TripAdvisor Recommendation Challenge

Beating BM25

Machine Learning for NLP - Project 1

Alessia SARRITZU
Alberto MARTINELLI

January 15, 2025

Contents

1 Introduction

The TripAdvisor Recommendation Challenge aims to develop a robust recommendation system that relies solely on user reviews to identify similar places based on a given query. The primary objective is to design a model capable of outperforming the BM25 baseline, a well-known algorithm for ranking documents based on their relevance to a query.

BM25 remains one of the most effective unsupervised techniques across various datasets. It builds upon the TF-IDF method by introducing additional elements that improve its performance:

- **A local component**: measures the term frequency within the document (TF).
- **A global component**: evaluates the term's scarcity within the entire corpus (IDF).
- **A preference for shorter documents**: gives higher relevance to shorter documents when two have identical TF-IDF scores.

The project's core challenge is to develop an innovative recommendation system that relies only on user reviews. Given a review or a set of reviews about a specific location, the system should recommend the most similar place in the dataset.

The system's effectiveness will be evaluated using the Mean Squared Error (MSE) between the ratings of the query location and the recommended location, averaged across multiple aspects such as service, cleanliness, overall, value, location, sleep quality, and rooms.

To achieve this, the project is divided into two main components:

- **BM25 Baseline**: Implementation of the BM25 algorithm using the *Rank-BM25* library to serve as the baseline for comparison.
- **Enhanced Recommendation Model**: Development of a novel model or approach to outperform BM25 in terms of MSE, integrating techniques such as advanced preprocessing, tokenization, and vector similarity calculations.

The dataset for this challenge consists of TripAdvisor hotel reviews. Only reviews with ratings strictly covering the predefined aspects are included. Reviews for the same location are concatenated to compute average ratings, which are used exclusively for evaluation purposes.

This report details the methodology, approaches, and results achieved. Additionally, it discusses challenges encountered and solutions implemented to enhance the recommendation system's performance. The goal is to develop an effective, unsupervised recommendation system that achieves a lower MSE than BM25, demonstrating its superiority in capturing review-based similarities between locations.

1.1 Colab Notebook

The Colab notebook used for this project can be accessed via the following link:
NLP - Project 1.

2 Data Preparation

The first step in the project involves preparing the dataset to ensure it meets the required conditions for effective model evaluation and recommendation. This is achieved through the `dataframe_preparation` function, which filters, processes, and structures the input data. The steps involved in this function are outlined below.

2.1 Data Filtering

The dataset is initially filtered to retain only reviews with ratings strictly covering the predefined aspects: *service*, *cleanliness*, *overall*, *value*, *location*, *sleep quality*, and *rooms*. This ensures that all reviews have consistent rating dimensions, facilitating accurate comparison between places. The filtered dataset is then reset with a new index.

2.2 Sampling for Model Testing

From the filtered dataset, a sample of 100 reviews is randomly selected to be used for testing the recommendation system. This sample serves as the basis for evaluating the performance of both the BM25 and enhanced models.

2.3 Concatenation of Reviews

Reviews corresponding to the same place (`offering_id`) are concatenated to form a single textual representation of the location. The ratings for each location are calculated as the mean of all corresponding reviews, ensuring a consistent rating structure. The concatenated reviews and averaged ratings form the final processed dataset.

- The final dataframe `final_df` contains the following fields:
  - The average ratings for each aspect: *service*, *cleanliness*, *overall*, *value*, *location*, *sleep quality*, and *rooms*.
  - A concatenated text representation of all reviews for the location.

2.4 Outputs of Data Preparation

The `dataframe_preparation` function returns the following two outputs:

1. `sample_df`: A sample of 100 reviews to use for testing and evaluation.
2. `final_df`: The processed dataset containing average ratings and concatenated reviews for each unique place.

The prepared data ensures that the recommendation system operates on a clean and structured dataset, allowing accurate evaluation using Mean Squared Error (MSE) as the metric.

3 Data Pre-Processing

The data pre-processing phase is a critical step in preparing the reviews for effective use in the recommendation models. This step focuses on transforming raw text data into a clean, standardized format suitable for token-based similarity measures such as BM25 and TF-IDF. Given the computational cost of preprocessing large datasets, the results are saved to a file, enabling faster execution in subsequent runs.

3.1 Pre-Processed File Check

To avoid redundant computations, the script first checks whether a preprocessed version of the dataset exists in the file `final_df_preprocessed.csv`. If the file is found, the preprocessed data is directly loaded into memory. This ensures that the lengthy preprocessing step is skipped, saving significant execution time.

3.2 Applying Pre-Processing

If the preprocessed file is not available, the raw dataset undergoes the following text preprocessing steps using the `preprocess_text` function:

- **Tokenization**: Splits the text into individual words.
- **Lemmatization**: Reduces words to their base forms (e.g., *running* to *run*).
- **Stop Word Removal**: Removes common words such as *the*, *and*, or *is*, which do not contribute significantly to the meaning of the review.
- **Lowercasing**: Converts all text to lowercase to ensure uniformity.

After preprocessing, the cleaned text data is saved to the file `final_df_preprocessed.csv`.

4 BM25 Model

The BM25 algorithm was implemented as the baseline model for the TripAdvisor Recommendation Challenge, designed to recommend the most similar place based on user reviews. This section elaborates on the implementation, workflow, and evaluation methodology adopted for the BM25 model.

4.1 Implementation of BM25

The BM25 algorithm operates by scoring candidate places based on their textual similarity to a query review. The implementation process was systematically designed to ensure accuracy and computational efficiency, as described below.

Initially, the function `apply_bm25` ensures that the query place itself is not recommended. To achieve this, the query location (`query_id`) is excluded from the dataset of candidate places, producing a filtered dataset, `documents_df`, that contains only valid recommendations. This step eliminates the possibility of trivial recommendations and ensures the system's robustness.

Each review in the filtered dataset is then tokenized into individual words. Tokenization is a crucial preprocessing step that enables the BM25 model to operate on granular textual units, laying the groundwork for computing relevance scores. The tokenized documents are passed to the `BM25Okapi` implementation from the `Rank-BM25` library, which initializes the BM25 model and processes the textual data. The model computes a relevance score for each candidate place by considering factors such as term frequency, inverse document frequency, and document length normalization.

Following the computation of relevance scores, the candidate place with the highest score is selected as the most relevant recommendation. The corresponding ratings for this place are then retrieved for further evaluation. To assess the recommendation's quality, the Mean Squared Error (MSE) between the ratings of the query place and the selected place is calculated. This metric serves as the baseline for evaluating BM25's performance against more advanced models.

4.2 Application Workflow

The BM25 algorithm was applied to all queries in the sample dataset (`sample_df`), adhering to a structured workflow. The process begins with preprocessing, where the reviews are tokenized, lemmatized, and stripped of stop words to ensure consistency and improve the relevance scoring process. This standardization ensures that noisy or irrelevant textual elements do not skew the results. Furthermore, this allows the comparison of BM25 performances against those of the Enhanced Model starting from the same data.

The results of the BM25 algorithm are stored in an output file, `results.csv`, which serves as a persistent log for performance evaluation. If the file does not already exist, it is initialized with the required columns: `row_id`, `offering_id`, and `bm25_mse`. The script processes the queries iteratively, invoking the `apply_bm25` function for each query.

The function calculates the MSE for the recommended place and appends the results to the output file. This iterative approach ensures that the system handles all queries in a systematic and orderly manner.

To optimize computational efficiency, the workflow incorporates a mechanism to avoid reprocessing previously evaluated queries. By maintaining a record of processed queries using the `row_id` column in the output file, the script skips redundant calculations, saving time and computational resources.

After all queries have been processed, the average MSE across the entire dataset is calculated and displayed. This average MSE serves as the baseline metric for comparing the BM25 model to more sophisticated approaches, such as the Enhanced Model.

5 Enhanced Model

The Enhanced Model (EM1) builds upon the BM25 baseline by incorporating advanced techniques for measuring semantic similarities between reviews. While BM25 relies on token frequency and document length normalization, EM1 leverages TF-IDF representations combined with cosine similarity. This approach aims to better capture the relationships between reviews.

5.1 Reasoning Behind the Model

BM25, despite its robustness, has several limitations that EM1 seeks to address:

- **Lack of Semantic Understanding**: BM25 treats words as independent tokens and does not account for semantic relationships.
- **Inflexibility for Multi-Word Queries**: BM25 struggles to capture nuances in multi-word phrases and their semantic context.
- **Exact Token Matching**: BM25 relies on exact term matches, making it less effective in identifying synonyms or paraphrases.

The hypothesis driving EM1 is that vector-based similarity, such as TF-IDF with cosine similarity, can better model the semantic relationships between reviews. By representing reviews as vectors, EM1 measures their closeness in a high-dimensional space, overcoming the reliance on exact token matches.

5.2 Key Differences Between BM25 and EM1

The following table outlines the key differences between BM25 and EM1:

Aspect	**BM25**	**EM1**
Relevance Measure	Token frequency and document length normalization	Cosine similarity of TF-IDF vectors
Semantic Understanding	Limited to token matching	Leverages vector-based similarity
Handling Synonyms	Poor	Better recognition of semantic relationships
Computational Complexity	Lower	Higher due to vectorization and similarity calculations

Table 1: Comparison of BM25 and EM1

5.3 Implementation of EM1

The EM1 model uses the following steps in its implementation:

1. **TF-IDF Vectorization**: Each review is transformed into a numerical vector using TF-IDF, which assigns weights based on the term's frequency in the document and its rarity across the corpus. This emphasizes terms that are more discriminative.

2. **Initial Attempt with Euclidean Distance**: Initially, Euclidean distance was employed to measure similarity between TF-IDF vectors. However, this approach failed to outperform BM25 due to its sensitivity to scale and focus on magnitude, rather than directional relationships.

3. **Switch to Cosine Similarity**: Based on the limitations of Euclidean distance, cosine similarity was adopted. This metric quantifies the angle between vectors, focusing on their directional closeness while normalizing differences in magnitude.

4. **Top Match Selection**: The place with the highest similarity score, as calculated by cosine similarity, is selected as the recommendation.

5.4 Application of EM1

The application of EM1 involves the following workflow:

- **Preprocessing**: The query and other reviews are preprocessed to ensure consistency (tokenization, lemmatization, and removal of stop words).
- **Vectorization and Similarity Computation**: TF-IDF vectors are computed for all reviews, and cosine similarity is used to measure relevance between the query and other places.
- **Recommendation**: The most similar place is identified based on the highest cosine similarity score.
- **Performance Evaluation**: The Mean Squared Error (MSE) between the ratings of the query place and the recommended place is calculated for all aspects.

5.5 Advantages of EM1

The EM1 model offers several advantages over BM25:

- **Improved Semantic Understanding**: EM1 captures relationships between words, enabling better handling of synonyms and variations in vocabulary.
- **Better Query Matching**: By considering the overall content of reviews, EM1 provides more relevant recommendations for complex or nuanced queries.
- **Flexibility for Enhancements**: The vector-based framework can be extended with additional techniques, such as word embeddings or pre-trained language models, for deeper semantic analysis.

5.6 Conclusion

The EM1 model addresses several limitations of BM25 by incorporating vector-based similarity techniques. The failed attempt with Euclidean distance highlighted the importance of choosing the right similarity metric. While Euclidean distance struggled with semantic nuances and magnitude sensitivity, cosine similarity demonstrated its strength in

capturing relationships within TF-IDF vectors. Despite being computationally more demanding, EM1 with cosine similarity delivered a significantly lower average MSE (**0.4356**) compared to BM25 (**0.4581**).

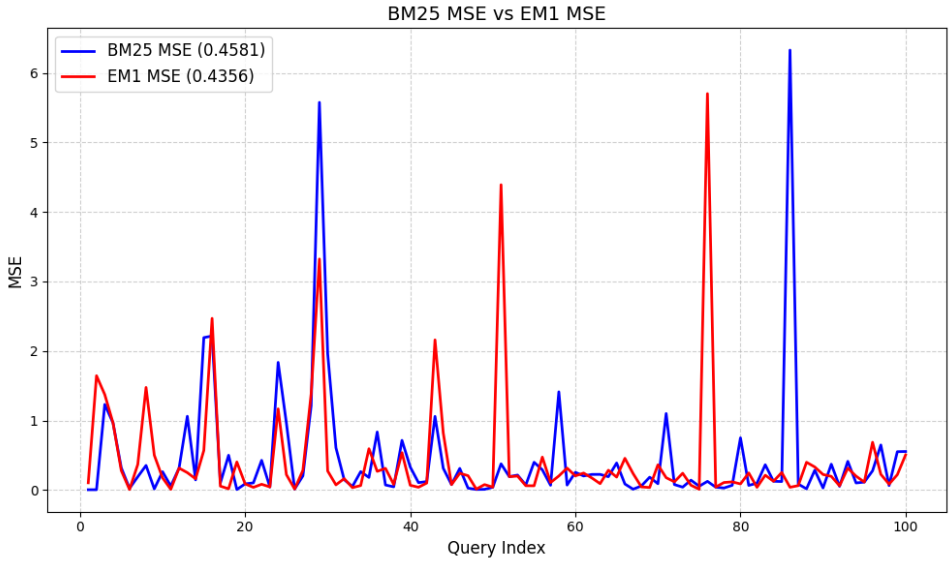

Figure 1: Comparison of BM25 MSE vs. EM1 MSE across Queries

Figure 1 illustrates the comparison between BM25 and EM1 MSE across multiple queries. The blue line represents BM25 MSE, while the red line represents EM1 MSE. From the graph, we observe the following:

- **Overall Performance**: EM1 achieves a lower average MSE (**0.4356**) compared to BM25 (**0.4581**), indicating better performance on most queries.
- **Outliers**: In some cases, such as near query indices around 20 and 80, EM1 demonstrates significant improvement by maintaining lower MSE values, while BM25 experiences spikes.
- **Consistency**: EM1 tends to be more consistent, as reflected by smoother transitions between query indices, whereas BM25 exhibits more erratic variations.
- **Key Insight**: EM1's ability to leverage semantic similarities allows it to excel in scenarios where BM25 struggles with exact token matches, as seen in multi-word queries or cases involving synonyms.

The results clearly demonstrate that EM1 provides a robust improvement over BM25, especially in complex scenarios. This highlights the effectiveness of vector-based similarity techniques for capturing semantic relationships, ultimately leading to more accurate recommendations. Future work could further refine EM1 by incorporating pre-trained embeddings or advanced models to enhance its semantic understanding capabilities.