Product Requirements Document

Grocery Association Rules Mining System (3→2 Pattern)

Author: Yair Levi
Date: October 15, 2025
Version: 1.0
Status: Final
Document Type: Technical Specification

---

1. Executive Summary

This document outlines the requirements for a grocery association rules mining system that discovers complex purchasing patterns where 3 items predict the purchase of 2 additional items. The system analyzes transaction data from a grocery store to identify actionable cross-selling opportunities and product bundling strategies.

1.1 Purpose

Develop a Python-based association rules mining tool specifically designed for grocery retail analytics, focusing on discovering 3→2 item relationships that can drive revenue through strategic product recommendations and placement.

1.2 Business Goals

- Identify high-value cross-selling opportunities in grocery retail
- Discover complex multi-item purchasing patterns (5-item combinations)
- Enable data-driven product placement and bundling strategies
- Provide actionable insights for marketing campaigns
- Support inventory management through purchase correlation analysis

1.3 Success Criteria

- Discover all 3→2 association rules meeting threshold criteria (0.3% support, 40% confidence, lift > 1)
- Process datasets with 1,000+ transactions within 60 seconds
- Identify the highest-lift rule for immediate business action
- Provide clear, interpretable output suitable for business stakeholders
- Generate actionable recommendations for each discovered rule

---

2. Product Overview

2.1 Target Users

User Type	Description	Primary Use Case
Retail Analysts	Analyze shopping patterns	Identify cross-selling opportunities
Marketing Teams	Campaign planning	Target customers with product bundles
Store Managers	Optimize layout	Position related products strategically
Merchandisers	Product placement	Create effective product displays
Data Scientists	Pattern discovery	Research consumer behavior

2.2 Key Differentiators

Unique Value Proposition:

- Focuses on complex 3→2 patterns (5-item combinations)

- Very low support threshold (0.3%) captures rare but valuable patterns
- Moderate confidence threshold (40%) balances discovery with reliability
- Manual metric calculation ensures transparency and educational value

Compared to Standard Tools:

- Mlxtend/Apriori: Focus on 1→1 or 1→N patterns
- This tool: Specifically designed for 3→2 complex associations
- Provides business-oriented output and recommendations

2.3 System Architecture

Input: grocery_dataset.txt
↓
[Data Loading & Validation]
↓
[Frequent Itemset Mining]
• 1-itemsets (individual items)
• 2-itemsets (pairs)
• 3-itemsets (triples)
• 5-itemsets (5 items together)
↓
[Rule Generation (3→2)]
• Split 5-itemsets into 3+2
• Calculate support, confidence, lift
• Apply threshold filters
↓
[Ranking & Analysis]
• Sort by lift
• Identify best rule
• Generate insights
↓
Output: Console report with rules and recommendations

---

3. Functional Requirements

3.1 Data Input (FR-1)

Priority: Critical
Complexity: Low

Requirements

FR-1.1: File Format Support

- Read text file named `grocery_dataset.txt`
- Support UTF-8 encoding

- Handle comma-separated values per line
- Each line represents one transaction
- Items separated by commas with optional whitespace

FR-1.2: Data Parsing

- Strip whitespace from item names
- Ignore empty lines
- Handle empty transactions gracefully
- Preserve original item names (case-sensitive)
- Support items with spaces in names

FR-1.3: Data Validation

Valid format examples:

bread, milk, eggs, butter, cheese

Whole Milk, Organic Eggs, Butter, Yogurt, Cheese

bread,milk,eggs,butter,cheese,yogurt

Invalid formats (should be handled):

(empty line) → Skip

bread, , eggs → Remove empty item

bread → Valid but cannot form 5-itemsets

FR-1.4: Dataset Statistics Display upon loading:

- Total number of transactions
- Number of unique items
- Sample items (first 15)
- Transaction size statistics (min, max, average)

Acceptance Criteria:

- Successfully loads files with 1,000+ transactions
- Handles Unicode characters in item names
- Reports clear error if file not found
- Shows helpful format examples on error
- Processing time < 5 seconds for 10,000 transactions

3.2 Frequent Itemset Mining (FR-2)

Priority: Critical
Complexity: High

Requirements

FR-2.1: 1-Itemset Mining

- Calculate support for every unique item
- Filter items with support ≥ 0.3%
- Store support values for later use

FR-2.2: 2-Itemset Mining

- Generate all pairs from frequent 1-itemsets
- Calculate support for each pair
- Filter pairs with support ≥ 0.3%
- Use frozenset for efficient storage

FR-2.3: 3-Itemset Mining

- Generate candidate 3-itemsets from frequent 2-itemsets
- Apply join operation (merge pairs sharing one item)
- Calculate support for candidates
- Filter triples with support ≥ 0.3%
- Prune infrequent itemsets early

FR-2.4: 5-Itemset Mining

- Generate candidate 5-itemsets from frequent 3-itemsets
- Apply join operation (merge triples)
- Calculate support for candidates
- Filter 5-itemsets with support ≥ 0.3%
- Store results for rule generation

FR-2.5: Support Calculation Formula

Support(X) = Count(transactions containing all items in X) / Total transactions

Implementation:
- Use set intersection for efficient subset checking
- Return decimal value (0.0 to 1.0)
- Precision: 6 decimal places (0.000001)

FR-2.6: Performance Optimization

- Use set operations instead of list comparisons
- Convert transactions to sets once
- Cache support calculations
- Generate candidates efficiently

Acceptance Criteria:

- Correctly identifies all frequent itemsets at each level
- Support calculations match manual verification
- Processes 10,000 transactions in < 30 seconds
- Memory usage < 500 MB for 1,000 unique items
- No duplicate itemsets in results

3.3 Association Rule Generation (FR-3)

Priority: Critical
Complexity: High

Requirements

FR-3.1: Rule Pattern (3→2) Generate rules of the form:

{Item1, Item2, Item3} → {Item4, Item5}

For each frequent 5-itemset:

- Generate all combinations of 3 items as antecedent
- Remaining 2 items become consequent
- Total of C(5,3) = 10 possible rules per 5-itemset

FR-3.2: Confidence Calculation

Confidence({A,B,C} → {D,E}) = Support({A,B,C,D,E}) / Support({A,B,C})

Requirements:
- Use pre-calculated support values
- Handle division by zero (return 0.0)
- Filter rules with confidence ≥ 40%
- Return decimal (0.0 to 1.0)
- Precision: 4 decimal places

FR-3.3: Lift Calculation

Lift({A,B,C} → {D,E}) = Support({A,B,C,D,E}) / (Support({A,B,C}) × Support({D,E}))

Interpretation:
- Lift > 1: Positive correlation (items appear together more than chance)
- Lift = 1: Independence (no correlation)
- Lift < 1: Negative correlation (items rarely together)

Requirements:
- Filter rules with lift > 1.0
- Return decimal value
- Precision: 4 decimal places

FR-3.4: Threshold Application Apply filters in order:

1. Support ≥ 0.3% (0.003) - Applied during itemset mining
2. Confidence ≥ 40% (0.40) - Applied during rule generation
3. Lift > 1.0 - Applied during rule generation

FR-3.5: Rule Storage Store each qualifying rule with:

python

```
{
    'antecedent': (item_idx1, item_idx2, item_idx3),  # Sorted tuple
    'consequent': (item_idx4, item_idx5),             # Sorted tuple
    'support': float,                                 # 0.003 to 1.0
    'confidence': float,                              # 0.40 to 1.0
    'lift': float                                     # > 1.0
}
```

Acceptance Criteria:

- All qualifying rules discovered (100% recall)
- No duplicate rules in output
- Confidence calculation accurate within 0.0001
- Lift calculation accurate within 0.0001
- Rules sorted by lift (descending)

3.4 Output and Reporting (FR-4)

Priority: High
Complexity: Medium

Requirements

FR-4.1: Console Output Structure

Header Section:

```
================================================================================
GROCERY ASSOCIATION RULES MINING
Author: Yair Levi
================================================================================

Dataset File: grocery_dataset.txt
Rule Pattern: {Item1, Item2, Item3} -> {Item4, Item5}
Minimum Support: 0.3%
Minimum Confidence: 40%
Minimum Lift: 1.0
================================================================================
```

FR-4.2: Dataset Information

✓ Dataset loaded successfully!
Total transactions: XXXX
Unique items: XXX
Sample items: item1, item2, item3, ...

Transaction sizes: min=X, max=X, avg=X.XX

FR-4.3: Mining Progress

==========
CALCULATING SUPPORT FOR ITEMSETS
==========

Step 1: Finding frequent 1-itemsets (individual items)...
✓ Found XXX frequent 1-itemsets (support >= 0.3%)

Step 2: Finding frequent 2-itemsets (pairs)...
✓ Found XXX frequent 2-itemsets (support >= 0.3%)

Step 3: Finding frequent 3-itemsets (triples)...
✓ Found XXX frequent 3-itemsets (support >= 0.3%)

Step 4: Finding frequent 5-itemsets (5 items together)...
Checking XXX candidate 5-itemsets...
✓ Found XXX frequent 5-itemsets (support >= 0.3%)

FR-4.4: Rules Table

```
================================================================================
ASSOCIATION RULES
================================================================================

Rule                                          Support       Confidence     Lift
--------------------------------------------------------------------------------
{bread, milk, eggs} -> {butter, cheese}                0.003200 ( 0.32%)  0.4500 (45.00%)  1.2345
{coffee, yogurt, cereal} -> {milk, orange juice}          0.003100 ( 0.31%)  0.4200 (42.00%)  1.1890
...
```

FR-4.5: Best Rule Highlight

```
================================================================================
⭐ BEST RULE BY LIFT ⭐
================================================================================

Rule: {bread, milk, eggs} -> {butter, cheese}

  Support:    0.003200 (0.320%)
  Confidence: 0.4500 (45.00%)
  Lift:       1.2345 ⭐ HIGHEST

Interpretation:
  • 0.32% of all transactions contain all 5 items
  • When {bread, milk, eggs} are purchased together,
    there's a 45.00% chance that {butter, cheese} are also purchased
  • These items appear together 1.23x more often than expected by chance

  Business Insights:
    ➜ Cross-sell opportunity: Recommend {butter, cheese} to customers buying
      {bread, milk, eggs}
    ➜ Product placement: Position these 5 items in proximity
    ➜ Bundle promotion: Create a 5-item package deal
    ➜ Marketing: Create targeted campaigns for customers buying the antecedent items
```

FR-4.6: Summary Statistics

```
================================================================================
SUMMARY STATISTICS
================================================================================

Total Rules Found: XX
Frequent 1-itemsets: XXX
Frequent 2-itemsets: XXX
Frequent 3-itemsets: XXX
Frequent 5-itemsets: XXX

Average Metrics:
  Support:    0.003XXX (0.3XX%)
  Confidence: 0.4XXX (4X.XX%)
  Lift:       1.XXXX

Maximum Metrics:
  Support:    0.00XXXX (0.XXX%)
  Confidence: 0.XXXX (XX.XX%)
  Lift:       X.XXXX

Minimum Metrics:
  Support:    0.003000 (0.300%)
  Confidence: 0.4000 (40.00%)
  Lift:       1.0001
```

FR-4.7: Top Rules Breakdown Display top 5 rules by:

- Lift (strongest associations)
- Confidence (most reliable)
- Support (most frequent)

FR-4.8: Item Analysis

```
================================================================================
ITEM COMBINATION ANALYSIS
================================================================================

--- Top 10 Items in Antecedents (Most Predictive) ---
 1. bread                       Appears in XX rules
 2. milk                        Appears in XX rules
...

--- Top 10 Items in Consequents (Most Predicted) ---
 1. butter                      Predicted in XX rules
 2. cheese                      Predicted in XX rules
...
```

FR-4.9: Error Messages If no rules found:

```
✗ No rules found meeting the specified criteria.

Possible reasons:
  • Support threshold too high (current: 0.3% - try 0.1% or lower)
  • Confidence threshold too high (current: 40% - try 30% or lower)
  • Dataset too small for 5-item combinations
  • Transactions don't have enough items (need at least 5 items per transaction)

Tip: Lower the thresholds in the code:
  MIN_SUPPORT = 0.001      # 0.1%
  MIN_CONFIDENCE = 0.30    # 30%
```

Acceptance Criteria:

- All sections displayed in correct order
- Numbers formatted consistently
- Percentages shown alongside decimals
- Clear visual separators between sections
- Output suitable for business presentation
- No technical jargon in business insights

---

4. Non-Functional Requirements

4.1 Performance (NFR-1)

Priority: High

Metric	Requirement	Target	Measurement
File Loading	< 10 seconds	3 seconds	10,000 transactions
1-itemset Mining	< 5 seconds	2 seconds	1,000 items
2-itemset Mining	< 10 seconds	5 seconds	500K pairs
3-itemset Mining	< 20 seconds	10 seconds	100K triples
5-itemset Mining	< 30 seconds	15 seconds	50K candidates
Rule Generation	< 10 seconds	5 seconds	1,000 rules
Total Execution	< 60 seconds	40 seconds	End-to-end
Memory Usage	< 1 GB	500 MB	Peak usage

Scalability:

- Transactions: 1,000 to 50,000
- Unique Items: 50 to 2,000
- Frequent 5-itemsets: Up to 10,000

4.2 Accuracy (NFR-2)

Priority: Critical

Calculation Precision:

- Support: 6 decimal places (0.000001)
- Confidence: 4 decimal places (0.0001)
- Lift: 4 decimal places (0.0001)

Correctness Verification:

- Manual verification of sample rules
- Cross-validation with known datasets
- Unit tests for all calculations
- Edge case handling (zeros, boundaries)

Tolerance:

- Support calculation error: < 0.0001
- Confidence calculation error: < 0.0001
- Lift calculation error: < 0.0001

4.3 Usability (NFR-3)

Priority: High

Ease of Use:

- Single command execution: `python grocery_association_rules.py`
- No command-line arguments required
- Clear progress indicators
- Self-explanatory output
- Helpful error messages with solutions

Configuration:

- Thresholds defined at top of file
- Easy to modify without code knowledge
- Clear comments explaining each parameter
- Example values provided

Documentation:

- Inline comments explaining logic

- Docstrings for all functions
- README with usage examples
- Sample dataset provided

4.4 Maintainability (NFR-4)

Priority: Medium

Code Quality:

- PEP 8 compliant formatting
- Meaningful variable names
- Functions < 50 lines
- Cyclomatic complexity < 10
- Comprehensive comments

Modularity:

- Separate functions for each major task
- Clear function interfaces
- Reusable components
- Easy to extend with new features

Testing:

- Unit tests for calculations
- Integration tests for workflow
- Edge case tests
- Performance benchmarks

4.5 Reliability (NFR-5)

Priority: High

Error Handling:

- File not found: Clear message with format example
- Empty dataset: Graceful exit with explanation
- Invalid format: Skip malformed lines, continue processing
- Memory overflow: Early detection, helpful message
- Zero division: Return 0.0, continue execution

Robustness:

- No crashes on edge cases
- Handles Unicode characters
- Manages large datasets gracefully
- Validates thresholds (0-1 range)

4.6 Portability (NFR-6)

Priority: Medium

Cross-Platform:

- Windows 10/11: Full support
- macOS (Intel/Apple Silicon): Full support
- Linux (Ubuntu 20.04+): Full support

Dependencies:

- Python 3.7+ (f-strings, type hints compatible)
- NumPy ≥ 1.19.0
- No platform-specific libraries
- Standard library only (except NumPy)

---

5. Technical Specifications

5.1 Technology Stack

Component	Technology	Version	Purpose
Language	Python	3.7+	Implementation
Array Operations	NumPy	≥1.19.0	Statistical calculations
Combinations	itertools	Built-in	Generate itemset combinations

5.2 Algorithm Details

Itemset Mining Algorithm:

Apriori-based approach with modifications:

1. Generate L1 (frequent 1-itemsets)
2. For k = 2 to 5:
   - Generate Ck (candidate k-itemsets) from L(k-1)
   - Calculate support for each candidate
   - Prune candidates with support < 0.3%
   - Store Lk (frequent k-itemsets)
3. Generate rules from L5

Complexity Analysis:

- Time: O(n × m × 2^k) where:
  - n = number of transactions
  - m = number of unique items
  - k = max itemset size (5)
- Space: O(m^k) for storing itemsets
- Worst case: O(n × m^5) for dense datasets

Optimizations:

1. **Early Pruning:** Remove infrequent itemsets immediately
2. **Set Operations:** Use set intersection for subset checking
3. **Candidate Generation:** Only combine compatible itemsets
4. **Support Caching:** Store calculated values
5. **Transaction Indexing:** Convert to sets once

5.3 Data Structures

Transaction Storage:

python

```
indexed_transactions: List[Set[int]]
# Each transaction converted to set of item indices
# Example: [{0, 1, 2, 3, 4}, {1, 2, 5, 6}, ...]
```

Item Mapping:

python

```
item_to_idx: Dict[str, int]
idx_to_item: Dict[int, str]
# Bidirectional mapping for efficient lookup
```

Support Storage:

python

```
item_support: Dict[int, float]            # 1-itemsets
pair_support: Dict[frozenset, float]      # 2-itemsets
triple_support: Dict[frozenset, float]    # 3-itemsets
five_support: Dict[frozenset, float]      # 5-itemsets
```

Rule Storage:

python

```
rules: List[Dict] = [
    {
        'antecedent': Tuple[int, int, int],
        'consequent': Tuple[int, int],
        'support': float,
        'confidence': float,
        'lift': float
    },
    ...
]
```

5.4 Configuration Parameters

python

```python
# File Configuration
DATASET_FILE = "grocery_dataset.txt"
ENCODING = 'utf-8'
DELIMITER = ','

# Mining Thresholds
MIN_SUPPORT = 0.003      # 0.3%
MIN_CONFIDENCE = 0.40    # 40%
MIN_LIFT = 1.0           # Must be greater than 1

# Output Configuration
DECIMAL_PLACES_SUPPORT = 6
DECIMAL_PLACES_CONFIDENCE = 4
DECIMAL_PLACES_LIFT = 4
TOP_N_RULES = 5          # For top rules display

# Performance Configuration
MAX_TRANSACTIONS = 50000
MAX_UNIQUE_ITEMS = 2000
CANDIDATE_LIMIT = 100000  # Max candidates to evaluate
```

---

6. Use Cases

6.1 Cross-Selling Optimization (UC-1)

Actor: Retail Marketing Manager
Goal: Identify product combinations for cross-selling campaigns
Preconditions: Transaction data available for at least 1 month

Main Flow:

1. Manager exports transaction data to `grocery_dataset.txt`
2. Runs mining script to discover patterns
3. Reviews discovered 3→2 rules
4. Identifies best rule by lift
5. Notes that customers buying {bread, milk, eggs} often buy {butter, cheese}
6. Creates targeted email campaign: "Customers who bought X might also like Y"
7. Implements recommendation on e-commerce site
8. Measures campaign effectiveness (conversion rate)

Postconditions: Campaign implemented, revenue increased by cross-selling

Success Metrics:

- Click-through rate: > 5%

- Conversion rate: > 2%
- Revenue per email: > $5

Alternative Flows:

- A1: No high-confidence rules found → Lower confidence threshold to 30%
- A2: Too many rules → Increase lift threshold to 1.5
- A3: Rules don't align with inventory → Focus on in-stock combinations

6.2 Store Layout Optimization (UC-2)

Actor: Store Manager
Goal: Optimize product placement to increase basket size
Preconditions: Physical store with movable displays

Main Flow:

1. Store manager collects 3 months of transaction data
2. Runs association rules mining
3. Discovers that {pasta, tomato sauce, ground beef} → {parmesan cheese, olive oil}
4. Currently these products are in different aisles
5. Creates end-cap display with all 5 products together
6. Adds signage: "Everything you need for spaghetti night!"
7. Monitors sales data for 1 month
8. Observes 25% increase in bundle purchases

Postconditions: Store layout optimized, average basket size increased

Success Metrics:

- Bundle purchase rate: +20%
- Average basket size: +$5
- Customer satisfaction: Maintained or improved

6.3 Product Bundling Strategy (UC-3)

Actor: Merchandising Director
Goal: Create profitable product bundles
Preconditions: Product cost and margin data available

Main Flow:

1. Director runs association rules analysis quarterly
2. Identifies rule: {chicken, rice, vegetables} → {soy sauce, ginger}
3. Calculates profitability:
    - Combined cost: $12
    - Normal selling price: $18
    - Bundle discount: 15% off = $15.30
    - Margin: $3.30 vs individual sales $6
4. Decides bundle is worthwhile due to:
    - Higher conversion (45% confidence)
    - Inventory movement for slow-moving items (ginger)
    - Customer perception of value
5. Creates "Asian Dinner Bundle" promotion
6. Tracks bundle sales for 2 months
7. Achieves 30% take rate on bundle offer

Postconditions: New bundle product created, incremental revenue generated

Success Metrics:

- Bundle sales: > 100 units/month

- Margin per bundle: > $3
- Customer reviews: > 4.0/5.0

6.4 Inventory Management (UC-4)

Actor: Inventory Manager
Goal: Optimize stock levels based on purchase correlations
Preconditions: Current inventory levels tracked in system

Main Flow:

1. Inventory manager runs weekly association analysis
2. Discovers multiple rules where {milk, eggs, bread} appear in antecedent
3. Recognizes these are "trigger items" that drive other purchases
4. Ensures these items are always in stock (never out-of-stock)
5. Adjusts reorder points for consequent items (butter, cheese, yogurt)
6. Sets up automatic alerts when trigger items run low
7. Monitors stockout rates over 3 months
8. Achieves 40% reduction in lost sales due to stockouts

Postconditions: Optimized inventory levels, reduced stockouts

Success Metrics:

- Stockout rate: < 2%
- Inventory turnover: +10%
- Lost sales: -40%

6.5 Customer Segmentation (UC-5)

Actor: Data Scientist
Goal: Segment customers based on purchase patterns
Preconditions: Customer IDs linked to transactions

Main Flow:

1. Data scientist exports customer transaction data
2. Runs association rules mining per customer segment
3. Discovers different patterns:
    - Health-conscious: {organic vegetables, quinoa, almond milk} → {avocado, chia seeds}
    - Family shoppers: {cereal, milk, snacks} → {juice boxes, cheese sticks}
    - Gourmet cooks: {specialty cheese, wine, prosciutto} → {olives, crackers}
4. Creates customer personas based on patterns
5. Tailors marketing messages per segment
6. Implements personalized recommendations
7. Measures engagement by segment
8. Observes 50% higher engagement in personalized campaigns

Postconditions: Customer segments identified, personalized marketing implemented

Success Metrics:

- Email open rate: +30%
- Recommendation click rate: +50%
- Customer lifetime value: +15%

---

7. Constraints and Assumptions

7.1 Technical Constraints

Hard Constraints:

- Python 3.7+ required (f-string syntax, ordered dicts)
- NumPy dependency must be installed
- Single-threaded execution (no multiprocessing)
- In-memory processing (no database)
- File-based input only (no API/streaming)

Soft Constraints:

- Recommended: 8GB+ RAM for large datasets
- Recommended: Modern CPU (2015+) for reasonable performance
- Practical dataset limit: 50,000 transactions
- Practical item limit: 2,000 unique items

7.2 Data Constraints

Dataset Requirements:

- Minimum transactions: 100 (for statistical significance)
- Minimum items per transaction: 5 (for 3→2 rules)
- Average items per transaction: 8-15 (ideal)
- File size: < 100 MB (for reasonable performance)

Data Quality:

- Item names: Consistent spelling and capitalization
- No special characters: Commas only as delimiters
- Encoding: UTF-8 or ASCII
- No missing data: Each line must have items or be empty

7.3 Business Constraints

Threshold Justification:

- **0.3% Support:** Captures rare but meaningful patterns
  - In 1,000 transactions, rule must appear 3+ times
  - In 10,000 transactions, rule must appear 30+ times
  - Too low → noise, too high → miss opportunities
- **40% Confidence:** Balances reliability and discovery
  - High enough for actionable insights
  - Low enough to find diverse patterns
  - Business acceptable for recommendations
- **Lift > 1.0:** Ensures positive correlation
  - Eliminates random or negative associations
  - Guarantees items appear together more than chance
  - Industry standard threshold

Retail Context:

- Grocery stores typically have 10,000+ SKUs
- Average basket: 10-20 items
- Transaction volume: 1,000-10,000 per day
- Seasonality affects patterns (adjust quarterly)

7.4 Assumptions

User Assumptions:

- Basic Python knowledge for threshold adjustment
- Understanding of retail metrics
- Access to transaction export capability
- Familiarity with CSV/text file format

Data Assumptions:

- Transaction data represents actual purchases
- Items are accurately recorded at POS
- Data is recent (< 6 months old)
- Represents typical shopping patterns
- No test/demo transactions included

Business Assumptions:

- Correlation implies causation (purchasing context)
- Historical patterns predict future behavior
- Cross-selling increases revenue
- Customers receptive to recommendations
- Physical product proximity influences purchases

---

8. Dependencies and Integrations

8.1 External Dependencies

Dependency	Version	Purpose	Critical	Fallback
Python	≥3.7	Runtime environment	Yes	None
NumPy	≥1.19.0	Statistical calculations	Yes	Pure Python (slower)

8.2 Standard Library Dependencies

Module	Purpose	Version
itertools	Combination generation	Built-in
collections	defaultdict (optional)	Built-in

8.3 Input/Output Interfaces

Input:

File: grocery_dataset.txt
Format: CSV (comma-separated)
Encoding: UTF-8
Example:
bread, milk, eggs, butter, cheese
coffee, yogurt, cereal, milk, orange juice

Output:

Medium: Console (stdout)
Format: Formatted text with tables
Encoding: UTF-8
Redirection: Supports > output.txt

8.4 Future Integrations

Planned (Version 2.0):

- Database connectivity (PostgreSQL, MySQL)
- CSV export of rules
- JSON API for programmatic access
- Excel integration via pandas
- Visualization with matplotlib
- Web interface with Flask

Potential (Version 3.0):

- Real-time stream processing
- Integration with POS systems
- E-commerce platform plugins
- BI tool connectors (Tableau, Power BI)
- Cloud deployment (AWS, Azure, GCP)
- Mobile app for rule browsing

---

9. Success Metrics and KPIs

9.1 Technical Performance Metrics

Metric	Target	Measurement Method	Frequency
Execution Time	< 40s	Time module	Per run
Memory Usage	< 500 MB	memory_profiler	Weekly
Rules Discovery Rate	> 95%	Manual verification	Per dataset
Calculation Accuracy	100%	Unit tests	Every commit
Code Coverage	> 80%	pytest-cov	Every commit
Bug Rate	< 5 per 1000 LOC	Issue tracking	Monthly

9.2 Business Impact Metrics

Metric	Baseline	Target	Measurement
Cross-Sell Conversion	1.5%	3.0%	A/B testing
Average Basket Size	$45	$52	POS data
Bundle Adoption Rate	0%	15%	Sales tracking
Recommendation CTR	2%	5%	Web analytics
Customer Lifetime Value	$500	$575	CRM data
Inventory Turnover	8x/year	10x/year	Inventory system
Stockout Rate	5%	2%	Inventory tracking
Marketing ROI	200%	300%	Campaign analytics

9.3 Quality Metrics

Metric	Target	Measurement
Output Clarity Score	> 4.5/5	User survey
Documentation Completeness	100%	Manual review
Code Readability	> 4.0/5	Peer review
Error Rate	< 0.1%	Error logs
User Satisfaction	> 85%	User feedback

9.4 Adoption Metrics

Metric	Target	Timeline
Active Users (analysts)	10+	3 months
Reports Generated	50+/month	6 months
Business Actions Taken	5+/month	3 months
Training Sessions	5	2 months
Documentation Views	100+	3 months

---

10. Testing Requirements

10.1 Unit Tests

Required Test Cases:

Data Loading Tests:

python

```
test_load_valid_file()
test_handle_missing_file()
test_parse_comma_separated()
test_handle_empty_lines()
test_handle_unicode_items()
test_handle_whitespace()
test_validate_transaction_format()
```

Support Calculation Tests:

python

test_support_single_item()
test_support_pair()
test_support_triple()
test_support_five_itemset()
test_support_with_zero_count()
test_support_with_full_coverage()
test_support_precision()

Confidence Calculation Tests:


python

test_confidence_basic()
test_confidence_with_zero_antecedent()
test_confidence_equals_one()
test_confidence_precision()
test_confidence_range_validation()

Lift Calculation Tests:


python

test_lift_positive_correlation()
test_lift_independence()
test_lift_negative_correlation()
test_lift_with_zero_support()
test_lift_precision()

Itemset Generation Tests:


python

test_generate_1_itemsets()
test_generate_2_itemsets()
test_generate_3_itemsets()
test_generate_5_itemsets()
test_candidate_pruning()
test_frequent_itemset_filtering()

Rule Generation Tests:


python

```
test_generate_3_to_2_rules()
test_rule_filtering_confidence()
test_rule_filtering_lift()
test_rule_uniqueness()
test_rule_sorting()
```

10.2 Integration Tests

End-to-End Workflow:


python

```
test_full_workflow_small_dataset()
test_full_workflow_large_dataset()
test_workflow_with_no_rules()
test_workflow_with_many_rules()
test_performance_benchmark()
```

Data Quality Tests:


python

```
test_various_item_names()
test_special_characters()
test_different_encodings()
test_varying_transaction_sizes()
test_duplicate_items_in_transaction()
```

10.3 Performance Tests

Benchmarks:


python

```
test_performance_1000_transactions()
test_performance_10000_transactions()
test_performance_50000_transactions()
test_memory_usage_large_dataset()
test_scalability_with_items()
```

Target Metrics:

- 1,000 transactions: < 5 seconds
- 10,000 transactions: < 40 seconds
- 50,000 transactions: < 120 seconds

10.4 Validation Tests

Cross-Validation:

- Compare with known association rules libraries (mlxtend)
- Verify against manual calculations (10 sample rules)
- Test with synthetic datasets with known patterns
- Validate against published research datasets

Edge Cases:

python

```
test_minimum_support_threshold()
test_single_frequent_5_itemset()
test_all_items_frequent()
test_no_frequent_5_itemsets()
test_maximum_dataset_size()
```

---

11. Risks and Mitigation

11.1 Technical Risks

Risk	Impact	Probability	Mitigation	Owner
Exponential complexity with many items	High	High	Implement early pruning, limit itemset generation	Dev Team
Memory exhaustion on large datasets	High	Medium	Add memory monitoring, chunked processing	Dev Team
Incorrect calculations	Critical	Low	Extensive unit tests, manual verification, peer review	QA Team
Performance degradation	Medium	Medium	Performance profiling, optimization, caching	Dev Team
Python/NumPy version incompatibility	Medium	Low	Document versions, test on multiple platforms	Dev Team
File encoding issues	Low	Medium	Support UTF-8, handle errors gracefully	Dev Team

11.2 Business Risks

Risk	Impact	Probability	Mitigation	Owner
Rules don't reflect current trends	High	Medium	Regular reanalysis (monthly/quarterly)	Business Analyst
Low-quality transaction data	High	Medium	Data validation, cleansing procedures	Data Team
Over-reliance on historical patterns	Medium	High	Combine with other analytics methods	Analytics Team
Seasonal pattern variations	Medium	High	Segment analysis by season, rolling windows	Business Analyst
Privacy/compliance concerns	High	Low	Anonymize data, follow regulations (GDPR)	Legal/Compliance
Misinterpretation of results	Medium	Medium	Clear documentation, training, examples	Product Owner

11.3 Operational Risks

Risk	Impact	Probability	Mitigation	Owner
User error in threshold setting	Medium	High	Default values, validation, documentation	Product Owner
Insufficient training	Medium	Medium	Training sessions, documentation, examples	Training Team
Tool abandonment after initial use	Medium	Medium	Regular check-ins, success stories, support	Product Manager
Conflicting recommendations	Low	Low	Prioritization logic, business rules	Business Analyst

11.4 Data Risks

Risk	Impact	Probability	Mitigation	Owner
Insufficient transaction volume	High	Medium	Define minimum data requirements (1000+ transactions)	Data Team
Transaction data not representative	High	Medium	Validate sampling, check for biases	Business Analyst
Item naming inconsistencies	Medium	High	Standardization process, data cleansing	Data Team
Duplicate transactions	Low	Low	Deduplication logic, validation checks	Dev Team

---

12. Future Enhancements

12.1 Phase 2 Features (Q1 2026)

Priority: High

- [ ] CSV export of discovered rules
- [ ] JSON output format option
- [ ] Configurable rule patterns (1→1, 2→1, 1→2, etc.)
- [ ] Temporal analysis (day of week, time of day patterns)
- [ ] Customer segmentation by purchase patterns
- [ ] Visualization of rule networks (graph display)

Priority: Medium

- [ ] Web-based interface for parameter tuning
- [ ] Batch processing of multiple datasets
- [ ] Rule comparison across time periods
- [ ] Statistical significance testing (chi-square)
- [ ] Confidence intervals for metrics
- [ ] Integration with pandas DataFrames

Priority: Low

- [ ] Email notification on completion
- [ ] Scheduled automated runs
- [ ] Rule filtering by specific items
- [ ] Negative association rules (items that don't co-occur)
- [ ] Multi-level association rules (hierarchical products)

12.2 Advanced Features (Version 2.0 - Q3 2026)

Algorithm Enhancements:

- FP-Growth algorithm implementation (faster than Apriori)
- Parallel processing with multiprocessing
- Incremental mining (update rules without full reprocessing)
- Dynamic threshold adjustment based on data distribution
- Weighted association rules (considering item prices/margins)

Business Intelligence:

- Real-time dashboard with key metrics
- Automated actionable insights generation
- A/B testing framework integration
- ROI calculator for recommended bundles
- Competitive analysis (compare with industry benchmarks)

Data Integration:

- Database connectivity (PostgreSQL, MySQL, SQLite)
- API endpoints for programmatic access
- Integration with e-commerce platforms (Shopify, WooCommerce)
- POS system direct integration
- CRM data enrichment

Advanced Analytics:

- Sequential pattern mining (order of purchases)
- Seasonal pattern detection
- Customer lifetime value prediction
- Churn prediction based on purchase patterns
- Market basket optimization algorithms

12.3 Enterprise Features (Version 3.0 - 2027)

- Multi-store analysis and comparison
- Cloud deployment (AWS Lambda, Azure Functions)
- Scalability to millions of transactions
- Real-time stream processing (Apache Kafka)
- Machine learning integration (predictive bundling)
- Mobile app for managers
- Advanced security and access control
- Multi-language support
- White-label customization

---

13. Documentation Requirements

13.1 User Documentation

Required Documents:

1. **Quick Start Guide** (1 page)
    - Installation steps
    - First run example
    - Basic interpretation
2. **User Manual** (10-15 pages)
    - Complete feature description
    - Configuration options

- Threshold tuning guide
- Troubleshooting section
- FAQ (10+ questions)

3. **Business Guide** (5 pages)
   - How to interpret results
   - Business actions for each metric
   - Case studies (3-5 examples)
   - ROI calculation examples
   - Best practices
4. **Tutorial Videos** (Optional)
   - Installation (5 min)
   - Running analysis (10 min)
   - Interpreting results (15 min)
   - Business applications (20 min)

13.2 Technical Documentation

Required Documents:

1. **Technical Specification** (This PRD)
   - Complete requirements
   - Algorithm details
   - API reference
2. **Developer Guide** (8-10 pages)
   - Code architecture
   - Function reference
   - Extension guide
   - Testing guide
   - Contributing guidelines
3. **Algorithm Documentation** (5 pages)
   - Mathematical formulas
   - Pseudocode
   - Complexity analysis
   - Optimization techniques
   - References to academic papers
4. **API Documentation** (Auto-generated)
   - Docstring extraction
   - Function signatures
   - Parameter descriptions
   - Return value specifications
   - Usage examples

13.3 Training Materials

Required Materials:

1. **Workshop Presentation** (30 slides)
   - Introduction to association rules
   - Business applications
   - Tool demonstration
   - Hands-on exercises
   - Q&A
2. **Exercise Workbook** (10 pages)
   - 5 practical exercises
   - Sample datasets
   - Step-by-step solutions
   - Discussion questions
3. **Cheat Sheet** (1 page)
   - Common commands
   - Metric interpretations

- Threshold guidelines
- Quick troubleshooting

13.4 Maintenance Documentation

1. **Change Log**
   - Version history
   - Feature additions
   - Bug fixes
   - Breaking changes
2. **Known Issues**
   - Current limitations
   - Workarounds
   - Planned fixes
3. **Release Notes**
   - Per-version summaries
   - Migration guides
   - Deprecation notices

---

14. Acceptance Criteria

14.1 Functional Acceptance

The system is accepted if:

✅ **Data Processing:**

- [x] Loads `grocery_dataset.txt` successfully
- [x] Handles 1,000+ transactions without errors
- [x] Parses comma-separated items correctly
- [x] Displays dataset statistics accurately

✅ **Itemset Mining:**

- [x] Discovers all frequent 1-itemsets (support ≥ 0.3%)
- [x] Discovers all frequent 2-itemsets (support ≥ 0.3%)
- [x] Discovers all frequent 3-itemsets (support ≥ 0.3%)
- [x] Discovers all frequent 5-itemsets (support ≥ 0.3%)
- [x] Support calculations accurate within 0.0001

✅ **Rule Generation:**

- [x] Generates all valid 3→2 rules from 5-itemsets
- [x] Applies confidence filter (≥ 40%) correctly
- [x] Applies lift filter (> 1.0) correctly
- [x] No duplicate rules in output
- [x] Rules sorted by lift (descending)

✅ **Calculations:**

- [x] Support calculated correctly (verified manually)
- [x] Confidence calculated correctly (verified manually)
- [x] Lift calculated correctly (verified manually)
- [x] All metrics within specified precision

✅ **Output:**

- ☑ Displays all sections in correct order
- ☑ Shows best rule by lift prominently
- ☑ Includes business insights and recommendations
- ☑ Summary statistics calculated correctly
- ☑ Top 5 rules displayed for each metric

14.2 Performance Acceptance

The system is accepted if:

✅ **Execution Time:**

- ☑ 1,000 transactions: < 10 seconds
- ☑ 10,000 transactions: < 60 seconds
- ☑ Processes average dataset (5,000 transactions) in < 40 seconds

✅ **Resource Usage:**

- ☑ Memory usage < 500 MB (standard datasets)
- ☑ CPU usage efficient (no infinite loops)
- ☑ No memory leaks during execution

✅ **Scalability:**

- ☑ Handles up to 50,000 transactions
- ☑ Supports up to 1,000 unique items
- ☑ Processes up to 10,000 frequent 5-itemsets

14.3 Quality Acceptance

The system is accepted if:

✅ **Code Quality:**

- ☑ All functions have docstrings
- ☑ Code follows PEP 8 style guide
- ☑ No critical or high-severity bugs
- ☑ Cyclomatic complexity < 10 per function

✅ **Testing:**

- ☑ Unit test coverage > 80%
- ☑ All critical paths tested
- ☑ Edge cases handled properly
- ☑ Performance benchmarks meet targets

✅ **Documentation:**

- ☑ README.md complete and accurate
- ☑ Inline comments explain complex logic
- ☑ User guide provides clear instructions
- ☑ Examples work as documented

✅ **Usability:**

- ☑ Single-command execution
- ☑ Clear error messages with solutions

- ☑ Output easy to understand
- ☑ No technical jargon in business sections

14.4 Business Acceptance

The system is accepted if:

✅ **Business Value:**

- ☑ Discovers actionable rules (confidence ≥ 40%)
- ☑ Identifies highest-lift rule for immediate action
- ☑ Provides clear business recommendations
- ☑ Results align with domain expert expectations

✅ **Stakeholder Satisfaction:**

- ☑ Retail analysts can use tool independently
- ☑ Marketing team understands rule interpretations
- ☑ Store managers can implement recommendations
- ☑ Executive team sees clear ROI potential

---

15. Deployment Plan

15.1 Deployment Phases

Phase 1: Internal Testing (Week 1-2)

- Deploy to development team
- Internal testing and bug fixes
- Documentation review
- Performance benchmarking

Phase 2: Pilot Release (Week 3-4)

- Deploy to 3-5 retail analysts
- Collect feedback on usability
- Validate business insights
- Refine output format

Phase 3: Limited Release (Week 5-6)

- Deploy to full analytics team (10-15 users)
- Conduct training sessions
- Monitor usage patterns
- Provide hands-on support

Phase 4: General Availability (Week 7+)

- Open to all authorized users
- Publish documentation
- Ongoing support and maintenance
- Quarterly enhancement releases

15.2 Training Plan

Training Sessions:

1. **Technical Training** (2 hours)

    - Target: Data analysts, data scientists
    - Content: Installation, configuration, execution
    - Hands-on: Running analysis on sample data
2. **Business Training** (2 hours)
    - Target: Marketing, merchandising, store managers
    - Content: Interpreting results, taking action
    - Hands-on: Case studies and group exercises
3. **Advanced Workshop** (4 hours)
    - Target: Power users
    - Content: Threshold tuning, advanced scenarios
    - Hands-on: Real dataset analysis

15.3 Support Plan

Support Channels:

- Email: analytics-support@company.com
- Slack: #association-rules-help
- Documentation: Internal wiki
- Office hours: Tuesdays 2-4 PM

Support SLAs:

- Critical issues: 4-hour response
- High priority: 24-hour response
- Medium/low: 48-hour response

---

16. Approval and Sign-off

16.1 Stakeholder Approval

Role	Name	Signature	Date
Author	Yair Levi	_____	Oct 15, 2025
Technical Lead	_____	_____	_____
Business Owner	_____	_____	_____
Retail Analytics Manager	_____	_____	_____
Marketing Director	_____	_____	_____
IT Manager	_____	_____	_____

16.2 Review History

Version	Date	Reviewer	Status	Comments
0.1	Oct 8, 2025	Yair Levi	Draft	Initial requirements
0.5	Oct 12, 2025	Tech Team	Review	Technical feedback
0.8	Oct 14, 2025	Business Team	Review	Business requirements
1.0	Oct 15, 2025	All	Approved	Final version

---

17. Revision History

Version	Date	Author	Changes
0.1	Oct 8, 2025	Yair Levi	Initial draft with core requirements
0.2	Oct 9, 2025	Yair Levi	Added use cases and technical specs
0.3	Oct 10, 2025	Yair Levi	Added performance requirements
0.4	Oct 11, 2025	Yair Levi	Added testing requirements
0.5	Oct 12, 2025	Yair Levi	Technical review feedback incorporated
0.6	Oct 13, 2025	Yair Levi	Added deployment and training plans
0.7	Oct 14, 2025	Yair Levi	Business review feedback incorporated
0.8	Oct 14, 2025	Yair Levi	Added acceptance criteria
0.9	Oct 15, 2025	Yair Levi	Final review and polish
1.0	Oct 15, 2025	Yair Levi	Approved final version

Appendix A: Mathematical Formulas

Support Formula

$$\text{Support}(\{A,B,C,D,E\}) = |\{t \in T \mid \{A,B,C,D,E\} \subseteq t\}| / |T|$$

Where:
- T = set of all transactions
- $\{A,B,C,D,E\}$ = 5-itemset
- $|\cdot|$ = cardinality (count)
- $\subseteq$ = subset relation

Example:
If 15 out of 5000 transactions contain {bread, milk, eggs, butter, cheese}
Support = 15/5000 = 0.003 = 0.3%

Confidence Formula

Confidence({A,B,C} → {D,E}) = Support({A,B,C,D,E}) / Support({A,B,C})
= P(D,E | A,B,C)

Where:
- {A,B,C} = antecedent (3 items)
- {D,E} = consequent (2 items)
- P(D,E | A,B,C) = conditional probability

Example:
Support({bread, milk, eggs, butter, cheese}) = 0.003
Support({bread, milk, eggs}) = 0.007
Confidence = 0.003 / 0.007 = 0.4286 = 42.86%

Interpretation: When customers buy {bread, milk, eggs},
they buy {butter, cheese} 42.86% of the time

Lift Formula

Lift({A,B,C} → {D,E}) = Support({A,B,C,D,E}) / (Support({A,B,C}) × Support({D,E}))
= Confidence({A,B,C} → {D,E}) / Support({D,E})
= P(A,B,C,D,E) / (P(A,B,C) × P(D,E))

Interpretation:
- Lift > 1: Positive correlation (occur together more than random)
- Lift = 1: Independence (no correlation)
- Lift < 1: Negative correlation (rarely occur together)

Example:
Support({bread, milk, eggs, butter, cheese}) = 0.003
Support({bread, milk, eggs}) = 0.007
Support({butter, cheese}) = 0.015
Lift = 0.003 / (0.007 × 0.015) = 0.003 / 0.000105 = 28.57

Interpretation: These 5 items appear together 28.57× more often than
expected by chance - very strong association!

Appendix B: Sample Dataset Format

Valid Dataset Example

bread, milk, eggs, butter, cheese
coffee, yogurt, cereal, milk, orange juice
chicken, rice, vegetables, soy sauce, ginger
pasta, tomato sauce, ground beef, parmesan cheese, olive oil
apples, bananas, grapes, oranges, strawberries
bread, jam, peanut butter, honey, butter
yogurt, granola, berries, honey, nuts
lettuce, tomatoes, cucumbers, carrots, dressing
cheese, crackers, wine, olives, prosciutto
ice cream, cookies, chocolate syrup, whipped cream, sprinkles

Dataset Statistics

- Transactions: 10
- Average items per transaction: 5
- Unique items: 50
- Minimum items: 5
- Maximum items: 5

Requirements for Valid Dataset

1. **Minimum Transaction Count:** 100+ (for statistical significance)
2. **Minimum Items Per Transaction:** 5 (to form 5-itemsets)
3. **Recommended Average:** 8-15 items per transaction
4. **Item Naming:** Consistent, no leading/trailing spaces
5. **Delimiter:** Comma only
6. **Encoding:** UTF-8

Appendix C: Threshold Tuning Guide

Support Threshold Guidelines

Support	Use Case	Expected Rules	Dataset Size
0.1%	Very rare patterns	100-500	10,000+ transactions
0.3%	**Rare but meaningful**	**20-100**	**5,000+ transactions**
0.5%	Uncommon patterns	10-50	3,000+ transactions
1.0%	Moderately common	5-20	1,000+ transactions
3.0%	Common patterns	1-10	500+ transactions
5.0%	Very common only	0-5	Any size

Recommendation: Start with 0.3% and adjust based on results

Confidence Threshold Guidelines

Confidence	Interpretation	Use Case
30%	Low reliability	Exploratory analysis
40%	**Moderate reliability**	**Standard recommendations**
50%	Good reliability	Marketing campaigns
60%	High reliability	Product bundles
70%	Very high reliability	Guaranteed offers
80%+	Near certainty	Premium bundles

Recommendation: Start with 40% for discovery, increase for action

Lift Threshold Guidelines

Lift	Interpretation	Action
> 1.0	Positive correlation	Investigate
> 1.5	Moderate association	Consider action
> 2.0	Strong association	Implement
> 3.0	Very strong association	Priority action
> 5.0	Exceptional association	Immediate action

Recommendation: Always use > 1.0, increase for prioritization

---

Document Control:

- Classification: Internal Use
- Distribution: Analytics Team, Management, Development Team
- Review Cycle: Quarterly or as needed
- Next Review: January 15, 2026
- Document Owner: Yair Levi
- Maintained By: Analytics Team