INTRODUCTION

1.1 INTRODUCTION TO SCM

Supply chain management (SCM) is the oversight of materials, information, and finances as they move in a process from supplier to manufacturer to wholesaler to retailer to consumer. Supply chain management involves coordinating and integrating these flows both within and among companies. It is said that the ultimate goal of any effective supply chain management system is to reduce inventory. As a solution for successful supply chain management, sophisticated software systems with Web interfaces are competing with Web-based application service providers (ASP) who promise to provide part or all of the SCM service for companies who rent their service.

A supply chain is the collection of steps that a company takes to transform raw components into the final product. Typically, supply chain management is comprised of five stages: plan, develop, make, deliver, and return.

1. **Plan:**

   A plan or strategy must be developed to address how a given good or service will meet the needs of the customers. A significant portion of the strategy should focus on planning a profitable supply chain. A big piece of SCM planning is developing a set of metrics to monitor the supply chain so that it is efficient, costs less and delivers high quality and value to customers.

2. **Develop:**

   It involves building a strong relationship with suppliers of the raw materials needed in making the product the company delivers. This phase involves not only identifying reliable suppliers but also planning methods for shipping, delivery, and payment. SCM managers can put together processes for managing their goods and services inventory, including receiving and verifying shipments, transferring them to the manufacturing facilities and authorizing supplier payments.

3. **Make:**

   At this stage the product is manufactured, tested, packaged, and scheduled for delivery. This is the manufacturing step. This is the most metric-intensive portion of the supply chain—one where companies are able to measure quality levels, production output and worker productivity.

4. **Deliver:**

   This stage, at the logistics phase, customer orders are received and delivery of the goods is planned. This is the part that many SCM insiders refer to as logistics, where companies coordinate the receipt of orders from customers, develop a network of warehouses, pick carriers to get products to customers and set up an invoicing system to receive payments.

5. **Return:**

   As the name suggests, during this stage, customers may return defective products. The company will also address customer questions in this stage. This can be a problematic part of the supply chain for many companies. Supply chain planners have to create a responsive and flexible network for receiving defective and excess products back from their customers and supporting customers who have problems with delivered products.

This project in the Supply Chain Management will provide us an understanding of key areas of logistics and supply chain management where relevant analysis of data needed. This will focus on several key supply chain functions and analyse data that may be available for the supply chain. This will be done through Excel-based approaches for analysing. If a company expects to achieve benefits from their supply chain management process, they will require some level of investment in technology. The backbone for many large companies has been the vastly expensive Enterprise Resource Planning (ERP) suites, such as SAP and Oracle.

Since the wide adoption of Internet technologies, all businesses can take advantage of Web-based software and Internet communications. Instant communication between vendors and customers allows for timely updates of information. The key in management of the today's supply chain emphasize on cutting costs and streamlining expenses of many companies .

1.2 NEED FOR SUPPLY CHAIN MANAGEMENT

Many people involved with companies don't have a clear understanding of what a supply chain is or how it fits into the company's overall strategy. Supply Chain Management (SCM) software can have tremendous financial benefits for companies Supply chains include a company's entire manufacturing and distribution process. They involve every step of the production from planning to manufacturing to handling defective goods. The overall goal of these chains is to keep the process running smoothly at all times and to keep all of the components (vendors, warehouses, etc.) connected.

It is well known that supply chain management is an integral part of most businesses and is essential to company success and customer satisfaction.

- **Boost Customer Service**

  • Customers expect the correct product assortment and quantity to be delivered.

  • Customers expect products to be available at the right location.

  • Right Delivery Time – Customers expect products to be delivered on time.

  • Right After Sale Support – Customers expect products to be serviced quickly.

- **Reduce Operating Costs**

  • Decreases Purchasing Cost – Retailers depend on supply chains to quickly deliver expensive products to avoid holding costly inventories in stores any longer than necessary.

  • Decreases Production Cost – Manufacturers depend on supply chains to reliably deliver materials to assembly plants to avoid material shortages that would shut down production. Decreases Total Supply Chain Cost – Manufacturers and retailers depend on supply chain managers to design networks that meet customer service goals at the least total cost. Efficient supply chains enable a firm to be more competitive in the market place

- **Improve Financial Position**

  • Increases Profit Leverage – Firms value supply chain managers because they help control and reduce supply chain costs. This can result in dramatic increases in firm profits.

  • Decreases Fixed Assets – Firms value supply chain managers because they decrease the use of large fixed assets such as plants, warehouses and transportation vehicles in the supply chain. Increases Cash Flow – Firms value supply chain managers because they speed up product flows to customers.

  • Lesser known, is how supply chain management also plays a critical role in society. SCM knowledge and capabilities can be used to support medical missions, conduct disaster relief operations, and handle other types of emergencies.

Whether dealing with day-to-day product flows or dealing with an unexpected natural disaster, supply chain experts roll up their sleeves and get busy. They diagnose problems, creatively work around disruptions, and figure out how to move essential products to people in need as efficiently as possible.

- **Ensure Human Survival**

  • SCM Helps Sustains Human Life – Humans depend on supply chains to deliver basic necessities such as food and water. Any breakdown of these delivery pipelines quickly threatens human life

  • SCM Improves Human Healthcare – Humans depend on supply chains to deliver medicines and healthcare. During a medical emergency, supply chain performance can be the difference between life and death.

  • SCM Protects Humans from Climate Extremes – Humans depend on an energy supply chain to deliver electrical energy to homes and businesses for light, heat, refrigeration and air conditioning. Logistical failure (a power blackout) can quickly result in a threat to human life.

- **Improve Quality of Life**

  • Foundation for Economic Growth – Societies with a highly developed supply chain infrastructure (modern interstate highway system, vast railroad network, numerous modern ports and airports) are able to exchange many goods between businesses and consumers quickly and at low cost. As a result, the economy grows. In fact, the one thing that most poor nations have in common is no or a very poorly developed supply chain infrastructure.

  • Improves Standard of Living – Societies with a highly developed supply chain infrastructure (modern interstate highway system, vast railroad network, numerous modern ports and airports) are able to exchange many goods between businesses and consumers quickly and at low cost. As a result, consumers can afford to buy more products with their income thereby raising the standard of living in the society

  • Job Creation – Supply chain professionals design and operate all of the supply chains in a society and manage transportation, warehousing, inventory management, packaging and logistics information

  • Opportunity to Decrease Energy Use – Supply chain activities involve both human and product transportation. As a by-product of these activities, scarce energy is depleted.

1.3 AIM OF THIS PROJECT

Aim of this project is to study Supply Chan application at L&T Construction & Mining Business to improve Customer support/fulfilment at after sales market and organizational performance, identifying challenges to balance demand and supplies, analysing data to have clear visibility and to stream lining its operations through Supply Chain Analytics.

CHAPTER 2

LITERATURE REVIEW

2.1 HISTORY OF SUPPLY CHAIN MANAGEMENT

Over the last 100 plus years of the history of supply chain management has evolved from an initial focus on improving relatively simple, but very labour-intensive processes to the present-day engineering and managing of extraordinarily complex global networks

In the 1940s and 1950s, the focus of logistics research was on how to use mechanisation to improve the very labour intensive processes of material handling and how to take better advantage of space using racking and better warehouse design and layout. In the mid 1950 s, this concept was extended transportation management with the development of intermodal containers together with ships, trains, and trucks to handle these containers.

By the 1960s, a clear trend had developed in shifting more time-dependent freight transportation to truck rather than rail. This led to the need for joint consideration of warehousing, material handling, and freight transportation, which emerged under the label of "Physical Distribution." The National Council of Physical Distribution Management was formed in 1963 to focus industry attention on this area and quickly became the predominant organization in the field.

Prior to the 1960s, virtually all transactions and record keeping were done manually. The computerization of this data opened the door to a huge opportunity for innovations in logistics planning, from randomized storage in warehouses to optimization of inventory and truck routing.

The 1980s marked the beginning of a sea-change in logistics in the history of supply chain management. The emergence of personal computers in the early 1980s provided tremendously better computer access to planners and a new graphical environment for planning. Perhaps the most important trend for logistics in the 1980s was that it had begun to get tremendous recognition in industry as being very expensive, very important, and very complex. Company executives became aware of logistics as an area where they had the opportunity to significantly improve the bottom line if they were willing to invest in trained professionals and new technology.

The logistics boom was fuelled further in the 1990s by the emergence of Enterprise resource planning **(ERP)** systems. These systems were motivated in part by the successes achieved by Material Requirements Planning systems developed in the 1970s and 1980s, in part by the desire to integrate the multiple databases that existed in almost all companies and seldom talked to each other, and in part by concerns that existing systems might have catastrophic failures as a result of not being able to handle the year 2000 date. In spite of some significant problems in getting the ERP systems installed and working, by 2000 most large companies had installed ERP systems.

The widespread recognition of the term "supply chain" has come primarily as a result of the globalization of manufacturing since the mid 1990 s, particularly the growth of manufacturing in China. U.S. imports from China grew from about $45 billion per year in 1995 to more than $280 billion per year in 2006.

2.2 BASE PAPER / EXISTING SYSTEM

2.2.1 Base Paper

Present scenario competition between organizations is among supply chain, Global economic climate is that most of companies experiencing decline in gross margin due to lack of visibility in respective supply chain. Therefore, effective supply chain management plays key role in improving organizational performance.

Scope of this project is to study Supply Chan application at L&T Construction & Mining Business to improve Customer support at after sales market and organizational performance. After sales & service division deals with 25000 to 30000 SKUs to support 70 to 75 verity of Equipment’s like Wheel Loaders, Excavators, Dozers and Dump trucks. This is a real challenge to the organization to balance demand and supplies with optimum cost. This project thus analyses the data to have clear visibility and stream lining its operations through Supply Chain Analytics to respond them proactively, increasing the both efficiency and profitability. Further it stipulates the linkage between organizational information, alerts and key performance indicators and its management.

2.2.2 Existing system:

L&T Construction & Mining Business uses ERP - SAP ECC 6.0 (Modules – Sales and Distribution, Materials Management & Finance) and uses standard reports and few of customized reports. Data analysis is done outside SAP using MS Excel on case to case basis. It is a challenge to get right data at right time for taking right decision making.

2.3 PROPOSED SYSTEM:

It is recommended to overcome present challenges in data, proposed to use Analytical techniques to get right data at right time for right decision. There are many analytical software solutions are available, it is proposed to use to use Jupyter notebook with python pandas and Excel. It is proposed to take SAP dump in Excel and load data in Jupyter to fulfil the requirement of this project.

CHAPTER 3

SPECIFICATIONS

3.1 HARDWARE REQUIREMENTS

- PROCESSOR : Intel Pentium or Higher Version
- RAM : Minimum 1GB
- HARD DISK : 60GB and above

3.2 SOFTWARE REQUIREMENTS

- SOFTWARE : Python 2.7 or greater
- SUPPORTED BROWSERS : Google Chrome / Mozilla Firefox /Internet explorer
- Jupyter notebook using Anaconda installation
- Windows Vista* or Newer Version, or MACos, or Linux (32/64 bit)

3.3 FUNCTIONAL REQUIREMENTS

The Functional Requirements Specification documents the Operations and activities that a system must be able to perform. Functional Requirements include:

- Descriptions of data entered into the system
- Descriptions of work-flows performed by the system
- Descriptions of system reports or other outputs
- How the system meets applicable regulatory requirements
- An integrated system operating in or near actual real time without reliance on periodic updates
- A common data base, or a data source supporting all applications.
- A consistent look and feel amongst all modules
- System installation with elaborate application integration by the in-house IT department

The Functional Requirements Specifications is designed to be read by a general audience. Readers should understand the system, but no particular technical knowledge should be required to understand the document.

These are the functional requirements specification documents for the project analysis. A software requirement specification helps to attenuate the time and energy needed by the developers to attain their desired goals and additionally minimizes the value of development.

3.4 NON-FUNCTIONAL REQUIREMENTS

Each attribute may be accustomed measure of the product performance. These attributes may be used for Quality assurance similarly as quality control. Quality assurance activities are directed towards prevention of introduction of defects and internal control activities are aimed toward detecting defects in product and services.

1. Reliability

Measure if product is reliable enough to sustain in any condition. Give systematically correct results. Product dependability is measured in terms of operation of project underneath different operating atmosphere and different inputs.

2. Maintainability

Different versions of the product ought to be easy to maintain. For development, it ought to be easy to feature code to existing system, ought to be easy to upgrade for brand new options and new technologies time to time. Maintenance should be value effective and simple. System be easy to take care of and correcting defects or making a change within the software system.

3. Usability

This can be measured in terms of ease of use. Application should be user friendly. Easy to use for input preparation, operation and also for interpreting of output. Simple is the key here.

4. Scalability

This can be measured in terms of Costing issues, Technical issues related to porting, scalable enough to add new functionalities in future.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

System design is the first design stage for devising the basic approach to solving the problem. During system design, developers decide the overall structures and styles. The system architecture determines the organization of the system into subsystems. In addition, the architecture provides the context for the detailed decisions that are made in later stages. During design, developers make decisions about how the problem will be solved, first at the high level and then with more detail.

Proposed Model will make use of the following:

4.2 DATA ANALYTICS USING PANDAS DATA FRAME

Pandas is an open source, BSD-licensed library providing high performance, easy to use data structures and data analysis tool for the Python Programming language. Pandas is a NUMFocus sponsored project. This will help ensure the success of development of pandas as a world-class open-source project, and makes it possible to donate to the project.

Pandas Library Features

- A fast and efficient **Data Frame** object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different formats: CSV and text files, Microsoft Excel, SQL databases, and the fast HDF5 format.
- Intelligent **data alignment** and integrated handling of **missing data**: gain automatic label-based alignment in computations and easily manipulate messy data into an orderly form.
- Flexible **reshaping** and pivoting of data sets.
- Intelligent label-based **slicing**, **fancy indexing**, and **sub setting** of large data sets.

- **Time series-functionality**: date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging. Even create domain-specific time offsets and join time series without losing data.
- Highly **optimized for performance**, with critical code paths written in Python or C.
- Python with pandas is in use in a wide variety of **academic and commercial** domains, including Finance, Neuroscience, Economics, Statistics, Advertising, Web Analytics, and more.

4.3 Machine Learning

Machine learning is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed. Machine learning focuses on the development of computer programs that can change when exposed to new data.

The process of machine learning is similar to that of data mining. Both systems search through data to look for patterns. However, instead of extracting data for human comprehension -- as is the case in data mining applications -- machine learning uses that data to detect patterns in data and adjust program actions accordingly. Machine learning algorithms are often categorized as being supervised or unsupervised. Supervised algorithms can apply what has been learned in the past to new data. Unsupervised algorithms can draw inferences from datasets.

Scikit Learn for machine learning Built on NumPy, SciPy and matplotlib, this library contains a lot of efficient tools for machine learning and statistical modelling including classification, regression, and clustering and dimensionality reduction.

We need ML in cases where we cannot directly write a program to handle every case so it's better to have a machine that learns from a large training set.

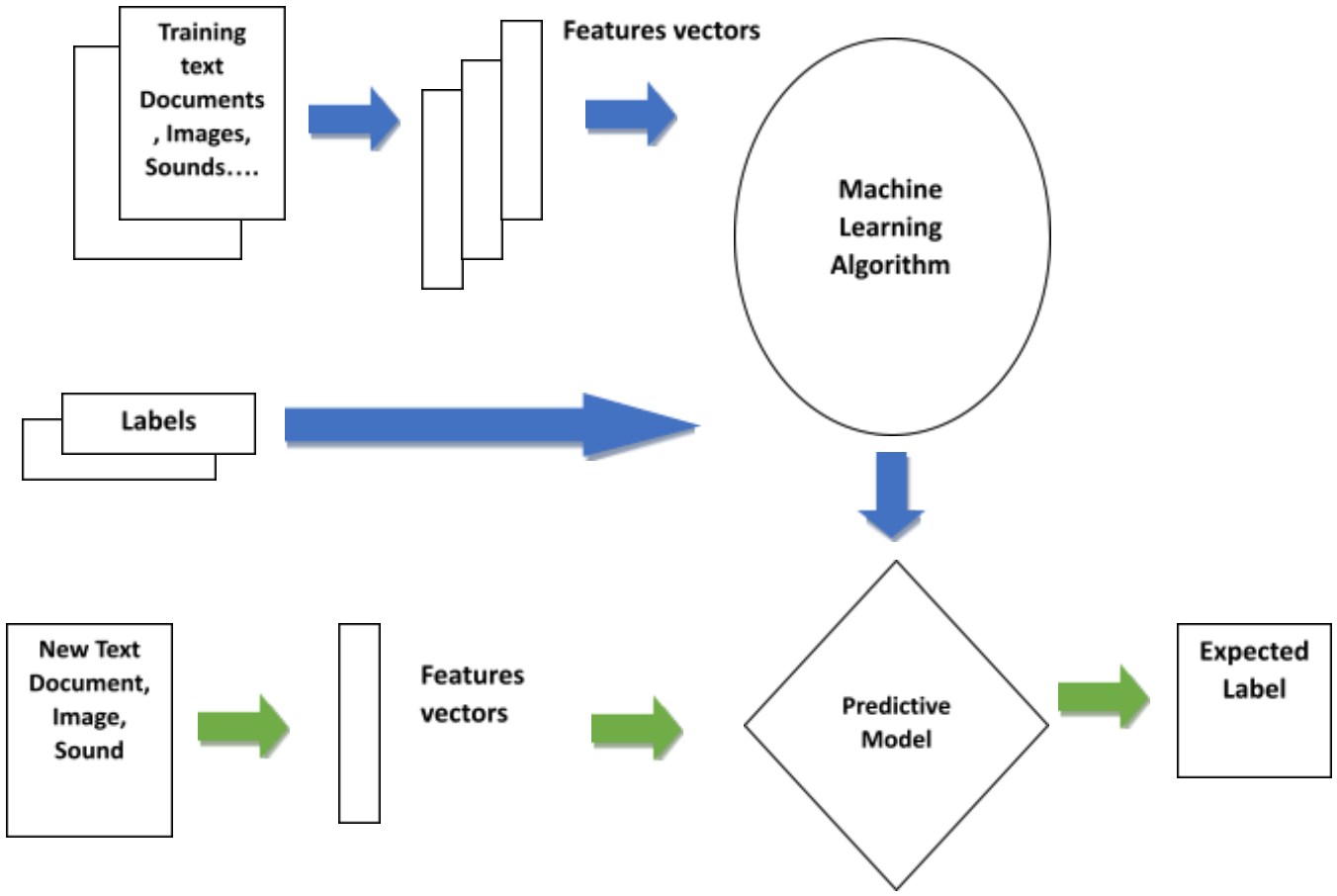
Two types of Machine learning are:

- Supervised learning
- Unsupervised learning

4.6.1 Supervised Learning

The set of data (training data) consists of a set of input data and correct responses corresponding to every piece of data. Based on this training data, the algorithm has to generalize such that it is able to correctly (or with a low margin of error) respond to all possible inputs.

In essence: The algorithm should produce sensible outputs for inputs that weren't encountered during training. Also called learning from exemplars.



Supervised learning can be classified into

- ✔ Regression Problems
- ✔ Classification Problems

4.3.1.1 Classification Problems

Consists of taking input vectors and deciding which of the N classes they belong to, based on training from exemplars of each class. “- Is discrete (most of the time). I.e. an example belongs to precisely one class, and the set of classes covers the whole possible output space.

How it's done: Find ‘decision boundaries’ that can be used to separate out the different classes. Given the features that are used as inputs to the classifier, we need to identify some values of those features that will enable us to decide which class the current input belongs to.

4.3.1.2 Regression Problems

Given some data, you assume that those values come from some sort of function and try to find out what the function is. In essence: You try to fit a mathematical function that describes a curve, such that the curve passes as close as possible to all the data points. So, regression is essentially a problem of function approximation or interpolation.

4.3.2 Unsupervised Learning

Conceptually different problem. No information about correct outputs are available. No Regression, no guesses about the function can be made. Information about the correct classes. But if we design our algorithm so that it exploits similarities between inputs so as to cluster inputs that are similar together, this might perform classification automatically.

ALGORITHMS USED FOR PREDICTING THE DEMANDS

4.3.3 Ridge Regression

Ridge regression addresses some of the problems of Ordinary Least Squares by imposing a penalty on the size of coefficients. The ridge coefficients minimize a penalized residual sum of squares,

$$\min_w ||Xw - y||_2{}^2 + \alpha||w||_2{}^2$$

Here, $\alpha \geq 0$ is a complexity parameter that controls the amount of shrinkage: the larger the value of , the greater the amount of shrinkage and thus the coefficients become more robust to co linearity. RidgeCV implements ridge regression with built-in cross-validation of the alpha parameter. The object works in the same way as GridSearchCV except that it defaults to Generalized Cross-Validation (GCV), an efficient form of leave-one-out cross-validation.

4.3.4 Lasso

The Lasso is a linear model that estimates sparse coefficients. It is useful in some contexts due to its tendency to prefer solutions with fewer parameter values, effectively reducing the number of variables upon which the given solution is dependent. For this reason, the Lasso and its variants are fundamental to the field of compressed sensing. Under certain conditions, it can recover the exact set of non-zero weights (see Compressive sensing: tomography reconstruction with L1 prior (Lasso)).

Mathematically, it consists of a linear model trained with ℓ_1 prior as regularizer. The objective function to minimize is:

$$\min_w \frac{1}{2n_{samples}}||Xw - y||_2^2 + \alpha||w||_1$$

The lasso estimate thus solves the minimization of the least-squares penalty with $\alpha||w||_1$ added, where α is a constant and $||w||_1$ is the ℓ_1-norm of the parameter vector.

4.3.5 Elastic Net

Elastic Net is a linear regression model trained with L1 and L2 prior as regularizer. This combination allows for learning a sparse model where few of the weights are non-zero like Lasso, while still maintaining the regularization properties of Ridge. We control the convex combination of L1 and L2 using the l1_ratio parameter.

Elastic-net is useful when there are multiple features which are correlated with one another. Lasso is likely to pick one of these at random, while elastic-net is likely to pick both.

A practical advantage of trading-off between Lasso and Ridge is it allows Elastic-Net to inherit some of Ridge's stability under rotation.

The objective function to minimize is in this case

$$\min_{w} \frac{1}{2n_{samples}} ||Xw - y||_2^2 + \alpha\rho||w||_1 + \frac{\alpha(1-\rho)}{2}||w||_2^2$$

4.4 ARCHITECTURAL DIAGRAM

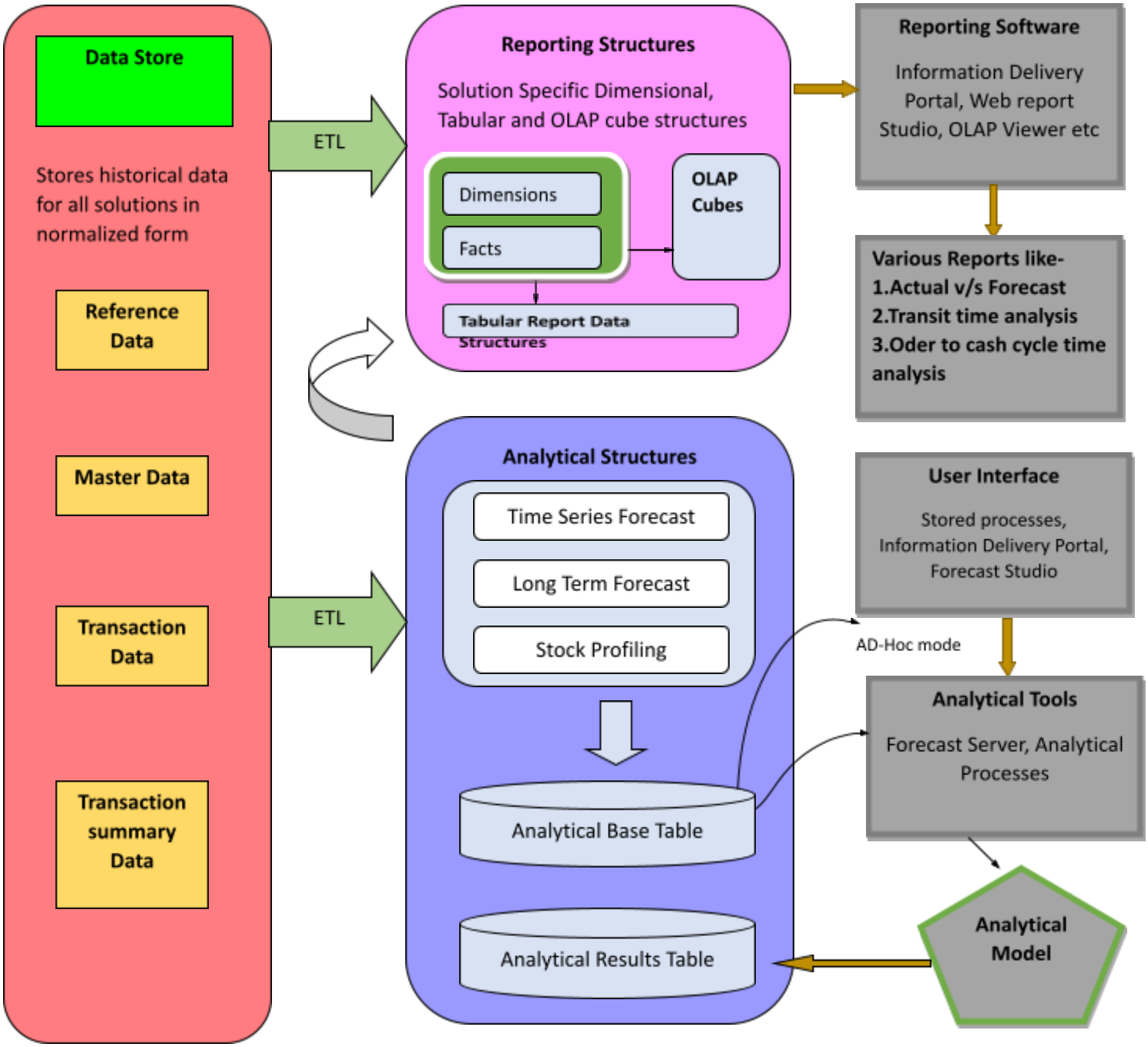


Fig. 4.1 Architectural Diagram

The **architectural diagram** is a created using available standards, in which the primary concern is to illustrate a specific set of trade-offs inherent in the structure and design of a system. The diagram indicates the how data is extracted, transformed and loaded into the analytical tools or machines and process carried out and tools and algorithms used to perform analysis.

4.5 DATA FLOW DIAGRAM

A Data Flow Diagram is a graphical representation of the "flow" of data through an information system. The above DFD indicates how there is flow of information and goods from customer to the company and back to the customer.

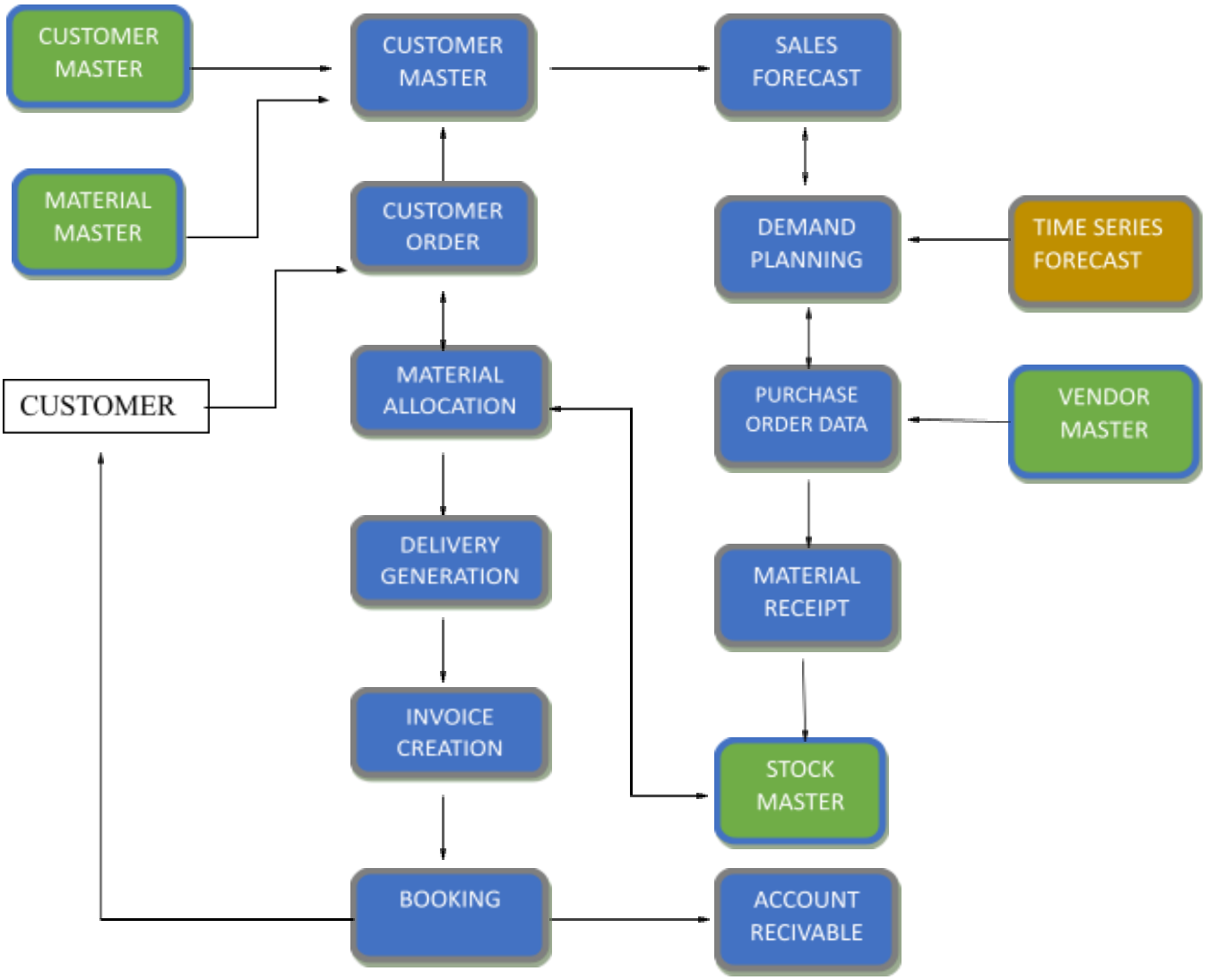
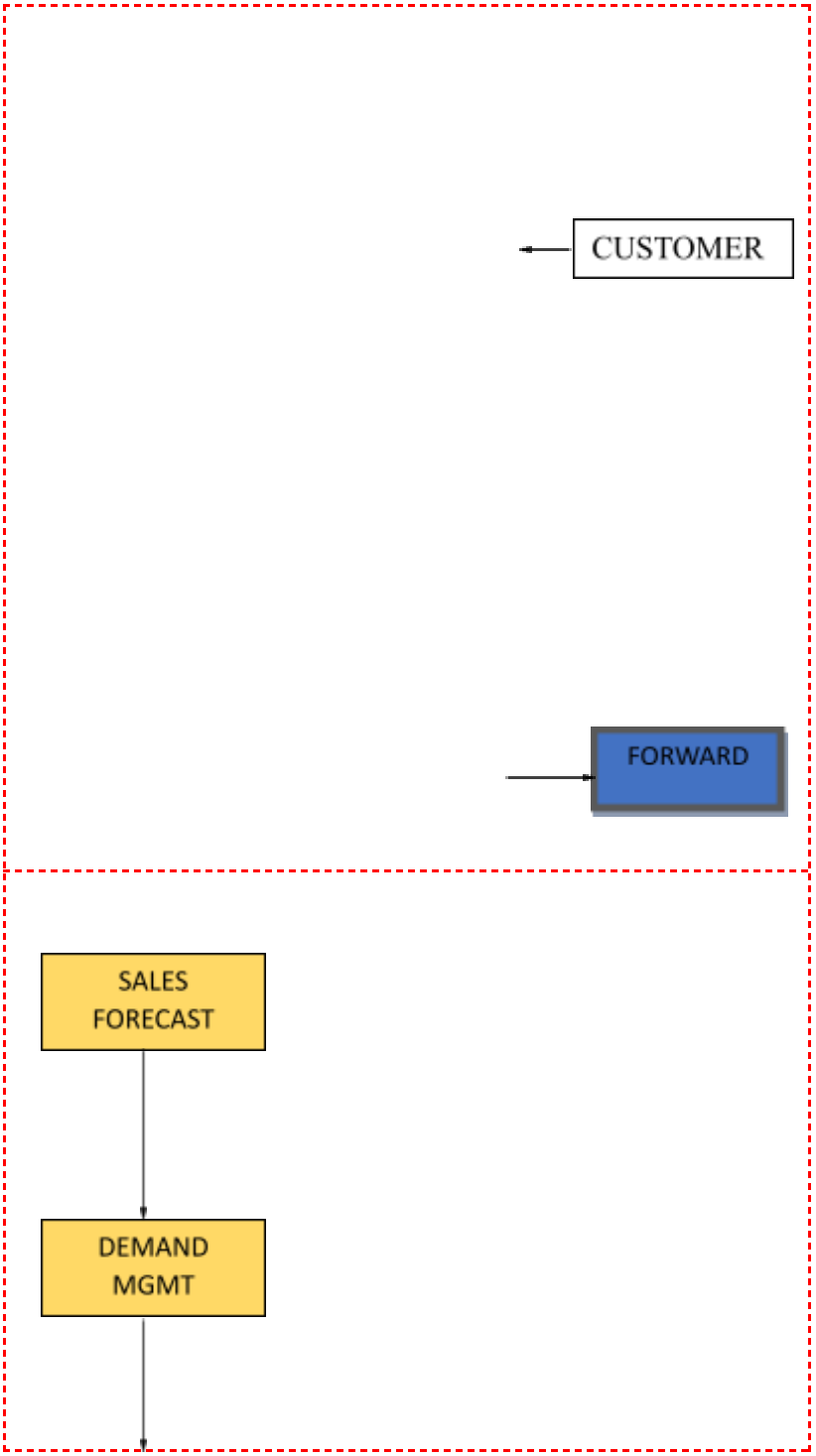
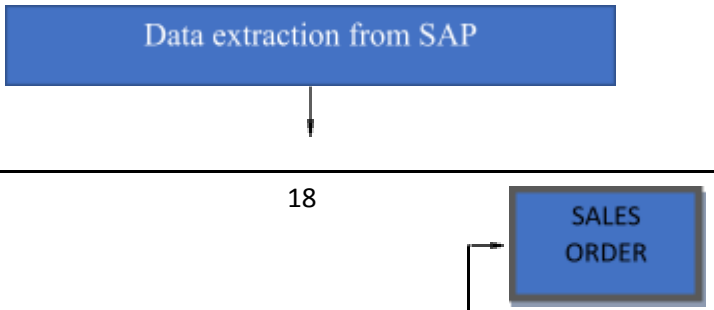


Fig 4.2 Data Flow Diagram



4.6 USE CASE DIAGRAM

Each use case represents a single, repeatable interactions that a user or actor experiences when using the system.



4.7 Project Work Flo Di

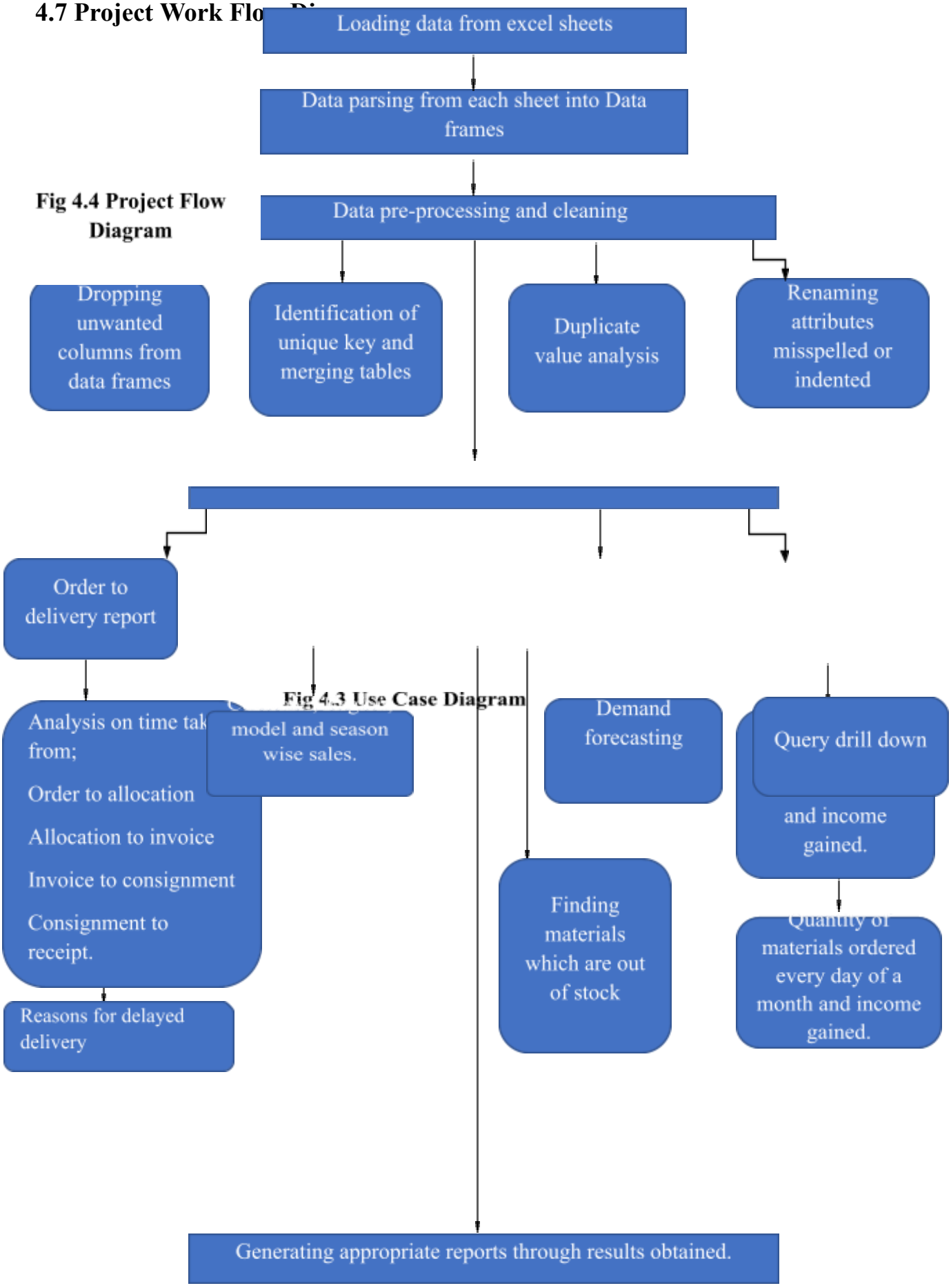


Fig 4.4 Project Flow Diagram

Fig 4.3 Use Case Diagram

CHAPTER 5

IMPLEMENTATION

5.1 STAGE 1: LOADING AND PRE-PROCESSING OF THE DATA.

The first stage in implementation involves loading the company's data from the excel sheet format to pandas dataframes, and then carrying out the early pre-processing on the data to detect and manipulate on the null values , missing values, duplicates and merge tables with strong relations for further implementation .

5.1.1 Load All The Modules We Need

1. For plotting

```
import matplotlib.pyplot as plt
import matplotlib
import seaborn as sns
import datetime
```

2. For ML

```
import sklearn
```

3. For data manipulation

```
import numpy as np
import pandas as pd
```

4. This makes all the plots to be shown within jupyter

```
%matplotlib inline
#Setting the default plot size
matplotlib.rcParams['figure.figsize'] = (20.0, 10.0)
```

5.1.2 Loading And Parsing Of Data From Excel To Pandas Dataframes

```
xlsx = pd.ExcelFile('data_File')

customer_order = xlsx.parse('customer order')

material_master = xlsx.parse('Material Master')

sales_past_demand = xlsx.parse('sales past demand')
```

5.1.3 Pre-Processing And Cleaning Of The Data

Considering the customer order dataframe.

5.1.3.1 Finding If There Are Null Values.

```
pd.isnull(customer_order).describe()
```

	SONO	ITEM	PTNO	DESC	DATE	ORD_QTY	CUST	PLNT	Price	customer PO ref	PO date	Value
count	39398	39398	39398	39398	39398	39398	39398	39398	39398	39398	39398	39398
unique	1	1	1	1	1	1	1	2	1	1	1	1
top	False	False	False	False	False	False	False	False	False	False	False	False
freq	39398	39398	39398	39398	39398	39398	39398	39388	39398	39398	39398	39398

We observe that there is missing data in PLNT. We fill it with the most common PLNT values.

```
customer_order.PLNT.fillna(930, inplace=True)
pd.isnull(customer_order).describe()
```

	SONO	ITEM	PTNO	DESC	DATE	ORD_QTY	CUST	PLNT	Price	customer PO ref	PO date	Value
count	39398	39398	39398	39398	39398	39398	39398	39398	39398	39398	39398	39398
unique	1	1	1	1	1	1	1	1	1	1	1	1
top	False	False	False	False	False	False	False	False	False	False	False	False
freq	39398	39398	39398	39398	39398	39398	39398	39398	39398	39398	39398	39398

Null values have been successfully removed from the dataframe. We carry out similar operations on all dataframes.

5.1.3.2 Processing Data With Incorrect Formats Or Ambiguous Values

We see that customer_order['PO date'] gives has some strings which are set to 00:00:00. We remove these by setting PO date to the DATE.

Fill in the missing 'PO date' with the corresponding values from 'DATE'. To do this, first get the month and day from each corresponding 'DATE' value. and using this and the year (2016) create a datetime.date and assign it to 'PO DATE'

```
tmp = customer_order[customer_order['PO date'] == datetime.time(0)]
day, month = [], []
for i in tmp['DATE'].str.split('-').values:
    day.append(int(i[0]))
    month.append(int(i[1]))
customer_order.loc[customer_order['PO date'] == datetime.time(0), 'PO date'] =
[datetime.date(2016, m, d) for m, d in zip(month, day)]
```

5.1.3.3 Merging tables with strong relations

Since there are multiple tables and there is a strong relation amongst these tables, we could merge the tables for easier access and manipulation. We use unique keys or if no such keys are present, we form them by concatenating two or more attributes.

Customer order and Customer master on customer’s unique code :

```
customer_order.head(2)
customer_master.head(2)
customer = pd.merge(customer_master, customer_order, left_on=['customer code'],
right_on=['CUST'])
customer.drop(['CUST'], axis=1, inplace=True)
```

5.1.3.4 Analysis Of Duplicates

We need to drop the duplicate values in the dataframes to eliminate redundancy. Considering the 'Bill' dataframe:

```
bill[bill.duplicated(['Sales ord', 'Delivery no', 'Description', 'Consignment details', 'Value'])]
bill.drop(bill[bill.duplicated(['Sales ord', 'Delivery no', 'Description', 'Consignment details', 'Value'])].index, inplace=True)
```

5.2 STAGE 2: DATA ANALYSIS

5.2.1 Time To Market Or Customer

The order to delivery analysis comprises finding the following:

1. Order to delivery note generation: The time between when order is placed by the customer to when the delivery note is generated by the company.
2. Delivery note to Invoice generation: The time interval between when the delivery note is generated to when the invoice for the corresponding order is sent to the customer.
3. Invoice to consignment: The time interval between when the invoice note is generated to when the consignment of product takes place.
4. Consignment to reaching customers (Recpt date): The time interval when the consigned product is delivered to the respective customer.

5.2.1.1 Order To Delivery Note Generation

We consider Customer and Delivery data table and map them using (SONO + Item).

```
tmp = customer[['SONO', 'PO date', 'ITEM']]
bill = bill.merge(tmp, how='left', left_on=['Sales ord', 'delivery Item'], right_on=['SONO', 'ITEM'])
order_to_delivery = bill[['PO date', 'Delivery date']]
order_to_delivery.loc[:, 'ORD_to_DEL'] = order_to_delivery['Delivery date'] - order_to_delivery['PO date']
 order_to_delivery.head(2)
```

	PO date	Delivery date	ORD_to_DEL
0	2016-05-16	2016-06-06	21 days
1	2016-05-13	2016-06-08	26 days

Implementation of function to segregate according to increasing order in the difference between dates

```
def display_days_difference(tmp, title):
    df = pd.DataFrame(columns=['Days', 'cum'])
     for i in xrange(7):
                df.loc[i] = [str(i) + ' days', tmp[tmp['Days'] <= i]['Count'].sum()]
                df.loc[7] = ['< 1 week', tmp[tmp['Days'] <= 7]['Count'].sum()]
                df.loc[8] = ['< 2 weeks', tmp[tmp['Days'] <= 14]['Count'].sum()]
                df.loc[9] = ['< 3 weeks', tmp[tmp['Days'] <= 21]['Count'].sum()]
                df.loc[10] = ['< 1 month', tmp[tmp['Days'] <= 30]['Count'].sum()]
                df.loc[11] = ['> 1 month', tmp['Count'].sum()]
      df['Count'] = df['cum']
      for i in xrange(11, 0, -1):
                df.loc[i, 'Count'] = df.loc[i, 'Count'] - df.loc[i - 1, 'Count']
```

Finding percentage and cumulative percentage of delivery note generation under each of the category

```
df['percentage'] = 100*df.Count/df.Count.sum()

df['cum percent'] = df.percentage.cumsum()
```

Display dataframe as table

```
from IPython.display import display
print "TABLE"
display(df)
```

Displaying line graphs and corresponding bar graphs on the same axes plane.

```
for (i, j) in [('Count', 'percentage'), ('cum', 'cum percent')]:
    fig, ax1 = plt.subplots()
    ax2 = ax1.twinx()
    ax1.bar([x for x in xrange(len(df))], df[i], width=.5, label=i)
    ax2.plot([x for x in xrange(len(df))], df[j], color='red', marker='o')
    for k in xrange(len(df)):
        ax1.text(k, df[i][k] + 300, str(float(df[j][k]))[:4], horizontalalignment='center')
    plt.xticks([x for x in xrange(len(df))], df['Days'])
    h1, l1 = ax1.get_legend_handles_labels()
    h2, l2 = ax2.get_legend_handles_labels()
    ax1.legend(h1+h2, l1+l2, loc=0)
    plt.title(title)
    plt.show()
```

Table For Order To Delivery Note Generation

	Days	cum	Count	percentage	cum percent
0	0 days	9542.0	9542.0	25.005241	25.005241
1	1 days	14206.0	4664.0	12.222222	37.227463
2	2 days	16884.0	2678.0	7.017820	44.245283
3	3 days	18502.0	1618.0	4.240042	48.485325
4	4 days	19722.0	1220.0	3.197065	51.682390
5	5 days	21309.0	1587.0	4.158805	55.841195
6	6 days	22532.0	1223.0	3.204927	59.046122
7	< 1 week	23801.0	1269.0	3.325472	62.371593
8	< 2 weeks	29070.0	5269.0	13.807652	76.179245
9	< 3 weeks	31730.0	2660.0	6.970650	83.149895
10	< 1 month	33919.0	2189.0	5.736373	88.886268
11	> 1 month	38160.0	4241.0	11.113732	100.000000

5.2.1.2 Delivery To Invoice Generation.

Finding difference between delivery date and bill date :

```
delivery_to_bill = bill[['Bill date', 'Delivery date']]
delivery_to_bill.loc[:, 'DEL_TO_BILL'] = delivery_to_bill['Bill date'] -
delivery_to_bill['Delivery date']
tmp = delivery_to_bill['DEL_TO_BILL'].value_counts().reset_index()
tmp['Days'] = tmp['Days'].apply(lambda x: x.days)
tmp.sort_values(by=['Days'], inplace=True)
```

Calling the function display_days_difference to display the graph :

display_days_difference(tmp, 'Service Efficiency: Allocation To Invoice')

Table For Delivery to Invoice Generation

	Days	cum	Count	percentage	cum percent
0	0 days	15599.0	15599.0	40.660515	40.660515
1	1 days	30304.0	14705.0	38.330205	78.990720
2	2 days	34298.0	3994.0	10.410802	89.401522
3	3 days	36301.0	2003.0	5.221041	94.622563
4	4 days	37425.0	1124.0	2.929830	97.552393
5	5 days	37699.0	274.0	0.714211	98.266604
6	6 days	37987.0	288.0	0.750704	99.017308
7	< 1 week	38104.0	117.0	0.304973	99.322281
8	< 2 weeks	38304.0	200.0	0.521322	99.843603
9	< 3 weeks	38335.0	31.0	0.080805	99.924408
10	< 1 month	38342.0	7.0	0.018246	99.942655
11	> 1 month	38364.0	22.0	0.057345	100.000000

5.2.1.3 Invoice To Consignment Generation

Determining time interval between invoice to consignment and calling function to display respective table and corresponding graphs.

```
bill_to_GC = bill[['Bill date', 'GC date']]
bill_to_GC.loc[:, 'BILL_TO_GC'] = bill_to_GC['GC date'] - bill_to_GC['Bill date']
bill_to_GC.loc[:, 'BILL_TO_GC'] = bill_to_GC['BILL_TO_GC'].dt.components.days
tmp = bill_to_GC['BILL_TO_GC'].value_counts().reset_index()
tmp.columns = ['Days', 'Count']
tmp['Days'] = tmp['Days'].apply(lambda x: x.days)
tmp.sort_values(by=['Days'], inplace=True)
tmp = tmp.reset_index().drop('index', axis=1)
display_days_difference(tmp, 'Service Efficiency: Invoice to Consignment')
```

Table For Invoice To Consignment Generation

	Days	cum	Count	percentage	cum percent
0	0 days	30915.0	30915.0	80.583359	80.583359
1	1 days	37753.0	6838.0	17.824002	98.407361
2	2 days	38253.0	500.0	1.303305	99.710666
3	3 days	38364.0	111.0	0.289334	100.000000
4	4 days	38364.0	0.0	0.000000	100.000000
5	5 days	38364.0	0.0	0.000000	100.000000
6	6 days	38364.0	0.0	0.000000	100.000000
7	< 1 week	38364.0	0.0	0.000000	100.000000
8	< 2 weeks	38364.0	0.0	0.000000	100.000000
9	< 3 weeks	38364.0	0.0	0.000000	100.000000
10	< 1 month	38364.0	0.0	0.000000	100.000000
11	> 1 month	38364.0	0.0	0.000000	100.000000

5.2.1.4 Consignment To Receipt

Determining time interval between consignment to receipt calling function to display respective table and corresponding graphs.

```
GC_to_recpt = bill[['GC date', 'Recpt date']]
GC_to_recpt.loc[:, 'GC_TO_RECPT'] = GC_to_recpt['Recpt date'] - GC_to_recpt['GC date']
tmp = GC_to_recpt['GC_TO_RECPT'].value_counts().reset_index()
tmp.columns = ['Days', 'Count']
tmp['Days'] = tmp['Days'].apply(lambda x: x.days)
tmp.sort_values(by=['Days'], inplace=True)
tmp = tmp.reset_index().drop('index', axis=1)
display_days_difference(tmp, 'Service Efficiency: Consignment to Reciept')
```

Table for Consignment to Receipt

	Days	cum	Count	percentage	cum percent
0	0 days	288.0	288.0	0.754717	0.754717
1	1 days	1139.0	851.0	2.230084	2.984801
2	2 days	4442.0	3303.0	8.655660	11.640461
3	3 days	8561.0	4119.0	10.794025	22.434486
4	4 days	11731.0	3170.0	8.307128	30.741614
5	5 days	14542.0	2811.0	7.366352	38.107966
6	6 days	16973.0	2431.0	6.370545	44.478512
7	< 1 week	18867.0	1894.0	4.963312	49.441824
8	< 2 weeks	26393.0	7526.0	19.722222	69.164046
9	< 3 weeks	30036.0	3643.0	9.546646	78.710692
10	< 1 month	32979.0	2943.0	7.712264	86.422956
11	> 1 month	38160.0	5181.0	13.577044	100.000000

5.2.2 Customer wise, Region wise and Model wise sales analysis

5.2.2.1 Customer wise sales

The next step involves determining the customers providing maximum profit to the company. Here we display the top 50 value adding customers.

Finding the total price corresponding to each customer

```
customer['Total_Price'] = customer['ORD_QTY'] * customer['Price']
customer.head(1)
tmp = customer[['customer code', 'Total_Price']]
tmp = tmp.groupby(['customer code']).sum().reset_index()
tmp.sort_values(by='Total_Price', ascending=False, inplace=True)
tmp = tmp.reset_index(drop=True)
```

Considering top 50 profit giving customers

```
tmp2 = tmp.loc[50:]
tmp = tmp.loc[:49]
tmp.loc[50] = ['Others', tmp2['Total_Price'].sum()]
```

Code for plotting the bar graph

```
sns.barplot(x='customer code', y='Total_Price', data=tmp)
plt.xticks(rotation=60)
plt.title("Top customers")
plt.plot()
```

5.2.2.2 Region wise Sales

Similar to customer wise sales, we now analyse on the profit incurred by the company in various regions within India

```
df = customer.groupby('Region')['Total_Price'].sum().reset_index()


 Sorting by income
df = df.sort_values(by='Total_Price', ascending=False)
df = df.reset_index(drop=True)


Plotting the region wise profit graph
sns.barplot(x = 'Region', y= 'Total_Price', data=df)
plt.xticks(rotation=60)
plt.plot()
```

5.2.2.3 Model Wise Sales

We now explicate on the profit incurred by various model releases by the company.

```
model_demand = material_master
model_demand = model_demand.groupby('Model').sum().reset_index()
model_demand.drop(['safety stock'], axis=1, inplace=True)
model = model_demand.sort_values(by='Demand', ascending=False)
model = model.reset_index(drop=True)
model.head(15)
```

Table showing model code and demand

	Model	Demand
0	PC200	129526
1	PC300	61608
2	PC71	44963
3	PC210	34785
4	D155	32093
5	300CK	29608
6	PC130	26286
7	PC450	21973
8	Others	14919

5.2.3 SEASON WISE DEMAND

Here we try to find out how the demand for various products in the company vary with the seasons.

First, we extract the month from 'PO date' of customer_order and find the profit for each month. Then, the months are grouped into seasons as mentioned below.

We consider 5 seasons:

1. Winter - November to Feb (11,12,1,2)
2. Spring - March to April (3,4)
3. Summer - May to June (5,6)
4. Monsoon - July to August (7,8)
5. Autumn - September to October (9,10)

Then, Total_Profit for each month is found.

Grouping the months into seasons and finding the profit in each season.

```
def test (row):
        if (row['month'] ==1)|(row['month']==2)      :
            return 'winter'
        if (row['month'] ==3)|(row['month'] ==4)     :
            return 'spring'
        if (row['month'] ==5)|(row['month'] ==6)     :
            return 'summer'
        if (row['month']==7)|(row['month']==8)       :
            return 'monsoon'
        if (row['month'] ==9)|(row['month'] ==10)    :
            return 'autumn'
       if (row['month'] ==11)|(row['month'] ==12)  :
           return 'winter'
 df['season'] = df.apply (lambda row: test(row),axis=1)
 df = df.drop(['month'], axis=1)
 df = df.groupby('season').sum().reset_index()
 df['percentage'] = 100*df.Total_Price/df.Total_Price.sum()
```

	season	Total_Price	percentage
0	autumn	115282947	16.943341
1	monsoon	190583361	28.010378
2	spring	136098629	20.002659
3	summer	229683886	33.757052
4	winter	8753860	1.286570

5.2.4 Finding Materials OUT_OF_STOCK

In this part of data analysis, we find materials which are out of stock, in stock, and materials where only few stocks are left.

First, we will merge material_master and stock_master to obtain safety stock and quantity for each materials in one table. Unwanted columns are dropped.

Then, the materials are grouped into IN_STOCK, OUT_OF_STOCK and NEW_STOCKS_NEEDED as show below.

```
# checking for various conditions
def test (row):
  if row['safety stock'] < row['Quantity']  :
    return 'IN_STOCK'
  if row['safety stock'] > row['Quantity']  :
    return 'OUT_OF_STOCK'
  if row['safety stock'] == row['Quantity']  :
    return 'NEW_STOCK_NEEDED'
  return 'Other'
stock_df['status'] = stock_df.apply (lambda row: test (row),axis=1)
```

Table indicating status of the first 5 materials in company's warehouse

	Material code	Model	Safety stock	Quantity	Status
0	01010-61435I.	PC450	8	86	IN_STOCK
1	01010-61455I.	D65	1	4	IN_STOCK
2	01010-61635I.	GD511	1	34	IN_STOCK
3	01010-61645I.	D475	8	4	OUT_OF_STOCK
4	01010-61650I.	HD465	5	8	IN_STOCK

After grouping the materials into out_of_stock, in_stock and new_stocks_neeeded, we will focus our analysis on the materials which are out_of_stock.

```
#Retrieving materials which are out of stock from stock_df table
ofs=stock_df.loc[stock_df['status'] == 'OUT_OF_STOCK']

#Finding how much stock is needed
ofs['Stock Needed'] = ofs['safety stock'] + ofs['Quantity']

#sorting by the value of stock needed(difference)
ofs=ofs.sort_values(by='Stock Needed',ascending=False)
ofs=ofs.reset_index(drop=True)
ofs.head(5)
```

	Material code	Model	safety stock	Quantity	status	Stock Needed
0	07063-51210I	PC200	229	4	OUT_OF_STOCK	233
1	01010-61865I.	D65	113	85	OUT_OF_STOCK	198
2	6136-51-5121I	D65	172	12	OUT_OF_STOCK	184
3	20Y-60-21311I	PC200	183	1	OUT_OF_STOCK	184
4	07063-01100I	D275	151	5	OUT_OF_STOCK	156

5.2.5 CHECKING POSSIBLE REASONS FOR DELAY FROM CONSIGNMENT TO DELIVERY

In section 5.2.1, we found the time taken between consignment to delivery(receipt). Here we are trying to find out the possible reasons for delay from consignment to delivery.

First, we extract consignment to delivery time from bill data frame. Then the table is grouped according to consignment details.

```
GC_to_recpt = bill[['GC date', 'Recpt date', 'Consignment details']]
GC_to_recpt.loc[:, 'GC_TO_RECPT'] = (GC_to_recpt['Recpt date'] –
GC_to_recpt['GC date']) .dt.days
GC_to_recpt = GC_to_recpt.sort_values(by='GC_TO_RECPT', ascending=False)
GC_to_recpt = GC_to_recpt.groupby('Consignment details') .sum(). reset_index()
.sort_values(by='GC_TO_RECPT', ascending=False).reset_index(drop=True)
```

	Consignment details	GC_TO_RECPT
0	442176696 GATI 21-07-2016	3922
1	22364639 NECC 26-07-2016	3834
2	442176603 GATI 23-07-2016	2313
3	21041 UNITY 11-07-2016	1895
4	LOCAL COLLECTION	1682

5.2.6 QUERY DRILL DOWN

The next part of data analysis is query drill down. Here, we will check if the order fluctuates as per the date, as we know there would be season for most orders. Like at the end of the fiscal year or start of new fiscal year.

Here we will first find the profit and demand for each month of the year and then the same for each day of the month. For this we are using customer data frame.

```
#For month of the year
df = customer[['PO date', 'Total_Price', 'ORD_QTY']]
df.loc[:,'month'] = df['PO date'].apply(lambda x: x.month)
df = df.drop(['PO date'], axis=1)
df = df.groupby(['month']).sum().reset_index()
#For day of the month
df = customer[['PO date', 'Total_Price', 'ORD_QTY']]
df.loc[:,'day'] = df['PO date'].apply(lambda x: x.day)
df = df.drop(['PO date'], axis=1)
df = df.groupby(['day']).sum().reset_index()
```

Month of the year

	month	Total_Price	ORD_QTY
0	1	534144	191
1	2	3298831	1099
2	3	27143439	1439

3	4	108955190	38685
4	5	120713266	38275
5	6	108970620	38315
6	7	104511662	34957
7	8	86071699	29733
8	9	115282947	35748
9	11	2934394	215
10	12	1986491	378
	day	**Total_Price**	**ORD_QTY**
0	1	12893564	3604
1	2	30516498	8774
2	3	24459472	5732
3	4	26489072	7810
4	5	26719262	10069

Day of the month

5.2.7 DEMAND FORECASTING

The next step of data analysis (predictive analysis) is demand forecasting. Here we are predicting the future demand of materials from the previous demands.

From sales past demand table we have demand for 36 months. We are implementing "Regression Analysis" to predict the future demands.

First, we will check for periodicity. This is done by analysing the subplots for materials. Then if periodicity is present we will proceed with prediction by applying various regression methods.

Instead of data per month, we divide the data as to have data per 3 months. This allows us to predict the demand for the next three months which would be aggregated better

than data per month. Since, we had demand for 36 months, know it is divided into 12 intervals of three months.(Q_0 to Q_11)

```
#Grouping the demand into interval of three months
df = pd.DataFrame()
df['Material code'] = sales_past_demand['Material code']
for i in range(1, 37, 3):
        df['Q_' + str((i-1)/3)] = sales_past_demand[['DEM' + str(x) for x in range(i, i + 3)]].
sum(axis=1)
```

Demand for interval of 3 months

	Material code	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_10	Q_11
0	01124-82250I.	0	6	0	0	0	0	8	0	0	0	0	3
1	01310-21216I.	100	102	112	83	87	39	173	74	59	60	40	75
2	01435-00814I.	0	2	0	0	0	0	8	0	1	0	0	3
3	01435-00865I.	2	0	0	0	0	0	0	0	0	0	0	0
4	01435-01025I.	2	1	7	20	1	1	14	9	4	6	12	13
5	015424KB.	0	1	6	5	7	3	1	4	2	0	0	1

We split the data into quarters (3 months). Since we have data of 36 months, this gives us 12 quarters. So, the idea is to train the model on 11 quarters so that it is able to predict the 12th quarter.

Now we have our desired inputs and desired outputs. But it wouldn't make sense to train the machine learning algorithm and test it on the same data, so we'll now split our data into training and tests sets (70% - 30%).

```
#Dividing the data into train and test set
 from sklearn.model_selection import train_test_split
 X_train, X_test, y_train, y_test = train_test_split(df.drop('Q_11', axis=1), df.Q_11,
 test_size=.3, random_state=42)
```

Finding the most accurate Regression method for demand prediction:

1. ElasticNet regression

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import ElasticNet

param = {'alpha': [1.0, 2, 5, 10, 50, 100, 1000], 'normalize': [True, False]}
reg = GridSearchCV(ElasticNet(), param)
reg.fit(X_train, y_train)
reg.score(X_test, y_test)
```

Accuracy = 0.8616707598397878

2. Lasso

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Lasso
param = {'alpha': [1.0, 2, 5, 10, 50, 100, 1000], 'normalize': [True, False]}
reg = GridSearchCV(Lasso(), param)
reg.fit(X_train, y_train)
reg.score(X_test, y_test)
```

Accuracy = 0.84968164809060198

3. Ridge

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge

param = {'alpha': [1.0, 10, 100], 'normalize': [True, False], 'solver' : ['auto', 'svd', 'cholesky',
'lsqr', 'sparse_cg', 'sag']}
reg = GridSearchCV(Ridge(), param)
reg.fit(X_train, y_train)
reg.score(X_test, y_test)
```

Accuracy = 0.87221892970289294

4. AdaboostRegressor

```
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import AdaBoostRegressor

param = {'n_estimators': [50, 100, 500], 'loss': ['linear', 'square', 'exponential']}
reg = GridSearchCV(AdaBoostRegressor(), param)
reg.fit(X_train, y_train)
reg.score(X_test, y_test)
```

Accuracy = 0.77895308772704985

Picking up the best model:

Since we got the highest accuracy with the Ridge model, we'll use it to do our predictions. Now using the Ridge regression lets train and then test on our data set.

```
#Training and testing the data using ridge regression

from sklearn.linear_model import Ridge
reg = Ridge(alpha=100, normalize=False, solver='sag')
reg.fit(X_train, y_train)
```

We will perform two iterations of ridge regression to predict the demand for future two quarters.

1. Iteration one:

In first iteration, we will split the data into test(X) and train(y) sets. Where, the ridge model is trained using train set and test the predicted values using test set.

We will assign y as the required output (Q_11) and the others as X the input (Q_1, Q_2 ... Q_10).

```
y = df['Q_11']
X = df.drop(['Q_11'], axis = 1)
```

Now we have our desired inputs and desired outputs. But it wouldn't make sense to train the machine learning algorithm and test it on the same data, so we'll now split our data into training and tests sets (70% - 30%).

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

Now using the Ridge regression lets train and then test on our data set

```
from sklearn.linear_model import Ridge

reg = Ridge(alpha=100, normalize=False, solver='sag')
reg.fit(X_train, y_train)
```

Now our model is trained on our data, we will try to predict the test set.

```
pred = reg.predict(X_test)
tempdf = pd.DataFrame()
tempdf.loc[:, 'real'] = y_test
tempdf.loc[:, 'pred1'] = pred
tempdf.reset_index(inplace=True, drop=True)
tempdf.head(10)
```

#Predicted values got from the implementation of Ridge regression

	real	pred1
0	0	0.649993
1	0	0.300784
2	0	0.191031
3	0	0.534626
4	0	3.720385
5	44	10.751811

Now, we will put these predicted values as demand for first future quarter i.e. Q_12.
And the sales_past_demand table would look like;

	Material code	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_10	Q_11	Q_12
0	01124-82250I.	0	6	0	0	0	0	8	0	0	0	0	4	2
1	01310-21216I.	100	102	112	83	87	39	173	74	59	60	40	77	75
2	01435-00814I.	0	2	0	0	0	0	8	0	1	0	0	0	3
3	01435-00865I.	2	0	0	0	0	0	0	0	0	0	0	0	0
4	01435-01025I.	2	1	7	20	1	1	14	9	4	6	12	5	13
5	015424KB.	0	1	6	5	7	3	1	4	2	0	0	0	1
6	01580-10806I.	2	12	3	2	5	4	4	8	0	0	6	0	5
7	01599-01011I.	4	0	0	1	2	2	0	0	0	0	0	0	0
8	01602-20825I.	0	2	0	0	0	0	0	0	0	0	0	0	0
9	01643-31445.	9	25	0	0	26	16	46	34	24	32	22	34	36
10	01643-32460.	0	24	0	0	16	0	36	0	0	0	0	0	9

2. Iteration two:

Second iteration is similar to first iteration. But, here we will train the ridge model according to train data Q_1 to Q_11 and test the predicted value with test data Q_12. The predicted values will be demand for second future quarter i.e. Q_13.
And the sales_past_demand table after implementing second iteration would look like,

	Material code	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_10	Q_11	Q_12	Q_13
0	01124-82250I.	0	6	0	0	0	0	8	0	0	0	0	4	2	1
1	01310-21216I.	100	102	112	83	87	39	173	74	59	60	40	77	75	58
2	01435-00814I.	0	2	0	0	0	0	8	0	1	0	0	0	3	1
3	01435-00865I.	2	0	0	0	0	0	0	0	0	0	0	0	0	0
4	01435-01025I.	2	1	7	20	1	1	14	9	4	6	12	5	13	11
5	015424KB.	0	1	6	5	7	3	1	4	2	0	0	0	1	1
6	01580-10806I.	2	12	3	2	5	4	4	8	0	0	6	0	5	5
7	01599-01011I.	4	0	0	1	2	2	0	0	0	0	0	0	0	0
8	01602-20825I.	0	2	0	0	0	0	0	0	0	0	0	0	0	0
9	01643-31445.	9	25	0	0	26	16	46	34	24	32	22	34	36	29

	Material code	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_10	Q_11	Q_12	Q_13
10	01643-32460.	0	24	0	0	16	0	36	0	0	0	0	0	9	3

Table representing the possible future demands of first 10 materials

CHAPTER 6

TESTING

Supply chain Management software testing is very much needed now to bring maximum profits to the organisation by the business owners.

The services rendered by us to our esteemed clients to enhance and boost the supply chain of our clients' business are the following ones

Unit Testing:

Unit Testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of software. It usually has one or a few inputs and usually a single output.

End-to-end Testing:

End-to-end testing is a technique used to test whether the flow of an application right from start to finish is behaving as expected. The purpose of performing end-to-end testing is to identify system dependencies and to ensure that the data integrity is maintained between various system components and systems.

Load Testing:

Load testing is the process of putting demand on a software system or computing device and measuring its response. Load testing is performed to determine a system's behaviour under both normal and anticipated peak load conditions.

Performance Testing:

Performance testing, a non-functional testing technique performed to determine the system parameters in terms of responsiveness and stability under various workload. Performance testing measures the quality attributes of the system, such as scalability, reliability and resource usage.

Compatibility Testing:

Compatibility testing is a non-functional testing conducted on the application to evaluate the application's compatibility within different environments. It can be of two types - forward compatibility testing and backward compatibility testing.

CHAPTER 7

RESULTS AND SNAPSHOT

7.1 ORDER TO DELIVERY REPORT

7.1.1 Order To Allocation (Delivery Note Generation)

Fig 7.1.1 Graphs indicating order to allocation time taken on individual and cumulative percentage basis

From the graphs above (Fig 7.1.1), the following observations are construed. We see that only around 50% of the allocation of goods ordered takes place within the first 3 days, and it takes around 1 month for 90% of the allocations to be completed.

This requires the company to focus on the main reasons behind this delay. One major reason could be the shortage of stocks in the company's warehouses as and when ordered.

We thereby try to find out the stocks which are consistently in demand and those unavailable at the time of order. We also predict on the probable future demands for each of these materials.

7.1.2 Allocation To Invoice Generation

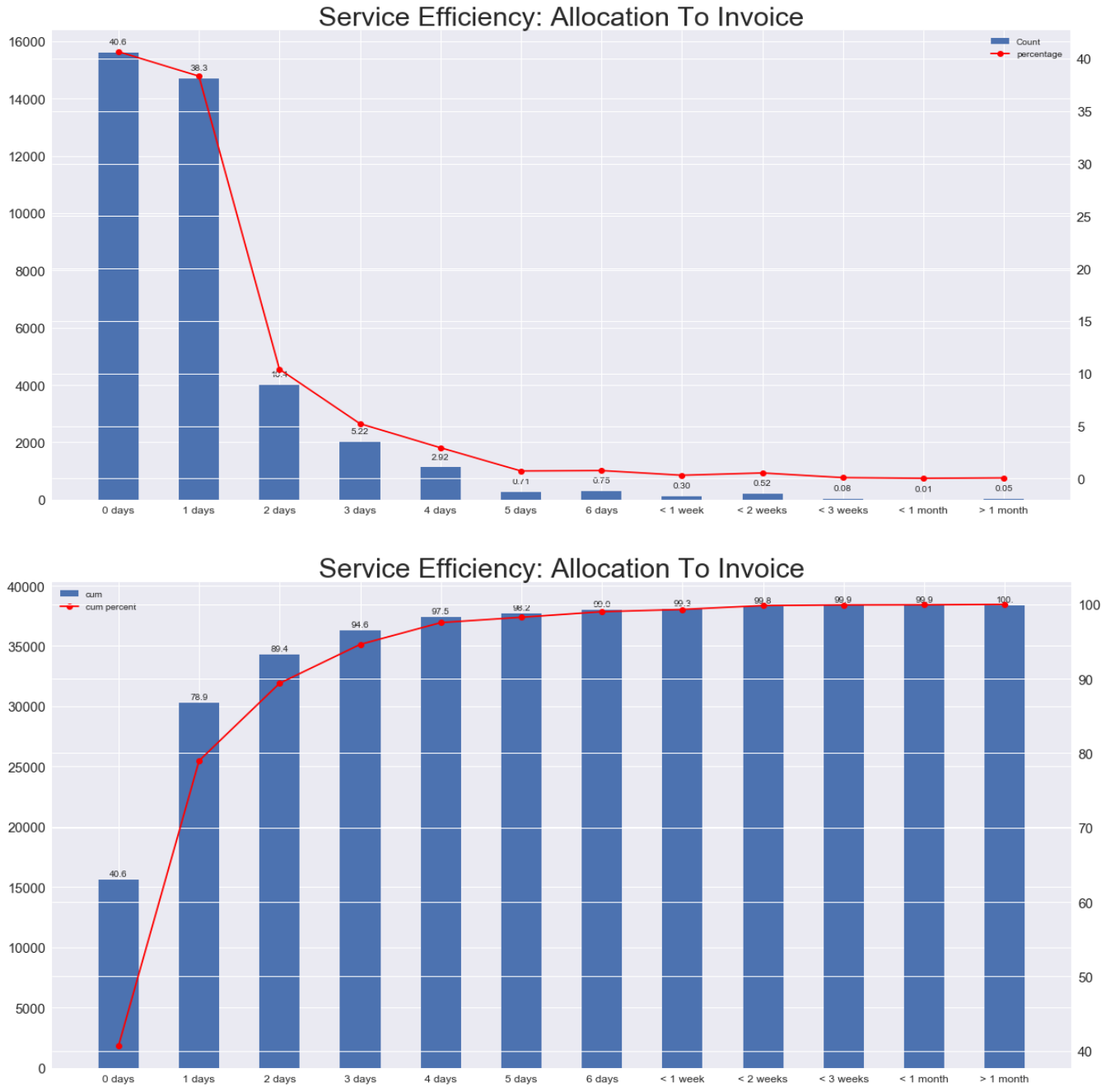


Fig 7.1.2 Graphs indicating time taken from allocation to invoice generation

From the graphs above (7.1.2) we infer that around 95% of the invoices for the orders made by the customers are generated within first 3 days. Hence no immediate measures are necessary on this aspect.

7.1.3 Invoice To Consignment

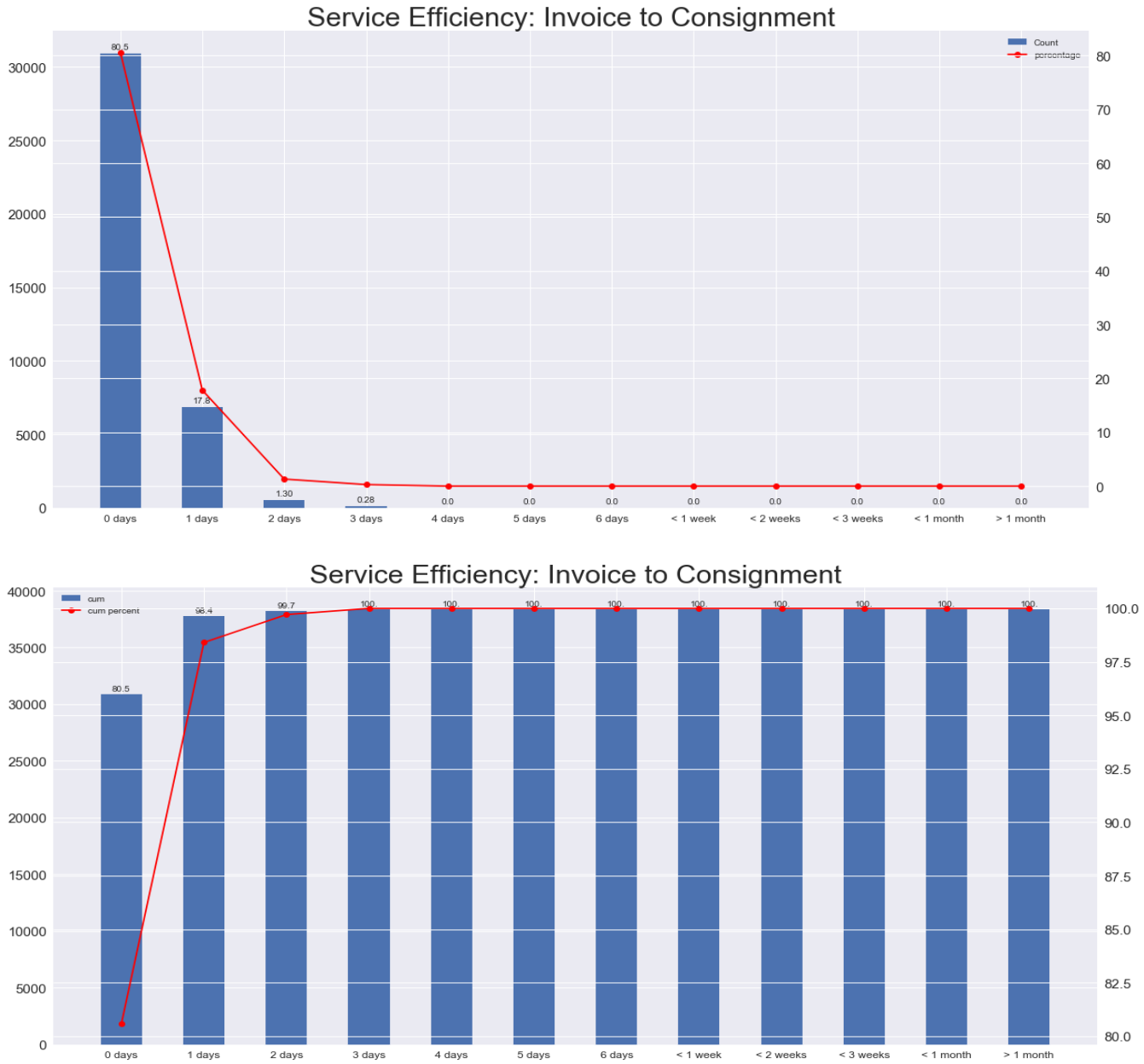


Fig 7.1.3 Graphs indicating time taken from to invoice to consignment

From the graphs above we infer that around 95% of the job is completed within 1 day. Hence no immediate measures are necessary on this aspect either.

7.1.4 Consignment To Reaching Customers (Receipt Date)

Fig 7.1.4 Graphs indicating time taken from consignment to delivery

From the graphs plotted, we construe it takes more than 1 month for all of the deliveries to reach their respective customers.

We decipher that one of the major reasons could be the mode of delivery being used. We thereby determine the modes of consignment taking major amount of time, the company needs to find alternatives in order to reduce the delivery time and increase their market value.

7.1.5 Order To Receipt

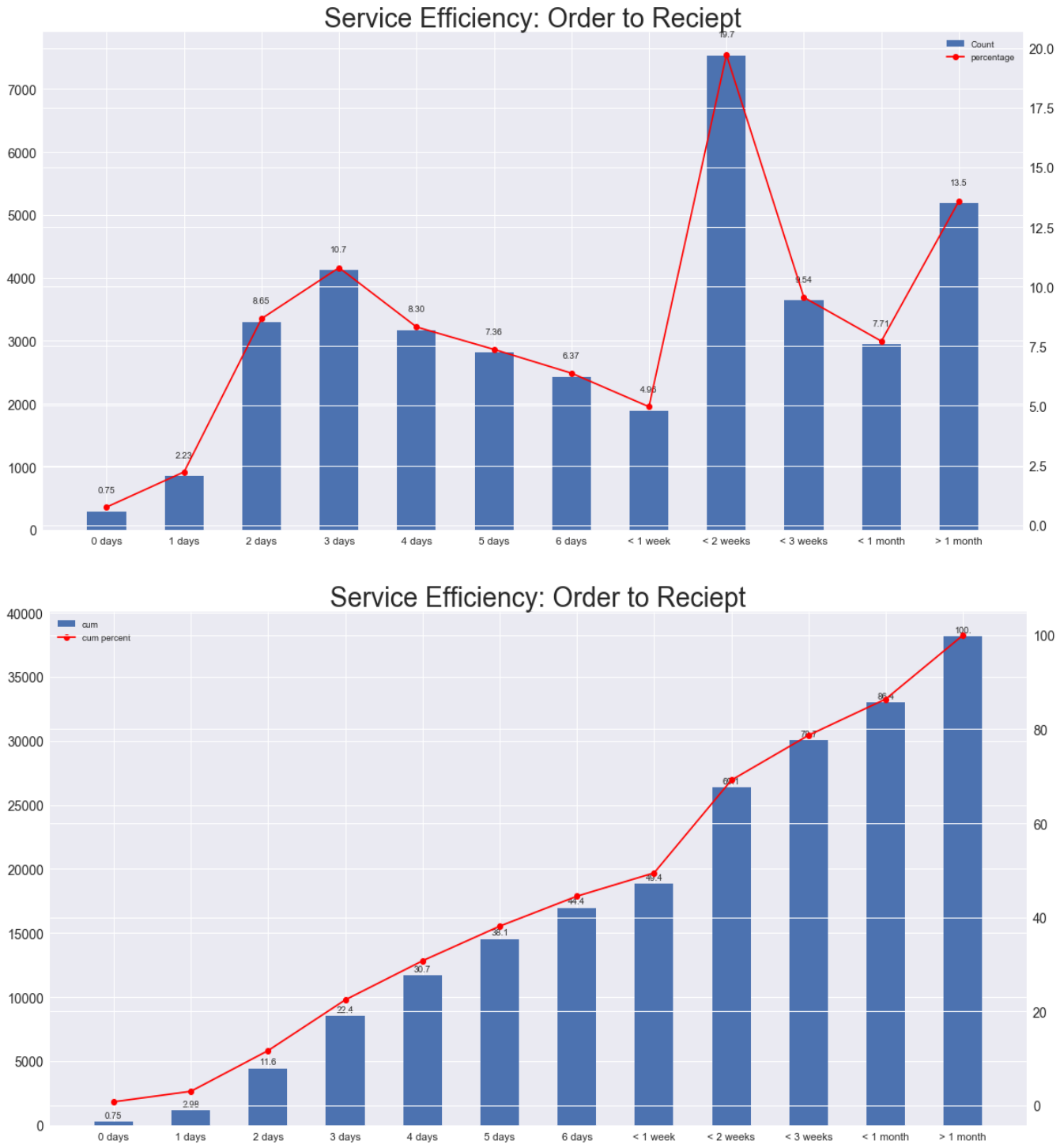


Fig 7.1.5 Graphs indicating total time taken from order to delivery of the goods

The above graph shows the total no. of days taken from the time order was placed by the customer to the day the order was delivered to him. It is observer that it takes around a month for all the orders to be delivered.

7.2 CUSOMER WISE, REGION WISE, MATERIAL WISE AND MODEL WISE PROFITS, AND SEASON WISE SALES.

7.2.1 Customer Wise Profit

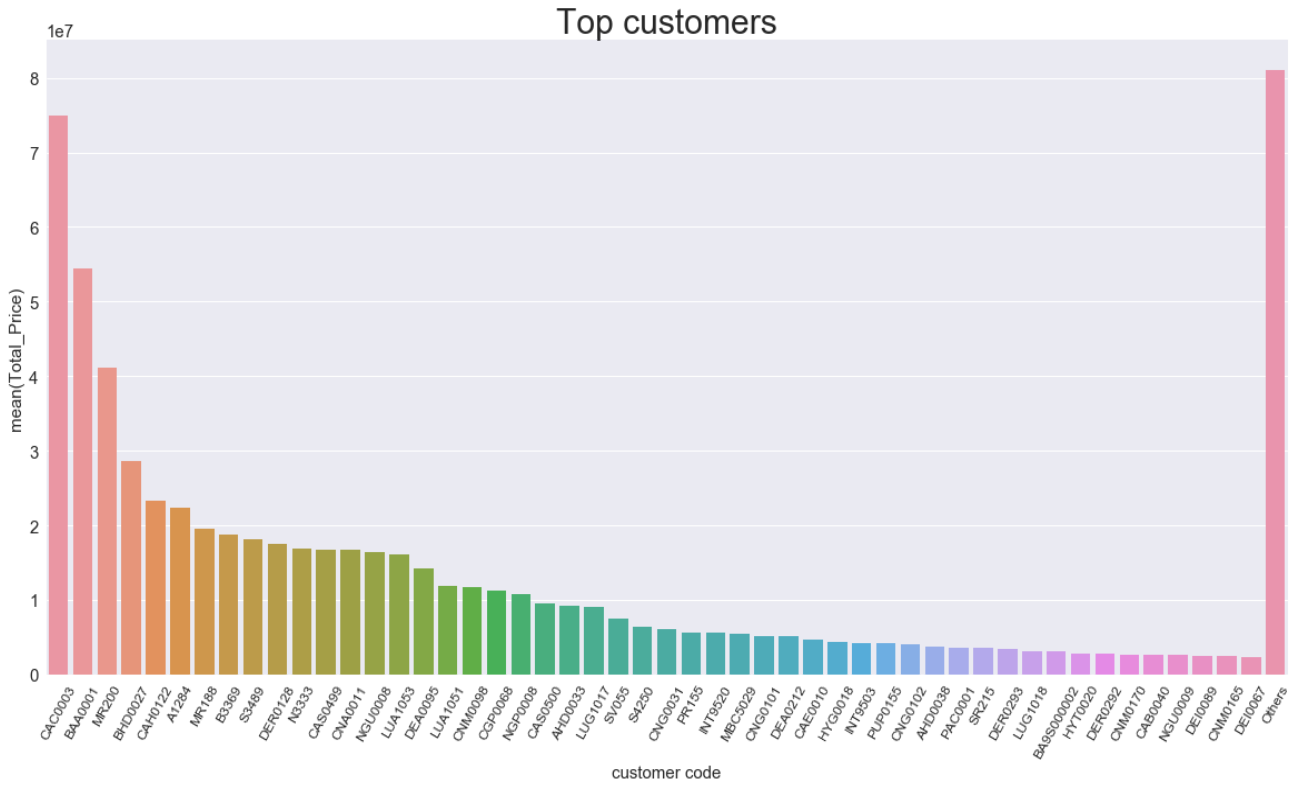


Fig 7.2.1 Graph showing the top 50 value adding customers to the company

The above graph shows the top 50 value adding customers , facilitating the company's profit. It is inferred that more than 90% of the customer side profit is from these top 50 customers , hence they are a major asset to the company.

Hence it would be highly valuable if company focuses more on these customers to maintian their trust , so they continue to invest . At the same time, attracting the customers who are not currently investing majorly on the company's products would be valuable for growth of company .

7.2.2 Region Wise Profit

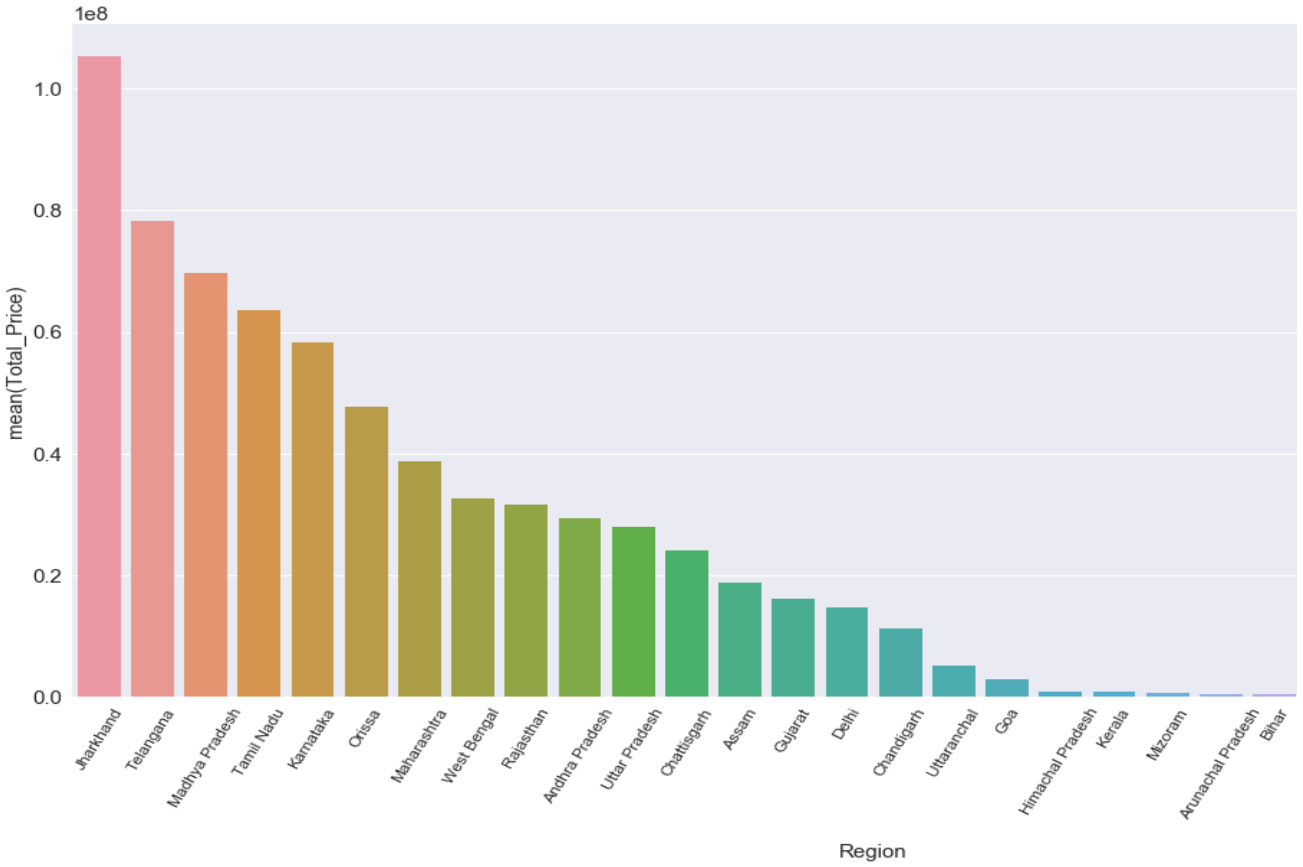


Fig 6.2.2 The graph shows the major regions investing on the company's products.

Similar measures are taken, as suggested in case of customer sales.

7.2.3 Material Wise Profit

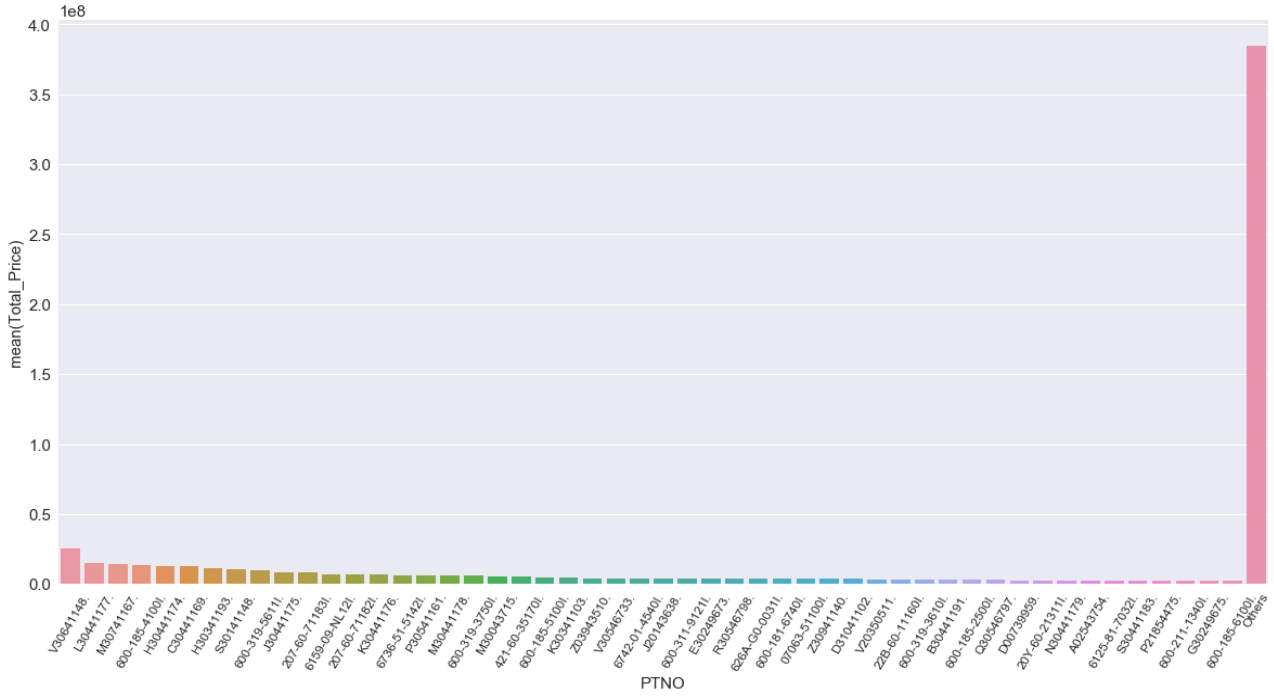


Fig 7.23 Above graph shows the major value adding materials produced by the company. The highest profit adding products should be noted and measures similar to customer sales are advised.

7.2.4 Model Wise Profit

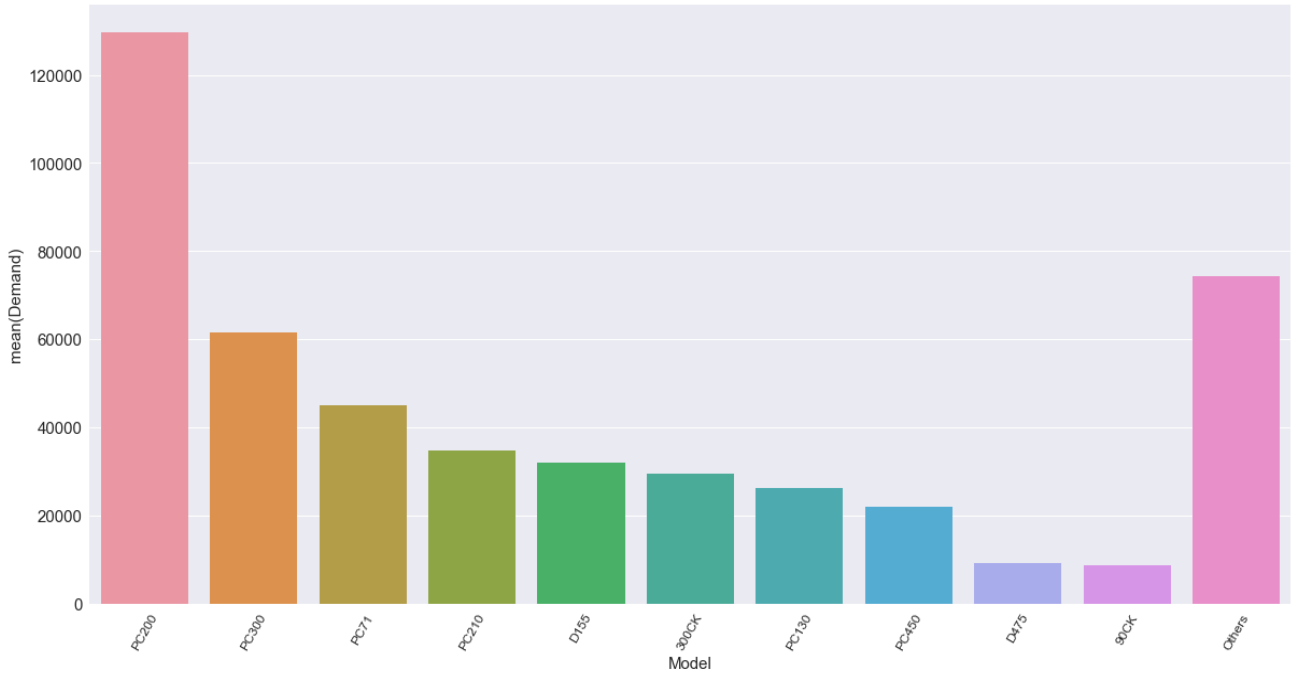


Fig 7.2.4 The above graph shows the top 10 value adding models released by the company , facilitating to it's profit.

It is inferred that more than 90% of the profit is from these top 10 models , hence they are major value adders to the company, performing significantly in the market .

Hence it would be highly valuable if the company works on releasing more such models, with similar features and performance .

7.2.5 Season Wise Sales

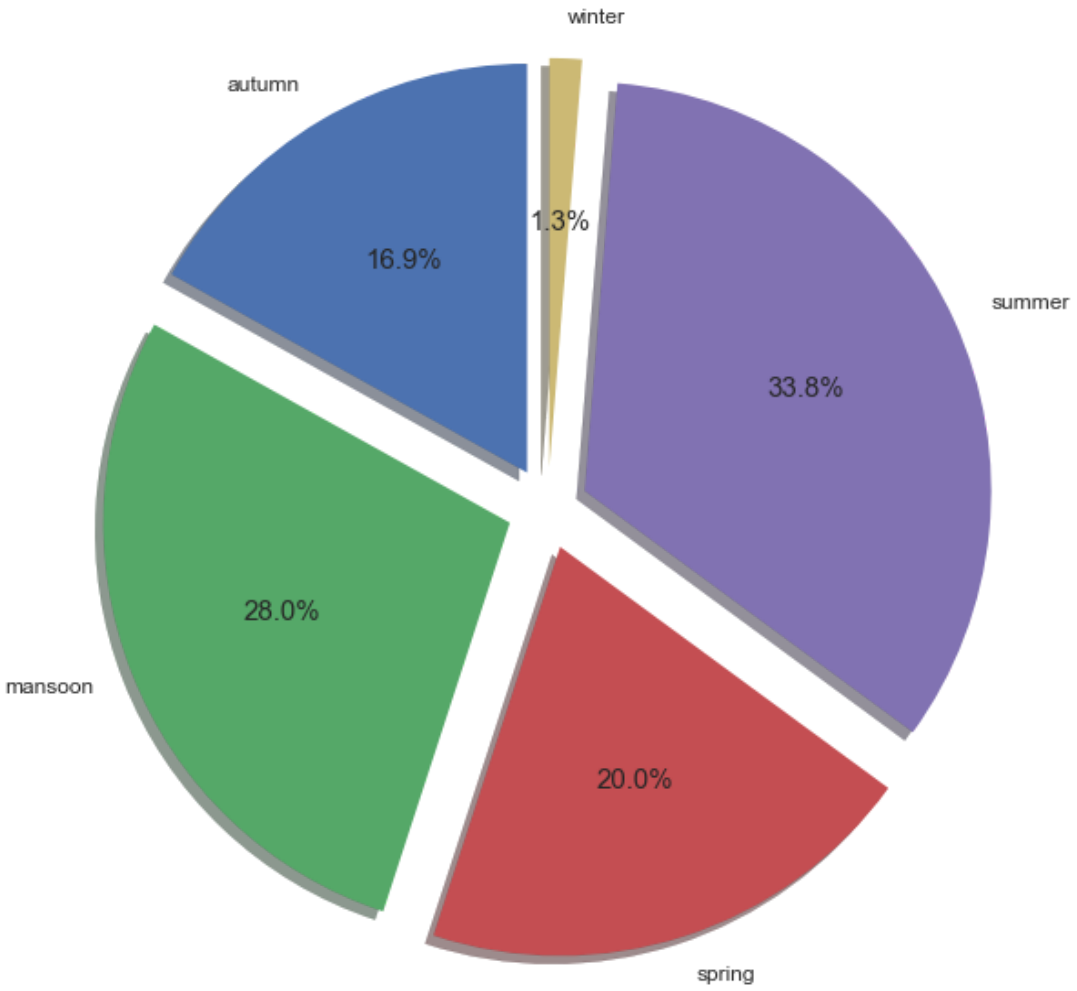


Fig 7.2.5 The above pie plot gives the season wise demand forecast i.e. how demand of the product in the company varies with seasons.

From the above graph, we can see that sales in summer is maximum i.e. 33.8% and sales in winter is minimum i.e. 1.3%. From this we can infer that, since the sales will be maximum in summer, company should boost up its production to satisfy its customers. And since the sales is minimum in winter company should measures to increase the sales.

7.3 DETERMINING MATERIALS WHICH ARE OUT OF STOCK

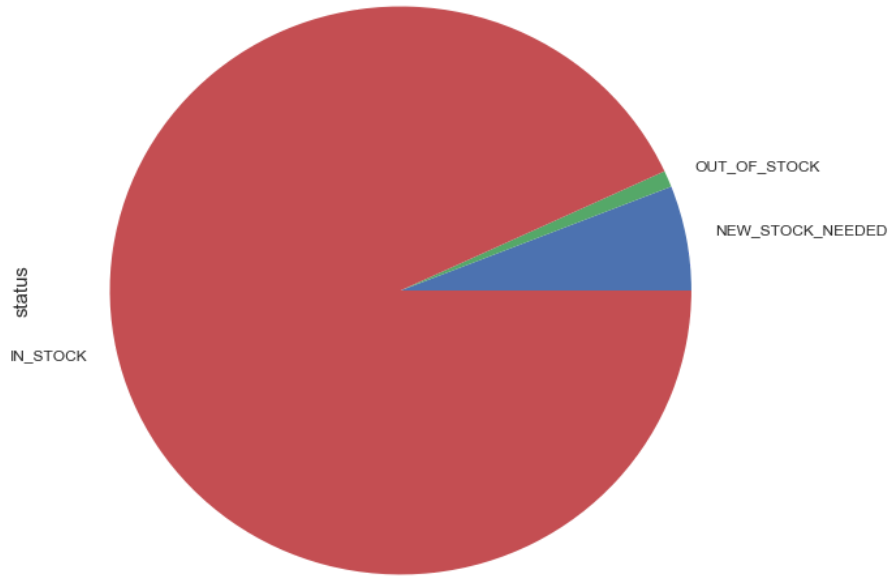


Fig 7.3.1 The above pie plot shows the amount of materials which are out_of_stock, in_stock and few_stock_needed. From this we can infer that few materials are in out_of_stock and new stocks needed. Action must be taken by the company to overcome this.

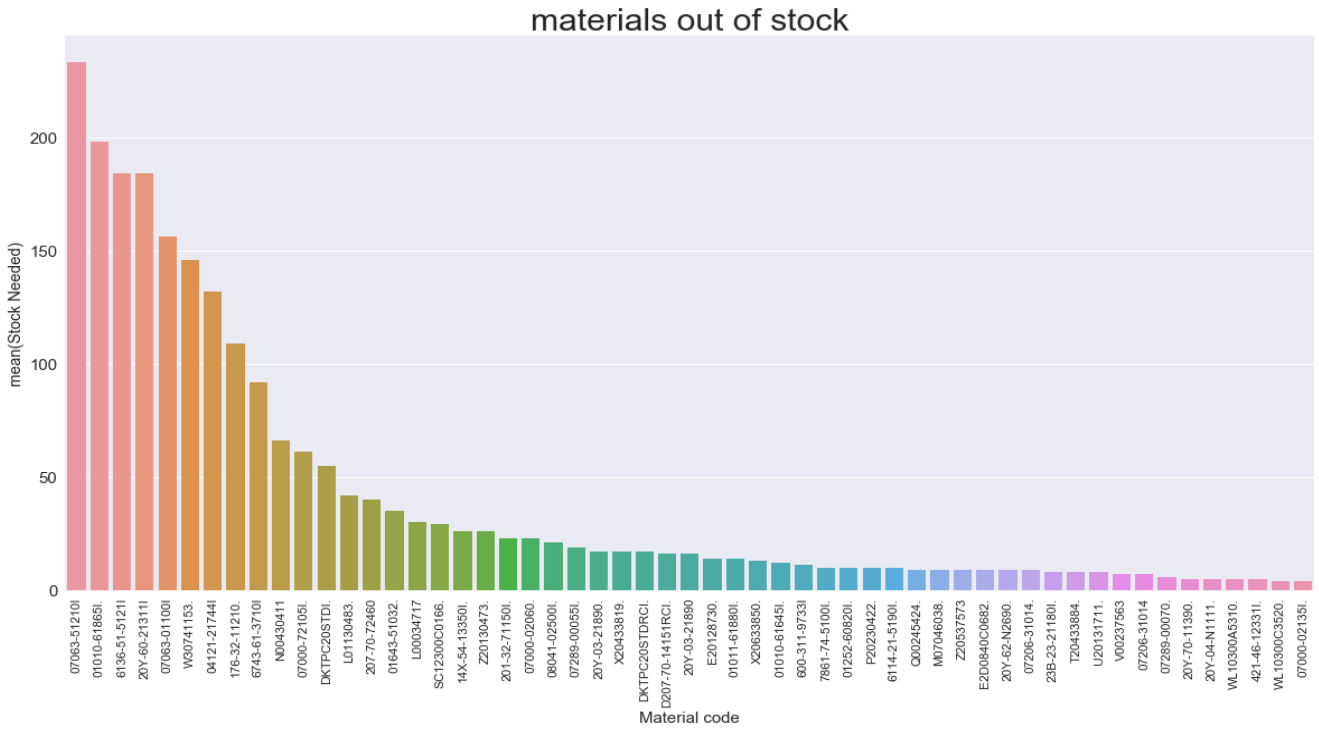


Fig 7.3.2 The above graph shows the materials which are out_of_stock . Here we can see few materials with higher values. We can say that, these materials are needed immediately and also the stock of rest of the materials should be added.

7.4 CHECKING POSSIBLE REASONS FOR DELAY FROM CONSIGNMENT TO DELIVERY

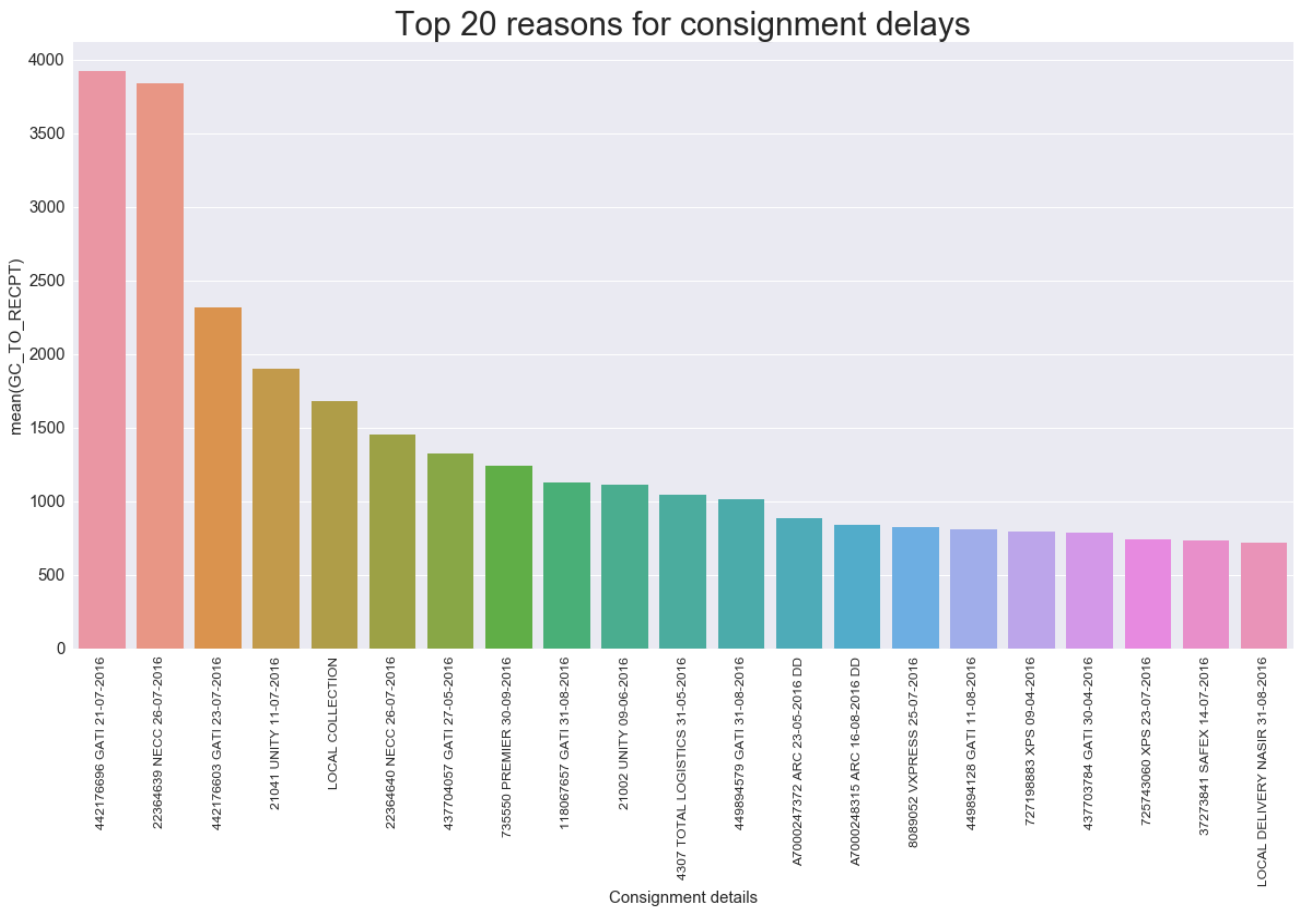


Fig 7.4 The above graphs shows he possible reasons for delay in consignment to delivery .The x axis gives the consignment details and y axis gives the mean time take by them. This can be reduced by making sure the materials are in stock and by improving the mode of courier services .

7.5 QUERY DRILL DOWN

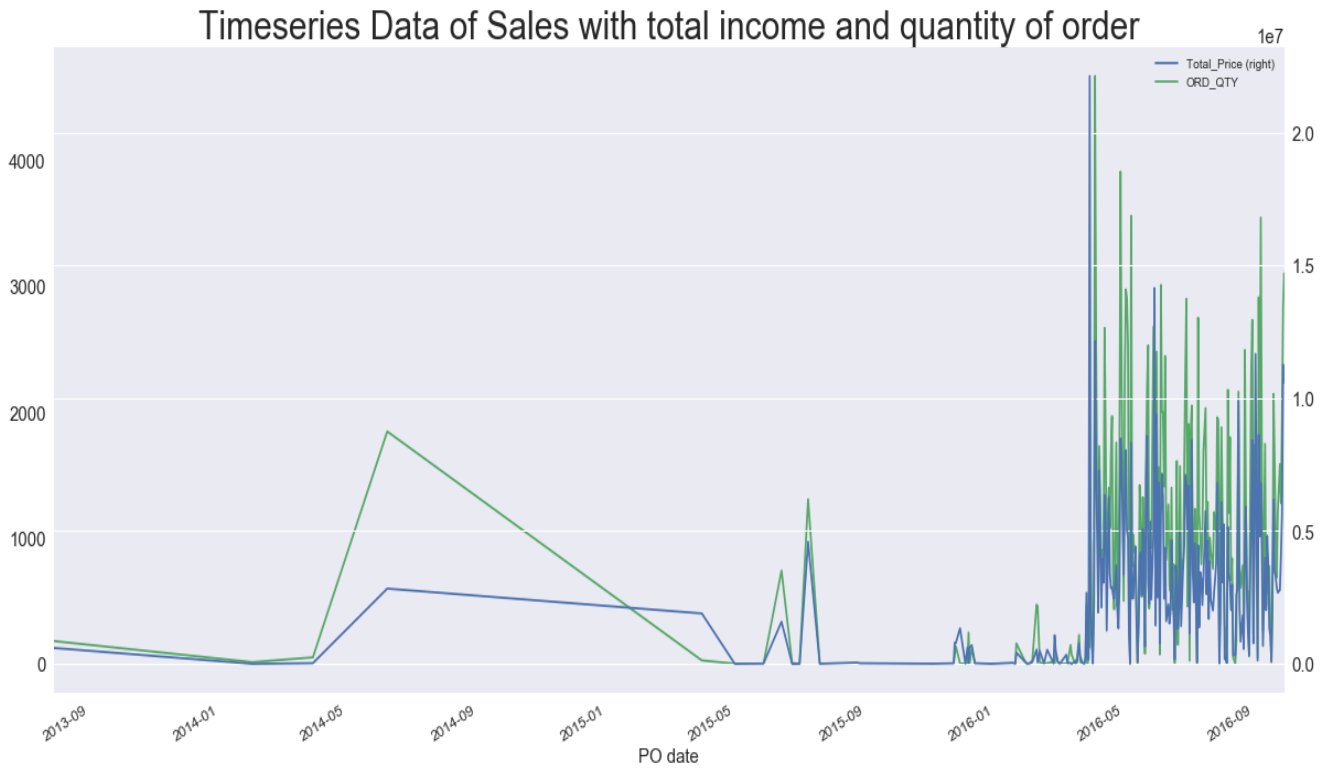


Fig 7.5 The above graph gives the total income and quantity of order with time.

Analysis is carried out on about 3 years of data. And from the above graph we can say that that is huge improvement in the sales from previous year to next year.

We will further narrow down this analysis to each month of the year and day of the month.

7.5.1 Income And Order Quantity For Months Of The Year

Fig 7.5.1 The above graphs give the sales for each month of the year. Here we can see that, the orders are higher in the middle of the year and lower at the beginning and end. This shows the optimal time to pump up the production of parts.

7.5.2 Income And Order Quantity For Days Of The Month

Fig 7.5.2 The above graph gives the sales for each day of the month. Here we can see that the demand is maximum at the end of the month.

7.6 DEMAND FORECASTING

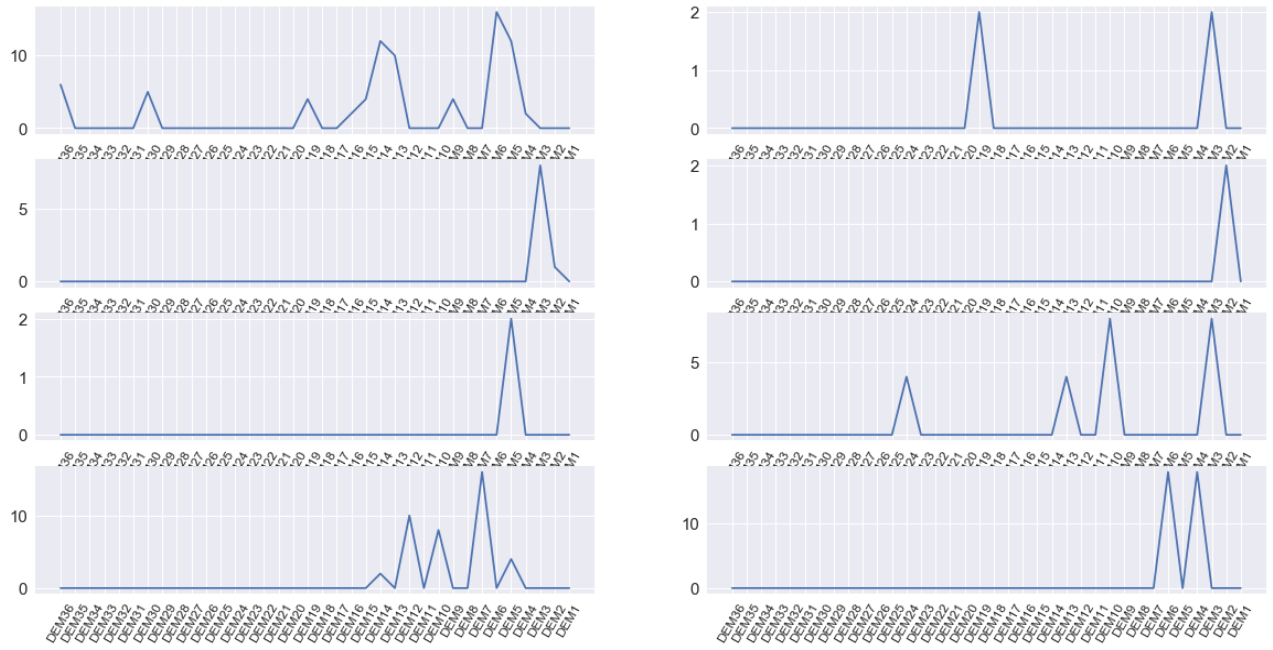


Fig 7.6.1 The above graph gives the demand of few materials for 36 months. Here we can see that the is no periodicity, hence we have to find other factors which influence these purchases or we could try to represent data in some other form. So, instead of data per month, we divide the data as to have data per three months. This allows us to predict the demand for next three months which would be aggregated better than data per month

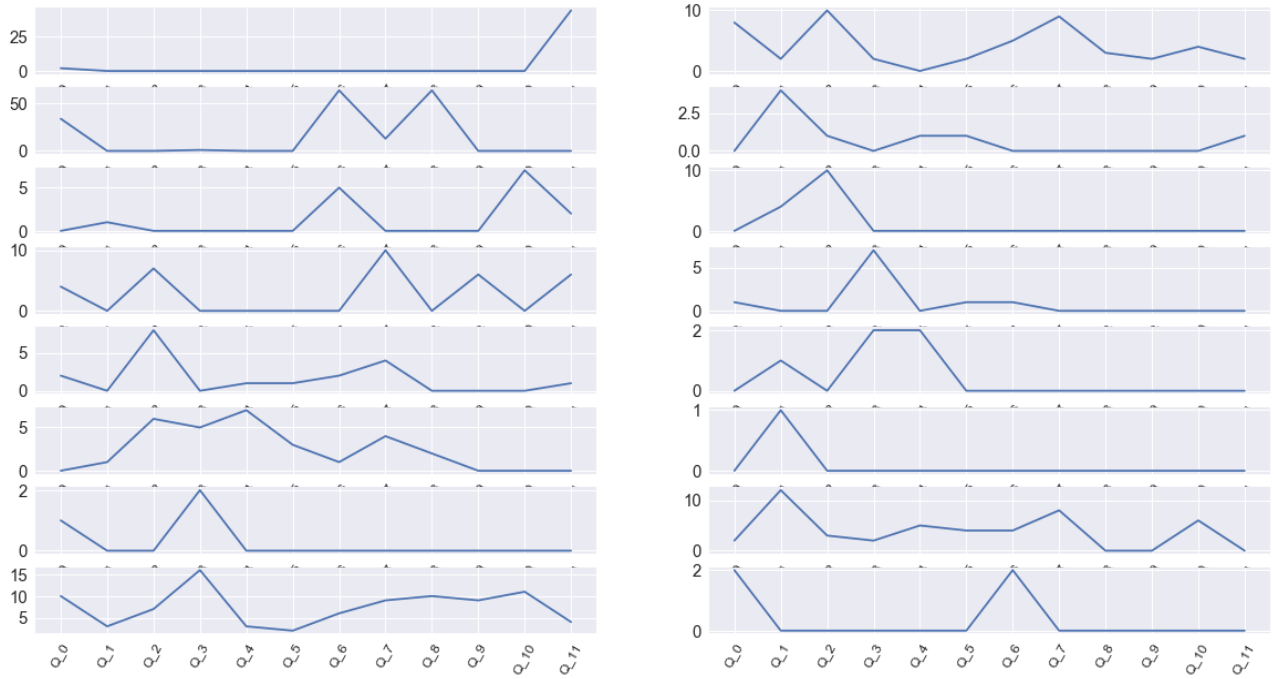
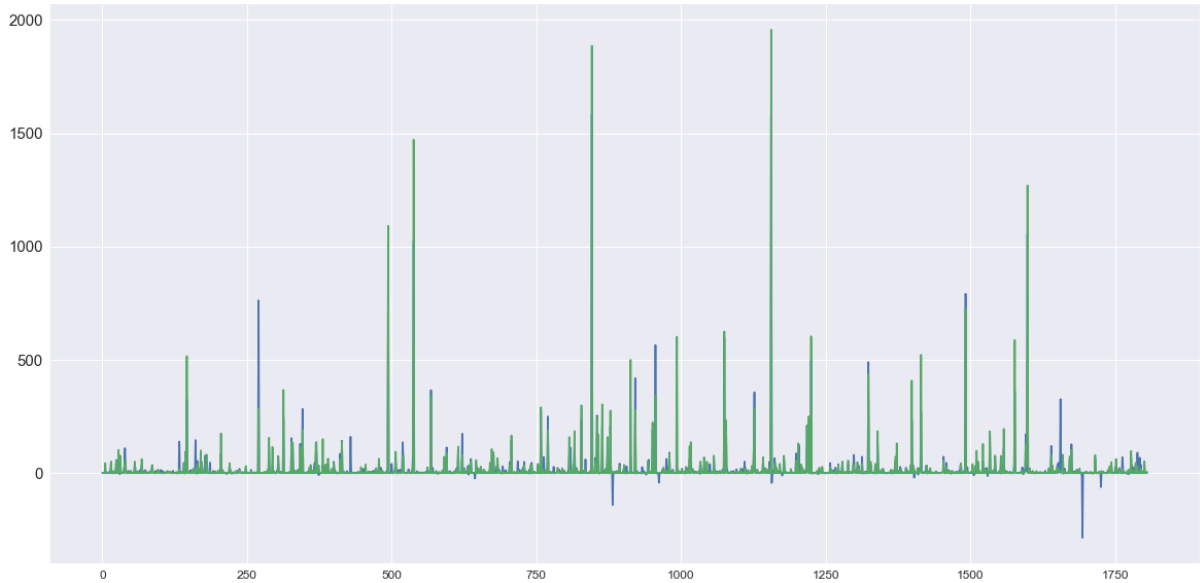


Fig 7.6.2 The above graph shows demand for interval of three months. Here we can see some sort of patterns. This could be predicted well.


2000
1500
1000
500
0
0
250
500
750
1000
1250
1500
1750


2000
1750
1500
1250
1000
750
500
250
0
0
25
50
75
100
125
150
175
200

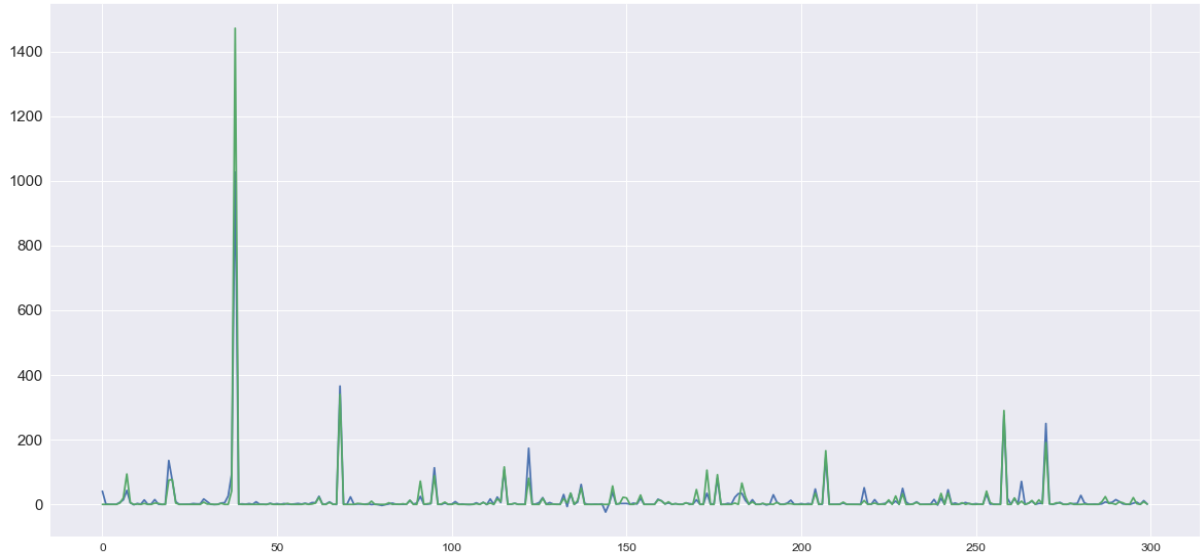

1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250
300

Fig 7.6.2 The above three graphs give the comparison between predicted values and actual values.

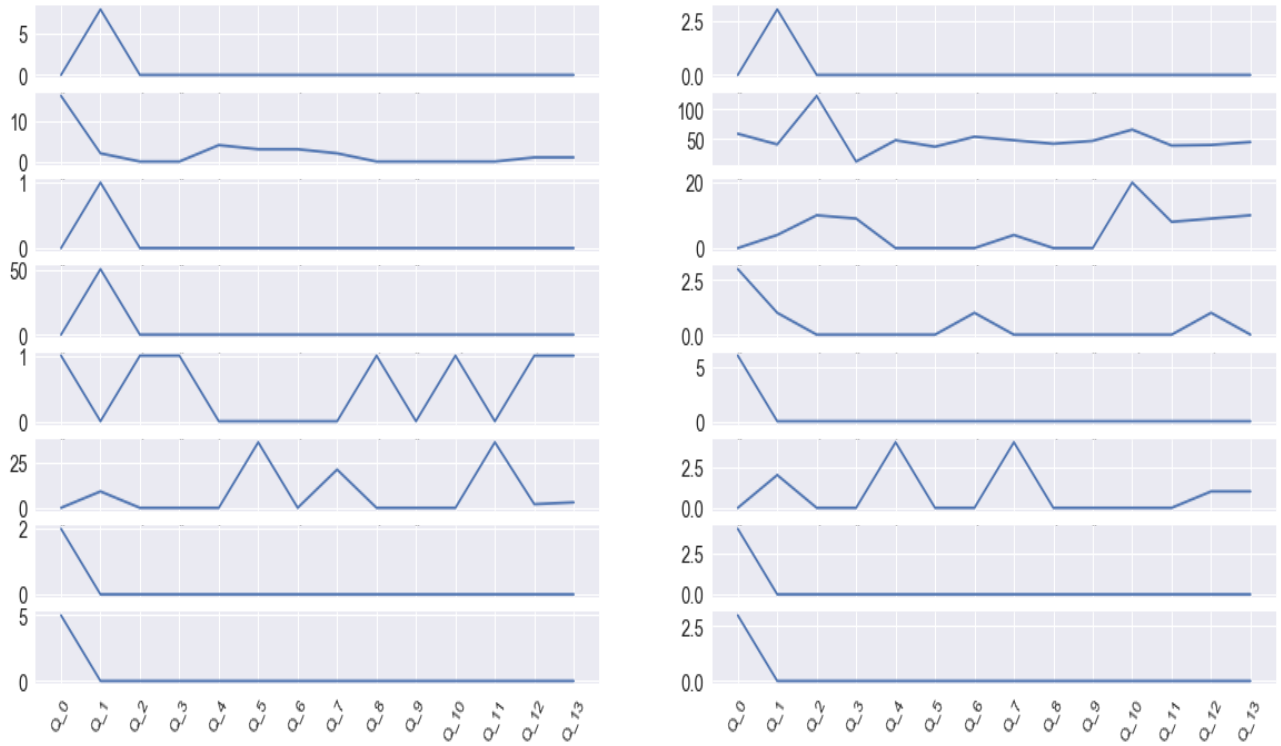


Fig 7.6.3 This graph gives the demand for materials after predicting for two more intervals. We have used Ridge regression to predict these values.

CONCLUSION

Efficient material & supply chain management is crucial for the success of any organization. Efficient material management methods and supply chain analytics helps in minimising working capitals and improves decision making process.

An effective supply management system is essential to improve entire gamut of SCM and Analytics plays a key role in improving. As a part of this project, taken a case study at L&T Construction & Mining Business order to cash cycle and analysed time taken between different legs of operation and finding and recommendation are:

1. **Order To Allocation (Delivery Note Generation)**

   Only 25 % of material/goods are allocated on the same day and 50% within 3 days. It needs immediate attention to reduce OTC (Order to Cash) cycle time

   We propose Alerts for material where stocks touched safety levels & Re-order levels on daily basis, which helps to place orders on time and to improve availability.

2. **Allocation To Invoice Generation**

   Time taking for conversion of allocated material to Invoice is 3 days (95 %), even then it looks good, but need attention.

   We propose to look into systems and movement of material picking & packing at warehouse. It is recommended to reduce this time to 1 day or 2 days to complete 100% of conversion. This is the area where organization variable assets converting to revenue.

3. **Invoice To Consignment**

   Time taking to move material out from where house after invoicing is 3 days (100 %) and this is to be maintained as it is under control.

4. **Consignment To Reaching Customers**

   Time taking to reach materials after consignment booking is 6 days (91 %). This looks good, but completing 100% is taking long time and it needs attention for funds flow and customer satisfaction.

   We propose to monitor/track consignment on regular basis and take corrective action.

5. **Order To Receipt (Total cycle time of OTC)**

This is the total time taken to fulfil customer requirement. As per analysis made it takes 15 days to complete 60% of orders and 30 days to complete 85 % of orders. In present competitive world, this time to be minimised to improve business.

As first step/immediate action, we propose to select effect methods/forecasting packages to improve material availability and also select right transport and mode of transport.

We also analysed data segments like customer, sales region, applicable models etc.., which gives insight about major contributes to business and improving areas for further business improvement.

BIBILOGRAPHY

BOOKS

[1] Essentials of Supply Chain Management by Michael H. Hugos

[2] Logistics and Supply Chain Management by Martin Christopher

[3] Strategic Supply Chain Management: The Five Core Disciplines for Top Performance by Shoshanah Cohen and Joseph Roussel

[4] Supply Chain Management: Strategy, Planning, and Operation by Sunil Chopra and Peter Meindl

[5] Designing and Managing the Supply Chain: Concepts, Strategies and Case Studies by David Simchi-Levi et al

[6] Integral Logistics Management: Operations and Supply Chain Management Within and Across Companies by Paul Schönsleben

[7] Logistics Management and Strategy: Competing through the Supply Chain by Alan Harrison and Remko Van Hoek

[8] Manufacturing Planning and Control for Supply Chain Management by F. Robert Jacobs et al

[9] Supply Chain Logistics Management by Donald Bowersox et al

[10] The Handbook of Logistics and Distribution Management: Understanding the Supply Chain by Alan Rushton et a

ARTICLES

[1] The Institute, Connecting the Dots with Big Data http://theinstitute.ieee.org/ns/quarterly_issues/tisep14.pdf

[2] http://ieeevis.org/year/2013/tutorial/visweek/visualization-and-analytics-python

[3] big data IEEE paper 2016 http://www.engpaper.com/big-data-2016.htm

[4] http://www.kdnuggets.com/2015/09/30-hbr-articles-analytics-big-data-science.html (Harvard business review articles on data science, big data, Analytics)

[5] https://scm.ncsu.edu/scm-articles/article/innovative-approaches-to-supply-chain-quality (Innovative Approaches to Supply Chain Quality)

LINKS

[1] https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-learn-data-science-python-scratch-2/

[2] https://www.lynda.com/Numpy-tutorials/Introduction-Data-Analysis-Python/419162-2.html

[3] http://bigdata-madesimple.com/step-by-step-approach-to-perform-data-analysis-using-python/

[4] https://www.coursera.org/learn/python-data-analysis

[5] https://pandas.pydata.org/pandas-docs/stable/10min.html

[6] https://www.dataquest.io/blog/pandas-python-tutorial/

[7] http://jupyter.readthedocs.io/en/latest/install.html

[8] https://pythonprogramming.net/python-pandas-data-analysis/