Algorithms for Massive Data - Market-basket Analysis

Antonio De Patto - 46430A

August 31, 2025

Abstract

I/We declare that this material, which I/We now submit for assessment, is entirely my/our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my/our work, and including any code produced using generative AI systems. I/We understand that plagiarism, collusion, and copying are grave and serious offenses in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me/us or any other person for assessment on this or any other course of study.

1 Introduction and Dataset Presentation

The goal of this project is to implement a system for discovering frequent item-sets and association rules, also known as market-basket analysis. We consider as baskets the textual reviews from the 'Books_rating.csv' dataset available on Kaggle. The chosen dataset contains approximately 3 million reviews for 212.404 unique books. Due to the dataset size, we initially applied sampling, considering 1% of the whole dataset (30.332 reviews).

More about the Market-basket Analysis Market-basket analysis is a data mining technique used to discover associations and correlations between items in transactional datasets. The fundamental concepts include frequent item-sets, which are sets of items that appear together in a sufficiently large number of transactions, and association rules, which capture the likelihood of occurrence of one item given the presence of others. The strength of these rules is typically evaluated using metrics such as support (the proportion of transactions containing the item-set), confidence (the probability that the consequent occurs given the antecedent), and lift (the ratio of observed co-occurrence to expected co-occurrence if items were independent).

In this project, the market-basket model is applied to textual reviews by treating words and meaningful bigrams as items in each basket. Mining frequent patterns in this context allows the identification of common word co-occurrences that may indicate sentiment, themes, or preferences in customer reviews.

2 Data Pre-processing Techniques

Textual reviews are noisy and unstructured, containing typos, punctuation, varied word forms, and irrelevant terms. To effectively use them for frequent item-set mining, it is crucial to perform several pre-processing steps that transform the raw text into a clean, structured representation:

1. **Tokenization**: This step splits each review into individual tokens using regular expressions. Tokenization is fundamental process of converting a sequence of text into smaller parts, known as tokens. These tokens can be as small as characters or as long as words. The primary reason this process matters is that it helps machines understand human language by breaking it down into bite-sized pieces, which are easier to analyze. In this case Word Tokenization is applied by breaking text into individual words.

2. **Lemmatization**: Lemmatization is the reduction of inflected words to their common root, or lemma, decreasing sparsity in the dataset and improving the statistical significance of frequent patterns. This step is particularly important in textual market-basket analysis, where slight variations in word forms can artificially fragment the occurrence counts of semantically identical items. In this projects the lammatization will be applied to the nouns, transforming words like 'books' into 'book' or 'stories' into 'story'.

3. **Stopword Removal**: Common words such as 'the', 'is', 'and' occur in almost every review but carry little semantic information for pattern mining. Removing stopwords reduces noise, decreases dimensionality, and improves the efficiency of pattern mining by focusing on informative tokens.

4. **Keep and Ban Tokens**: Certain words with strong sentiment polarity, such as good, excellent, or terrible, were explicitly retained regardless of their frequency, ensuring that sentiment-related patterns are preserved. Conversely, meaningless or overly generic terms, which could lead to spurious associations, were banned. This selective filtering helps the algorithm concentrate on tokens capable of expressing a positive or negative evaluation of a book, which can be useful in identifying meaningful patterns within reviews.

5. **Frequent Words Removal**: In addition to predefined stopwords, the top k most frequent tokens in the dataset were automatically identified and filtered out if they were not part of the keep list. This adaptive approach allows the system to dynamically remove high-frequency terms that do not contribute to distinguishing itemsets, further enhancing the quality of discovered patterns. In our case, the 50 most frequent words were automatically removed, unless they were included in the list of words to keep for sentiment analysis or other words we retain important, as 'character', 'novel', 'story' and 'author', to which the sentiment can be related to.

6. **Bigrams**: Many semantic concepts in text are composed of multiple words, such as easy_read or highly_recommended. Meaningful bigrams were extracted to capture these compound expressions, which are treated as single tokens in the basket. Including bigrams allows the model to discover associations between multi-word phrases that carry richer meaning than individual tokens alone. This prevents the extraction of uninformative associations, like the obvious correlation between 'high' and 'recommended', that is now transformed into a single token 'high_recommended'. Bigrams are retained in the final basket only if at least one of their two words appears in the list of terms deemed important for sentiment analysis. This list primarily includes adjectives that convey a positive evaluation, such as good or amazing, as well as adjectives expressing a negative evaluation, such as bad or dull. Consequently, bigrams like good_book or dull_character will be included in the bigram set, whereas bigrams such as last_year will be discarded, since neither of their words is present in the list of terms to retain.

After all pre-processing steps, each review is represented as a structured basket containing tokens and meaningful bigrams. This transformation converts unstructured natural language into discrete items suitable for frequent pattern mining.

3 Algorithm Functioning and Implementation

For frequent pattern mining we adopt the Frequent Pattern Growth algorithm. Its functioning is based on two main phases:

1. **Data Compression**: First FP-Growth compresses the dataset into a smaller structure called the Frequent Pattern Tree (FP-Tree). This tree stores information about item sets (collections of items) and their frequencies without need to generate candidate sets like Apriori does.

2. **Mining the Tree**: The algorithm then examines this tree to identify patterns that appear frequently based on a minimum support threshold. It does this by breaking the tree down into smaller "conditional" trees for each item making the process more efficient.

3. **Generating Patterns**: Once the tree is built and analyzed the algorithm generates the frequent patterns (itemsets) and the rules that describe relationships between items.

Thanks to this structure, FP-Growth requires only two passes over the dataset, leverages prefix compression, and significantly reduces memory usage compared to alternative algorithms like Apriori, PCY, multistage and multihash hashing-based techniques, randomized approaches such as Toivonen or SON. However, these either require multiple dataset scans, rely on approximations, or involve higher computational and communication overheads. In contrast, FP-Growth offers an exact, memory-efficient, and Spark-native solution, making it the most suitable choice for large-scale textual baskets.

3.1 The parameters of the model

The parameters related to the minimum support and confidence play a central role in controlling the behavior of the FP-Growth algorithm and the quality of the resulting patterns.

- **Minimum Support Threshold**: specifies the minimum proportion of baskets in which an itemsets must appear to be considered frequent. In our implementation, the Minimum Support Threshold is equal to 0.01 means that a token or bigram must appear in at least 1% of the sampled reviews to be included in the frequent itemsets. This threshold balances between discovering meaningful patterns and avoiding extremely rare combinations that may not be statistically significant. Lower values produce a very large number of itemsets, increasing computation time and noise, while higher values could discard interesting associations.
- **Minimum Confidence Threshold**: controls the strength of the association rules generated from frequent itemsets. A rule of the form $A \rightarrow B$ is kept only if the conditional probability of B given A exceeds the threshold. By setting the Minimum Confidence Threshold equal to 0.2, we allow rules where at least 20% of the baskets containing the antecedent also contain the consequent, filtering out weak associations while retaining moderately strong and interpretable patterns.

These values were selected after preliminary experimentation on a sample of the dataset (1% of the total reviews) to ensure that the number of frequent itemsets and association rules remained manageable for visualization and analysis, while still capturing relevant co-occurrences and sentiment-related patterns across the reviews.

4 Analysis and Results

The pipeline consists of three main phases. First, textual pre-processing is applied: reviews are tokenized and lemmatized, converted to lowercase, and stripped of standard and domain-specific stop-words. Overly frequent words are filtered out, while sentiment-relevant terms, like 'good', 'recommend' or 'bad' are preserved.

```
Bigrams found: 515
Single token: 1993
Bigram examples: ['able_wonderful', 'action_must', 'actor_really', 'actuality_really', 'add_much']
```

Figure 1: Bigrams and Tokens found by the Algorithm

To capture compound expressions, meaningful bigrams are generated and merged with the remaining tokens. Finally, duplicates are removed yielding the final baskets of items. Some examples found by the algorithm can be found in figure 1. To be precise, 515 bigrams are considered important because one of the two words is contained in the list of words considered important for sentiment analysis.

Text Processing As illustrated in Figure 2, the complete sequence of pre-processing steps leading to the construction of the final basket is reported. Starting from the original 'review/text' column, the text was first tokenized by splitting words according to non-alphabetic characters. Next, lemmatization was applied at the noun level (stories $\rightarrow$ story, books $\rightarrow$ book), followed by the removal of stop words such as 'is' and 'the'. Subsequently, less relevant tokens were filtered out both on the basis of frequency (while still retaining terms such as 'author', 'story', 'novel' and 'character') and by means of a list that serves as an additional safeguard. Finally, bigrams were generated and filtered: each bigram

is preserved only if at least one of its components belongs to the list of words considered important for sentiment analysis. We observe that the procedure produces highly informative bigrams such as 'didnt_like', 'truly_great', or 'everyone_like', which capture relevant aspects of sentiment expressed in the reviews. For example, 'truly_great' emphasizes the concept of satisfaction even more than 'great'. Similarly, 'didnt_like' allows us to save a token as negative that would otherwise have been incorrectly classified as positive, thus interfering with the distribution of words expressing a positive sentiment. On the other hand, some bigrams such as *lebron_high* are not meaningful for our analysis. Nevertheless, due to their uniqueness and low frequency, these latter cases do not significantly influence the overall results. Through this procedure, we obtained the final basket that can be directly passed to the model.

review/text	tokens	tokens_lem	tokens_nostopwords
"LeBron James is ...	[lebron, james, i...	[lebron, james, i...	[lebron, james, b...
This book is a cl...	[this, book, is, ...	[this, book, is, ...	[book, classic, p...
The plot and char...	[the, plot, and, ...	[the, plot, and, ...	[plot, character,...
Although the titl...	[although, the, t...	[although, the, t...	[although, title,...
This book is the ...	[this, book, is, ...	[this, book, is, ...	[book, best, funn...
There is no other...	[there, is, no, o...	[there, is, no, o...	[book, africa, ar...
"According to the...	[according, to, t...	[according, to, t...	[according, autho...
Hazlitt is one of...	[hazlitt, is, one...	[hazlitt, is, one...	[hazlitt, one, pr...
Teacher no longer...	[teacher, no, lon...	[teacher, no, lon...	[teacher, longer,...
"Economics in One...	[economics, in, o...	[economics, in, o...	[economics, one, ...
A great and acces...	[a, great, and, a...	[a, great, and, a...	[great, accessibl...
I have to laugh a...	[i, have, to, lau...	[i, have, to, lau...	[laugh, negative,...
Henry Hazlitt was...	[henry, hazlitt, ...	[henry, hazlitt, ...	[henry, hazlitt, ...
i enjoy folk/fair...	[i, enjoy, folk, ...	[i, enjoy, folk, ...	[enjoy, folk, fai...
I was a big fan o...	[i, was, a, big, ...	[i, wa, a, big, f...	[wa, big, fan, ja...
A terrible, child...	[a, terrible, chi...	[a, terrible, chi...	[terrible, childi...
I don't know who ...	[i, don, t, know,...	[i, don, t, know,...	[know, worse, hig...
Our four-year-old...	[our, four, year,...	[our, four, year,...	[four, year, old,...
this book is womd...	[this, book, is, ...	[this, book, is, ...	[book, womderful,...
I am truly amazed...	[i, am, truly, am...	[i, am, truly, am...	[truly, amazed, o...

tokens_filtered	bigrams_raw	meaningful_bigrams	basket
[lebron, james, b...	[lebron james, ja...	[lebron_high, hig...	[james, basketbal...
[classic, pertain...	[classic pertains...	[partner_much, mu...	[classic, pertain...
[plot, character,...	[plot character, ...	[everyone_like, l...	[plot, character,...
[although, title,...	[although title, ...	[century_interest...	[although, title,...
[best, funny, fel...	[best funny, funn...	[best_funny, felt...	[laugh, head, exa...
[africa, arabia, ...	[africa arabia, a...	[arabia_like, lik...	[africa, ramble, ...
[according, autho...	[according author...	[]	[according, autho...
[hazlitt, promine...	[hazlitt prominen...	[contains_lot, lo...	[hazlitt, promine...
[teacher, longer,...	[teacher longer, ...	[came_great, grea...	[teacher, longer,...
[economics, lesso...	[economics lesson...	[simply_must, mus...	[economics, lesso...
[great, accessibl...	[great accessible...	[great_accessible...	[nothing, topic, ...
[laugh, negative,...	[laugh negative, ...	[thinking_best, b...	[laugh, negative,...
[henry, hazlitt, ...	[henry hazlitt, h...	[truly_great, gre...	[henry, hazlitt, ...
[enjoy, folk, fai...	[enjoy folk, folk...	[enjoy_folk, esp_...	[fairytale, gener...
[big, fan, jack, ...	[big fan, fan jac...	[adventure_like, ...	[big, fan, jack, ...
[terrible, childi...	[terrible childis...	[terrible_childis...	[absolutely, rede...
[worse, higgins, ...	[worse higgins, h...	[]	[worse, higgins, ...
[four, old, dog, ...	[four old, old do...	[able_wonderful, ...	[four, old, dog, ...
[womderful, enjoy...	[womderful enjoye...	[didnt_like, like...	[womderful, enjoy...
[truly, amazed, n...	[truly amazed, am...	[stir_many, many_...	[truly, amazed, n...

Figure 2: Dataset Text Processing

The analysis of the results, obtained by introducing generic bigrams and removing the single tokens contained within them are reported in Figure 3. The output shows some interesting but also problematic patterns.

Frequent Itemsets:

items	freq
[story]	5149
[character]	3907
[author]	3067
[novel]	2752
[old]	1925
[thought]	1909
[man]	1893
[long]	1754
[part]	1680
[child]	1645
[real]	1586
[family]	1552
[woman]	1516
[history]	1507
[fact]	1501
[right]	1461
[review]	1440
[place]	1435
[without]	1427
[series]	1426

Association Rules:

antecedent	consequent	confidence	lift	support
[main]	[character]	0.5894924309884239	4.568830382073321	0.02186189359664476
[plot]	[character]	0.4291845493562232	3.32637249527919	0.0198144050724877
[tale]	[story]	0.40704070407040704	2.393785115547873	0.012218883128034081
[novel, story]	[character]	0.3833943833943834	2.9714781990185117	0.010369538654601896
[fiction]	[story]	0.3633177570093458	2.1366527481064286	0.010270466629239456
[novel, character]	[story]	0.35280898876404493	2.074851231067012	0.010369538654601896
[girl]	[story]	0.3427601809954751	2.0157547175614647	0.010006274561606288
[short]	[story]	0.3422818791946309	2.0129418496586946	0.01178957101813018
[novel]	[character]	0.3234011627906977	2.506503867536503	0.029391367524190085
[plot]	[story]	0.32331902718168815	1.9014223076497765	0.014926851821274066
[main]	[story]	0.3205699020480855	1.8852548463620271	0.011888643043492618
[family]	[story]	0.3176546391752577	1.8681103377094541	0.01628083616789406
[around]	[story]	0.3162393162393162	1.8597868974641163	0.013440771440837489
[set]	[story]	0.3147340889276373	1.850934734281955	0.011921667051946765
[character]	[story]	0.3135398003583312	1.8439111855992671	0.04045441035632905
[place]	[story]	0.3121951219512195	1.8360032021372847	0.014794755787457481
[man]	[story]	0.31061806656101426	1.8267286217778351	0.019418116971037944
[child]	[story]	0.3100303951367781	1.8232725568337111	0.016842244311614542
[true]	[story]	0.29770992366412213	1.75081650776331	0.012879363297117004
[novel]	[story]	0.2976017441860465	1.7501803099043844	0.027046662923945707

Figure 3: Results of the FP-Growth Algorithm

In the frequent itemsets we observe a dominance of common words such as story, character, author, and novel. These tokens capture the narrative context but they do not strongly convey evaluative or emotional meaning. The association rules confirm this tendency: most of the discovered relationships connect structural elements (for example 'main' → 'character', or 'plot' → 'story'), with relatively high lift values. This suggests that the bigram-based approach is effective in detecting common descriptive combinations, but it largely misses the evaluative vocabulary. The reason is clear: by discarding the single tokens when they appear inside bigrams, highly informative words such as 'good', 'bad', or 'recommend' disappear from the distribution. The overall effect is a frequency landscape that privileges narrative structure but weakens sentiment analysis.

Bigrams found: 3
Single token: 2201
Bigram examples: ['didnt_anything', 'didnt_like', 'didnt_mind']

bigrams_raw	meaningful_bigrams	basket
[lebron james, ja...	[]	[lebron, james, b...
[classic pertains...	[]	[classic, pertain...
[plot character, ...	[]	[plot, character,...
[although title, ...	[]	[although, title,...
[best funny, funn...	[]	[best, funny, fel...
[africa arabia, a...	[]	[africa, arabia, ...
[according author...	[]	[according, autho...
[hazlitt prominen...	[]	[hazlitt, promine...
[teacher longer, ...	[]	[teacher, longer,...
[economics lesson...	[]	[economics, lesso...
[great accessible...	[]	[great, accessibl...
[laugh negative, ...	[]	[laugh, negative,...
[henry hazlitt, h...	[]	[henry, hazlitt, ...
[enjoy folk, folk...	[]	[enjoy, folk, fai...
[big fan, fan jac...	[]	[big, fan, jack, ...
[terrible childis...	[]	[terrible, childi...
[worse higgins, h...	[]	[worse, higgins, ...
[four old, old do...	[]	[four, old, dog, ...
[womderful enjoye...	[didnt_like]	[womderful, enjoy...
[truly amazed, am...	[]	[truly, amazed, n...

Figure 4: Dataset Text Processing after Bigrams Fix

For this reason we proceed with the reformulation of the problem. The second set of results, obtained by keeping single tokens but encoding negative bigrams such as 'didnt_like' or 'dont_recommend', produces a more interpretable distribution. In the frequent itemsets, alongside structural tokens like 'story' and 'character', we now see evaluative words such as 'good', 'great', 'recommend' and 'interesting'. Their high frequency indicates that sentiment related words play a central role in shaping readers' opinions. The association rules also become more meaningful: combinations like 'plot' + 'well' → 'character' or 'novel' + 'story' → 'character' still highlight structural aspects, but rules involving tokens such as 'good', 'like', or 'recommend' introduce evaluative dimensions. Importantly, by separating negated bigrams from positive single tokens, the model prevents misleading interpretations: the word 'like' no longer artificially inflates positive sentiment when it actually appears in negative contexts such as 'didn't like'.

Frequent Itemsets:

items	freq
[story]	7427
[like]	6726
[good]	5690
[great]	5685
[character]	5239
[well]	5168
[many]	4648
[much]	4492
[love]	4276
[really]	4117
[author]	3886
[novel]	3815
[best]	3220
[recommend]	2783
[thought]	2553
[interesting]	2413
[character, story]	2385
[better]	2373
[must]	2311
[lot]	2258

Association Rules:

antecedent	consequent	confidence	lift	support
[main, novel]	[character]	0.8666666666666667	5.017535153019025	0.010715109953512908
[main, story]	[character]	0.8262476894639557	4.783530954214782	0.014737397382216215
[main]	[character]	0.7379767827529021	4.272489749509119	0.029342916488081502
[main, like]	[character]	0.7379454926624738	4.272308596668352	0.011605288318881673
[plot, well]	[character]	0.6569767441860465	3.803542971541702	0.011176683920741157
[plot, novel]	[character]	0.6529968454258676	3.7805014923863314	0.013649401602321058
[novel, well, character]	[story]	0.6429942418426103	2.6259133363845715	0.011044805644390228
[plot, like]	[character]	0.6330708661417322	3.6651407598673185	0.013253766773268274
[novel, well, story]	[character]	0.6320754716981132	3.65937795993042	0.011044805644390228
[part, character]	[story]	0.6259541984732825	2.5563237907490413	0.01351752332597013
[plot, good]	[character]	0.6161616161616161	3.567245271959912	0.010055718571758267
[plot, story]	[character]	0.6096324461343473	3.5294448795000744	0.015858362731199102
[place, character]	[story]	0.6067193675889329	2.477770989408903	0.01012165770993373
[novel, like, story]	[character]	0.6053067993366501	3.5044017046535476	0.012033892717022189
[character, good, like]	[story]	0.6	2.450329877474081	0.01157231874979394
[well, character, like]	[story]	0.5925233644859813	2.419796171835775	0.010451353400811051
[man, character]	[story]	0.581954887218045	2.376635745820725	0.012759223236952293
[told]	[story]	0.5739231664726426	2.343835136970745	0.01625399756025189
[plot]	[character]	0.5732734418865806	3.3189457464901464	0.03366193003857439
[many, character]	[story]	0.560337552742616	2.288353078259901	0.021891793874254065

Figure 5: Results of the FP-Growth Algorithm after Bigrams Fix

Interpretations and Conclusions Comparing the two approaches, we can therefore draw some conclusions. The initial bigram strategy was useful as an exploratory step, since it showed how frequent co-occurrences of descriptive terms can be detected and structured. However, it was not ideal because it sacrificed the evaluative vocabulary that is crucial for sentiment-oriented analysis. The refined strategy, based on encoding only negation-related bigrams, strikes a better balance: it retains the richness of sentiment tokens while still capturing meaningful expressions. In this way, the analysis becomes more faithful to the real distribution of opinions in the corpus, highlighting both the narrative structure of the texts and the evaluative judgments expressed by the readers. Moreover, by filtering in this selective way, the resulting feature space remains more compact and interpretable, which is especially valuable when scaling the model to larger datasets. This not only improves computational efficiency but also facilitates a clearer understanding of how negation interacts with sentiment in shaping user opinions. In practice, this method allows us to focus on those linguistic constructions that genuinely alter polarity, rather than cluttering the model with redundant or weakly informative bigrams.

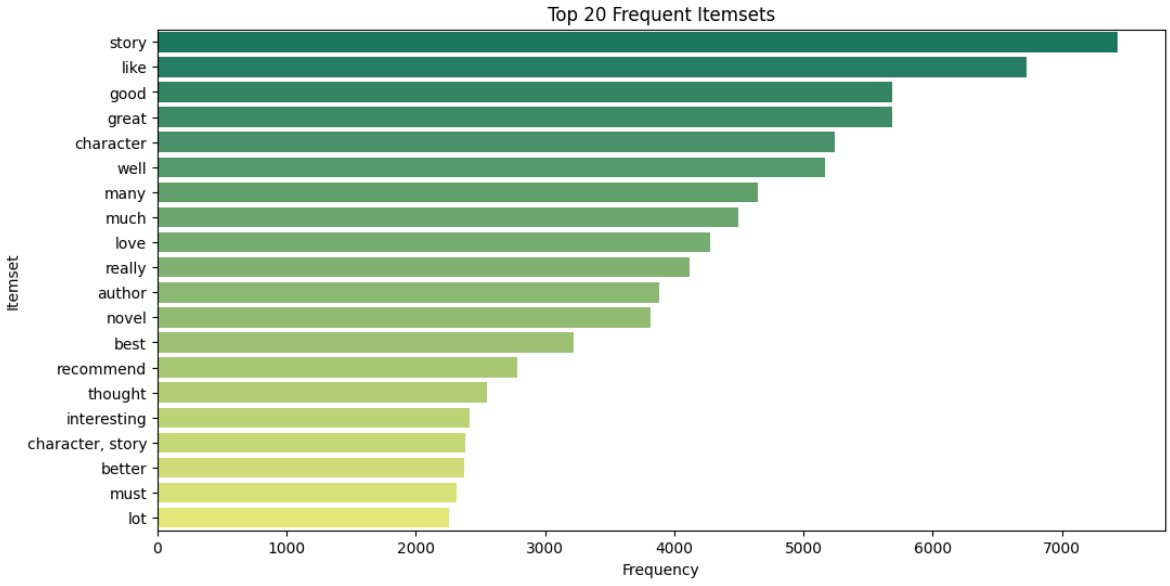


Figure 6: Top 20 frequent Itemsets Ordered by Frequency - Histogram

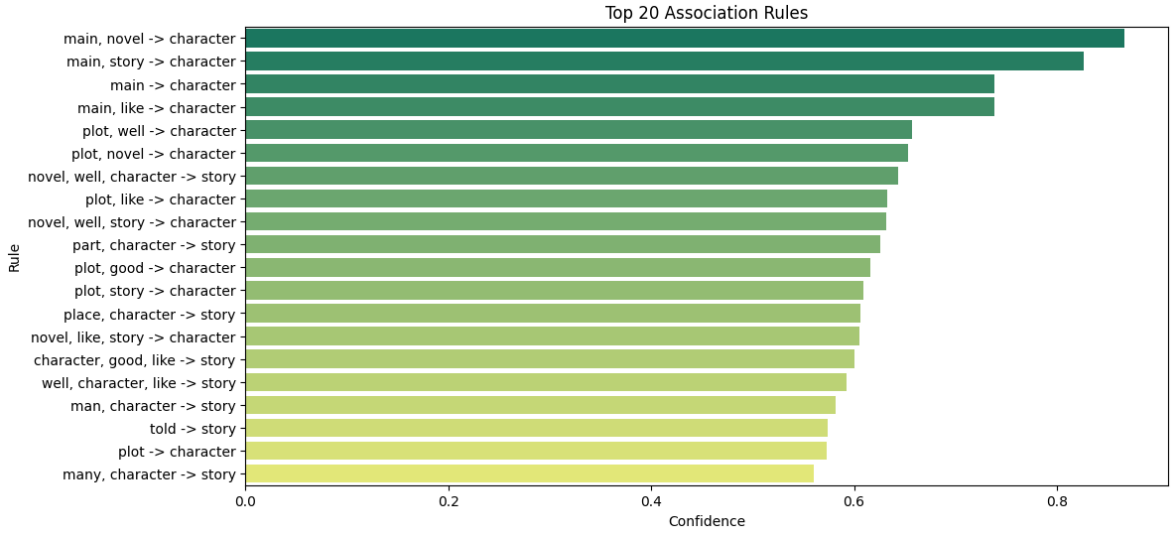


Figure 7: Top 20 Association Rules Ordered by Confidence - Histogram

By reducing the lift and confidence to lower values, we can obtain more sensible association rules because they are less trivial, as shown in figure 8. The filtered association rules (lift > 1.2, confidence > 0.3) provide a clearer picture of how readers connect descriptive and evaluative language and more sentimet related association of words can be done.

antecedent	consequent	confidence	lift	support
[amazing]	[story]	0.3515881708652793	1.4358449993960936	0.01058323167716198
[past]	[like]	0.33189655172413796	1.4966925825668789	0.010154627279021463
[past]	[story]	0.41810344827586204	1.7074856186421397	0.012792192806040025
[place, like]	[story]	0.521669341894061	2.130436624342098	0.010715109953512908
[man]	[character]	0.3056066176470588	1.7692984004300327	0.021924763443341797
[man]	[like]	0.35202205882352944	1.5874488650277239	0.02525468992120273
[man]	[story]	0.42922794117647056	1.7529167475189886	0.03079357752794171
[great, good]	[like]	0.3870192307692308	1.745269147853336	0.015924301869374567
[great, good]	[story]	0.3766025641025641	1.538000857923101	0.015495697471234051
[favorite]	[story]	0.35220994475138123	1.4383842512796747	0.016814480234743332
[language]	[story]	0.33988212180746563	1.3880388631401965	0.01140747090435528
[tale]	[character]	0.31588613406795224	1.8288112869660356	0.011341531766179816
[tale]	[story]	0.549127640036731	2.2425731048813904	0.01971580231446375
[without]	[like]	0.3369495851946394	1.5194793143827843	0.017407932478322508
[without]	[story]	0.34716017868538607	1.4177615968367367	0.01793544558372622
[beautiful]	[story]	0.41875	1.7101260603204524	0.013253766773268274
[mean]	[like]	0.3717217787913341	1.6762850538982983	0.010748079522600639
[author]	[like]	0.30030880082346884	1.354247136154718	0.03847548712538327
[author]	[story]	0.32501286670097784	1.327314563068178	0.041640565757805545
[novel, many]	[well]	0.3399778516057586	1.9953305373557013	0.01012165770993373

Figure 8: Association Rules considering Lift > 1.2 and Confidence > 0.3

Importance of 'story' and 'character' tokens As expected, words like 'story' and 'character' frequently appear as consequents in many rules. For example, 'man' $\rightarrow$ 'story' (confidence 0.43, lift 1.75) or 'tale' $\rightarrow$ 'character' (confidence 0.32, lift 1.83) show that readers describe narratives mainly in terms of 'character' and 'story'. Similarly, 'author' $\rightarrow$ 'story' has high support (0.0416), reflecting the natural link between 'author' and 'story'. Naturally, the correlation with book-related words is high, confirming the general tendency of the reviews to engage with the themes of the book and the story rather than limiting themselves to a direct judgment without any comment on the plot or the author

How positive words highlight the story Positive adjectives and evaluative words often co-occur with mentions of 'story', highlighting the narrative focus of reviewers. For example, 'amazing' → 'story' and 'beautiful' → 'story' have high confidence (0.35–0.42) and lift (> 1.4). Similarly, 'favorite' → 'story' shows that personal appreciation is closely tied to 'story'. These patterns suggest that positive words naturally emphasize 'story' in reviews. So we can venture to say that in general most of the reviews that are made are of a positive nature, since bigrams like 'didnt_like' or 'dont_like' are not reported in the most frequent words or in the most frequent association rules.

The token 'like' as a central expression of appreciation The word 'like' frequently appears as a consequent, connected to antecedents such as 'past', 'man', 'without', 'mean', and even combinations like 'great', 'good'. This indicates that 'like' acts as a general evaluative outcome: many descriptive contexts lead to expressions of liking. Lift values (1.5–1.7) confirm that these associations are stronger than chance, reflecting a strong evaluative tendency.

How combined words reveal stronger evaluations Some multi-word antecedents show even stronger associations with sentiment-related consequents. For example, 'place', 'like' → 'story' (confidence 0.52, lift 2.13) or 'great', 'good' → 'like' (confidence 0.39, lift 1.74) demonstrate that when evaluative terms appear together, their predictive power increases. This indicates that combinations of words can better capture subtle sentiment in reviews, showing how multiple descriptors work together to emphasize appreciation for the story or other narrative elements.

Less frequent but insightful associations Even less common rules can provide valuable information about evaluative patterns. For instance, 'novel', 'many' → 'well' (confidence 0.34, lift 2.0) highlights how evaluative adverbs connect to broader narrative descriptions. These rules often reflect judgments about style, quality, or narrative technique rather than purely emotional reactions, demonstrating that even rare word combinations contribute to understanding how reviewers express their opinions.

Overall conclusions Compared to simpler bigram strategies, these rules show a balanced mix of structural anchors ('story', 'character') and evaluative outcomes ('like', 'well', positive adjectives). Lift values above 1.3 indicate that the associations are meaningful, while confidence values above 0.3 show reasonably strong predictive links. This demonstrates that the refined pre-processing — keeping single tokens and meaningful negated bigrams — successfully preserves evaluative richness while capturing structural regularities.

5 Parameter Tuning and Trade-offs

In order to evaluate the effect of different parameter settings on the quality and scalability of the algorithm, several experiments were carried out by varying both the minimum support and the minimum confidence thresholds. The results are summarized in figure 9 and visualized in the heatmaps presented in figure 10.

support	confidence	num_itemsets	num_rules	training_time
0.005	0.1	8337	10833	1281.045831
0.005	0.2	8337	7750	1260.319609
0.005	0.3	8337	4584	1280.921283
0.010	0.1	1971	1849	905.379267
0.010	0.2	1971	1506	895.289781
0.010	0.3	1971	712	893.978231
0.020	0.1	461	296	580.551028
0.020	0.2	461	202	600.343250
0.020	0.3	461	100	593.063365

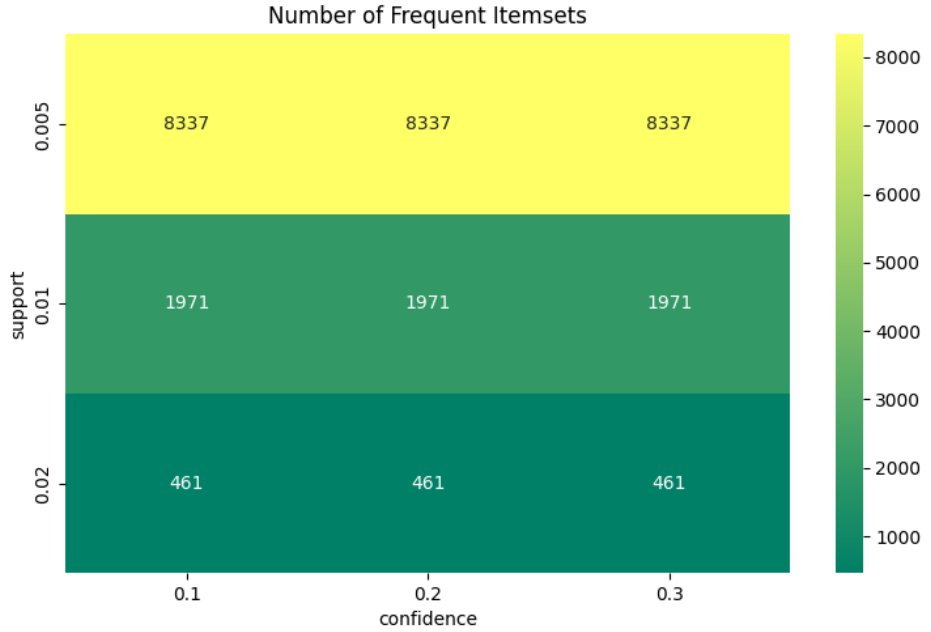
Figure 9: Parameter Tuning Results

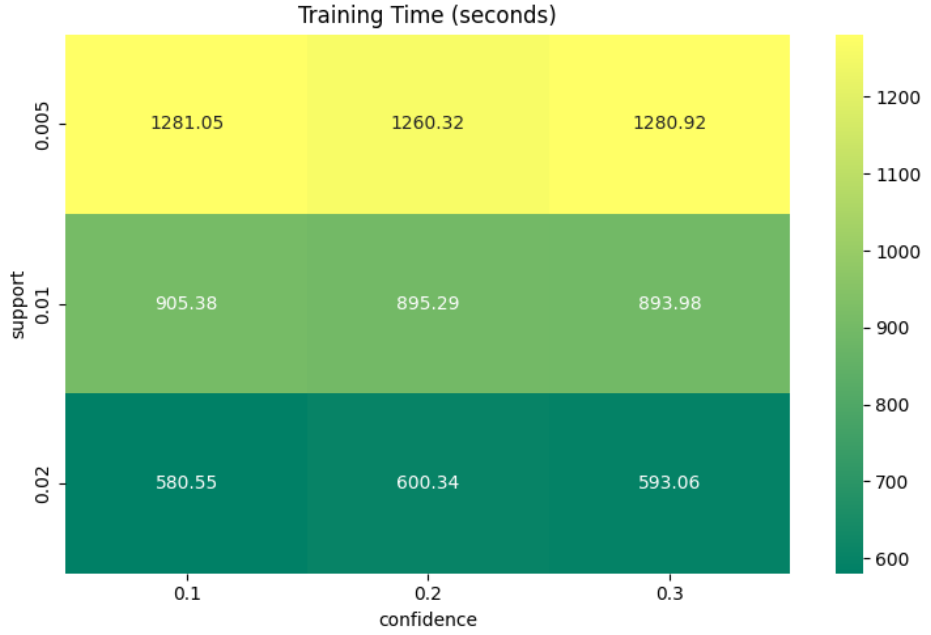
When the minimum support was set to a very low value (0.005), the algorithm generated the largest number of frequent itemsets (8,337) and rules (over 10,000 for confidence = 0.1). However, this configuration also required the longest training time (approximately 1280 seconds, equal to 21 minutes) due to the explosion of the search space. Increasing the confidence threshold reduced the number of rules substantially (from 10,833 to 4,584), while the number of itemsets remained

constant. This confirms that confidence acts mainly as a filter on rule reliability, while support controls the combinatorial growth of candidate itemsets.

At a medium support level (0.01), the number of itemsets dropped sharply to 1,971, with a corresponding decrease in rules (from 1,849 to 712 as confidence increased). Training time was significantly reduced (approximately 900 seconds, equal to 15 minutes), yielding a more balanced trade-off between computational efficiency and interpretability. This setting proved to be the most informative, as it retained a sufficiently large rule set without incurring excessive computational cost.

Finally, at high support (0.02), only 461 itemsets and 100–296 rules were discovered. Training was fast (approximately 590 seconds, equal to 10 minutes), but the rule space became extremely sparse, limiting the ability to capture meaningful associations. The heatmaps clearly illustrate this effect, with the rules-per-itemsets ratio dropping as low as 0.22 for the strictest configuration (support = 0.02, confidence = 0.3). Overall, these results highlight the inherent trade-off between quantity, quality, and computational cost in association rule mining. A medium support threshold combined with a moderate confidence level offers the most insightful and computationally feasible configuration.





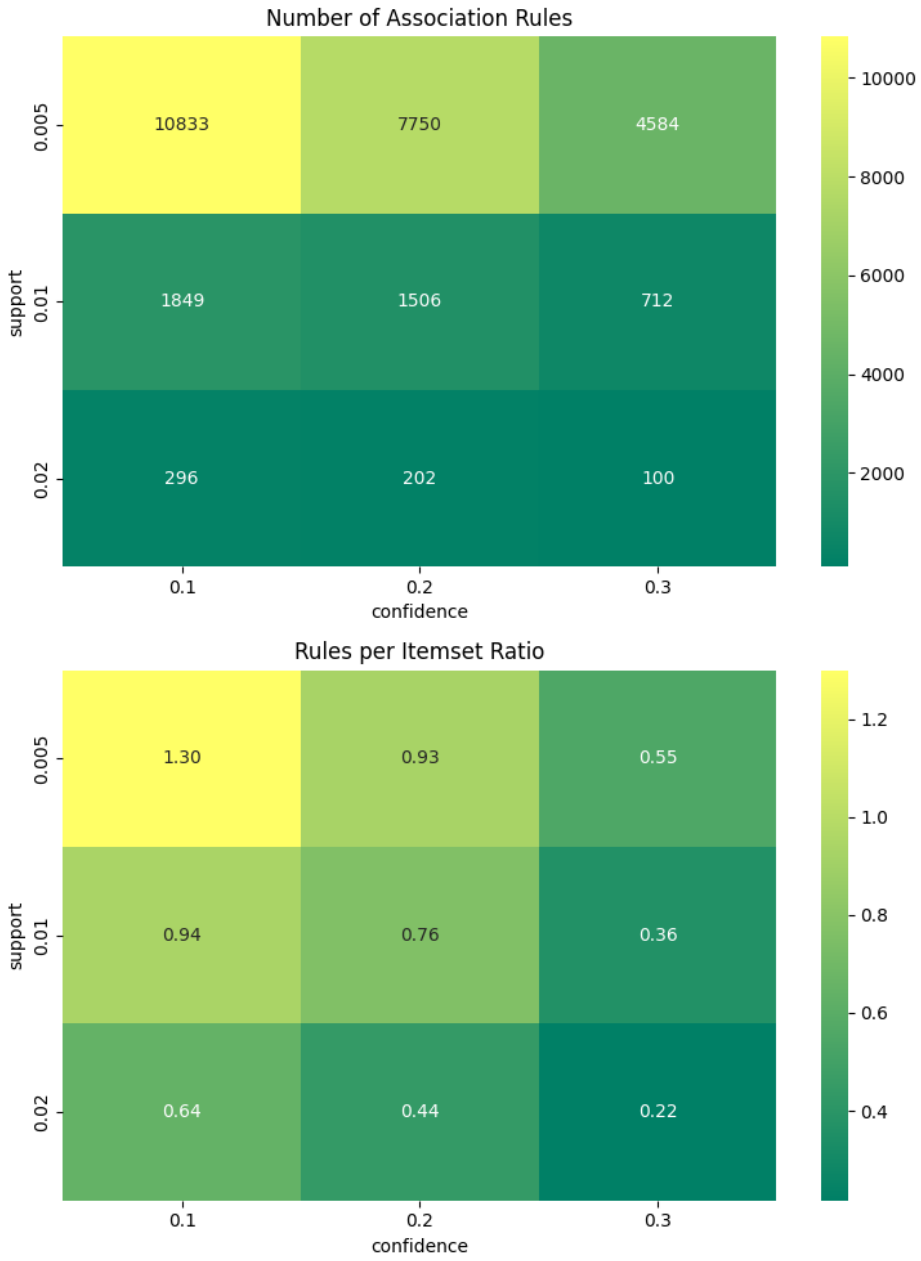


Figure 10: Parameter Tuning Results - Heatmap

6 Conclusions

In this project, we applied market-basket analysis to textual book reviews, using Frequent Pattern Growth to extract frequent itemsets and association rules from pre-processed baskets of tokens and meaningful bigrams. Careful text pre-processing, including tokenization, lemmatization, stopword removal, selective filtering, and the creation of sentiment relevant bigrams allowed us to capture both structural elements (like 'story' and 'character') and evaluative language (such as 'good', 'like', or 'recommend'). The analysis revealed that positive words and combined expressions strongly correlate with narrative elements, while even less frequent associations provide insights into stylistic or sentiment patterns. Parameter tuning highlighted the trade-off between rule quantity, interpretability, and computational cost, showing that medium support and moderate confidence thresholds yield the most informative and manageable set of rules.