Review-Based Car Search System for Improved Decision Making

Rodrigo Campos Rodrigues
up202108847@edu.fe.up.pt
Universidade do Porto - Faculdade de Engenharia
Porto, Portugal

Christopher Schneider
up202401397@edu.fe.up.pt
Universidade do Porto - Faculdade de Engenharia
Porto, Portugal

Luis Vieira Relvas
up202108661@edu.fe.up.pt
Universidade do Porto - Faculdade de Engenharia
Porto, Portugal

Daniel Samuel Edim
up202400301@edu.fe.up.pt
Universidade do Porto - Faculdade de Engenharia
Porto, Portugal

Abstract

This research project project tackles a crucial issue in the automotive industry, as car assessments profoundly impact consumer choices. The primary objective is to create a search engine for vehicle evaluations that adequately fulfills consumers' informational requirements. To do this, data is collected, processed, and organized into a complete dataset that includes critical review components. The system utilizes sophisticated retrieval techniques, including complex search algorithms, and is assessed using a variety of queries. Experimental findings indicate the system's efficacy, validated by key performance indicators such as the Mean Average Precision (MAP). An Enhanced System attains a MAP of 0.349, surpassing other evaluated configurations. These findings underscore the system's resilience in providing precise, relevant, and user-centric outcomes. Furthermore, enhancements like query expansion, ranking optimization, and user-driven feedback methods are examined. The research significantly improves decision-making in vehicle purchases by facilitating rapid and accurate information retrieval.

CCS Concepts

• **Information systems → Information retrieval**.

Keywords

Data Sources, Data Collection, Data Preparation, Data Characterization, Data Retrieval, Cars, Reviews.

ACM Reference Format:
Rodrigo Campos Rodrigues, Christopher Schneider, Luis Vieira Relvas, and Daniel Samuel Edim. 2024. Review-Based Car Search System for Improved Decision Making. In *Proceedings of PRI (G12).* ACM, New York, NY, USA, 17 pages.

1 Introduction

In the contemporary digital era, the vast quantity of internet information often incapacitates people from making educated judgments. Large amounts of data are generated every day across various sectors, with online consumer reviews being one key resource for informed decision-making. Car reviews play an important role in modeling consumer preferences and brand reputations. These reviews, spanning from expert evaluations to end-users, offer valuable insights into vehicle performance, safety, reliability, and customer satisfaction. Data retrieval is crucial for understanding market trends, improving product offerings, and enhancing customer experiences. However, the unstructured nature of car reviews, often consisting of text, ratings, and multimedia, presents challenges in terms of effective data extraction, organization, and interpretation. This paper dives into the methodologies and techniques of data retrieval with a particular focus on car reviews trying to transform raw review content into valuable information for consumers.

2 Data Extraction

Dataset	Features	Brands	Car Reviews	Size
Car Reviews	8	50	1145	12,3 (MB)

Table 1: Dataset Volume

2.1 Authority of the source

Our dataset emerged from the website CarsGuide.com.au [5]. Founded in November 2011 the Sydney based company "specializes in automotive sales and purchases, as well as car news, advice and reviews"[6]. CarsGuide is regarded as one of the more authoritative platforms for car reviews in Australia, being widely used by consumers, car buyers, and sellers for reliable car information. The content is typically produced by experienced automotive journalists or industry professionals, enhancing its credibility. The company backing from major automotive industry players, such as the Cox Automotive Group, lending it legitimacy. This commercial affiliation, however, means that while reviews aim for objectivity, there may be commercial incentives influencing certain content or car reviews. However, due to the nature of our project objective (the creation of a search engine prototype), this fact is not particularly important. Thus, CarsGuide.com.au is an authoritative source, especially when extracting expert opinions.

2.2 Scraping the data

The second step to take is data extraction. After exploring the most varied range of existing databases, we concluded that starting from the source would be more interesting and beneficial to our specific needs. Our main goal in choosing a source for data scraping was to identify a website where the car reviews followed a consistent and well-structured format. This consistency is vital for efficient data extraction, as it simplifies the process of parsing and organizing the information and prevents missing values. Our first choice was Edmunds Website [4], a complete website that has all the reviews structured and offers a varied quantity of car reviews. We didn't manage to scrape from this website since it has protection against automated bots. So, after careful consideration, we selected CarsGuide [5] as the primary data source. This website met our requirements, offering a uniform structure across its car reviews, which is essential for automating the extracting process and ensuring that the data collected is reliable and comprehensive. When scraping data directly from CarsGuide [5], we aimed to get key attributes such as vehicle specifications, performance evaluations, interior details, and expert opinions, ignoring the reviews that don't meet our requirements. The dataset is then composed of floating-point numbers and text, respectively ratings and reviews. This step helped us with data preparation and cleaning, as we didn't have to deal with *null* or missing values because we only scraped what we wanted and when information was complete. This raw data serves as the base to be analyzed in the future, allowing us to apply advanced techniques to uncover meaningful insights from user reviews. Regarding the license, we have been trying to contact the company that manages the website although, at the moment of the delivery, we didn't get any reply, so our dataset will be only used for academic purposes.

3 Data Preparation

This section outlines the data preparation pipeline [1] we followed to build the final dataset, designed to optimize our future search engine by producing a well-structured and organized dataset. Firstly, we developed a custom Python scraper to extract data from the source. The scraper processed multiple reviews from the website, verifying whether each review followed the specific structure we required by examining the relevant HTML tags. The specific structure consists of a combination of 8 categories: Price and Features, Design, Practicality, Under the bonnet, Efficiency, Driving, Safety, and Ownership. For each category, a review and the respective rating is required. So, the initial scraping produced a dataset, saved in a CSV file, with the following structure: car name, overall rating, and ratings for key features mentioned above, as well as their respective reviews. At this point, the data contained a lot of noise such as HTML tags, special characters, and more. Secondly, we applied a data-cleaning script to remove these HTML tags and the special characters that appear in the middle of the cells. Additionally, the script cleaned the "Name" column by removing the word "review," leaving only the car's name and model. Thirdly we extracted the car's year and brand from the car column, which were then assigned to new columns, "Year" and "Brand". After these three main steps we obtained an organized and structured dataset. The final dataset included the original data plus two additional columns— "Year" and "Brand" — which provided valuable information for further analysis.

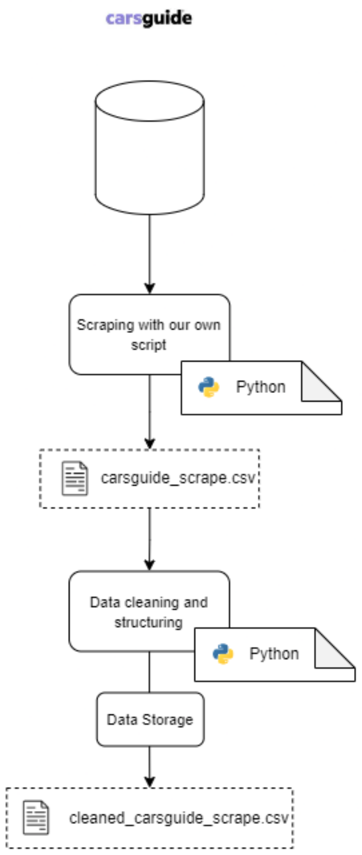
carsguide
Scraping with our own script
Python
carsguide_scrape.csv
Data cleaning and structuring
Python
Data Storage
cleaned_carsguide_scrape.csv

Figure 1: Data preparation pipeline

4 Data Characterization

In this section, we will characterize the documents that are produced by the pipeline. To display the plots we used the Python libraries: Pandas [1], SeaBorn [2] and MatPlotLib [3].

4.1 Average Rating per Brand

This plot [2] represents the average overall rating for each brand. It is calculated by the mean of the overall rating of each car that belongs to that brand. As expected, the brands that have the best ratings are the luxury ones just as Rolls-Royce, Tesla, McLaren, and Ferrari.

4.2 Average Number of Words per Review for each Brand

This metric [3] gives insights into how detailed the reviews tend to be for each brand. The top 5 brands with the highest average number of words per review are Rolls-Royce, Tesla, Ram, Genesis, and Porsche suggesting that these reviews are more likely to provide more in-depth feedback about the respective car. Regarding the first three however, it is important to point out that the quantity

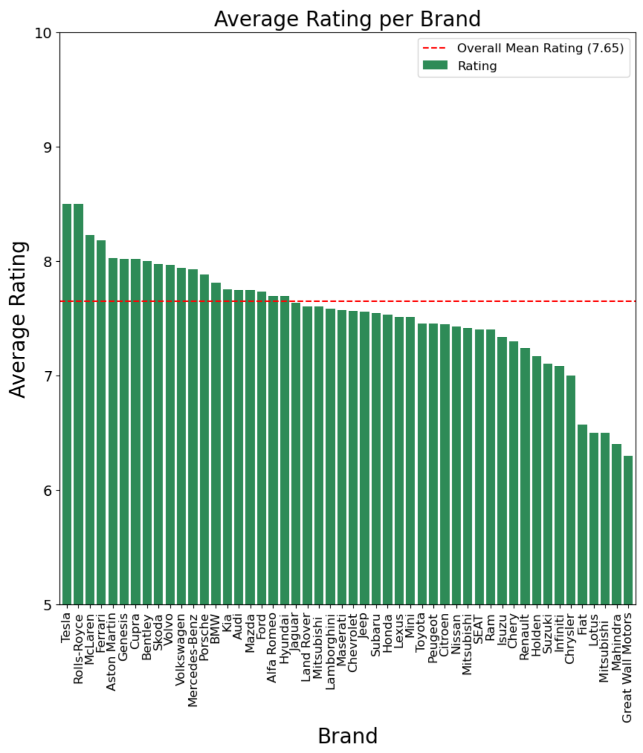

Figure 2: Average Rating per Brand

of their reviews does not exceed 1% of the dataset's volume (see figure [4]).

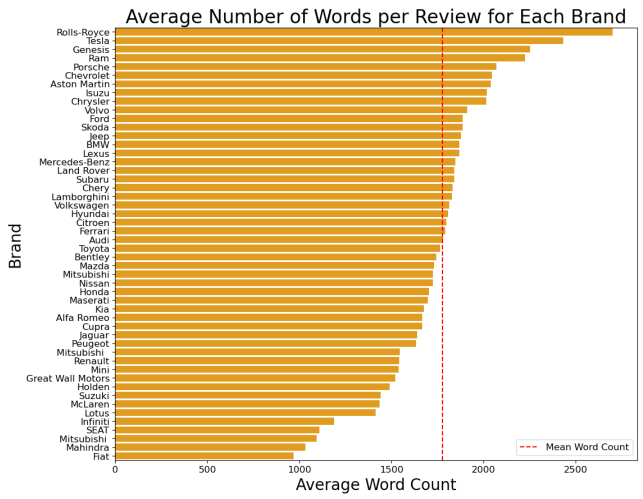

Figure 3: Average Number of Words per Review for each Brand

4.3 Number of Reviews per Brand

This figure [4] shows the percentage of reviews for each car brand. The top 5 brands are Audi, Mercedes-Benz, Toyota, BMW and Volkswagen. These brands receive the most feedback due to their popularity and larger customer base.

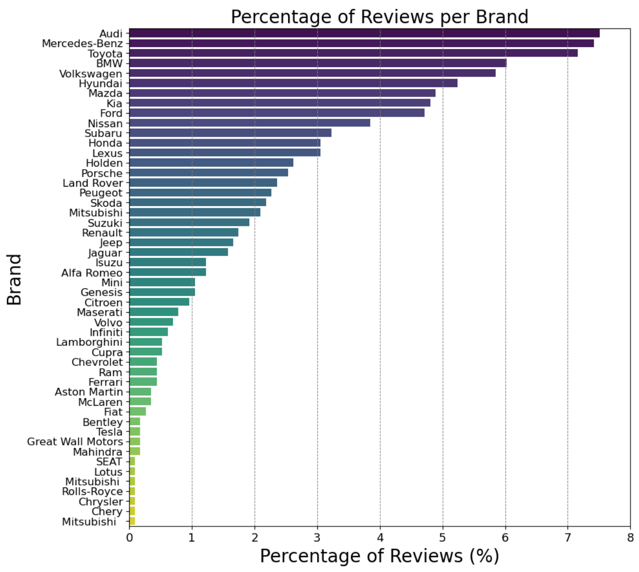

Figure 4: Percentage of Reviews per Brand

4.4 Number of Reviews per Year

Figure [5] shows that the top 5 years with the highest number of reviews are 2017, 2018, 2019, 2020, and 2021. Here, 2017 stands out as the peak year with the highest number of reviews.

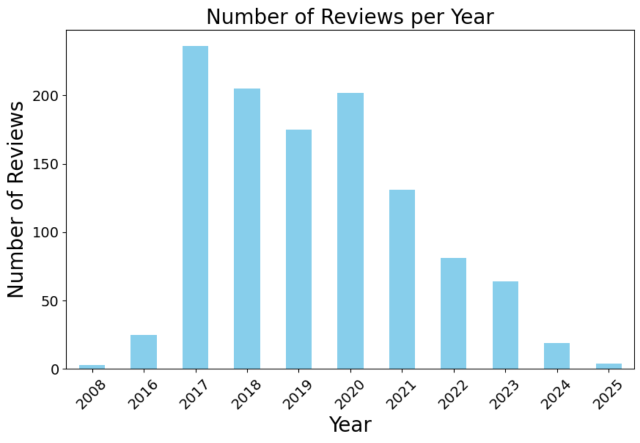

Figure 5: Number of Reviews per Year

4.5 Word Cloud

Word clouds are useful in the context of the project in order to check which keywords users search frequently and indicate terms that are more relevant or important. Moreover, the search engine will identify, prioritize, and refine the search to get better results.

In the first Word Cloud [6] related to the car models, some brands stand out. Comparing this, with the figure [4], which illustrates the percentages of reviews for each brand, it is obvious that there is a correlation between those two. The brands with the highest number of reviews are also the ones most frequently mentioned in our dataset, indicating that there is a strong relationship between review volume and brand visibility.

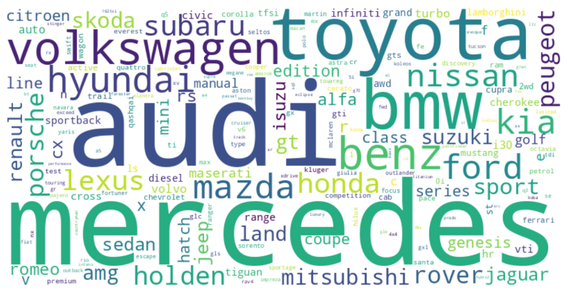

Figure 6: Word Cloud Car Models

In the second Word Cloud [7] which refers to the Car Reviews, it is clear that the words that are more highlighted are related to the details of the cars. This was one of our objectives while web scraping: getting detailed information about the cars, and not only general feedback that would not add anything to the customers.

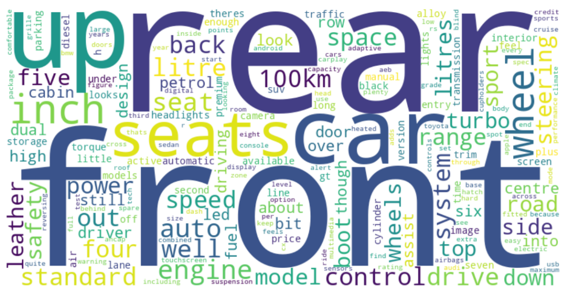

Figure 7: Word Cloud Car Reviews

4.6 Conceptual Data Model

The Conceptual Data Model describes how the system is designed at a high level. The Car class includes attributes for the model and year of the vehicle. Each car is associated with a specific Brand, while a brand can have multiple cars. The Brand class is defined by its name attribute. Every car can have one or more Reviews, each providing an overall evaluation. These reviews are composed of sub-reviews that assess different aspects of the car, such as price, design, practicality, driving performance, and safety. Each sub-review has a numeric rating associated with it, and the overall rating for the review is calculated as the mean of these sub-review ratings. Conceptual Data Model: (Figure. 8)

5 Prospective Search Tasks

In the future, we plan to design a car review search engine that provides users with an efficient tool to discover and evaluate opinions on a wide range of car models, brands, and features. The word clouds allowed us to start thinking about the most common words that users would search for, such as details about a specific model and the physical perks the vehicle might have. From the car reviews word cloud, the most frequent words are "front" and "rear". This shows a strong interest among the clients in the specific features and details of the vehicles they are considering buying. As a result, potential buyers are not only looking for general information about the cars but are also focused on details related to the vehicle's design, performance, and functionality. Therefore, it would be beneficial to perform in-depth searches and comparisons highlighting these crucial aspects of the car. As a result, we agreed on the following information needs:

- **Cars with cruise control and lane keep assist.**
- **Family cars with a reasonable price.**
- **Comfortable and smooth driving experience.**
- **Modern design cars.**
- **High performance sports cars.**

6 Collection and Indexing

This section highlights the collection and indexing processes used in the developed search engines. *Information Retrieval* is the process of identifying and retrieving information system resources relevant to an information need [7]. The search engines will be implemented in Solr. Solr[8] is an open-source search platform that runs on top of the Apache Lucene library. The collection is important to format uniformly the data, allowing for a consistent structure across documents and improving retrieval quality. Indexing then structures this collected data for fast and relevant retrieval. Together, collection and indexing allow the information retrieval system to handle large volumes of data, transforming raw information into an organized, searchable format that meets users' information needs effectively.

6.1 Document Definition and Structure

Defining a document is an important step as it establishes the structure and characteristics of the data to be indexed and retrieved. A document in Solr is a collection of fields and each field holds specific information that represents a distinct attribute. A well-constructed document definition is essential for optimizing search performance, relevance, and retrieval efficiency within an information retrieval system.

The document structure has been defined to support comprehensive indexing and retrieval of car review data, with a focus on textual content and rating-based numerical attributes. It is to be recalled that a document contains information about feature reviews, overall ratings and feature ratings.

6.2 Indexed fields and Processing

Indexing is a necessary procedure in Information Retrieval systems as it directly affects the efficiency and effectiveness of data retrieval. Indexed fields enable fast access to relevant information by organizing and optimizing data for search queries. Transforming the raw data into a structured format improves response time and facilitates complex search tasks. Effective indexing strategies also contribute to scalability, enabling the system to handle large volumes of data while maintaining the required high performance. Consequently, the careful selection and configuration of indexed fields influence the system's ability to meet user information needs accurately and efficiently. Solr, as a powerful and flexible search platform, provides a rich set of indexing capabilities to optimize data retrieval. Solr's advanced features include configurable analyzers, tokenizers, and filters allowing for fine-grained control over how the text data is processed during indexing. Tokenizers are the first step in the text-analysis process, as they break down a stream of text into smaller units called tokens. Filters are applied after the tokenizers to refine and standardize documents before they are indexed. They perform essential operations such as converting text to lowercase, removing special characters, and many more. Our focus will be on the textual fields that describe the reviews as they will be our main source of data. The other columns, apart from the ones mentioned before, will not be indexed as they are not relevant for search purposes. The only number that will be indexed is the overall rating of each car. Even though the rating features are not indexed, they are still searchable. They can function as filters because they are declared as floating points. In Solr, when assigning this type to a field, *docValues*, by default, is enabled, which enables performing queries using such fields[14]. We opted to use *docValues* instead of *stored*, since it may improve the performance due to memory accesses[15]. In Solr, the most diverse filters are applied to process the text data during indexing and query execution. These filters play an important role in transforming the text in ways that optimize search accuracy and performance. Below is a brief explanation of the function of each filter used in the schema:

- **solr.StandardTokenizerFactory**: is a tokenizer that divides text into individual tokens based on spaces, punctuation, and other word boundaries.
- **solr.ASCIIFoldingFilterFactory**: is designed to replace the special characters with their corresponding ASCII representation.
- **solr.LowerCaseFilterFactory**: converts all tokens in the text to lowercase.
- **solr.SynonymGraphFilterFactory**: allows for the expansion of search terms by incorporating predefined synonyms. It works by referring to a list of synonym mappings and, during indexing, replaces the terms with their corresponding synonyms. We gather our synonyms.txt from an open-source synonyms file in Github[12].

6.3 Schema Details

Field Type Name	Query Tokenizer	Query Filters
review	solr.StandardTokenizerFactory	solr.ASCIIFoldingFilterFactory solr.LowerCaseFilterFactory solr.EnglishMinimalStemFilterFactory solr.SynonymGraphFilterFactory

Table 2: Boosted schema configuration

Field	Type	Indexed
Name	review	yes
Rating	floating point	yes
Reviews' Features	review	yes
Features' Ratings	floating point	no
URL	string	no
Year	integer	yes
Brand	string	yes

Table 3: Document Field Types

7 Retrieval

To retrieve information, we constructed two search engines. One loaded with a simple schema and the other loaded with an enhanced, boosted schema. The simple schema uses the solr.StandardTokenizerFactory and the filters solr.ASCIIFoldingFilterFactory, solr.LowerCaseFilterFactory. On the other hand, the boosted schema includes all the tokenizers, query analyzers, and filters described in (Table. 2). When querying both systems, we tried to always follow the same structure to properly evaluate the system and emphasize the differences between them. We use the eDisMax [9] query parser in both systems, although we will not boost any term or field in the simple search engine. More about Solr filters, can be explored here [10].

For the Simple Search Engine:

- **q (query)**: Specifies the main query string, focusing on the terms most relevant to the user's intentions.
- **q.op (query operator)**: Specifies the default logical operator applied between terms in a query string when multiple terms are present without explicit operators.
- **(fl (fields list)**: Controls which fields are returned in the query response.
- **fq (filter query**: Used to refine the main query by applying additional constraints without impacting the score of the documents.
- **qf (query filter)** : Designates specific fields that should be searched in response to the query.
- **sort**: Specifies the sorting order of the search results based on one or more fields.
- **start**: Defined the starting point for results, which is used for pagination.
- **rows**: Specifies the number of documents to be returned in a single query response.
- **defType**: Determines which query parser Solr should use.

The Boosted Search Engine uses all of the above parameters and more:

- **pf (phrase fields)**: This parameter boosts results that match exact match phrases in specified fields.
- **ps (phrase slop)**: Specifies the acceptable distance between terms in a phrase query.

Considering the parameters outlined above, the only changes between queries in our implementation are the specific field being searched and the query content itself, all other parameters remain the same across queries. Also, we assigned different weights in the query field parameter to prioritize the importance of the specific fields. The higher relevance will always be given to the column that contains the feature we are searching for. The phrase slop parameter in the boosted search engine is set to 4, allowing for an average of four tokens to be between query terms to enhance flexibility in phrase matching. Applying this boosted weighting approach consistently across queries has optimized the system's ability to handle queries effectively, leading to improved retrieval performances.

8 Evaluation

Evaluation of the resulting outputs is essential in objectively determining the system's performance. The primary consideration in the evaluation should be relevancy. A document is considered to be relevant if it satisfies the topic or information need of a query, even though this is not always easy to evaluate. To help us with the evaluation metrics, we used *trec eval*[13]. It is a widely used evaluation tool in the field of information retrieval designed for analyzing effectiveness and efficiency of search engines and information retrieval systems. With *trec eval* we can generate evaluation metrics such as precision and recall, average precision (*AvP*), mean average precision (*MAP*), and *Precision at N*. The last three will be explored further to evaluate the queries. Before diving into complex evaluation metrics, let's define what precision and recall are. **Precision** quantifies the proportion of recovered documents that are pertinent. If 10 papers are retrieved and 5 are relevant, then the precision is 0.5. **Recall** reflects the proportion of appropriate documents that have been retrieved. If 10 relevant results exist and only 5 are retrieved, the recall is 0.5. Also significant in the context of evaluation is ranking. For that reason, we need to adapt precision and recall measurements to specific levels within the returned document set.

A common approach uses precision-recall curves: a plot that traces how precision evolves as recall increases. The area under the curve is considered a measure of the system's overall quality. A larger area indicates that the system maintains higher precision as recall increases, whereas a smaller area suggests that the system retrieves many documents, although few are relevant. These evaluation measures are significant, yet we will further investigate additional options to meet our requirements. Precision at N (*P@N*) and *Mean Average Precision* are examples that belong to the Rank-Based Evaluation metrics. Within a ranked result list, **Precision at N** describes the precision at a designated threshold. The mean of the AvP computed at the query degree is **Mean Average Precision**, a metric that is known for its stability. Our search systems will be evaluated primarily based on these final two metrics.

Before diving into the individual queries, to compare the documents retrieved from the different search engines, we created a qrels.txt file, that contains the relevant documents for each query. The qrels file is formatted in the following way: query ID, iteration, document ID, and relevance. To generate this file, we retrieved 40 documents from each query for each system. After having the 80 documents we started analyzing manually every document by checking if it was relevant or not. After this step, we are now ready to start evaluating our queries, by comparing the documents generated for a specific query with the documents that are in the qrels.txt.

8.1 Cars with Cruise Control and Lane Keep Assist.

Information Need: Cars with cruise control and lane keep assist.
Examination: This query aims to identify cars equipped with both cruise control and lane-keep assist features. The search should include keywords such as 'cruise control', and 'lane keep assist' in vehicle descriptions, reviews, and specifications. Results should prioritize cars offering both features and not only one of them.

Simple
Query: `(cruise control) AND (lane keep assist)`
df: `Safety_Feature`
sort: `score desc`
Boosted
Query: `("cruise control") AND ("lane keep assist")`
qf: `Safety_Feature^3`
pf: `Safety_Feature^5`
ps: `4`
sort: `score desc`

Metric	Simple System	Enhanced System
P@20	0.850	0.950
AvP	0.474	0.635

Table 4: Query 1 and respective metrics

Result Evaluation: The metrics (Tab. 4) indicate a meaningful improvement, in the boosted search engine, due to the use of the phrase slops. In the baseline, we are accepting every document that has the words cruise, control, lane, keep, and assist, while in the enhanced we are exactly expecting the document retrieved to include the phrases "cruise control" and "lane keep assist", of course with some tolerance, defined in the ps field (*see* Fig. 9 & 10)

8.2 Family cars with a reasonable price

Information Need: Affordable Cars suitable for families with kids.
Examination: This aim is to find inexpensive, family-friendly cars. The search should include 'reasonable' and 'family car,' in vehicle descriptions, reviews, and specifications. Results should prioritize cars with large interiors, substantial trunk space, and child seat-compatible seating.

Simple
Query: `(reasonable AND price) AND (family car)`
qf: `Practicality_Feature Price_Feature`
sort: `score desc`
Boosted
Query: `(reasonable price) AND ("family car")`
qf: `Practicality_Feature^7 Price_Feature^7`
pf: `Practicality_Feature^11 Price_Feature^11`
ps: `4`
bq: `Practicality_Feature:(family)^10`
sort: `score desc`

Metric	Simple System	Enhanced System
P@20	0.550	0.600
AvP	0.289	0.381

Table 5: Query 2 and respective metrics

Result Evaluation: The evaluation (Tab. 5) indicates a modest improvement in ranking quality and relevance, suggesting that the enhanced system better addresses the specified information need. However, with an AvP of only 0.6, there is room for further refinement to ensure the results more comprehensively meet the requirements for affordable, family-friendly cars. The gap between the accuracies for the baseline and the boosted system search engines happens because in the enhanced system we are giving more relevance to the word "family". With this change, the order of the documents will also be different, giving us better results. Besides this change, this is not the only impact, similar to the other queries the phrase slops, and the boosts given to the fields also have their importance (*see* Fig. 11 & 12).

8.3 Comfortable and Smooth Driving Experience

Information Need: Vehicles that offer a smooth and comfortable driving experience.
Examination: The goal is to identify cars that prioritize comfort and driving ease. Search results should focus on vehicles with advanced suspension systems, noise reduction features, ergonomic seating, and intuitive driver-assistance systems. Relevant reviews and specifications should emphasize ride quality, reduced road vibrations, and effortless steering.

Simple
Query: `("comfortable ride") AND (smooth ride)`
qf: `Driving_Feature Practicality_Feature`
fq: `Ownership_Feature : [7 TO *]`
sort: `Rating desc, score desc`
Boosted
Query: `("comfortable ride") AND (smooth ride)`
qf: `Driving_Feature^7 Practicality_Feature^7`
pf: `(Driving_Feature)^9 Practicality_Feature^9`
ps: `4`
fq: `Ownership_Feature : [7 TO *]`
bq: `(comfortable)^5 (smooth)^5`
sort: `Rating desc, score desc`

Metric	Simple System	Enhanced System
P@20	0.850	0.900
AvP	0.617	0.620

Table 6: Query 3 and respective metrics

Result Evaluation: Table 6 reflects a significant enhancement in both precision and ranking quality, demonstrating that the improved system more effectively satisfies the search task. The results indicate advancements in recognizing automobiles that prioritize comfort and smooth driving; yet, further optimization will be required to better fit with the given criteria. The boosts given to the fields and terms are also important to better answer the user's needs (*see* Fig. 13 & 14).

8.4 Modern Design Cars

Information Need: Cars with a modern design.
Examination: This query seeks to locate automobiles that display a contemporary aesthetic design. The inquiry must encompass buzzwords including 'modern', 'current', 'sleek', 'innovation', and 'sophisticated' within vehicle descriptions, evaluations, and specifications. Results must emphasize vehicles that use innovative design features such as angular contours, avant-garde lighting systems, streamlined and ergonomic interiors, expansive digital interfaces, and sophisticated technological integrations.
Result Evaluation: The metrics (Tab. 7) indicate a moderate improvement from the baseline to the improved system. The overall AvP score suggests that there is still substantial room for improvement in the comprehensive fulfillment of the query requirements across all retrieved results, despite the fact that the enhanced system more effectively aligns with the modern design-focused Information Need, particularly by better ranking relevant results within the top 20 (*see* Fig. 15 & 16).

Simple
Query: `modern AND design`
df: `Design_Feature`
sort: `score desc`
Boosted
Query: `("modern design")`
qf: `Design_Feature^3`
pf: `Design_Feature^5`
ps: `4`
bq: `Design_Feature: modern^3`
Sort: `score desc`

Metric	Simple System	Enhanced System
P@20	0.75	0.8
AvP	0.385	0.474

Table 7: Query 4 and respective metrics

8.5 High Performance Sports Cars

Information Need: Sports cars are designed for exceptional performance and speed.
Examination: The objective is to find vehicles optimized for high performance, with specifications emphasizing speed, acceleration, handling, and aerodynamics. Searches should include terms like 'high performance' and 'sport'. Relevant results should prioritize cars with advanced engine technology and a sporty driving feeling.

Simple
Query: `(sport*) AND ("high performance")`
qf: `Driving_Feature Under_Bonnet_Feature`
fq: `Rating : [7 TO *]`
sort: `score desc`
Boosted
Query: `(sport*) AND (high performance)`
qf: `Driving_Feature^7 Under_Bonnet_Feature^7`
pf: `Driving_Feature^9 Under_Bonnet_Feature^9`
ps: `4`
bq: `(sport^10)`
fq: `Rating: [7 TO *]`
sort: `score desc`

Metric	Simple System	Enhanced System
P@20	0.6	0.8
AvP	0.342	0.564

Table 8: Query 5 and respective metrics

Result Evaluation: A significant enhancement in precision and overall ranking quality showed the system's improved capacity to meet information requirements (see Tab. 8). The findings indicate substantial advancement in recognizing high-performance sports automobiles, efficiently emphasizing essential characteristics such as velocity, maneuverability, and sophisticated engineering. In this query, we also opted to use the wildcard feature from Solr to not only search for sport but also to search for its derivations just like sporty. Nonetheless, there remains potential for further enhancement (*see* Fig. 17 & 18).

Metric	Simple System	Enhanced System
MAP	0.421	0.535

Table 9: Overall MAP for both IR systems

9 Evaluation Metrics Discussion

From the table above (Table 9), it is noticeable that the Enhanced System performed better than the Simple System, as expected. As explained above this difference relies on how both systems are constructed, starting at the indexing. The Simple System schema is almost a default schema, while the Enhanced System contains features that improve its query analysis. For example, the synonyms factory, makes the schema not only look for the terms present on the query but also for synonyms of those words, which increases the number of relevant documents retrieved. Concerning queries, we always tried to maintain a similar structure between all queries for both systems, and the impact of the boosts, phrase slops, wildcards, and independent terms is clear. The MAP is the mean of the AvP returned from the queries, and it is calculated for each system.

10 Improvements on Information Retrieval

In the last chapters, we presented a preliminary iteration of the information retrieval system. Assessing it according to clearly defined information requirements and metrics based on accuracy and recall, our examination from an alternative viewpoint also facilitated the identification of the faults and limits of the selected methodologies. Consequently, this phase will include the implementation and assessment of the following features:

- **Semantic Search**
- **More Like This**
- **Re-Ranking Documents with LLM**

The first two are designed to rectify the observed deficiencies in the previous search engines. Again using actual information requirements as the starting point and adapting them to the encountered situation, the analysis of these aspects will be performed according to the same methodology as prior evaluations. Four systems will be examined: the simple system, the enhanced system, the semantic system, and a hybrid system, consisting of the improved system plus the proposed semantic reasoning. Analyzing each system modification independently enables a discussion on its permanency in the last search engine as well as an isolated study of its influence on the success of the system.

The last one, re-ranking documents in the user interface, will be employed to improve the user's overall experience while using our system.

The identical five information requirements from the previous evaluation will be employed to evaluate the system's capacity to manage the challenges posed by the natural language attributes of these queries. In response to the proliferation of available search methods, we decided to augment the volume of documents produced by a query from 20 to 40, leading to larger qrel files and differences in the evaluation metrics calculated before. There may be also differences in the P@20 due to two reasons: a re-evaluation of the queries or the sorting field is different from the score, which is generated by the Solr.

10.1 Semantic Search

Semantic search is a concept that demonstrates a system's capacity to comprehend the context and intent of a user's query, thereby surpassing simplistic keyword matching. This method, also known as "dense vector search," enables search systems to identify documents that are semantically related to the query. In the present scenario, dense vectors are distinguished from sparse vectors, which are typically connected with inverted indices. Sparse vectors typically generate large vectors with the majority of values being zeros, with dimensions that correspond to the number of phrases in the corpus. Conversely, dense vectors limit the scope of semantic meaning to a limited number of dimensions. These elements may be significantly fewer than the terms in a corpus, allowing for a more concise representation of the document's meaning. We used a deep learning model, the **all-MiniLM-L6-v2** model, and employed the Python sentence-transformers module to generate our document embeddings for each query. The model's quality is comparable to that of its larger counterparts, despite its diminutive size. The semantic schema is built on top of the Boosted schema(Tab. 2 so the query tokenizers and filters are the same, although there is a minor difference. We introduced a new **DenseVectorField** field type into the current schema to store the embeddings **for each feature**. **Semantic Schema:**

Key	Value
name	embeddingVector
class	solr.DenseVectorField
vectorDimension	384
similarityFunction	cosine
knnAlgorithm	hnsw

Table 10: Dense vector configuration in Solr schema

The **Knn Query Parser** concurrently applies a nearest neighbor algorithm search method to the new collection to identify the dense vectors that are closest to a specified query. In a given query, if it has more than one field, the final approach joins the Top20 results from each field, re-ordering them by their score. The outcome of these phases is our semantic and hybrid systems. In the latter, a specialized **Boolean Query Parser** [18] is employed to integrate our advanced system, which utilizes lexical search terms via the *Edismax* query parser, together with the semantic search capability that incorporates embeddings. We want to ensure that both the enhanced search words and the dense vector fields are considered when retrieving documents. An illustration on how this combination occurs on all employed examples can be found in Tables 18, 19, 20, 21 and 22.

10.2 More Like This

This integrated function analyzes the text in the given document and then identifies other indexed documents that display textual and contextual similarities. The results of this query consist of documents with the highest similarity scores. This approach to the feature is particularly appropriate and essential for the current project. Due to the intrinsic subjectivity in selecting a vehicle, consumers are anticipated to seek additional cars that correspond with their preferences.

Consequently, the method identifies three analogous cars with comparable material from a car review. The results are calculated and arranged according to Solr's internal match score.

11 Evaluation of Semantic Systems

11.1 Cars with Cruise Control and Lane Keep Assist

Final Result Evaluation: The introduction of the hybrid system demonstrates a significant improvement over the semantic system, addressing some of its limitations in handling structured and technical information needs. With P@20 rising to 0.8 and AvP to 0.301, the hybrid approach effectively combines the strengths of semantic understanding with the precision of enhanced lexical search (Table 11). While it still lags behind the enhanced system in both metrics, the hybrid system outperforms the semantic system, offering a balanced trade-off between contextual relevance and keyword precision.

The semantic system alone underperforms in this query, with low P@20 (0.65) and AvP (0.1986), highlighting its difficulty in precisely matching technical feature-based requirements. The hybrid system mitigates this issue by leveraging both dense vector fields for context and lexical search for exact matches, leading to a marked improvement in ranking quality and relevance (*see* Fig. 19, 20, 21, & 22).

Metric	Simple System	Enhanced System	Semantic System	Hybrid System
P@20	0.9	0.95	0.65	0.8
AvP	0.3594	0.4476	0.1986	0.301

Table 11: Query 1 metrics comparison

11.2 Family cars with a reasonable price

Final Result Evaluation: The semantic search system demonstrates an interesting dynamic for this query, which involves a balance between cost-effectiveness and family-friendly features. While the semantic system's P@20 (0.5) is slightly lower than both the baseline (0.55) and enhanced system (0.6), its AvP (0.2516) surpasses both, indicating better ranking quality across the entire set of retrieved results (Table 12). The hybrid system emerges as a balanced improvement over the other approaches for this query, achieving the highest P@20 (0.65) while maintaining a competitive AvP (0.2374). Its integration of semantic reasoning and lexical search enables it to prioritize relevant results at the top ranks while still addressing the broader intent of the query. This performance indicates that the hybrid system effectively captures the nuances of cost-effectiveness and family-friendly features, delivering a well-rounded retrieval approach. In contrast, the semantic system excels in identifying contextually relevant documents across the entire result set, remaining the lowest however in P@20, reflecting a challenge in ranking the most relevant documents prominently. This underperformance in top-ranked precision suggests that the semantic system's reliance on dense vectors may introduce results that align with the general intent but miss exact keyword matches critical to this query (*see* Fig. 23, 24, 25, & 26).

Metric	Simple System	Enhanced System	Semantic System	Hybrid System
P@20	0.55	0.6	0.5	0.65
AvP	0.1935	0.2234	0.2556	0.2374

Table 12: Query 2 metrics comparison

11.3 Comfortable and Smooth Driving Experience

Final Result Evaluation: For experience-driven queries like this, the results highlight the continued dominance of the baseline and enhanced systems like in Query 1, which achieve identical P@20 (0.8) and similar AvP scores (0.4148 and 0.411, respectively) (Table 13). These results indicate their reliability in ranking highly relevant results for descriptive requirements focused on comfort and smoothness. Both methods advantageously utilize exact keyword matching and proximity elements that correspond effectively with structured query specifications. The semantic system, while demonstrating strengths in contextual understanding, struggles to prioritize top-ranked results (P@20 = 0.7) and exhibits a drop in ranking quality overall (AvP = 0.2709). This indicates that its dependence on dense vector embeddings may inadequately encapsulate particular descriptive terms such as "comfortable ride" and "smooth ride," instead yielding more generalized contextually relevant outcomes. Interestingly, the hybrid system does not improve on the semantic system's limitations, with P@20 (0.65) and AvP (0.2677) falling below both the baseline and enhanced systems. This indicates that the hybrid integration may not sufficiently balance lexical precision and semantic reasoning for descriptive queries. Conversely, the baseline and upgraded systems demonstrate comparable performance in this case, indicating their efficacy in prioritizing top-ranked outcomes while preserving overall ranking quality (*see* Fig. 27, 28, 29, & 30).

Metric	Simple System	Enhanced System	Semantic System	Hybrid System
P@20	0.8	0.8	0.7	0.65
AvP	0.4148	0.411	0.2709	0.2677

Table 13: Query 3 metrics comparison

11.4 Modern Design Cars

Final Result Evaluation: This query highlights the strength of semantic and hybrid systems in addressing abstract, concept-driven information needs. The semantic system excels with the highest P@20 (0.9) and AvP (0.4099), demonstrating its ability to align with user intent for nuanced queries involving "modern design" (Table 14)). Its reliance on dense vector embeddings enables it to retrieve documents that indirectly reference modernity through synonymous or conceptually related terms, which traditional keyword-based approaches may overlook.

The hybrid system, while slightly behind the semantic system, achieves strong performance with a P@20 of 0.85 and an AvP of 0.3691. It integrates semantic reasoning with the system's improved lexical accuracy, successfully balancing contextual comprehension and exact keyword alignment, hence yielding strong outcomes for concept-intensive inquiries.

For context-rich queries involving abstract concepts like "modern design", the semantic search system excels, demonstrating its potential to outperform traditional systems in retrieving results that align with user intent. This reinforces the value of semantic search in addressing information needs that depend on a deeper understanding of context and meaning (*see* Fig. 31, 32, 33, & 34).

Metric	Simple System	Enhanced System	Semantic System	Hybrid System
P@20	0.75	0.75	0.9	0.85
AvP	0.2594	0.2239	0.4099	0.3691

Table 14: Query 4 metrics comparison

11.5 High Performance Sports Cars

Final Result Evaluation: The advanced system excels in this technical inquiry, achieving P@20 (0.8) and AvP (0.441), which highlight its capability in delivering accurate, attribute-focused outcomes (Table 15). Its emphasis on rigorous keyword matching and weighting guarantees the accurate identification and ranking of documents that prioritize speed, acceleration, and handling.

The semantic system, although equivalent to the baseline system in P@20 (0.6), has a substantial deficiency in AvP (0.1769), signifying worse overall ranking quality. This indicates that the contextual comprehension offered by dense vector embeddings has difficulty prioritizing outcomes with very specific performance characteristics. Instead, it obtains more general contextually pertinent documents that do not satisfy the rigorous standards of this query.

Unexpectedly, the hybrid system underperforms on both measures (P@20 = 0.55, AvP = 0.1203). This suggests that the combination of semantic reasoning with the improved system fails to provide the anticipated synergy for this question type. The hybrid technique may generate semantic noise, diminishing its capacity to rank articles that precisely meet the technical specifications (*see* Fig. 35, 36, 37, & 38).

Metric	Simple System	Enhanced System	Semantic System	Hybrid System
P@20	0.6	0.8	0.6	0.55
AvP	0.2491	0.441	0.1769	0.1203

Table 15: Query 5 metrics comparison

12 Final System Evaluation and Characterization

The Mean Average Precision (MAP) scores for each system, summarized in Tables 16 and 17, provide a quantitative basis for their characterization.

The Semantic System, with a MAP of 0.262, prioritizes contextual reasoning and semantic linkages in queries and documents. This strategy enhances the system's capacity to handle intent-driven inquiries, however it fails to attain the accuracy of keyword-based systems. This performance gap indicates that existing semantic techniques may need more modification to include precision into their comprehension of user queries.

The Hybrid System, integrating semantic and keyword-based retrieval, attains a MAP of 0.260, roughly matching the performance of the Semantic System. This outcome, although still below expectations, indicates the hybrid system's unexploited potential. The hybrid system demonstrates versatility by combining semantic reasoning with lexical accuracy, especially for complex queries that include numerous intentions or structures. The somewhat reduced MAP suggests that more tuning is required to enhance ranking accuracy, especially in precision-oriented applications.

Metric	Simple System	Enhanced System
MAP	0.295	0.349

Table 16: Overall MAP for Simple and Enhanced IR systems

Metric	Semantic System	Hybrid System
MAP	0.262	0.260

Table 17: Overall MAP for Semantic and Hybrid IR systems

The Enhanced System attains the highest overall MAP score (0.349), confirming its status as the most formidable method among the evaluated systems. This result highlights the significance of efficient exact-matching algorithms, which perform very well in contexts characterized by accurate keyword alignment and structured queries. This system's enhancements effectively reconcile proximity factors with ranking efficiency, resulting in enhanced performance for descriptive or narrowly focused searches.

Conversely, the Simple System exhibits competitive, albeit inferior, MAP scores (0.295), indicating its fundamental dependence on keyword-based retrieval. Despite its simple design, the system's performance underscores the continued significance of conventional lexical methods, especially for structured and precise queries when semantic interpretation is of lesser importance.

Additional optimization may improve the hybrid system's performance to compete with or exceed the semantic system in addressing extremely particular informational requirements. The baseline and enhanced algorithms maintain a distinct edge for queries that depend on accurate descriptive language, owing to their emphasis on exact keyword matching and proximity considerations. Semantic and hybrid systems, although typically effective for larger intent-based inquiries, have limits in meeting the precision demanded by some queries. Future improvements may entail optimizing the methodology to more effectively reconcile contextual comprehension with high-ranking accuracy, especially for complex inquiries.

13 User Interface

To provide a convenient user experience we developed a web based application to query our car reviews database via the Solr API [16] through a Next.js [17] application. Using the search bar in the middle of the screen, it is possible to enter the information needs. The engineered system will enable prospective customers to examine and refine automobile models according to preferences, including design, interior attributes, engine performance, or other criteria established during the study process. To improve the overall user experience, More-Like-This and Re-Ranking features were implemented. The More-Like-This feature, as explained before, enables the user to search for similar documents to the current one, without leaving the page. This helps the user find documents that are not being explicitly searched for but can still be relevant. By showing related documents, it also simplified the whole search process. The Re-Ranking feature is processed by the LLM *Gemini* [19] and, refines the order of the search results based on relevance to the user's query. While the initial ranking from Solr provides a good starting point, this implementation will minimize the time and effort needed to locate the desired information, ensuring that the users see the most relevant and consistent documents first. To evaluate the Re-Ranking feature we ran individual-assessments.

Query 2 Before Re-Rank
R R R N R N R N N R R R N R R N R N N R P@20 = 0.6
Query 2 After Re-Rank
R R R R N R R R N R N N R R R R R N N P@20 = 0.7
Query 3 Before Re-Rank
R R R R R R R R R R R R N R N R R N N P@20 = 0.8
Query 3 After Re-Rank
R R R R R R R R R R R R R R R R R N N P@20 = 0.9

14 Conclusions and Possible Future Work

This study introduced the creation and assessment of a search engine focused on automobile reviews to tackle the excessive volume of unstructured automotive data accessible online. Commencing with data collection, processing, and characterization, we effectively organized a dataset that connects automobiles with comprehensive user and expert evaluations across essential aspects, including design, safety, and efficiency. Since it was our first experience with scraping, we needed to do extensive study to accomplish our objective. Throughout this approach, as previously elucidated, we encountered websites that prohibited our automated procedure, hindering our advancement. During the second milestone, the most challenging job was identifying information requirements queries that would optimize our information retrieval systems. Furthermore, we always endeavored to maintain a uniform structure throughout the search engines to get reliable and superior outcomes. The third and final milestone, in addition to the foundational retrieval systems established in milestone two, included substantial enhancements such as query expansion techniques, ranking optimization, and methods for incorporating user input. In addition to these advancements that improved system accuracy and user happiness, as shown by experimental assessments, a user interface was created and implemented.

Thus, the tasks related to all three milestones have been completed. The final findings highlight the system's capacity to provide pertinent and customized solutions, enabling users to make educated choices about automobile purchases. Future efforts may concentrate on augmenting datasets, integrating multimedia evaluations, and optimizing feedback systems to improve retrieval precision and user satisfaction.

References

[1] McKinney, Wes. *Pandas: Powerful Python Data Analysis Toolkit.* Accessed October 1, 2024. https://pandas.pydata.org/.
[2] Waskom, Michael. *seaborn: statistical data visualization.* Accessed October 13, 2024. https://seaborn.pydata.org/.
[3] Hunter, John D. *Matplotlib: A 2D Graphics Environment.* Accessed October 1, 2024. https://matplotlib.org/.
[4] Edmunds. *Car Reviews, Pricing, and Specifications.* Accessed September 26, 2024. https://www.edmunds.com/.
[5] CarsGuide. *Car Reviews and News.* Accessed September 29, 2024. https://www.carsguide.com.au/.
[6] Wikipedia. *CarsGuide - History.* Accessed October 05, 2024. https://en.wikipedia.org/wiki/CarsGuide.
[7] Wikipedia. *Definition of Information Retrieval.* Accessed November 14, 2024. https://en.wikipedia.org/wiki/Information_retrieval.
[8] Apache Solr. *Solr: Open-Source Search Platform.* Accessed November 14, 2024. https://solr.apache.org/.
[9] Apache Solr. *The Extended DisMax.* Accessed November 14, 2024. https://solr.apache.org/guide/6_6/the-extended-dismax-query-parser.html.
[10] Apache Solr. *Filters.* Accessed November 14, 2024. https://solr.apache.org/guide/solr/latest/indexing-guide/filters.html.
[11] Apache Solr. *The Standard Query Parser.* Accessed November 14, 2024. https://solr.apache.org/guide/8_7/the-standard-query-parser.html.
[12] Hanson Robotics. *Synonyms List for Solr.* Accessed November 14, 2024. https://github.com/hansonrobotics/hr-solr/blob/master/synonyms_1.txt.
[13] U.S. NIST. *Trec Eval Tool for Information Retrieval Evaluation.* Accessed November 14, 2024. https://github.com/usnistgov/trec_eval.
[14] Apache Solr. *docValues.* Accessed November 18, 2024. https://solr.apache.org/guide/solr/latest/indexing-guide/docvalues.html.
[15] Apache Solr. *stored.* Accessed November 18, 2024. https://solr.apache.org/guide/6_6/defining-fields.html.
[16] Apache Solr. *stored.* Accessed December 14, 2024. https://solr.apache.org/guide/8_5/client-apis.html.
[17] Node JS. *stored.* Accessed December 14, 2024. https://nodejs.org/en/about.
[18] Apache Solr. *stored.* Accessed December 10, 2024. https://solr.apache.org/guide/7_7/other-parsers.html
[19] Google Gemini. *LLM* Accessed December 16, 2024. https://en.wikipedia.org/wiki/Gemini_(language_model)

15 Annexes

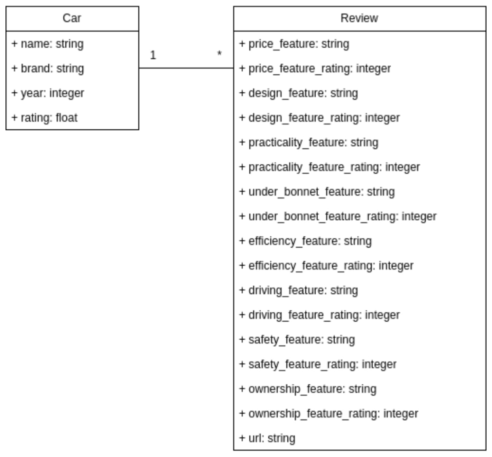
Car
+ name: string
+ brand: string
+ year: integer
+ rating: float
1
*
Review
+ price_feature: string
+ price_feature_rating: integer
+ design_feature: string
+ design_feature_rating: integer
+ practicality_feature: string
+ practicality_feature_rating: integer
+ under_bonnet_feature: string
+ under_bonnet_feature_rating: integer
+ efficiency_feature: string
+ efficiency_feature_rating: integer
+ driving_feature: string
+ driving_feature_rating: integer
+ safety_feature: string
+ safety_feature_rating: integer
+ ownership_feature: string
+ ownership_feature_rating: integer
+ url: string

Figure 8: Conceptual Data Model

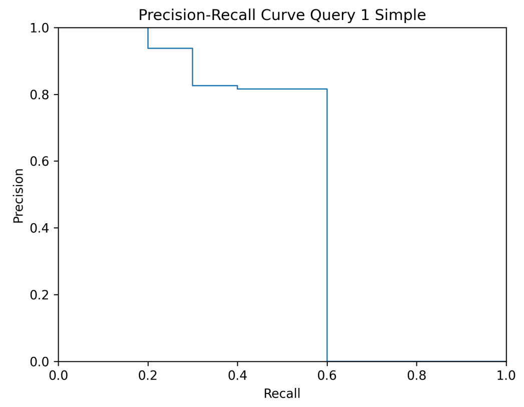

Figure 9: Query 1 metrics of simple system

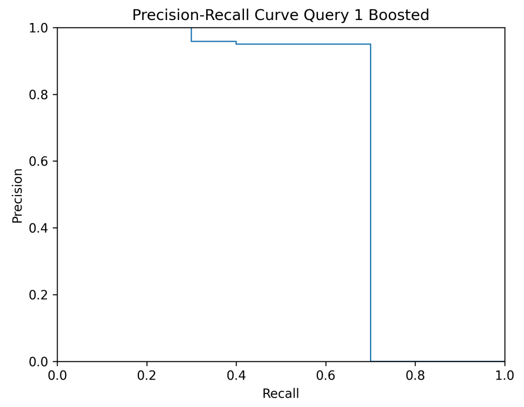

Figure 10: Query 1 metrics of enhanced system

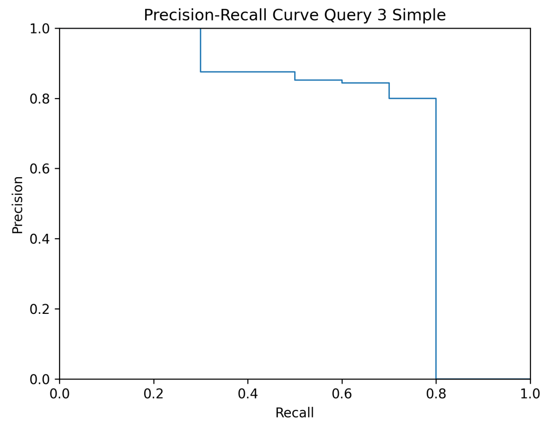

Figure 13: Query 3 metrics of simple system

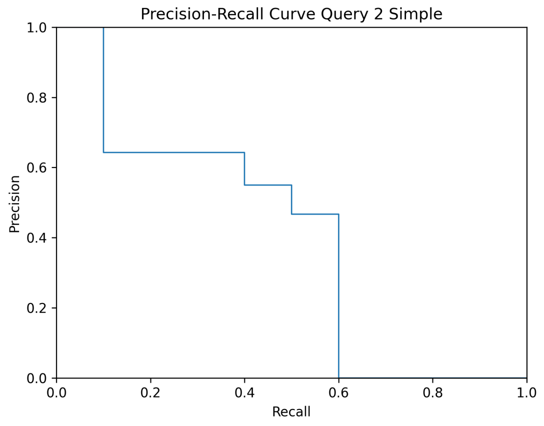

Figure 11: Query 2 metrics of simple system

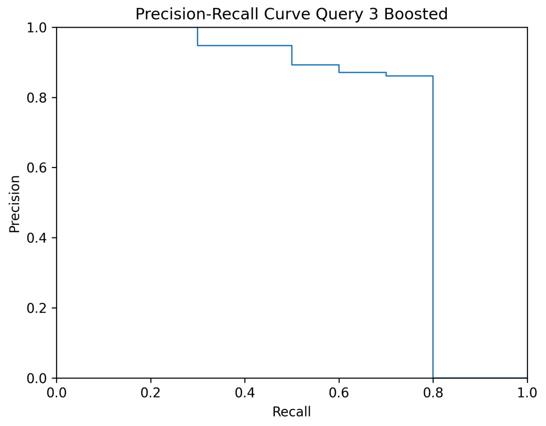

Figure 14: Query 3 metrics of enhanced system

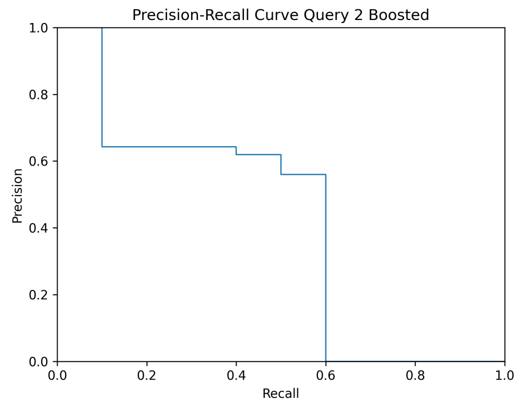

Figure 12: Query 2 metrics of enhanced system

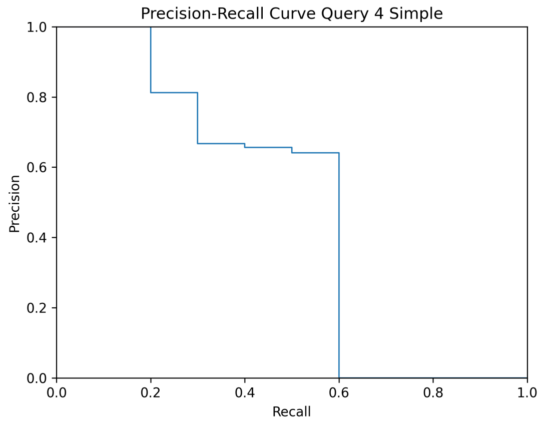

Figure 15: Query 4 metrics of simple system

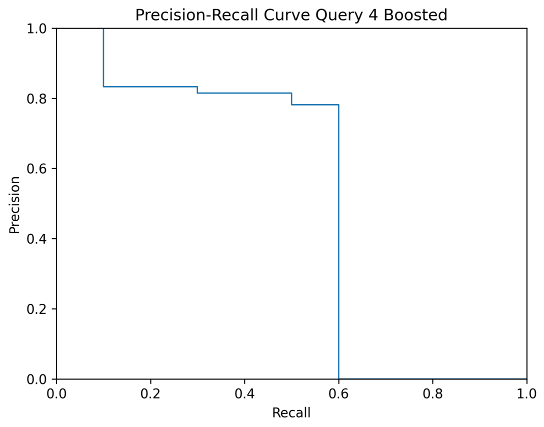

Figure 16: Query 4 metrics of enhanced system

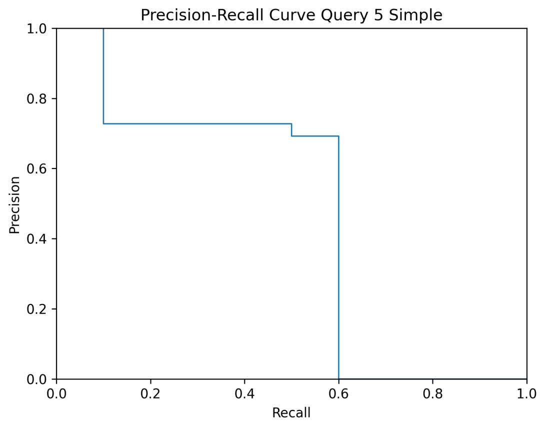

Figure 17: Query 5 metrics of simple system

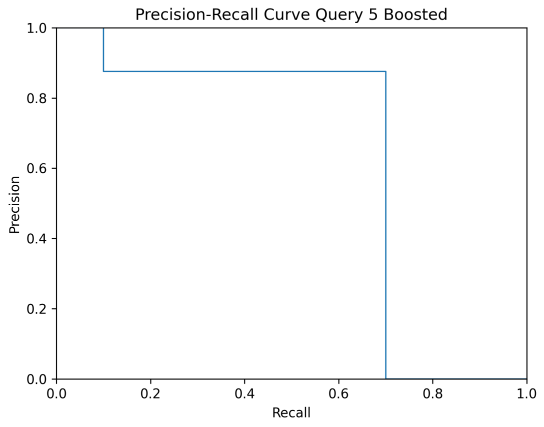

Figure 18: Query 5 metrics of simple system

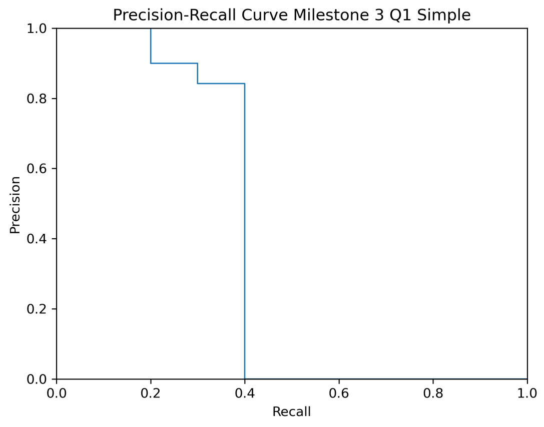

Figure 19: Query 1 P&R-Curve Simple System

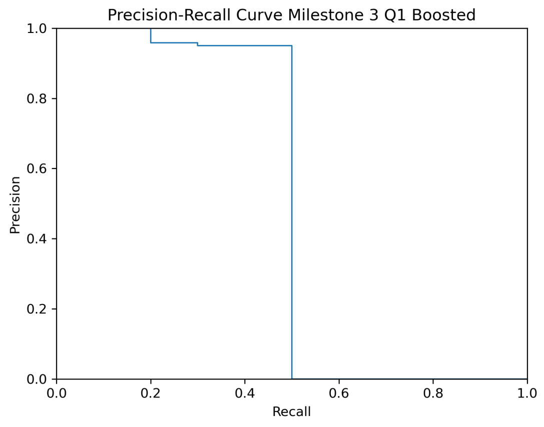

Figure 20: Query 1 P&R-Curve Enhanced System

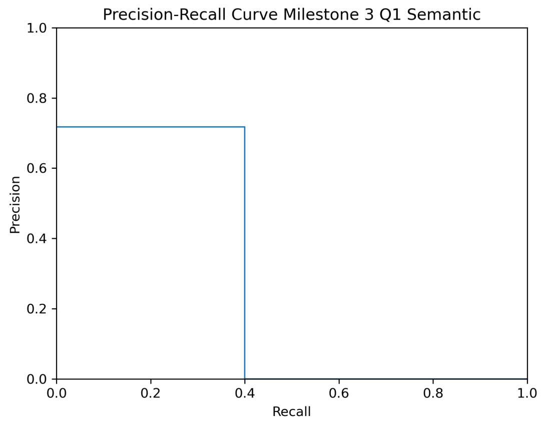

Figure 21: Query 1 P&R-Curve Semantic System

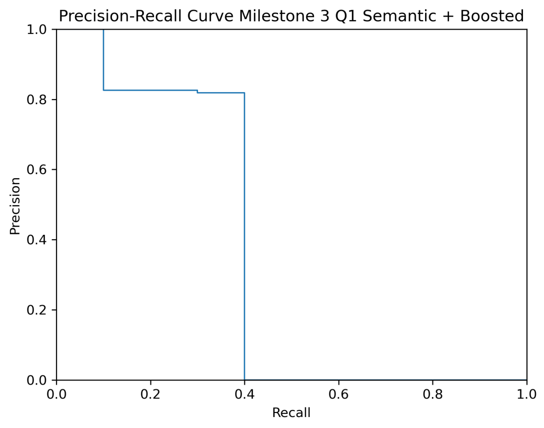

Figure 22: Query 1 P&R-Curve Hybrid System

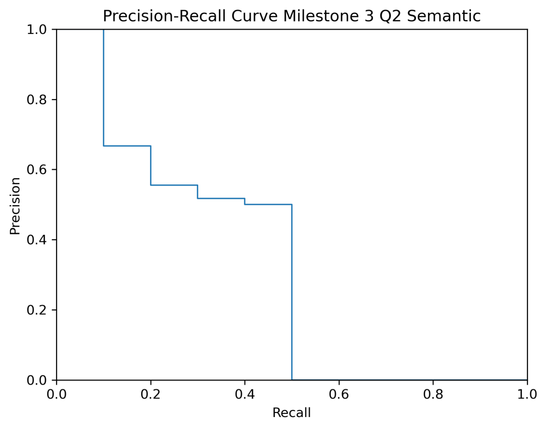

Figure 25: Query 2 P&R-Curve Semantic System

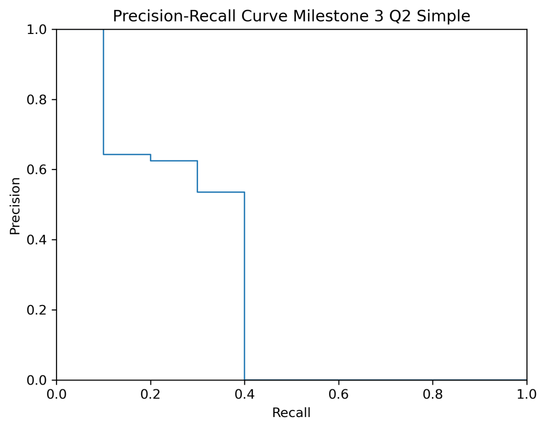

Figure 23: Query 2 P&R-Curve Simple System

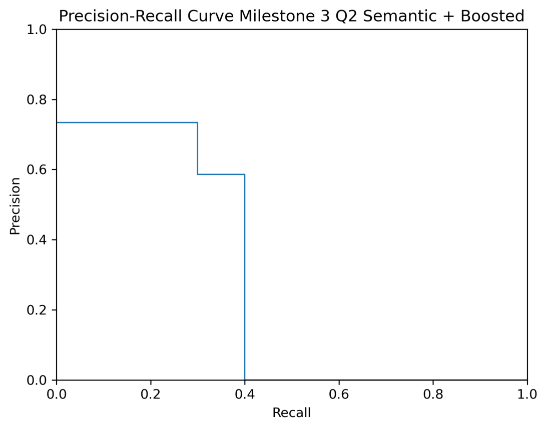

Figure 26: Query 2 P&R-Curve Hybrid System

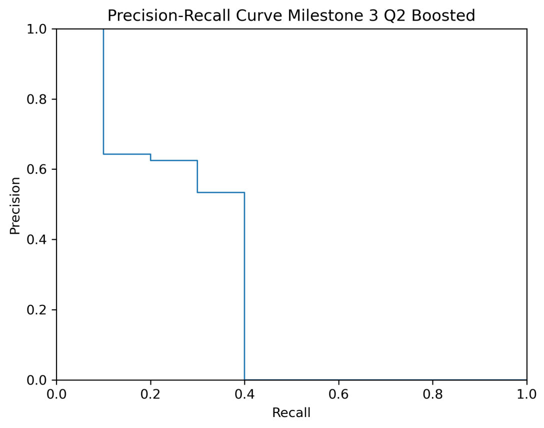

Figure 24: Query 2 P&R-Curve Enhanced System

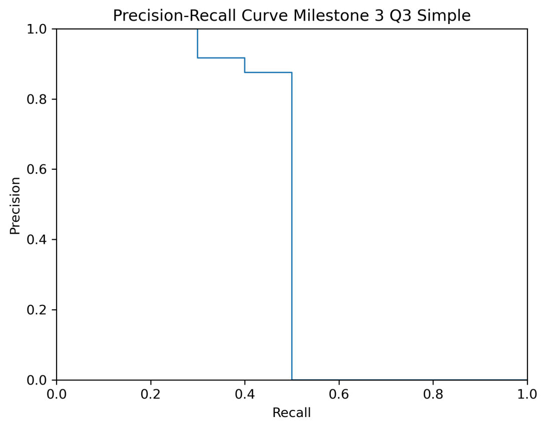

Figure 27: Query 3 P&R-Curve Simple System

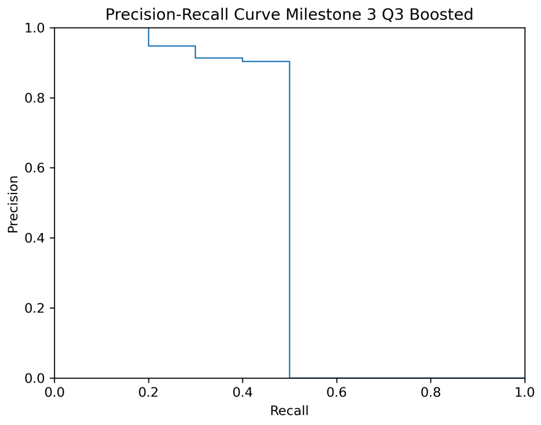

Figure 28: Query 3 P&R-Curve Enhanced System

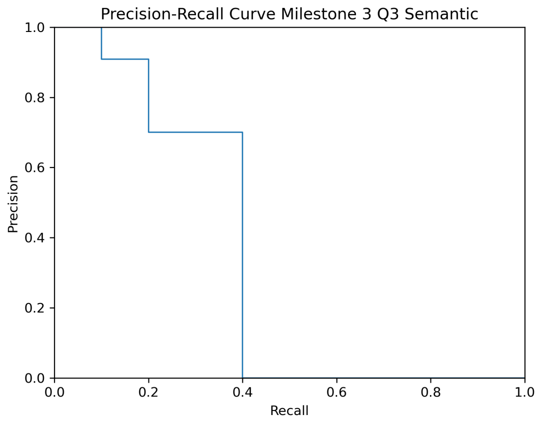

Figure 29: Query 3 P&R-Curve Semantic System

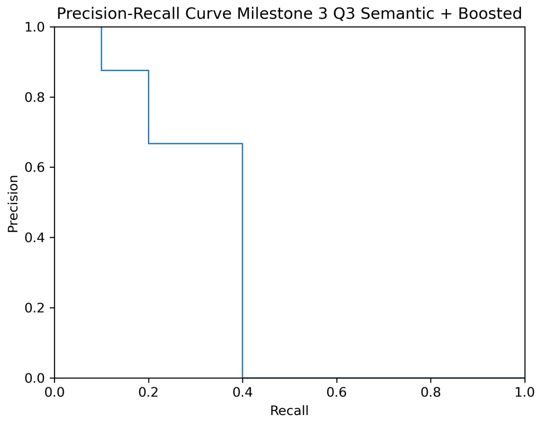

Figure 30: Query 3 P&R-Curve Hybrid System

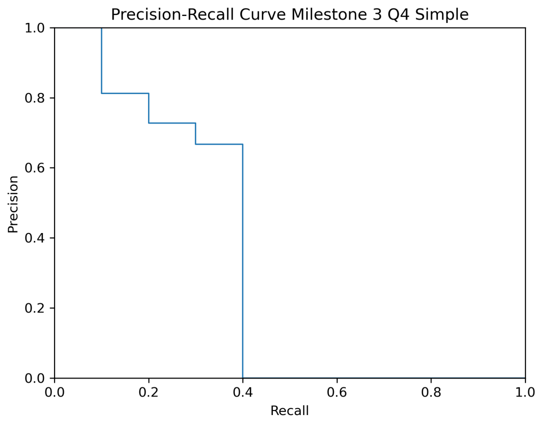

Figure 31: Query 4 P&R-Curve Simple System

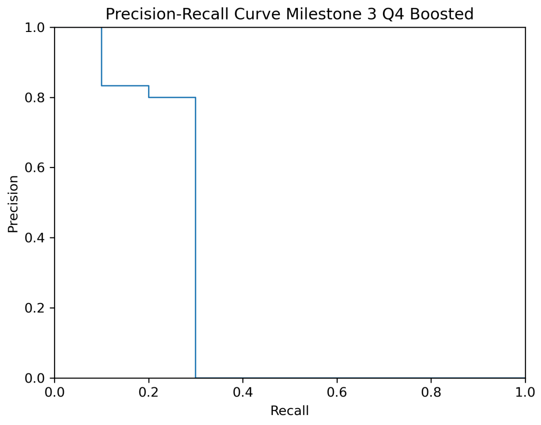

Figure 32: Query 4 P&R-Curve Enhanced System

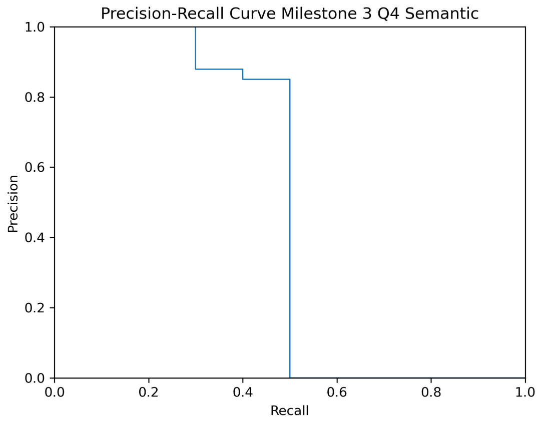

Figure 33: Query 4 P&R-Curve Semantic System

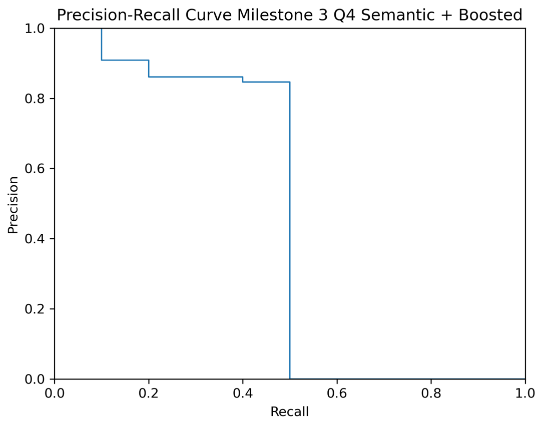

Figure 34: Query 4 P&R-Curve Hybrid System

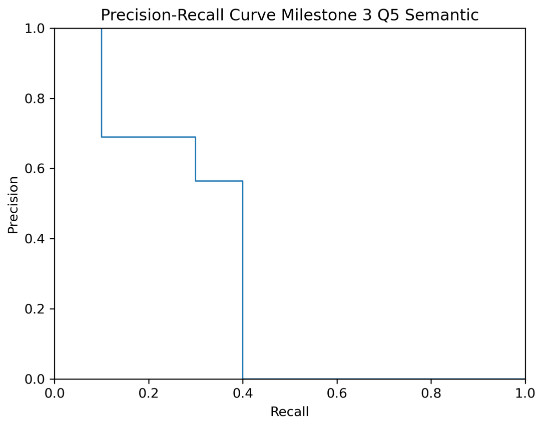

Figure 37: Query 5 P&R-Curve Semantic System

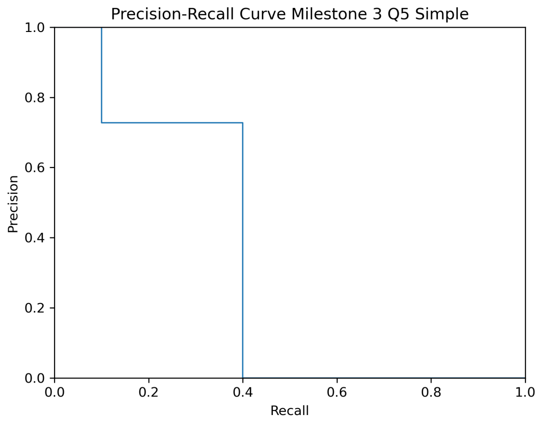

Figure 35: Query 5 P&R-Curve Simple System

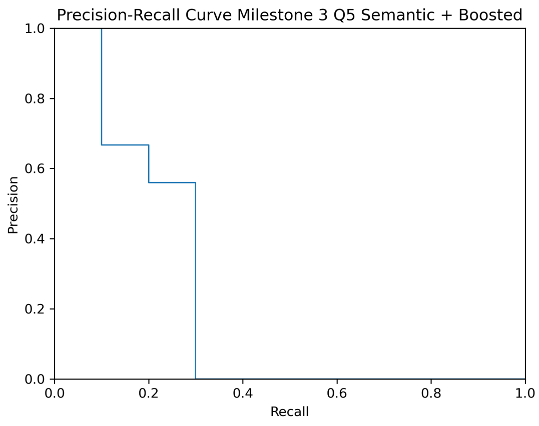

Figure 38: Query 5 P&R-Curve Hybrid System

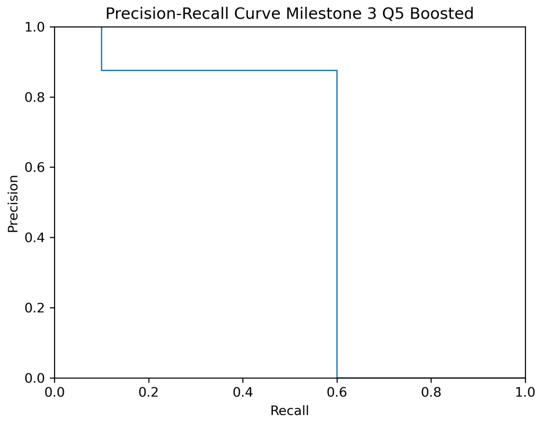

Figure 36: Query 5 P&R-Curve Enhanced System

Query 1 Semantic
query_text: `cars with cruise control and lane keep assist`
semantic_query: `f"{{!knn f=Safety_Feature_Vector topK=40}}{query_text_to_embeddings}"`
Query 1 Hybrid
semantic_query_text: `cars with cruise control and lane keep assist`
query_text: `str("('cruise control') AND ('lane keep assist')")`
lexical_search: `"{!edismax qf='Safety_Feature^7' pf='Safety_Feature^10' ps='4'}" + query_text`
semantic_search: `f"{{!knn f=Safety_Feature_Vector topK=40}}{semantic_query_text_to_embeddings}"`
final_query: `f"{{!bool must="{lexical_search}" must="{semantic_search}"}}"`
Query Excerpt
q: `final_query,`
fl: `"Name, Rating, score"`
rows: `40,`

Table 18: Semantic and Hybrid Query 1

Query 2 Semantic in Practicality Feature
query_text: `family cars with reasonable price`
semantic_query: `f"{{!knn f=Practicality_Feature_Vector topK=40}}{query_text_to_embeddings}"`

Query 2 Semantic in Price Feature
query_text: `family cars with reasonable price`
semantic_query: `f"{{!knn f=Price_Feature_Vector topK=40}}{query_text_to_embeddings}"`

Query 2 Hybrid
semantic_query_text: `family cars with reasonable price`
query_text: `str("(reasonable price) AND ('family car')")`
lexical_search: `"{!edismax qf='Practicality_Feature^7 Price_Feature^7' pf='Practicality_Feature^10 Price_Feature^10' ps='4'}" + query_text`
semantic_search: `f"{{!knn f=(Practicality + Price)_Feature_Vector topK=40}}{semantic_query_text_to_embeddings}"`
final_query: `f"{{!bool must="{lexical_search}" must="{semantic_search}"}}"`

Query Excerpt
q: `final_query,`
fl: `"Name, Rating, score"`
rows: `40`

Table 19: Semantic and Hybrid Query 2

Query 3 Semantic in Driving Feature
query_text: `cars with a comfortable and smooth ride`
semantic_query: `f"{{!knn f=Driving_Feature_Vector topK=40}}{query_text_to_embeddings}"`

Query 3 Semantic in Practicality Feature
query_text: `cars with a comfortable and smooth ride`
semantic_query: `f"{{!knn f=Practicality_Feature_Vector topK=40}}{query_text_to_embeddings}"`

Query 3 Hybrid
semantic_query_text: `cars with a comfortable and smooth ride`
query_text: `str("(comfort ride) AND (smooth ride)")`
lexical_search: `"{!edismax qf='Driving_Feature^7 Practicality_Feature^7' pf='Driving_Feature^10 Practicality_Feature^10' ps='4'}" + query_text`
semantic_search: `f"{{!knn f=(Driving + Practicality)_Feature_Vector topK=40}}{semantic_query_text_to_embeddings}"`
final_query: `f"{{!bool must="{lexical_search}" must="{semantic_search}"}}"`

Query Excerpt
q: `final_query,`
fl: `"Name, Rating, score"`
rows: `40`

Table 20: Semantic and Hybrid Query 3

Query 4 Semantic
query_text: `cars with modern design`
semantic_query: `f"{{!knn f=Design_Feature_Vector topK=40}}{query_text_to_embeddings}"`

Query 4 Hybrid
semantic_query_text: `cars with modern design`
query_text: `str("('modern design')")`
lexical_search: `"{!edismax qf='Design_Feature^7' pf='Design_Feature^10' ps='4'}" + query_text`
semantic_search: `f"{{!knn f=Design_Feature_Vector topK=40}}{semantic_query_text_to_embeddings}"`
final_query: `f"{{!bool must="{lexical_search}" must="{semantic_search}"}}"`

Query Excerpt
q: `final_query,`
fl: `"Name, Rating, score"`
rows: `40`

Table 21: Semantic and Hybrid Query 4

Query 5 Semantic in Driving Feature
query_text: `sport cars with high performance`
semantic_query: `f"{{!knn f=Driving_Feature_Vector topK=40}}{query_text_to_embeddings}"`

Query 5 Semantic in Under The Bonnet Feature
query_text: `sport cars with high performance`
semantic_query: `f"{{!knn f=Under_Bonnet_Feature_Vector topK=40}}{query_text_to_embeddings}"`

Query 5 Hybrid
semantic_query_text: `sport cars with high performance`
query_text: `str("(sport*) AND ('high performance')")`
lexical_search: `"{!edismax qf='Driving_Feature^7 Design_Feature^7' pf='Driving_Feature^10 Under_Bonnet_Feature^10' ps='4'}" + query_text`
semantic_search: `f"{{!knn f=(Driving + Under_Bonnet)_Feature_Vector topK=40}}{semantic_query_text_to_embeddings}"`
final_query: `f"{{!bool must="{lexical_search}" must="{semantic_search}"}}"`

Query Excerpt
q: `final_query,`
fl: `"Name, Rating, score"`
rows: `40,`

Table 22: Semantic and Hybrid Query 5