PERSONALIZED PROPERTY MATCHING USING RAG FOR SPECIAL NEEDS

Data Analysis and Visualization 094295

Authors:

Itay Bachar - 206948218

Sewar Hino - 323016485

Adir Toledano - 313535379

Technion

Spring 2023-2024

Abstract:

In This project we develop a Retrieval-Augmented Generation (RAG) system to improve property recommendations for Airbnb users by leveraging synthetic question-answer (QA) pairs for evaluation. Given the lack of ground truth data in the Airbnb dataset, we generated synthetic QA pairs based on Airbnb property descriptions. This dataset enhances the reliability of the RAG system in retrieving relevant properties. Our system, demonstrated through a prototype application, enables users to query property features like accessibility, infrastructure, and amenities, providing tailored property recommendations. The results confirm the RAG system's efficacy in returning relevant listings, showing promise for application across similar recommendation domains.

Introduction:

In recent years, personalized recommendation systems have become essential for platforms like Airbnb, as users increasingly seek properties that meet unique, specific needs. Beyond standard filters like location, price, and number of rooms, users frequently prefer features tied to accessibility, amenities, or infrastructure—such as wheelchair access, elevators, or essential services like internet and kitchen facilities. Traditional recommendation systems struggle to address these nuanced user preferences, often resulting in generic listings that fail to capture detailed requirements.

This project addresses this gap by developing a Retrieval-Augmented Generation (RAG) system that combines information retrieval with generative capabilities to deliver property recommendations tailored to the needs of users with special requirements. Unlike standalone language models (LLMs), which generate responses based on general data, the RAG approach ensures that responses are grounded in relevant property data, aligning recommendations with user-defined details, such as specific amenities or accessibility features.

To explore the effectiveness of different indexing methods in RAG pipelines, we implemented and compared two models: one using FAISS and the other using Pinecone. Each model follows the core RAG stages: (1) embedding and indexing property descriptions, (2) retrieving relevant listings based on user queries, (3) using a language model to generate responses incorporating these listings, and (4) presenting recommendations through an interactive demo application. For evaluation, we generated synthetic question-answer pairs using an LLM, which serve as ground truth data to assess the models' retrieval and generative performance.

Ultimately, this project aims to improve user satisfaction by delivering tailored recommendations. By comparing the FAISS and Pinecone-based RAG models against a baseline LLM model, we gain insights into how different indexing methods impact

recommendation quality, establishing a scalable approach applicable across platforms requiring personalized, context-aware recommendations.

Methodology:

Dataset and Preprocessing:

The Airbnb dataset included property descriptions, amenities, and numerical attributes like price and guest capacity. Data preprocessing involved removing irrelevant columns, handling missing values with averages or placeholders, and standardizing data types. Exploratory Data Analysis (EDA) highlighted key patterns (e.g., price distribution, number_of_reviews), guiding subsequent modeling steps (see Appendix for visuals).

Pipeline Overview:

Our RAG pipeline, implemented using FAISS and Pinecone, provided personalized property recommendations through the following steps:

1. **Data Embedding and Indexing**: Property descriptions were embedded using `all-MiniLM-L6-v2` and indexed in both FAISS and Pinecone to compare retrieval performance.

2. **Query Retrieval:** User queries were embedded and matched against indexed data to retrieve the top 5 relevant properties.

3. **Response Generation:** The `distilgpt2` model generated contextually relevant responses from the retrieved descriptions.

4. **Demo Application:** A demo application with a FastAPI backend and Streamlit front-end allowed users to:

- Submit queries and preferences (e.g., “wheelchair-accessible property with an elevator”).
- Compare FAISS and Pinecone results interactively.
- Access ranked recommendations through a user-friendly interface.

QA Dataset Generation

To address the lack of labeled data, we created a synthetic QA dataset using Cohere’s LLM:

1. **Sample Selection**: 500 Airbnb listings were selected to represent diverse features.

2. **Attribute-Based Prompting**: Prompts targeted accessibility, infrastructure, and amenities, yielding realistic QA pairs.

3. **Validation**: QA pairs were refined to ensure consistency in format (e.g., `Q: [Question] A: [Answer]`).

The resulting dataset (`synthetic_qa_pairs_expanded.csv`) served as ground truth for evaluating RAG accuracy.

RAG System Development and Evaluation:

1. **Embedding and Retrieval**: Descriptions were embedded using `all-MiniLM-L6-v2`, indexed in FAISS and Pinecone, and used for top-K retrieval.

2. **Generative Responses**: The `distilgpt2` model generated concise recommendations from retrieved data.

3. **QA Integration**: Synthetic QA pairs improved context and response alignment.

4. **Demo Integration**: The FastAPI backend processed queries and returned ranked results, while the Streamlit front-end enabled interactive exploration of recommendations.

5. **Evaluation Metrics**:

- BLEU, ROUGE, BERTScore : assessed answer quality against the QA pairs.

- Recall at K and MRR evaluated retrieval performance, highlighting limitations (e.g., 0.00 scores under specific conditions).

This pipeline demonstrated comparable performance across FAISS and Pinecone, with the demo showcasing its practical application for personalized, user-specific recommendations.

Experiments:

Experimental Setup:

Our experiments evaluated the performance of two RAG implementations (FAISS and Pinecone) in generating property recommendations tailored to specific user needs. We focused on retrieval accuracy, response relevance, and system limitations in handling complex queries. Additionally, a demo application was used to simulate real-world interactions and assess the system's usability and performance.

1. Synthetic Queries: Synthetic queries were created to test the models' ability to handle detailed requirements. Examples included "wheelchair-accessible property with an elevator" and "quiet neighborhood with fast internet access." These queries covered diverse topics such as accessibility, amenities, and infrastructure.

2. Comparison of FAISS and Pinecone: Queries were processed through both FAISS and Pinecone, with retrieved results summarized using `distilgpt2`. The demo application enabled an interactive comparison, providing insights into retrieval efficiency and response generation quality for similar scenarios.

3. Evaluation Metrics:

- BLEU, ROUGE, and BERTScore: These metrics assessed the generative quality of responses. BLEU and ROUGE scores were generally low due to differences in phrasing between generated and reference answers.

- Precision, Recall, and F1-Score: Both systems performed well in precision but showed lower recall, reflecting limitations in retrieving all relevant items.

- Recall at K and MRR: Both FAISS and Pinecone had low Recall at 5 and MRR scores, indicating difficulties in ranking highly specific listings within the top-K results.

4. Synthetic QA Dataset as Ground Truth: A synthetic QA dataset generated using Cohere's LLM served as ground truth for evaluation. This dataset captured common user questions and answers, providing a foundation for comparing generated responses. However, traditional metrics like BLEU and ROUGE struggled to fully capture recommendation quality due to phrasing differences.

5. User Interaction Simulations via Demo Application:

A FastAPI backend and Streamlit front-end powered the demo application. Users provided preferences and queries, and the system returned ranked recommendations. The app revealed the system's strengths in handling standard queries but highlighted challenges with rare or unique features. Both FAISS and Pinecone implementations were tested in the demo, allowing side-by-side comparisons of retrieval and generation quality in a practical context.

Summary of Results:

Experiments demonstrated comparable performance across FAISS and Pinecone-based RAG systems. Key findings include:

- Comparable Retrieval Quality: Both implementations achieved similar scores across BLEU, ROUGE, and BERTScore, underscoring their effectiveness in retrieving relevant descriptions. Metrics included:

- LLM Model Performance: BLEU: 0.04, ROUGE-1: 0.21, BERTScore F1: 0.85

- RAG Model Performance: BLEU: 0.01, ROUGE-1: 0.08, BERTScore F1: 0.81

- Recall and Ranking Challenges: Both systems exhibited low Recall at K and MRR scores (0.00 in some cases), suggesting potential gaps in ranking highly relevant items among the top results. Enhancing retrieval strategies or dataset coverage could mitigate these issues.

- Consistent Generation Quality: Generated responses were coherent and relevant across both implementations, suggesting that differences in indexing did not affect the language model's output.

- Scalability Considerations: FAISS excelled in local environments, while Pinecone demonstrated scalability advantages for larger datasets, critical for real-time applications.

Comparison of FAISS and Pinecone:

The minimal performance differences between FAISS and Pinecone likely reflect redundancy in the dataset and similar embedding handling. Both systems use similarity-based retrieval, yielding comparable outcomes when applied to embeddings from the same model.

By integrating the demo application into our experiments, we highlighted the RAG system's practical usability and its ability to adapt to diverse user needs, offering a comprehensive evaluation framework.

Discussion:

This project highlights the potential of a Retrieval-Augmented Generation (RAG) system to enhance personalized property recommendations in the Airbnb domain. By using both FAISS and Pinecone implementations with a synthetic QA dataset as ground truth, we addressed some limitations of traditional recommendation models in meeting specific user needs, such as accessibility and amenities. The addition of Cohere-based QA generation allowed us to construct relevant ground truth data, while both FAISS and Pinecone proved effective for large-scale data retrieval.

Our key insights demonstrate that while the RAG system delivered high precision in retrieving relevant properties, recall rates were limited, particularly for queries requiring niche or rare features. This may stem from challenges in ranking specificity within the top-K results. Additionally, generative quality metrics such as BLEU and ROUGE did not fully capture the contextual relevance of recommendations, underestimating the system's true performance when assessed using BERTScore.

A unique aspect of this project was the development of an interactive demo application, where users could input specific property requirements and receive ranked, tailored recommendations. This interface demonstrates the system's practical applications, offering an intuitive experience for users with diverse needs.

Key Findings and Future Improvements

- Synthetic QA Generation: The QA dataset provided a useful evaluation framework, but limitations in the volume of generated pairs, due to API constraints, impacted system coverage.

Future work could benefit from exploring self-hosted LLMs to increase scalability and control costs.

- Enhanced Relevance and Ranking: The close performance results between FAISS and Pinecone suggest that indexing choice may have a limited effect on retrieval relevance. Future improvements could focus on re-ranking models that better capture niche preferences and on tuning the retrieval models with domain-specific embeddings.

- Expansion of QA Coverage: Extending the synthetic QA dataset to encompass a broader array of user scenarios and property features could improve system robustness. This enhancement could increase response accuracy for diverse and nuanced queries, making the RAG system more adaptable across different recommendation-based platforms.

Overall, the RAG system effectively delivered precise, contextually relevant responses, though with room for improvement in recall and niche feature recognition. These refinements would allow the system to better support complex, user-specific queries in the hospitality and other recommendation-intensive domains.

Appendix:

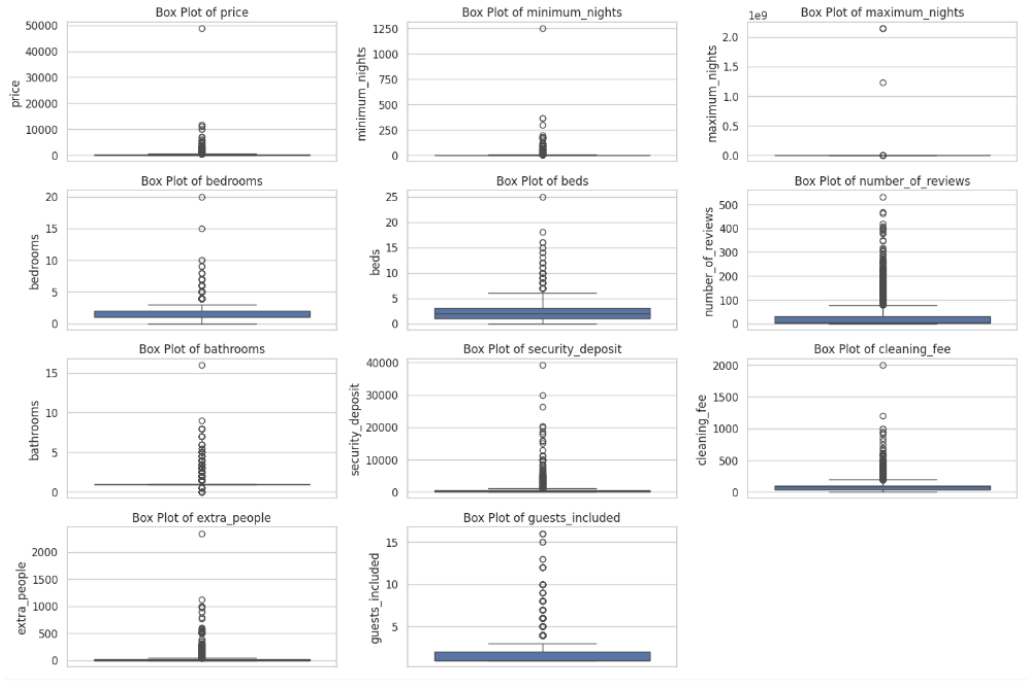
1. GitHub Repository:

https://github.com/ItayBachar1/DataAnalysisLab.git

2. Visualizations:

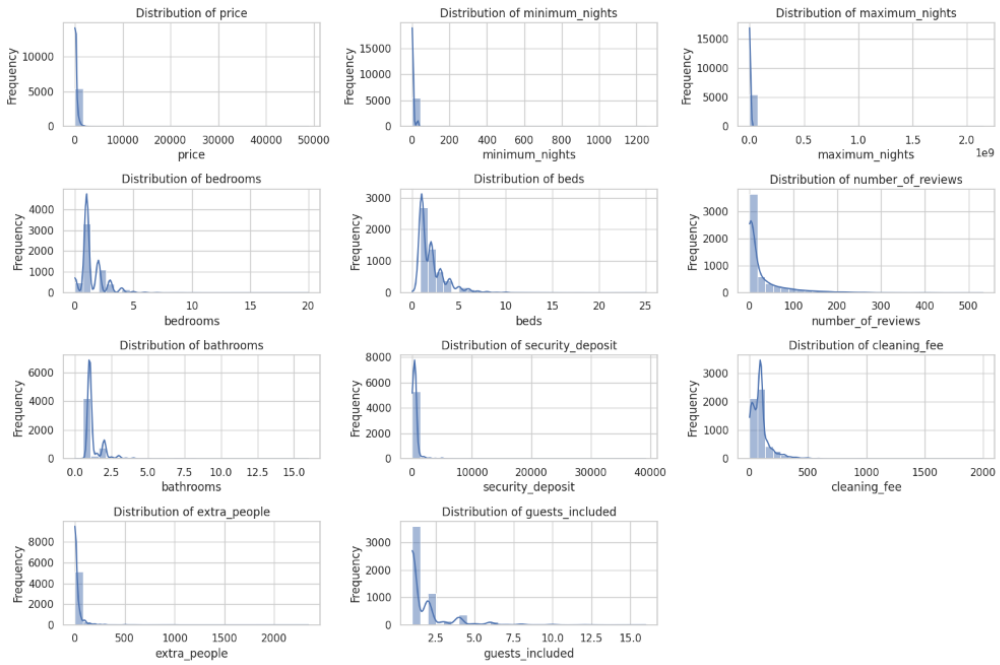
Description: This scatter plot shows the relationship between the number of reviews and the price per night for Airbnb listings. Each point represents a listing, plotted by its number of reviews and price.

Insight: The scatter plot helps determine if there is a correlation between a listing's popularity (based on review count) and its pricing. For example, popular listings with low prices could suggest high demand due to affordability.



Description: This box plot displays the variation in prices across different property types (e.g., apartment, villa, condo). Each box represents the interquartile range of prices within a property type, with whiskers indicating variability outside the upper and lower quartiles.

Insight: By showing the distribution of prices within each property type, this plot enables a direct comparison across categories, revealing which types tend to be priced higher or lower, and highlighting any outliers.



Description: The distributions illustrate various features of Airbnb listings, including price, minimum and maximum stay nights, and characteristics like the number of bedrooms, beds, and bathrooms. Most distributions are right-skewed, with the majority of listings having low values for price, minimum nights, and fees, as well as fewer bedrooms and bathrooms.

Insight: The dataset predominantly represents affordable, smaller accommodations suitable for short-term stays and small groups. This suggests that Airbnb caters largely to budget-conscious travelers or small families. Features like low cleaning fees, minimal security deposits, and flexible occupancy options further enhance accessibility and affordability, which could be important for recommendations and search filtering.

3. References:

- Diamant, N. (n.d.). *RAG Techniques* [GitHub repository]. GitHub. Retrieved from https://github.com/NirDiamant/RAG_Techniques

- Antoniou, A., Storkey, A., & Edwards, H. (2017). Data augmentation generative adversarial networks. arXiv preprint arXiv:1711.04340. Retrieved from https://arxiv.org/abs/1711.04340

- Johnson, J., Douze, M., & Jégou, H. (2017). Billion-scale similarity search with GPUs. IEEE Transactions on Big Data, 7(3), 535–547. Retrieved from https://arxiv.org/abs/1702.08734

- Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning-based recommender system: A survey and new perspectives. ACM Computing Surveys (CSUR), 52(1), 1–38. Retrieved from https://doi.org/10.1145/3285029

- Gunawardana, A., & Shani, G. (2009). *A survey of accuracy evaluation metrics of recommendation tasks*. Journal of Machine Learning Research, 10, 2935–2962. Retrieved from https://www.jmlr.org/papers/volume10/gunawardana09a/gunawardana09a.pdf