Research on Optimized Storage and Analysis System of Web Log Based on Django’s MVC Framework

Huan Tian, Jing Zhao*, Jiaqi Shen
Department of Electronics and Information Engineering, Lanzhou Vocational Technical College, Lanzhou, Gansu, China

Corresponding author’s email: 120839127@qq.com

Abstract. Association rule analysis algorithm is widely used in Web log analysis, but the existing association rule analysis algorithm will significantly reduce the analysis and mining performance when the amount of Web log is relatively large. This paper proposes an improved clustering algorithm, which first clusters users with the same interests and hobbies, and then mines association rules for users in the same category, thereby reducing data dispersion. Based on Django’s MVC framework, it optimizes the storage and storage of Web logs. In the analysis part, users can configure the support and confidence of association rule mining through the front-end, and at the same time query the results of mining through Hive, and use encryption algorithms in the data transmission process to ensure data security.

Keywords. HDFS; Web log mining; clustering; FP-Growth algorithm

1. Introduction
The overall requirements of web log mining system design are safety, efficiency and ease of use. Use an optimized distributed file storage architecture to save log data, and use encryption algorithms to ensure data security during log transmission. At the same time, it uses distributed computing tools to extract useful features in Web log data, combines with improved clustering algorithm to classify the log data, and then finds the rule sequence that meets the attribute requirements through the association rule mining algorithm. Users can manage the tasks created by the system, set the parameters of the association rule mining algorithm and obtain the results of the mining tasks in time.

2. Optimization of K-means clustering algorithm
For the mining of association rules of massive Web log data, the main algorithms used are Apriori and FP-Growth algorithms. They all have their own advantages and disadvantages in the execution efficiency of the algorithm [1]. The Apriori algorithm requires constant access to the database, and the overhead of the database is obviously unacceptable. The FP-Growth algorithm needs to store it in memory when constructing FP-Tree, but the FP-Tree constructed by massive Web log data will consume most of the memory, which will seriously affect the performance of the cluster [2]. Therefore, this paper proposes an FP-Growth association rule mining algorithm based on the improved K-means clustering algorithm. First, the proposed clustering optimization algorithm is used to reduce data dispersion, and users with the same hobbies are classified into one category, and then the same Mining association rules for class data [3].
2.1. Analysis of K-means clustering algorithm

Compared with other clustering algorithms, K-means algorithm has the advantages of simple execution process and fast convergence speed and is easy to implement. To measure the performance of the K-means clustering algorithm, it is often explained by the sum of the square error (SSE). The specific calculation method of the sum of the square error is shown in formula (1).

\[
SSE = \sum_{i=1}^{k} \sum_{x \in C_i} dist(x, c_i)
\]

(1)

Where \(C_i\) represents the particle in the i-th cluster, \(x\) represents any data point in the i-th cluster to the particle points of the cluster, and the K-means The distance calculation method uses the Euclidean distance calculation method. Therefore, SSE represents the sum of the distances between all data points and the mass point of the cluster to which the point belongs. If the SSE value is larger, it means that the clustering effect of each cluster is not very good, and the data points are not very dense; If the value is smaller, it means that the clustering effect between each cluster is better. The calculation here can only start when the initial mass points are determined, so it is only a local optimal solution, because the K-means algorithm does not have a clear method to confirm the initial mass points. If the initial mass points are not well selected, it will cause SSE is too large. Kim et al. determined the initial value of K according to the idea of maximum and minimum distance. First, calculate the set of minimum distance between each point and each cluster point, and then select the point with the largest distance as the new cluster point. This can avoid the clustering effect being too close due to the selection of cluster points, but this method can not solve the influence of abnormal points and the consumption of iterative calculation of new cluster points.

2.2. Optimization of K-means algorithm

Based on the analysis of the K-means algorithm, in view of the shortcomings of the K-means algorithm and combined with the data characteristics of the Web log itself, this chapter proposes some improved solutions, mainly including the following:

- Web log data preprocessing. Among the massive Web log data, not all data records belong to normal user access. Some of the abnormal data can be eliminated according to the request status code is not in the normal range, the request method is not GET, and the requested resource type is not a page request. Filtering out these logs that do not meet the request and the status code error can reduce the number of abnormal points in the Web log data, which can prevent some of the extreme attribute data from having a serious impact on the calculation of the sample distance.

- Optimization of the number of initial clusters. The selection of the initial clustering center will seriously affect the final clustering effect, so random selection of k clustering centers is not ideal. The Web log analysis in this article is based on a big data platform, so the number of clusters can be determined based on distributed computing. First of all, because the calculation method of SSE only considers the local optimality, it does not consider the difference of the particles in each cluster. Therefore, this chapter proposes the calculation method of the global optimal solution to determine the optimal initial cluster number. The specific definition of the function is as follows:

\[
OPT(k) = \frac{\sum_{i=1}^{k} \sum_{x \in C_i} sim(x, c_i)}{(n-k) / k}
\]

(2)

- In formula (2), k represents the number of clusters, n represents the number of all data, and sim represents the distance between two data points. The specific calculation method of the distance will be explained in the next section. The formula \(\sum_{i=1}^{k} \sum_{j=1}^{n} sim(c_i, c_j)\) represents the sum of square errors of the mass points between each cluster. The larger the value, the farther
the distance between each cluster, the more obvious the data aggregation, and the better the clustering effect. The formula represents the sum of squared errors within the group, which represents the convergence of each cluster. The smaller the value, the better the clustering effect in each cluster. Therefore, based on the algorithm of the global loss function, in accordance with the definition of the global loss function, this paper determines the number of cluster families by finding the k value with the largest fluctuation. The value of the largest fluctuation can be determined by finding the turning point at which the rate of change suddenly becomes larger, because if the rate of change tends to be flat, it means that it is meaningless to continue to increase the number of clusters. When each point when it is a cluster, the global loss function is zero.

- Iterative process optimization. In the K-means algorithm, the clustering point for the next iteration is determined by the mean point of all the data in the cluster. The cluster center formed in this way is probably not in the real high-density area of the data, which leads to the final clustering. The class results will have a certain deviation, and the iteration cost will become higher. This paper proposes an optimized binary clustering algorithm based on the binary K-means algorithm combined with the maximum distance idea. First find out the cluster with the largest square error sum in the group, calculate the K points with the largest distance from the cluster particles, and perform the binary clustering of the cluster according to these K points, and then obtain K binary cluster sets, choose the square error and the smallest division replace the original cluster. Keep repeating the above steps until the number of clusters obtained is equal to the initial set K.

3. Web log storage and analysis system interface design
As a complete Web log mining system, the system mainly implements the following interfaces in function:

- User interface, which is mainly used to verify the legitimacy of users. It is mainly divided into ordinary users and administrator users. Users of different levels have different permissions;
- Log storage interface, this interface is mainly used for users to upload Web log data that needs to be analyzed. Through this interface, the back-end optimized HDFS storage architecture can be triggered to save data;
- Data download interface, which is mainly used to download Web log data and the execution results of mining tasks, including classified data and corresponding association rule mining results;
- Task creation interface, this interface is used to create mining tasks, the user can select data batches and set the parameters of the association rule mining algorithm through this interface;
- Status query interface, through which users can view the execution status of tasks. If the task execution fails, you can view the failure log, and then restart the task; if the execution is successful, you can download the mining results.

4. System framework design
The import of Web logs is mainly based on the HTTP protocol. Through the configuration of the client, select the Web log data to be imported [4]. The uploaded Web logs are stored in the HDFS after the file merging module, as the basis of data analysis. Then, based on the Spark cluster, mining the Web logs. Web log mining mainly includes cluster analysis and association rule mining. The results of log mining are stored in MySQL and Hive tables, and are persisted to HDFS at the same time. Users can obtain mining information through the result display module. Figure 1 shows the overall system design architecture diagram.
Figure 1. System overall design architecture diagram

4.1. System function design
Figure 2 shows the functional structure diagram of the Web log mining system. The system is mainly divided into three modules: The log management module is mainly responsible for the upload and download functions of logs; the log mining module is mainly responsible for user management tasks, viewing task status and querying task execution results; the user management module is mainly responsible for managing the user's login registration information.
4.2. Database Design
When users use the system to perform mining tasks, in addition to storing Web logs in HDFS, they also need to store some user and task-related tables in the MySQL database to show users related information about task execution and data mining the result of. At the same time, data block tables are also needed to store user-related information, which mainly involves the following four tables:

- **User information table.** As shown in Table 1, it is a user information table, which is mainly provided to users of the super user management system. The field `user_role` is the user's role, the value is `admin` for super user, and the value is `engineer` for ordinary user. Super users can modify common user information by adding, deleting, modifying and checking.

| Field Name     | Types         | Field constraint | Field description |
|----------------|---------------|------------------|-------------------|
| user_id        | INTEGER       | Primary key      | User ID           |
| user_name      | VARCHAR[64]   | non empty        | user name         |
| user_role      | VARCHAR[64]   | non empty        | User role         |
| Creat_time     | DATETIME      | non empty        | Registration time |
| remarks        | VARCHAR[64]   | naught           | Remarks           |

- **File storage table.**
When the Web log is imported into HDFS, it is necessary to store the log file information in MySQL according to the batch number of the file. Among them, the information of the log file in Mysql is consistent with the information of the HDFS file. As shown in Table 2, the main fields included in the log storage information, the `path` field indicates the location of the Web log file on the HDFS, and the `batch_id` is the unique representation of each batch of logs.

| Field Name     | Types         | Field constraint | Field description |
|----------------|---------------|------------------|-------------------|
| batch_id       | INTEGER       | Primary key      | Batch ID          |
| batch_name     | VARCHAR[64]   | non empty        | Batch name        |
| path           | VARCHAR[64]   | non empty        | Storage path      |
| creat_time     | DATETIME      | non empty        | Storage time      |

- **Task execution information table.**
Table 3 shows the main fields and descriptions of the task execution information table. The table mainly stores the information of users performing mining tasks. Among them, `user_id` and `batch_id` are respectively associated with user information and file storage. `Task_name` represents the name of the task execution, which is mainly composed of the timestamp and the uploaded folder name. The `status` field indicates the execution status of the task (0: ready, 1: executing, 2: executing successfully, 3: executing failed). The `conf_info` field indicates that the user's execution task is the selected configuration information, that is, the support and confidence when mining log association rules.

| Field Name     | Types         | Field constraint | Field description |
|----------------|---------------|------------------|-------------------|
| task_id        | INTEGER       | Primary key      | Batch ID          |
| user_id        | INTEGER       | Foreign key      | User ID           |
| batch_id       | INTEGER       | Foreign key      | Log batch ID      |
| task_name      | VARCHAR[64]   | non empty        | mission name      |
| creat_time     | DATETIME      | non empty        | Storage time      |
| start_time     | DATETIME      | non empty        | Starting time     |
• Result information table

As shown in Table 4, it is a table of related information stored in the execution result after the task is successfully executed. This table mainly stores the result information of Web log clustering mining and association rule mining. Among them, the field result_id represents the unique identifier of each mining result, and the task_id is the foreign key that associates the specific information of each mining task. Cluster_num represents the number of clusters in this batch of Web journals mined by the improved K-means algorithm, and the log information of each cluster is stored on the HDFS, and the specific path information is stored in the field cluster_path. The freq_num table 7K is the total number of frequent items mined by the cluster-based FP-Growth algorithm, and the information of the specific frequent items is stored on the HDFS through the path field freq_path. fp_growth_num represents the total number of association rule mining of FP_Growth conversion method, that is, after combining with the improved K-means clustering algorithm, the total number of association rules mined by the association rule mining algorithm for each type of users with similar interests, and The specific association rule information is also stored on the HDFS through the path fp_growth_path. At the same time, the data about Web log information and execution result information stored in HDFS will be imported into the Hive table, which makes it convenient for users to query the results through the Hive table and display the corresponding execution results to the user through the front-end page.

Table 4. Result information table

| Field Name | Types      | Field constraint | Field description                      |
|------------|------------|------------------|----------------------------------------|
| result_id  | INTEGER    | Primary key      | Result ID                              |
| task_id    | INTEGER    | Foreign key      | Batch ID                               |
| user_id    | INTEGER    | Foreign key      | User ID                                |
| batch_id   | INTEGER    | Foreign key      | Log batch ID                           |
| task_name  | VARCHAR[64]| non empty        | mission name                           |
| creat_time | DATETIME   | non empty        | Storage time                           |
| Cluster_num| INTEGER    | non empty        | Number of clusters                     |
| Cluster_path| VARCHAR[256]| non empty   | Cluster storage path                   |
| Freq_num   | INTEGER    | non empty        | Number of frequent items               |
| Freq_path  | VARCHAR[256]| non empty    | Frequent item path                     |
| Fp_growth_num| INTEGER    | non empty        | Total number of association rules      |
| Fp_growth_path| VARCHAR[256]| non empty | Rule storage path                      |
| conf_info  | VARCHAR[256]| non empty        | Configuration information              |

5. System implementation

5.1. MVC framework construction
Django is an open source Web application framework written by Python. Figure 3 shows the overall structure of Django. The code management of the entire system framework of Django is mainly composed of the following files:

- **Urls.py**: This file is used to receive the user's request to access API, and then jump to the view according to the user's request. The corresponding interface in py. As shown in Table 5-5, the mapping relationship between the system url and the interface is defined, for example, "url(r'Alogin', views.login, name='login')", the user accesses the system through the HTML protocol. Page, it will pass urls. The py file requests the user login interface in the view, and then returns the login result. In addition to the user login interface, it mainly includes the interface for uploading and downloading logs, the interface for creating tasks, and the interface for requesting task execution result information.

- **Views.py**: User-defined interface, that is, receive urls. The user request forwarded by py, and then the specific implementation logic of each request is defined in this file. As shown in Table 5-6, it is the service interface for users to request execution result information. First, obtain the server address of the user's download request, then search for the corresponding task_id, and then obtain the specific execution result information according to the task information.

- **Models.py**: Related to database operations, when users request task status and corresponding execution result information, they need to connect to the database and then get specific data. As shown in Table 5-7, model. The class definition for obtaining task execution information in py is mainly to connect to the database, and then connect each field in the database with the previous. The fields displayed in the table on the end correspond to the specific logic in view. Realized in py.

- **Admin.py**: Complete the background configuration by adding configuration code.

- **Settings.py**: Store the configuration information of Djaango, such as the location of static files, etc.

![Django structure diagram](image)

**Figure 3.** Django structure diagram

### 5.2. System function module realization

For web log collection, real-time message systems are generally used to collect, such as kafta and nsq message queue [5]. According to the different business scenarios of the website, different topics are used to collect different Web logs, and then the log files are saved to disk through HDFS. The mining system designed in this paper mainly refers to the mining of the association rules of offline Web logs. The analyzed Web logs are based on the log data left after the user visits the website, rather than
processing the logs being generated in real time. Therefore, to mine and analyze the logs, the user needs to select the location of the imported log data, and then start uploading the Web log.

In order to prevent the web log data uploaded by users from being tampered with, it is necessary to strengthen the security of the web log during the upload process. First, select the encryption algorithm. Commonly used encryption algorithms include symmetric encryption and asymmetric encryption algorithms. However, the sub-asymmetric encryption algorithm needs to use the public key and the secret key, and the encryption and decryption process takes a long time. Therefore, the AES symmetric encryption algorithm is selected to encrypt the uploaded Web log. However, because the secret keys used in the encryption and decryption process of the symmetric encryption algorithm are the same, the security is relatively low. In order to strengthen security, MD5 of the log is used as a secondary check [6]. That is to calculate MD5 for the encrypted log, and then use the comma as the separator to connect the encrypted string to form a new string. In order to reduce the bandwidth consumption in the log transmission process, the new string is compressed through the gzip compression algorithm and then uploaded to the server through the HTTP protocol.

When the server receives the message, it first decompresses gzip, and then obtains the encrypted string and the corresponding MD5 value through the separator. Then calculate the MD5 value. If the MD5 value calculated for the encrypted string is the same as the transmitted MD5 value, it means that the transmitted data has not been modified; if the MD5 value is not the same, it means that the data has been modified during transmission, and then the data is discarded. After the MD5 value is verified, the encrypted string is decrypted with the same AES key, and the decoded string is the uploaded Web log. After the backend obtains the decrypted Web log, it inputs the log into the file pre-processing module [7].

After receiving the decrypted log data, the data is pre-processed. Data pre-processing is a necessary process in Web log mining and the core work of the entire data preparation [8]. Data pre-processing is the foundation of the entire mining process. If the data is not pre-processed well, it will directly affect the rules and patterns generated in the mining process, and it is also a guarantee of mining quality. Data pre-processing mainly includes the stages of data cleaning, user identification, session identification and path supplementation [9].

- **Data cleaning**
  In the original Web log, there are many requests with status codes of 3XX series and 4XX series [10]. These requests indicate redirection or request errors, and also include some requests for web resources with suffixes such as gif and jpg. It is meaningless to analyze user behaviour, so it needs to be filtered out from the original data, and only the GET request with the status code of 2XX series needs to be retained.

- **User identification**
  The user identification stage divides the visits of different users from the data after data cleaning, that is, the user IP is the key and the value is the user's access item, and each access item is composed of the access link and the access time [11].

- **Session recognition and path supplement**
  Session recognition refers to identifying a complete browsing process of a user, that is, a series of page sequence collections visited by the user from visiting the site to leaving the site. This is called a session of the user. The system sets a time threshold of 30 minutes for each session, that is, the time of a session will not exceed the threshold [12]. Due to the impact of the cache of the website proxy server, the user's access request will not generate the corresponding log, so it is necessary to add the access request missed by these servers to the user session to provide a complete data source for Web log mining.

### 6. Conclusion

This paper designs a visual Web log mining system based on big data platform. On the basis of Django's MVC framework, with the help of the open source Bootstrap framework, a Web log storage and mining system for users is realized. This chapter details the internal implementation details of
each module function, and shows the overall framework of the language system and the specific information of each module. Using this system, users can realize the Web log storage, cluster analysis and association rule mining proposed in this paper through simple front-end operations.

Acknowledgments
Thanks for the following fund for supporting this article: Research Project of Lanzhou Vocational and Technical College “Research on the Construction of Innovation and Entrepreneurship Teams in Higher Vocational Colleges Based on the Integration of Production and Education under the ‘Internet +’ Perspective”.

References
[1] E. Steegmans, P. Rigole, T. Holvoet, et al. 2013 Intelligent Buildings: A Multi-Agent System Approach. Artificial Intelligence and Applications Proceedings pp 771-776
[2] Hani Hagras, Victor Callaghan, Martin Colley, et al. 2013 A hierarchical fuzzy-genetic multi-agent architecture for intelligent buildings online learning, adaptation and control Information Sciences 150 pp 33-57
[3] Vladimir Marik and Pavel Vrba 2014 Developing Agents for Manufacturing at Rockwell Automation Agent Link News 12 pp 9-11
[4] Wellman M.P, Birmingham W.P, and Durfee E.H. 2016 The Digital Library as a Community of Information Agents IEEE expert 11 (3) pp 10-11
[5] Wooldridge M and Jennings N.R. 2015 Intelligent Agents: Theory and Practice The Knowledge Engineering Review 10 (2) pp 115-152
[6] Davidsson Paul and Boman Magnus 2015 Distributed monitoring and control of office buildings by embedded agents Information Sciences 171 (4) pp 293-307
[7] Kraus Sarit 2017 Negotiation and cooperation in multi-agent environments Artificial Intelligence 94 pp 79-97
[8] Flax, and Barry. M. 2011 Intelligent Buildings IEEE Communications Magazine 29 (4)
[9] Loveday 2012 Artificial Intelligent for Buildings Applied Energy 41 (3)
[10] Wulf. OPC DX[EB/OL].http://www.hpe.fzk.de/hgf/scs-workshop2002/Wulf.pdf. 2012-11-20.
[11] Kujuro 2013 Systems Evolution in Intelligent Buildings IEEE Communications Magazine 31 (10)
[12] Dave Thompson 2020 A BACnet gain Engineered System 17 pp 108-116