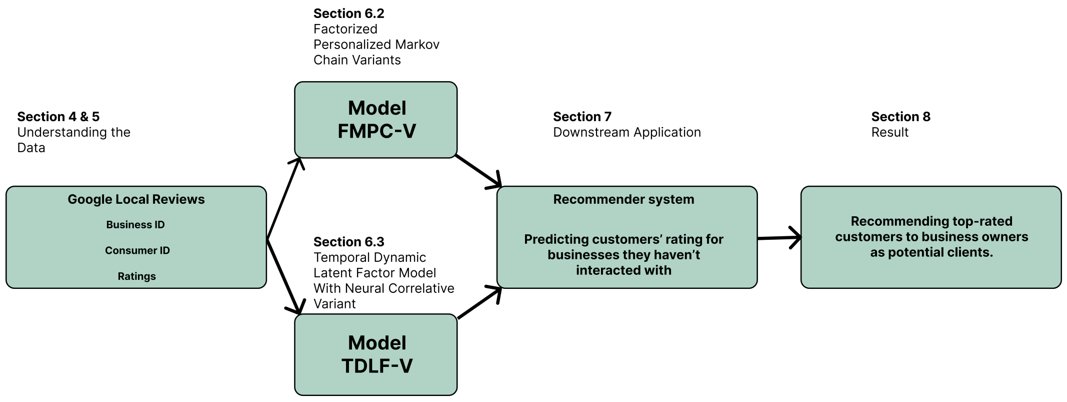
Recommendation Based on Sequential Dynamics for Business Owners

Kaiwen Bian[*]
Halıcıoğlu Data Science Institute
UC San Diego
La Jolla, CA 92093
`kbian@ucsd.edu`

Guoxuan Xu
Halıcıoğlu Data Science Institute
UC San Diego
La Jolla, CA 92093
`g7xu@ucsd.edu`

Jiaying Chen
Cognitive Science Department
UC San Diego
La Jolla, CA 92093
`jic098@ucsd.edu`

Weijie Zhang
Halıcıoğlu Data Science Institute
UC San Diego
La Jolla, CA 92093
`wez042@ucsd.edu`


Section 4 & 5
Understanding the Data
Google Local Reviews
Business ID
Consumer ID
Ratings
Section 6.2
Factorized Personalized Markov Chain Variants
Model FMPC-V
Section 6.3
Temporal Dynamic Latent Factor Model With Neural Correlative Variant
Model TDLF-V
Section 7
Downstream Application
Recommender system
Predicting customers' rating for businesses they haven't interacted with
Section 8
Result
Recommending top-rated customers to business owners as potential clients.

1 Introduction

Nowadays, online platforms such as Yelp, Google Reviews, and Reddit have become the primary interface between business and their customers. Such platforms allow users to comment and rate the business, creating a wealth of metadata that reflects consumer preferences and perceptions. These data provide a unique opportunity for business owners to make informed, data-driven decisions. We developed a hybrid recommender system that combines collaborative and content-based filtering, recommending potential customers to specific business owners. The model was trained using key business features, historical interactions between businesses and customers, and temporal trends. We trained two advanced models, with one taking the temporal feature and one taking the sequential feature of the dataset to predict the target value, which is the rating that customers might leave on the business. Once trained and given a learned business ID, the model will identify customers who have never interacted with the business but would give a high rating.

[*]documentations at https://kbian.org/RSDB/

2 Related Works

We use a dataset collected from Google Maps local user review by UC San Diego researchers. This dataset serves as a foundational data source for two significant works [7, 13]. In [13], the author proposes a personalized multi-modal framework to generate diverse textual and visual explanations for recommendation. Specifically, they collect a large-scale dataset from Google Maps user review and filter a subset of high-quality visual data specifically for generating visual explanations. In [7], the author introduces a new method called cluster-assisted contrastive learning (CCL), which mitigates the issue of noisy negatives in negative sampling. They collected large-scale textual data from Google Maps user reviews in the restaurant category, which was then used in topic phase mining. Our work builds on the datasets established by these two papers, which offer rich contextual data about user reviews and business information.

Similar datasets studies in the past include Yelp reviews data [1] and Amazon product reviews data [4], which have been extensively used for recommender systems, sentiment analysis, text mining, etc. These datasets share some similar characteristics, such as user-generated content, diverse textual information, and user-business matching, making them suitable for developing all kinds of recommendation system task. However, the Google Maps Local Review dataset we used distinguishes itself from other datasets through its focus on users reviews for diverse local physical business and geospatial context. The dataset aligns with our problem definition of recommending potential customers to business owners.

Sequential recommendation focuses on predicting the user's next interactions with items by modeling the dynamics and temporal preference. To address this challenge, some of the state-of-the-art methods employ advanced models such as Recurrent Neural Networks recommendations [9], Transformer architecture recommendations [12], and Large Language Model recommendations [2], etc. For instance, in [9], the author proposed FitRec, an LSTM-based framework for context-aware modeling for fitness data. The method uses within-activity and across-activity fitness data to model the user's workout profile and short-term heart rate dynamic. The model uses a two-layer stacked LSTM and an encoder-decoder network with attention for modeling user interaction. The methods aim to recommend potential workout plans for users and to predict heart rate changes during workouts. In [12], the author introduces BERT4Rec, a Bidirectional Encoder Representation from Transformers for sequential Recommendation to model user interactions. The Transformer architecture is known for the self-attention mechanism, which captures complex dependencies across the entire context sequences. The author trained a bidirectional model to predict the masked items by conditioning on both left and right contexts. The approaches allow for a better representation of the sequential user interaction dynamic. The recent advancements in LLM have shown its remarkable ability, showing promise for sequential recommendation. For example, in [2], the author proposes the Hierarchical Large Language Model (HLLM), a two-tier architecture for sequential recommendation. The method is composed of two LLMs, the Item LLM extracts context-rich features from the item description, while the Users LLM predicts future user interest based on interaction history. HLLM offers excellent scalability and achieves state-of-the-art performance on recommendation, specifically for item feature extraction and user interest prediction. The conclusion and result of our work can be found in section 8 and section 9.

3 Predictive Task

Our predictive task is to develop a hybrid recommender system that combines collaborative and content-based filtering to recommend potential customers to small business owners using Google Local Review data [7, 13]. In particular, we will train a sequential recommendation model specifically for predicting the user's future rating with the business. To evaluate the model, we will use metrics: Root Mean Square Error(RMSE), Mean Absolute Scaled Error(MASE), R-square($R\hat{2}$), and Accuracy. These metrics will provide fair judgment on the models' ability to predict the right target value and variability of the data. After we have obtained the trained model, we will apply our model in the downstream recommender system application. The downstream recommender will use the model to predict users' ratings of a given business they have never interacted with, then from there, the system will rank the predicted users' ratings and recommend these users who are more likely to give high ratings to the given business. We hope our downstream recommender application can help the business owner find their target customers easily, and hopefully, they can use this information to expand their business. To ensure a more privacy-preserving recommendation system, we have removed all personally identifiable information (PII) during the data processing, feature engineering, and downstream recommendation process. (Further implementation details of our downstream application are described in section 7)

In order to evaluate our model for predicting rating, we use train-test-split to split our data into the training set, testing set, and validation sets that respect the temporal order of observations. We use the training set and the validation set to train and tune our model for good hyperparameters. We then use the testing data to evaluate our model performance for unseen data. We use the following three metrics to evaluate our performance: Mean Square Error/Root Mean Square Error, R-square, and Mean absolute scaled error. In addition, we also employ qualitative measures to assess our model and downstream application (which we further discuss and explain in section 8.2).

For comparison with the baseline model, we use a basic latent factor model [6] that only models the interaction between user and items through latent rating interactions and bias terms. We use the Factorized Personalized Markov Chains [11] and the Temporal Dynamic Latent Factor Model with Neural Correlative [5] as our final model choice. We slightly modify the model to better fit with our data and recommendation objective. These models are good at sequential recommendation and can easily be scalable to large-scale datasets. We discuss our final model in detail with mathematical formation in section 6.

Data preprocessing and feature engineering are described below in section 4 - Processing Data and section 5 - Feature Engineering.

4 Processing Data

The Dataset we use contains review information on Google Maps made by customers on business across the United States. Due to limited resources and time contain, we sample a subset of data review data in California.

4.1 Data Overview

The raw data comes in two separate files: business metadata and review data. The business metadata contains information about 15,291 businesses in California, structured into 13 columns. Each row represents a business, and the columns include their unique identifier, business name, address, and so on. On top of that basic information, they also include operating hours, longitude & latitude, and business categories, which we use extensively in feature engineering. The review dataset represents the interaction between customers and businesses, where each row is a review left by the customer on the business. Some features of the review dataset include the review time, rating, etc.

4.2 Data Cleaning

Since we want to model the interaction between customers and specific businesses, we prepare data with procedures that will produce a single dataframe where each row represents interactions between a single user and a specific business and the rating made by the user. In cleaning business metadata, we rearrange and keep key columns for readability and clarity. Out of all the cleaning procedures, these are the cortical steps we took:

1. We dropped the "State" column to avoid multicollinearity. The "State" column captures whether the store was open or not at the time the dataset was created, which is covered by the "Hours" column.
2. To clean review information data, we conduct a similar procedure. For instance, we removed any review with a timestamp before February 8th, 2005. Therefore, we dropped all entries with review time after February 8th, 2005.
3. To handle missing values in both datasets, we removed rows containing missing data because the percentage of missing values was minimal, ensuring the analysis remained unaffected.
4. To ensure the relevance of the reviews dataset to business, we applied a series of filtering to exclude outliers and less significant data points. We first focused on eliminating business categories with limited representations. Rows with categories that appear fewer than 20

times in the entire dataset were removed to reduce noise and focus on more prominent business categories. For other numerical columns, we removed outliers that were three standard deviations from the mean value.

At last, we remove reviews from anywhere outside of the California boundary. With cleaned business metadata and review data, we merge the data set using the common identifier of businesses. It is worth pointing out that the rating(target value) is unequally distributed, with a rating of 5 taking a higher proportion. We can visualize our ratings by the heat map below.

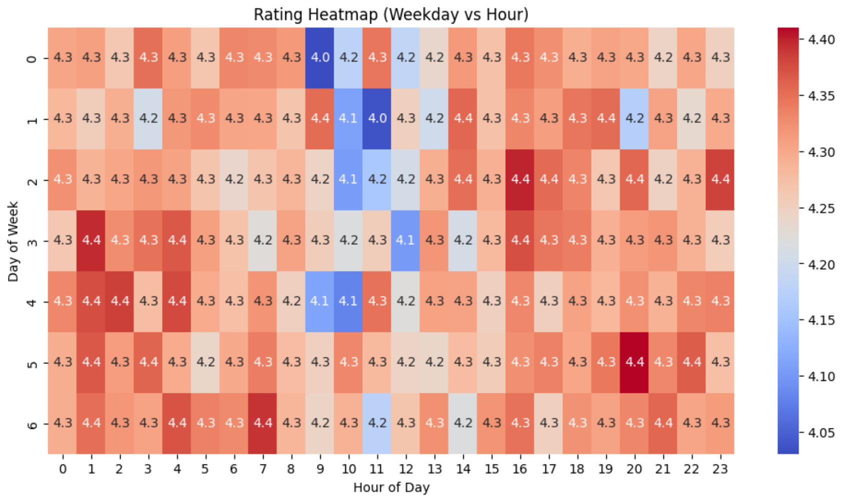


Figure 1: Distribution of rating across week and hours of the day

5 Feature Engineering

We conduct feature engineering with a few assumptions. Our model takes a hybrid approach to recommendation, using a mix of static features such as geographic location, business categories and interaction between user and business, as well as dynamic temporal features like sliding-window rolling average to capture sequential trend for prediction.

5.1 Static features

For locations, we created 20 bins based for both longitude and latitude, resulting in a total of 400 bins for different areas, and one-hot encoding was applied to capture spatial characteristics. These features help represent the spatial distribution of businesses.

The figure below illustrates the distribution of average review ratings and the number of reviews across various locations. Notably, areas with a higher review count generally align with densely populated regions within the state. This suggests a correlation between population clusters and business activity, reflected in the volume of reviews. The visualization highlights both the concentration of reviews and the average sentiment expressed by users in these areas.

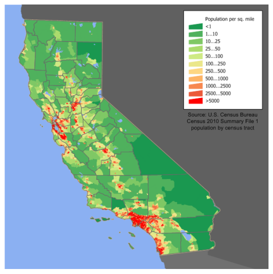


Figure 2: California Population Heatmap (2010)

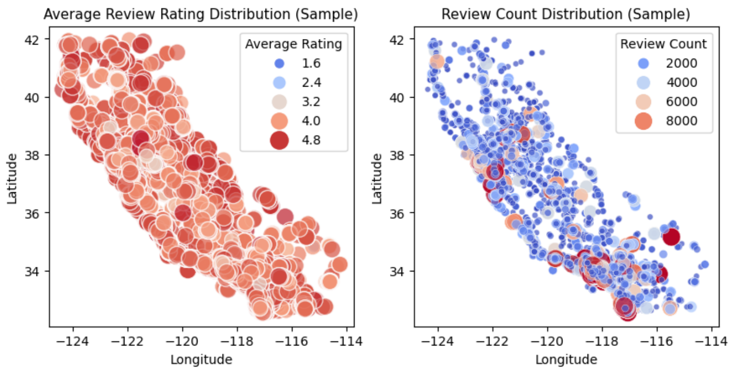
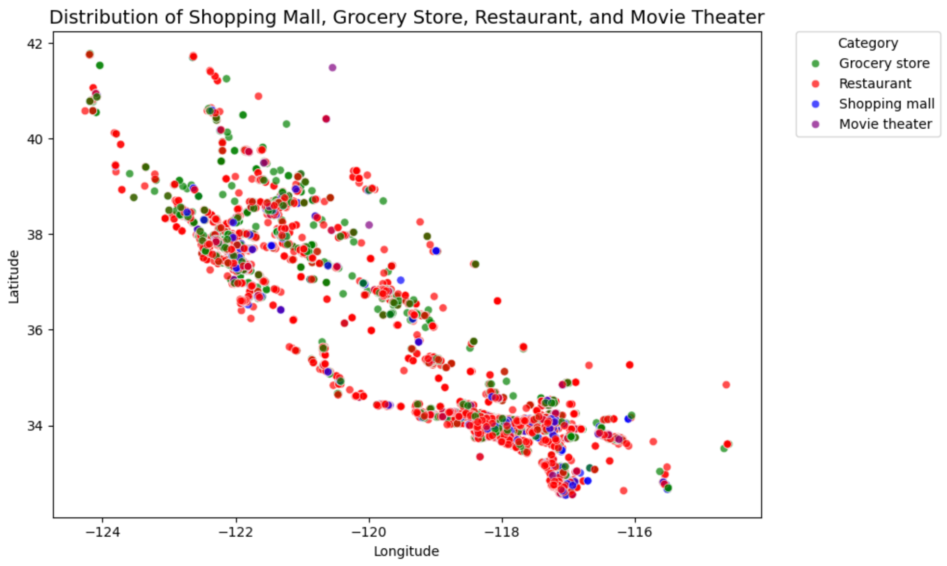


Figure 3: Location Map

Another interesting feature shown by average review rating distribution is that people tend to give high ratings (4.0-4.8) compared to lower. This correlates to our exploration of temporal data, which will be talked about below.

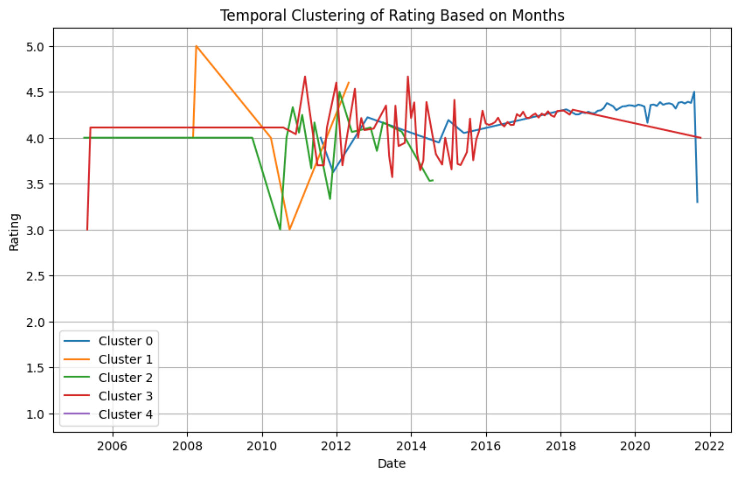
We also observed and picked out some key categories in our data frame (Grocery stores, Restaurants, Shopping malls), and applied one-hot encoding to capture the presence of these categories in each business entry. This allows us to incorporate categorical data into numerical form for downstream analysis. This graph tracks the distribution of those four categories.



5.2 Temporal features

Upon inspection, the dataset contains reviews across a 20 years time period (2005-2021), providing a rich temporal context for analysis. We created a figure capturing a clustering analysis of rating based on months across the year.

By grouping months with similar numerical characteristics, the KMeans algorithm identifies five unique clusters. These clusters highlight periods of similarity in user behavior or business performance over time, which leads us to consider creating time-based features for a temporal model. We present an additional graph showing the review trend for 10 top businesses over time based on our dataset. We



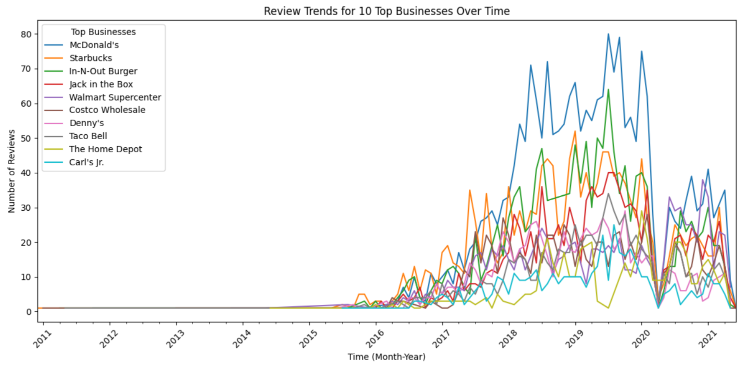


Figure 4: Enter Caption

can see a clear peak in business reviews from 2017 to 2020, and a sharp decrease in 2020 potentially caused by the covid-19 pandemic.

In our preprocessing pipeline, we emphasized standardizing key variables to ensure consistency across the dataset. Two specific standardization processes are noteworthy:

We standardized the review time column by converting raw unix timestamps into a normalized z-score representation. This transformation ensures that temporal data is comparable by preventing outliers from skewing our data, and allowing direct comparison of model coefficients. The normalization involves subtracting the mean and dividing by the standard deviation, with an additional step to calculate standardized user-specific average review times.

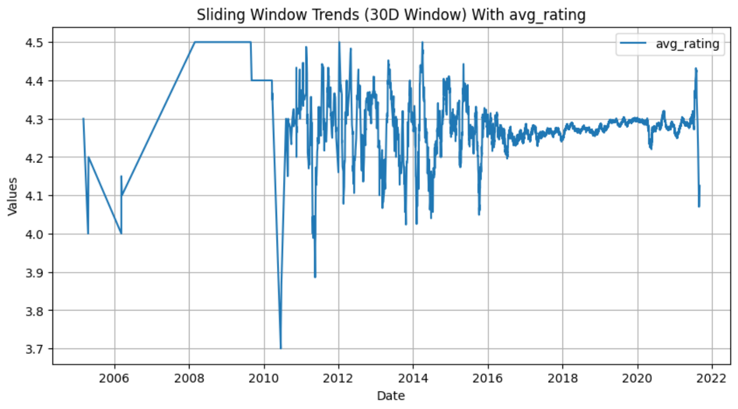
We also standardized the 'hours' column by taking into account the various formats it can take. We developed custom parsing logic to handle diverse time representations and computing the total weekly operating hour for each business, elaborated below.

Besides standardization, we also engineered other temporal features to facilitate usage in our model:

We categorized review timestamps into discrete bins for time-series analysis. We also calculated and normalized each user's average review submission time, which helps capture temporal preferences and habits of reviewers.

We were able to extract two features from the "hours" column: whether the business is open during weekends, which could affect its customer base and review frequency, and the total hours of operation per week that captures availability as a predictor for customer interaction.

An interesting trend to notice is how people's rating seem to "converge" over time. By inspecting the average rating over 30 days period by a sliding window approach, we see people tend to give more diverse ratings during the ten year period from 2006 to 2016, while the range of average ratings



becomes smaller and smaller marching into the early 2020s, which corresponds to the high amount of near 4.0 ratings in the graph of average review by location above.

6 Modeling Dynamics in Time

There are usually two approaches to tackling dynamics in time: one that focuses on using temporal features (timestamps) and one that focuses on looking at the order of things (sequence). We will introduce two common approaches (one from each family) in this section, which are written in more detail in [8].

6.1 Baseline Latent Factor (BLF)

For our baseline evaluation, we are using a plain latent factor model [6] that only models the interaction between user and item through rating interactions (γ_u and γ_i) and bias terms (user bias β_u, item bias β_i, and global bias β_g). We also added regularization on each of the terms mentioned above, which we can frame the whole optimization using MSE with the following.

$$\arg\min_{\beta,\gamma} \sum_{u,i} (\beta_g + \beta_u + \beta_i + \gamma_u \cdot \gamma_i - R_{u,i})^2 + \lambda \left[\sum_u \beta_u^2 + \sum_i \beta_i^2 + \sum_i \|\gamma_i\|_2^2 + \sum_u \|\gamma_u\|_2^2 \right]$$

6.2 Factorized Personalized Markov Chain Variants (FMPC-V)

Factorized Personalized Markov Chains [11] (FPMC) extends from Markov Chain model using first-order Markovian property. It includes both user-item interactions and item-item sequential transitions and represent them as a factorized latent factor model. The core idea of FPMC is to do two things:

1. Item should match user preferences.
2. Next item should be consistent with previous item.

Such model would be robust to capture the short term temporal dependency of the ratings made on users. To achieve this, we can use the scoring function $f(i \mid u, j)$ that represent the probability of seeing item i giving the joint probability of item j and the user u. We can use tensor decomposition to reduce such probabilistic formulation into a latent factro formulation.

$$f(i \mid u, j) = \underbrace{\gamma_{ui} \cdot \gamma_{iu}}_{f(i|u)} + \underbrace{\gamma_{ij} \cdot \gamma_{ji}}_{f(i|j)} + \underbrace{\gamma_{uj} \cdot \gamma_{ju}}_{f(u,j)}$$

Where $f(i \mid u)$ captures the compatibility between the user u and the next item i, which represents a user-item interaction (personalized). $f(i \mid j)$ captures the sequential relationship between the previous item j and the next item i, which represents thes sequential term (Markov chain). $f(u, j)$ captures the relationship between the user u and the previous item j. Notice that this last term is usually neglected as user's compatibility with previous item, already rated, is trivial and doesn't matter. Our variants expands upon this system and incorporated feature components into the model, the feature takes in many format (for specifics reference section 5 for more details), but for demonstration purpose, we can separate them into categorical and numerical features, which we use dense layer to incorporate into the embeddings:

$$f(i \mid u, j, \mathbf{F}) = \underbrace{\gamma_{ui} \cdot \gamma_{iu}}_{\text{user/next item compatibility}} + \underbrace{\gamma_{ij} \cdot \gamma_{ji}}_{\text{next/previous item compatibility}} + \underbrace{\beta_u + \beta_i}_{\text{user and next-item biases}}$$
$$+ \underbrace{\mathbf{w}^\top \mathbf{F}_{\text{cat}}}_{\text{categorical embeddings}} + \underbrace{\mathbf{v}^\top \mathbf{F}_{\text{num}}}_{\text{numerical embeddings}} + \underbrace{b_g}_{\text{global bias}}$$

Traditionally, we would optimize FPMC with Bayesian Personalized Ranking [10]. However, we are making a non-binary categorical prediction (rating). Thus we used plain MSE (mean square error) for the optimization process, which can be framed as the following:

$$\arg \min_{\gamma, \beta, \mathbf{w}, \mathbf{v}} \sum_{u,i,j} \left(\gamma_{ui} \cdot \gamma_{iu} + \gamma_{ij} \cdot \gamma_{ji} + \beta_u + \beta_i + \mathbf{w}^\top \mathbf{F} + \mathbf{v}^\top \mathbf{F} + \beta_g - R_{u,i} \right)^2$$
$$+ \lambda \left(\sum_u \beta_u^2 + \sum_i \beta_i^2 + \sum_u \|\gamma_u\|_2^2 + \sum_i \|\gamma_i\|_2^2 + \|\mathbf{w}\|_2^2 + \|\mathbf{v}\|_2^2 \right)$$

Notice that the features are directly embedded using embedding methods with neural networks.

6.3 Temporal Dynamic Latent Factor Model With Neural Correlative Variants (TDLF-V)

The second approach that we use in this system extends from the model that won the Netflix price [5]. We are essentially trying to incorporate temporal features into the model. Notice that this is a very over-engineering approach, and the final model itself may be very specific to the dataset and domain that it serves. This model extends from a traditional latent factor model and incorporated temporal features, specifically, it includes bias terms that is specific to items $\beta_i(t)$ and assumes that the latent factor with k component for user changes over time $\gamma_{u,k}(t)$. The bias term includes both binning and periodic changes:

$$\beta_i(t) = \beta_i + \beta_{i,\text{bin}}(t) + \beta_{i,\text{period}}(t)$$

In addition, we are using a neural correlative approach [3], so the latent factor is passed through a neural network like the following (notice that the features are also shoved along into the neural network by $\mathbf{w}^\top \mathbf{F}$):

$$f(\gamma_u, \gamma_i) = \text{NN}([\gamma_u, \gamma_i])$$

The whole prediction is farmed to be:

$$\hat{r}_{u,i,t,\mathbf{F}} = \underbrace{\mu}_{\text{Global bias}} + \underbrace{\beta_i}_{\text{Static item bias}} + \underbrace{\beta_i(t)}_{\text{Dynamic item bias}} + \underbrace{\beta_u}_{\text{Static user bias}} + \underbrace{f(\gamma_{u,k}(t), \gamma_{i,k})}_{\text{Interaction score}} + \underbrace{\mathbf{w}^\top \mathbf{F}_{\text{item}}}_{\text{Item-specific feature effect}}$$

The optimization process can be framed as the bellow with regularization on each component (still with MSE loss).

$$\arg \min_{\alpha, \beta, \gamma, \mathbf{W}} \sum_{u,i} \left(\mu + \beta_i + \beta_i(t) + \beta_u + f(\gamma_{u,k}(t), \gamma_{i,k}) + \mathbf{w}^\top \mathbf{F}_{\text{item}} - R_{u,i} \right)^2$$
$$+ \lambda \left(\sum_u \beta_u^2 + \sum_i \beta_i^2 + \sum_u \|\gamma_u\|_2^2 + \sum_i \|\gamma_i\|_2^2 + \sum \|\mathbf{w}\|_2^2 \right)$$

We have strong confidence in this model since this model would be effective in capturing the long-term relationships between businesses and customers. During the implementation process, we realized that the deviation function caused our model to perform poorly. We realize that the deviation was made depending on the dataset itself. In the end, we decide to remove the deviation term.

We also fine-tune models to get the best performance. All of the parameters can be observed from Table 2.

7 Downstream Application

Our downstream application provides business owners with actionable insights by identifying and recommending users likely to interact positively with their services. This is the core intention of this recommendation system. The application will take in a business ID (gmap_id), which is the focal point for generating recommendations. The associated business categories of the business ID are extracted. Customers(reviewers_id) that are used to interact with the extracted categories are identified from the dataset, forming a candidate pool of potential customers. Then, the trained model will predict the ratings of the given business ID with all candidate customers (reviewers_id). The model will use the business' metadata and users' latest review timestamps to predict the rating. Candidate users will be ranked based on their likelihood of giving high ratings. By default, the application will recommend the top 20 users to business owners.

8 Results

The performance of both advanced and baseline model is evaluate quantitatively and qualitatively. Quantitatively, we deploy several metrics measuring the performance of the model. Qualitatively, we run our model on the downstream applications.

8.1 Quantitative Performance

We use four metrics to comprehensively evaluate model performance: Mean Squared Error (MSE), Root Mean Squared Error (RMSE), $R\hat{2}$ (coefficient of determination), Mean Absolute Scaled Error (MASE), and Accuracy. The results are summarized in Table 1. Among the models, the basic Latent Factor model achieves the best performance with an MSE of 0.98, indicating that its predictions are closest, on average, to the actual ratings. It also has the highest $R\hat{2}$, suggesting it explains the most variance in the dependent variable. In contrast, the FPMC Variants model yields an MSE of 1.13 and a negative $R\hat{2}$, indicating that its predictions are worse than simply using the mean of the dependent variable. This suggests that the advanced FPMC model underperforms relative to the baseline model. However, the Temporal Dynamic model shows notable improvement, achieving a competitive MSE of 0.99 and the highest accuracy of 0.42, demonstrating its potential for better classification performance. We highly emphasize the accuracy metrics because positive prediction is critical to our final recommendation.

8.2 Qualitative Performance

To evaluate the model's performance, we randomly selected five business IDs (gmap_id) and analyzed the users recommended by each model. Interestingly, all three models produced highly similar sets of recommended users, indicating consistent prediction patterns across different approaches. Upon closer examination, the recommendations were contextually relevant and demonstrated the model's ability to capture nuanced user behaviors.

For instance, when the target business belonged to the category "Sushi restaurant," the majority of recommended users had a history of frequent interactions with categories such as fast food and breakfast restaurants. This suggests the model effectively identifies users with dining preferences that overlap or align with the target business type. Furthermore, these users consistently provided high ratings, averaging 5, to the restaurants they previously interacted with. This trend highlights the model's ability to prioritize users who not only exhibit similar preferences but also demonstrate a likelihood of giving favorable feedback, making them highly valuable for business owners.

Model	MSE	RMSE	R2	MASE	ACC
Latent Factor Model	0.968255	0.983999	0.092328	0.817679	0.382959
FPMC Variants	1.286426	1.134207	-0.205936	1.064545	0.238363
Temporal Dynamic Variants	0.980515	0.990210	0.080835	0.809500	0.427182

Table 1: Performance Metrics for Different Models

Model	**Patience**	**Min Delta**	**Learning Rate Schedule**	**Embedding Dim.**
Basic Latent Factor Model	5	0.001	0.001	20
Temporal Latent Factor Model	5	0.001	0.00066501	90
Factorized Personalized Markov Chain	5	0.001	0.00022001	20

Model	**Dense Units**	**L2 Regularizer**	**Time Bins**
Basic Latent Factor Model	NAN	0.001	NAN
Temporal Latent Factor Model	128	0.00121	30
Factorized Personalized Markov Chain	20	0.0066295	NAN

Table 2: Hyperparameters for Different Models (Split Representation)

9 Conclusion & Discussion

9.1 future improvement

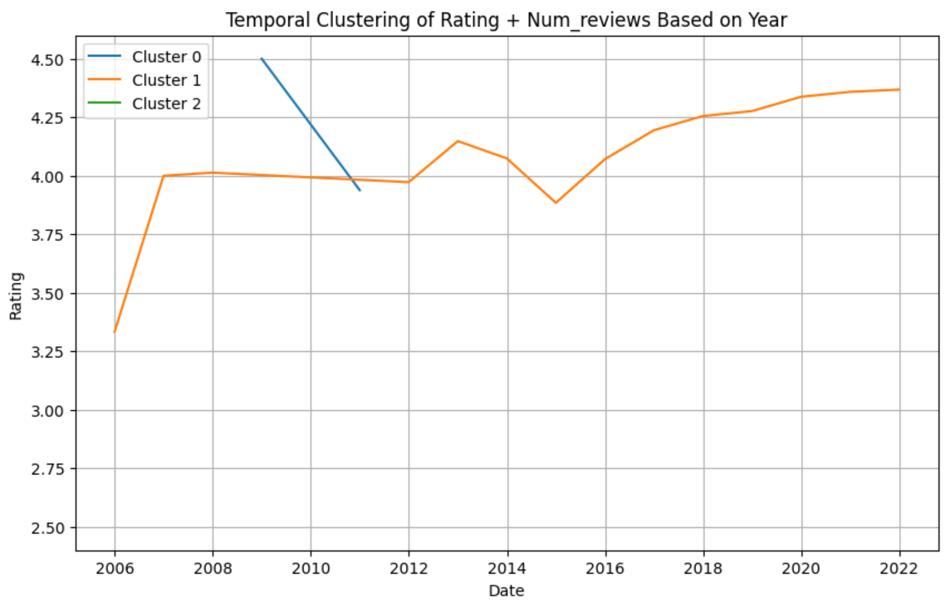
One limitation of the project is its reliance on a relatively small dataset. While there are 666,324,103 reviews in total, we only train the data set on 1% of the data. This may not fully capture the diversity of the full dataset. Future work should aim to incorporate training on the entire dataset. In addition, we only focus on the training model with the California data set. The model might be more robust and powerful in predicting the creation, considering consumers' interactions from one state to another. Moreover, standard methods are lacking in the validation of downstream applications. In this project, we evaluate the downstream result based on common sense and manual inspection. Thus, future work can be done on developing methods to validate the downstream application result and the reason for the similar performance of the three models. Both of the advanced models did not perform well in terms of R-square and RMSE. We hypothesize the root cause to be the quality of data. The limited data resources cause the advanced model to be bad in terms of MSE. For future work, more data and features can be created to achieve a better performance.

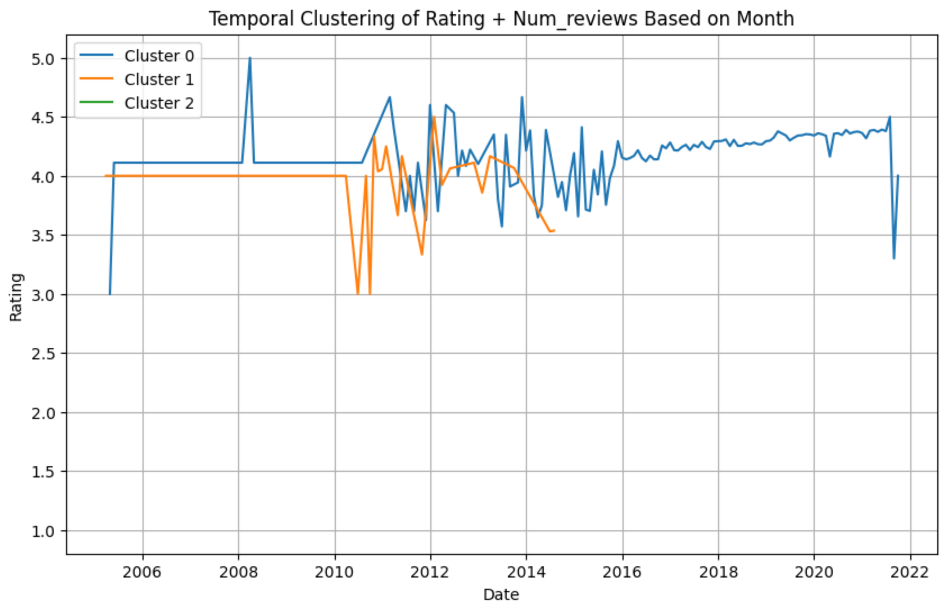
9.2 conclusion

This study developed a hybrid recommender system leveraging collaborative and content-based filtering to help business owners identify potential customers. The model effectively analyzed sequential dynamics using Google Local Review data, producing robust and contextually relevant recommendations. While the approach shows promise, future work should focus on scaling to larger datasets, incorporating cross-state data, and standardizing validation methods to enhance applicability and impact.

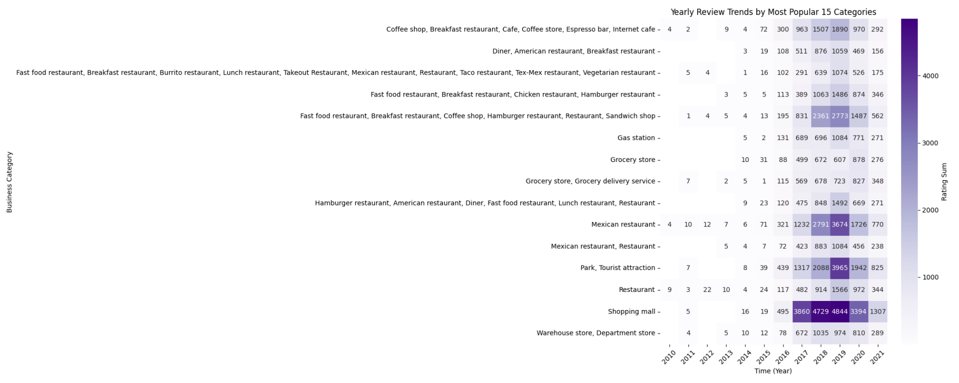
10 Appendix

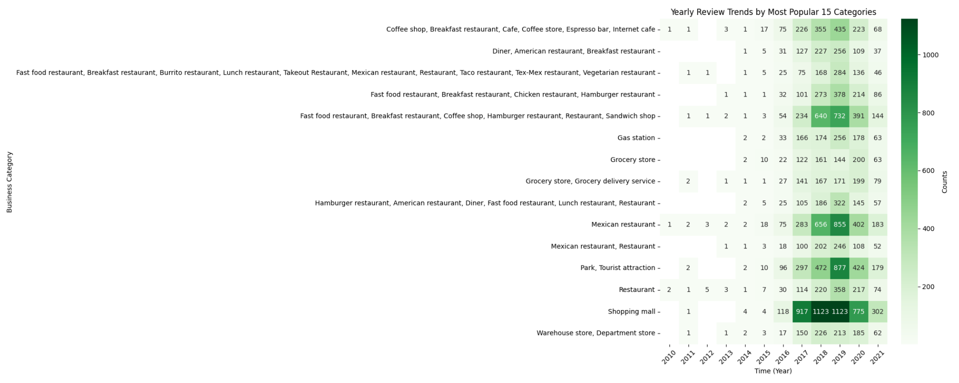
10.1 Additional temporal clustering

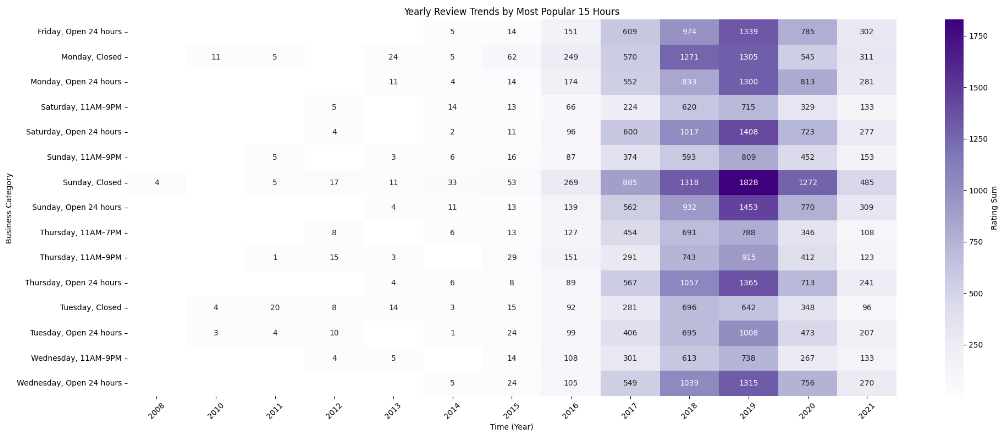


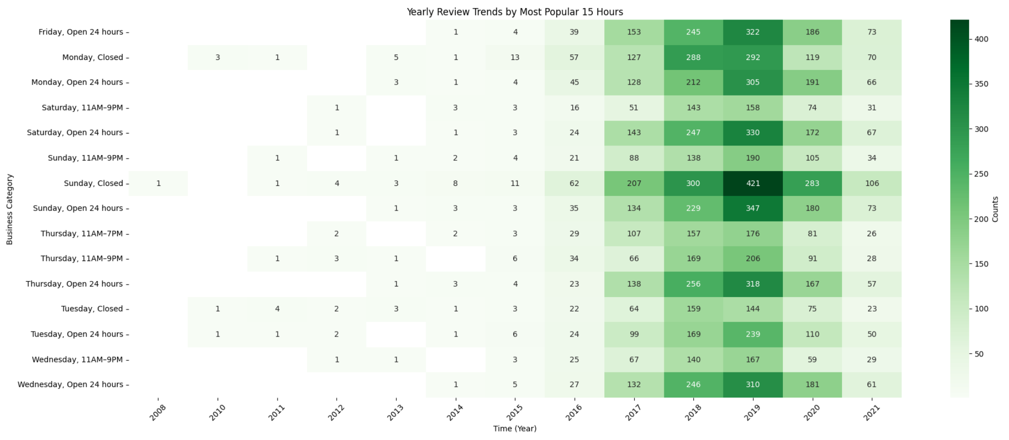


10.2 Additional heat maps









10.3 Additional rolling average

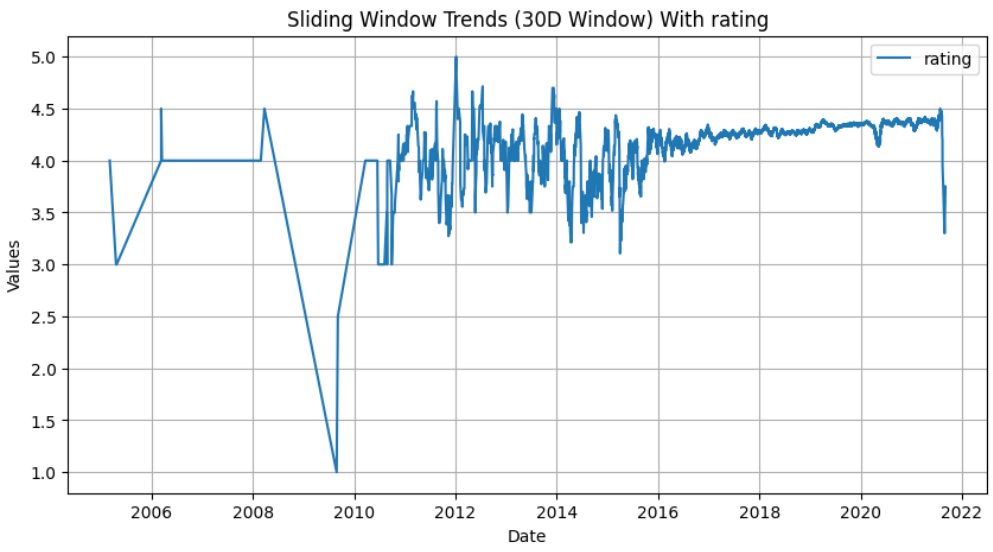
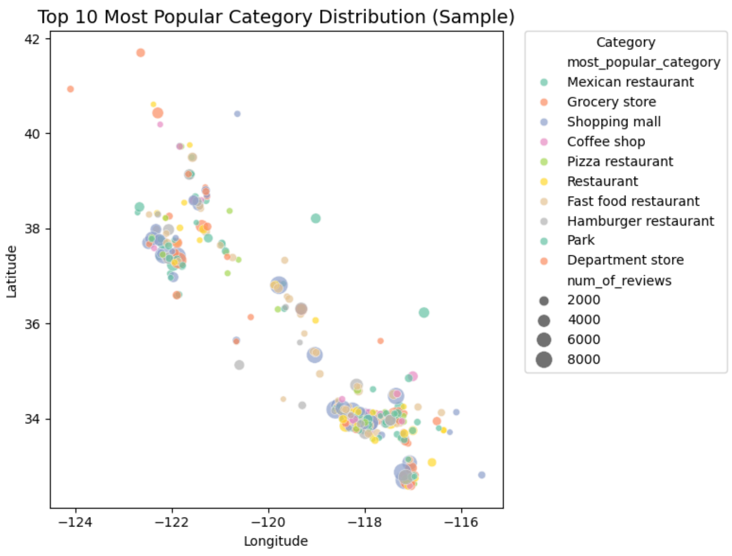
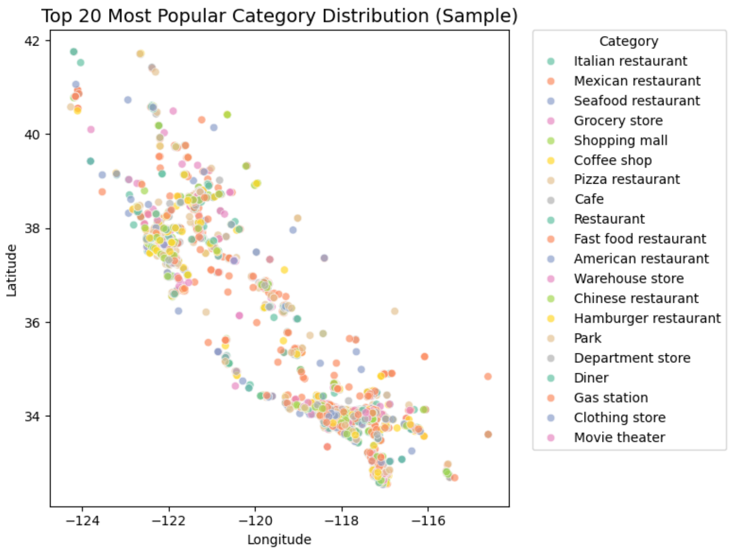


Figure 5: Enter Caption

10.4 Additional geographical visualization





References

[1] Yelp dataset, [n. d.]. `https://www.yelp.com/dataset`.

[2] Junyi Chen, Lu Chi, Bingyue Peng, and Zehuan Yuan. Hllm: Enhancing sequential recommendations via hierarchical large language models for item and user modeling. *arXiv preprint arXiv:2409.12740*, 2024.

[3] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web (WWW)*, pages 173–182. ACM, 2017.

[4] Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*, 2024.

[5] Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 447–456. ACM, 2009.

[6] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[7] Jiacheng Li, Jingbo Shang, and Julian McAuley. Uctopic: Unsupervised contrastive learning for phrase representations and topic mining. *arXiv preprint arXiv:2202.13469*, 2022. Accepted as ACL 2022 main conference paper.

[8] Julian McAuley. *Personalized Machine Learning*. Cambridge University Press, 2022.

[9] Jianmo Ni, Larry Muhlstein, and Julian McAuley. Modeling heart rate and activity data for personalized fitness recommendation. *ACM*, 2019. Proceedings of the World Wide Web Conference (WWW '19).

[10] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. *AUAI Press*, 2009. Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI2009).

[11] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. *Proceedings of the 19th International Conference on World Wide Web (WWW)*, pages 811–820, 2010.

[12] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. *arXiv preprint arXiv:1904.06690*, 2019. To appear in CIKM 2019.

[13] An Yan, Zhankui He, Jiacheng Li, Tianyang Zhang, and Julian McAuley. Personalized showcases: Generating multi-modal explanations for recommendations. *arXiv preprint arXiv:2207.00422*, 2023. Accepted to SIGIR-23, with additional dataset details.