2010-01-01

E-Quality: Using Dimensional Index Values Towards Improving Classification Accuracy

Prashanth Devaram
University of Texas at El Paso, pdevaram@miners.utep.edu

E-QUALITY: USING DIMENSIONAL INDEX VALUES TOWARDS IMPROVING CLASSIFICATION ACCURACY

PRASHANTH DEVARAM

Department of Industrial Engineering

APPROVED:

_____
Tzu-Liang (Bill) Tseng, Ph.D., Chair

_____
Tao Xu, Ph.D.

_____
Eric D. Smith, Ph.D.

_____
Jianmei Zhang, Ph.D.

_____
Patricia D. Witherspoon, Ph.D.
Dean of the Graduate School

E-QUALITY: USING DIMENSIONAL INDEX VALUES TOWARDS IMPROVING CLASSIFICATION ACCURACY

By

PRASHANTH DEVARAM, B. Tech.

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Department of Industrial Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

December 2010

Acknowledgements

I express my gratitude to all my family members and friends for their support without which this thesis would not have been possible. I would like to express my sincere thanks to Dr. Tseng for believing in me and choosing me for this research. I would like to thank Dr. Eric Smith, Dr. Jianmei Zhang and Dr. Xu Tao for being my committee members and dedication of their valuable time with this research. I would also like to thank Mr. Frank Medina and Mr. David Espalin from W.M. Keck Center for their valuable support without which this thesis would not have been possible.

Abstract

E-quality is a process through which inspection of the process and quality of the part produced is done online resulting in the improvement of the process and reduction in the amount of time consumed for the overall process. Automated quality control involves using a methodology to classify the parts based on the dimensions of the features on a part. However, achieving 100% classification accuracy is not an easy task, especially in area of quality control where small differences in dimensions result in part fall into a different category. In this study, a novel approach for modifying the data before being used for training of Support vector Machines (SVM) is presented. A new methodology for classifying the parts into different categories is also presented and the classification accuracies of both the approaches are compared with that of the traditional SVM approach. SVM was used as benchmark keeping in view of its higher generalization ability especially when the data set is small and class overlap is non-existent, primary attribute of quality control data. The data extracted from Machine Vision Systems (MVS) in a robotic set up was used as case study to demonstrate the three procedures. **Results show that the proposed new (sine) methodology yielded superior results compared to the rest in the current scenario with 100% classification accuracy. Moreover, it was found that with the proposed methodology, the classification accuracy can be improved up to the level of 8th decimal point by using more accurate 'C' value.** In the current work, accuracy up to 4th decimal point was demonstrated. Any number of features on a part can be used without limit, for classification with the proposed new methodology, accuracy being unaffected.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

The ever increasing competition among the production groups in turn forced them to move towards shorter product life cycle, remote quality control, smaller products and network based production and distribution systems to stay in the competition. E-manufacturing is the process of integration of design, manufacturing, quality and business functions with integrated information networks. The competition in the manufacturing industry made them to place high emphasis on quality and reliability of the products. In other words, industry is striving towards achieving zero defect manufacturing. How precise the product may be produced, there is at least a little chance that a defective product is produced. Industries like those that produce brake wires for automobiles are forced to inspect each and every part as a defective product may result in an accident. In such scenarios, where each product is to be inspected for quality, internet based inspection systems help perform quality inspection reliably, accurately and in very less time. E-quality is the process through which monitoring the process and inspection of the parts produced is performed online by integrating the machines into the information network [1]. Ability to control the equipment remotely offers tremendous benefits. Designers located remotely can visualize their designs and carry out inspections remotely. Inspections settings can be adjusted according to the requirements. A similar set up developed at Industrial Systems Engineering Laboratory (ISEL) at University of Texas at El Paso (UTEP) is explained in the following section.

1.2 Internet Based Inspection Set Up

In a manufacturing setting, to achieve zero defect products, parts are to be inspected at each stage to eliminate bad parts so that no defective product is produced at the end. Inspecting the parts as they are moving on a conveyer between stages of production minimizes the time for inspection. However, it also necessary to remove the bad parts away from the conveyer as they are moving on. Robots customized to pick and place are used to perform these functions. The current set up consists of a small conveyer on which parts move. A sensor attached to the conveyer detects the parts' arrival. Cognex machine vision system was used to inspect parts on the conveyer. The machine vision system performs the inspection by comparing the image of the current part to the standard image that is pre-loaded into the memory of the camera. Yamaha YK 350X SCARA (selective compliance assembly robot arm) robot was used to pick bad parts on the conveyer and place them away. The robot makes use of vacuum to pick the parts from the conveyer. When the sensor detects the parts' arrival, a signal is sent to the robot controller (RCX240 robotic controller was used in the current set up). The robot controller sends signals to the conveyor and the machine vision system. The conveyor stops and the machine vision system perform the inspection process. If the part is good, the camera sends a signal to the robot controller and the robot controller in turn signals the conveyer to move. Else, there will not be any signal from the camera and the robot controller sends a signal to the robot to pick up the bad part and later the conveyor to move on. When the signal is sent to the robot, the robot moves to the appropriate position and the vacuum is switched on so that the suction tip of the robot holds the part by means of vacuum pressure. The robot then moves to a different position where the part is to be placed and the vacuum is switched off. The robot is directly linked to the robot

controller and all other equipment like conveyor, machine vision system (MVS) and vacuum are operated from the robotic controller by means of digital signals (Inputs and Outputs). All the logical operations to be performed are written in the form of programming code (Appendix I) into the robotic controller that can be controlled remotely through an onboard Ethernet card, which is an optional device for connecting over the internet. TCP/IP (Transmission Control Protocol/Internet Protocol), a standard internet protocol can be used to communicate with the controller. The unit uses 10BASE-T specifications and UTP (unshielded twisted-pair) or STP (shielded twisted-pair) cables can be used. PCs can access the controller using Telnet. Commands can be sent to the controller once the connection is established. Application Programming Interface (API) is developed by embedding the Telnet procedure in the form of visual basic codes to have improved visualization of robotic movements and quality control operations. Web cameras were fixed in such positions that the remote operators can view the entire process through them and make necessary changes to the program in the robot controller. The connection between API and the controller was established by using Winsock components using various ActiveX controls that communicate through IP addresses. Figure 1 shows the overall setting of the system.

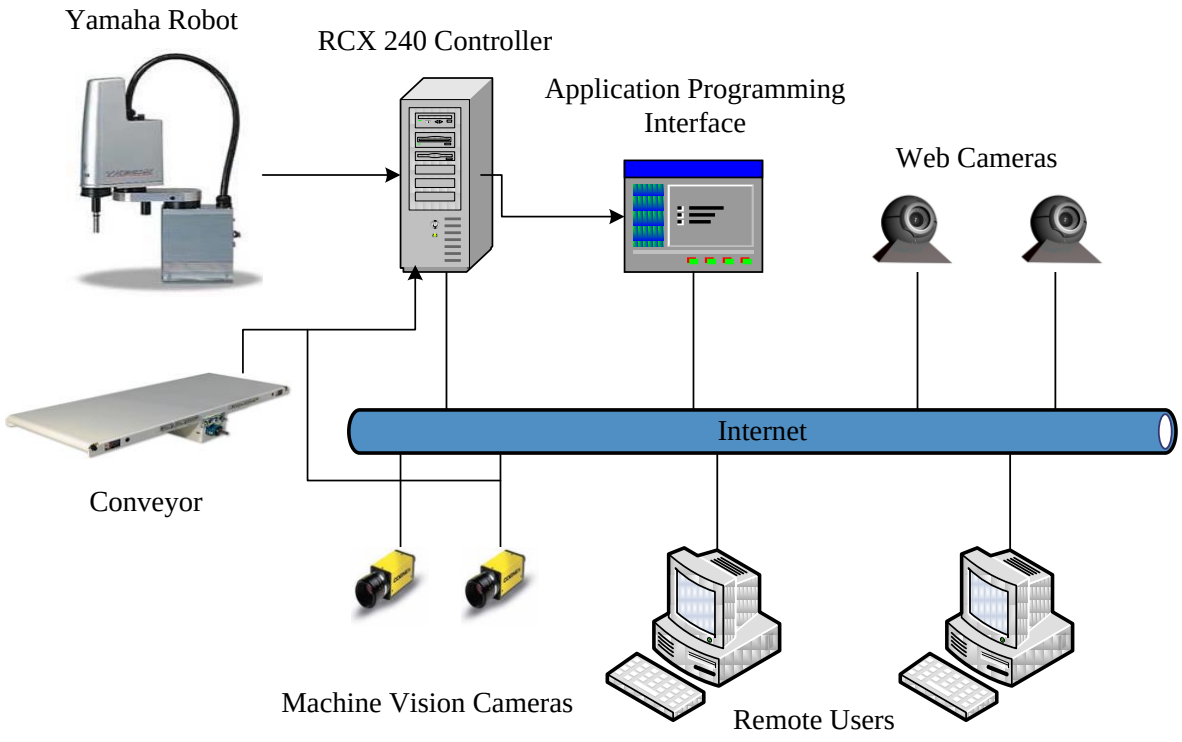


Figure 1.1. Overall setting of the inspection system

Figure 1.2 below shows the set up at ISEL (Industrial Systems Engineering Laboratory), UTEP.

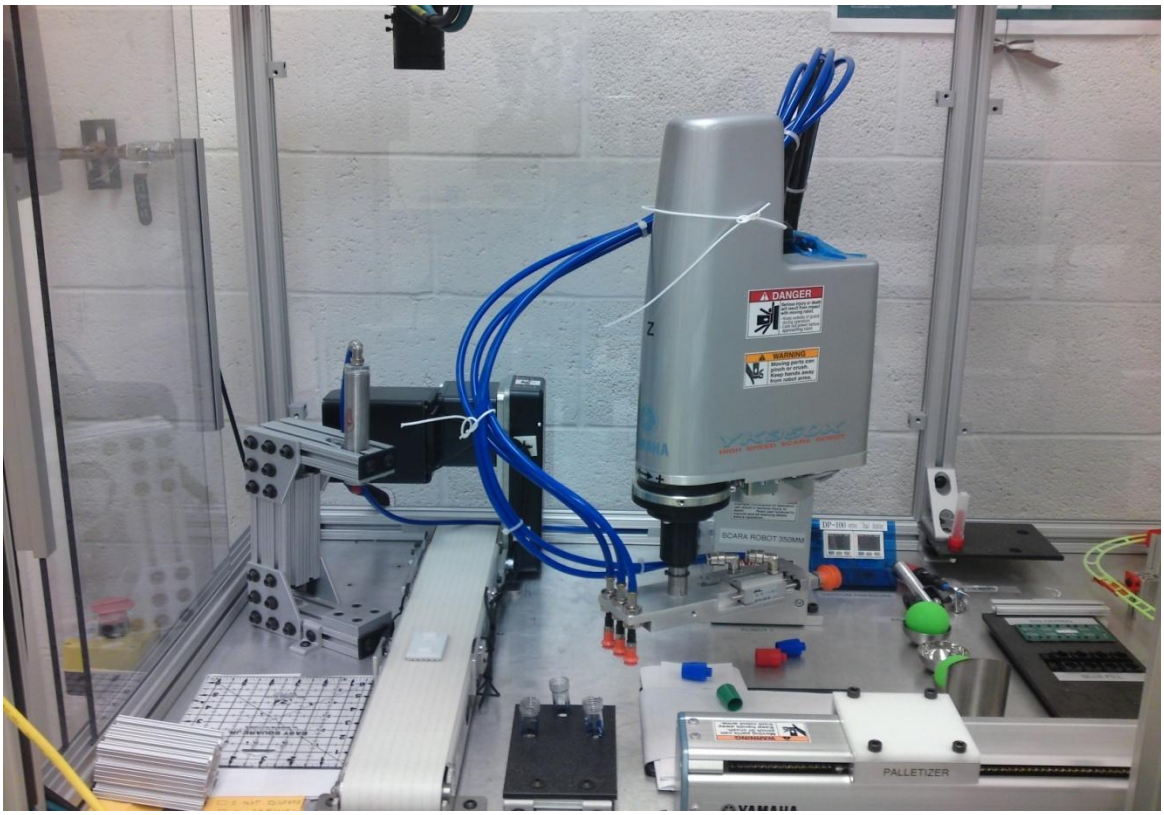


Figure 1.2. Inspection System Set Up at ISEL, UTEP.

1.3 Motivation of the research

Process of reworking a part that is not good to make it meet the requirements is common in any manufacturing industry. A part will be sent to scrap only if it cannot be reworked. Any inspection is expected to give out the result as whether the part is good, bad or can be reworked. **But, the Cognex Insight software in the current scenario does not do the multi-classification of the dimensional values obtained.** The objective of this research is to try using Support Vector Machines, a machine learning approach for prediction of the outcomes. A new approach of using index values in SVM was tested for improved classification accuracy. A completely new methodology was also tried that was found to give 100% accuracy with certain limitations.

Chapter 2

Literature Review

Achieving 100% classification accuracy is not an easy task, especially when dealing with a case where exactness matters. **Quality control in this case involves analyzing the dimensional values of features where slight difference in value makes the part fall into a different category.** The three best known methodologies are Neural Networks, Support vector Machines and Fuzzy Logic. **Support Vector Machines was preferred due to its higher generalization ability especially when the data set is small and the class overlap is scarce or non-existent [2,3].** The following section presents previous works dealing with ways and combinations of methods with SVMs towards improving classification accuracy.

2.1 Classification Accuracy using Support Vector Machines

Ming-HuwiHorng et al. [4] used multi-class fuzzy support vector machines for identifying injury to supraspinatus muscle of the rotator cuff. Five types of multi-class support vector machines along with fuzzy logic were tested for classifying the supraspinatus images into different disease groups. One against all fuzzy SVM (OAA-FSVM) was found to yield best classification accuracy of 90% compared to one against all SVM (OAA-SVM), one against all decision tree based, one against one voting based and one against one directed acyclic graph methods. Jin-Hyuk Hong and Sung-Bae Cho [5] used a new method integrating one vs. rest SVM (OVR SVM) and Naïve Bayes classifiers (NBs) for classifying cancer data. The proposed method yielded better results with an accuracy of 81.5% compared to existing OVR SVMs and

NBs worked out either individually or combined by sum and product strategies. Sukanta Mondal et al. [6] tested BLAST algorithm, I sort predictor, least hamming distance algorithm, least Euclidean distance algorithm and multi-class support vector machines for classifying conotoxins into A, M, O and T super families to find SVMs outperform other methods with an accuracy of 88.1% with squared correlation of 0.75 for five classes using jackknife cross-validation test. Kai-Quan Shen et al. [7] tested probabilistic based multi-class SVM for establishing a robust electroencephalography (EEG) based mental fatigue measurement and monitoring system. Average testing accuracy of 87.2% was achieved compared to that of 85.4% with one vs. one SVM (OVO SVM). Approach of aggregating the confidence estimates was found to yield increased classification accuracy with the value increasing with increase in the number of epochs considered for aggregation. Der-Chiang Li et al. [8] developed and tested kernel construction technique for increased classification accuracy with small data sets using support vector machines. Echocardiogram data, Wisconsin diagnostic breast cancer data, BUPA liver disorders data and Pima Indians diabetes data from UCI repository of machine learning data base were used for testing. Class possibility based kernel was found to give better classification performance compared to polynomial and Gaussian kernels on support vector machines. Cheng-Jin Du et al. [9] tested three multi classification methods called, one vs. all, one vs. one and directed acyclic graph (DAG) on the pizza images obtained through computer vision for classifying them into four categories for pizza base and five categories for pizza sauce spread and topping. One vs. one method yielded best accuracy levels of 89.17%, 87.5% and 80.83% for pizza base, sauce spread and topping respectively. C.Z. Cai et el. [10] tested SVM for classification of distinctly related proteins that include RNA-binding proteins, protein

homodimers, proteins responsible for drug absorption, proteins involved in drug delivery, drug excretion proteins, class-I drug metabolizing enzymes and class-II drug metabolizing enzymes into functional classes based on the features generated from the analysis of physiochemical properties of proteins. Classification accuracy of independent evaluation sets was found to be in the range of 86.5% to 99.4%. V. Sugumaran et al. [11] developed and tested a fault diagnostic system where critical features are selected by using decision tree algorithm based on information gain and prediction by Gaussian kernel based multiclass SVM. Accuracy of 94% was achieved. J.H. Hong et al. [12] developed a novel method for classification of finger prints dynamically by integrating Naïve Bayes (NB) classifiers and SVMs. Naïve Bayes was used for determining the sequence of the one vs. all SVM models to be evaluated based on singular points and pseudo ridges. One vs. all SVM models with Gaussian kernel is evaluated based on finger codes. Proposed method yielded an accuracy of 90.8% for five class problem and 94.9% for four class problem with 1.8% rejection during feature extraction phase of finger code from which it can be concluded that combination of different methods and features will result in increased classification accuracy. Xue-Wen et al. [13] proposed a new method combining independent component analysis (ICA) filter bank, recursive feature elimination method (RFE) and least square SVM with Gaussian kernel for feature extraction, feature selection and texture classification respectively. One against all strategy was used to combine the results from classifiers. The proposed RFE_max method was found to yield better results. Limitations with genetic systems, fuzzy logic and neural networks like requiring expensive evaluation processes, linguistic rules which are hard to generate and inability in determining the number of layers and number of neurons per layer respectively and the effectiveness of SVM in pattern recognition

drove Li-Chang Chao et al. [14] to propose a novel wafer defect recognition system using multi class SVM with a new defect cluster index. Gaussian kernel with one against one approach was used. Accuracy of 91.3725% was achieved with the proposed method compared to 62.7451% with RBF (radial basis function) neural network. Jayadeva et al. [15] proposed the extension of fuzzy proximal support vector machines methodology to a multi-class problem by using one from rest (OFR) criteria for separation of each class from the rest. Iris and Wane data were used for the test. Tuning data set was used for finding optimal values for parameters. Fivefold cross validation was used for evaluating accuracy. Fuzzy proximal SVM and Proximal SVM were both tested. Fuzzy proximal SVM was found to better accuracy levels with improved process speed. EmreComak et al. [16] proposed a new training algorithm to overcome the problems related to outliers in the training set. Training data was mapped into higher space by using RBF kernel function and simple clustering scheme based on Euclidean distance measure was used for normalization of distance values. A K number that represents the number of neighbor for i th cluster of class j are computed independently for both classes. Pair wise SVM was used for separating the hyper planes. Pair wise SVM with Euclidean distance measure was used for multi class problems. Iris, Wane and Thyroid data were used for testing. The proposed method was found to yield better results compared to FLS (Fuzzy least square)-SVM, LS-SVM and LS-SVM with K-NN (K nearest neighborhood) methods with reduced training times. In health care, it is important to know the confidence levels of an outcome as it is critical in making a decision regarding the possible diagnosis. Ben Van Calster et al. [17] evaluated the performances of algorithms like standard statistical MLR (multi-class logistic regression) with manual checks for transformations, interactions, and the linearity in logit assumption, standard binary LR (logistic

regression) models with similar checks, Bayesian MLP (multi-layer perceptron), combinations of binary Bayesian LS-SVMs (least square support vector machines) and multi class kernel logistic regression based on LS-SVM theory on the data collected at St. George's hospital in London. Binary logistic regression models combined using pair wise coupling isolated more than 80% patients as failing PUL (pregnancy of unknown location) or IUP (Intra-uterine pregnancy) with very high level of confidence leaving 20% at risk for ectopic pregnancy (EP) for which more attention is required. SVM methods are found to yield good results with MKLR (Multi-class kernel logistic regression), a close affiliate to SVM standing in top 5 for all the performance measures. David Meyer et al. [18] compared binary SVM to 16 other classification methods and 9 regression methods accessible from R software version 1.6.1 with 21 data sets used for classification and 12 for regression to conclude that simple statistical procedures and ensemble methods providing competitive results without the need for delicate and computationally expensive hyper parameter tuning. All the approaches mentioned above are summarized in the table 2.1 given below.

Table 2.1. Review of various works on SVM

Author	Method	Application	Remarks
Ming-Huwi Horng.,2009.	Various multi-class SVMs.	Classify the supraspinatus image into different disease groups.	One Against All Fuzzy SVM was found to yield the best classification accuracy of 90%.
Jin-Hyuk Hong.,	One Vs. Rest SVMs and	Classify cancer data	Better accuracy of 81.5%

Sung-Bae Cho., 2008	Naïve Bayes		compared to simple SVMs.
SukantaMondal et al.	Various methods including MSVMs	Classify conotoxins	SVMs outperformed with 88.1% accuracy.
Kai-QuanShen et al.	Probabilistic based multi-class SVM	Electroencephalography based mental fatigue measurement	PWC-SVM performed better compared to One Vs. One SVM.
Der-Chiang Li et al.	Possibility based kernel o be used with SVM	Echocardiogram data, Wisconsin diagnostic breast cancer data, BUPA liver disorders data and Pima Indians diabetes data	Proposed kernel performed better compared to Gaussian and polynomial kernials.
Cheng-Jin Du et al.	SVMs (One vs. All, One vs. One) and Directed Acyclic Graph (DAG)	Pizza quality	One vs. One found to be the best followed by DAG.
C.Z. Cai et al.	Gaussian Kernel based SVM	Classification of functionally distinct proteins	Accuracy levels of 86.5-99.4% were achieved.
V. Sugumaran et al.	Information gain and multiclass SVM	Identify faulty bearing conditions	Accuracy of 94% was achieved
J.H. Hong et al.	Naïve Bayes classifiers and OVA SVMs with Gaussian kernel	Finger print classification	Accuracy of 90.8% for five class problem and 94.9% for four class problem with 1.8% rejection

Wen Chen et al.	ICA filter bank, a new recursive feature elimination method (RFE) and LS-SVM	Texture Analysis	Proposed RFE_max method performed better.
Li-Chang Chao et al.	Multi class SVM (Gaussian Kernel and One Against One approach) with new defect cluster index	Wafer defect recognition	Proposed cluster index yielded better results.
Jayadeva et al.	Multi class proximal support vector machines	Iris data and Wane data	Proper membership value assignment improved results.
EmreComak et al.	Training algorithm based on Euclidean distance measure and K nearest neighbor with pair wise SVM	Iris, Wane and Thyroid data	Proposed method performed better.
Van Calster et al.	Probabilistic algorithms	Classification of pregnancies of unknown location	Methods affiliated to SVM are found to perform good but not superior.
David Meyer et al.	Benchmarked SVM to 16 classification and 9 regression methods	21 datasets for classification and 12 for regression	SVMs has high potential but not superior to all methods in all cases.

From the review it was found that in general, SVM was preferred over neural networks for its risk minimization capability. Alterations in the SVM and combinations with different methods were found to give increased classification accuracy and dominate other methods. However, the alterations and combinations were tested on specific applications and the results cannot be

generalized. In most cases, the methods were found to have developed to improve classification accuracies in particular cases they dealt with. It was observed authors mentioning that simple statistical approaches specific for an application may outperform all other algorithms [18]. It can be concluded that, alteration in SVM depending on the data or a problem specific simple statistical approach may lead to improved classification accuracy.

Chapter 3

Methodology

In the current section, SVM and two new approaches were explained. In the first case, the data extracted was modified to obtain new set of values and was used for analysis with SVM. Secondly, a completely new methodology for classifying the data was developed. This chapter illustrates the various methodologies including the traditional SVM process.

3.1 Support Vector Machines

In this method, Support vectors machines methodology is applied to the training data set to obtain the classifier equation and any incoming data point of a part will be classified based on those classifier equations. Figure 3.1 below shows the flow diagram of the procedure using the part from the case study as example.

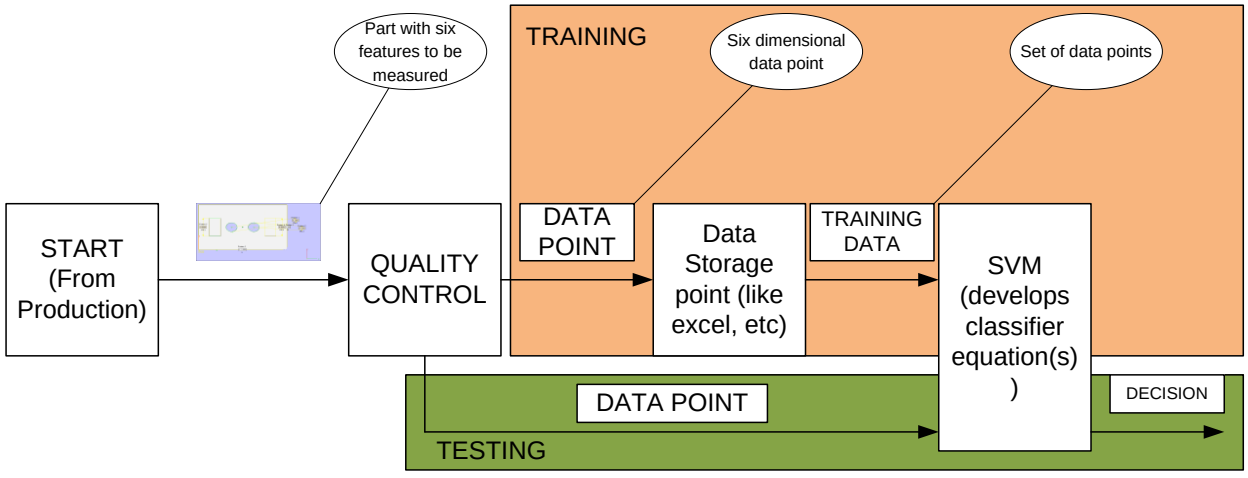


Figure 3.1. Flow diagram of the procedure using SVM.

3.1.1 Binary Support Vector Machines

The SV algorithm is a nonlinear generalization of the Generalized Portrait algorithm developed in the sixties [19]. It is based on the frame work of statistical learning theory, which has been developed over the last three decades by Vapnik and Chervonenki [20]. The present form of SVM was developed to a great extent by Vapnik and co-workers at AT&T Bell Laboratories [21-23]. It is this industrial context that led the research on SVM to be oriented towards real-world applications. The decision function for the support vector classification is of the form

$$f(x) = \mathrm{sgn}(\sum_{i=1}^{m} y_i \alpha_i \langle \Phi(x), \Phi(x_i) \rangle + b) \quad (1)$$

Where

m is the total number of data points considered for training,

sgn (x) = 1, if x is positive,

sgn(x) = -1, if x is negative,

y_i = ±1 depending on the class to which ith data point belongs to in the output space,

α_i is the Lagrange multiplier obtained by solving the quadratic problem,

$\langle x, y \rangle$ is the dot product of x and y,

$\Phi(x)$ is the mapping function that maps input value x into higher dimensional space,

b is a constant.

Equation (1) gives rises to equation (2).

$$f(x) = \text{sgn}(\sum_{i=1}^{m} y_i \alpha_i k(x, x_i) + b) \quad (2)$$

Where

$$k(x, x_i) = \langle \Phi(x), \Phi(x_i) \rangle .$$

A mapping function is used to map the data points into higher dimensional feature space where they can be linearly separated. Figure 3.2 below demonstrates the use of a mapping function.

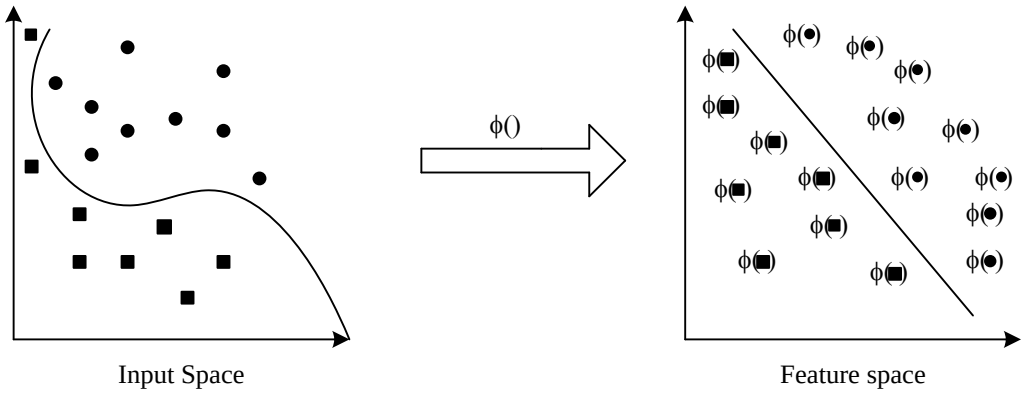


Figure 3.2. Demonstration of mapping function.

Equation (2) leads to the following quadratic problem

$$\text{Maximize}_{\alpha \in R^m} W(\alpha) = \sum_{i=1}^{m} \alpha_i - 1/2 \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \quad (3)$$

$$\text{Subject to } \alpha_i \geq 0 \text{ for all i = 1,……, m and } \sum_{i=1}^{m} \alpha_i y_i = 0 \quad (4)$$

Where

α is the Lagrange multiplier,

y = ±1 is the class index value in the output space,

x is the input data point vector,

m is the total number of data points (vectors),

$k(x_i, x_j)$ is the kernel function.

To allow for the possibility of examples violating, slack variables were introduced.

$\xi_i \geq 0$ for all i = 1, … , m (5)

So that the constraints get relaxed to

$y_i(\langle w, x\rangle + b) \geq 1 - \xi_i$ for all i = 1, … , m (6)

Where

$\langle w, x\rangle = \sum \alpha_i y_i \langle x_i, x\rangle$ and

$w = \sum \alpha_i y_i x_i$

Figure 3.3 demonstrates the working principle of SVM with slack variables included.

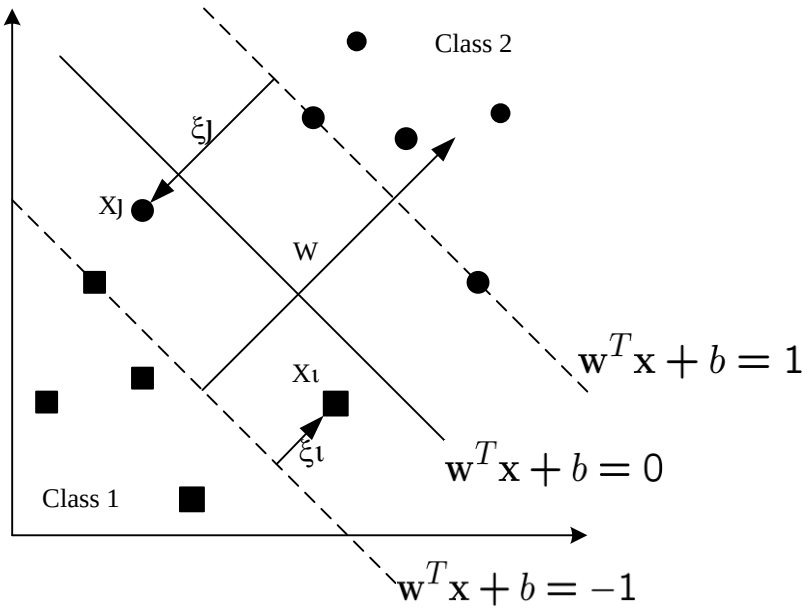


Figure 3.3. Geometric representation of SVM with slack variables included.

In the figure 3.3 shown above, ξ_i, ξ_j are the amount of slacks for the data points X_i and X_j respectively, where X_i belongs to class 1 and X_j belongs to class 2. $W^T X + b = 0$ is the classifier plane that separates the classes 1 and 2. $W^T X + b = 1$ and $W^T X + b = -1$ are the equations used to maximize the separation space between the two classes where W is the vector shown in the figure.

A good generalizing classifier is found by controlling both the classifier capacity and the sum of the slacks. The latter can be used to set the upper bound on training errors. The soft margin classifier can be obtained by minimizing the objective function

$$\tau(w,\xi) = \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{m} \xi_i \tag{7}$$

Subject to constraints (5) and (6),

Where

C > 0 determines the trade-off between margin maximization and training error minimization

Incorporation of kernel and rewriting in terms of Lagrange multipliers leads to problem of maximizing (3),

Subject to $0 \le \alpha_i \le C$ for all i = 1, …, m and $\sum_{i=1}^{m} \alpha_i y_i = 0$ (8)

The only difference from the separable case being the upper bound C on the Lagrange multipliers α_i. By this, the influence of individuals that could be outliers gets limited. The solution takes the form (1). Threshold b can be computed by exploiting the fact that for all support vectors, x_i with $\alpha_i <$ C, the slack variable ξ_i is zero, and hence

$$\sum_{j=1}^{m} \alpha_j y_j k(x_i, x_j) + b = y_i \quad (9)$$

Choosing b amounts to shifting the hyper plane and [24] suggests shifting the hyper plane such that support vectors with zero slack variables lie on the ±l lines.

3.1.2 Multi-Class Support Vector Machines

As support Vector machines employ direct decision functions, an extension to multi-class is not straight forward. There are roughly four types of support vector machines that can handle multi-class problems. In one-against-all support vector machines, data of each class is compared with all the rest of the data as single class. In pair wise support vector machines, data from each class is compared with that of all other classes. In error correcting output codes SVM, “don’t care” outputs will be introduced and a unified scheme that includes one-against-all and pair wise

formulations are used. In all at once SVM, all the decision functions of classes are solved simultaneously. More information can be found in [3]. In the current study, pair wise support vector machines was used. Pair wise strategy was selected keeping in view of the superior results obtained by Chih-Wei Hsu and Chih-Jen Lin [25] when tested on standard data like iris, wine, dna, shuttle, etc. compared to other strategies.

In pair wise support vector machines, a total of n (n-1)/2 decisions functions will be determined for all the combinations of pairs of classes (using binary SVM). $D_{ij}(x) = -D_{ji}$, where $D_{ij}(x)$ is the decision function for class i against class j and i ≠ j. After arriving at decision functions, for classification of data into classes, a voting strategy was employed.

X is classified into the class

$$\arg^{\max}_{i=1,..,n} D_i(x) \tag{10}$$

for

$$D_i(x) = \sum_{j \neq i,\, j=1}^{n} sign(D_{ij}(x)) \tag{11}$$

Where

$D_{ij}(x)$ is the decision function for class i against class j.

sign(x) = 1 for x ≥ 0

sign(x) = -1 for x ≤ 0.

In case two classes have same identical number of votes, the class with the smallest index is selected to avoid unclassifiable regions [25].

3.1.3 Example

The following example demonstrates the SVM methodology used in the current study. Let us consider six one dimensional data points:$x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 4$, $x_5 = 5$, $x_6 = 6$ where x_1, x_2 belongs to class 1, x_3, x_4 belongs to class 2 and x_5, x_6belongs to class 3.Since, we use pair wise support vector machines, we solve for classes 1 and 2 first. Let us consider a polynomial kernel with degree=2, gamma=1 and coefficient=1 which results in K(x, y) = (xy+1) ^2. We assume C=100.

We can solve for support vectors by using the equation

Max $\sum_{i=1,2,3,4} \alpha_i - 1/2 \sum_{i=1,2,3,4} \sum_{j=1,2,3,4} \alpha_i \alpha_j y_i y_j (x_i x_j + 1)^2$

Subject to $0 \le \alpha_i \le 100$, $\sum_{i=1,2,3,4} \alpha_i y_i = 0$

By using a quadratic programming solver, we get $\propto_1 = 0$, $\propto_2 = 0.074$, $\propto_3 = 0.074$, $\propto_4 = 0$. Figure 3.4 below shows the screen shot from maple software.

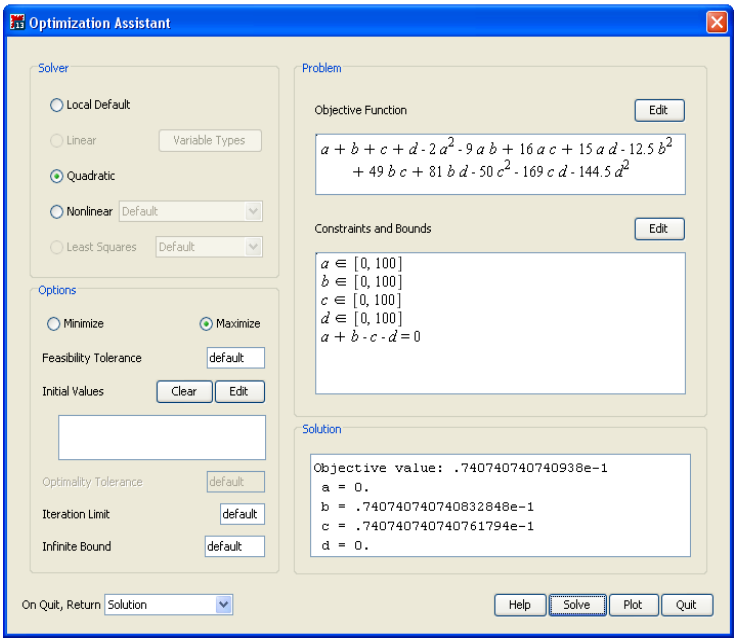


Figure 3.4. Screen shot from Maple 13 where a=$\propto_1$, b=$\propto_2$, c=$\propto_3$, d=$\propto_4$

By definition, non-zero values of alpha are considered as support vectors.

The decision function will be

$$D_{12} = 0.074(+1)(2y+1)^2 + 0.074(-1)(3y+1)^2 + b$$

$$D_{12} = -0.37y^2 - 0.148y + b$$

'b' can be recovered by solving for $D_{12}(2) = 1$ or $D_{12}(3) = -1,$ as all these points lie on

$y_i(\langle w, x_i \rangle + b) = 1$ and both give b=2.77.

Hence the classifier equation is $D_{12} = -0.37y^2 - 0.148y + 2.77$

Similarly,

$D_{13} = -0.084y^2 - 0.024y + 1.384$ and

$D_{23} = -0.216y^2 - 0.048y + 4.64$

From $D_{ij} = -D_{ji}$,

$D_{21} = 0.37y^2 + 0.148y - 2.77$,

$D_{31} = 0.084y^2 + 0.024y - 1.384$,

$D_{32} = 0.216y^2 + 0.048y - 4.64$.

Suppose a new data point x = 2.2 is to be tested.

$D_{12} = 0.6536$,

$D_{13} = 0.9246$,

$D_{21} = -0.6536$,

$D_{23} = 3.48896$,

$D_{31} = -0.9246$,

$D_{32} = -3.48896$.

From which $D_1 = 2$, $D_2 = 0$ and $D_3 = -2$. Hence, data point x = 2.2 is classified into class 1.

3.2 Support Vector Machines with modified data

The deficiency in accuracy with support vector machines is due to the improper development of separation boundary lines as the data points are close and the difference between values of different categories is very low. Hence, a new procedure of modifying the data points was developed so that the values of all the features for a particular category or class fall into a definite range and can be separated out easily by using support vector machines. Prior information about the limits is necessary, a general phenomenon in any industry. The procedure is an iterative process and has to be repeated twice. In the first iteration, the good ones were separated out from the rest and in the second iteration, the parts to be reworked are separated leaving out the bad ones. In the first iteration, both rework and bad parts will be evaluated under a single category. Figure 3.5 below shows the flow diagram of the proposed procedure using the part from the case study as example.

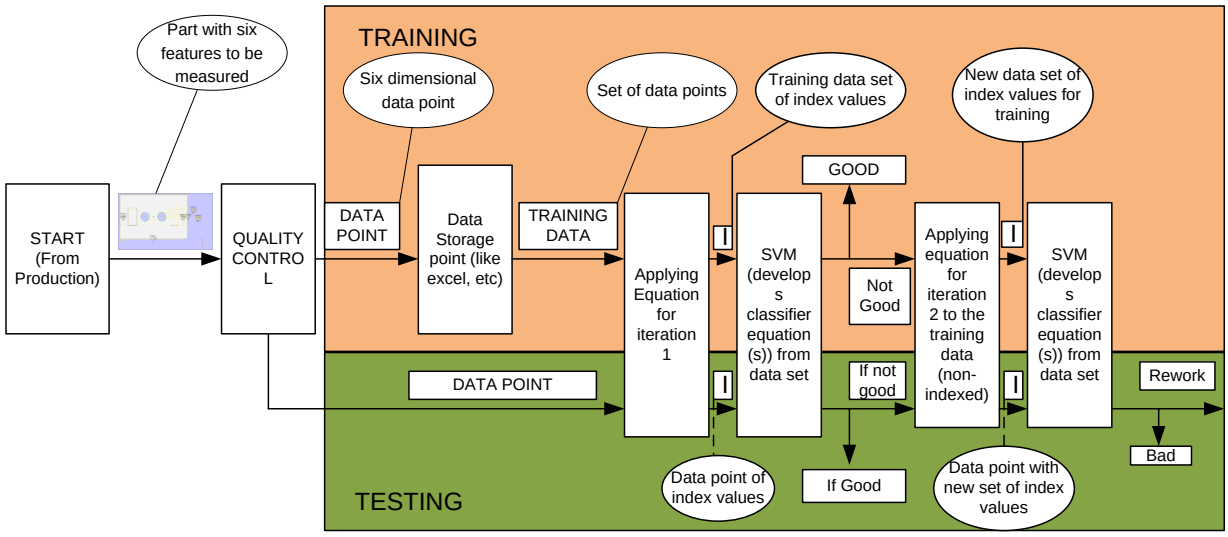


Figure 3.5. Proposed procedure that uses modified data with SVM

In the first iteration, equation (12) is applied to the original data to obtain the index values for each value of a feature in the sample point. The values obtained will be such that if the measured dimension of a particular feature is within limits, the value will be less than or equal to1 or else more than one. The idea is to bring uniformity in values of all the features so that SVM can classify the part into a category easily.

$$\frac{|X_4 - X|}{X - X_1} \tag{12}$$

Where

X_4 is the measured dimension of a particular feature in a data point.

X is the average of lower and upper tolerance limits.

X_1 is the lower tolerance limit.

X_2 is the upper tolerance limit

X_1, X_2, and X_3 are lower tolerance, upper tolerance and rework limits for a feature respectively. X is the average of lower and upper tolerance limits. X_4 is the measured dimension of a particular feature in the sample point. Rework has no role to play in the first iteration. SVM will be applied on the modified data containing index values for first iteration. The parts that are classified as not good will be exported to the send iteration.

In the second iteration, rework limit will replace the upper tolerance limit if greater than it or will replace lower tolerance limit if less than it. The new average of limits is called X^1 and will replace X in equation (12) to form equation (13). The new lower tolerance limit is called X_1^1 and will replace X_1 in equation (12). New set of index values will be calculated for the parts categorized as not good by applying equation (13) to the dimensions of the features.

$$\frac{|X_4 - X^1|}{X^1 - X_1^1} \tag{13}$$

Where

X_4 is the measured dimension of a feature (on a part) in a data point.

X^1 is the average of new upper and lower tolerance limits.

X_1^1 is the new lower tolerance limit.

SVM will be applied on the new set of index values obtained to separate out the rework parts from bad ones.

3.3 New Methodology

A simple statistical approach was proposed keeping in view of the current experiment and the kind of data. **The proposed approach consists of two iterations and completely eliminates the need for training, saving lot of time and labor**. First iteration is meant to isolate the good

parts from the rest. Second iteration is meant to separate the reworkable parts from bad ones. Suppose X_1, X_2, X_3 are dimension limits for a particular feature to be known prior to getting into the process of prediction. X_1 and X_2 are lower and upper tolerance limits. X_3 is the rework limit for the part that can be reworked. X is the midpoint of lower and upper limits. X_4 is the actual dimension measured for a feature. For that feature that does not have a rework limit, lower tolerance limit is considered as rework limit. In the first step, equation (14) is applied to each individual feature of the sample point. The values obtained will be rounded off to two decimal points. If any of the features is not within the limits, the corresponding value will be greater than zero or else will be equal to zero. The sum of values generated for all the features is calculated. If the sum of the values is equal to zero, indicates a good part or else will be processed through step 2 for further analysis.

$$Sin[\{\frac{|X_4 - X|}{X - X_1}\} * c] \tag{14}$$

Where

c is a constant equal to 0.57296 degrees or 0.0100001 radians.

X_4 is the measured dimension of a feature (on a part) in a data point.

X is the average of lower and upper tolerance limits.

X_1 is the lower tolerance limit.

In second step, for each individual feature, if the rework limit is less than the lower tolerance limit, then the lower tolerance limit value is replaced with the rework limit value. Similarly, if

the rework limit is higher than the upper tolerance limit, then the upper tolerance limit value is replaced with the rework limit value. New midpoint X^1 is calculated. The obtained values are substituted into the equation (15) obtained by changing the variables in equation (14) accordingly.

$$Sin[\{\frac{|X_4 - X^1|}{X^1 - X_1^1}\} * c] \quad (15)$$

Where

X_4 is the measured dimension of a feature on a part (in a data point).

X_1^1 stands for the new lower tolerance limit.

X^1 stands for the average of new lower and upper tolerance limits and

C is a constant equal to 0.57296 degrees or 0.0100001 radians.

Sum of the corresponding values of the features is calculated as in iteration 1. If the sum obtained equals to zero, the sample is classified as rework, else a bad part.

Figure 3.6 below shows the geometric representation of the proposed methodology. The methodology was demonstrated using normal distribution as measurements of features on parts produced by any controlled production process will resemble normal distribution and is the perfect chart to demonstrate the quality control process.

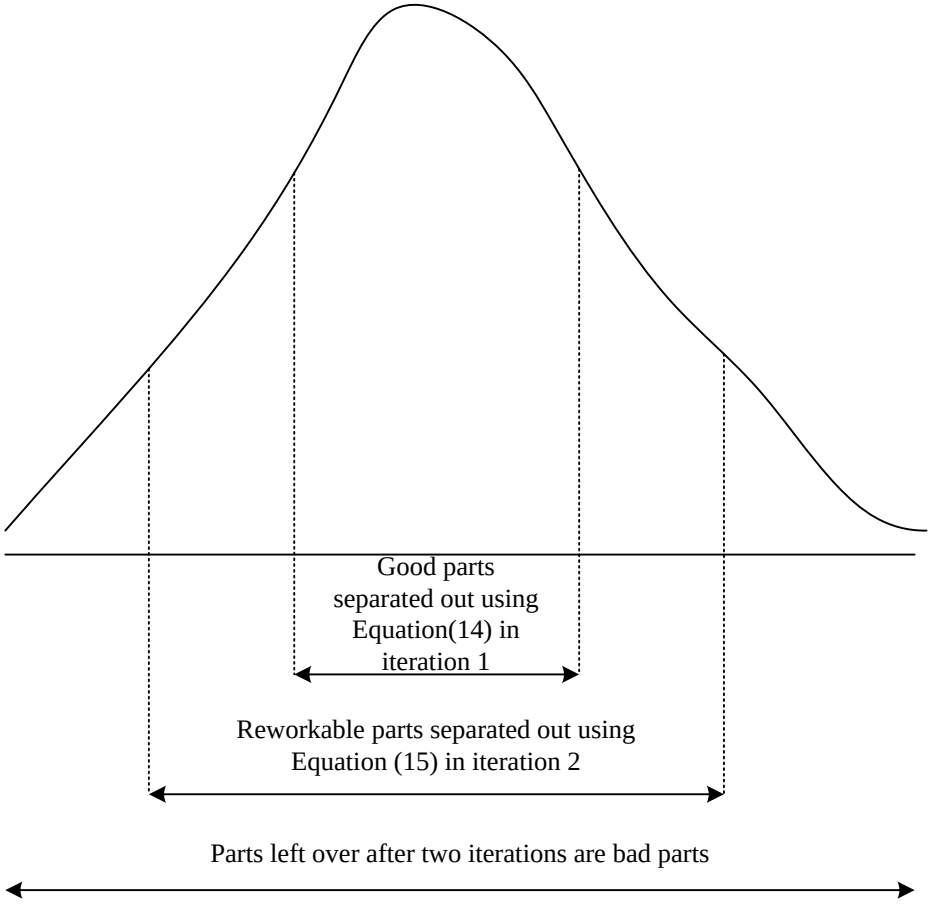


Figure 3.6. Geometric representation of the proposed sine methodology.

Figure 3.7 below shows the flow diagram of the proposed sine methodology using the part from the case study as example.

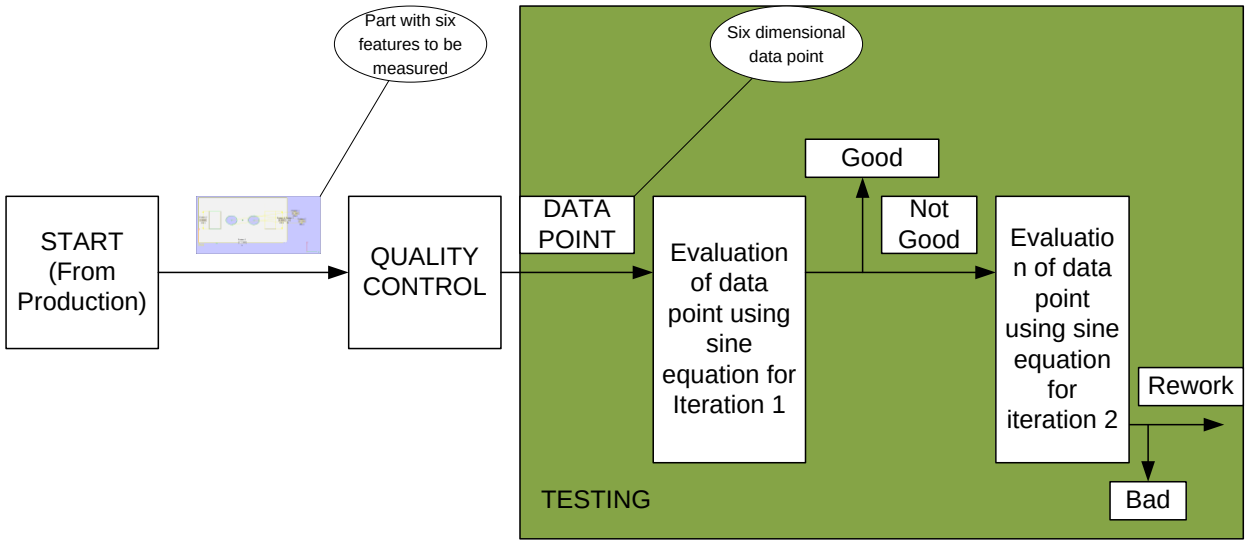


Figure 3.7. Flow diagram of the proposed sine methodology.

3.3.1 Necessity for a trigonometric function

In the current methodology, equations 13 and 14 are obtained by multiplying equations 11 and 12 respectively with constant 'c' and applying sine function to them. **By applying equations 11 or 12 to features in a data point (for a part), index values are obtained. For features that lie in the range of a particular class, index values are obtained between 0 and 1 and for features that does not lie within the range as greater than 1. After arriving at index values, it is required to classify the parts (data points).** The main essence of the new methodology is that it eliminates the need for SVM in performing classification, process which requires analyzing the combined effect of index values of all the features in a data point (for a part). Only if index values for all the features are between 0 and 1, a part (data point) can be classified into a particular class. But, to verify at data point level, a combined value of index values of all the features is required. By summing up the index values of all the features in a data point (for a part), a random value is obtained, based on which, making a decision is not possible. Hence, **it is**

required to reduce the index values for features that are between 0 and 1 to zero, so that summing up all the resultant values for features of a data point (part) will be equal to zero and can be verified easily. Hence, to modify the index values, a function that can minimize the index values that are less than or equal to 1 by a decimal point or two is required so that the values obtained can be rounded down and then summed up to make a final decision. **The importance of using a function is to turn the value '1' into a value with same number of decimals as for example with '0.9'.** Trigonometric functions are found to serve the purpose.

In general, all the trigonometric functions are positive in first quadrant where $\theta \in \{0, 90\}$ in degrees or $\{0, \pi/4\}$ in radians i.e., $\sin \theta \in \{0, 1\}$, $\text{Cos } \theta \in \{1, 0\}$, $\text{Tan } \theta \in \{0, \infty\}$, $\text{cosec } \theta \in \{\infty, 1\}$, $\sec \theta \in \{1, \infty\}$, $\cot \theta \in \{\infty, 0\}$. However, from the above scenario, it can be concluded that a function with increasing values with increase in θ is required. Sine, tangent and secant functions have increasing values and can be used to form equations 13 and 14 with appropriate c values as shown below.

Sine function: With sine function, to convert index value '1' into a value with one decimal, c equals to 5.7391 ~ Inverse Sine (0.1) can be used and to convert '1' into a value with two decimals, c equals to 0.57296 ~ Inverse Sine (0.01) can be used.

Tangent function: With tangent function, to convert index value '1' into a value with one decimal, c equals to 5.71059 ~ Inverse Tan (0.1) can be used and to convert '1' into a value with two decimals, c equals to 0.57293 ~ Inverse Tan (0.01) can be used.

Secant function: With secant function, to convert index value '1' into a value with one decimal, c equals to 0.01186791 ~ Inverse Sec (0.1) can be used and to convert '1' into a value with two decimals, c equals to 0.011182301 ~ Inverse Sec (0.01) can be used.

For all the above functions, values to convert '1' into values with more than two decimals can also be obtained, but will not affect the final result and hence are not mentioned here. The c values given above were found to work up to the values with 4^{th} decimal point difference. If using with values much closer i.e., values with decimal points more than 4, a more accurate c value as well may be chosen. The example below demonstrates the proposed methodology with the above mentioned sine function and corresponding c value for values with 4 decimal points difference.

3.3.2 Example

Suppose a data point (of a part with six features to be measured on it) with dimensional values y1=0.2085, y2=0.21, y3=1.952, y4=1.184, y5=0.486, y6 =0.486 is given with limits as shown in the figure 3.8 below.

Upper tolerance limit	0.2685	0.2665	1.9745	1.245	0.519	0.5195
Lower tolerance limit	0.2085	0.2065	1.9145	1.185	0.459	0.4595
Rework limit	0.1785	0.1765	2.0045	1.275	0.459	0.4595

Figure 3.8. Limits for dimensions of features on a part.

Since the proposed methodology involves application of equations 13 and 14 to all features of a data point iteratively, let us consider y1 for detailed analysis. Here, X_4 = y1 for equation 13.

From the values for upper and lower tolerance limits,

X = (0.2685+0.2085)/2 = 0.2385.

Hence, we get Index value of feature y1 for iteration 1 as

|(0.2085-0.2385)|/(0.2385-0.2085) = 1,

where X_4 (measured dimension) and X_1 (lower tolerance limit) are both equal to 0.2085.

Multiplying index value '1' with 0.0100001 (for converting into two decimal values) and applying sine will result in 0.009999933.

Rounding down the value 0.009999933 to two decimal places will result in a value 0.00.

Similarly, for y2, y3, y4, y5 and y6, values 0.00, 0.00, 0.01, 0.00 and 0.00 are obtained respectively.

For y4, 0.01 is obtained as 1.184 is not within 1.185 and 1.245.

Summing up the values for y1 to y6 will result in 0.01, from which it can be concluded as a not good part and hence is moved to iteration 2.

In the second iteration, new value of X called X^1 is calculated, with rework limit replacing the upper or lower tolerance limit.

Let us consider y4 for this iteration.

X^1 = (1.275+1.185)/2 = 1.23 is obtained

Where

X_1^1 = 1.185.

X_2^1 = 1.275.

New index value for second iteration for feature y4 is obtained as

|(1.184-1.23)|/(1.23-1.185) = 1.022222.

Multiplying index value with 0.0100001 and applying sine will result in 0.010333.

Rounding down the obtained value to two decimals will give 0.01.

Similarly, 0.00, 0.00, 0.00, 0.00 and 0.00 are obtained for y1, y2, y3, y5 and y6 respectively. Summing up the values will result in 0.01 and hence is concluded as a bad part.

From the example, it can be noted that the given c value worked well for the dimensions as close as with 4th decimal difference between them.

Chapter 4

Case Study

To demonstrate the proposed methodologies and their classification capabilities over multi-class SVMs for quality control in an industrial set up, a part that contains all the three types of features was designed i.e., positive, negative and standard features. Positive features are those created by adding material in a production house (ex: extrusions). In the other way, features that can be reduced in size to meet the specifications are defined as positive features in the present case. Negative features are those created by removing the material (ex: holes on a part, formed by taking out the material by drilling). In the other way, features that can be increased in size to meet the specifications (ex: holes can be made larger, if small) are defined as negative features. Positive features have a rework limit on lower side and negative features have rework limit on the higher side of the dimension limits. A term “standard feature” is coined and is defined as a feature that cannot be reworked at all. The part was designed using Google SketchUp and modified using Magics software version 14.01 to produce parts with varied dimensions. Figure shows the part design file with the dimensions of six features considered for the current case study.

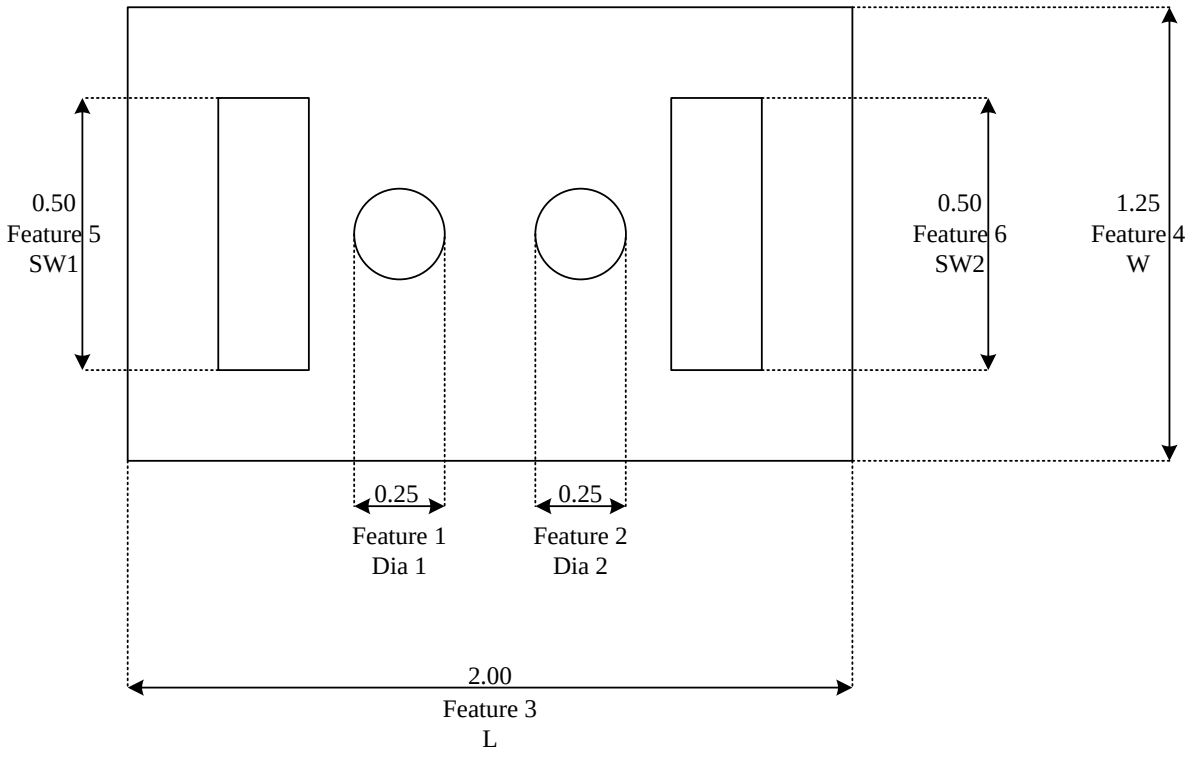


Figure 4.1. Part Design and dimensions of various features of a good (basic) part. Length of part (L): 2 inches, width of part (W): 1.25 inches, Diameter of the circles: 0.25 inches (Dia1, Dia2), width of extruded rectangles (SW1, SW2): 0.5 inches.

Three types of features were identified for the current purpose. Length and width of the part were considered as positive features. Diameters of the two circles were considered as negative features and width of extruded rectangles were considered as standard features for the current case study. Though the width of the extruded rectangles can be considered as positive features, they considered as standard features keeping in view of the requirement and the fact that change in the dimensions of the feature may affect the surface roughness. The part design was edited to produce parts with varied dimensions. Part dimensions were varied by 0.03 inches or 0.06 inches to produce rework and bad parts. The digital files created were realized using Fused Deposition

Modeling machine version 3000, a rapid prototyping machine that creates parts layer by layer using extrusion technique. Figure 4.2 below shows the parts being produced.

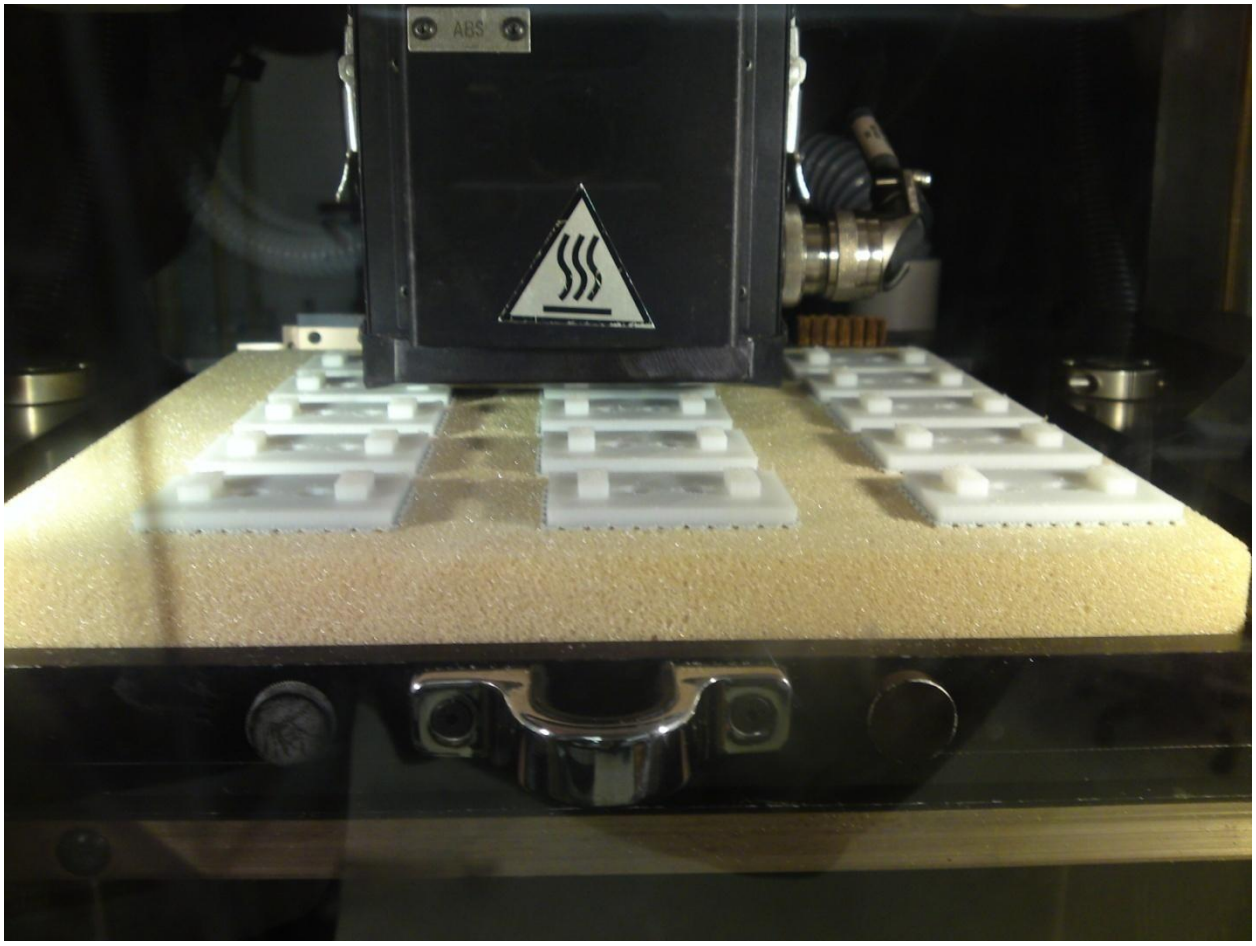


Figure 4.2. A close up view showing the FDM 3000 head building the parts.

A total of 103 parts were produced and the dimensions of the six features marked were measured using the inspection setup described in chapter 1. The data collected is shown in Appendix II. By analyzing the dimensions, 44 of the parts were identified as good, 28 as rework and 31 as bad. Two third of the parts were marked as train and one third as test for analyzing

with various methodologies. Marking is meant to maintain standard. Parts marked as train were used for training and the rest for testing.

Chapter 5

Analysis

Analysis of the data obtained with the three methods and the results generated are explained in the current chapter. For performing the analysis with SVM, commercially available software, STATISTICA 9 was used. All the four types of kernels were tried to find out the best classifier that provides superior classification accuracy.

5.1 Support Vector machines

The data obtained from the case study explained above was analyzed with various kernels without making any modifications to the data.

5.1.1 Linear Kernel

The data was analyzed with linear kernel. Fivefold cross validation was employed. Software enables to give a range and an increment for the training parameter through which it automatically selects the value that generates the best results. A range of 0 to 500 with an increment of 0.01 was given. Overall accuracy of 84.466% was achieved. Figure 5.1 below shows the results obtained.

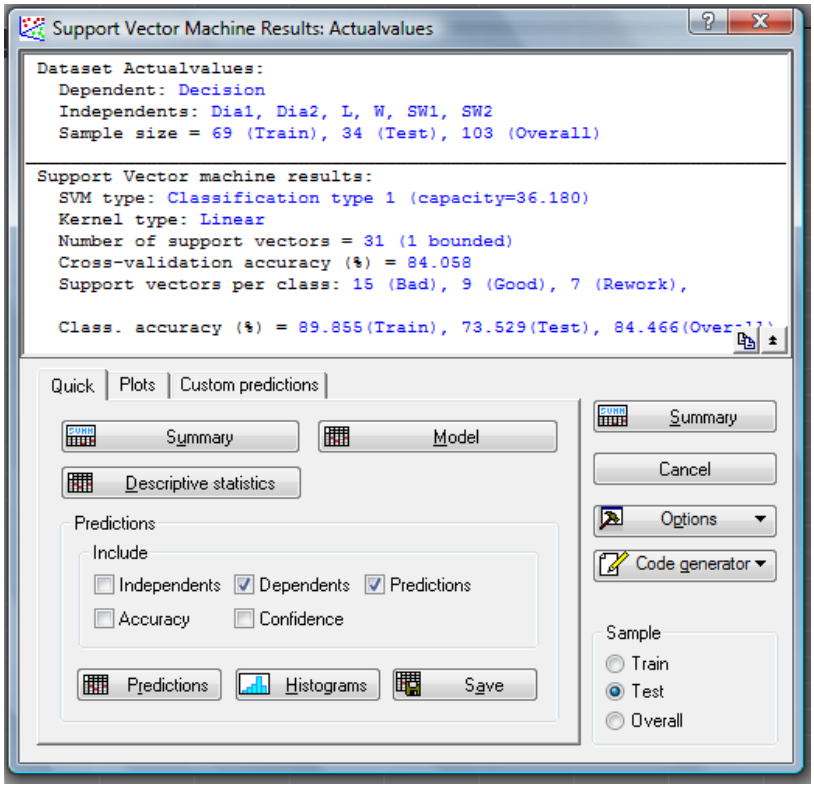


Figure 5.1. Result obtained with linear kernel

5.1.2 Polynomial Kernel

With polynomial kernel, user is required to provide the values for degree, gamma and coefficient parameters. Several iterations were run by varying degree from 1 to 25. Table 5.1 below shows the results obtained for various iterations with different sets of values. In the column for number of support vectors, numbers in brackets indicate the number of bounded vectors. The set of values that generated highest accuracy with least number of support vectors was highlighted.

Table 5.1. Results obtained with different sets of values for degree, gamma and coefficient

No	Degree	Gamma	Coefficient	Training Parameter	Training Accuracy	Testing Accuracy	Overall Accuracy	No of SVs
1	1	0.1	0	75.6	89.855	73.529	84.466	42(5)
2	1	0.1	0.1	75.7	89.855	73.529	84.466	42(5)
3	1	0.1	0.5	75.6	89.855	73.529	84.466	42(5)
4	1	0.1	1	75.6	89.855	73.529	84.466	42(5)
5	1	0.1	2	75.6	89.855	73.529	84.466	42(5)
6	1	0.1	5	75.6	89.855	73.529	84.466	42(5)
7	1	0.1	10	75.6	89.855	73.529	84.466	42(5)
8	1	0.1	25	75.6	89.855	73.529	84.466	42(5)
9	1	0.5	0	72.4	89.855	73.529	84.466	31(1)
10	1	1	0	36.2	89.855	73.529	84.466	31(1)
11	1	2	0	18.1	89.855	73.529	84.466	31(1)
12	1	5	0	7.3	89.855	73.529	84.466	31(1)
13	1	10	0	3.7	89.855	73.529	84.466	31(1)

14	1	25	0	64.9	79.71	58.824	72.816	35(0)
15	2	0.1	0	97.8	82.609	70.588	78.641	50(11)
16	2	5	10	0.8	91.304	73.529	85.437	27(0)
17	2	10	10	1.6	98.551	73.529	90.291	28(0)
18	2	15	15	0.1	91.304	73.529	85.437	28(0)
19	3	15	15	0.1	100	73.529	91.262	29(0)
20	4	15	15	0.1	100	73.529	91.262	30(0)
21	5	17.5	17.5	0.1	100	73.529	91.262	32(0)
22	5	20	20	0.1	100	73.529	91.262	32(0)
23	6	20	20	0.1	100	73.529	91.262	34(0)
24	8	30	30	0.1	100	73.529	91.262	30(0)
25	10	45	45	0.1	100	76.471	92.233	26(0)
26	12	60	60	0.1	100	76.471	92.233	25(0)
27	**13**	**65**	**65**	**0.1**	**100**	**76.471**	**92.233**	**24(0)**
28	14	70	70	0.1	98.551	73.529	90.291	24(0)
29	15	75	75	0.1	100	76.471	92.233	24(0)

30	16	80	80	0.1	100	76.471	92.233	24(0)
31	17	85	85	0.1	98.551	73.529	90.291	24(0)
32	20	100	100	0.1	95.652	76.471	89.32	23(0)
33	21	105	105	0.1	97.101	79.412	91.262	24(0)
34	23	115	115	0.1	94.203	73.529	87.379	22(0)
35	25	125	125	0.1	95.652	73.529	88.35	24(0)

From the table, it can be found that the best overall classification accuracy of 92.233% was obtained. Figure 5.2 below shows the screen shot of the best result with STATISTICA.

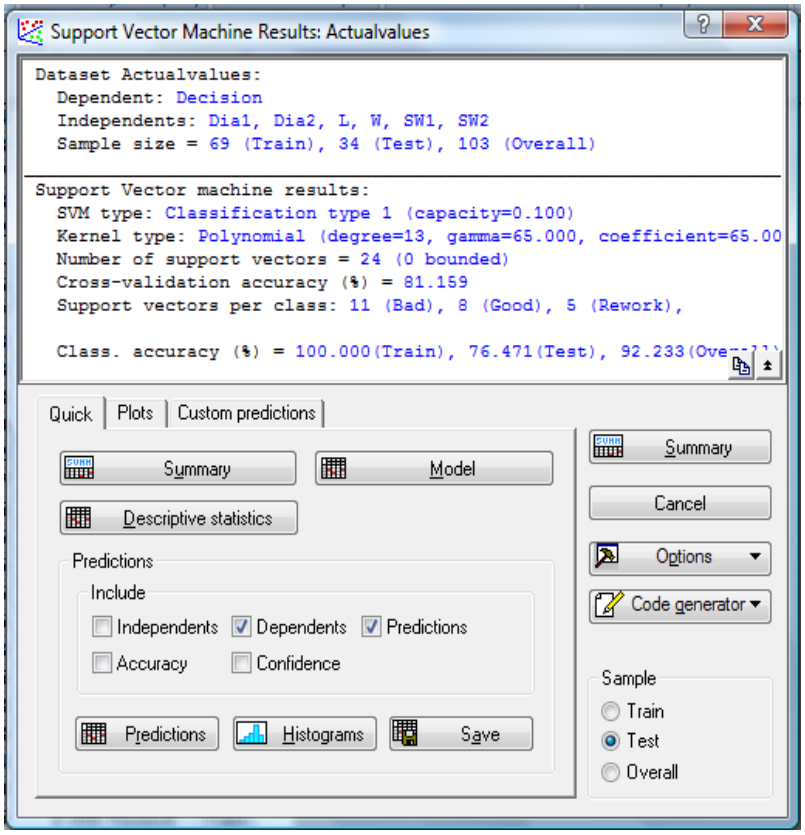


Figure 5.2. Screen shot of the best result with polynomial kernel

5.1.3 Radial Basis Kernel

With radial basis function kernel, user is required to provide the gamma value and a range for the training parameter. Several iterations were run with gamma value ranging from 0.01 to 25. Results were tabulated as shown in the table 5.2 below. The set of values that gave the best result was highlighted.

Table 5.2. Results obtained with radial basis kernel

Trial No	Gamma	Training Parameter	Training Accuracy	Testing Accuracy	Overall Accuracy	No of support vectors
1	0.1	44.3	89.855	73.529	84.466	41(5)
2	0.25	40.7	89.855	73.529	84.466	34(1)
3	0.5	46.7	91.304	70.588	84.466	30(0)
4	0.75	45.4	92.754	76.471	87.379	34(0)
5	0.8	41.7	92.754	76.471	87.379	34(0)
6	1	31.4	92.754	73.529	86.408	34(0)
7	1.5	7.8	91.304	85.294	89.32	37(0)
8	2	0.9	88.406	79.412	85.437	51(9)
9	2.75	0.9	89.855	82.353	87.379	50(9)
10	3	1.9	89.855	82.353	87.379	44(3)
11	3.25	1.7	89.855	82.353	87.379	44(3)
12	3.5	1.5	89.855	82.353	87.379	44(4)
13	3.75	1.3	91.304	82.353	88.35	45(4)

14	4.5	1.2	91.304	82.353	88.35	48(5)
15	5	1.3	91.304	82.353	88.35	50(4)
16	6	1.1	91.304	82.353	88.35	52(5)
17	**6.5**	**7.5**	**100**	**76.471**	**92.233**	**51(0)**
18	6.75	7.6	100	76.471	92.233	52(0)
19	7	7.6	100	73.529	91.262	51(0)
20	8	11.7	100	73.529	91.262	56(0)
21	8.15	13.2	100	76.471	92.233	56(0)
22	9	1	92.754	79.412	88.35	57(3)
23	9.5	1.2	94.203	79.412	89.32	58(1)
24	10	3.9	97.101	76.471	90.291	60(0)
25	11	3.6	98.551	73.529	90.291	58(0)
26	12	3.3	98.551	70.588	89.32	58(0)
27	12.75	3.3	100	70.588	90.291	58(0)
28	13.5	3	100	67.647	89.32	58(0)
29	14	3.3	100	67.647	89.32	58(0)

30	15.5	3	100	67.647	89.32	59(0)
31	17	2.6	100	67.647	89.32	60(0)
32	19.5	2.3	100	67.647	89.32	60(0)
33	21	2.3	100	67.647	89.32	61(0)
34	23	2.3	100	67.647	89.32	60(0)
35	25	1.8	100	61.765	87.379	60(0)

From the table, it can be found that the best classification accuracy of 92.233% was achieved. Gamma 6.5 was chosen as it has least number of support vectors. Figure 5.3 below shows the screen shot of the result from STATISTICA.

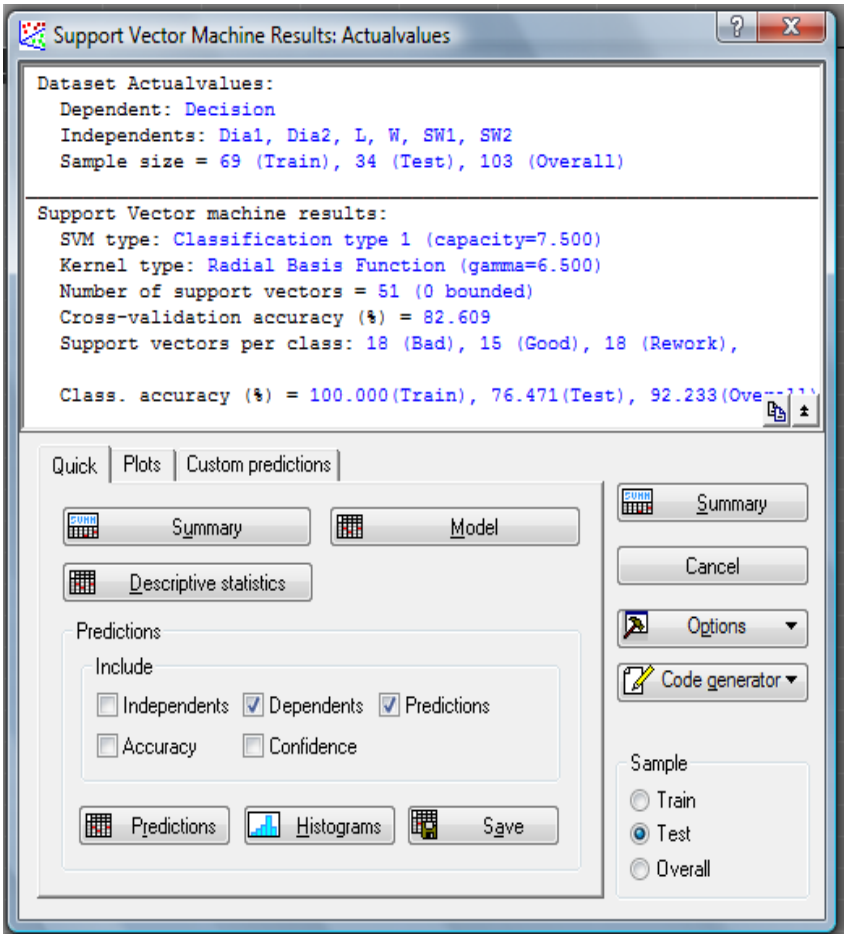


Figure 5.3. Screen shot of the best result with radial basis function kernel

5.1.4 Sigmoid Kernel

With sigmoid kernel, user is required to input the values for gamma, coefficient and the range for training parameter. Several iterations were made by varying the gamma and coefficient values. Results were tabulated as shown below in the table 5.3. The best result was highlighted.

Table 5.3. Results obtained with sigmoid kernel

Trial No	Gamma	Coefficient	Training Parameter	Training Accuracy	Testing Accuracy	Overall Accuracy	No of SVs
1	0.01	0	266.1	82.609	70.588	78.641	50(10)
2	0.05	0	740.5	89.855	73.529	84.466	31(1)
3	**0.1**	**0**	**399**	**89.855**	**73.529**	**84.466**	**30(1)**
4	0.1	10	0	34.783	20.588	30.097	0(0)
5	0.15	0	154.8	89.855	73.529	84.466	32(1)
6	0.25	0	286.4	85.507	67.647	79.612	26(2)
7	0.6	0	5.4	68.116	70.588	68.932	58(20)
8	1	1	6.2	50.725	50	50.585	66(32)
9	1	5	0	34.783	20.588	30.097	0(0)
10	2	50	0	34.783	20.588	30.097	0(0)
11	5	5	0	34.783	20.588	30.097	0(0)
12	50	5	0	34.783	20.588	30.097	0(0)

The best overall classification accuracy of 84.466% was obtained. Figure 5.4 below shows the screen shot of the result from STATISTICA. The accuracy values were found to improve for gamma value up to 0.1 and decrease later.

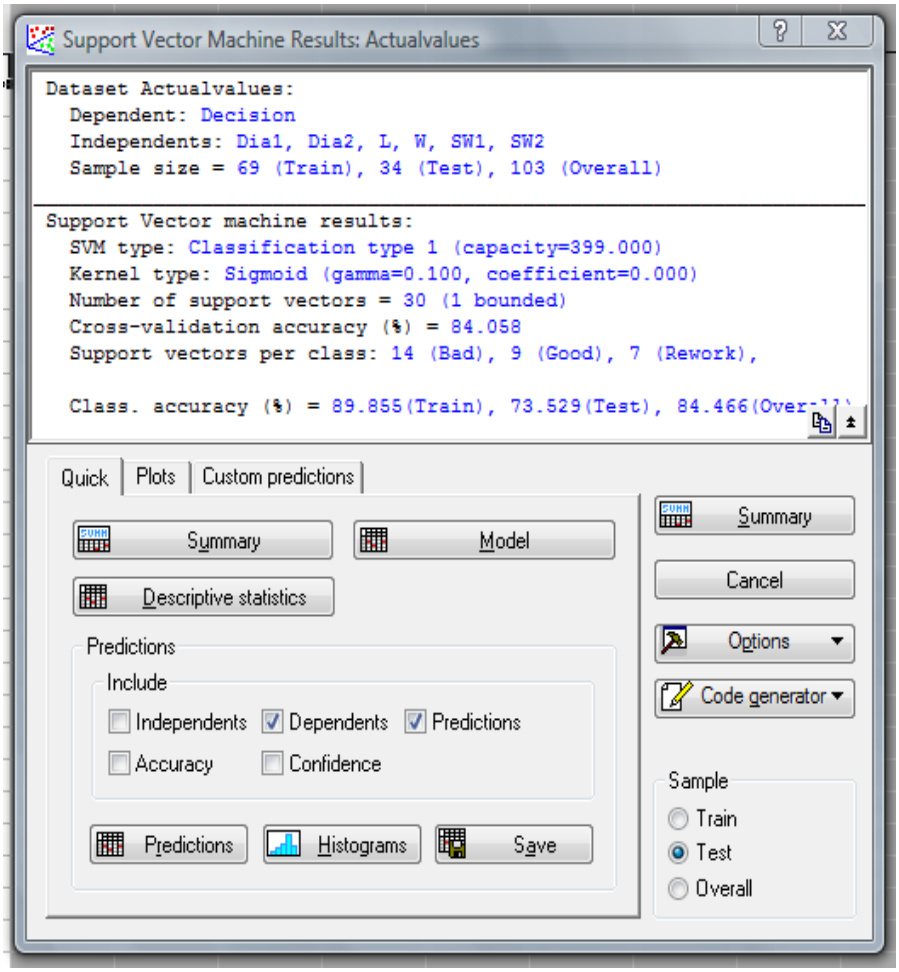


Figure 5.4. Screen shot of the best result with sigmoid kernel

5.1.5 Analysis of SVM results

Of all the four kernels, polynomial kernel was found to yield the best classification accuracy of 92.233% with least number of support vectors. Table 5.4 below shows the confusion matrix for the polynomial kernel with the best set of values.

Table 5.4. Confusion matrix for polynomial kernel (degree 13, gamma 65, coefficient 65)

	Predicted			
Actual	Good	Rework	Bad	Total
Good	43	1	1	45
Rework	4	23	0	27
Bad	2	0	29	31

Out of 45 good parts, 43 were predicted as good, 1 was predicted as rework and 1 as bad. Out of 27 rework parts, 4 were predicted as good. Out of 31 bad parts, 2 were predicted as good. In all, 95 parts were classified correctly and 8 were predicted wrong.

5.2 Support Vector Machines with Modified Data

For the first iteration, index values were calculated for all the 103 parts and were analyzed with all the four kernels to find the one that gives out the best classification accuracy.Index values for 1st iteration were shown in Appendix III. The parts that were classified as not good in the first

iteration are sent to the second iteration. Index values were calculated again with the new limits and analyzed with SVM to differentiate rework and bad parts. The kernel that gave the best result in the first iteration was chosen for the second iteration as the data is similar.

5.2.1 Iteration 1

5.2.1.1 Linear kernel

In the software, linear kernel was selected. Fivefold cross validation with training parameter range of 0 to 500 with an increment of 0.01 was given. Classification Accuracy of 83.95% was achieved. Figure 5.5 below shows the screen shot of the result generated using STATISTICA software.

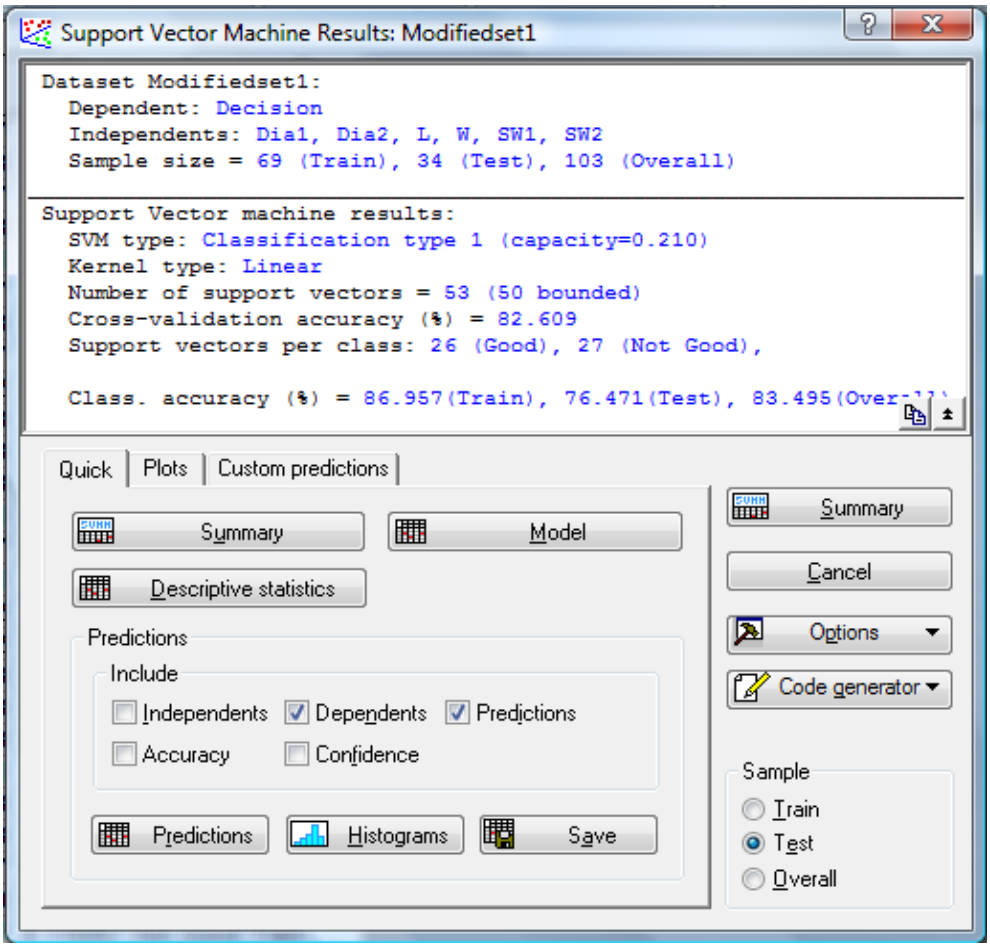


Figure 5.5. Screen shot of the result generated using linear kernel with modified data for 1st iteration

5.2.1.2 Polynomial Kernel

Several iterations were performed with different set of values for degree, gamma and coefficient. A range of 0 to 100 with an increment of 0.1 was given for the training parameter with fivefold cross validation. Results were tabulated as shown in table 5.5 below.

Table 5.5. Results generated using polynomial kernel with modified data for 1st iteration

Trial No	Degree	Gamma	Coefficient	Training Parameter	Training Accuracy	Testing Accuracy	Overall Accuracy	No of support vectors
1	1	0.01	0	20.3	86.957	79.412	84.466	55(51)
2	1	0.1	0	2.1	86.957	76.471	83.495	53(50)
3	1	0.5	0	0.5	86.957	76.471	83.495	50(48)
4	1	0.5	1	0.5	86.957	76.471	83.495	50(48)
5	1	1	0	0.3	86.957	76.471	83.495	47(46)
6	1	1	1	0.3	86.957	76.471	83.495	46(46)
7	1	2	0	0.2	84.058	79.412	82.524	43(41)
8	1	2	1	0.2	84.058	79.412	82.524	43(41)
9	1	5	0	72.1	81.159	79.412	80.583	45(5)
10	1	5	3	72.1	81.159	79.412	80.583	45(5)
11	1	10	0	45.6	73.913	64.706	70.874	38(2)
12	1	10	5	45.6	73.913	64.706	70.874	38(2)

13	2	5	0	7.3	97.101	79.412	91.262	22(1)
14	2	5	3	3.5	98.551	79.412	92.233	24(3)
15	2	5	5	3.5	98.551	79.412	92.233	24(3)
16	3	10	10	0.1	100	85.294	95.146	24(0)
17	3	20	20	0.1	100	85.294	95.146	24(0)
18	**4**	**20**	**20**	**0.1**	**100**	**85.294**	**95.146**	**22(0)**
19	5	25	25	0.1	100	85.294	95.146	24(0)
20	6	40	40	0.1	100	85.294	95.146	24(0)
21	7	40	40	0.1	100	85.294	95.146	24(0)
22	10	50	50	0.1	97.101	79.412	91.262	26(0)
23	12	50	50	0.1	95.652	79.412	90.291	23(0)
24	12	70	70	0.1	98.551	79.412	92.233	31(0)
25	15	90	90	0.1	100	79.412	93.204	26(0)
26	17	90	90	0.1	95.652	79.412	90.291	21(0)
27	17	110	110	0.1	94.203	82.353	90.291	22(0)
28	20	100	100	0.1	88.406	82.353	86.408	20(0)

29	20	120	120	0.1	97.101	82.353	92.233	29(0)
30	20	120	135	0.1	95.652	79.412	90.291	22(0)
31	20	135	135	0.1	95.652	82.353	91.262	24(0)
32	23	135	135	0.1	91.304	79.412	87.379	22(0)
33	23	150	150	0.1	92.754	85.294	90.291	20(0)
34	23	160	160	0.1	91.304	85.294	89.32	21(0)
35	25	160	160	0.1	92.754	82.353	89.32	23(0)
36	25	175	175	0.1	91.304	85.294	89.32	22(0)
37	25	190	190	0.1	94.203	85.294	91.262	22(0)
38	25	200	200	0.1	92.754	82.353	89.32	21(0)

The best classification accuracy of 95.146% was found to be achieved with set of values degree=4, gamma=20, coefficient=20. The result was highlighted in the table above. The figure 5.6 below shows the screen shot of the same.

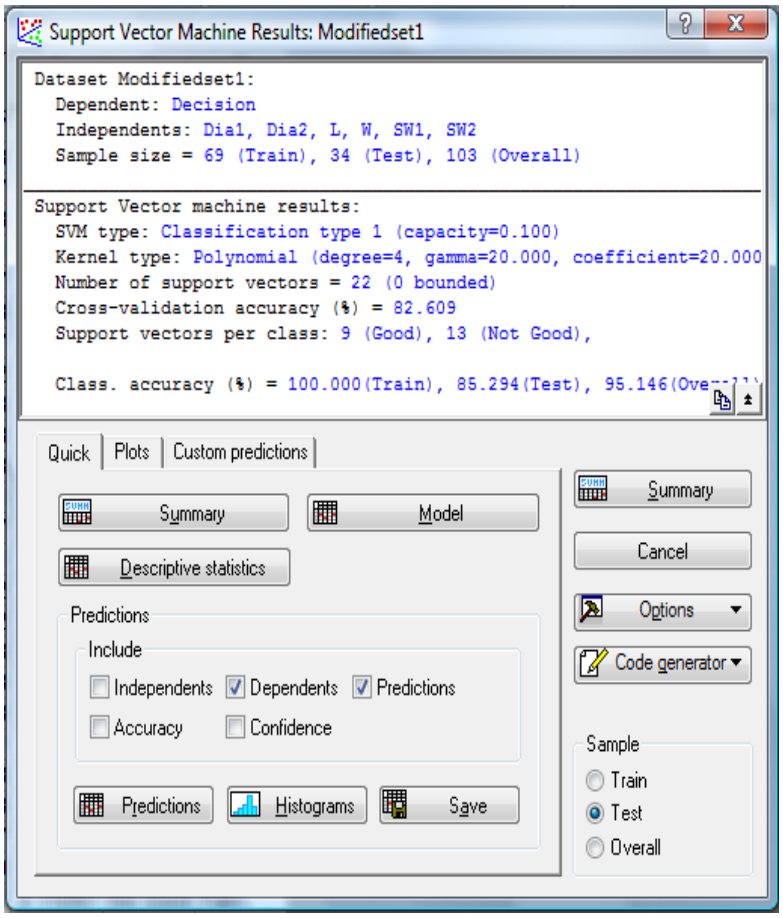


Figure 5.6. Screen shot of the result generated using polynomial kernel with modified data for iteration 1

5.2.1.3 Radial Basis Kernel

Table 5.6 below shows the results generated with various values of gamma using the radial basis function kernel. The best result is highlighted.

Table 5.6. Results generated using radial kernel with modified data for 1st iteration

Trial No	Gamma	Training Parameter	Training Accuracy	Testing Accuracy	Overall Accuracy	No of support vectors
1	0.01	10.2	86.957	76.471	83.495	55(51)
2	0.05	2.1	86.957	76.471	83.495	55(51)
3	0.1	1.3	85.507	79.412	83.495	51(48)
4	0.25	4.3	88.406	79.412	85.437	32(25)
5	0.5	2.2	88.406	79.412	85.437	32(27)
6	0.75	1.9	88.406	82.353	86.408	31(23)
7	1	21.8	97.101	82.353	92.233	26(8)
8	1.25	18.6	98.551	82.353	93.204	26(6)
9	1.5	13.8	98.551	82.353	93.204	26(7)
10	1.75	11.5	98.551	82.353	93.204	26(8)
11	2	10	98.551	82.353	93.204	27(7)
12	2.5	8	98.551	82.353	93.204	29(7)
13	**2.75**	**32.1**	**100**	**85.294**	**95.146**	**26(0)**

14	3	6.7	98.551	85.294	94.175	32(7)
15	3.25	1.8	91.304	85.294	89.32	38(15)
16	3.5	1.3	91.304	82.353	88.35	39(19)
17	4.5	0.8	92.754	79.412	88.35	43(22)
18	5	0.9	91.304	79.412	87.379	42(21)
19	6	0.8	92.754	76.471	87.379	48(20)
20	7.25	3.4	98.551	82.353	93.204	49(3)
21	8.5	2.5	98.551	82.353	93.204	52(5)
22	9.25	2.6	98.551	79.412	92.233	52(3)
23	9.75	2.7	98.551	76.471	91.262	52(3)
24	10	0.8	94.203	76.471	88.35	54(19)
25	10.5	2.3	98.551	79.412	92.233	54(4)
26	12.5	2	98.551	76.471	91.262	54(5)
27	14	1.9	98.551	76.471	91.262	57(5)
28	15	1.9	98.551	76.471	91.262	57(5)
29	16.5	1.9	98.551	76.471	91.262	57(4)

30	18	1.8	98.551	76.471	91.262	58(4)
31	19.5	1.5	98.551	76.471	91.262	58(5)
32	22	1.5	98.551	76.471	91.262	58(5)
33	24	0.9	98.551	76.471	91.262	60(16)
34	28	1.5	98.551	76.471	91.262	60(4)
35	35	1	98.551	76.471	91.262	61(13)

The best accuracy of 95.146% was achieved with gamma=2.75. Figure 5.7 below shows the screen shot of the best result.

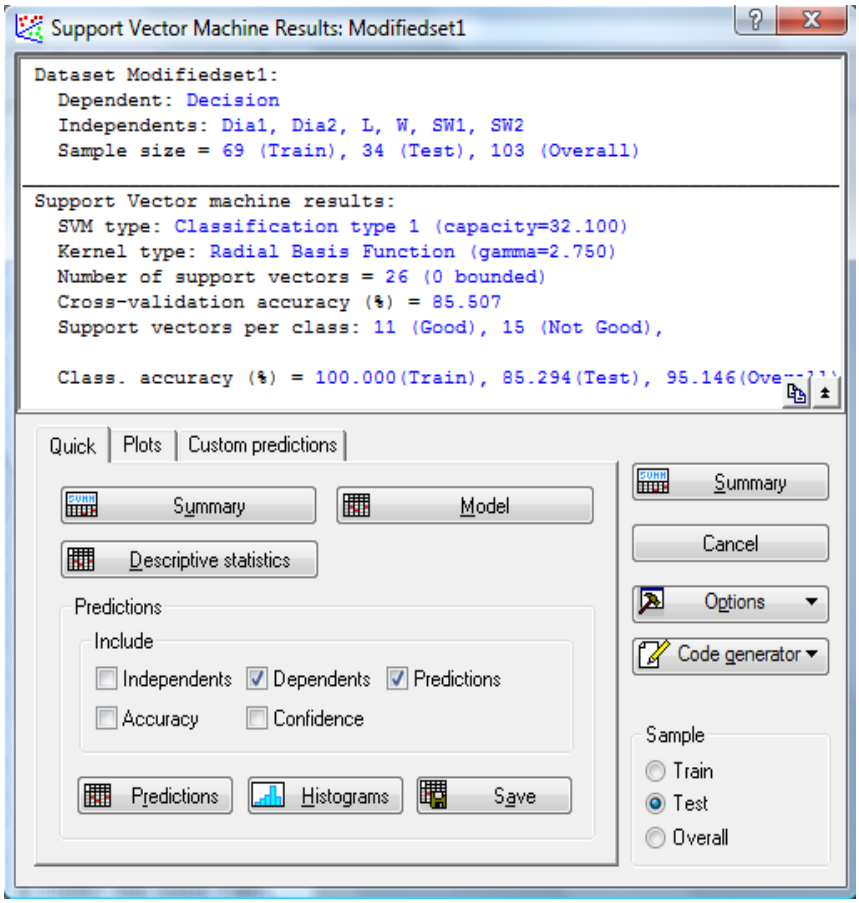


Figure 5.7. Screen shot of the result using radial kernel with modified data for 1st iteration

5.2.1.4 Sigmoid Kernel

Table 5.7 below shows the results generated with various set of values using the sigmoid kernel. The best result is highlighted.

Table 5.7. Results generated using sigmoid kernel with modified data for 1st iteration

Trial No	Gamma	Coefficient	Training Parameter	Training Accuracy	Testing Accuracy	Overall Accuracy	No of SVs
1	**0.001**	**0**	**202.6**	**86.957**	**79.412**	**84.466**	**55(51)**
2	0.01	0	20.3	86.957	79.412	84.466	55(51)
3	0.05	0	4.1	86.957	76.471	83.495	55(51)
4	0.1	0	2.1	86.957	76.471	83.495	54(50)
5	0.5	0	2.9	85.507	79.412	83.495	34(29)
6	1	0	1.8	82.609	79.412	81.553	32(30)
7	1.5	0	0.5	81.159	85.294	82.524	45(43)
8	2	0	0.3	81.159	73.529	78.641	54(54)
9	3	0	0.4	66.667	52.941	62.136	54(54)
10	3	3	44.6	59.42	55.882	58.252	58(58)
11	3	0.3	1.1	62.319	41.176	55.34	50(48)
12	5	0	0.1	57.971	52.941	56.311	58(58)
13	5	0.5	0.1	57.971	52.941	56.311	58(58)

14	5	5	0.1	57.971	52.941	56.311	58(58)
15	5	15	0.1	57.971	52.941	56.311	58(58)
16	6	0	0.1	57.971	52.941	56.311	58(58)
17	7	0	0.3	49.275	41.176	46.602	58(58)
18	9	0	0.1	57.971	52.941	56.311	58(58)
19	9	6	0.1	57.971	52.941	56.311	58(58)
20	10	0	0.1	57.971	52.941	56.311	58(58)
21	10	10	0.1	57.971	52.941	56.311	58(58)
22	15	0	0.1	57.971	52.941	56.311	58(58)
23	15	1.5	0.1	57.971	52.941	56.311	58(58)
24	25	0	0.1	57.971	52.941	56.311	58(58)
25	25	250	0.1	57.971	52.941	56.311	58(58)

The best classification accuracy of 84.466% was achieved at gamma=0.001, 0.01 and 0.05. Figure 5.8 below shows the screen shot of the result.

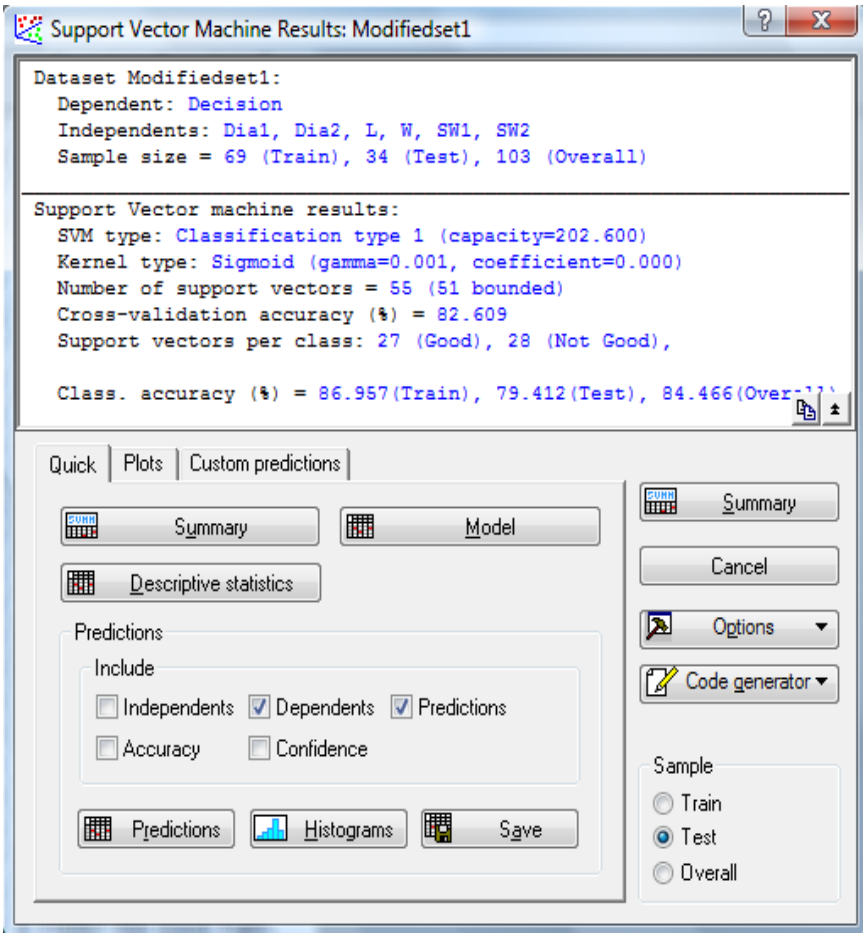
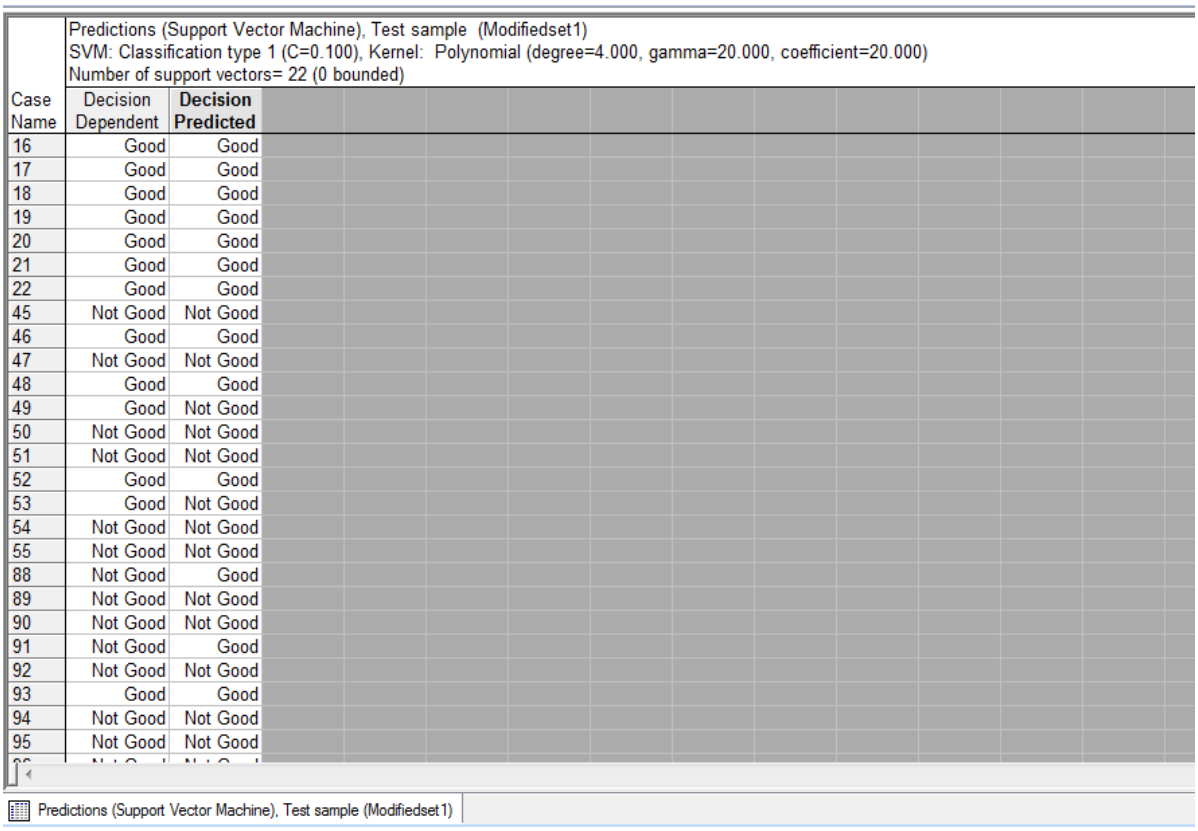


Figure 5.8. Screen shot of the result using sigmoid kernel with modified data for 1st iteration

5.2.1.5 Analysis of the results

It was found that both the polynomial and radial basis kernels resulted in highest classification accuracy. However, polynomial kernel generated the result with least number of support vectors and hence selected. The predictions for the test data are analyzed to find out the wrong predictions. STATISTICA provides option to see the predictions for all the cases tested. Figure 5.9 below shows the screen shot of the same.



Predictions (Support Vector Machine), Test sample (Modifiedset1)
SVM: Classification type 1 (C=0.100), Kernel: Polynomial (degree=4.000, gamma=20.000, coefficient=20.000)
Number of support vectors= 22 (0 bounded)

Case Name	Decision Dependent	Decision Predicted
16	Good	Good
17	Good	Good
18	Good	Good
19	Good	Good
20	Good	Good
21	Good	Good
22	Good	Good
45	Not Good	Not Good
46	Good	Good
47	Not Good	Not Good
48	Good	Good
49	Good	Not Good
50	Not Good	Not Good
51	Not Good	Not Good
52	Good	Good
53	Good	Not Good
54	Not Good	Not Good
55	Not Good	Not Good
88	Not Good	Good
89	Not Good	Not Good
90	Not Good	Not Good
91	Not Good	Good
92	Not Good	Not Good
93	Good	Good
94	Not Good	Not Good
95	Not Good	Not Good

Predictions (Support Vector Machine), Test sample (Modifiedset1)

Figure 5.9. Screen shot of the predictions with modified data for 1st iteration

Table 5.8 below shows list of all the test cases and the predictions made by SVM using polynomial kernel with set of values that generated highest accuracy.

Table 5.8. Predictions with modified data for 1st iteration

Part No	Actual	Predicted
1	Good	Good
2	Good	Good

3	Good	Good
4	Good	Good
5	Good	Good
6	Good	Good
7	Good	Good
8	Not Good	Not Good
9	Good	Good
10	Not Good	Not Good
11	Good	Good
12	Good	Not Good
13	Not Good	Not Good
14	Not Good	Not Good
15	Good	Good
16	Good	Not Good
17	Not Good	Not Good
18	Not Good	Not Good

19	Not Good	Good
20	Not Good	Not Good
21	Not Good	Not Good
22	Not Good	Good
23	Not Good	Not Good
24	Good	Good
25	Not Good	Not Good
26	Not Good	Not Good
27	Not Good	Not Good
28	Not Good	Not Good
29	Not Good	Not Good
30	Not Good	Not Good
31	Good	Good
32	Good	Good
33	Not Good	Not Good
34	Good	Not Good

From the table, it can be found that a total of 5 cases were predicted wrong. Since the training accuracy was 100%, it can be said that a total of 5 wrong predictions out of 103 cases was made. Going through the table, it can be found that cases 88 and 91 which are originally rework are wrongly predicted as good and cases 49, 53 and 103 which are originally good are wrongly predicted as not good. All the cases predicted as not good including those of wrong predictions and training cases are exported to the second iteration. A total of 59 cases were obtained of which 40 were training and 19 test cases. The good ones that are predicted as not good are marked as rework for the analysis in the second iteration.

5.2.2 Iteration 2

5.2.2.1 Polynomial kernel

For all the 59 cases imported from first iteration, actual values obtained from case study were evaluated with new limits to obtain the new set of data. The new set of index values were presented in Appendix IV. The new set of data was then analyzed with SVM using polynomial kernel. Polynomial kernel was used as it resulted in best classification accuracy for the similar type of data. Several trials were made with different set of values starting out with the values from the previous iteration. Table 5.9 below shows the results from various trials. Best result is highlighted.

Table 5.9. Results using polynomial kernel with modified data for iteration 2

Trial No	Degree	Gamma	Coefficient	Training Parameter	Training Accuracy	Testing Accuracy	Overall Accuracy	No of SVs
1	2	10	10	0.1	100	78.947	93.22	12(4)
2	2	15	15	0.1	100	84.211	94.915	7(2)
3	3	15	15	0.1	100	84.211	94.915	6(0)
4	4	15	15	0.1	100	84.211	94.915	9(0)
5	4	15	20	0.1	100	84.211	94.915	9(0)
6	4	20	20	0.1	100	84.211	94.915	9(0)
7	**5**	**30**	**30**	**0.1**	**100**	**94.737**	**98.305**	**9(0)**
8	6	40	40	0.1	100	94.737	98.305	10(0)
9	7	45	45	0.1	100	94.737	98.305	11(0)
10	10	70	70	0.1	100	94.737	98.305	12(0)
11	15	90	90	0.1	100	94.737	98.305	14(0)
12	20	100	100	0.1	100	84.211	94.915	15(0)
13	30	120	120	0.1	100	84.211	94.915	13(0)

Figure 5.10 below shows the screen shot of the best result.

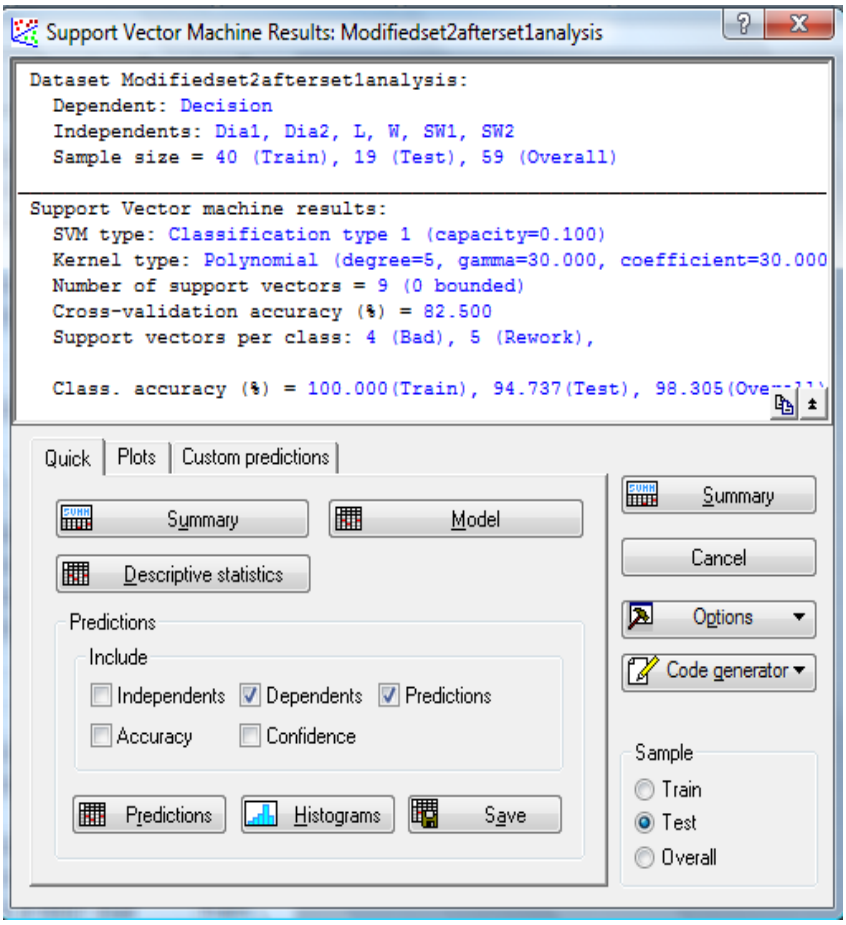


Figure 5.10. Screen shot of the best result with modified data for iteration 2

5.2.2.2 Analysis of the result

The predictions made are analyzed to find out the wrong predictions. Figure 5.11 below shows the screen shot of the predictions.

Predictions (Support Vector Machine), Test sample (Modifiedset2afterset1analysis)
SVM: Classification type 1 (C=0.100), Kernel: Polynomial (degree=5.000, gamma=30.000, coefficient=30.000)
Number of support vectors= 9 (0 bounded)

Case Name	Decision Dependent	Decision Predicted
13	Rework	Rework
14	Rework	Rework
15	Rework	Rework
16	Rework	Rework
17	Rework	Rework
18	Rework	Rework
19	Rework	Rework
20	Rework	Rework
49	Bad	Bad
50	Rework	Rework
51	Bad	Bad
52	Bad	Bad
53	Bad	Rework
54	Bad	Bad
55	Rework	Rework
56	Rework	Rework
57	Bad	Bad
58	Bad	Bad
59	Rework	Rework

Figure 5.11. Screen shot of the predictions with modified data for 2nd iteration

From the figure 5.11, it can be seen that case 53, which is case number 95 in the original data was predicted wrongly as rework.

5.2.3 Analysis of the results with modified data

By analyzing the results from sections 5.2.1 and 5.2.2, it be concluded that the two good ones which are predicted as not good in the first iteration and marked as rework are classified into the rework category in the second iteration. In all, one bad part is wrongly predicted as rework, two rework parts are wrongly predicted as good and three good parts are wrongly predicted as rework. Table 5.10 below shows the confusion matrix developed by combining the results from both iterations.

Table 5.10. Confusion matrix for the results from SVM with modified data

	Predicted			
Actual	Good	Rework	Bad	Total
Good	42	3	0	45
Rework	2	25	0	27
Bad	0	1	30	31

From the matrix, it can be seen that 6 cases were predicted wrong out of 103 cases resulting in the classification accuracy of 94.175%.

5.3 New Methodology

The new methodology is created in excel sheet as shown in figure below. Each column stands for a feature measured on parts. Depending on the type of feature, it is programmed to automatically take the rework limits for the second iteration. The user needs to enter the limits tolerance and rework limits for all the features in the first three rows. The measured values will be entered in the fourth row. All the remaining processing will be done automatically and the final result will be displayed in one of the cells.

Upper tolerance limit	0.2685	0.2665	1.9745	1.245	0.519	0.5195	
Lower tolerance limit	0.2085	0.2065	1.9145	1.185	0.459	0.4595	
Rework limit	0.1785	0.1765	2.0045	1.275	0.459	0.4595	
Measured Dimension	0.21	0.21	1.952	1.185	0.486	0.483	
Iteration 1							
X	0.2385	0.2365	1.9445	1.215	0.489	0.4895	
Xtg	0.03	0.03	0.03	0.03	0.03	0.03	
Index value	0.95	0.88333	0.25	1	0.1	0.21667	
Sine Value	0.0095	0.00883	0.0025	0.01	0.001	0.00217	
0	0	0	0	0	0	0	Good
Iteration 2							
	0.2235	0.2215	1.9595	1.23	0.489	0.4895	
	0.045	0.045	0.045	0.045	0.03	0.03	
Index value	0.3	0.25556	0.16667	1	0.1	0.21667	
Sine Value	0.003	0.00256	0.00167	0.01	0.001	0.00217	
0	0	0	0	0	0	0	Rework
	Final Result		Good				

Figure 5.12. Prediction of case number 103 with the developed methodology

With the developed methodology, all the 103 cases were predicted correctly resulting in 100% classification accuracy. Figure 5.12 shows the prediction of case 103, distinguishing the values with 4^{th} decimal point difference. Table 5.11 below shows the confusion matrix.

Table 5.11. Confusion matrix with the new methodology

	Predicted			
Actual	Good	Rework	Bad	Total
Good	45	0	0	45
Rework	0	27	0	27
Bad	0	0	31	31

Chapter 6

Conclusions

The insights into the results obtained in the previous chapter and conclusions based on them were presented in this chapter. The purpose of this work is to improve the classification process by selecting the best methodology that gives out superior classification accuracy at high speed. SVM is selected as bench mark keeping in view of its higher generalization ability compared to other methods, especially when the data overlap is non-existent. From the literature review, it is learnt that alterations in working with SVM may improve the results and simple statistical procedure designed for a particular case may dominate all other well-known methods. Hence, two new approaches were designed and tested along with traditional SVM. The first approach included altering the data obtained before applying SVM and the second approach is based on a new methodology developed exclusively for the current experiment. The following results were generated with the three approaches.

1. With traditional SVM, the classification accuracy achieved was **92.233%**.
2. With the modified data using SVM, the classification accuracy achieved was **94.175%**
3. With the new methodology, the classification accuracy achieved was **100%**.

From the results, it can be concluded that the proposed method fared better compared to the other two alternatives with 100% classification accuracy and the proposed alteration of the data resulted in improved classification accuracy compared to the one without alteration of the data. Moreover, the proposed sine method works faster without need for any training. **The proposed sine method is tested for the current experiment and is expected to work in any similar**

scenario. In terms of the data type, it works on the data where one class encloses the other, similar to the current case study. With other types of data, the methodology is yet to be tested and may be required to be modified to suit that particular type of data. **However, in the industry applications (with or without E-Quality) where parts are to be classified as good, bad and rework, the proposed methodology can be used right away.** The accuracy in the demonstrated experiment is limited by the accuracy with which the camera measures the dimensions and the methodology can classify dimensions to the accuracy level of 8 decimal points. The methodology can accommodate as many features as the user needs and can classify the parts into good, rework or bad classes with 100% accuracy.

Chapter 7

Future Research

1. In the current work, the proposed methodology was applied to one type of part, developed my mimicking an index part, commonly used in RP industry to benchmark the RP machines. It needs to be tested on wide variety of parts to completely validate the methodology through various types of applications.
2. The proposed procedure of applying SVM to index values (modified data) needs further testing to completely validate the results, as the results may vary with different types and amounts of data.

References

1. Richard Chiou., PrathabanMookiah., and Yongjin Kwon., 2008, "Manufacturing e-quality through integrated web-enabled computer vision and robotics", International Journal of Advanced Manufacturing Technology, 43, 720-730.
2. Suykens, J.A.K., Vandewalle, J.,1999, "Least squares support vector machine classifiers", Neural Processing Letters, 9, 293-300.
3. Shigeo Abe., 2005, "Support Vector Machines for pattern classification", ISBN: 1-85233-929-2.
4. Ming-HuwiHorng., 2009,"Multi-class support vector machine for classification of the ultrasonic images of supraspinatus", Expert Systems with Applications, 36 (4), 8124-8133.
5. Jin-Hyuk Hong., Sung-Bae Cho., 2008,"A probabilistic multiclass strategy of one-vs.-rest support vector machines for cancer classification",Neurocomputing, 71(16-18), 3275-3281.
6. SukantaMondal., RajasekaranBhavna., Rajasekaran Mohan Babu., SuryanarayanaraoRamakumar., 2006,"Pseudo amino acid composition and multi-class support vector machines approach for conotoxins superfamily classification", Journal of Theoretical Biology, 243 (2), 252-260.
7. Kai-QuanShen., Xiao-Ping Li., Chong- Jin Ong., Shi-Yun Shaho., EinarP.V.Wilder – Smith., 2008,"EEG-based mental fatigue measurement using multi-class support vector machines", Clinical Neurophysiology, 119 (7), 1524-1533.

8. Der-Chiang Li., Chiao-Wen Liu., 2009,“A class possibility based kernel to increase classification accuracy for small data sets using support vector machines”, Expert systems with applications, doi: 10.1016/j.eswa.2009.09.019.
9. Cheng-Jin Du., Da-Wen Sun., 2008,“Multi –classification of pizza using computer vision and support vector machine”, Journal of Food Engineering, 86(2), 234-242.
10. C.Z. Cai., W.L. Wang., L.Z.sun., Y.Z.Chen., 2003,“Protein function classification Via support vector machine approach”, Mathematical Biosciences, 185(2), 111-122.
11. V.Sugumaran., G.R. Sabareesh., K.I. Ramachandran., 2008,“Fault diagonistics of roller bearing using kernel based kernel based neighbouringhood score multi-class support vector machine”, Expert systems with Apllications, 34(2), 3090-3098.
12. Jin-Hyuk Hong., Jun-Ki Min., Ung-Keun Cho., Sung-Bae Cho., 2008,“Fingerprint classification using one-versus-all support vector machines dynamically ordered with naïve bayes classifiers”, Pattern Recognition, 41(2), 662-671.
13. Xue-Wen Chen., XiangyanZeng., Deborah Van Alphen., 2006,“Multi-class feature selection for texture classification”, Pattern Recognition Letters, 27(14), 1685-1691.
14. Li-Chang Chao., Lee-Ing Tong., 2009,“Water defect pattern recognition by multi-class support vector machines by using a novel defect cluster index”, Sensors and Actuators, 122(1), 227-235.
15. Jayadeva.,ReshmaKhemchandani., Suresh Chandra., 2005,“Fuzzy linear proximal support vector machines for multi-category data classification”,Neurocomputing, 67, 426-435.

16. EmreComack., AhmetArslan., 2008,“A new training method for support vector machines: Clustering k-NN support vector machines”, Expert Systems with Applications, 35(3), 564-568.
17. Ben Van Calster., George Condous., Emma Kirk., Tom Bourne., Timmerman., Sabine Van Huffel., 2009,“An application of methods for the probabilistic three-class classification of pregnancies of unknown location”, Artificial Intelligence in Medicine, 46(2), 139-154.
18. David Meyer., Friedrich Leisch., Kurt Hornik., 2003, “The support vector machine under test”,Neurocomputing, 55(1-2), 169-186.
19. Vapnik V., and Lerner A., 1963,“Pattern recognition using generalized portrait method. Automation and Remote Control”, 24, 774–780
20. Vapnik V., and Chervonenkis A., 1974,“Theory of Pattern Recognition [in Russian]”, Nauka, Moscow (German Translation: Wapnik W. &Tscherwonenkis A., Theorie der Zeichenerkennung, Akademie-Verlag, Berlin, 1979).
21. Sch¨olkopf B., Burges C., and Vapnik V., 1995,“Extracting support data for a given task. In: Fayyad U.M. and Uthurusamy R. (Eds.)”, Proceedings, First International Conference on Knowledge Discovery & Data Mining, Menlo Park, AAAI Press.
22. Sch¨olkopf B., Burges C., andVapnikV., 1996,“Incorporating invariances in support vector learning machines”, In: von der Malsburg C., von Seelen W., Vorbr¨uggen J. C., and Sendhoff B. (Eds.), Artificial Neural Networks ICANN‘96, 47–52, Berlin, Springer Lecture Notes in Computer Science, Vol. 1112.

23. Vapnik V., Golowich S., and Smola A., 1997,“Support vector method for function approximation, regression estimation, and signal processing”, In: Mozer M.C., Jordan M.I., and Petsche T. (Eds.) Advances in Neural Information Processing Systems 9, MA, MIT Press, Cambridge, 281–287.
24. Vapnik V., and Chervonenkis A., 1964, “A note on one class of perceptrons”, Automation and Remote Control, 25.
25. Hsu C.-W., and Lin C.-J., 2002, “A comparison of methods for multi-class support vector machines”, IEEE Transactions on Neural networks, 13(2), 415-425.

Appendix

Appendix I: Robot Code

```
ACCEL 80 /*SETTING ACCELERATION TO 80% OF TOTAL AVAILABLE*/

DECEL 80

SPEED 60

MOVE P, P0, Z=0 /*MOVING TO P0, INITIAL START UP POINT*/

*MAIN:

DO2(5)=0 /*SETTING THE TRIGGER OF CAMERA TO OFF*/

DELAY 5000

DO2(6)=1 /*SETTING THE CONVEYER TO ON*/

DELAY 1600

WAIT DI(30)=0/*WAITING FOR THE SENSOR TO DETECT THE PART*/

DO2(6)=0 /* SETTING THE CONVEYER TO STOP */

DELAY 50

DO2(5)=1 /*TRIGGERING THE CAMERA*/

DELAY 10000

IF DI(37)=0 THEN /*CHECKING IF THE PART IS BAD (DI(37) IS THE OUTPUT FROM
CAMERA)*/

MOVE P, P101, Z=0 /*MOVE TO THE POINT OVER THE BAD PART ON CONVEYER*/

DO2(0)=1 /*TRIGGERING THE SUCTION TIP 1 TO BE ON*/

DO2(1)=1 /*TRIGGERING SUCTION TIPS 2 & 3 TO BE ON*/

MOVE P, P0, Z=0 /*MOVE TO DROP OFF POINT*/

DO2(0)=0 /*TRIGGERING SUCTION PORT 1 TO BE OFF*/
```

```
DO2(1)=0 /*TRIGGERING SUCTION PORTS TO BE OFF SO THAT PART IS DROPPED*/
ELSE /* CHECKING IF THE PART IS GOOD*/
GOTO *MAIN
END IF
GOTO *MAIN
```

Appendix II: Experimental Data

Dia 1	Dia 2	L	W	SW1	SW2	Decision	Sample
0.24	0.241	1.947	1.217	0.494	0.489	Good	Train
0.24	0.241	1.946	1.217	0.493	0.49	Good	Train
0.242	0.239	1.946	1.22	0.487	0.495	Good	Train
0.239	0.241	1.95	1.217	0.493	0.487	Good	Train
0.239	0.242	1.943	1.218	0.492	0.487	Good	Train
0.235	0.238	1.942	1.214	0.491	0.485	Good	Train
0.241	0.24	1.948	1.214	0.485	0.49	Good	Train
0.241	0.242	1.947	1.214	0.487	0.495	Good	Train
0.239	0.239	1.955	1.214	0.483	0.489	Good	Train
0.24	0.235	1.938	1.219	0.493	0.488	Good	Train
0.236	0.241	1.953	1.212	0.488	0.484	Good	Train
0.241	0.232	1.951	1.219	0.49	0.486	Good	Train
0.236	0.233	1.946	1.214	0.492	0.484	Good	Train
0.239	0.239	1.956	1.219	0.487	0.492	Good	Train
0.242	0.24	1.943	1.219	0.486	0.497	Good	Test
0.24	0.242	1.946	1.219	0.495	0.488	Good	Test
0.239	0.242	1.95	1.218	0.495	0.486	Good	Test
0.24	0.241	1.942	1.219	0.485	0.499	Good	Test

0.239	0.238	1.947	1.215	0.485	0.489	Good	Test
0.235	0.231	1.946	1.21	0.493	0.48	Good	Test
0.235	0.233	1.954	1.213	0.486	0.488	Good	Test
0.235	0.24	1.933	1.214	0.488	0.488	Good	Test
0.205	0.211	1.944	1.239	0.48	0.488	Rework	Train
0.212	0.213	1.977	1.219	0.484	0.487	Rework	Train
0.241	0.241	1.965	1.249	0.489	0.489	Rework	Train
0.241	0.241	1.976	1.247	0.486	0.486	Rework	Train
0.207	0.241	1.94	1.247	0.484	0.488	Rework	Train
0.209	0.211	1.969	1.219	0.479	0.485	Good	Train
0.242	0.241	1.971	1.243	0.485	0.485	Good	Train
0.21	0.24	1.95	1.212	0.481	0.484	Good	Train
0.21	0.24	1.966	1.249	0.486	0.487	Rework	Train
0.239	0.209	1.974	1.248	0.492	0.484	Rework	Train
0.212	0.241	1.944	1.249	0.483	0.491	Rework	Train
0.243	0.208	1.972	1.214	0.488	0.484	Good	Train
0.211	0.24	1.975	1.217	0.482	0.485	Rework	Train
0.236	0.209	1.942	1.212	0.484	0.484	Good	Train
0.214	0.213	1.971	1.214	0.482	0.483	Good	Train
0.242	0.243	1.975	1.216	0.483	0.486	Rework	Train
0.213	0.239	1.972	1.245	0.482	0.487	Good	Train
0.239	0.242	1.942	1.241	0.479	0.483	Good	Train

0.208	0.208	1.971	1.24	0.481	0.488	Rework	Train
0.211	0.241	1.98	1.217	0.486	0.485	Rework	Train
0.239	0.239	1.947	1.241	0.481	0.48	Good	Train
0.239	0.239	1.949	1.24	0.479	0.475	Good	Train
0.241	0.213	1.947	1.248	0.487	0.485	Rework	Test
0.211	0.213	1.948	1.215	0.481	0.485	Good	Test
0.21	0.206	1.946	1.239	0.483	0.483	Rework	Test
0.209	0.211	1.945	1.212	0.478	0.481	Good	Test
0.21	0.211	1.951	1.239	0.48	0.481	Good	Test
0.208	0.212	1.944	1.217	0.489	0.486	Rework	Test
0.209	0.211	1.98	1.24	0.479	0.477	Rework	Test
0.209	0.24	1.947	1.214	0.481	0.483	Good	Test
0.24	0.243	1.972	1.214	0.483	0.488	Good	Test
0.243	0.243	1.981	1.213	0.481	0.485	Rework	Test
0.207	0.209	1.975	1.24	0.481	0.482	Rework	Test
0.238	0.268	1.947	1.245	0.485	0.487	Bad	Train
0.268	0.272	1.952	1.185	0.479	0.486	Bad	Train
0.179	0.208	1.971	1.21	0.486	0.482	Rework	Train
0.293	0.269	1.975	1.211	0.483	0.481	Bad	Train
0.272	0.27	1.945	1.214	0.512	0.516	Bad	Train
0.272	0.27	1.944	1.219	0.506	0.514	Bad	Train
0.267	0.269	1.996	1.247	0.474	0.484	Bad	Train

0.269	0.294	1.978	1.212	0.478	0.476	Bad	Train
0.268	0.295	1.972	1.213	0.489	0.477	Bad	Train
0.211	0.184	1.977	1.211	0.483	0.481	Rework	Train
0.21	0.179	1.975	1.213	0.476	0.484	Rework	Train
0.24	0.242	1.943	1.183	0.481	0.489	Bad	Train
0.268	0.266	1.936	1.22	0.515	0.517	Good	Train
0.268	0.273	1.947	1.184	0.488	0.48	Bad	Train
0.238	0.238	1.915	1.213	0.482	0.514	Good	Train
0.272	0.269	1.972	1.211	0.491	0.482	Bad	Train
0.272	0.263	1.981	1.21	0.472	0.478	Bad	Train
0.273	0.269	1.944	1.247	0.488	0.482	Bad	Train
0.27	0.272	1.949	1.247	0.487	0.484	Bad	Train
0.271	0.269	1.938	1.22	0.485	0.489	Bad	Train
0.271	0.269	1.996	1.246	0.481	0.486	Bad	Train
0.24	0.243	1.969	1.218	0.486	0.515	Good	Train
0.237	0.241	1.977	1.218	0.487	0.517	Rework	Train
0.24	0.244	1.919	1.184	0.484	0.485	Bad	Train
0.267	0.269	1.941	1.184	0.481	0.485	Bad	Train
0.271	0.272	1.941	1.247	0.488	0.486	Bad	Train
0.209	0.207	1.944	1.216	0.484	0.512	Good	Train
0.242	0.271	1.952	1.243	0.487	0.482	Bad	Train
0.27	0.239	1.942	1.244	0.492	0.478	Bad	Train

0.269	0.271	1.999	1.247	0.48	0.49	Bad	Train
0.238	0.24	1.946	1.181	0.483	0.482	Bad	Train
0.273	0.269	1.941	1.215	0.483	0.484	Bad	Train
0.207	0.211	1.94	1.217	0.485	0.514	Rework	Test
0.268	0.27	1.977	1.21	0.483	0.48	Bad	Test
0.207	0.211	1.976	1.27	0.483	0.485	Rework	Test
0.208	0.21	1.947	1.218	0.513	0.485	Rework	Test
0.241	0.242	1.951	1.181	0.477	0.48	Bad	Test
0.243	0.24	1.973	1.218	0.486	0.518	Good	Test
0.27	0.27	1.946	1.217	0.484	0.489	Bad	Test
0.24	0.24	1.912	1.211	0.515	0.487	Bad	Test
0.241	0.241	1.912	1.182	0.491	0.483	Bad	Test
0.21	0.212	1.979	1.273	0.487	0.486	Rework	Test
0.209	0.208	1.972	1.27	0.487	0.485	Rework	Test
0.208	0.21	1.945	1.182	0.488	0.483	Bad	Test
0.24	0.24	1.923	1.185	0.486	0.483	Good	Test
0.237	0.241	1.921	1.212	0.484	0.511	Good	Test
0.211	0.21	1.944	1.183	0.487	0.483	Bad	Test
0.21	0.21	1.952	1.185	0.486	0.483	Good	Test

Appendix III: Modified data set for Iteration 1

Dia 1	Dia 2	L	W	SW1	SW2	Decision	Sample
0.05	0.15	0.08333	0.06667	0.16667	0.01667	Good	Train
0.05	0.15	0.05	0.06667	0.13333	0.01667	Good	Train
0.11667	0.08333	0.05	0.16667	0.06667	0.18333	Good	Train
0.01667	0.15	0.18333	0.06667	0.13333	0.08333	Good	Train
0.01667	0.18333	0.05	0.1	0.1	0.08333	Good	Train
0.11667	0.05	0.08333	0.03333	0.06667	0.15	Good	Train
0.08333	0.11667	0.11667	0.03333	0.13333	0.01667	Good	Train
0.08333	0.18333	0.08333	0.03333	0.06667	0.18333	Good	Train
0.01667	0.08333	0.35	0.03333	0.2	0.01667	Good	Train
0.05	0.05	0.21667	0.13333	0.13333	0.05	Good	Train
0.08333	0.15	0.28333	0.1	0.03333	0.18333	Good	Train
0.08333	0.15	0.21667	0.13333	0.03333	0.11667	Good	Train
0.08333	0.11667	0.05	0.03333	0.1	0.18333	Good	Train
0.01667	0.08333	0.38333	0.13333	0.06667	0.08333	Good	Train
0.11667	0.11667	0.05	0.13333	0.1	0.25	Good	Test
0.05	0.18333	0.05	0.13333	0.2	0.05	Good	Test
0.01667	0.18333	0.18333	0.1	0.2	0.11667	Good	Test
0.05	0.15	0.08333	0.13333	0.13333	0.31667	Good	Test
0.01667	0.05	0.08333	0	0.13333	0.01667	Good	Test
0.11667	0.18333	0.05	0.16667	0.13333	0.31667	Good	Test

0.11667	0.11667	0.31667	0.06667	0.1	0.05	Good	Test
0.11667	0.11667	0.38333	0.03333	0.03333	0.05	Good	Test
1.11667	0.85	0.01667	0.8	0.3	0.05	Not Good	Train
0.88333	0.78333	1.08333	0.13333	0.16667	0.08333	Not Good	Train
0.08333	0.15	0.68333	1.13333	0	0.01667	Not Good	Train
0.08333	0.15	1.05	1.06667	0.1	0.11667	Not Good	Train
1.05	0.15	0.15	1.06667	0.16667	0.05	Not Good	Train
0.98333	0.85	0.81667	0.13333	0.33333	0.15	Good	Train
0.11667	0.15	0.88333	0.93333	0.13333	0.15	Good	Train
0.95	0.11667	0.18333	0.1	0.26667	0.18333	Good	Train
0.95	0.11667	0.71667	1.13333	0.1	0.08333	Not Good	Train
0.01667	0.91667	0.98333	1.1	0.1	0.18333	Not Good	Train
0.88333	0.15	0.01667	1.13333	0.2	0.05	Not Good	Train
0.15	0.95	0.91667	0.03333	0.03333	0.18333	Good	Train
0.91667	0.11667	1.01667	0.06667	0.23333	0.15	Not Good	Train
0.08333	0.91667	0.08333	0.1	0.16667	0.18333	Good	Train
0.81667	0.78333	0.88333	0.03333	0.23333	0.21667	Good	Train
0.11667	0.21667	1.01667	0.03333	0.2	0.11667	Not Good	Train
0.85	0.08333	0.91667	1	0.23333	0.08333	Good	Train
0.01667	0.18333	0.08333	0.86667	0.33333	0.21667	Good	Train
1.01667	0.95	0.88333	0.83333	0.26667	0.05	Not Good	Train
0.91667	0.15	1.18333	0.06667	0.1	0.15	Not Good	Train

0.01667	0.08333	0.08333	0.86667	0.26667	0.31667	Good	Train
0.01667	0.08333	0.15	0.83333	0.33333	0.48333	Good	Train
0.08333	0.78333	0.08333	1.1	0.06667	0.15	Not Good	Test
0.91667	0.78333	0.11667	0	0.26667	0.15	Good	Test
0.95	1.01667	0.05	0.8	0.2	0.21667	Not Good	Test
0.98333	0.85	0.01667	0.1	0.36667	0.28333	Good	Test
0.95	0.85	0.21667	0.8	0.3	0.28333	Good	Test
1.01667	0.81667	0.01667	0.06667	0	0.11667	Not Good	Test
0.98333	0.85	1.18333	0.83333	0.33333	0.41667	Not Good	Test
0.98333	0.11667	0.08333	0.03333	0.26667	0.21667	Good	Test
0.05	0.21667	0.91667	0.03333	0.2	0.05	Good	Test
0.15	0.21667	1.21667	0.06667	0.26667	0.15	Not Good	Test
1.05	0.91667	1.01667	0.83333	0.26667	0.25	Not Good	Test
0.01667	1.05	0.08333	1	0.13333	0.08333	Not Good	Train
0.98333	1.18333	0.25	1	0.33333	0.11667	Not Good	Train
1.98333	0.95	0.88333	0.16667	0.1	0.25	Not Good	Train
1.81667	1.08333	1.01667	0.13333	0.2	0.28333	Not Good	Train
1.11667	1.11667	0.01667	0.03333	0.76667	0.88333	Not Good	Train
1.11667	1.11667	0.01667	0.13333	0.56667	0.81667	Not Good	Train
0.95	1.08333	1.71667	1.06667	0.5	0.18333	Not Good	Train
1.01667	1.91667	1.11667	0.1	0.36667	0.45	Not Good	Train
0.98333	1.95	0.91667	0.06667	0	0.41667	Not Good	Train

0.91667	1.75	1.08333	0.13333	0.2	0.28333	Not Good	Train
0.95	1.91667	1.01667	0.06667	0.43333	0.18333	Not Good	Train
0.05	0.18333	0.05	1.06667	0.26667	0.01667	Not Good	Train
0.98333	0.98333	0.28333	0.16667	0.86667	0.91667	Good	Train
0.98333	1.21667	0.08333	1.03333	0.03333	0.31667	Not Good	Train
0.01667	0.05	0.98333	0.06667	0.23333	0.81667	Good	Train
1.11667	1.08333	0.91667	0.13333	0.06667	0.25	Not Good	Train
1.11667	0.88333	1.21667	0.16667	0.56667	0.38333	Not Good	Train
1.15	1.08333	0.01667	1.06667	0.03333	0.25	Not Good	Train
1.05	1.18333	0.15	1.06667	0.06667	0.18333	Not Good	Train
1.08333	1.08333	0.21667	0.16667	0.13333	0.01667	Not Good	Train
1.08333	1.08333	1.71667	1.03333	0.26667	0.11667	Not Good	Train
0.05	0.21667	0.81667	0.1	0.1	0.85	Good	Train
0.05	0.15	1.08333	0.1	0.06667	0.91667	Not Good	Train
0.05	0.25	0.85	1.03333	0.16667	0.15	Not Good	Train
0.95	1.08333	0.11667	1.03333	0.26667	0.15	Not Good	Train
1.08333	1.18333	0.11667	1.06667	0.03333	0.11667	Not Good	Train
0.98333	0.98333	0.01667	0.03333	0.16667	0.75	Good	Train
0.11667	1.15	0.25	0.93333	0.06667	0.25	Not Good	Train
1.05	0.08333	0.08333	0.96667	0.1	0.38333	Not Good	Train
1.01667	1.15	1.81667	1.06667	0.3	0.01667	Not Good	Train
0.01667	0.11667	0.05	1.13333	0.2	0.25	Not Good	Train

1.15	1.08333	0.11667	0	0.2	0.18333	Not Good	Train
1.05	0.85	0.15	0.06667	0.13333	0.81667	Not Good	Test
0.98333	1.11667	1.08333	0.16667	0.2	0.31667	Not Good	Test
1.05	0.85	1.05	1.83333	0.2	0.15	Not Good	Test
1.01667	0.88333	0.08333	0.1	0.8	0.15	Not Good	Test
0.08333	0.18333	0.21667	1.13333	0.4	0.31667	Not Good	Test
0.15	0.11667	0.95	0.1	0.1	0.95	Good	Test
1.05	1.11667	0.05	0.06667	0.16667	0.01667	Not Good	Test
0.05	0.11667	1.08333	0.13333	0.86667	0.08333	Not Good	Test
0.08333	0.15	1.08333	1.1	0.06667	0.21667	Not Good	Test
0.95	0.81667	1.15	1.93333	0.06667	0.11667	Not Good	Test
0.98333	0.95	0.91667	1.83333	0.06667	0.15	Not Good	Test
1.01667	0.88333	0.01667	1.1	0.03333	0.21667	Not Good	Test
0.05	0.11667	0.71667	1	0.1	0.21667	Good	Test
0.05	0.15	0.78333	0.1	0.16667	0.71667	Good	Test
0.91667	0.88333	0.01667	1.06667	0.06667	0.21667	Not Good	Test
0.95	0.88333	0.25	1	0.1	0.21667	Good	Test

Appendix IV: Modified data set for Iteration 2

Dia 1	Dia 2	L	W	SW1	SW2	Decision	Sample
0.41111	0.23333	0.34444	0.2	0.3	0.05	Rework	Train
0.25556	0.18889	0.38889	0.24444	0.16667	0.08333	Rework	Train
0.38889	0.43333	0.12222	0.42222	0	0.01667	Rework	Train
0.38889	0.43333	0.36667	0.37778	0.1	0.11667	Rework	Train
0.36667	0.43333	0.43333	0.37778	0.16667	0.05	Rework	Train
0.3	0.41111	0.14444	0.42222	0.1	0.08333	Rework	Train
0.34444	0.27778	0.32222	0.4	0.1	0.18333	Rework	Train
0.25556	0.43333	0.34444	0.42222	0.2	0.05	Rework	Train
0.27778	0.41111	0.34444	0.28889	0.23333	0.15	Rework	Train
0.41111	0.47778	0.34444	0.31111	0.2	0.11667	Rework	Train
0.34444	0.3	0.25556	0.22222	0.26667	0.05	Rework	Train
0.27778	0.43333	0.45556	0.28889	0.1	0.15	Rework	Train
0.38889	0.18889	0.27778	0.4	0.06667	0.15	Rework	Test
0.3	0.34444	0.3	0.2	0.2	0.21667	Rework	Test
0.3	0.23333	0.18889	0.2	0.3	0.28333	Rework	Test
0.34444	0.21111	0.34444	0.28889	0	0.11667	Rework	Test
0.32222	0.23333	0.45556	0.22222	0.33333	0.41667	Rework	Test
0.36667	0.47778	0.27778	0.35556	0.2	0.05	Rework	Test
0.43333	0.47778	0.47778	0.37778	0.26667	0.15	Rework	Test
0.36667	0.27778	0.34444	0.22222	0.26667	0.25	Rework	Test

0.32222	1.03333	0.27778	0.33333	0.13333	0.08333	Bad	Train
0.98889	1.12222	0.16667	1	0.33333	0.11667	Bad	Train
0.98889	0.3	0.25556	0.44444	0.1	0.25	Rework	Train
1.54444	1.05556	0.34444	0.42222	0.2	0.28333	Bad	Train
1.07778	1.07778	0.32222	0.35556	0.76667	0.88333	Bad	Train
1.07778	1.07778	0.34444	0.24444	0.56667	0.81667	Bad	Train
0.96667	1.05556	0.81111	0.37778	0.5	0.18333	Bad	Train
1.01111	1.61111	0.41111	0.4	0.36667	0.45	Bad	Train
0.98889	1.63333	0.27778	0.37778	0	0.41667	Bad	Train
0.27778	0.83333	0.38889	0.42222	0.2	0.28333	Rework	Train
0.3	0.94444	0.34444	0.37778	0.43333	0.18333	Rework	Train
0.36667	0.45556	0.36667	1.04444	0.26667	0.01667	Bad	Train
0.98889	1.14444	0.27778	1.02222	0.03333	0.31667	Bad	Train
1.07778	1.05556	0.27778	0.42222	0.06667	0.25	Bad	Train
1.07778	0.92222	0.47778	0.44444	0.56667	0.38333	Bad	Train
1.1	1.05556	0.34444	0.37778	0.03333	0.25	Bad	Train
1.03333	1.12222	0.23333	0.37778	0.06667	0.18333	Bad	Train
1.05556	1.05556	0.47778	0.22222	0.13333	0.01667	Bad	Train
1.05556	1.05556	0.81111	0.35556	0.26667	0.11667	Bad	Train
0.3	0.43333	0.38889	0.26667	0.06667	0.91667	Rework	Train
0.36667	0.5	0.9	1.02222	0.16667	0.15	Bad	Train
0.96667	1.05556	0.41111	1.02222	0.26667	0.15	Bad	Train

1.05556	1.12222	0.41111	0.37778	0.03333	0.11667	Bad	Train
0.41111	1.1	0.16667	0.28889	0.06667	0.25	Bad	Train
1.03333	0.38889	0.38889	0.31111	0.1	0.38333	Bad	Train
1.01111	1.1	0.87778	0.37778	0.3	0.01667	Bad	Train
0.32222	0.41111	0.3	1.08889	0.2	0.25	Bad	Train
1.1	1.05556	0.41111	0.33333	0.2	0.18333	Bad	Train
0.98889	1.07778	0.38889	0.44444	0.2	0.31667	Bad	Test
0.36667	0.23333	0.36667	0.88889	0.2	0.15	Rework	Test
0.38889	0.45556	0.18889	1.08889	0.4	0.31667	Bad	Test
1.03333	1.07778	0.3	0.28889	0.16667	0.01667	Bad	Test
0.36667	0.41111	1.05556	0.42222	0.86667	0.08333	Bad	Test
0.38889	0.43333	1.05556	1.06667	0.06667	0.21667	Bad	Test
0.3	0.21111	0.43333	0.95556	0.06667	0.11667	Rework	Test
0.32222	0.3	0.27778	0.88889	0.06667	0.15	Rework	Test
0.34444	0.25556	0.32222	1.06667	0.03333	0.21667	Bad	Test
0.27778	0.25556	0.34444	1.04444	0.06667	0.21667	Bad	Test
0.3	0.25556	0.16667	1	0.1	0.21667	Rework	Test

Curriculum Vita

Prashanth Devaram was born in Khammam, India on August 6, 1986 to Mr. Purushotham Reddy Devaram and Mrs. Karuna Devaram. He graduated from Jawaharlal Nehru Technological University, Anantapur with Bachelor of Technology degree in Mechanical Engineering in May, 2007. He started to pursue his Master of Science degree in Industrial Engineering at University of Texas at El Paso from fall 2007. At UTEP, he worked as research assistant at Industrial Systems Engineering Laboratory and was a member of Institute of Industrial Engineers.

Permanent Address: 712 Prospect St Apt #11, El Paso, Texas, 79902.

This thesis was typed by Prashanth Devaram.