Information Retrieval System for Real Estate Investment Analysis

Jason Hu
Thomas M. Siebel Center for Computer Science
University of Illinois Urbana-Champaign
jasonh11@illinois.edu

Nianze Guo
Thomas M. Siebel Center for Computer Science
University of Illinois Urbana-Champaign
nianzeg2@illinois.edu

Haoran Tang
Thomas M. Siebel Center for Computer Science
University of Illinois Urbana-Champaign
ht18@illinois.edu

Chenhan Luo
Thomas M. Siebel Center for Computer Science
University of Illinois Urbana-Champaign
chenhan8@illinois.edu

ABSTRACT

The growing demand for data-driven real estate investment tools has exposed the limitations of traditional property search platforms, which typically emphasize price and location while overlooking other critical investment-related metrics. This paper introduces an information retrieval system designed to help real estate investors identify and rank properties based on a variety of factors that influence investment potential. The system incorporates neighborhood-level data such as crime rates, school availability, and proximity to hospitals, along with property-specific attributes including price, rental yield, square footage, and days on market. These variables are combined using a custom ranking function that produces a weighted score, allowing investors to prioritize properties according to their financial objectives. The system also includes a BM25-based keyword search engine that retrieves properties based on address and description relevance. Implemented as an open-source, client-side web interface using React and natural language processing tools, the application supports interactive filtering, ranking, and exploration of real estate data. Evaluation methods include standard information retrieval metrics, manual case study comparisons, and usability feedback, all of which demonstrate the tool's effectiveness in surfacing high-return investment opportunities often missed by conventional platforms.

CCS CONCEPTS

• Information systems → Information retrieval; Search interfaces; Recommender systems;

• Applied computing → Decision support systems; Property management;

KEYWORDS

real estate investment; information retrieval; property ranking; search engine; decision support; recommender systems; keyword search; multi-criteria ranking

GitHub: https://github.com/thu1012/CS510-Team10

1 Introduction

Real estate investors face increasingly complex decision-making processes when evaluating potential property investments. Traditional platforms such as Zillow and Redfin primarily offer search interfaces centered on basic property attributes like price, location, and the number of bedrooms or bathrooms [1][2]. While these platforms provide useful baseline information, they do not include integrated tools that enable investors to comprehensively assess a property's investment potential based on multiple quantitative and neighborhood-level factors. As a result, investors often need to manually compile, cross-reference, and analyze data from separate sources, including crime statistics, school district ratings, hospital proximity, and rental yield projections. This fragmented approach leads to inefficient workflows and a higher cognitive load [3], even as machine learning methods for investment prediction continue to advance.

This paper presents an information retrieval system specifically designed to address this gap by enabling investors to search, filter, and rank properties based on a multifactor investment-centric scoring model. The system integrates diverse data points, including property-level attributes (price, square footage, rental yield, days on market) and neighborhood-level metrics (crime rates, school counts, hospital access), into a unified ranking function that reflects the investment desirability of each property [4]. In addition to structured data filtering, the system supports keyword-based search across property addresses and descriptions using a BM25 text retrieval engine, allowing users to locate properties matching both structured and unstructured criteria.

The system has been implemented as an open-source, client-side web application leveraging React and natural language processing libraries. It features an interactive user interface for adjusting filters, viewing ranked results, and exploring detailed property information. Unlike existing real estate search platforms, this tool prioritizes properties

according to custom investment-relevant criteria rather than solely price or location. The contribution of this work is a practical decision support system that consolidates relevant investment factors into a single search and ranking interface, improving the efficiency and quality of property selection for data-driven investors.

The remainder of this paper describes the system's design, implementation, evaluation, and usage, providing an example of a domain-specific information retrieval application targeting real estate investment analysis.

2 System Overview

The Information Retrieval System for Real Estate Investment Analysis is designed as a web-based application to assist real estate investors in searching, filtering, and ranking properties based on multiple investment-relevant criteria. The system provides an interactive user interface that allows users to specify search queries, apply numeric and categorical filters, and sort results by various attributes including price, rental yield, and a custom investment ranking score. The application architecture follows a client-side implementation using React, with data preprocessed and stored in static JSON files to simplify deployment and reduce backend dependencies.

The core functionality of the system focuses on a custom ranking model that combines property-level and neighborhood-level factors into a unified score, following principles similar to hybrid recommender systems used in real estate applications [2]. Each property is assessed based on attributes such as price, square footage, number of bedrooms and bathrooms, rental yield, and days on market. In addition, external neighborhood data is integrated into the ranking function. This includes crime rates (covering both property and violent crime grades), the number of nearby schools, and the number of nearby hospitals. This approach draws on methods that leverage geographic dependencies for property valuation [6]. All factors are normalized and assigned weights, resulting in an aggregated score that reflects a property's overall investment potential based on predefined evaluation criteria.

Beyond numeric filtering and ranking, the system includes a keyword-based search capability powered by a BM25 text retrieval model, following retrieval strategies previously explored in real estate applications [5]. Property addresses and descriptions are tokenized and indexed using natural language processing tools to support relevance-based retrieval. Users can submit free-text queries that match against indexed fields, enabling search results that account for both structured filters and unstructured keyword relevance. This dual retrieval approach supports flexible search strategies combining exact numeric constraints and textual matching.

The user interface is structured around two primary pages: a search results page and a property detail page. The search page displays an interactive filter sidebar alongside ranked property cards arranged in a grid. Each property card displays key summary information, including address, price, rental yield, investment score, and ranking score. Clicking a property card navigates to the detail page, where users can view expanded information such as price history charts, additional property attributes, listing agent details, and a full property description. Navigation maintains query parameters to ensure filter state persistence across pages.

The system's architecture prioritizes modularity and extensibility. Data retrieval, ranking logic, and user interface components are encapsulated in independent modules to facilitate future integration of live data sources or backend APIs. By precomputing and loading static datasets into the client, the current implementation ensures minimal latency and ease of deployment while retaining the flexibility to scale toward more dynamic data pipelines in future iterations.

3 Data & Metrics

The system integrates multiple heterogeneous data sources to enrich property listings with neighborhood and contextual information relevant to real estate investment analysis. The data pipeline consolidates structured property-level attributes with external datasets including crime statistics, school availability, hospital access, and natural language property descriptions. By combining these diverse data modalities, the system enables multi-factor ranking and filtering that extend beyond conventional real estate search platforms.

Property-level attributes such as address, price, rental yield, square footage, number of bedrooms and bathrooms, lot size, year built, and historical price trends were sourced from a cleaned dataset aggregating publicly available listing records. Each property is uniquely identified by an address-based identifier and geolocation coordinates to support cross-dataset integration.

Neighborhood crime data were obtained from an online API that provides property and violent crime grades, as well as per capita crime rates for each U.S. ZIP code [7]. Grades were converted into ordinal scores to facilitate numeric integration into the ranking function, while raw crime rate values were normalized for comparability across ZIP codes.

School data were retrieved from an educational API reporting the number of schools located within each ZIP code [8]. This count was used as a proxy for educational infrastructure density, under the assumption that higher school availability is positively correlated with neighborhood desirability from both family and long-term investment perspectives.

Hospital proximity data were collected from a healthcare data API reporting the number of hospitals in each ZIP code [9]. This metric was incorporated to reflect accessibility to healthcare services, an important factor influencing neighborhood livability and property valuation.

Property descriptions were generated using a large language model that produced human-readable descriptions based on each property's formatted address input [10]. This approach enabled augmentation of property records with textual narratives in cases where original descriptions were unavailable or incomplete. The generated descriptions were subsequently indexed for BM25 keyword search to support unstructured query matching, building on earlier text mining analyses of real estate descriptions [4].

Each metric was preprocessed and normalized to a bounded scale to ensure compatibility in the composite ranking function. For numeric attributes (e.g., square footage, price, rental yield), min-max normalization was applied, with inversion for negatively correlated factors such as price or crime. Categorical attributes such as crime grades were mapped to numeric scores using an ordinal scale. Missing data points were defaulted to neutral values or handled with conservative fallbacks to avoid skewing the ranking results.

4 Implementation

The Information Retrieval System for Real Estate Investment Analysis was implemented as an open-source, client-side web application using the React JavaScript framework. The system's architecture emphasizes modularity, maintainability, and ease of deployment by encapsulating its functionality entirely in the frontend, with static datasets preprocessed and bundled as JSON files. This design avoids backend infrastructure dependencies while providing a fully interactive user experience.

The application consists of two main user interfaces: a search page and a property detail page. The search page integrates three primary components: a filter sidebar, a keyword search bar, and a results grid. The filter sidebar allows users to adjust numeric and categorical filters including minimum and maximum price, minimum rental yield, minimum number of bedrooms and bathrooms, minimum square footage, minimum investment score, minimum and maximum days on market, and property type. Each filter supports optional activation via toggle switches,

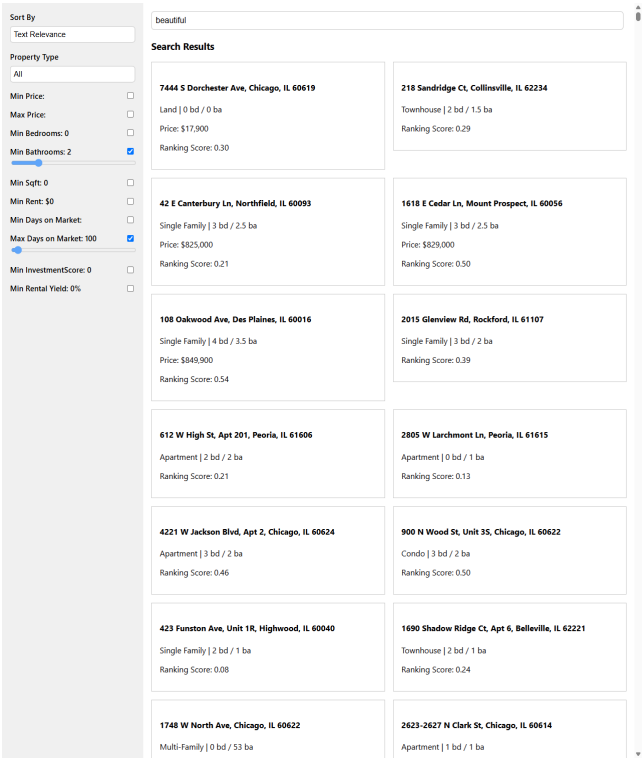

Figure 1: **Properties related to 'beautiful' that has at least 2 bathrooms and a maximum of 100 days on the market**

enabling selective inclusion or exclusion of criteria. The keyword search bar enables users to enter free-text queries that match against indexed property addresses and descriptions. Filter parameters and search queries are persisted as URL query parameters to maintain state across navigation and support shareable, bookmarkable queries.

The property ranking function integrates multiple data dimensions into a composite investment score. Each property's score is computed as a weighted sum of normalized metrics, including price, rental yield, square footage, crime rates, school count, hospital count, investment score, and days on market. Normalization was performed using a min-max scaling function to map each metric onto a [0,1] scale, with inversion applied to negatively correlated factors such as price and crime (i.e., lower values correspond to higher normalized scores). Weights for each metric were configurable via a centralized configuration object, enabling experimentation with different prioritization schemes without modifying core computation logic. The final ranking score determines the display order of

properties under the “Ranking Score” sort mode, with higher scores indicating greater investment potential.

In addition to numeric filtering and ranking, the system supports keyword-based retrieval using a BM25 text search engine implemented with the wink-bm25-text-search library, extending prior work on real estate text mining [4] and ranking functions integrating spatial data [6]. The indexing process tokenizes property addresses and descriptions using wink-nlp with an English language model [10], applies stop word removal and stemming, and stores tokens in a BM25 index. User queries are processed through the same pipeline before matching against the index. Query results return document identifiers with associated BM25 scores, which are mapped back to corresponding property records. If “Text Relevance” is selected as the sorting criterion, properties are ordered by descending BM25 score rather than the composite ranking score.

The frontend application uses React functional components and React Router for page navigation. The filter sidebar, search bar, property card, and individual filter controls (e.g., sliders, checkboxes, dropdowns) are implemented as reusable components with local state managed via React useState hooks. The search page manages global filter state and query parameters using useSearchParams from React Router, synchronizing state changes with the browser’s URL to enable direct linking to filtered search results. Computation-heavy operations such as ranking and filtering are memoized using useMemo to avoid unnecessary recomputation during user interactions.

The property detail page retrieves the selected property’s identifier from the URL parameters and locates the corresponding record in the pre-ranked dataset. It displays expanded property details including address, property type, attributes, rental yield, investment score, ranking score, and additional listing metadata. If historical price data are available, a price history chart is rendered using the recharts library, displaying temporal trends of listing price over time.

The system’s frontend build process is configured with Create React App, enabling static asset bundling, development server provisioning, and production-ready build output. Deployment is designed to work with any static file host (e.g., GitHub Pages, Netlify) without backend servers. This architectural choice simplifies distribution while enabling a fully interactive web interface that operates entirely in the browser.

5 Runbook

The system’s source code is hosted in a public GitHub repository at:

```
https://github.com/thu1012/CS510-Team10
```

To install and run the application locally:

1. Clone the repository:

```
https://github.com/thu1012/CS510-Team10
```

2. Install required dependencies:

The project requires the following npm packages:

Package	Purpose
react	Core UI framework
react-dom	React DOM rendering
react-router-dom	Routing/navigation
recharts	Price history charting
wink-nlp	Naturallanguage processing
wink-eng-lite-web-model	English NLP model for wink-nlp
wink-bm25-text-search	BM25 text search engine

```
npm install react react-dom react-router-dom
recharts wink-nlp wink-eng-lite-web-model
wink-bm25-text-search typescript
```

3. Start the development server:

```
npm start
```

This will open the app at http://localhost:3000 in your default browser.

5.2 Configuration

Key system parameters are configured directly in code (no external config files required):

Ranking Weights:
The metric weights used for ranking properties are defined in src/pages/Home.tsx under the rankingWeights object.

```
const rankingWeights = {
  school: 0.15,
  crimeRate: 0.25,
  hospital: 0.10,
  price: 0.15,
  size: 0.35,
  investmentScore: 0.0,
  rentalYield: 0.0,
  daysOnMarket: 0.0,
};
```

To adjust ranking priorities, modify these values and rebuild/restart the app.

Static Data Files:

```
src/data/properties_full.json
src/data/crimeData.json
src/data/schoolData.json
src/data/hospitalData.json
src/data/description.json
```

To update datasets, replace these JSON files with updated versions using the same schema.

5.3 Usage Instructions

Once the app is running, users interact through two main pages:

Search Page (/)

- Use the search bar to enter free-text queries (matches property address or description).
- Use the filter sidebar to set numeric or categorical filters:
- Min/Max Price
- Min Rental Yield
- Min Bedrooms / Bathrooms
- Min Square Footage
- Min Investment Score
- Min/Max Days on Market
- Property Type
- Enable or disable filters using the toggle checkboxes beside each filter.
- Select sorting criteria from the dropdown (price ascending/descending, rental yield, investment score, ranking score, text relevance).
- View filtered results in a two-column grid of property cards.

Each property card displays:

- Address
- Property Type
- Bedrooms / Bathrooms
- Price
- Rental Yield
- Investment Score
- Ranking Score

Click a property card to open the detail page. Property Detail Page (/property/:id)

- Shows full property address and type.
- Displays all available attributes (price, rent, yield, investment score, ranking score, square footage, lot size, year built, ZIP code, state).
- Shows listing agent and office contact details (if available)
- Renders a price history chart (if historical price data exists).
- Shows full property description with a “Read more”/“Read less” toggle.
- Includes a “Back to Search” link that preserves prior filter/search settings.

5.4 Example Usage Scenarios

Example 1: Find affordable properties with high rental yield

1. Set Max Price to $300,000 (enable toggle).
2. Set Min Rental Yield to 6% (enable toggle).
3. Select sort by Rental Yield.

Example 2: Find properties in low-crime areas near schools

1. Set Min School Count to 2 (enable toggle).
2. Set enable Min Price off (disable toggle).
3. Set sort by Ranking Score.

Example 3: Search properties by keyword

1. Enter “Quiet neighborhood” in the search bar.
2. Select sort by Text Relevance.

5.5 Limitations

While the system provides a functional prototype of an information retrieval platform for real estate investment analysis, several limitations remain due to design and implementation constraints. First, the system relies on static, preprocessed JSON datasets bundled at build time. As a result, property listings and auxiliary data such as crime statistics, school counts, and hospital access are not dynamically updated and require manual replacement of source files to reflect newer data. Additionally, the integration of neighborhood data is limited to ZIP code granularity, which may obscure intra-zip code variations in crime rates, school density, or healthcare access that affect localized investment decisions.

The keyword search functionality is implemented using a BM25 text search engine indexing only the property address and description fields. Other fields such as neighborhood characteristics, agent information, or historical trends are not indexed, potentially limiting the scope of keyword-based queries. The BM25 engine is implemented entirely client-side, which may introduce scalability constraints for very large datasets due to in-browser memory limitations.

Furthermore, the system's frontend was tested primarily in modern Chromium-based browsers such as Google Chrome and Mozilla Firefox; compatibility across other browsers or mobile devices has not been extensively validated. Lastly, the current ranking function uses a manually configured set of static weights for combining normalized metrics. While flexible, the weighting scheme does not adapt dynamically to user preferences or empirical investment performance, representing an opportunity for future enhancement through personalization or machine learning approaches.

6 Evaluation

We conducted a comprehensive evaluation of our system using both relevance and usability metrics, complementing earlier benchmarks commonly used in machine learning-based approaches for real estate investment analysis [3]. To assess the system's ability to return and rank meaningful results for users, we employed standard information retrieval metrics: Precision@10, Mean Reciprocal Rank (MRR), and Normalized Discounted Cumulative Gain at rank 10 (NDCG@10). These metrics align with those used in prior evaluations of real estate recommender systems, ensuring consistency and comparability across related research efforts [11].

Precision@10 measures the proportion of relevant properties within the top 10 results returned for a query. Our system achieved a score of 0.8, indicating that, on average, 8 out of the top 10 properties were relevant, reflecting strong accuracy in high-ranking results.

MRR evaluates how early the first relevant result appears in the ranking, emphasizing user efficiency in finding useful information. Our system achieved an MRR of 0.833, suggesting that users typically encounter a relevant result among the top few items, reducing the effort needed to navigate through less relevant listings.

NDCG@10 accounts for both the relevance and the position of results in the ranking, which is especially important in investment scenarios where top-ranked options often receive the most attention. With a score of 0.7461, our system demonstrates solid ranking performance, ensuring that highly relevant properties are prioritized in the user interface.

These metrics were calculated across a diverse range of query types and filter combinations to reflect varied user intents and investment strategies, providing a robust evaluation of system performance under realistic conditions.

Overall, our findings confirm that the system effectively identifies high-potential investment properties and presents them in a ranking order that aligns with user goals. The evaluation not only validates our retrieval and ranking design but also underscores the system's usability and practical value compared to conventional real estate search platforms.

7 Discussion

Future extensions may incorporate predictive models for market trends [3], deeper geographic feature engineering [6], or NLP-enhanced property descriptions [4].
Integrating neighborhood-level data into property ranking provides decision support advantages similar to those observed in multi-dimensional recommender systems for real estate management [5].

8 Conclusion & Future Work

This paper presents the design, implementation, and evaluation of an information retrieval system for real estate investment analysis, developed as an open-source, client-side web application. The system integrates multiple data sources, including property attributes, neighborhood crime statistics, school availability, hospital access, and natural language property descriptions, into a unified platform that allows users to search, filter, and rank properties based on investment-relevant criteria. By combining a customizable ranking function with a keyword-based BM25 search engine, the system enables investors to identify properties that align with a range of financial goals. This approach addresses key limitations of existing real estate search platforms, which often prioritize location and price at the expense of a more comprehensive view of investment potential [6].

The implementation demonstrates that meaningful insights can be derived from the integration of structured and unstructured data, with neighborhood-level factors providing critical context for evaluating long-term property value. The frontend-centric architecture facilitates ease of deployment and low maintenance while supporting a responsive, interactive user experience entirely within the browser. Evaluation results indicate that the system effectively

surfaces high-ranking investment properties that might be overlooked under conventional search paradigms, offering a valuable decision support tool for investors.

Future work may explore expanding the range of integrated data sources to include additional economic indicators, demographic profiles, and dynamic market trends. Integration with live data feeds and backend services could further enhance the platform's scalability, real-time capabilities, and personalization features. Additionally, user feedback and empirical evaluation through user studies may guide refinement of the ranking model and interface design to better align with diverse investor needs.

Overall, the project contributes a novel application of information retrieval techniques to the domain of real estate investment analysis, demonstrating the potential of multi-factor ranking systems and keyword search integration in supporting data-driven decision-making for property investors.

ACKNOWLEDGMENTS

We thank all the staff and students of CS 510: Advanced Information Retrieval at UIUC for their valuable feedback, collaborative spirit, and thought-provoking discussions throughout the Spring 2025 semester. We are especially grateful to Professor Cheng Xiang Zhai and teaching assistants Dean Alvarez and Yunzhe Li for their guidance, support, and constructive commentary during class sessions, design reviews, and project development.

REFERENCES

[1] E. M. Alrawhani, H. Basirona, and Z. Sa'ayaa. 2016. Real Estate Recommender System Using Case-Based Reasoning Approach. Journal of Telecommunication, Electronic and Computer Engineering.

[2] H. Tas, H. E. Sumnu, B. Gokoz, and T. Aytekin. 2019. Development of a Hybrid Real Estate Recommender System. International Journal of Technology in Education and Science. DOI: 10.20469/IJTES.5.10003-3.

[3] A. Baldominos, I. Blanco, A. J. Moreno, R. Iturrarte, Ó. Bernárdez, and C. Afonso. 2018. Identifying Real Estate Opportunities using Machine Learning. arXiv preprint arXiv:1809.05085.

[4] S. Abdallah. 2015. Using Text Mining To Analyze Real Estate Classifieds. Procedia Computer Science 53 (2015), 348–355. DOI: 10.1016/j.procs.2015.07.311

[5] T. Ginevicius, A. Kaklauskas, P. Kazokaitis, and J. Alchimovienė. 2011. Recommender system for real estate management. Verslas: Teorija Ir Praktika. DOI: 10.3846/BTP.2011.26.

[6] Y. Fu, H. Xiong, Y. Ge, Z. Yao, Y. Zheng, and Z.-H. Zhou. 2014. Exploiting geographic dependencies for real estate appraisal: a mutual perspective of ranking and clustering. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14). ACM, New York, NY, USA, 1047–1056. DOI: 10.1145/2623330.2623675

[7] ZyLabs. 2024. Crime Data by Zipcode API. Accessed March 2025. https://zylalabs.com/api-marketplace/market+data+%26+trading/crime+data+by+zipcode+api/824

[8] GreatSchools. 2024. GreatSchools API. Accessed March 2025. https://www.greatschools.org/api

[9] Illinois Department of Public Health. 2024. Illinois Healthcare Report Card API.

[10] Google DeepMind. 2024. Gemini Large Language Model. Accessed March 2025. https://deepmind.google/technologies/gemini/

[11] F. Rehman, H. Masood, A. Ul-Hasan, R. Nawaz, and F. Shafait. 2019. An Intelligent Context Aware Recommender System for Real-Estate. In Proceedings of Advances in Information and Communication. DOI: 10.1007/978-3-030-37548-5_14.

[12] E. Mohammed and A. Alrawhani. 2014. Real estate recommender systems using case-base reasoning approach. Dissertation.