

# IRWA Project Part 3: Ranking and Filtering Report

---

Authors: Lucas Andreu, Pau Chaves, Pol Bonet, Joan Company

Course: Information Retrieval and Web Analytics (IRWA)

Date: November 20, 2025

## 1. Project Summary

This third part of the project builds on the preprocessing and indexing developed in Parts 1 and 2. Using the inverted index created previously, we focus on the ranking and filtering stage of our fashion-product search engine.

For each user query, we first apply a conjunctive (AND) Boolean filter, so only documents that contain all query terms become candidates. On top of this candidate set, we implement and compare three lexical ranking models:

- TF-IDF + cosine similarity
- BM25
- A custom score that combines TF-IDF relevance with product-specific numeric features (rating, discount, price and stock availability)

In addition, we train a Word2Vec-based semantic ranker, which represents queries and documents as averaged word embeddings and ranks them using cosine similarity. This allows us to analyse how semantic similarity compares to purely lexical models in our domain.

## 2. TF-IDF vs BM25 vs Custom Score

In this section we describe the three ranking functions used on top of the conjunctive candidate set, and we qualitatively compare their behaviour on example queries from the fashion catalogue.

## 2.1 TF-IDF + Cosine similarity

TF-IDF (Term Frequency-Inverse Document Frequency) is a technique used to weigh the importance of words in a document.

- TF (Term Frequency): How many times a word appears in a document (more appearances = more importance).
- IDF (Inverse Document Frequency): Reduces the weight of common words across the entire corpus (e.g., "the," "of") and increases the weight of rare or specific words.

Weights are computed as:

$$tf_{i,j} = 1 + \log_2(f_{i,j}), \quad idf_i = \log_2 \left( \frac{N}{df_i} \right)$$

Cosine Similarity: It measures the similarity between two vectors (query and document) by calculating the cosine of the angle between them. Values close to 1 indicate high similarity (relevant documents); values close to 0 indicate low similarity.

Is computed as:

$$\cos(\vec{q}, \vec{d}) = \frac{\vec{q} \bullet \vec{d}}{|\vec{q}| |\vec{d}|} = \frac{\vec{q}}{|\vec{q}|} \bullet \frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2} \sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

Dot product
Unit vectors

$q_i$  is the tf-idf weight of term  $i$  in the query ( $w_i, q$ )

$d_i$  is the tf-idf weight of term  $i$  in the document ( $w_i, d$ )

*The pros of this approach are:*

- Simple and fast

- Is symmetric and interpretable, a high score is obtained when a document contains query terms with high tf-idf.

*However the cons of tf-idf are:*

- It does not capture synonyms
- Query term frequency has limited impact, as IDF often overshadows term repetition in the query.
- It does not model saturation effect (each additional occurrence keeps increasing the score linearly before the log)

## 2.2 BM25

It is an advanced probabilistic ranking function, an evolution of TF-IDF. Unlike TF-IDF, it models saturation and document length.

Weights are computed as:

$$RSV_d = \sum_{t \in q} \log \left[ \frac{N}{df_t} \right] \cdot \frac{(k_1 + 1)tf_{td}}{k_1((1 - b) + b \times (L_d/L_{ave})) + tf_{td}}$$

N: Number of documents

K1: Tuning parameter controlling the document term frequency scaling

B: Tuning parameter controlling the scaling by document length

*The pros of this approach are:*

- Better term handling: It models saturation, giving diminishing returns for very high term frequency (TF).
- Explicit length normalization: It down-weights very long documents appropriately.
- Typically yields more robust rankings than plain TF-IDF on heterogeneous document collections.

*However the cons of BM25 are:*

- More complex, needs hyperparameters k1 and b (in our case we used the standard ones k1 = 1.5 and b = 0.75)
- Still purely lexical; it cannot capture synonyms or deeper semantic similarity.

## 2.3 Custom Score

Our custom score is designed to combine textual relevance with business-relevant product attributes. It uses the TF-IDF + cosine score as a base and multiplies it by a numeric boost derived from product metadata.

Formally, for a document  $d$  and query  $q$ :  $\text{score}_{\text{custom}}(q,d) = \text{score}_{\text{tfidf}}(q,d) \times \text{boost}(d)$

The boost function  $\text{boost}(d)$  is computed using:

- Average rating (normalized to  $[0,1]$ )
- Discount percentage (scaled so large discounts give higher boosts)
- Price (slight preference for cheaper items)
- Stock availability (in-stock products are favored; out-of-stock products are penalized)

All factors are clipped to avoid extreme values, so the boost stays in a reasonable range and does not completely override textual relevance.

*The pros of this approach are:*

- Incorporates business logic directly into ranking: promotes highly rated, discounted and in-stock products that are also textually relevant.
- Flexible: new numeric features can be added without changing the underlying index.
- Still interpretable, since the final score decomposes into text relevance  $\times$  metadata boost.

*However the cons of Custom Score are:*

- Requires feature engineering and manual scaling of numeric fields.
- Sensitive to noisy metadata (e.g., unreliable ratings or mis-labelled stock status).
- The multiplicative combination assumes independence between factors and may not be optimal without further tuning or learning-to-rank.

## 2.4 Results Comparison

To compare the behaviour of the three ranking functions, we inspect the top-k results for representative queries from the fashion domain (e.g. “women full sleeve sweatshirt cotton” and “men slim jeans blue”).

1. TF-IDF + Cosine Similarity:

Query: *women full sleeve sweatshirt cotton*

Product ID	Score	Title
------------	-------	-------

SWSFZVTTQCB4SJ7F	0.7305	Full Sleeve Solid Women Sweatshirt
SWSFQGS456JAZCQB	0.6561	Full Sleeve Printed Women Sweatshirt
SWSFZVTNGM4HG8BC	0.6339	Full Sleeve Printed Women Sweatshirt
SWSFYTYMNTBNARUN	0.6237	Full Sleeve Solid Women Sweatshirt
SWSFYRKYAHH4HHSM	0.6070	Full Sleeve Printed Women Sweatshirt

- The top-ranked items all have titles containing “Full Sleeve ... Women Sweatshirt”, showing TF-IDF’s strong bias toward exact lexical overlap, especially in short titles.
- The word “cotton” hardly affects the ranking. It appears mostly in descriptions where its TF-IDF weight is low, so items labeled as cotton do not rise to the top.
- The scores decrease smoothly, indicating that many sweatshirts have near-identical titles, and TF-IDF distinguishes them mostly by minor term-frequency differences.

Query: *men slim jeans blue*

JEAFRAQXEKGUPNUN	0.4843	Slim Men Blue Jeans
JEAFSKYHZHSZZC9S	0.4559	Slim Men Blue Jeans
JEAFQF6JBUSEXHVF	0.4447	Slim Men Blue Jeans
JEAFSKYHRVZSABPR	0.4423	Slim Men Blue Jeans
JEAFSKYHTE76YWH7	0.4423	Slim Men Blue Jeans

- All top-ranked items are exactly titled “Slim Men Blue Jeans”, showing that TF-IDF strictly requires exact matching of “slim”, “men”, “jeans”, and “blue”.
- The results show very limited diversity; many jeans share almost identical titles, causing TF-IDF scores to cluster tightly.
- Variants such as dark blue, light blue, tapered, or skinny jeans never appear, as the method cannot generalize beyond literal token overlap.

## 2. BM25:

Query: *women full sleeve sweatshirt cotton*

Product ID	Score	Title
SWSFVZRFS7GHGKSF	10.2591	Full Sleeve Solid Women Sweatshirt
SWSFYFFFFYZ896TJ	9.9860	Full Sleeve Printed Women Sweatshirt
SWSFXMFDPVRHYYPH	9.9796	Full Sleeve Striped Women Sweatshirt
SWSFXMFPPZGDQGMW	9.9796	Full Sleeve Striped Women Sweatshirt
SWSFYFFYQ7Z3ZKN6	9.9541	Full Sleeve Printed Women Sweatshirt

- BM25's top results also match the query well, but the list includes Striped Women Sweatshirts that TF-IDF did not rank as highly. BM25 rewards repeated occurrences of query terms in longer descriptions.
- Two striped versions appear back-to-back with almost identical scores, reflecting BM25's sensitivity to term saturation.
- The ordering differs more from TF-IDF because BM25 uses document-length normalization, which can boost items with richer descriptions even if their titles are less perfectly aligned.

Query: *men slim jeans blue*

JEAEVJGUSXRNSHRY	10.2389	Slim Men Dark Blue Jeans
JEAFSKYHRVZSABPR	10.1051	Slim Men Blue Jeans
JEAFSKYHTE76YWH7	10.1051	Slim Men Blue Jeans
JEAEHGRJSGGYEYYX	10.0824	Slim Men Light Blue Jeans
JEAFWZXTFGMP9GCN	9.9718	Slim Men Blue Jeans

- BM25 introduces more variation: the top results include Dark Blue and Light Blue versions of slim jeans, which TF-IDF overlooked.
- This occurs because BM25 benefits from richer descriptions where colour-related terms appear more frequently.
- Slim Blue Jeans items still appear in top ranks, but BM25 orders them differently, reflecting a balance of title match and descriptive detail.

### 3. Custom Score

Query: *women full sleeve sweatshirt cotton*

Product ID	Score	Title
SWSFZVTNGM4HG8BC	1.2278	Full Sleeve Printed Women Sweatshirt
SWSFYTYMNTBNARUN	1.2057	Full Sleeve Solid Women Sweatshirt
SWSFY38ADYPVZHYZ	1.1757	Full Sleeve Printed Women Sweatshirt
SWSFY382UZHFBNCB	1.1277	Full Sleeve Printed Women Sweatshirt
SWSFZ2H4KMYXZXX7	1.1058	Full Sleeve Solid Women Sweatshirt

- Sweatshirts with better ratings, discounts, or stock availability rise noticeably compared to TF-IDF. For example, printed sweatshirts move to the top even when they were mid-ranked in the lexical methods.
- Since many sweatshirts share nearly the same title, the metadata boost becomes the main factor differentiating products with similar textual relevance.
- The custom score still retrieves the same general type of product (full-sleeve women's sweatshirts), but now prioritizes those that are more attractive from a business or user perspective.

Query: *men slim jeans blue*

JEAFQF6JBUSEXHV	0.7882	Slim Men Blue Jeans
JEAFSKYHZHSZZC9S	0.7670	Slim Men Blue Jeans

JEAFSKYHRVZSABPR	0.7660	Slim Men Blue Jeans
JEAFESND4QWQUBZD	0.7660	Slim Men Blue Jeans
JEAFEKQZ4C2Z6GCX	0.7643	Slim Men Blue Jeans

- The custom ranking keeps the same general set of “Slim Men Blue Jeans,” but rearranges them based on ratings, discounts, and stock.
- Items with stronger metadata outperform those that had slightly higher lexical similarity.
- Because the AND filter requires all query terms, and the content is very uniform, metadata becomes the decisive factor in ranking.
- This model effectively answers: “Which of the strongly relevant jeans are best for the user?”

### 3. Word2Vec + Cosine Similarity

#### 3.1 Overview

Word2Vec allows us to capture semantic similarity through the representation of words as numerical vectors (embeddings). To obtain the embedding for the entire document instead of just the words, the word vectors in each document are averaged.

$$\text{doc\_vec} = \frac{1}{|d|} \sum_{w \in d} \mathbf{v}(w)$$

$$\text{query\_vec} = \frac{1}{|q|} \sum_{w \in q} \mathbf{v}(w)$$

Then we proceed to calculate the score using cosine similarity.

#### Pipeline:

1. The query is preprocessed and its average Word2Vec vector is calculated, which is then compared via cosine similarity to the precomputed average Word2Vec vector of every document to determine relevance.
2. The top-20 documents with the highest similarity score are then returned. This approach allows for retrieval when exact words do not match, relying instead on the related meanings of the query and document content.

## 3.2 Training

A skip-gram Word2Vec model was trained using all tokenized text from three specific fields, resulting in word embeddings specialized for fashion product text. All documents were cached to allow fast search.

The text fields used for training were:

- Title\_clean
- Description\_clean
- metadata\_clean

Model hyperparameters used:

- Vector size: 100
- Window: 5
- Min\_count: 2
- Workers: 4
- Sg: 1 (skip-gram mode)

## 3.3 Results

The model was evaluated on the five queries defined in Part 2. Below are two representative examples showing the Top-5 ranked items returned by Word2Vec + cosine similarity.

Query: *women full sleeve sweatshirt cotton*

Product ID	Score	Title
SWSFY5ZHUEZPZZYV	0.8426	Full Sleeve Printed Women Sweatshirt
SWSF9W528G7VEGCV	0.8423	Full Sleeve Striped Women Sweatshirt
SWSFY5ZHEJ2HYWDG	0.8421	Full Sleeve Printed Men Sweatshirt
SWSF9W5YHFAAHNJZ	0.8404	Full Sleeve Striped Women Sweatshirt
SWSFMJF98EY2FXBH	0.8377	Full Sleeve Solid Women Sweatshirt

The model captures semantic similarity: products with full sleeve, sweatshirt, and women dominate the top results, even if the word cotton does not always appear explicitly.

Interestingly, one men's sweatshirt appears in the top-5, showing that Word2Vec semantic similarity can blur fine-grained distinctions (gender in this case) when many terms overlap.

Query: *men slim jeans blue*

Product ID	Score	Title
JEAFSKYHRVZSABPR	0.7829	Slim Men Blue Jeans
JEAFSKYHTE76YWH7	0.7829	Slim Men Blue Jeans
JEAFSKYH539HTZB8	0.7803	Tapered Fit Men Blue Jeans
JEAFXUHCWV9C5WNX	0.7783	Super Skinny Men Blue Jeans
JEAFXUHA5YF8WYQY	0.7780	Tapered Fit Men Blue Jeans

The method returns highly relevant jeans products: slim, tapered, skinny, and blue, demonstrating how averaging embeddings preserves overall semantic meaning.

### Pros and Cons of Word2Vec Ranking:

#### Pros

- Captures semantic similarity beyond exact text matching.
- Robust to vocabulary variation (e.g., hoodie vs sweatshirt).
- Performs well on short queries and descriptive product text.
- Lightweight and fast to compute once trained.

#### Cons

- Averaging word vectors loses word order and syntax information.
- Important terms may be diluted by long descriptions.
- Cannot distinguish subtle attributes (e.g., women vs men).
- No contextual understanding (Word2Vec is non-contextual).

## 3.4 Improvements

To improve our search, we could use other types of embeddings like Doc2Vec or Sentence Embeddings, which are not limited like Word2Vec. This is because Word2Vec's averaging of word vectors causes the semantic meaning of the phrase to be lost, which is crucial for obtaining good results. In contrast, these two are designed to capture this very thing.

Doc2Vec extends Word2Vec by generating a single vector for the entire document instead of just individual words.

Pros:

- Captures the semantics of the whole document.
- More accurate than Word2Vec averaging because it models intra-document relationships.

Cons:

- Requires more training data and is more computationally expensive than Word2Vec.

Another way could be Sentence Embeddings, where models generate dense embeddings optimized for semantic similarity using advanced techniques like transformers (e.g., BERT).

Pros:

- Captures bidirectional context and complex relationships (synonyms, polysemy).
- Excellent for short, semantic queries.
- Can use pre-trained models, reducing the need for custom training.

Cons:

- More resource-intensive (often requires a GPU).
- Less interpretable than Word2Vec.

## 4. Conclusions

When comparing all ranking models (TF-IDF, BM25, Custom Score, and Word2Vec):

- TF-IDF performs well for keyword-heavy queries but lacks semantic understanding.
- BM25 consistently outperforms TF-IDF thanks to term saturation and length normalization.
- Our custom score allows domain-specific weighting but requires careful tuning.

**Github Repository:** <https://github.com/lucassandreu/irwa-search-engine>

**Tag Delivery 3:** <https://github.com/lucassandreu/irwa-search-engine/releases/tag/IRWA-2025-part-3>

- Word2Vec introduces semantic retrieval, retrieving relevant items even when vocabulary differs, although it may lose precision on fine-grained distinctions.

Overall, combining lexical models (TF-IDF/BM25) with semantic embeddings (Word2Vec, Sentence Transformers) would yield the most robust search engine.

### **AI Usage Policy:**

We used ChatGPT to review report definitions, pros and cons of the ranking methods, and to know possible ways to improve Word2Vec ranking.