APPLICATION OF MACHINE LEARNING IN SUPPLY CHAIN MANAGEMENT – LEVERAGING SUPERVISED MACHINE LEARNING FOR EFFICIENT SUPPLIER SELECTION

Thesis submitted
In Partial Fulfillment of the Requirements for the Degree of

MASTER OF TECHNOLOGY

in

Industrial Engineering and Management

by

Subadeep Das
(Roll No. 2K22/IEM/12)

Under the Supervision of

Dr. Suresh Kumar Garg
Professor, Department of Mechanical Engineering
Delhi Technological University

To the

Department of Mechanical Engineering

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-110042, India

June, 2024

DEPARTMENT OF MECHANICAL ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

I want to convey my heartfelt gratitude to my professor, **Dr. S. K Garg**, for their support and encouragement during the research and writing of this project. His expertise in the subject matter greatly contributed to the depth and quality of the project He showed us the path to achieve our targets by explaining all the tasks to be done and explained to us the importance of this project as well as its industrial relevance. He was always ready to help us and clear our doubts regarding any hurdles in this project. Without his constant support and motivation, this project would not have been successful.

I would also like to extend my heartfelt appreciation to the **Hon'ble Vice-Chancellor, Delhi Technological University,** and **Prof. B. B. Arora**, **Head of the Department of Mechanical Engineering, Delhi Technological University** and **Assoc. Prof. Pravin Kumar** for their support and encouragement. And also for providing us with this platform to explore new avenues in life and carry out research.

Lastly, I would like to acknowledge my family and friends for their unwavering support, encouragement, and understanding throughout this research endeavour.

Place: Delhi   Subadeep Das
Date: June, 2024   (2K22/IEM/12)

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CANDIDATE'S DECLARATION

I Subadeep Das (2K22/IEM/12) hereby certify that the work which is being presented in the thesis entitled "Application of Machine Learning in Supply Chain Management – Leveraging Supervised Machine Learning for Efficient Supplier Selection" in partial fulfillment of the requirements for the award of the Degree of Master of Technology, submitted in the Department of Mechanical Engineering, Delhi Technological University is an authentic record of my own work carried out during the period from January, 2024 to May, 2024 under the supervision of Dr Suresh Kumar Garg.

The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.

Candidate's Signature

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CERTIFICATE BY THE SUPERVISOR

Certified that Subadeep Das (2K22/IEM/12) has carried out their search work presented in this thesis entitled "Application of Machine Learning in Supply Chain Management – Leveraging Supervised Machine Learning for Efficient Supplier Selection" for the award of Master of Technology from Department of Mechanical Engineering, Delhi Technological University, Delhi, under my supervision. The thesis embodies results of original work, and studies are carried out by the student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date: June, 2024

Signature

Dr. Suresh Kumar Garg

Professor

Department of Mechanical Engineering

Delhi Technological University, Delhi

Application of machine learning in supply chain management – leveraging supervised machine learning for efficient supplier selection

Subadeep Das

ABSTRACT

The aim of this paper is to discuss the inefficiencies in supplier selection which occur in the shipping and logistics industries. These inefficiencies interrupt the regular flow of goods in the global economy and obstruct optimal performance. Traditional methods that rely on human review and subjective standards sometime leads to poor selection and missed opportunities. The purpose of this work is to introduce new methods that use supervised machine learning(SML) algorithms to increase the accuracy and efficiency of vendor selection process. With the aim of enhancing selection accuracy through the use of supervised machine learning algorithms in our research, "Application of Machine Learning in Supply Chain Management – Leveraging Supervised Machine Learning for efficient Supplier Selection" focuses on the supplier selection process based on available features and instances. This research uses a two-phase methodology. Initially, we examine historical information from previous procurement activities, encompassing vendor performance metrics. Second, we create a predictive model for vendor selection using a variety of machine learning algorithms, including Random Forests, Decision Tree, and KNearest Neighbor models. These algorithms were trained and compared to find the model with best accuracy in forecasting vendors for upcoming contracts. While comparing the different Supervised Machine Learning (SML) models and SML with sparrow search algorithm as feature selection technique, it has been found that, the accuracy of vendor selection was significantly higher for the Random Forest which is 92.69% followed by the decision tree 86.4% KNearest Neighbor(KNN) 85.4%, Adaboost 84.21%. The random forest with the SSA algorithm's accuracy was the highest. The study also emphasizes how important it is to train machine learning models using large data sets. The shipping logistics industry can greatly optimize its resources by integrating machine learning algorithms with data analytics. This study demonstrate how the application of supervised machine learning algorithms can fundamentally alter the process of selecting suppliers for logistics and transportation projects. Predictive modelling and historic data can help organisation make well informed decisions, reduce disruptions, and enhance the overall effectiveness of their supply chain operations. The study findings indicate that more research is needed into merging SML algorithms optimization techniques to consistently improve supplier selection's accuracy. This would boost operational efficacy and industrial competitiveness.

Keywords: Supply chain management, Shipping Industries, Machine learning, Random Forests, Decision Tree, and KNearest Neighbor, Supplier selection, Sparrow Search Algorithm.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

SCM	Supply Chain Management
ML	Machine Learning
AI	Artificial Intelligence
SSP	Supplier Selection Process
SC	Supply Chain
API	Application Programming Interface
ARV	Antiretroviral
CSV	Comma Separated Values
USAID	United States Agency For International Development
SML	Supervised Machine Learning
KNN	K Nearest Neighbour
SSA	Sparrow Search Algorithm
PPV	Positive Predictive Value
CNN	Convolutional Neural Network
PCA	Principal Component Analysis

CHAPTER 1

INTRODUCTION

For the supply chain (SC) to be successful and efficient, the supplier selection process (SSP) is essential. This procedure was traditionally based on the manual evaluation of multiple criteria, which frequently resulted in arbitrary choices and inconsistent results. However, a more objective and data-driven approach is required due to the ever-increasing complexity of SCs, which demands increased competitiveness, cost reduction, and risk mitigation (Gunasekaran., 2017). By uncovering hidden patterns and insights from massive volumes of supplier data, machine learning (ML) algorithms become increasingly potent instruments to revolutionise the SSP landscape(Wu et al., 2019). These algorithms have the ability to predict possible risks, find the best suppliers, and automate laborious operations, all of which can improve the SSP's efficacy and efficiency (Büyüközkan et al., 2011).

1.1 Supply Chain Management:

In order to provide end users with the goods and services they require, networks, channels, and node groups that are interconnected or related to one another collaborate. Effective supply chain management (SCM) has become a crucial differentiator for companies in today's globalised and dynamic marketplace, influencing important performance metrics like cost, responsiveness, and customer satisfaction(Christopher, M. 2020). Conventional supply chain management techniques were predicated on fragmented systems and data silos, which frequently led to inaccuracies, delays in decision-making, and inefficiencies (Gunasekaran et al.,2017). According to some sources (Ang et al., 2017; Scheibe & Blackhurst, 2018), supply chain management (SCM) is the "layout, making plans, execution, control, and tracking supply chain processes to create internet value, building a challenging architecture, leveraging global logistics, matching supply and demand, and measuring overall performance". SCM exercises aim for an integrated, multidisciplinary, multimethod approach and heavily rely on commercial engineering, structural engineering, logistics, information technology, operations research, and advertising (Haji et al., 2022).

The SC is based on the notion that well almost every commodity that involves a marketplace effect is the result of the efforts of various companies that comprise a supply chain. Even though SC has been in the vicinity for a while, most agencies have only recently paid attention to them as a value-add to their operations (Scheibe & Blackhurst, 2018)(Bajomo et al., 2022). Fig 1.1 shows the supply chain flow.

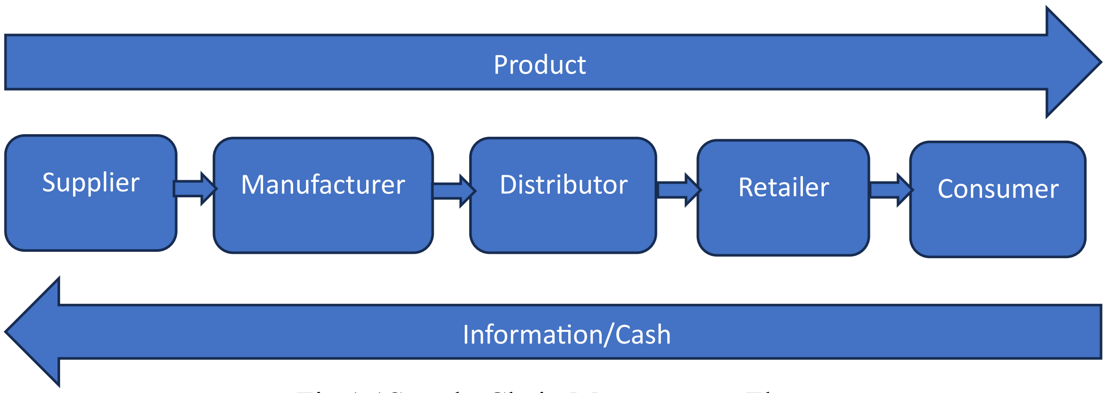


Fig.1.1Supply Chain Management Flow

1.2 Logistics:

The overall method of handling how assets are preserved and transferred to their final location. First, a military-based total time frame will be used to describe how the army procured, stored, and relocated systems and elements. With the growing complexity of providing businesses with resources and materials, and ultimately with the expansion of the SC, the concept of business logistics has evolved. Food and other consumables, as well as tangible goods like systems, materials, and resources, may also be included in the resources managed in logistics. Within supply chain management is the field of logistics management. Using in a way that best meets the needs of customers requires the efficient movement and storage of goods and services. A subset of supply chain management is logistics management. Logistics is the execution and management of the start factor and necessary for using in order to meet the needs of clients, as well as the efficient movement and storage of goods. With specialised simulation tools, logistics complexity can be analysed, visualised, and improved.

Any successful supply chain (SC) is built on logistics, which includes the strategic planning, coordination, and implementation of the actual physical transfer of goods and information (Ballou et al.,2019). Numerous factors, such as cost, responsiveness, inventory levels, and customer satisfaction, are impacted by its efficiency (Mentzer et al.,2009). Effective logistics operations are more important than ever in a marketplace that is becoming more complex and globalised, necessitating constant innovation and optimisation. Selection of supplier requires deliberation of both recurring and disturbance occurrences, i.e., frequent occurrences with minimal effect and infrequent occurrences with high impact (Gupta et al., 2019; Rajagopal et al., 2017).

1.3 Transportation:

Within the intricate network of supply chain management (SCM), transportation acts as the lifeblood, ensuring the timely and efficient movement of goods from raw materials to finished products. Four specific modes of transportation are used for cargo in logistics. These include carriages like wagons, trains, cargo ships, and aircraft carriers. Depending on the requirements of the company, many of these transportation techniques are employed in logistics (Wang & Jiao, 2022).

For cargo in logistics, there are specifically four types of transportation. These include automobiles, railroads, aircraft, and ships. In logistics, most of these modes of transportation are employed based on the needs of the business. Exceptional modes of transportation are employed based on the type of goods to be supplied, the range of goods, the location of the goods' delivery, and the freight cost associated with shipping. Each mode of transportation has unique advantages and disadvantages.

1.4 Artificial intelligence (AI) :

"Artificial intelligence (AI), is broadly defined as the capability of a virtual pc or laptop-managed robotic to carry out tasks on the whole related to smart beings." smart beings ideally mean humans. AI, when used for decision-making purposes in the supply chain, minimizes mistakes that may do with the aid of humans. AI in warehouse control makes the planning process lots less difficult with the aid of rushing up analysis time.

Additionally, even with the need to input a few demand-influencing factors and historical reports, demand prediction is much easier because AI will quickly compile all reliable predictions. The various subfields within AI studies focus on individual needs and the application of particular instruments. Traditional AI research objectives include planning, reasoning, natural language processing, ideation, and the capacity to manipulate and influence devices. Fig 1.4 represents nested hierarchy AI>>ML>>Deep Learning.

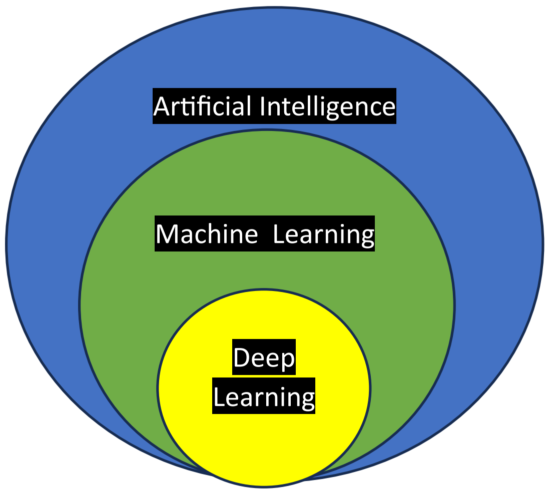


Fig.1.2 AI v/s ML v/s Deep Learning

Once limited to science fiction, artificial intelligence (AI) has quickly developed into a disruptive force in many different fields. This wide range of technologies includes approaches like computer vision, natural language processing, and machine learning that can mimic human intelligence. Industry, academia, and society as a whole have shown a great deal of interest in and investment in artificial intelligence (AI) due to its potential to automate tasks, solve complex problems, and reveal hidden insights. The most recent technological and clinical advancement is the sale of international web interconnection and logistics assiduity. It is anticipated that in the next ten to fifteen years, the rapidly developing field of statistical technological advancements, including big data analysis and its overall implications, will reach a developed period and give rise to logistics networks that possess extensive content and high-achieving features. References: Duan et al. (n.d.–c); Raman et al. (2018)

1.5 Machine Learning:

A subset of artificial intelligence called machine learning uses data to help a device identify patterns and build models of them in order to forecast an outcome while taking into account the inputted data. In logistics, machine learning may be responsible for analysing data sets and looking for better ways to handle operations (Lee & Chens, 2001). This can involve improving forecasting or accuracy requests, supporting inventory optimisation, or reacting to acquisition. Compared to the way things are typically handled in the goods industry, machine learning offers many advantages (Wuest et al., 2016a). The subsequent are 5 blessings of system mastering in Freight enterprise: 1. Analysing large quantities of information 2. Appearing repetitive features without making a mistake 3. Multitasking while appearing speedy 4. Working continuous root 5. Constantly improving forecasting as it learns from the data every time. The freight business creates huge volumes of records, with about four hundred million magnificence 8 shipments each year in the United States on its own. Computer systems may additionally sing lots of information bits for each cargo, including pickup and dropoff times, facility wait times, pricing, tender recognition, gasoline applied, and GPS coordinates throughout the cargo (Tsolaki et al., 2022b). The extra records supplied to ML models, the more powerful and effective they grow. To the most fundamental degree, machine studying uses programmed algorithms that get hold of and analyze input information to expect output values inside an acceptable range. As new statistics are given to those algorithms, they research and optimize their operations to improve and decorate performance, growing ‘intelligence’ over the years and getting more robust(Kosasih & Brintrup, 2022a). Fig 1.3 represent major machine learning technique. According to (Iqbal H. Sarker 2021)There are three different routes that invite exploration of ML's expansive terrain:

1.5.1 Supervised Learning:

One kind of machine learning is called supervised learning, in which an algorithm learns from labelled data to classify or predict new, unseen data. See it as similar to teaching a teacher through labelled examples; for instance, by displaying images of dogs and cats with tags identifying which is which. At that point, the instructor can identify, fairly accurately, whether a new image depicts a dog or a cat.

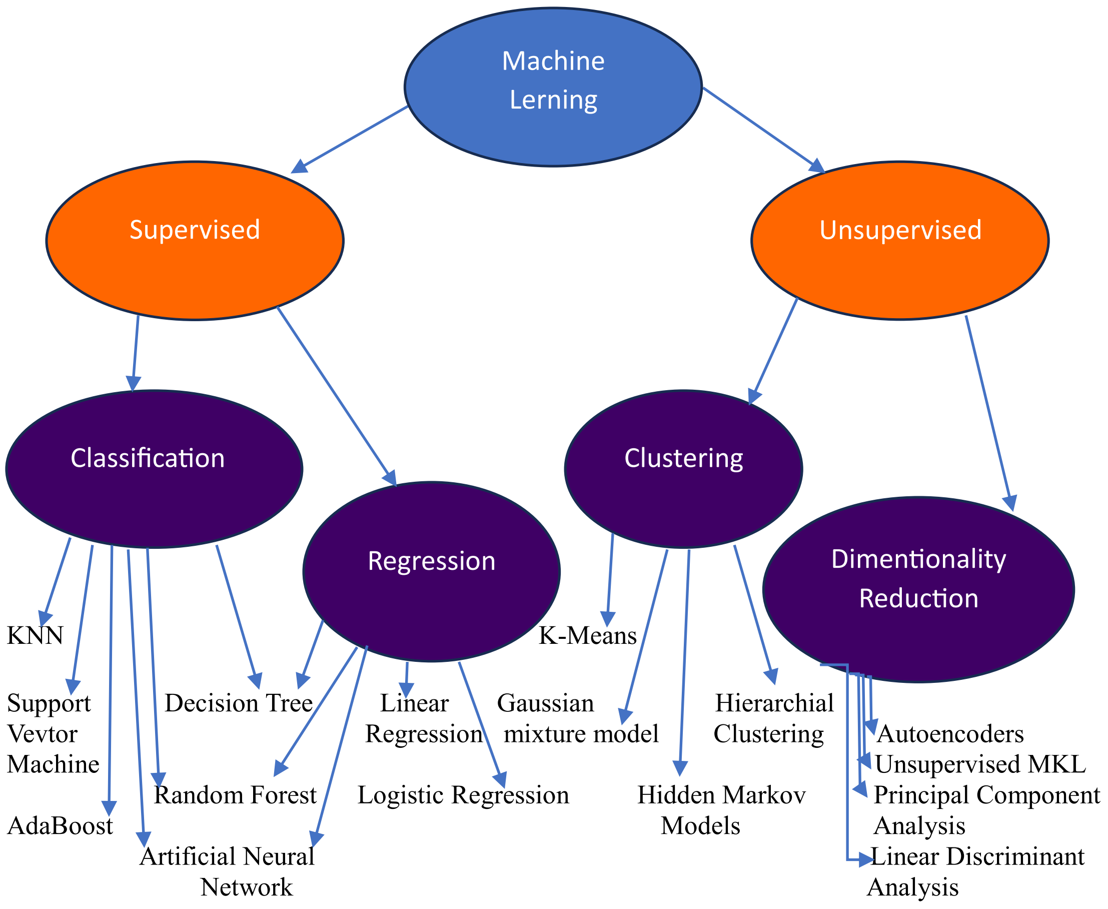


Fig.1.3 Categorization of major machine learning technique

1.5.2 Unsupervised Learning:

When an algorithm learns something on its own from unlabeled data, it is referred to as unsupervised learning. Imagine being thrust into a foreign nation with no knowledge or direction; you would have to use your own sense of observation to make sense of the world around you and learn about its structures and patterns.

1.5.3 Reinforcement Learning (RL):

The field of machine learning that focuses on autonomous agents making decisions is called reinforcement learning, or RL. Any system that can behave in reaction to its surroundings and make decisions without direct guidance from a human user is considered an autonomous agent. Examples of autonomous agents are robots and self-driving autos. Through trial and error and without any assistance from a human user, an autonomous agent can learn how to complete a task through reinforcement learning.1 It specifically tackles the issue of making successive decisions in unpredictable situations and holds potential for advancements in artificial intelligence.

1.5.4 Python Libraries:

Pandas In the dynamic landscape of data analysis, Python reigns supreme, and within its ecosystem, one library stands out as a versatile and robust workhorse: Pandas. More than just a collection of functions, Pandas has become the de facto standard for data wrangling and manipulation, empowering analysts across diverse fields to conquer data complexities with exceptional efficiency and ease. This comprehensive exploration delves into the history, applications, technical attributes, and resources that make Pandas the quintessential tool for data wranglers in the modern era. The genesis of Pandas can be traced back to 2008, where it emerged within the confines of AQR Capital Management as an internal tool for tackling financial data challenges. Recognizing its broader potential, the developers released Pandas as an open-source project in 2009, igniting a wave of enthusiastic adoption and propelling it into the hands of researchers, students, and professionals beyond the realm of finance. This open-source ethos, championed by a dedicated community and supported by NumFOCUS, has fueled Pandas' continuous evolution, solidifying its position as a cornerstone of the data analysis landscape. While its financial roots run deep, Pandas transcends the limitations of its origin story. Its reach extends across a vast spectrum of disciplines, empowering experts in fields ranging from neuroscience and bioinformatics to marketing and web analytics.

1.5.5 Matplotlib:

Matplotlib: Matplotlib produces stunning yet powerful visuals. A thriving community of around 700 contributors and approximately 26,000 comments can be found on GitHub for this Python charting package. It is often used for data visualization because of the graphs and plots that it generates. Additionally, it offers an object-oriented API that may be utilized to incorporate those plots into programs. Matplotlib's primary attributes are With the benefit of being free and open-source, it can be used as a substitute for MATLAB. It is compatible with numerous backends and output formats, so you can use it with any operating system or desired output format. To utilize MATLAB like a cleaner, Pandas itself can be used as wrappers around the MATLAB API. Minimal memory usage and better runtime behavior. Matplotlib is very helpful for correlation analysis of variables, displaying the models' 95 percent confidence intervals, identifying outliers with a scatter plot, and visualizing the data distribution to obtain immediate insights.

1.5.6 Sklearn:

The open-source machine learning toolkit Scikit-Learn was developed on top of well-known Python modules such as matplotlib, NumPy, and SciPy. For tasks like classification, regression, clustering, dimensionality reduction, and more, it offers a broad range of tools. Because Scikit-Learn has a straightforward and consistent API, it is easy to use for both novices and seasoned practitioners. The foundation of the library is the idea of estimators, which are entities that can be taught to generate predictions using data. These Estimators contain the learning and prediction algorithms as well as the parameters that govern how they behave. With its extensive feature set and robust component set, Scikit-Learn is an excellent tool for machine learning applications. A variety of methods for data preprocessing, including feature extraction, feature scaling, and handling missing values, are offered by Scikit-Learn. By doing this, the data is better prepared for feeding into a machine learning model. Numerous machine learning algorithms, such as clustering, regression, and classification, are included in the library. The uniform implementation of these algorithms facilitates the transition between various approaches. Tools for assessing model performance using measures such as accuracy, precision, recall, F1-score, and others are included in Scikit-Learn. It also allows rigorous model performance assessment methods such as cross-validation. Tuning Improving model performance requires hyperparameter tuning. For a given model, Scikit-Learn offers tools for both grid search and randomized search to determine the optimal set of hyperparameters. Pipelines enable you to combine model training and several data processing stages into a single object. This makes creating and implementing machine learning workflows easier.

CHAPTER 2

LITERATURE REVIEW

An efficient supply chain entails one that makes the optimal consumption of one's resources, whether they are financial, human, technical, or tangible. This reduces expenses for components and packaging while also reducing time waste. This kind of supply chain is often used in the industry that is related to emergency equipment and health. Studies are conducted on the efficient supply chain (Fahimnia et al., 2017).

The field of smart transport has attracted a lot of researchers, as it has been approached with the help of machine learning and internet of things techniques. Many machine scheduling models (Kosasih & Brintrup, 2022b) assume either an infinite number of transporters or even that jobs are distributed instantly from one location to another, with no associated transportation time. Prior studies have examined various transportation modes, geographical locations, and product categories that are transported through advanced technologies.

In order to operate efficiently, supply chains require close cooperation between the different parties involved in the process. While these distribution channels guarantee excellent customer service, they don't work well when dealing with erratic market demands. According to Hipólito et al., they generally adhere to a "just-in-time" management policy, which involves producing goods at retail facilities on demand to meet customers' needs immediately without holding intermediate inventory.

The demand for smooth services in logistics creates rapid growth in services like transportation in native American countries and Europe. MNCs have main impact in growth in their supplier and distributors, who require faster delivery and production of goods to consumers worldwide. Economic globalization has necessitated this need for efficient transportation and logistics services to support the global movement of goods (Rondinelli & Berry, 2000). In the 21st century, the world economy majorly impacted with intermodal and multi-modal transport network which plays essential roles in corporate logistics systems. The agile nature of an organization influences its capacity to develop in supply chain and gives innovative products to customers in prompt and cost-saving fashion in a constantly shifting, and worldwide competitive environment (Mehralian et al., 2015). The capacity of complete supply chain and its adjacent parts can quickly adjust with changes in operations to overcome dynamic changes and disturbance in demands of market called agility of supply chain. The goal is to manage work within organization by predicting future changes, and creates structure of networks to capitalize on new opportunities. The aim to enable business to operate with agility to deal with unpredictable demands. (Ismail & Sharifi, 2006).

Through dynamic order placement and the use of available data, digital-based manufacturing provides new opportunities to improve supplier selection. To improve this selection process, a combination of two methodologies, simulation, and machine learning, is developed and used to discover the best resilient supplier based on data. The study focuses on how well this technique performs in terms of enhancing supplier selection decisions. The study focuses on the performance of this approach in improving supplier selection decisions. (Cavalcante et al., 2019). Risk pertains to the supply chain due to uncertainty; the risk may of on-time delivery, reliability, finances, credit risk so on. There are studies considering the various risk criteria in the selection of suppliers and arrangement customer orders considering supply chain commotion (Ang et al., 2017; Wuest et al., 2016b)

For industries experiencing prolonged periods of low demand interspersed with periods of exceptionally high demand, the flexible supply chain model works best. This model stands out for its versatility, its capacity to modify fundamental manufacturing processes to meet client demands, and its simplicity in turning on and off. Take IBM's research on the flexible supply chain model, for instance (Cheng et al., 2006). A case study that uses a mid-size European company to apply an appropriate model. The study expanded its ability to handle uncertain conditions to include handling traditional supply chain networks. According to (Chatzikontidou et al., 2017), the analysis revealed that the latter model offers cost-benefits, making it less realistic but still cost-effective for the company implementing the process.

Numerous data-driven metaheuristics show extremely good outcomes and are outstanding optimisation algorithms. A detailed taxonomy based on search components is proposed to provide a thorough classification of these algorithms. This taxonomy includes both the low-level and high-level metaheuristic components used in the search process, as well as the target optimisation problem (Talbi, 2020).

The advent of the popular uprising of industry 4.0, the use of machine learning, big data, and AI-based technologies, the rapid adaptation of industries growing, and the advancement of technology all aptly reflect the thoughts and reflections of today's industry sectors and the possibilities to manage emotions, opinions, and contribute to the future industries (Haji et al., 2022; Lasi et al., 2014). Industry 4.0 is the best option because it seeks to increase communication amongst industrial decision-makers and offer tools for centrally controlling the organization's various activities (Nagar et al., 2021).

Scikit-learn library provides sophisticated atmosphere giving exceptional functionalities for many famous machine learning algorithms, by maintain simple interface coupled with python programming language. Because of this features, growing demand for data analysis and web industries by non-experts, as well as addition to other than computer engineering, such as physics or biology (Pedregosa FABIANPEDREGOSA et al., 2011). ML models such as Support Vector Machines (SVMs) and Decision Trees possess hyperparameters that require tuning for optimal performance. SVMs, for instance, require specification of hyperparameters such as regularization strength (often represented as C), input data scaling, choice of similarity kernel, and specific parameters associated with the selected kernel. Decision Trees, on the other hand, have hyperparameters relating to internal node generation and the pruning technique used during or after training. Despite being a well-known machine learning algorithm, Neural Networks are often regarded as difficult to incorporate into the Scikit-learn (sklearn) library due to the large number of hyperparameters they possess. (Bergstra et al., 2013).

Adoption of big data advanced technologies can generate significant real worth and pecuniary gain for businesses though and will shortly become an industry standard. By incorporating big data and SCM, this research offers a fresh characterization of the Supply Chain Operations Reference model (Ntabe et al., 2015). This study focuses on big data and how it might help existing supply chain practices by providing fresh opportunities, value addition, and operational excellence.

Public, corporate, and non-governmental organisations (NGOs) have altered their product and service offerings significantly as a result of the COVID-19 pandemic, especially in the aviation industry. The systems used for air freight transport have been greatly impacted by this. Given the enormous human and financial costs that the pandemic has caused worldwide, this paper focuses on the opportunities and challenges related to managing air freight transport, with a particular emphasis on sustainability (Bartle et al., 2021).

Previously studies conducted, like (Lee & Chens, 2001; Tsolaki et al., 2022b; Wuest et al., 2016a) have explored various machine learning methods and their applications in international freight transportation management. These researches used ML approaches for activities like demand forecasting, functioning, maintaining assets, and delivery of on-time performance predicting. The results demonstrate the adaptability and effectiveness of ML approaches across different areas of freight transportation management.

In order to operate efficiently, supply chains require close cooperation between the different parties involved in the process. While these distribution channels guarantee excellent customer service, they don't work well when dealing with erratic market demands. According to (Hipólito et al., 2020), they generally adhere to a "just-in-time" management policy, which involves producing goods at retail facilities on demand to meet customers' needs immediately without holding intermediate inventory.

Traditional methods struggle with large datasets, while ML techniques can efficiently manage and analyze vast amounts of data, providing more accurate and timely decisions. ML excels at recognizing patterns in large datasets and identifying critical supplier criteria precisely. ML model can adapt to new data and evolving conditions in the supply chain.ML integrates with other advanced technologies enhancing the overall decision-making framework and providing a competitive edge in supply chain management.

CHAPTER 3

MATERIAL & METHODOLOGY

To enhance supplier selection in the logistic sector, his study uses two-phase methodology. Initially, historical information from past procurement endeavors is reviewed. Next, various base ML algorithms are used to create a predictive model for supplier selection. The study compares the performance of these ML algorithms in terms of supplier selection accuracy, with a focus on identifying the most effective approach.

3.1 Problem Statement:

To find the best machine learning model for supplier selection based on model accuracy parameters. from the given features (freight cost, weight, unit prices, line-item value, and line-item quantity etc).

3.2 Data Acquisitions:

Data acquisition" refers to the procedure of gathering, acquiring, or importing data to perform analysis. It is an essential phase in any project that is data-driven and entails multiple important elements:

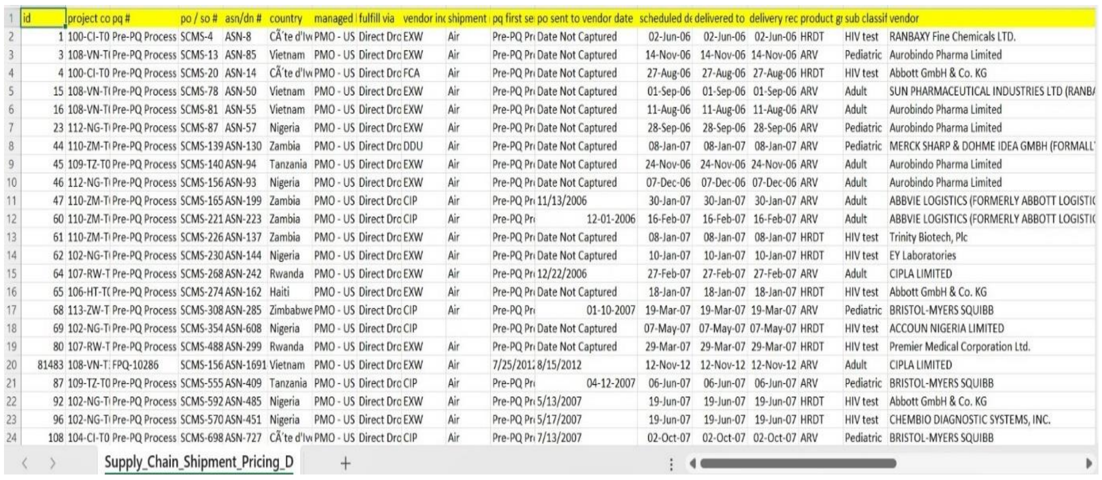
3.2.1. Locating Data Sources:

Find the locations of the pertinent data. Databases, files, APIs, external sources, and any other method that allows for the extraction of data could be included in this.

3.2.2. Data Collection:

Compile the information from the sources you've located. This may entail an automated extraction process, a manual collection process, or both. The type of data and its sources determine the methodology.

The dataset used in this study to make a choice of suppliers is acquired from Supply Chain Shipment Pricing Dataset | Development Data Library (usaid.gov) and can be downloaded as a CSV file. It contains data about the pricing and shipping of medical supplies which help us reach antiretroviral (ARV) and HIV lab materials for supporting countries. The cost of commodities and related supply chain costs— depicted through the transportation of these needed resources— are reflected in the dataset for use by those nations. Viewing this dataset with PQR information doesn't only double but rather enriches our perception on global expenditure towards specific health provisions. The data makes it possible to understand price changes' variations, cyclical trends, as well as volumes delivered to each nation since they're all outlined here. Any conclusions reached regarding the costs of shipping specific goods to particular countries, as well as lead times by product and country details may not be reliable. The unwieldy raw dataset containing 33 features and over 10,000 rows finds its way into legibility through a rather mundane savior: Microsoft Excel. But before this raw data can even begin its journey as input, there need to be feature selection parameters set up for supplier selection using Python libraries on Jupyter notebook software— a technological enabler that makes it possible for us, among others include using "pandas" library to import the dataset as a data frame. Python indeed stands out among the cacophony of programming languages with its wide acceptance within the data science community where it is hailed as the chosen one for processing these voluminous sets of information.



1	id	project co	pq #	po / so #	asn/dn #	country	managed	fulfill via	vendor inc	shipment	pq first se	po sent to vendor date	scheduled de	delivered to	delivery rec	product gr	sub classif	vendor
2	1	100-CI-T0	Pre-PQ Process	SCMS-4	ASN-8	CÃ´te d'Iv	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	02-Jun-06	02-Jun-06	02-Jun-06	HRDT	HIV test	RANBAXY Fine Chemicals LTD.
3	3	108-VN-T(	Pre-PQ Process	SCMS-13	ASN-85	Vietnam	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	14-Nov-06	14-Nov-06	14-Nov-06	ARV	Pediatric	Aurobindo Pharma Limited
4	4	100-CI-T0	Pre-PQ Process	SCMS-20	ASN-14	CÃ´te d'Iv	PMO - US	Direct Drc	FCA	Air	Pre-PQ Pr	Date Not Captured	27-Aug-06	27-Aug-06	27-Aug-06	HRDT	HIV test	Abbott GmbH & Co. KG
5	15	108-VN-T(	Pre-PQ Process	SCMS-78	ASN-50	Vietnam	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	01-Sep-06	01-Sep-06	01-Sep-06	ARV	Adult	SUN PHARMACEUTICAL INDUSTRIES LTD (RANB/
6	16	108-VN-T(	Pre-PQ Process	SCMS-81	ASN-55	Vietnam	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	11-Aug-06	11-Aug-06	11-Aug-06	ARV	Adult	Aurobindo Pharma Limited
7	23	112-NG-T(	Pre-PQ Process	SCMS-87	ASN-57	Nigeria	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	28-Sep-06	28-Sep-06	28-Sep-06	ARV	Pediatric	Aurobindo Pharma Limited
8	44	110-ZM-T(	Pre-PQ Process	SCMS-139	ASN-130	Zambia	PMO - US	Direct Drc	DDU	Air	Pre-PQ Pr	Date Not Captured	08-Jan-07	08-Jan-07	08-Jan-07	ARV	Pediatric	MERCK SHARP & DOHME IDEA GMBH (FORMALL'
9	45	109-TZ-T0	Pre-PQ Process	SCMS-140	ASN-94	Tanzania	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	24-Nov-06	24-Nov-06	24-Nov-06	ARV	Adult	Aurobindo Pharma Limited
10	46	112-NG-T(	Pre-PQ Process	SCMS-156	ASN-93	Nigeria	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	07-Dec-06	07-Dec-06	07-Dec-06	ARV	Adult	Aurobindo Pharma Limited
11	47	110-ZM-T(	Pre-PQ Process	SCMS-165	ASN-199	Zambia	PMO - US	Direct Drc	CIP	Air	Pre-PQ Pr	11/13/2006	30-Jan-07	30-Jan-07	30-Jan-07	ARV	Adult	ABBVIE LOGISTICS (FORMERLY ABBOTT LOGISTI(
12	60	110-ZM-T(	Pre-PQ Process	SCMS-221	ASN-223	Zambia	PMO - US	Direct Drc	CIP	Air	Pre-PQ Pr	12-01-2006	16-Feb-07	16-Feb-07	16-Feb-07	ARV	Adult	ABBVIE LOGISTICS (FORMERLY ABBOTT LOGISTI(
13	61	110-ZM-T(	Pre-PQ Process	SCMS-226	ASN-137	Zambia	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	08-Jan-07	08-Jan-07	08-Jan-07	HRDT	HIV test	Trinity Biotech, Plc
14	62	102-NG-T(	Pre-PQ Process	SCMS-230	ASN-144	Nigeria	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	10-Jan-07	10-Jan-07	10-Jan-07	HRDT	HIV test	EY Laboratories
15	64	107-RW-T	Pre-PQ Process	SCMS-268	ASN-242	Rwanda	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	12/22/2006	27-Feb-07	27-Feb-07	27-Feb-07	ARV	Adult	CIPLA LIMITED
16	65	106-HT-T(	Pre-PQ Process	SCMS-274	ASN-162	Haiti	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	18-Jan-07	18-Jan-07	18-Jan-07	HRDT	HIV test	Abbott GmbH & Co. KG
17	68	113-ZW-T	Pre-PQ Process	SCMS-308	ASN-285	Zimbabwe	PMO - US	Direct Drc	CIP	Air	Pre-PQ Pr	01-10-2007	19-Mar-07	19-Mar-07	19-Mar-07	ARV	Pediatric	BRISTOL-MYERS SQUIBB
18	69	102-NG-T(	Pre-PQ Process	SCMS-354	ASN-608	Nigeria	PMO - US	Direct Drc	CIP		Pre-PQ Pr	Date Not Captured	07-May-07	07-May-07	07-May-07	HRDT	HIV test	ACCOUN NIGERIA LIMITED
19	80	107-RW-T	Pre-PQ Process	SCMS-488	ASN-299	Rwanda	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	Date Not Captured	29-Mar-07	29-Mar-07	29-Mar-07	HRDT	HIV test	Premier Medical Corporation Ltd.
20	81483	108-VN-T:	FPQ-10286	SCMS-156	ASN-1691	Vietnam	PMO - US	Direct Drc	EXW	Air	7/25/201:	8/15/2012	12-Nov-12	12-Nov-12	12-Nov-12	ARV	Adult	CIPLA LIMITED
21	87	109-TZ-T0	Pre-PQ Process	SCMS-555	ASN-409	Tanzania	PMO - US	Direct Drc	CIP	Air	Pre-PQ Pr	04-12-2007	06-Jun-07	06-Jun-07	06-Jun-07	ARV	Pediatric	BRISTOL-MYERS SQUIBB
22	92	102-NG-T(	Pre-PQ Process	SCMS-592	ASN-485	Nigeria	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	5/13/2007	19-Jun-07	19-Jun-07	19-Jun-07	HRDT	HIV test	Abbott GmbH & Co. KG
23	96	102-NG-T(	Pre-PQ Process	SCMS-570	ASN-451	Nigeria	PMO - US	Direct Drc	EXW	Air	Pre-PQ Pr	5/17/2007	19-Jun-07	19-Jun-07	19-Jun-07	HRDT	HIV test	CHEMBIO DIAGNOSTIC SYSTEMS, INC.
24	108	104-CI-T0	Pre-PQ Process	SCMS-698	ASN-727	CÃ´te d'Iv	PMO - US	Direct Drc	CIP	Air	Pre-PQ Pr	7/13/2007	02-Oct-07	02-Oct-07	02-Oct-07	ARV	Pediatric	BRISTOL-MYERS SQUIBB

Fig.3.1 MS Excel SCM Delivery History Dataset

Overview of the Supply Chain Shipment Pricing Dataset

ID	Field Name	Field Description
1	ID	Primary Key Identifier of the line of data in our analytical
2	Project Code	Project code
3	PQ#	Price quote (PQ)
4	PO#	Order number: Purchase Order for Direct drop deliveries
5	ASN/DN#	Shipment No.: ASN for Direct drop deliveries, DN for from RDC deliveries
6	Country	Destination country
7	Managed By	SCMS managing office: Program management office(PMO) in the US or the relevant SCMS Field office
8	Fillfill Via	Method through which the shipment is fulfilled: via Direct Drop
9	Vendor Inco Term	The vendor inco term (also known as International Commercial Terms) for Direct Drop deliveries
10	Shipment Mode	Method by which commodities are shipped
11	PQ First Sent to Client Date	Date the PQ which first sent to client
12	PO Sent to Vendor Date	Date PO is first sent to client
13	Scheduled Delivery Date	Current anticipated delivery
14	Delivered to Client Date	Date of delivery to client
15	Delivery Recorded Date	Date on which delivery to client was recorded in SCMS information system
16	Product Group	Product group for item, i.e. ARV, HRDT
17	Sub Classification	Identifies relevant product sub classifications, such as whether ARVs are paediatric or adult, whether a malaria product is an artemisinin -based combination therapy(ACT), etc.
18	Vendor	Vendor Name

ID	Field Name	Field Description
19	Item Description	Product name and formulation from partnership for supply chain management(PFSCM) Item Master
20	Molecule/Test Type	Active drugs or test kit type
21	Brand	Generic or Branded name for the item
22	Dosage	item dosage and unit
23	Dosage Form	Dosage form for the item(tablet, oral solution, injection, etc.).
24	Unit of Measure	Pack quantity (pills or test kits) used to compute unit price
25	Line Item Quantity	Total quantity (packs) of commodity per line item
26	Line ItemValue	Total value of commodity per line item
27	Pack Price	Cost per pack(months supply of ARVs, pack of 60 test kit)
28	Unit Price	Cost per pill or per test
29	Manufacturing Site	identifies manufacturing site for the line item for direct drop and RDC deliveries
30	First Line Designation	Designates if the line in question shows the aggregated freight costs and weight associated with all items on the ASN/DN
31	Weight(KG)	weight for all lines on an ASN/DN
32	Freight Cost(USD)	Freight Charges Associated with all lines on the repective ASN/DN
33	Line Item Insurance(USD)	Line item cost of insurance, created by applying an annual flat rate(%) to commodity cost.

Table 3.1 Data Dictionary

This dataset, hosted by the (USAID) Development Data Library, provides detailed information about health commodity shipments and associated costs. It focuses specifically on Antiretroviral (ARV) and HIV lab shipments to countries supported by USAID.

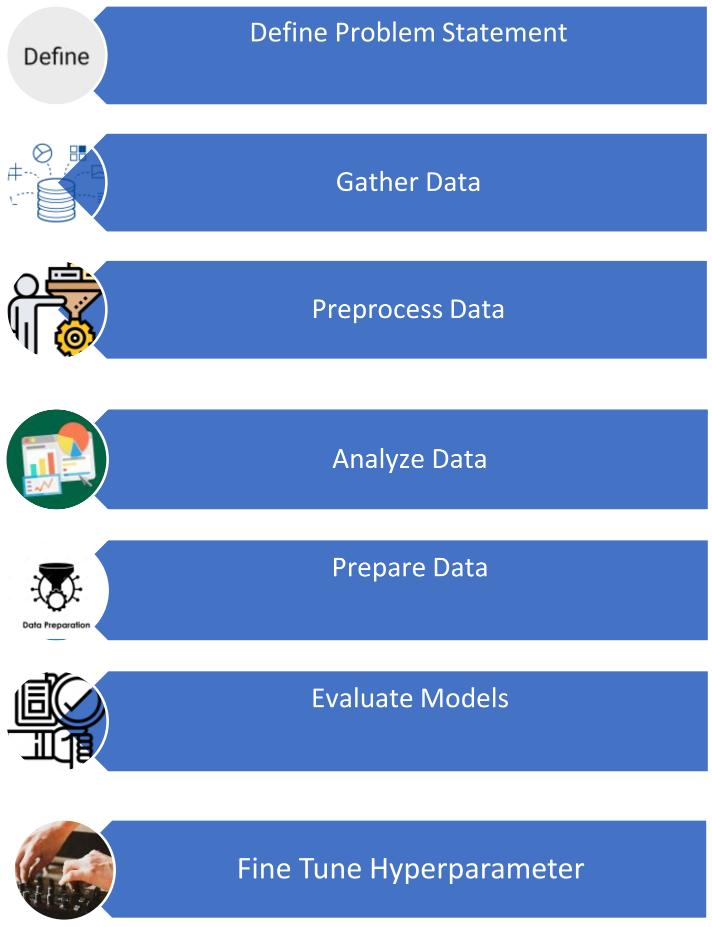
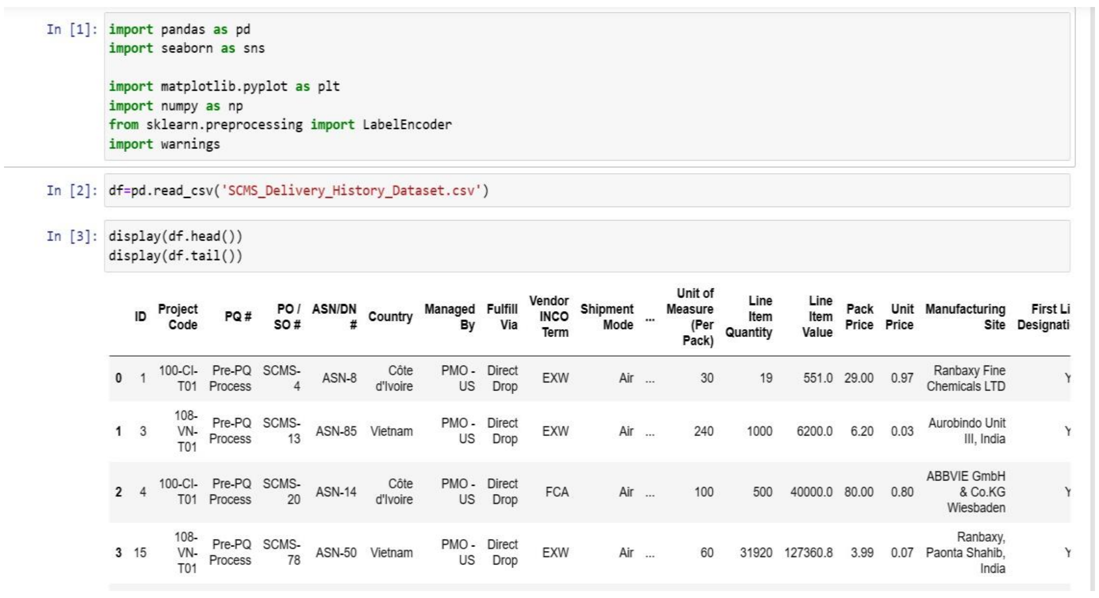


Figure.3.2 Methodology

3.3 Data Cleaning and Preprocessing:

A truly valuable asset that will improve customer satisfaction and yield accurate results is exceptional information. In order to fix data as needed, it involves identifying null values and statistical errors and then adding, deleting, or altering the data. Records cleaning makes the data better, makes it easier to provide consistent, more accurate statistics, and gives an agency's decision-makers trustworthy records.

Regression and classification issues can both be resolved with Supervised Machine Learning algorithms. The categorization problem for supplier selection is examined in this paper. In order to train the model and verify its efficacy, the SML algorithm uses raw data that has previously been made available as input data. It also provides test data.



```
In [1]: import pandas as pd
        import seaborn as sns

        import matplotlib.pyplot as plt
        import numpy as np
        from sklearn.preprocessing import LabelEncoder
        import warnings

In [2]: df=pd.read_csv('SCMS_Delivery_History_Dataset.csv')

In [3]: display(df.head())
        display(df.tail())
```

	ID	Project Code	PQ #	PO / SO #	ASN/DN #	Country	Managed By	Fulfill Via	Vendor INCO Term	Shipment Mode	...	Unit of Measure (Per Pack)	Line Item Quantity	Line Item Value	Pack Price	Unit Price	Manufacturing Site	First Li Designati
0	1	100-CI-T01	Pre-PQ Process	SCMS-4	ASN-8	Côte d'Ivoire	PMO - US	Direct Drop	EXW	Air	...	30	19	551.0	29.00	0.97	Ranbaxy Fine Chemicals LTD	Y
1	3	108-VN-T01	Pre-PQ Process	SCMS-13	ASN-85	Vietnam	PMO - US	Direct Drop	EXW	Air	...	240	1000	6200.0	6.20	0.03	Aurobindo Unit III, India	Y
2	4	100-CI-T01	Pre-PQ Process	SCMS-20	ASN-14	Côte d'Ivoire	PMO - US	Direct Drop	FCA	Air	...	100	500	40000.0	80.00	0.80	ABBVIE GmbH & Co.KG Wiesbaden	Y
3	15	108-VN-T01	Pre-PQ Process	SCMS-78	ASN-50	Vietnam	PMO - US	Direct Drop	EXW	Air	...	60	31920	127360.8	3.99	0.07	Ranbaxy, Paonta Shahib, India	Y

Fig.3.3 Python libraries on the Jupyter notebook software

3.4 Data Analysing:

Explore the data to understand its structure, characteristics, and patterns. This involves summarizing key statistics, generating visualizations, and gaining a preliminary understanding of the dataset.

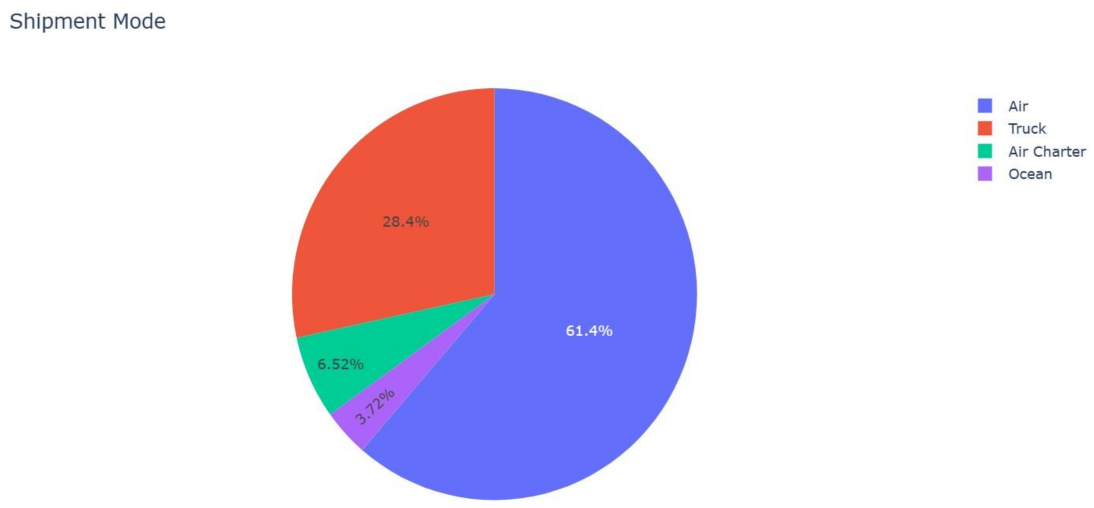


Fig.3.4 Percentage value of shipment mode

Majority of the health commodities are shipped by Air which constitute about 61.4%, followed by truck (28.4%), Air Charter (6.52%), Ocean (3.72). Air Charter and Ocean account for smaller proportions, potentially indicating specialized use cases or limitations.

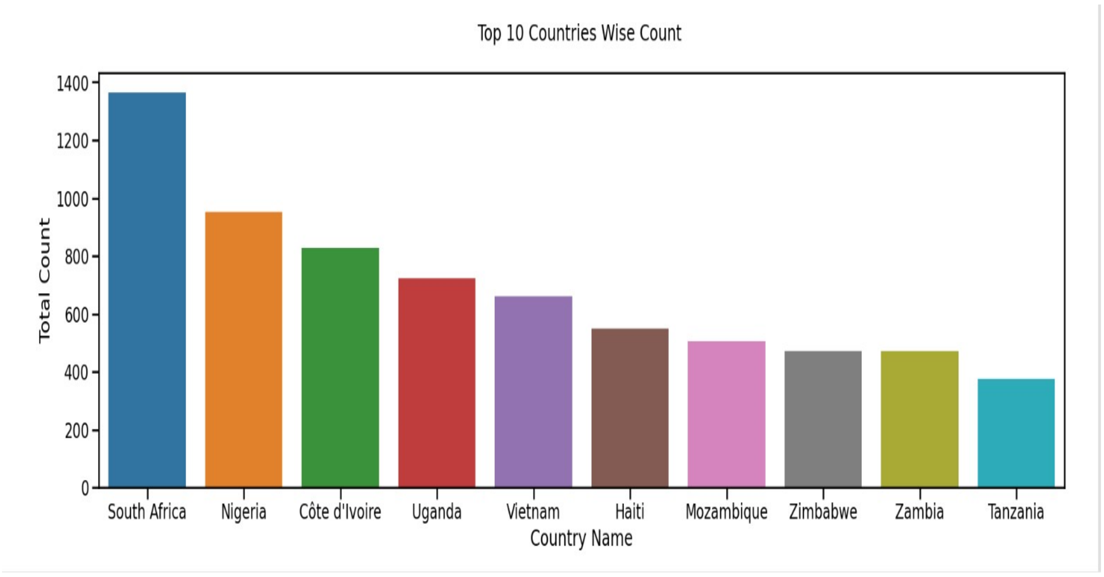


Fig.3.5 Top 10 countries by importing health commodities

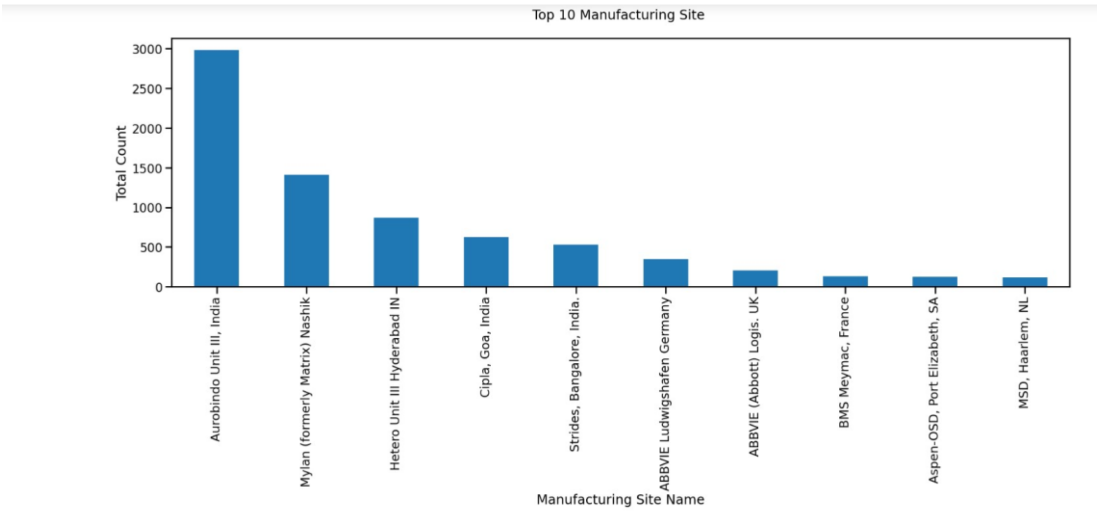


Fig.3.6 Top 10 manufacturing sites for health commodities

The chart reveals the 10 sites that have contributed the most deliveries in the dataset. Aurobindo Unit III is the most active site, with 3000+deliveries. Mylan (formerly Matrix) Nashik and Hetero Unit III Hyderabad follow closely with 2500 and 2000 deliveries, respectively.

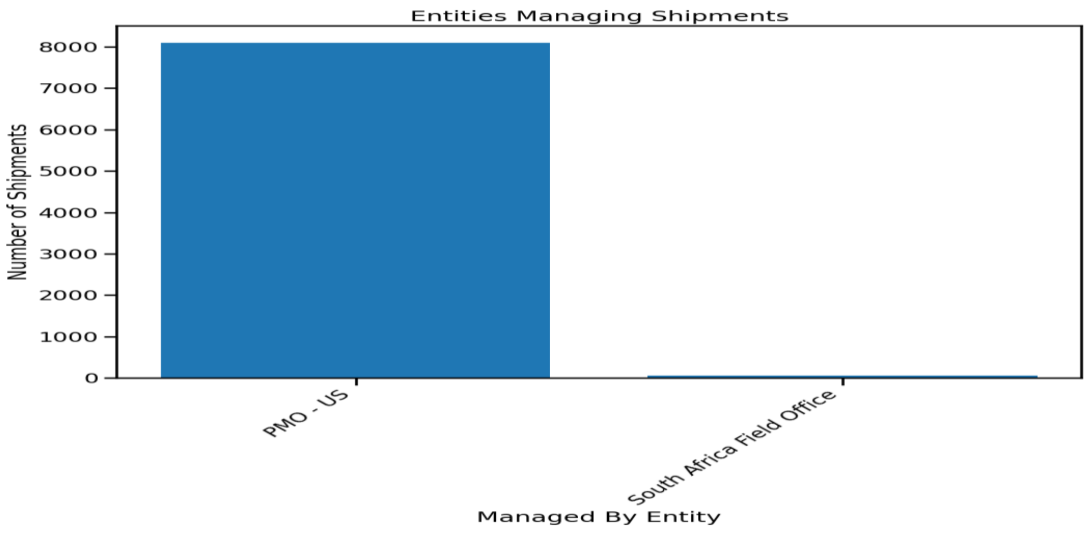


Fig.3.7 Entities Managing shipments
(PMO - US 99.3% and South Africa Field Office - 0.7%)

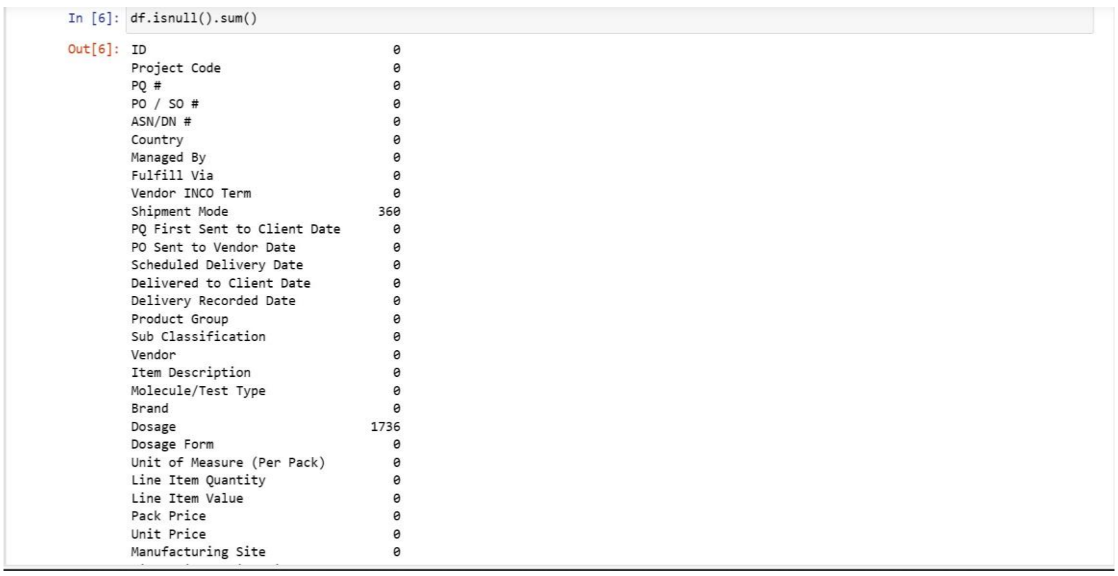
3.5 Data transformation:

It is a critical step in data analysis, involving adjustments to raw data for effective modelling. Tasks include scaling numerical values for uniformity, encoding categorical variables for machine learning compatibility, creating new features to enhance model performance, and applying various modifications to optimize data for accurate and meaningful analysis.

3.5.1. We used the information on the cited website (data.world) to select the suppliers. The.csv file contains the dataset. More than 33 features and more than 10,324 rows make up the raw dataset. We used Microsoft Excel to read out the dataset quickly.

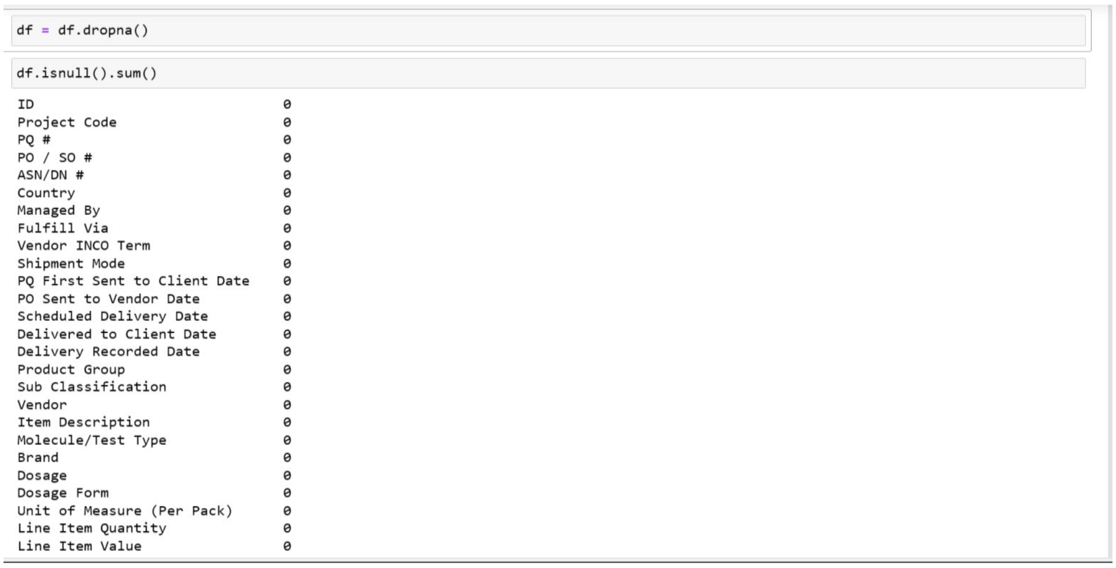
3.5.2. We have to choose the pertinent supplier selection feature before utilising the raw data as input. To do this, we used a variety of Python libraries on the Jupyter notebook software. Pandas is the library that was used to import the dataset as a data frame.

3.5.3. We handle dataset inconsistencies, such as null values and data type errors, before choosing pertinent features.

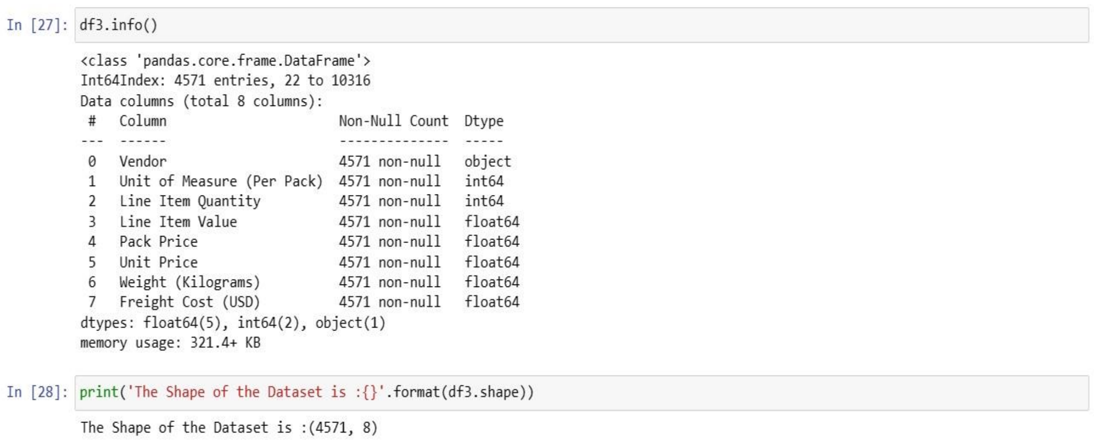


```
In [6]: df.isnull().sum()

Out[6]: ID                               0
        Project Code                     0
        PQ #                             0
        PO / SO #                        0
        ASN/DN #                         0
        Country                          0
        Managed By                       0
        Fulfill Via                      0
        Vendor INCO Term                 0
        Shipment Mode                  360
        PQ First Sent to Client Date     0
        PO Sent to Vendor Date           0
        Scheduled Delivery Date          0
        Delivered to Client Date         0
        Delivery Recorded Date           0
        Product Group                    0
        Sub Classification               0
        Vendor                           0
        Item Description                 0
        Molecule/Test Type               0
        Brand                            0
        Dosage                        1736
        Dosage Form                      0
        Unit of Measure (Per Pack)       0
        Line Item Quantity               0
        Line Item Value                  0
        Pack Price                       0
        Unit Price                       0
        Manufacturing Site               0
```

Fig.3.8 Finding Null Values from the given dataset.



```
df = df.dropna()

df.isnull().sum()

ID                              0
Project Code                    0
PQ #                            0
PO / SO #                       0
ASN/DN #                        0
Country                         0
Managed By                      0
Fulfill Via                     0
Vendor INCO Term                0
Shipment Mode                   0
PQ First Sent to Client Date    0
PO Sent to Vendor Date          0
Scheduled Delivery Date         0
Delivered to Client Date        0
Delivery Recorded Date          0
Product Group                   0
Sub Classification              0
Vendor                          0
Item Description                0
Molecule/Test Type              0
Brand                           0
Dosage                          0
Dosage Form                     0
Unit of Measure (Per Pack)      0
Line Item Quantity              0
Line Item Value                 0
```



```
In [27]: df3.info()

         <class 'pandas.core.frame.DataFrame'>
         Int64Index: 4571 entries, 22 to 10316
         Data columns (total 8 columns):
          #   Column                      Non-Null Count  Dtype
         ---  ------                      --------------  -----
          0   Vendor                      4571 non-null   object
          1   Unit of Measure (Per Pack)  4571 non-null   int64
          2   Line Item Quantity          4571 non-null   int64
          3   Line Item Value             4571 non-null   float64
          4   Pack Price                  4571 non-null   float64
          5   Unit Price                  4571 non-null   float64
          6   Weight (Kilograms)          4571 non-null   float64
          7   Freight Cost (USD)          4571 non-null   float64
         dtypes: float64(5), int64(2), object(1)
         memory usage: 321.4+ KB

In [28]: print('The Shape of the Dataset is :{}'.format(df3.shape))

         The Shape of the Dataset is :(4571, 8)
```

Fig.3.9 Reduction of the dataset.

3.5.4. The above steps cause a reduction of the dataset from 10324 rows to only close to 4571 data rows per class/feature.

3.5.5. Data visualization is a fundamental technique for better understanding the dataset. To create pair plot graphs for each feature with one another and show the relationship between them, we used the Seaborn library.

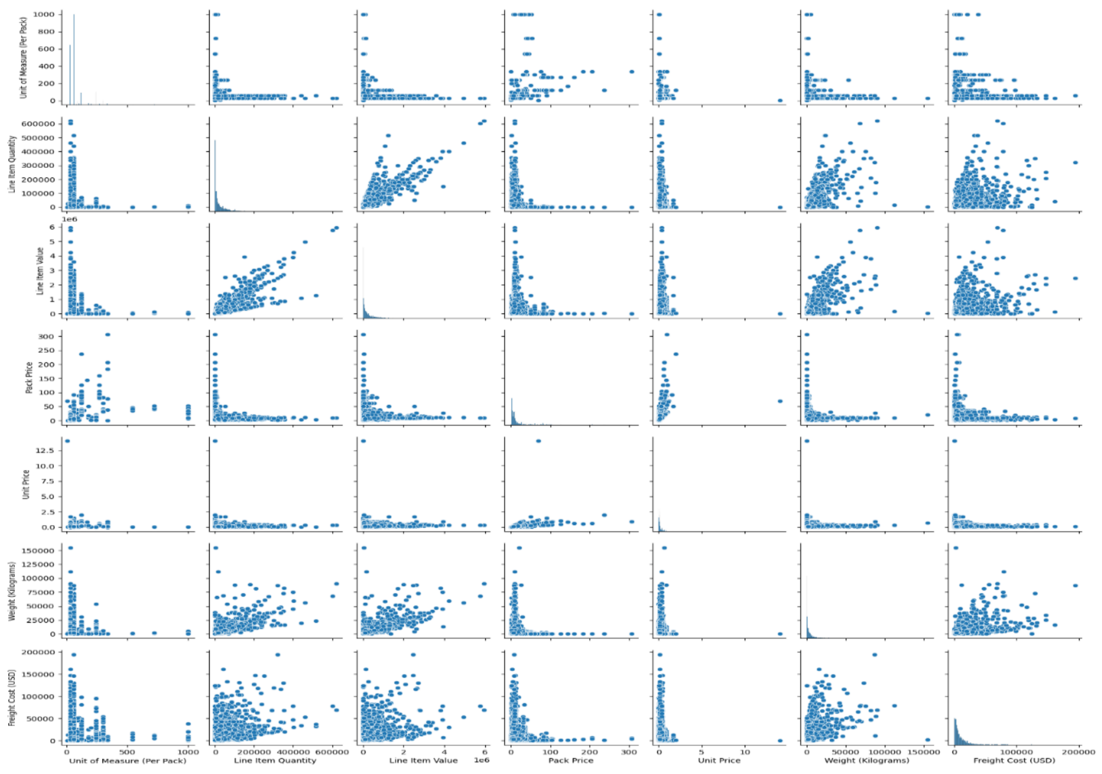
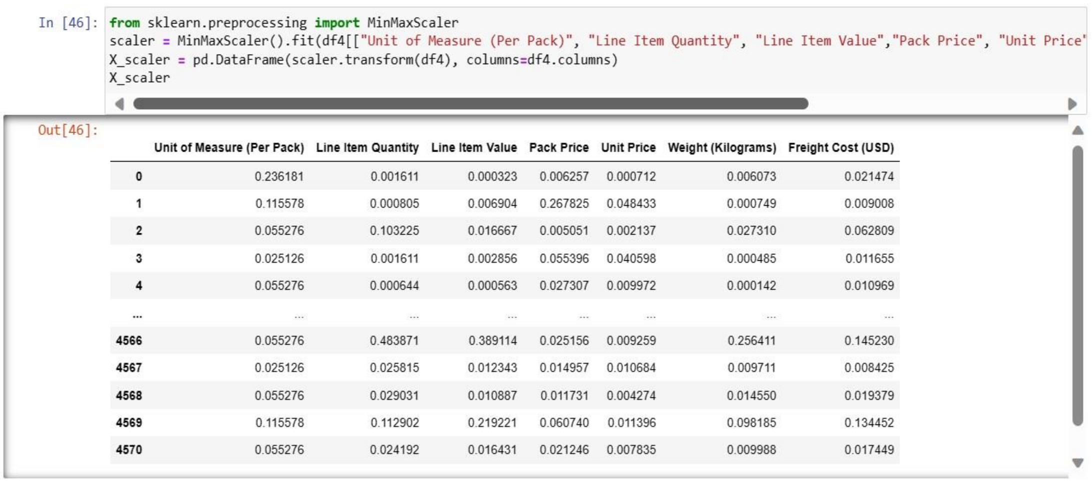


Fig.3.10 Visualizing features data

Strong positive correlation: The scatter plot of weight vs cost clearly shows a diagonal pattern of points sloping upward, indicating a strong positive correlation between the two. This implies that delivery costs typically rise in tandem with a shipment's weight.

3.5.6. Because the dataset contains a variety of features with varying range magnitudes and sets of units to define, we used a python library called sklearn to help with standardisation by importing preprocessing features into the code.



```
In [46]: from sklearn.preprocessing import MinMaxScaler
         scaler = MinMaxScaler().fit(df4[["Unit of Measure (Per Pack)", "Line Item Quantity", "Line Item Value","Pack Price", "Unit Price'
         X_scaler = pd.DataFrame(scaler.transform(df4), columns=df4.columns)
         X_scaler
```

Out[46]:

	Unit of Measure (Per Pack)	Line Item Quantity	Line Item Value	Pack Price	Unit Price	Weight (Kilograms)	Freight Cost (USD)
0	0.236181	0.001611	0.000323	0.006257	0.000712	0.006073	0.021474
1	0.115578	0.000805	0.006904	0.267825	0.048433	0.000749	0.009008
2	0.055276	0.103225	0.016667	0.005051	0.002137	0.027310	0.062809
3	0.025126	0.001611	0.002856	0.055396	0.040598	0.000485	0.011655
4	0.055276	0.000644	0.000563	0.027307	0.009972	0.000142	0.010969
...	...	...	...	...	...	...	...
4566	0.055276	0.483871	0.389114	0.025156	0.009259	0.256411	0.145230
4567	0.025126	0.025815	0.012343	0.014957	0.010684	0.009711	0.008425
4568	0.055276	0.029031	0.010887	0.011731	0.004274	0.014550	0.019379
4569	0.115578	0.112902	0.219221	0.060740	0.011396	0.098185	0.134452
4570	0.055276	0.024192	0.016431	0.021246	0.007835	0.009988	0.017449

Fig.3.11 Scaling of independent feature

A MinMaxScaler is a tool for preprocessing data in machine learning. Scaling features to a given range—typically between 0 and 1—transforms them. This may be advantageous for a number of reasons:

a. Better model performance: The size of features affects the performance of many machine learning algorithms, including neural networks and distance-based algorithms like K-Nearest Neighbours. The MinMaxScaler makes sure that no feature dominates the model because of its larger scale by scaling all features to a similar range. This may result in increased generalizability and accuracy of the model.

b. Quicker training: The model may take longer to converge during training if its features are on different scales. It is possible to achieve faster training times by reducing variance by scaling features to a common range.

c. Easier data visualisation: MinMax scaling can facilitate the visual comparison and interpretation of feature relationships when you wish to visualise your data on the same scale, such as in a scatter plot.

3.5.7. Assuming that the freight cost and quantity are significant factors in supplier selection, we applied the correlation function to the dataset's independent features to ascertain the correlation between them and which feature influences them.

3.5.8. Since we assume that the freight cost and quantity are significant factors in supplier selection, we used the correlation function to the independent features of the dataset to find the correlation between them and which feature effects the quantity and freight cost.

3.6 Model Building and Evaluation:

To evaluate a model, a dataset is typically split into training and testing sets. For classification tasks, models are assessed using metrics like recall, accuracy, precision, and F1 score(Fig 3.12). Feature reduction techniques aim to increase model efficiency by choosing the most relevant features. Application of machine learning for imbalance data set.

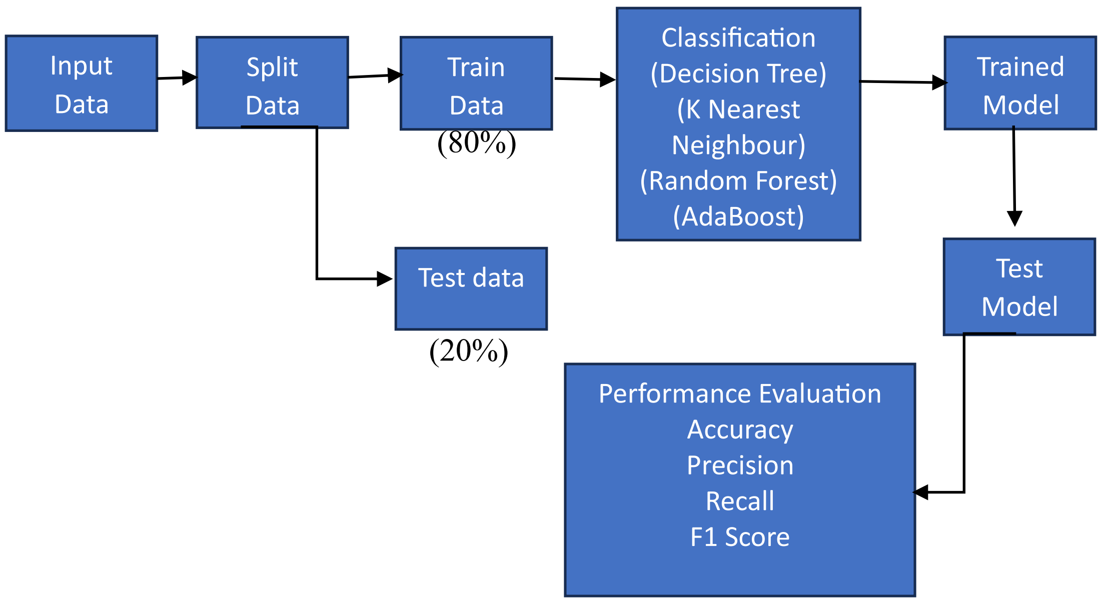


Fig.3.12 Flow Chart of Classifier

3.6.1 Accuracy:

The most widely used assessment metric in the data community is accuracy. However, when the data set is balanced across the sample classes, accuracy is an excellent reporting metric. Furthermore, relying solely on this measure could be deceptive in trying to solve the problem because most classification problems—especially those that arise in the real world—are inherently imbalanced. For this reason, it is advised that we take into account additional metrics when assessing model performance. Mathematically, accuracy can be calculated as folloes: Accuracy = T P + T N / T P + T N + F P + F N

3.6.2 Precision:

The ratio of True Positives observations to all Positive observations is called precision. That is, what percentage of the cases the classifier marked as positive were in fact accurate? It is also known as Positive Predictive Value (PPV) and is a metric for determining how accurate a classifier is. Precision =TP / T P + F P.

3.6.3 Recall/Sensitivity:

One indicator of a classifier's completeness is recall/sensitivity. The ability of the classifier to locate all positive instances is known as recall, and it can be defined as the ratio of postive predictions to positive class values. Recall = T P / T P + F N

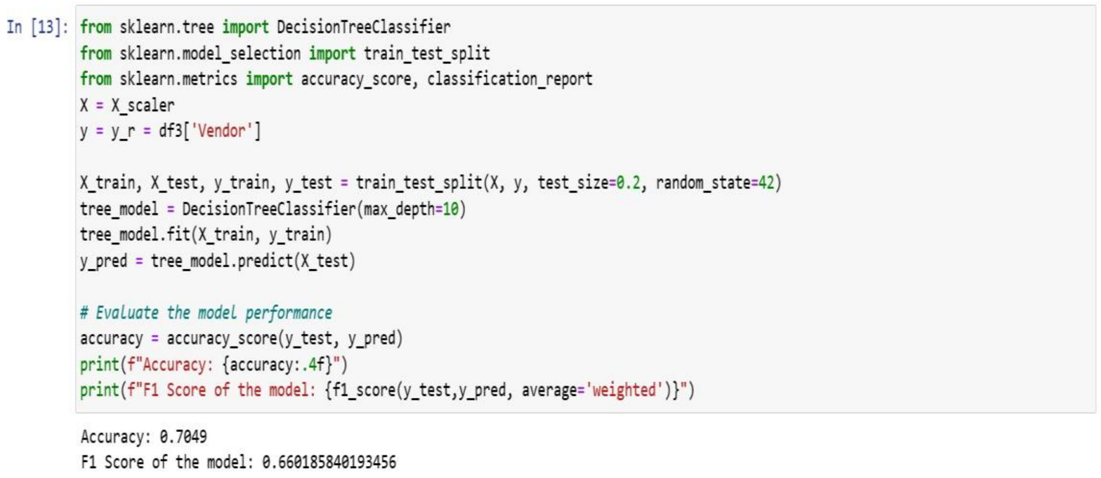
3.6.4 F1-score:

The F1-score is typically regarded as the accuracy of the model that accounts for Precision and Recall. And when working with asymmetric or imbalanced data sets, this is especially helpful and accurate in comparison to accuracy. The F1-Score considers the weighted harmonic mean of the precision (exactness) and recall/sensitivity (completeness) of the classifiers. The F1- score will increase with higher Precision and Recall. Precision and Recall are optimised into a single value by this measure.

F1 − Score = 2 ∗ (Recall ∗ Precision) /Recall + Precision

Techniques like feature importance analysis and Principal Component Analysis (PCA) are used. A thorough evaluation ensures that models perform accurately, generalise well, and effectively use pertinent features for effective predictions in both classification and regression scenarios. These factors are achieved by finding a balance between training and testing sets, selecting the right metrics, and optimising feature selection.

Decision Tree Classifier: Decision Tree algorithms are a kind of structural model that resembles a probability tree and continuously divides data to classify or predict based on the answers to the previous set of questions. To help you make better decisions, the model examines the information and offers answers to the queries.

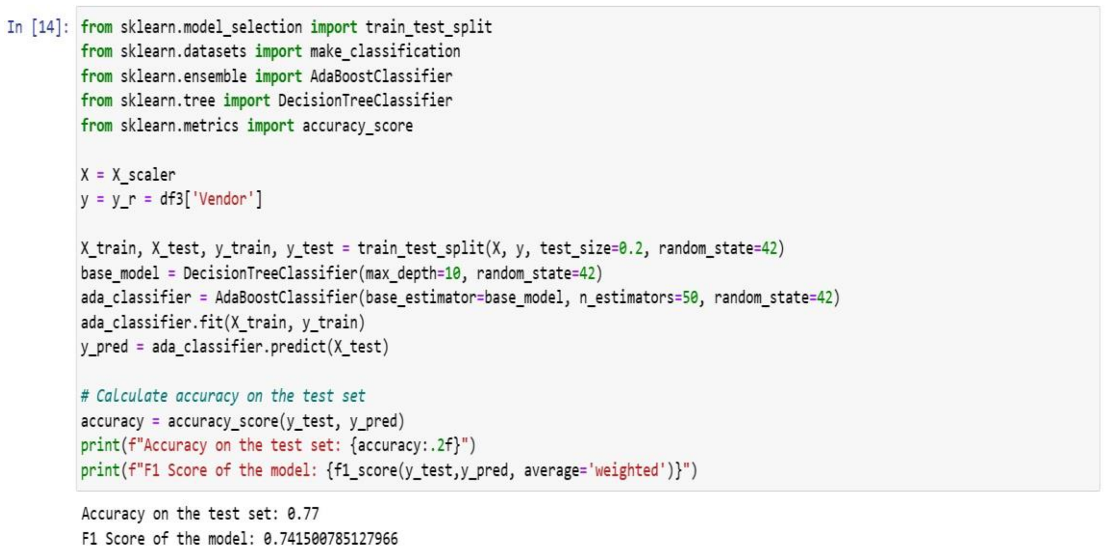


```
In [13]: from sklearn.tree import DecisionTreeClassifier
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score, classification_report
         X = X_scaler
         y = y_r = df3['Vendor']

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
         tree_model = DecisionTreeClassifier(max_depth=10)
         tree_model.fit(X_train, y_train)
         y_pred = tree_model.predict(X_test)

         # Evaluate the model performance
         accuracy = accuracy_score(y_test, y_pred)
         print(f"Accuracy: {accuracy:.4f}")
         print(f"F1 Score of the model: {f1_score(y_test,y_pred, average='weighted')}")

         Accuracy: 0.7049
         F1 Score of the model: 0.660185840193456
```

Fig.3.13 Output of Decision Tree Classifier

Accuracy represent that model correctly classified 70.49% of the data samples it was tested on. In other words, out of every 100 data points the model was evaluteed on, it made correct predictions for 70.49 of them. An F1 Score of 0.660185 indicates that the model balances precision and recall reasonably well. In simpler terms, it’s not simply picking up many suppliers and hoping some are good(imprecise) nor is it missing any good ones (low recall value).

AdaBoost Classifier: Popular ensemble learning technique AdaBoost combines weak learner to produce a robust classifier. Decision trees are frequently used as base estimators in it. Freund and Schapire's (1997) research paper, "A Decision-Theoretic Generalisation of On-Line Learning and an Application to Boosting," claims that AdaBoost performs exceptionally well in terms of enhancing predictive performance through the sequential adjustment of weights of misclassified instances. Its versatility and adaptability across different datasets are improved by incorporating decision trees as base models.

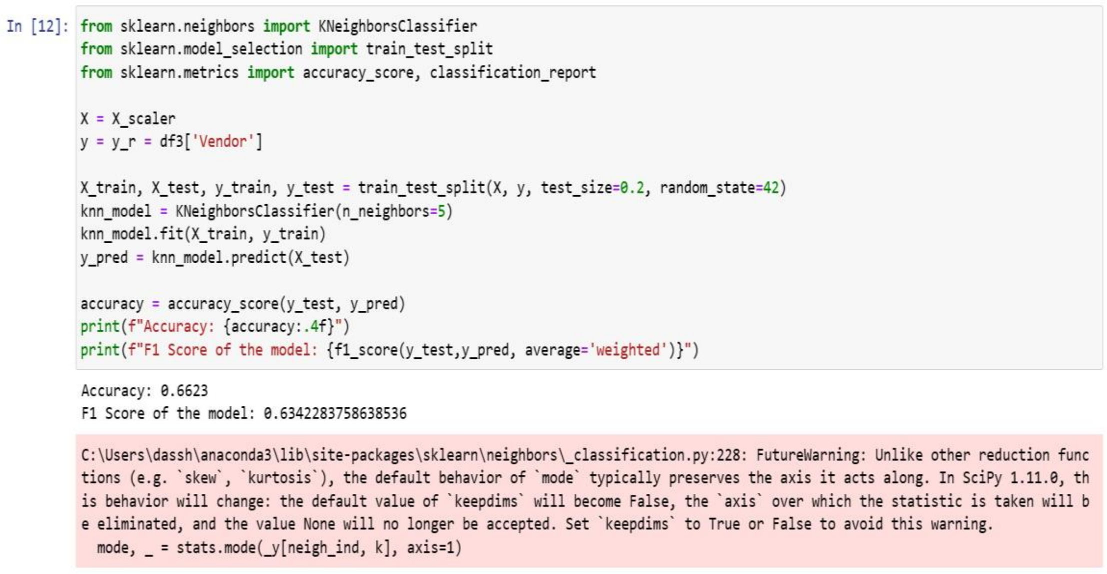


```
In [14]: from sklearn.model_selection import train_test_split
         from sklearn.datasets import make_classification
         from sklearn.ensemble import AdaBoostClassifier
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.metrics import accuracy_score

         X = X_scaler
         y = y_r = df3['Vendor']

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
         base_model = DecisionTreeClassifier(max_depth=10, random_state=42)
         ada_classifier = AdaBoostClassifier(base_estimator=base_model, n_estimators=50, random_state=42)
         ada_classifier.fit(X_train, y_train)
         y_pred = ada_classifier.predict(X_test)

         # Calculate accuracy on the test set
         accuracy = accuracy_score(y_test, y_pred)
         print(f"Accuracy on the test set: {accuracy:.2f}")
         print(f"F1 Score of the model: {f1_score(y_test,y_pred, average='weighted')}")

         Accuracy on the test set: 0.77
         F1 Score of the model: 0.741500785127966
```

Fig.3.14: Output of AdaBoost Classifier

The accuracy of 0.77 suggests the model is doing well at identifying good suppliers most othe time. The F1-score of 0.7415 further indicates a good balance between precision and recall. But, there is still a 23% chance the model might recommend a bad supplier and 25.85% chance it might miss a good one. It is crucial to ensure high precision to avoid wasting time and resources on unqualified suppliers. While missing out on a good supplier isn’t ideal, it is less damaging than integrating a bad one.

KNN Classifier: A statistical technique called K-Nearest Neighbours determines whether or not two data points can be grouped together by calculating how close one data point is to the other. The degree of similarity between the data points is indicated by their proximity to one another.

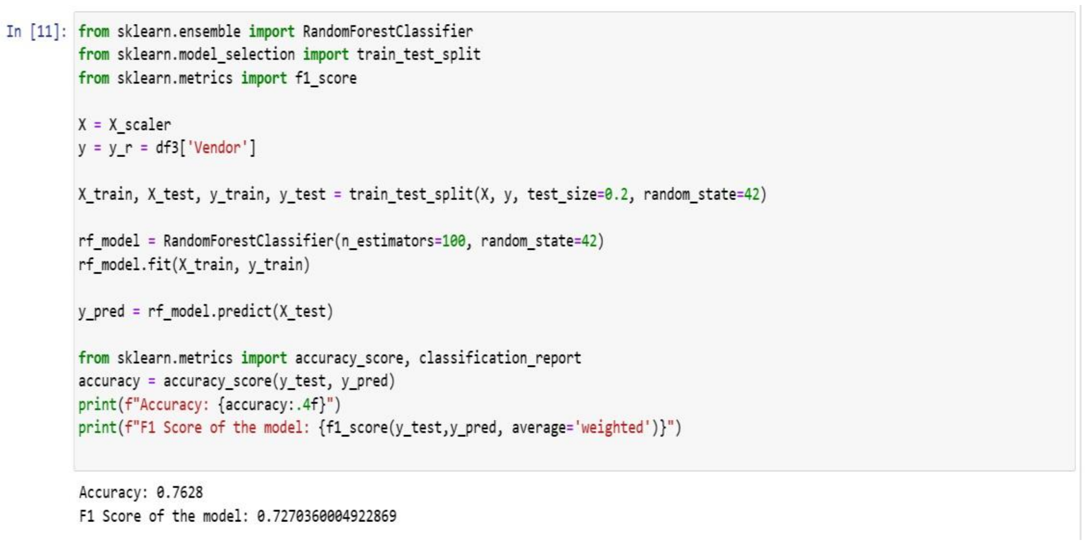


```
In [12]: from sklearn.neighbors import KNeighborsClassifier
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score, classification_report

         X = X_scaler
         y = y_r = df3['Vendor']

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
         knn_model = KNeighborsClassifier(n_neighbors=5)
         knn_model.fit(X_train, y_train)
         y_pred = knn_model.predict(X_test)

         accuracy = accuracy_score(y_test, y_pred)
         print(f"Accuracy: {accuracy:.4f}")
         print(f"F1 Score of the model: {f1_score(y_test,y_pred, average='weighted')}")

         Accuracy: 0.6623
         F1 Score of the model: 0.6342283758638536

         C:\Users\dassh\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction func
         tions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, th
         is behavior will change: the default value of `keepdims` will become False, the `axis` over which the statistic is taken will b
         e eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warning.
           mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

Fig.3.15: Output of KNN Classifier

Out of 100 shipments, the model might incorrectly identify the most cost-effective supplier in 34 cases. This could lead to higher transportation costs. Excess shipping costs can eat into our profits. The model might recommend suppliers that aren't truly the cheapest option. The model might miss out on identifying some of the cost-effective suppliers altogether. This means you might miss opportunities to save money on shipping.

Random Forest Classifier: Random forests are a kind of tree-based algorithm that is comparable to decision trees. Instead of using a single Decision Tree, as in a Decision Tree, Random Forest forms what is essentially a forest of trees by using multiple Decision Trees to make decisions.



```
In [11]: from sklearn.ensemble import RandomForestClassifier
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import f1_score

         X = X_scaler
         y = y_r = df3['Vendor']

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
         rf_model.fit(X_train, y_train)

         y_pred = rf_model.predict(X_test)

         from sklearn.metrics import accuracy_score, classification_report
         accuracy = accuracy_score(y_test, y_pred)
         print(f"Accuracy: {accuracy:.4f}")
         print(f"F1 Score of the model: {f1_score(y_test,y_pred, average='weighted')}")

         Accuracy: 0.7628
         F1 Score of the model: 0.7270360004922869
```

Fig.3.16: Output of Random Forest Classifier

With 76% accuracy, the model can potentially help in achieving significant cost savings on shipping health commodities. F1-score of 0.7270 means the model is doing a moderately good job at identifying the most cost-effective suppliers, but there is room for improvement.

3.7 Imbalanced Dataset

An imbalanced dataset occurs when the distribution of classes within the data is highly skewed. In other words, one or more classes have significantly more data points than the others. This can be a challenge for machine learning models because they tend to Favor the majority class during training, leading to poor performance on the minority class.

Here are some characteristics of an imbalanced dataset:

Skewed class distribution: One class can represent 80-90% of the data, while the remaining classes share the remaining 10-20%.

Challenges for models: Models trained on such data often suffer from:

Overfitting: Models learn the dominant class too well and ignore the nuances of the minority class.

Poor performance metrics: Precision, recall, and other metrics may appear good overall due to the majority class dominance, but can be misleading for the minority class.

Consequences of ignoring the issue: Imbalanced data can lead to biased and inaccurate predictions, particularly for minority groups, which can have ethical and practical implications depending on the application.

Dealing with imbalanced data: There are various techniques to handle imbalanced data, including.

Resampling: Oversampling the minority class or under sampling the majority class to balance the distribution.

Cost-sensitive learning: Assigning higher weight to the misclassification of minority class examples during training.

Using specific algorithms: Choosing algorithms robust to class imbalance, like Random Forests or Support Vector Machines with appropriate modifications

The difficulty of imbalances classification is compounded by properties such as data size, label noise, and data distribution. (fig.3.17)

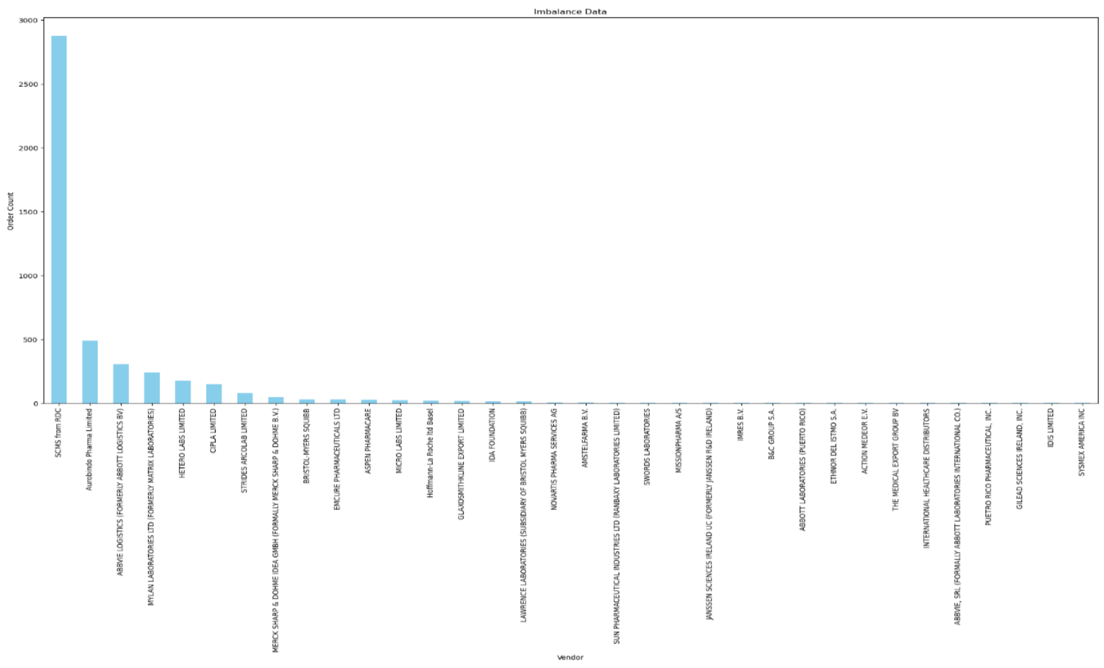


Fig.3.17: Representing Imbalanced Data Set

3.8 Techniques to Balance Unbalance Dataset:

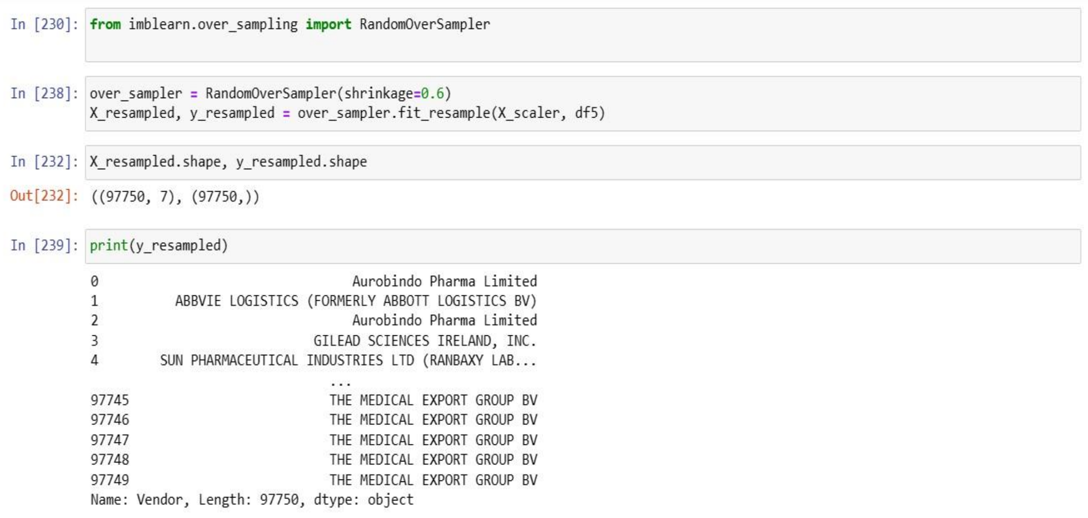
Use the following techniques to convert an unbalanced dataset into a balanced dataset:

3.8.1. Over-sampling: Random oversampling is the process of randomly copying minority class examples and adding them to the training set examples. This involves selecting and adding instances from the minority class to a new, more balanced training set multiple times. These classes are selected from the original dataset and added to the newly created training dataset in such a way that it is ‘put back’, or returned, to the original dataset, enabling them to be selected again.

3.8.2. Under sampling: Random under sampling is the process of arbitrarily removing instances belonging to the majority class from the training data set.

As a result, there are fewer examples in the modified version of the training dataset’s majority class. Until an optimal classifier is achieved—that is, a comparable number of examples for every class—this process can be repeated. This method might be better suited for datasets that exhibit class imbalance, but a feasible model like that might be possible if the minority class has a sufficient number of examples.

As per the aforementioned comprehension, the dataset under consideration exhibits imbalanced data. The Oversampling method is necessary to eliminate inaccuracies and enhance machine learning training due to the insufficient amount of data.



```
In [230]: from imblearn.over_sampling import RandomOverSampler

In [238]: over_sampler = RandomOverSampler(shrinkage=0.6)
          X_resampled, y_resampled = over_sampler.fit_resample(X_scaler, df5)

In [232]: X_resampled.shape, y_resampled.shape
Out[232]: ((97750, 7), (97750,))

In [239]: print(y_resampled)
          0                               Aurobindo Pharma Limited
          1            ABBVIE LOGISTICS (FORMERLY ABBOTT LOGISTICS BV)
          2                               Aurobindo Pharma Limited
          3                           GILEAD SCIENCES IRELAND, INC.
          4          SUN PHARMACEUTICAL INDUSTRIES LTD (RANBAXY LAB...
                                                 ...
          97745                          THE MEDICAL EXPORT GROUP BV
          97746                          THE MEDICAL EXPORT GROUP BV
          97747                          THE MEDICAL EXPORT GROUP BV
          97748                          THE MEDICAL EXPORT GROUP BV
          97749                          THE MEDICAL EXPORT GROUP BV
          Name: Vendor, Length: 97750, dtype: object
```

Fig.3.18 Over Sampling

Although RandomOverSampler does not require sophisticated mathematics, the following are some of its main ideas:

1. Sampling and Probability

Finding Class Proportions: The initial step determines how much of each class there was in the original dataset.

Sampling at Random with Replacement: Until the target proportion is attained, minority class samples are sampled at random with replacement. This implies that samples may be chosen more than once.

Shrinkage Parameter: To produce a more balanced distribution, it may choose to exclude a certain percentage of majority class samples.

2. Algorithm

Count Class Occurrences: Determine how many times each class appears in the target vector (y).

Determine the Minority Class: Find the class in which there are the fewest instances.

Determine the desired sample count: Multiply the total number of samples by the minority class proportion that is desired.

Replicate Minority Samples at Random: Until the desired number is reached, pick minority class samples at random time after time.

Apply Shrinkage (Optional): Take a random portion of the majority class samples and remove it if shrinkage is set.

Combine Resampled Data: To create the final resampled dataset, merge the minority class samples that have been resampled with the remaining majority class samples, possibly shrinking the majority class samples as well.

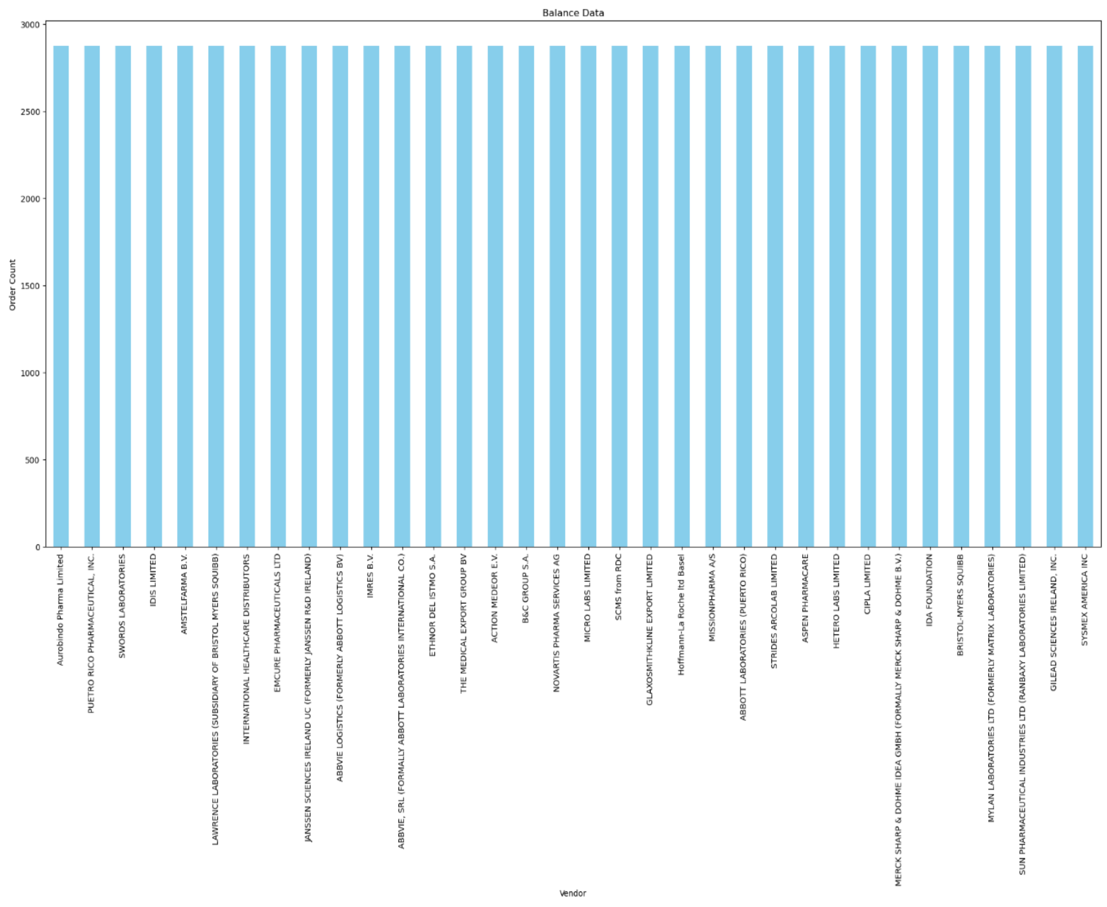
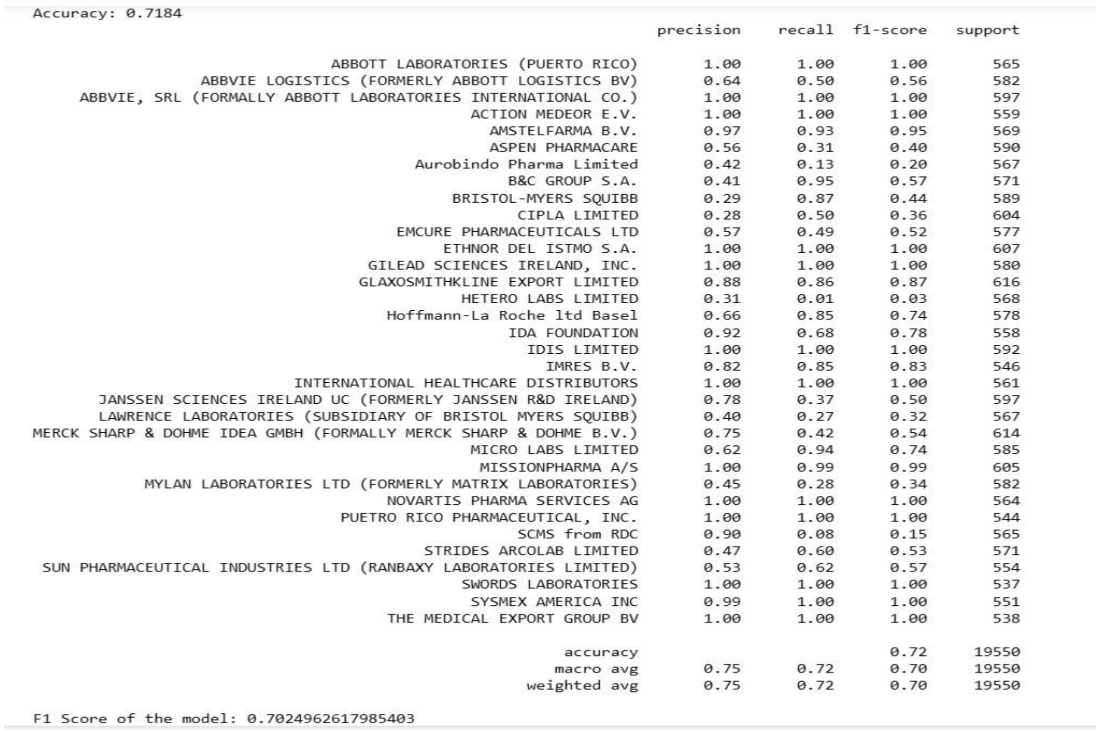


Fig.3.19 Representing a balanced Data Set

Decision Tree Classifier (Balanced Data Set): The training score is consistently higher than the validation score, as seen in Fig. 3.21. This is due to the decision tree classifier's flawless memorization of the training set. This does not imply, however, that the classifier will be good at generalising to fresh data. A more accurate indicator of the classifier’s performance on fresh data is the validation score. As the max depth of the decision tree increases, the train accuracy generally increases as well. However, the test accuracy may also increase, then decrease, or stay about the same. This is because a decision tree with a higher max depth may be more likely to overfit the data.

In the specific graph it appears that the test accuracy peaks at a max depth of 4 and starts to decrease after that. In conclusion, the plot shows that the decision tree classifier is overfitting the training data. This means that the classifier is not likely to perform well on new data. We can try to reduce overfitting by using a smaller number of trees or by using a different machine learning algorithm.



```
Accuracy: 0.7184
                                                                         precision    recall  f1-score   support

                                  ABBOTT LABORATORIES (PUERTO RICO)       1.00      1.00      1.00       565
                       ABBVIE LOGISTICS (FORMERLY ABBOTT LOGISTICS BV)       0.64      0.50      0.56       582
        ABBVIE, SRL (FORMALLY ABBOTT LABORATORIES INTERNATIONAL CO.)       1.00      1.00      1.00       597
                                              ACTION MEDEOR E.V.       1.00      1.00      1.00       559
                                               AMSTELFARMA B.V.       0.97      0.93      0.95       569
                                               ASPEN PHARMACARE       0.56      0.31      0.40       590
                                        Aurobindo Pharma Limited       0.42      0.13      0.20       567
                                                  B&C GROUP S.A.       0.41      0.95      0.57       571
                                           BRISTOL-MYERS SQUIBB       0.29      0.87      0.44       589
                                                  CIPLA LIMITED       0.28      0.50      0.36       604
                                      EMCURE PHARMACEUTICALS LTD       0.57      0.49      0.52       577
                                            ETHNOR DEL ISTMO S.A.       1.00      1.00      1.00       607
                                   GILEAD SCIENCES IRELAND, INC.       1.00      1.00      1.00       580
                                  GLAXOSMITHKLINE EXPORT LIMITED       0.88      0.86      0.87       616
                                              HETERO LABS LIMITED       0.31      0.01      0.03       568
                                     Hoffmann-La Roche ltd Basel       0.66      0.85      0.74       578
                                                  IDA FOUNDATION       0.92      0.68      0.78       558
                                                   IDIS LIMITED       1.00      1.00      1.00       592
                                                     IMRES B.V.       0.82      0.85      0.83       546
                             INTERNATIONAL HEALTHCARE DISTRIBUTORS       1.00      1.00      1.00       561
            JANSSEN SCIENCES IRELAND UC (FORMERLY JANSSEN R&D IRELAND)       0.78      0.37      0.50       597
           LAWRENCE LABORATORIES (SUBSIDIARY OF BRISTOL MYERS SQUIBB)       0.40      0.27      0.32       567
      MERCK SHARP & DOHME IDEA GMBH (FORMALLY MERCK SHARP & DOHME B.V.)       0.75      0.42      0.54       614
                                              MICRO LABS LIMITED       0.62      0.94      0.74       585
                                                MISSIONPHARMA A/S       1.00      0.99      0.99       605
                MYLAN LABORATORIES LTD (FORMERLY MATRIX LABORATORIES)       0.45      0.28      0.34       582
                                       NOVARTIS PHARMA SERVICES AG       1.00      1.00      1.00       564
                                    PUETRO RICO PHARMACEUTICAL, INC.       1.00      1.00      1.00       544
                                                   SCMS from RDC       0.90      0.08      0.15       565
                                            STRIDES ARCOLAB LIMITED       0.47      0.60      0.53       571
     SUN PHARMACEUTICAL INDUSTRIES LTD (RANBAXY LABORATORIES LIMITED)       0.53      0.62      0.57       554
                                              SWORDS LABORATORIES       1.00      1.00      1.00       537
                                              SYSMEX AMERICA INC       0.99      1.00      1.00       551
                                        THE MEDICAL EXPORT GROUP BV       1.00      1.00      1.00       538

                                                        accuracy                           0.72     19550
                                                       macro avg       0.75      0.72      0.70     19550
                                                    weighted avg       0.75      0.72      0.70     19550

F1 Score of the model: 0.7024962617985403
```

Fig.3.20 Classification report of Decision Tree Classifier

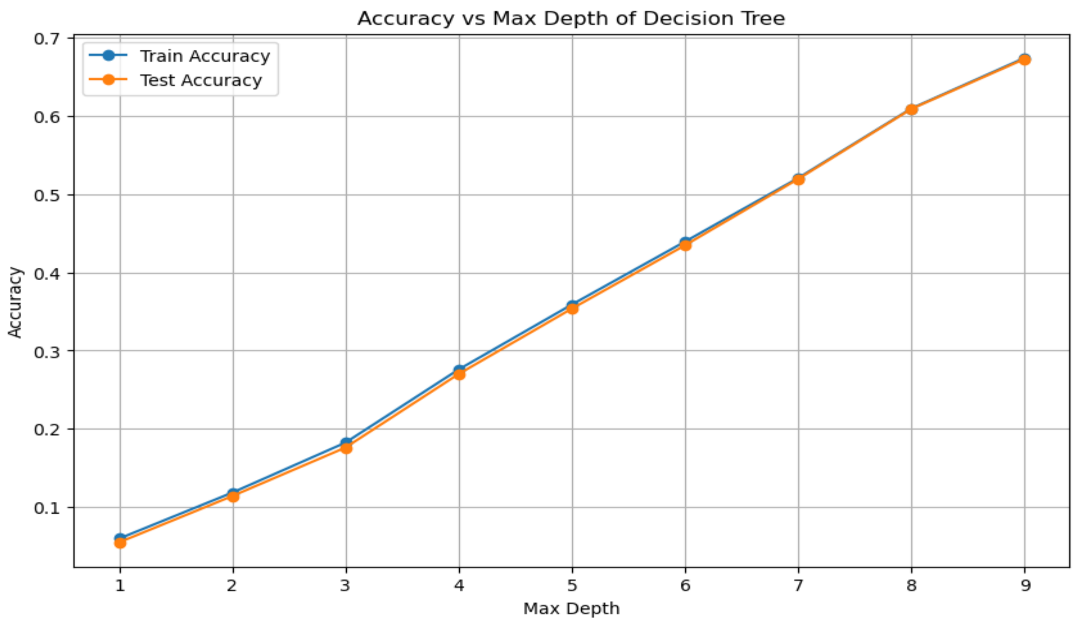
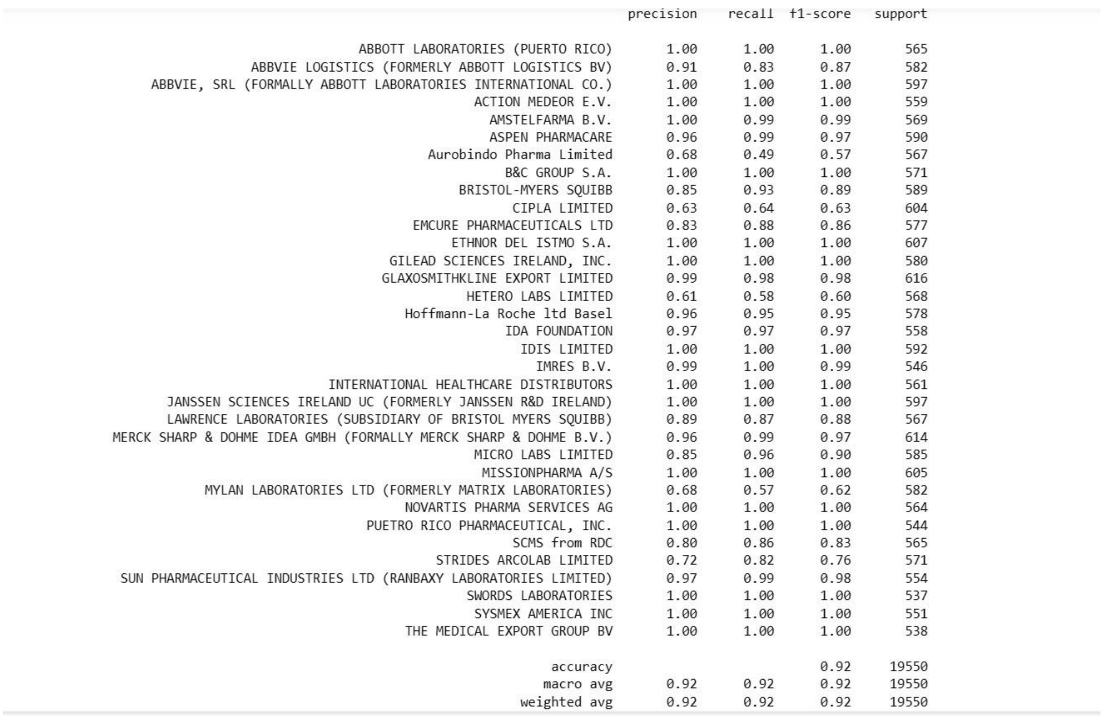


Fig.3.21 Decision Tree Classifier Train Score v/s Validation Score

Random Forest Classifier (Balanced Data Set): As the number of tree in the forest increases, both the training score and the validation score increase. However, the validation score starts to level off after a certain number of trees, while the training score continues to increase. This is because the training score is an optimistic estimate of the classifier's performance, while the validation score is a more realistic estimate of how well the classifier will perform on new data.

The gap between the training score and the validation score is known as the generalization gap. It is important to minimize the generalization gap, as it indicates how well the classifier will generalize to unseen data.

In conclusion, the graph shows that a random forest classifier can achieve good accuracy on a given dataset. However, it is important to be aware of the generalization gap and to take steps to minimize it.



```
                                                                   precision    recall  f1-score   support

                                ABBOTT LABORATORIES (PUERTO RICO)       1.00      1.00      1.00       565
                  ABBVIE LOGISTICS (FORMERLY ABBOTT LOGISTICS BV)       0.91      0.83      0.87       582
     ABBVIE, SRL (FORMALLY ABBOTT LABORATORIES INTERNATIONAL CO.)       1.00      1.00      1.00       597
                                               ACTION MEDEOR E.V.       1.00      1.00      1.00       559
                                                 AMSTELFARMA B.V.       1.00      0.99      0.99       569
                                                 ASPEN PHARMACARE       0.96      0.99      0.97       590
                                         Aurobindo Pharma Limited       0.68      0.49      0.57       567
                                                   B&C GROUP S.A.       1.00      1.00      1.00       571
                                             BRISTOL-MYERS SQUIBB       0.85      0.93      0.89       589
                                                    CIPLA LIMITED       0.63      0.64      0.63       604
                                       EMCURE PHARMACEUTICALS LTD       0.83      0.88      0.86       577
                                            ETHNOR DEL ISTMO S.A.       1.00      1.00      1.00       607
                                    GILEAD SCIENCES IRELAND, INC.       1.00      1.00      1.00       580
                                   GLAXOSMITHKLINE EXPORT LIMITED       0.99      0.98      0.98       616
                                              HETERO LABS LIMITED       0.61      0.58      0.60       568
                                      Hoffmann-La Roche ltd Basel       0.96      0.95      0.95       578
                                                   IDA FOUNDATION       0.97      0.97      0.97       558
                                                     IDIS LIMITED       1.00      1.00      1.00       592
                                                       IMRES B.V.       0.99      1.00      0.99       546
                            INTERNATIONAL HEALTHCARE DISTRIBUTORS       1.00      1.00      1.00       561
      JANSSEN SCIENCES IRELAND UC (FORMERLY JANSSEN R&D IRELAND)       1.00      1.00      1.00       597
      LAWRENCE LABORATORIES (SUBSIDIARY OF BRISTOL MYERS SQUIBB)       0.89      0.87      0.88       567
MERCK SHARP & DOHME IDEA GMBH (FORMALLY MERCK SHARP & DOHME B.V.)       0.96      0.99      0.97       614
                                               MICRO LABS LIMITED       0.85      0.96      0.90       585
                                                MISSIONPHARMA A/S       1.00      1.00      1.00       605
           MYLAN LABORATORIES LTD (FORMERLY MATRIX LABORATORIES)       0.68      0.57      0.62       582
                                      NOVARTIS PHARMA SERVICES AG       1.00      1.00      1.00       564
                                 PUETRO RICO PHARMACEUTICAL, INC.       1.00      1.00      1.00       544
                                                    SCMS from RDC       0.80      0.86      0.83       565
                                          STRIDES ARCOLAB LIMITED       0.72      0.82      0.76       571
 SUN PHARMACEUTICAL INDUSTRIES LTD (RANBAXY LABORATORIES LIMITED)       0.97      0.99      0.98       554
                                              SWORDS LABORATORIES       1.00      1.00      1.00       537
                                               SYSMEX AMERICA INC       1.00      1.00      1.00       551
                                      THE MEDICAL EXPORT GROUP BV       1.00      1.00      1.00       538

                                                         accuracy                           0.92     19550
                                                        macro avg       0.92      0.92      0.92     19550
                                                     weighted avg       0.92      0.92      0.92     19550
```

Fig.3.22 Classification report of Random Forest Classifier

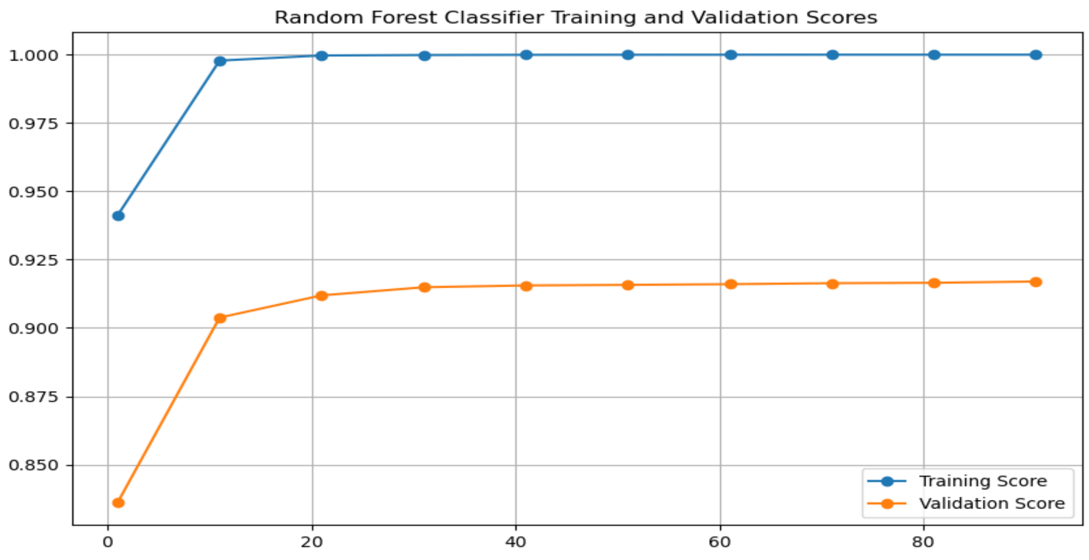
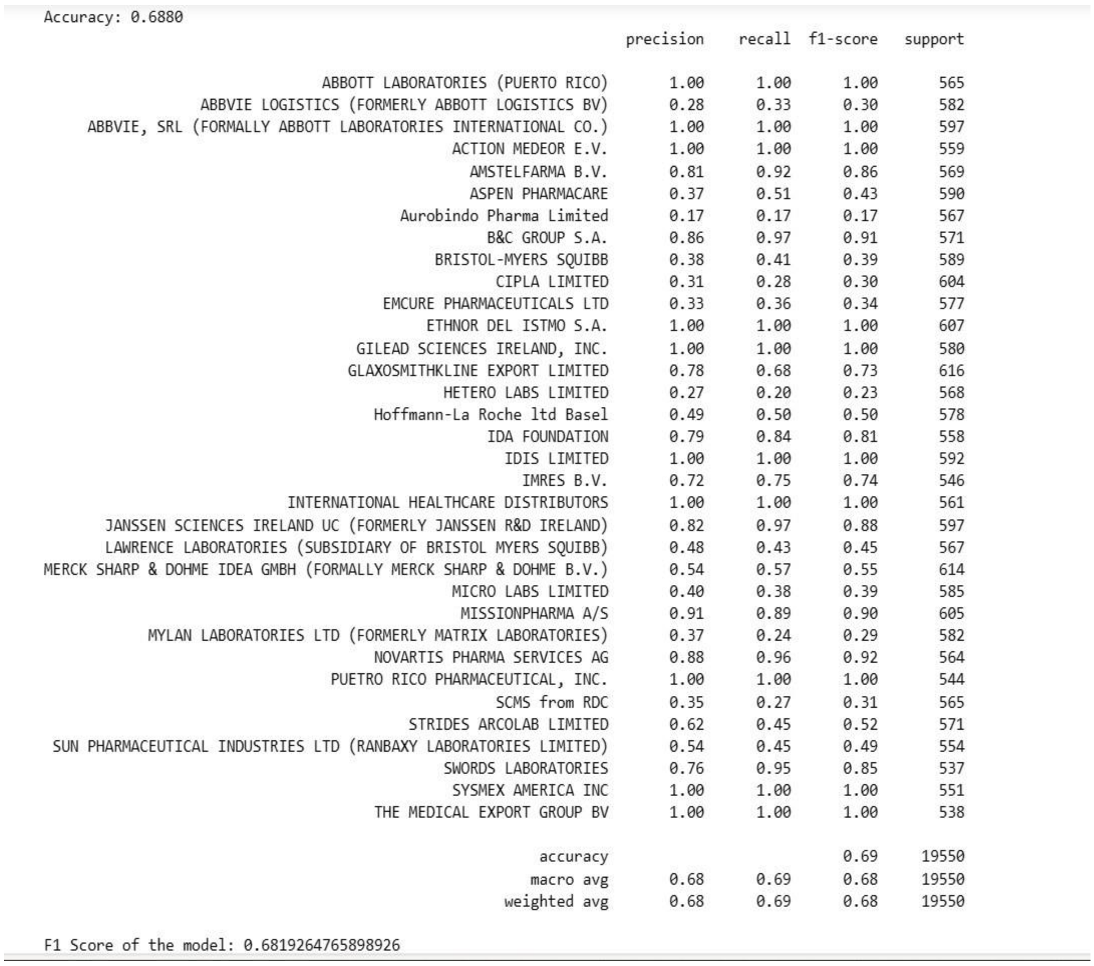


Fig.3.23 Random Forest Classifier Decision Tree Classifier Train Score v/s Validation Score

KNN Classifier (Balanced Data Set): As the model is trained, both the training and validation scores rise in Figure 3.24, which is encouraging. Nonetheless, the validation score consistently falls short of the training score. This is so that the model will always function well on the training data since the training data is used to fit the model. However, because the validation data isn't used to train the model, it offers a more impartial assessment of the model's effectiveness.

The model appears t be generalising well to new data, based on the fact that the validation score is growing even though it is lower than the training score. To put it in another way, the model is picking up patterns from the training set that apply to previously undiscovered date.



```
Accuracy: 0.6880
                                                                    precision    recall  f1-score   support

                     ABBOTT LABORATORIES (PUERTO RICO)                   1.00      1.00      1.00       565
              ABBVIE LOGISTICS (FORMERLY ABBOTT LOGISTICS BV)            0.28      0.33      0.30       582
ABBVIE, SRL (FORMALLY ABBOTT LABORATORIES INTERNATIONAL CO.)             1.00      1.00      1.00       597
                                      ACTION MEDEOR E.V.                 1.00      1.00      1.00       559
                                        AMSTELFARMA B.V.                 0.81      0.92      0.86       569
                                        ASPEN PHARMACARE                 0.37      0.51      0.43       590
                                 Aurobindo Pharma Limited                0.17      0.17      0.17       567
                                          B&C GROUP S.A.                 0.86      0.97      0.91       571
                                     BRISTOL-MYERS SQUIBB                0.38      0.41      0.39       589
                                           CIPLA LIMITED                 0.31      0.28      0.30       604
                               EMCURE PHARMACEUTICALS LTD                0.33      0.36      0.34       577
                                     ETHNOR DEL ISTMO S.A.               1.00      1.00      1.00       607
                               GILEAD SCIENCES IRELAND, INC.             1.00      1.00      1.00       580
                            GLAXOSMITHKLINE EXPORT LIMITED               0.78      0.68      0.73       616
                                      HETERO LABS LIMITED                0.27      0.20      0.23       568
                              Hoffmann-La Roche ltd Basel                0.49      0.50      0.50       578
                                          IDA FOUNDATION                 0.79      0.84      0.81       558
                                            IDIS LIMITED                 1.00      1.00      1.00       592
                                              IMRES B.V.                 0.72      0.75      0.74       546
                      INTERNATIONAL HEALTHCARE DISTRIBUTORS              1.00      1.00      1.00       561
          JANSSEN SCIENCES IRELAND UC (FORMERLY JANSSEN R&D IRELAND)     0.82      0.97      0.88       597
          LAWRENCE LABORATORIES (SUBSIDIARY OF BRISTOL MYERS SQUIBB)     0.48      0.43      0.45       567
MERCK SHARP & DOHME IDEA GMBH (FORMALLY MERCK SHARP & DOHME B.V.)        0.54      0.57      0.55       614
                                       MICRO LABS LIMITED                0.40      0.38      0.39       585
                                        MISSIONPHARMA A/S                0.91      0.89      0.90       605
             MYLAN LABORATORIES LTD (FORMERLY MATRIX LABORATORIES)       0.37      0.24      0.29       582
                               NOVARTIS PHARMA SERVICES AG               0.88      0.96      0.92       564
                          PUETRO RICO PHARMACEUTICAL, INC.               1.00      1.00      1.00       544
                                            SCMS from RDC                0.35      0.27      0.31       565
                                   STRIDES ARCOLAB LIMITED               0.62      0.45      0.52       571
 SUN PHARMACEUTICAL INDUSTRIES LTD (RANBAXY LABORATORIES LIMITED)        0.54      0.45      0.49       554
                                      SWORDS LABORATORIES                0.76      0.95      0.85       537
                                       SYSMEX AMERICA INC                1.00      1.00      1.00       551
                                THE MEDICAL EXPORT GROUP BV              1.00      1.00      1.00       538

                                                 accuracy                                  0.69     19550
                                                macro avg                0.68      0.69      0.68     19550
                                             weighted avg                0.68      0.69      0.68     19550

F1 Score of the model: 0.6819264765898926
```

Fig 3.24 Classification report of KNN Classifier

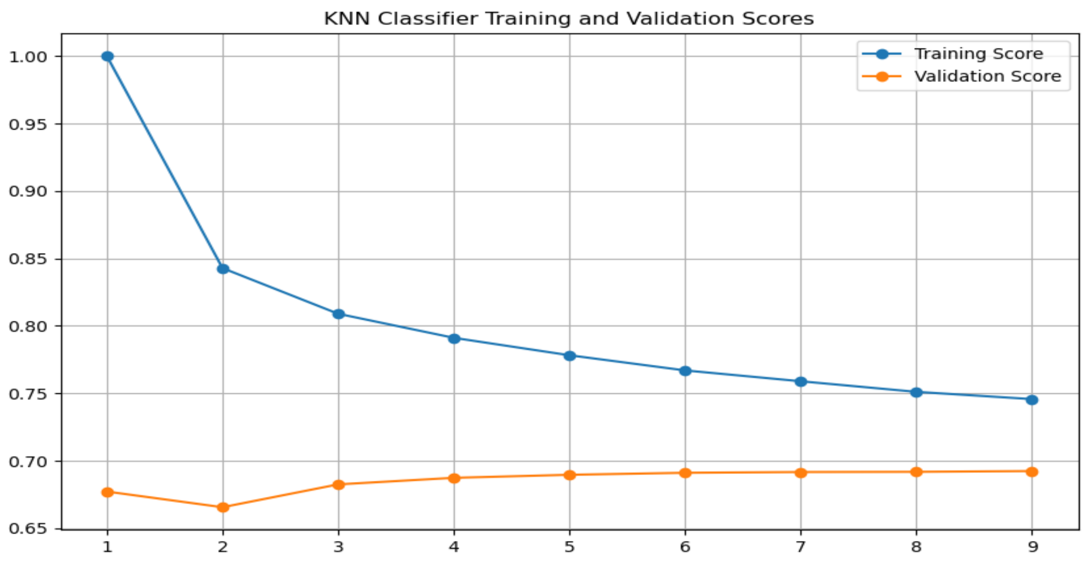
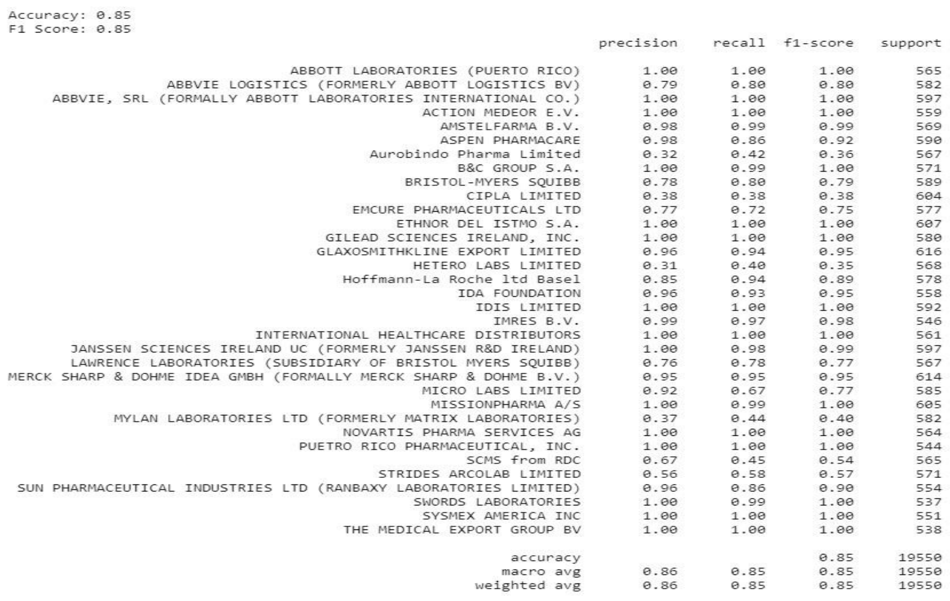


Fig.3.25 KNN Classifier Train Score v/s Validation Score

AdaBoost Classifier (Balanced Data Set): Fig 3.22 shows both the training score and validation score rise with the number of base learners. But while the training score keeps rising, the validation score begins to plateau after a certain amount of base learners. The reason for this is that the model is beginning to overfit the training set, which indicates that it is becoming too adept at identifying certain patterns in the training set and is struggling to make good generalisations to new data.



```
Accuracy: 0.85
F1 Score: 0.85
                                                                   precision    recall  f1-score   support

                                ABBOTT LABORATORIES (PUERTO RICO)       1.00      1.00      1.00       565
                  ABBVIE LOGISTICS (FORMERLY ABBOTT LOGISTICS BV)       0.79      0.80      0.80       582
     ABBVIE, SRL (FORMALLY ABBOTT LABORATORIES INTERNATIONAL CO.)       1.00      1.00      1.00       597
                                               ACTION MEDEOR E.V.       1.00      1.00      1.00       559
                                                 AMSTELFARMA B.V.       0.98      0.99      0.99       569
                                                 ASPEN PHARMACARE       0.98      0.86      0.92       590
                                         Aurobindo Pharma Limited       0.32      0.42      0.36       567
                                                   B&C GROUP S.A.       1.00      0.99      1.00       571
                                             BRISTOL-MYERS SQUIBB       0.78      0.80      0.79       589
                                                    CIPLA LIMITED       0.38      0.38      0.38       604
                                       EMCURE PHARMACEUTICALS LTD       0.77      0.72      0.75       577
                                            ETHNOR DEL ISTMO S.A.       1.00      1.00      1.00       607
                                    GILEAD SCIENCES IRELAND, INC.       1.00      1.00      1.00       580
                                   GLAXOSMITHKLINE EXPORT LIMITED       0.96      0.94      0.95       616
                                              HETERO LABS LIMITED       0.31      0.40      0.35       568
                                      Hoffmann-La Roche ltd Basel       0.85      0.94      0.89       578
                                                   IDA FOUNDATION       0.96      0.93      0.95       558
                                                     IDIS LIMITED       1.00      1.00      1.00       592
                                                       IMRES B.V.       0.99      0.97      0.98       546
                            INTERNATIONAL HEALTHCARE DISTRIBUTORS       1.00      1.00      1.00       561
       JANSSEN SCIENCES IRELAND UC (FORMERLY JANSSEN R&D IRELAND)       1.00      0.98      0.99       597
       LAWRENCE LABORATORIES (SUBSIDIARY OF BRISTOL MYERS SQUIBB)       0.76      0.78      0.77       567
MERCK SHARP & DOHME IDEA GMBH (FORMALLY MERCK SHARP & DOHME B.V.)       0.95      0.95      0.95       614
                                               MICRO LABS LIMITED       0.92      0.67      0.77       585
                                                MISSIONPHARMA A/S       1.00      0.99      1.00       605
           MYLAN LABORATORIES LTD (FORMERLY MATRIX LABORATORIES)       0.37      0.44      0.40       582
                                      NOVARTIS PHARMA SERVICES AG       1.00      1.00      1.00       564
                                 PUETRO RICO PHARMACEUTICAL, INC.       1.00      1.00      1.00       544
                                                    SCMS from RDC       0.67      0.45      0.54       565
                                          STRIDES ARCOLAB LIMITED       0.56      0.58      0.57       571
 SUN PHARMACEUTICAL INDUSTRIES LTD (RANBAXY LABORATORIES LIMITED)       0.96      0.86      0.90       554
                                              SWORDS LABORATORIES       1.00      0.99      1.00       537
                                               SYSMEX AMERICA INC       1.00      1.00      1.00       551
                                      THE MEDICAL EXPORT GROUP BV       1.00      1.00      1.00       538

                                                         accuracy                           0.85     19550
                                                        macro avg       0.86      0.85      0.85     19550
                                                     weighted avg       0.86      0.85      0.85     19550
```

Fig.3.26 Classification report of AdaBoost Classifier

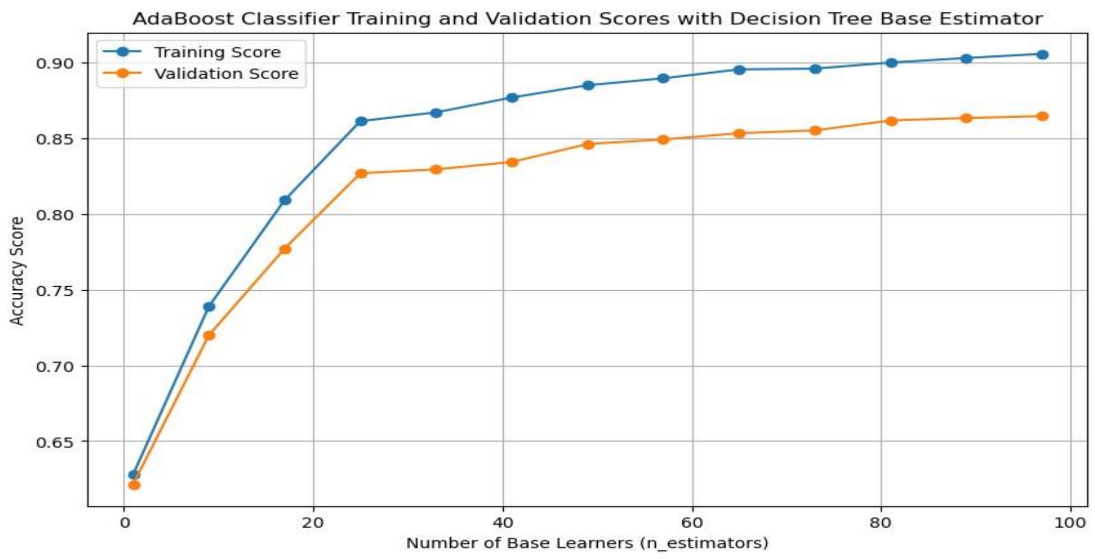


Fig.3.27 AdaBoost Classifier Train Score v/s Validation Score

Table 4.1 represents the model F-1 Score as well as model accuracy. The Random Forest model has the best F-1 Score of 91% and a better accuracy of 92% followed by the AdaBoost model with an F-1 score of 84% and an accuracy of 85%. The least good model is the KNN model with an f-1 score of 68% and accuracy of 69 % followed by the Decision Tree model with an accuracy of 72% and an F-1 score of 70%.

Machine Learning Model	F1- Score	Accuracy
Random Forest	0.90	91%
AdaBoost	0.84	85%
Decision Tree	0.70	72%
KNN	0.68	69%

Table 3.2 F1-SCORE and Accuracy of the four different models

3.9 Sparrow search algorithm (SSA):

For a given set, finding the best solution or bag of solutions to maintain a minimum or maximum as an objective f(x), while keeping a specific constraint in mind, is the purpose of an optimization issue. Meta heuristic algorithms can tackle any optimization challenge. Large-scale data problems can also be handled quickly and in a short amount of time. However, it does not guarantee a global optimal or bounded solution. (Talbi, 2020).

P-metaheuristics, or population-based metaheuristics, are iterative improvements to a population of solutions. The concept begins with a set of solutions, then loops through them to generate new numbers that replace the existing population. The creation phase generates new population solutions, whilst the substitution phase replaces existing solutions with new ones; this approach is repeated until a certain endpoint is reached.(Osaba et al., n.d.). Mathematical interpretation of algorithm (Xue & Shen, 2020). We must employ digital sparrows to find food in the simulation experiment. The following matrix represents the position of sparrows:

$$X = \begin{vmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,d} \end{vmatrix} \tag{3.8.1}$$

where n stands for the number of sparrows to be improved and d is the size of the variables to be maximized. After that, the vector can be used to express the fitness score of all sparrows:

$$Fx = \begin{bmatrix} f([x_{1,1} & x_{1,2} & \cdots & x_{1,d}]) \\ f([x_{2,1} & x_{2,1} & \cdots & x_{2,d}]) \\ \vdots & \vdots & \vdots & \vdots \\ f([x_{n,1} & x_{n,2} & \cdots & x_{n,d}]) \end{bmatrix} \tag{3.8.2}$$

where the value of each row in F_x indicates the individual's fitness value and n represents the number of sparrows. Food is prioritized in the SSA throughout the search process for farmers with higher fitness ratings. Furthermore, because the producers oversee finding food and govern the entire population's movement. As a result, producers have more options for finding food than scroungers. The producer's location is updated as follows during each iteration in compliance with rules (1) and (2).

$$X_{i,j}^{a+1} = \begin{cases} X_{i,j}^{a} . \exp\left(\frac{-i}{a.iter_{max}}\right) & if\ R_2 < ST \\ X_{i,j}^{a} + Q. & if\ R_2 > ST \end{cases} \tag{3.8.3}$$

where a is the current iteration and j denotes the number of iterations (1, 2..., d). $X_{i,j}^{a}$ denotes, at iteration a, the jth parameter of the ith sparrow is used. $iter_{max}$ is a constant that represents the maximum number of possible iterations. $\alpha \in [0, 1]$ is a random number. R_2 and ST denote the alert value ($R_2 \in [0, 1]$) and the secure threshold ($ST \in [0.5, 1.0]$, respectively. Q is a random number that carries normal distribution. L represents a one-dimensional matrix, with each element within being 1.

When $R_2 < ST$, the producer enters the wide search mode, suggesting that no predators are present. If $R_2 \geq ST$, it signifies that some sparrows have detected the predator and that all sparrows must evacuate promptly to another safe region.

$$X_p^{a+1} = \begin{cases} Q.exp\left(\frac{X_{worst}^a - X_{i,j}^a}{i^2}\right) & if\ I > n/2 \\ X_p^{a+1} + |X_{i,j}^a - X_p^{a+1}|.A^+.L & otherwise \end{cases} \tag{3.8.4}$$

where X_p represents the producer's best position. X_{worst} denotes the current global worst place. B is a one-dimensional array in which each member is randomly assigned a value of 1 or -1, and $B^+ = B^T (BB^T)^{-1}$. When i is more than n/2, the ith scrounger with the lowest fitness value is most happen to be starving.

The placements of these sparrows are produced at random in the population.

$$X_p^{a+1} = \begin{cases} X_{best}^a + \beta.|X_{i,j}^a - X_{bset}^a| & if\ f_i > f_g \\ X_{i,j}^a + K.\left(\frac{|X_{i,j}^a - X_{worst}^a|}{(f_i - f_w) + \varepsilon}\right) & if\ f_i = f_j \end{cases} \tag{3.8.5}$$

where X_{best} represents the current best location on the planet. The size of the step control parameter is the normal distribution with a mean of 0 and a variance of 1. K ∈ [-1, 1] is a chance number. fi is the sparrow's current fitness value. The present global best and worst fitness values are fg and f_w. The most minimal constant is used to avoid zero-division error. To keep things simple, when $f_i > f_g$ denotes that the sparrow is on the periphery of the group. X_{best} signifies the center of the population and is safe in its neighbourhood. Because fi = f_g, the sparrows in the central part of the group are aware of the risk and must relocate nearer to the others. K denotes the sparrow's motion direction and is also the step size control coefficient.

Feature Selection: The sparrow search algorithm aims to find the best subset of features that maximizes the accuracy of the AdaBoost classifier, Random Forest, K-Nearest Neighbours, Decision Tree, and AdaBoost. This is a type of feature selection technique where the model learns which features are most relevant for prediction.

S No.	Optimization of ML model with SSA	No. of Iteration	Accuracy of optimized model
1)	Implementation of SSA in Decision Tree Classifier.	N= 100	86.4%
2)	Implementation of SSA in Random Forest Classifier.	N= 100	92.96%
3)	Implementation of SSA in AdaBoost Classifier.	N= 50	84.21%
4)	Implementation of SSA in KNN Classifier.	N=100	85.41%

Table 3.3 Comparison of machine learning model accuracies of three different cases

The result in the table shows that the random forest classifier with SSA achieved the highest accuracy (92.96%) compared to all other models. An accuracy 0.9296 suggests the model is very good at distinguishing good suppliers from bad ones. There's only a chance of 7.04% chance the model might misclassify a supplier. This suggests that using SSA to select relevant features can significantly improve the accuracy of machine learning models used in supplier selection for logistics. The code for Implementation of SSA in ML Classifier is provided in the Appendices.

CHAPTER 4

RESULT AND DISCUSSION

The results of the study can be evaluated by looking at how well the model predicts suppliers. The secondary sources from which the dataset was gathered can be found on the cited website.

Accuracy comparison of models with a default parameter setting. We compared the models' accuracy in this study. The accuracy of four base classification supervised machine learning algorithm models along with base model after RandomOversampling and base model with SSA is displayed in the bar graph below Fig 4.1.

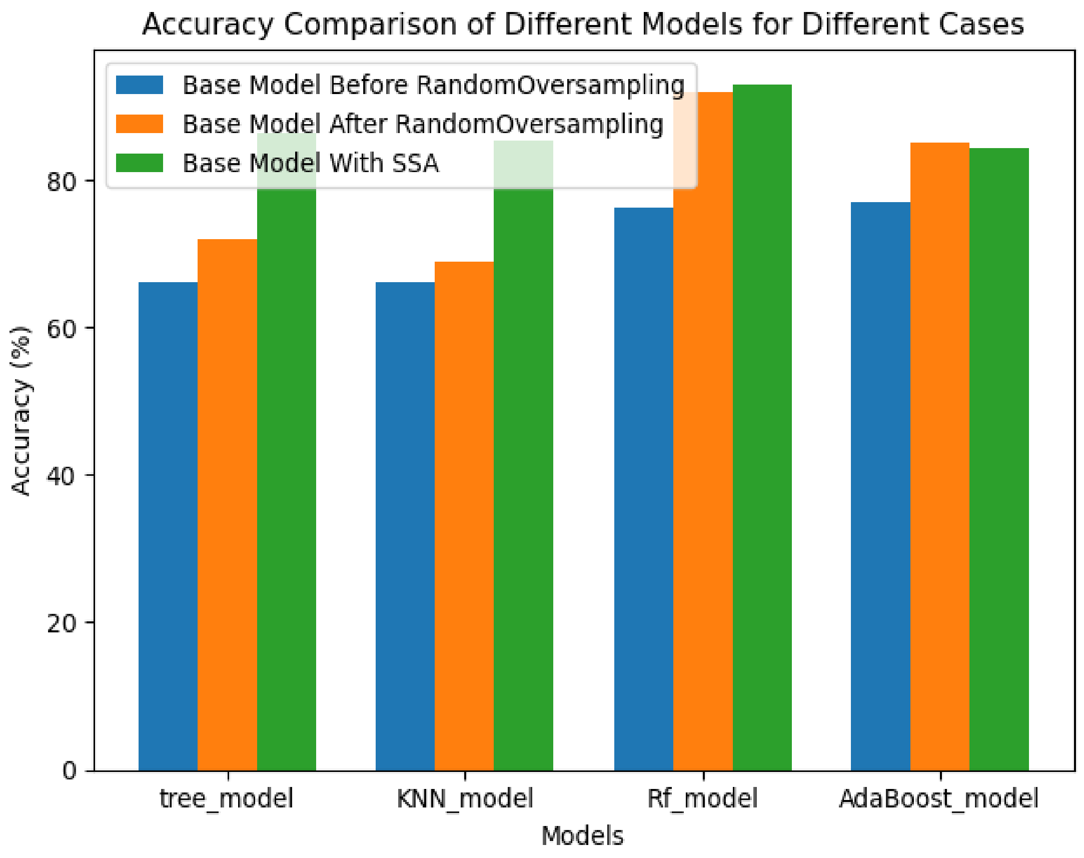


Fig 4.1: Comparison of machine learning model accuracies of three different cases

Table 4.1 represents the Accuracy of the four different models of three different cases. The Random Forest model has the best accuracy of 92.96% followed by the Decision Tree model with an accuracy of 86.41% and KNN model with an accuracy of 85.41% . The least good model is the Adaboost model with an accuracy of 84.41%.

	Models	Accuracy
Base Model Before RandomOversampling	AdaBoost(4K+ Data)	77%
	Random Forest(4K+ Data)	76.28%
	KNN(4K+ Data)	66.23%
	Decision Tree(4K+ Data)	66.01%
Base Model After RandomOversampling	Random Forest(90K+ Data)	91%
	AdaBoost(90K+ Data)	85%
	Decision Tree(90K+ Data)	72%
	KNN(90K+ Data)	69%
Base Model With SSA	Random Forest with SSA	92.96%
	Decision Tree with SSA	86.40%
	KNN with SSA	85.42%
	AdaBoost with SSA	84.21%

Table 4.1. Accuracy of the four different models of 3 different cases

Transportation logistics management and supply chain models could be greatly improved by applying SML to database files. This is because supervised machine learning models enable concept reduction in risk model management by analysing historical data, identifying trends, and simulating a variety of simplifying assumptions. Numerous analytics techniques exist that can be tried to solve conventional production issues, given the increasing reliance on statistical methods in SCM. To fully explore SML's potential to enhance SCM and transportation logistics management, more research is necessary.

CHAPTER 5

CONCLUSION

This research employed various supervised machine learning algorithm to predict the class accurately. Due to substantial amount of missing data under “PO Sent to Vendor Date” we cannot use lead time as one of the independent feature for our machine learning model. For better accuracy use the base machine learning models for vendor selection with other metaheuristic algorithms also such as genetic algorithm, ant colony optimisation or by manipulation of hyperparameters. Given the narrow scope of our information source (freight cost, weight, unit prices, line-item value, and line-item quantity), machine learning (ML) may be a better fit for larger, more complex data sets, which could be advantageous for organisations with a data-driven culture.

FUTURE SCOPE

Future research could benefit from combining different machine learning algorithms. It could also be more likely to demonstrate more reliable results in a practical application if there are large real-world data sets available. In order to improve the supplier selection process in the shipping logistics sector, future research could investigate sophisticated algorithms like convolutional neural networks (CNNs) and deep learning.

REFERENCE

Ang, E., Iancu, D. A., & Swinney, R. (2017). Disruption risk and optimal sourcing in multitier supply networks. *Management Science*, 63(8), 2397–2419.

Bajomo, M., Ogbeyemi, A., & Zhang, W. (2022). A systems dynamics approach to the management of material procurement for Engineering, Procurement and Construction industry. *International Journal of Production Economics*, 244.

Ballou, R.H., & Srivastava, S.K. (2019). *Business logistics / supply chain management: Integrating operations across the enterprise*. Pearson Education.

Bartle, J. R., Lutte, R. K., & Leuenberger, D. Z. (2021). Sustainability and air freight transportation: Lessons from the global pandemic. Sustainability (Switzerland), 13(7).

Bergstra, J., Yamins, D., & Cox, D. D. (n.d.). Hyperopt: A Python Library for Optimizing the Hyperparameters of Machine Learning Algorithms. In PROC. OF THE 12th PYTHON IN SCIENCE CONF. (SCIPY (Vol. 2013, Issue 1)

Büyüközkan, G. and Güleryüz, Ö., 2011. A fuzzy multi-criteria decision framework for sustainable supplier selection. *Journal of Cleaner Production*, 19(9-10), pp.980-987.

Cavalcante, I. M., Frazzon, E. M., Forcellini, F. A., & Ivanov, D. (2019). A Supervised Machine Learning Approach to Data-driven Simulation of Resilient Supplier Selection in Digital Manufacturing.

Chatzikontidou, A., Longinidis, P., Tsiakis, P., & Georgiadis, M. C. (2017). Flexible supply chain network design under uncertainty. Chemical Engineering Research and Design, 128, 290–305.

Cheng, F., Ettl, M., Lin, G., Schwarz, M., Hermes, E., & Yao, D. D. (2006). RC23917 (W0603-153) IBM Research Report Designing Flexible Supply Chain Contracts with Options Designing Flexible Supply Chain Contracts with Options.

Christopher, M. (2020). Logistics & supply chain management: Creating value-adding networks. Pearson Education.

Duan, Y., Edwards, J. S., & Dwivedi, Y. K. (n.d.-a). Artificial intelligence for decision making in the era of Big Data-evolution, challenges and research agenda (An opinion paper for *International Journal of Information Management*).

Fahimnia, B., Jabbarzadeh, A., Ghavamifar, A., & Bell, M. (2017). Supply chain design for efficient and effective blood supply in disasters. *International Journal of Production Economics*, 183, 700–709.

Gunasekaran, A., Dubey, R. and Papadopoulos, T., 2017. Sustainable operations: building resilience and the role of information systems. *International Journal of Production Research*, 55(8), pp.2206-2226.

Gunasekaran, A., Dubey, R., & Papadopoulos, T. (2017). Sustainable operations: Building resilience and the role of information systems. *International Journal of Production Research*, 55(8), 2206-2226.

Gupta, S., Soni, U., & Kumar, G. (2019). Green supplier selection using multi-criterion decision making under fuzzy environment: A case study in automotive industry. *Computers and Industrial Engineering*, 136, 663–680.

Hipólito, T., Lemos Nabais, J., Ayala Botto, M., & Negenborn, R. R. (2020). Effective Continuousflow Supply Chains Using Centralized Model Predictive Control.

Iqbal H. Sarker (2021) Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science Volume 2*, article number 160.

Ismail, H. S., & Sharifi, H. (2006). A balanced approach to building agile supply chains. *International Journal of Physical Distribution and Logistics Management*, 36(6), 431–444.

Kosasih, E. E., & Brintrup, A. (2022a). A machine learning approach for predicting hidden links in supply chain with graph neural networks. *International Journal of Production Research*, 60(17), 5380–5393. https://doi.org/10.1080/00207543.2021.1956697

Kosasih, E. E., & Brintrup, A. (2022b). A machine learning approach for predicting hidden links in supply chain with graph neural networks. *International Journal of Production Research*, 60(17), 5380–5393.

Lasi, H., Fettke, P., Kemper, H. G., Feld, T., & Hoffmann, M. (2014). Industry 4.0. Business and Information Systems Engineering, 6(4), 239–242.

Lee, C.-Y., & Chens, Z.-L. (2001). Machine scheduling with transportation considerations. In J. Sched (Vol. 4).

Lee, C.-Y., & Chens, Z.-L. (2001). Machine scheduling with transportation considerations. In J. Sched (Vol. 4).

Macchion, L., Fornasiero, R., & Vinelli, A. (2017). Supply chain configurations: a model to evaluate performance in customised productions. *International Journal of Production Research*, 55(5), 1386–1399.

McCarthy, J., Minsky, M. L., Papert, S. A., Shannon, C. E. (1959). Proposals for research on artificial intelligence. MIT Lincoln Laboratory.

McKinsey Global Institute. (2017). Jobs lost, jobs gained: Workforce transitions in a time of automation.

Mehralian, G., Zarenezhad, F., & Ghatari, A. R. (2015). Developing a model for an agile supply chain in pharmaceutical industry. *International Journal of Pharmaceutical and Healthcare Marketing*, 9(1), 74–91.

Mentzer, J.T., Dewitt, W., Bradford, J.S., Spekman, R.E., & Cachon, L.P. (2009). *Supply chain management: Creating and capturing value*. Cengage Learning.

Nagar, D., Raghav, S., Bhardwaj, A., Kumar, R., Lata Singh, P., & Sindhwani, R. (2021). Machine learning: Best way to sustain the supply chain in the era of industry 4.0. Materials Today: Proceedings, 47, 3676–3682.

Ntabe, E. N., LeBel, L., Munson, A. D., & Santa-Eulalia, L. A. (2015). A systematic literature review of the supply chain operations reference (SCOR) model application with special attention to environmental issues. *International Journal of Production Economics*, 169, 310– 332.

Pedregosa FABIANPEDREGOSA, F., Michel, V., Grisel OLIVIERGRISEL, O., Blondel, M., Prettenhofer, P., Weiss, R., Vanderplas, J., Cournapeau, D., Pedregosa, F., Varoquaux, G., Gramfort, A., Thirion, B., Grisel, O., Dubourg, V., Passos, A., Brucher, M., Perrot andÉdouardand, M., Duchesnay, andÉdouard, & Duchesnay EDOUARDDUCHESNAY, Fré. (2011). Scikit-learn: Machine Learning in Python Gaël Varoquaux Bertrand Thirion Vincent Dubourg Alexandre Passos PEDREGOSA, VAROQUAUX, GRAMFORT ET AL. Matthieu Perrot. In *Journal of Machine Learning Research* (Vol. 12).

Jiankai, X., Bo, S., (2020) A novel swarm optimization approach: Sparrow search algorithm. *Systems Science and Control Engineering*, Vol.8, 22-34.

Rajagopal, V., Prasanna Venkatesan, S., & Goh, M. (2017). Decision-making models for supply chain risk mitigation: A review. In *Computers and Industrial Engineering* (Vol. 113, pp. 646– 682).

Raman, S., Patwa, N., Niranjan, I., Ranjan, U., Moorthy, K., & Mehta, A. (2018). Impact of big data on supply chain management. *International Journal of Logistics Research and Applications*, 21(6), 579–596.

Rondinelli, D., & Berry, M. (2000). Multimodal Transportation, Logistics, and the Environment: Managing Interactions in a Global Economy. In European *Management Journal* (Vol. 18, Issue 4).

Russell, S. J., & Norvig, P. (2021). Artificial intelligence: A modern approach (4th ed.). Pearson Education.

Scheibe, K. P., & Blackhurst, J. (2018). Supply chain disruption propagation: a systemic risk and normal accident theory perspective. *International Journal of Production Research*, 56(1–2), 43–59

Talbi, E.-G. (2020). Machine learning into metaheuristics: A survey and taxonomy of data-driven metaheuristics.

Tsolaki, K., Vafeiadis, T., Nizamis, A., Ioannidis, D., & Tzovaras, D. (2022a). Utilizing machine learning on freight transportation and logistics applications: A review. In ICT Express. Korean Institute of Communication Sciences.

Wang, C., & Jiao, Y. (2022). Shipping companies' choice of low sulfur fuel oil with government subsidy and different maritime supply chain power structures. *Maritime Policy and Management*, 49(3), 323–346

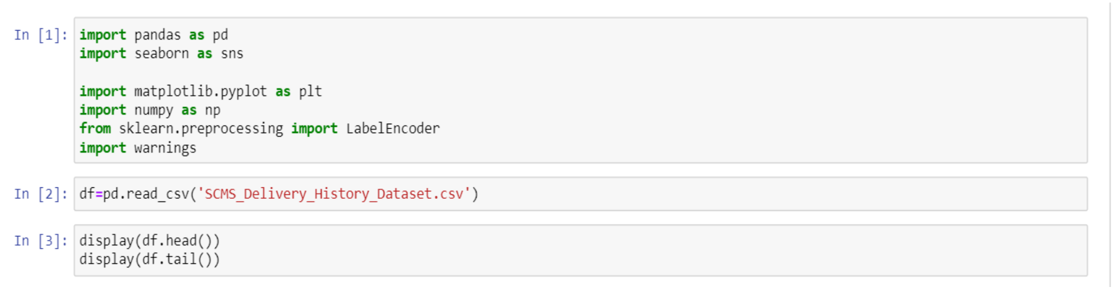
Wu, M., Zhang, W., Guo, S., Huang, W., Huang, H. and He, G., 2019. Application of machine learning techniques in supplier selection: A review and future directions. *International Journal of Production Research*, 57(13), pp.4205-4229.

Wuest, T., Weimer, D., Irgens, C., & Thoben, K. D. (2016a). Machine learning in manufacturing: Advantages, challenges, and applications. *Production and Manufacturing Research*, 4(1), 23– 45.

Zadeh, L.A., 1965. Fuzzy sets. Information and control, 8(3), pp.338-353.

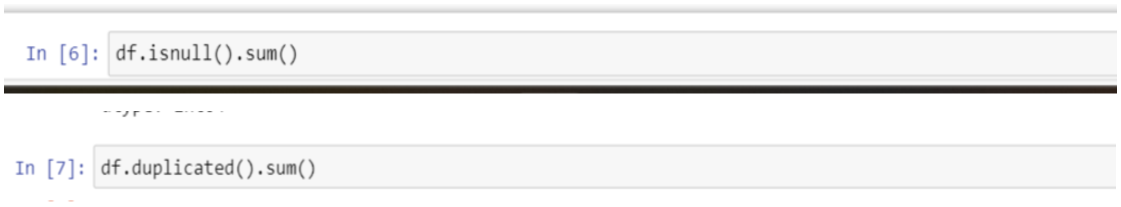
APPENDIX

Fig.1 Importing Python libraries on the Jupyter Notebook software. Python libraries are pre-written code snippets that offer tools and reusable functions for common tasks.



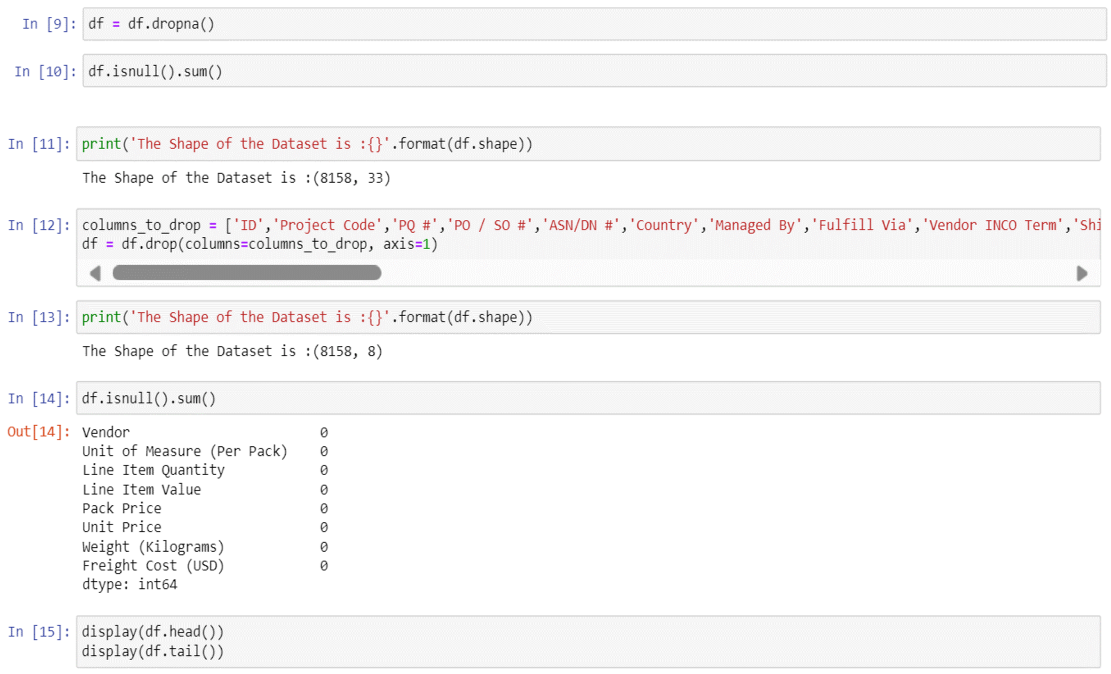
```
In [1]: import pandas as pd
        import seaborn as sns

        import matplotlib.pyplot as plt
        import numpy as np
        from sklearn.preprocessing import LabelEncoder
        import warnings

In [2]: df=pd.read_csv('SCMS_Delivery_History_Dataset.csv')

In [3]: display(df.head())
        display(df.tail())
```

Fig.2 Finding Null Va lues and Duplicate Values from the given dataset.



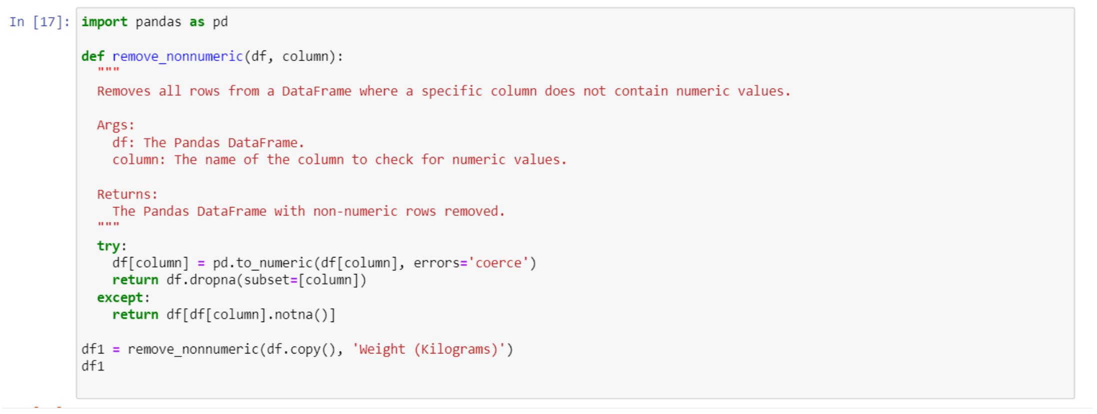
```
In [6]: df.isnull().sum()

In [7]: df.duplicated().sum()
```

Fig.3 Dropping Null values from the given dataset.

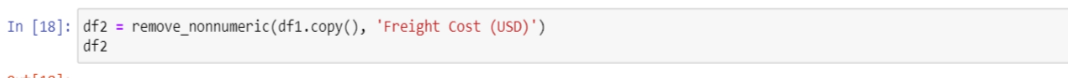


```
In [9]: df = df.dropna()

In [10]: df.isnull().sum()

In [11]: print('The Shape of the Dataset is :{}'.format(df.shape))

         The Shape of the Dataset is :(8158, 33)

In [12]: columns_to_drop = ['ID','Project Code','PQ #','PO / SO #','ASN/DN #','Country','Managed By','Fulfill Via','Vendor INCO Term','Shi
         df = df.drop(columns=columns_to_drop, axis=1)

In [13]: print('The Shape of the Dataset is :{}'.format(df.shape))

         The Shape of the Dataset is :(8158, 8)

In [14]: df.isnull().sum()

Out[14]: Vendor                      0
         Unit of Measure (Per Pack)  0
         Line Item Quantity          0
         Line Item Value             0
         Pack Price                  0
         Unit Price                  0
         Weight (Kilograms)          0
         Freight Cost (USD)          0
         dtype: int64

In [15]: display(df.head())
         display(df.tail())
```

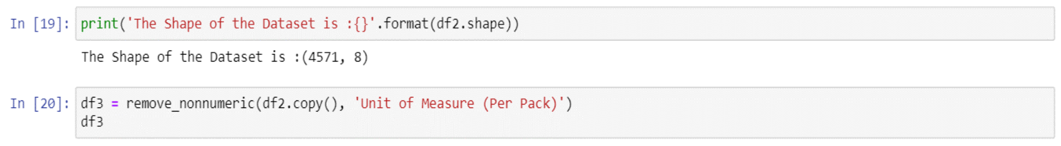
Fig.4 Removing non numeric column from the given dataset.



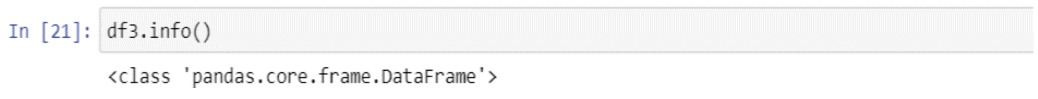
```
In [17]: import pandas as pd

         def remove_nonnumeric(df, column):
           """
           Removes all rows from a DataFrame where a specific column does not contain numeric values.

           Args:
             df: The Pandas DataFrame.
             column: The name of the column to check for numeric values.

           Returns:
             The Pandas DataFrame with non-numeric rows removed.
           """
           try:
             df[column] = pd.to_numeric(df[column], errors='coerce')
             return df.dropna(subset=[column])
           except:
             return df[df[column].notna()]

         df1 = remove_nonnumeric(df.copy(), 'Weight (Kilograms)')
         df1
```



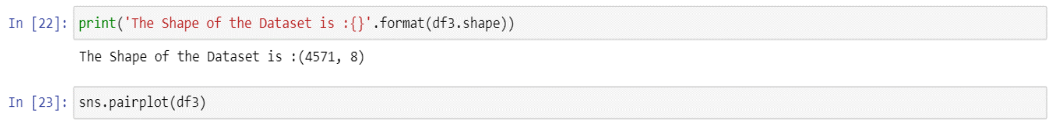
```
In [18]: df2 = remove_nonnumeric(df1.copy(), 'Freight Cost (USD)')
         df2
```



```
In [19]: print('The Shape of the Dataset is :{}'.format(df2.shape))

         The Shape of the Dataset is :(4571, 8)

In [20]: df3 = remove_nonnumeric(df2.copy(), 'Unit of Measure (Per Pack)')
         df3
```

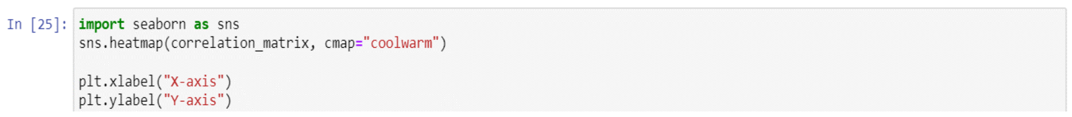


```
In [21]: df3.info()

         <class 'pandas.core.frame.DataFrame'>
```



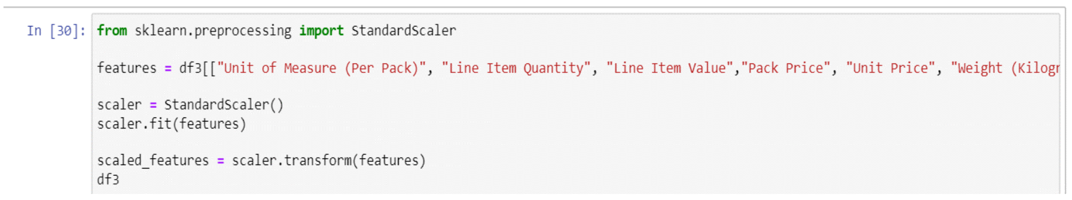
```
In [22]: print('The Shape of the Dataset is :{}'.format(df3.shape))

         The Shape of the Dataset is :(4571, 8)

In [23]: sns.pairplot(df3)
```

Fig.5 Pair plot graphs for each feature with one another and show the relationship between them, we used the Seaborn library.



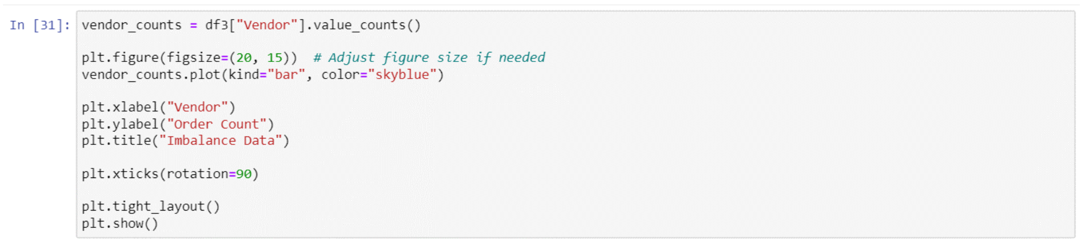
```
In [25]: import seaborn as sns
         sns.heatmap(correlation_matrix, cmap="coolwarm")

         plt.xlabel("X-axis")
         plt.ylabel("Y-axis")
```

Fig.6 Scaling of independent feature



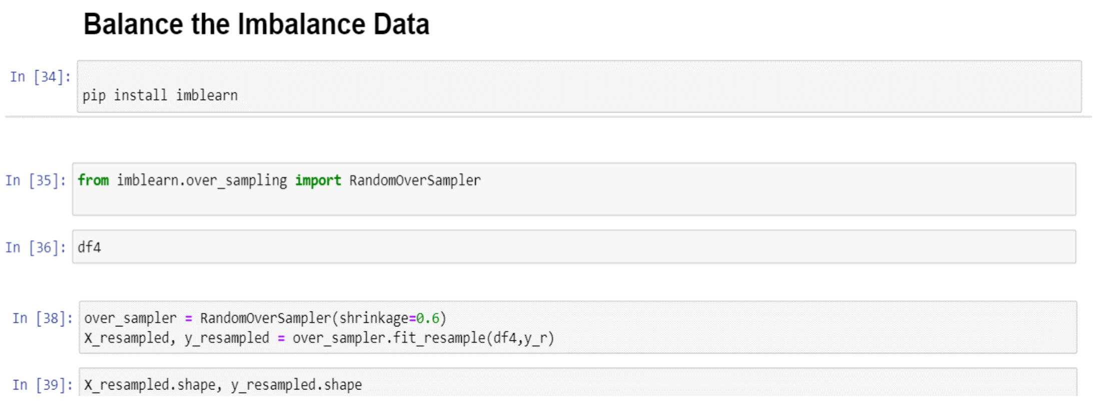
```
In [30]: from sklearn.preprocessing import StandardScaler

         features = df3[["Unit of Measure (Per Pack)", "Line Item Quantity", "Line Item Value","Pack Price", "Unit Price", "Weight (Kilogr

         scaler = StandardScaler()
         scaler.fit(features)

         scaled_features = scaler.transform(features)
         df3
```

Fig.7 Code for Representation of Imbalanced Data Set



```
In [31]: vendor_counts = df3["Vendor"].value_counts()

         plt.figure(figsize=(20, 15))  # Adjust figure size if needed
         vendor_counts.plot(kind="bar", color="skyblue")

         plt.xlabel("Vendor")
         plt.ylabel("Order Count")
         plt.title("Imbalance Data")

         plt.xticks(rotation=90)

         plt.tight_layout()
         plt.show()
```

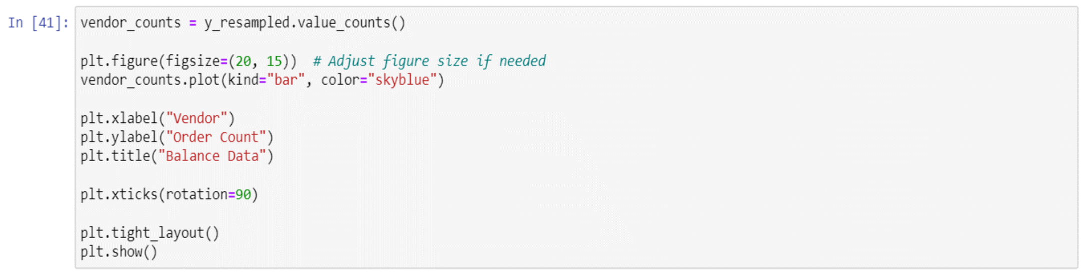
Fig.8 Code for balancing of Imbalanced Data Set



Balance the Imbalance Data

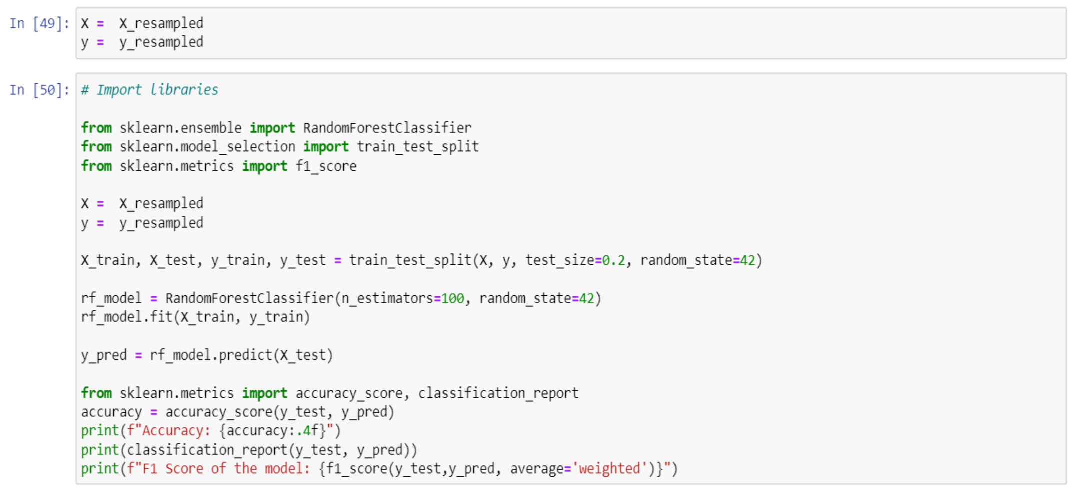
```
In [34]: pip install imblearn

In [35]: from imblearn.over_sampling import RandomOverSampler

In [36]: df4

In [38]: over_sampler = RandomOverSampler(shrinkage=0.6)
         X_resampled, y_resampled = over_sampler.fit_resample(df4,y_r)

In [39]: X_resampled.shape, y_resampled.shape
```

Fig.9 Code for Representation of balanced Data Set



```
In [41]: vendor_counts = y_resampled.value_counts()

         plt.figure(figsize=(20, 15))  # Adjust figure size if needed
         vendor_counts.plot(kind="bar", color="skyblue")

         plt.xlabel("Vendor")
         plt.ylabel("Order Count")
         plt.title("Balance Data")

         plt.xticks(rotation=90)

         plt.tight_layout()
         plt.show()
```

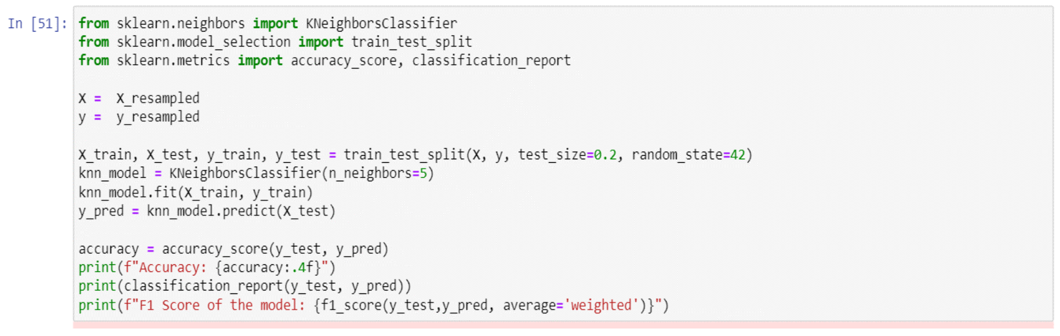
Fig.10 Code for Output of Random Forest Classifier



```
In [49]: X =  X_resampled
         y =  y_resampled
```

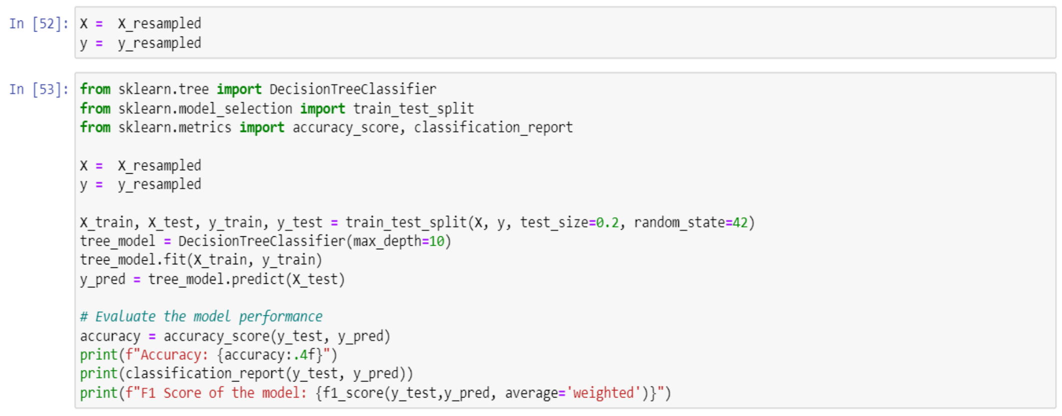
```
In [50]: # Import libraries

         from sklearn.ensemble import RandomForestClassifier
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import f1_score

         X =  X_resampled
         y =  y_resampled

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
         rf_model.fit(X_train, y_train)

         y_pred = rf_model.predict(X_test)

         from sklearn.metrics import accuracy_score, classification_report
         accuracy = accuracy_score(y_test, y_pred)
         print(f"Accuracy: {accuracy:.4f}")
         print(classification_report(y_test, y_pred))
         print(f"F1 Score of the model: {f1_score(y_test,y_pred, average='weighted')}")
```

Fig.11 Code for Output of KNN Classifier



```
In [51]: from sklearn.neighbors import KNeighborsClassifier
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score, classification_report

         X =  X_resampled
         y =  y_resampled

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
         knn_model = KNeighborsClassifier(n_neighbors=5)
         knn_model.fit(X_train, y_train)
         y_pred = knn_model.predict(X_test)

         accuracy = accuracy_score(y_test, y_pred)
         print(f"Accuracy: {accuracy:.4f}")
         print(classification_report(y_test, y_pred))
         print(f"F1 Score of the model: {f1_score(y_test,y_pred, average='weighted')}")
```

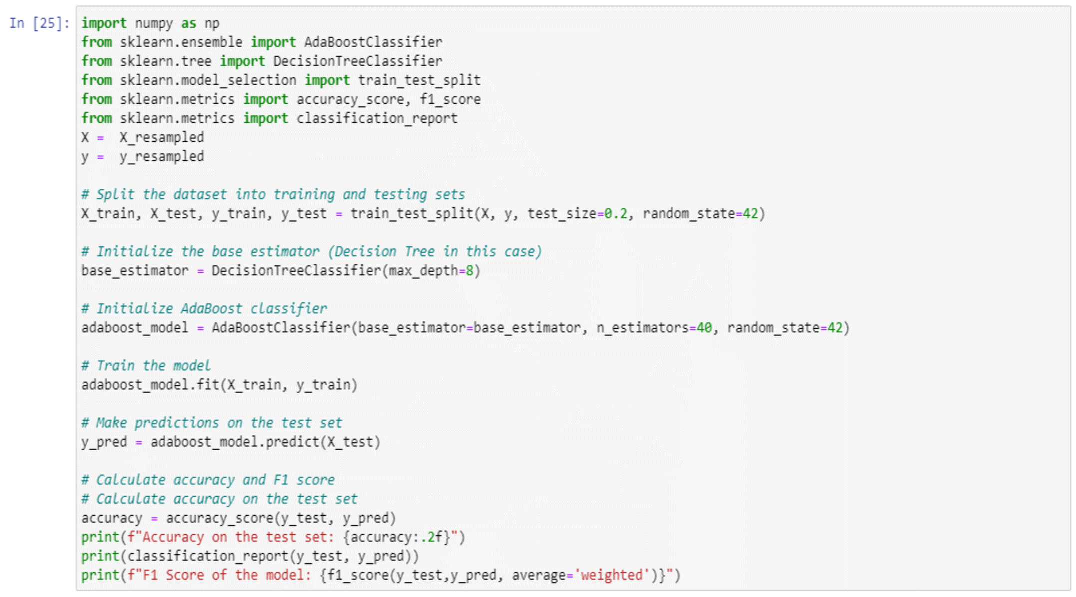
Fig.12 Code for Output of Decision Tree Classifier



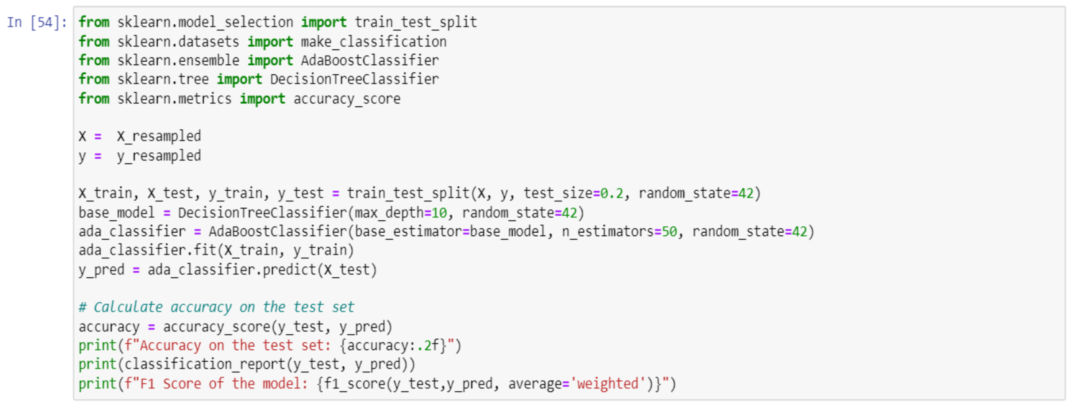
```
In [52]: X =  X_resampled
         y =  y_resampled
```

```
In [53]: from sklearn.tree import DecisionTreeClassifier
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score, classification_report

         X =  X_resampled
         y =  y_resampled

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
         tree_model = DecisionTreeClassifier(max_depth=10)
         tree_model.fit(X_train, y_train)
         y_pred = tree_model.predict(X_test)

         # Evaluate the model performance
         accuracy = accuracy_score(y_test, y_pred)
         print(f"Accuracy: {accuracy:.4f}")
         print(classification_report(y_test, y_pred))
         print(f"F1 Score of the model: {f1_score(y_test,y_pred, average='weighted')}")
```

Fig.13 Code for Output of AdaBoost Classifier with Decision Tree as base estimator.

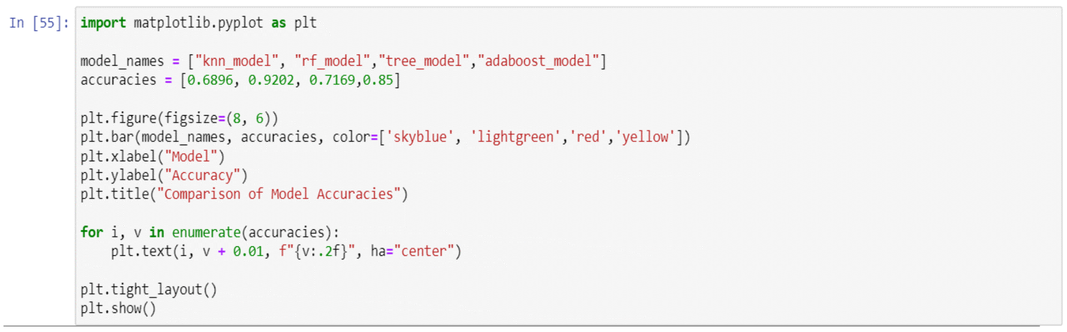


```
In [25]: import numpy as np
         from sklearn.ensemble import AdaBoostClassifier
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score, f1_score
         from sklearn.metrics import classification_report
         X =  X_resampled
         y =  y_resampled

         # Split the dataset into training and testing sets
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # Initialize the base estimator (Decision Tree in this case)
         base_estimator = DecisionTreeClassifier(max_depth=8)

         # Initialize AdaBoost classifier
         adaboost_model = AdaBoostClassifier(base_estimator=base_estimator, n_estimators=40, random_state=42)

         # Train the model
         adaboost_model.fit(X_train, y_train)

         # Make predictions on the test set
         y_pred = adaboost_model.predict(X_test)

         # Calculate accuracy and F1 score
         # Calculate accuracy on the test set
         accuracy = accuracy_score(y_test, y_pred)
         print(f"Accuracy on the test set: {accuracy:.2f}")
         print(classification_report(y_test, y_pred))
         print(f"F1 Score of the model: {f1_score(y_test,y_pred, average='weighted')}")
```



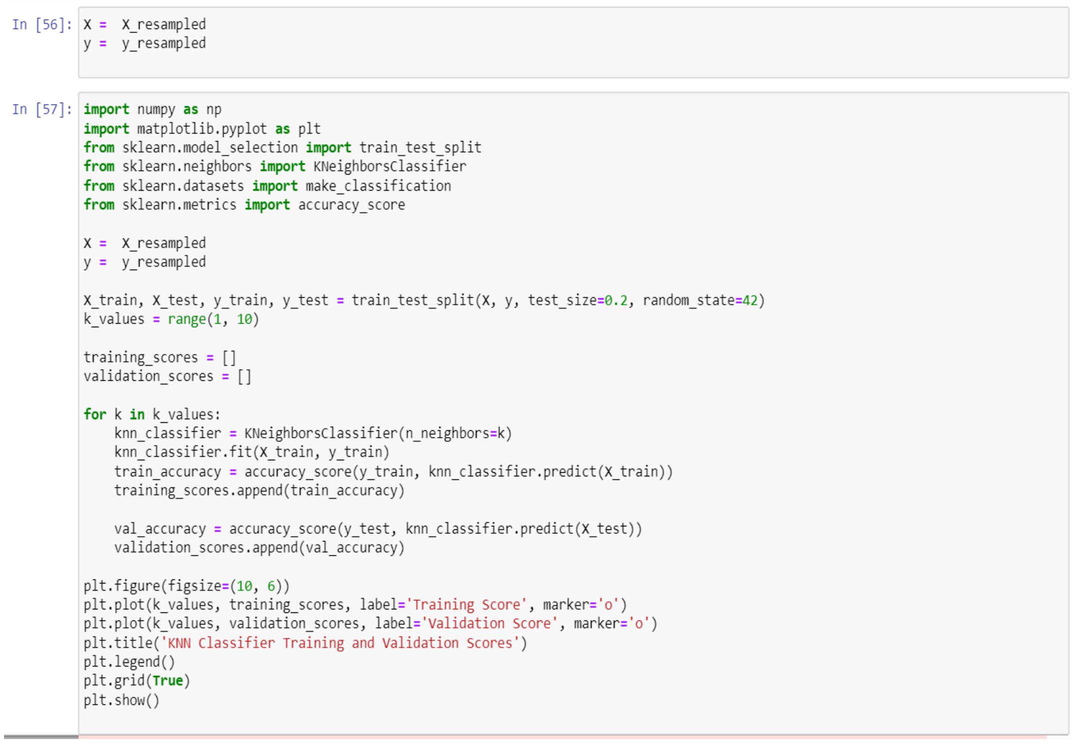
```
In [54]: from sklearn.model_selection import train_test_split
         from sklearn.datasets import make_classification
         from sklearn.ensemble import AdaBoostClassifier
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.metrics import accuracy_score

         X =  X_resampled
         y =  y_resampled

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
         base_model = DecisionTreeClassifier(max_depth=10, random_state=42)
         ada_classifier = AdaBoostClassifier(base_estimator=base_model, n_estimators=50, random_state=42)
         ada_classifier.fit(X_train, y_train)
         y_pred = ada_classifier.predict(X_test)

         # Calculate accuracy on the test set
         accuracy = accuracy_score(y_test, y_pred)
         print(f"Accuracy on the test set: {accuracy:.2f}")
         print(classification_report(y_test, y_pred))
         print(f"F1 Score of the model: {f1_score(y_test,y_pred, average='weighted')}")
```

Fig.14 Code for Comparison of Model Accuracies in Pyplot.



```
In [55]: import matplotlib.pyplot as plt

         model_names = ["knn_model", "rf_model","tree_model","adaboost_model"]
         accuracies = [0.6896, 0.9202, 0.7169,0.85]

         plt.figure(figsize=(8, 6))
         plt.bar(model_names, accuracies, color=['skyblue', 'lightgreen','red','yellow'])
         plt.xlabel("Model")
         plt.ylabel("Accuracy")
         plt.title("Comparison of Model Accuracies")

         for i, v in enumerate(accuracies):
             plt.text(i, v + 0.01, f"{v:.2f}", ha="center")

         plt.tight_layout()
         plt.show()
```

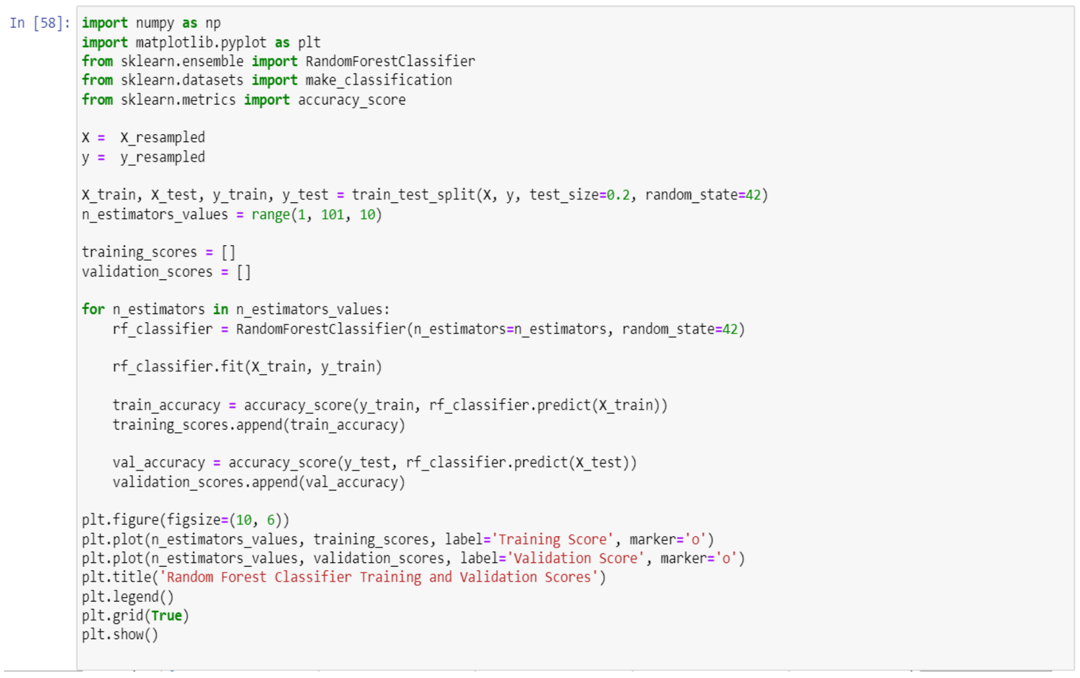
Fig.16 KNN Classifier Train Score v/s Validation Score code.



```
In [56]: X =  X_resampled
         y =  y_resampled
```

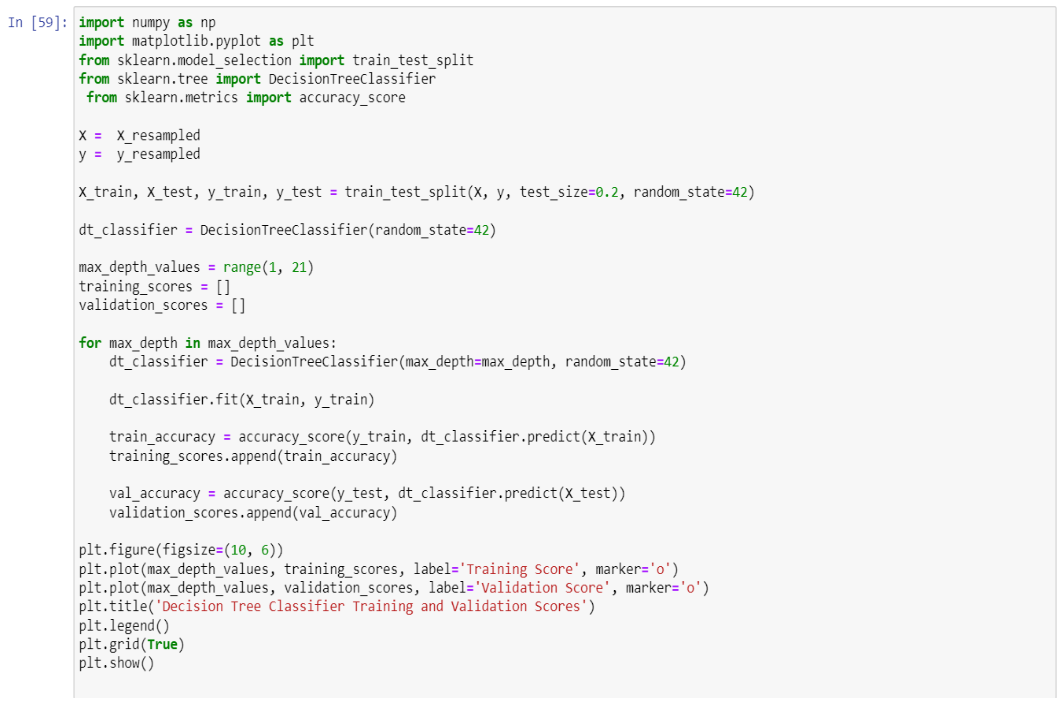
```
In [57]: import numpy as np
         import matplotlib.pyplot as plt
         from sklearn.model_selection import train_test_split
         from sklearn.neighbors import KNeighborsClassifier
         from sklearn.datasets import make_classification
         from sklearn.metrics import accuracy_score

         X =  X_resampled
         y =  y_resampled

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
         k_values = range(1, 10)

         training_scores = []
         validation_scores = []

         for k in k_values:
             knn_classifier = KNeighborsClassifier(n_neighbors=k)
             knn_classifier.fit(X_train, y_train)
             train_accuracy = accuracy_score(y_train, knn_classifier.predict(X_train))
             training_scores.append(train_accuracy)

             val_accuracy = accuracy_score(y_test, knn_classifier.predict(X_test))
             validation_scores.append(val_accuracy)

         plt.figure(figsize=(10, 6))
         plt.plot(k_values, training_scores, label='Training Score', marker='o')
         plt.plot(k_values, validation_scores, label='Validation Score', marker='o')
         plt.title('KNN Classifier Training and Validation Scores')
         plt.legend()
         plt.grid(True)
         plt.show()
```

Fig.17 Code for Random Forest Classifier Train Score v/s Validation Score.



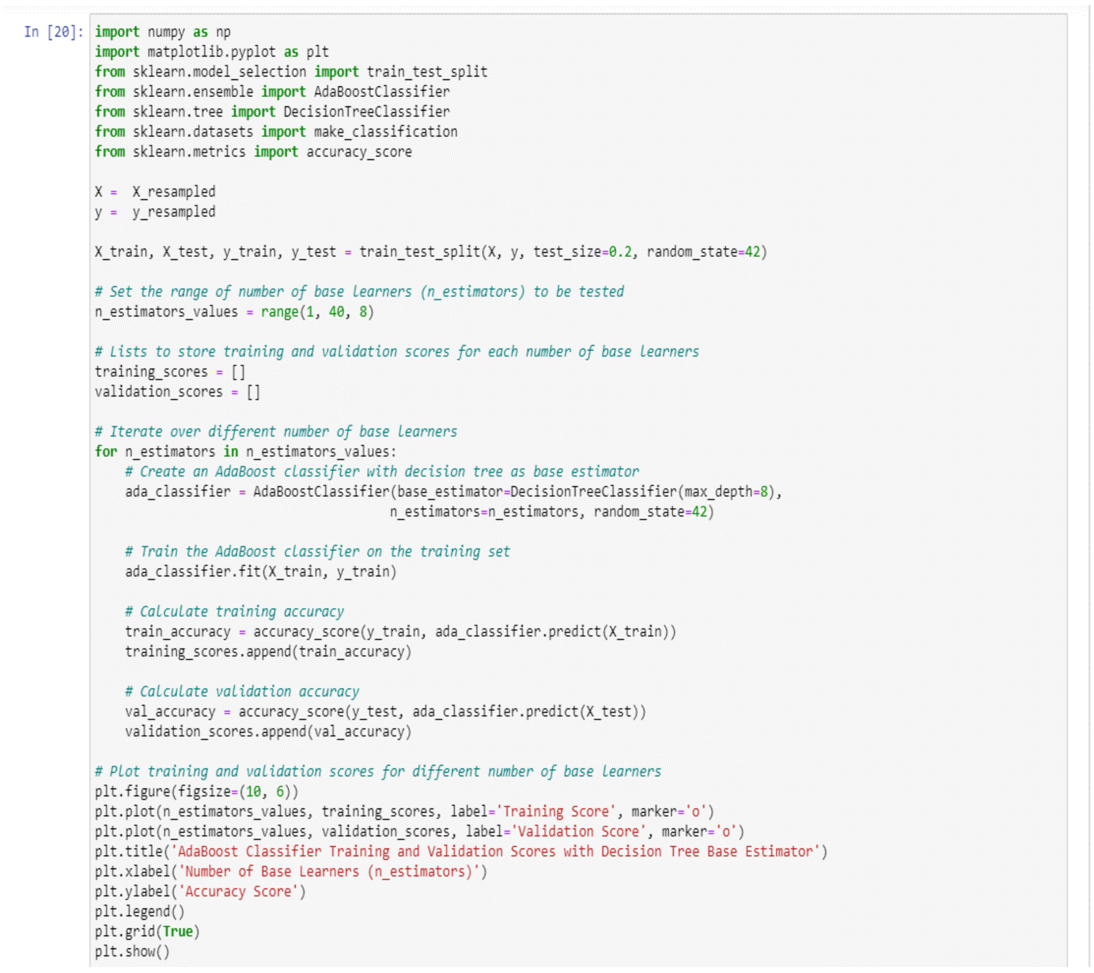
```
In [58]: import numpy as np
         import matplotlib.pyplot as plt
         from sklearn.ensemble import RandomForestClassifier
         from sklearn.datasets import make_classification
         from sklearn.metrics import accuracy_score

         X =  X_resampled
         y =  y_resampled

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
         n_estimators_values = range(1, 101, 10)

         training_scores = []
         validation_scores = []

         for n_estimators in n_estimators_values:
             rf_classifier = RandomForestClassifier(n_estimators=n_estimators, random_state=42)

             rf_classifier.fit(X_train, y_train)

             train_accuracy = accuracy_score(y_train, rf_classifier.predict(X_train))
             training_scores.append(train_accuracy)

             val_accuracy = accuracy_score(y_test, rf_classifier.predict(X_test))
             validation_scores.append(val_accuracy)

         plt.figure(figsize=(10, 6))
         plt.plot(n_estimators_values, training_scores, label='Training Score', marker='o')
         plt.plot(n_estimators_values, validation_scores, label='Validation Score', marker='o')
         plt.title('Random Forest Classifier Training and Validation Scores')
         plt.legend()
         plt.grid(True)
         plt.show()
```

Fig.18 Code of Decision Tree Classifier for Train Score v/s Validation Score.



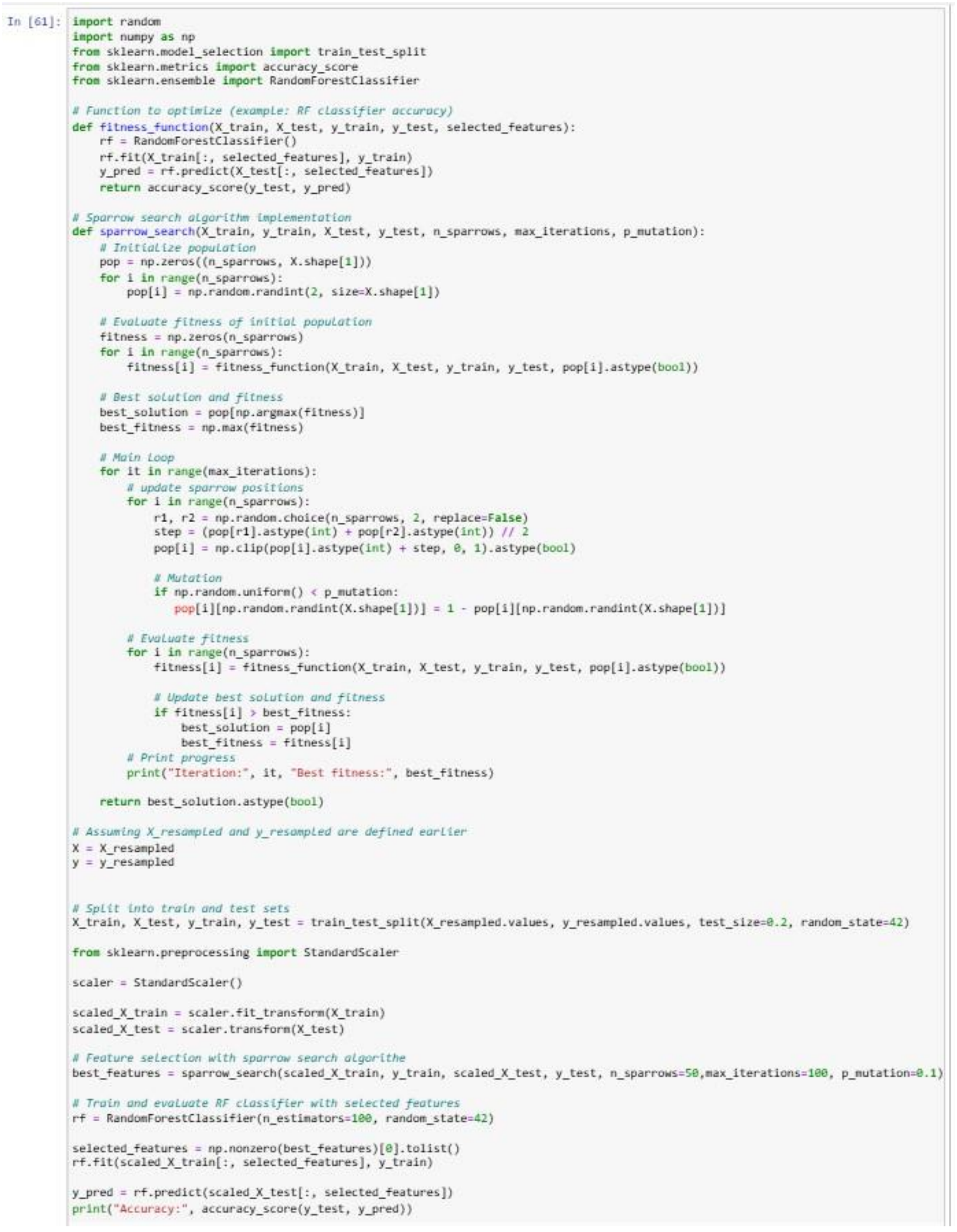
```
In [59]: import numpy as np
         import matplotlib.pyplot as plt
         from sklearn.model_selection import train_test_split
         from sklearn.tree import DecisionTreeClassifier
          from sklearn.metrics import accuracy_score

         X =  X_resampled
         y =  y_resampled

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         dt_classifier = DecisionTreeClassifier(random_state=42)

         max_depth_values = range(1, 21)
         training_scores = []
         validation_scores = []

         for max_depth in max_depth_values:
             dt_classifier = DecisionTreeClassifier(max_depth=max_depth, random_state=42)

             dt_classifier.fit(X_train, y_train)

             train_accuracy = accuracy_score(y_train, dt_classifier.predict(X_train))
             training_scores.append(train_accuracy)

             val_accuracy = accuracy_score(y_test, dt_classifier.predict(X_test))
             validation_scores.append(val_accuracy)

         plt.figure(figsize=(10, 6))
         plt.plot(max_depth_values, training_scores, label='Training Score', marker='o')
         plt.plot(max_depth_values, validation_scores, label='Validation Score', marker='o')
         plt.title('Decision Tree Classifier Training and Validation Scores')
         plt.legend()
         plt.grid(True)
         plt.show()
```

Fig.19 Code for AdaBoost Classifier Train Score v/s Validation Score.



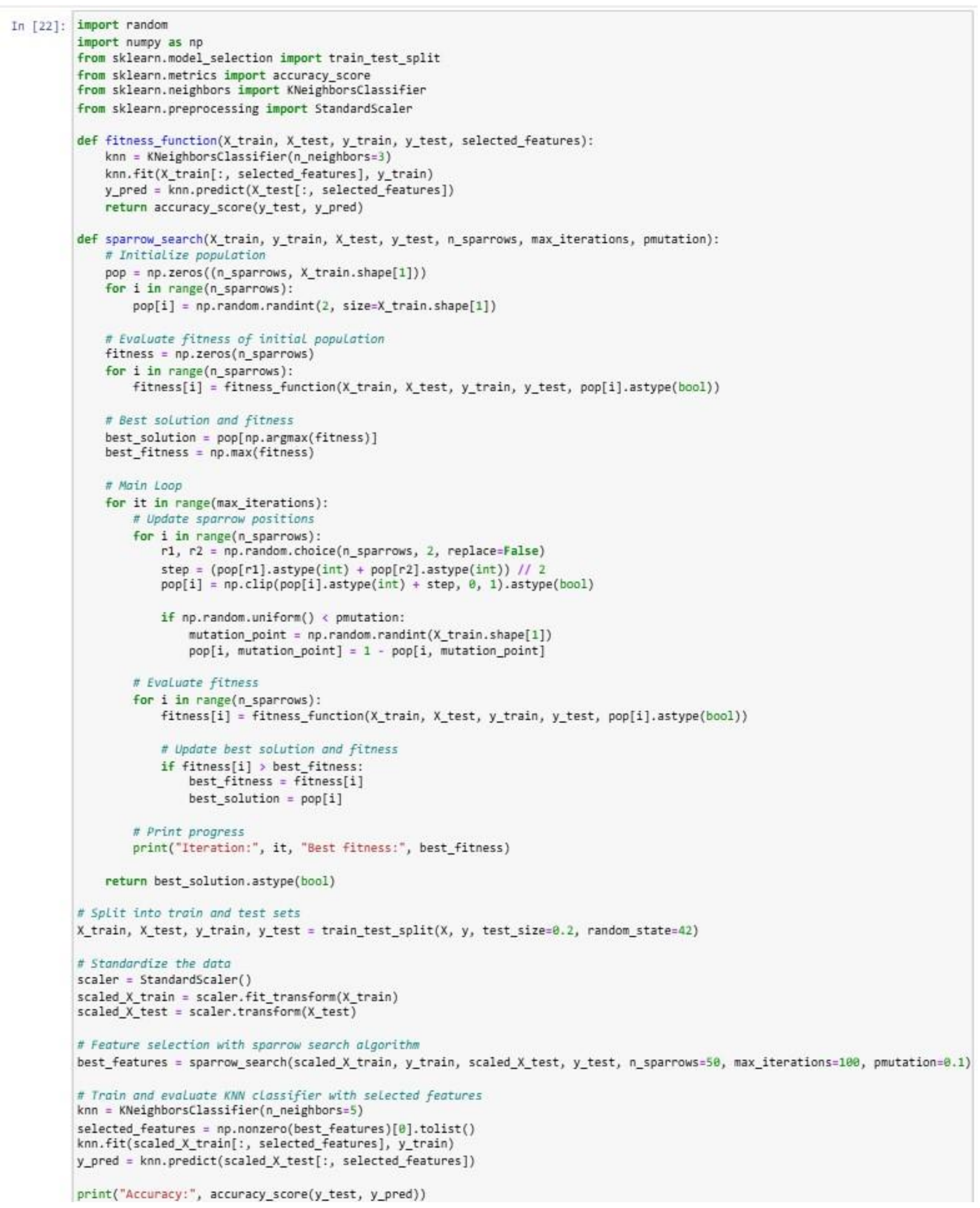
```
In [20]: import numpy as np
         import matplotlib.pyplot as plt
         from sklearn.model_selection import train_test_split
         from sklearn.ensemble import AdaBoostClassifier
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.datasets import make_classification
         from sklearn.metrics import accuracy_score

         X =  X_resampled
         y =  y_resampled

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # Set the range of number of base learners (n_estimators) to be tested
         n_estimators_values = range(1, 40, 8)

         # Lists to store training and validation scores for each number of base learners
         training_scores = []
         validation_scores = []

         # Iterate over different number of base learners
         for n_estimators in n_estimators_values:
             # Create an AdaBoost classifier with decision tree as base estimator
             ada_classifier = AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=8),
                                                 n_estimators=n_estimators, random_state=42)

             # Train the AdaBoost classifier on the training set
             ada_classifier.fit(X_train, y_train)

             # Calculate training accuracy
             train_accuracy = accuracy_score(y_train, ada_classifier.predict(X_train))
             training_scores.append(train_accuracy)

             # Calculate validation accuracy
             val_accuracy = accuracy_score(y_test, ada_classifier.predict(X_test))
             validation_scores.append(val_accuracy)

         # Plot training and validation scores for different number of base learners
         plt.figure(figsize=(10, 6))
         plt.plot(n_estimators_values, training_scores, label='Training Score', marker='o')
         plt.plot(n_estimators_values, validation_scores, label='Validation Score', marker='o')
         plt.title('AdaBoost Classifier Training and Validation Scores with Decision Tree Base Estimator')
         plt.xlabel('Number of Base Learners (n_estimators)')
         plt.ylabel('Accuracy Score')
         plt.legend()
         plt.grid(True)
         plt.show()
```

Fig.20 Code for Output of Random Forest with Sparrow Search Algorithm.



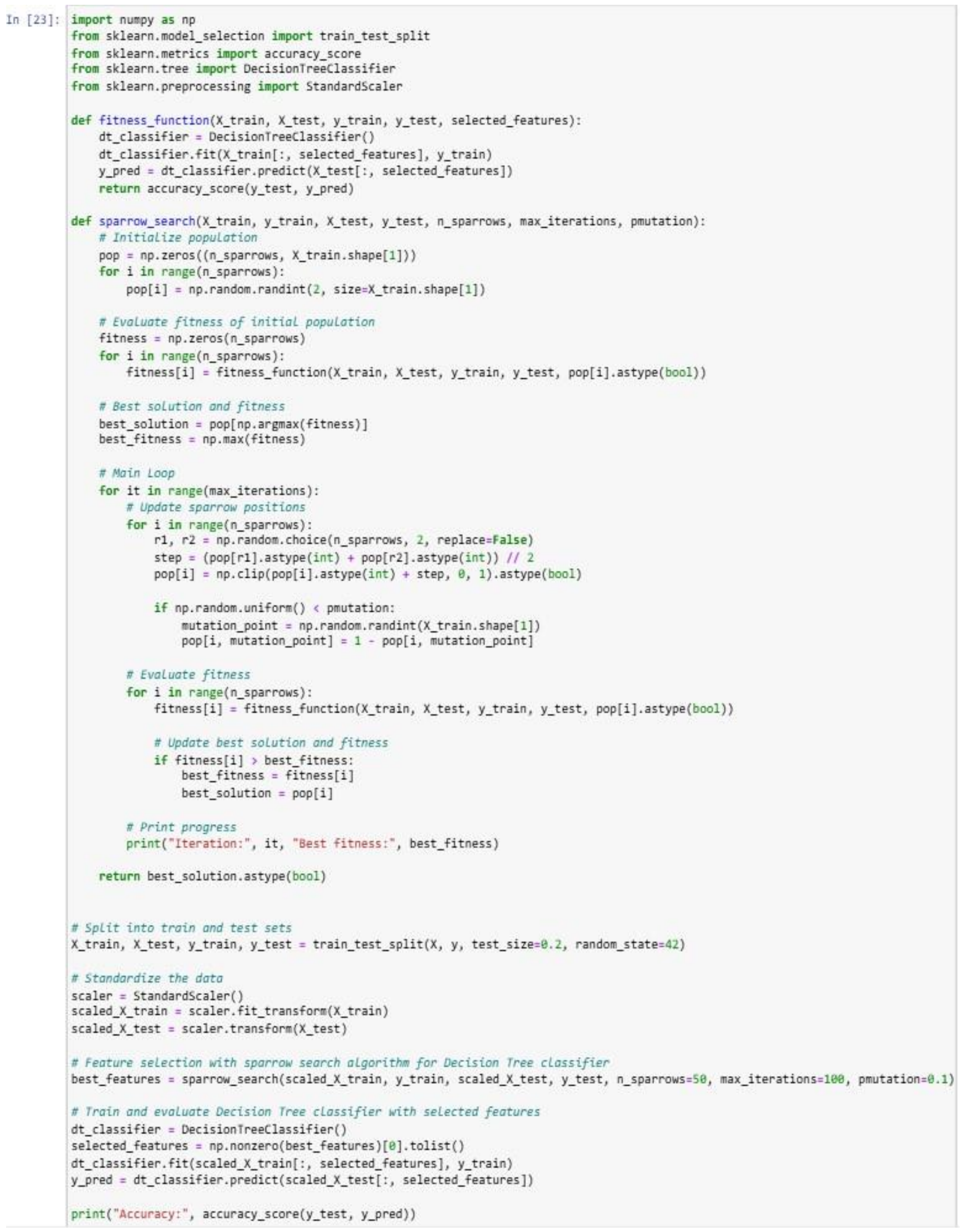
```
In [61]: import random
         import numpy as np
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score
         from sklearn.ensemble import RandomForestClassifier

         # Function to optimize (example: RF classifier accuracy)
         def fitness_function(X_train, X_test, y_train, y_test, selected_features):
             rf = RandomForestClassifier()
             rf.fit(X_train[:, selected_features], y_train)
             y_pred = rf.predict(X_test[:, selected_features])
             return accuracy_score(y_test, y_pred)

         # Sparrow search algorithm implementation
         def sparrow_search(X_train, y_train, X_test, y_test, n_sparrows, max_iterations, p_mutation):
             # Initialize population
             pop = np.zeros((n_sparrows, X.shape[1]))
             for i in range(n_sparrows):
                 pop[i] = np.random.randint(2, size=X.shape[1])

             # Evaluate fitness of initial population
             fitness = np.zeros(n_sparrows)
             for i in range(n_sparrows):
                 fitness[i] = fitness_function(X_train, X_test, y_train, y_test, pop[i].astype(bool))

             # Best solution and fitness
             best_solution = pop[np.argmax(fitness)]
             best_fitness = np.max(fitness)

             # Main Loop
             for it in range(max_iterations):
                 # update sparrow positions
                 for i in range(n_sparrows):
                     r1, r2 = np.random.choice(n_sparrows, 2, replace=False)
                     step = (pop[r1].astype(int) + pop[r2].astype(int)) // 2
                     pop[i] = np.clip(pop[i].astype(int) + step, 0, 1).astype(bool)

                     # Mutation
                     if np.random.uniform() < p_mutation:
                        pop[i][np.random.randint(X.shape[1])] = 1 - pop[i][np.random.randint(X.shape[1])]

                 # Evaluate fitness
                 for i in range(n_sparrows):
                     fitness[i] = fitness_function(X_train, X_test, y_train, y_test, pop[i].astype(bool))

                     # Update best solution and fitness
                     if fitness[i] > best_fitness:
                         best_solution = pop[i]
                         best_fitness = fitness[i]
                 # Print progress
                 print("Iteration:", it, "Best fitness:", best_fitness)

             return best_solution.astype(bool)

         # Assuming X_resampled and y_resampled are defined earlier
         X = X_resampled
         y = y_resampled


         # Split into train and test sets
         X_train, X_test, y_train, y_test = train_test_split(X_resampled.values, y_resampled.values, test_size=0.2, random_state=42)

         from sklearn.preprocessing import StandardScaler

         scaler = StandardScaler()

         scaled_X_train = scaler.fit_transform(X_train)
         scaled_X_test = scaler.transform(X_test)

         # Feature selection with sparrow search algorithe
         best_features = sparrow_search(scaled_X_train, y_train, scaled_X_test, y_test, n_sparrows=50,max_iterations=100, p_mutation=0.1)

         # Train and evaluate RF classifier with selected features
         rf = RandomForestClassifier(n_estimators=100, random_state=42)

         selected_features = np.nonzero(best_features)[0].tolist()
         rf.fit(scaled_X_train[:, selected_features], y_train)

         y_pred = rf.predict(scaled_X_test[:, selected_features])
         print("Accuracy:", accuracy_score(y_test, y_pred))
```

Fig.21 Code for Output of KNN Classifier with Sparrow Search Algorithm.



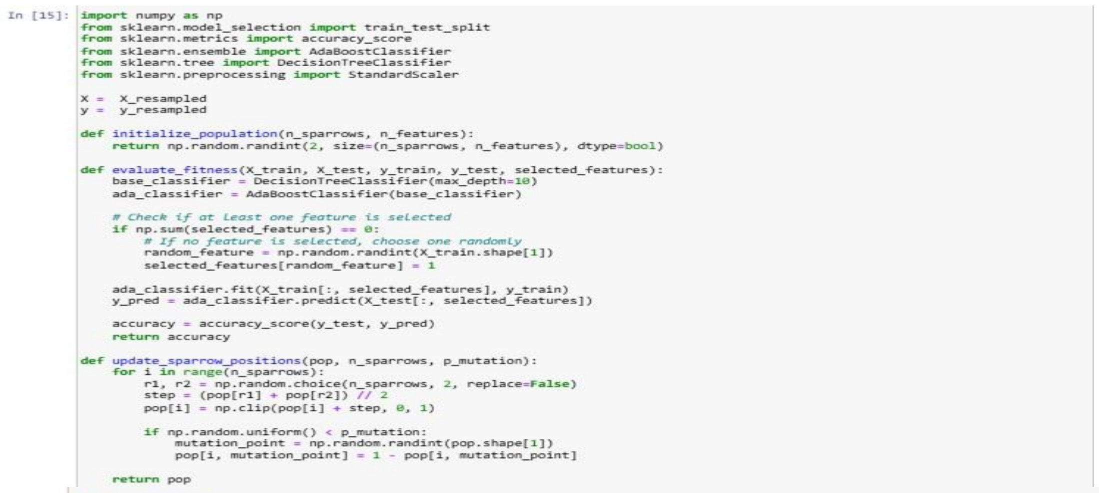
```
In [22]: import random
         import numpy as np
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score
         from sklearn.neighbors import KNeighborsClassifier
         from sklearn.preprocessing import StandardScaler

         def fitness_function(X_train, X_test, y_train, y_test, selected_features):
             knn = KNeighborsClassifier(n_neighbors=3)
             knn.fit(X_train[:, selected_features], y_train)
             y_pred = knn.predict(X_test[:, selected_features])
             return accuracy_score(y_test, y_pred)

         def sparrow_search(X_train, y_train, X_test, y_test, n_sparrows, max_iterations, pmutation):
             # Initialize population
             pop = np.zeros((n_sparrows, X_train.shape[1]))
             for i in range(n_sparrows):
                 pop[i] = np.random.randint(2, size=X_train.shape[1])

             # Evaluate fitness of initial population
             fitness = np.zeros(n_sparrows)
             for i in range(n_sparrows):
                 fitness[i] = fitness_function(X_train, X_test, y_train, y_test, pop[i].astype(bool))

             # Best solution and fitness
             best_solution = pop[np.argmax(fitness)]
             best_fitness = np.max(fitness)

             # Main Loop
             for it in range(max_iterations):
                 # Update sparrow positions
                 for i in range(n_sparrows):
                     r1, r2 = np.random.choice(n_sparrows, 2, replace=False)
                     step = (pop[r1].astype(int) + pop[r2].astype(int)) // 2
                     pop[i] = np.clip(pop[i].astype(int) + step, 0, 1).astype(bool)

                     if np.random.uniform() < pmutation:
                         mutation_point = np.random.randint(X_train.shape[1])
                         pop[i, mutation_point] = 1 - pop[i, mutation_point]

                 # Evaluate fitness
                 for i in range(n_sparrows):
                     fitness[i] = fitness_function(X_train, X_test, y_train, y_test, pop[i].astype(bool))

                     # Update best solution and fitness
                     if fitness[i] > best_fitness:
                         best_fitness = fitness[i]
                         best_solution = pop[i]

                 # Print progress
                 print("Iteration:", it, "Best fitness:", best_fitness)

             return best_solution.astype(bool)

         # Split into train and test sets
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # Standardize the data
         scaler = StandardScaler()
         scaled_X_train = scaler.fit_transform(X_train)
         scaled_X_test = scaler.transform(X_test)

         # Feature selection with sparrow search algorithm
         best_features = sparrow_search(scaled_X_train, y_train, scaled_X_test, y_test, n_sparrows=50, max_iterations=100, pmutation=0.1)

         # Train and evaluate KNN classifier with selected features
         knn = KNeighborsClassifier(n_neighbors=5)
         selected_features = np.nonzero(best_features)[0].tolist()
         knn.fit(scaled_X_train[:, selected_features], y_train)
         y_pred = knn.predict(scaled_X_test[:, selected_features])

         print("Accuracy:", accuracy_score(y_test, y_pred))
```

Fig.22 Code for Output of Decision Tree Classifier with Sparrow Search Algorithm.

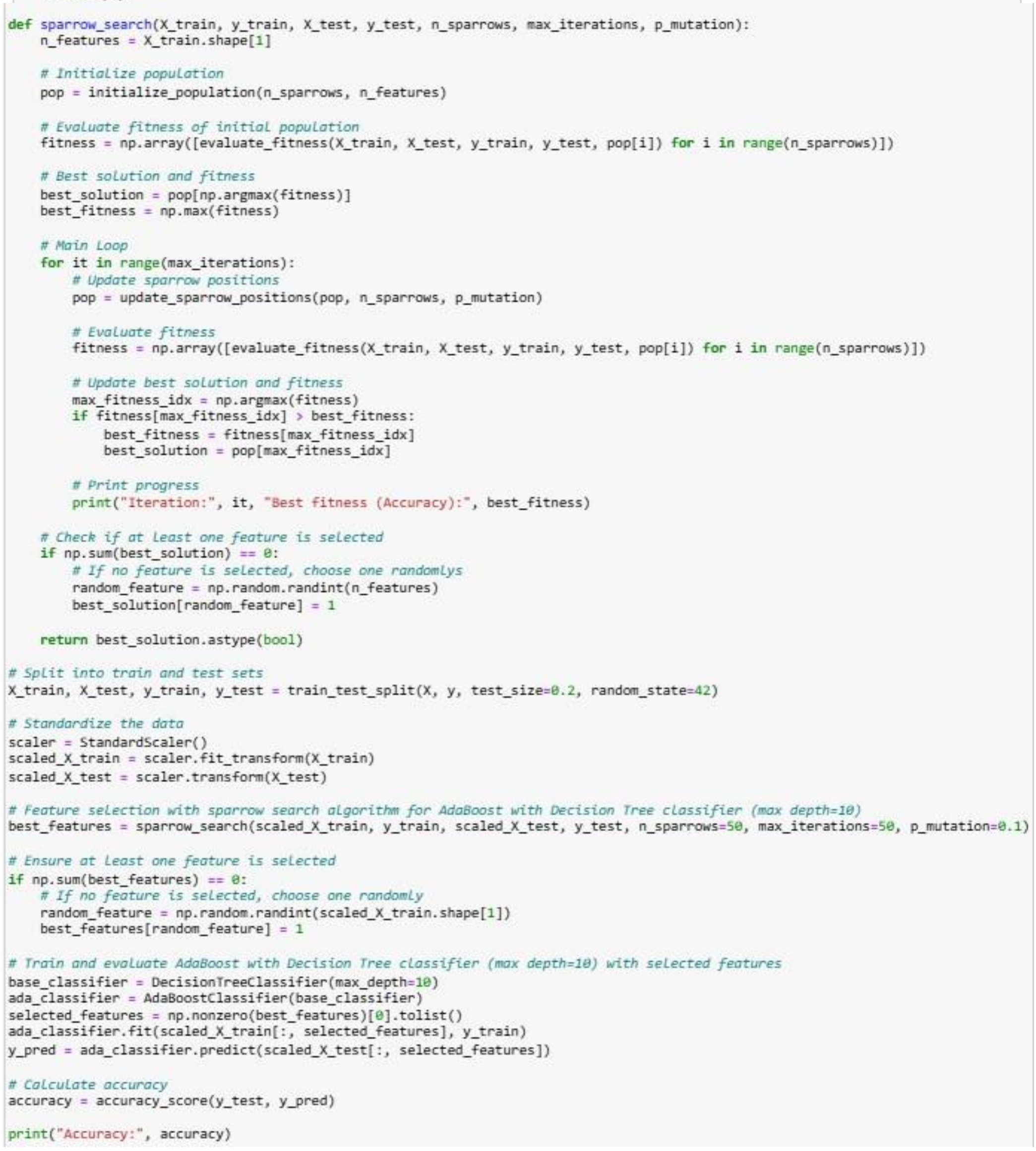


```
In [23]: import numpy as np
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.preprocessing import StandardScaler

         def fitness_function(X_train, X_test, y_train, y_test, selected_features):
             dt_classifier = DecisionTreeClassifier()
             dt_classifier.fit(X_train[:, selected_features], y_train)
             y_pred = dt_classifier.predict(X_test[:, selected_features])
             return accuracy_score(y_test, y_pred)

         def sparrow_search(X_train, y_train, X_test, y_test, n_sparrows, max_iterations, pmutation):
             # Initialize population
             pop = np.zeros((n_sparrows, X_train.shape[1]))
             for i in range(n_sparrows):
                 pop[i] = np.random.randint(2, size=X_train.shape[1])

             # Evaluate fitness of initial population
             fitness = np.zeros(n_sparrows)
             for i in range(n_sparrows):
                 fitness[i] = fitness_function(X_train, X_test, y_train, y_test, pop[i].astype(bool))

             # Best solution and fitness
             best_solution = pop[np.argmax(fitness)]
             best_fitness = np.max(fitness)

             # Main Loop
             for it in range(max_iterations):
                 # Update sparrow positions
                 for i in range(n_sparrows):
                     r1, r2 = np.random.choice(n_sparrows, 2, replace=False)
                     step = (pop[r1].astype(int) + pop[r2].astype(int)) // 2
                     pop[i] = np.clip(pop[i].astype(int) + step, 0, 1).astype(bool)

                     if np.random.uniform() < pmutation:
                         mutation_point = np.random.randint(X_train.shape[1])
                         pop[i, mutation_point] = 1 - pop[i, mutation_point]

                 # Evaluate fitness
                 for i in range(n_sparrows):
                     fitness[i] = fitness_function(X_train, X_test, y_train, y_test, pop[i].astype(bool))

                     # Update best solution and fitness
                     if fitness[i] > best_fitness:
                         best_fitness = fitness[i]
                         best_solution = pop[i]

                 # Print progress
                 print("Iteration:", it, "Best fitness:", best_fitness)

             return best_solution.astype(bool)


         # Split into train and test sets
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

         # Standardize the data
         scaler = StandardScaler()
         scaled_X_train = scaler.fit_transform(X_train)
         scaled_X_test = scaler.transform(X_test)

         # Feature selection with sparrow search algorithm for Decision Tree classifier
         best_features = sparrow_search(scaled_X_train, y_train, scaled_X_test, y_test, n_sparrows=50, max_iterations=100, pmutation=0.1)

         # Train and evaluate Decision Tree classifier with selected features
         dt_classifier = DecisionTreeClassifier()
         selected_features = np.nonzero(best_features)[0].tolist()
         dt_classifier.fit(scaled_X_train[:, selected_features], y_train)
         y_pred = dt_classifier.predict(scaled_X_test[:, selected_features])

         print("Accuracy:", accuracy_score(y_test, y_pred))
```

Fig.23 Code for Output of AdaBoost Classifier with Sparrow Search Algorithm

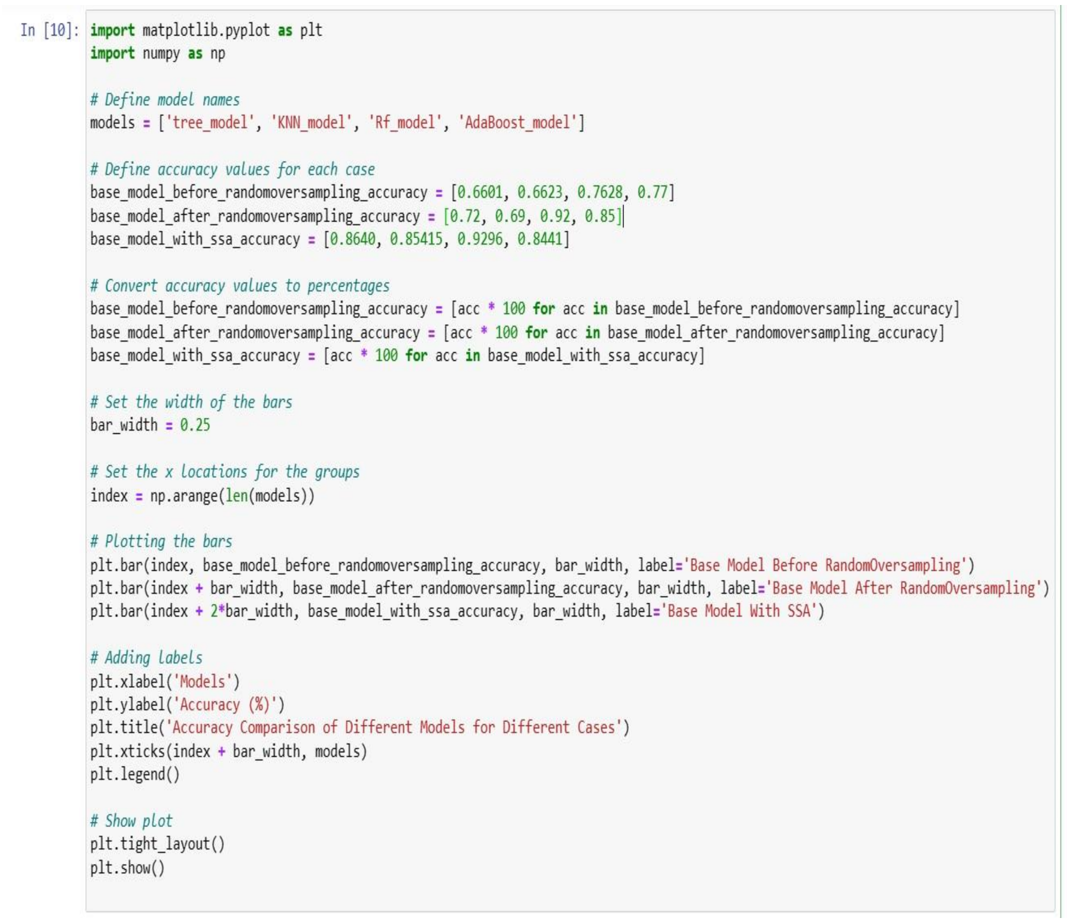


```
In [15]: import numpy as np
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import accuracy_score
         from sklearn.ensemble import AdaBoostClassifier
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.preprocessing import StandardScaler

         X =  X_resampled
         y =  y_resampled

         def initialize_population(n_sparrows, n_features):
             return np.random.randint(2, size=(n_sparrows, n_features), dtype=bool)

         def evaluate_fitness(X_train, X_test, y_train, y_test, selected_features):
             base_classifier = DecisionTreeClassifier(max_depth=10)
             ada_classifier = AdaBoostClassifier(base_classifier)

             # Check if at least one feature is selected
             if np.sum(selected_features) == 0:
                 # If no feature is selected, choose one randomly
                 random_feature = np.random.randint(X_train.shape[1])
                 selected_features[random_feature] = 1

             ada_classifier.fit(X_train[:, selected_features], y_train)
             y_pred = ada_classifier.predict(X_test[:, selected_features])

             accuracy = accuracy_score(y_test, y_pred)
             return accuracy

         def update_sparrow_positions(pop, n_sparrows, p_mutation):
             for i in range(n_sparrows):
                 r1, r2 = np.random.choice(n_sparrows, 2, replace=False)
                 step = (pop[r1] + pop[r2]) // 2
                 pop[i] = np.clip(pop[i] + step, 0, 1)

                 if np.random.uniform() < p_mutation:
                     mutation_point = np.random.randint(pop.shape[1])
                     pop[i, mutation_point] = 1 - pop[i, mutation_point]

             return pop
```



```
def sparrow_search(X_train, y_train, X_test, y_test, n_sparrows, max_iterations, p_mutation):
    n_features = X_train.shape[1]

    # Initialize population
    pop = initialize_population(n_sparrows, n_features)

    # Evaluate fitness of initial population
    fitness = np.array([evaluate_fitness(X_train, X_test, y_train, y_test, pop[i]) for i in range(n_sparrows)])

    # Best solution and fitness
    best_solution = pop[np.argmax(fitness)]
    best_fitness = np.max(fitness)

    # Main Loop
    for it in range(max_iterations):
        # Update sparrow positions
        pop = update_sparrow_positions(pop, n_sparrows, p_mutation)

        # Evaluate fitness
        fitness = np.array([evaluate_fitness(X_train, X_test, y_train, y_test, pop[i]) for i in range(n_sparrows)])

        # Update best solution and fitness
        max_fitness_idx = np.argmax(fitness)
        if fitness[max_fitness_idx] > best_fitness:
            best_fitness = fitness[max_fitness_idx]
            best_solution = pop[max_fitness_idx]

        # Print progress
        print("Iteration:", it, "Best fitness (Accuracy):", best_fitness)

    # Check if at least one feature is selected
    if np.sum(best_solution) == 0:
        # If no feature is selected, choose one randomlys
        random_feature = np.random.randint(n_features)
        best_solution[random_feature] = 1

    return best_solution.astype(bool)

# Split into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Standardize the data
scaler = StandardScaler()
scaled_X_train = scaler.fit_transform(X_train)
scaled_X_test = scaler.transform(X_test)

# Feature selection with sparrow search algorithm for AdaBoost with Decision Tree classifier (max depth=10)
best_features = sparrow_search(scaled_X_train, y_train, scaled_X_test, y_test, n_sparrows=50, max_iterations=50, p_mutation=0.1)

# Ensure at least one feature is selected
if np.sum(best_features) == 0:
    # If no feature is selected, choose one randomly
    random_feature = np.random.randint(scaled_X_train.shape[1])
    best_features[random_feature] = 1

# Train and evaluate AdaBoost with Decision Tree classifier (max depth=10) with selected features
base_classifier = DecisionTreeClassifier(max_depth=10)
ada_classifier = AdaBoostClassifier(base_classifier)
selected_features = np.nonzero(best_features)[0].tolist()
ada_classifier.fit(scaled_X_train[:, selected_features], y_train)
y_pred = ada_classifier.predict(scaled_X_test[:, selected_features])

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)

print("Accuracy:", accuracy)
```

Fig.24 Code for Comparison of machine learning model accuracies of three different cases



In [10]:
```python
import matplotlib.pyplot as plt
import numpy as np

# Define model names
models = ['tree_model', 'KNN_model', 'Rf_model', 'AdaBoost_model']

# Define accuracy values for each case
base_model_before_randomoversampling_accuracy = [0.6601, 0.6623, 0.7628, 0.77]
base_model_after_randomoversampling_accuracy = [0.72, 0.69, 0.92, 0.85]
base_model_with_ssa_accuracy = [0.8640, 0.85415, 0.9296, 0.8441]

# Convert accuracy values to percentages
base_model_before_randomoversampling_accuracy = [acc * 100 for acc in base_model_before_randomoversampling_accuracy]
base_model_after_randomoversampling_accuracy = [acc * 100 for acc in base_model_after_randomoversampling_accuracy]
base_model_with_ssa_accuracy = [acc * 100 for acc in base_model_with_ssa_accuracy]

# Set the width of the bars
bar_width = 0.25

# Set the x locations for the groups
index = np.arange(len(models))

# Plotting the bars
plt.bar(index, base_model_before_randomoversampling_accuracy, bar_width, label='Base Model Before RandomOversampling')
plt.bar(index + bar_width, base_model_after_randomoversampling_accuracy, bar_width, label='Base Model After RandomOversampling')
plt.bar(index + 2*bar_width, base_model_with_ssa_accuracy, bar_width, label='Base Model With SSA')

# Adding labels
plt.xlabel('Models')
plt.ylabel('Accuracy (%)')
plt.title('Accuracy Comparison of Different Models for Different Cases')
plt.xticks(index + bar_width, models)
plt.legend()

# Show plot
plt.tight_layout()
plt.show()
```

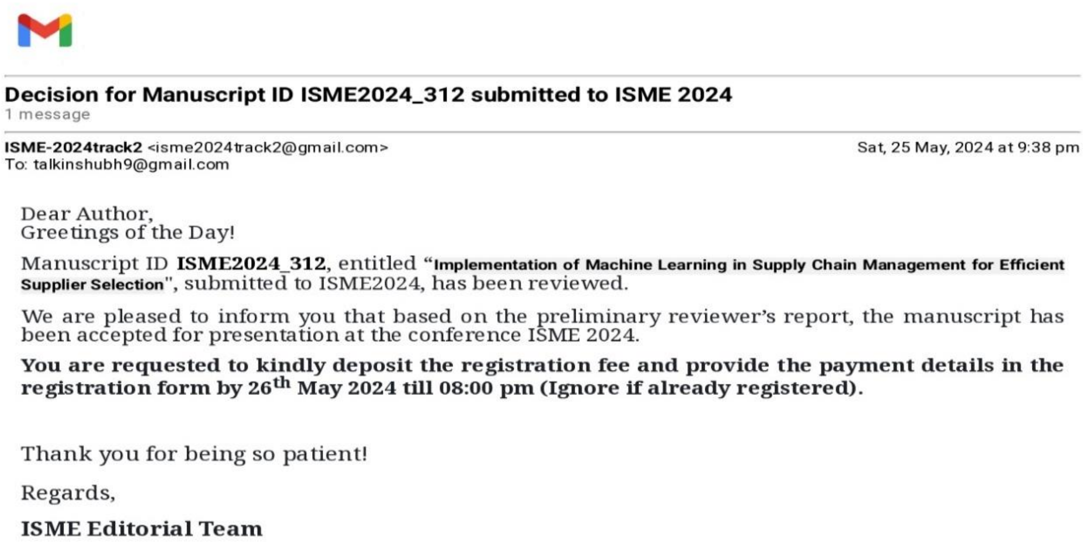
DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-42

LIST OF PUBLICATIONS AND THEIR PROOF

Title	Conference
Implementation of Machine Learning in Supply Chain Management for Efficient Supplier Selection.	22nd ISME International Conference on "Recent Advances in Mechanical Engineering for Sustainable Development"



Decision for Manuscript ID ISME2024_312 submitted to ISME 2024
1 message

ISME-2024track2 <isme2024track2@gmail.com> Sat, 25 May, 2024 at 9:38 pm
To: talkinshubh9@gmail.com

Dear Author,
Greetings of the Day!

Manuscript ID **ISME2024_312**, entitled "**Implementation of Machine Learning in Supply Chain Management for Efficient Supplier Selection**", submitted to ISME2024, has been reviewed.

We are pleased to inform you that based on the preliminary reviewer's report, the manuscript has been accepted for presentation at the conference ISME 2024.

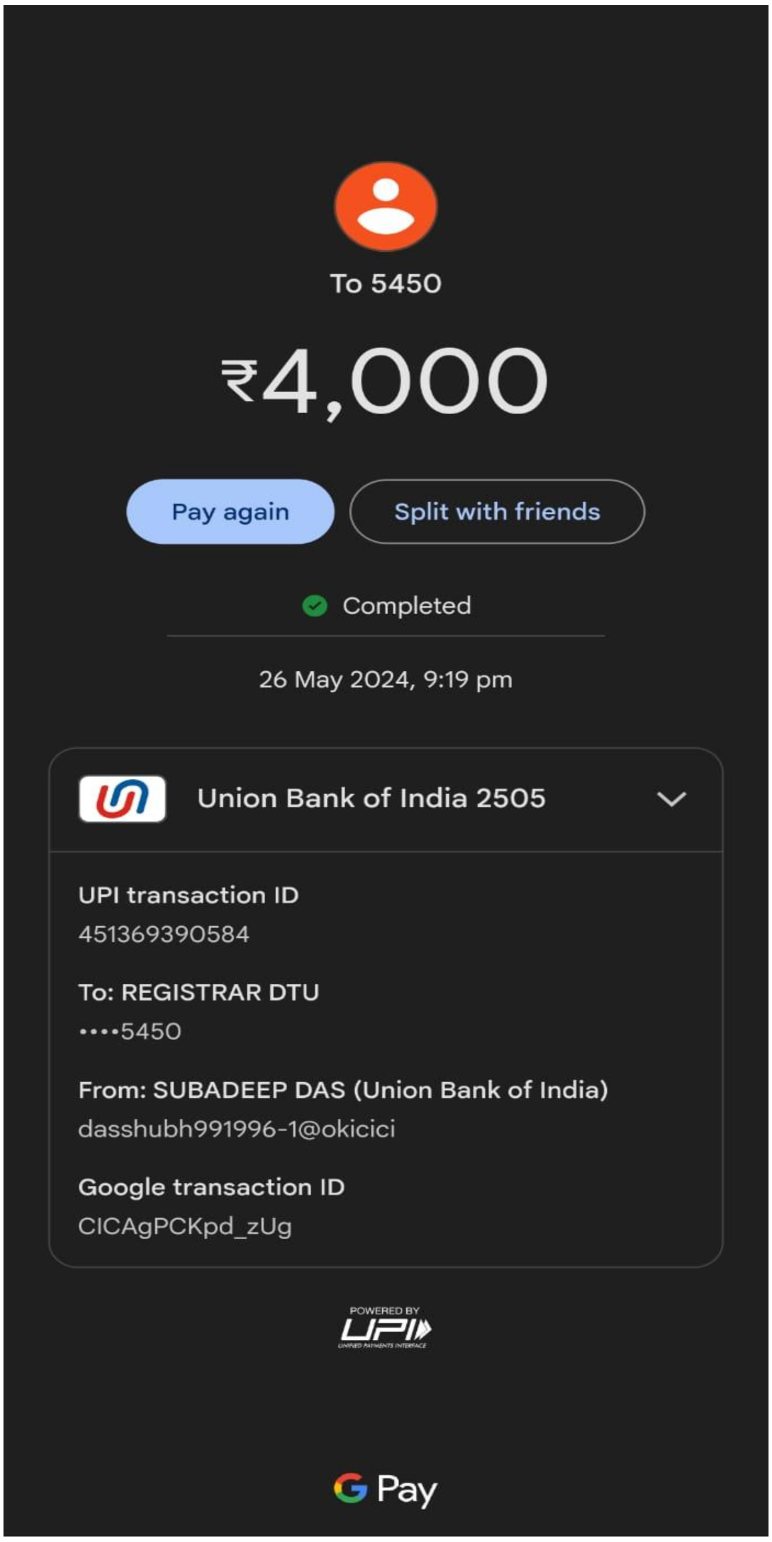
You are requested to kindly deposit the registration fee and provide the payment details in the registration form by 26th May 2024 till 08:00 pm (Ignore if already registered).

Thank you for being so patient!

Regards,

ISME Editorial Team

22nd ISME International Conference on “Recent Advances in Mechanical Engineering for Sustainable Development” Fee Recipte.


To 5450

₹4,000

Completed

26 May 2024, 9:19 pm

Union Bank of India 2505

UPI transaction ID
451369390584

To: REGISTRAR DTU
••••5450

From: SUBADEEP DAS (Union Bank of India)
dasshubh991996-1@okicici

Google transaction ID
CICAgPCKpd_zUg

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

PLAGIARISM VERIFICATION

Title of the Thesis Application of machine learning in supply chain management – leveraging supervised machine learning for efficient supplier selection.

Total Pages: Name of the scholar: Subadeep Das

Supervisor(s): Dr. S. K. Garg

Department: Department of Mechanical Engineering

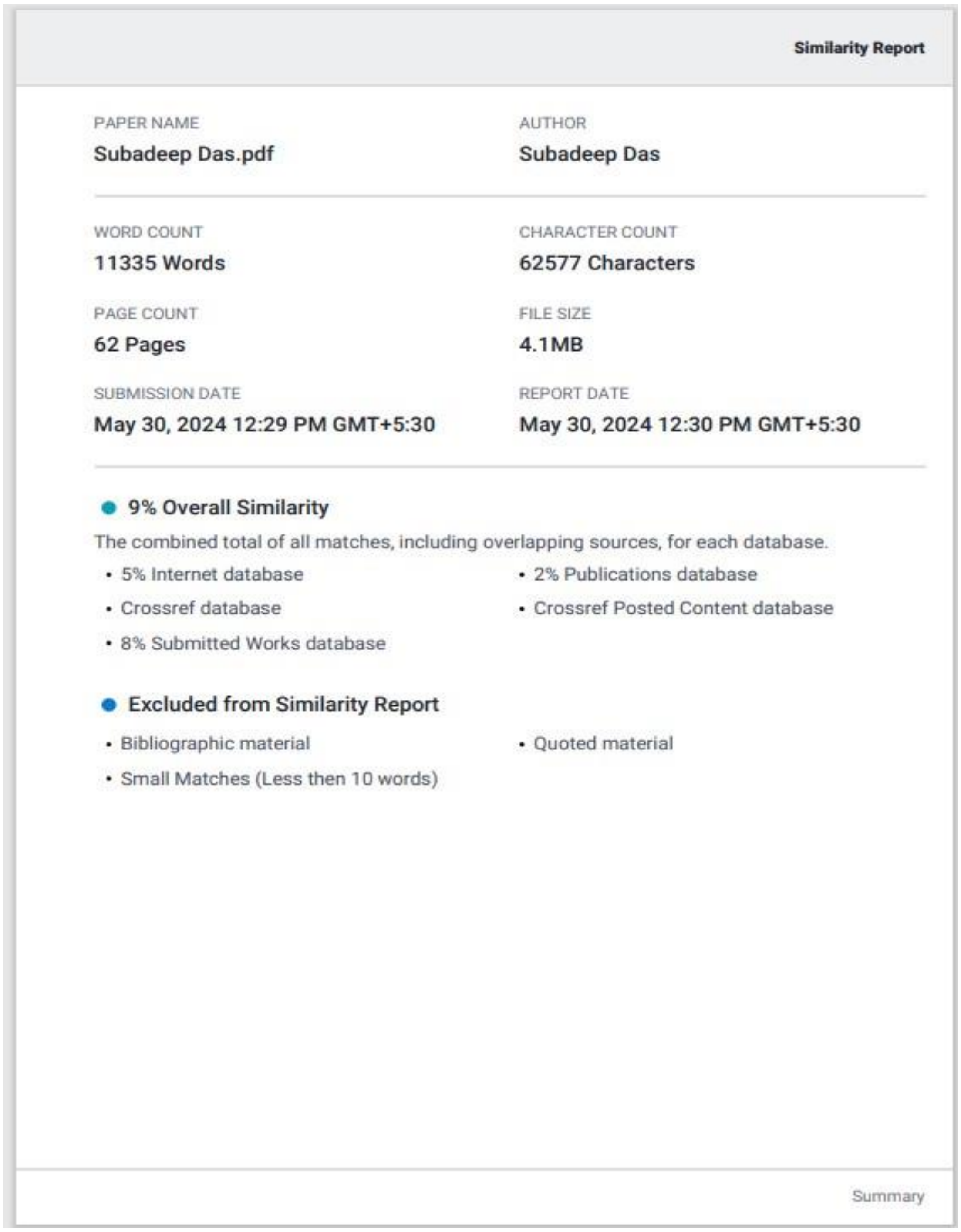
This is the report that the above thesis was scanned for similarity detection. The process and outcome are given below:

Software used: Turnitin Similarity Index: 9% Total Word Count: 11335

Date: 30 May 2024

Candidate's Signature **Signature of Supervisor(s)**

PLAGIARISM REPORT



Similarity Report

PAPER NAME
Subadeep Das.pdf

AUTHOR
Subadeep Das

WORD COUNT
11335 Words

CHARACTER COUNT
62577 Characters

PAGE COUNT
62 Pages

FILE SIZE
4.1MB

SUBMISSION DATE
May 30, 2024 12:29 PM GMT+5:30

REPORT DATE
May 30, 2024 12:30 PM GMT+5:30

● **9% Overall Similarity**

The combined total of all matches, including overlapping sources, for each database.

- 5% Internet database
- 2% Publications database
- Crossref database
- Crossref Posted Content database
- 8% Submitted Works database

● **Excluded from Similarity Report**

- Bibliographic material
- Quoted material
- Small Matches (Less then 10 words)

Summary

BRIEF PROFILE

My name is Subadeep das, I am from Balasore Odisha. I have completed my Bachelor of Technology in Mechanical Engineering from the Institute of Engineering and Technology, Mahatma Jyotiba Phule Rohilkhand University, Bareilly, Uttar Pradesh with 75.6 %. At present, I am pursuing Master of Technology in Industrial Engineering and Management from Delhi Technological University, New Delhi. I am currently working on a project topic "Application of machine learning in supply chain management – leveraging supervised machine learning for efficient supplier selection".

Software Skills

Power Bi, Tableau, Python, SQL, MS Excel, QM

Position of Responsibility

Placement Coordinator at Delhi Technological University, New Delhi (2023-2024)