Deliverable 1: Final Year Dissertation

Sahil Pattni

BSc. Computer Science (Honours)

Supervisor: Neamat El Gayar

December 10, 2020

Declaration

I, Sahil Manojkumar Pattni, confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: Sahil Manojkumar Pattni

Date: December 10, 2020

Abstract

In this digital age, data is being generated at an exponential rate, with data analytics being used by corporations and small businesses alike to produce insights, reduce costs, optimize operations and increase profits. Generating association rules allows us to find non-intuitive associations that bring new insights to the management, allowing them to leverage this information to maximize their profits. A prime example would be the *'Beers and Diapers'* urban legend, where a company looked at their point-of-sale data and found a strong association between beers and diapers being co-purchased, which seems rather unintuitive.

In this study, the author will extract a minimum spanning tree (MST) from a data set using machine learning techniques. The author will then use this minimum spanning tree to segment products together, and produce association rules and extract the most interesting ones. The resulting ruleset will be compared to rules generated by the established Apriori Algorithm [1], which is the foundation of association rule mining. Additionally, the author will study how the grouping of the MST compares to clustering algorithms, and how the structure of our MST differs when the dataset is limited to a given city (as opposed to the overall MST), leading to insights that may help the management in such firms make better informed decisions about the type of promotions they would like to run, and the most optimal locations for such.

Contents

1 Introduction

1.1 Context

For businesses which deal with the sale of a heterogeneous physical assets - such as groceries, hypermarkets and select retail outlets - operations such as inventory management and product placement play an instrumental role in determining the business' financial success. These involve asking questions such as:

- Which products should be placed at the entrance of the store? Which should be placed closer to the exit?
- Which products will benefit the most by being placed at eye-level?
- Which products should be placed next to each other to maximize the purchase volume?

One way to find optimal solutions for such questions is to employ the use of Association Rule Mining (also known as Market Basket Analysis). This set of techniques assess frequent itemsets (e.g. from sales data) and generate association rules between products. Several algorithms and techniques exist for association rule mining, such as the Apriori Algorithm and FP-Growth [2]. One problem with these algorithms is that they tend to generate an enormous amount of rules, of which many of the rules themselves are large. This makes it grossly inconvenient for the end-user to retrieve any actionable information from the results.
The author construct a network of products - where each vertice represents a product or product category, and an edge between two vertices represents the simultaneous occurrence of the the two products. The result is a graph (i.e. a network), whose architecture is visually informative of the products and their associations. From this graph, a minimum spanning tree (MST) can be extracted, which will prune the graph down to only the strongest connections. The insights that can be gained from the MST can help the management of the mentioned businesses to maximize their profit. Furthermore, association rules can now be extracted from the MST.

1.2 Aims

The aim of this study is to study the effectiveness of a minimum spanning tree as a market basket analysis tool. Additionally, this dissertation will study how the architecture of our minimum spanning tree changes when the dataset is limited to a certain city. The model will be built from a large transactional database.

1.3 Objectives

The research objectives for this project have been laid out below, in the order that they will be carried out.

1. Acquire a suitable dataset upon which the MST can be constructed.
2. Explore and evaluate MST extraction algorithms.
3. Generate an undirected graph $G(E, V)$ where the edges E are the correlation values between the product vertices.
4. Extract a minimum spanning tree from this graph using the technique determined in step 2.
5. Analyze this MST and use its architecture to determine product clusters and generate association rules.
6. Generate association rules using the Apriori and FP-Growth algorithm and compare them in both their time complexity and their *'interestingness'* (i.e. the unintuitive rules they generate).
7. Validate the product grouping present in the MST against parent categories (if present), and a clustering algorithm (e.g. such as the K-Means algorithm).

2 Background

2.1 Core Concepts

Graphs

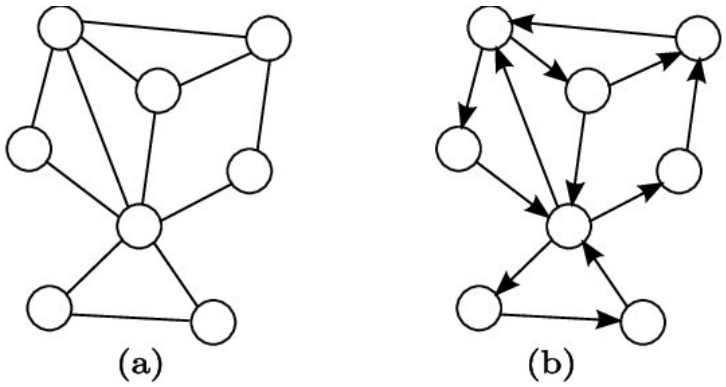

Figure 1: Undirected and Directed Graphs

In discrete mathematics and more specifically - graph theory, a graph is a data structure that contains a set of nodes (i.e. vertices) connected by lines (i.e. edges). These edges may be directed - such as in Figure 1:(a), or undirected - such as in Figure 1:(b). A graph G with a set of vertices V and a set of edges E can be represented via the notation $G = (V, E)$. For the scope of this dissertation, the author will be building undirected graphs, where the weight between two vertices is the same in both directions.

2.1.1 Minimum Spanning Trees

Given an undirected $G = (V, E)$, a *spanning tree* can be described as a subgraph that is a tree which includes all the vertices V of G with the minimum number of edges required. A *minimum spanning tree* (MST) is the spanning tree with the smallest sum of edge weights. This means that if the graph has n vertices, each spanning tree - including the minimum spanning tree - will have $n - 1$ edges. There are two widely used algorithms to extract the minimum spanning tree from a graph: Prim's algorithm and Kruskal's algorithm.

2.1.2 Prim's Algorithm

Independently discovered by three authors, Prim's algorithm [3][4][5] is a greedy algorithm[1] to find the minimum spanning three of an undirected, weighted graph G. To successfully

[1]Selecting the locally optimal choice at each iteration of the solution

implement the algorithm, three sets need to maintained: a set of *discovered* edges, and two sets of vertices: a set of *undiscovered* vertices, and a set of *discovered* vertices. Figure 2 illustrates Prim's algorithm being applied to a graph. The algorithm is as follows:
Initialize an empty set of discovered edges: E, and two sets of vertices: an empty set D of the discovered vertices, and UD as the set of undiscovered vertices.

- Pick an arbitrary vertex as a starting point (in the case of Figure 2, the top right node) Add this vertex to D and remove it from UD.
- While UD is not empty:
  - Find the edge e_i with the smallest weight such that it connects together a vertex V_i in D and V_j in UD (to avoid forming cycles).
  - Append V_j to D and remove it from UD (i.e. V_j is now discovered).
  - Append e_i to E.

Once D contains all the vertices of G, the algorithm terminates, and the set D represents the minimum spanning tree, and $\sum_{i=1}^{n} e_i$ is the weight of the MST. The time complexity of this algorithm is $O(V^2)$.

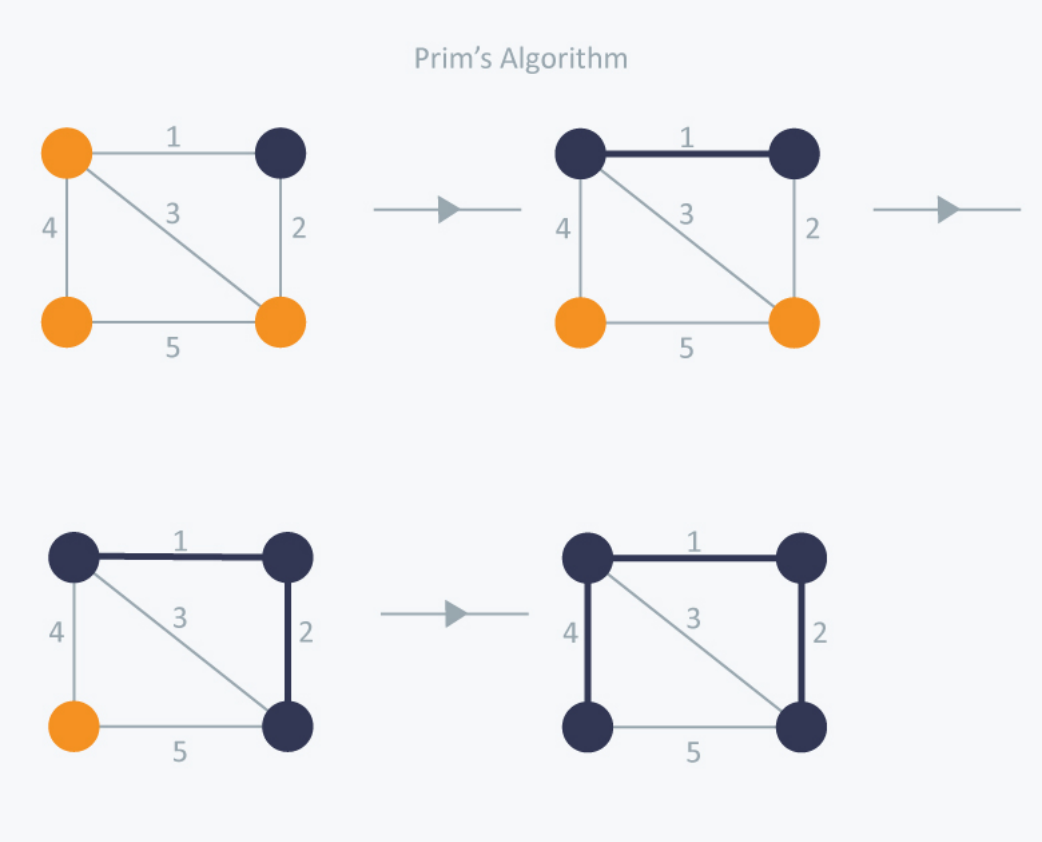

Figure 2: Prim's Algorithm applied to a graph

2.1.3 Kruskal's Algorithm

Another greedy algorithm, Kruskal's algorithm [6] also extracts the MST from a graph. Unlike Prim's algorithm, Kruskal's doesn't select an edge that connects directly to the already build spanning tree, but rather picks the global optimal solution. The algorithm is as follows: Maintaining a set of edges E, and an initially empty set of chosen edges C:

- Sort the set of edges E in ascending order.
- While the number of elements in C is not $n - 1$:
  - Select the smallest edge e_i from E.
  - If adding it does not form a cycle with the spanning tree formed so far, append e_i to C.

The algorithm will terminate when $n - 1$ edges have been selected. The time complexity for this algorithm is $O(ElogE)$.

2.1.4 Clustering and K-Means

Illustrated in Figure 3, clustering is the act of segregating data into a number of groups (i.e. clusters) such that a data point in a cluster is similar to other data points in that cluster.

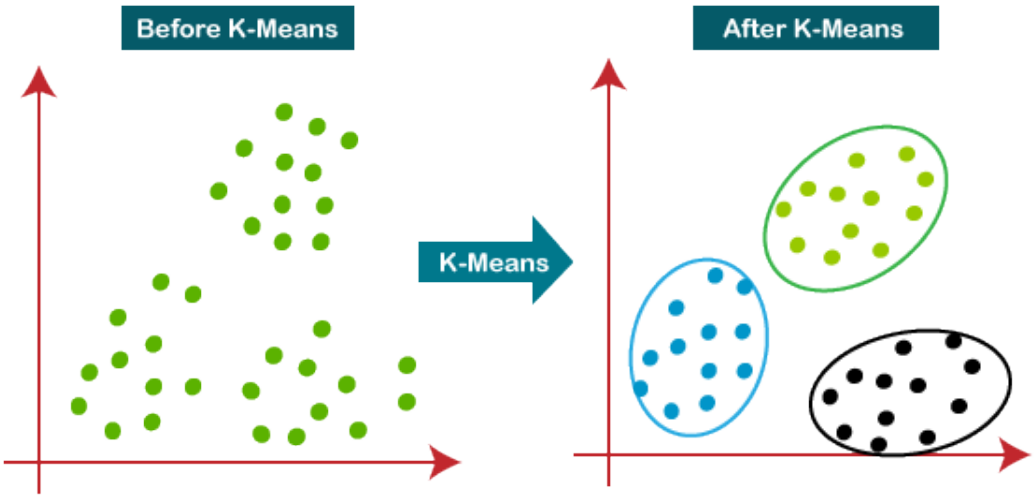

Figure 3: An example of clustering

K-Means Clustering [7] is an unsupervised machine learning algorithm where the data is partitioned into k clusters, where each observation belongs to the cluster with the nearest mean (i.e. centroid).

2.1.5 Binary Purchase Vectors

Binary purchase vectors play an instrumental role in the construction of the MST, helping us determine the association between products. What is a binary purchase vector, however? It is a vector whose length is the number of unique products in the dataset, where each element is a 1 or 0 depending on whether the product was purchasd or not. Imagine a grocer who only sells five items: milk, eggs, bread, apples and oranges. If a customer purchases bread and oranges, the binary purchase vector for that transaction would look like this:

milk	eggs	bread	apples	oranges
0	0	1	0	1

2.1.6 Market Basket Analysis and Apriori Rule

Market Basket Analysis (MBA), also known as Affinity Analysis, is a data mining and analysis technique that identifies co-occurrence patterns between products purchased together, and produces association rules for these products as such: $\{A\} \rightarrow \{B\}$ which implies a strong relationship between the purchase of product A and product B. In the case of our *beers and pampers* example, the association rule could be represented as $\{Beers\} \rightarrow \{Pampers\}$. The contents of a purchase basket (i.e. the contents of a customer's basket when they check out) is called a *itemset*, which - as the name suggests - is a set of all the items in the basket. For example, if a customer had bought detergent, bread and soda, the itemset would be $\{bread,\ detergent,\ soda\}$. Association rules are generated by looking at different combinations of the itemset (e.g. $\{bread,\ soda\} \rightarrow \{detergent\}$ and $\{soda\} \rightarrow \{bread\}$). The equation [8] to calculate the number of rules for an itemset of length d is as follows :

$$number\ of\ rules = \sum_{k=1}^{d-1} \left(\binom{d}{k} \times \sum_{j=1}^{d-k} \binom{d-k}{j} \right) \tag{1}$$

Analyzing the equation, it becomes apparent that the problem with association rule generation from itemsets is that the number of rules produced grows exponentially with the size of the itemset, as can be seen in Figure 4, using Equation 1.

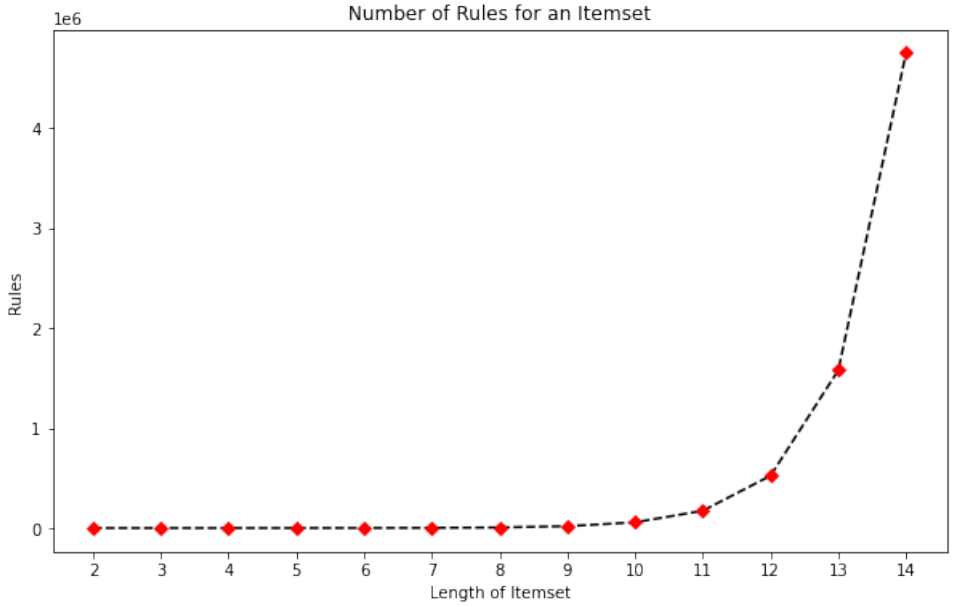

Figure 4: Number of Association Rules for an Itemset

Clearly, checking all itemsets in a database would be computationally expensive, and so

only the *frequent itemsets* - itemsets that have a support value above a minimum threshold $min_{support}$ - will be considered. Even so, checking each itemset's support score via brute force is unacceptably time consuming, therefore the search for frequent itemsets can be optimized using the Apriori Principle [9], which states that *all subsets of a frequent itemset must be frequent.* To understand how the Apriori Principle can be applied here, we first look at three key metrics:

Support

The support of a set of products is the fraction of transactions in which these set of products are present. For example, for a list of transactions T, for the product X where $T(X)$ denotes that the set of transactions in which X was present:

$$support(\{A\}) = \frac{T(A)}{T}$$

and similarly,

$$support(\{A\} \rightarrow \{B\}) = \frac{T(A,B)}{T}$$

denotes the support for the rule $\{A\} \rightarrow \{B\}$. We can prune rules with a low support as they indicate a rule does not occur enough to draw any reasonable conclusions from.

Confidence

Confidence is the measure of how likely a product will be in a basket given that another product is in it. That is to say, the confidence of a rule $\{A\} \rightarrow \{B\}$ is the conditional probability that $\{B\}$ occurs in the basket given that $\{A\}$ is present. The confidence of a rule can be denoted as:

$$confidence(\{A\} \rightarrow \{B\}) = \frac{T(A,B)}{T(A)} \equiv \frac{support(\{A\} \rightarrow \{B\})}{support(\{A\})}$$

Lift

The lift of a rule $\{A\} \rightarrow \{B\}$ is the rise (i.e. **lift**) that $\{A\}$ gives to our $confidence(\{A\} \rightarrow \{B\})$.

$$lift(\{A\} \rightarrow \{B\}) = \frac{confidence(\{A\} \rightarrow \{B\})}{support(\{B\})}$$

To make this easier to understand, imagine that $confidence(\{A\} \rightarrow \{B\}) = 0.5$, and $support(\{B\}) = 0.4$. This means that the presence of $\{A\}$ increases the probability of $\{B\}$ being in the same basket by 25% ($\frac{0.5}{0.4} = 1.25$), therefore providing us with a lift value of 1.25. A lift value below 1 would indicate that the occurrence of $\{A\}$ in a basket decreases the likelihood of $\{B\}$ occurring

in the same basket (i.e. a low product association).

As we established above, the Apriori Principle states that all subsets of a frequent itemset must be frequent. The Apriori Principle is a result of the *anti-monotone property of support* [10], which means for $\{A, B, C\}$, $support(\{A, B\}) \geq support(\{A, B, C\})$. We can use this to prune the frequent itemsets much more efficiently, because if $support(\{A, B\}) < min_{support}$, then any itemset containing the set $\{A, B\}$ will also fall below $min_{support}$. Once the frequent itemsets have been pruned, association rules can be generated from the remaining itemsets. The resulting association rules can be even further pruned by removing those that fall below a confidence threshold $min_{confidence}$. Finally, the remaining rules can be ranked according to their lift to find the rules with the highest associations. Figure 5 is an illustration of how pruning one set $\{A, B\}$ leads to the pruning of all its children.

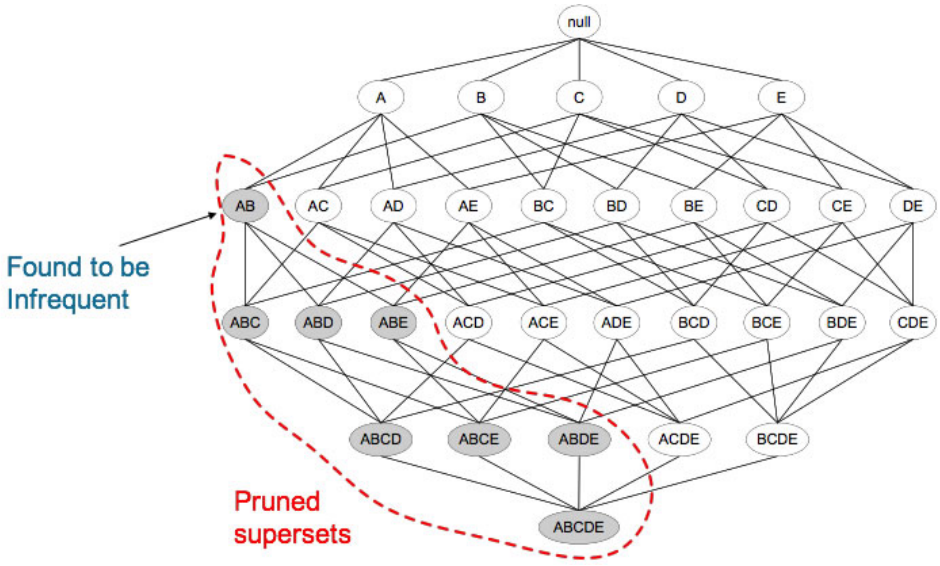

Figure 5: Pruning Infrequent Rules

2.2 Related Work

C. Zhong et al [11] proposed a novel framework to extract the minimum spanning tree of a graph based on the K-Means clustering algorithm. Their methodology can be separated into two distinct phases. In the first phase, the data is partitioned into $\sqrt{n}$ clusters via K-Means and the Kruskal's algorithm is applied to each of the clusters individually. Once the $\sqrt{n}$ MSTs have been constructed, they are combined to form an approximate MST. In the second phase, new partitions are constructed based on the borders of the clusters identified in phase 1. Based on these new partitions, a second approximate MST is constructed. Finally, both graphs are

merged such that the resulting graph has $2(n-1)$ edges. The Kruskal's algorithm is run on this graph to get the final approximation of the MST.

Critical Analysis

The authors have proposed an efficient way to approximate an MST, with their methodology having a complexity of $O(N^{1.5})$, which is faster than the standard MST algorithm which has a complexity of $O(N^2)$. This may not intuitively seem like a large decrease, but as Figure 6 illustrates, there is a significant increase in efficiency over the standard algorithm.

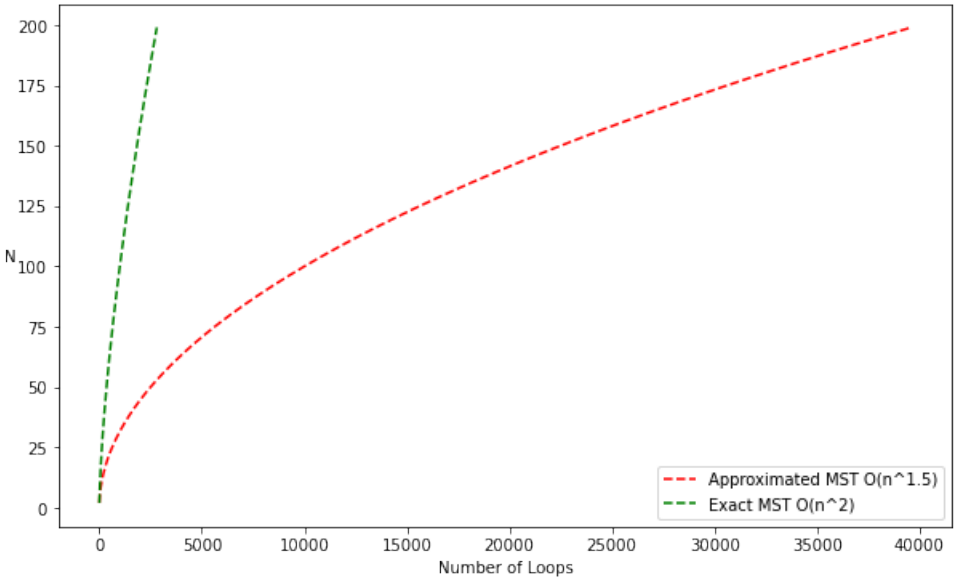

Figure 6: Efficiency of K-Means optimized MST vs. exact MST

The K-Means optimized algorithm is $\frac{\sqrt{N}-1}{\sqrt{N}}\%$ faster than the standard MST algorithm.

R. Aggarwal et al. [9] proposed a novel algorithm to generate all statistically significant association rules between items in a database, laying the foundations for association rule mining. Given a set of items $I = I_1, I_2, I_3, ...I_m$, the authors define an association rule to be of the form $X \rightarrow I_j$ where X is a set of items such that $X \in I, I_j \notin X$. The hypothetical database stated was a list of transactions, T, where each transaction t was a binary vector of length m, representing a basket purchase, where $t[k] = 1$ if I_k been purchased in that basket. The authors stated that their methodology for association rule mining could be split into two steps: the generation of candidate itemsets, and the generation of statistically significant association rules from these itemsets.

To address the first subproblem, the authors provided the pseudo-code for their candidate

itemset generation, where all itemsets possible were generated from tuples (samples) from the database, and those itemsets whose support score[2] is above the minimum support threshold are considered candidates (called *large itemsets*). Since a brute-force check would be sub-optimal (the authors note this could take up to 2^m passes of the database), the authors devised a methodology to check for candidate itemsets where on the k^{th} pass of the database, they would only check itemsets of length k, to see if they satisfied the support constraint. On the $(k+1)^{th}$ pass, they need only check those itemsets that are 1-extensions (i.e. itemsets extended by exactly one item) of the *large itemsets* found in the previous pass. This is because of what would later be known as the *Apriori Principle*, where if an itemset Y is *large*, then all subsets of Y must also be *large*. Therefore, if they found the itemset $\{A, B\}$ was *small* (i.e. did not satisfy the support constraint), then sets containing $\{A, B\}$ (e.g. $\{A, B, C\}, \{A, B, D\}, \{A, B, C, D\}$) would also be small, and need not be checked. This means, however, that if an itemset I is *large*, then another pass over the dataset would be required to check the support of the subsets of I. To avoid this, the authors devised a measure to calculate the expected support, $\bar{s}$, of an itemset, and use this to measure the support of itemsets $I = (X + I_j)$ not only where I is expected to be large, but also where $X + I_j$ is expected to be small but X is expected to be large, further helping them prune the number of itemsets to check. The authors proceed to define a method that allows the algorithm to be more memory efficient[3]. The authors also defined method to further prune itemsets from the search, namely *remaining tuples optimization* and *pruning function optimization.*

To address the second subprobem, the authors proposed the following methodology: for each large itemset $Y = I_1, I_2, ...I_k, k \geq 2$ from the set of non-pruned large itemsets, generate a set of association rules of form $X \rightarrow I_j$ such that the consequent is I_j and the antecedent (i.e. the precedent set in the rule) is a subset X of Y such that X is of length $k - 1$ and $I_j \notin X$. Therefore, each large itemset will produce k rules. From the generated rules, the authors discarded those rules whose confidence scores[4] fell below the minimum confidence threshold c.

The authors tested their methodology on a sales dataset with $46,783$ transactions, with 63 *items* (in this case, the department from which the customer bought an item). They used a configuration of a minimum support threshold of 1% and a minimum confidence threshold of

[2]see: Section 2.1.6

[3]This may no longer be required due to the advances in computational power in the the 27 years since this paper was written.

[4]see: Section 2.1.6

50%. The rules produced seemed to follow with general intuition, such as:

{Auto Accessories, Tires} → {Automotive Services}

Furthermore, the authors assessed the accuracy of their expectation method, by measuring the ratio of correctly estimated itemsets for both small and large, against various values for the minimum support threshold, and visualizing the result. To isolate the effect of their expectation method, they disabled their pruning optimization functions. They were able to conclude that their estimation accuracy was satisfactory, as their accuracy was above 98% for support thresholds except the first, where it was 96%. The authors also tested the effectiveness of their pruning optimization functions, namely the *remaining tuples* and the *pruning function* optimization functions, against multiple minimum support threshold values. They were able to conclude that their pruning efficiency increased as the support threshold increased.

Critical Analysis

The authors have proposed a novel methodology that has been the bedrock of numerous research publications, including most of the papers in this literature review. Their estimation function performed with high accuracy, meaning it can reduce the number of passes through a database the algorithm has to take by a significant amount. Additionally, their pruning techniques allowed them to eliminate a large proportion of itemsets from the space.

M. Zekić-Sušac et al. [12] proposed a novel measure for the *interestingness* of association rules, identifying that a dominant, universally used measure did not exist. The authors' goal was to combine objective measures such as the support, confidence and lift scores with more subjective measures. Instead of the Apriori approach, their methodology has them generate association rules via the *tree-building technique* - which compresses a large database into a Frequent-Pattern tree, citing that this technique was more efficient than the Apriori algorithm. The author's employed the heuristical unexpectedness measure(i.e. significantly contradicting a user's prior beliefs) and the heuristical actionability measure (i.e. if the end user can use the information to their advantage(e.g. a promotion)) as their subjective measures, and a minimum confidence threshold of 51% as their subjective measure. Since a subjective measure would require a human subject, the authors' used the estimations of a sales manager from a Croatian retail chain, and stored his responses in binary format for the subjective measures (i.e. 0 if a rule was unexpected else 1, 0 if a rule is not useful else 1). The dataset used for this paper was a real transactional dataset with 14,012 transactions and a set of 1,230 unique items (which

was later pruned to 7,006 transactions and a set of 278 products) from the same Croatian retail chain their test subject worked at. The authors then generated association rules from the first-level hierarchical grouping of items from the dataset (items with a minimum support of 25%), of which 36 rules were identified as statistically significant. From this set of rules, only two rules satisfied both subjective measures and the confidence constraint, and therefore these two rules were identified as highly interesting. The authors then generated association rules from the second-level hierarchical grouping of items, where items that represented the same product (but had different a manufacturer, brand etc.) were grouped together. Of the rules generated, 15 satisfied their confidence constraints and had a support value able 10%. 5 rules from this set satisfied both their subjective and objective measures, more than the previous experiment The authors were able to conclude that the increase in accuracy and number of interesting rules resulted from the second level of grouping which generalized the products.

Critical Analysis

The authors have presented a well thought out approach to combining the subjective metrics with objective ones to produce a human-verified association rule set. A few caveats to note, however: their study only involved one subject, which is rarely regarded to be statistically acceptable. An ideal study would require multiple, randomly chosen subjects to offset any bias that the singular subject would have had, and in addition, the larger their subject size, the closer their collective estimations will model the total population's. Another drawback of their approach is that by using human intuition as a metric, they're promoting association rules that satisfy pre-existing notions about human behavior (e.g. if someone buys milk, they'll *probably* get eggs too), however these types of rules are usually regarded as common knowledge, whereas the beauty of association rule mining is in its ability to surface association rules that - while true - seem unintuitive, and therefore are less likely to be known by the management of such organizations.

M. A. Valle et al. [13] proposed a novel methodology to study the structure and behavior of consumer market baskets from the topology of a minimum spanning tree which represented the interdependencies between products, and use this information to complement the association rule generation process. The input to their proposed methodology was a correlation matrix between the set of all one-hot encoded purchase vectors such that each vector denoted the presence or non-presence of each product from the dataset in that vector. The dataset used for the MST

construction was a list of $1,046,804$ transactions containing a set of $3,240$ unique products from a large supermarket chain branch in Santiago, Chile. When building this correlation matrix, the authors opted to use the Pearson's Coefficient (which is equivalent to the coefficient ϕ for binary data such as theirs) over the traditionally used Jaccard distance to compute the similarity between the binary product vectors, as the former provides both a positive and negative association between products. Additionally, they used the distance function $d_{ij} = \sqrt{2(1-\phi_{ij}}$ to transform the correlation matrix into a distance measurement (i.e. the weight of the edges). The authors constructed a MST for 220 product subcategories, and noted that there was a significant level of grouping between product sub-categories that belonged to the same parent category. To remove edges from the MST that were not statistically significant, the authors used the mutual information [14] measure $\sum_{x,y} log_2 \frac{r(x,y)}{p(x)q(y)}$ between product subcategories p and q, and were able to prune 14 edges, all of which were connected to a terminal node, therefore effectively pruning 14 vertices from the MST too. To identify the most influential regions of the MST, the authors defined an influence zone of distances that were in the 10^{th} percentile. To generate meaningful association rules, for each MST product i, the authors ran a search for the set of all association rules R_i such that $P_i \rightarrow P_j (i \neq q)$. Then from the resulting set of rules, they searched for rules that obeyed $P_i \rightarrow P_m$ where m a product node connected to the product i in the minimum spanning tree. For both resulting sets of rules for each product, the mean of their lift scores were observed, and the authors determined that the rules that were reinforced by the MST had a higher mean, and that a majority of these rules had a lift score above the 90^{th} percentile.

To identify the clusters each of the products should be identified under, the authors constructed a hierarchical tree using the average linkage clustering method, and by using an unspecified cut distance, they were able to produce 17 taxonomic groups (i.e. clusters). Cross-referencing their results with the actual parent categories of the products, they were able to conclude that the MST did indeed categorize the product sub-categories into clusters with a reasonable degree of accuracy. The authors then compared their MST to another methodology, namely the structured association map (SAM) [15], using the Jaccard distance as a measure of similarity, and were able to generate interesting 2x2 rules (i.e. $\{A, B\} \rightarrow \{C, D\}$), all with lift scores above 1.0, with one rule even having a lift score of 106.46. They concluded that while both approaches provided different information, they both visually identify the strongest relationships between the products, and provide useful information to reduce the search space for association rules.

Critical Analysis

The authors' approach seems to be novel, thorough and well structured. Their methodology successfully employed the use of minimum spanning trees to complement the association rule generation process with sound results. One caveat of their approach is that they only used the MST to generate 1x1 rules (i.e. $\{A\} \rightarrow \{B\}$). Using the distance score in conjunction with the importance function they defined (i.e. $\sum_{k \in K_u} \frac{1}{w_{uk}}$), they could have defined a system to produce $p \times q$ rules (where $p \geq 1, q \geq 1$;i.e. rules that have one *or more* items in the antecedent and consequent), then rank them using their respective lifts. Additionally, while the authors did cross reference their clustering results against the real parent categories of the products, they did not compare their results to that of a clustering algorithm (e.g. K-Means), which would have given a reasonable benchmark to compare the results of the MST clustering to. While 1x1 rules are easily understandable and tend to have high lift values when extracted from the MST, $p \times q$ rules would provide a layer of insight as to how a range of products (perhaps a cluster) related to another.

3 Requirements & Research

3.1 Requirements Analysis

This section will identify both the functional and non-functional requirements required for the implementation of the model proposed in this document. By stating the specific requirements of the model, it serves as a high-level view of how the model should function.

Table 1: Functional Requirements

ID	Category	Description
FR1	Dataset	The dataset must not have any empty values.
FR2	Dataset	The dataset must have a unique identifier for each transaction.
FR3	Model	A list of binary purchase vectors must be extracted from the dataset.
FR4	Model	The model must return a storable data type representation of a MST.
FR5	Model	The model must produce $p \times q$ association rules.

Table 2: Non-Functional Requirements

ID	Category	Description
NFR1	Dataset	The dataset must contain at least 500,000 unique transactions.
NFR2	Hardware	The data inputs and the MST construction will run on a computer with at least 8 GB of ram, and a quad-core CPU with clock speed aabove 2.5GHz.

3.2 Research Methodology

3.2.1 Technical Choices

We will be building our model on Python 3.8. It is a versatile, fast language with both object-oriented and functional paradigms, with vast support and numerous libraries for data analytics and machine learning, many of which act as an interface for functions written in faster languages such as C/C++. We will be using the following libraries to help optimize our development process:

- pandas
- numpy

- scikit-learn
- matplotlib
- graphviz
- efficient-apriori

3.2.2 Implementation

The primary objective of this dissertation is to analyze the viability of a framework based on the minimum spanning tree as a 'one-in-all' tool for market basket analysis, from association rule generation to clustering. This section will describe how we plan to achieve these goals:
A candidate dataset has been identified: a dataset of $27,000,000$ transactions occurring across multiple branches of a Brazilian gas station store [16]. Each row in the dataset contains the purchase information for a particular product for a given transaction. The dataset has been checked for errors and inconsistencies, and any found have been remedied. Additionally, since the transaction id is an unreliable identifier for a market basket[5], we have assigned each row a unique alphanumeric code consisting of the purchase's transaction number, city and date, which will act as a robust unique identifier for each basket. F Additionally, we have removed redundant or unnecessary features from the dataset, this will allow us to make quicker passes through all the data when generating the binary purchase vectors. Once the dataset is prepared, we will need to extract a binary purchase matrix from the dataset. Since each product purchase has an assigned transaction number, to find all product purchases in a transaction, we will need one pass of the database. Therefore, to build n binary purchase vectors, we will need n passes of the database (i.e. a complexity of $O(n^2)$). This will prove to be quite computationally expensive, as there are approximately $5,200,000$ unique baskets in our dataset. Even when only considering $473,641$ baskets with 40 unique products, the resulting matrix (see Fig. 7) took over 6 hours to construct. We will be looking into more efficient ways to generate out input data.

[5] transaction numbers reset once they have reached an upper limit, transaction ids are not synced across stores so the same transaction id could refer to multiple transactions across different stores.

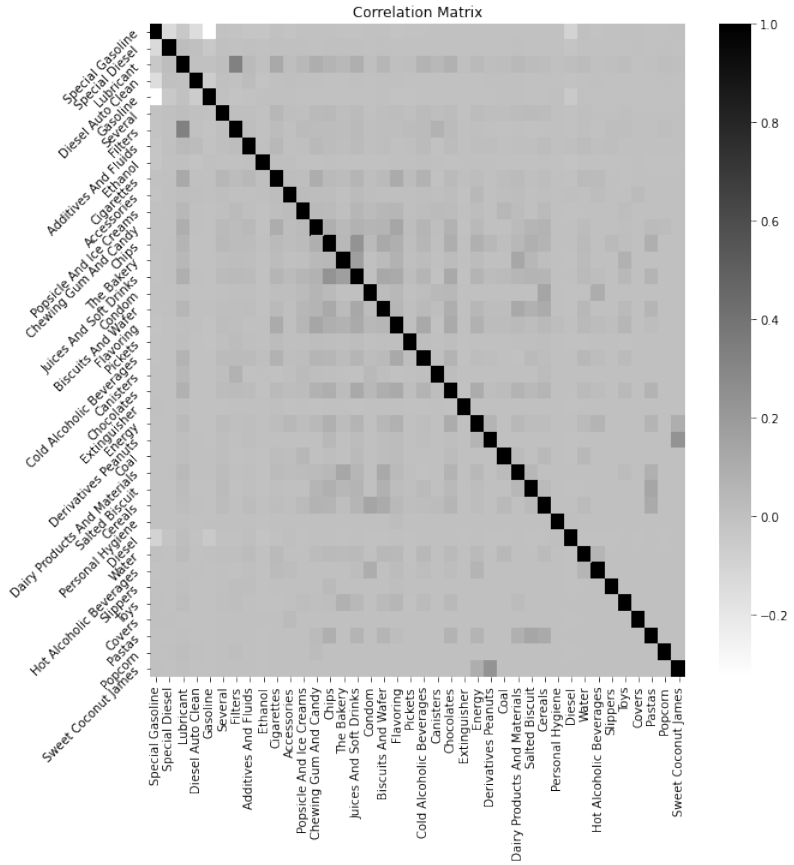

Figure 7: Correlation Matrix for 473,641 baskets

Once the correlation matrix has been generated and a distance function applied, a graph will have to be constructed. Given the size of our dataset, extracting a MST from our graph using traditional algorithms such as Prim's and Kruskal's may be ineffective, and we may have to look into more recent approaches. For example, **C. Zhong et al.** [11] proposed a divide-and-conquer algorithm for extracting a minimum spanning tree based on the K-Means clustering algorithm, which may be worth considering over Prim's and Kruskal's. Using the appropriate extraction method, we will retrieve our MST from the graph. We will then repeat the above methodology, but after aggregating the products (e.g. renaming all the brands of lubricant to 'lubricant', similar to the second-level hierarchical grouping in [12]) , which will allow us to study how these generalized products affect the structure of the MST. From the resulting minimum spanning trees, we will then be able to extract the most interesting association rules (i.e. those with the highest statistical significance). We will also repeat this process several times, limiting the dataset to a particular city each iteration to observe how the architecture of the MST differs by city as compared to the dataset as a whole.

4 Evaluation Strategy

Our model will need to evaluated to ensure that:

1. The model satisfies all the listed functional requirements listed in Table 1.
2. We are able to extract high-value association rules from the MST.
3. The MST architecture indicates a level of product grouping that is consistent with their parent subcategories.

Functional Requirement Validation

`FR1: The dataset must not have any empty values.`

To evaluate this, we can pass through the dataset, checking for empty/null values. The result of this evaluation is binary - no empty values found or more than 0 found.

`FR2: The dataset must have a unique identifier for each transaction.`

To evaluate this, we can a script to validate if each product purchase's associated transaction identifier matches the expected value. The result of this evaluation is also binary - either all unique transaction identifiers are as expected, or otherwise.

`FR3: A list of binary purchase vectors must be extracted from the dataset.`

Using the unique transaction identifier for each product in the database, we can generate our binary purchase vectors. To evaluate this, we can ascertain if the dimensions matrix of binary purchase vectors are consistent with our expected value.

Rule "Interestingness" Validation

There are two ways to validate the *interestingness* of an association rule:

- Objective measures: Metrics such as the support, confidence and lift scores will be used to rank the rule and evaluate its relative position against all the other rules. Additionally, we will be generating a greater number of association rule using the Apriori algorithm, and observing tour rule's relative position to these rules ranked by the aforementioned metrics.

- Subjective measures: Due to the ongoing pandemic, we will not be pursuing subjective measures to validate our rules, as subjective evaluation will require a third party to be present.

MST Grouping Validation

One way to validate the MST is to check the grouping of products in the MST. Ideally, grouping of similar products should be present (e.g. strong grouping between dairy products). We can set a threshold on the edge distance, such that any products within that threshold radius can be considered a cluster. One objective way to evaluate this cluster is to build a confusion matrix with the products against their parent categories and evaluate the proportion of products grouped together under their correct parent category. It would also be interesting to compare the results of the MST grouping to the clusters generated by an unsupervised clustering algorithm such as the K-Means clustering algorithm.

5 Project Management

To ensure that the project is of optimal quality, a plan must be charted out that specifies a timetable and deadlines for milestones in the prohect,

5.1 Methodology

There are several formal approaches for development planning, but the most optimal process for us is the Waterfall method [17], as our methodology can be defined as a linear, sequential workflow, where the requirements have been explicitly stated and change is unlikely.

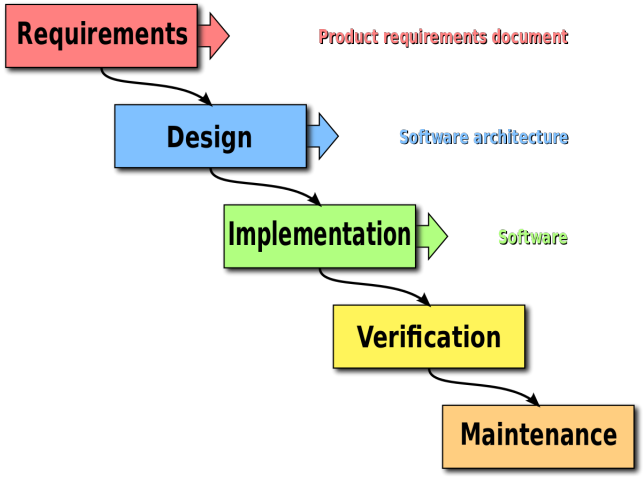

Figure 8: An outline of the waterfall method

5.2 Gantt Chart

(To be added)

5.3 Risk Analysis

The probabilities of our risks can be discretized and approximated to the following values:

Very Low	Low	Moderate	High	Very High

The impact of a given risk can be defined as follows:

Tolerable	Serious	Catastrophic

Given these metrics, we can describe our risk analysis as follows:

Risks				
ID	Risk	Probability	Impact	Priority
R1	Data loss/corruption	Low	Catastrophic	High
R2	Code loss/corruption	Low	Catastrophic	High
R3	Inability to meet deadlines	Low	Serious	Medium
R4	Degradation of health	Low	Serious	Low
R5	Incorrect estimation of model development time	Low	Serious	Low
R6	Unrealistic computation times	High	Catastrophic	High

The table above lists the risks identified with their probabilities, their impact, and the priority given to them. Each risk has given an alphanumeric ID to cross reference it against the table below, which describes the risk management for each of the risks identified.

Risk Management			
ID	Avoidance	Minimization	Contingency
R1	Store backups of datasets locally and on cloud storage at frequent intervals.	Do not overwrite existing datasets.	Retrieve the last checkpoint data.
R2	Maintain a git repository with two online remotes, with frequent commits.	Only commit when the code is known to work correctly.	Roll-back to a previous, working commit.
R3	Set milestone deadlines in the project plan.	Collaborate with supervisor to come up with a robust yet flexible project plan.	Restructure plan to offset any delays.
R4	Stay at home as much as possible, take necessary precautions outside.	Self-isolate and work to the best of my ability.	Restructure plan to offset delay, and if necessary, request an extension.
R5	Create realistic plan with reasonable tolerances to account for unexpected delays.	Attempt to finish all tasks within their appropriate period.	Restructure plan with updated tolerances.

Risk Management			
ID	Avoidance	Minimization	Contingency
R6	Perform computations on a small subset of the dataset.	Prototype model with very small dataset subsets and increase once the model is robust.	Reduce the dataset size until the computation is affordable.

5.4 Professional, Ethical, Legal & Social Issues

To the best of my knowledge, there are no professional, ethical, legal and social issues present. The only potential issue (both ethically and legally) was data anonymization, and the author of the dataset has explicitly stated [16] that the names of the products, customers and employees have been modified to be unidentifiable.

References

[1] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," *Proceedings of the 20th International Conference on Very Large Databases*, pp. 487–489, 1994.

[2] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," 2, vol. 29, New York, NY, USA: Association for Computing Machinery, May 2000, pp. 1–12. DOI: 10.1145/335191.335372. [Online]. Available: https://doi.org/10.1145/335191.335372.

[3] R. C. Prim, "Shortest connection networks and some generalizations," *Bell System Technical Journal*, vol. 6, pp. 1389–1401, 6 1957. DOI: 10.1002/j.1538-7305.1957.tb01515.x.

[4] V. Jarník, "O jistém problému minimálním," *Práce Moravské Přírodovědecké Společnosti*, pp. 57–63, 1930.

[5] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, 1959, ISSN: 1. DOI: 10.1007/BF01386390.

[6] J. B. K. Jr., "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proceedings of the American Mathematical Society*, vol. 7, pp. 48–50, 1956. DOI: 10.1090/S0002-9939-1956-0078686-7.

[7] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982. DOI: 10.1109/TIT.1982.1056489.

[8] J. C. Pennsylvania. (2020). "Generating association rules," Junita College Pennsylvania, [Online]. Available: http://faculty.juniata.edu/rhodes/ml/assocRules.html.

[9] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '93, Washington, D.C., USA: Association for Computing Machinery, 1993, pp. 207–216, ISBN: 0897915925. DOI: 10.1145/170035.170072. [Online]. Available: https://doi.org/10.1145/170035.170072.

[10] P.-N. Tan, M. Steinbach, A. Karpatne, and V. Kumar, "Definition 6.2," in *Introduction to Data Mining*, 2nd ed. 2019, ch. 6, p. 334, ISBN: 9780134080284.

[11] C. Zhong, M. Malinen, D. Miao, and P. Fränti, "A fast minimum spanning tree algorithm based on k-means," *Information Sciences*, vol. 295, pp. 1–17, 2015, ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2014.10.012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025514009943.

[12] M. Zekic-Susac and A. Has, "Discovering market basket patterns using hierarchical association rules," *Croatian Operational Research Review*, vol. 6, pp. 475–487, Oct. 2015. DOI: 10.17535/crorr.2015.0036.

[13] M. A. Valle, G. A. Ruz, and R. Morrśs, "Market basket analysis: Complementing association rules with minimum spanning trees," *Expert Systems with Applications*, vol. 97, pp. 146–162, May 2018. DOI: 10.1016/j.eswa.2017.12.028.

[14] T. M. Cover and J. A. Thomas, *Elements of information theory*. 2006.

[15] J. W. Kim, "Construction and evaluation of structured association map for visual exploration of association rules," *Expert Systems with Applications*, vol. 74, 2017. DOI: 10.1016/j.eswa.2017.01.007.

[16] Kaggle. (2020). "Sales data for a chain of brazilian stores," Kaggle, [Online]. Available: https://www.kaggle.com/marcio486/sales-data-for-a-chain-of-brazilian-stores.

[17] P. Manager. (2016). "The ultimate guide to the waterfall model," [Online]. Available: https://www.projectmanager.com/waterfall-methodology.