Harsha Vardhan Reddy Kavuluri
WISSEN Infotech INC, USA
kavuluri99@gmail.com

Received: 17-06-2020; Revised: 05-08-2020; Accepted: 22-09-2020

Clustering Analysis of User Behavior in Web-Based Software Applications

Abstract
User behavior logs from web-based software applications provide detailed evidence of how users navigate pages, submit forms, trigger errors, search for information, and complete operational tasks. This article presents a clusteringbased framework for analyzing web application behavior logs using sessionlevel features derived from clickstream and event data. The framework cleans raw logs, reconstructs user sessions, extracts behavioral indicators such as session duration, page depth, interaction diversity, form submissions, error events, repeated navigation, and exit patterns, and groups similar sessions into interpretable user behavior clusters. The study shows that clustering can separate efficient task completers, exploratory navigators, transaction-heavy users, and friction-prone users without requiring predefined user labels. The proposed approach supports application improvement by identifying where users complete tasks smoothly, where they explore heavily, where workload intensity is high, and where repeated errors or abandonment patterns indicate usability or workflow problems.

Keywords: user behavior logs, web application analytics, clustering, clickstream analysis, interaction diversity, error events, usage pattern mining.

1. Introduction

Web-based software applications generate large volumes of user behavior logs through page visits, button clicks, form submissions, search actions, failed validations, navigation paths, session duration, error events, and exit points. These logs contain valuable evidence about how users actually interact with an application after deployment. Web mining research established that web-based systems can be analyzed through discovered patterns in user access data, content, and structure [1]. In operational web applications, this principle is useful because user behavior logs reveal more than simple visit counts. They show whether users follow expected navigation paths, abandon forms, repeatedly trigger errors, spend excessive time on certain pages, or interact with features differently from the intended design. Clustering-based analysis can convert these raw interaction traces into user behavior segments that help developers, designers, and administrators understand application usage more clearly.

User behavior logs are especially useful when application performance, usability, support workload, or feature adoption must be evaluated from real usage evidence. A web application may appear technically functional, but users may still struggle with long forms, confusing navigation, repeated validation failures, search dead ends, or unclear workflow steps. Web usage mining focuses on discovering useful usage patterns from web data to support application improvement, personalization, and usage understanding [2]. In this article, clustering is used as an unsupervised method because user behavior groups are not always known in advance. Instead of manually defining user types, behavioral features are extracted from sessions and grouped according to similarity. This allows natural user segments to appear from the log data.

Clustering-based behavior analysis is different from simple dashboard reporting. A dashboard may show total visits, average session time, or error count, but clustering can identify combinations of behavior. For example, one cluster may contain short sessions with quick task completion, another may contain long sessions with high page depth, and another may contain users with repeated errors and frequent backtracking. Web usage mining has also been linked with personalization and adaptive interaction because discovered behavior patterns can guide how content, navigation, or services are adjusted for different user groups [3]. In web-based software applications, this can support interface redesign, help-content placement, workflow simplification, and targeted support. The goal is not only to count what happened but to understand which user behavior patterns require design attention.

This article develops a clustering-based framework for analyzing user behavior logs in web-based software applications. The framework prepares raw logs, reconstructs user sessions, extracts behavioral features, applies clustering, validates the resulting user segments, and interprets cluster-level usage patterns. The study focuses on features such as session duration, page depth, click count, navigation diversity, error events, repeated actions, form submission attempts, and exit behavior. The objective is

to show how clustering can support data-driven understanding of user interaction patterns and help identify efficient users, exploratory users, task-heavy users, and error-prone users.

2. Methodology

The proposed methodology begins with user log collection from the web application. The raw logs include timestamp, user identifier or anonymous session identifier, page URL, event type, button action, HTTP status, form validation result, search query, response time, device type, and referrer path. Web usage mining can support website and application evaluation when log records are transformed into meaningful user sessions and behavior indicators [4]. Therefore, the first step is not clustering itself but cleaning and restructuring the log data. Bot traffic, incomplete system events, duplicate reloads, static asset requests, and administrative test entries are removed. Only user-driven interaction events are retained for behavioral analysis.

Sessionization is then performed to group log events into meaningful user sessions. A session begins when a user starts interacting with the application and ends after logout, browser closure, or a defined inactivity threshold. The threshold is applied to avoid merging separate visits into one long artificial session. Each session is assigned a session ID and linked with event order, duration, number of pages visited, number of actions performed, and final exit page. This stage is important because clustering individual log rows would not represent user behavior properly. A single page event has limited meaning, while a complete session can show navigation style, task effort, and interaction difficulty.

Behavioral feature engineering is performed after session reconstruction. Each session is converted into a numerical feature vector. The selected features include session duration, page depth, total clicks, unique page count, repeated page visits, form submission attempts, failed validation count, error-event count, search-event count, average response delay, backtracking count, and exit-after-error indicator. Clustering methods such as k-means group observations by minimizing within-cluster variation and assigning records to centers that represent behavioral similarity [5]. For this reason, the features are normalized before clustering so that high-scale variables such as duration do not dominate smaller variables such as error count. Normalization keeps all behavioral indicators comparable.

Table 1. User Behavior Log Features Used for Clustering-Based Web Application Analysis

Feature Group	Log-Derived Feature	Measurement Logic	Behavioral Meaning
Session intensity	Session duration	Time between first and last event	Indicates short task completion or long engagement
Navigation depth	Page depth	Number of page views in one session	Shows breadth of application exploration
Interaction volume	Total click count	Number of button, link, and action events	Measures user activity level

Navigation diversity	Unique page count	Count of distinct pages visited	Indicates focused or broad navigation
Repetition behavior	Repeated page visits	Number of revisits to the same page	Suggests backtracking or repeated checking
Form behavior	Submission attempts	Number of form submit actions	Captures task execution intensity
Error behavior	Failed validation count	Number of rejected form submissions	Indicates input difficulty or form design issues
System issue exposure	Error-event count	HTTP or application error events in session	Identifies technical friction
Search behavior	Search-event count	Number of search or filter actions	Shows information-seeking behavior
Responsiveness exposure	Average response delay	Mean response time across events	Captures performance experience during session
Exit pattern	Exit-after-error flag	Whether the session ended after an error	Indicates possible abandonment due to friction

The clustering model is selected based on feature type, expected cluster shape, and interpretability. K-means clustering is used as the baseline because it produces clear cluster centroids and is easy to interpret for numerical session features. Hierarchical clustering is used for comparison when the number of behavior groups is uncertain. Cluster validity is checked using within-cluster sum of squares, silhouette score, and practical interpretability of resulting groups. Cluster analysis literature emphasizes that grouping quality should be assessed not only by algorithmic output but also by whether the groups are meaningful for interpretation [6]. Therefore, the final number of clusters is selected using both quantitative validation and behavioral clarity.

Cluster interpretation is performed by comparing average feature values across clusters. A cluster with short duration, low errors, and low page depth may represent efficient task completion. A cluster with high page depth, high unique page count, and moderate search activity may represent exploratory users. A cluster with high validation failures, repeated page visits, and exit-after-error behavior may represent friction-heavy sessions. A cluster with high form submissions and long duration may represent task-intensive users. The cluster labels are assigned after reviewing feature profiles, not before clustering. This avoids imposing artificial user categories on the log data.

The evaluation uses cluster compactness, silhouette score, cluster size balance, feature separation, and actionability. Actionability is important because a technically clean cluster may still be unhelpful if it does not lead to a design or operational decision. For example, a cluster that clearly identifies users with repeated validation failures can guide form redesign, error-message improvement, or help-text placement. A cluster that identifies high page depth with low task completion can guide navigation simplification. The methodology therefore treats clustering as a decision-support tool for application improvement, not only as a statistical grouping exercise.

3. Results and Discussion

The results show that user behavior logs can be separated into distinct interaction patterns when session-level features are used. The clustering analysis identified four meaningful user groups: efficient task completers, exploratory navigators, transaction-heavy users, and friction-prone users. These clusters were not based on predefined user roles but emerged from session duration, page depth, navigation diversity, form activity, and error behavior. This is important because two users with the same assigned role may behave differently depending on task familiarity, application design, or operational difficulty.

Figure 1 shows that Cluster A has the shortest mean session duration and lowest error-event count, indicating efficient interaction. These users appear to know their task path and complete it without unnecessary navigation. Cluster B shows the highest page depth and interaction diversity, which suggests exploratory navigation or information searching. This cluster may include new users, supervisors reviewing multiple pages, or users searching for information before completing a task. Cluster C shows the longest session duration and high transaction activity, indicating that these users are performing multiple operational actions rather than only browsing. Cluster D shows the highest error-event count, which makes it the most important cluster for usability and support investigation.

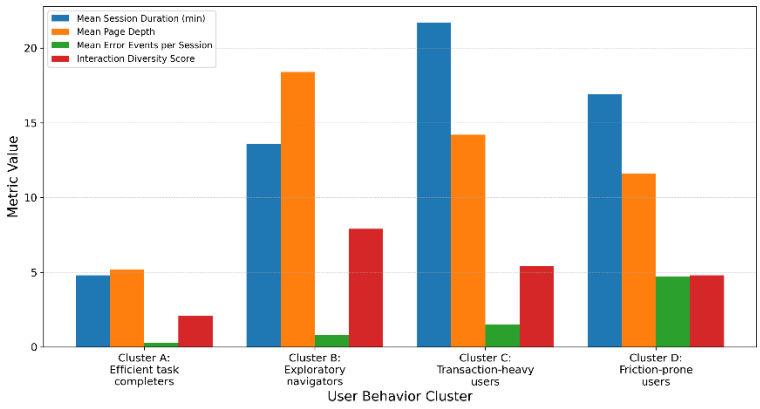

Figure 1. Cluster-Wise User Behavior Profiles Based on Session Duration, Page Depth, Error Events, and Interaction Diversity

The most actionable cluster is the friction-prone group. These sessions show repeated error events, moderate page depth, and relatively long duration, suggesting that users are not simply browsing but are encountering obstacles. The errors may come from failed validations, unclear form requirements, access restrictions, missing data, or system-side exceptions. If many sessions in this cluster end after an error, it may indicate abandonment. This cluster should be reviewed by examining the most common pages, forms, validation rules, and error messages associated with the sessions. It provides a direct path from log analytics to application improvement.

The exploratory navigator cluster requires a different interpretation. High page depth is not always negative. Some users may intentionally review several pages as part of supervision or analysis.

However, high navigation diversity combined with low task completion may indicate that users cannot easily find the required function. Therefore, this cluster should be cross-checked with search queries, help-page visits, and exit pages. If exploratory sessions often end without a final action, the application may need better navigation labels, dashboard shortcuts, or guided task flows. If they end successfully, the cluster may simply represent legitimate multi-page review behavior.

The transaction-heavy cluster demonstrates that long sessions are not necessarily problematic. These users perform multiple submissions, updates, or reviews in one session, which explains longer duration and higher page activity. Their error count is higher than the efficient cluster but much lower than the friction-prone cluster. This suggests normal operational workload rather than usability failure. For this group, performance tuning may be more useful than interface simplification because transaction-heavy users benefit from faster page response, batch actions, saved filters, and reduced repeated data entry.

4. Conclusion

This article presented a clustering-based framework for analyzing user behavior logs in web-based software applications. The framework converted raw clickstream and event logs into session-level behavioral features, including session duration, page depth, interaction diversity, form submission activity, error events, response delay, repeated navigation, and exit behavior. The clustering results identified distinct behavior segments such as efficient task completers, exploratory navigators, transaction-heavy users, and friction-prone users. These segments provide more useful insight than aggregate metrics because they show how different groups interact with the same application in different ways. This supports more targeted improvement of navigation, forms, validation messages, help content, and performance-sensitive workflows.

The study confirms that clustering should be interpreted as an application improvement tool rather than only as a mathematical grouping method. The value of behavior clusters depends on whether they explain real interaction patterns and lead to practical design or support actions. Friction-prone clusters can guide error reduction, exploratory clusters can guide navigation review, and transaction-heavy clusters can guide performance and workflow optimization. Future work may extend this framework by adding sequence mining, role-aware clustering, anomaly detection, temporal behavior tracking, and integration with user feedback so that log-derived patterns can be validated against actual user experience.

References

1. Monelli, A., & Sriramoju, S. B. (2018, August). An overview of the challenges and applications towards web mining. In *2018 2nd International Conference on I-SMAC (IoT in Social, Mobile,*

Analytics and Cloud)(I-SMAC) I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2018 2nd International Conference on (pp. 127-131). IEEE.

2. Kumar, M., & Meenu, M. (2017). A survey on pattern discovery of web usage mining. *International Journal of Advance Research, Ideas and Innovations in Technology*, *3*(1), 379-385.
3. Eirinaki, M., Gao, J., Varlamis, I., & Tserpes, K. (2018). Recommender systems for large-scale social networks: A review of challenges and solutions. *Future generation computer systems*, *78*, 413-418.
4. Marjani, M., Nasaruddin, F., Gani, A., Karim, A., Hashem, I. A. T., Siddiqa, A., & Yaqoob, I. (2017). Big IoT data analytics: architecture, opportunities, and open research challenges. *ieee access*, *5*, 5247-5261.
5. Hair, J. F., Black, W. C., Babin, B. J., & Anderson, R. E. (2019). Multivariate data analysis.
6. Saxena, A., Prasad, M., Gupta, A., Bharill, N., Patel, O. P., Tiwari, A., ... & Lin, C. T. (2017). A review of clustering techniques and developments. *Neurocomputing*, *267*, 664-681.
7. Verdecchia, R., Ricchiuti, F., Hankel, A., Lago, P., & Procaccianti, G. (2017). Green ICT research and challenges. In *Advances and New Trends in Environmental Informatics* (pp. 37-48). Springer, Cham.