8-2021

Data Science and the Ice-Cream Vendor Problem

Makafui Azasoo
East Tennessee State University

Data Science and the Ice-Cream Vendor Problem

A thesis

presented to

the faculty of the Department of Mathematics

East Tennessee State University

In partial fulfillment

of the requirements for the degree

Master of Science in Mathematical Sciences

---

by

Makafui Ama Azasoo

August 2021

Jeff Randall Knisley, Ph.D., Chair

Michelle Joyner, Ph.D.

Anant Godbole, Ph.D.

JeanMarie L Hendrickson, Ph.D.

Keywords: newsvendor models, sample average approximation, ice-cream vendor problem, discrete event simulation, L^1-regularization

ABSTRACT

Data Science and the Ice-Cream Vendor Problem

by

Makafui Ama Azasoo

Newsvendor problems in Operations Research predict the optimal inventory levels necessary to meet uncertain demands. We examine an extended version of a single period multi-product newsvendor problem known as the ice cream vendor problem. In this problem, there are two products – ice cream and hot chocolate – which may be substituted during medium temperatures. This problem is a data-driven extension of the newsvendor problem with no assumption of a specific demand distribution. Using Discrete Event Simulation, we simulate a real-world scenario of the problem via a demand whose expected value is a function of temperature. Sample average approximation is used to transform the stochastic newsvendor program into a feature-driven linear program based on some exogenous factors. The resulting problem is a multi-product newsvendor linear program with L^1-regularization. The solution to this problem yields the expected cost to the ice cream vendor as well as the optimal order quantities for both products.

ACKNOWLEDGMENTS

I would like to express my utmost gratitude to my advisor Dr. Jeff Knisley. I am most thankful for his support and continuous patience with me throughout this. I also grateful for his guidance which which helped me grow and believe in myself once again. I would also like to thank Dr. Michelle Joyner, Dr. Anant Godbole and Dr. JeanMarie L Hendrickson for been apart of my committee.

Additionally, I would like to thank my dad and Emma for believing in me and been there for me, always.

TABLE OF CONTENTS

LIST OF TABLES

1 INTRODUCTION

Inventory control management is integral for any business. Effective inventory control creates a balance between the outflow and inflow of inventory in order to satisfy customer demands. The newsvendor model is an essential model in inventory control management typically used for products available for a single selling period [45].

The newsvendor problem is the dilemma of a newspaper vendor who orders newspapers each morning in order to meet the demand during the day. In this case, it is not possible to order extra units during the day to fulfill any unanticipated demand. Papers ordered for a particular day are only useful for that day; leftover papers cannot be sold at a later date. If the vendor knows the demand at the time he orders, he would order that quantity. However, demand is random and unknown at the time he orders. The model assumes that, if demand during the day is less than the ordered quantity, he will incur a cost for the excess inventory. If the demand exceeds the ordered quantity, then some unfulfilled demand is lost and he forgoes some profit. In other scenarios, any unsold inventory is discounted and sold at a salvage value or disposed of.

The newsvendor problem hence seeks to find the optimal number of copies of newspapers to buy in order to maximize profits or minimize loss [28]. The newsvendor model is a well-known problem formulation in Operations Management for making inventory decisions under uncertain demand. [39].

Demand estimates are speculative and fluctuate based on different factors [64]. These uncertainties may be caused by some factors such as customer preference, sea-

sonal fluctuations, price of substitute products etc. However, certain industries such as transportation, pharmaceuticals, etc. are more sensitive to fluctuating demand than others. In the problem of the vendor making a decision as to the optimum number of newspapers to purchase, demand is likely to be higher if the headlines are about a political scandal than if about plan to expand a local library. For business decision making, analysis and eventual decision is based on knowing the demand characteristics, accounting for more factors. Consideration of several specific extensions such as price can also help in determining optimal order quantity in a varied range of problems.

In its adoption to real world scenarios, the newsvendor problem captures the fundamentals of diverse business contexts [2, 53, 54] especially in inventory management [13]. In these industries, the newsvendor problem is a one-time business decision with no opportunity to replenish [10]. One of the application areas is the inventory management of seasonal products such as electronics, fashion items, Christmas trees etc. [17]. Products with a limited shelf-life such as produce, dairy products also face similar constraints. It is also widely used in the health industry (drugs with short expiration dates), manufacturing and even liquid markets [18]. The newsvendor model is also used to manage capacity and estimate booking requests in service sectors like airlines, theaters and hotels. For many of these products, ordering decisions are often made before any demand is known. There is also a finite selling season beyond which inventory is essentially obsolete [31]. Additionally, the processing length and transportation lead-times, are often longer than the market lifespan of these products due to restrictions such as budgeting, minimum output, and capacity [14].

Table 1: Notation

Notation	*Definition*
x	order quantity
D	demand variable (random variable)
c	cost per unit
c_o	uderage cost per unit
c_u	overage cost per unit
C	cost function
E_D	expected demand
$f(D)$	density function of D
$F(D)$	cumulative distribution function of D

1.1 Classical Single Product Newsvendor Problem

In the generic setting of all newsvendor-type problems, stocking decisions are based on finding an optimal between expected costs due to excess inventory or expected cost incurred due to stockouts in a single selling period. In [44], "it occurs whenever the amount needed of a given resource is random, a decision must be made regarding the amount of the resource to have available prior to finding out how much is needed, and the economic consequences of having too much and too little are known". The cost incurred per unit of unused inventory is known as overage cost. Costs incurred per unit of unmet demand or lost sales is known as underage cost [20, 39].

Consider a certain product with a retail unit cost of $\$c$ sold at a unit price of $\$p$, and that the product is discounted or dispose of at a unit salvage price $\$s$ after the selling period. Assume also that, the vendor orders quantity (x) to satisfy demand D, at the start of the selling period. If demand D, is less than x, ($D < x$), then, underage cost, c_u is incurred. If demand D is greater than x, ($D > x$), then an

overage cost c_o is incurred to the vendor.

In view of the notation in Table 1 and [57], the goal of the newsvendor problem is to find the optimal order quantity that minimizes the expected value of

$$C(x, D) = \begin{cases} c_o(x - D) & \text{if } D < x \\ c_u(D - x) & \text{if } D > x \end{cases}$$

We define $(y)^+ = \max(y, 0)$. This allows us to write the total cost as

$$C(x, D) = c_o(x - D)^+ + c_u(D - x)^+$$

If D is known before ordering x, then, the optimal order quantity(x^*) which minimizes the expected value of cost will occur at $x^* = D$.

At an optimal order quantity x^*, minimization of the expected cost is equivalent to the maximization of the expected profit [26, 39]. Throughout this thesis, the cost minimization is used as the objective function of an inventory newsvendor model.

Demand estimations are inherently difficult. The accuracy of the decision may rely on alternate assumptions which are appropriate and possible in different settings. However, distribution assumptions are also speculative and often erroneous [28]. Actual demand distribution and it corresponding parameters are unknown and may even change over time [27, 46]. With new developed products, it becomes increasingly difficult to define probabilities since there may be no available data for demand prediction [36]. In certain cases, demand is assumed to be known or regular (deterministic). Here, trends in historical sale data may be used to predict demand. If demand is assumed to be unknown, then the demand is assumed to be a random

variable in each future period drawn independently from some known probability distribution.

A typical method for resolving demand uncertainty is the use of stochastic models which assume a specific distribution for the demand. Under stochastic demand, the goal of the newsvendor problem is to determine the minimum order quantity that minimizes the expected cost. In this setting, the expected cost is estimated since it is a variable for which the value is modeled by a distribution. The Stochastic objective is defined as

$$\min_x E_D(C(x, D)) = E_D[(c_o(x - D)^+ + c_u(D - x)^+)]$$

If the random demand is stochastic, then it has a cumulative distribution $F(x) = \Pr(D < x)$ and density

$$f(x) = \frac{dF(x)}{d(x)}$$

If demand D is discrete and a probability distribution $\Pr(D)$, the expected cost is given by

$$E(C(x, D)) = \sum_{D=0}^{\infty} C(x, D)\Pr(D)$$

which can equivalently be written as

$$\min E_D(C(x, D)) = \sum_{D=0}^{\infty} (c_o(x - D)^+ + c_u(D - x)^+)\Pr(D)$$

Because discrete random demand can be difficult to work with, it is frequently approximated as a continuous random variable [11].

If D is continuous, then the expected cost becomes

$$E(C(x, D)) = \int_{D=-\infty}^{\infty} C(x, D)f(D)dD$$

If $D \geq 0$, then $\Pr(D) = 0$. The expected cost function E(C(x,D)), is then equivalent to

$$\min E_D(C(x,D)) = \int_{D=0}^{x} (x-D)^+ f(D)dD + \int_{D=x}^{\infty} (D-x)^+ f(D)dD$$

Theorem 1.1 *If $f^{'}(x^*) = 0$ for a convex function, then x^* is a global minimum of f, where $f^{'}(x)$ is the gradient function or total derivative.*

If $E[\nabla C(x^*, D)] = 0$ and $C(x,D) = c_o(x-D) + c_u(D-x)$, then,

$$\nabla C(x,D) = \begin{cases} c_o & \text{if } D < x^* \\ \\ -c_u & \text{if } D > x^* \end{cases}$$

Note that $F^{'}(x) = f(x)$ and by Leibniz's Rule, $T^{'}(x) = xf(x)$. The first derivative of the expected cost with respect to x, is

$$\begin{aligned} E[\nabla C(x^*, D)] &= c_o Pr(D < x^*) - c_u Pr(D > x^*) \\ &= c_o Pr(D < x^*) - c_u(1 - Pr(D < x^*)) \\ &= (c_o + c_u) Pr(D < x^*) - c_u \end{aligned}$$

The argmin x^* (optimal order quantity) satisfies

$$Pr(D < x^*) = \frac{c_u}{c_o + c_u}$$

which is equivalent to

$$F(x^*) = \frac{c_u}{c_o + c_u}$$

The ratio $c_u/(c_o + c_u)$ is the Critical Fractile (C.F). The Critical Fractile is the probability that the order quantity satisfies demand in the likelihood of underage or overage.

1.2 Extensions of the Newsvendor Problems

Traditionally, the newsvendor only dealt with inventory decision making. Today, however, newsvendor models have been improved and are effective for controlling inventories by assessing trade-offs among cost components [50]. These new models are durable, simple and easily adapt to specific industries. In 1963, Hadley and Whitlin introduced the newsvendor model [21] as applied to solving inventory problems. Their model in itself was an extension of the traditional newsvendor problem which included multiple products with an assumed demand distribution and a budget constraint. There exists a large literature of research particularly in the extension of the model to find solutions for different products in a single-sale period [23, 29, 30, 43]. Some of these extensions concentrated on creating a computationally efficient algorithm to find optimal solutions to multi-product newsvendor problems [16]. More recent studies have used Lagrangian relaxation approach with constraints [58]. In the growing interest and application of multi-product newsvendor problems with constraints, conditions such as the Karush-Kuhn-Tucker have also been proposed and used [1].

Multi-product newsvendor models form a class of models in operation research used to build stochastic linear programs [7]. The multi-product newsvendor problem is framed as a vendor who at the start of a sale period must decide the order quantities for different products x_i ($i = 1,2,...n$) to meet uncertain customer demand [65]. Similar to the single-product, any unsold product(s) after the selling period is discounted at a unit salvage price or disposed of. Suppose we have the sale of n items, thus, $x_1, x_2, ..., x_n$ order quantities to satisfy $D_1, D_2, ..., D_n$ demands, c_i is the cost per unit, c_{ui} is the underage cost per unit of the i^{th} item ($x_i < D_i$) and c_{oi} is overage cost per

unit of the i^{th} item $(x_i > D_i)$.

Then the expected cost function is

$$E(C(x_i, D)) = \sum_{i=1}^{n} (c_{ui}(D_i - x_i)^+ + c_{oi}(x_i - D_i)^+)$$

Suppose that we have a constraint such as capacity ($\mathbf{C}$), that is, the vendor cannot order so many there will be nowhere to store them, then,

$$x_1 + x_2 + ... + x_n \leq \mathbf{C}$$

We may also have a budget constraint $\mathbf{B}$ (only a limited amount of money to spend),

$$c_i x_1 + ... + c_n x_n \leq \mathbf{B}$$

Hence for a deterministic demand, the cost minimization is

$$\min_x \sum_{i=1}^{n} (c_{ui}(D_i - x_i)^+ + c_{oi}(x_i - D_i)^+)$$

$$\text{subject to :} \quad x_1 + x_2 + ... + x_n \leq \mathbf{C}$$

$$c_i x_1 + ... + c_n x_n \leq \mathbf{B}$$

For the stochastic demand (here demand is random); the problem is

$$\min_x E(\sum_{i=1}^{n} (c_{ui}(D_i - x_i)^+ + c_{oi}(x_i - D_i)^+)$$

$$\text{subject to :} \quad x_1 + x_2 + ... + x_n \leq \mathbf{C}$$

$$c_i x_1 + ... + c_n x_n \leq \mathbf{B}$$

For single product newsvendor problems, constraints do not play a major role since there is only one order quantity. Because the objective function and the constraints

are both linear, multi-product newsvendor problems are formulated as linear programs. If the demand is known beforehand, then the minimum cost happens when the sum of the order quantities for each specific period equals the demands for those periods respectively.

In certain cases, unsold items are sometimes kept and resold at a later date or in subsequent periods. The amount carried forward to the future period is however random. The vendor's goal is to find an order quantity sequence that minimizes the expected total cost while maintaining appropriate inventory for the period. Extra inventory maybe held to protect against backorder or stockout (safety stock). One of the most important constraints in supply chains is

$$x_i - D_i \geq S_i$$

where S_i is the safety stock [33]. Multi-period newsvendor problems are dynamic programs and can be broken down into sequences of simpler problems and solved independently for the entire sale-period [4].

1.3 Stochastic Demand and Connection to Supply Chain

A supply chain is made up of networks that consists of processing raw of materials, manufacturing, transporting of the processed goods, distribution to centres and retail outlets and eventually, selling to customers [35]. In a supply chain, decisions concerning quantities to order are made before sale. Inventory decision problems are often posed as newsvendor problems. The newsvendor model is an integral and foundational concept of supply chain and inventory control management [3]. In a supply a chain, history of past demands may not necessarily be enough to accurately estimate

the exact future demand of a given product. This uncertainty along with supply, cost etc. are some of the challenges encountered in the supply chain management.

1.4 A New Extension - The Ice-Cream Vendor Problem

The ice cream vendor problem is an extension of the conventional single product newsvendor problem but with a related set of factors that allow a better prediction of demand. In this problem, an ice cream vendor must order ice cream and hot chocolate to fulfil demand D (during the day). Ice cream costs $\$c_1$ per serving while hot chocolate costs $\$c_2$ per serving. Any ice cream unsold at the end of the day is salvaged for $\$s_1$ per serving and hot chocolate salvaged for $\$s_2$ per serving. If the demand D exceeds the amount of ice cream or hot chocolate, then the vendor issues a 'coupon' which costs additional $\$b_1$ and $\$b_2$ dollars per item respectively.

Intuitively, the vendor would use the weather to predict what the demand might be that day. There would be greater demand for ice-cream when the temperature is high than when it is low. On colder days, demand for hot chocolate will be high, the vendor is likely to sell more hot chocolate than ice cream. The demand D, a random variable, is dependent on temperature and if the temperature is not too high, not too low, then a customer may substitute one product for the other. Because the demand distribution depends temperature, the ice cream vendor cannot estimate any order quantity independent of a weather forecast. What would be the vendor's order quantities for the number of servings of ice cream and hot chocolate at a minimized cost?

In this problem, D has an exogenous factor (temperature). Thus, the random

variable D is determined by changes or influences outside the model and which is also unexplained by the model itself. In the ice cream vendor problem, demand may have no known distribution. Nonetheless, certain assumptions can be made on the temperature categories to reduce the ice cream vendor's problem to three (3) newsvendor problems:

1. Demand has an exogenous factor (temperature) (Simulate via a distribution whose mean is a function of temperature)

2. Reduces to 3 Newsvendor Problems for temperature categories:

    - Hot (High Temperatures)
    - Cold (Low Temperatures)
    - Medium (High or low Temperatures)

3. Implements substitution of one product for another (here, the substitution is parameter(temperature) dependent).

    - high temperature - ice cream (no substitution)
    - medium temperature - substitution (either ice-cream or hot chocolate)
    - low temperature - hot chocolate (no substitution)

Determining optimal inventory levels to meet uncertain demand of substitutable items is a well-known topic in Operations Research. Substitution occurs when at least two (2) or more products can be easily replaced for another and in most instances, for the same purposes. In [14], product substitution can be defined as the use of a product

to fulfill the demand for another product within a particular product category. The ice cream vendor problem is a data-driven extension of the conventional newsvendor problem that do not require the demand to have an assumed distribution.

In this thesis, our objective is to determine the minimum expected cost to the ice cream vendor given demand with some features. We use simulations to generate data as a means of insuring sufficient data as well as a means of comparison between theoretical and empirical solutions. The Simpy package is used to simulate as a discrete event simulation of a real-world scenario of an ice cream vendor problem via a demand whose expected value is a function of temperature. Using sample average approximation, the vendor's problem is transformed into a feature-driven linear program dependent on exogenous factors such as temperature and rainfall.

2 STOCHASTIC LINEAR PROGRAMMING

2.1 Probability and Randomness

We frequently hear and use phrases like "it is very likely that it will snow now or later", "he will probably not make it to the meeting next week". These type of expressions are based on the idea of randomness, which is studied using probability theory.A traditional view of probability is based on the idea of equally likely outcomes [15]. Probability enables us to quantify or measure uncertainties in the likely outcome of an event [25]. The mathematical theory of probability provides fundamental tools for developing and assessing mathematical models for random occurrences. In general, probability models assume a specified distribution. In this section, we introduce some concepts which serve as foundation for subsequent section to solve the ice-cream vendor problem.

Definition 2.1 *Consider a sample space S. Let A be a set of events. Assuming P is a real-valued function defined on A, then P is a probability measure if it satisfies the following axioms.*

1. $P(A) \geq 0$
2. $P(S) = 1$
3. *If A_i is a sequence of events in A, then for every infinite sequence of disjoint events, A_1, A_2, A_3,...we have*

$$P\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} P(A_i)$$

When a probability measure is specified across a sample space, we can estimate the probabilities regarding the possible values of a random variable [15]. The distribution of these probabilities defines the random variable.

Definition 2.2 *A random variable X is a discrete random variable if its defined space is finite or infinitely countable.*

Its probability mass function is defined as the function f such that for each real number x,

$$p(x) = \Pr(X = x)$$

Definition 2.3 *For any value x, the probability mass function of a discrete random variable X satisfies*

1. $0 \leq \Pr(x) \leq 1$

2. $\sum_{x} \Pr(x) = 1$

Similarly, random variables are real-valued functions that are defined on a sample space. Though these concepts, probability and randomness are closely associated, they however do not coincide.

Definition 2.4 *A random variable X has a continuous distribution if its cumulative distribution function $F(x)$ is a continuous function for all $x \in \mathbb{R}$. [15].*

For a random variable X associated with an interval [a,b],

$$\Pr(a \leq X \leq b) = \int_a^b f(x)dx$$

for the probability density distribution of function f, it satisfies that

1. $f(x) \geq 0$ for all x

2. $\int_{-\infty}^{\infty} f(x)dx = 1$

Definition 2.5 *The cumulative distribution function F(x) of a random variable X is defined as*

$$F(x) = \Pr(X \leq x)$$

$F(x)$ is monotone and non-decreasing [22, 25]. That is if $a < b$, then $F(a) \leq F(b)$. The probability that a random variable X takes on any value from $[a, b]$ is

$$P(a < x \leq b) = F(b) - F(a) = \int_a^b f(x)dx$$

$F(x)$ as defined above follows for the respective distribution for each random variable X.

Suppose that the random variable X, has a discrete distribution with probability function $Pr(x)$, then $F(x)$ is

$$F(x) = \Pr(X \leq x) = \sum_x Pr(x)$$

For a continuous random variable X with probability density function $f(x)$

$$F(x) = \Pr(X \leq x) = \int_{-\infty}^{x} f(x)dx$$

If $F(x)$ is continuous at all x (Fundamental Theorem of Calculus) [15, 22, 32] then

$$\frac{d}{d(x)}F(x) = f(x)$$

Definition 2.6 *The expected value of a random variable X is the mean or average of the probability distribution.*

Suppose X be a continuous random variable with probability density function $f(x)$. If the integral $\int_{-\infty}^{\infty} |x| f(x) < \infty$, then the mean, or expected value of X is said to exist and is defined to be

$$E(X) = \int_{-\infty}^{\infty} x f(x) dx$$

If X is a discrete random variable with probability function Pr(x) and if $E(|X|) < \infty$, then the expectation,

$$E(X) = \sum_{x} x \Pr(x)$$

The expectation of any real-valued function $g(x)$ may be determined using the above definitions (2.6) of expectation to the distribution of a function g.

Theorem 2.1 *Let Y = g(X). Suppose X is a continuous random variable with probability density function f(x) for any real-valued function g(x). If $\int_{-\infty}^{\infty} |g(x)| f(x) dx < \infty$, then,*

$$E(Y) = E[g(X)] = \int_{-\infty}^{\infty} g(x) f(x) dx$$

If random variable X discrete with probability distribution Pr(x) for any real valued function g(x) and if $\sum_x |g(x)| x \Pr(x) < \infty$, then

$$E(Y) = E[g(X)] = \sum_{x} g(x) \Pr(x)$$

Many experiments include examining the times at which random events occur. Accurately forecasting arrival is critical for achieving optimal operational efficiency [12].

Exponential distributions are used to model the length of time until the next event occurs.

Definition 2.7 *A random variable X is said to have an exponential distribution with parameter λ (the mean arrival rate), $\lambda > 0$, if its probability density function*

$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & x \geq 0 \\ 0 & \textit{otherwise} \end{cases}$$

This implies that, $\lambda = \frac{1}{x}$. where x is the mean inter-arrival time. A counting process that is related to the exponential distribution, is the Poisson process. The Poisson arrival process is a suitable model used to model arrivals at a fixed time interval. [63].

2.1.1 Convexity

Definition 2.8 *[60] A convex set is a set $C \subseteq \mathbb{R}^n$ such that if for any $x_1, x_2 \in C$ and $\alpha \in [0, 1]$ we have*

$$\alpha x_1 + (1 - \alpha)x_2 \in C.$$

Geometrically, convex set has a line segment connecting any two (2) points in the set. The empty set, Φ and $\mathbb{R}^n$ are both convex. The convexity of a set is preserved by scaling and translation. The intersection of convex sets are also convex sets [49]. Notably, half-spaces, hyperplanes are also convex sets. These form the basis for a lot of practical optimization problems [9].

Definition 2.9 *A convex function is a function f(x) $\rightarrow \mathbb{R}$ such that for all $x_1, x_2 \in X$ and $\alpha \in [0, 1]$ then*

$$f(\alpha x_1 + (1 - \alpha)x_2) \leq \alpha f(x_1) + (1 - \alpha)f(x_2)$$

If $f(x)$ and $g(x)$ are convex functions, then we have that

1. $af(x) + bg(x)$ is convex for convex f, g and $a, b > 0$

2. the pointwise maximum, $\max(f(x), g(x))$ is convex for convex $f(x)$ and $g(x)$

2.1.2 Convex Optimization

In Machine Learning and Operations Research, many problems can be framed as convex optimization problems [34, 41]. Convex optimization is generally repeatable. This is important for diverse business or industry decision making. Its application encompasses fields such as communication and networks, estimation and signal processing, and autonomous control systems.

Practically, in formulating a convex optimization problem, we are thinking of optimization under constraints. An optimization problem is framed as

$$\begin{aligned} \min_{x} \quad & f_0(x) \\ \text{subject to:} \quad & f_i(x) \leq 0 \quad \forall i = 1, \ldots, m \\ & h_i(x) = 0, \quad \forall i = 1, \ldots, n \end{aligned}$$

1. $f_0(x) : \mathbb{R}^n \to \mathbb{R}$: objective/cost function

2. $f_i(x) : \mathbb{R}^n \to \mathbb{R} : i \in 1, \ldots, m$: inequality constraint

3. $h_i(x) : \mathbb{R}^n \to \mathbb{R} : i \in 1, \ldots, n$: equality constraint

4. optimal solution x^* has the smallest value of $f_0(x)$ among all vectors that satisfy the constraint.

5. $x = (x_1, ..., x_n)$: optimization variables

If the objective function and the constraints are convex functions thus, all $f_i(x), i \in 1, \ldots, m$ are convex and all $h_i(x), j \in 1, \ldots, n$ are affine [9] then the optimization problem is said to be convex. This together with the objective function ensures that all local minima are global minima.

2.1.3 Regularization

Models are often tested to see how best they fit a data set. In such instances, the entire data set is split into a training set (data set on which the model is developed) and a test set (how best the model works and fits the data). Sometimes these models are too flexible or complex for the data set. That is, the model follows too closely to the randomness in the data (over-fitting). This leads to high test errors which reduces the predictive accuracy of a model [8]. By adding a penalty term to the loss function, regularization attempts to limit the set of learn-able models. Regularization is useful in models with high variance (models with parameters that fluctuates with different training sets). In optimization, adding a regularization term to a cost function help tuning a preferred level of model complexity to improve prediction accuracy.

In regularization, a variable, w, is added to a loss function to penalize complexity or flexibility to prevent extreme values. In L^1 regularization, the loss function becomes

$$f(w) = L(w, X, y) + \lambda \|w\|_1$$

where $\lambda \geq 0$. L^1 regularization forces sparsity [40]. L^2 regularization forces small

rms (root mean square) [37]. The loss function then becomes

$$f(w) = L(w, X, y) + \frac{\lambda}{2}\|w\|_2^2$$

An important property of the L^1-regularization is that it performs feature selection which is relevant to this thesis.

In subsequent sections of this thesis, (Section 3), the L^1 regularization would be used to improve the utility of an optimization problem into a linear program which is solvable.

2.2 Linear Programming

A linear program is a convex optimization in which the objective and constraints are linear. A linear program may be framed in the form

$$\min_x \quad c^T x$$

$$\text{s.t:} \quad Ax \leq b$$

$$x \geq 0$$

A set of linear constraints can be expressed in different forms. This makes transformation of linear programs quite easy. Most inventory and daily life activities can be framed into a linear optimization problem [9]. An example is maximizing the output of a factory, the x is the quantity of items that may be produced, c is the profit and the constraints in A and b can be the materials available.

Stochastic linear programming allows us to optimize programs where certain parameters are random. In a two stage stochastic linear programming model, the decision variables are grouped into two stages. The initial stage is based on whether

they are applied before or after the random variable's result is noticed and a second decision (recourse function) enable us to model a reaction to the output, which serves as a recourse action. A standard two-stage stochastic linear program can be posed as

$$\begin{aligned} \min_{x} \quad & c^T x + E_\xi[Q(x,\xi)] \\ \text{s.t:} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

where $c \in \mathbb{R}^n$ is given. The function $Q(x,\xi)$ in itself can be defined by an optimization problem. In the second stage, $Q(x,\xi)$ is the optimal value of

$$\begin{aligned} \min_{x} \quad & c^T x + q^T y \\ \text{s.t:} \quad & T(\xi)x + W(\xi)y = h(\xi) \\ & y \geq 0 \end{aligned}$$

where $(q(\xi), h(\xi), T(\xi), W(\xi))$ is the random data of the problem.

Assuming that there are K outcomes and a random variable $\xi = (h, q, T, W)$ with probabilities p_k, $1 \leq k \leq K$, the expectation $E_\xi[Q(x,\xi)]$ in the first stage becomes

$$E_\xi[Q(x,\xi)] = \sum_{k=1}^{K} Q(x,\xi) p_k$$

The second stochastic program then

$$\min \quad c^T x + \sum_{k=1}^{K} (q_k)^T y_k p_k$$

$$\text{s.t:} \quad Ax = b$$

$$T_k x + W_k y_k = h_k$$

$$x \geq 0, \quad y_k \geq 0 \quad k = 1, \ldots, K$$

When the variables are continuous, these formulations result in infinite-dimensional problems which are complex and difficult to solve [51].

2.3 Stochastic Linear Programs - Sampling Methods

From a statistical standpoint, it makes sense to minimize the objective function on average. However, objective functions are often complicated and difficult to solve. The principle of Sample Average Approximation (SAA) provides solutions to these problems through the use of sampling and optimization methods. The sample average approximation method uses Monte Carlo simulation to solve stochastic optimization problems [55]. The principles of Monte Carlo methods are based on the strong law of large numbers. Suppose that we do not know what distribution our real-valued function $f(x)$ is, instead, we have a fixed set ξ whose distribution does not depend on x. The principle of SAA, is to select a random sample $\xi_1, \xi_2, ...\xi_n$ which is independently and identically distributed (i.i.d) as vector ξ and set

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} f(x, \xi_i)$$

Given $h_1, h_2, .., h_n$ which is independently and identically distributed random sample of N realization of random vector ξ, the sample average of the expectation function $E_\xi[Q(x, \xi)]$ become

$$E_\xi[Q(x, \xi)] = \frac{1}{N} \sum_{k=1}^{N} Q(x, h_k)$$

Since $f_n(x)$ is deterministic, an optimization approach can be used to solve the two-stage stochastic program in the form of:

$$\min c^T x + \frac{1}{N} \sum_{k=1}^{N} Q(x, h_k) \tag{1}$$

$$\text{s.t: } Ax = b \tag{2}$$

$$T_n x + W_n y_k = h_k \tag{3}$$

$$x \geq 0, \quad y_k \geq 0 \quad k = 1, \ldots, K \tag{4}$$

Equation (2) represents the SAA objective function, which corresponds to the original two-stage stochastic program's objective function. Because sampling and optimization are independent, the sampling technique and optimization method can be split into modules. This makes the programming simpler.

2.4 The Data Driven Newsvendor

With accessible and large data, machine learning and data science inputs can be employed to gain greater insights on company issues or business scenarios to make informed decisions. In some instances, trends in data set can be indicative in the prediction of demand [5, 6]. Instead of making assumptions about the distribution of the demand, data-driven models which do not depend on a parametric demand distribution are used. This approach is the utilization of the principle of Sample Average Approximation [56].

2.4.1 Nonparametric Newsvendor Models

In this, assuming the sample demand $D_1, D_2, ...D_n$ are identically and independently distributed and drawn from a distribution with a cumulative distribution function $F(k)$. A formulation of the newsvendor problem would be to compute an x^* that minimizes expected cost of the vendor.

$$\min_x \frac{1}{n} \sum (\, c_o(x - D_i)^+ + c_u(D_i - x)^+ \,)$$

Since the objective function of the model is the expectation of the loss under empirical distribution,

$$F(k) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}(D_i \leq k)$$

An inventory level x^* is chosen so that

$$F(x^*) = \frac{c_u}{c_o + c_u}$$

If the demand D_i is ordered, that is, $D_1, D_2, ..., D_n$, the solution to the newsvendor problem corresponds to the $\lceil n(\frac{c_u}{c_o+c_u}) \rceil - th$ value. That is, $X_{\lceil n(\frac{c_u}{c_o+c_u}) \rceil}$.

2.4.2 Linear Parametric Models

If we have sample demands, $D_1, D_2, ..., D_n$, with features $Z_1, Z_1, ..., Z_n \in \mathbb{R}^p$ respectively, then our goal is to predict x^* using these features. In these type of problems, the optimal decision depends on recent data. In the ice cream vendor problem, suppose we want to predict demand for either ice cream or hot chocolate. Then the feature Z_i may be a vector of features such as temperature, probability of rainfall, time of day etc. With this, we could use a parametric model that takes the features

and predict the corresponding demand. A set of corresponding parameters β of the model is chosen to make good predictions of the corresponding demand that minimizes the expected cost to the vendor. In a linear parametric newsvendor, the x_i's are assumed to be linearly related to the features. That is,

$$x_i = Z_i^T \beta$$

Z_i^T denotes the transpose of Z_i. The objective function is

$$\min_{\beta} \frac{1}{n} \sum (\, (c_o(Z_i^T \beta - D_i)^+ + c_u(D_i - Z_i^T \beta)^+ \,)$$

This problem can then be formulated as a linear program to solve the newsvendor problem. Thus, given feature Z_{new}, the optimal order quantity becomes

$$x^* = Z_{new}^T \beta^*$$

3 THE ICE CREAM VENDOR PROBLEM

Distribution assumptions about demand to estimate order quantities of products are problematic and may lead to misspecified ordering decisions [59]. This can result in an increase in expected costs to a vendor. The ice cream vendor problem is a data-driven extension of the conventional newsvendor problem that does not require the demand to have an assumed distribution. In this thesis, our goal is to find the minimum expected cost to the vendor given demand with some features.

3.1 Derivation, Motivation and Explanation of Problem

Similar to the newsvendor, the ice cream vendor has to make an initial one time order decision before the beginning of the day. The demand for both ice cream and hot chocolate assumed to be random. The vendor incurs a cost for overstocking, and he also incurs a cost if he orders too few and has to turn customers away. To maintain goodwill to customers, he issues coupons which incurs additional costs. We present the ice cream problem as a linear parametric newsvendor problem in which we consider two products ice cream and hot chocolate. In order to achieve our goal:

1. We simulate a real-world scenario of an ice cream vendor problem via a demand whose mean is a function of temperature.

2. We reduce the problem to a Newsvendor problem for temperature categories (hot, medium, cold). Using sample average approximation, the vendor's problem is transformed into a feature-driven linear program dependent on exogenous factors such as temperature and rainfall (develop an approach that utilizes data-

driven inputs to predict demand). With enough data, the expectation of the newsvendor loss can be estimated from the features independent of the demand distribution.

3. We use data science methods to produce an interpretable model that improves the utility of the newsvendor solution of the ice cream vendor problem.

It's worth noting that, in the ice cream vendor problem, there are two products – ice cream and hot chocolate – which may be substituted for one another if the outside temperature is not too hot or too cold. We present a novel approach by further extending our model to account for substitution. We assume that customers may decide to either substitute ice cream or chocolate for the other as a result of changed weather conditions. Our method is based on data science methods that leverage large available datasets independent of a demand distribution.

3.2 Simulation of the Ice Cream Problem

Since no assumption is been made about the demand distribution, the Simpy package (in Python) is used to simulate a discrete event simulation of an ice cream vendor problem via a demand whose expected value is a function of temperature. We considered probability of rainfall and temperature as features for the demand for a given day. In the simulation, we instantiate a time to next arrival process generated by the features. We defined a function that triggers a sold event if a product sells out. We also assume that a vendor sells his products for 5 hours each day, and the vendor expected 600 arrivals per 300 minutes (0.5 minutes on average) independent of temperature or the probability of rain. For this simulation, we assume:

- $c = 1$
- $c_o = 2$
- $c_u = 7$
- maximum temperature for a given day = 74
- minimum temperature for a given day = 36
- average temperature for a given day = 55

The simulation is run for 5 different probability categories $(0.0, 0.2, 0.4.0.6, 0.8)$ over temperature ranges between 36 and 74 (degrees Fahrenheit). Using the features, we generated a sample size of three thousand (3000) demands as a function of temperature. With a sample size of 100, SAA was used to calculate the average expected cost and the order quantity x^*. To generate a temperature dependent demand, the simulation is run separately for ice cream and hot chocolate with fixed cost parameters (but for probability of rainfall and temperature).

Table 2: Simulated Average Cost for Ice-Cream

Pr(rain)	Theoretical x^*	Simulation x^*	Average Cost
0.0	788	812	4913.97
0.2	642	635	3834.99
0.4	490	468	2728.92
0.6	322	311	1608.29
0.8	156	140	560

From Table 2, the theoretical optimal order quantity and the simulated generated order quantity are close for the given features. At a 0.2 probability of rainfall, which

could also mean high temperatures, the minimum expected cost to the vendor is highest. This could be attributed to the high order quantity ($x^* = 812$) for the day. Because naturally, a customer will want an ice cream or something cold if it hot. Though the lowest average cost to the vendor is lowest at 0.8, it may incur the highest back order cost given the order quantity. Like the newspaper vendor, a back order cost may be high or expensive to the ice cream vendor since unsold products are often discarded or salvaged.

Table 3: Simulated Average Cost for Hot Chocolate

Pr(rain)	Theoretical x^*	Simulation x^*	Average Cost
0.0	760	797	5110.12
0.2	626	631	4014.62
0.4	456	446	2886.22
0.6	284	290	1809.13
0.8	152	135	712

From Table 3, high average cost could be attributed to overstocking of the products. Over stocking in this instance may not be expensive to the vendor (chocolate is in a powdered form before it is prepared) since such products can be carried over and resold in the next sale-period. In comparison of the average costs for the products, medium temperatures yield just about the same minimum costs. This could be due to a substitution effect. The vendor's average cost for ice cream is 9.96% less than hot chocolate.

Due to the similarities between the simulation results and the theoretical results, we assert that the ice cream vendor problem is reduced to a newsvendor problem for a fixed temperature category.

3.3 Model Formulation - The Data-Driven Newsvendor

As established in Section 2.4, the SAA can be used to make predictions of the vendors minimum cost if there is sample demand and corresponding features. With this approach, the optimal order quantity is estimated directly from the feature rather than calculating the mean demand and error distribution and then solving the newsvendor problem. Thus, optimal order quantity x^* of the newsvendor model is a function of the feature data. This implies that x^* is a linear function of the features (Theorem 2.3).

Mathematically, the ice cream vendor problem can be framed as

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} (\, c_o(Z_i^T \beta - D_i)^+ + c_u(D_i - Z_i^T \beta)^+ \,)$$

If we define nonnegative variables $s_i \geq Z_i^T \beta - D_i$ and $t_i \geq D_i - Z_i^T \beta$, then the program can be reformulated as a linear program of the form

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} (c_o s_i + c_u t_i) \tag{5}$$

$$\text{s.t:} \quad s_i \geq -(Z_i^T \beta - D_i) \tag{6}$$

$$t_i \geq (Z_i^T \beta - D_i) \tag{7}$$

$$s_i, t_i \geq 0 \tag{8}$$

The objective function in (6) minimizes the sum of underage and overage costs. The constraints in (7-9) ensure that any variation from the estimated demand are assigned to the appropriate underage and overage costs. The argin β^*, is the value of β that minimizes the expected costs with respect to the feature data.

3.3.1 Implementation of the Model

We evaluate our proposed approach using the data generated from the discrete event simulation for each of the products. We use the GLPK (GNU Linear Programming Kit) package executed in Python to solve for the minimum expected cost of the ice cream vendor. GLPK is for solving large-scale linear programming (LP). In implementation of the model, we assumed the same varying underage and overage costs for both ice cream and hot chocolate.

Table 4: Expected Minimum Cost for Ice Cream and Hot Chocolate

c_o	c_u	E(Ice Cream)	E(Hot Chocolate)
2	1	158999.690	158999.669
0.5	0.5	58946.246	58766.205
0.75	3	145181.537	145181.537
4	1	192739.835	192736.835
2	2.5	265256.199	265256.199
0.11	0.45	214426.582	21428.412
2	3	290703.915	2990307.063
1.5	0.8	125922.56	124362.919
1.62	1.40	181454.737	180707.615
0.25	0.33	30053.828	34040.921

From Table 4, in most parts, the expected minimum costs are comparable. However, it can also be observed that underage costs are higher for hot chocolate. Overage costs are comparatively higher for ice cream.

3.4 Improving the Utility of the Model - Data-Driven Regularization

Just as in regression models, we can regularize this problem by either using L^1-regularization or L^2-regularization. This is motivated by the interpretation of a regu-

larizing parameter as a term that penalizes complexity or flexibility to prevent extreme values (highest predictive accuracy or lowest bias) in a model to one that has desirable or appropriate properties [8]. In both approaches, the statistical advantages still hold. L^1-regularization penalizes non-zero values and forces sparsity [66]. L^2-regularization forces small rms (root mean square) due to significant amount of correlation between data features [19]. In our model, we interpret 'appropriateness' or 'desirability' as having the minimum expected cost to the newsvendor problem. The L^1 regularized version of the problem then becomes

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} (c_o(Z_i^T \beta - D_i)^+ + c_u(D_i - Z_i^T \beta)^+) + \lambda \|\beta\|_1 \tag{9}$$

where $\lambda > 0$ is a tuning parameter, and $\|\beta\|_1 = \sum_{k=1}^{p} |\beta_k|$ is the ℓ_1 norm of a p-dimensional vector. It is important to note that varying λ has an effect on the expected cost function.

Theorem 3.1 *A norm $\|.\|$ is a convex function.*

In order to describe our reformulation of the problem as an LP, suppose $x \in \mathbb{R}^2$, consider the constraint

$$\|x\|_1 = |x_1| + |x_2| \leq 0 \tag{10}$$

The constraint $|x_1| + |x_2| - 1 \leq 0$ can also be written as intersection of four (4)

half-spaces:

$$x_1 + x_2 - 1 \leq 0$$

$$x_1 - x_2 - 1 \leq 0$$

$$-x_1 + x_2 - 1 \leq 0$$

$$-x_1 - x_2 - 1 \leq 0$$

The constraint in equation (10) above, can be represented as a projection of a set. For example, $|x| \leq 1$ can be the projection of a polytope when a variable t_1 is added:

$$-t_1 \leq x_1 \leq 0$$

$$t_1 = 0$$

The constraint $|x_1| + |x_2| - 1 \leq 0$ can now be written as the projection of a polytope by the introduction of two new variables t_1, t_2 which then becomes:

$$-t_1 \leq x_1 \leq t_1$$

$$-t_2 \leq x_2 \leq t_2$$

$$t_1 + t_2 = 1$$

$$\sum_{k=1}^{p} |x_k| - 1 \leq 0 \tag{11}$$

The constraint in equation (11) can be written as the intersection of half-spaces, that is

$$\sum_{k=1}^{p} \pm x_k - 1 \leq 0$$

Here, there are p number of variables and 2^p constraints. $\pm$ shows all the possible permutation of signs in the set of constraints. As a polytope (which requires less constraints), the constraints becomes

$$
\begin{aligned}
&-t_k \leq x_k \leq t_k \; \forall k = 1, \ldots, p \\
&\sum_{k=1}^{p} t_k = 1
\end{aligned}
$$

In this, there are 2_p variables $x_1, ..., x_p$, $t_1, ..t_p$ and $2_p + 1$ constraints. The constraints in equation (11) requires less inequalities when represented as the projection of a polytope. With this intuition, we reformulate the ice cream vendor problem in equation (9) as an LP in the form

$$
\begin{aligned}
&\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} (\, c_{o,i} s_i + c_{u,i} t_i \,) + \lambda U \\
\text{s.t:} \quad &s_i \geq -(Z_i^T \beta - D_i) \\
&t_i \geq (Z_i^T \beta - D_i) \\
&s_i, t_i \geq 0 \\
&\beta_k \geq -v_k, \; \forall k = 1, \ldots, p \\
&\beta_k \leq v_k, \; \forall k = 1, \ldots, p \\
&\sum_{k=1}^{p} v_k = U
\end{aligned}
$$

3.4.1 Implementation of the Regularized Model

We implement the model using the same cost parameters in Subsection 3.3.1. Through cross-validation a tuning parameter $\lambda = 0.001$ was chosen for both ice cream and hot chocolate respectively.

Table 5: Expected Minimum Costs with L^1-Regularization

c_o	c_u	E(Ice Cream)	E(Hot Chocolate)
2	1	161008.737	158999.669
0.5	0.5	59738.246	59846.205
0.75	3	145629.986	145181.537
4	1	195098.304	192736.835
2	2.5	265474.960	265256.199
0.11	0.45	214426.582	21426.582
2	3	290703.915	290703.915
1.5	0.8	124362.941	125922.560
1.62	1.40	180707.633	181454.737
0.25	0.33	34040.937	34071.612

From Table 5, in comparison of the expected minimum costs for ice cream and hot chocolate, it is about 0.2% cheaper for the vendor to stock up on ice cream hot than chocolate (this amount could translate into huge cost savings for companies with a big budget).

To put these results into perspective, we compare the utility of the models from Table 4 and from Table 5. Using the same c_o and c_u, it is about 1.57% more expensive to use the L^1 regularized model. This could be due to the influence of the regularization parameter which penalized the model and hence. This however does not imply the accuracy of the model has been jeopardized, but instead, may have improved.

3.5 Incorporating Substitution

In today's marketplace, several products may have the same or similar utility. If a customer's product of choice is not in stock, they might opt for another product in the same category [62]. Substitution provides customers with alternatives. Particularly, in substitution, when the substitutable product has excess, customers choose this product which is in the same category as their first-choice product. For instance, when at a restaurant, a customer might turn to a Pepsi-Cola if Coca-Cola was currently unavailable.

In most instances, substitution may be beneficial to customer [47]. Although this is good, due to the stochastic nature of demand it becomes inherently difficult to forecast demand and substitution rates accurately. A disadvantage of substitution is that substitute products can cut into a business' profitability, as consumers may end up choosing one product over the other [42].

Stochastic programming models with an assumed distribution have been proposed to solve these substitution demands [48, 52]. In most cases, these estimates are speculative. In particular, technology companies like Samsung and Apple releases new products (phones, computers, earphones etc.) each year. It becomes almost impossible to accurately predict demand and substitution rates of these new products because there is not enough data. This can cause misspecified decisions which ultimately results in huge costs to the company. Product substitution is an extensive area of study in inventory management. Monte Carlo simulation approach has been used to find optimal solutions for a two substitute product newsvendor problems [29]. In [24] partial product substitution on a multi-product competitive newsvendor prob-

lem was studied. They came to the conclusion that competition always resulted in a greater overall inventory level.

Most often, vendors that sell perishable product also sell a variety of other products in the same product category (produce, fashion industry etc.). Having optimal inventory levels is challenging especially due to stock-outs and the high demand for a product due to a substitution effect. Substitution in multi-product newsvendor problem poses new challenges [52]. A major challenge of this substitution behavior is how to distribute demand across the different products. Other challenges may be determining the order quantity and how to deal with salvage [42]. This results in challenging models that are difficult to work with and require sophisticated programs to solve.

In this section, we further extend the model of the ice cream vendor problem to include substitution. We extend the concept of a data-driven newsvendor problem to a two-product newsvendor problem with feature-driven demand which is a function of temperature. Our method builds on the regularized version of the linear parametric model ice cream vendor. We assess our approach based on medium temperatures where a customer may either substitute ice cream or hot chocolate. In this, the substitution may not be necessarily be due to product stock out [38]. It is therefore important to note that, due to the exogenous factors such as temperature related demand, safety stocking of these kind of products may not always be ideal. Here we consider substitution between two products - ice cream and hot chocolate - but the approach immediately generalizes to any number of products.

3.5.1 Substitution - Model Formulation

To formulate this effect, we supposed that the ice cream vendor's products are similar and can be proportionally substituted by the other. Similar to the parametric model, we also assumed that we have samples of demand $D_1, D_2, ...D_n \in \mathbb{R}$ for each product along with corresponding features $Z_i, Z_2, ...Z_n \in \mathbb{R}^p$ (such as probability of rain, temperature, etc). Due to substitution effects, when one product i runs out, a portion of the demand that cannot be met shifts to the other, j. This results in an increased demand of the substitute and subsequently increase in sales.

Consider α_{ij} as the rate of unmet demand of one product (percentage of unmet demand for product j replaced by product i [61]). Substitution means replacing the demand D_i by its effective demand. The effective demand is expressed as

$$\tilde{D}_i = D_i + \sum_{i \sim j} \alpha_{ij}(D_j - X_j)^+ \tag{12}$$

where $i \sim j$ if product j can be substituted for product i. If $\alpha_{ij} = 0$, then the model reduces to n newsvendor problems. The L^1 regularized version of the substitution problem then becomes

$$\min_{\beta} \frac{1}{n} \sum_{i=n}^{n} (\, c_{o,1}s_i + c_{u,1}t_i + c_{o,2}s_i + c_{u,2}t_i \,) + \lambda_1 U_1 + \lambda_1 U_2 \tag{13}$$

Suppose that $s_i \geq Z_i^T\beta - D_i + \alpha_{i,j}(D_i - Z_i^T\beta)^+$ and $t_i \geq (D_i + \alpha_{i,j}(D_i - Z_i^T\beta)^+ - Z_i^T\beta)$.

s_i, t_i bounded below by 0 or a linear expression whichever is larger. Let non-negative $w_i = (D_i - Z_i^T\beta)^+$, Ice cream = 1 and Hot chocolate =2

We reformulate the problem in equation (13) as an L^1-regularized LP with corresponding constraints in the form

$$\min_\beta \frac{1}{n} \sum_{i=n}^{n} (c_{o,1} s_i + c_{u,1} t_i + c_{o,2} s_i + c_{u,2} t_i) + \lambda_1 U_1 + \lambda_1 U_2$$

$$\begin{aligned}
\text{s.t: } s_{1,i} &\geq D_{1,i} - \alpha_{ij} w_i - Z_{1,i}^T \beta \\
s_{2,i} &\geq D_{2,i} - \alpha_{ij} w_i - Z_{2,i}^T \beta \\
t_{1,i} &\geq Z_{1,i}^T \beta - D_{1,i} + \alpha_{ij} w_i \\
t_{2,i} &\geq Z_{2,i}^T \beta - D_{2,i} + \alpha_{ij} w_i \\
s_{1,i}, t_{1,i} &\geq 0 \\
s_{2,i}, t_{2,i} &\geq 0 \\
\beta_k &\geq -v_{1,k} \; \forall k = 1, \ldots, p \\
\beta_k &\leq v_{1,k} \; \forall k = 1, \ldots, p \\
\beta_k &\geq -v_{2,k}, \; \forall k = 1, \ldots, p \\
\beta_k &\leq v_{2,k} \; \forall k = 1, \ldots, p \\
\sum_{k=1}^{p} v_{1,k} &= U_1 \\
\sum_{k=1}^{p} v_{2,k} &= U_2
\end{aligned}$$

3.5.2 Implementation of the Substitution Model

We evaluate our proposed model by setting the overage and underage parameters to be the same as the regularized models for ice cream and hot chocolate for comparison. Through cross-validation a tuning parameter $\lambda = 0.001$ was chosen for both ice cream and hot chocolate respectively. We assumed equal rate of unmet demand

Table 6: Comparison between $L^1 Regularized$ Models and the Substitution Model

c_o	c_u	E($L^1 Regularized$ Models)	E(Substitution Model)	Amount Saved
2	1	320008.4	317308.646	2699.754
0.5	0.5	117712.451	116907.568	804.883
0.75	3	290811.5	277329.255	13482.27
4	1	387835.1	406893.564	-19058.43
2	2.5	530731.2	517337.001	13394.16
0.11	0.45	235853.2	40816.986	195036.2
2	3	581407.8	557844.398	523563.4
1.5	0.8	251845.1	247662.632	4182.488
1.62	1.40	362909.5	353142.256	9767.218
0.25	0.33	60107.66	57831.759	2275.897

α_1=0.05 for both products. We report the resulting expected minimum costs from Table 6 in Figure 1. We observed that the substitution model is able to minimize costs even with higher overage costs. It however under performs with very high overage costs (example, $c_o = 4, c_u = 1$). This translates into real world situations where a vendor sells perishable products and incurs a cost for overstocking. We also see a smaller difference in the performance of the substitution model when the overage costs are similar ((example, $c_o = 0.5, c_u = 0.5$).

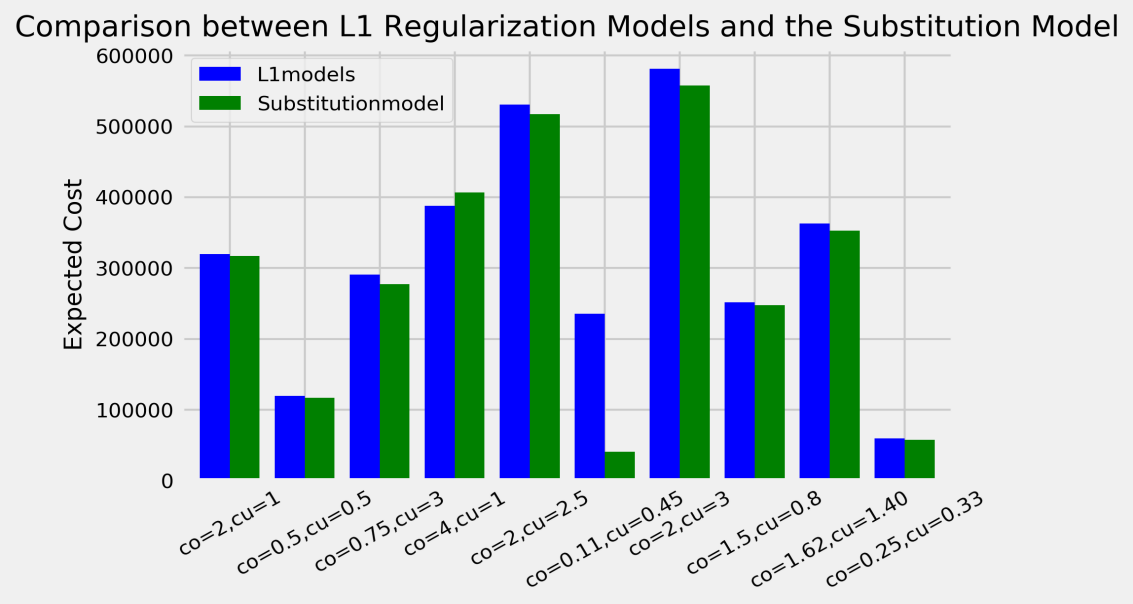

Figure 1: Comparison between Models

3.5.3 Interpretation and Conclusion of Results

In comparison of the average costs from Table 6, the L^1 substitution model outperforms with the least expected costs for all the cost parameters except except for very high overage costs. We estimate that, substitution model minimizes the vendor's cost by 17.77% less than if he were to use the L^1 regularization models for ice cream and hot chocolate. This could be attributed to the substitution effect. Overall, we find that accounting for substitution is essential in the decision making process of the ice cream vendor. However, a limitation of the substitution model is that lost demands may be unobserved leading to severe underage costs.

In conclusion, with the substitution model, there will no need reduce the problem to solve an optimization problem for each temperature category (hot, cold or medium temperatures). With β^*, and new features (recent data), a decision rule of the optimal

order quantity can be made. Since there is no distribution assumption, ultimately, the risk of making misspecified ordering decisions are reduced.

4 FUTURE DIRECTION

In future analysis, the ice cream vendor problem could be extended to multi-periods where products are reordered during a selling-period. This extension can be made to include more features and product.

BIBLIOGRAPHY

[1] Layek L Abdel-Malek and Roberto Montanari. An analysis of the multi-product newsboy problem with a budget constraint. *International Journal of Production Economics*, 97(3):296–307, 2005.

[2] Kenneth R Baker and Gary D Scudder. Sequencing with earliness and tardiness penalties: a review. *Operations research*, 38(1):22–36, 1990.

[3] Alain Bensoussan, Metin Cakanyıldırım, and Suresh P Sethi. On the optimal control of partially observed inventory systems. *Comptes Rendus Mathematique*, 341(7):419–426, 2005.

[4] Alain Bensoussan, Metin Cakanyıldırım, and Suresh P Sethi. A multiperiod newsvendor problem with partially observed demand. *Mathematics of Operations Research*, 32(2):322–344, 2007.

[5] Dimitris Bertsimas and Nathan Kallus. From predictive to prescriptive analytics. *arXiv preprint arXiv:1402.5481*, 2014.

[6] Dimitris Bertsimas and Nathan Kallus. From predictive to prescriptive analytics. *Management Science*, 66(3):1025–1044, 2020.

[7] Dimitris Bertsimas and Aurélie Thiele. A data-driven approach to newsvendor problems. *Working Papere, Massachusetts Institute of Technology*, 2005.

[8] Peter J Bickel, Bo Li, Alexandre B Tsybakov, Sara A van de Geer, Bin Yu, Teófilo Valdés, Carlos Rivero, Jianqing Fan, and Aad van der Vaart. Regularization in statistics. *Test*, 15(2):271–344, 2006.

[9] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[10] Apostolos Burnetas, Stephen M Gilbert, and Craig E Smith. Quantity discounts in single-period supply contracts with asymmetric demand information. *IIE Transactions*, 39(5):465–479, 2007.

[11] Gerard Cachon and Christian Terwiesch. *Matching supply with demand*. McGraw-Hill Publishing, 2008.

[12] Mehmet Tolga Cezik and Pierre L'Ecuyer. Staffing multiskill call centers via linear programming and simulation. *Management Science*, 54(2):310–323, 2008.

[13] Fangruo Chen and Yu-Sheng Zheng. Near-optimal echelon-stock (r, nq) policies in multistage serial systems. *Operations research*, 46(4):592–602, 1998.

[14] Sunil Chopra and Peter Meindl. Supply chain management. strategy, planning & operation. In *Das summa summarum des management*, pages 265–275. Springer, 2007.

[15] Morris H DeGroot. *Probability and statistics*. Pearson, 2012.

[16] Steven J Erlebacher. Optimal and heuristic solutions for the multi-item newsvendor problem with a single capacity constraint. *Production and Operations Management*, 9(3):303–318, 2000.

[17] Marshall Fisher and Ananth Raman. Reducing the cost of demand uncertainty through accurate response to early sales. *Operations research*, 44(1):87–99, 1996.

[18] Guillermo Gallego and Ilkyeong Moon. The distribution free newsboy problem: review and extensions. *Journal of the Operational Research Society*, 44(8):825–834, 1993.

[19] David K Guilkey and James L Murphy. Directed ridge regression techniques in cases of multicollinearity. *Journal of the American Statistical Association*, 70(352):769–775, 1975.

[20] Peijun Guo. One-shot decision theory: a fundamental alternative for decision under uncertainty. In *Human-centric decision-making models for social sciences*, pages 33–55. Springer, 2014.

[21] George Hadley and Thomson M Whitin. Analysis of inventory systems. Technical report, 1963.

[22] Robert V Hogg, Joseph McKean, and Allen T Craig. *Introduction to mathematical statistics*. Pearson Education, 2005.

[23] Di Huang, Qiu Hong Zhao, and Cheng Cheng Fan. Simulation-based optimization of inventory model with products substitution. In *Innovative quick response programs in logistics and supply chain management*, pages 297–312. Springer, 2010.

[24] Di Huang, Hong Zhou, and Qiu-Hong Zhao. A competitive multiple-product newsboy problem with partial product substitution. *Omega*, 39(3):302–312, 2011.

[25] Richard A Johnson, Irwin Miller, and John E Freund. *Probability and statistics for engineers*, volume 2000. Pearson Education London, 2000.

[26] Irwin W Kabak and Allen I Schiff. Inventory models and management objectives. *Sloan Management Review (pre-1986)*, 19(2):53, 1978.

[27] Samuel Karlin and Herbert Scarf. The nature and structure of inventory problems. *Studies in the Mathematical Theory of Inventory and Production, op. cit*, pages 16–36, 1958.

[28] Moutaz Khouja. The single-period (news-vendor) problem: literature review and suggestions for future research. *omega*, 27(5):537–553, 1999.

[29] Moutaz Khouja, Abraham Mehrez, and Gad Rabinowitz. A two-item newsboy problem with substitutability. *International journal of production economics*, 44(3):267–275, 1996.

[30] J. R. Kirkwood. *An Introduction to Analysis*. PWS Publishing Company and Waveland Press, 2 edition, 1995.

[31] Panagiotis Kouvelis and Genaro J Gutierrez. The newsvendor problem in a global market: Optimal centralized and decentralized control policies for a two-market stochastic inventory system. *Management Science*, 43(5):571–585, 1997.

[32] Richard J Larsen and Morris L Marx. *An introduction to mathematical statistics*. Prentice Hall, 2005.

[33] Haitao Li and Keith Womer. Modeling the supply chain configuration problem with resource constraints. *International Journal of Project Management*, 26(6):646–654, 2008.

[34] Xinfu Liu, Ping Lu, and Binfeng Pan. Survey of convex optimization for aerospace applications. *Astrodynamics*, 1(1):23–40, 2017.

[35] John T Mentzer, William DeWitt, James S Keebler, Soonhong Min, Nancy W Nix, Carlo D Smith, and Zach G Zacharia. Defining supply chain management. *Journal of Business logistics*, 22(2):1–25, 2001.

[36] Stephen M Millett. Should probabilities be used with scenarios. *Journal of futures Studies*, 13(4):61–68, 2009.

[37] Robert Moore and John DeNero. L1 and l2 regularization for multiclass hinge loss models. In *Symposium on machine learning in speech and language processing*, 2011.

[38] Sebastian Müller, Jakob Huber, Moritz Fleischmann, and Heiner Stuckenschmidt. Data-driven inventory management under customer substitution. *Available at SSRN 3624026*, 2020.

[39] Steven Nahmias and Ye Cheng. *Production and operations analysis*, volume 6. McGraw-hill New York, 2009.

[40] Henrik Ohlsson. *Regularization for sparseness and smoothness: Applications in system identification and signal processing*. PhD thesis, Linköping University Electronic Press, 2010.

[41] Daniel P Palomar and Yonina C Eldar. *Convex optimization in signal processing and communications*. Cambridge university press, 2010.

[42] Qin-hua Pan, Xiuli He, Konstantina Skouri, Sheng-Chih Chen, and Jinn-Tsair Teng. An inventory replenishment system with two inventory-based substitutable products. *International Journal of Production Economics*, 204:135–147, 2018.

[43] Mahmut Parlar. Game theoretic analysis of the substitutable product inventory problem with random demands. *Naval Research Logistics (NRL)*, 35(3):397–409, 1988.

[44] Evan L Porteus. The newsvendor problem. In *Building Intuition*, pages 115–134. Springer, 2008.

[45] Yan Qin, Ruoxuan Wang, Asoo J Vakharia, Yuwen Chen, and Michelle MH Seref. The newsvendor problem: Review and directions for future research. *European Journal of Operational Research*, 213(2):361–374, 2011.

[46] Ruozhen Qiu, Minghe Sun, and Yun Fong Lim. Optimizing (s, s) policies for multi-period inventory models with demand distribution uncertainty: Robust dynamic programing approaches. *European Journal of Operational Research*, 261(3):880–892, 2017.

[47] Kumar Rajaram and Christopher S Tang. The impact of product substitution on retail merchandising. *European Journal of Operational Research*, 135(3):582–601, 2001.

[48] Uday S Rao, Jayashankar M Swaminathan, and Jun Zhang. Multi-product inventory planning with downward substitution, stochastic demand and setup costs. *IIE Transactions*, 36(1):59–71, 2004.

[49] Ralph Tyrell Rockafellar. *Convex analysis*. Princeton university press, 2015.

[50] Evren Sahin and Yves Dallery. Assessing the impact of inventory inaccuracies within a newsvendor framework. *European Journal of Operational Research*, 197(3):1108–1118, 2009.

[51] Tjendera Santoso, Shabbir Ahmed, Marc Goetschalckx, and Alexander Shapiro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1):96–115, 2005.

[52] Maurice E Schweitzer and Gérard P Cachon. Decision bias in the newsvendor problem with a known demand distribution: Experimental evidence. *Management Science*, 46(3):404–420, 2000.

[53] Louis O Scott. Option pricing when the variance changes randomly: Theory, estimation, and an application. *Journal of Financial and Quantitative analysis*, pages 419–438, 1987.

[54] Kevin H Shang and Jing-Sheng Song. Newsvendor bounds and a heuristic for optimal policies in serial supply chains. *Manufacturing and Service Operations Management*, 4(1):2–4, 2002.

[55] Alexander Shapiro. Simulation-based optimization—convergence analysis and statistical inference. *Stochastic Models*, 12(3):425–454, 1996.

[56] Alexander Shapiro. Monte carlo sampling methods. *Handbooks in operations research and management science*, 10:353–425, 2003.

[57] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczynski. Lectures on stochastic programming. 2009.

[58] Jianmai Shi, Guoqing Zhang, and Jichang Sha. Jointly pricing and ordering for a multi-product multi-constraint newsvendor problem with supplier quantity discounts. *Applied Mathematical Modelling*, 35(6):3001–3011, 2011.

[59] Edward A Silver, David F Pyke, and Douglas J Thomas. *Inventory and production management in supply chains*. CRC Press, 2016.

[60] Barry Simon. *Convexity: an analytic viewpoint*, volume 187. Cambridge University Press, 2011.

[61] Hajnalka Vaagen, Stein W Wallace, and Michal Kaut. Modelling consumer-directed substitution. *International Journal of Production Economics*, 134(2):388–397, 2011.

[62] Tom Van Woensel, Karel Van Donselaar, Rob Broekmeulen, and Jan Fransoo. Consumer responses to shelf out-of-stocks of perishable products. *International Journal of Physical Distribution & Logistics Management*, 2007.

[63] Dennis Wackerly, William Mendenhall, and Richard L Scheaffer. *Mathematical statistics with applications*. Cengage Learning, 2014.

[64] Haifeng Wang, Bocheng Chen, and Houmin Yan. Optimal inventory decisions in a multiperiod newsvendor problem with partially observed markovian supply capacities. *European Journal of Operational Research*, 202(2):502–517, 2010.

[65] Bin Zhang and Shaofu Du. Multi-product newsboy problem with limited capacity and outsourcing. *European Journal of Operational Research*, 202(1):107–113, 2010.

[66] Peng Zhao and Bin Yu. On model selection consistency of lasso. *The Journal of Machine Learning Research*, 7:2541–2563, 2006.

APPENDICES

Appendix A: Code Implementation of Ice-Cream

The codes shows the simulation of a temperature dependent demand for ice-cream of the ice cream vendor problem. It also shows both the codes for the linear parametric model model and the regularized model.

A.1 Embedded Code Example – Python

```
#!/usr/bin/env python
# coding: utf-8

get_ipython().run_line_magic('matplotlib', 'inline')
from matplotlib import pyplot as plt

import simpy
import collections
import random
import numpy as np
import statistics
import math

import pandas as pd

from random import choice
import csv
import pandas as pd
pd.set_option('display.max_rows',None)

ls *.exe

  """An ice cream vendor must order ice cream and hot
     chocolate to fulfil
demand D (during the day). Ice cream cost c1 dollars per
   serving while
hot chocolate costs c2 dollars per serving. Any ice cream
   remaining at the
end of the day can be salvaged for s1 per serving and hot
   chocolate
salvaged for s2 per serving.
```

```
If the demand D exceeds the amount of ice cream or hot
   chocolate, then
the vendor issues a    coupon    which costs additional b1
    and b2 dollars per
item respectively.
.
 """

Vendor = collections.namedtuple('Vendor', 'counter, 
   product, available, '
                                                'sold_out, 
                                                   when_sold_out
                                                   , '
                                                'num_renegers'
                                                   )
c = 1
h = 3
b = 7

# monitor for capture data for simulation
Data = []

def customer(env, product, num_tickets, vendor):

    with vendor.counter.request() as my_turn:
        # Wait until its our turn or until the product is
           sold out
        result = yield my_turn | vendor.sold_out[product]

        # Check if it's our turn or if product is sold out
        if my_turn not in result:
            vendor.num_renegers[product] += num_tickets
            return

        # Check if enough tickets left.
        if vendor.available[product] < num_tickets:
            # Customer leaves after some discussion
            vendor.num_renegers[product] += num_tickets

            yield env.timeout(0)
            return

        # Buy tickets
        vendor.available[product] -= num_tickets
        if vendor.available[product] < 2:
            # Trigger the "sold out" event for the product
            vendor.sold_out[product].succeed()
            vendor.when_sold_out[product] = env.now
            vendor.available[product] = 0
        yield env.timeout(0)
```

```
Demand = {"Ice␣Cream" : 0, "Hot␣Chocolate": 0}
def customer_arrivals(env, vendor,pr_rain):
    """Create new *customer* until the sim time reaches
       3600."""
    while True:
        ### FEATURES INCORPORATED HERE
        ### Generate Temperature Dependent Demand

        ### FEATURES GENERATE "TIME TO NEXT ARRIVAL"
        ##Default: 0.5 minutes on average ==>  600
           arrivals per 300 minutes
        ## Assume a demand of 600 per day, where a day is
           5 hours
        ## D =600/5 ## demand per day
        DailyDemand = 600 ## per 5hrs (per day)
        # t = 0, Demand is 120 per hour
        # t = 5 hrs, Demand is 130 per hour


        t= env.now
        Tmax = 74 #maximum temp. for a given day
        Tmin = 36 #minimum temp. for a given day
        M = 55 # average temp. for a given day
        A = -19 # Amplitude (M+A =Tmax, M-A =Tmin)
           = math.pi/12  ##
           = 11 ## time at which max temp occurs
        Temp =  M+ A*np.cos(  *(t-  ))      ##Temp(t)
        m = 2 ##(130-120)/(5-0)
        pr_rain = 0.0
        D = 600/5 +m*(Temp - Tmin)
        D =(1-pr_rain)*D

        # data capture
        Data.append([Temp,pr_rain,D])


        TimeToNextArrival = np.random.exponential(60/D)  #
           # E(TimeToNextArrival) = 0.5 minutes
        yield env.timeout(TimeToNextArrival)

        product = "Ice␣Cream" #random.choice(vendor.
           product)
        num_tickets = 1 #random.randint(1, 2)
        Demand[product] += num_tickets
        if vendor.available[product]:
            env.process(customer(env, product, num_tickets
               , vendor))
        else:
```

```
            vendor.num_renegers[product] += num_tickets


def runSim(TICKETS):
    env = simpy.Environment()
    SIM_TIME = 300
    pr_rain = 0.0

    global Demand
    Demand = {"Ice Cream" : 0, "Hot Chocolate": 0}

    counter = simpy.Resource(env, capacity=2)
    products = ['Ice Cream', 'Hot Chocolate']
    available = {product: TICKETS for product in products}
    sold_out = {product: env.event() for product in
       products}
    when_sold_out = {product: None for product in products
       }
    num_renegers = {product: 0 for product in products}

    vendor = Vendor(counter, products, available, sold_out
       , when_sold_out,
                    num_renegers)

    env.process( customer_arrivals(env, vendor, pr_rain) )
    env.run(until = env.now + SIM_TIME)

    Cost = 0
    Underage = {"Ice Cream" : 0, "Hot Chocolate": 0}
    Overage = {"Ice Cream" : 0, "Hot Chocolate": 0}
    for product in products:
        Underage[product] = Demand[product] - TICKETS
        Overage[product] = 0
        if (Underage[product] < 0):
            Overage[product] = -Underage[product]
            Underage[product] = 0
        Cost += h*Overage[product] + b*Underage[product]

    return Cost


Demand

#%%timeit
#Order Quantity of 100
AveCost = np.array( [ runSim(100) for _ in range(100) ]).
   mean()
AveCost
```

```
from tqdm.notebook import tqdm, trange

AveCost = []
for order_quantity in trange(200,800,2):
    AveCost.append( np.array( [ runSim(order_quantity) for
        _ in range(10) ]).mean() )
#AveCost

np.array(AveCost).mean()

plt.plot(range(200,800,2), AveCost);

CF = (b-c)/(b+h)
CF


xstar_emp = 2*np.argmin(AveCost)+200
xstar_emp


I0 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
I0 = I0[:600]
I0
I0.to_csv('I0.csv',index=False,header = True)

I2 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
I2 = I2[:600]
I2
I2.to_csv('I2.csv',index=False,header = True)


I4 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
I4 = I4[:600]
I4
I4.to_csv('I4.csv',index=False,header = True)


I6 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
I6 = I6[:600]
I6
I6.to_csv('I6.csv',index=False,header = True)


I8 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
I8 = I8[:600]
I8
I8.to_csv('8.csv',index=False,header = True)
```

```
I8 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
I8 = I8[:600]
I8
I8.to_csv('I8.csv',index=False,header = True)


# In[137]:


IC = pd.concat([I0,I2,I4,I6,I8],ignore_index=True)
IC
IC.to_csv('IC.csv',index=False,header = True)


IC=pd.read_csv('IC_all.csv')
IC_demand =IC['D']
IC_demand

def ToGLPKdat(Df):

    """Df should be a DataFrame


    Quick and Dirty dat writer for Matrix Game

    GLPK model. Df is Payoff Matrix."""

    Lines = 'set␣DEMAND␣:=␣'
    for row in Df.index:
        Lines += '%s␣' % row
    Lines += ';\nset␣FEATURES␣:=␣'
    for col in Df.columns[0:-1]:
        Lines += '%s␣' % col
    Lines +=';\n\nparam␣F:␣'
    for row in Df.index:
        Lines += '%s␣' % row
    Lines += ':='
    for col in Df.columns[0:-1]:
        Lines += '\n%s␣' % col
        for row in Df.index:
            Lines += '%s␣' % Df.loc[row,col]
    Lines += ';␣\n␣\nparam␣D1:␣'
    for col in Df.columns[-1]:
        Lines += '%s' % col
    Lines += ':=␣'
    Lines += '\n%s␣' % Df['D']
```

```
    Lines += ';␣\n␣\nend;'
    with open('cream.dat','w') as FileObject:
        print('Writing␣to␣cream.dat')
        FileObject.write(Lines)

    return Lines


print(ToGLPKdat(IC))


### LINEAR PARAMETRIC MODEL FOR ICE-CREAM--- ESTIMATING
   THE MINIMUM EXPECTED COST

%%script glpsol -m /dev/stdin -d cream.dat

set FEATURES;
set DEMAND;

##parameters

param co :=0.25;
param cu := 0.33 ;
param D{DEMAND} >= 0;
param F{FEATURES, DEMAND} >= 0;

## variables

var Beta{FEATURES} >= 0;
var S{DEMAND} >= 0;
var T{DEMAND} >= 0;
var v{FEATURES} >=0;


##Objective
minimize totalCost : sum{d in DEMAND} (co*S[d] + cu*T[d])
    ;
    s.t. Underage{d in DEMAND}:
        S[d] >= D[d] - sum{f in FEATURES} F[f,d]*Beta[f];

    s.t. Overage{d in DEMAND}:
        T[d] >= sum{f in FEATURES}F[f,d]*Beta[f] -D[d];

    s.t. lowerbound{f in FEATURES}:
        Beta[f] >=-v[f];
    s.t. upperbound{f in FEATURES}:
        Beta[f] <=v[f];

solve;
```

```
##display
display Beta;

for {f in FEATURES}:

printf "\ntotalCost = %.3f\n\n", totalCost;



end;


### USING CROSS-VALIDATION TO FIND THE APPROPRIATE TUNING
   PARAMETER FOR THE L1-REGULARIZATION

from sklearn.linear_model import Lasso
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
import pandas as pd
import csv


x_trainIC,x_testIC,y_trainIC,y_testIC = train_test_split(
   IC,IC_demand, test_size = 0.3)

## Using Grid Search to find a suitable tuning parameter

LS = Lasso(random_state=0, max_iter=10000)  ## model
alphas= [ 1e-5,1e-4, 1e-3,1e-2, 1, 5, 10,50,100,1000]  ##
   Define the alpha values to test


tuned_parameters = [{'alpha': alphas}]
clf = GridSearchCV(LS, tuned_parameters, cv=5,
   return_train_score = False)
clf.fit(IC,IC_demand)
clf.cv_results_


scores = clf.cv_results_['mean_test_score']
scores

IC=pd.DataFrame(clf.cv_results_)
IC

IC[['param_alpha','mean_test_score']]
## appropriate tuning parameter for the data can be= [1e
   -5,1e-4, 1e-3,1e-2]
```

```
### THE L-1 REGULARIZATION MODEL - ICE-CREAM
%%script glpsol -m /dev/stdin -d cream.dat

set FEATURES;
set DEMAND;

##parameters

param D{DEMAND} >= 0;
param F{FEATURES, DEMAND} >= 0;
###
param co := 1.62 ;
param cu := 1.40;
param L := 0.01;

## variables

var Beta{FEATURES} >= 0;
var S{DEMAND} >= 0;
var T{DEMAND} >= 0;
var v{FEATURES} >=0;
var U;

##Objective
minimize totalCost : sum{d in DEMAND} (co*S[d] + cu*T[d])
   +L*U ;
     s.t. Underage{d in DEMAND}:
         S[d] >= D[d] - sum{f in FEATURES} F[f,d]*Beta[f];

     s.t. Overage{d in DEMAND}:
         T[d] >= sum{f in FEATURES}F[f,d]*Beta[f] -D[d];

     s.t. lowerbound{f in FEATURES}:
         Beta[f] >=-v[f];
     s.t. upperbound{f in FEATURES}:
         Beta[f] <=v[f];
     s.t. sumofV:
          sum{f in FEATURES} v[f] = U;
solve;


##display
display Beta;

for {f in FEATURES}:

printf "\ntotalCost = %.3f\n\n", totalCost;
```

```
end;
```

Appendix B: Code Implementation of Hot Chocolate

The codes shows the simulation of a temperature dependent demand for hot chocolate of the ice cream vendor problem. It also shows both the codes for the linear parametric model model and the regularized model.

```
#!/usr/bin/env python
# coding: utf-8

get_ipython().run_line_magic('matplotlib', 'inline')
from matplotlib import pyplot as plt

import simpy
import collections
import random
import numpy as np
import statistics
import math

import pandas as pd

from random import choice
import csv
import pandas as pd
pd.set_option('display.max_rows',None)

ls *.exe


  """An ice cream vendor must order ice cream and hot
     chocolate to fulfil
demand D (during the day). Ice cream cost c1 dollars per
   serving while
hot chocolate costs c2 dollars per serving. Any ice cream
   remaining at the
end of the day can be salvaged for s1 per serving and hot
   chocolate
salvaged for s2 per serving.

If the demand D exceeds the amount of ice cream or hot
   chocolate, then
the vendor issues a    coupon    which costs additional b1
    and b2 dollars per
item respectively.
.
 """
```

```
Vendor = collections.namedtuple('Vendor', 'counter, product, available, '
                                          'sold_out, when_sold_out, '
                                          'num_renegers')
c = 1
h = 3
b = 7

# monitor for capture data for simulation
Data = []

def customer(env, product, num_tickets, vendor):

    with vendor.counter.request() as my_turn:
        # Wait until its our turn or until the product is sold out
        result = yield my_turn | vendor.sold_out[product]

        # Check if it's our turn or if product is sold out
        if my_turn not in result:
            vendor.num_renegers[product] += num_tickets
            return

        # Check if enough tickets left.
        if vendor.available[product] < num_tickets:
            # Customer leaves after some discussion
            vendor.num_renegers[product] += num_tickets

            yield env.timeout(0)
            return

        # Buy tickets
        vendor.available[product] -= num_tickets
        if vendor.available[product] < 2:
            # Trigger the "sold out" event for the product
            vendor.sold_out[product].succeed()
            vendor.when_sold_out[product] = env.now
            vendor.available[product] = 0
        yield env.timeout(0)

Demand = {"Ice Cream" : 0, "Hot Chocolate": 0}
def customer_arrivals(env, vendor,pr_rain):
    """Create new *customer* until the sim time reaches
       3600."""
    while True:
        ### FEATURES INCORPORATED HERE
```

```
        ### Generate Temperature Dependent Demand

        ### FEATURES GENERATE "TIME TO NEXT ARRIVAL"
        ##Default: 0.5 minutes on average ==>  600
           arrivals per 300 minutes
        ## Assume a demand of 600 per day, where a day is
           5 hours
        ## D =600/5 ## demand per day
        DailyDemand = 600 ## per 5hrs (per day)
        # t = 0, Demand is 120 per hour
        # t = 5 hrs, Demand is 130 per hour


        t= env.now
        Tmax = 74 #maximum temp. for a given day
        Tmin = 36 #minimum temp. for a given day
        M = 55 # average temp. for a given day
        A = 19 # Amplitude (M+A =Tmax, M-A =Tmin)
           = math.pi/12  ##
           = 11 ## time at which max temp occurs
        Temp =  M+ A*np.cos(  *(t-  ))      ##Temp(t)
        m = 2 ##(130-120)/(5-0)
        pr_rain = 0.0
        D = 600/5 +m*(Temp - Tmin)
        D =(1-pr_rain)*D

        # data capture
        Data.append([Temp,pr_rain,D])


        TimeToNextArrival = np.random.exponential(60/D)  #
           # E(TimeToNextArrival) = 0.5 minutes
        yield env.timeout(TimeToNextArrival)

        product = "Hot␣Chocolate" #random.choice(vendor.
           product)
        num_tickets = 1 #random.randint(1, 2)
        Demand[product] += num_tickets
        if vendor.available[product]:
            env.process(customer(env, product, num_tickets
               , vendor))
        else:
            vendor.num_renegers[product] += num_tickets


def runSim(TICKETS):
    env = simpy.Environment()
    SIM_TIME = 300
    pr_rain = 0.0
```

```
    global Demand
    Demand = {"Ice␣Cream" : 0, "Hot␣Chocolate": 0}

    counter = simpy.Resource(env, capacity=2)
    products = ['Ice␣Cream', 'Hot␣Chocolate']
    available = {product: TICKETS for product in products}
    sold_out = {product: env.event() for product in
       products}
    when_sold_out = {product: None for product in products
       }
    num_renegers = {product: 0 for product in products}

    vendor = Vendor(counter, products, available, sold_out
       , when_sold_out,
                    num_renegers)

    env.process( customer_arrivals(env, vendor, pr_rain) )
    env.run(until = env.now + SIM_TIME)

    Cost = 0
    Underage = {"Ice␣Cream" : 0, "Hot␣Chocolate": 0}
    Overage = {"Ice␣Cream" : 0, "Hot␣Chocolate": 0}
    for product in products:
        Underage[product] = Demand[product] - TICKETS
        Overage[product] = 0
        if (Underage[product] < 0):
            Overage[product] = -Underage[product]
            Underage[product] = 0
        Cost += h*Overage[product] + b*Underage[product]

    return Cost

Demand

#%%timeit
#Order Quantity of 100
AveCost = np.array( [ runSim(100) for _ in range(100) ]).
   mean()
AveCost


from tqdm.notebook import tqdm, trange

AveCost = []
for order_quantity in trange(200,800,2):
    AveCost.append( np.array( [ runSim(order_quantity) for
        _ in range(10) ]).mean() )
```

```
#AveCost

np.array(AveCost).mean()

plt.plot(range(200,800,2), AveCost);

CF = (b-c)/(b+h)
CF

xstar_emp = 2*np.argmin(AveCost)+200
xstar_emp


H0 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
H0 = H0[:600]
H0
H0.to_csv('H0.csv',index=False,header = True)

H2 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
H2 = H2[:600]
H2
H2.to_csv('H2.csv',index=False,header = True)


H4 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
H4 = H4[:600]
H4
H4.to_csv('H4.csv',index=False,header = True)


H6 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
H6 = H6[:600]
H6
H6.to_csv('H6.csv',index=False,header = True)

H8 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
H8 = H8[:600]
H8
H8.to_csv('H8.csv',index=False,header = True)


HC = pd.concat([H0,I2,I4,I6,I8],ignore_index=True)
HC
HC.to_csv('HC.csv',index=False,header = True)

H8 = pd.DataFrame(Data, columns= ['Temp', 'pr_rain','D'])
H8 = H8[:600]
H8
H8.to_csv('H8.csv',index=False,header = True)
```

```
HC = pd.concat([H0,H2,H4,H6,H8],ignore_index=True)
HC
HC.to_csv('HC.csv',index=False,header = True)

## GENERATING A GLPK DATAFILE FOR HOT CHOCOLATE


def ToGLPKdat(Df):

    """Df should be a DataFrame"""

    Lines = 'set␣DEMAND␣:=␣'
    for row in Df.index:
        Lines += '%s␣' % row
    Lines += ';\nset␣FEATURES␣:=␣'
    for col in Df.columns[0:-1]:
        Lines += '%s␣' % col
    Lines +=';\n\nparam␣F:␣'
    for row in Df.index:
        Lines += '%s␣' % row
    Lines += ':='
    for col in Df.columns[0:-1]:
        Lines += '\n%s␣' % col
        for row in Df.index:
            Lines += '%s␣' % Df.loc[row,col]
    Lines += ';␣\n␣\nparam␣D1:␣'
    for col in Df.columns[-1]:
        Lines += '%s' % col
    Lines += ':=␣'
    Lines += '\n%s␣' % Df['D']

    Lines += ';␣\n␣\nend;'
    with open('coco.dat','w') as FileObject:
        print('Writing␣to␣coco.dat')
        FileObject.write(Lines)

    return Lines

print(ToGLPKdat(HC))

### LINEAR PARAMETRIC MODEL FOR HOT CHOLATE--- ESTIMATING
   THE MINIMUM EXPECTED COST

%%script glpsol -m /dev/stdin -d coco.dat

set FEATURES;
set DEMAND;
```

```
##parameters

param co :=0.25;
param cu := 0.33 ;
param D{DEMAND} >= 0;
param F{FEATURES, DEMAND} >= 0;

## variables

var Beta{FEATURES} >= 0;
var S{DEMAND} >= 0;
var T{DEMAND} >= 0;
var v{FEATURES} >=0;


##Objective
minimize totalCost : sum{d in DEMAND} (co*S[d] + cu*T[d])
    ;
    s.t. Underage{d in DEMAND}:
        S[d] >= D[d] - sum{f in FEATURES} F[f,d]*Beta[f];

    s.t. Overage{d in DEMAND}:
        T[d] >= sum{f in FEATURES}F[f,d]*Beta[f] -D[d];

    s.t. lowerbound{f in FEATURES}:
        Beta[f] >=-v[f];
    s.t. upperbound{f in FEATURES}:
        Beta[f] <=v[f];

solve;


##display
display Beta;

for {f in FEATURES}:

printf "\ntotalCost = %.3f\n\n", totalCost;

end;

### USING CROSS-VALIDATION TO FIND THE APPROPRIATE TUNING
   PARAMETER FOR THE L1-REGULARIZATION

from sklearn.linear_model import Lasso
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
import pandas as pd
import csv
```

```
### USING CROSS-VALIDATION TO FIND THE APPROPRIATE TUNING
   PARAMETER FOR HOT CHOCOLATE
HC=pd.read_csv('HC_all.csv')
HC_demand =HC['D']
HC_demand


x_trainHC,x_testHC,y_trainHC,y_testHC = train_test_split(
   HC,HC_demand, test_size = 0.3)

## Using Grid Search to find a suitable tuning parameter

LS = Lasso(random_state=0, max_iter=10000)  ## model
alphas= [ 1e-5,1e-4, 1e-3,1e-2, 1, 5, 10,50,100,1000]  ##
   Define the alpha values to test


tuned_parameters = [{'alpha': alphas}]
clf = GridSearchCV(LS, tuned_parameters, cv=5,
   return_train_score = False)
clf.fit(HC,HC_demand)
clf.cv_results_

hC=pd.DataFrame(clf.cv_results_)
h
hC[['param_alpha','mean_test_score']]
## appropriate tuning parameter for the data can be= [1e
   -5,1e-4, 1e-3,1e-2]


### PARAMETRIC NEWSVENDOR MODEL FOR HOT CHOCOLATE WITH L1-
   REGULARIZATION

%%script glpsol -m /dev/stdin -d cream.dat

set FEATURES;
set DEMAND;

##parameters
param D{DEMAND} >= 0;
param F{FEATURES, DEMAND} >= 0;

param co := 1.62 ;
param cu := 1.40;
param L := 0.01;

## variables
```

```
var Beta{FEATURES} >= 0;
var S{DEMAND} >= 0;
var T{DEMAND} >= 0;
var v{FEATURES} >=0;
var U;

##Objective
minimize totalCost : sum{d in DEMAND} (co*S[d] + cu*T[d])
   +L*U ;
    s.t. Underage{d in DEMAND}:
        S[d] >= D[d] - sum{f in FEATURES} F[f,d]*Beta[f];

    s.t. Overage{d in DEMAND}:
        T[d] >= sum{f in FEATURES}F[f,d]*Beta[f] -D[d];

    s.t. lowerbound{f in FEATURES}:
        Beta[f] >=-v[f];
    s.t. upperbound{f in FEATURES}:
        Beta[f] <=v[f];
    s.t. sumofV:
         sum{f in FEATURES} v[f] = U;
solve;


##display
display Beta;

for {f in FEATURES}:

printf "\ntotalCost = %.3f\n\n", totalCost;

end;
```

Appendix C: Code Implementation of the Substitution Model

The codes shows the implementation of the substitution model.

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


get_ipython().run_line_magic('matplotlib', 'inline')
import matplotlib.pyplot as plt
import numpy as np

import pandas as pd
pd.set_option('display.max_rows',None)
import csv

ls *.exe

##Rearranging columns for the temperature related demand
   for Ice-Cream and Hot Chocolate
ICE = pd.read_csv('IC_all.csv')
ICE.rename(columns = {'D':'HOT_Demand'}, inplace = True)
print(ICE)

HOT = pd.read_csv('HC_all.csv')
HOT.rename(columns = {'D':'HOT_Demand'}, inplace = True)
print(HOT)

### COMBINING THE DATA FOR ICECREAM AND HOT CHOCOLATE
DATA= pd.concat([ICE,HOT],axis=1)
np.unique(A,return_index=True)


def ToGLPKdat(Df):

    """Df should be a DataFrame"""

    Lines = 'set␣DEMAND␣:=␣'
    for row in Df.index:
        Lines += '%s␣' % row
    Lines += ';\nset␣FEATURES␣:=␣'
    for col in Df.columns[0:-2]:
        Lines += '%s␣' % col
    Lines +=';\n\nparam␣F:␣'
    for row in Df.index:
        Lines += '%s␣' % row
    Lines += ':='
```

```
    for col in Df.columns[0:-2]:
        Lines += '\n%s ' % col
        for row in Df.index:
            Lines += '%s ' % Df.loc[row,col]
    Lines += '; \n \nparam D1: '
    for col in Df.columns[-2]:
        Lines += '%s' % col
    Lines += ':= '
    Lines += '\n%s ' % Df['ICE_Demand']
    Lines += '; \n \nparam D2: '
    for col in Df.columns[-1]:
        Lines += '%s' % col
    Lines += ':= '
    Lines += '; \n \nend;'
    Lines += '\n%s ' % Df['HOT_Demand']
    Lines += '; \n \nend;'
    with open('alldata.dat','w') as FileObject:
        print('Writing to alldata.dat')
        FileObject.write(Lines)

    return Lines


print(ToGLPKdat(alldata))

%%script glpsol -m /dev/stdin -d alldata.dat

set FEATURES;
set DEMAND;


##parameters
## IC =icecream , HC = Hot Chocolate
### D1 = Demand for Icecream, D2 = Demand for Hot
   Chocolate
# coIC = Overage cost (Icecream)
#cuIC = underage cost (Icecream)
#c1 = some small % of the demand for icecream
#c2 = some small % of the demand for hot chocolate


param coIC := 0.2;
param cuIC := 0.33;
param coHC :=  0.2;
param cuHC := 0.33;
param L1 := 0.001;
param L2 := 0.001;
param c1:= 0.05;
param c2:= 0.05;
```

```
param D1{DEMAND} >= 0;
param D2{DEMAND} >= 0;
param F{FEATURES, DEMAND} >= 0;

## variables
var Beta{FEATURES} >= 0;
var S1{DEMAND} >= 0;
var T1{DEMAND} >= 0;
var S2{DEMAND} >= 0;
var T2{DEMAND} >= 0;
var v1{FEATURES} >=0;
var v2{FEATURES} >=0;
var U1;
var U2;
var x >= 0;
##Objective
minimize totalCost : sum{d in DEMAND} (coIC*S1[d] + cuIC*
   T1[d])+L1*U1  + sum{d in DEMAND}(coHC*S2[d] + cuHC*T2[d
   ]) +L2*U2 ;
    s.t. Underage1{d in DEMAND}:
        S1[d] >= D1[d] - c1*(D2[d]- sum{f in FEATURES} F[f
           ,d]*Beta[f]) -sum{f in FEATURES} F[f,d]*Beta[f
           ];

    s.t. Overage1{d in DEMAND}:
        T1[d] >= sum{f in FEATURES}F[f,d]*Beta[f] -D1[d] +
            c1*(D2[d]- sum{f in FEATURES} F[f,d]*Beta[f]);

    s.t. Underage2{d in DEMAND}:
        S2[d] >= D2[d] - c2*(D1[d]- sum{f in FEATURES} F[f
           ,d]*Beta[f]) -sum{f in FEATURES} F[f,d]*Beta[f
           ];

    s.t. Overage2{d in DEMAND}:
        T2[d] >= sum{f in FEATURES}F[f,d]*Beta[f] -D2[d] +
            c2*(D1[d]- sum{f in FEATURES} F[f,d]*Beta[f]);

    s.t. lowerboundIC{f in FEATURES}:
        Beta[f] >=-v1[f];
    s.t. upperboundIC{f in FEATURES}:
        Beta[f] <=v1[f];
    s.t. sumofvIC:
         sum{f in FEATURES} v1[f] = U1;
    s.t. LowerboundHC{f in FEATURES}:
        Beta[f] >=-v2[f];
    s.t. UpperboundHC{f in FEATURES}:
        Beta[f] <=v2[f];
    s.t. sumofvHC:
```

```
          sum{f in FEATURES} v2[f] = U2;
solve;

display Beta;

for {f in FEATURES}:
printf "\ntotalCost = %.3f\n\n", totalCost;

end;
```

VITA

MAKAFUI AMA AZASOO

Education:	B.S. Actuarial Science, University for Development Studies, Tamale, 2016
	M.S. Mathematical Sciences, East Tennessee State University Johnson City, Tennessee 2021
Professional Experience:	Intern, Junior Data Analyst, Burkina Faso, Summer-2015
	Data Update Officer, S.S.N.I.T, Ghana, 2016-2017
	Service Excellence, Barclays Bank Gh. Ltd, Ghana, 2018-2019
	Graduate Assistant, East Tennessee State University, Johnson City, Tennessee, 2019-2021