$$

$f(x,\theta)$ is an expectation function. It can be any kind of function.

The purpose of this experiment is to use the "non-linear" regression process to fit the model of the relationship between the appropriate dependent variable (Invocation times) and the independent variables (Visits, Favorites, and Comments) [13]. The following diagram is an analysis example.

The Preliminary Analysis of the Data and Parameter Settings

Firstly, we make a composite scatter plot between the independent variable (Invocationtimes) and the other three dependent variables as shown in the following figures (Fig. 1):

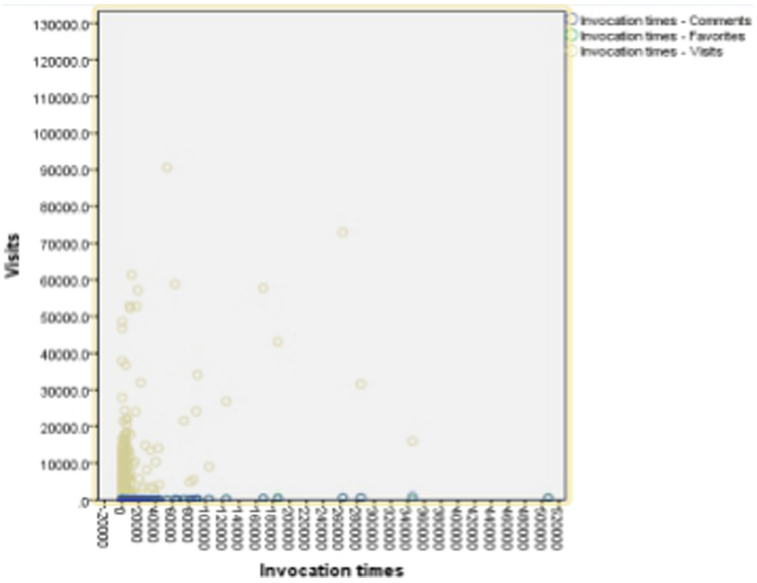

Fig. 1. The composite scatter diagram

Through observation, it is recommended to use a nonlinear model of the class with $Y = a * (b + c * EXP(d * Xe + f))$ exponential.

The Setting of the Loss Function, Looking for Whether There is a Strong Influence Point in the Data

According to this table, we can draw the following conclusions: 1, 15, 30, 34, 35, 36, 37, 90, 100, 101, 128, 153, 155, 186, 262 may be the impact points and its standardized residuals are greater than 3 (Table 1).

Table 1. Casewise diagnostics

Case number	Std. residual	Invocation times	Predicted value	Residual
1	−7.317	1744	104182.22	−102438.224
15	11.517	214748	53496.01	161251.992
30	6.670	139180	45794.64	93385.358
34	3.587	214748	164527.74	50220.257
35	−9.250	7279	136791.94	−129512.939
36	4.541	118237	54661.68	63575.322
37	−3.866	1715	55845.75	−54130.748
72	4.735	75290	8992.62	66297.379
90	5.546	79561	1907.97	77653.034
100	3.218	47408	2350.19	45057.812
101	3.225	45615	456.16	45158.841
128	4.147	63822	5757.18	58064.819
153	4.626	68278	3511.50	64766.497
155	5.899	85238	2641.63	82596.374
186	3.569	49612	−351.63	49963.628
262	3.833	105284	51616.45	53667.347

a. Dependent Variable: Invocation times

In order to reduce the influence of strong influence points, we use the least absolute deviations to calculate the residual value.

The Independent Variable is Visits

The functional form of setting and analyzing variables and models (Table 2):

Invocationtimes = 1.581 * EXP(−48.887 * Visits ** (−1/49.892) + 49.563)

Table 2. ANOVA

Source	Sum of squares	df	Mean squares
Regression	358720824316.627	4	89680206079.157
Residual	486073597544.478	691	703435018.154
Uncorrected total	844794421861.105	695	
Corrected total	805089065634.983	694	

Dependent variable: Invocation times

a. R squared = 1 − (Residual Sum of Squares)/(Corrected Sum of Squares) = .396.

The "ANOVA" table gives the structure of the analysis of variance. The R-square value is 0.396. The fitted model can account for variations in the dependent variable greater than 39.6%.

The fitting trend is shown in the following figure (Fig. 2):

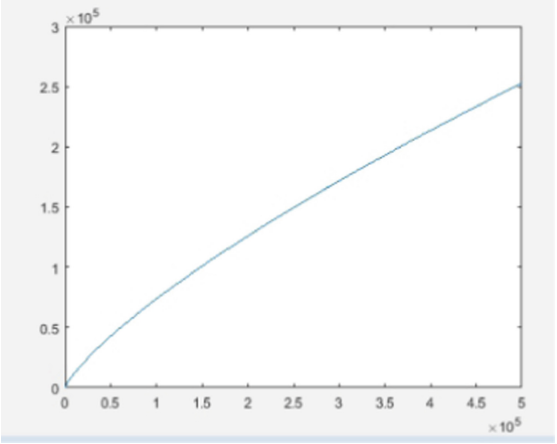

Fig. 2. Fitting trend map

The Independent Variable is Favorites

The functional form of setting and analyzing variables and models (Table 3):

Invocationtimes = 318985.678 * EXP(0.992 * Favorites2 + 4.141E − 6)

Table 3. ANOVA

Source	Sum of squares	df	Mean squares
Regression	490399653302.685	3	163466551100.895
Residual	354394768558.420	692	512131168.437
Uncorrected total	844794421861.105	695	
Corrected total	805089065634.983	694	

Dependent variable: Invocation times

a. R squared = 1 − (Residual Sum of Squares)/(Corrected Sum of Squares) = .560.

The "ANOVA" table gives the structure for analysis of the variance. The R-square value is 0.560. The fitted model can account for variations in the dependent variable greater than 56.0%.

The fitting trend is shown in the following figure (Fig. 3):

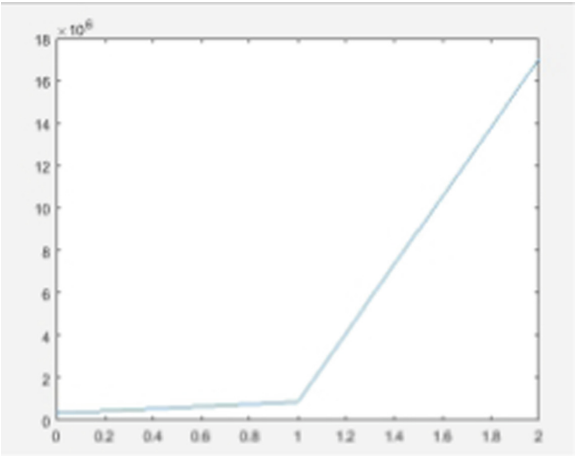

Fig. 3. Fitting trend map

The Independent Variable is Comments

The functional form of setting and analyzing variables and models is (Table 4):

Invocation times = 424303.081 * (1 – (−0.002) * EXP(116.870 * Comments − 4.761))

Table 4. ANOVA

Source	Sum of squares	df	Mean squares
Regression	618756082710.866	4	154689020677.717
Residual	226038339150.239	691	327117712.229
Uncorrected total	844794421861.105	695	
Corrected total	805089065634.983	694	

Dependent variable: Invocation times

a. R squared = 1 − (Residual Sum of Squares)/(Corrected Sum of Squares) = .719.

The "ANOVA" table gives the structure for analysis of variance. The R-square value is 0.719. The fitted model can account for variations in the dependent variable greater than 71.9%. This indicates that the fitting effect of the model is better.

The fitting trend is shown in the following figure (Fig. 4):

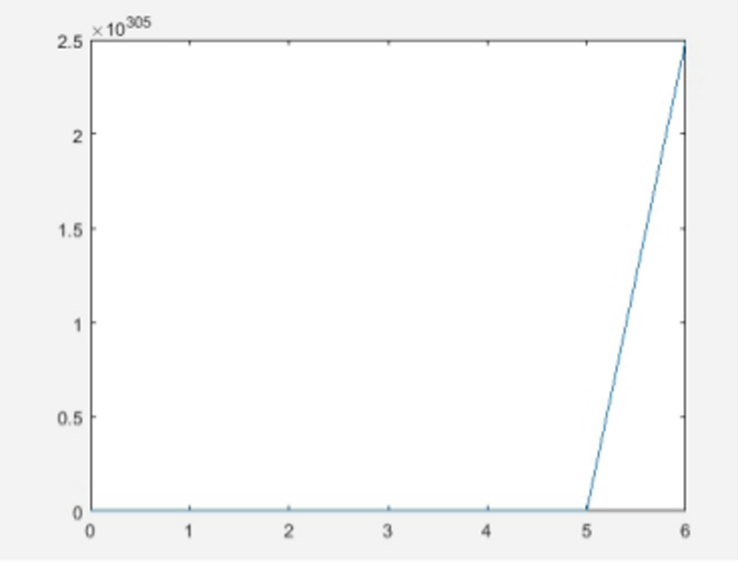

Fig. 4. Fitting trend map

3.2 Summary of Data Analysis

Statistical results show that the four variables are important factors to indicate how popular a service is. Moreover, the four variables are related to each other. Nonlinear regression methods are used to quantify their relationships. The result is that the fitted model should be improved in future to fit the service data.

4 Inspiration for Our SaaS Platform

This section shows the meaning of analyzing the crawled data. According to results of data mining and analysis above, our new platform [14] is improved in the aspect of design, development and operations. All data used in this paper can be downloaded from our website.

As mentioned earlier in this paper, we already know that invocation times of services are strongly related to visits, favorites and comments. Therefore, during designing, we extend service user privileges so that they can directly and freely invoke services in the platform and obtain results returned by service providers. Furthermore, service user in the platform can fully interact with each other to find potential co-developers, give suggestions, and even composite new services based on other users history of composing services.

5 Conclusions

Existing SaaS platforms are investigated before we build a new distributed SaaS platform. We analysis API website data by our data mining method and IBM software. Outlier detection, prediction, and statistical methods are used to evaluate the websites performance, and give suggestions to improve the design and development of our API website. The result of data analysis tells us which indicators are important to popularize a cloud service. Moreover, it helps us improve our own API website by comprehensively analyzing other successful API websites.

Acknowledgment. This work was supported by grants from Natural Science Foundation of Inner Mongolia Autonomous Region (2015BS0603) and Open Foundation of State key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications, SKLNST-2016-1-01).

References

1. Couto, R.S., Sadok, H., Cruz, P., Silva, F.F.D., Sciammarella, T., Campista, M.E.M., et al.: Building an IaaS cloud with droplets: a collaborative experience with openstack. J. Netw. Comput. Appl. (2018)
2. Kaufman, S., Garber, D.: Pro Windows Server AppFabric. Apress, New York (2010)
3. Malawski, M., Gajek, A., Zima, A., Balis, B., Figiela, K.: Serverless execution of scientific workflows: experiments with hyperflow, AWS lambda and Google cloud functions. Future Gener. Comput. Syst. (2017)
4. Villamizar, M., Ochoa, L., Castro, H., Salamanca, L., Verano, M., Lang, M., et al.: Cost comparison of running web applications in the cloud using monolithic, microservice, and aws lambda architectures. SOCA **11**(2), 1–15 (2017)
5. Abrahao, S., Insfran, E.: Models@runtime for monitoring cloud services in Google App Engine. In: IEEE 13th World Congress on Services, pp. 30–35 (2017)
6. Nishida, S., Shinkawa, Y.: A performance prediction model for Google App Engine. In: IEEE International Conference on P2p, Parallel, Grid, Cloud and Internet Computing, vol.3, pp. 134–140 (2014)
7. Basu, A., Vaidya, J., Dimitrakos, T., Kikuchi, H.: Feasibility of a privacy preserving collaborative filtering scheme on the Google App Engine: a performance case study. In: Proceedings of the 27th Annual ACM Symposium on Applied Computing, pp. 447–452 (2012)
8. Prodan, R., Sperk, M., Ostermann, S.: Evaluating high-performance computing on Google App Engine. IEEE Computer Society Press (2012)
9. Prodan, R., Sperk, M.: Scientific computing with Google App Engine. Future Gener. Comput. Syst. **29**(7), 1851–1859 (2013)
10. Park, S.H., Synn, J., Kwon, O.H., Sung, Y.: Apriori-based text mining method for the advancement of the transportation management plan in expressway work zones. J. Supercomput. **74**(3), 1283–1298 (2018)
11. Huang, P., et al.: Locality-regularized linear regression discriminant analysis for feature extraction. Inf. Sci. **429**, 164–176 (2018)

12. Bravi, L., Piccialli, V., Sciandrone, M.: An optimization-based method for feature ranking in nonlinear regression problems. IEEE Trans. Neural Netw. **28**(4), 1005–1010 (2017)
13. Yu, L., Junxing, Z., Yu, P.S.: Service recommendation based on topics and trend prediction. In: Wang, S., Zhou, A. (eds.) CollaborateCom 2016. LNICST, vol. 201, pp. 343–352. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59288-6_31
14. http://www.servicebigdata.cn