Location Recommendation System based on Google Review Rating Prediction

Dang Huynh Vinh Tan[1,2], Nguyen Thi Hong Nhung[1,2], Nguyen Viet Kha[1,2], Nguyen Huy Hoang[1,2] and Trong Hop Do[1,2]

[1] University of Information Technology, Ho Chi Minh City, Vietnam
[2] Vietnam National University, Ho Chi Minh City, Vietnam
{21520442, 21522436, 21520949, 21522093}@gm.uit.edu.vn, hopdt@uit.edu.vn

Abstract The recommendation system is a data analysis system that analyzes user data and makes predictions, suggesting recommendations believed to be suitable for the user's desires and preferences at any given time, helping to save time searching, accessing content easily, and enhancing customer experience. In this project, we aim to build a recommendation system to suggest locations based on predicting ratings from reviews on Google. We use a self-collected dataset in various areas of Ho Chi Minh City to build and train the model. We utilize a regression model for training, and then this data will go through Spark Streaming and Spark SQL for processing, querying, and data extraction, and will be stored in the MongoDB database management system. After that, we use the k-Nearest Neighbors (kNN) model with the "Brute Force" algorithm to find the nearest locations.

Keywords: Big Data · Recommendation System · Spark Streaming · Machine Learning · Kafka · Dash · Plotly

1 Introdution

This research project focuses on predicting the ratings of location reviews on Google, aiming to enhance the system's capability to recommend places for users based on historical data and sentiments within reviews. We collect data from Google Maps in Ho Chi Minh City, then explore detailed information about user reviews and locations, primarily with the goal of estimating review scores by applying diverse features within the dataset. Additionally, we construct a recommendation system and compile user review statistics to improve user location search through reviews.

The dataset encompasses millions of users, local businesses, and reviews, providing a diverse set of business types. Our objective is to develop prediction models based on features such as review text, location information, and user review history. Through this, we aim to categorize and suggest similar locations to users.

By applying machine learning models and data analysis, we hope to create an intelligent location recommendation system, enhancing users' experiences in

searching and selecting places based on predicted historical data. Furthermore, we aspire to apply the research results practically, supporting decision-making and location selection based on community emotions and opinions on the Google Maps platform.

2 Relevant Research

The recommendation system is increasingly being applied extensively in user data analysis and prediction, suggesting and providing recommended content for users. Currently, location recommendation systems are widely integrated across various sectors such as e-commerce, tourism, and entertainment. These systems not only assist in predicting review scores but also support users in searching and selecting locations that align with their preferences and personal interests. This helps save time, accelerates the search process, and facilitates users in easily accessing places of interest. With the growing application of these systems, location recommendation systems are garnering increasing attention and research from the community, aiming to improve performance and better cater to user needs.

3 Identification and Analysis of Structure

3.1 Collecting Data

Collecting data from Google Maps in various areas of Ho Chi Minh City using the Apify tool. Apify is a cloud-based platform for web scraping, data extraction, and automation. It provides tools for data collection from websites and APIs, allowing users to run scalable web scraping tasks, automate workflows, and store and process data in the cloud.

3.2 Datasets Overview

The data we collected included two parts:

1. Places Data: This dataset included 8,210 places with a 3.24 MB file size. Table 1 shows the description of this dataset.
2. Reviews Data: This dataset included 1,737,239 reviews with a 482 MB file size. Table 2 shows the description of this dataset.

3.3 Data Cleaning

Due to the massive global volume of data in the original dataset, the study team encountered two issues with their subsequent analysis. The first issue is that processing all the data becomes challenging owing to limited computing capability. The second issue is that in order to improve the accuracy of text analysis, the study team wants to restrict the language to only Vietnamese. The dataset is cleaned using the following procedures:

#	Attribute	Meaning	Datatype
1	placeId	Id of a place	String
2	title	Name of place	String
3	categoryName	Place category	String
4	location/lat	Latitude of a place	Float
5	location/lng	Longitude of a place	Float
6	address	Place address	String
7	permanentlyClosed	Operation status	Bool
8	url	Url to GoogleMap	String
9	categories	List of all categories of place	String

Table 1. Places Data description.

#	Attribute	Meaning	Datatype
1	reviewId	Id of a review	String
2	placeId	Id of a place	String
3	title	Name of place	String
4	location/lat	Latitude of a place	Float
5	location/lng	Longitude of a place	Float
6	reviewerId	Id of reviewer	String
7	name	Name of reviewer	Bool
8	stars	Rating point	Int
9	text	Reviewer text	String
10	publishedAtDate	Review time	Datetime
11	categories	List of all categories of place	String
12	categoryName	Place category	String

Table 2. Reviews Data description.

1. Eliminate duplicate places and duplicate reviews.
2. Remove rows with missing data in important columns.
3. Remove unnecessary columns.
4. Verify that the data collection contains reviews for the places.
5. Remove reviews text in languages other than Vietnamese using langdetect library.
6. Take out any reviews that are too brief. Sometimes users reviews simply provide their assessment of the review site, leaving out any further remarks or more general remarks like this: "Đẹp", "Tốt",...
7. Take the icon out of the review text. User-added icons occasionally appear in some reviews text.
8. Normalize Unicode characters for reviews text.
9. Separate words from punctuation with spaces for reviews text.

After the above process, we had a total of 456,792 reviews and 6,298 locations.

3.4 Data Transformation

The research team wants to add some attributes to the data set after analyzing the data. As a result, the following steps are followed when performing the data transformation process:

1. Using UnixReviewTime datatype to format "publishedAtDate" column.
2. Add reviews text length and reviews hour as attributes
3. Add a column content a list of sentences of each reviews text

4 Model Creation

To create a model, we need a fixed-sized representation of each review. The research team's goal in the scenario under consideration is to use feature vectors to characterize the rating. This will enable us to evaluate user ratings of various locations and give particular attributes weights in order to perform prediction problems. We must perform some preprocessing on reviews, including eliminating punctuation, capitalization, acronym replacement, expressive character removal, etc. The review's sentiment is then graded by assigning a "negative," "positive," or "neutral" feeling to each sentence. After that, we can figure out the review's sentiment ratio as a whole. Lastly, we forecast user rating scores using that parameter.

4.1 Regression model with emotion ratio weights

First, the research team create a regression model based on basic information about emotion rates as features for the model.

$$\text{Rating} = \theta_0 + \theta_1 \times \text{ProportionOfPositive} + \theta_2 \times \text{ProportionOfNegative}$$

Next, assign a 90%:10% ratio to the features and labels in the training, validation, and testing sets. Using ridge regression, we trained the linear model on the training set, adjusted the model's parameters on the validation set, and evaluated the outcomes on the test set. The training set's Mean Squared Error (MSE) is 4.759583, whereas the testing set's MSE is 4.793305, according to the results.

4.2 Model Explanation

While examining the data set, the team took into account how the user's review score was affected by the length of the review set, the time the review was written, and certain details about each location. The group chose to incorporate these elements into the model. We obtain the subsequent formula:

$$\begin{aligned}\text{Rating} = \theta_0 + \theta_1 \times \text{ProportionOfPositive} + \theta_2 \times \text{ProportionOfNegative} \\ + \theta_3 \times \text{ReviewHour} + \theta_5 \times \text{ReviewLength}\end{aligned}$$

We conduct evaluation in a manner similar to the original model, and the findings indicate that the train's MSE of 3.466059 and the test's MSE of 3.497937 are both better than the original model.

5 Recommendation System

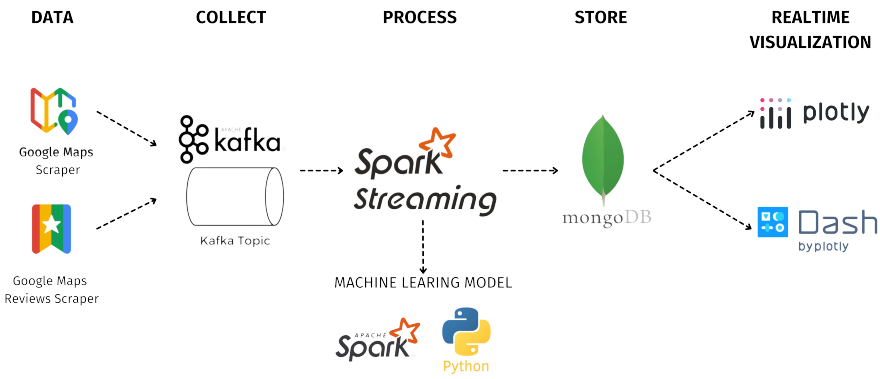

Figure 1. Overview of System Architecture

The proposed system architecture consists of: real-time data retrieved from Google Maps related to Google Reviews preprocessed using Python and stored in Kafka; Spark Streaming reads the data stream from Kafka and feeds it into Spark Mllib, Python, and Spark DataFrame for processing and analysis; The processed and analyzed data will be passed through a Linear Regression model; This data will go through Spark Streaming and Spark SQL for processing, querying, and data extraction, and will be stored in the MongoDB database management system; To create a visual interface for the system, we used Dash and Plotly. Dash is an open-source library released under the MIT license. Written on Plotly.js and React.js, Dash is ideal for building and deploying data applications with custom UIs (user interfaces), while Plotly is a powerful data visualization library.

The system's processing components consist of three elements: Offline processing system, Online simulation processing system, and Web application as follows:

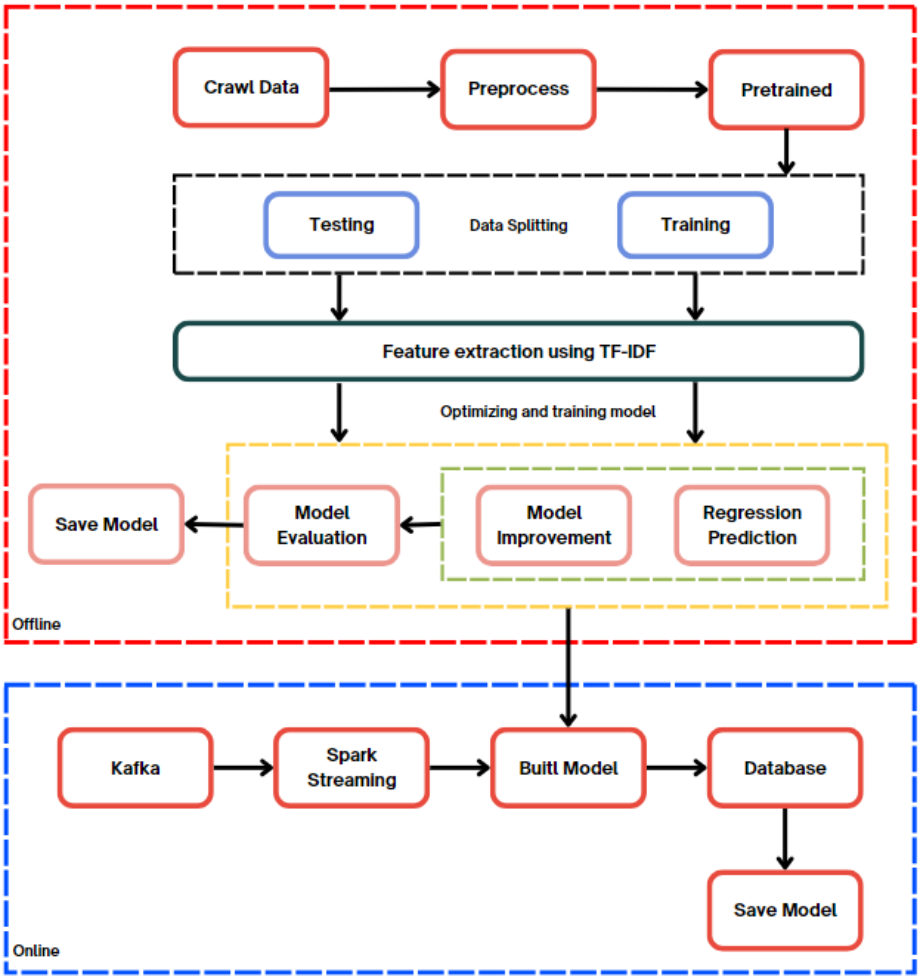

Figure 2. Offline processing and Online simulation processing system

– Offline processing system: The offline processing component is used for training machine learning models based on data collected from Google Maps, where the model used is a regression model. The steps are as follows:
  - Step 1: Prepare the working environment and collect data from Google Maps.
  - Step 2: Data preprocessing involves several steps including data cleaning, removing rows with missing data, and sorting the data based on GPS coordinates.
  - Step 3: Utilize the Linear Regression machine learning model in Spark MLlib to train the model.

  - Step 4: The data is pushed to the MongoDB database management system, then passed through the recommendation system. The recommendation system is built to provide location recommendations based on user preferences and location. Data is collected from MongoDB and used to build the recommendation model using TF-IDF and KNN.
  - Step 5: Data preprocessing: Split the values in the 'location' column into 'location/lat' and 'location/lng', convert data from list format in the 'categories' column into a string.
  - Step 6: Model Building: Utilize the k-Nearest Neighbors (kNN) model with the "Brute Force" algorithm to find the nearest locations based on the TF-IDF vector.
  - Step 7: The recommended locations are sorted based on the geographical distance (haversine distance) from the input location and filtered by the ID of the input location. The results are sorted in descending order of average rating.
  - Step 8: The final result will include information about the input location and a list of 10 recommended locations.

- Online simulation processing system: consists of the following steps:
  - Step 1: The simulated online data stream is stored in JSON format and saved to Kafka.
  - Step 2: Spark Streaming reads data from Kafka for analysis and processing. This involves preprocessing the data and conducting analysis similar to the offline processing system.
  - Step 3: Use the pre-trained machine learning model to analyze, predict the data, and store the results in MongoDB.

- Web application: Dash reads the result data stored in MongoDB and utilizes Plotly to create charts and maps to visualize the reporting and statistical data.

6 Experimentation and Evaluation of Results

6.1 Results

This section compares the model accuracy and provides an evaluation of the findings. The following table illustrates how the outcomes are compared using the Mean Squared Error (MSE) parameter:

Model	Train's MSE	Test's MSE
Initial model	4.759583	4.793305
Explanation model	3.466059	3.497937

Table 3. Model comparison

The explanation model outperforms the original model in terms of Mean Squared Error (MSE) outcomes, as indicated by Table 3's findings.

6.2 Demo

To enhance the vividness of location recommendations, the team has developed a simple real-time web application with visualizations as shown in some images below:

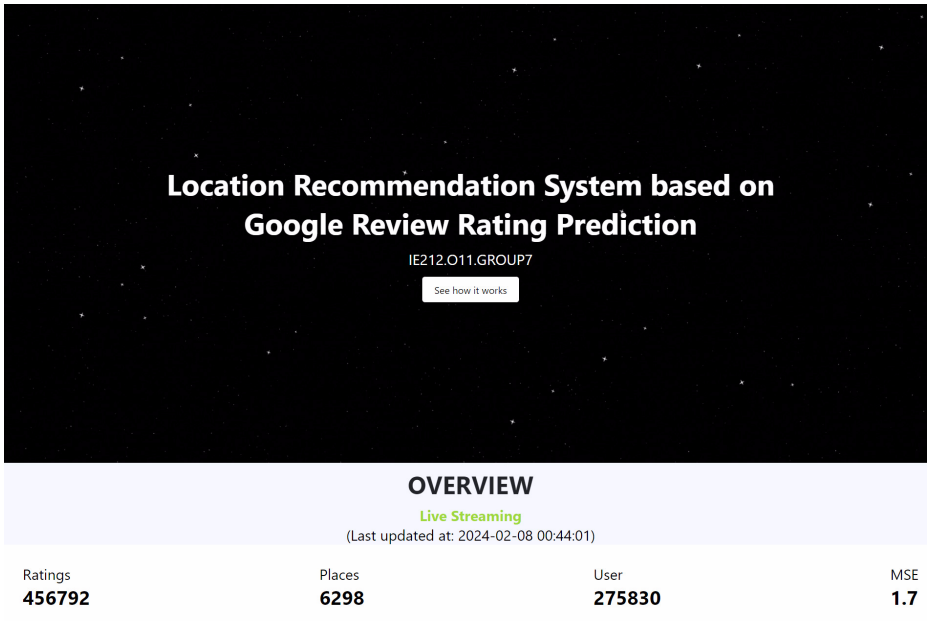

Figure 3. Overview

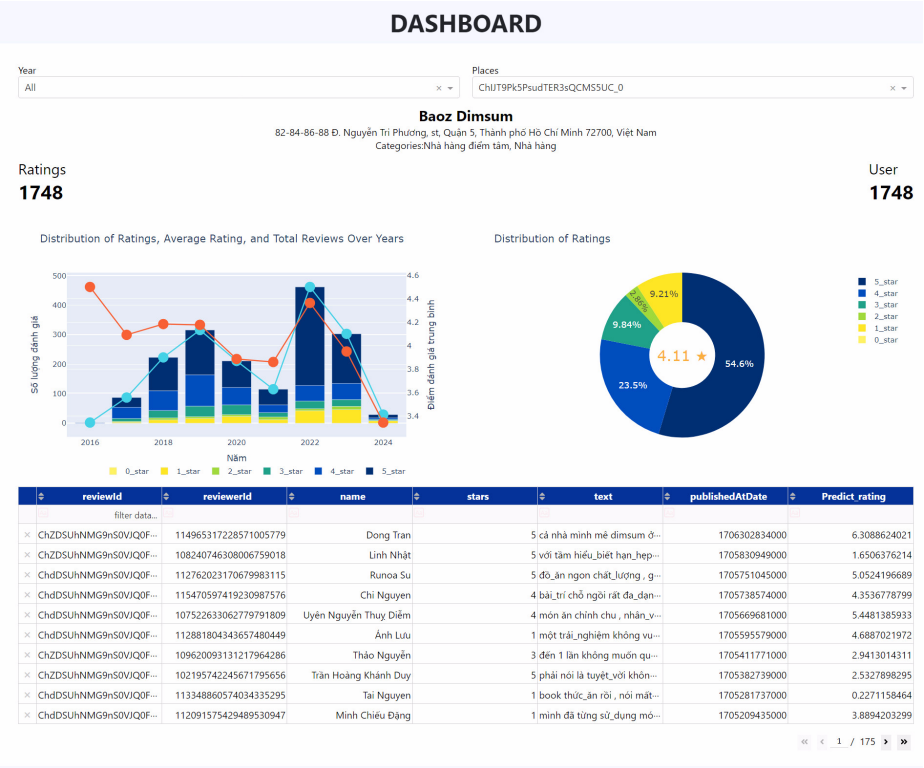

Figure 4. Data statistic

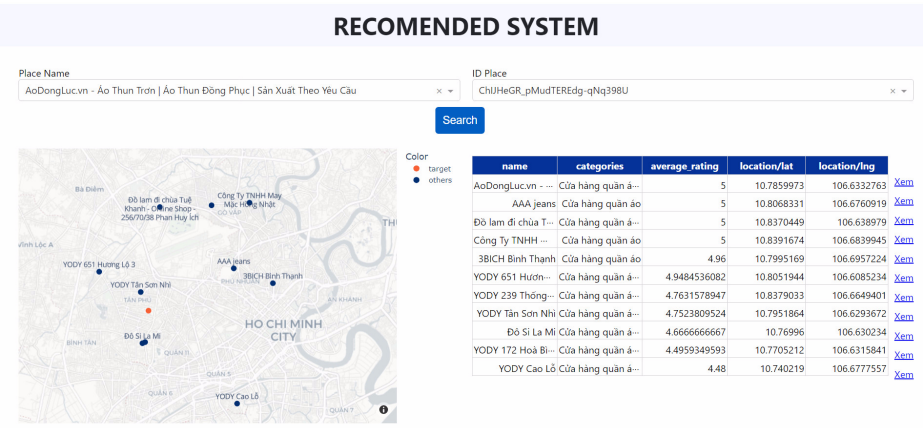

Figure 5. Recommendation system

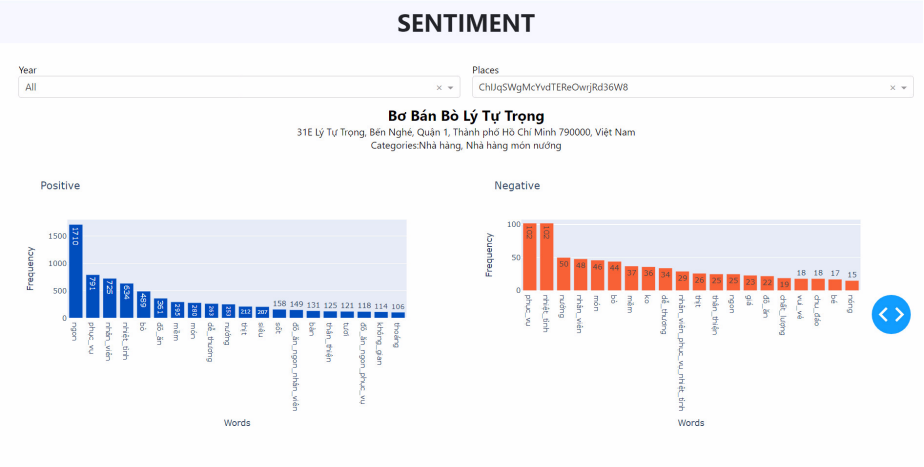

Figure 6. Sentiment

7 Conclusion and Future Development Direction

In the conclusion of this report, the team focuses on building a Location Recommendation System based on real-time prediction of Google Review Scores. The report introduces and explores related works, conducts identification and analysis of structures, builds prediction models, presents the system design and architecture, along with the experimentation process and result evaluations.

Additionally, the team has developed a real-time data updating web application, enhancing the usability of the product. However, the application has a slightly slow page loading speed.

For future development, the team suggests integrating additional data sources from other applications such as social media platforms to expand and improve the accuracy of the rating and recommendation system. Furthermore, plans include integrating support for multiple languages and advanced processing techniques to ensure high-quality and accurate results during the location recommendation process. The team also aims to develop new features and enhance performance to meet the increasingly diverse and complex needs of users.

References

1. Rajiv Pasricha, Julian McAuley, Translation-based factorization machines for sequential recommendation, RecSys, 2018
2. He, Wang-Cheng Kang, Julian McAuley, Translation-based recommendation Ruining, RecSys, 2017