

Demand Estimation of Full-Cut Promotion on E-Commerce Company

Submitted as part of the honors requirements.

By: Wong Jing Hui

Division of Mathematical Sciences

School of Physical & Mathematical Sciences

Nanyang Technological University

2018/2019

Project Code: MH4900

Project Supervisor:
Assistant Professor Yan Zhen Zhen

Deliverables:
Report: 1 Volume

Date: 29/4/2019

ABSTRACT

Full-cut promotion is a type of promotion where customers spend a minimum threshold amount in a single order to enjoy a fixed discount amount for that order. This type of promotion is popular in China, particularly in E-commerce. However, there is no existing literature that estimates the promotion driven demand and optimizes the threshold and discount amount in the full-cut promotion.

The purpose of this research is to construct a methodology that enables E-commerce companies to measure the performance of full-cut promotion and optimize the promotion for revenue management. The proposed methodology has four main steps: exploratory data analysis, proposed model to measure promotion driven demand, customer segmentation and implementation of a benchmark model and lastly, machine learning and optimization. The benchmark model we have implemented is a modified conditional gradient approach from Jagabathula's paper. The significance of the research is that we have enabled E-commerce companies to measure a customer's attraction towards the full-cut promotion and created an approach to segment customers simply based on the product-level features in an aggregated transactional dataset. Many existing customer segmentation techniques involve customer demographics whereby empirical researchers do not have access to such information. Further, we have constructed a model, using the outputs from demand estimation and customer types, to estimate their utility towards the full-cut promotion and hence, optimize the threshold and discount amount. We have also used various machine learning models to investigate the relationship between features like market price towards the attraction for full-cut promotion. Therefore, E-commerce will not only be able to obtain the optimal full-cut promotion from a given transactional dataset, they will also obtain insights on why customers react towards the different discount & threshold amounts differently as well as how to fine-tune full-cut promotions.

Subject Descriptors:

Machine Learning

Optimization

E-commerce

Keywords:

Full-cut promotion demand estimation,

Clustering and customer type estimation,

Optimization of Full-cut promotions

Implementation Environment:

Windows 10, Python 3.7.0, R 3.4.4, MATLAB 9.5.0

ACKNOWLEDGEMENTS

My work in this honors thesis is only a small part of a greater project in understanding the methods used to answer questions in revenue management of promotions in E-commerce. I have benefited greatly from the weekly discussions with my Final Year Project (FYP) supervisor, Asst Prof Yan Zhen Zhen. I was fortunate to be able to learn more about machine learning and optimization techniques through some of the intellectual discussions at our meetings.

I owe a great many favors to a great number of people who have always been there to encourage and support me throughout the trying year of my FYP in Nanyang Technological University. My deepest thanks to Asst Prof Yan Zhen Zhen, the advisor of the FYP project and a mentor who has guided and supervised me patiently throughout the entire process, introducing me to other research opportunities in the field, encouraged and supported me to promote my ideas and direction. She gave me the huge opportunity working on this project proposed by her and access to the dataset from VIPshop. I could not have been able to continue my research without her persistent trust, even though I am far from the most capable of candidates to accomplish these assignments. Despite the difficulty that I had with the demands of the project, she continued to patiently give me feedback on how to improve and finish the report, particularly in the past two months when my research begun to cover topics in optimization which I do not have prior knowledge in and I became disgruntled with the difficulty to decide which materials to include in the honors thesis. I am especially grateful to her for being so tolerant and supportive.

I also extend my thanks to my family, friends who have been accommodating and supporting me throughout the year. I have only thanked a small group of people and I ask for forgiveness from those I may have excluded accidentally.

CONTENTS

1	INTRODUCTION	5
2	DATA ACQUISITION AND PROCESSING	6
3	EXPLORATORY DATA ANALYSIS	8
4	PROPOSED MODEL FOR PROMOTION DRIVEN DEMAND	13
5	CLUSTERING BASED ON PROMOTION DRIVEN DEMAND	15
	5.1 K-means and K-medoids Clustering.	17
	5.2 Hierarchical Clustering	21
6	CUSTOMER TYPE ESTIMATION.	31
	6.1 Conditional Gradient Setup and Algorithm	31
	6.2 Results and Comparative study.	34
7	MACHINE LEARNING MODELS AND RESULTS	37
	7.1 Decision Regression Tree	38
	7.2 Decision Regression Tree (Bagging)	40
	7.3 Decision Regression Tree (GBM)	43
	7.4 Decision Regression Tree (XGBoost)	45
	7.5 Decision Regression Tree (GBM with H2O package)	46
	7.6 Decision Regression Tree (Light GBM)	48
	7.7 K-Nearest Neighbor	49
	7.8 Feedforward Neural Network (FNN)	50
8	DISCRETE CHOICE MODELS	52
9	OPTIMISATION AND FUTURE WORK	55
10	CONCLUSIONS	57
11	REFERENCES	58
12	APPENDIX	61
	12.1 Multiple Regression Analysis.	61
	12.2 Comparative study of decision tree algorithms.	64
	12.3 Discussion of bagging and boosting methods	65
	12.4 Discussion of Feedforward neural networks.	69
	12.5 Clustering and Silhouette score.	70
	12.6 Customer type estimation and Comparative study.	72

1 INTRODUCTION

Promotion is an important element of the competitive dynamics in E-commerce and it contributes a significant difference to a company's profits. Full-cut promotion is a popular type of promotion used in E-commerce companies such as Taobao and VIPshop [1]. In a full-cut promotion, customers spend a minimum threshold amount in a single order to enjoy a fixed discount to that order [2]. Therefore, full-cut promotions are defined by a pair of discount and threshold amount. However, there is no existing literature that estimates the promotion-driven demand and optimizes the threshold and discount amount in the full-cut promotion. This is due to the nature of full-cut promotions: the fixed discount towards a customer's entire purchase amount may lead to a discontinuous function of the customer purchase amount, which is difficult to optimize since it usually becomes a non-convex optimization problem.

The purpose of this research is to construct a methodology that enables E-commerce companies to measure the customer's attraction to full-cut promotions and optimize the promotion for revenue management. The methodology has four main steps: data exploratory analysis of the transactional dataset, proposed model to measure promotion driven demand, customer segmentation and implementation of a benchmark model, and lastly, machine learning and optimization. The benchmark model we have implemented is a modified conditional gradient approach from Jagabathula's paper [33] and we have shown that our proposed approach is accurate in segmenting customers based on their promotion-driven demand and predicting their choice probabilities. Further, we have discussed how the outputs from demand estimation and customer types can be used to estimate their utility towards the full-cut promotion to optimize the threshold and discount amount. The scope of this research is based on a dataset provided by VIPshop (third largest E-commerce company in China [3]), consisting of 664 orders, 620 different users and 95 categories of products, with information such as the market price, price of the products and total purchase amount.

This research enables E-commerce companies to measure a customer's attraction towards the full-cut promotion and segment customers simply based on the product-level features in an aggregated transactional dataset. Existing techniques for customer segmentation often involve using customer demographics, which is usually sparse with missing or invalid information whereas aggregated transactional dataset is readily available and less sparse. Further, empirical researchers do not have access to customer demographics due to privacy issues. Our methodology also sets up the optimization framework for full-cut promotions and carry out machine learning to investigate the relationship between features like market price towards the attraction for full-cut promotion. Therefore, E-commerce will not only be able to obtain the optimal full-cut promotion from a given transactional dataset, they will also obtain insights on why customers react towards the different discount & threshold amounts differently as well as how to fine-tune full-cut promotions.

2 DATA ACQUISITION AND PROCESSING

The dataset provided by VIPshop consists of 664 orders, 620 different users and 95 categories of products with different features such as market price, total number of products in the order, cut, discount amount as shown in table 2.1. There are other features which are excluded from this dataset as it is irrelevant towards the scope of the research. For example, features such as the product size and discount amount for other types of promotions were excluded from this study. Additionally, features such as number of refunded goods and refunded amount were excluded from this study. In this research, we assumed that the customer's demand towards the full-cut promotion is independent of the presence of other promotions such as free shipping and red packet promotion. This is intuitive because the different types of promotions are stackable, and customers do not have to choose one promotion over the other to enjoy a higher discount amount. Therefore, full-cut promotion driven demand is not affected by the presence of other promotions. All sizes of the same product have the same price and design and therefore assumed to have the same full-cut promotion driven demand. It is also assumed that product refunds do not affect the full-cut promotion driven demand. This is intuitive as product refunds affect the demand for the product but not the customer's demand for the promotion. In this research, it is assumed that there are no out-of-stock situations which can potentially lead to a lower number of orders and possibly underestimating the full-cut promotion-driven demand. It is also assumed that all customers are aware of the information for all promotions at the time of purchase in VIPshop. The following steps were performed to clean and process the dataset:

1. Orders with an invalid order ID are removed.
2. Transactions in each order with 0 good count or invalid discount amount are removed.
3. Orders with transactions that have missing information are replaced with the discount amount and amount paid based on the weighted ratio in the entire order.
4. The minimum price of a good in each order is calculated and added as a new feature.
5. The total amount paid in each order discounted with the minimum price of the good is calculated and added as a new feature.

After processing the dataset, we have the following features: market price, total goods count (total goods cnt), discount amount (cut fav amount), threshold, minimum price (min price), original purchase amount (order amt – min price) are used in this research as shown in table 2.1. All the features in table 2.1 are continuous variables. If the customer makes additional purchases to reach the threshold amount, we assumed that the customers will purchase a single category of good to reach the threshold amount. Further, we assumed that the lowest priced good in the customer's order is the good that the customer purchases to reach this threshold amount. This assumption is intuitive as customers would generally want to spend the least amount of money to subscribe to the promotion, and therefore maximize the utility of the promotion. The features: minimum price and total amount paid discounted with minimum price are important for us to understand the level of attraction of the customer towards the full-cut promotion. This is because these 2 features tell us the additional amount the customer is willing to pay to reach the threshold amount and help us to understand the influence of the original purchase amount on whether the customer will subscribe to the promotion.

Feature Categories	Features	Details	Examples
Quantitative	market_price	Prices of product from competitors such as retail or e-commerce stores	655
	total_goods_cnt	Total number of goods purchased in the order	1,2,8
	cut_fav_amt	Discount amount in the order	0,100,200,300,400,500
	min_price	Price of the good_id which is the lowest amongst the other prices of good_id in an order	¥40
	price-min_price	order price minus the minimum price. order price of good_id that the user originally intends to purchase without full-cut promotion	¥212
Output	Target	Apply logit transformation to price and then normalize to a range between 0 to 1. Target reflects the full-cut promotion driven-demand in each order.	1, 0.7478, 0.6818

Table 2.1 Features in the VIPshop dataset

3 EXPLORATORY DATA ANALYSIS

In an aggregated transactional dataset, there may be multiple promotions occurring across different transactions and we do not have information on which customers or transactions have subscribed to which type of promotions. Therefore, in Exploratory Data Analysis, we aim to construct a methodology to identify the discount and threshold amount for full-cut promotions and understand the distribution of discount amount, threshold amount based on the frequencies of orders. This provides an insight on the customers' preferences to which specific type of full-cut promotion. Further, this methodology enables us to identify all the existing full-cut promotions from a noisy transactional dataset and subsequently measure the customers' attraction towards each type of promotion and apply machine learning techniques to investigate the relationship between promotion driven demand and the features. This is useful in understanding specifically what discount and threshold amounts that the customers respond more favorably towards and therefore, help companies to re-adjust their threshold and discount amounts accordingly. In exploratory data analysis, we seek to understand the characteristics of a customer's order in terms of market price, price and other features described in table 2.1. A summary of the dataset is provided in appendix (table 12.1) to understand the mean, median, minimum and maximum values to provide an insight on each feature's distribution. Due to the limited number of transactions in the dataset, we have simulated the data via sampling to obtain 10,000 orders to train the models in Section 4. The method of sampling to simulate the dataset is discussed later in this section.

We explore the dataset by plotting the histogram of frequencies for the discount amount in each order as shown in figure 3.1. The steps to identify the discount amounts are as follows. Firstly, based on figure 3.1, we can observe that ¥100, ¥200, ¥300 have a spike in frequencies. Secondly, we plot the discount amounts per order based on intervals of ¥0 - ¥100, ¥100 - ¥200, ¥210 - ¥325 as shown in figures 3.2 to 3.4 so that we can observe the spike in frequencies more clearly. Indeed, these figures also show a spike in frequencies for ¥100, ¥200, ¥300. Thirdly, we can observe a trend of a spike in frequency followed by a decreasing trend of frequencies until the next successive spike in frequency. We proposed that this spike in frequency is the discount amount and hence, ¥100 ¥200, ¥300 are proposed as the discount amounts of full-cut promotions in table 3.1. This is intuitive as the variation of discount amounts is caused by the different types of promotions stacking together to discount the customer's total order amount. Since over 95% of the sales are from full-cut promotions, majority of the frequencies for specific discount amount will likely be solely from full-cut promotions.

After identifying the discount amounts, the next step is to identify the threshold amounts. We plot the histogram of frequencies for the threshold amount of all the orders with discount amounts ¥100, ¥200, ¥300 separately as shown in figures 3.5 to 3.7. From figure 3.5, the spike in frequency occurs at ¥199 followed by the decreasing trend of frequencies. This trend is similar to the

observed trend for discount amounts. This is intuitive as customers would want to spend an amount that is near or at the threshold amount to maximize their monetary benefit from this full-cut promotion. Similarly, from figures 3.6 and 3.7, we can identify that the spikes in frequencies occur at ¥398 and ¥597. Since other types of promotions can be stacked with the full-cut promotions, it is possible that orders with discount amounts similar to full-cut promotions have purchase amounts below the threshold amount. Hence, instead of observing the first bin with the non-zero frequency of spent amount, we observe the amount spent in orders that have a highest frequency compared to the rest of the orders with the same discount amounts (¥100, ¥200, ¥300). We obtained the different types of full-cut promotion (defined by the discount amount and threshold amount pairs) in table 3.1.

A data simulation is carried out to obtain a larger dataset of orders to achieve a higher accuracy for the training of machine learning models and for customer segmentation. A total of 10,000 orders are simulated by first carrying out a random sampling with replacement of the total good counts from the existing 664 orders. This first step of random sampling determines the order size for each new simulated order. The second step of random sampling is carried out to determine which goods are selected for each order (from the existing 95 goods). For example, the first random sampling is sampled from a list of “3,3,3,2,2...” is 3, indicating the new simulated order has a size of 3 goods and the second sampling is from a list of good id “532964606, 532964607...”. The respective total amount spent, discount amount and market price are obtained from pivot table of goods and calculated based on their weighted ratios. This method of simulation ensures that new orders are generated based on the list of existing goods and the purchasing behaviors in the original dataset.



Figure 3.1 Histogram plot for full-cut discount amount in each order



Figure 3.2 Histogram plot for full-cut discount amount in each order (¥0-¥100 interval)



Figure 3.3 Histogram plot for full-cut discount amount in each order (¥100-¥200 interval)



Figure 3.4 Histogram plot for full-cut discount amount in each order (¥210-¥325 interval)



Figure 3.5 Histogram plot for amount spent per order (for ¥100 discount amount)



Figure 3.6 Histogram plot for amount spent per order (for ¥200 discount amount)



Figure 3.7 Histogram plot for amount spent per order (for ¥300 discount amount)

Full-Cut Discount Amount	Threshold Amount
¥100	¥199
¥200	¥398
¥300	¥597

Table 3.1 Proposed Full-cut discount amount and threshold amount

4 PROPOSED MODEL FOR PROMOTION DRIVEN DEMAND

In section 3, we have introduced an approach to identify the types of full-cut promotion from a noisy aggregated transactional dataset. In this section, we would like to propose a method to measure a customer's attraction towards each type of full-cut promotion. Hence, we propose that the customer's attraction towards the full-cut promotion is the promotion driven demand, which is defined as the probability the customer will purchase additional product(s) to reach the minimum threshold amount to enjoy the full-cut promotion.

The output (Target) is a continuous value between 0 to 1 and reflects the full-cut promotion driven demand in each order. This value is obtained by finding the maximum value between 0 and the output of a modified logit transformation on the price of each order (before full-cut promotion) and then normalizing the resulting value into a range between 0 to 1. The number of goods purchased (unique or non-unique) in each order is denoted as T . Specifically, for each order $i = 1, 2, 3 \dots \mathcal{N}$ and $Threshold_j = ¥199, ¥398, ¥597$ for $j = 1, 2, 3$

$$y_i = \begin{cases} \max\{0, \log(\frac{Threshold_j/Price_i}{1 - Threshold_j/Price_i})\}, & Price_i \geq Threshold_j, T \geq 2 \\ 0, & Price_i < Threshold_j \text{ or } T < 2 \end{cases}$$

$$Target_i = \begin{cases} 1, & Price_i = Threshold_j, T \geq 2 \\ \frac{y_i - \min(y)}{\max(y) - \min(y)}, & Price_i > Threshold_j, T \geq 2 \\ 0, & Price_i < Threshold_j \text{ or } T < 2 \end{cases}$$

The logit transformation is represented as $\log(\frac{p(X)}{1-p(X)})$. In our modified logit transformation, $Threshold_j/Price_i$ represents $p(X)$, which is the probability of a customer being attracted to the full-cut promotion in the case of $Price_i \geq Threshold_j$. This is intuitive as we have assumed that all customers are aware of the full cut promotions and a customer who spends an order amount near to the threshold amount is likely to indicate that the customer is attracted to this full-cut promotion. Hence the closer the amount, $Price_i$, to the threshold amount, $Threshold_j$, the higher the probability of the customer being attracted to the full-cut promotion, given that $Price_i \geq Threshold_j$. When the $Price_i \gg Threshold_j$, $\log(\frac{Threshold_j/Price_i}{1 - Threshold_j/Price_i}) < 0$, we take the value of 0 by applying the maximum function. This is intuitive since when the amount spend exceeds the threshold amount by a large margin, it is likely that the customer is no longer interested in the promotion since the discount amount is only a small fraction to the entire purchase amount and the promotion driven demand is assigned as 0. When $Price_i < Threshold_j$, the customer is not subscribed to the full-cut promotion and hence the promotion driven demand is assigned as 0.

The above formula for normalization scales the promotion-driven demand, $Target_i$, to a range between 0 to 1, which enables a comparison to a consistent range of values of demand to different orders. We use a logit transformation as it is a conventional econometric model used for modelling consumer choice probabilities.

When $\mathcal{T} \geq 2$ and price that is near to the threshold amount, this indicates that the customer is strongly influenced by the full-cut promotion. When price is lower than the promotion threshold, this indicates that the customer is aware of the promotion but chooses not to purchase additional goods to enjoy the full-cut promotion. Hence, the full cut promotion-driven demand is 0. When $\mathcal{T} \geq 2$ and the purchase amount in the order is more than the promotion threshold, this indicates that the customer is less influenced by the full-cut promotion since the discount amount contributes a smaller proportion to the purchase amount. If the customer is strongly influenced by the promotion, the customer will want to maximize the utility of promotion by spending an amount closer to the threshold amount. In this instance, by spending additional amount to reach the next successive threshold amount. The promotion driven demand is defined as the willingness of a customer to purchase due to the promotion. Therefore, we assumed that customers who purchase only a single product in an order are not influenced by the full-cut promotion.

5 CLUSTERING BASED ON PROMOTION DRIVEN DEMAND

In the previous section, we proposed a model to measure a customer's attraction to the full-cut promotion via promotion driven demand. In this section, we will segment and understand different types of customers with varying attractions to the full-cut promotion. Additionally, the segmentation based solely on promotion driven demand enables us to create different customer types using only product-level features and does not require customer demographics such as salary, age. The significance of segmenting customers in this approach is that companies can understand which types of customer are more attracted to full-cut promotions and hence, allocate their resources more effectively to have targeted marketing towards these customer segments to optimize their revenue from promotions.

Clustering is defined as the partitioning of data objects into groups in a way that objects from the same group have similar features, while objects from different groups have different features, displaying the heterogeneity of objects in the data. There are supervised and unsupervised clustering algorithms. In supervised clustering algorithms, the clusters (outputs) are already known before the training of the model whereas in unsupervised clustering algorithms, the clusters (outputs) are unknown and the algorithm goes through a learning process to find the appropriate number of clusters based on the features. In this section, we are using unsupervised clustering algorithms to segment customers based on their full-cut promotion demand, specifically, K-means, K-medoids and Hierarchical clustering methods.

Popular unsupervised clustering algorithms are: K-means (and the variation K-medoids) and Hierarchical clustering [28]. Hierarchical clustering outputs a tree like structure (dendrogram) enabling the visualization of taxonomic relationships between objects and clusters [29]. Specifically, the two most similar clusters are combined and continue to combine until all objects are in the same cluster [29]. In K-means clustering, the data objects are subdivided into K clusters (pre-specified) and each cluster is represented by the means of the data points belonging to the cluster [28]. Both hierarchical clustering and K-means method are sensitive to outliers. For the full details on hierarchical clustering, refer to the following paper in [29] and K-means clustering in [28]. As quoted from [31], "clustering is in the eye of the beholder.", that means the evaluation of clustering algorithms are highly subjective and is based on two main factors: interpretability of clusters and clustering structure. Due to the subjectivity of interpretability of clusters, we implemented a variety of popular clustering algorithms to compare the silhouette score as well as the cluster intuitiveness and interpretability in each algorithm.

The first step involves cleaning and visualizing the datapoints to understand the properties such as the distribution and sparsity of the data. The output based on our proposed model in section 4 is used to train the following clustering algorithms: K-means, K-medians, K-medoids and Hierarchical clustering. Figure 5.1 shows the data distribution on the full-cut promotion driven demand. From figure 5.1, we can observe that the histogram is left-skewed with majority of the

customers having 0 to 0.1 full-cut promotion demand. Table 5.1 shows a summary of clustering models used, with the optimal number of clusters obtained and the mean silhouette score. The mean silhouette approach measures the quality of a clustering and determines how well each observation lies within its cluster. A value of mean silhouette score between 0.7 to 1.0 indicates a strong clustering structure has been observed [36]. A value of mean silhouette score between 0.51 to 0.70 indicates a reasonable clustering structure has been observed [36]. A value of mean silhouette score of less than 0.50 indicates that the clustering structure is weak and could be artificial [36]. Additional details on the definition of silhouette score will be discussed in Appendix section 12.5. From table 5.1, K-means and Single Linkage (Hierarchical clustering) have the highest silhouette score of 0.64 and 0.66 respectively. We have chosen K-means due to the high silhouette score and interpretability of clusters. For Single Linkage, there is a poor interpretability of cluster as cluster 2 and 3 only contains 5 observations.

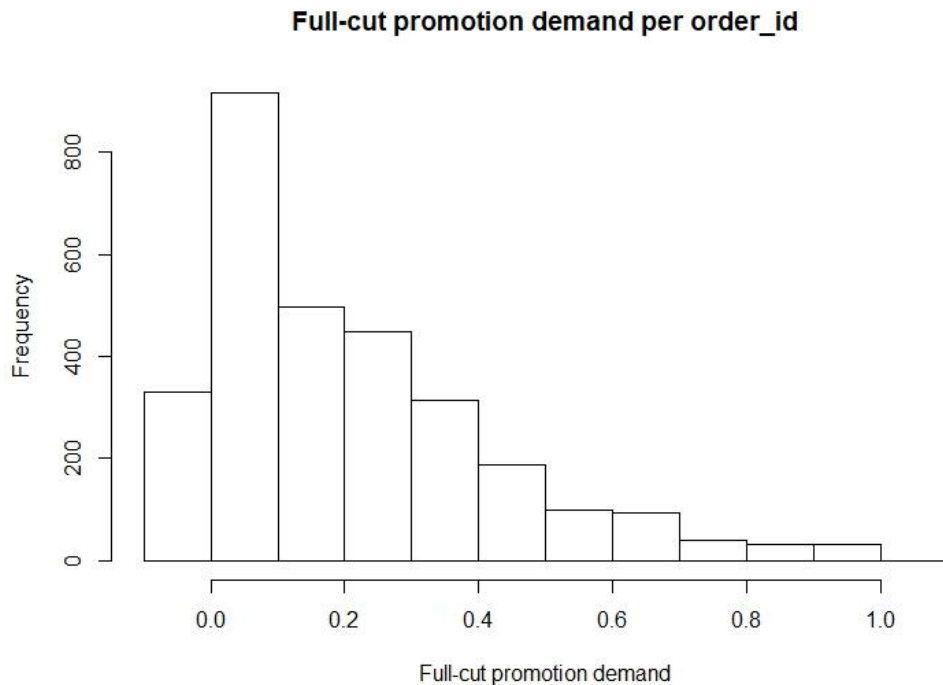


Figure 5.1 Histogram plot for full-cut driven demand per order

	Optimal number of clusters	Mean Silhouette score
K-means	3	0.6367
K-medoids	3	0.6209
Hierarchical clustering (Ward Linkage)	3	0.5842
Hierarchical clustering (Ward Linkage)	4	0.6201
Hierarchical clustering (Single Linkage)	3	0.6557
Hierarchical clustering (Complete Linkage)	4	0.6011
Hierarchical clustering (Average Linkage)	3	0.6139
Hierarchical clustering (Average Linkage)	4	0.6018

Table 5.1 Summary of clustering models used

5.1 K-means and K-medoids clustering

K-means clustering involves partitioning the data objects into pre-specified K by minimizing the squared distance of each point to the point belonging to the center of a cluster [28]. Instead of minimizing the squared distance. K-medoid involves minimizing the absolute distance between the points and the selected centroid. Hence, K-medoid may be more robust to noise and outliers than K-means, when the points belonging to a cluster are connected but are less compact and contain outliers [32]. Since the number of clusters need to be pre-specified, the elbow method is used to determine the optimal number of clusters based on the percentage of variance, known as, within clusters sum-of-square (WSS). The optimal number of clusters is the cluster number whereby the successive cluster number has the most significant decrease in the amount of WSS reduced. Figure 5.1.1 and 5.1.2 are the elbow method plot for K-means and K-medoids. In both

figures, the optimal number of clusters is 3 since from 3 to 4 clusters, since there is a significant decrease in WSS reduced. Figure 5.1.3 shows a histogram plot of each of the 3 clusters in K-means and K-medoids, with both clustering algorithms having similar results. Each of the corresponding clusters shows a similar distribution, with cluster 1 showing a zero or low full-cut promotion demand, cluster 2 showing a lower full-cut promotion demand, between the range of 0.15 to 0.5 and cluster 3 showing a higher full-cut promotion demand, between 0.5 to 1. Additionally, the figures for the silhouette scores and plots for K-means and K-medoids are provided in figures 5.1.4 to 5.1.5. The histogram plots indicate that customers from cluster 1 are not interested in the full-cut promotion and customers from cluster 2 have some interest in full-cut promotions. E-commerce companies need to target these customers to understand the factors that drive their attraction as well as the factors that negatively influence their attraction. Hence, feedback should be obtained from these customers to understand and tailor the full-cut promotions better for this segment. The customers from cluster 3 are highly interested in full-cut promotion. By identifying the characteristics and feedback from each customer segments, E-commerce companies can understand how to alter the full-cut promotions to attract customers from cluster 2 and more importantly, optimize the revenue management of promotions from customers in cluster 2 and 3.

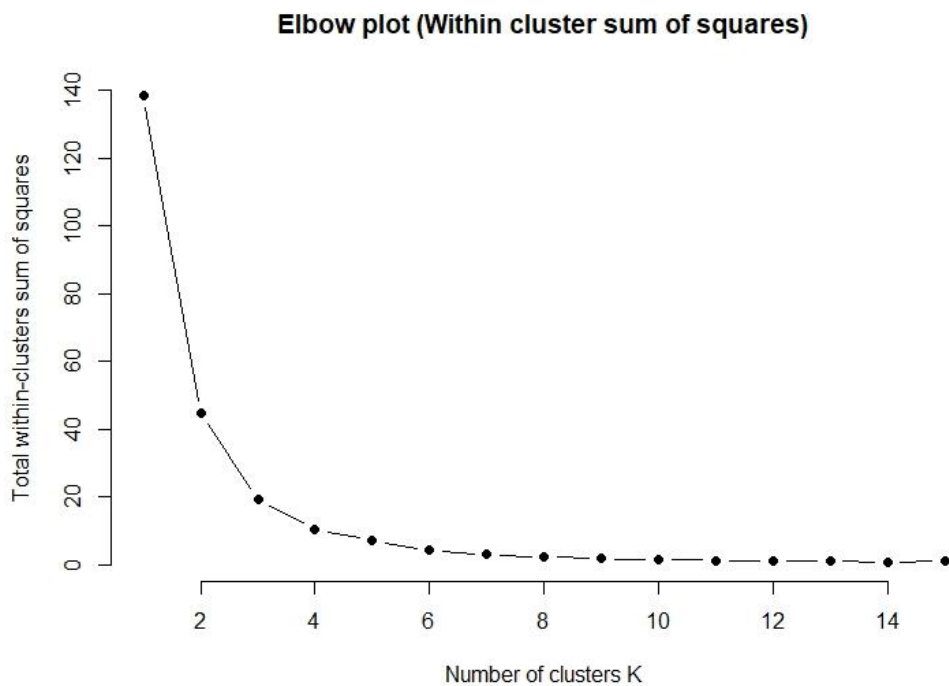


Figure 5.1.1 Elbow plot for determining number of clusters for K-means

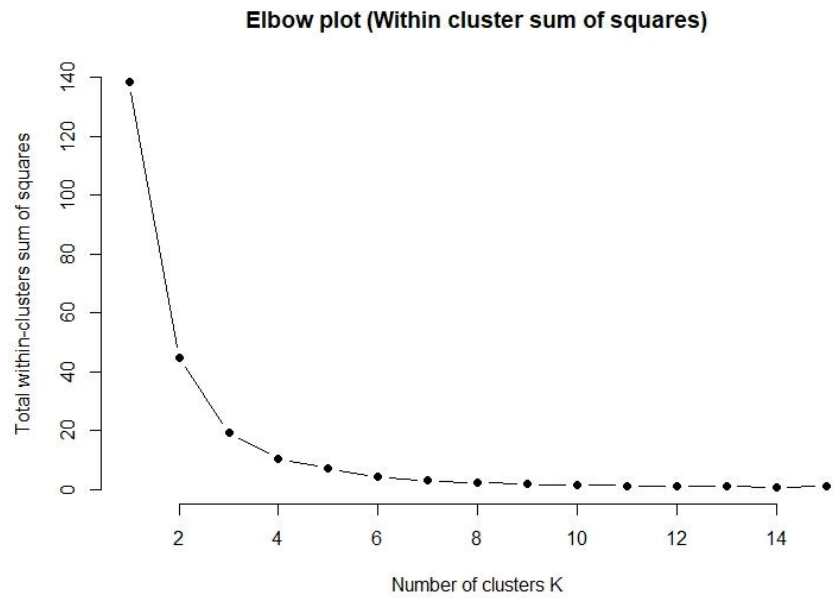


Figure 5.1.2 Elbow plot for determining number of clusters for K-medoids

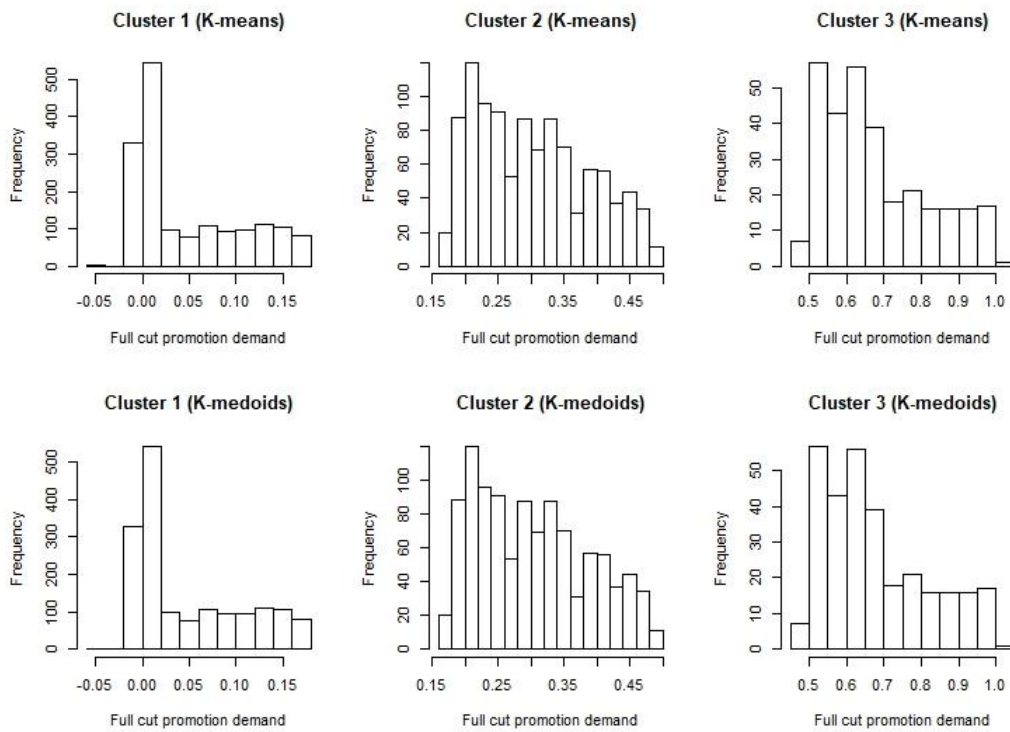


Figure 5.1.3 Histogram plot of clusters for K-medoids and K-means

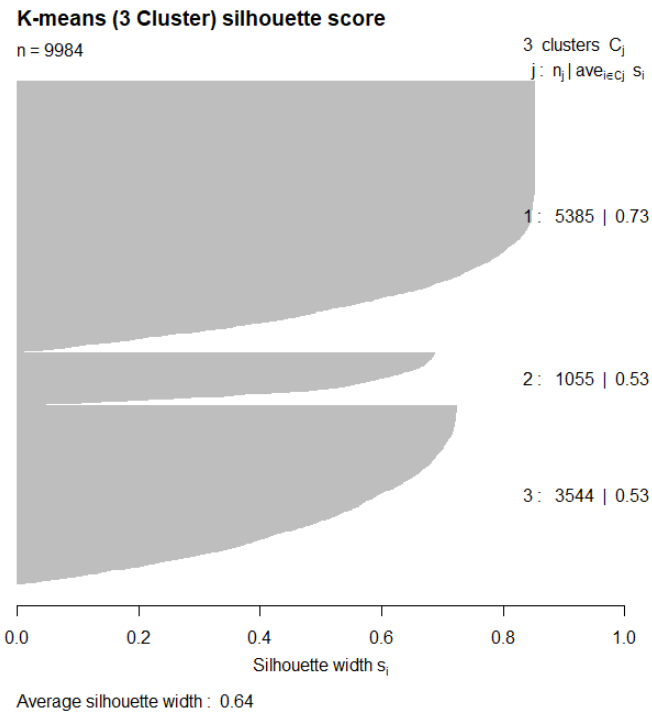


Figure 5.1.4 Silhouette plot for K-means (3 clusters)

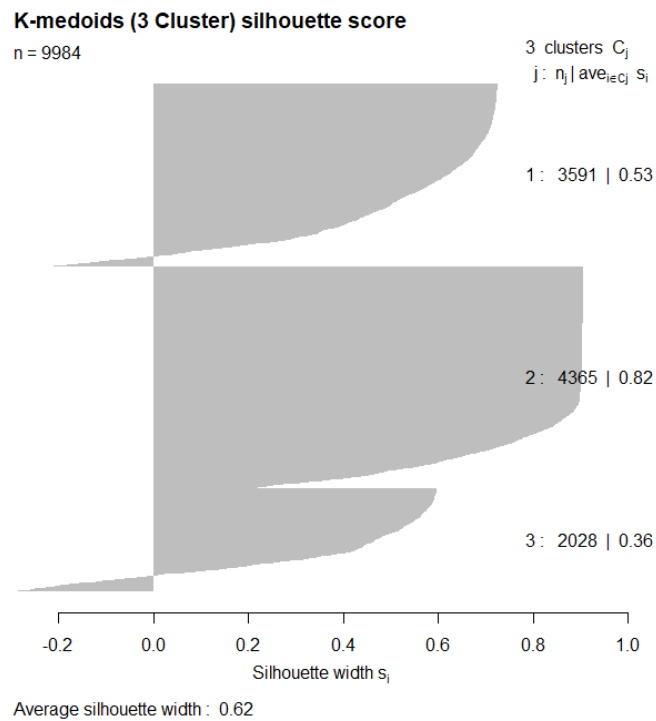


Figure 5.1.5 Silhouette plot for K-medoids (3 clusters)

5.2 Hierarchical Clustering

Hierarchical clustering involves creating a dendrogram to visualize the taxonomic relationships between objects and clusters [29]. This is achieved by the first assigning each object into its own clusters and then repeatedly combining the two most similar clusters until all objects are in the same cluster and only one remaining cluster. This produces a tree (dendrogram) with level indicating the possible number of clusters, and therefore, the number of clusters do not need to be pre-specified unlike K-means clustering [29]. Similarity between clusters are calculated based on the distance between clusters and there are a few measures to do so: Single Linkage, Complete Linkage, Average Linkage and Ward Linkage [29]. In Single Linkage, the distance between one cluster and another cluster is equal to the shortest distance from any data point of one cluster to any data point in another [29]. The similarity of the pair of clusters is based on the closest pair of data points and hence, highly affected by outliers. In Complete Linkage, the distance between one cluster and another cluster is equal to the longest distance from any point of a cluster to any point in the other cluster [29]. This tends to produce more compact clusters and is also highly affected by outliers. In Average Linkage method, the distance between one cluster and another cluster is equal to the average distance between all possible pairs of points in each cluster [29]. Ward's Linkage aims to minimize the total within-cluster variance, whereby the union of every possible cluster pair at each step is considered based on the minimum increase in error sum-of-squares (ESS) [29].

Figures 5.2.1, 5.2.2, 5.2.3 ,5.2.4 show the dendrograms for Hierarchical Clustering based on the following respective similarity measures: Ward's Linkage, Single Linkage, Complete Linkage and Average Linkage. The optimal number of clusters is obtained by cutting across the dendrogram at a specific depth. This depth is chosen is based on the interpretability of clusters as well as the clustering structure using mean silhouette score. The optimal number of clusters are the number of branches that passes through the red line as shown in the figures below. Therefore, the dendrogram based on Ward's Linkage shows that the optimal number of clusters is either 3 or 4. The dendrogram based on Single Linkage shows that the optimal number of clusters is 3. The dendrogram based on Complete Linkage shows that the optimal number of clusters is 4. The dendrogram based on Average Linkage shows that the optimal number of clusters is 3 or 4.

From the results of both the hierarchical clustering and K-means , K-medoids clustering, the most common optimal number of clusters is 3 and is also the most interpretable amongst the other results with a second highest mean silhouette score of 0.6367. Even though hierarchical clustering via Single Linkage has the highest silhouette score of 0.6557, we have not chosen this clustering as the clusters 2 and 3 only contains 4 and 1 observations respectively. Therefore, these clusters are not interpretable and are not meaningful. Therefore, in this research, we decided to segment the customers into 3 clusters based on the result from K-means clustering. The first cluster contains customers which has a predicted full-cut promotion driven demand, y_i ,

where $0 \leq y_i < 0.15$ and the second cluster where $0.15 \leq y_i < 0.5$ and the third cluster where $0.5 \leq y_i < 1.0$. Intuitively the first cluster (majority of the customers) represents customers who are not interested in the full-cut promotion, while the second cluster represents customers who have shown some interest in the full-cut promotion and the third cluster represents customers who are very interested in the full-cut promotion. Suppose that $y_i \geq 0.5$ represents customers who has subscribed to the full-cut promotion, then this result shows companies should spend more resources to attract customers in second clusters towards this promotion. The first step is to gather feedback from customers to understand their reason being interested or not interested in the promotions and then alter the promotion accordingly, such as the discount amount, time period or the threshold amount.

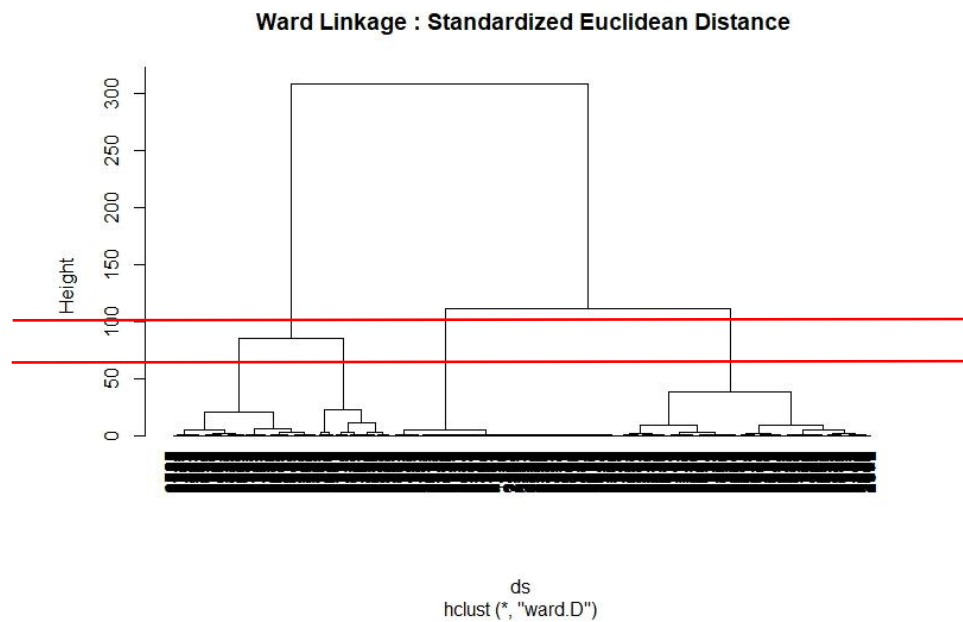


Figure 5.2.1 Dendrogram of Hierarchical Clustering based on Ward's Linkage

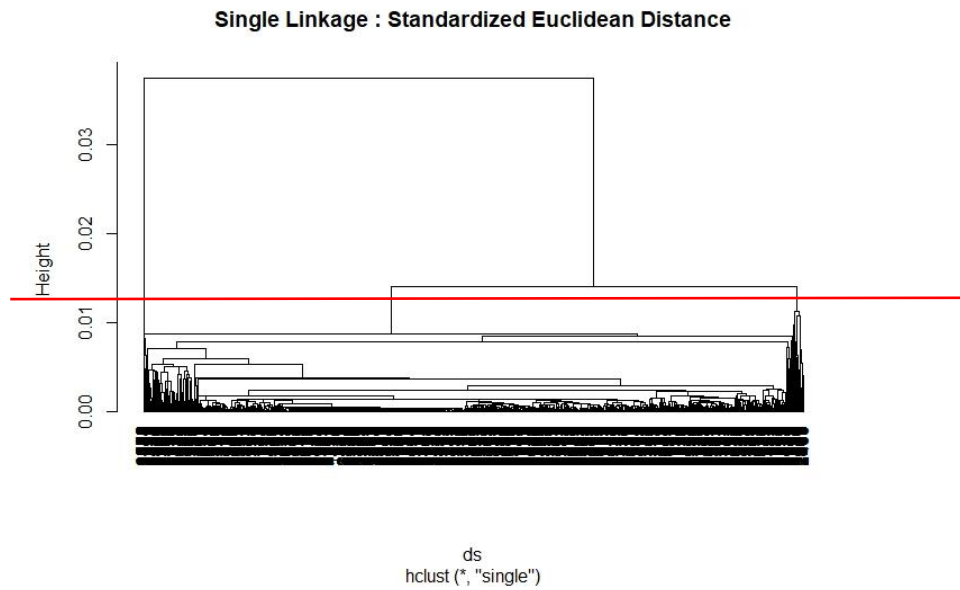


Figure 5.2.2 Dendrogram of Hierarchical Clustering based on Single Linkage

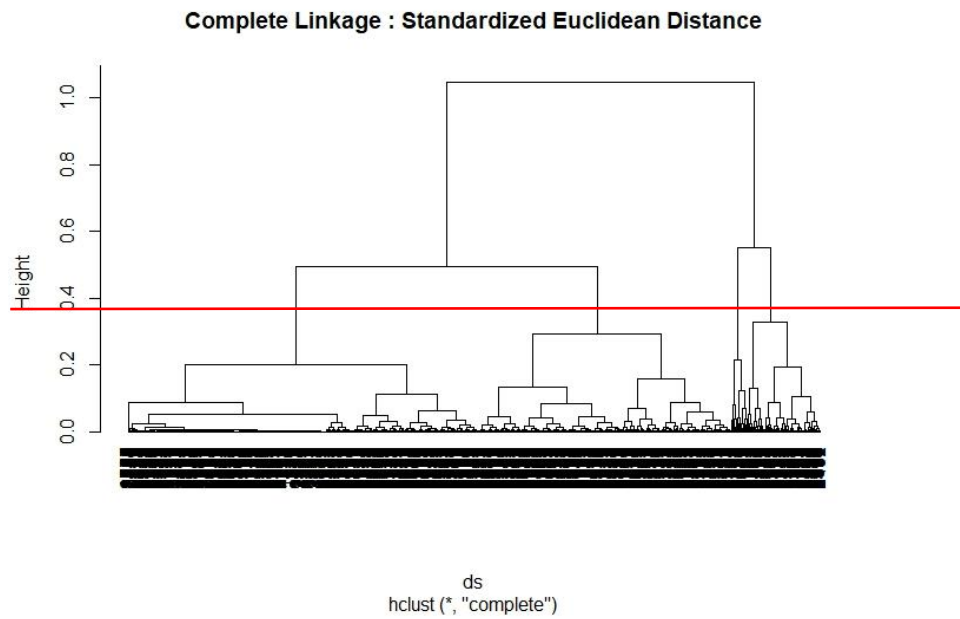


Figure 5.2.3 Dendrogram of Hierarchical Clustering based on Complete Linkage

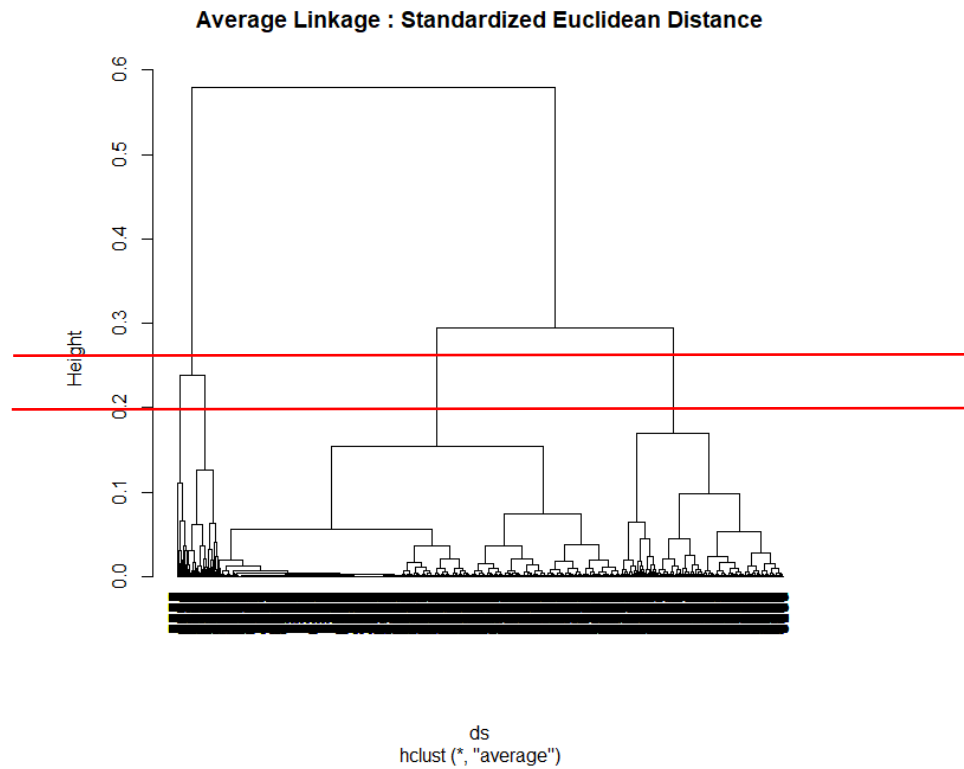


Figure 5.2.4 Dendrogram of Hierarchical Clustering based on Average Linkage

Figure 5.2.5 to 5.2.16 shows the histogram plot for the frequencies based on full-cut promotion demand in each cluster as well as the silhouette scores for each cluster in the respective clustering models.

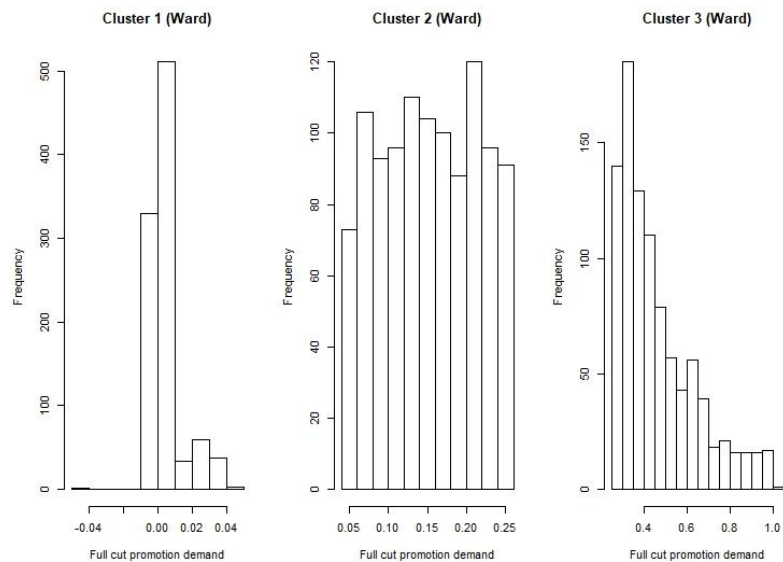


Figure 5.2.5 Histogram plot of clusters for Ward's Linkage (3 clusters)

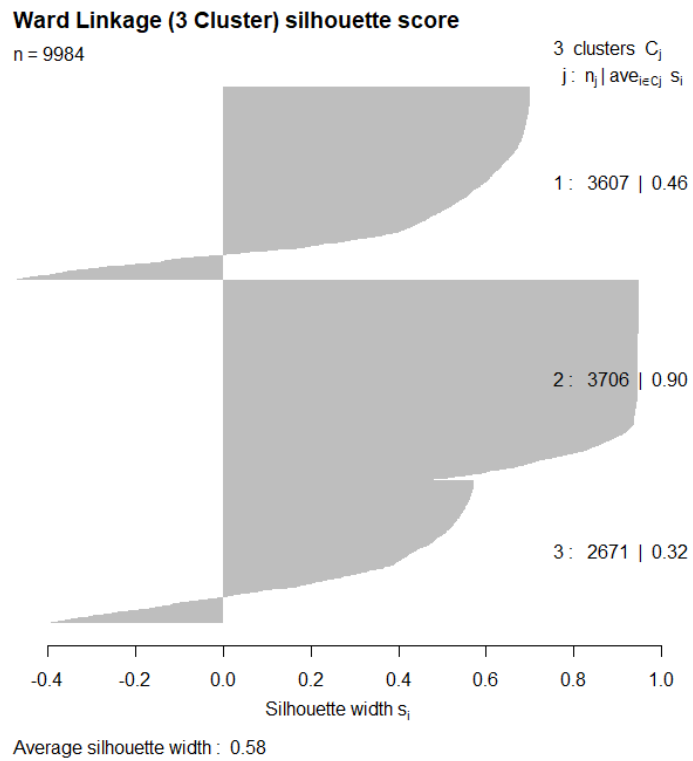


Figure 5.2.6 Silhouette plot for Ward's Linkage (3 clusters)

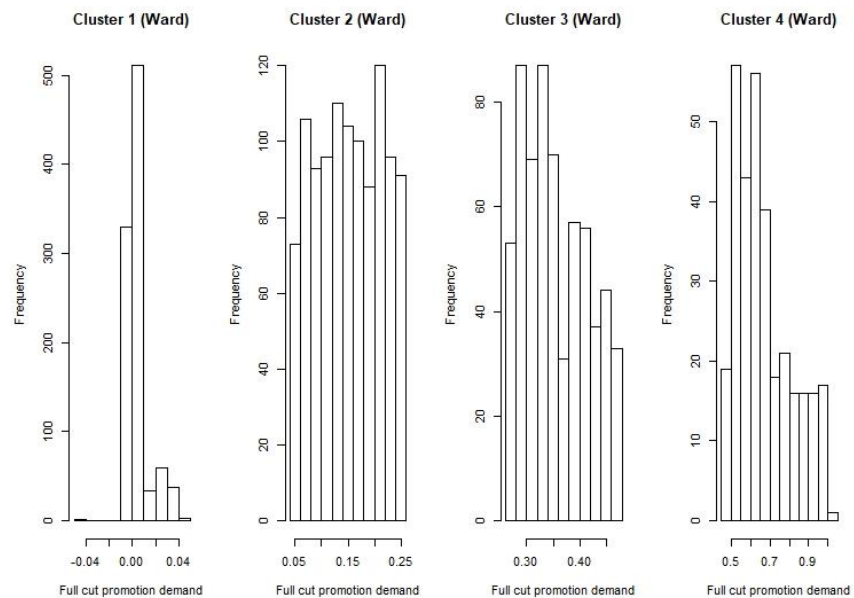


Figure 5.2.7 Histogram plot of clusters for Ward's Linkage (4 clusters)

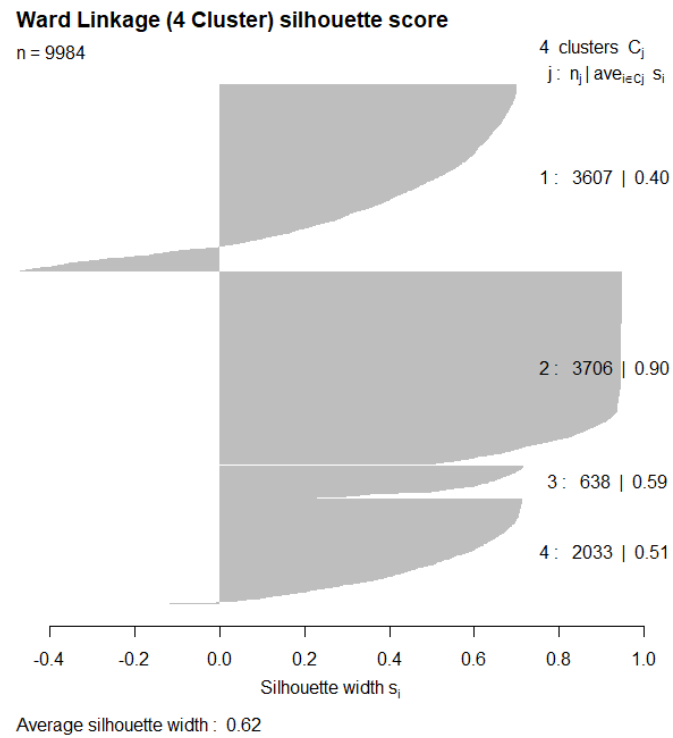


Figure 5.2.8 Silhouette plot for Ward's Linkage (4 clusters)

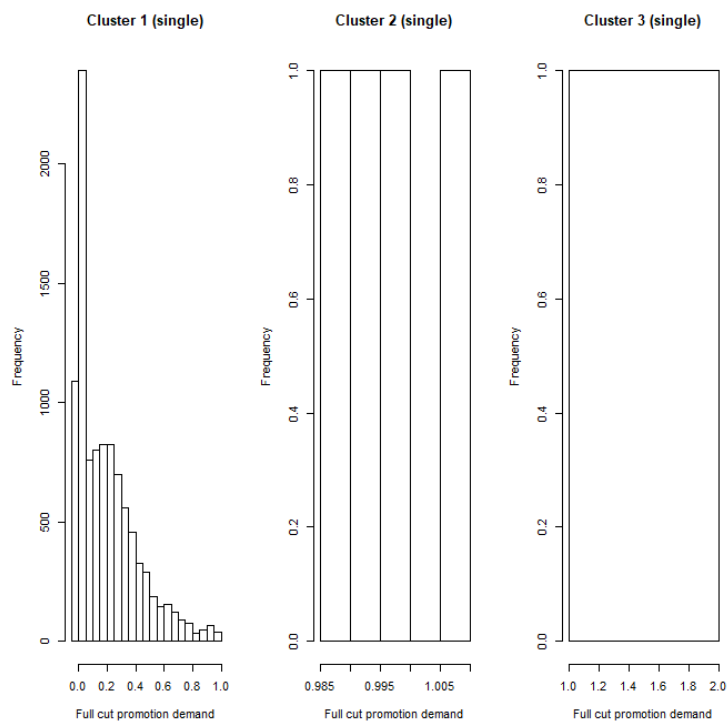


Figure 5.2.9 Histogram plot of clusters for Single Linkage (3 clusters)

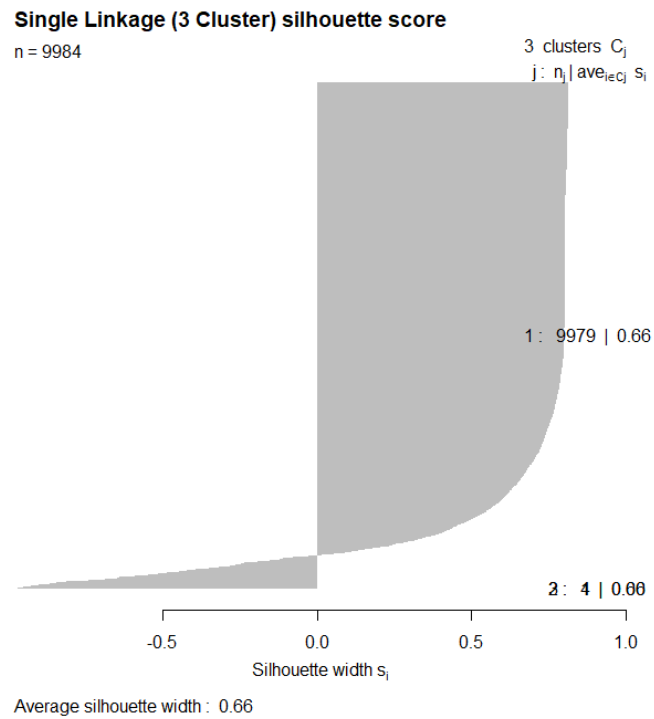


Figure 5.2.10 Silhouette plot for Single Linkage (3 clusters)

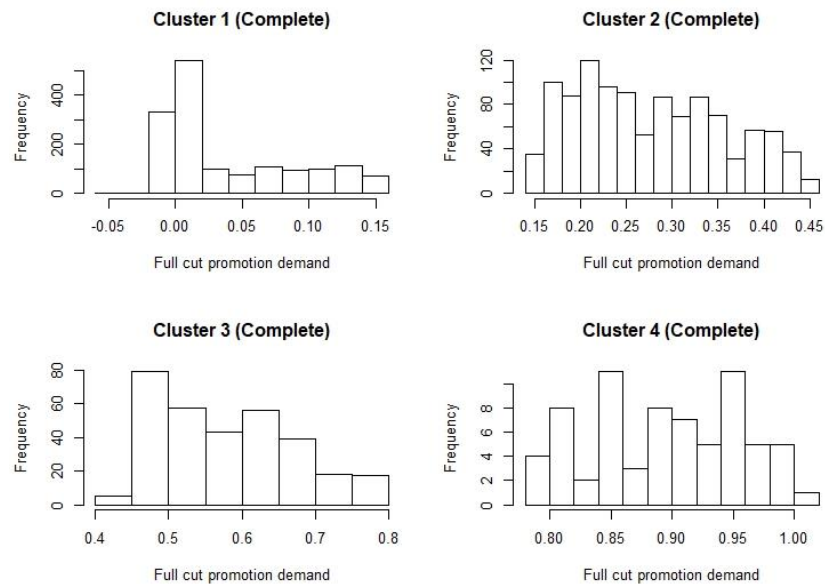


Figure 5.2.11 Histogram plot of clusters for Complete Linkage (4 clusters)

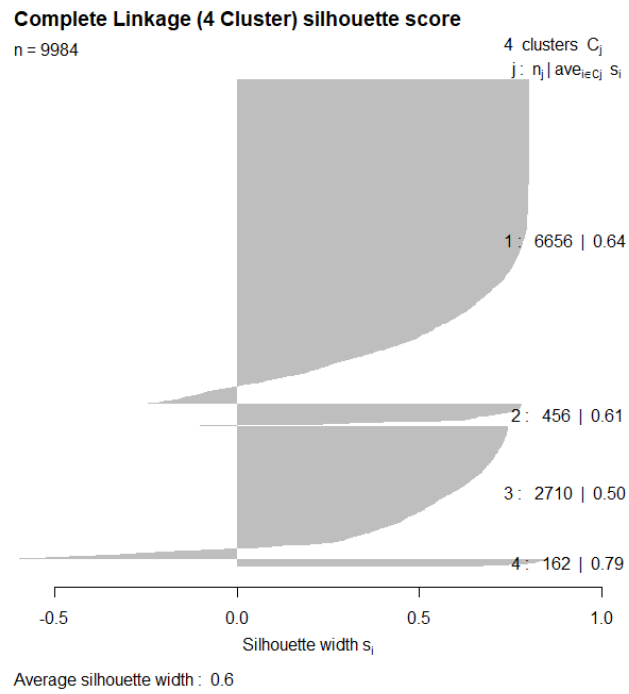


Figure 5.2.12 Silhouette plot for Complete Linkage (4 clusters)

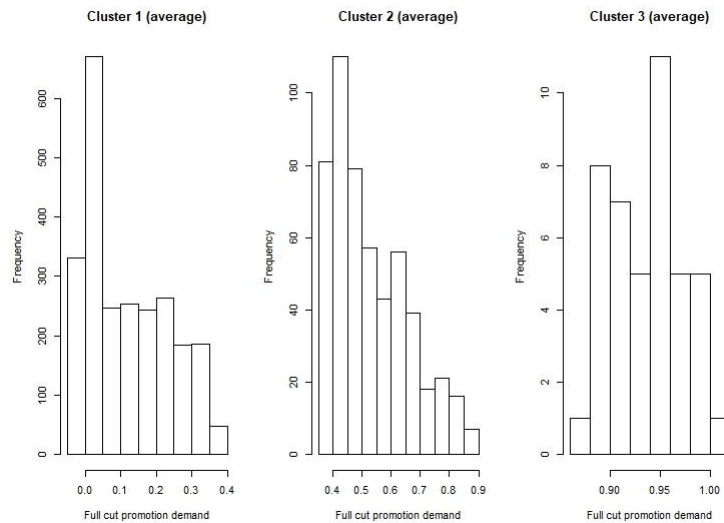


Figure 5.2.13 Histogram plot of clusters for Average Linkage (3 clusters)

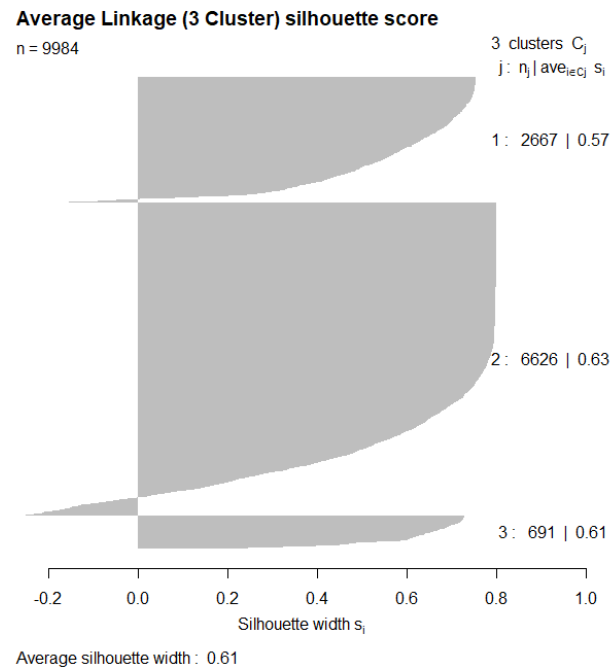


Figure 5.2.14 Silhouette plot for Average's Linkage (3 clusters)

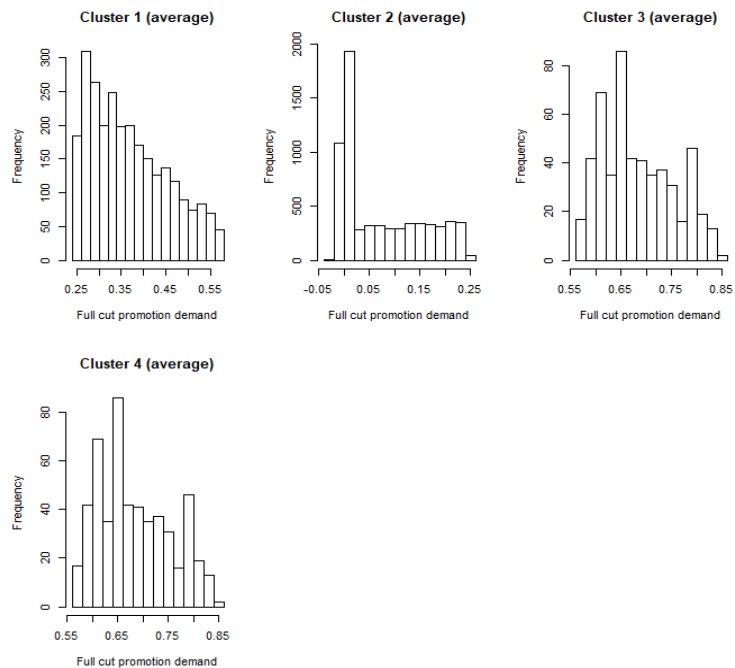


Figure 5.2.15 Histogram plot of clusters for Average Linkage (4 clusters)

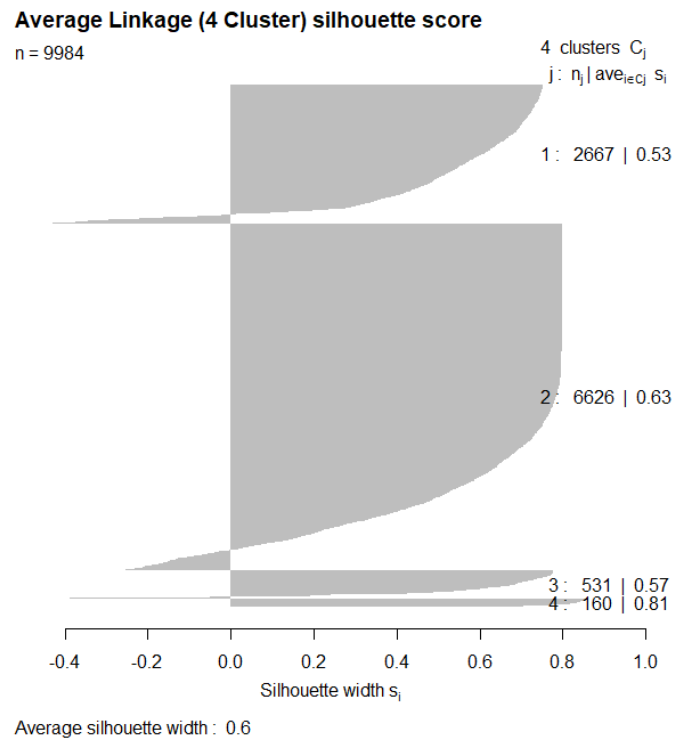


Figure 5.2.16 Silhouette plot for Average's Linkage (4 clusters)

6 CUSTOMER TYPE ESTIMATION

In section 4, we proposed a model to measure the customer's attraction to the full-cut promotion via promotion driven demand. In section 5, using this proposed model, we segment and identified 3 customer types based on their attraction. In this section, we carry out a comparative study of our proposed approach with other existing techniques by comparing the customer types and their respective demand via choice probabilities. We seek to show that our proposed approach is viable and the benchmark model we have implemented is Conditional Gradient approach. This comparative study enables us to understand how well our proposed approach performs against existing models. There are two main existing techniques that identify customer types using an aggregated sales dataset: Conditional Gradient Approach from Jagabathula's paper [33] and BLP model from Berry, Levinsohn, Pakes [39]. Both techniques are similar in that they estimate a customer's demand via their choice probabilities and identify customer types by positing a distribution of customer preferences over the products. However, the models used are different in that BLP model considers a joint distribution of the consumer and product-level features whereas Conditional Gradient approach considers a mixture distribution on product-level features. Further, the BLP model makes certain consumer price-taking assumptions as well as considering Nash equilibrium into its prediction of equilibrium product prices and quantities. In this research, we have implemented Conditional Gradient approach as it is a technique that is designed recently.

We will first discuss the implementation of Conditional Gradient in subsection 6.1 and then the results and comparative study in subsection 6.2. To identify the different customer types and the corresponding customer choice probabilities, we need to first identify the distribution of the orders across all goods based on the features. Such a distribution is usually assumed to be a mixture of logit models. A common approach for modelling the mixing distribution is to approximate the set of all mixing distributions with another large but finite class of distributions and search for the optimal (best fitting) mixing distribution via minimizing a loss function [33]. However, this may lead to errors from model misspecifications and the resulting optimizing problem will be non-convex and this is computationally difficult to solve [33]. In this research, we use the conditional gradient approach, to reformulate the estimation of the mixture distribution as a constrained convex program, without resorting to approximating the space of all mixing distributions to a finite large space [33]. We follow closely to Jagabathula's work on conditional gradient [33] by attempting to implement the model onto the E-commerce setting with a few modifications.

6.1 Conditional Gradient Setup and Algorithm

We consider a universe of $[n] = \{1, 2, 3, \dots, n\}$, $n = 95$ goods. We denote N_j as the number of times good j has been purchased across all the orders in the dataset. Hence, $N = \sum_{j \in [n]} N_j$ represents the total number of purchases for all goods in the dataset. Unlike in Jagabathula's paper which considers time intervals, in this research we do not consider any time interval. This is because at every given time interval, we assume that the product assortment does not change and customers

can purchase from the entire universe of goods, whereas in Jagabathula's paper, customers are exposed to different sets of goods, with new goods introduced and old goods being removed. **Data** = $\{(N_j: j \in [n])\}$ Additionally, for every N_j , there is a $D = 4$ -dimensional feature vector z_j representing the following features: market price, discount amounts, count of goods, goods amount. The goal is to estimate the best fitting mixing distribution based on **Data** from the space of all possible mixing distributions Q , by minimizing the loss function, which measures the mismatch between proportion of sales in **Data** and the predicted output from mixing distributions. We assumed that the customers make the choices of purchasing the good j according to Multinomial Logit model (MNL) as shown in $f_j(w)$:

$$f_j(w) = \frac{\exp(w' \cdot z_j)}{\sum \exp(w' \cdot z_l)}$$

$$g_j(Q) = \int f_j(w) dQ(w)$$

where $f_j(w)$ denotes the probability that product j is purchased and w denotes the 4-dimensional preference vector for the features. $Q = \{Q: Q \text{ is a distribution over } \mathbb{R}^4\}$ and Q is the set of all possible mixing distributions. The loss function is a Squared Loss (SQ) function which is defined as follows:

$$\min_Q \text{loss}(g(Q), \mathbf{Data})$$

$$\text{loss}(g(Q), \mathbf{Data}) = \text{SQ}(g(Q), \mathbf{Data}) = \frac{1}{2} \sum_{j \in [n]} (g_j(Q) - b_j)^2$$

where $b_j = N_j/N$ and $g_j(Q)$ represents the customer's choice probability on good j

In Jagabathula's paper, instead of optimizing through the entire set of all possible mixture distributions, we can simply optimize through the convex hull of choice probability vectors:

$$\min_Q \text{loss}(g(Q), \mathbf{Data}) \equiv \min_{g \in \text{conv}(\bar{P})} \text{loss}(g, \mathbf{Data})$$

$$P = \{f(w): w \in \mathbb{R}^D\} \quad \text{where } f(w) = (f_j(w): j \in [n])$$

$$\text{conv}(\bar{P}) = \left\{ \sum_{f \in F} \alpha_f \mathbf{f} : F \subset \bar{P} \text{ is finite and } \sum_{f \in F} \alpha_f = 1, \alpha_f \geq 0, \text{ for all } \mathbf{f} \in F \right\}$$

$\mathbf{g}(Q)$ represents the mixture likelihood vectors, which are choice probabilities across all goods under the mixture distribution Q . The equations from [33] (as shown above) show that the objective function (minimum of loss function) only depends on the set of mixture distributions Q through $\mathbf{g}(Q)$. Further, this set of mixture likelihood vectors is convex and is analogous to $\mathbf{f}(\mathbf{w}) = (f_j(\mathbf{w}) : j \in [n])$. This means that $\text{conv}(\bar{P})$ is the convex hull of the closure of the set of all choice probabilities across all goods and weights \mathbf{w} . Convex hull of \bar{P} is defined as the smallest convex set containing \bar{P} . This is crucial as the convex sum of any number of elements in the set \bar{P} must still be a valid choice probability, that is, it must still be an element of the set \bar{P} . \bar{P} consists of a set of 100 customer choice probabilities and represents the set of choice probabilities in the entire dataset. \bar{P} is built by sampling with replacement multiple transactions over the entire dataset and aggregating the transactions to form a customer's choice probability. In this research, we assumed the homogeneity of the customers in their product needs and price sensitivities. This is because the transactional dataset only involves product-level features and lacks the customer demographics such as salary, and this makes determining the price sensitivities for each product difficult. This assumption is also intuitive as our scope of research involves understanding the customer's attraction towards full-cut promotion and this is based on whether the customer purchases additional goods to reach the threshold amount. This means that the differences in product needs do not matter significantly as we are only concern about the dollar amount and count of goods that the customer purchases to reach this threshold amount, instead of specifically which type of goods he or she purchases as well as the price for each good. For example, the promotion driven demand measure for a customer who purchased 2 products (that are priced ¥50 each) will be the same as customer who purchased 4 products (that are priced ¥25 each).

The steps in the conditional gradient algorithm is as shown in Algorithm 1 below. We begin by using an initial choice probability for goods across all transactions, denoted as $\mathbf{g}^{(0)}$, which is obtained from simulating a uniform distribution of values from 0 to 1 where the sum of the choice probabilities in each transaction is equals to 1. We also initialize the number of iterations, $k = 1$, and $\alpha^{(0)}$ as a unit vector. α^0 denotes the corresponding likelihood for each of the customer's choice probability $\mathbf{g}^{(0)}$. Additionally, the subscript denotes the iteration k . For each iteration k , we seek to obtain the customer choice probability, $\mathbf{f}^{(k)}$ in the support finding step, the corresponding likelihood $\alpha^{(k)}$ in the proportions update step, and update the $\mathbf{g}^{(k)}$ as inputs for the support finding step in the next iteration. In the support finding step, the conditional gradient algorithm generates a sequence of solutions iteratively $\mathbf{f}^{(1)}, \mathbf{f}^{(2)}, \dots$, that converges to $\mathbf{f}^{(k)}$ at the end of k iterations. At each iteration, the conditional gradient algorithm obtains an optimal descent direction by optimizing the hyperplane to the convex Squared Loss function over the space of choice probabilities denoted as \bar{P} . The choice probability for customer type k is obtained by finding the minimum choice probability in \bar{P} of this hyperplane. The optimal descent direction obtained at each iteration k is taken as the choice probability of new customer type k subtracted with the $\mathbf{g}^{(k-1)}$ that is obtained in the step 3 of the previous iteration. In the proportions update step, the

likelihood of the customer type k is denoted as $\alpha^{(k)}$ and is calculated by finding the choice probabilities in a simplex that minimizes the Squared Loss function of the weighted sum of all the customer types identified. Essentially, this can be interpreted as searching for the likelihood vector of $\alpha^{(k)}$ that minimizes the error function. The input for the next iteration in our optimization of descent direction (step 4, support finding step) is updated with the new $\alpha^{(k)}$ and g^k by simply re-weighting g^k with the new convex sum of choice probabilities and likelihood. The final solution at each iteration k is based on the convex sum of both the customer type k and the previous solution g^{k-1} .

Algorithm 1 [33]

- 1: Initialize: $k = 0$; $g^{(0)}$ sampled based on standard uniform distribution, $\alpha^{(0)} = (1)$
 - 2: While stopping condition is not met **DO**
 - 3: $k \leftarrow k + 1$
 - 4: Compute $f^{(k)} \in \text{argmin}_{v \in \bar{P}} \langle \nabla \text{loss}(g^{(k-1)}), v - g^{(k-1)} \rangle$ (Support finding step)
 - 5: Compute $\alpha^{(k)} \in \text{argmin}_{\alpha \in \Delta^k} \text{loss}(\alpha_0 g^{(0)} + \sum_{s=1}^k \alpha_s f^{(s)})$ (Proportions update step)
 - 6: Update $g^{(k)} = \alpha_0^{(k)} g^{(0)} + \sum_{s=1}^k \alpha_s^{(k)} f^{(s)}$
 - 7: **end while**
 - 8: Output: Probabilities $\alpha_0^{(k)}, \alpha_1^{(k)}, \alpha_2^{(k)}, \dots, \alpha_k^{(k)}$ for each corresponding customer types $g^{(0)}, f^{(1)}, \dots, f^{(k)}$
-

6.2 Results and Comparative study

The results show that the mixing distribution is made up of 3 different customer types g^1, g^2, g^3 . Each of the g^1, g^2, g^3 represents a choice probability across all 95 goods and has a corresponding likelihoods which are: $\alpha^1 = 0.002, \alpha^2 = 0.996$ and $\alpha^3 = 0.002$. In this subsection, we would like to carry out a comparative study of the choice probabilities for each of the customer types (g^1, g^2, g^3) obtained in this section with our proposed approach in section 4 and 5. Using the proposed model for promotion driven demand in section 4, we segment the customers into 3 clusters via K-means. We seek to obtain the vector of choice probabilities for each of the cluster, CP^i , based on the market share of each product in each cluster. The entry in the vector of choice probabilities is denoted as $CP_j^i = \frac{N_j^i}{N^i}$ where CP_j^i denotes the choice probability for product j in cluster i , N_j^i denotes the number of times good j in cluster i was purchased and N^i denotes the

total number of purchases across all goods in cluster i . Hence, we obtain the choice probabilities CP^1, CP^2, CP^3 . for each corresponding cluster. Next, we obtain the customer's weighted choice probabilities based on $g = \alpha^1 * g^1 + \alpha^2 * g^2 + \alpha^3 * g^3$ and $CP = w^1 * CP^1 + w^2 * CP^2 + w^3 * CP^3$, whereby $w^i = N^i / \sum_{i=1}^3 N^i$, which denotes the size of the cluster, that is, the weight for each cluster based on their number of purchases across all goods. The figure 6.2.1 below shows the plot of choice probabilities in our proposed approach CP and the choice probabilities g obtained based on the conditional gradient approach. The x-axis is the good ID and each point in the plot represents the customer's predicted choice probability for that particular good ID.

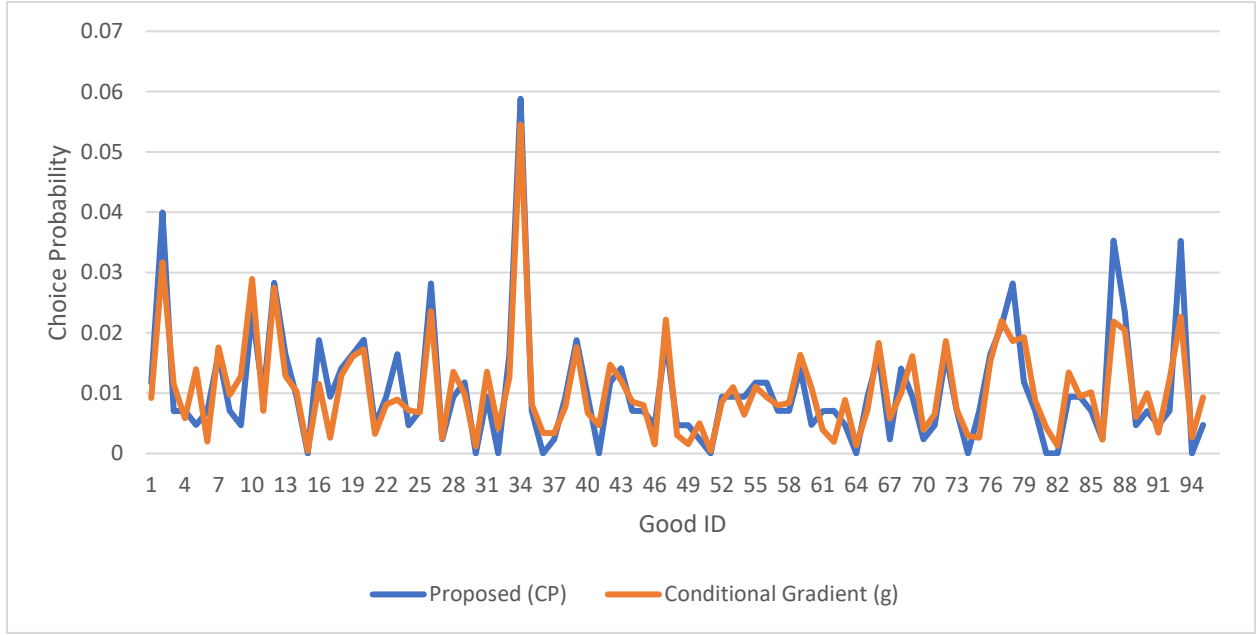


Figure 6.2.1 Plot of choice probabilities in CP and g

To compare the accuracy of the choice probabilities CP (obtained in section 4, 5) with g (obtained in section 6, based on Jagabathula's paper), we also compute the absolute difference for choice probability of CP and g for every good, that is, $|CP_j - g_j|$. This represents the absolute deviation in terms of market share of our proposed approach from the Conditional Gradient approach in Jagabathula's paper. To measure the percentage difference in this market share, we compute $|CP_j - g_j|/g_j$. The figure for this absolute percentage error, denoted as $|CP_j - g_j|/g_j$, across the good ID is provided in the Appendix section 12.5 (figure 12.8). Since each good is purchased a different number of times, each good will therefore have a different weightage towards representing the entire set of transactions in the dataset, and hence, a different weightage to the promotion driven demand. To compute the average percentage difference of the market share in CP and g , we have to weight each percentage difference $percntdiff_j = |CP_j - g_j|/g_j$ with $\delta_j = N_j / \sum_{j=1}^{95} N_j$. The figure for this weightage across good ID, denoted as $\delta_j = N_j / \sum_{j=1}^{95} N_j$, is provided in the Appendix section 12.5 (figure 12.9). The average percentage difference in market

share between the propose technique **CP** and the technique proposed in Jagabathula’s paper, **g** is denoted as $\sum_j \delta_j * percentdiff_j = 29.6\%$. This means that the choice probability, **CP**, obtained in section 4 and 5, on average is around 70% close to the choice probability predicted in **g**. This shows that our proposed approach in section 4 and 5 is a viable approach in identifying the customer types based on the product-level features. Additionally, based on the figures 12.8 and 12.9, we can observe that majority of the error in the choice probabilities between our proposed approach and the Conditional Gradient approach lies in good ID which has a small number of purchases. With a larger dataset consisting of higher number of transactions for these good IDs, we believe that our proposed approach which uses data-driven clustering models can yield a lower error based on the benchmark against Jagabathula’s approach.

7 MACHINE LEARNING MODELS AND RESULTS

In the previous sections, we have measured the customer's promotion driven demand and segmented them and then benchmarked the performance of our proposed approach with existing models. We have shown that our proposed approach is viable. As discussed in the introduction, our goal is to estimate the demand and optimize the full-cut promotions for revenue management. However, optimization of full-cut promotions only enables us to identify the best type of full-cut promotion in a particular setting, based on the transactional dataset. Therefore, the motivation for applying machine learning models is to understand the relationship between the features (such as market price, number of products purchased in an order) and the promotion driven demand. This can provide insights into how the changes in features (apart from discount and threshold amount) can impact the customers' attraction towards the full-cut promotion. This enables the E-commerce companies to understand on what features to fine-tune to drive the customers' attraction beyond just optimizing the threshold and discount amounts. E-commerce companies can also understand how and why customers react towards different discount and threshold amounts from the customers' preferences towards certain features.

We train various machine learning models to estimate the demand for the full-cut promotion using the features in table 2.1. The machine learning models used are: Decision Regression Trees (Basic Regression Trees with CART, Bagged Trees, Boosted Trees with GBM, XGBoost, H2O, LightGBM), K-Nearest Neighbors and Feedforward Neural Network. The model performance is assessed by comparing the Mean Square Errors (MSE) and Mean Absolute Error (MAE) of the pseudo-test set via 10-fold cross-validation and test-set via random sampling 30% of the dataset. All the models are trained by setting the seed as 123. Models on regression analysis are not used in this research due to the violation of assumptions in regression analysis. This is discussed in more details in Appendix section 12.1. A larger emphasis is placed on decision regression trees as decision trees produces results that are generally more interpretable than artificial neural networks. Additionally, a decision tree models the human decision-making process more closely due to the stepwise flow of nodes and leaves [4]. Table 7.1 below summarizes results of the model performance. Based on the results, we observe that boosted regression tree with GBM, XGBoost and H2O have the best model performance.

	MSE (Pseudo Test set)	MAE (Pseudo Test set)	MSE (Test set)	MAE (Test set)
Regression Tree (CART)	-	-	0.638%	4.223%
Regression Tree (Bagging)	-	-	0.582%	4.014%
Regression Tree (Boosted, GBM)	-	-	0.143%	1.178%
Regression Tree (XGBoost)	-	-	0.1491 %	1.392%
Regression Tree (H2O)	-	-	0.140%	1.226%
Regression Tree (Light GBM)	0.836%	5.172%	0.778%	5.051%
Artificial Neural Networks	0.894%	4.890%	0.847%	4.844%
K-Nearest Neighbors	0.546%	4.080%	1.283%	6.910%

Table 7.1 Summary of machine learning model performance

7.1 Decision Regression Tree

A decision tree is a tree in which each branch node represents a choice between several alternatives, and each leaf node represents a decision. Decision tree starts with a root node and is split into two nodes via recursive partitioning based on a decision tree learning algorithm. The output is a decision tree which can be linearized into decision rules [5], each branch represents the thought process and conditions that a customer goes through and each leaf node represents the possible scenario of the customer's decision, or in this case, how much promotion driven demand the customer has.

ID3, CART and C4.5 are the most common decision tree algorithms which use different pruning strategies and splitting criteria for splitting the nodes at each level [6]. Amongst the different decision tree algorithms, CART (Classification and Regression Trees) is used in this research as it can handle both categorical and numeric inputs, handle outliers and generate regression trees.

CART is characterized in its construction of binary trees, namely each internal node has exactly two outgoing edges. The splits are selected using the twoing criteria and the obtained tree is pruned by cost-complexity Pruning. CART identify splits that minimize the prediction squared error. ID3 is not selected as it is unable to handle numeric input features such as market price. CART is preferred to C4.5 as C4.5 is susceptible to outliers, which are orders with $Sales_i \gg PromotionThreshold$. This is discussed in more detail in Appendix section 12.2 and the comparison of characteristics across different decision tree algorithms is shown in table 12.4.

Decision tree is a data-driven approach that does not require any assumptions on the dataset. The regression trees generated in this research uses CART algorithm and is pruned based on the cost-complexity criterion with the parameters minsplit, maxdepth of the tree and complexity parameter (CP). Minsplit is the minimum number of data points required to attempt a split before it is forced to create a terminal node. Maxdepth is the maximum number of internal nodes between the root node and the terminal nodes. CP is used to control and select the optimal tree size and is a cost complexity parameter, α , which penalizes the objective function for the number of terminal nodes of the tree, denoted as $|T|$. Specifically, the equation for cost complexity criterion is

$$minimize\{SSE + \alpha|T|\}$$

The values for the parameters are obtained using a grid search of minsplit from 5 to 30 and maxdepth from 8 to 30. For each combination of minsplit and maxdepth, a function is created to obtain the optimal CP value. A 10-fold cross validation is carried out to calculate the cross-validated error for each model. The model with the least error is selected as the optimal model and is then trained and tested to identify the Mean Squared Error (MSE) and Mean Absolute Error (MAE). Figure 7.1 shows the regression tree plot and the model performance is summarized in table 7.2 below. The regression tree plot shows that the most important feature is the discount amount followed by the original purchase amount of the customer (before making the additional purchase of the minimum priced good). The customer thinks about the discount amount from the full-cut promotion first and then think about what their purchase amount are before deciding if they are interested in the full-cut promotion. For example, customers will observe whether the discount amount is more than ¥50 and then observe whether their original purchase amount is more than ¥178 and ¥167. If the original purchase amount less than ¥167 and the discount amount is more than ¥50, then the customer has an estimated full-cut promotion driven demand of 0.64.

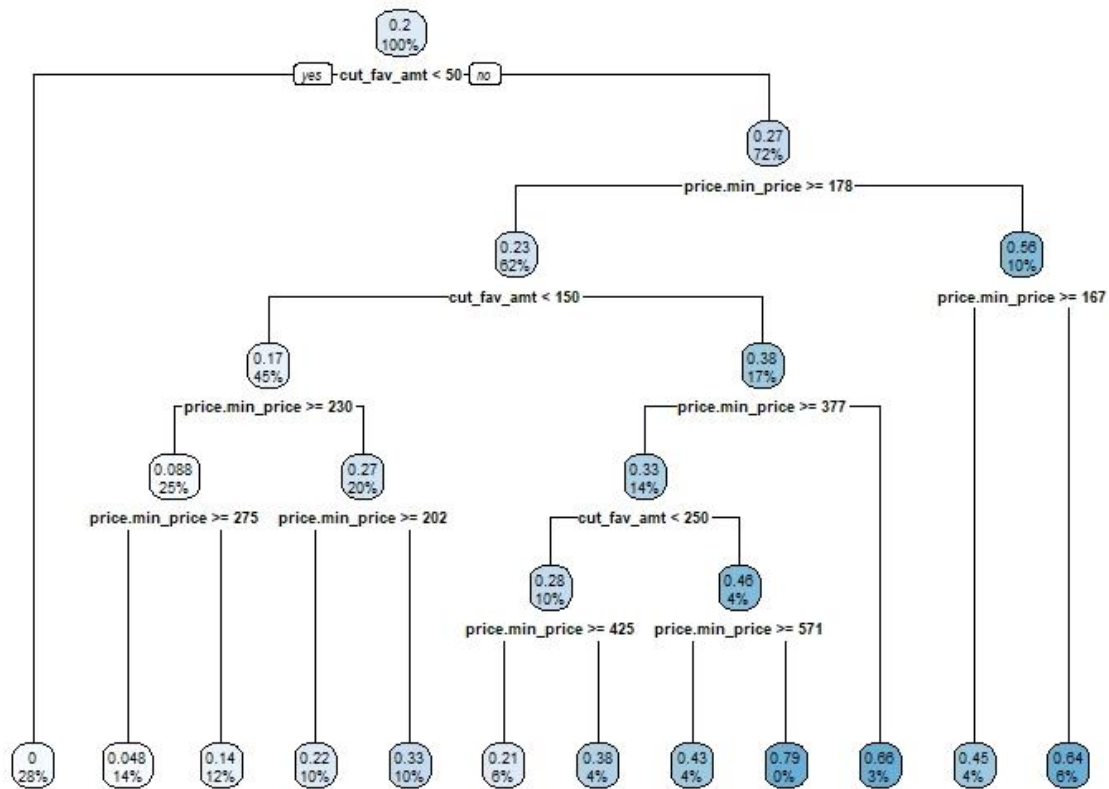


Figure 7.1 Regression Tree (CART)

	MSE (Test set)	MAE (Test set)
Regression Tree (CART)	0.638%	4.223%

Table 7.2 Summary of Regression Tree model performance

7.2 Decision Regression Tree (Bagging)

The construction of a single decision tree can be non-robust and a small change in the dataset can cause a large change in the output regression tree [7]. Although parameter tuning and pruning help to reduce the variance of the output, we can further reduce overfitting and variance of regression tree through bagging, which is a combination of bootstrapping and aggregating [8]. Bagging involves the construction of bootstrap samples so that we can generate an unpruned decision tree from each of this bootstrap sample dataset. The results from the multiple decision trees are averaged to create an overall average predicted value. Generally, a bootstrap sample

will contain 63% of the entries in the training dataset and the remaining 33% untrained entries in the training dataset is known as the Out-of-Bag (OOB) samples [8]. These samples are used as pseudo test dataset. The bagging process is discussed in the Appendix section 12.3 in more details. We set the parameter, number of trees, from a range of 10 to 50 and the optimal value is discovered to be 29 trees based on the lowest error (MSE) as shown in Figure 7.2. The bagged regression tree is then trained and tested using the optimal number of trees, to identify the Mean Squared Error (MSE) and Mean Absolute Error (MAE). Since multiple trees are created, bagged regression trees do not have a single regression tree plot. Instead, the results at each split for all the regression trees are aggregated to determine the variable importance of the features in figure 7.3. The process to identify the variable importance for each feature is discussed in Appendix section 12.3. The model performance has improved compared to regression trees used in section 7.2 and the results are summarized in table 7.3 below. This also shows that process of bagging helps to improve the accuracy and variance of the result at the expense of sacrificing the visibility of a single regression tree.

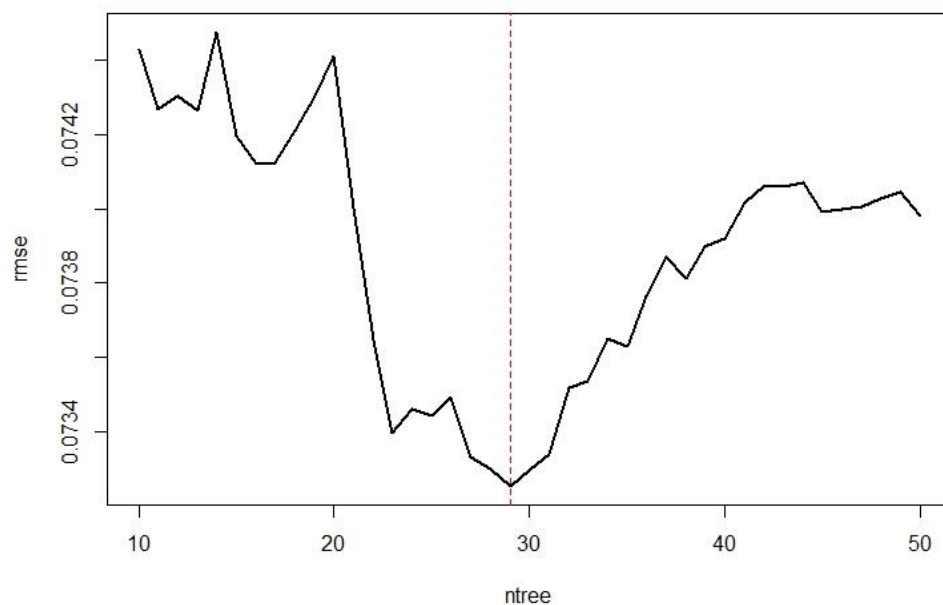


Figure 7.2 Optimal Number of Trees for Bagging

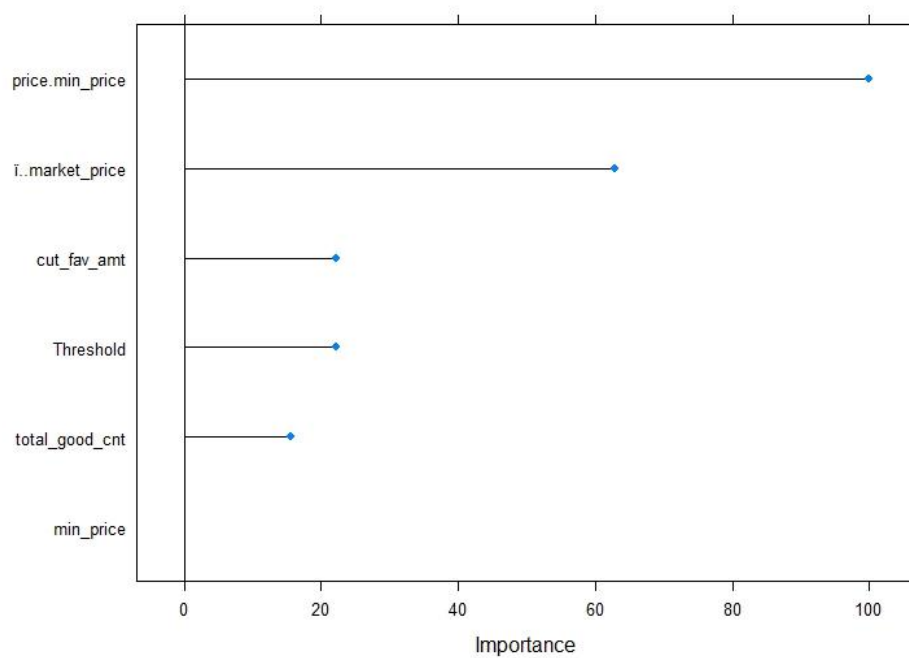


Figure 7.3 Variable Importance plot for Regression Tree (Bagging)

	MSE (Test set)	MAE (Test set)
Regression Tree (Bagging)	0.582%	4.014%

Table 7.3 Summary of Regression Tree model performance

The trees constructed in bagging may not be completely independent of each other since all the input features are considered at every split of every tree. Generally, random forest is introduced to decorrelate the trees constructed in bagging by considering a random subset of the input features at every split in each tree. This enables the construction of decision trees which have a different structure and hence, more likely to be independent from each other. This helps to achieve a greater reduction in variance of output. However, since there are only 6 input features, the application of random forests will not significantly improve the model performance and therefore is not employed in this research. Instead, we will discuss the use of boosted decision trees as a training model in the next subsection.

7.3 Decision Regression Tree (GBM)

Bagging and Random Forests involve building an ensemble of models where each individual model predicts the outcome and the ensemble simply aggregates the predicted values. In Boosting, an ensemble of shallow and weak successive trees is added sequentially with each tree learning and improving based on the residuals of the previous models. The construction of successive weak trees is computationally cheap and allows the algorithm to learn slowly; making incremental improvements in the reduction of residuals. This enables the model to avoid overfitting as the learning and construction process can be terminated with a stopping criterion. There are many variants of Gradient Boosting Machines (GBM) which are used to implement Boosting. Additional details on GBM as well as other boosting techniques such as Adaptive Boosting (AdaBoost) are provided in Appendix section 12.3. In this research, we used the original R implementation of GBM, XGBoost, H2O and Light GBM to train the models and carry out a comparative study.

The values of the parameters: minimum number of observations allowed in the terminal nodes, depth of trees, learning rate and subsampling are tuned using grid search. Specifically, the range of depth of trees is {1,3,5}, learning rate is {0.01,0.1,0.3}, minimum number of observations is {5,10,15}, and subsampling is {0.65,0.8,1}. To reduce the computation cost for the tuning of parameters, instead of 10-fold cross validation, a simple valuation set approach of 75% of pseudo-training set is used to evaluate performance on the remaining 25% pseudo-test set. The top 10 models with the lowest cross-validation error is identified and a second grid search is conducted based on the range of values for the tuning parameters. The model with the optimal values for the tuning parameters is identified and the model is then trained and tested to identify the Mean Squared Error (MSE) and Mean Absolute Error (MAE). The optimal values are: number of trees = 5824, depth of trees = 3, learning rate = 0.05, minimum number of observations = 5, subsampling = 0.65. Additionally, a plot of the importance of input features is created to understand which feature has the largest influence on the output. In figure 7.5, the partial dependence plot shows the average change in output as discount amount varies while holding all other features constant. The model performance is summarized in table 7.4 below and figure 7.4 shows the feature importance plot. The results show that the discount amount has the largest influence on the full-cut promotion driven demand. Additionally, the partial dependence plot shows that at specific discount amounts, the predicted full-cut promotion demand increases for the customers.

	MSE (Test set)	MAE (Test set)
Regression Tree (GBM)	0.143%	1.178%

Table 7.4 Summary of Regression Tree model performance

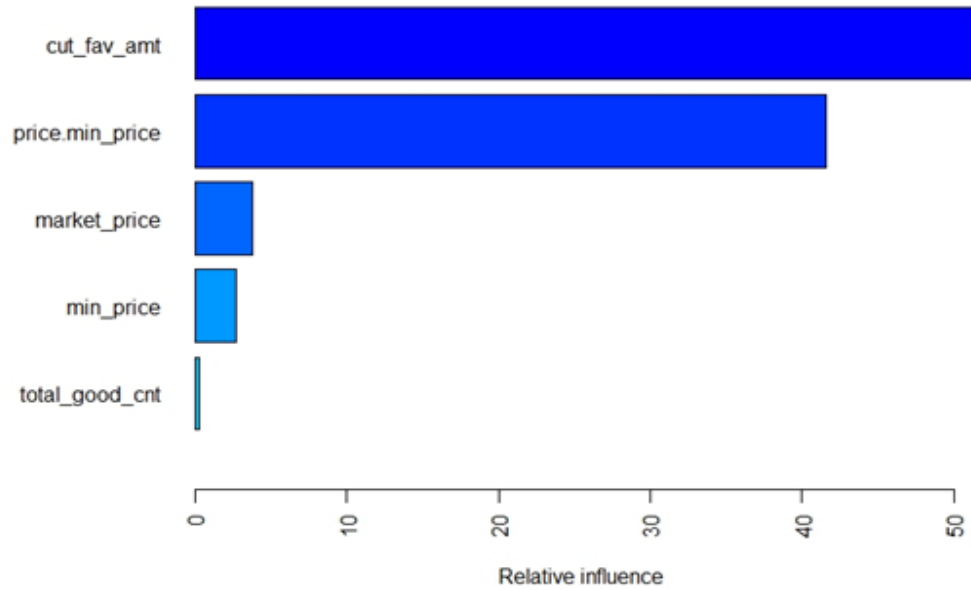


Figure 7.4 Variable Importance plot

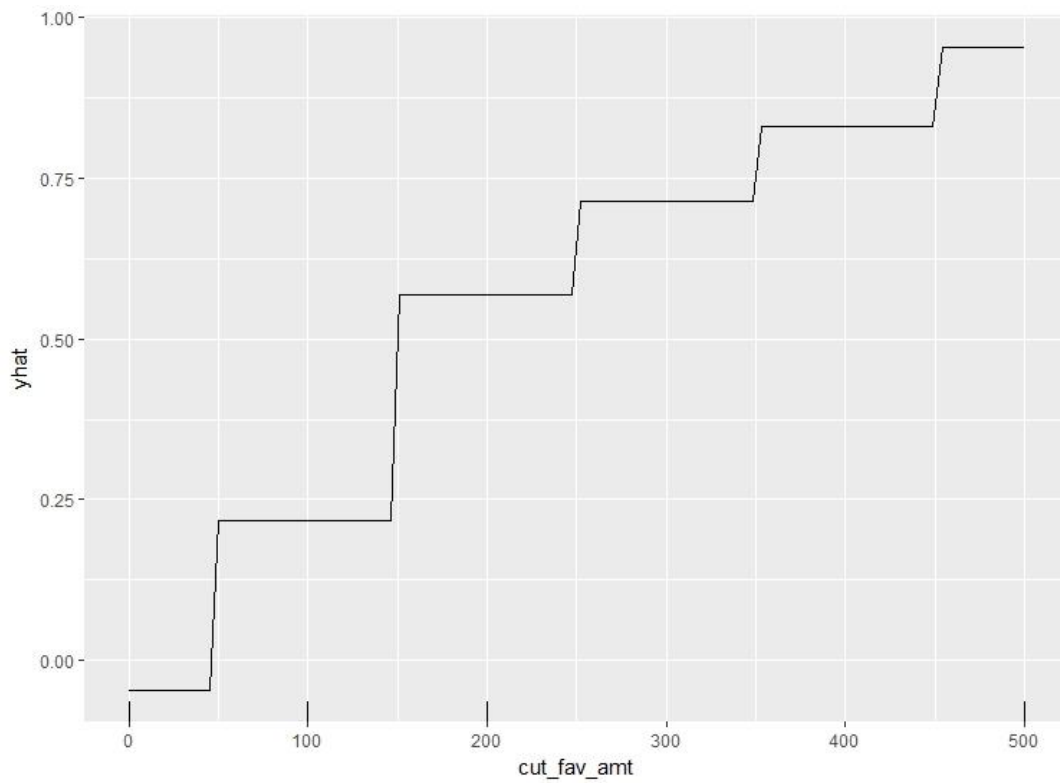


Figure 7.5 Partial Dependence plot for discount amount

7.4 Decision Regression Tree (XGBoost)

One of the disadvantages of Gradient Boosting Machines (GBM) is the high computational costs; GBMs often need to construct many trees (>1000) and this can be time and memory exhaustive [14]. Additionally, the high model flexibility results in multiple parameters to be tuned (number of iterations, tree depth, regularization parameters, etc.) [14]. This requires a large grid search during tuning, which adds to the computational costs.

Extreme Gradient Boosting (XGBoost) is an efficient implementation of gradient boosting framework. XGBoost includes an efficient linear model solver and tree learning algorithms and enables the feature of parallel computation on a single machine [15]. While it is not possible to parallelize the ensemble because each tree is dependent on the previous, XGBoost parallelizes the building of the several nodes within each depth of each tree. This enables XGBoost to perform faster than GBM. Additionally, XGBoost uses a more regularized model formalization to control over-fitting, which gives it better performance [15]. The discussion and comparative study for various boosting methods: XGBoost, GBM are discussed in more details in Appendix section 12.3. The values of the parameters: minimum number of observations required in each terminal node, depth of trees, learning rate, percent of columns to sample for each tree and subsampling are tuned using grid search. Specifically, the range of depth of trees is {1, 3, 5, 7}, learning rate is {0.01, 0.05, 0.1, 0.3}, minimum number of observations required in each terminal node is {1, 3, 5, 7}, and subsampling is {0.65, 0.8, 1} and percent of columns to sample from for each tree is {0.8, 0.9, 1}.

A 10-fold cross validation is used to evaluate performance of each model. The top 10 models with the lowest cross-validation error is identified and a second grid search is conducted based on the range of values for the tuning parameters. The model with the optimal values for the tuning parameters is identified and the model is then trained and tested to identify the Mean Squared Error (MSE) and Mean Absolute Error (MAE). Additionally, a plot of the importance of input features is created to understand which feature has the largest influence on target as shown in figure 7.6. The model performance is summarized in table 7.5 below and figure 7.6 shows the feature importance plot.

	MSE (Test set)	MAE (Test set)
Regression Tree (XGBoost)	0.1491 %	1.392%

Table 7.5 Summary of Regression Tree model performance

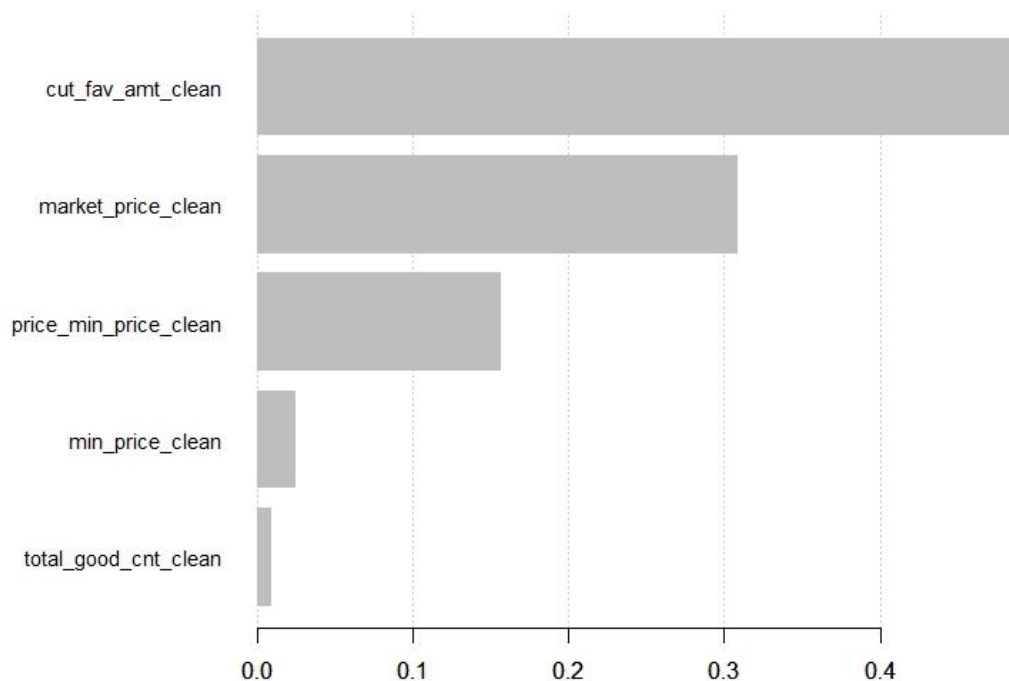


Figure 7.6 Variable Importance plot for XGBoost

7.5 Decision Regression Tree (GBM with H2O)

H2O is a powerful R package containing various algorithms for GBM and is also an efficient java-based interface [16]. For additional details on H2O and the features in GBM from H2O, refer to the Appendix section 12.3. Even though XGBoost is approximately computationally faster than the GBM, executing 10-fold cross validation in the grid search to obtain the optimal model with the least cross-validated error will be computationally expensive (a running time of approximately 6 hours, due to the large number of tuning parameters). GBM implemented with the H2O package enables a random discrete grid search to reduce the running time. Instead of going through the combinatorial permutations of the parameters, random discrete grid search jumps from one random combination to another and stop once a certain level of improvement has been made, certain amount of time has been exceeded, or a certain number of models have been ran. By setting a strong stopping criterion, using a random discrete search path can reduce the running time while obtaining a relatively good performing model [16].

The values of the parameters in the grid search follows the grid search parameters in XGBoost. A 10-fold cross validation is used to evaluate performance of each model. The top 10 models with the lowest cross-validation error is identified and a second grid search is conducted based on the

range of values for the tuning parameters. The model with the optimal values for the tuning parameters is identified and the model is then trained and tested to identify the Mean Squared Error (MSE) and Mean Absolute Error (MAE). Additionally, a plot of the importance of input features is created to understand which feature has the largest influence on the output. The model performance is summarized in table 7.6 below and figure 7.7 shows the feature importance plot. The results show that discount amount has the highest influence on the full-cut promotion driven demand.

	MSE (Test set)	MAE (Test set)
Regression Tree (GBM with H2O)	0.140%	1.226%

Table 7.6 Summary of Regression Tree model performance

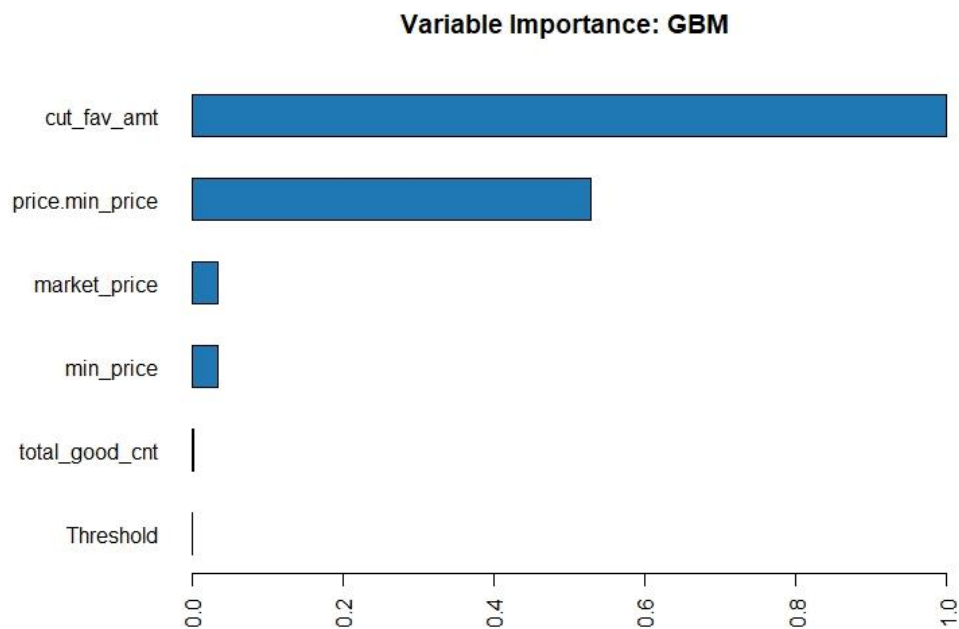


Figure 7.7 Feature Importance plot for H2O

7.6 Decision Regression Tree (Light GBM)

Light Gradient Boosting Machine (Light GBM) is a gradient boosting framework that uses tree-based learning algorithm. In Light GBM, the trees are grown vertically (leaf-wise) while in other algorithms, the trees are grown horizontally (level-wise), therefore reducing residuals to a larger extent and leading to greater accuracy of the predicted outputs [17]. The differences in leaf-wise and depth-wise tree growing strategies are shown in the figure 12.5 from Appendix section 12.3. Another unique feature of Light GBM is its ability to handle a large size dataset, while taking a lower memory and shorter running time [17].

The values of the parameters: learning rate, subsampling, regularization parameters lambda and alpha, number of boosting iterations, number of leaves in a full tree, boosting type and percent of columns to sample for each tree are tuned using grid search. Specifically, the range of learning rate is {0.001, 0.005, 0.01, 0.03, 0.05, 0.07, 0.1}, subsampling is {0.65, 0.7, 0.75, 0.8, 1}, lambda is {1, 1.2, 1.4}, alpha is {1, 1.2}, number of boosting iterations is {40, 50, 100}, number of leaves is {6, 8, 12, 16} and boosting type is set as traditional gradient boosting decision tree, and percent of columns to sample from for each tree is {0.65, 0.66}.

A 10-fold cross validation is used to evaluate performance of each model. The top 10 models with the lowest cross-validation error is identified and a second grid search is conducted based on the range of values for the tuning parameters. The model with the optimal values for the tuning parameters is identified and the model is then trained and tested to identify the Mean Squared Error (MSE) and Mean Absolute Error (MAE). Additionally, a plot of the importance of input features is created to understand which feature has the largest influence on target. The model performance is summarized in table 7.7 below and figure 7.8 shows the feature importance plot. The top 2 features which have the highest influence on full-cut promotion are the discount amount and customer's original purchase amount.

	MSE (Pseudo set)	MAE (Pseudo set)	MSE (Test set)	MAE (Test set)
Regression Tree (Light GBM)	0.836%	5.172%	0.778%	5.051%

Table 7.7 Summary of Regression Tree model performance

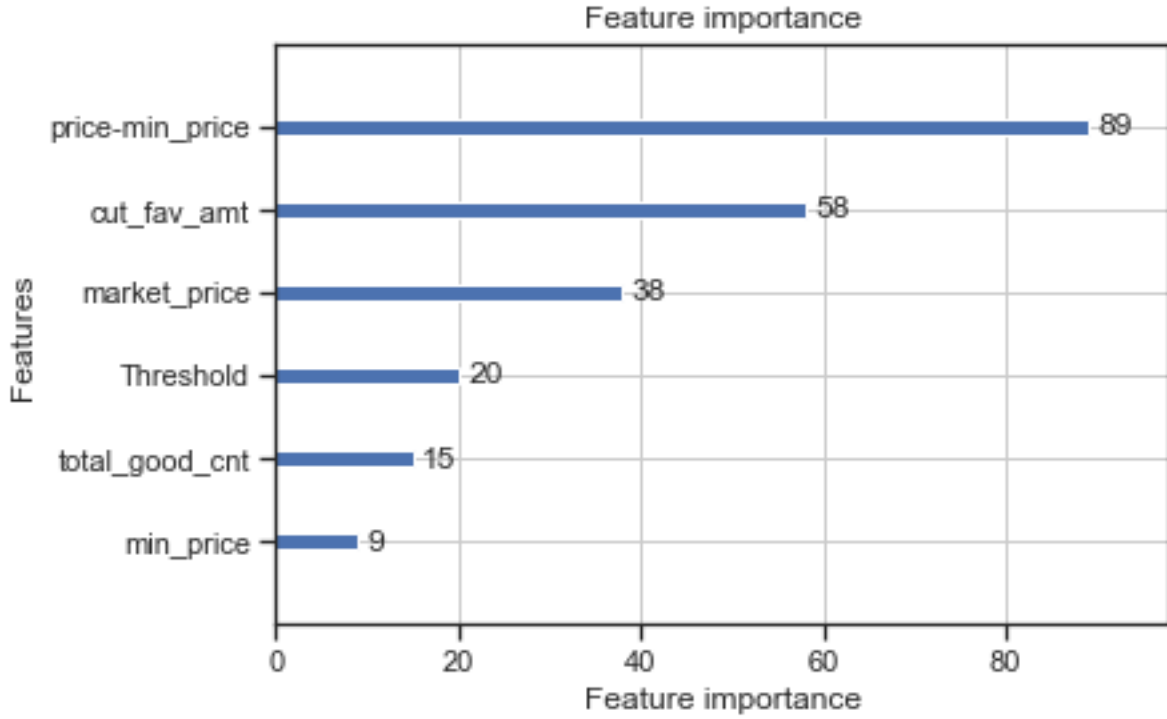


Figure 7.8 Feature Importance plot

7.7 K-Nearest Neighbor

The K-Nearest Neighbor (KNN) regression is a non-parametric model and does not make any assumptions on the underlying statistical distribution of the dataset. KNN is chosen as the baseline model it is simple and easy to implement. Given an input of the features \mathcal{X}_i , KNN regression finds the neighborhood of \mathcal{X}_i , denoted as $\mathcal{N}_{\mathcal{X}_i}$, containing the nearest \mathcal{K} neighbors via Euclidean distance [18]. The output, *target*, is calculated based on the mean of $target_i$, where $\mathcal{X}_i \in \mathcal{N}_{\mathcal{X}_i}$. The input features in training and test datasets are standardized separately into the range of 0 to 1 value, as the features are measured in different units. The optimal \mathcal{K} was selected by performing 10-fold cross-validation approach over a range of \mathcal{K} values from 1 to 100. The optimal $\mathcal{K} = 4$. The results of the model performance 4-NN Classifier is shown below. Figure 7.9 shows the cross-validated accuracy across multiple \mathcal{K} values. The cross-validated accuracy refers to the accuracy rate, $(\hat{f}(x_i) - y_i)^2$ obtained from 10-fold cross validation. The table 7.8 summarizes the model performance of 4-NN Classifier.

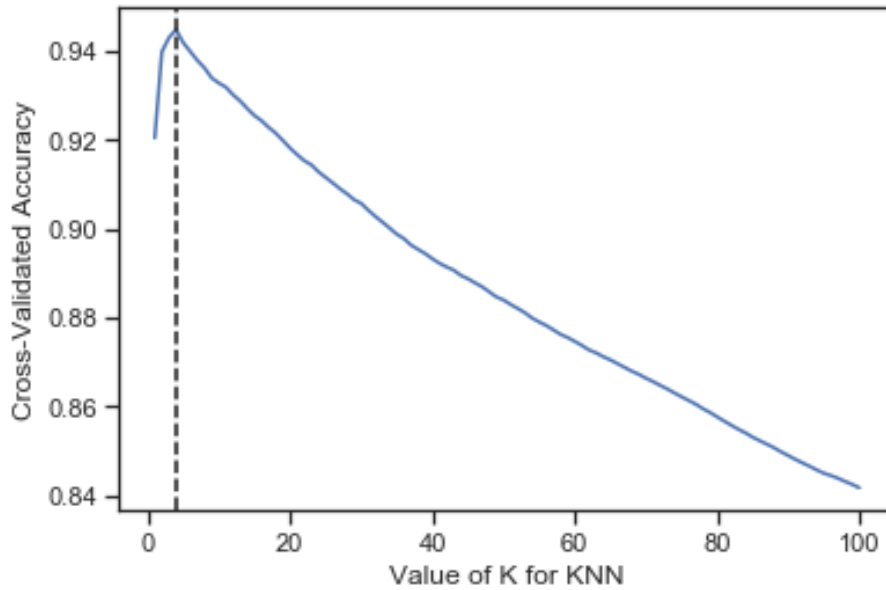


Figure 7.9 Cross-Validated Accuracy over K values from 1 to 100

	MSE (Pseudo Test set)	MAE (Pseudo Test set)	MSE (Test set)	MAE (Test set)
K-Nearest Neighbors	0.546%	4.080%	1.283%	6.910%

Table 7.8 Summary of 4-NN model performance

7.8 Feedforward Neural Network

Neural network has evolved to encompass a large class of models and learning methods. The discussion of artificial neural network, multi-layer feedforward neural network is provided in Appendix section 12.4. We use a Feedforward Neural Network involving 2 hidden layers with a back-propagation algorithm to compute the weights that are passed onto each neuron. The activation function in the output layer is a linear function and the activation in the hidden layers is chosen to be ReLu (Rectified Linear Unit) as ReLu is a more popular approach than Sigmoid or Tanh activation functions. This is because ReLu is a simpler function which enables a faster running time, and yet at the same time resolves the issue of vanishing gradient and non-zero centered problem in a Sigmoid function [38]. Two hidden layers are selected to prevent overfitting and the number of neurons within each hidden layer is set as 4. This is based on the rule-of-

thumb that the optimal size of the hidden layer is the mean of the size of input and output layers [20,21,22].

The input features are standardized before the weights for each input feature is calculated in the hidden and output layers. An issue with FNN is that output depends on the initial values to which they are randomized. Therefore, FNN was ran for 5 times with 100 epochs for each run and the best result was recorded. Figure 7.10 shows the performance of the predicted output from FNN against the actual test labels. The table 7.9 summarizes the model performance of FNN.

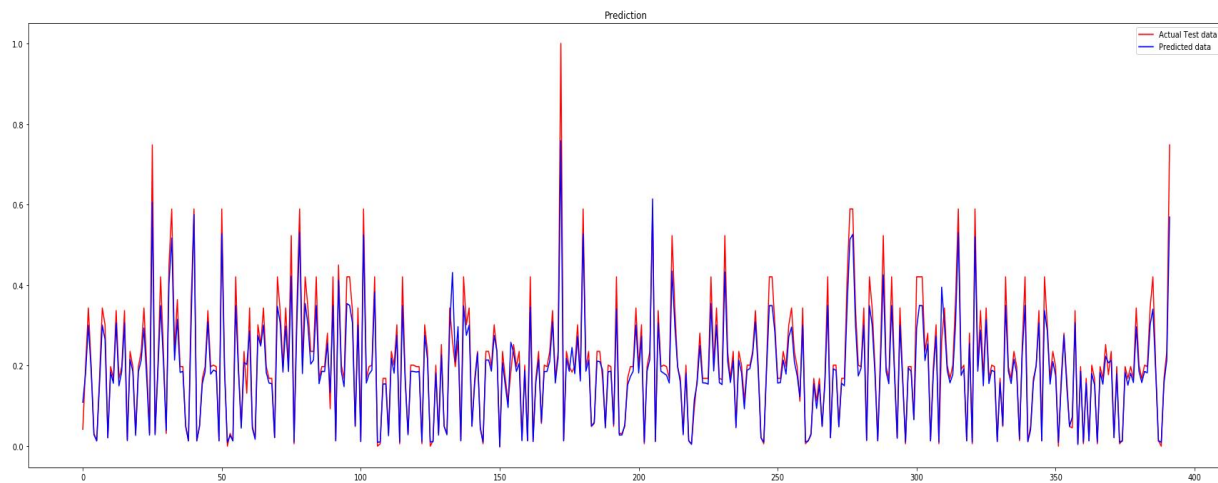


Figure 7.10 Predicted Output (Blue) vs Actual Output (Red) in Test dataset

	MSE (Pseudo Test set)	MAE (Pseudo Test set)	MSE (Test set)	MAE (Test set)
Artificial Neural Networks	0.894%	4.890%	0.847%	4.844%

Table 7.9 Summary of ANN model performance

8 DISCRETE CHOICE MODELS

In this section, we seek to use the traditional and popular discrete choice models to model consumer choices between subscribing or not subscribing to the full-cut promotion and hence, estimate this promotion driven demand. Discrete choice models estimate the probability that a person chooses a choice over an assortment of choices and this can be used to forecast how people's choices will change based on the changes in attributes such as demographics or changes in attributes of the alternatives [25]. There are various discrete choice models available, such as: Multinomial Logistic Regression, Multinomial Probit Regression, Binary Logistic Regression, Binary Probit Regression, Multiple Logistic Regression, Ordinal Logistic Regression, Nested Logit Regression. The most widely used method to estimate discrete choice models under complete information is the Multinomial Logistic Regression or Binary Logistic Regression which uses the Maximum Likelihood Estimation (MLE) method [26, 27], to maximize the probability of a choice(s) being selected against the other assortment of choices. In this section, Binary Logistic Regression is selected since there are only two choices: customers subscribing to full-cut promotion or not subscribing to full-cut promotion.

In this research, the choice set contains two choices for customer i , with $y_i = 1$ defined as customer i subscribing to the promotion, and $y_i = 0$ defined as customer i who is not subscribing to the promotion. Logistic regression uses the cumulative distribution function of the logistic distribution to define the transformation on the regression model, whereas the probit model uses the cumulative distribution function of the standard normal distribution to do so. In multiple discrete choices, multiple regression model must be carried out for each pair of outcomes: since each analysis is potentially run on a different sample, this may end up with the probability of choosing all possible outcome categories greater than 1. Ordinal logistic regression is not selected as the choices do not have an ordinal utility property. The nested logistic model is a generalization of the multinomial logit model based on the idea that some choices may be joined in groups (nests). Since there are only two choices, a nested logistic model is not relevant for this research. In the binary logistic regression model, the choice, y_i is trained on two features: minimum price of the good in an order (x_1) and the customer's purchase amount in an order discounted with the minimum priced good (x_2),

$$y_i = \sigma(x_i) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2)}}$$
$$\ln\left(\frac{y_i}{1 - y_i}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

The feature, minimum price (x_1), is defined as the price of the good which is the lowest amongst the other prices of good in an order. The feature, order price - minimum price (x_2), is defined as the entire purchase amount in the order discounted with the minimum price (x_1) and this provides the intuition of the customer's original purchase amount. x_1 represents the lowest price that the

user purchased in order to reach the threshold amounts (¥199, ¥398, ¥597) and subscribe to the full-cut promotion. A higher minimum price x_1 indicates that a customer is willing to pay a higher amount of money to subscribe to the promotion and hence a higher demand to enjoy the full-cut promotion, whereas a lower minimum price indicates a lower willingness. x_2 represents the total purchase amount that the customer originally intends to purchase without making additional purchase to enjoy the full-cut promotion. Hence, this investigates the relationship between the customer original purchase amount and whether the customer will subscribe to the promotion. For example, we are examining the effect of whether the user is more willing to subscribe to the promotion with a threshold ¥199 given that his purchase is ¥150, compared to a user whose purchase is ¥179 and a threshold of ¥199. The output, whether $y_i = 0$ or 1 is based on the probability of the user choosing $y_i = 1$. This probability represents the customer's utility for each choice over the other and in turn represents the full-cut promotion driven demand. The error component ε_i is a random variable which represents the impact of all the unobserved variables has on the utility of a customer i choosing a specific choice. The regression coefficients β_0 , β_1 , β_2 (which are estimated using maximum likelihood estimation) where the probability of the choice i made from one customer j based on one of his or her order is:

$$P_i = \prod_j P_{ij}^{y_{ij}}$$

The following log-likelihood function used in the MLE is:

$$\ln L = \sum_i \ln P_i = \sum_i \sum_j y_{ij} \ln P_{ij}$$

Additionally, this study assumes that the lowest priced good is the item that the user purchased to subscribe to the promotion. In reality, it is possible for a scenario where the price of goods that the customer originally intends to purchase have already met the threshold amount. It is also assumed that the users are aware of the existence of full-cut promotions. The binary logistic model assumes that collinearity is low between the features, the error terms are independent and identically distributed and the Independence of Irrelevant Alternatives (IIA). IIA assumes that adding or deleting alternative choices into the choice set does not affect the odds among the remaining outcomes. From the results of Binary Logistic Model, the final negative log-likelihood value is 16.16 and the residual deviance is 38.32.

$\beta_0 = -666.6, \beta_1 = 3.38, \beta_2 = 3.35$. One-unit increase in the variable x_1 is associated with the increase in the log odds of being the user subscribing to the promotion vs. not subscribing to the promotion by the value of 3.38. One-unit increase in the variable x_2 is associated with the increase in the log odds of being the user subscribing to the promotion vs. not subscribing to the promotion by the value of 3.35. The p-value from the 2-tailed Z tests is below the significance level of 0.05, rejecting the null hypothesis, therefore indicating that the Binary Logistic model is significant in

explaining the relationship between the features and output. This shows that both the minimum price and original purchase amount has approximately the same amount of positive influence towards the customer's full-cut promotion driven demand. The higher the original purchase amount, the higher the customer's full-cut promotion driven demand. The higher the minimum price of the good in an order, the higher the customer's full-cut promotion driven demand. This is intuitive as it shows that a customer who spends more is attracted to promotions to seek discounts. Additionally, an order which contains a higher minimum priced good is likely to show that the customer of that order is willing to spend more to reach the threshold amount and hence more willing to subscribe to the full-cut promotion.

The seed is set at 123 and the dataset is randomly split with 70% of the dataset being the training set and the remaining 30% as testing set. The misclassification rate is based on the number of incorrectly predicted choices in the testing set. Additionally, the misclassification rate is applied and calculated on the predicted output of XGBoost, by classifying the values (full-cut promotion demand) from 0.5 to 1 to an output of 1 and 0 otherwise. This misclassification rate for Binary Logit Model is 6.487% and is significantly higher than XGBoost, which is 1.669%. This shows that the XGBoost has a higher accuracy in predicting the customer's full-cut promotion demand and can fit the data better than a binary logit model which samples from a logistic distribution. This also shows that companies or researchers in operations and revenue management should seek to model full-cut promotion driven demand using data-driven approaches like decision trees instead of the popular and traditional approach of regression and discrete choice models.

	Misclassification rate
Binary Logit Model	6.487 %
XGBoost	1.669%

Table 8.1 Comparison of model performance

9 OPTIMISATION AND FUTURE WORK

In this section, we seek to create a methodology for companies to optimize their discount and threshold amount in the full-cut promotion for revenue management. This involves trying to maximize the traditional utility model in econometrics as an objective function and based on the information: full-cut promotion driven demand (in section 4), information of clusters of customers (in section 5), price of each good. Additionally, we will also discuss possible avenues for future work, incorporating different forms of optimization techniques and extending towards other contexts beyond E-commerce. We begin by obtaining the preference vectors, $w^{(1)}, w^{(2)}, \dots, w^{(k)}$ for each of the corresponding choice probabilities for each customer type k (Section 6). Each preference vector is a 1 by n vector. This is carried out by optimizing based on the following objective function [34, 35]:

$$\min_{w^{(k)}} \sum_{j=1}^n w_j^{(k)'} \cdot z_j - p_j - \ln(x_{kj}) + \ln(x_{k0})$$

Whereby x_{kj} denotes the choice probability for good j by customer type k , x_{k0} denotes the choice probability of the customer not buying good j . Essentially, $x_{k0} = 1 - x_{kj}$. The objective function shown below models the utility of the customer:

$$\max_{x \in \Delta_n} E \left[\sum_{j=0}^n (U_{ij} - \alpha p_j) x_{kj} + \gamma_i \cdot \mathbf{1}_{\{\text{Order amount} \geq \theta\}} \cdot \delta \right] + \varepsilon$$

$$\Delta_n = \{x \in \mathbb{R}_+^{n+1} \mid \sum_{j=0}^n x_{ij} = 1\}$$

where $j \in \text{product universe } [n]$ and $j = 0$ denotes the outside option, x_{ij} denotes the choice probability of customer i purchasing good j . Δ_n is the unit simplex and denotes the constraint space for choice probabilities across goods. α denotes the homogenous price sensitivity parameter across all goods, where $\alpha > 0$. p_j denotes the price of good j , U_{ij} denotes the customer i 's deterministic valuation for good j . We evaluate $U_{ij} = w_k' * z_{ij}$, where customer i belongs to customer type k with a preference vector of w_k based on the features z_{ij} of good i . Therefore, $U_{ij} - \alpha p_j$ represents customer i 's surplus on good j . In this research,

$\gamma_i \cdot \mathbf{1}_{\{Order\ amount \geq \theta\}} \cdot \delta$ denotes the customer's utility as a result of the full-cut promotion discount, where γ_i denotes the customer i 's sensitivity parameter for full-cut promotion. θ denotes the threshold amount and δ denotes the discount amount in the full-cut promotion. Hence, this denotes a positive award in terms of the discount amount from the full-cut promotion, scaled with the customer's attraction towards this promotion. γ_i is obtained from the full-cut promotion demand in section 4. For the price sensitivity across goods, α , we have assumed to be homogenous in this research. Alternatively, α can be set as a vector with each element α_j reflecting the customer's overall price sensitivity towards each product j . E-commerce companies can carry out surveys to measure this price sensitivities in their E-commerce websites, include the demographics of both the customers who purchased or did not purchased the goods.

First, we set constraints and space for the threshold and discount amounts, θ and δ . For every pair of threshold and discount amount (θ, δ) , using the expectation maximization model, we seek to obtain the optimal choice probability x_i for each customer i , and across all goods j , based on the customer i 's features z_{ij} and sensitivity towards full-cut promotion, γ_i . After obtaining all the choice probabilities for each customer i , we find the expectation of all the choice probabilities and denote it as \mathbf{x} . Hence for each type of full-cut promotion, defined by the unique pair of threshold and discount amount, we have a corresponding choice probability \mathbf{x} . To identify the type of full-cut promotion that leads to the highest revenue, we simply take the product of \mathbf{x} with the vector of profit margin for each product, denoted as \mathbf{p} . Hence, we obtain the optimal full-cut promotion (θ^*, δ^*) with a highest corresponding value, $(\mathbf{x} \cdot \mathbf{p})^*$.

The future work in this research includes the use of various optimization techniques to solve the above objective function with a comparative study of the performance of each optimization technique. Another area of future work involves relaxing the assumptions made in this research. It is interesting to consider the price sensitivity across products as a non-homogenous factor. In this research, we assumed that there are no stock-out scenarios. This assumption may be violated, and the sales of the stock-out products will be underestimated and hence, underestimating the full-cut promotion driven demand in section 4. Future areas of work may examine the possibility of stock-out products. Additionally, in section 6, we have assumed that there are no changes in the product assortment across the different time intervals. This is primarily due to limited information in our dataset to capture the choice probabilities across all products at each time interval. In reality, the product assortment changes over time intervals as companies delist old products and include new product offerings. Therefore, a future area of work can involve the consideration of time intervals whereby at each time interval, customers purchase from an offered set of products and offered set across time intervals are different.

10 CONCLUSIONS

In this research paper, we have introduced a methodology for companies to identify the types of full-cut promotions in an aggregated transactional dataset. This also enables companies to identify the types of the discount and threshold amounts that customers are more attracted to base on frequencies in the histogram plot. Next, we proposed a model to measure the customer's attraction towards each type of full-cut promotion (promotion driven demand) and segment the customers using K-means. We have also implemented a benchmark model, Conditional Gradient approach, from Jagabathula's paper to carry out a comparative study of the customer types and their corresponding demand. This comparative study shows that the customer choice probabilities in our proposed approach is 70% close to the choice probabilities predicted from the Conditional Gradient approach and therefore suggests that our proposed approach is viable in estimating demand and segmenting customers. The significance of this proposed method is that E-commerce companies and empirical researches can now segment customers based on their attraction to the promotion and to do so simply from product-level features in an aggregated transactional dataset instead of using customer demographics, which are sparse. Additionally, empirical researchers often do not have access to customer demographic information due to privacy issues.

Next, we applied machine learning models to investigate the relationship between features such as market price on the output, promotion driven demand. The boosted regression tree with Gradient Boosting Machine is shown to have the highest accuracy on our test sets and the features of discount amount and original purchase amount are the two most important features towards influencing a customer's promotion driven demand. We have also carried out a comparison of discrete choice models with the machine learning models and it is shown that traditional models for modelling consumer choices have a lower accuracy than data-driven machine learning models due to the model misspecifications incurred in traditional models. Optimization of full-cut promotions only enables E-commerce companies to find the optimal full-cut promotion in a specific context, based on the transactional dataset. The machine learning models enable E-commerce companies to understand how other features such as market price impact the customer's preferences. The results of machine learning provide the insight that E-commerce companies should drive the customers' attraction towards full-cut promotion by increasing the customer's original purchase amount and then consider changing the discount amount appropriately.

Lastly, we provided a framework for optimization of full-cut promotions, using an expectation maximization model, as an area of future research. The optimization of full-cut promotion enables companies to identify the threshold and discount amount that leads to optimal revenue from full-cut promotion. Other future areas of research involve relaxing certain assumptions such as varying the product assortment across time intervals and introducing stock-out scenarios.

11 REFERENCES

- [1] 法制晚报. (2010 年 12 月 13 日). 商家促销方式多 "满减" "满赠" 一字之差折扣不同. Retrieved from <http://www.chinanews.com/life/2010/12-13/2718668.shtml>
- [2] 买手业务后台. 满减促销, 洋码头买手促销. Retrieved from <https://seller.ymatou.com/sellerbook/seller-promotions/promotion-2.html>
- [3] Analysis 易观. (2016 年 7 月 28 日). 中国网络零售 B2C 市场年度综合报告 2016. Retrieved from <https://www.analysys.cn/article/analysis/detail/1000147/>
- [4] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 112, p. 315). New York: springer.
- [5] Quinlan, J. R. (1987). Simplifying decision trees. *International journal of man-machine studies*, 27(3), 221-234.
- [6] Singh, S., & Gupta, P. (2014). Comparative study ID3, cart and C4. 5 decision tree algorithm: a survey. *Int J Adv Inf Sci Technol [Internet]*, 27, 97-103.
- [7] Rokach, L., & Maimon, O. (2005). Decision trees. In *Data mining and knowledge discovery handbook* (pp. 165-192). Springer, Boston, MA.
- [8] Witten, I. H., Frank, E., & Mark, A. (2011). *Hall. Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*.
- [9] El Seddawy, A. B., Sultan, T., & Khedr, A. Applying Classification Technique using DID3 Algorithm to improve Decision Support System under Uncertain Situations.
- [10] Mazid, M. M., Ali, S., & Tickle, K. S. (2010, February). Improved C4. 5 algorithm for rule based classification. In *Proceedings of the 9th WSEAS international conference on Artificial intelligence, knowledge engineering and data bases* (pp. 296-301). World Scientific and Engineering Academy and Society (WSEAS).
- [11] Aslam, J. A., Popa, R. A., & Rivest, R. L. (2007). On Estimating the Size and Confidence of a Statistical Audit. *EVT*, 7, 8.
- [12] Breiman, L. (2017). *Classification and regression trees*. Routledge.
- [13] Khandelwal, P. (June, 2017). Which algorithm takes the crown: Light GBM vs XGBOOST?. Retrieved from <https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/>
- [14] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189-1232.
- [15] Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785-794). ACM.
- [16] Click, C., Malohlava, M., Candel, A., Roark, H., & Parmar, V. (2016). Gradient boosting machine with h2o. *H2O. ai*, 11, 12.

- [17] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems* (pp. 3146-3154).
- [18] Indyk, P., & Motwani, R. (1998, May). Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing* (pp. 604-613). ACM.
- [19] Svozil, D., Kvasnicka, V., & Pospichal, J. (1997). Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, 39(1), 43-62.
- [20] Blum, A. (1992). *Neural networks in C++: an object-oriented framework for building connectionist systems*. John Wiley & Sons, Inc..
- [21] Berry, M. J., & Linoff, G. S. (2004). *Data mining techniques: for marketing, sales, and customer relationship management*. John Wiley & Sons.
- [22] Boger, Z., & Guterman, H. (1997, October). Knowledge extraction from artificial neural network models. In *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation* (Vol. 4, pp. 3030-3035). IEEE.
- [23] Gupta, V. (October, 2017). Understanding Feedforward Neural Networks. Retrieved from <https://www.learnopencv.com/understanding-feedforward-neural-networks/>
- [24] Sharma, S. (September, 2017). Activation Functions in Neural Networks. Retrieved from <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>
- [25] Hanemann, W. M. (1984). Discrete/continuous models of consumer demand. *Econometrica: Journal of the Econometric Society*, 541-561.
- [26] Domencich, T. A., & McFadden, D. (1975). *Urban travel demand-a behavioral analysis* (No. Monograph).
- [27] Ben-Akiva, M. E., Lerman, S. R., & Lerman, S. R. (1985). *Discrete choice analysis: theory and application to travel demand* (Vol. 9). MIT press.
- [28] Boutsidis, C., Drineas, P., & Mahoney, M. W. (2009). Unsupervised feature selection for the k -means clustering problem. In *Advances in Neural Information Processing Systems* (pp. 153-161).
- [29] Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3), 241-254.
- [30] Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems* (pp. 849-856).
- [31] Estivill-Castro, V. (2002). Why so many clustering algorithms: a position paper. *SIGKDD explorations*, 4(1), 65-75.
- [32] Park, H. S., & Jun, C. H. (2009). A simple and fast algorithm for K-medoids clustering. *Expert systems with applications*, 36(2), 3336-3341.
- [33] Jagabathula, S., Subramanian, L., & Venkataraman, A. (2018). A Conditional Gradient Approach for Nonparametric Estimation of Mixing Distributions.
- [34] Song, J. S., & Xue, Z. (2007). Demand management and inventory control for substitutable products. Working paper.

- [35] Yan, Z., Cheng, C., Natarajan, K., Teo, C. (2018). Marginal Estimation + Price Optimization with Choice Models. Working paper.
- [36] Kaufman, L., & Rousseeuw, P. J. (2009). Finding groups in data: an introduction to cluster analysis (Vol. 344). John Wiley & Sons.
- [37] Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics, 20, 53-65.
- [38] Adventures in Machine Learning. (April, 2018). Vanishing gradient problems and Relus. Retrieved from <https://adventuresinmachinelearning.com/vanishing-gradient-problem-tensorflow/>
- [39] Berry, S., Levinsohn, J., & Pakes, A. (1995). Automobile prices in market equilibrium. Econometrica: Journal of the Econometric Society, 841-890.

12 APPENDIX

12.1 Multiple Regression Analysis

```
> (summary(vipdat[-1]))
Types_goods_id      Total_goods_cnt      Total_Market_price      Total_price      Average_Market_price      Average_price
Min.   : 1.000      Min.   : 1.000      Min.   : 99.0      Min.   : 40.0      Min.   : 99.0      Min.   : 40.00
1st Qu.: 1.000      1st Qu.: 1.000      1st Qu.: 159.0      1st Qu.: 69.0      1st Qu.:134.0      1st Qu.: 52.00
Median : 2.000      Median : 2.000      Median : 344.0      Median :143.0      Median :159.0      Median : 66.33
Mean   : 2.294      Mean   : 2.294      Mean   : 376.1      Mean   :156.2      Mean   :169.1      Mean   : 70.07
3rd Qu.: 3.000      3rd Qu.: 3.000      3rd Qu.: 517.0      3rd Qu.:211.0      3rd Qu.:185.7      3rd Qu.: 77.38
Max.   :10.000      Max.   :10.000      Max.   :1510.0      Max.   :620.0      Max.   :349.0      Max.   :168.00

Full_cut_goods_money_Pre_return      Total_cut_goods_money_Pre_return      Total_goods_amt_Pre_return_full_cut_only
Min.   : 0.00      Min.   : 0.00      Min.   : 19.90
1st Qu.: 21.78      1st Qu.: 21.78      1st Qu.: 47.40
Median : 62.11      Median : 62.11      Median : 95.63
Mean   : 66.50      Mean   : 66.53      Mean   : 89.70
3rd Qu.:100.00      3rd Qu.:100.00      3rd Qu.:114.00
Max.   :300.00      Max.   :300.00      Max.   :320.00

Total_goods_amt_Pre_return      Total_goods_amt_Post_return_full_cut_only      Total_goods_amt_Post_return
Min.   : 13.00      Min.   : 0.00      Min.   : 0.00
1st Qu.: 45.54      1st Qu.: 40.00      1st Qu.: 40.00
Median : 90.28      Median : 74.98      Median : 70.78
Mean   : 87.82      Mean   : 80.11      Mean   : 78.03
3rd Qu.:111.03      3rd Qu.:110.00      3rd Qu.:107.00
Max.   :308.00      Max.   :319.00      Max.   :308.00

Full_cut_goods_money_Post_return      Total_cut_goods_money_Post_return      sum_return_goods_cnt      Total_return_goods_amt
Min.   : 0.00      Min.   : 0.00      Min.   :0.0000      Min.   : 0.000
1st Qu.: 0.00      1st Qu.: 0.00      1st Qu.:0.0000      1st Qu.: 0.000
Median : 53.67      Median : 53.67      Median :0.0000      Median : 0.000
Mean   : 58.87      Mean   : 58.90      Mean   :0.2581      Mean   : 9.589
3rd Qu.:100.00      3rd Qu.:100.00      3rd Qu.:0.0000      3rd Qu.: 0.000
Max.   :300.00      Max.   :300.00      Max.   :9.0000      Max.   :238.081

Full_cut_amount      Price_Threshold
100-110:192      200-209: 89
0-10 :130      40-49 : 69
20-30 : 58      60-69 : 55
50-60 : 45      110-119: 51
30-40 : 41      210-219: 44
90-100 : 34      70-79 : 40
(other):116      (other):268
```

Table 12.1 Summary of features in the VIPshop dataset

Multiple Linear Regression (MLR) is a common approach for demand prediction as the model is easy to implement and interpret. MLR assumes a linear relationship between the dependent and independent variables, multivariate normality, no or little multicollinearity and homoscedasticity. More generally, regression assumes no or little multicollinearity and homoscedasticity. However, from the dataset, the input features violate the general assumption of multicollinearity and homoscedasticity and hence, regression is not used in the scope of our research. Figure 12.1 shows the correlation across the variables and there is a strong positive correlation between the variables. To ensure a low multicollinearity, the variables on market price and total number of goods purchased will need to be removed. However, these features are critical to our research as the modelling of the threshold and discount amount depends on market price. Further the full-cut promotion demand defined as the willingness of a customer to purchase due to the full-cut promotion. Therefore, the number of goods purchased will likely to have a strong influence on full-

cut promotions. Figure 12.2 shows the linearity relationship between each independent variable against the dependent variable. The initial Residual against Fitted Values plot shows a non-random scatterplot and therefore indicates the violation of the homoscedasticity assumption. Box-cox transformation is applied onto the training labels with the optimal $\alpha = 0.5$, based on the results from the function `box_cox_normplot`. Figure 12.3 shows the graph of Residuals against the fitted values after Box-Cox transformation, which still indicates a possible violation of the homoscedasticity assumption. With the violation of multicollinearity and homoscedasticity, regression models are not used in this research study.

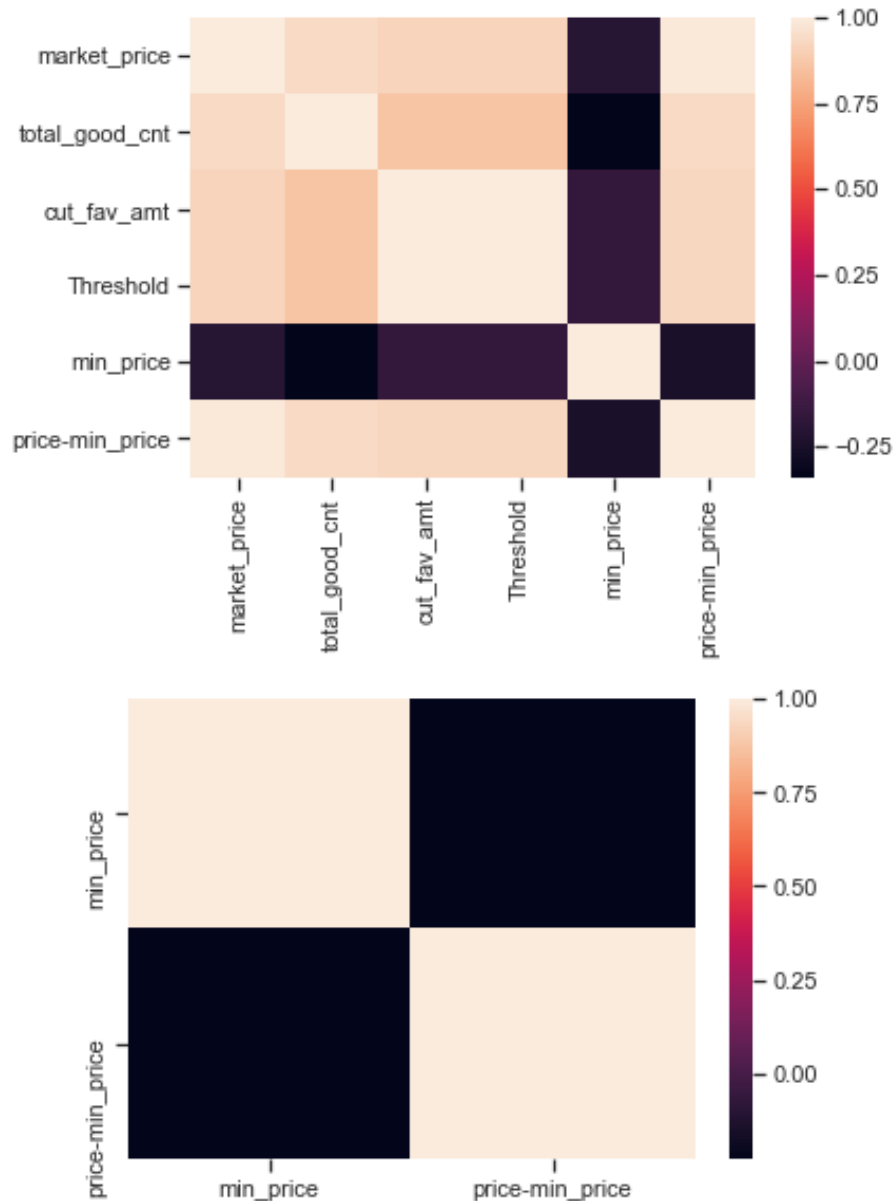


Figure 12.1. Correlation amongst independent variables

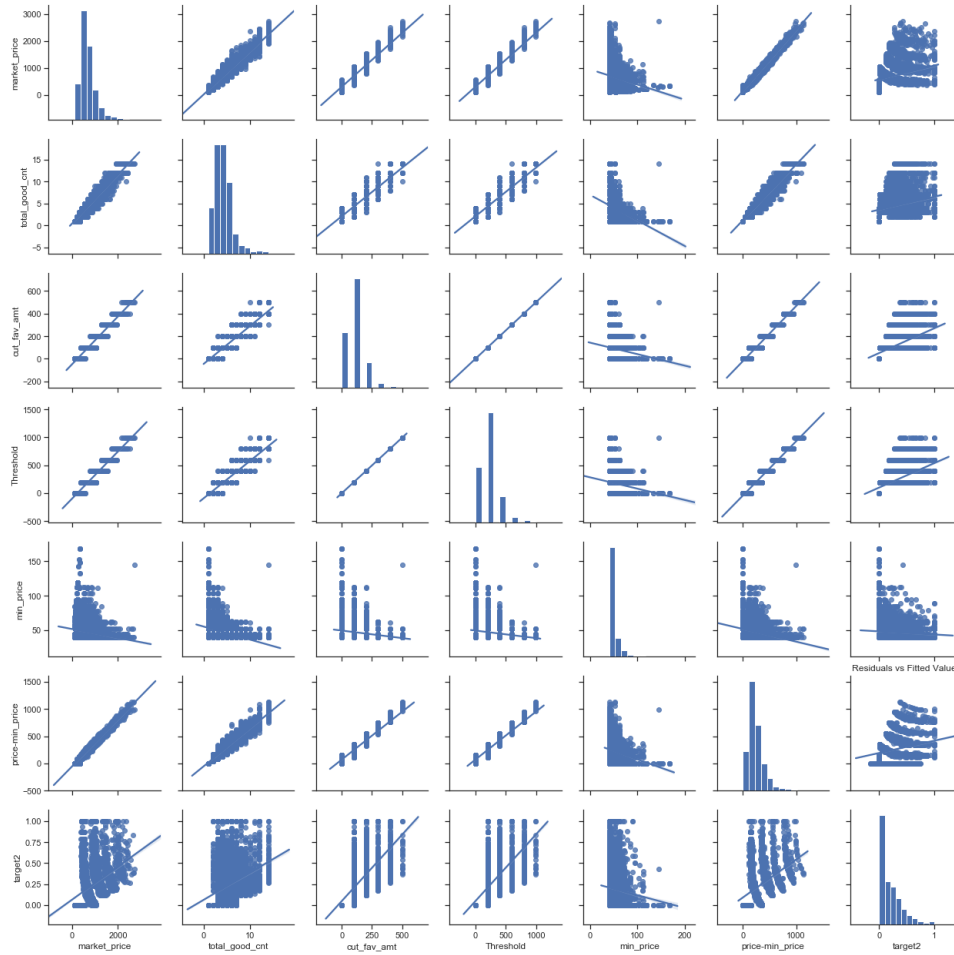


Figure 12.2 Scatterplot across dependent and independent variables

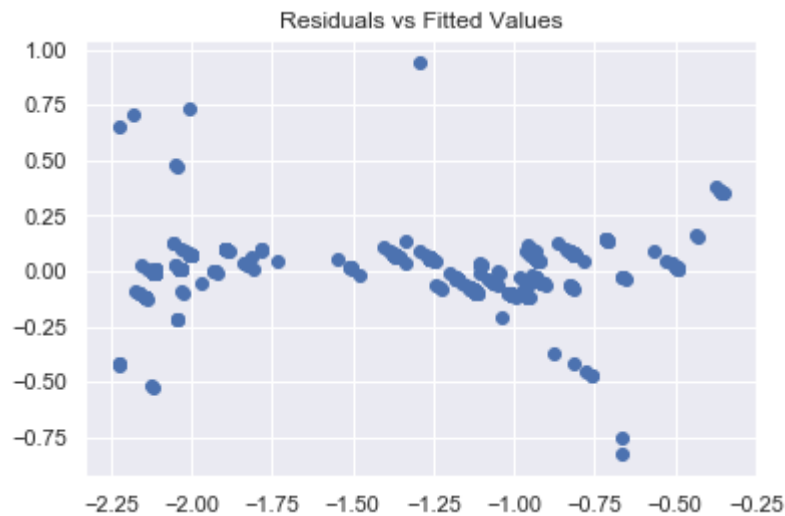


Figure 12.3 Residuals against Fitted Values after Box-cox transformation

12.2 Comparative study of Decision Tree algorithms

Classification and Regression Trees (CART) is a binary decision tree algorithm for processing both continuous and categorical features and/or target variables. Furthermore, decision tree model results provide clear information on the importance of significant factors for prediction or classification [9]. CART algorithm partitions the subsets recursively into two subsets so that successive subsets are more homogeneous than the prior subsets. The partitioning stops based on a homogeneity criterion or a stopping criterion based on time or number of iterations are satisfied. The same predictor variable may be used several times in the process of growing the decision tree. The most significant advantage is that CART can handle both numerical and categorical variables. Another advantage is that CART can output regression trees where their leaves predict a real number and instead of a class. This is especially useful for us to understand the extent of the full-cut promotion demand for the customers. The third advantage is unlike ID3, which does not have a pruning strategy, CART uses a cost-complexity pruning to identify the most significant features and eliminate the non-significant features. CART looks for splits that minimize the prediction squared error, that is the least-squared deviation. The prediction in each leaf is based on the weighted mean for node [9]. The disadvantage of CART is that it splits only by one feature.

ID3 determines the classification of the customers by testing the values of the properties. It builds a decision tree in a top-down fashion by searching through the entire dataset, starting from a set of customers and a specification of properties on the features. At each node of the tree, one feature is tested based on maximizing information gain and minimizing entropy, and the results are used to split the customers. This partitioning process is done recursively until the set in a given sub-tree is homogeneous, and the output becomes one leaf node in the tree. ID3 involves a greedy search and does not explore the possibility of alternate choices. The advantages of ID3 is that the prediction rules based on the splitting of the nodes are easily to interpret and ID3 outputs a short tree. ID3 can also build the tree quickly. The disadvantages of ID3 is that the data can be over-fitted, particularly when a small dataset is trained. Additionally, ID3 only splits by one feature and does not handle numeric attributes and missing values. Since our features involve numerical features, hence ID3 is not relevant for this scope of research.

C4.5 is a variant of ID3 that can handle both numerical and categorical features and outputs a decision tree by recursively partitioning the dataset. C4.5 grows the tree by considering the depth of the tree. C4.5 considers all the possible tests that can split the data and selects a test that gives the highest information gain. For each discrete attribute, one test is used to produce many outcomes based on the number of distinct values of the attribute. For each continuous attribute, the data is sorted, and the entropy gain is calculated based on binary splits on each distinct value of this feature. This process is repeated for all continuous attributes. The C4.5 algorithm allows pruning of the resulting decision trees and helps to reduce the test errors. The advantages of C4.5 is that it can handle both continuous and categorical features and C4.5 can also handle missing values in features by not using these values in calculation of gain and entropy values. C4.5 also has an error-based pruning strategy that goes back to remove any branches in the tree. However, the disadvantage of C4.5 is that it may construct empty branches which leads to multiple nodes with zero values [10]. These nodes and empty branches are not relevant in segmenting the

customers and leads to a more complex tree. C4.5 may also lead to over-fitting due to the noises or anomalies in the dataset.

Characteristic(→) Algorithm(↓)	Splitting Criteria	Attribute type	Missing values	Pruning Strategy	Outlier Detection
ID3	Information Gain	Handles only Categorical value	Do not handle missing values	No pruning is done	Susceptible on outliers
CART	Twoing Criteria	Handles both Categorical and Numeric value	Handle missing values	Cost- Complexity pruning is used	Can handle Outliers
C4.5	Gain Ratio	Handles both Categorical and Numeric value	Handle missing values	Error Based pruning is used	Susceptible on outliers

Figure 12.4 Basic characteristics of decision tree algorithms [6]

12.3 Discussion of bagging and boosting methods

Bagging is a technique involving bootstrapping and aggregating to improve the variance and accuracy of the output in machine learning algorithms and it is particularly useful for decision tree methods. Bagging begins by carrying out bootstrapping, which is to generate m datasets by sampling the entire original dataset uniformly with replacement. This means that some observations may be repeated in each bootstrapped dataset and for a large number of observations the unique number of observations in each bootstrapped dataset is expected to $(1 - 1/e) \approx 63.2\%$ [11]. The decision tree algorithm is then applied onto each bootstrapped dataset to obtain m number of decision trees. The final output is obtained by aggregating the results over the m number of decision trees, that is

$$\hat{f}_{bag}(x) = \frac{1}{m} \sum_{b=1}^m \hat{f}^{*b}(x)$$

Where x represents the predictors and $\hat{f}^{*b}(x)$ represents the output for each decision tree obtained from the b th bootstrap dataset.

The accuracy of the outputs is based on the mean squared error (for quantitative outputs) or misclassification rate (for categorical outputs) using the out-of-bagged samples (which are samples that are not sampled into the bootstrapped dataset. The feature importance is calculated based on the mean decrease impurity, which is defined as the total decrease in node impurity (weighted by the probability of reaching that node (which is approximated by the proportion of samples reaching that node)) averaged over all trees obtained from the bagged samples [12].

Boosting involves creating weak classifiers iteratively and adding them to a final strong classifier in the form of a linear regression. At each iteration, the weak classifiers are typically weighted in a manner that is related to the weak classifier's accuracy. After a weak classifier is added, the weights in the dataset are readjusted, so that observations in the dataset that are previously misclassified will be assigned a higher weight while observations that are previously classified correctly will lose weight. This allows the future weak classifiers to focus more correctly classifying the previously wrongly classified observation. There are two main types of boosting algorithms: Adaptive Boosting (Adaboost) and Gradient Boosting. Both of Adaboost and Gradient Boosting builds the weak classifiers iteratively by increasing the weights of misclassified observations sequentially. However, the main variation between is their method of weighting and how they create the weak classifiers.

In AdaBoost, the algorithm is as follows:

1. All n observations in the dataset are assigned with equal weights, $\frac{1}{n}$.
2. A model, $h_t(x)$, is built on a subset of data.
3. Using this model, predictions are made on the entire dataset. Errors are calculated by comparing the predictions and actual values. We calculate also the weighted sum error for misclassified points, denoted as $\varepsilon_t = \sum_{i=1}^n w_{i,t} \cdot \mathbb{1}_{h_t(x_i) \neq y_i}$.
4. This model is added to build the final classifier

$$F_t(x) = F_{t-1}(x) + \alpha_t h_t(x), \text{ where } \alpha_t = \frac{1}{2} \ln \left(\frac{1-\varepsilon_t}{\varepsilon_t} \right)$$

5. While creating the next model, $h_{t+1}(x)$, the weights are updated with the following error function, $Error(h_t(x), y, i) = e^{-y_i f(x_i)}$

$$w_{i,t+1} = w_{i,t} * e^{-y_i \alpha_t h_t(x_i)} \text{ for all } i$$

6. Renormalize the weights such that $\sum_i w_{i,t+1} = 1$
7. This shows that the weights are determined using an exponential error function. The higher the error the more is the weight assigned to the observation.

8. This process is repeated until the error function does not change, or the maximum limit of the number of estimators is reached.

For Gradient Boosting, unlike AdaBoost, fits the new classifier to the residual errors made by the previous classifier.

In Gradient Boosting, the algorithm is as follows:

1. An initial model is built on a subset of data, $h_t(x) = \operatorname{argmin}_{\gamma} \sum_{i=1}^n L(y_i, \gamma)$, where $t = 0$, $L(y_i, \gamma)$ is a loss function such as squared loss $\frac{1}{2}(y - F(x))^2$.
2. Using this model, predictions are made on the whole dataset. Errors are calculated by comparing the predictions and actual values. Instead of calculating the weighted sum error of misclassified points, we calculate the residuals:

$$r_{it} = - \left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)} \right]_{F(x)=h_t(x)} \quad \text{for } i = 1, \dots, n$$

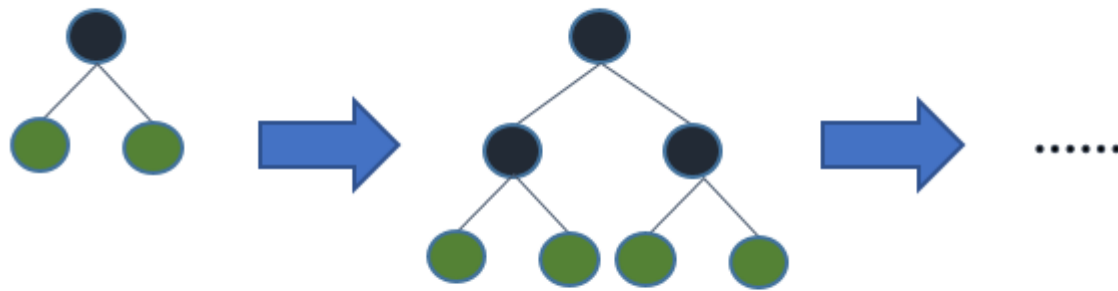
3. A new classifier $h_{t+1}(x)$ is created by fitting to the errors (residuals). This new classifier is added to build on the final classifier:

$$F_t(x) = F_{t-1}(x) + \alpha_t h_t(x),$$

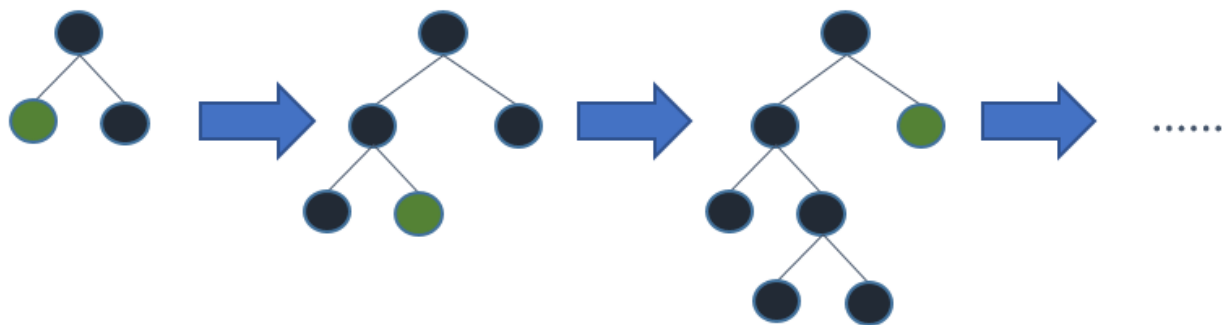
$$\text{where } \alpha_t = \operatorname{argmin}_{\gamma} \sum_{i=1}^n L(y_i, F_{t-1}(x_i) + \gamma h_t(x_i))$$

4. The predictions made by this new model are combined with the predictions of the previous. New errors are calculated using this predicted value and actual value.
5. This process is repeated until the error function does not change, or the maximum limit of the number of estimators is reached.

Extreme Gradient Boosting (XGBoost) is a variant implementation of the Gradient Boosting. This algorithm has high predictive power, a faster running time compared to traditional gradient boosting, and includes a regularization technique to reduce overfitting and improve overall performance. XGBoost also implements cross-validation at each iteration of the boosting process. Unlike XGBoost, whereby the algorithm splits the tree depth wise, Light Gradient Boosting Machine (Light GBM) splits the tree leaf wise as shown in the figure 12.5 below. Light GBM has a faster running time in training the models as it assigns the continuous feature values into discrete bins, hence using a lower memory usage and making it ideal for training large datasets.



Level-wise tree growth



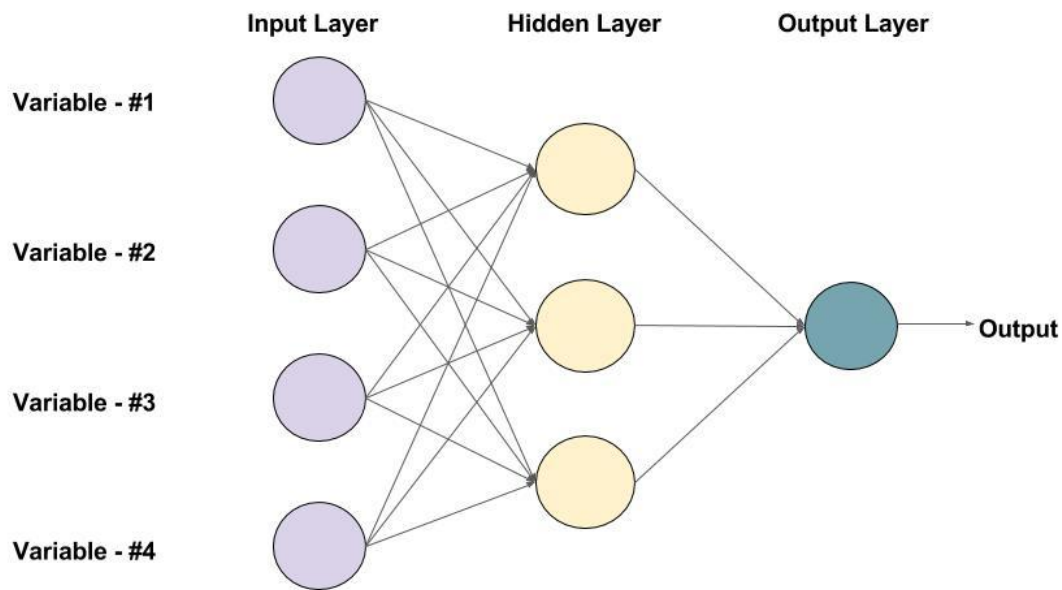
Leaf-wise tree growth

Figure 12.5 Diagram of level (depth)-wise and leaf-wise decision [13]

H2O is a community of physicists, mathematicians, and computer scientists who are passionate in exploring machine learning algorithms and has now become a platform for introducing fast, scalable, open-source machine learning algorithms. Some of these machine learning algorithms are: generalized linear modeling, naive bayes, principal components analysis, k-means clustering, distributed random forest, gradient boosting [16]. H2O GBM is a variant implementation of the Gradient Boosting Machine in R as it has additional functionalities such as supervised learning for regression and classification tasks, a faster Java implementation of the algorithms and a grid search for hyperparameter optimization and model selection. For the discussion of other additional functionalities from H2O, refer to the booklet section 4.1 [16].

12.4 Discussion of Feedforward Neural Networks

Artificial neural networks (ANN) are networks of neurons operating communicating with one another. The design of ANNs was motivated by the structure of a human brain, but the processing elements and the architectures used in ANN have gone far from their biological inspiration. A Multi-Layer Feedforward neural network (FNN) which consists of neurons, that are ordered into layers as shown in figure 12.6. The first layer is called the input layer, the last layer is called the output layer, and the layers between are hidden layers and the nodes in the hidden layers are known as neurons. Each of the inputs, x_i for $i = 1, \dots, n$ is passed into every neuron with weights, $w_{i,t}$ for input i and neuron t in the hidden layer and each neuron contains an activation function $\sigma^{(l)}$ where l represents the l^{th} layer in the neural network. There are a few popular types of activation functions (sigmoid, hyperbolic tangent, ReLu) as shown in figure 12.7. In this research, we have set our activation function as ReLu. One of the advantages of ReLu is that the function and its derivative both are monotonic. In particular, its derivative is constant and this makes it to be useful a deep neural network where the gradient may vanish or increases exponentially, since the gradient in a layer gets multiplied by the gradient in the successive gradient during the backpropagation process. FNN uses a backpropagation algorithm to calculate the weights for the variables or outputs from the prior neuron. For an explanation of the backpropagation algorithm, refer to the research paper in [19].



An example of a Feed-forward Neural Network with one hidden layer (with 3 neurons)

Figure 12.6 Diagram of feedforward neural network [23]







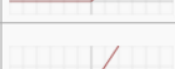


Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parametric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

Figure 12.7 Diagram of activation functions [24]

12.5 Clustering and Silhouette Score

Silhouette score measures how well an element belongs to its cluster, and the mean silhouette score measures the average strength across all the clusters' membership. Hence, silhouette score validates the consistency within clusters of data and its silhouette plot provides a graphical representation of how well each observation has been classified [37]. After clustering the full-cut promotion demand output by k-means (with $k = 3$ clusters), for each observation $i \in \text{cluster } C_i$, we compute,

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j)$$

where $a(i)$ is the average distance between observation i and all the other observations in the same cluster. $d(i, j)$ is the distance between observation i and j . The resulting summation is divided by $|C_i| - 1$ and not $|C_i|$ as we do not include the distance $d(i, i)$ in the summation. Next, we compute the average dissimilarity, $b(i)$, of the observation i with C_i by computing the smallest average distance of observation i to all the observations in C_i .

$$b(i) = \min_{i \neq j} \frac{1}{|C_j|} \sum_{j \in C_j} d(i, j)$$

Silhouette score for observation i is defined as:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \text{ if } |C_i| > 1$$

$$s(i) = 0 \text{ if } |C_i| = 1$$

Hence,

$$s(i) = \begin{cases} 1 - \frac{a(i)}{b(i)}, & \text{if } a(i) < b(i) \\ 0, & \text{if } a(i) = b(i) \\ \frac{b(i)}{a(i)} - 1, & \text{if } a(i) > b(i) \end{cases}$$

$$\text{mean silhouette score} = \sum_i s(i)$$

The silhouette score ranges from -1 to $+1$. For $s(i)$ to be close to 1 we require $a(i) \ll b(i)$. As $a(i)$ measures dissimilarity of observation i to its cluster, a small value of $a(i)$ means it is well matched. A large $b(i)$ implies that observation i is poorly matched to its nearest neighboring cluster. Thus a $s(i)$ close to one means that the data is appropriately clustered. If $s(i)$ is close to negative one, by the same logic, observation i would be more appropriate to be clustered to its nearest neighboring cluster. When $s(i)$ is near zero, this means that the observation is on the border of two clusters. A high value of silhouette score between 0.7 to 1.0 indicates a strong clustering structure has been observed [36]. A moderate value of silhouette score between 0.51 to 0.70 indicates a reasonable clustering structure has been observed [36]. A low value of silhouette score of less than 0.50 indicates that the clustering structure is weak and could be artificial [36]. The silhouette score can be computed with different measures of distance, such as the Euclidean distance or Manhattan distance. In this research, we have chosen Euclidean distance.

12.6 Customer type estimation and Comparative study

Figure 12.8 is a plot for the weightage of each good ID, such that good ID $\in product\ universe\ [n]$, $n = 95$. The weightage is computed as $\delta_j = N_j / \sum_{j=1}^{95} N_j$ as discussed in section 6.2. Figure 12.9 is a plot for the percentage absolute error of each good ID, such that good ID $\in product\ universe\ [n]$, $n = 95$. This error is computed as $percentdiff_j = |CP_j - g_j| / g_j$ as discussed in section 6.2. The average percentage difference in market share between the propose technique CP and the technique proposed in Jagabathula's paper, g is denoted as $\sum_j \delta_j * percentdiff_j = 29.6\%$. From the figures below, we can clearly observe that the good ID which has a lower $percentdiff_j$ will have a higher corresponding weightage δ_j , and vice versa. For example, "good ID 2" and "good ID 34" have high weightages, representing 2.8% and 5.5% of the total transactions and they both have a very low percentage difference in choice probabilities of 2.9% and 7.8%. Conversely, "good ID 51" and "good ID 30" have a low weightage, representing only 0.044% and 0.12% of the total transactions and they both have a very high percentage difference in choice probability of 98% and 99%. This shows that our proposed approach can identify customer types and predict the choice probabilities more accurately for good ID with a higher number of transactions (weightage) due to our data-driven approach. Hence, we believe that our proposed approach can yield an even lower error based on the benchmark against Jagabathula's approach with a larger dataset that includes more transactions for every good ID. The low average percentage difference of 29.6% also shows that our proposed approach is a viable one for predicting promotion driven demand and identifying customer types.

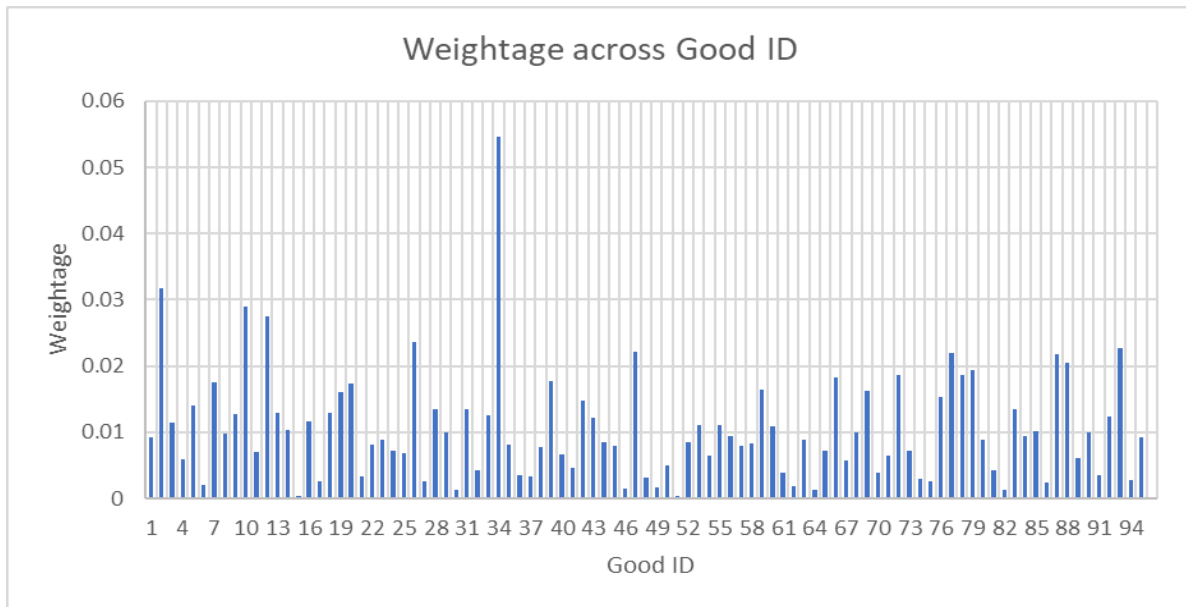


Figure 12.8 Weightage across good ID

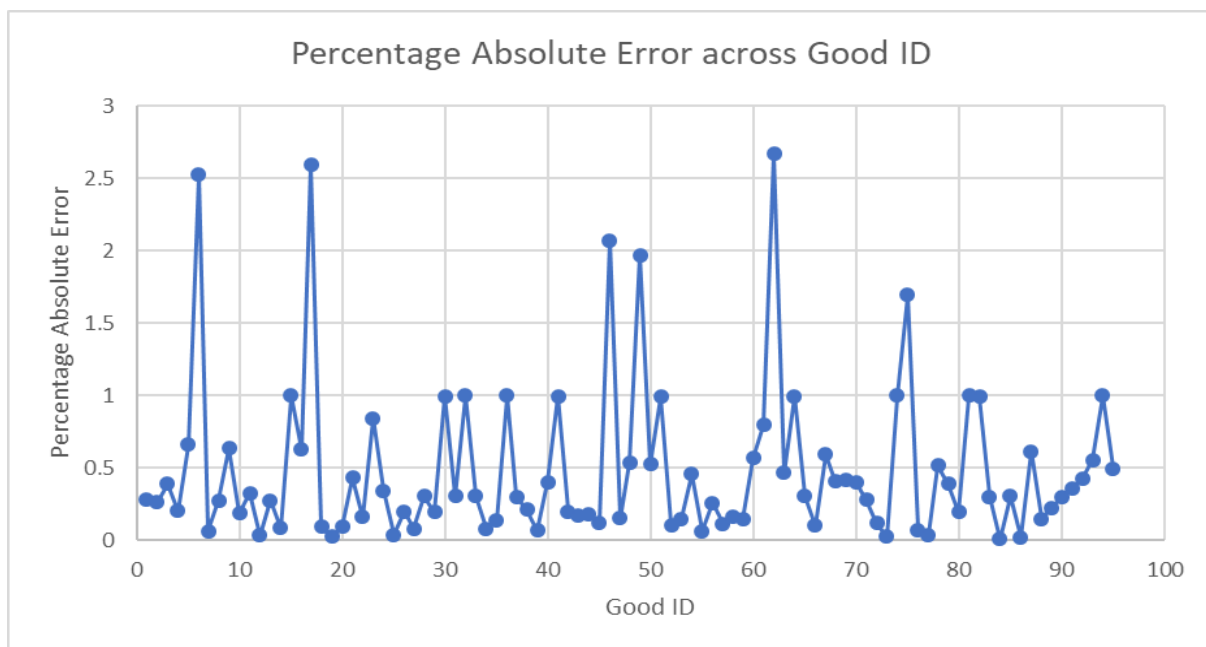


Figure 12.9 Percentage absolute error across good ID