MULTI-OBJECTIVE DYNAMIC HYBRID FLOW SHOP SCHEDULING USING MACHINE LEARNING APPROACH

Saiara Samira Sajid

Student ID: 0417082016

Session: April 2017

DEPARTMENT OF INDUSTRIAL AND PRODUCTION ENGINEERING
BANGLADESH UNIVERSITY OF ENGINEERING & TECHNOLOGY
DHAKA-1000, BANGLADESH

MULTI-OBJECTIVE DYNAMIC HYBRID FLOW SHOP SCHEDULING USING MACHINE LEARNING APPROACH

By
Saiara Samira Sajid

A Thesis Submitted to the
Department of Industrial and Production Engineering
BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY
In Partial Fulfillment of the Requirements for the Degree of
MASTER OF SCIENCE IN INDUSTRIAL AND PRODUCTION ENGINEERING

DEPARTMENT OF INDUSTRIAL AND PRODUCTION ENGINEERING
BANGLADESH UNIVERSITY OF ENGINEERING & TECHNOLOGY
DHAKA-1000, BANGLADESH

December 2019

CERTIFICATE OF APPROVAL

The thesis titled **"Multi-objective Dynamic Hybrid Flow Shop Scheduling Using Machine Learning Approach"** submitted by Saiara Samira Sajid, Student No: 0417082016, Session-April 2017, has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Industrial & Production Engineering on December 29, 2019.

BOARD OF EXAMINERS

Dr. Ferdous Sarwar
Associate Professor
Department of IPE, BUET, Dhaka.

Chairman
(Supervisor)

Dr. Nikhil Ranjan Dhar
Professor & Head
Department of IPE, BUET, Dhaka.

Member
(Ex-Officio)

Dr. A. K. M. Masud
Professor
Department of IPE, BUET, Dhaka.

Member

Dr. Abdullahil Azeem
Professor
Department of IPE, BUET, Dhaka.

Member

Dr. Mohammed Forhad Uddin
Professor
Department of Mathematics, BUET, Dhaka.

Member
(External)

CANDIDATE’S DECLARATION

It is hereby declared that this thesis or any part of it has not been submitted elsewhere for the award of any degree or diploma.

____

Saiara Samira Sajid

To the Almighty

To my family

ACKNOWLEDGEMENT

All credits go to the Almighty, for His boundless grace in the successful completion of this thesis.

At the very beginning, the author expresses her deepest gratitude to the supervisor of this thesis- Dr. Ferdous Sarwar, Associate Professor, Department of Industrial & Production Engineering (IPE), Bangladesh University of Engineering and Technology (BUET). Without his affectionate guidance, valuable suggestions and inspirations throughout this work, this research would not have been possible.

Moreover, the author is thankful to the board members- Dr. Nikhil Ranjan Dhar, Professor & Head, Department of IPE, BUET, Dr. A. K. M. Masud, Professor, Department of IPE, BUET, Dr. Abdullahil Azeem, Professor, Department of IPE, BUET, and Dr. Mohammed Forhad Uddin, Professor, Department of Mathematics, BUET, for their constructive remarks and kind evaluations of this research.

Finally, the author would like to thank all of her colleagues, friends, and well-wishers for their co-operation and motivation to complete the thesis work. The author would also like to extend her thanks to her parents whose continuous inspiration, sacrifice and support encouraged her to complete the thesis successfully.

ABSTRACT

Keeping pace with the rapidly changing production system is a challenge, where each day new technology is being invented and the market is becoming more competitive. One effective way to sustain is to have a production planning system that can react to sudden changes in the production phase and also is capable of finding an optimum solution among production challenges. This thesis aims to propose a semi-automated dynamic hybrid flow shop scheduling model that can provide an optimum production schedule considering capacity limitations, operators learning effect, machine break-down conditions, etc. In order to make the production scheduling semi-automated, a machine learning algorithm, Support Vector Machine (SVM) is used to formulate a job classification model that can classify jobs based on their priority level. Furthermore, a scheduling model is developed that utilizes each job's corresponding priority level information. The model aims to address three objectives: minimization of make-span, minimization of tardiness and maximization of efficiency. In this work maximization of efficiency is calculated in terms of machine idle time. To make this model applicable to real-life production challenges, uncertainties related to processing time and machine break-down are considered. Finally, a meta-heuristic algorithm, Particle Swarm Optimization is used to find the optimum schedule. The job classification approach used in this thesis has not been explored by other researchers so far. This proposed method has proved its efficacy in depicting real-life production challenges and providing an optimum result.

TABLE OF CONTENTS

Contents **Page No**

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACO	Ant Colony Optimization
AIS	Artificial Immune System
DPSO	Discrete Particle Swarm Optimization
FSP	Flow-shop Scheduling Problem
GA	Genetic Algorithm
HFS	Hybrid Flow Shop
HGA	Hybrid Genetic Algorithm
HQGA	Hybrid Quantum-inspired Genetic Algorithm
KKT	Karush-Kuhn-Tucker
MILP	Mixed Integer Linear Programming
MOACSA	Multi-objective Ant colony System Algorithm
NEH	Nawaz-Enscore-Ham
NP	Nondeterministic Polynomial
PCA	Principal Component Analysis
PSO	Particle Swarm Optimization
SA	Simulated Annealing
SAM	Standard Allocated Minutes
SVM	Support Vector Machine
TS	Tabu Search
VND	Variable Neighborhood Descent

CHAPTER 1: INTRODUCTION

1.1 Background of the Study

To improve organizational effectiveness with customer satisfaction, it requires some important decision-making processes. Among then the decision of scheduling has a significant importance level. Scheduling mainly deals with resource (i.e. machines) allocation considering some performance criteria such as make-span, tardiness, flow-time and so on. It is a process by which limited resources are assigned to sequential or parallel activities over a given time frame. Intelligent scheduling methods are needed to ensure optimum allocation of scarce resources within limited execution time.

Manufacturing industries are facing different challenges such as customized product demand, shorter product life cycles, continuously changing market demand, and global competition. Without improving the performance of the production scheduling systems under increasing market fluctuations and internal uncertainties in the manufacturing process (e.g. machine breakdown, tool failure, and change of processing times) it will be difficult for manufacturing companies to survive in the competitive environment. There is also continuous advancement in technology advancement and manufacturing industries need to keep pace with that. So, scheduling should be done in such a way that ensures efficient use of available resources. The efficient use of resources can be ensured by optimum scheduling.

An optimum schedule can be obtained by optimizing various performance measures. One measure can be the minimization of the completion time of the last job which is termed as make-span and another can be the minimization of the number of jobs completed after their respective due dates or minimizing the number of days by which a job is delayed from its due date. It is also important to choose the objectives properly so that the optimized schedule can meet the requirements properly.

The performance of an organization can be measured by two dimensions, those are technological and organizational dimensions. The technological dimension focuses on satisfying the required quality at a lower cost. These requirements along with the rapid technological advancement of products lead companies to opt for mass production. Production cycle times, expected delivery date, inventory, and work in process management, etc are addressed by the organizational dimension. Therefore, companies need to have powerful production planning and control methods.

By specifying the resource configuration and the nature of the tasks scheduling models are can be categorized. For instance, when a job needs to be machined by one machine, it is likely to be processed on a single stage. Whereas, if a job requires multiple operations or machines it will be multi-stage scheduling which is called flexible scheduling or hybrid scheduling. In static scheduling, at the beginning of the scheduling process, all jobs to be scheduled are available, whereas in dynamic scheduling the set of jobs to be processed is continuously changing. Generally, it is easy to deal with static scheduling rather than dynamic scheduling. Based upon the certainty of parameter scheduling can be of two types, one is deterministic and the other one is stochastic, when all parameters are known with certainty, the scheduling model is called deterministic. On the other hand, if uncertainty exists with the scheduling parameters it is stochastic scheduling. In this work, stochastic scheduling is being addressed.

Based upon the application there are several types of scheduling such as- single machine scheduling, job shop scheduling, flow-shop scheduling, flexible or hybrid job shop scheduling. flexible or hybrid flow-shop scheduling. In job shop scheduling problem a set of $\boldsymbol{n}$ jobs to be processed by a set of machines, each job is processed on machines in a predefined order. In this case, the objective is to find an optimal ordering of all the jobs with respect to their varied routing requirements through the machines. The job flow through machines is multidirectional flow. The flow-shop scheduling problem (FSP) consists of $\boldsymbol{m}$ machines and $\boldsymbol{n}$ jobs, where the objective is to sequence $\boldsymbol{n}$ jobs on $\boldsymbol{m}$ machines in an optimum way. All machines are situated in a defined series and all jobs need to be processed on each of the machines. The routing of the jobs through the different machines are unidirectional flow. A job can move forward in the queue when the preceding operation is completed. A hybrid flow shop (HFS) consists of a series of production stages with several parallel machines, where a set of operations are done on jobs in a predefined order **[Linn & Zhang, 1999].** Some examples of HFS are an automotive assembly line, semiconductor production, textile production line, etc

Shop scheduling problems can be represented by the combinatorial optimization class of problems. In combinational optimization problems, an optimal solution is searched in a finite set of potential solutions. At the very beginning, solving scheduling problems were done by exact methods that guarantee to find an optimal solution. Later on, researchers have identified job scheduling problems as NP-hard (non-deterministic-time hard) problems. That means it can be solved by exact methods. Then researchers started to use various hybrid algorithms as well as meta-heuristic algorithms to find a local solution in spite of the global one **[Framinan, Gupta & Leisten, 2004]**. In this work, the PSO algorithm has been used to find the optimum scheduling.

In earlier research works some specific objectives such as reducing tardiness and maintaining stability **[Rahmani & Ramezanian, 2016]** or reduce mean flow time (Shahvari & Logendran, 2018) were considered. A number of works **[Novak, Sucha & Hanzalek, 2019;Jamili, 2019]** have been done considering uncertainty in processing time while scheduling, under some special considerations of a single machine, workers rest time, etc. Whereas, this work will consider uncertainties in processing time for multiple machines, machine break-down as well as focus on multiple objectives such as tardiness, mean flow-time.

In the last decade, the use of artificial intelligence had grabbed the attention of researchers in various fields such as- image processing, data mining, even in medical sectors. Using artificial intelligence, with the help of historical data which is termed as training data set various prediction models and decision-making models can be developed. These models can imitate the human decision-making process. So, considering the computational complexity of scheduling problems due to its NP-hard nature, the use of AI in this sector had proved its efficacy. Though a very small amount of work has been done using a machine learning algorithm in scheduling. Mostly scheduling models are focus on job shop type scheduling and finding a suitable dispatch rule. However, this thesis work focuses on finding an optimum schedule for hybrid flow shop.

This research has used Support Vector Machine (SVM) an artificial intelligence (AI) algorithm to classify the jobs as per their priority level along with a heuristic algorithm to solve it. In previous works, SVM has proved its efficacy in assigning dispatching rules to jobs in flexible manufacturing systems **[Priore et al, 2006; Priore et al, 2018; Y. H. Liu, Huang & Lin, 2005].** Both AI and a heuristic algorithm have been used to make this model adaptive to real-life changes with faster computation capacity. Previous works did not focus on both prioritizing the job and considering the uncertainties related to the parameters together. This work combines these two aspects which makes this model more beneficial to manufacturing industries.

To sustain in the competitive global market, an optimal production sequence is a prerequisite that will help to achieve the planned production quantity. This work develops a scheduling model, that provides an optimum job sequencing based on each job's priority level and then provides an optimum job assignment to machines. This model can be used in the manufacturing industry to obtain the optimized schedule which corresponds to minimized make-span, minimized tardiness and maximized efficiency. The model segregates the jobs based on their priorities which will help them to identify important jobs. Furthermore, it provides optimum scheduling for a set of jobs in a hybrid flow shop considering uncertainties and multiple objectives. This scheduling model will help manufacturing industries to make the production planning decisions in a semi-automated manner.

1.2 Objectives with Specific Aims

Hybrid flow shop problem is known to be an NP-hard problem and most research works concentrate on developing a heuristic procedure, to provide a better result **[Framinan, Gupta & Leisten, 2004].** However, little attention had been given to develop a common framework that resembles real-life production situations. Furthermore, very little work has been done by combining machine learning concept with meta-heuristics. The uncertainty considerations were seen limited to small production scale, whereas, in real life, hybrid flow shop is applicable in mass production. This work aims to bridge these research gaps and develop a hybrid flow show shop scheduling model, that can replicate real-life scenarios.

The specific objectives of this research are:

- To develop a multi-objective hybrid flow shop scheduling model considering uncertainty on production which can resemble real-life production scheduling.
- To incorporate a machine learning algorithm (SVM) to classify jobs based on their priority level which will consider qualitative information associated with the jobs.

1.3 Outline of Methodology

a. Different assessment criteria for prioritizing among a set of jobs has been identified from expert opinion.

b. The criteria identified in (i) have been used as features of the training data set of SVM (Support Vector Machine). This trained multiclass classifier SVM model has been used to classify future jobs. In this work, jobs have been classified among three classes. Each class will have a different priority level.

c. Based on this priority level, a priority index has been assigned to each job. The weightage value has been determined by management.

d. A multi-objective job scheduling model has been formulated to minimize tardiness (delay between compilation time and due date of the respective job), minimize mean flow time (amount of time spent by a job in the shop) and maximize overall efficiency. This model has used the priority index value determined in (iii) to schedule a set of jobs with the highest priority index in descending order.

e. In order to validate the proposed model, data have been collected from the manufacturing industry. Data set consisted of processing time, delivery date, material availability status and sequence of operation of each job, the capacity of the industry, etc.

f. The multi-objective scheduling model has been solved using PSO (Particle Swarm Optimization), a meta-heuristic algorithm, to obtain the optimum value of the objective functions.

g. A sensitivity analysis of the model has been performed to examine the stability of the scheduling model.

1.4 Organization of the Thesis

This thesis has been chalked out into seven chapters along with references and appendices

Chaprter 1:	Introduction	The background of the research work, methodologies to develop a scheduling model for dynamic hybrid flow shop is mentioned in this chapter.
Chaprter 2:	Literature Review	It includes the related literature on hybrid flow shop scheduling, uncertainty is in flow shop, use of PSO and SVM in flow shop scheduling.
Chaprter 3:	Theoretical Framework	The theory for SVM and how the basic SVM has emerged into the current form is discussed in this chapter. As well as, the basic theory for the PSO algorithm is included in this chapter.
Chaprter 4:	Model Formulation	It describes the formulation of the model for a hybrid flow shop scheduling which specifies the objective functions and constraints associated with the model.
Chaprter 5:	Model Implementation	In this chapter, the formulated model and SVM model are implemented in the manufacturing industry.
Chaprter 6:	Result Analysis	This chapter discusses the result obtained after the model implementation. It also includes a sensitivity analysis of the model in different scenarios.
Chaprter 7:	Conclusion and Future Works	It contains the conclusion of this work and mentions the scope of future research associated with this work.

CHAPTER 2: LITERATURE REVIEW

2.1 Hybrid Flow Shop

A flow shop is a scheduling approach where, *m* machines are set in a series and each job has to be processed on each of the machines, *m*, following the same route. A new job in the queue can be processed when the processing of an earlier job is done. Whereas, a flexible or hybrid flow shop is a generalization of the flow shop and parallel machine environments. In a hybrid flow shop instead of m machines in series, there are c stages in series with at each stage a number of identical machines in parallel. Though some stages may have only one machine, at least one stage must have multiple machines. The flow of jobs through the shop is unidirectional. Each job is processed by one machine in each stage and it must go through one or more stages **[Linn & Zhang, 1999]**. A hybrid flow shop is suitable for mass production of any product. The performance of a flow shop can be measured in terms of make-span(i. e. the completion time of the last job to leave the system), tardiness (i.e. the difference between the required delivery date and actual production completion date), etc. Most research works had considered the above-mentioned measures to determine the performance of a flow shop or hybrid flow shop.

Wang & Xia, **[2005]** considered the learning effect in the flow shop scheduling problem, where an optimal solution for two-machine flow shop scheduling was obtained with minimized make-span considering the learning effect.

Availability constraints were considered in the work of Aggoune**, [2003].** In their problem, they considered the fact that a machine may not be continuously available due to preventive maintenance activity. They used GA and Tabu search methods to solve the make-span minimization problem.

The main objective of Luo et al, **[2013]** was to improve production efficiency, which they related to the reduction of energy consumption. It became important with the advent of green manufacturing. They solved the optimization problem by using a multi-objective ant colony algorithm.

Precedence constraints, several time lags(i.e. either machine setup time, machine removal time or transportation time), and due-dates were considered by Botta-Genoulaz, **[2000]** while solving a single objective scheduling model to minimize maximum lateness.

A realistic situation was considered by Ruiz, Şerifoğlu & Urlings, **[2008],** where per machine sequence-dependent setup times were considered anticipatory and non-anticipatory along with machine lags, release dates for machines, machine eligibility and precedence relationships among jobs were also considered. In that work, the optimization criterion was the minimization of make-span.

Production planning and control function in a textile plant can be modeled by using the concept of hybrid flow-shop. A scheduling problem can be divided into two parts, one is 'capacity loading function' and the other one is 'scheduling function', both were considered for a given planning horizon. Also, manufacturing environments like food processing, ceramic tile manufacturing, and several others can be modeled by a hybrid flow shop scheduling system Ruiz, Şerifoğlu & Urlings, **[2008].**

2.2 Algorithms to Solve Hybrid Flow Shop Scheduling

In 1954 Jonshon first presented a paper on flow shop which can be considered as a pioneer in the research field of flow shop**.** The proposed algorithm is known by Jonshon's rule, which is a simple technique to sequence a set of jobs optimally **[Gupta & Stafford, 2006].** A two-stage hybrid flow shop model was developed to minimize make-span was one of the very first contributions in this area, a heuristic approach was used for the solution. Hybrid flow shop scheduling being an NP-Hard problem, it is difficult to get a direct solution. For this purpose, meta-heuristic approaches need to be used to find the near-optimal solution Sayadi, Ramezanian & Ghaffari-Nasab**, [2010].** From literature, it has been seen that, for complex situations, heuristic and meta-heuristic methods and hybrid procedures are much more useful than other exact methods **[Sun et al, 2011]** .

2.2.1 Single Objective Hybrid Flow Shop Scheduling

Flow shop scheduling problem being an NP-hard problem, many research works had been conducted to find an efficient solution method. Osman & Potts, **[1989]** proposed Simulated annealing as a heuristic to obtain approximate solutions, where the objective was to minimize the maximum completion time or minimize the make-span. Murata, Ishibuchi & Tanaka, **[1996]** compared the genetic algorithm with other search algorithms such as local search, taboo search and simulated annealing where the objective was to minimize make-span and concluded that computer simulations had better performance. However, with the increase of complexity of the problem performance of computer simulations may degrade. Ruiz & Stützle, **[2007]** proposed a new iterated algorithm that is applied to two phases iteratively. In the first phase, jobs were eliminated from the incumbent solution and in the second phase, eliminated jobs were reinserted into the sequence by using Nawaz-Enscore-Ham (NEH) heuristic. This algorithm was named as the greedy algorithm and the two stages were named destruction and construction respectively. Ruiz, Maroto & Alcaraz, **[2006]** used robust GA with new genetic operators such as hybridization with local search and efficient population initialization. Design of Experiments was used for a complete evaluation of parameters. The considered optimization criterion was the minimization of make-span. Rajendran & Ziegler, **[2004]** used the ant colony algorithm to solve a single objective flow shop scheduling problem, where the objective was to minimize total make-span. An improved cuckoo search algorithm was developed by Marichelvam, Prabaharan & Yang**, [2014]** to solve hybrid flow shop scheduling problems with a single objective of minimizing the make-span.

Artificial immune system (AIS), a new approach was proposed by Engin & Döyen, **[2004]** to solve an NP-hard, two-stage hybrid flow shop problem with a single objective of minimizing the make-span. Sayadi et al., **[2010]** used discrete firefly meta-heuristics to solve a permutational flow shop problem, where the objective was to minimize the make-span. An effective hybrid genetic algorithm (HGA) was implemented to solve a permutation flow shop scheduling with limited buffers, where the objective was to minimize total completion time (or make-span).

2.2.2 Multi-objective Hybrid Flow Shop Scheduling

To measure the performance of a hybrid flow shop it requires developing a multi-objective optimization problem. To handle these multi-objective optimization problems Ishibuchi & Murata, **[1998]** had used genetic algorithm to a two-objective function problem, where the result was in a Pareto front and two objectives: to minimize the make-span and to minimize the total tardiness. A hybrid algorithm was proposed by Ishibuchi & Murata, **[1998]** where local search procedure was applied for each solution, that was generated using genetic operations. In this approach, the local search procedure was modified as only a small number of neighborhood solutions were examined. The fitness function for this approach used the weighted sum of multiple objectives.

A hybrid quantum-inspired genetic algorithm (HQGA) for the multi-objective flow shop scheduling problem (FSSP) was proposed by Li & Wang, **[2007**], where a randomly weighted linear-sum function was used.

Ant colony optimization (ACO) was used by Yagmahan & Yenisey, **[2008]** to solve a multi-objective flow shop scheduling problem, where the objectives were to minimize the make-span, total flow time and total machine idle time. In that work, total machine idle time was defined by the difference between make-span and flow time. Multi-objective ant colony system algorithm (MOACSA) was proposed, which combined ant colony optimization approach and a local search strategy in order to solve a multi-objective flow shop scheduling problem with both objectives of make-span and total flowtime.

A discrete firefly algorithm was extended by Marichelvam, Prabaharan & Yang**, [2014]** to solve hybrid flow shop scheduling problems with two objectives, which were minimization of make-span and mean flow time

2.2.3 Use of Particle Swarm Optimization in Hybrid Flow Shop Scheduling

Flow shop scheduling problem is one of the hardest combinatorial optimization problems **[Ercan, 2008b] .**To solve this, a similar particle swarm optimization algorithm (SPSOA) was used to solve a scheduling problem of minimizing make-span.

B. Liu, Wang & Jin, **[2007]** had proposed a modified PSO which is PSOMA. The model had three modifications, use of ranked-order value rule to convert continuous particle positions values to job permutation; use of Nawaz-Enscore-Ham (NEH) and finally used simulated annealing (SA) to avoid premature convergence. Similar to the modified PSO, B. Liu, Wang & Jin**, [2008]** proposed HPSO for the permutation flow shop scheduling problem which had limited buffers between consecutive machines to make-span. No-wait flow shop scheduling can also be solved by the proposed hybrid PSO **[B. Liu, Wang & Jin, 2007a].** Another modified PSO was proposed by Pan, Fatih Tasgetiren & Liang, **[2008]** where the particle and velocity were redefined and the proposed modification was able to provide a better result with total flowtime criterion.

Pan et al., **[2008]** presented a discrete particle swarm optimization (DPSO) algorithm to solve the no-wait flow shop scheduling problem with two criteria: make-span and total flowtime. The main contribution of the work was it presented particles as discrete job permutations. In addition to this, DPSO is hybridized with the variable neighborhood descent (VND) algorithm. Tang & Wang**, [2010]** used modified PSO along with greedy method to minimize total weighted completion time, where they used a job permutation concept.

Li, Wang & Liu**, [2008]** have employed a parallel evolution mechanism in PSO to solve multi-objective hybrid flow shop scheduling problem. In their work, they used ranked-order value for initialization, Nawaz-Enscore-Ham method to modify the local search approach. Kuo et al., **[2009]** proposed a novel hybrid flow shop algorithm, that combined random-key encoding scheme, individual enhancement scheme, and particle swarm optimization (PSO) altogether**.** Choong, Phon-Amnuaisuk & Alias, **[2011]** combined particle swarm optimization (PSO) with simulated annealing (SA) and tabu search (TS), to develop a hybrid model and found that memetic techniques produced improved solutions over conventional methods with faster convergence. Similarly, Liao, Tjandradjaja & Chung**, [2012]**used particle swarm optimization and bottleneck heuristic to solve a hybrid flow shop scheduling problem to minimize make-span. For group scheduling, another hybrid PSO was proposed by Hajinejad, Salmasi & Mokhtari, **[2011],** with a single objective of minimizing total flow time. Ranked order value-based encoding scheme was developed for their work.

A cost optimization problem of a hybrid flow shop was studied by Han et al, **[2012],** where total production cost was a function of time-based scheduling. The costs include processing costs, waiting costs, and product storage costs.

B. Liu, Wang & Jin**, [2005]** considered a hybrid flow shop with stochastic processing time and the objective was to minimize make-span. PSO along with simulated annealing was used to solve the NP-hard problem.

Many optimization and search problems have used PSO for its simplicity and ability to tackle difficult NP-hard problems successfully **[Ercan, 2008a] .**The performance of PSO is better than all existing GA and ACS algorithms **[Tseng & Liao, 2008]** .So, PSO can be a preferable method among other available solution approaches. That is why for this present work PSO has been selected for solving multi-objective constrained scheduling problems.

2.3 Uncertainties in Flow Shop

Uncertainty in processing time can be described in three ways (1) bounded form, (2) probability description and (3) fuzzy description **[Karunakaran et al, 2017].** In this work, we have used the probability distribution concept. Scheduling under uncertainty can be classified into two classes. One is preventive and the other one is reactive. The preventive scheduling approach requires historical data on the problem **[Karunakaran et al, 2017].** Uncertainty is addressed by preventive scheduling in current work.

Another way to address uncertainty is to model the tasks to have multiple processing times concerning their criticality. This approach converts these scheduling problems into deterministic scheduling with alternative processing times **[Novak et al., 2019].**

To address uncertainty in processing time Janak, Lin & Floudas**, [2007]** had proved that the true value of the processing time can be represented in terms of the nominal processing time as follows

$$\tilde{\alpha}_{ij} = (1 + \epsilon \xi_{\alpha_{ij}})\alpha_{ij} \quad \textbf{(2.1)}$$

Where $\xi_{\alpha ij}$ is a random variable with known distribution. For the case where the uncertainty ϵ is uniform in the interval $[-1, 1]$.

A reactive scheduling framework was developed by Janak et al**, [2006]** that used mixed-integer linear programming (MILP) to develop a short time scheduling model. They identified the jobs, which were not affected by uncertainty to avoid rescheduling of all jobs. Another approach to reactive scheduling is a two-step procedure, where the first step produces an initial solution and the second adopts to unexpected situations **[Rahmani & Heydari, 2014].**

Considering processing time as a deterministic value may cause an error in the result. For this purpose Behnamian & Zandieh, **[2013]** had considered position-dependent learning effects, they assumed process and setup times as a function of the number of repetitions of production item.

Apart from processing time, another source of uncertainty on the production floor is machine break-down. Mirabi, Ghomi & Jolai, **[2013]** had considered machine break-down situation, where they multiplied machine break-down probability with machine repair time to determine addition time required for machine unavailability**.** Jamili, **[2019]** had considered uncertainty from workers' rest time concept. Because, if a job is scheduled during a worker's rest time, there will be a difference in scheduled production quantity and realized production quantity.

Xiong, Xing & Chen**, [2013]** developed robust scheduling for a flexible job-shop problem with random machine break-down. They had two objectives make-span and robustness. They identified five different ways to address the robustness of the model.

2.4 Machine Learning Concept in Flow Shop Scheduling

After reviewing solution approaches for flow shop, it was observed that heuristics are moving towards artificial intelligence search techniques that reflect new solution methods **[H. Wang, 2005]**.

It is really important to define a dispatching rule in a flexible manufacturing system **[Shaw, Park & Raman, 1992].** So, they developed a framework incorporating machine learning capabilities to build an inductive learning module. Their proposed framework could help to classify jobs based on their distinct manufacturing patterns. The draw-back of dispatching rule is that one dispatching rule may not be best for all jobs. Dispatching rule may vary based upon a job **[Priore et al., 2006].** A case-based reasoning approach was proposed by Priore et al., **[2006]** that could specify different dispatch rules for different jobs. Priore et al.**, [2018]** compared the performance of various machine learning algorithms to identify dispatching rules in a flexible manufacturing system.

Dorronsoro & Pinel, **[2017]** combined Virtual Savant (VS) with a parallel genetic algorithm (called PA-CGA) which provided accurate results in extremely low run times. So, it can be said that combining machine learning concept with heuristics increases the efficiency of the solver.

Among various machine learning algorithms, the support vector machine is superior **[Y. H. Liu et al., 2005].** A support vector machine scheduler was developed to identify dispatching rules in a dynamic flexible manufacturing system and the throughput result was better than static dispatching rules.

2.5 Summary of Literature Review

From the previous research works, it can be concluded that various researches had focused on solving scheduling problems with various approaches. In the very first, researchers tried to solve the scheduling problem by exact methods. When scheduling problems were identified as NP-hard problems, researches started to explore new optimization techniques focusing on local solutions instead of global once. During that phase, various optimization algorithms (i.e. GA, TS, SA, PSO, ACO, Firefly Algorithm) were explored. Furthermore, various hybrid algorithms were used by combining the above-mentioned algorithms. With time production complexity increased and researchers started to consider uncertainties in the production floor. After the advancement of artificial intelligence, some work had been done by combing machine learning with metaheuristics and that proved to give better results compared to previous approaches.

CHAPTER 3: THEORETICAL FRAMEWORK

3.1 Support Vector Machine (SVM)

Machine learning is divided into two categories, one is supervised learning and the other one is unsupervised learning. If prior information about the predicting data set is present, then a supervised learning algorithm can be used. Otherwise, unsupervised algorithms need to be implemented. For the work, previous information about predicting data set is available, so a supervised machine learning algorithm is being used for this purpose. For this work, the SVM training algorithm is used for classification which is a supervised clustering algorithm. An SVM model is basically a representation of examples in a feature space is such a way that examples belonging to different classes can be separated by a boundary line or plane. Then the SVM model is used to predict the class of new examples. This prediction is done based on the distance between the boundary line and their position on which they fall on.

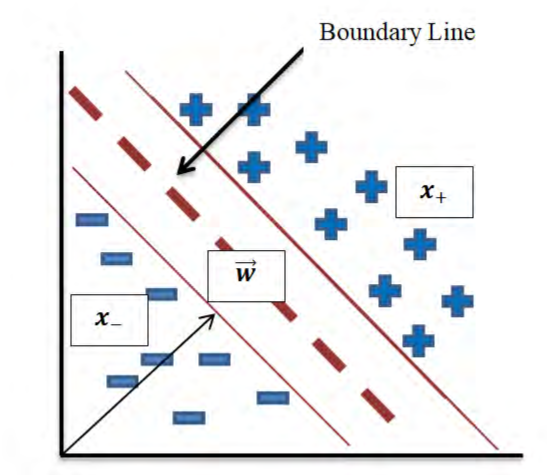

Figure 3.1: Presentation of SVM

In **Figure 3.1** x_+ and x_- corresponds to two different classes, those are separated by the boundary line.

3.1.1 Mathematical Model

Ɦ is a hyperplane with given dot product space, which consists of the following set of pattern vectors.

$x_{1,}x_{2,}x_{3,}$ … … … … … $x_m \in$ Ɦ

The hyperplane Ɦ can be written as following Eq. (**3.1**):

$$\{x \in Ɦ| < w, x > +b = 0\}, w \in Ɦ, b \in \mathbb{R} \quad \textbf{(3.1)}$$

Here, w is a vector orthogonal to the hyperplane Ɦ and b is a threshold value. In Eq. (**3.1**), $< w, x >$ is the length x of along the direction of w, when w is an unit vector . Furthermore, $(w, b) \in$ Ɦ $\times \mathbb{R}$ is called canonical form of hyperplane Ɦ with respect to $x_{1,}x_{2,}x_{3,}$ … … … … … $x_m \in$ Ɦ, if it is scaled such that

$$min|< w, x_i > +b| = 1 \quad \textbf{(3.2)}$$

$$\forall i = 1,2, \ldots \ldots \ldots \ldots m$$

It can be said that the point closest to the hyperplane has a distance of $1/_{\|w\|}$ in Eq. (**3.2**). From Eq. (**3.2**), it is seen that two pairs of points (w, b) and $(-w, -b)$ satisfies the canonical hyperplane. However, for pattern recognition, these two hyperplanes are different for their different orientation and both of them correspond to the decision functions given by following Eq. (**3.3**):

$$f_{w,b}: Ɦ \rightarrow \{\pm 1\} \quad \textbf{(3.3)}$$

$$x \rightarrow f_{w,b}(x) = sgn(< w, x > +b)$$

In Eq. **(3.3)**, $sgn(< w, x > +b)$ means $(< w, x > +b)$ will be either +1 or -1.These decision functions are the inverse of each other. However, by using $y_i \in \{\pm 1\}$ which is associated with x_i, these two hyperplanes can be represented as a single identity.

For pattern recognition, the target is to find a solution for $f_{w,b}$ that satisfies $f_{w,b}(x_i) = y_i$ for all *i,* which means, it can separate the training data set correctly. To achieve a large margin between hyperplanes $\|w\|$ should be kept small so that the distance between hyperplanes ($1/_{\|w\|}$) is large enough.

3.1.2 Optimal Hyperplane

In order to find an optimal hyperplane, the target is to find a decision function

$$f_{w,b}(x) = sgn(< w, x > +b) \quad \textbf{(3.4)}$$

Which satisfies

$$f_{w,b}(x_i) = y_i \quad \textbf{(3.5)}$$

Where example sets are (x_1, y_1),…….., (x_m, y_m), $x_i \in$ Ƕ, $y_i \in \{\pm 1\}$.

If such $f_{w,b}(x)$ exists which satisfies Eq. **(3.5)**, from canonicality of Eq. **(3.2)**, it implies

$$y_i(< x_i, w > +b) \geq 1 \quad \textbf{(3.6)}$$

So, a generalized hyperplane can be constructed by solving the following problem:

$$minimize\ \tau(w) = \frac{1}{2}\|w\|^2 \quad \textbf{(3.7)}$$

$$\forall w \in \text{Ƕ}, b \in \mathbb{R}$$

Subjected to,

$$y_i(< x_i, w > +b) \geq 1 \quad \textbf{(3.8)}$$

$$\forall i = 1,2, \dots \dots \dots \dots m$$

The optimization problem presented in Eq. **(3.7**) and Eq. **(3.8**), is called the primal optimization problem.

3.1.3 Lagrangian Transformation and Support Vectors

To solve the constrained primal optimization problem a dual problem needs to be derived, where both primal and dual problems have the same solution. For this purpose, Lagrangian is introduced in following Eq. **(3.9**).

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i(< x_i, w > +b) - 1)) \quad \textbf{(3.9)}$$

$$for\ \alpha_i \geq 0$$

This Lagrangian should be maximized with respect to α_i, and minimized with respect to *w* and *b*. As a result at the saddle point, derivatives of L with respect to primal variables will be zero which implies,

$$\frac{\partial}{\partial b} L(w, b, \alpha) = 0 \quad \textbf{(3.10)}$$

$$\frac{\partial}{\partial w} L(w, b, \alpha) = 0 \quad \textbf{(3.11)}$$

Solving Eq. **(3.10**) and Eq. **(3.11**), leads to following formations

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \quad \textbf{(3.12)}$$

$$w = \sum_{i=1}^{m} \alpha_i y_i x_i \qquad \textbf{(3.13)}$$

From Eq. **(3.13)**, it is seen that solution vector *w* is an expansion of training samples and it has a unique expression. However, *w* being unique α_i can have similar values.

According to KKT theorem **[Schölkopf & Smola, 2000]** only the Lagrange multipliers α_i that are non-zero at the saddle point, correspond to the constraint in Eq. **(3.8)**. Formally for all *i*=1,……. m, it is given,

$$\alpha_i[y_i(< x_i, w > +b) - 1] = 0 \qquad \textbf{(3.14)}$$

The patterns x_i , for which $\alpha_i > 0$ are called Support Vectors. From Eq. **(3.15)**, it can be seen that they lie exactly on the margin. So, the remaining examples in training set become irrelevant, for them Eq. **(3.8)**, are satisfied automatically.

Now, substituting the extreme conditions obtained in Eq. **(3.12)** and Eq. **(3.13)**, into Lagrangian in Eq. **(3.9)**, following dual formulation of the problem can be obtained.

$$maximize\, W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j\, y_i y_j < x_i, x_j > \qquad \textbf{(3.15)}$$

$$\forall\, \alpha_i \in \mathbb{R}^m$$

Subjected to,

$$\alpha_i \geq 0 \qquad \textbf{(3.16)}$$

$$\forall i = 1,2, \dots \dots \dots \dots m$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \qquad \textbf{(3.17)}$$

Now by substituting the Eq. **(3.13**), into decision function, Eq. **(3.3**), an expression is obtained in terms of dot products between the pattern to be classified and Support Vectors,

$$f(x) = sgn(\sum_{i=1}^{m} \alpha_i \, y_i < x, x_i > +b) \qquad \textbf{(3.18)}$$

3.1.4 Nonlinear Support Vector Machine

In the last section, all data set considered are linear. However, to deal with more general decision surfaces, Kernel transformation is used to nonlinearly transform the data set $x_{1,} x_{2,} \dots \dots, x_m \in \mathcal{X}$ into high-dimensional feature space. For linear separation in the feature space map $\phi: x_i \rightarrow x_i^*$ is used.

Cover'r theorem characterizes the number of possible linear separations of m points in an N-dimensional space. If m < N + 1, then 2^m is possible. According to this Theorem number of separation can be given by $2\sum_{i=1}^{N} \binom{m-1}{i}$ **[Berge, 1957].**

With the increase in the number of N, the number of elements is the sum increases. So, it can be said that the number of separations increases with the increase in dimensionality.

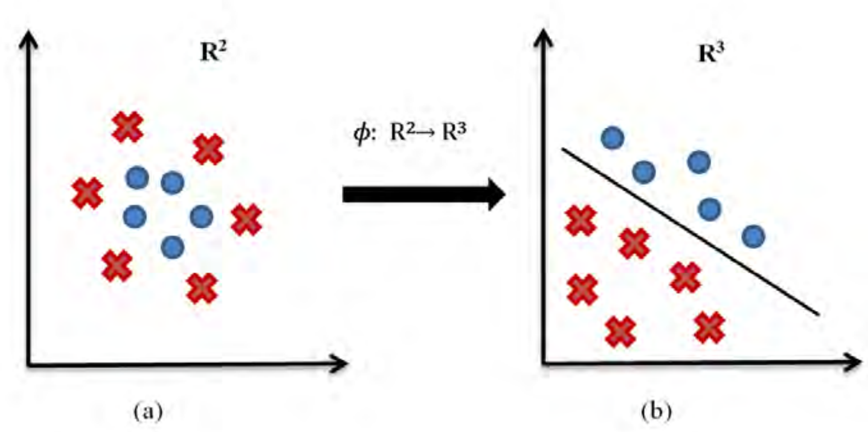

Figure 3.2: Mapping non-linear data into higher dimensional feature space

In **Figure 3.2(a)** nonlinear data is shown and in **Figure 3.2 (b)** those data are transferred to higher dimensional feature space and separating it by a hyperplane.

In order to make Eq. **(3.15**) and Eq. **(3.18**), suitable for a general decision surface, $< x, x_i >$ is substituted by $< \phi(x), \phi(x_i) >$ in higher dimensional space. As this substation is computationally expensive, a positive kernel is used to make the calculation easier.

$$maximize\ W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j\, y_i y_j < \phi(x), \phi(x_i) > \qquad \textbf{(3.19)}$$

$$\forall\, \alpha_i \in \mathbb{R}^m$$

$$f(x) = sgn(\sum_{i=1}^{m} \alpha_i\, y_i < \phi(x), \phi(x_i) > +b) \qquad \textbf{(3.20)}$$

$$(< \phi(x), \phi(x_i) >) = k(x, x_i) \qquad \textbf{(3.21)}$$

Using this transformation of Eq. **(3.21**) into Eq. **(3.18**), a new decision function can be obtained as following Eq. **(3.22**).

$$f(x) = sgn(\sum_{i=1}^{m} \alpha_i\, y_i k(x, x_i) + b) \qquad \textbf{(3.22)}$$

Now to calculate threshold value *b*, implying to KKT conditions **[Schölkopf & Smola, 2002]** to Eq. **(3.15**) , α_i> 0 following formulation can be obtained

$$\sum_{i=1}^{m} \alpha_i\, y_i k\left(x_j, x_i\right) + b = y_j \qquad \textbf{(3.23)}$$

So, the threshold value can be obtained as following Eq. **(3.24**).

$$b = y_j - \sum_{i=1}^{m} \alpha_i\, y_i k\left(x_j, x_i\right) \qquad \textbf{(3.24)}$$

There are some popular forms of Kernel functions. Some of them are given by following Eq. **(3.25**), Eq. **(3.26**), Eq. **(3.27**) and Eq. **(3.28**): **[Team, D. F., 2018]**

Polynomial Kernel Classifier with a degree of d

$$k(x, x_i) =< x, x_i >^d \tag{3.25}$$

Gaussian Kernel

$$k(x, x_i) = \exp(\frac{-\|x - x_i\|^2}{2\sigma^2}) \tag{3.26}$$

Radial basis function classifier with Gaussian Kernel of width c>0

$$k(x, x_i) = \exp(\frac{-\|x - x_i\|^2}{c}) \tag{3.27}$$

Sigmoid kernel

$$k(x, x_i) = \tanh(\mathcal{B} < x, x_i > +\mathcal{C}) \tag{3.28}$$

Where $\mathcal{B} > 0$ and $\mathcal{C} \in \mathbb{R}$

3.1.5 Principle Component Analysis (PCA)

PCA can reduce the dimensionality of the feature space and reducing dimensionality helps to create a classification model that has less overfitting problems. PCA linearly transforms predictors of training data set to remove redundant dimensions, then it creates a new set of predictors. However, implementing PCA may result in an under-fitting problem. So, the use of PCA needs to be considered properly, prior to implementing it.

3.1.6 Soft Margin

To implement SVM for classification, in reality, it becomes difficult to develop a separating hyperplane because in many cases this hyperplane may not exist. It has also shown that it is an

NP-hard problem to find a hyperplane whose training error is less. A new approach is using slack variables to deal with the difficulty **[Schölkopf & Smola, 2002a].**

$$\xi_i \geq 0 \qquad \textbf{(3.29)}$$

$$\forall i = 1,2, \dots \dots \dots \dots m$$

Eq. **(3.29)**, is used for relaxing Eq. **(3.8)**

$$y_i(< x_i, w > +b) \geq 1 - \xi_i \qquad \textbf{(3.30)}$$

$$\forall i = 1,2, \dots \dots \dots \dots m$$

By using Eq. **(3.29)**, into Eq. **(3.7)**, an updated objective function can be obtained considering the slack variable concept.

$$minimize\ \tau(w, \xi) = \frac{1}{2}\|w\|^2 + \frac{C}{m}\sum_{i=m}^{m} \xi_i \qquad \textbf{(3.31)}$$

$$\forall w \in \mathrm{H}\Gamma,\ \forall\ \xi_i \in \mathbb{R}^m$$

In Eq. **(3.31)**, *C* is a positive constant which determines the trade-off between minimizing the training error and maximizing the margin. However, it is not possible to have prior information about *C*. So, it becomes difficult to predict. In order to solve this problem parameter *C* replaced by *v* and a new parameter ρ is added.

$$minimize\ \tau(w, \xi, \rho) = \frac{1}{2}\|w\|^2 - v\rho + \frac{1}{m}\sum_{i=m}^{m} \xi_i \qquad \textbf{(3.32)}$$

$$\forall w \in \mathrm{H}\Gamma,\ \forall\ \xi_i \in \mathbb{R}^m, \rho b \in \mathbb{R}$$

Subjected to:

$$y_i(< x_i, w > +b) \geq \rho - \xi_i \qquad \textbf{(3.33)}$$

$$\forall i = 1,2, \dots \dots \dots \dots m$$

And $\xi_i \geq 0, \rho \geq 0$ **(3.34)**

$$\forall i = 1,2, \ldots \ldots \ldots \ldots m$$

Here, v is a parameter and ρ is an additional variable to be optimized. For, $\xi = 0$, Eq. (**3.33**), says that two classes can be separated by a margin $^{2\rho}/_{||w||}$.

So, the final SVM model can be defined by the above-stated optimization problem from Eq. (**3.32**) to Eq. (**3.34**)

3.1.7 Multiclass Classification SVM

SVM can be used for multi-class classification. In SMV multi-class classification can be done in two approaches. One is called One Vs One classification, where if there are m number of classes, m times SVM classification is done. Another one is called one Vs All, in this approach classification is done by a certain class versus considering the rest of the classes as one class and it continues until the correct class is identified.

3.1.7.1 One versus the Rest:

To develop M-class classifier, a set of binary classifiers $f^1, f^2, \ldots \ldots \ldots \ldots f^m$ are developed and each of them is trained to separate one class from the rest of the classes. The main shortcoming of this approach is that it is unclear whether real value outputs are on a comparable scale or not.

3.1.7.2 Pairwise Classification:

In pairwise classification, a classifier is trained for each possible pair of classes. For M classes, the number of a binary classifier is $^{(M-1)M}/_2$ .This number is usually larger than the number of one-versus-the-rest classifiers; for instance, if M = 15, 105 binary classifiers are required rather than 15 as in one vs rest method. This method may require larger training times; however, the individual problems are significantly smaller. In this method when a test pattern is to be classified, it requires to evaluate all 105 binary classifiers and classify according to which of the classes gets the highest number of votes**.** There are two reasons first, the training sets are smaller, and second, the problems to be learned are usually easier, since the classes have less overlap, pairwise classification is much faster.

3.1.7.3 Error-Correcting Output Coding:

Error-Correcting Output Codes is an ensemble method designed for multiclass classification problems. If a set of a binary classifier $f^1, f^2, \ldots \ldots \ldots \ldots f^L$ is properly designed it will have consistency binary responses. However, if the responses are inconsistent or the data set is not large enough, a binary classifier is not reliable. To deal with these cases robustness against some error was proposed, by designing a clever set of binary problems. In this approach, the closest match between the vector of responses and the rows of the matrix is determined using the Hamming distance (the number of entries where the two vectors differ; essentially, the L_∞ distance). In this case, it is possible to guarantee the correct classification of all test examples which may lead to at most one error amongst the binary classifiers. This method is suitable for multiclass classification; however, it may have some limitations in using crucial quantity in classifiers.

3.1.8 Cross-Validation

To estimate the expected error leave-one-out method can be used. The leave-one-out method is defined by leaving out one of the training examples and use remaining for training the model and using the left out training set for testing the model. This procedure is repeated until all training examples are used to test the accuracy of the model. This procedure averages the error generated from all texting set.

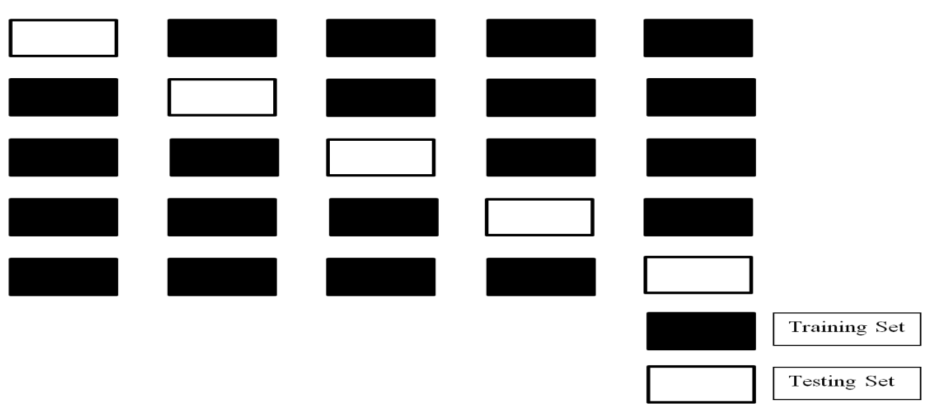

Figure 3.3: 5-Fold Cross-validation of Training Data Set

3.2 Particle Swarm Optimization (PSO)

The theory of particle swarm optimization (PSO) is proliferating. It has several applications. The algorithm of PSO is inspired by the behavior of animal societies like birds and fishes that don‘t have any leader in their group or swarm. Considering animal troops with no leader have to find food by random search following the other members having the closest position with a food source which in terms of PSO may be regarded as a potential solution. The troops simultaneously communicate among members who already have a better situation and reach the desired destination. Animals with better conditions keep on informing the other troop members to move simultaneously towards the place. This keeps on happening repeatedly until the best conditions or a food source discovered. The animal’s social behavior is followed by the PSO algorithm to find optimal values. Particle swarm optimization consists of a swarm of particles, where a particle is the representation of a potential solution.

The advantage of using an optimization method PSO is that it does not use the gradient of the problem to be optimized, so the method can be readily employed for a host of optimization problems. This is especially useful when the gradient is too laborious or even impossible to derive. This versatility comes at a price, however, as PSO does not always work well and may need tuning of its behavioral parameters so as to perform well on the problem at hand.

3.2.1 PSO Procedure

A particle swarm optimization algorithm requires the definition of the following concepts of swarm size, information links, initialization, equations of motions.

3.2.1.1 Swarm Size

The size of the swarm is fixed at the beginning. Higher swarm size fastens the search in terms of iteration, however, it may require more compared to smaller swarm size. Rather it is more important to reduce computational time. Apparently, smaller swarm sizes can increase computational time. So, using an optimum number of swarm size is necessary to make the algorithm efficient.

3.2.1.2 Initialization

Initialization means initially randomly placing the particles according to a uniform distribution in a search space. This stage is virtually present in all the algorithms of stochastic iterative optimization. Moreover, the particles have velocities which are vectors by definition or, more precisely, an operator, which, applied to a position, will give another position. Basically, it is a displacement, called velocity because during the increment of the iterations it is always implicitly regarded as equal to 1.

3.2.1.3 Equations of Motions

The velocity of each particle is updated using the following Eq. **(3.35)**.

$$v_i\,(t+1) = wv_i(t) + c_1r_1[\hat{x_i}(t) - x_i(t)] + c_2r_2[g(t) - x_i(t)] \qquad \textbf{(3.35)}$$

Where,

i	=	The particle index
w	=	The inertial coefficient usually between 0.8 and 1.2
c_1, c_2	=	Acceleration coefficients, $0 \leq c_1, c_2 \leq 2$
r_1, r_2	=	Random values which are generated for every velocity update, $0 \leq r_1, r_2 \leq 1$
$v_i(t)$	=	Particle's velocity at time t
$x_i(t)$	=	Particle's position in time t
$\hat{x_i}(t)$	=	Particle's individual best solution as of time t, which is also called $\boldsymbol{p_{best}}$
$g(t)$	=	The swarm's best solution as of time t

3.2.2 Pseudo-code Algorithm

```
For each particle
        {Initialize
} End

(Estimate intensity of a particle as an object)
Do
For each particle
        {Calculate fitness
value}
If fitness value is better than pbest [local best]
        {Set pbest =current fitness
value }
If pbest is better than gbest [global best]
        {Set gbest = pbest
} End
For each particle
        {- Calculate particle velocity according to the equation of v
        - update particle position according to the equation of present x

} End
```

3.2.3 Modified PSO

From the basic form of PSO, further, there have been some modifications to make it applicable to more generalized cases.

3.2.3.1 Constrained PSO

In constrained PSO, particles are initiated after constrained are satisfied. This method reduces the feasible space wherein the solution to the problem can be found. Optimization algorithms need to ensure that a feasible solution is found. That is the optimization algorithm should find a solution that both optimizes the objective function satisfies all constraints. If it is not possible to satisfy all constraints, the algorithm has to balance the trades off between optimal objective function value and the number of constraints violated.

3.2.3.2 Multi-objective PSO

In order to solve multi-objective problems using PSO, the following approaches can be used.

A) Aggregating approaches

In this method, all objectives are transformed into a single objective and it is done by multiplying all objectives with a weight depending on their importance level. Then their summation is taken to convert into a single objective. This method requires prior knowledge about the problem.

B) Lexicographic ordering

In this approach, all objectives are ranked based on their importance level. It starts by optimizing the most important objective and using that solution to optimize the second most important objective function. This process continues until all objectives are optimized. This method is suitable for problems with less number of objectives.

C) Sub-population approach

A sub-population approach works by dividing the swarm of particles into sub-swarm. These sub-swarms search for a solution on their own and share the information with other sub-swarms.

D) Pareto based approaches

Pareto based approach is one of the most applied methods to solve multi-objective PSO. This approach finds a set of solutions, which is called non-dominated solutions.

CHAPTER 4: MODEL FORMULATION

4.1 Problem Identification

For mass production of any product, the flow shop process is preferable as it can ensure high production volume in less time. So, when the production planning of a set of jobs can be done in an optimized way, it will ensure maximum profitability. During the production planning, each job has some associated qualitative values, which helps the planner to prioritize a job. These qualitative values need to be considered during production scheduling. Though previously no scheduling model had addressed these qualitative values such as product value, customer priority level, material in hand status, available production time, this work will address these qualitative values associated with a particular job to interpret the priority level of the job in a better way. Apart from this, it is also required to achieve multiple objectives during scheduling which includes minimizing make-span, minimize delay in delivery, maximize efficiency. Moreover, a real-life scheduling problem involves various uncertainties that can originate from machine break-down, processing time of a job in a different machine and so on. So, in this work processing time of jobs is considered as stochastic values rather than deterministic ones. Adding to this, it also aims to minimize make-span, minimize delay in delivery, maximize efficiency so that the model can be more adaptive to real-life scenarios and requires less human involvement during the planning phase.

The development of optimized scheduling is divided into two steps. The first step involves data analysis of qualitative values associated with each job. After the data analysis jobs are classified into three classes. Furthermore, jobs belonging to the same class are ranked according to their earliness of the delivery date. In this way, all jobs are sequenced before the implementation of the model. This model mainly assigns jobs to suitable machines in such a way that all objectives are fulfilled.

4.2 Objective Functions

In this work, three objectives are being addressed. These objectives are minimizing make-span, minimizing total tardiness and maximizing efficiency by minimizing machine idle time.

a) Make-Span: Make-span means the completion time of a job which has maximum value among a set of jobs on the production floor. This criterion had been mostly studied in previous research works.
b) Tardiness: Tardiness is also termed as the lateness of a job. This is calculated as a difference between real completion time and the required delivery date of a job. This criterion is quite important because when an organization fails to meet a committed deadline it results in a loss of goodwill and market value of the organization.
c) Efficiency: Generally efficiency is calculated in terms of a ratio of output versus input. Whereas, in this work, efficiency is calculated in terms of machine idle times. It is considered that frequent line or machine change over results in lower production quantity as machines are kept idle for a new set up and also it requires additional time for the operator's skill development.

For this purpose, three different objective functions are developed. Finally, this multi-objective optimization is solved by a weighted average method.

4.2.1 Minimize Make-span

The first objective is to minimize make-span, which means minimize the making time of a job which spends a maximum amount of time on the production floor. The notation for make-span is $\boldsymbol{C_{i(max)}}$. In this notation $\boldsymbol{C_i}$ is defined as the completion time of job $\boldsymbol{i}$. In the formulation subscript, $\boldsymbol{max}$ is used to address the completion time of a job which spends maximum time on the production floor compared to other jobs. The following Eq. **(4.1)** is the first objective function.

Minimize make-span:

$$Min\!: C_{i(max)} \tag{4.1}$$

4.2.1.1 Completion Time Calculation

In order to calculate make-span, it is prerequisite to calculation the completion time of each job in each stage, which is defined by $\boldsymbol{C_{ij}}$, this means the completion time of job $\boldsymbol{i}$ in stage $\boldsymbol{j}$. It can be calculated by following Eq. (**4.2**), which is applicable when calculating completion time in stage-1. To calculate the completion time of a job, which is sequenced in position $\boldsymbol{i}$ two considerations are done, those are job waiting time and a particular job is assigned to which machine. A job may need to wait in a queue before its operation starts in a particular machine. So, to consider the waiting time is necessary. In order to consider that waiting time for that particular machine, the processing time, $\boldsymbol{P^o_{ijk}}$ of all jobs that precedes a particular job, $\boldsymbol{i}$ in a particular machine is multiplied with the assignment decision variable, $\boldsymbol{X_{ijk}}$ and their summation is added to have the value of waiting time. If any of the preceding jobs are not assigned to that particular machine, their corresponding assignment decision variable will be zero and $\boldsymbol{X_{ijk}.P^o_{ijk}}$ for that not assigned job will be zero. As a result, its processing time will not be considered in the waiting time calculation. Furthermore, $\boldsymbol{\sum_{i=1}^{i=i} X_{ijk}.P^o_{ijk}}$ is multiplied by $\boldsymbol{X_{ijk}}$ to check if job, $\boldsymbol{i}$ is assigned to machine $\boldsymbol{k}$ or not. This calculation is done for all machines in stage-1. As one job can be assigned to only one machine in one stage, the processing time will be available for only one machine and for the rest of the machine it will be zero. So, by taking the summation of processing times in all machines in stage-1 will give the completion time of stage-1.

$$C_{ij} = \sum_{k=1}^{k=M_j} \{(\sum_{i=1}^{i=i} X_{ijk}.P^o_{ijk}).X_{ijk}\} \qquad \textbf{(4.2)}$$

$$\forall\ i \in \{1,2,3,\ldots\ldots N\}$$
$$\text{for } j = 1$$

Now, to calculate the completion time for other stages apart from stage 1 some additional considerations are done. It is really important to calculate the waiting time of each job when it is assigned to a particular machine. While calculating the completion time for the remaining stages, there can be two scenarios.

a. If the mean processing time in the preceding stage is smaller than the mean processing time of the current stage, then the completion time of the current stage is calculated by following Eq. (**4.3**) and Eq. (**4.4**).

$$C_{ij} = \sum_{k=1}^{k=M_j} (C_{i,j-1} + X_{ijk}.P^{o}_{ijk}).X_{ijk} \quad \textbf{(4.3)}$$

for i =1

∀j ϵ {2,3,.......... L}

$$C_{ij} = \sum_{k=1}^{k=M_j} [max\{(C_{1,j-1}.X_{1,j,k}), (C_{2,j-1}.X_{2,j,k}), \dots \dots .(C_{i-1,j-1}.X_{i-1,j,k})\} \\ + max(C_{i,j-1}, \sum_{i=1}^{i=i-1} X_{ijk}.P^{o}_{ijk}) + X_{ijk}.P^{o}_{ijk}].X_{ijk} \quad \textbf{(4.4)}$$

∀ i ϵ {2,3,………N}

∀j ϵ {2,3,.......... L}

In scenario (a) which is presented by Eq. **(4.3**) and Eq. **(4.4**) at first completion time ($\boldsymbol{C_{i-1,j-1}}$) of all preceding jobs in the earlier stage is multiplied by the assignment decision variable $\boldsymbol{X_{i-1,j,k}}$ corresponding to the particular machine of current stage. Then among all preceding jobs which will also be assigned to the particular machine $\boldsymbol{k}$ in stage $\boldsymbol{j}$ prior to job $\boldsymbol{i}$, the job with largest completion time in an earlier stage is considered and its completion time added to the processing time of job $\boldsymbol{i}$ in machine $\boldsymbol{k}$ in stage ***j.*** Then the completion time($\boldsymbol{C_{i,j-1}}$) of job $\boldsymbol{i}$ in stage ***j-1*** is compared with the waiting time ($\boldsymbol{\sum_{i=1}^{i=i-1} X_{ijk}.P^{o}_{ijk}}$) of job $\boldsymbol{i}$ in stage $\boldsymbol{j}$ in machine $\boldsymbol{k}$. Among them the larger one on is added to get final completion time in stage $\boldsymbol{j}$. Then this value is multiplied by assignment variable, $\boldsymbol{X_{ijk}}$**.** Completion time of job *i* is stage j will be calculated for only that machine for which assignment variable $\boldsymbol{X_{ijk}}$ is 1

b. If the mean processing time in the preceding stage is greater than the mean processing time of the current stage, then the completion time of the current stage is calculated by following Eq. **(4.6**).

$$C_{ij} = \sum_{k=1}^{k=M_j} \{max(C_{i,j-1}, \sum_{i=1}^{i=i-1} X_{ijk}.P^{o}_{ijk}) + (X_{ijk}.P^{o}_{ijk})\}.X_{ijk} \quad \textbf{(4.5)}$$

∀ i ϵ {1,2,………N}

∀j ϵ {2,3,.......... L}

In scenario (b) which is presented by Eq. **(4.5**) the completion time($\boldsymbol{C_{i,j-1}}$) of job $\boldsymbol{i}$ in stage ***j-1*** is compared with the waiting time ($\boldsymbol{\sum_{i=1}^{i=i-1} X_{ijk}.P^{o}_{ijk}}$) of job $\boldsymbol{i}$ in stage $\boldsymbol{j}$ in machine $\boldsymbol{k}$ and the larger value is added to the summation of waiting time and processing time of job $\boldsymbol{i}$ in stage $\boldsymbol{j}$ in machine $\boldsymbol{k}$ which is $\boldsymbol{X_{ijk}.P^{o}_{ijk}}$**.**

So, for each stage, the completion time is updated considering the completion time of the earlier stage.

In this manner completion time of the last stage is calculated, which is actually the final completion time a particular job. It is represented by Eq. **(4.6)** and Eq. **(4.7)**.

$$C_i = \sum_{j=1}^{j=L} \sum_{k=1}^{k=M_j} X_{ijk} \,. P_{ijk}^{o} \tag{4.6}$$

for i=1

$$C_i = C_{iL} \tag{4.7}$$

∀ i ϵ {2,3,………N}

4.2.1.2 Processing Time Calculation

To calculate the processing time of each job in each machine there are three considerations.

I. Processing time is considered as a stochastic element
II. Machine break down is considered during the processing time calculation
III. When a product is assigned to a machine which is not suitable for that product it requires additional set up a time

Processing time is considered as a stochastic element it is calculated using the following Eq. **(4.8)** so that the model is more adaptive to real-life situations

$$P_{ijk} = (1 + \varphi . \varepsilon) . P_{ijk}^{\sim} \tag{4.8}$$

During production to encounter machine break-down situations is very common. So, calculating the impact of machine break down in the processing time is considered where machine break-down probability can be calculated by following Eq. **(4.9)**.

$$\rho_{jk} = \frac{BT_{jk}}{BT_{Total}} \tag{4.9}$$

It is assumed that all machines are not suitable for all jobs. If a job is assigned to such a machine, where its operator has less idea about the job, in that case, operation time will have a learning time adding to the normal processing time. So, each machine has a processing time including learning curve and it is represented as P_{ijk}^{L}.

$$P_{ijk}^{o} = Y_{ijk}.P_{ijk} + (1 - Y_{ijk}).P_{ijk}^{L} \quad \textbf{(4.10)}$$

$$P_{ijk}^{o} = Y_{ijk}.P_{ijk} + (1 - Y_{ijk}).P_{ijk}^{L} + \rho_{jk}.RT_{jk} \quad \textbf{(4.11)}$$

If a job is assigned to a machine that is suitable for it, then processing time without the learning curve is considered. However, if a job is assigned to a machine that is not suitable for it then processing time with the learning curve is considered. For a particular machine, a job cannot have two processing times, both with and without a learning curve. It is ensured by $\boldsymbol{Y_{ijk}.P_{ijk} + (1 - Y_{ijk}).P_{ijk}^{L}}$ in Eq. **(4.10)**. To consider machine break downtime, machine break down probability is multiplied by the machine repair time and this is added to Eq. **(4.10)**. So the final processing time is calculated as per Eq. **(4.11)**.

4.2.2 Minimize Total Tardiness

The second objective function is to minimize delay in delivery, which is called minimize tardiness. For this work instead of considering tardiness, the tardiness of each job multiplied with the jobs respective priority index value is considered. This modification is done because a delay in the delivery of a job with higher priority will have a greater impact than the job with a lower priority level.

Minimize total tardiness:

$$Min: \sum_{i=1}^{i=N}[max\{0, \alpha_i . (C_i - DD_i)\}] \quad \textbf{(4.12)}$$

In the given Eq. **(4.12)**, $(\boldsymbol{C_i - DD_i})$ represents the difference between the completion time $(\boldsymbol{C_i})$ of a job $\boldsymbol{i}$ required delivery date $(\boldsymbol{DD_i})$ of that job. This difference is further multiplied by the priority index $(\boldsymbol{\alpha_i})$ of the respective job so that the priority level of a job comes into consideration during the calculation. If a job is finished prior to its required delivery date, $(\boldsymbol{C_i - DD_i})$ gives negative value. Considering this negative value while calculating tardiness, will give an erroneous result. It is not desired to have a finished product much prior to its required delivery date as it will result in an additional holding cost of that product. So, it is ensured that, if a job is finished earlier than its required delivery date negative value is not taken by taking a maximum value between zero and $\boldsymbol{\alpha_i} \, . \, (\boldsymbol{C_i} - \boldsymbol{DD_i})$ in Eq. **(4.12)** . By doing so, it is ensured that while calculating the tardiness of a particular job only nonnegative values are considered.

4.2.3 Maximize Efficiency

The third objective is to maximize efficiency, which can be obtained by minimizing the machine idle time. In a real production scenario, if a machine is suitable for a particular type of product and a different type of product is assigned to that machine it requires additional time for machine set up according to the new type of product. It also requires some learning time for the operator to adjust to the new product type. This additional machine set up time and the time for operators learning about new products reduces productivity. This can be minimized if a product that is suitable for a particular machine can be assigned to that machine. It will result in minimizing the machine idle time and subsequently maximizing efficiency.

Maximize Efficiency:

$$Min: \sum_{i=1}^{i=N} \sum_{j=1}^{j=L} \sum_{k=1}^{k=M_j} X_{ijk} \, . \, (1 - Y_{ijk}) . \, (P_{ijk}^{L} - P_{ijk} + T_{set\,up}) \qquad \textbf{(4.13)}$$

In Eq. **(4.13)**, $\boldsymbol{Y_{ijk}}$ is a binary variable, it is defined as, if machine $\boldsymbol{k}$ of stage $\boldsymbol{j}$ is suitable for product $\boldsymbol{i}$, then $\boldsymbol{Y_{ijk}}$ will have value 1, otherwise it will be zero. So, if the product $\boldsymbol{i}$ is assigned to a machine which is suitable for it, then $(\boldsymbol{1} - \boldsymbol{Y_{ijk}})$ will be zero. For that particular assignment value of the objective function will be zero. However, if the product $\boldsymbol{i}$ is assigned to a machine which is not suitable for it, the value of $(\boldsymbol{1} - \boldsymbol{Y_{ijk}})$ will be 1. It will require additional time for processing. This additional time is formulated in Eq. **(4.13)**, as $(\boldsymbol{P_{ijk}^{L}} - \boldsymbol{P_{ijk}} + \boldsymbol{T_{set\,up}})$**.** Here, $\boldsymbol{P_{ijk}^{L}} - \boldsymbol{P_{ijk}}$ gives the additional time required for a worker to learn about new product and $T_{set\,up}$ provides the additional time required for adjustment of the machine as per new products requirement.

Finally, this multi-objective model is formulated by the weighted aggregation approach and the objective function is given below:

$$w_1 . C_{i(max)} + w_2 . \sum_{i=1}^{i=N}[max\{0, \alpha_i . (C_i - DD_i)\}] \tag{4.14}$$
$$+ w_3 . \sum_{i=1}^{i=N} \sum_{j=1}^{j=L} \sum_{k=1}^{k=M_j} X_{ijk} . \left(1 - Y_{ijk}\right) . (P_{ijk}^{L} - P_{ijk} + T_{set\ up})$$

In this case, all the separate objective functions of minimizing make-span, minimizing tardiness and maximizing efficiency are multiplied by three weighted value w_1, w_2, w_3 respectively.

4.3 Decision Variable

This model has only a set of binary decision variables, which can have only two values either one or zero. If job ***i*** is assigned to machine ***k*** in stage ***j***, X_{ijk} will be 1, otherwise, it will be zero. If there is N number of jobs and in total L stages there are M_{total} machines, then the total number of decision variables will be N **x** M_{total}.

So following is the decision variable.

$$X_{ijk} \quad = \quad \begin{cases} 1, & if\ product\ i\ is\ assigned\ to\ machine\ k\ in\ stage\ j \\ 0, & \quad else \end{cases}$$

4.4 Constraints

The model formulation involves a set of constraints. The first constraint is that a job can be assigned to only one machine in a particular stage. For example, in stage 2 there is a total of 10 machines. So a job can be assigned to only one of those 10 machines in stage 2. This constraint will restrict the multiple assignments of the same job in different machines. Because one operation is done in one stage and it will be done by one machine on a particular job. This constraint is presented by following Eq. **(4.15)**.

$$\sum_{k=1}^{k=M_j} X_{ijk} = 1 \tag{4.15}$$

$\forall$ i ϵ {1,2,………N}

$\forall$j ϵ {1,2,…......... L}

The second constraint is, the decision variable, X_{ijk} can have only two values one or zero. As during job assignment, there can be only two scenarios, either a job is assigned to a particular machine or it is not assigned to that machine

$$X_{ijk} \in \{0,1\} \tag{4.16}$$

4.5 Assumptions

Each job follows the same order of stages

I. Time to transport a job between two-stage is negligible.
II. One job passes a particular machine only once.
III. A machine can operate at most on one job at a time.
IV. A job can be assigned to only one machine in one stage.
V. Machine break down probability of a particular machine is known.
VI. In each stage probability distribution of processing time of a job in a particular machine is known.
VII. Each machine is suitable for a particular type of product. Except for suitable jobs to that machine, other jobs will require additional machine set up time and learning time for the operator.
VIII. In each stage for the first job, a machine has no waiting time.

4.6 Parameters

N	=	Total number of jobs
L	=	Total number of stages
M_j	=	Total number of machines in stage j
M_{total}	=	Total number of machines in all stages
DD_i	=	Due date of job i
RD_i	=	Release date of job i
C_{ij}	=	Completion time of job i in stage j
C_i	=	Completion time of job i
ρ_{jk}	=	Probability of machine breakdown in stage j at machine k
RT_{jk}	=	Repair time of machine k in stage j
$P_{ijk}^{\sim}$	=	Mean Processing time of job i in stage j at machine k without a learning curve
P_{ijk}	=	Processing time of job i in stage j at machine k without a learning curve
P_{ijk}^{L}	=	Processing time of job i in stage j at machine k with a learning curve
P_{ijk}^{o}	=	Considered processing time of job i in stage j at machine k
$T_{set\ up}$	=	Machine set-up time
BT_{jk}	=	Busy time of machine k in stage j
BT_{total}	=	Total busy time of all machines
α_i	=	Priority index of job i
ε	=	A random variable with a known probability distribution
φ	=	A given uncertainty level
Y_{ijk}	=	$\begin{cases} 1, & \text{if a product i can be done in machine k in stage j} \\ 0, & \text{else} \end{cases}$
i	=	Job index
j	=	Stage/Operation Index
k	=	Machine Index
w_1, w_2, w_3	=	Weight of objective functions respectively

Table 4.1: Model Formulation

Individual objective functions

Minimize make-span : $Min: C_{i(max)}$

Minimize tardiness: $Min: \sum_{i=1}^{i=N}[max\{0, \alpha_i \,.\, (C_i \; - \; DD_i)\}]$

Maximize Efficiency: $Min: \sum_{i=1}^{i=N}\sum_{j=1}^{j=L}\sum_{k=1}^{k=M_j} X_{ijk} \,.\, \left(1 - Y_{ijk}\right).\, (P_{ijk}^{L} - P_{ijk} \; + T_{set\ up})$

Objective function

$w_1 .\, C_{i(max)} + w_2 .\, \sum_{i=1}^{i=N}[max\{0, \alpha_i \,.\, (C_i \; - \; DD_i)\}]$

$+w_3 .\, \sum_{i=1}^{i=N}\sum_{j=1}^{j=L}\sum_{k=1}^{k=M_j} X_{ijk} \,.\, \left(1 - Y_{ijk}\right).\, (P_{ijk}^{L} - P_{ijk} \; + T_{set\ up})$

Subjected to,

$\sum_{k=1}^{k=M_j} X_{ijk} = 1$

$\forall$ i $\in$ {1,2,………N}

$\forall$j $\in$ {1,2,…......... L}

$X_{ijk} \in \{0,1\}$

Where,

$C_{ij} = \sum_{k=1}^{k=M_j}\{\left(\sum_{i=1}^{i=i} X_{ijk} .\, P_{ijk}^{o}\right).\, X_{ijk}\}$

$\forall$ i $\in$ {1,2,3,……N}

for j =1

If the mean processing time in the preceding stage is smaller than the mean processing time of current stage:

$C_{ij} = \sum_{k=1}^{k=M_j}(C_{i,j-1} + X_{ijk} .\, P_{ijk}^{o}).\, X_{ijk}$

for i =1

$\forall$j $\in$ {2,3,…......... L}

$C_{ij} = \sum_{k=1}^{k=M_j}[max\{(C_{1,j-1} .\, X_{1,j,k}), (C_{2,j-1} .\, X_{2,j,k}), \dots \dots (C_{i-1,j-1} .\, X_{i-1,j,k})\}$

$+max(C_{i,j-1,}\; \sum_{i=1}^{i=i-1} X_{ijk} \,.\, P_{ijk}^{o}) + X_{ijk} .\, P_{ijk}^{o}].\, X_{ijk}$

$\forall$ i $\in$ {2,3,………N}

$\forall$j $\in$ {2,3,…......... L}

Table 4.2: Model Formulation (Continued)

If the mean processing time in the preceding stage is greater than the mean processing time of current stage:

$$C_{ij} = \sum_{k=1}^{k=M_j} \{max(C_{i,j-1},\ \sum_{i=1}^{i=i-1} X_{ijk} \,.\, P_{ijk}^{o}) + (X_{ijk}.\, P_{ijk}^{o})\}.\, X_{ijk}$$

∀ i ϵ {1,2,………N}

∀j ϵ {2,3,…......... L}

$$C_i = \sum_{j=1}^{j=L} \sum_{k=1}^{k=M_j} X_{ijk} \,.\, P_{ijk}^{o}$$

for i=1

$$C_i = C_{iL}$$

∀ i ϵ {2,3,………N}

$$P_{ijk}^{o} = Y_{ijk}.\, P_{ijk} + \left(1 - Y_{ijk}\right).\, P_{ijk}^{L} + \ \rho_{jk}.\, RT_{jk}$$

4.7 Proposed Framework of SVM Guided Hybrid Flow Shop Scheduling:

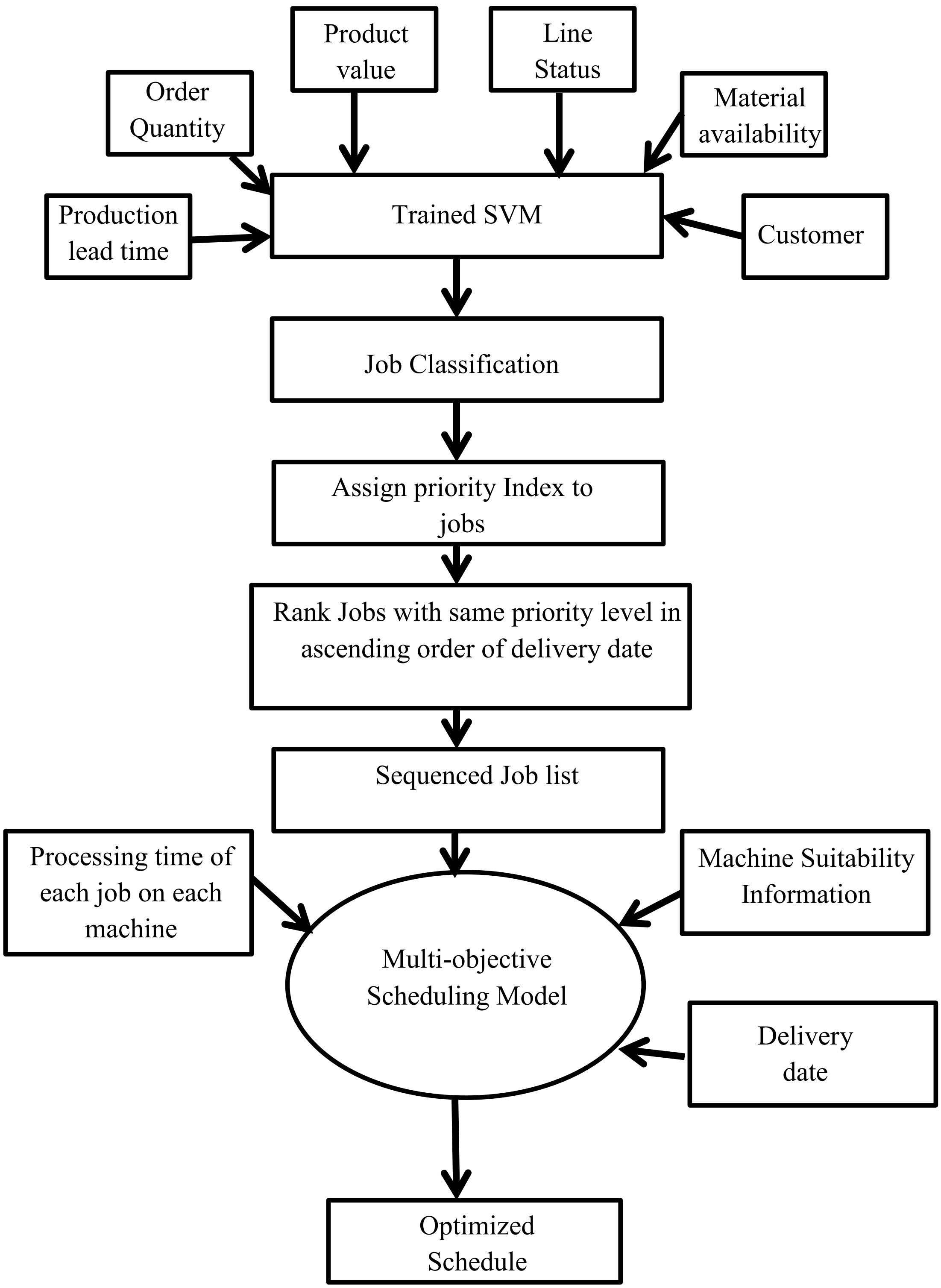

Figure 4.4: Proposed SMV guided scheduling framework

CHAPTER 5: MODEL IMPLEMENTATION

5.1 Case Study

In order to implement the hybrid flow shop scheduling model developed in the preceding chapter a textile industry is selected, where a certain portion of the entire production floor is considered for implementation. In the textile industry, the whole production process is divided into three stages, these are respectively cutting, sewing and finishing section. In the sewing stage, each production line comprises of several machines. However, once a product enters a line it passes through all the machines sequentially. So, to reduce model complexity, each line is considered as a single machine unit as there is no change in the sequence once a product enters a particular sewing line. The illustration of the hybrid flow shop is given below:

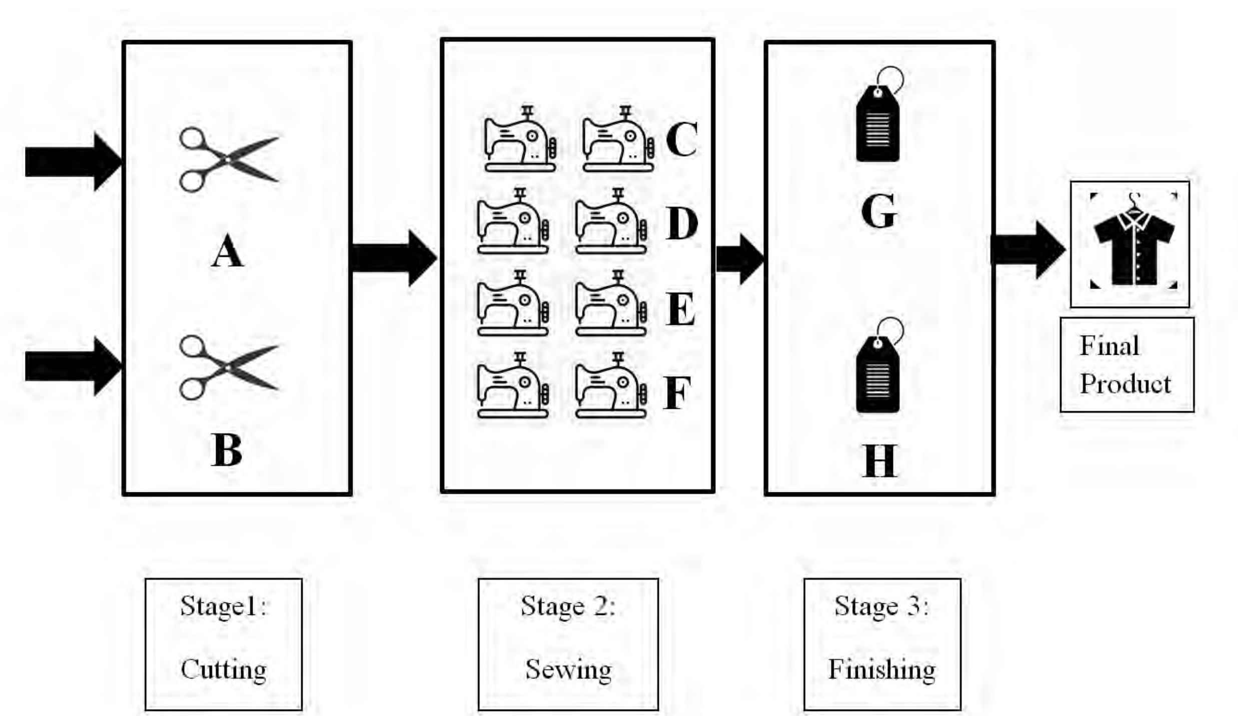

Figure 5.1: Hybrid flow Shop for textile industry

5.2 Job Classification Model Formulation

In this work, job scheduling is divided into two stages, where the first stage is to classify the jobs based on their importance level. For this purpose, a job classification model is developed by implementing Support Vector Machine. To develop the job classification model, at first support vector machine is trained using historical data. A sample of training data set used to develop the job classification model is given in **Appendix B.**

The features used in the training data set are-

i. The quantity of the order- As flow-shop is suitable for batch production, each job is assumed to have an order quantity. The quantity of each batch is considered as a feature of the training data set.
ii. The value of the job- This feature considers the price of each job, as each job.
iii. Production lead time available- This is the difference between the job creation date and the required delivery date of the product.
iv. Material Availability Status- The material availability status of a job is considered as a feature. As, if the material required for the production is not available, production cannot start and it should be scheduled later once material is available.
v. Line Running Status- This feature considers if any similar product is being produced on the production floor. If a similar product is already being produced, it means for that job operator does not require learning time which means it is beneficient to schedule that job.
vi. Customer- The customer who is placing the order has an impact on identifying the job priority level. As, if a job is created from a customer who is really important, by default the importance of that order will be more than other jobs.

Above mentioned six features are used to train SVM so that it can predict the importance level of any future job based on previous importance level of other jobs. This job classification help to identify jobs with higher priority. SVM classifies each job into three classes and each class has a priority index. For this model jobs are classified into below 3 classes with the respective priority index value.

Table 5.1: Job Classification and Priority Index

Class Name	Priority Index
Urgent	3
Moderate	2
Minor	1

For identifying a prediction model which is can predict the priority level of jobs based on the above-mentioned features different SVM models are checked by using MATLAB toolbox. Those models are:

i. Linear SVM- Uses linear Kernel
ii. Quadratic SVM- Uses quadratic Kernel
iii. Cubic SVM- Uses cubic Kernel
iv. Fine Gaussian SVM- Uses Gaussian Kernel with Kernel scale (√(No of features))⁄4
v. Medium Gaussian SVM- Uses Gaussian Kernel with Kernel scale √(No of features)
vi. Coarse Gaussian- Uses Gaussian Kernel with Kernel scale (√(No of features))*4

Table 5.2: Job to be Classified

Job ID	Ordered Quantity	Customer	Production Lead Time (days)	Value ($)	Line Running	Material Availability
971	462	Europe	18	2231	Yes	Available
77358	528	Europe	17	2550	Yes	Available
45388	528	Morocco	33	2550	Yes	Available
76526	2838	Europe	17	13708	Yes	Available
9493	390	Malaysia	15	1677	Yes	Available
46896	660	Europe	17	2838	Yes	Available
98804	2040	Europe	18	8772	Yes	Available
37304	5330	Europe	17	22919	Yes	Available
41555	700	Malaysia	96	3010	No	Not available
51144	1140	Russia	46	4902	Yes	Available
15068	700	China	79	5327	No	Not available
79878	1090	Europe	86	8295	No	Not available
21738	1150	Europe	114	8752	No	Not available
19376	1600	Europe	100	12176	No	Not available
6370	1620	Europe	93	12328	No	Not available
22216	1810	Europe	107	13774	No	Not available
22224	432	Europe	19	2575	Yes	Available
6324	432	Europe	102	2575	No	Not available
40919	2232	Europe	101	13303	Yes	Available
52303	3536	Europe	59	56116	Yes	Available

In **Table 5.3** the accuracy of different SVM models given. Six SVM models are checked to find the model which has the highest accuracy level. For this purpose, at first, models are checked with 5-fold cross-validation with PCA (principal component analysis). It can be seen that with PCA the model accuracy is less than models' without PCA. Furthermore, new models are checked by using a 10-fold cross-validation method. Finally, it is observed in **Table 5.3** that model-22 has the best accuracy level with a training time of 4.31 seconds at a speed of 630 observations per second.

Table 5.3: Prediction Accuracy Level of a Model

Model No	SVM type	PCA	Accuracy (%)	Cross-Validation
Model-1	Linear SVM	On	77.7	5-Fold
Model-2	Quadratic SVM	On	78.4	5-Fold
Model-3	Cubic SVM	On	60.8	5-Fold
Model-4	Fine Gaussian SVM	On	75	5-Fold
Model-5	Medium Gaussian SVM	On	77	5-Fold
Model-6	Coarse Gaussian	On	67.6	5-Fold
Model-7	Linear SVM	Off	85.1	5-Fold
Model-8	**Quadratic SVM**	**Off**	**91.2**	**5-Fold**
Model-9	**Cubic SVM**	**Off**	**93.9**	**5-Fold**
Model-10	Fine Gaussian SVM	Off	81.8	5-Fold
Model-11	Medium Gaussian SVM	Off	85.1	5-Fold
Model-12	Coarse Gaussian	Off	75.7	5-Fold
Model-13	**Cubic SVM**	**Off**	**92.6**	**5-Fold**
Model-14	Linear SVM	On	77.7	10- Fold
Model-15	Quadratic SVM	On	79.7	10- Fold
Model-16	Cubic SVM	On	60.8	10- Fold
Model-17	Fine Gaussian SVM	On	74.3	10- Fold
Model-18	Medium Gaussian SVM	On	77.7	10- Fold
Model-19	Coarse Gaussian	On	70.3	10- Fold
Model-20	Linear SVM	Off	85.8	10- Fold
Model-21	Quadratic SVM	Off	93.2	10- Fold
Model-22	**Cubic SVM**	**Off**	**95.3**	**10- Fold**
Model-23	Fine Gaussian SVM	Off	84.5	10- Fold
Model-24	Medium Gaussian SVM	Off	86.5	10- Fold
Model-25	Coarse Gaussian	Off	81.8	10- Fold
Model-26	**Cubic SVM**	**Off**	**92.6**	**10- Fold**

As model-22 has best accuracy level, this model is studied further to understand how features are co-related and impact of each feature in the job classification. Firstly, the confusion matrix for model-22 is given in **Figure 5.2** where it can be seen that the model can 100% correctly jobs belonging to “Minor” class and 90% and 94% accuracy for “Moderate” and “Urgent” class respectively.

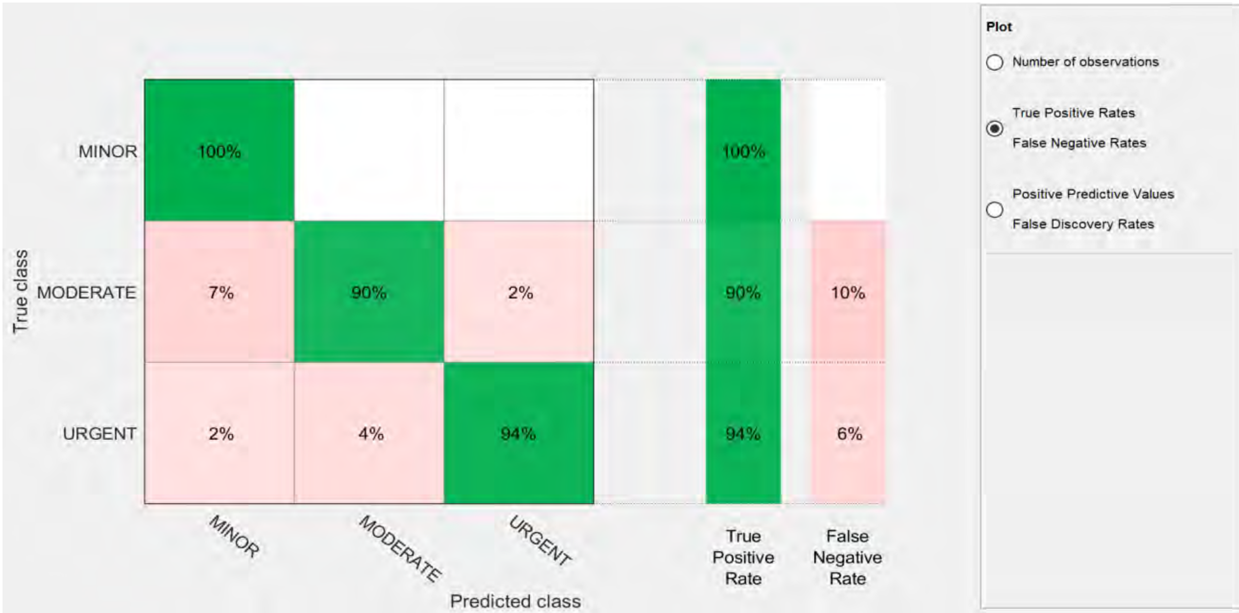

Figure 5.2: Confusion Matrix for Model-22

In **Figure 5.3**, **Figure 5.4**, **Figure 5.5**, **Figure 5.6**, **Figure 5.7**, **Figure 5.8** scatter plots are given which reflect which features can separate classes with less overlapping. From the scatter plot in **Figure 5.4** and **Figure 5.6**, it can be seen that order quantity, production lead time and product value these three features are more useful for job classification.

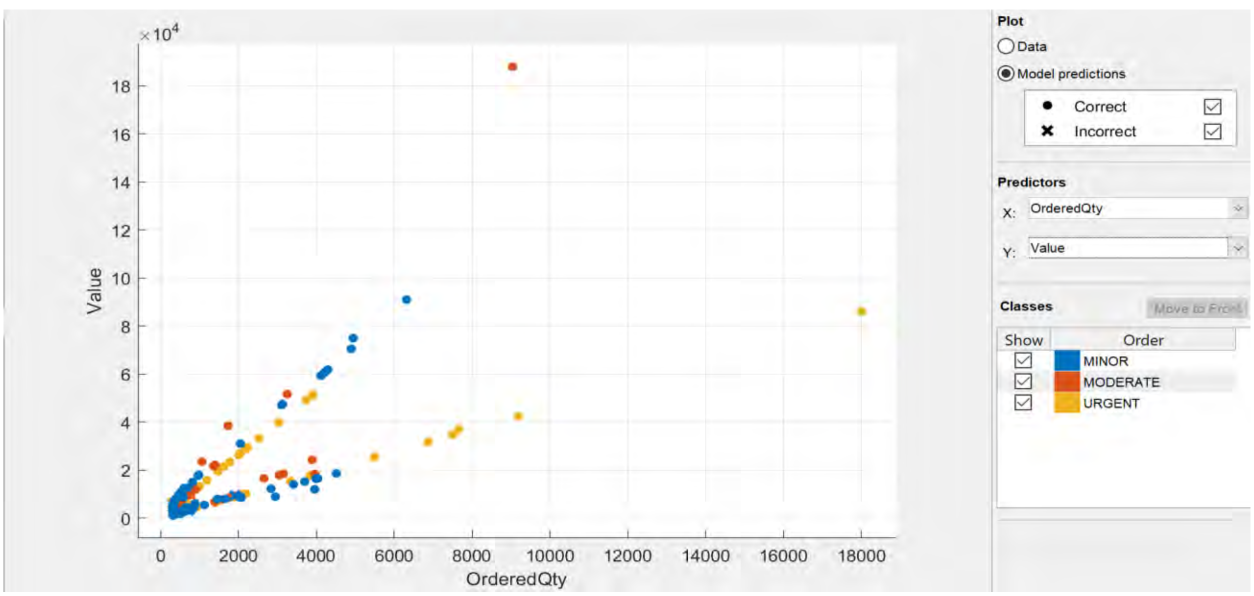

Figure 5.3: Scatter Plot for order value versus order quantity

In **Figure 5.3** it can be seen that by plotting predicted data with rest to features product value and order quantity of respective job ID, the model cannot separate “Minor”, “Moderate” and “Urgent” class with a certain boundary gap.

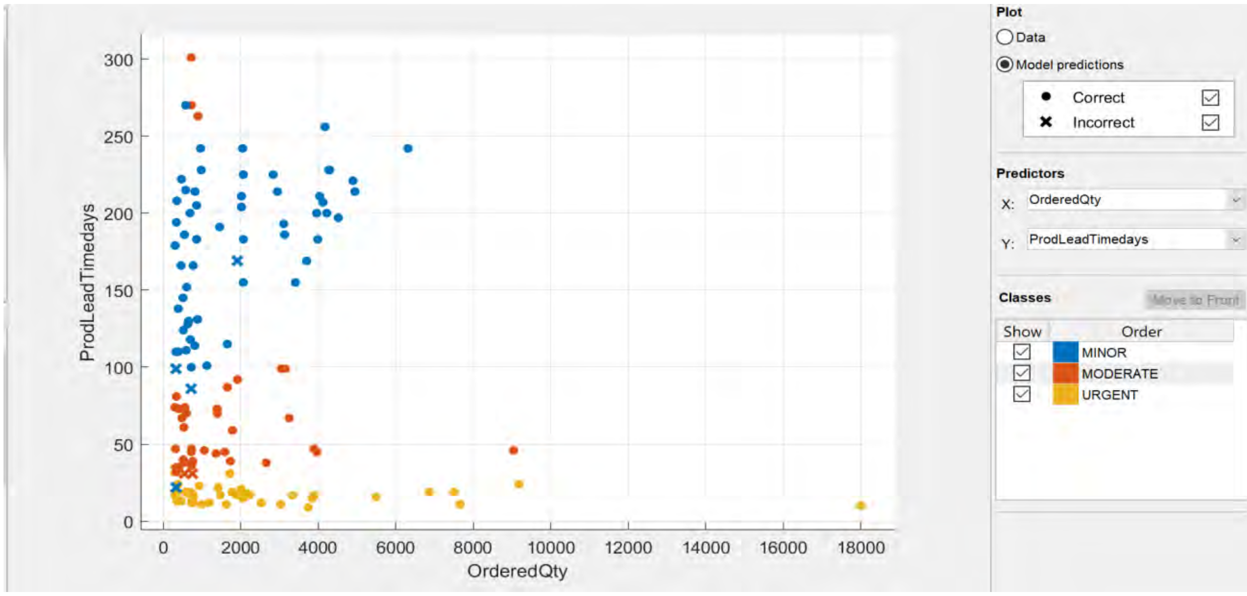

Figure 5.4: Scatter Plot for production lead time versus order quantity

Whereas, in **Figure 5.4** when data points are presented with respect to production lead time and order quantity, they are easily separable. So, it can be said that production lead time and order quantity have a greater impact on job classification.

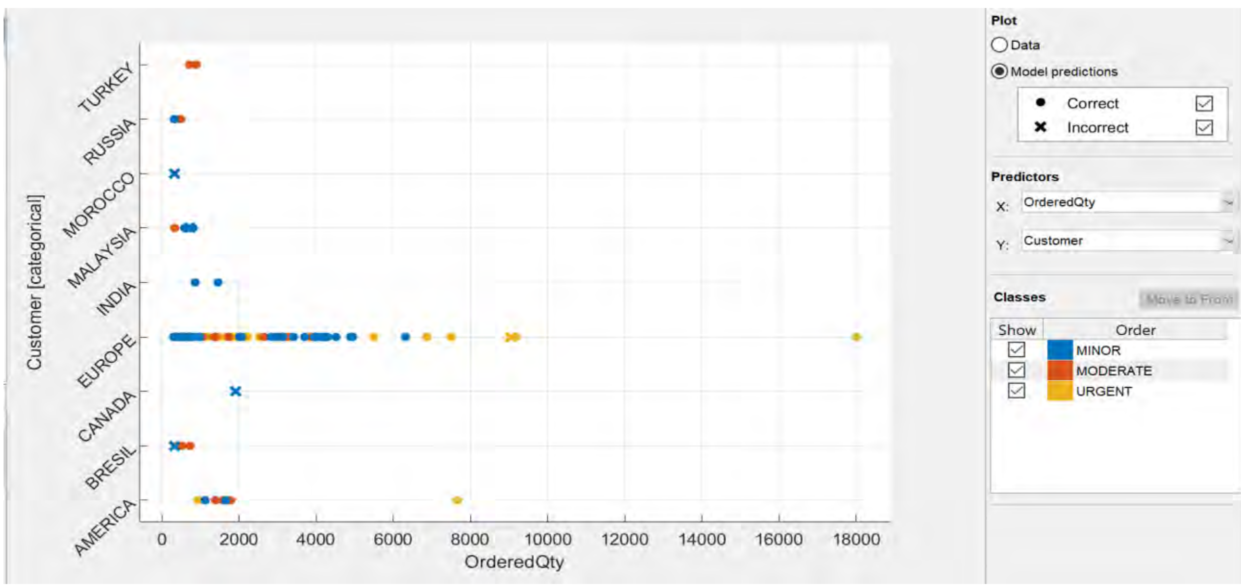

Figure 5.5: Scatter Plot for customer versus order quantity

From **Figure 5.5** it can be observed that when data points are presented with respect to customer and order quantity, one particular customer "Europe Zone" is placing more orders compared to others. However, this is not separating the classes in a visible way.

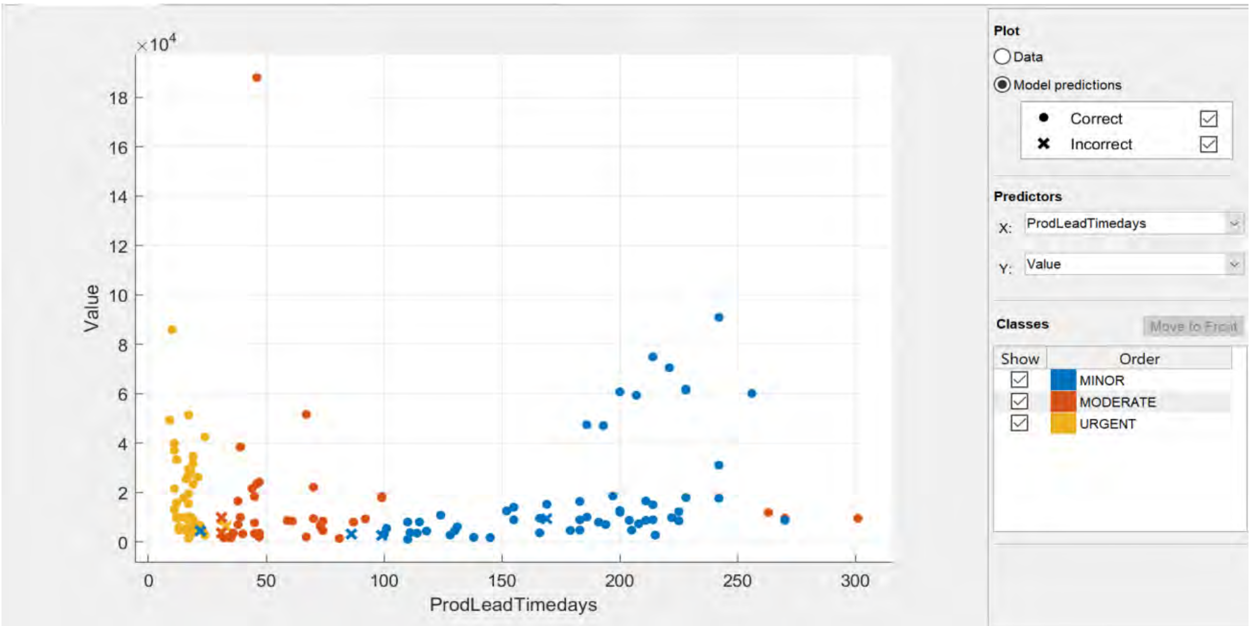

Figure 5.6: Scatter Plot for order value versus production lead time

Figure 5.6 illustrates that when data points are plot in a two-dimensional space with respect to product value and available lead time, these features can separate the classes in a better way.

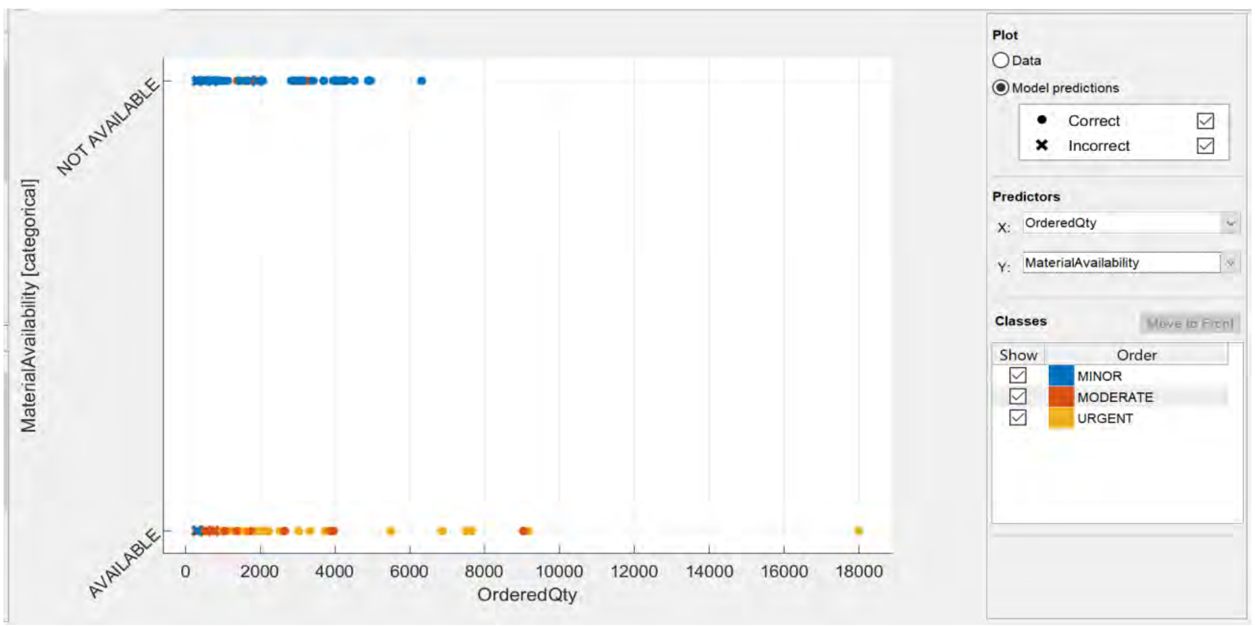

Figure 5.7: Scatter Plot for material availability versus order quantity

Figure 5.7 represents job classification with respect to material availability status and order quantity. It can be observed that irrespective to order quantity if the material is not available a job is considered to be in “Minor” class and when the material is available it can be both on urgent or moderate class based on its quantity.

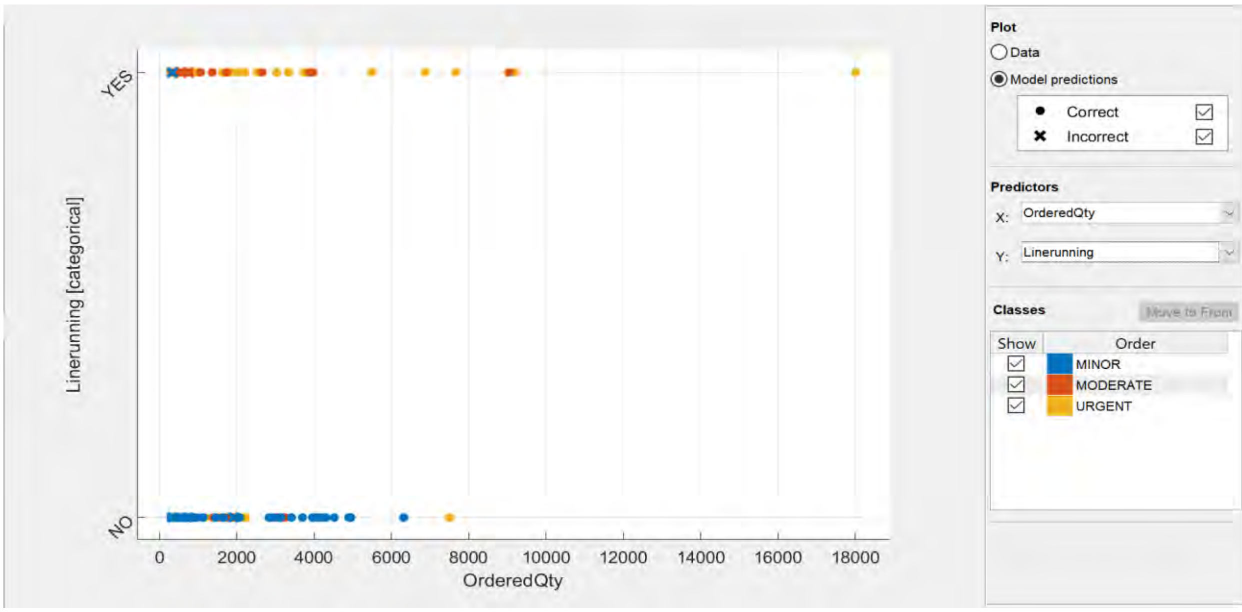

Figure 5.8: Scatter Plot for line running status versus order quantity

In **Figure 5.8** it can be observed that, if lines are running for a particular type of product, then any job corresponding to that type will be either in "Moderate" or "Urgent" class. Whereas, if line is not running, the predicted model tends to predict it in the "Minor" class.

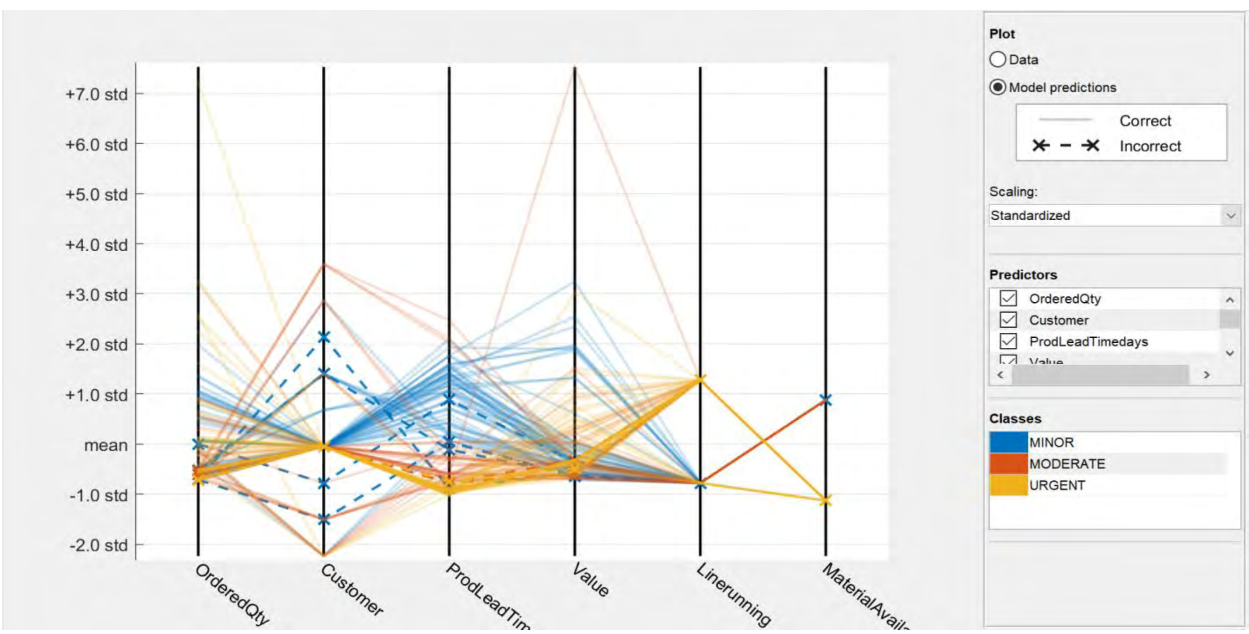

Figure 5.9: Parallel plot for Model-22

Figure 5.9 shows the parallel plot of all features which reflects how each feature has an impact on the classification model. In **Figure 5.9** blue lines, red lines and orange lines respectively represent “Minor”, “Moderate”, “Urgent” class. The area between the features, where these lines are easily separable have a greater impact on the classification model compared to other features. All parameters are standardized here.

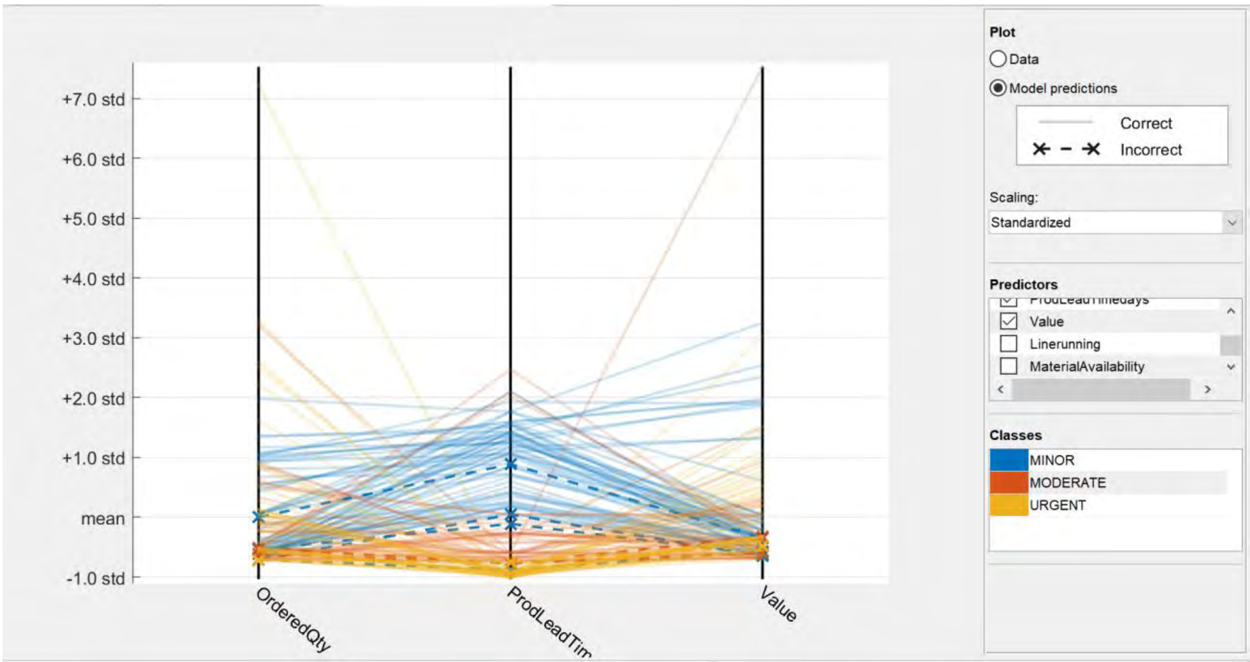

Figure 5.10: Parallel plot for Model-22 (modified)

In **Figure 5.10** the parallel plot in **Figure 5.9** is modified by keeping only three features those are order quantity, production lead time and order value, which separates the classes in a better way.

In **Appendix A** the job classification obtained using all 26 models is given. Among them, the prediction result of models - 8, 9, 13, 22 and 26 are given below as all these models have accuracy above 90%. It can be seen that all models have the same prediction for 19 out of 20 jobs. So, the prediction is considered to be consistent. Finally, the prediction of model-22 which is Cubic SVM with 10 fold cross-validation is used for job classification as it has the highest accuracy level among all models.

Table 5.4: Job Classification Model Comparison

Job ID	Model-8	Model-9	Model-13	Model-22	Model-26
971	Urgent	Urgent	Urgent	Urgent	Urgent
77358	Urgent	Urgent	Urgent	Urgent	Urgent
45388	Urgent	Urgent	Urgent	Urgent	Urgent
76526	Urgent	Urgent	Urgent	Urgent	Urgent
9493	Urgent	Urgent	Urgent	Urgent	Urgent
46896	Urgent	Urgent	Urgent	Urgent	Urgent
98804	Urgent	Urgent	Urgent	Urgent	Urgent
37304	Urgent	Urgent	Urgent	Urgent	Urgent
41555	Minor	Minor	Minor	Minor	Minor
51144	Moderate	Moderate	Moderate	Moderate	Moderate
15068	Minor	Minor	Minor	Minor	Minor
79878	Moderate	Moderate	Moderate	Moderate	Moderate
21738	Minor	Minor	Minor	Minor	Minor
19376	Moderate	Moderate	Moderate	Moderate	Moderate
6370	Moderate	Moderate	Moderate	Moderate	Moderate
22216	Moderate	Moderate	Minor	Moderate	Minor
22224	Urgent	Urgent	Urgent	Urgent	Urgent
6324	Minor	Minor	Minor	Minor	Minor
40919	Moderate	Moderate	Moderate	Moderate	Moderate
52303	Moderate	Moderate	Moderate	Moderate	Moderate

5.3 Uncertainty Considerations

The model developed considers some uncertainties related to the production floor, which may arise from machine breaks down or uncertain processing time. To deal with these uncertainties it requires below data preparations:

I. Calculating machine break down probability for each machine
II. Calculating the processing time of each job on each machine considering processing time as a stochastic element

5.3.1 Machine Break-down Probability Calculation

The probability of the break-down of a single machine can be calculated from the Eq. **(4.9)** given in section **4.2.1.2.** This empirical formula was derived by Al-Hinai & Elmekkawy, **[2011]** which is:

$$\rho_m = \frac{BT_m}{BT_{Total}} \tag{5.1}$$

Where,

ρ_m = Machine break down the probability of machine m

BT_m = Busy time of machine m

BT_{Total} = Busy time of all machines

The considered hybrid flow shop consists of 8 machines. Two machines (A, B) in stage 1, four machines (C, D, E, F) in stage 2 and two machines (G, H) in stage 3. As mentioned in section **5.1**, to avoid calculation complexity in stage 2 each line consisting of 15 separate machines is considered as a single machine. However, to calculate the machine break down probability on machines in stage 2, it requires two-step calculations.

In **Table 5.5** all busy times are given in unit of hours and for stage 2, the busy times of all machines are calculated separately to calculate the machine break down the probability of each machine in one particular line of stage 2, which is further considered as a single machine.

Table 5.5: Machine Busy Time

Stage-1															
A	47														
B	50														
Stage-2															
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
C	48	48	48	48	48	48	48	48	48	48	48	48	48	48	48
D	47	47	47	47	47	47	47	47	47	47	47	47	47	47	47
E	50	50	50	50	50	50	50	50	50	50	50	50	50	50	50
F	40	40	40	40	40	40	40	40	40	40	40	40	40	40	40
Stage-3															
G	45														
H	48														

By taking the summation of all machine's machine busy time, total machine busy time is

$BT_{Total} = 2965\ hours$

One example of a machine break-down the probability of machine A is shown below:

$\rho_A = \frac{BT_A}{BT_{Total}} = \frac{47}{2965} = 0.016$

In this same procedure machine, break-down probability of machine B, G, H can be calculated.

However, to calculate the machine break down the probability of lines C, D, E, F it requires calculating the machine break-down probability of each machine separately and taking the summation of all machines belonging to one line, to calculate the machine break-down probability of that line. In this case, the summation is taken because the probability of each machine break-down is mutually exclusive and the line cannot work if any of the machines face a breakdown.

Calculations of machine break down the probability of machine -1 of line C

$\rho_{C,1} = \frac{BT_{C,1}}{BT_{Total}} = \frac{48}{2965} = 0.017$

Following the same procedure, machine break down the probability of each machine inline C, D, E, F are given in **Table 5.6**

Table 5.6: Machine Break-down Probability in Stage-2

Machine No	C	D	E	F
1	0.017	0.017	0.018	0.014
2	0.017	0.017	0.018	0.014
3	0.017	0.017	0.018	0.014
4	0.017	0.017	0.018	0.014
5	0.017	0.017	0.018	0.014
6	0.017	0.017	0.018	0.014
7	0.017	0.017	0.018	0.014
8	0.017	0.017	0.018	0.014
9	0.017	0.017	0.018	0.014
10	0.017	0.017	0.018	0.014
11	0.017	0.017	0.018	0.014
12	0.017	0.017	0.018	0.014
13	0.017	0.017	0.018	0.014
14	0.017	0.017	0.018	0.014
15	0.017	0.017	0.018	0.014
Total	0.255	0.255	0.27	0.21

In stage-2, as all machines are in series in each line, so the probability that there will be a break-down in a line is the summation of the probability of machine break down of every single machine, which can be given by below equation.

$$\rho_{Ņ} = \sum_{i=1}^{r} \rho_{Ņ,i} \qquad \textbf{(5.2)}$$

Where, $Ņ = C, D, E, F$

r= machine number in corresponding line

Table 5.7: Machine Break-down Probability

Machine No	Machine break-down probability	Machine No	Machine break-down probability
A	0.016	E	0.27
B	0.017	F	0.21
C	0.255	G	0.015
D	0.255	H	0.016

This machine break-down probability is multiplied by the repair time of the corresponding machine. Further, this additional time for machine break-down is added with the processing time of all jobs, done on that machine.

5.3.2 Processing Time Calculation

The processing time of different operations of a job in different machines are not deterministic values, there are stochastic. Processing time may depend on various external factors such as the operator's skill level, machine condition and so on. So, in this work, we have tried to address these uncertainties by using a formulation proposed by Janak, Lin, & Floudas, **[2007]**, where the real value of processing time is presented in terms nominal values as following Eq. (**5.3**):

$$P_m = (1 + \varphi.\varepsilon).P_m^{\sim} \quad \textbf{(5.3)}$$

In Eq. (**5.3**), φ is a given uncertainty level and in this case, we have considered 30% of uncertainty so, $\varphi = 0.3$ is used for future calculations. In Eq. (**5.3**), ε is a random number that follows normal distribution within an interval of [-1, 1].

The value of processing time will defer based on learning curve. If the operator is skilled in a particular job, in that case, the operator will require less time compared to the operator who is new to that job. This can have a vital effect on processing time. This work aims to calculate processing time considering two scenarios: processing time with learning curve and processing time without learning curve.

5.3.2.1 Processing Time without Learning Curve

In textile industries processing time of a particular job is calculated using SAM (standard allocated minutes). In this work SAM is used to calculate the daily output of a particular product, further the value of output per day of a product is used to calculate the processing time of a particular job. The formula used to calculate the daily output of a product is given below:

$$Per\ day\ output = \frac{(No\ of\ operator \times Working\ hour\ perday \times 60 \times Efficiency)}{SAM} \quad \textbf{(5.4)}$$

A sample calculation for calculating per day output is of job ID- 98804 for a particular machine is given below:

SAM= 22.90
No of operator = 20
Working hour per day = 8hours
Efficiency = 60%

$$Per\ day\ output = \frac{(20 \times 8 \times 60 \times 0.6)}{22.90} = 252$$

Using Eq. **(5.4)**, per day output will be 252 Pieces and the total order quantity was 2040. So, for that particular machine processing time of job ID- 98804 will be 9 days. This processing time is used in Eq. **(5.3)**, to have a stochastic processing time. For this purpose, this processing time 9days is used as nominal value and 100 values are generated using random variables with normal distribution. Finally, the mean value of those hundred observations is considered as processing time that will be a stochastic value. For each job in all 8 machines processing time is calculated following this method. The table of processing time is given in **Appendix C.**

5.3.2.2 Processing Time with Learning Curve

Processing time considering learning curve is greater than processing time without learning curve, as skilled operators require less time to finish an operation compared to a new operator. Calculating the processing time considering learning curve is the same as calculating processing time without learning curve. However, processing time with learning curve considers the additional time for changing the machine set up and training an operator about the operation.

Considering learning curve effect when processing time of job ID- 98804 is calculated it is increased to 16days for that same machine if the operator is new. While calculating processing time with learning curve, Eq. **(5.3**), is used to determine the stochastic value.

Whether a machine is suitable for a job or the operators are skilled about that particular job, this information was collected from the production floor and below matrix is formed.

Table 5.8: Machine Suitability Matrix

		Machines							
		A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
Job ID	**9493**	1	1	0	1	1	0	1	1
	77358	1	1	1	0	0	1	1	1
	76526	1	1	1	0	0	1	1	1
	46896	1	1	0	1	1	0	1	1
	37304	1	1	0	1	1	0	1	1
	971	1	1	1	0	0	1	1	1
	98804	1	1	0	1	1	0	1	1
	22224	1	1	0	1	1	0	1	1
	45388	1	1	1	0	0	1	1	1
	51144	1	1	0	1	1	0	1	1
	52303	1	1	1	0	0	1	1	1
	79878	1	1	0	0	0	0	1	1
	6370	1	1	0	0	0	0	1	1
	19376	1	1	0	0	0	0	1	1
	40919	1	1	0	1	1	0	1	1
	22216	1	1	0	0	0	0	1	1
	15068	1	1	0	0	0	0	1	1
	41555	1	1	0	1	1	0	1	1
	6324	1	1	0	1	1	0	1	1
	21738	1	1	0	0	0	0	1	1

By using the Machine Suitability Matrix processing time of each job on each machine can be determined based on the effect of learning curve. The below-given formulation is used for this calculation.

$$P_{im}^{o} = Y_{im}.P_{im} + (1 - Y_{im}).P_{im}^{L} \qquad \textbf{(5.5)}$$

In Eq **(4.10**), Y_{im} is the machine suitability matrix, P_{im} is the processing time without learning curve and P_{im}^{L} is processing time with learning curve for a particular job *i* and machine *m*. If machine m is suitable job *i* processing time without learning curve will be considered else processing time with learning curve will be considered for that job machine pair. Finally after adding machine break-down repair time with Eq. **(4.10**), final processing time is calculated and that is used for scheduling model.

$$P_{im}^{o} = Y_{im}.P_{im} + (1 - Y_{im}).P_{im}^{L} + \rho_{m}.RT_{m} \qquad \textbf{(5.6)}$$

By using Eq. **(4.11**), final processing time for job set is given below:

Table 5.9: Processing Time

		Machines							
		A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
Job ID	**9493**	0.5	0.5	6.1	2.1	2.1	7.2	1.1	1.1
	77358	0.6	0.6	3.1	7	7.9	3.1	1.1	1
	76526	2.1	2.6	15	22	22	14.1	4.1	4.1
	46896	0.6	0.6	7	3.1	3	8.1	1	1
	37304	3.1	3.5	32.1	22.2	20.2	27.8	4	4.1
	971	0.5	0.5	3	7	8.1	3	1	1
	98804	1.1	1	16.1	8.9	8	15.8	2	2
	22224	0.6	0.5	7	3.1	3	8.1	1	1.1
	45388	0.5	0.5	3.1	7	8	3.1	1	1.1
	51144	0.6	0.5	9.1	5.1	5	10.1	1	1
	52303	3.1	3.5	28.8	32.1	28.1	26	4	4
	79878	1	1	15.9	16.1	16.2	16.1	1.5	1.6
	6370	1	1.1	19.9	20	20.2	20	1.5	1.6
	19376	1	1	20.3	19.6	20.2	19.8	1.5	1.5
	40919	1	1.1	18.9	12.2	12.1	20.2	1.6	1.5
	22216	1.1	1.1	20.9	21.1	21.5	20.9	1.5	1.6
	15068	1	1	13.1	13.1	10.2	10.1	1.6	1.6
	41555	0.6	0.5	7	3.1	3	7.9	1	1
	6324	0.5	0.5	7	3.1	3.1	8.2	1.1	1
	21738	1.1	1.1	15.9	16.3	17.1	17.2	1.6	1.6

5.4 Job Scheduling

After job classification according to their priority level and considering uncertainty in processing time, these inputs are used for the job scheduling model. Jobs belonging to the same class are sequenced based on their delivery date. Jobs with earlier delivery dates are sequenced prior to jobs with subsequent delivery dates. Finally, jobs are sequenced as below position:

Table 5.10: Job Sequencing

Job ID	Priority Level	Order Sequence	Job ID	Priority Level	Order Sequence
9493	Urgent	1	52303	Moderate	11
77358	Urgent	2	79878	Moderate	12
76526	Urgent	3	6370	Moderate	13
46896	Urgent	4	19376	Moderate	14
37304	Urgent	5	40919	Moderate	15
971	Urgent	6	22216	Moderate	16
98804	Urgent	7	15068	Minor	17
22224	Urgent	8	41555	Minor	18
45388	Urgent	9	6324	Minor	19
51144	Moderate	10	21738	Minor	20

In the job scheduling model, the job index i will be the associated sequence number of the job, for solving the model Job ID is not considered further.

5.4.1 Job Completion Time

The scheduling model developed in Chapter **4** involves a calculation of job completion times prior to implementing the model.

<u>Completion in Stage-1 (Cutting)</u>

The cutting stage has two machines A and B. So, when jobs are in the cutting stage, the completion time of each job follows below formula:

$$C_{i1} = \sum_{k=1}^{k=2}\{(\sum_{i=1}^{20} X_{i1k}.P_{i1k}^{o}).X_{i1k}\} \qquad \textbf{(5.7)}$$

$$\forall\ i \in \{1,2,3,\ldots\ldots 20\}$$

In Eq. (**4.2**), the value of P_{i1k}^{o} is a matrix which is processing time of each job in machines A and B, these values are used from **Table 5.9**.

Completion in Stage-2 (Sewing)

It is seen from **Table 5.9** that the mean sewing time, which is the processing time of stage-2 is greater than the processing time of stage-1. So, the time required for a job to complete the sewing stage is given by the below formula as mentioned in section **4.2.1.1**.

$$C_{12} = \sum_{k=1}^{k=4}(C_{11} + X_{12k}.P_{12k}^{o}).X_{12k} \quad \textbf{(5.8)}$$

$$C_{i2} = \sum_{k=1}^{k=4}[max\{(C_{1,1}.X_{1,2,k}),(C_{2,1}.X_{2,2,k}),\dots\dots(C_{i-1,1}.X_{i-1,2,k})\} + max(C_{i,1}, \sum_{i=1}^{i=i-1}X_{i2k}.P_{i2k}^{o}) + X_{i2k}.P_{i2k}^{o}].X_{i2k} \quad \textbf{(5.9)}$$

$\forall$ i ϵ {2,3,………20}

Completion in Stage-3 (Finishing)

The mean completion time of the finishing stage is smaller than the mean completion time of the sewing stage. So, the job completion time of the finishing stage is calculated using the below formula as mentioned in section **4.2.3.**

$$C_{i3} = \sum_{k=1}^{k=2}\{max(C_{i,2}, \sum_{i=1}^{i=i-1}X_{i3k}.P_{i3k}^{o}) + \left(\sum_{i=1}^{i=i}X_{i3k}.P_{i3k}^{o}\right)\}.X_{i3k} \quad \textbf{(5.10)}$$

$\forall$ i ϵ {1,2,………N}

$\forall$j ϵ {2,3,…........ L}

Eq. **(4.3)**, Eq. **(4.4)** and Eq. **(4.5)**, are used for the calculation of make-span and tardiness.

5.4.2 Efficiency Calculation

To calculate efficiency from the concept of minimizing machine idle time, processing time with a learning curve and without learning curve given in **Appendix D** and machine suitability matrix is given in **Table 5.8** is used. Efficiency is calculated by the below equation:

$$\sum_{i=1}^{i=20}\sum_{j=1}^{j=3}\sum_{k=1}^{k=M_j}X_{ijk}.\left(1-Y_{ijk}\right).(P_{ijk}^{L} - P_{ijk} + T_{set\,up}) \quad \textbf{(5.11)}$$

Consequently, the value of all three objectives functions are calculated and constrained multi-objective PSO algorithm is used to find the optimum jib assignment matrix $\boldsymbol{X_{ijk}}$.

CHAPTER 6: RESULT AND ANALYSIS

6.1 Job Assignments

After implementing the model in a textile industry mentioned in **Chapter 5,** the optimum scheduling result can be obtained. This model aims to optimize three objectives which are minimizing make-span, minimizing tardiness and maximize efficiency by minimizing machine idle time or the time required for a learning curve. As this model is a multi-objective optimization problem, to solve it, the weighted aggregation approach is used. While using the weighted aggregation approach, the weight of each objective is changed 12 times so that for different importance levels of a particular objective, the corresponding job schedule can be obtained. The result summary is given in **Table 6.1**, where make-span, tardiness, efficiency, and value of the objective function are given in days.

Table 6.1: Function Value

Combination No	Weight of Make-span	Weight of Tardiness	Weight of Efficiency	Make-span	Tardiness	Efficiency	Objective Function
1	0.5	0.2	0.3	98	270	73.8	137.64
2	0.5	0.3	0.2	89.3	261.6	68.2	136.77
3	0.3	0.5	0.2	113.7	200	62.6	176.18
4	0.3	0.2	0.5	101.6	267.7	57.7	112.87
5	0.2	0.5	0.3	109.8	181.95	52.1	128.565
6	0.2	0.3	0.5	110.2	199.8	64	113.98
7	0.6	0.3	0.1	90	153.2	52.7	105.23
8	0.6	0.1	0.3	89.89	183.3	64.4	91.584
9	0.3	0.6	0.1	110.7	216	77.4	170.55
10	0.3	0.1	0.6	134.5	210	57.7	113.97
11	0.1	0.6	0.3	101	290.8	72.6	196.26
12	0.1	0.1	0.6	110.3	263.7	82.6	75.93

For combination 1, the job assignment is given in **Table 6.2**, and in each stage job assignments to each machine are illustrated in **Figure 6.1**, **Figure 6.2** and **Figure 6.3** respectively. In **Figure 6.4**, the PSO curve obtained for this combination is shown, where the best function value obtained is 137.64 days.

Table 6.2: Job Allocation (Combination-1)

		Machine No							
Job ID	Sequence	A	B	C	D	E	F	G	H
9493	1	0	1	0	0	0	1	1	0
77358	2	0	1	1	0	0	0	1	0
76526	3	0	1	0	0	0	1	0	1
46896	4	1	0	1	0	0	0	0	1
37304	5	0	1	1	0	0	0	1	0
971	6	1	0	1	0	0	0	1	0
98804	7	1	0	0	1	0	0	1	0
22224	8	1	0	0	0	1	0	1	0
45388	9	0	1	0	0	1	0	0	1
51144	10	0	1	0	0	1	0	0	1
52303	11	1	0	1	0	0	0	0	1
79878	12	1	0	0	0	1	0	1	0
6370	13	1	0	0	0	0	1	0	1
19376	14	0	1	0	0	0	1	0	1
40919	15	0	1	0	1	0	0	0	1
22216	16	1	0	0	0	1	0	1	0
15068	17	0	1	0	0	1	0	0	1
41555	18	1	0	0	1	0	0	1	0
6324	19	1	0	0	0	0	1	0	1
21738	20	0	1	0	1	0	0	1	0

From **Table 6.1**, it can be that this job assignment is obtained by giving the lowest weight to tardiness. By comparing the required delivery date of each order with its completion time it can be observed that jobs in **Table 6.3** cannot be completed before the due date. So, it will be the decision of the producer whether he needs to increase his capacity to complete them within the due date or needs to cancel those orders.

Table 6.3: List of delay jobs (Combination -1)

Job ID	Sequence	Delay	Job ID	Sequence	Delay
76526	3	8.4	98804	7	2.3
37304	5	32.5	52303	11	33.4
971	6	41.6	15068	17	11.3

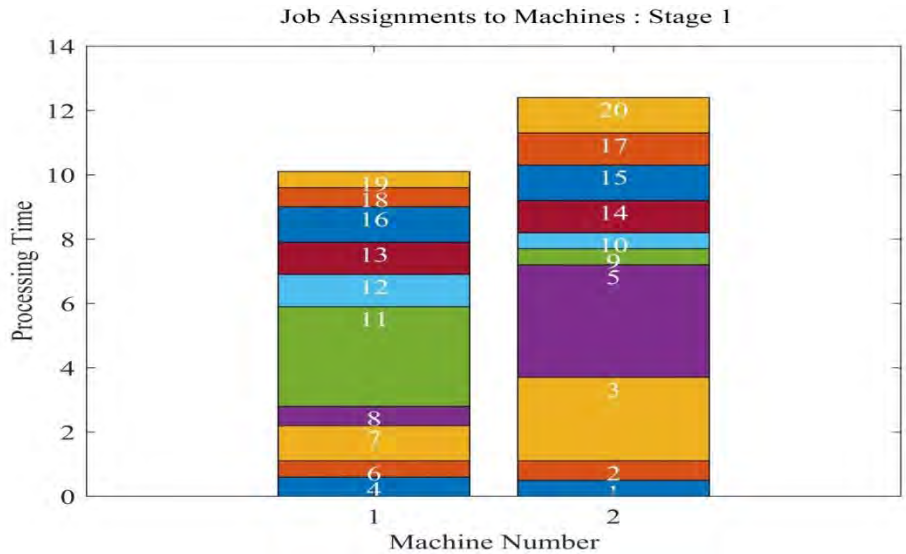

Figure 6.1: Job allocation in stage-1 (Combination-1)

Figure 6.1 illustrates that according to job allocation from combination-1, in first stage the machine-1 (A) will process jobs in sequence- 4, 6, 7, 8, 11, 12, 13, 16, 18, 19 in the given order and the machine-2 (B) will process jobs in sequence – 1, 2, 3, 5, 9, 10, 14, 15, 17, 20 sequentially where job in sequence 20 finishes stage-1 as the last job.

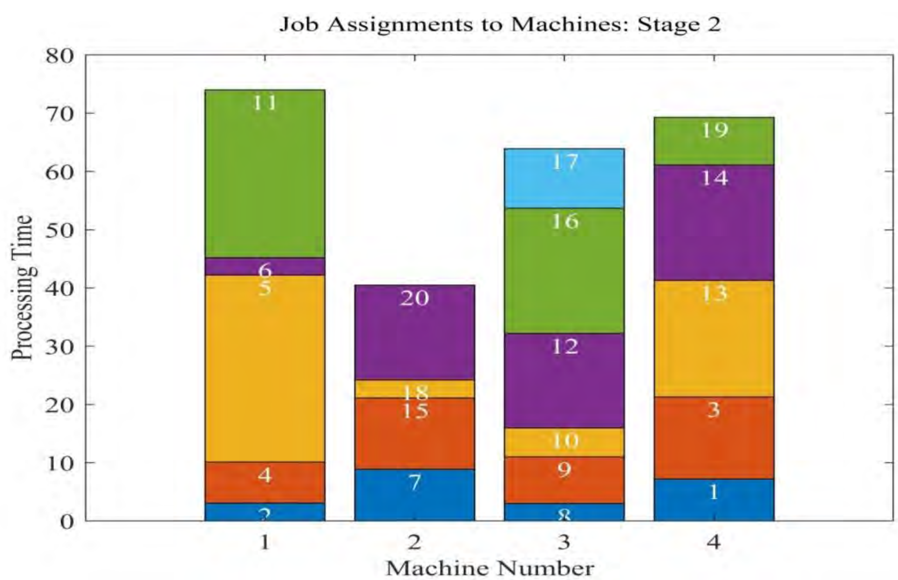

Figure 6.2: Job allocation in stage-2 (Combination-1)

Figure 6.2 illustrates that according to job allocation from combination-1, in first stage the machine-1 (C) will process jobs in sequence- 2, 4, 5, 6, 11 in the given order; the machine-2 (D) will process jobs in sequence – 7, 15, 18, 20; the machine-3 (E) will process jobs in sequence – 8, 9, 10, 12,16,17; the machine-4 (F) will process jobs in sequence – 1, 3, 13, 14, 19 sequentially.

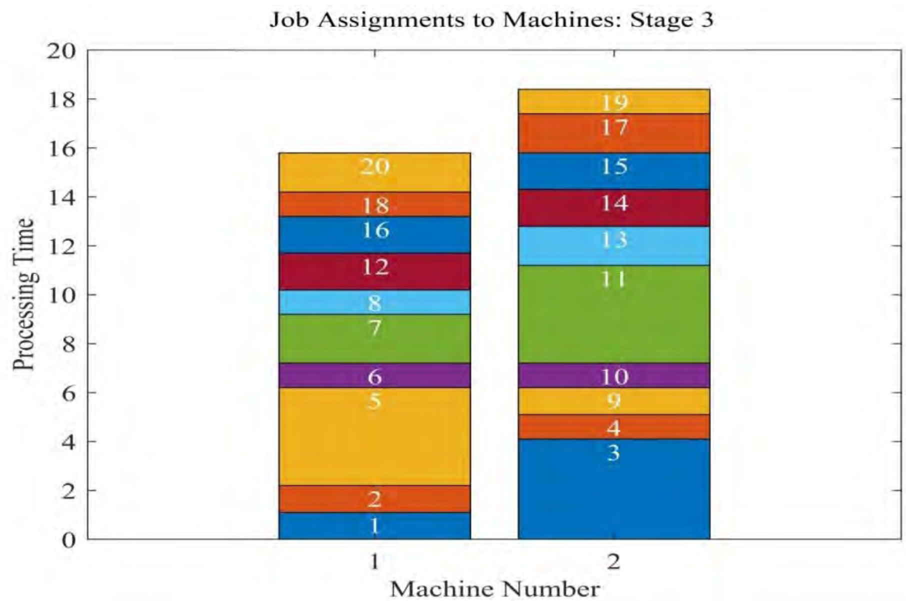

Figure 6.3: Job allocation in stage-3 (Combination-1)

Figure 6.3 illustrates that according to job allocation from combination-1, in first stage the machine-1 (G) will process jobs in sequence- 1, 2, 5, 6, 7, 8, 12, 16, 18, 20 in the given order and the machine-2 (H) will process jobs in sequence – 3, 4, 9, 10, 11, 13, 14, 15, 17, 19 sequentially.

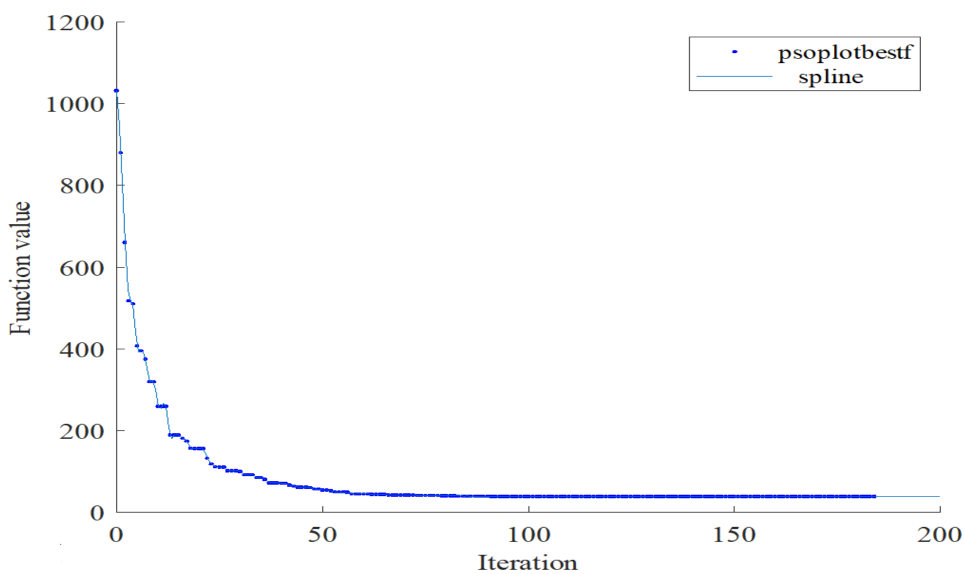

Figure 6.4: PSO Plot (Combination-1)

Figure 6.4 illustrates the PSO curve, where the value of the objective function is decreasing with each iteration and finally at iteration no 180 the algorithm terminates as there is no improvement in objective function value.

For combination 2, the job assignment is given in **Table 6.4**, and in each stage job assignments to each machine are illustrated in respectively **Figure 6.5**, **Figure 6.6** and **Figure 6.7**. In **Figure 6.8**, the PSO curve obtained for this combination is shown, where the best function value obtained is 136.7 days.

Table 6.4: Job Allocation (Combination-2)

					Machine No				
	Sequence	A	B	C	D	E	F	G	H
9493	1	1	0	1	0	0	0	1	0
77358	2	0	1	0	0	1	0	0	1
76526	3	1	0	0	1	0	0	0	1
46896	4	1	0	1	0	0	0	0	1
37304	5	0	1	0	0	1	0	1	0
971	6	1	0	0	0	0	1	0	1
98804	7	1	0	0	1	0	0	0	1
22224	8	1	0	0	0	1	0	0	1
45388	9	0	1	1	0	0	0	1	0
51144	10	1	0	0	1	0	0	0	1
52303	11	1	0	0	0	0	1	0	1
79878	12	0	1	1	0	0	0	1	0
6370	13	0	1	0	1	0	0	1	0
19376	14	0	1	0	0	1	0	0	1
40919	15	1	0	0	0	1	0	0	1
22216	16	0	1	0	0	0	1	0	1
15068	17	0	1	1	0	0	0	0	1
41555	18	1	0	1	0	0	0	1	0
6324	19	0	1	0	1	0	0	0	1
21738	20	1	0	0	0	0	1	1	0

Job ID

From **Table 6.1**, it can be that this job assignment is obtained by giving second importance to tardiness. By comparing the required delivery date and completion time of each job, job IDs mentioned in **Table 6.3**, cannot be completed before the due date.

Table 6.5: List of delay jobs (Combination -2)

Job ID	Sequence	Delay	Job ID	Sequence	Delay
76526	3	12.7	98804	7	24.6
46896	4	2.7	22224	8	26.4
37304	5	16.8	51144	10	6

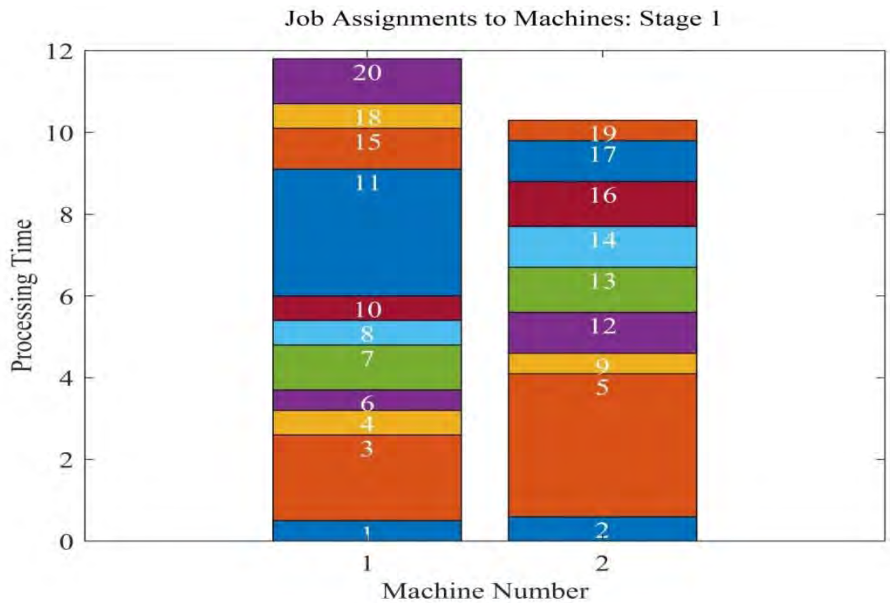

Figure 6.5: Job allocation in stage-1 (Combination-2)

Figure 6.5 illustrates that according to job allocation from combination-2, in first stage the machine-1 (A) will process jobs in sequence- 1, 3, 4, 6, 7, 8, 10, 11, 15, 18, 20 in the given order and the machine-2 (B) will process jobs in sequence – 2, 5, 9, 12, 13, 14, 16, 17, 19 sequentially.

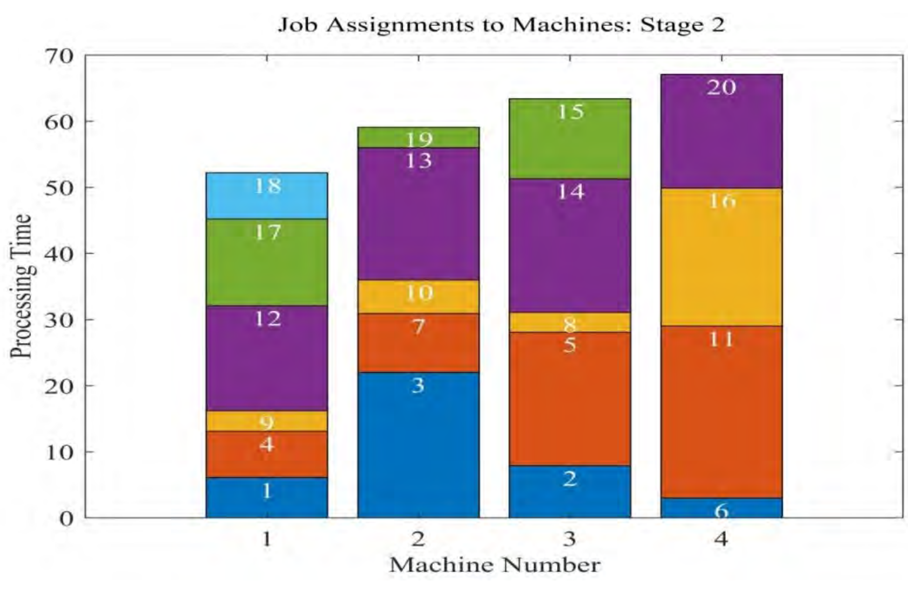

Figure 6.6: Job allocation in stage-2 (Combination-2)

Figure 6.6 illustrates that according to job allocation from combination-2, in first stage the machine-1 (C) will process jobs in sequence- 1, 4, 9, 12, 17, 18 in the given order; the machine-2 (D) will process jobs in sequence – 3, 7, 10, 13, 19; the machine-3 (E) will process jobs in sequence – 2, 5, 8, 14, 15; the machine-4 (F) will process jobs in sequence – 6, 11, 16, 20 sequentially.

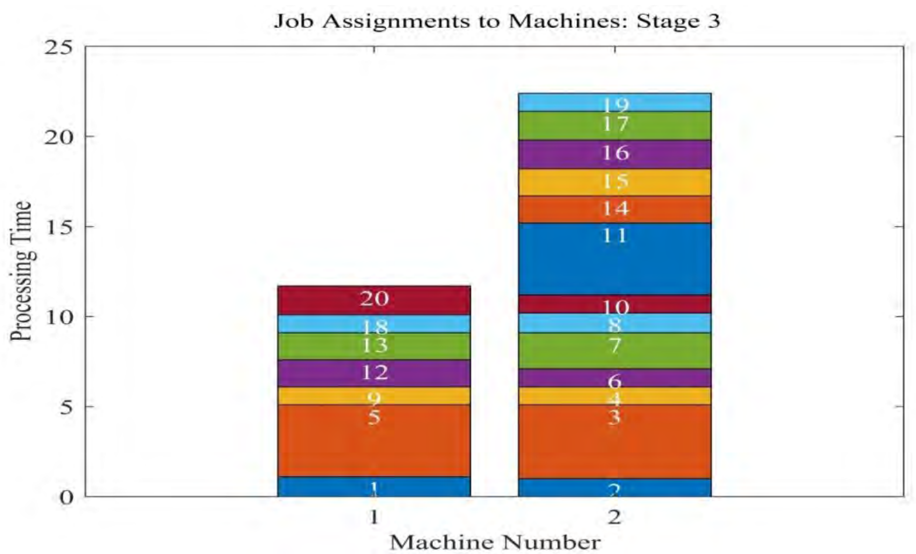

Figure 6.7: Job allocation in stage-3 (Combination-2)

Figure 6.7 illustrates that according to job allocation from combination-1, in first stage the machine-1 (G) will process jobs in sequence- 1, 5, 9, 12, 13, 18, 20 in the given order and the machine-2 (H) will process jobs in sequence – 2, 3, 4, 6, 7, 8, 10, 11, 14, 15, 16, 17, 19.

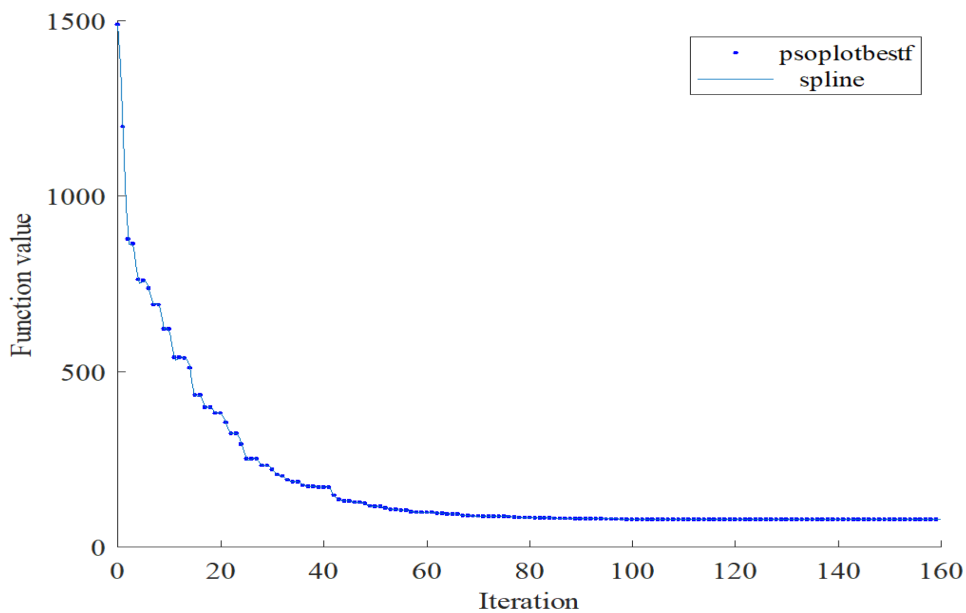

Figure 6.8: PSO Plot (Combination-2)

Figure 6.8 illustrates the PSO curve with the decreasing value of the objective function with each iteration and finally at iteration no 160 the algorithm terminates as there is no improvement in objective function value.

For combination 3, the job assignment is given in **Table 6.6**, and in each stage job assignments to each machine are illustrated respectively in **Figure 6.9**, **Figure 6.10** and **Figure 6.11**. **Figure 6.12**, represents the corresponding PSO curve with the best function value obtained of 176.18 days.

Table 6.6: Job Allocation (Combination-3)

		Machine No							
Job ID	Sequence	A	B	C	D	E	F	G	H
9493	1	0	1	0	1	0	0	1	0
77358	2	1	0	1	0	0	0	1	0
76526	3	0	1	0	1	0	0	1	0
46896	4	1	0	0	0	0	1	0	1
37304	5	0	1	0	1	0	0	1	0
971	6	0	1	1	0	0	0	0	1
98804	7	0	1	0	0	1	0	0	1
22224	8	0	1	1	0	0	0	0	1
45388	9	0	1	1	0	0	0	1	0
51144	10	1	0	0	0	1	0	0	1
52303	11	0	1	0	0	0	1	0	1
79878	12	1	0	0	1	0	0	0	1
6370	13	1	0	0	1	0	0	1	0
19376	14	1	0	0	0	1	0	1	1
40919	15	1	0	0	0	1	0	0	1
22216	16	1	0	0	0	0	1	1	0
15068	17	1	0	0	0	0	1	0	1
41555	18	1	0	0	0	0	1	1	1
6324	19	1	0	0	0	1	0	0	1
21738	20	0	1	1	0	0	0	0	1

From **Table 6.1**, it can be that this job assignment is obtained by giving first priority to tardiness. By comparing the required delivery date and completion time of each job, job IDs mentioned in **Table 6.7**, cannot be completed before the due date.

Table 6.7: List of delay jobs (Combination -3)

Job ID	Sequence	Delay	Job ID	Sequence	Delay
76526	3	14.9	45388	9	3.1
37304	5	42.7	6370	13	8.8
98804	7	2.1	15068	17	9
22224	8	8.8	41555	18	17.7

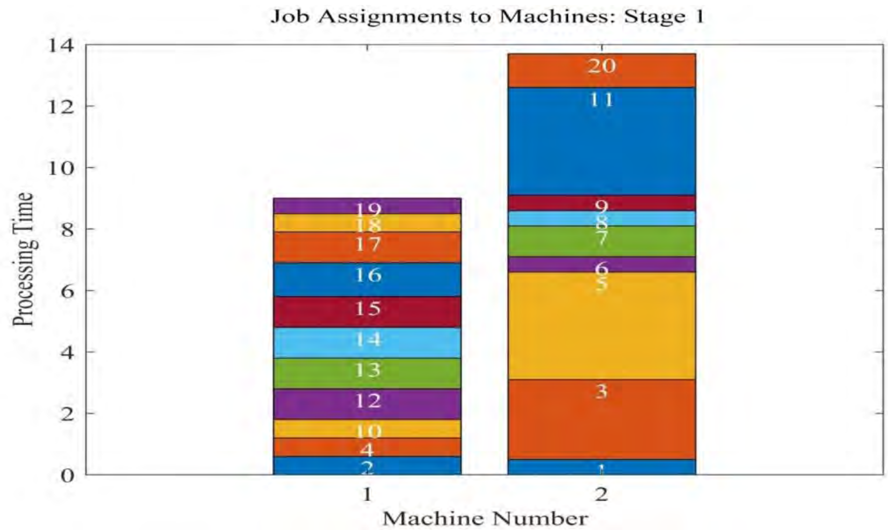

Figure 6.9: Job allocation in stage-1 (Combination-3)

Figure 6.9 illustrates that according to job allocation from combination-1, in first stage the machine-1 (A) will process jobs in sequence- 2, 4, 10, 12, 13, 14, 15, 16, 17, 18, 19 in the given order and the machine-2 (B) will process jobs in sequence – 1, 3, 5, 6, 7, 8, 9, 11, 20 sequentially.

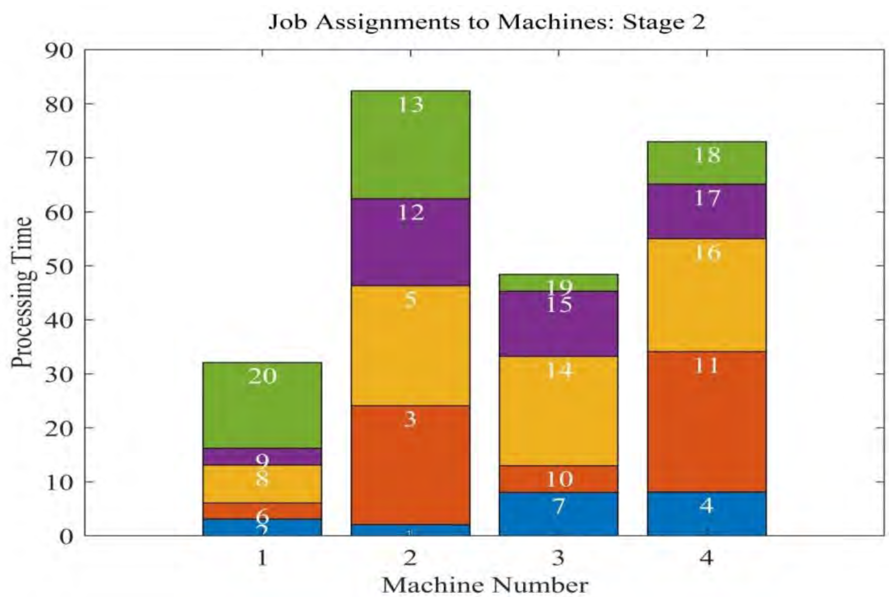

Figure 6.10: Job allocation in stage-2 (Combination-3)

Figure 6.10 illustrates that according to job allocation from combination-1, in first stage the machine-1 (C) will process jobs in sequence- 2, 6, 8, 9, 20 in the given order; the machine-2 (D) will process jobs in sequence – 1, 3, 5, 12, 13; the machine-3 (E) will process jobs in sequence – 7, 10, 14, 15, 19; the machine-4 (F) will process jobs in sequence – 4, 11, 16, 17, 18 sequentially.

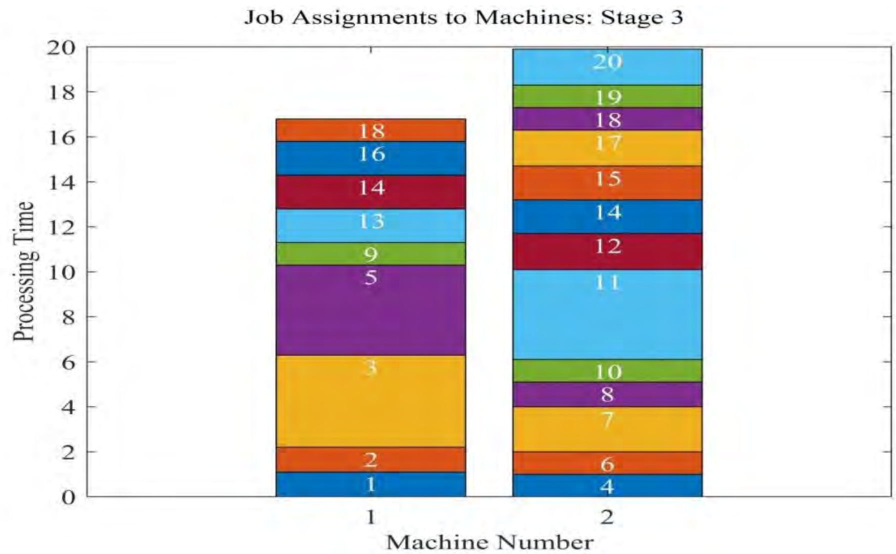

Figure 6.11: Job allocation in stage-3 (Combination-3)

Figure 6.11 illustrates that according to job allocation from combination-1, in first stage the machine-1 (G) will process jobs in sequence- 1, 2, 3, 5, 9, 13, 14, 16, 18 in the given order and the machine-2 (H) will process jobs in sequence – 4, 6, 7, 8, 10, 11, 12, 14, 15, 17, 18, 19, 20 sequentially.

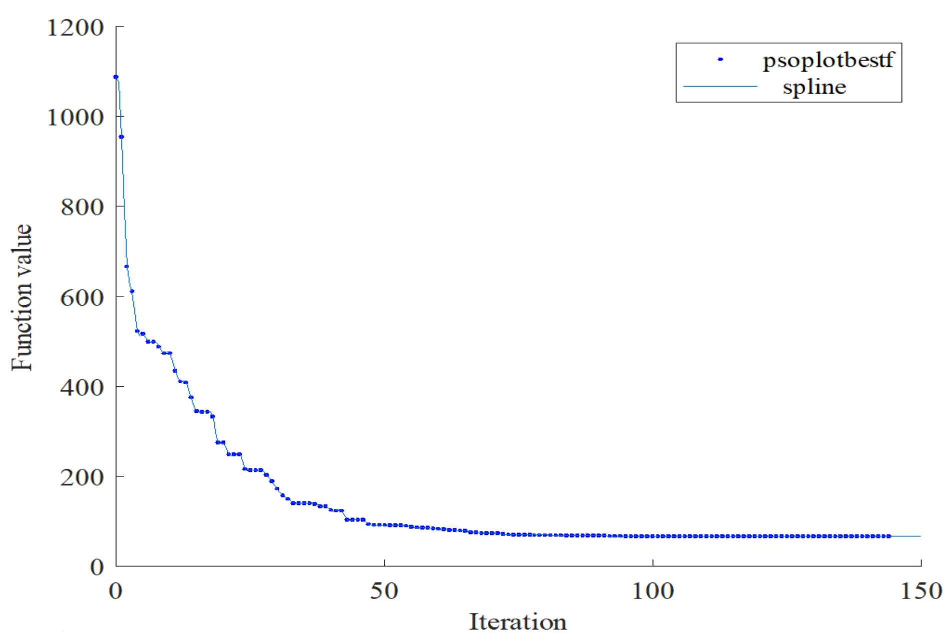

Figure 6.12: PSO Plot (Combination-3)

Figure 6.12 illustrates the PSO curve, where the value of the objective function is converging with each iteration and at iteration no 140 the algorithm terminates as there is no improvement in objective function value.

For combination 4, the job assignment is given in **Table 6.8**, and in each stage job assignments to each machine are illustrated respectively in **Figure 6.13**, **Figure 6.14** and **Figure 6.15**. **Figure 6.16**, represents the corresponding PSO curve with the best function value obtained of 112.87 days

Table 6.8: Job Allocation (Combination-4)

						Machine No				
		Sequence	A	B	C	D	E	F	G	H
Job ID	9493	1	0	1	0	1	0	0	0	1
	77358	2	1	0	1	0	0	0	0	1
	76526	3	0	1	1	0	0	0	0	1
	46896	4	0	1	0	0	1	0	1	0
	37304	5	0	1	0	0	1	0	0	1
	971	6	1	0	1	0	0	0	0	1
	98804	7	1	0	0	0	1	0	1	0
	22224	8	1	0	0	1	0	0	1	0
	45388	9	0	1	1	0	0	0	0	1
	51144	10	1	0	1	0	0	0	0	1
	52303	11	0	1	0	1	0	0	0	1
	79878	12	1	0	0	0	1	0	1	0
	6370	13	0	1	0	0	0	1	1	0
	19376	14	0	1	0	0	1	0	1	0
	40919	15	1	0	0	1	0	0	1	0
	22216	16	1	0	0	0	0	1	1	0
	15068	17	0	1	0	0	0	1	0	1
	41555	18	1	0	0	0	0	1	0	1
	6324	19	0	1	0	0	1	0	1	0
	21738	20	0	1	0	0	0	1	1	0

From **Table 6.1**, it can be that this job assignment is obtained by giving less importance to tardiness. By comparing the required delivery date and completion time of each job, job IDs mentioned in **Table 6.9**, cannot be completed before the due date. In this case, the number of delay jobs is comparatively high though none of the jobs will be delayed more than 25 days.

Table 6.9: List of delay jobs (Combination -4)

Job ID	Sequence	Delay	Job ID	Sequence	Delay
76526	3	7.9	45388	9	6.7
37304	5	24.4	51144	10	8.4
971	6	17.5	52303	11	4.5
98804	7	23.4	15068	17	2.2

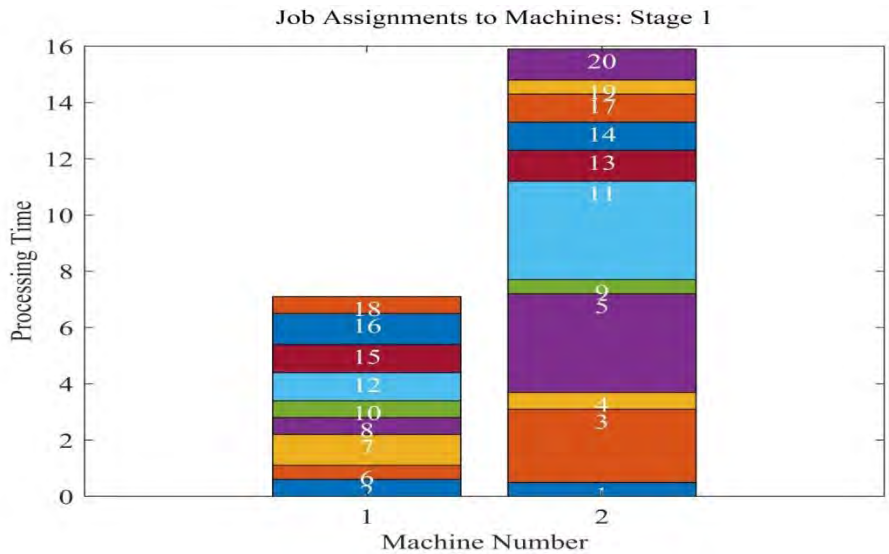

Figure 6.13: Job allocation in stage-1 (Combination-4)

Figure 6.13 illustrates that according to job allocation from combination-1, in first stage the machine-1 (A) will process jobs in sequence- 2, 6, 7, 8, 10, 12, 15, 16, 18 in the given order and the machine-2 (B) will process jobs in sequence – 1, 3, 4, 5, 9, 11, 13, 14, 17, 19.

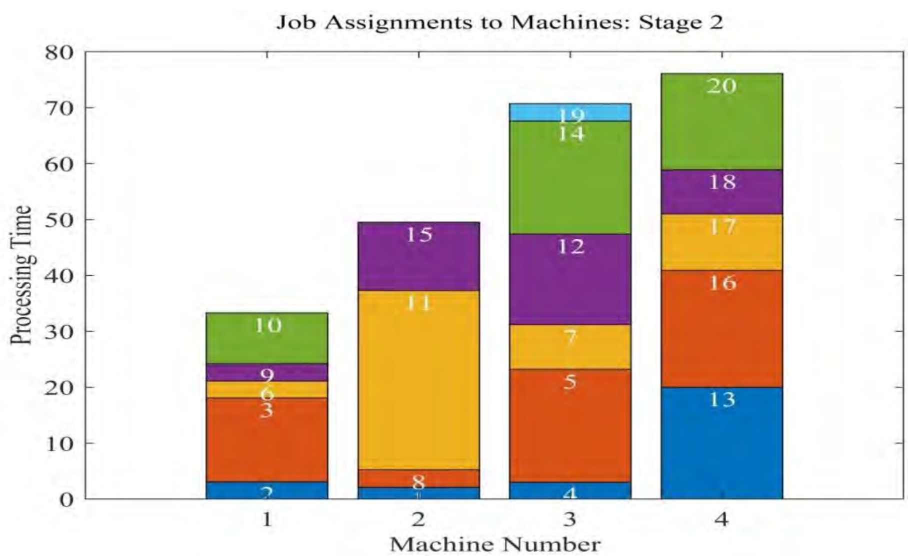

Figure 6.14: Job allocation in stage-2 (Combination-4)

Figure 6.14 illustrates that according to job allocation from combination-1, in first stage the machine-1 (C) will process jobs in sequence- 2, 3, 6, 9, 10 in the given order; the machine-2 (D) will process jobs in sequence – 1, 8, 11, 15; the machine-3 (E) will process jobs in sequence – 4, 5. 7, 12, 14, 19; the machine-4 (F) will process jobs in sequence – 13, 16, 17, 18, 20 sequentially.

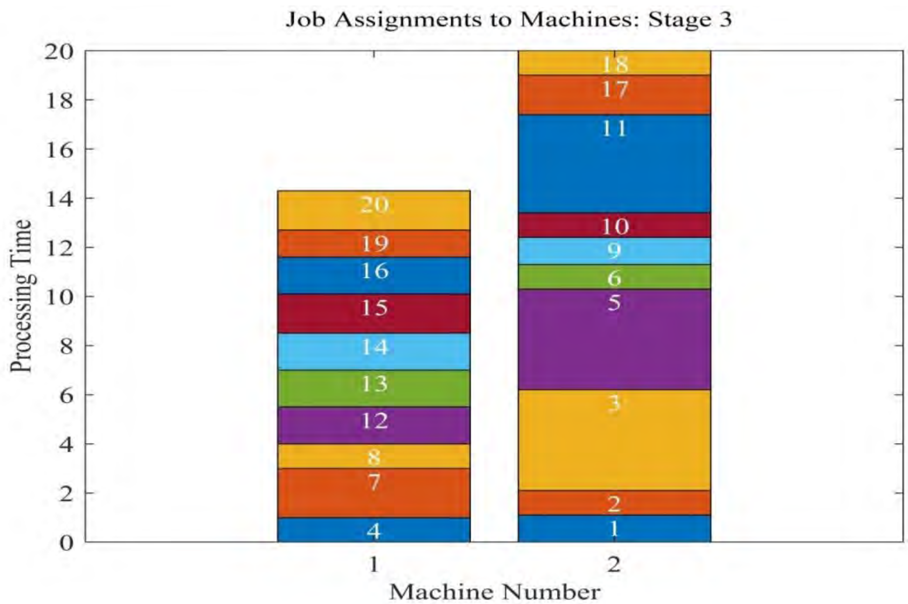

Figure 6.15: Job allocation in stage-3 (Combination-4)

Figure 6.15 illustrates that according to job allocation from combination-1, in first stage the machine-1 (G) will process jobs in sequence- 4, 7, 8, 12, 13, 14, 15, 16, 19, 20 in the given order and the machine-2 (H) will process jobs in sequence – 1, 2, 3, 5, 6, 9, 10, 11, 17, 18 sequentially.

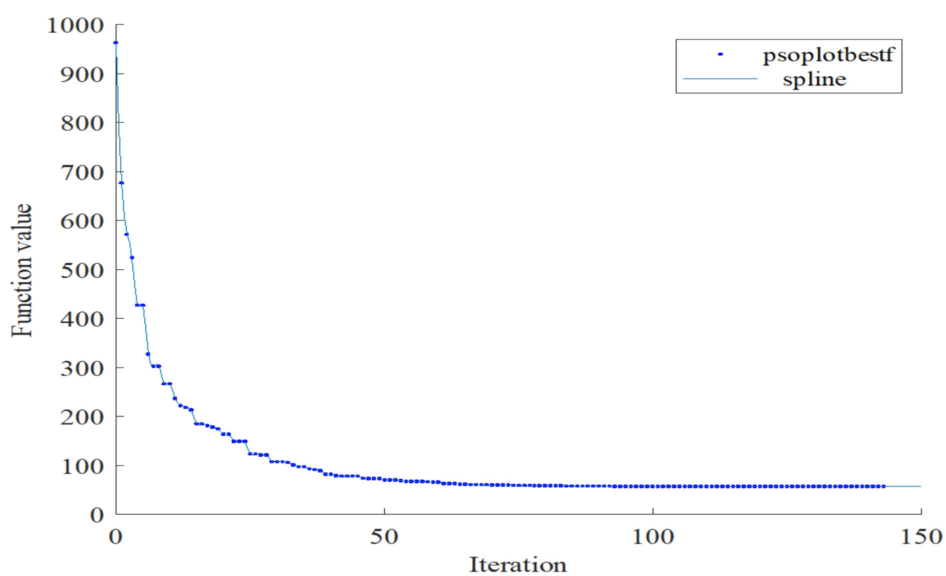

Figure 6.16: PSO Plot (Combination-4)

Figure 6.16 demonstrates the convergence of PSO curve where optimum function value is obtained at iteration 140.

For combination 7, the job assignment is given in **Table 6.10**, and in each stage job assignments to each machine are illustrated respectively in **Figure 6.17**, **Figure 6.18** and **Figure 6.19**. **Figure 6.20**, represents the corresponding PSO curve with the best function value obtained of 105.23 days

Table 6.10: Job Allocation (Combination-7)

		Machine No							
Job ID	Sequence	A	B	C	D	E	F	G	H
9493	1	1	0	0	0	1	0	1	0
77358	2	1	0	1	0	0	0	1	0
76526	3	0	1	0	0	0	1	1	0
46896	4	0	1	0	0	1	0	0	1
37304	5	1	0	0	1	0	0	1	0
971	6	1	0	0	0	1	0	0	1
98804	7	0	1	0	1	0	0	0	1
22224	8	0	1	1	0	0	0	0	1
45388	9	1	0	1	0	0	0	1	0
51144	10	0	1	0	0	1	0	0	1
52303	11	0	1	1	0	0	0	0	1
79878	12	1	0	0	0	0	1	0	1
6370	13	1	0	1	0	0	0	0	1
19376	14	0	1	0	0	1	0	0	1
40919	15	0	1	0	1	0	0	1	0
22216	16	1	0	0	0	0	1	0	1
15068	17	0	1	0	0	0	1	0	1
41555	18	1	0	0	1	0	0	1	0
6324	19	1	0	0	1	0	0	1	0
21738	20	1	0	0	0	1	0	1	0

From **Table 6.1**, it can be that this job assignment is obtained by giving moderate importance to tardiness. By comparing the required delivery date and completion time of each job, job IDs mentioned in **Table 6.9**, cannot be completed before the due date.

Table 6.11: List of Delay Jobs (Combination -7)

Job ID	Sequence	Delay
76526	3	6
37304	5	19.7
971	6	0.4
98804	7	21.3
15068	17	11

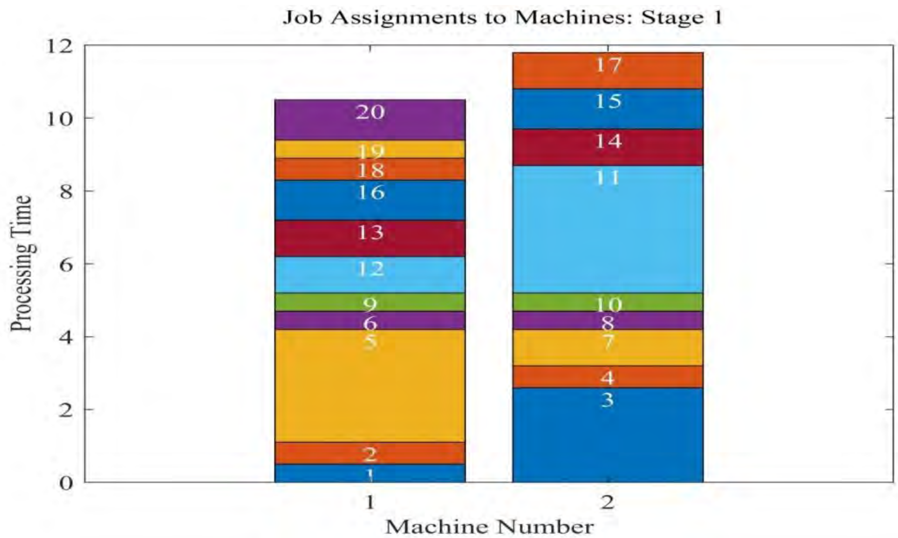

Figure 6.17: Job allocation in stage-1 (Combination-7)

Figure 6.17 illustrates that according to job allocation from combination-1, in first stage the machine-1 (A) will process jobs in sequence- 1, 2, 5, 6, 9, 12, 13, 16, 18, 19, 20 in the given order and the machine-2 (B) will process jobs in sequence – 3, 4 7, 8, 10, 11, 14, 15, 17 sequentially.

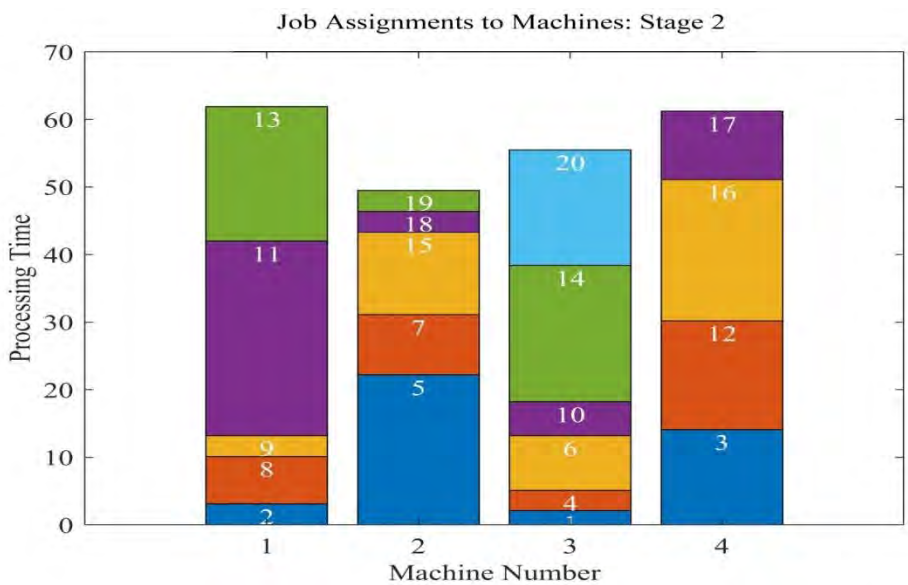

Figure 6.18: Job allocation in stage-2 (Combination-7)

Figure 6.18 illustrates that according to job allocation from combination-1, in first stage the machine-1 (C) will process jobs in sequence- 2, 8, 9, 11, 13 in the given order; the machine-2 (D) will process jobs in sequence – 5, 7, 15, 18, 19; the machine-3 (E) will process jobs in sequence – 1, 4, 6, 10, 14, 20; the machine-4 (F) will process jobs in sequence – 3, 12, 16, 17 sequentially.

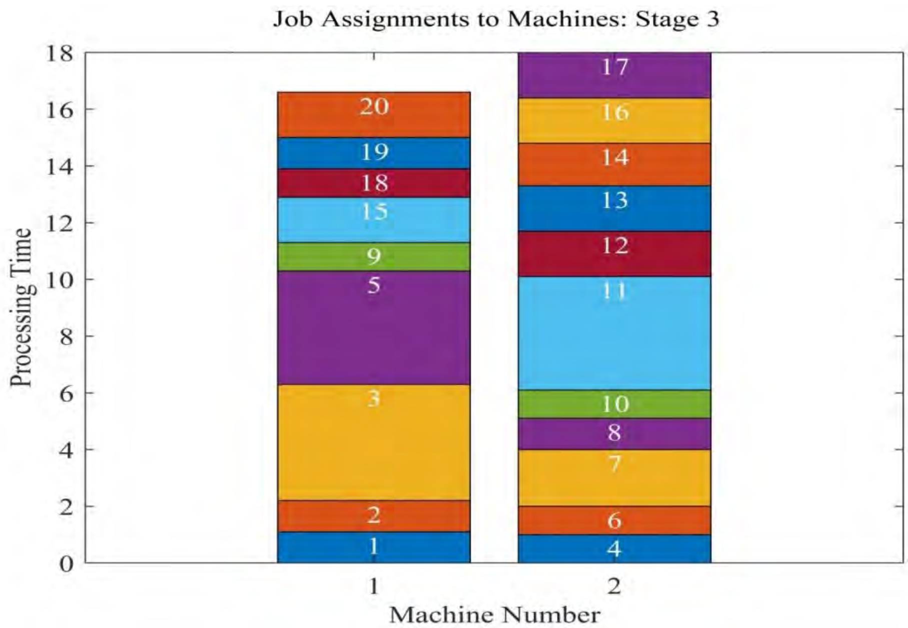

Figure 6.19: Job allocation in stage-3 (Combination-7)

Figure 6.19 illustrates that according to job allocation from combination-1, in first stage the machine-1 (G) will process jobs in sequence- 1, 2, 3, 5, 9, 15, 18, 19, 20 in the given order and the machine-2 (H) will process jobs in sequence – 4, 6, 7, 8, 10, 11, 12, 13, 14, 16, 17 sequentially.

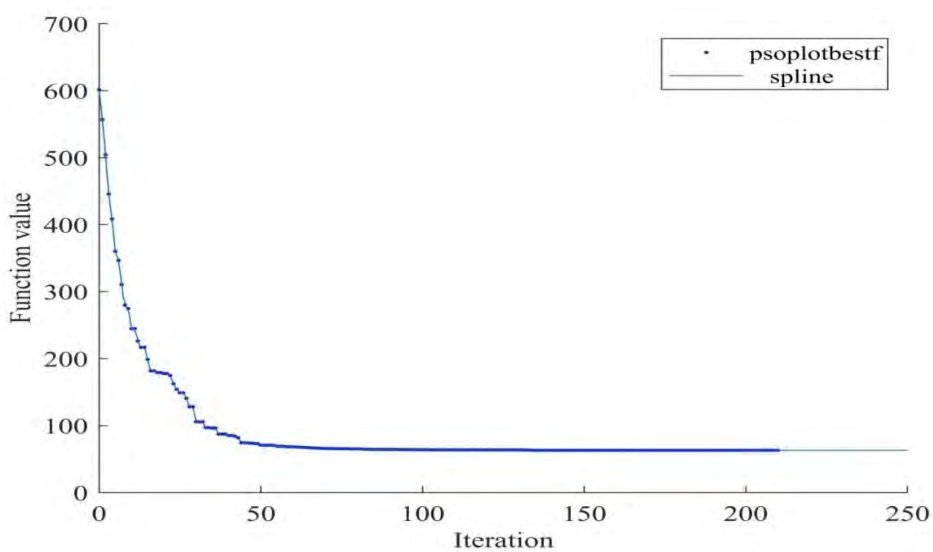

Figure 6.20: PSO Plot (Combination -7)

Figure 6.20 illustrates the PSO curve, where the value of the objective function is decreasing with each iteration and finally at iteration no 210 the algorithm terminates as there is no improvement in objective function value.

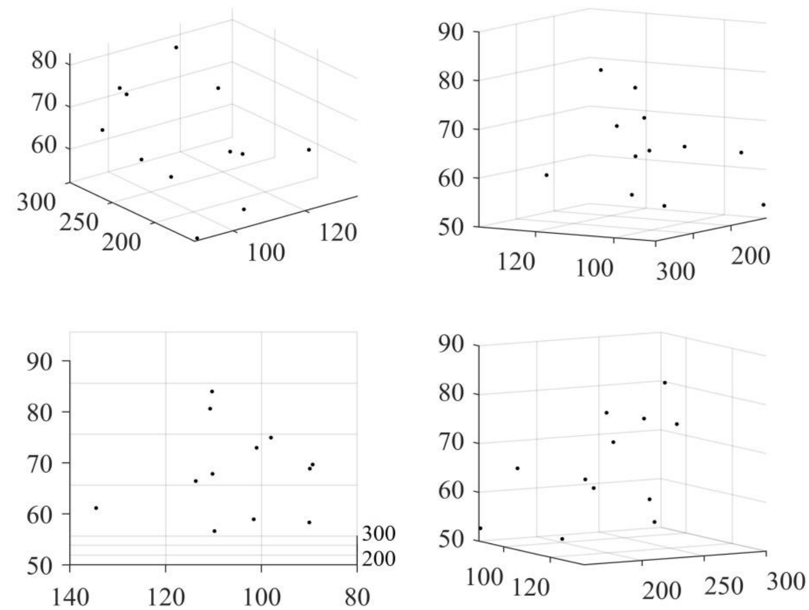

Figure 6.21: Pareto Front

Figure 6.21, represents the value of the objective functions in 3D-Plot, using these points Pareto surface presented in **Figure 6.22** is generated.

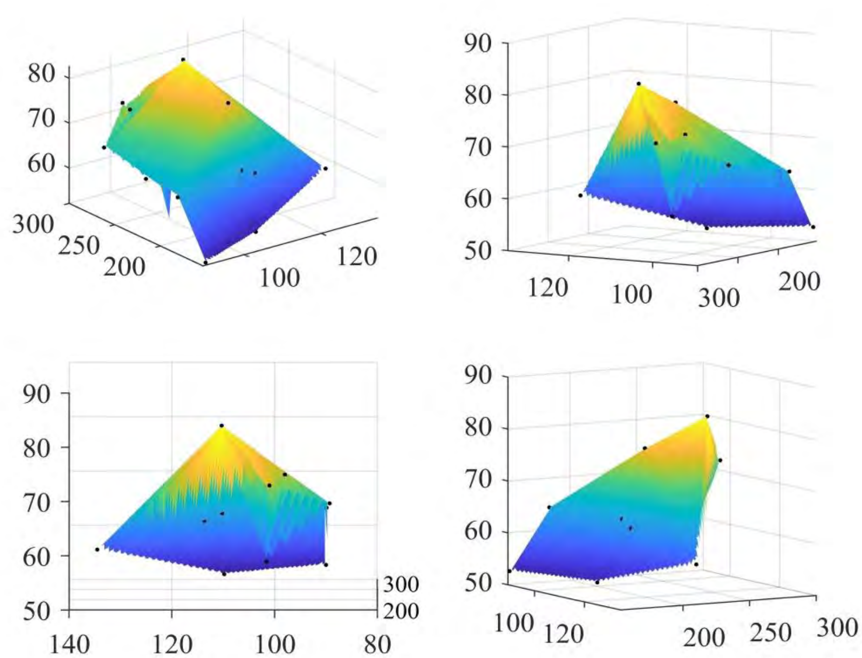

Figure 6.22: Pareto Surface

6.2 Sensitivity Analysis

In order to validate the proposed model sensitivity analysis is conducted by changing the number of jobs and examining its impact on make-span, tardiness, and efficiency. In **Table 6.12** make-span, tardiness and efficiency are given in days. From **Figure 6.23**, it is observed that make-span is in increasing trend with the increase in job number. It can be understood that if the job number is increasing make-span will increase with a fixed number of machines. Similarly in **Figure 6.24** and **Figure 6.25** tardiness and machine idle time both are in increasing trend with the increase in job number for a fixed number of machines. For fewer jobs machine idle time is less because fewer jobs require less line change over, which results in less idle time. Consequently, less machine idle time leads to higher efficiency. So, with the increase in the number of jobs for a fixed number of machines, machine idle time increases comparatively resulting in a decreasing efficiency.

Table 6.12: Change of Objective Function Values with Job Number Variation

Number of Jobs	Make-span	Tardiness	Machine Idle Time
10	55	45.6	15
12	64	57	30
14	73	127	48.5
16	88	147	56.5
18	110	177	65
20	112	181.95	69

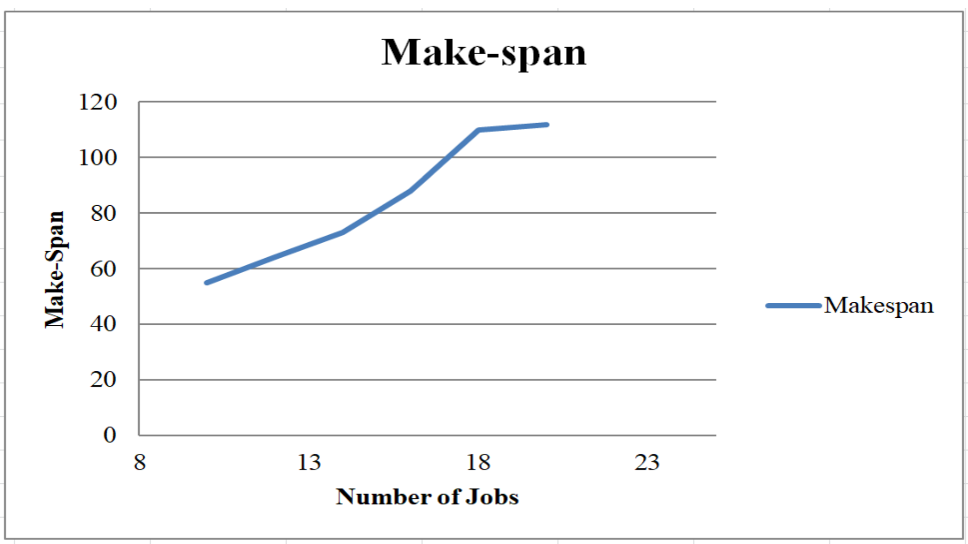

Figure 6.23: Make-span versus the number of jobs

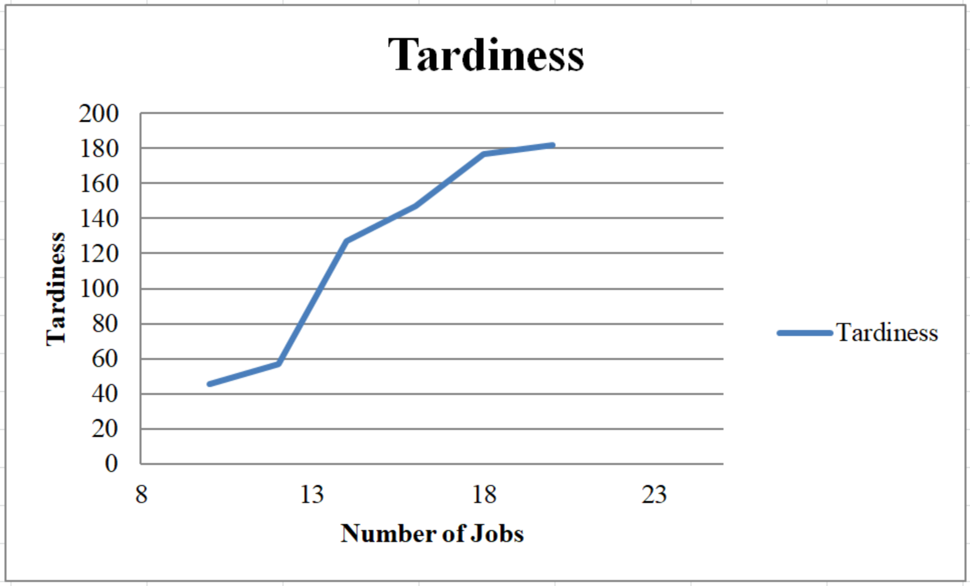

Figure 6.24: Tardiness versus the number of jobs

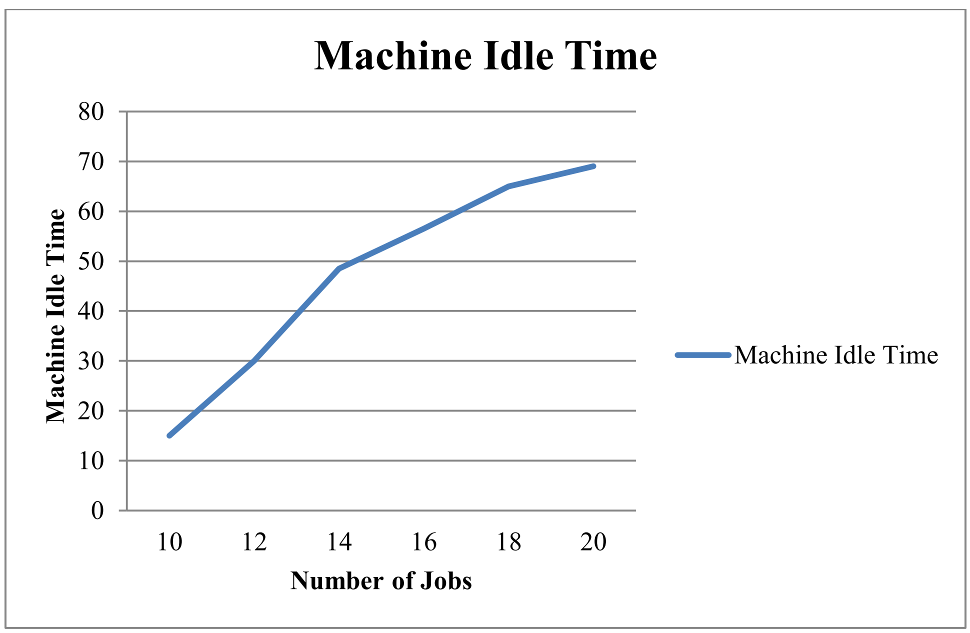

Figure 6.25: Machine idle time versus the number of jobs

From the result analysis, it can be seen that job assignments changes with the different priority level of make-span, tardiness, and efficiency. Depending upon the scheduler's requirement priority level of the objectives can be changed to get the desired output. The result also gives an idea that if the job number can be reduced or the machine number can be increased better results can be obtained with fewer delay jobs and lower completion time of each job. This helps to provide an overview of available production capacity versus the required production capacity.

CHAPTER 7: CONCLUSIONS AND FUTURE RESEARCH

7.1 Conclusions

The noteworthy contributions of the research work are given below:

i. Developing a job classification model, that can classify jobs based on their priority level. In the production system, there is some associated qualitative and quantitative information for each job, which becomes quite difficult to address through existing scheduling algorithms, whereas, this information plays a vital role during job sequencing. The proposed job classification model developed by implementing SVM is capable to address the limitation of existing models, which is a major contribution of this work.

ii. As mentioned in the literature review, hybrid flow shop scheduling is an NP-hard problem that is quite difficult to solve. The model formulation is done in such a way that it has only two constraints and there is only one decision variable present. Adding to this, the scheduling model is also capable of reacting to uncertainties.

iii. Finally, the job scheduling approach proposed in this thesis is a novel approach, where the computational time is relatively less compared to existing scheduling models. As the first step that involves job classification requires less computational effort.

7.2 Future Research

For future research work, this thesis can be modified by incorporating the following considerations:

i. Incorporating the concept of green manufacturing, which can be done by considering energy consumption on the production floor.

ii. Considering machine longevity, the current status of a machine, etc while calculating machine break-down probability.

iii. Furthermore, in this thesis, PSO is used to solve the scheduling problem, due to its efficacy and simplicity. However, other hybrid methods can be used to understand which algorithm offers a better result. As well as optimized weightage value for each objective can be used.

REFERENCES

Aggoune, R. (2003). Minimizing the makespan for the flow shop scheduling problem with availability constraints. *European Journal of Operational Research*, *153*(3 SPEC. ISS.), 534–543.

Al-Hinai, N. & Elmekkawy, T. Y. (2011). Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm. *International Journal of Production Economics*, *132*(2), 279–291.

Behnamian, J. & Zandieh, M. (2013). Earliness and Tardiness Minimizing on a Realistic Hybrid Flowshop Scheduling with Learning Effect by Advanced Metaheuristic. *Arabian Journal for Science and Engineering*, *38*(5), 1229–1242.

Berge, C. (1957). Two theorems in graph theory. *Proceedings of the National Academy of Sciences of the United States of America*, *43*(9), 842.

Botta-Genoulaz, V. (2000). Hybrid flow shop scheduling with precedence constraints and time lags to minimize maximum lateness. *International Journal of Production Economics*, *64*(1), 101–111.

Choong, F., Phon-Amnuaisuk, S. & Alias, M. Y. (2011). Metaheuristic methods in hybrid flow shop scheduling problem. *Expert Systems with Applications*, *38*(9), 10787–10793.

Dorronsoro, B. & Pinel, F. (2017). Combining machine learning and genetic algorithms to solve the independent tasks scheduling problem. *2017 3rd IEEE International Conference on Cybernetics, CYBCONF 2017 - Proceedings*, 0–7.

Engin, O. & Döyen, A. (2004). A new approach to solve hybrid flow shop scheduling problems by artificial immune system. *Future Generation Computer Systems*, *20*(6 SPEC. ISS.), 1083–1095.

Ercan, M. F. (2008a). A hybrid particle swarm optimization approach for scheduling flow-shops with multiprocessor tasks. *Proceedings of the International Conference on Information Science and Security, ICISS 2008*, 13–16.

Ercan, M. F. (2008b). A performance comparison of PSO and GA in scheduling hybrid Flow-shops with multiprocessor tasks. *Proceedings of the ACM Symposium on Applied Computing*, 1767–1771.

Framinan, J. M., Gupta, J. N. D. & Leisten, R. (2004). A review and classification of heuristics for permutation flow-shop scheduling with makespan objective. *Journal of the Operational Research Society*, *55*(12), 1243–1255.

Gupta, J. N. D. & Stafford, E. F. (2006). Flowshop scheduling research after five decades. *European Journal of Operational Research*, *169*(3), 699–711.

Hajinejad, D., Salmasi, N. & Mokhtari, R. (2011). A fast hybrid particle swarm optimization algorithm for flow shop sequence dependent group scheduling problem. *Scientia Iranica*, *18*(3 E), 759–764.

Han, Z., Ma, X., Yao, L. & Shi, H. (2012). Cost optimization problem of hybrid flow-shop based on PSO algorithm. *Advanced Materials Research*, *532–533*, 1616–1620.

Ishibuchi, H. & Murata, T. (1998). A multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, *28*(3), 392–403.

Jamili, A. (2019). Job shop scheduling with consideration of floating breaking times under uncertainty. *Engineering Applications of Artificial Intelligence*, *78*, 28–36.

Janak, S. L., Floudas, C. A., Kallrath, J. & Vormbrock, N. (2006). Production scheduling of a large-scale industrial batch plant. II. Reactive scheduling. *Industrial and Engineering Chemistry Research*, *45*(25), 8253–8269.

Janak, S. L., Lin, X. & Floudas, C. A. (2007). A new robust optimization approach for scheduling under uncertainty. II. Uncertainty with known probability distribution. *Computers and Chemical Engineering*, *31*(3), 171–195.

Karunakaran D., Mei Y., Chen G., Zhang M. (2017). Dynamic job shop scheduling under uncertainty using genetic programming In Leu G., Singh H., Elsayed S. (Eds.), *Intelligent and Evolutionary Systems* (pp.195-210). Canberra, Australia: Springer, Cham.

Kuo, I. H., Horng, S. J., Kao, T. W., Lin, T. L., Lee, C. L., Terano, T. & Pan, Y. (2009). An efficient flow-shop scheduling algorithm based on a hybrid particle swarm optimization model. *Expert Systems with Applications*, *36*(3 part 2), 7027–7032.

Li, B. Bin & Wang, L. (2007). A hybrid quantum-inspired genetic algorithm for multiobjective flow shop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *37*(3), 576–591.

Li, B. Bin, Wang, L. & Liu, B. (2008). An effective PSO-based hybrid algorithm for multiobjective permutation flow shop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, *38*(4), 818–831.

Liao, C. J., Tjandradjaja, E. & Chung, T. P. (2012). An approach using particle swarm optimization and bottleneck heuristic to solve hybrid flow shop scheduling problem. *Applied Soft Computing Journal*, *12*(6), 1755–1764.

Linn, R., & Zhang, W. (1999). Hybrid flow shop scheduling: a survey. *Computers and Industrial Engineering*, *37*(1), 57–61.

Liu, B., Wang, L. & Jin, Y. H. (2005). Hybrid particle swarm optimization for flow shop scheduling with stochastic processing time. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3801 LNAI*, 630–637.

Liu, B., Wang, L. & Jin, Y. H. (2007a). An effective hybrid particle swarm optimization for no-wait flow shop scheduling. *International Journal of Advanced Manufacturing Technology*, *31*(9–10), 1001–1011.

Liu, B., Wang, L. & Jin, Y. H. (2007b). An effective PSO-based memetic algorithm for flow shop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *37*(1), 18–27.

Liu, B., Wang, L. & Jin, Y. H. (2008). An effective hybrid PSO-based algorithm for flow shop scheduling with limited buffers. *Computers and Operations Research*, *35*(9), 2791–2806.

Liu, Y. H., Huang, H. P. & Lin, Y. S. (2005). Dynamic scheduling of flexible manufacturing system using support vector machines. *Proceedings of the 2005 IEEE Conference on Automation Science and Engineering, IEEE-CASE 2005*, *2005*, 387–392.

Luo, H., Du, B., Huang, G. Q., Chen, H. & Li, X. (2013). Hybrid flow shop scheduling considering machine electricity consumption cost. *International Journal of Production Economics*, *146*(2), 423–439.

Marichelvam, M. K., Prabaharan, T. & Yang, X. S. (2014). Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan. *Applied Soft Computing Journal*, *19*, 93–101.

Marichelvam, Mariappan Kadarkarainadar, Prabaharan, T. & Yang, X. S. (2014). A discrete firefly algorithm for the multi-objective hybrid flowshop scheduling problems. *IEEE Transactions on Evolutionary Computation*, *18*(2), 301–305.

Mirabi, M., Ghomi, S. M. T. F. & Jolai, F. (2013). A two-stage hybrid flowshop scheduling problem in machine breakdown condition. *Journal of Intelligent Manufacturing*, *24*(1), 193–199.

Murata, T., Ishibuchi, H. & Tanaka, H. (1996). Genetic algorithms for flowshop scheduling problems. *Computers and Industrial Engineering*, *30*(4), 1061–1071.

Novak, A., Sucha, P. & Hanzalek, Z. (2019). Scheduling with uncertain processing times in mixed-criticality systems. *European Journal of Operational Research*, *279*(3), 687–703.

Osman, I. & Potts, C. (1989). Simulated annealing for permutation flow-shop scheduling. *Omega*, *17*(6), 551–557.

Pan, Q. K., Fatih Tasgetiren, M. & Liang, Y. C. (2008). A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem. *Computers and Operations Research*, *35*(9), 2807–2839.

Priore, P., De La Fuente, D., Puente, J. & Parreño, J. (2006). A comparison of machine-learning algorithms for dynamic scheduling of flexible manufacturing systems. *Engineering Applications of Artificial Intelligence*, *19*(3), 247–255.

Priore, P., Ponte, B., Puente, J. & Gómez, A. (2018). Learning-based scheduling of flexible manufacturing systems using ensemble methods. *Computers and Industrial Engineering*, *126*(September), 282–291.

Rahmani, D. & Heydari, M. (2014). Robust and stable flow shop scheduling with unexpected arrivals of new jobs and uncertain processing times. *Journal of Manufacturing Systems*, *33*(1), 84–92.

Rahmani, D. & Ramezanian, R. (2016). A stable reactive approach in dynamic flexible flow shop scheduling with unexpected disruptions: A case study. *Computers and Industrial Engineering*, *98*, 360–372.

Rajendran, C. & Ziegler, H. (2004). Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. *European Journal of Operational Research*, *155*(2), 426–438.

Ruiz, R., Maroto, C. & Alcaraz, J. (2006). Two new robust genetic algorithms for the flowshop scheduling problem. *Omega*, *34*(5), 461–476.

Ruiz, R., Şerifoğlu, F. S. & Urlings, T. (2008). Modeling realistic hybrid flexible flowshop scheduling problems. *Computers and Operations Research*, *35*(4), 1151–1175.

Ruiz, R. & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research*, *177*(3), 2033–2049.

Sayadi, M. K., Ramezanian, R. & Ghaffari-Nasab, N. (2010). A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems. *International Journal of Industrial Engineering Computations*, *1*(1), 1–10.

Schölkopf, B. & Smola, A. J. (2002). Risk and Loss Functions. In *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (pp.61–84). Massachusetts, United States of America: The MIT Press.

Schölkopf, B. & Smola, A. J. (2002). Pattern Recognition. In *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (pp.189–211). Massachusetts, United States of America:The MIT Press.

Shahvari, O. & Logendran, R. (2018). A comparison of two stage-based hybrid algorithms for a batch scheduling problem in hybrid flow shop with learning effect. *International Journal of Production Economics, 195,*227-248.

Shaw, M. J., Park, S. & Raman, N. (1992). Intelligent Scheduling with Machine Learning Capabilities: The Induction of Scheduling Knowledge. *IIE Transactions (Institute of Industrial Engineers)*, *24*(2), 156–168.

Sun, Y., Zhang, C., Gao, L. & Wang, X. (2011). Multi-objective optimization algorithms for flow shop scheduling problem: A review and prospects. *International Journal of Advanced Manufacturing Technology*, *55*(5–8), 723–739.

Tang, L. & Wang, X. (2010). An improved particle swarm optimization algorithm for the hybrid flowshop scheduling to minimize total weighted completion time in process industry. *IEEE Transactions on Control Systems Technology*, *18*(6), 1303–1314.

Team, D. F. (2018). Kernel functions-Introduction to SVM kernel & examples - DataFlair.

Tseng, C. T. & Liao, C. J. (2008). A particle swarm optimization algorithm for hybrid flow-shop scheduling with multiprocessor tasks. *International Journal of Production Research*, *46*(17), 4655–4670.

Wang, H. (2005). Flexible flow shop scheduling: Optimum, heuristics and artificial intelligence solutions. *Expert Systems*, *22*(2), 78–85.

Wang, J. B. & Xia, Z. Q. (2005). Flow-shop scheduling with a learning effect. *Journal of the Operational Research Society*, *56*(11), 1325–1330.

Xiong, J., Xing, L. N. & Chen, Y. W. (2013). Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. *International Journal of Production Economics*, *141*(1), 112–126.

Yagmahan, B. & Yenisey, M. M. (2008). Ant colony optimization for multi-objective flow shop scheduling problem. *Computers and Industrial Engineering*, *54*(3), 411–420.

Appendix A

Job classification result for model 1 to model 9:

Job ID	Model-1	Model-2	Model-3	Model-4	Model-5	Model-6	Model-7	Model-8	Model-9
971	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
77358	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
45388	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
76526	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
9493	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
46896	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
98804	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
37304	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
41555	Minor	Minor	Moderate	Minor	Minor	Minor	Minor	Minor	Minor
51144	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Moderate	Moderate
15068	Minor	Minor	Minor	Minor	Minor	Minor	Minor	Minor	Minor
79878	Minor	Minor	Moderate	Minor	Minor	Minor	Moderate	Moderate	Moderate
21738	Minor	Minor	Moderate	Minor	Minor	Minor	Minor	Minor	Minor
19376	Minor	Minor	Moderate	Minor	Minor	Minor	Moderate	Moderate	Moderate
6370	Minor	Minor	Moderate	Minor	Minor	Minor	Moderate	Moderate	Moderate
22216	Minor	Minor	Moderate	Minor	Minor	Minor	Moderate	Moderate	Moderate
22224	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
6324	Minor	Minor	Moderate	Minor	Minor	Minor	Minor	Minor	Minor
40919	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Moderate	Moderate	Moderate
52303	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Moderate	Moderate	Moderate

Job classification result for model 10 to model 18:

Job ID	Model-10	Model-11	Model-12	Model-13	Model-14	Model-15	Model-16	Model-17	Model-18
971	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
77358	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
45388	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
76526	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
9493	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
46896	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
98804	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
37304	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
41555	Minor	Minor	Moderate	Minor	Minor	Minor	Moderate	Minor	Minor
51144	Moderate	Urgent	Urgent	Moderate	Urgent	Urgent	Urgent	Urgent	Urgent
15068	Minor	Minor	Minor	Minor	Minor	Minor	Minor	Minor	Minor
79878	Moderate	Moderate	Moderate	Moderate	Minor	Minor	Moderate	Minor	Minor
21738	Minor	Minor	Minor	Minor	Minor	Minor	Moderate	Minor	Minor
19376	Moderate	Moderate	Moderate	Moderate	Minor	Minor	Moderate	Minor	Minor
6370	Moderate	Moderate	Moderate	Moderate	Minor	Minor	Moderate	Minor	Minor
22216	Moderate	Moderate	Moderate	Minor	Minor	Minor	Moderate	Minor	Minor
22224	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Minor	Urgent
6324	Minor	Minor	Moderate	Minor	Minor	Minor	Moderate	Minor	Minor
40919	Moderate	Moderate	Urgent	Moderate	Urgent	Urgent	Urgent	Urgent	Urgent
52303	Urgent	Moderate	Urgent	Moderate	Urgent	Urgent	Urgent	Urgent	Urgent

Job classification result for model 19 to model 26:

Job ID	Model-19	Model-20	Model-21	Model-22	Model-23	Model-24	Model-25	Model-26
971	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
77358	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
45388	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
76526	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
9493	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
46896	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
98804	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
37304	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
41555	Minor	Minor	Minor	Minor	Minor	Minor	Moderate	Minor
51144	Urgent	Urgent	Moderate	Moderate	Moderate	Urgent	Urgent	Moderate
15068	Minor	Minor	Minor	Minor	Minor	Minor	Minor	Minor
79878	Minor	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate
21738	Minor	Minor	Minor	Minor	Minor	Minor	Minor	Minor
19376	Minor	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate
6370	Minor	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate
22216	Minor	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Minor
22224	Minor	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent	Urgent
6324	Minor	Minor	Minor	Minor	Minor	Minor	Moderate	Minor
40919	Urgent	Moderate	Moderate	Moderate	Moderate	Moderate	Urgent	Moderate
52303	Urgent	Moderate	Moderate	Moderate	Urgent	Moderate	Urgent	Moderate

Appendix B

Training data set for SVM:

Job ID	Ordered Quantity	Customer	Prod Lead Time (days)	Value ($)	Line running	Material Availability	Priority Level
95773	312	Europe	47	1941	Yes	Available	Moderate
44475	330	Malaysia	35	1594	No	Not available	Moderate
18109	378	Europe	24	2597	Yes	Available	Urgent
4151	396	Bresil	35	1913	No	Not available	Moderate
7142	396	Bresil	34	1913	No	Not available	Moderate
6887	516	Bresil	40	3210	No	Not available	Moderate
11107	546	Europe	31	3751	Yes	Available	Urgent
21618	720	Europe	45	3326	Yes	Available	Moderate
3581	726	Bresil	36	3507	Yes	Available	Moderate
97807	726	Bresil	47	3507	No	Not available	Moderate
49142	756	Europe	31	9866	Yes	Available	Urgent
72126	760	Europe	17	10055	Yes	Available	Urgent
24639	1716	America	31	8288	No	Available	Urgent
90539	2004	Europe	21	26152	No	Available	Urgent
71269	2046	Europe	15	9882	Yes	Available	Urgent
73130	2046	Europe	17	9882	Yes	Available	Urgent
73982	2160	Europe	18	9979	No	Available	Urgent
73423	2178	Europe	18	28815	No	Available	Urgent
55983	2652	Europe	38	16495	Yes	Available	Moderate

Training data set for SVM:

Job ID	Ordered Quantity	Customer	Prod Lead Time (days)	Value ($)	Line running	Material Availability	Priority Level
55983	2652	Europe	38	16495	Yes	Available	Moderate
95772	3888	Europe	47	24183	Yes	Available	Moderate
13322	3902	Europe	17	51350	Yes	Available	Urgent
14738	3960	Europe	45	18295	Yes	Available	Moderate
86166	5490	Europe	16	25364	Yes	Available	Urgent
9130	9036	Europe	46	187949	Yes	Available	Moderate
36309	304	Europe	33	6278	Yes	Available	Moderate
39705	330	Malaysia	32	1594	No	Not available	Moderate
79135	360	Europe	19	4738	Yes	Available	Urgent
98520	360	Europe	13	4738	Yes	Available	Urgent
78068	590	Europe	19	7764	Yes	Available	Urgent
79134	622	Europe	19	8186	Yes	Available	Urgent
77829	624	Europe	19	8212	Yes	Available	Urgent
39706	630	Malaysia	18	2911	Yes	Available	Urgent
78070	652	Europe	19	8580	Yes	Available	Urgent
78067	684	Europe	19	9001	Yes	Available	Urgent
78750	726	Europe	12	9605	Yes	Available	Urgent
79133	1772	Europe	19	23320	Yes	Available	Urgent
98523	352	Europe	13	4657	Yes	Available	Urgent

Training data set for SVM:

Job ID	Ordered Quantity	Customer	Prod Lead Time (days)	Value ($)	Line running	Material Availability	Priority Level
98519	458	Europe	13	6027	Yes	Available	Urgent
94131	738	Europe	18	9764	Yes	Available	Urgent
2164	768	Europe	12	10161	Yes	Available	Urgent
78749	1186	Europe	12	15691	No	Available	Urgent
79132	2526	Europe	12	33242	Yes	Available	Urgent
99486	7500	Europe	19	34650	No	Available	Urgent
2150	400	Europe	73	6060	No	Not available	Moderate
6376	554	Europe	74	8393	No	Not available	Moderate
55245	300	Europe	17	1386	Yes	Available	Urgent
51340	326	Morocco	22	4290	Yes	Available	Urgent
47761	336	Europe	16	4445	Yes	Available	Urgent
77830	524	Europe	38	6896	Yes	Available	Moderate
37986	736	Europe	13	9737	Yes	Available	Urgent
78069	756	Europe	39	9949	Yes	Available	Moderate
14756	762	Europe	17	10028	Yes	Available	Urgent
49186	777	Europe	16	5338	Yes	Available	Urgent
55164	924	America	23	4463	Yes	Available	Urgent
68004	992	Europe	11	13124	Yes	Available	Urgent
35425	1056	Europe	46	23443	Yes	Available	Moderate

Appendix C

Processing time without learning curve effect:

Job ID		Machines							
		A	B	C	D	E	F	G	H
	9493	0.5	0.5	2	2.1	2.1	2.1	1.1	1.1
	77358	0.6	0.6	3.1	3	3	3.1	1.1	1
	76526	2.1	2.6	15	15.1	14	14.1	4.1	4.1
	46896	0.6	0.6	3.1	3.1	3	3	1	1
	37304	3.1	3.5	22.2	22.2	20.2	20.3	4	4.1
	971	0.5	0.5	3	3.1	3.1	3	1	1
	98804	1.1	1	9	8.9	8	7.9	2	2
	22224	0.6	0.5	3	3.1	3	3.1	1	1.1
	45388	0.5	0.5	3.1	3.1	3.1	3.1	1	1.1
	51144	0.6	0.5	5.1	5.1	5	5.1	1	1
	52303	3.1	3.5	28.8	28.4	26	26	4	4
	79878	1	1	9.1	8.9	8	7.9	1.5	1.6
	6370	1	1.1	13.1	13.1	11.7	12	1.5	1.6
	19376	1	1	13.2	13.1	12.2	12	1.5	1.5
	40919	1	1.1	12.1	12.2	12.1	12	1.6	1.5
	22216	1.1	1.1	14	13.9	13.1	13.2	1.5	1.6
	15068	1	1	6.1	6.1	5.1	5	1.6	1.6
	41555	0.6	0.5	3.1	3.1	3	3	1	1
	6324	0.5	0.5	3.1	3.1	3.1	3	1.1	1
	21738	1.1	1.1	9.1	9	9.2	9.1	1.6	1.6

Appendix D

Processing time with learning curve effect:

	Machines							
Job ID	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
9493	0.6	0.5	6.1	6.1	7	7.2	1.1	1.1
77358	0.5	0.6	7.1	7	7.9	8	1.1	1
76526	2	2.6	22.2	22	22	22.1	4.1	4
46896	0.6	0.6	7	7	8.1	8.1	1.1	1.1
37304	3.1	3.5	32.1	31.7	28.2	27.8	4	4.1
971	0.5	0.5	7	7	8.1	8.1	1.1	1.1
98804	1.1	1	16.1	16.2	15.9	15.8	2.1	2
22224	0.6	0.6	7	7	8	8.1	1.1	1.1
45388	0.6	0.5	7.1	7	8	8.2	1	1
51144	0.6	0.6	9.1	9	10.2	10.1	1	1.1
52303	3.1	3.6	32.1	32.1	28.1	28	6.1	6
79878	1.1	1.1	15.9	16.1	16.2	16.1	1.5	1.6
6370	1	1.1	19.9	20	20.2	20	1.5	1.5
19376	1.1	1.1	20.3	19.6	20.2	19.8	1.5	1.5
40919	1.1	1	18.9	19.2	19.9	20.2	1.6	1.6
22216	1	1	20.9	21.1	21.5	20.9	1.6	1.6
15068	1	1	13.1	13.1	10.2	10.1	1.5	1.6
41555	0.5	0.6	7	7.1	8.1	7.9	1	1
6324	0.6	0.5	7	7.1	8	8.2	1.1	1
21738	1.1	1.1	15.9	16.3	17.1	17.2	1.6	1.6

Appendix E

Likert Scale to Select Criteria for the Job Classification:

	Very Important	**Important**	**Moderately Important**	**Slightly Important**	**Unimportant**
Order Value	1	2	3	4	5
Available Production Lead Time	1	2	3	4	5
Responsible Customer	1	2	3	4	5
Customer Urgency Level	1	2	3	4	5
Material Availability Status	1	2	3	4	5
Product Type	1	2	3	4	5
Production Status of Same Product	1	2	3	4	5
Finished Goods Inventory Status	1	2	3	4	5
Material Expiry Date	1	2	3	4	5
Order Quantity	1	2	3	4	5

Using this chart, among 9 criteria 6 criteria has been chosen. The first 6 criteria that have been ranked 'very important' and 'important' by most responses have selected as features of the training data set of SVM.