ALGORITHMIC MANAGEMENT IN A WORK CONTEXT BY PREDICTING THE EFFECTIVENESS OF WORK CENTERS USING MACHINE LEARNING

Major Project submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

T. HIMAJA SREE (221810305060)

T. MANOJ (221810305059)

Y. VISHNU VARDHAN (121810303031)

T.V.L.N. SAMHIGNA (221810305058)

Under the Guidance of

Dr. S. Phani Kumar

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM

(Deemed to be University)

HYDERABAD

APRIL 2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM

(Deemed to be University)

DECLARATION

We hereby declare that the Major-Project entitled "**ALGORITHMIC MANAGEMENT IN A WORK CONTEXT BY PREDICTING THE EFFECTIVENESS OF WORK CENTERS USING MACHINE LEARNING**" is an original work done in the Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University) submitted in partial fulfillment of the requirements for the award of the degree of B.Tech. In Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree or diploma.

Date: 4/4/22

T. HIMAJA SREE (221810305060)

T. MANOJ (221810305059)

Y.VISHNU VARDHAN (121810303031)

T.V.L.N.SAMHIGNA (221810305058)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM

(Deemed to be University)

CERTIFICATE

This is to certify that Major Project entitled "**ALGORITHMIC MANAGEMENT IN A WORK CONTEXT BY PREDICTING THE EFFECTIVENESS OF WORK CENTERS USING MACHINE LEARNING**" is submitted by **T. HIMAJA SREE (221810305060), T. MANOJ (221810305059), Y. VISHNU (121810303031), SAMHIGNA (221810305058)** in partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering. The Major Project has been approved as it satisfies the academic requirements.

Under the guidance of	**Head of the Department**
Dr. S. Phani Kumar	**Dr. S. Phani Kumar**
Professor	**Professor**

ACKNOWLEDGMENT

Our Major Project would not have been successful without the help of several people. We would like to thank the personalities who were part of our Major Project in numerous ways, those who gave us outstanding support from the birth of the Major Project.

We are extremely thankful to our honorable Pro-Vice Chancellor, **Prof. N. Siva Prasad** for providing necessary infrastructure and resources for the accomplishment of our Major Project.

We are highly indebted to **V. K Mittal,** Director of engineering, **Prof. N. Seetha Ramaiah**, Associate Director of Engineering, for their support during the tenure of the Major Project.

We are very much obliged to our beloved **Prof. S. Phani Kumar**, Head of the Department of Computer Science & Engineering for providing the opportunity to undertake this Major Project and encouragement in completion of this Major Project.

We hereby wish to express our deep sense of gratitude to **Dr. S Aparna**, Assistant Professor, Department of Computer Science and Engineering, School of Technology and to our **Guide Dr. S. Phani Kumar**, HOD, Department of Computer Science and Engineering, School of Technology for the esteemed guidance, moral support and invaluable advice provided by them for the success of the Major Project.

We are also thankful to all the staff members of Computer Science and Engineering department who have cooperated in making our Major Project a success. We would like to thank all our parents and friends who extended their help, encouragement and moral support either directly or indirectly in our Major Project.

Sincerely,

T. HIMAJA SREE (221810305060)

T. MANOJ (221810305059)

Y. VISHNU VARDHAN (121810303031)

T.V.L.N.SAMHIGNA (221810305058)

TABLE OF CONTENTS

1. ABSTRACT

The rapid development of machine-learning algorithms, which underpin contemporary artificial intelligence systems, has created new opportunities for the automation of work processes and Management functions. The effectiveness of this algorithmic management depends on various Developmental aspects like on-time delivery, satisfaction of employees, managers and customer And cost of implementation. Also the quality aspects like on-time completion of unit & Regression testing, satisfaction of various teams involved in integration testing, coverage of end To end functionality in system testing and customer satisfaction during the user acceptance testing plays a role in deciding the effectiveness of algorithmic management. This project aims at Analyzing the dataset comprising of the development and quality aspects of Work centers by Using QDA Classifier, Gaussian Process classification, Decision Tree Classifier and Random Forest Classifier.

The Gaussian Processes Classifier is a classification machine learning algorithm. Gaussian Processes are a generalization of the Gaussian probability distribution and can be used as the Basis for sophisticated non-parametric machine learning algorithms for binary classification Tasks and regression. Discriminant analysis is a technique that is used by the researcher to analyze the research data when the criterion or the dependent variable is categorical and the predictor or the independent variable is interval in nature. LDA (Linear Discriminant Analysis) is used when a linear boundary is required between classifiers and QDA (Quadratic Discriminant Analysis) is used to find a non-linear boundary between classifiers. Decision Tree is a supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

2. INTRODUCTION

In recent years, the "gig economy" has revolutionized the way millions of people work. Detractors argue that the gig economy erodes workplace regulations and standards while encouraging businesses to treat workers as disposable. Proponents argue that the gig economy gives people more flexibility and opportunities while lowering barriers to entry into the labor market, while detractors argue that it erodes workplace regulations and standards while encouraging businesses to treat workers as disposable.

As the name implies, algorithmic management is the use of computer algorithms and artificial intelligence approaches to manage a team of humans. Algorithmic management tries to automate substantial elements of the managerial decision-making process by gathering huge amounts of data, particularly data on employee performance.

While estimating the prevalence of algorithmic management is difficult, there are a few indicators. For example, 40% of human resources departments at multinational corporations are presently utilizing AI software. Here are a few examples of how algorithmic management has begun to permeate the mainstream:

- HI revue, a video interviewing software platform, is testing a face analysis AI that looks at things like a candidate's facial expressions, tone of voice, and language use. HireVue claims that the new technology can speed up the recruiting process by 90%, but others believe that it may exacerbate existing socioeconomic disparities.
- Algorithms control workers at Amazon's warehouse in Melbourne, Australia, determining which things need to be selected, transported, stored, and dispatched. Employees say they are under pressure to raise their "pick rate," which is a metric that measures how many things are grabbed from shelves every hour.

In many conventional organizational contexts, algorithm implementation is still dependent on dash-boards or "decision support" systems that produce suggestions to managers on actions to take.

Algorithmic competences are talents that enable employees to form symbiotic connections with algorithms. Algorithmic competences include critical thinking in addition to data-driven analytical skills that enhance interactions between employees and algorithms.

Thus, algorithmic management refers to the employment of such computer techniques in the administration of an organization. As technology advancements have expanded the possibilities for collecting and processing input data from cameras, sensors, audio devices, biometrics, and text, the potential economic benefits of algorithmic decision making have also expanded.

There is currently no large-scale representative study into algorithmic management being conducted. As a result, the research that does exist is centered on qualitative case studies, most of which has been conducted empirically in the context of platform work. Digital labor platforms are digital infrastructures that serve as mediators in multi-sided marketplaces by bringing together two or more separate user groups.

Thus, platforms have the potential to empower employees to independently contract with a diverse range of clients and consumers, and to varied degrees, pick the clients and jobs they accept, how they carry out those services, and, in the majority of situations, the rates they charge to accomplish them.

2.1 Machine Learning Algorithms Implemented

In this project we have used **QDA Classifier, Gaussian Process classification, Decision Tree Classifier and Random Forest Classifier** to analyze the dataset. We will be comparing all the accuracies of these algorithms and consider the best accuracy rate algorithm.

Gaussian Process Classifier is a classification machine learning algorithm. Gaussian Processes are an extension of the Gaussian probability distribution that may be used to underpin advanced non-parametric machine learning methods for classification and regression.

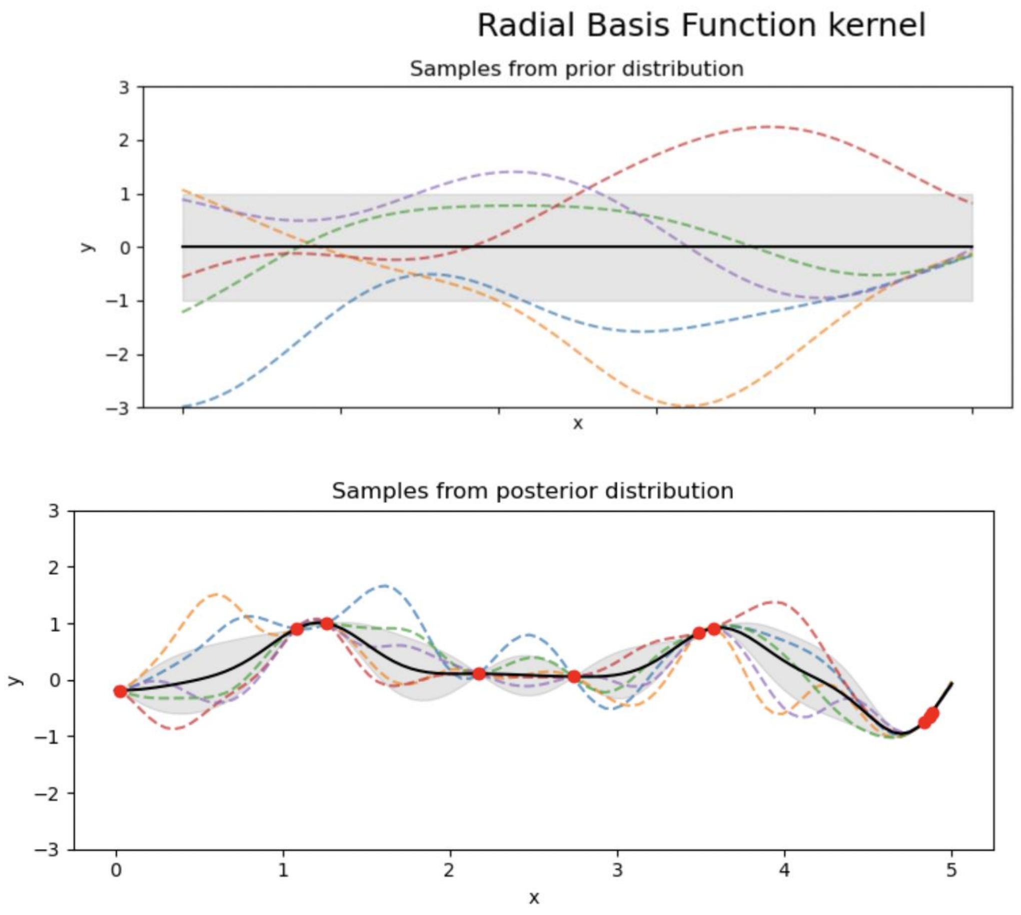
Kernels for Gaussian Processes: Kernels (also known as "covariance functions" in the context of GPs) are an important component of GPs that influence the form of the GP's prior and posterior. They encapsulate the assumptions on the function being learnt by defining the "similarity" of two data points and assuming that similar data points should have similar goal values. We implemented the **Radial Basis Function Kernel** which is a stationary kernel. It's sometimes referred to as the "squared exponential" kernel. It is defined by a length-scale parameter l>0, which can be a scalar (isotropic variant of the kernel) or a vector with the same number of dimensions as the inputs x. (anisotropic variant of the kernel).

The kernel is provided by:

$$k(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)^2}{2l^2}\right)$$

Where d (.,.) denotes the Euclidean distance. Because this kernel is endlessly differentiable, GPs using it as a covariance function have mean square derivatives of all orders and are thus exceedingly smooth.

The prior and posterior of a GP generated from an RBF kernel are depicted in the picture below:



Linear Discriminant Analysis (LDA) and **Quadratic Discriminant Analysis** (QDA) are two classic classifiers, featuring linear and quadratic decision surfaces, respectively.
These classifiers are appealing because they have readily calculated closed-form solutions, are naturally multiclass, have proved to operate well in reality, and have no hyper parameters to modify.

Decision Tree is a supervised learning approach that may be used to solve classification and regression issues; however, it is most commonly used to solve classification problems. It is a tree-structured classifier in which internal nodes contain dataset attributes, branches represent decision rules, and each leaf node represents the result. A Decision tree has two nodes: the Decision Node and the Leaf Node. Decision nodes are used to make any decision and have several branches, whereas Leaf nodes are the result of those decisions and have no more branches.

Random Forest is a well-known machine learning algorithm from the supervised learning approach. It may be applied to both classification and regression issues in machine learning. It is built on the notion of ensemble learning, which is the process of merging numerous classifiers to solve a complicated issue and enhance the model's performance. "Random Forest is a classifier that comprises a number of decision trees on various subsets of the provided dataset and takes the average to enhance the predicted accuracy of that dataset," as the name implies. Instead of depending on a single decision tree, the random forest collects the forecasts from each tree and predicts the final output based on the majority vote of predictions.

3. Literature Review

Machine learning (ML) algorithms increasingly affect many elements of ordinary human lives through the suggestions and activities they advise, whether it's restaurants to try, movies to watch, or routes to travel. Algorithms also affect organizational behavior by partially or completely automating worker management, coordination, and administration (Crowston and Bolici, 2019). This trend, sometimes known as "algorithmic management" or "management-by-algorithm," is defined as the outsourcing of managerial duties to algorithms (Lee, 2018; Lee et al., 2015; Noponen, 2019). The data that powers predictive modeling approaches is a distinguishing aspect of algorithmic management, with many acknowledging that the political economics of data acquisition is a crucial driver in changing labor standards (Dourish, 2016; Newlands, 2020; Shestakofsky, 2017). Prior algorithmic management research has concentrated on platform-mediated gig employment, where employees on non-standard contracts typically commit only for a short period of time to a certain firm (Harms and Han, 2019; Jarrahi and Sutherland, 2019). Algorithmic systems in the gig economy analyze worker performance, do job matching, provide employee rankings, and may even mediate conflicts between workers, according to Huws (2016). (Duggan et al., 2019; Wood et al., 2018). The main contribution of this research is to examine how the advent of algorithmic management in both standard and non-standard work situations interacts with pre-existing organizational dynamics, roles, and skills.

First, they defined algorithmic management as a socio-technical term, examining how it evolves as a result of organizational decisions. Following that, they discussed how algorithmic management affects power relations at work, boosting managers' control over workers while diminishing managerial authority. Then, we'll look at how algorithmic management changes organizational roles by cultivating both algorithmic competence and opposing attitudes like algorithm aversion and cognitive complacency. We explicitly address how the linked technical and organizational opacity of algorithms impacts employees' abilities, as well as information and knowledge sharing, in this paper.

Finally, they wrapped up by putting this essay within bigger conversations about the future of labor, responsibility, and determining next moves.

4. PROBLEM IDENTIFICATION & OBJECTIVES

In the existing system, the work to the employees is assigned and tracked manually by the managers. The effectiveness of the work assigned, whether the work has been done with full efficiency or not, is being checked by managers manually in workplaces.

In algorithmic management the assigning and tracking of the tasks is done automatically through the work management system. The proposed project aims at determining the effectiveness of such algorithmic management, by analyzing a dataset consisting of development aspects and quality aspects, using quadratic discriminant analysis and Gaussian process analysis. Algorithmic management enables processes to be scaled by coordinating the actions of vast, disaggregated workforces, for example, or by utilizing data to optimize for desired outcomes such as decreased labor costs.

The project is useful to the work center management to determine the effectiveness of algorithmic management. It is useful to the data analysts to understand more about quadratic discriminant analysis and Gaussian process analysis. The project finally leads to the improvement of the effectiveness of the work center.

5. SYSTEM ANALYSIS

5.1 Requirement Analysis

5.1.1 Hardware Requirements

1. It requires a minimum of 2.16 GHz processor.
2. It requires a minimum of 4 GB RAM.
3. It requires 64-bit architecture.
4. It requires a minimum storage of 500GB.

5.1.2 Software Requirements

1. It requires a 64-bit Windows Operating System.
2. Python Qt Designer for designing user interfaces.
3. SQLite3 server for storing database Entities.
4. Pyuic for converting the layout designed user interface (UI) to python code.
5. Python language for coding.

5.2 Feasibility Analysis

As the name implies, a feasibility study is used to determine the viability of an idea, such as ensuring a project is legally and technically feasible as well as economically justifiable. It tells us whether a project is worth the investment—in some cases, a project may not be doable. There can be many reasons for this, including requiring too many resources, which not only prevents those resources from performing other tasks but also may cost more than an organization would earn back by taking on a project that isn't profitable.

5.2.1 Economical Feasibility

This assessment typically involves a cost/ benefits analysis of the project, helping organizations determine the viability, cost, and benefits associated with a project before financial resources are allocated. It also serves as an independent project assessment and enhances project credibility helping decision makers determine the positive economic benefits to the organization that the proposed project will provide.

6. SYSTEM DESIGN & OVERVIEW OF TECHNOLOGIES

6.1 Architecture Diagram

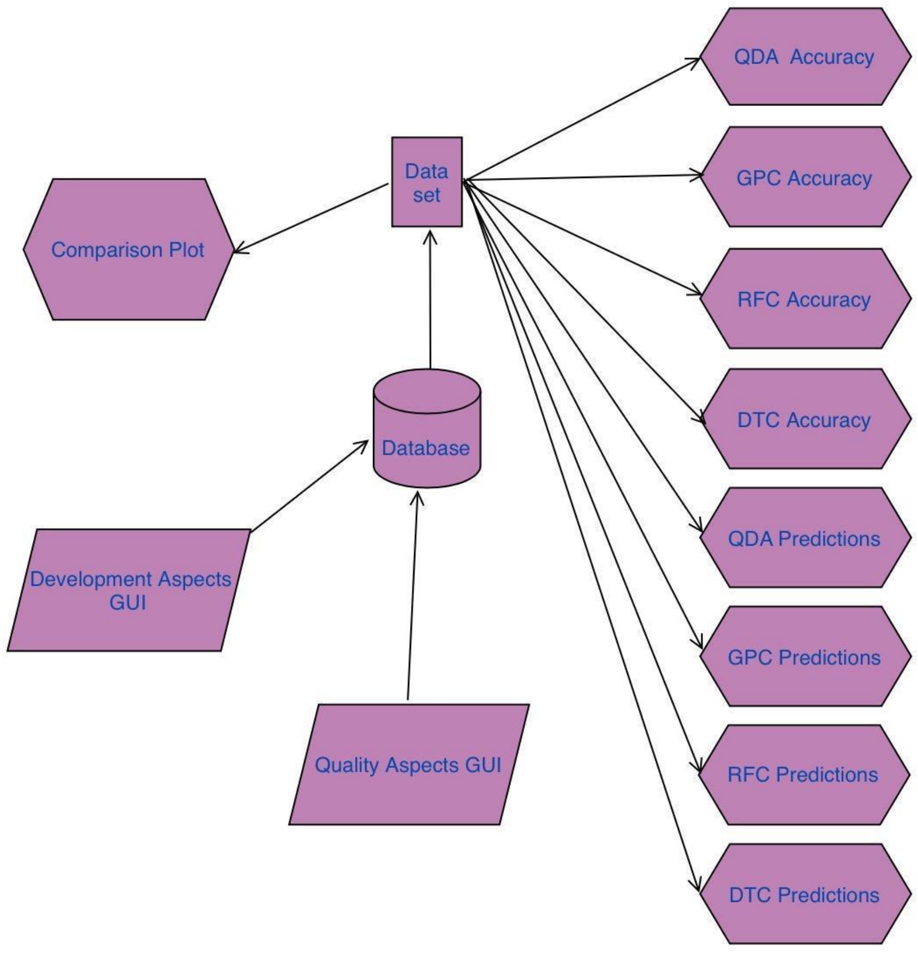


Fig 1

In Fig 1, we can see that first Developmental Aspects and Quality Aspects are entered and stored into the database. From the data set, we use our ML algorithms to predict and know the accuracies of it. We used QDA, GPC, RFC, DTC algorithms to predict the effectiveness of the data set and their accuracies respectively.

6.2 UML Diagrams

UML (Unified Modeling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG), and UML 1.0 specification draft was proposed to the OMG in January 1997. It was initially started to capture the behavior of complex software and non-software systems, and now it has become an OMG standard. This tutorial gives a complete understanding of UML.

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

UML was created by the Object Management Group (OMG), and UML 1.0 specification draft was proposed to the OMG in January 1997.

OMG is continuously making efforts to create a true industry standard.

- UML stands for Unified Modeling Language.
- UML is different from the other common programming languages such as C++, Java, COBOL, etc.
- UML is a pictorial language used to make software blueprints.
- UML can be described as a general-purpose visual modeling language to visualize, specify, construct, and document software systems.
- Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well. For example, the process flow in a manufacturing unit, etc.
  UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object- oriented analysis and design. After some standardization, UML has become an OMG standard.

Various UML diagrams are:

- Use case Diagram
- Sequence Diagram
- Class Diagram
- Activity Diagram
- Collaboration Diagram
- State chart Diagram

6.2.1 Use Case Diagram

The objective of a use case diagram is to capture a system's dynamic aspect. However, this description is too broad to characterize the objective adequately, given the purpose of the other four diagrams (activity, sequence, collaboration, and State chart) is the same? We'll look into a specific objective that will set it apart from the other four diagrams.
Use case diagrams are used to collect a system's needs, covering both internal and external

impacts. These are largely design specifications. As a result, when a system's features are assessed, use cases are created and actors are identified. After the primary work is completed, use case diagrams are created to show the outside perspective.

In a nutshell, the following are the aims of use case diagrams:

• This is a tool for gathering a system's requirements.

• Can be used to acquire a perspective of a system from the outside. • Recognize the system's external and internal influences.

• To demonstrate how the requirements interact with each other by using actors.

Where do we use USE CASE Diagram?

There are five diagrams in UML to depict a system's dynamic view, as we've already discussed. Every model now serves a distinct purpose. Actually, these distinct goals are various aspects of a working system.

Different types of diagrams are required to comprehend the dynamics of a system. One of them is a use case diagram, which is used to collect system needs and actors.

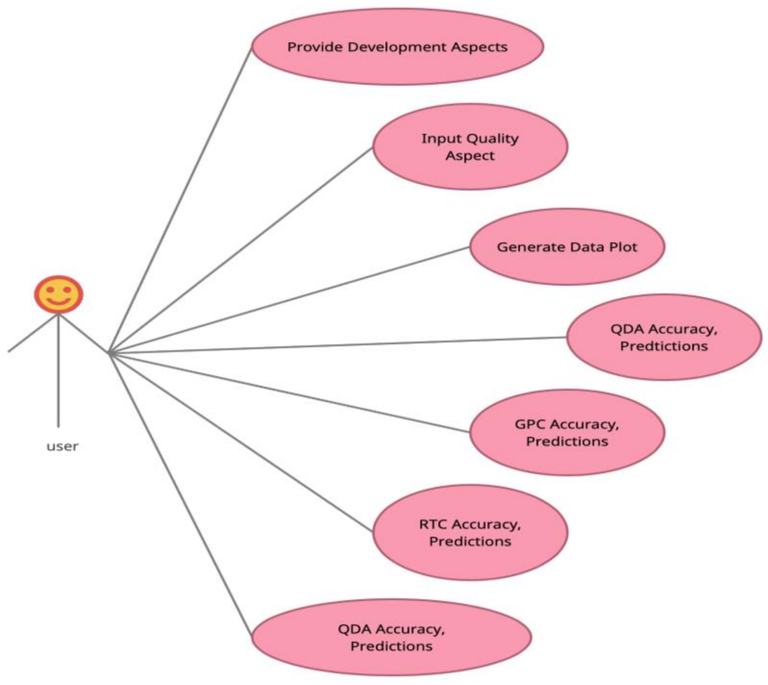

Provide Development Aspects
Input Quality Aspect
Generate Data Plot
QDA Accuracy, Predtictions
GPC Accuracy, Predictions
RTC Accuracy, Predictions
QDA Accuracy, Predictions
user

Fig 2

Description of Use case diagram

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

As we can see in Fig 2 the user is interacting with the system by a UI through which the customer can perform above mentioned operations like providing Work center Development Aspects, Quality Aspects, and then calculating the accuracies of QDA, GPC, DTC, and RFC along with the plotting of Training vs. testing dataset.

6.2.2 Sequence Diagram

Diagrams depicting the sequence of events. The Unified Modeling Language (UML) is a software engineering modeling language that tries to establish standard ways to depict a system's architecture. UML is used to create several types of diagrams, including interface, structure, and behavior diagrams. The most widely used interaction diagram is a sequence diagram.

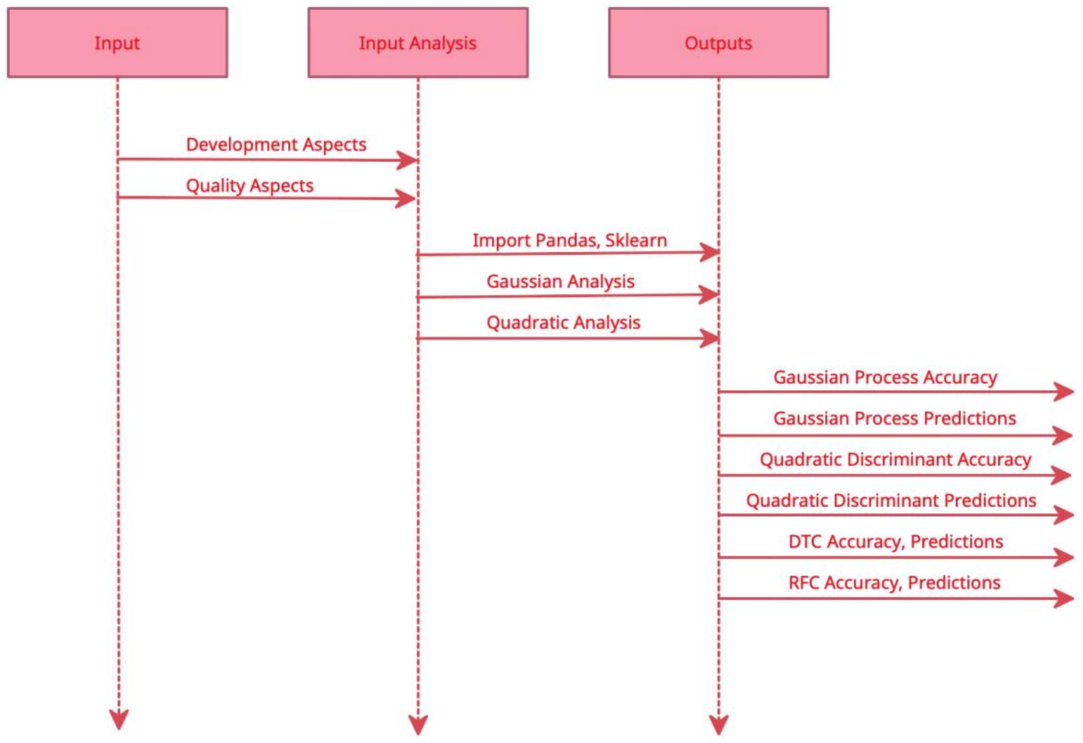

Input
Input Analysis
Outputs
Development Aspects
Quality Aspects
Import Pandas, Sklearn
Gaussian Analysis
Quadratic Analysis
Gaussian Process Accuracy
Gaussian Process Predictions
Quadratic Discriminant Accuracy
Quadratic Discriminant Predictions
DTC Accuracy, Predictions
RFC Accuracy, Predictions

Fig 3

Description of sequence diagram

In Fig 3 mentioned sequence diagram we have to go in sequence: Enter the needed details as shown in the above figure, Provide the quality Aspects, Work center Development Aspects and then calculate the accuracies of QDA, GPC, DTC & RFC along with the predictions.

6.2.3 Activity Diagram

Another significant diagram in UML for describing the system's dynamic features is the activity diagram. An activity diagram is essentially a flowchart that depicts the movement of information from one action to the next. The action can be thought of as a system operation. Control is passed from one operation to the next. This flow can be sequential, branching, or running simultaneously. Different elements such as fork, join, and others are used in activity diagrams to cope with different types of flow control.

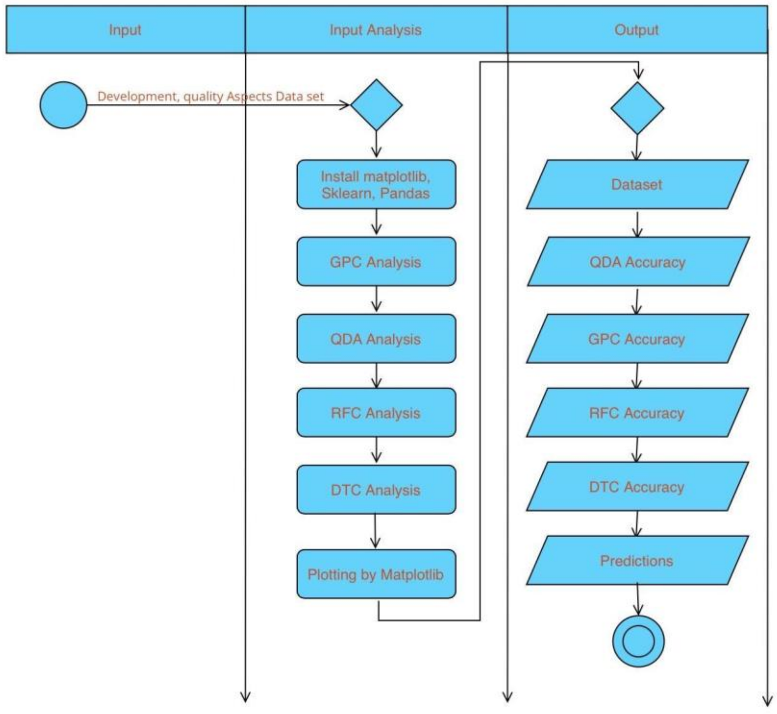


Fig 4

Description of Activity diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So, the control flow is drawn from one operation to another. In the activity diagram(Fig 4) we can see that first provide the dataset consisting of Work center Development Aspects, quality Aspects, and then train the model, test the model and calculate the accuracies along with the predictions of QDA, GPC, DTC & RFC.

6.2.4 Data Flow Diagram

A data flow diagram (DFD) shows how information flows through a process or system. It shows data inputs, outputs, storage sites, and paths between each destination using predetermined symbols such as rectangles, circles, and arrows, as well as short text labels. Data flowcharts can range from simple, even hand-drawn process overviews to multi-level DFDs that go further into how data is processed. They can be used to investigate an existing system or to create a new one. A DFD, like all the best diagrams and charts, can frequently graphically "express" things that are difficult to describe in words, and they are appropriate for both technical and nontechnical audiences, ranging from developers to CEOs.

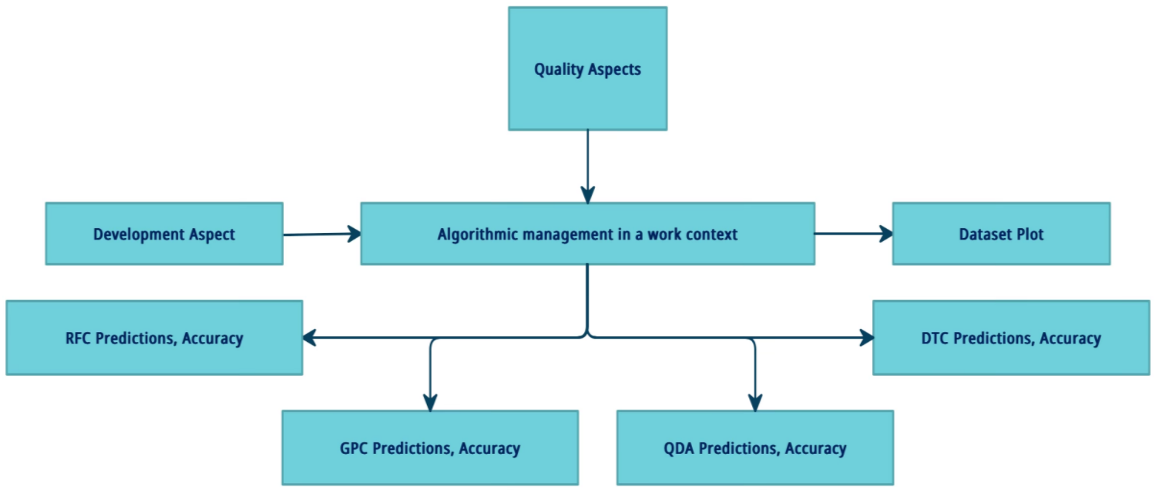

Quality Aspects
Development Aspect
Algorithmic management in a work context
Dataset Plot
RFC Predictions, Accuracy
DTC Predictions, Accuracy
GPC Predictions, Accuracy
QDA Predictions, Accuracy

Fig 5

Description of Data Flow Diagram

Work center Development Aspects as well as Quality Aspects are to be provided as Input to the system, using the corresponding user interfaces. The system then calculates QDA Accuracy & Predictions, GPC Accuracy & Predictions, RFC Accuracy & Predictions, and DTC Accuracy & Predictions.

6.3 Overview of Technologies

6.3.1 Python

Python is an object-oriented scripting language that is high-level, interpreted, and interactive. Python is written in a way that makes it very easy to understand. It typically employs English keywords in place of punctuation in other languages, and it has fewer syntactic structures than other languages. Python is a must-know language for students and working professionals who want to become exceptional software engineers, especially if they work in the Web Development field.

The important benefits of learning Python:

• Python is interpreted-The interpreter processes Python at runtime. Before running your software, you don't need to compile it. Like PERL and PHP, this is a scripting language.

• Python is Interactive-When writing programmers, you can sit at a Python prompt and interact directly with the interpreter.

Python supports the Object-Oriented programming style or approach, which encapsulates code within objects.

• Python is a Beginner's Language-Python is an excellent language for beginning programmers, as it allows for the creation of a wide range of programmers, ranging from simple text processing to web browsers and games.

6.3.2 SK-Learn

SciKit-Learn is abbreviated as SK-Learn. Scikit-learn is Python's most usable and robust machine learning library. It uses a Python consistency interface to give a set of efficient tools for machine learning and statistical modeling, such as classification, regression, clustering, and dimensionality reduction. NumPy, SciPy, and Matplotlib are the foundations of this package, which is mostly written in Python.

6.3.3 Numpy

NumPy (numerical Python) is a library that consists of multidimensional array objects and a set of procedures for manipulating them. NumPy may be used to conduct logical and mathematical operations on arrays. The architecture and environment of NumPy are covered in this session. It also covers array functions, indexing types, and other related topics. There's also a primer on Matplotlib. For a better understanding, all of this is taught with examples.

6.3.4 Pandas

Pandas is a Python library that provides high-performance, easy-to-use data structures and data analysis tools under the BSD license. Python with Pandas is utilized in a variety of academic and commercial disciplines, including finance, economics, statistics, analytics, and so on. In this article, we'll go through the different capabilities of Python Pandas and how to put them to use.

6.3.5 Matplotlib

Matplotlib is a widely used Python tool for data visualization. It's a cross-platform library for plotting data in arrays in 2D. It provides an object- oriented API for embedding plots in Python GUI toolkits like PyQt and WxPython Tkinter. It works in Python and IPython shells, as well as Jupyter notebooks and web application servers.

6.3.6 Sys Module

The sys module provides information about constants, functions and methods of the Python interpreter. Dir (system) gives a summary of the available constants, functions and methods. Another possibility is the help () function. Using help (sys) provides valuable detailed information. The module sys informs e.g. about the maximum recursion depth (sys.getrecursionlimit ()) and provides the possibility to change (sys.setrecursionlimit ()). The current version number of Python can be accessed as well by using this module. Lots of scripts need access to the arguments passed to the script, when the script was started. argv argv (or to be precise sys.argv) is a list, which contains the command-line arguments passed to the script. The first item of this list contains the name of the script itself. The arguments follow the script name. Every serious user of a UNIX or Linux operating system knows standard streams, i.e. input, standard output and standard error. They are known as pipes. They are commonly abbreviated as stdin, stdout, stderr. The standard input (stdin) is normally connected to the keyboard, while the standard error and standard output go to the terminal (or window) in which you are working. These data streams can be accessed from Python via the objects of the sys module with the same names, i.e. sys.stdin, sys.stdout and sys.stderr.

6.4 Machine Learning

Machine learning is a sub-field of computer science that gives “computers the ability to learn without being explicitly programmed”. Machine Learning is basically making a machine learn from the provided data. Machine Learning algorithms use historical data as input to predict new output values.

There are two types of variables in ML:

- DependentVariables:
  These are the type of variables which depend on other variables. These are known as “Labels”.
- IndependentVariables:
  These are the types of variables which don't depend on other variables. These are known as “Features”.

Machine learning is significant because it provides businesses with insights into customer behavior and business operational patterns, as well as assisting in the development of new products. Machine learning is a key component of many of today's most successful businesses, like Facebook, Google, and Uber. For many businesses, machine learning has become a key competitive differentiation.

Machine Learning can be divided into three categories:

6.4.1 Supervised Learning

Data scientists feed algorithms with labeled training data and identify the variables they want the algorithm to examine for correlations in supervised learning. The algorithm's input and output are both provided.

Supervised Machine Learning challenges can be divided into two categories:

1. Classification:
   This is determined by the output variable type. It's a classification problem if the output variable is categorical, that is, if the output must fall into one of two groups. (For example, red, blue, purple, and so on...)
2. Regression:
   A regression problem exists when the output variable is a real number and the goal is to forecast a value in the continuous order. (For instance, height ranges from 0 to 10 feet.)

   Some of the algorithms under Supervised Learning:

   - Logistic Regression
   - Decision Tree
   - Random Forest
   - KNN

6.4.2 Unsupervised Learning

Machine learning algorithms that train on unlabeled data are used in this sort of machine learning. The algorithm looks for important connections in data sets. Algorithms' training data, as well as the predictions or suggestions they produce, are preset. Unsupervised Machine Learning tasks are divided into two categories:

- Clustering:
  When you identify groupings within the input data, you have a clustering problem. (For instance, grouping voting habits by gender.)

- Association:

  When you uncover rules inside the input, this is known as association.

Some of the algorithms under Unsupervised Learning:

- Apriori algorithm
- K-means.

6.4.3 Reinforcement Learning

Reinforcement Learning is most commonly used to train a computer how to perform a multi-step process with well-defined constraints. Data scientists train an algorithm to do a task and provide it with positive or negative feedback as it figures out how to complete it. However, for the most part, the algorithm selects what actions to take along the way on its own.

Reinforcement learning, unlike supervised and unsupervised machine learning, is focused on determining the optimum course to take in a situation in order to maximize reward. The choices are made one after the other. The algorithm will either have a positive or negative reward for each step it takes on the way to total reward. The overall reward is calculated by adding all positive and negative rewards received along the way. The goal is to identify the most rewarding path.

6.5 Machine Learning Algorithms

6.5.1 Gaussian Process Classifier

The Gaussian Processes Classifier is a classification machine learning algorithm. Gaussian Processes are a generalization of the Gaussian probability distribution and can be used as the basis for sophisticated non-parametric machine learning algorithms for classification and regression. They are a type of kernel model, like SVMs, and unlike SVMs, they are capable of predicting highly calibrated class membership probabilities, although the choice and configuration of the kernel used at the heart of the method can be challenging.

6.5.2 Quadratic Discriminant Analysis

Discriminant analysis is a technique that is used by the researcher to analyze the research data when the criterion or the dependent variable is categorical and the predictor or the independent variable is interval in nature. Quadratic Discriminant Analysis (QDA) is a generative model. QDA assumes that each class follows a Gaussian distribution. The class-specific prior is simply the proportion of data points that belong to the class. The class-specific mean vector is the average of the input variables that belong to the class.

6.5.3 Random Forest Classifier

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

6.5.4 Decision Tree Classifier

Decision Tree is a supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given dataset. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

6.6 Metrics

Precision: Precision is the ratio of true positives (TP) by the sum of true positives (TP) and false positives (FP).

$$Precision = \frac{True\ Positives\ (TP)}{True\ Positives\ (TP) + False\ Positives\ (FP)}$$

Let us consider a data set with two target classes (say positive and negative) then precision tells us, out of total predicted positive values how many were actually positive. Precision should be used based on the use case. Take an example use case of spam detection. If our model detects a mail as spam which was not actually a spam mail then the user might miss an important mail i.e. here false positives should be reduced. So, in this use case we need to use precision as a metric to measure the quality of our classifier.

Recall: Recall is the ratio of true positives (TP) by the sum of true positives (TP) and false negatives (FN).

$$Recall = \frac{True\ Positives\ (TP)}{True\ Positives\ (TP) + False\ Negatives\ (FN)}$$

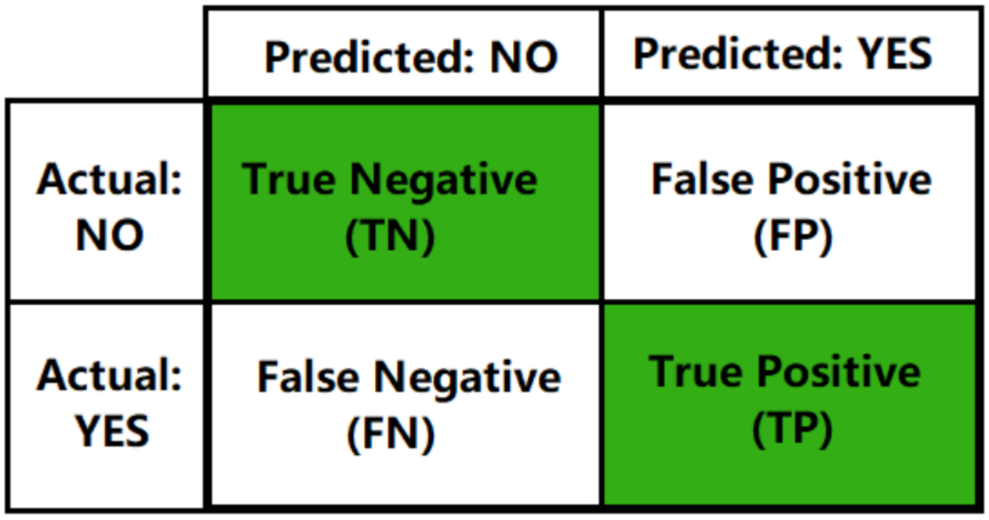
Let us consider a data set with two target classes (say positive and negative) then recall tells us, out of total actual negative values how many did our classifier predict negatively. Similar to precision, recall should also be used based on the use case. Take an example use case of cancer prediction. Consider a person who is actually having cancer but was predicted as a non-cancer patient by our classifier which can lead to mistreatment of the person i.e., here false negatives should be reduced. So, in this case, we need to use recall as a metric to measure the quality of our classifier.

F1Score: F1 score should be used when both precision and recall are important for the use case. F1 score is the harmonic mean of precision and recall. It lies between [0,1].

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

The **F1-score** combines the precision and recall of a classifier into a single metric by taking their harmonic mean. It is primarily used to compare the performance of two classifiers.

Confusion Matrix: A confusion matrix is an N dimensional square matrix, where *N* represents total number of target classes or categories. Confusion matrix can be used to evaluate a classifier whenever the data set is imbalanced. Let us consider a binary classification problem i.e. the number of target classes are 2. A typical confusion matrix with two target classes (say “Yes” and “No”) looks like:



	Predicted: NO	Predicted: YES
Actual: NO	True Negative (TN)	False Positive (FP)
Actual: YES	False Negative (FN)	True Positive (TP)

7. IMPLEMENTATION

7.1 CODING

7.1.1 Main UI

In Fig 6, this is our User Interface where we enter the values of Developmental aspects and Quality aspects, can generate the Training vs Testing Data Plot, Check for QDA and GPC accuracies as well.

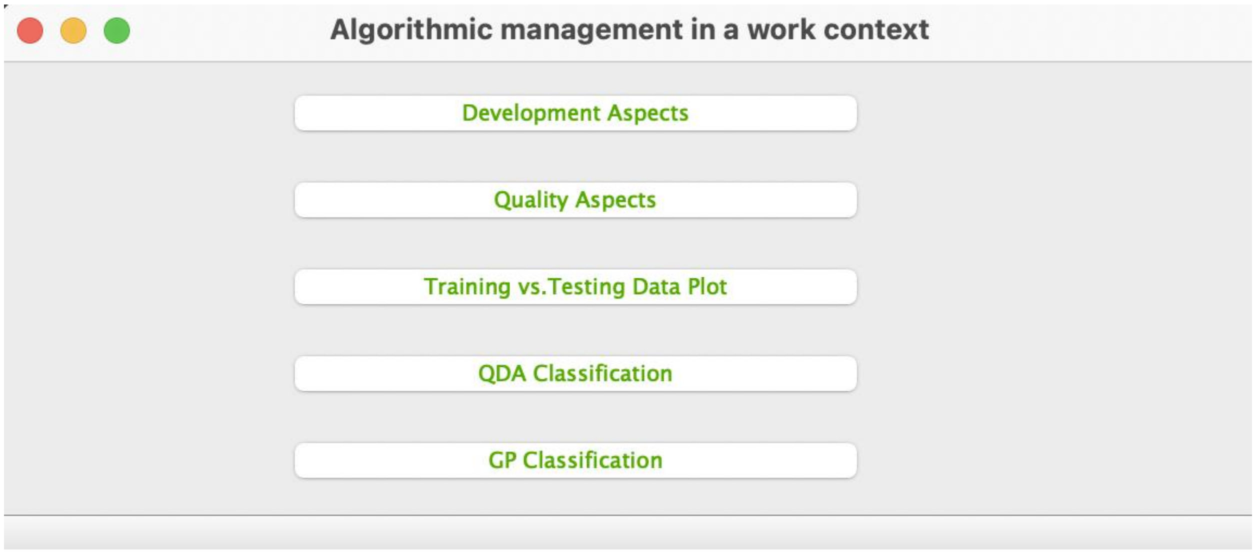
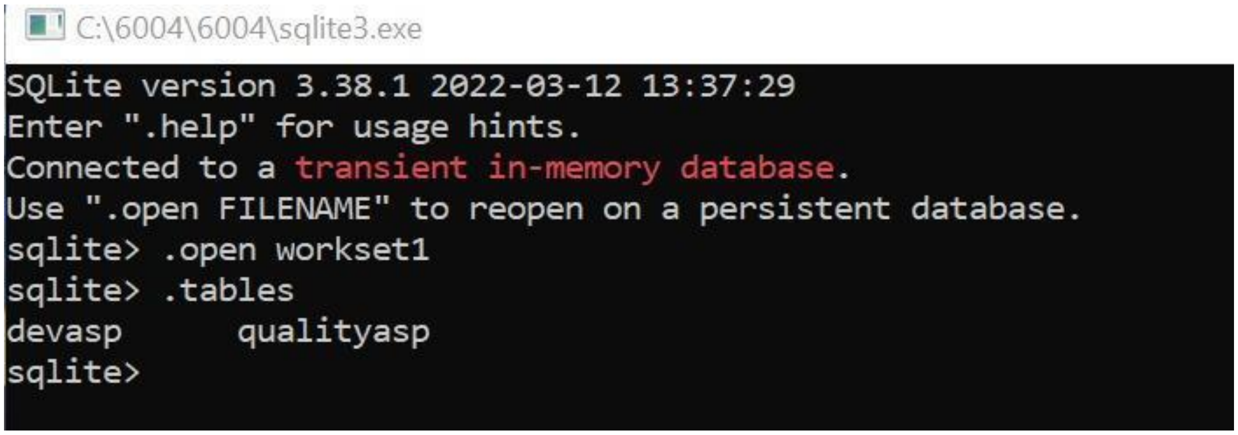

Algorithmic management in a work context
Development Aspects
Quality Aspects
Training vs.Testing Data Plot
QDA Classification
GP Classification

Fig 6

7.1.2 Creating Tables

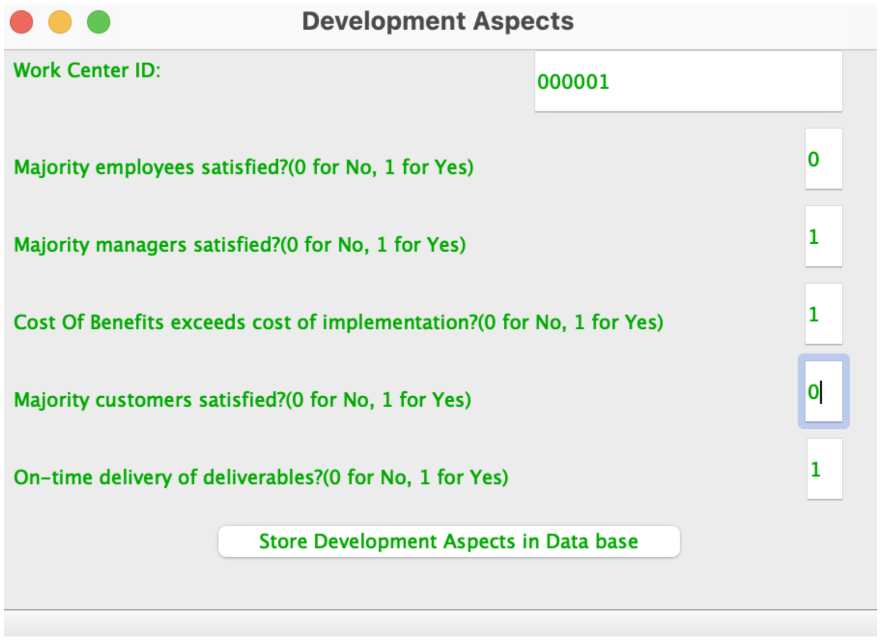


```
C:\6004\6004\sqlite3.exe
SQLite version 3.38.1 2022-03-12 13:37:29
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .open workset1
sqlite> .tables
devasp      qualityasp
sqlite>
```

Fig 7

In Fig 7 we have created two tables ‘devasp’ and ‘qualityasp’in the database. In these tables our particular data which we enter through the UI will be stored.

7.1.3 Storing Data into Database



Work Center ID: 000001

Majority employees satisfied?(0 for No, 1 for Yes) 0

Majority managers satisfied?(0 for No, 1 for Yes) 1

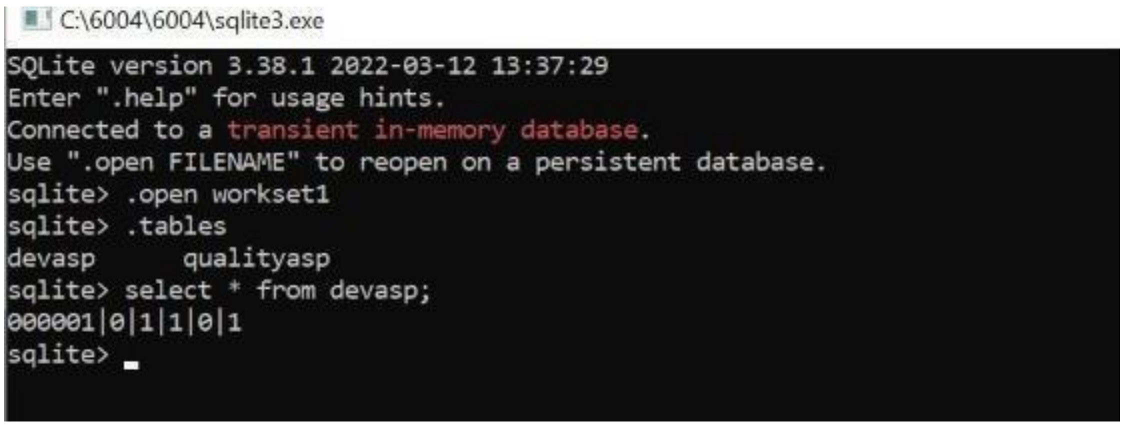
Cost Of Benefits exceeds cost of implementation?(0 for No, 1 for Yes) 1

Majority customers satisfied?(0 for No, 1 for Yes) 0

On-time delivery of deliverables?(0 for No, 1 for Yes) 1

Fig 8

In Fig 8 we can see the developmental aspects where we need to enter the data in binary format. The work center ID should be entered first. Then the parameters are needed to be entered (Yes=1, No=0)

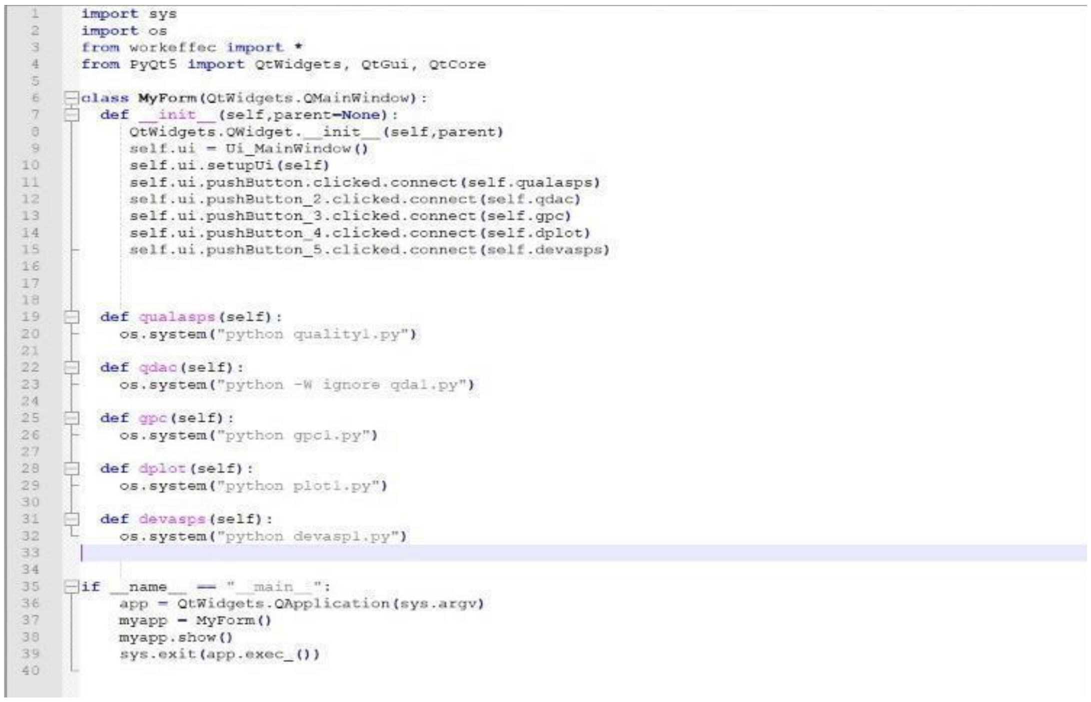


```
SQLite version 3.38.1 2022-03-12 13:37:29
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .open workset1
sqlite> .tables
devasp      qualityasp
sqlite> select * from devasp;
000001|0|1|1|0|1
sqlite> _
```

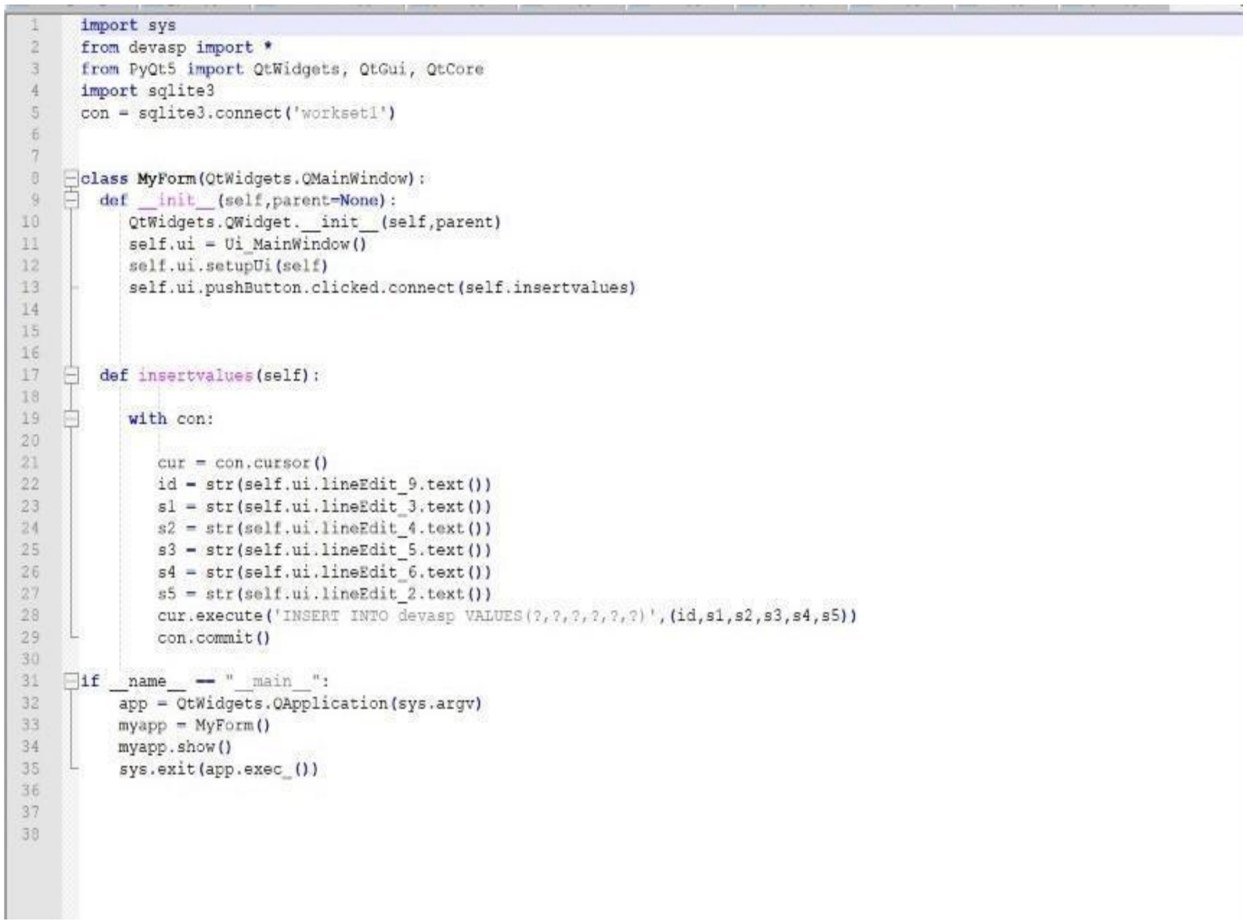
Fig 9

Now, in Fig 9 the entered data has been stored in the database. The stored values are reflected. In the same way we store Quality aspects as well.

7.1.4 UI code and storing data of developmental aspects into data base



```python
import sys
import os
from workeffec import *
from PyQt5 import QtWidgets, QtGui, QtCore

class MyForm(QtWidgets.QMainWindow):
  def __init__(self,parent=None):
     QtWidgets.QWidget.__init__(self,parent)
     self.ui = Ui_MainWindow()
     self.ui.setupUi(self)
     self.ui.pushButton.clicked.connect(self.qualasps)
     self.ui.pushButton_2.clicked.connect(self.qdac)
     self.ui.pushButton_3.clicked.connect(self.gpc)
     self.ui.pushButton_4.clicked.connect(self.dplot)
     self.ui.pushButton_5.clicked.connect(self.devasps)



  def qualasps(self):
    os.system("python quality1.py")

  def qdac(self):
    os.system("python -W ignore qda1.py")

  def gpc(self):
    os.system("python gpc1.py")

  def dplot(self):
    os.system("python plot1.py")

  def devasps(self):
    os.system("python devaspl.py")


if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```



```python
import sys
from devasp import *
from PyQt5 import QtWidgets, QtGui, QtCore
import sqlite3
con = sqlite3.connect('workset1')


class MyForm(QtWidgets.QMainWindow):
  def __init__(self,parent=None):
     QtWidgets.QWidget.__init__(self,parent)
     self.ui = Ui_MainWindow()
     self.ui.setupUi(self)
     self.ui.pushButton.clicked.connect(self.insertvalues)



  def insertvalues(self):

     with con:

        cur = con.cursor()
        id = str(self.ui.lineEdit_9.text())
        s1 = str(self.ui.lineEdit_3.text())
        s2 = str(self.ui.lineEdit_4.text())
        s3 = str(self.ui.lineEdit_5.text())
        s4 = str(self.ui.lineEdit_6.text())
        s5 = str(self.ui.lineEdit_2.text())
        cur.execute('INSERT INTO devasp VALUES(?,?,?,?,?,?)',(id,s1,s2,s3,s4,s5))
        con.commit()

if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    myapp = MyForm()
    myapp.show()
    sys.exit(app.exec_())
```

7.2 TESTING

Software testing is an integral part of software quality assurance since it is the final examination of the specification, design, and coding. We planned, through testing, because software is becoming more visible as a system element and the costs associated with a software failure are motivating elements. Testing is the process of running a software to look for errors. Test design for software and other engineering items can be as difficult as the product design itself.

Verification:

Verification is the process of ensuring that the product meets the requirements set forth at the beginning of the development phase. In another way, we want to make sure the product works as expected.

Validation:

Validation is the process of ensuring that at the end of the development phase, the product meets the defined requirements. To put it another way, to ensure that the product meets the needs of the consumer.

There of basically two types of testing:

1. Black-Box testing — when a product is meant to execute a specific function, tests can be performed to ensure that each function is completely operational.
2. White-Box testing — by understanding the product's internal workings, tests may be carried out to guarantee that the product's internal operation meets standards and that all internal components have been appropriately exercised.

This package was tested using both white box and black box testing methodologies. The boundary and intermediate conditions of the whole loop designs have been tested. The test data was created with the goal of ensuring that all conditions and logical decisions were met. The use of exception handlers has taken care of error handling.

The main goal of software development is to find and fix bugs. A sequence of test steps, including unit, integration, validation, and system tests, are developed and executed to achieve this goal. Each test stage is carried out using a set of systematic test techniques that aid in the development of test cases. The abstraction level with which software is regarded grows with each testing phase. Testing is the best way to ensure software quality, and it is more of an umbrella activity than a separate step. This is a separate activity from the software endeavor that has its own phases of analysis, design, implementation, execution, and maintenance.

7.2.1 Unit Testing

Rather than diving into the specifics at the statement level, this testing method treats a module as a single unit that is checked at interfaces and communicates with other modules. The module will be viewed as a black box that will accept input and output. The module calculates and generates outputs for a given set of input combinations.

7.2.2 Integrated Testing

During the creation of software, testing is an important quality control measure. Its primary goal is to discover faults. When sub functions are joined, they may not yield the results that are desired. The problems can be represented using global data structures. Integrated testing is a method of building the programme structure while running the tests. The objective is to generate unit test modules and build a programme structure that has been detected by design in order to reveal faults related with interfacing. In a non-incremental integration, all of the modules are merged ahead of time, and the entire programme is tested. In an endless loop function, faults will emerge. In incremental testing, the programme is built and tested in little chunks, with defects identified and repaired in between.

Different types of incremental integration strategies are:

1. Top – Down Integration
2. Bottom – Up Integration
3. Regression Testing.

7.2.3 Top-Down Integration Test

Modules are connected by working their way down the control hierarchy, starting with the main application. Subordinate modules are incorporated into the framework in one of two ways: breadth first or depth first.

This procedure is of five steps:

- Depending on the integration technique, selected subordinates are replaced one at a time with genuine modules, or all modules are straight to the main programme.
- After each set of tests is completed, a new stub is replaced with the genuine module.
- Regression testing can be used to check that no new mistakes have appeared. This process continues from step 2 until the entire program structure is reached. In top-down integration strategy decision making occurs at upper levels in the hierarchy and is encountered first.
- If major control problems do exist, early recognition is essential.
- If depth-first integration is selected, a complete function of the software may be implemented and demonstrated .Some problems occur when processing at low levels in hierarchy is required to adequately test upper-level steps to replace low-level modules at the beginning of the top-down testing. So, no data flows upward in the program structure.

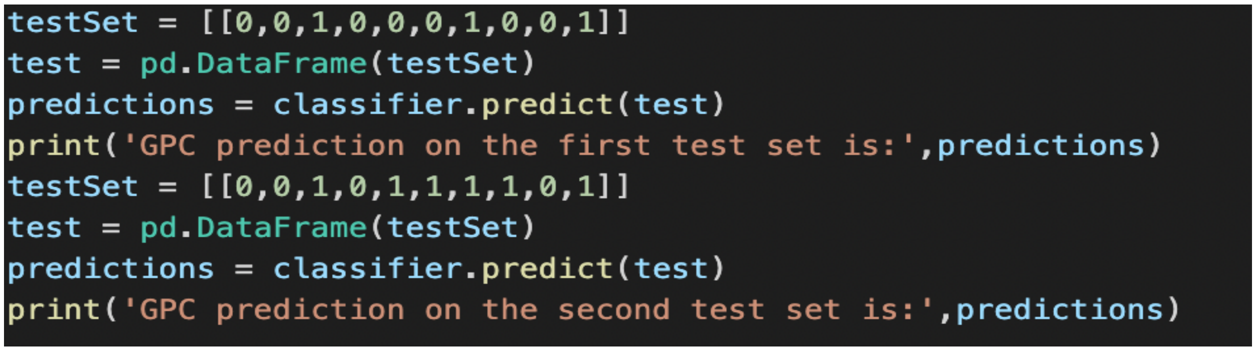
7.2.4 Bottom-Up Integration Test

- Atomic modules are used to start building and testing. Because modules are integrated from the bottom up, processing for modules that are subordinate to a certain level is always available, and stubs are no longer required. This plan is put into action in the steps that follow.
- Low-level modules are combined into clusters that perform a specific software sub function.
- A driver is created to coordinate the input and output of test cases.
- The formed cluster is finally tested.

7.3 Testing Our Test Cases Using ML Algorithms

7.3.1 Gaussian Process Classifier

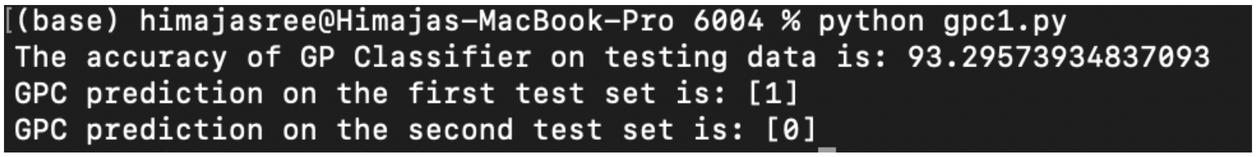
Predictions & Accuracy:



```
testSet = [[0,0,1,0,0,0,1,0,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('GPC prediction on the first test set is:',predictions)
testSet = [[0,0,1,0,1,1,1,1,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('GPC prediction on the second test set is:',predictions)
```

Fig 11

In Fig 11 we implemented Gaussian Process Classifier on our dataset and predicted outcomes of the testing sets as shown.



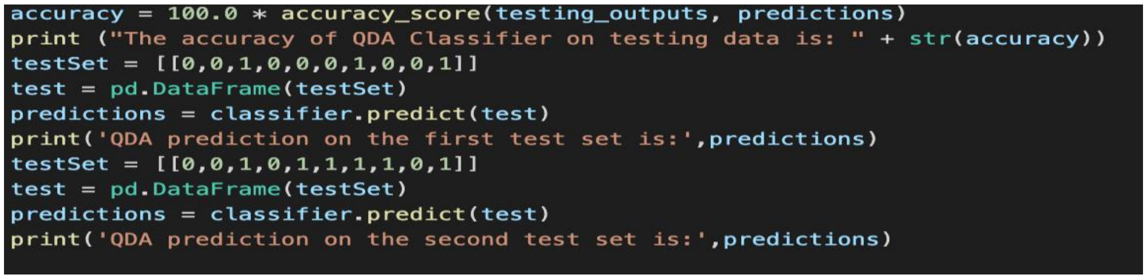
```
(base) himajasree@Himajas-MacBook-Pro 6004 % python gpc1.py
 The accuracy of GP Classifier on testing data is: 93.29573934837093
 GPC prediction on the first test set is: [1]
 GPC prediction on the second test set is: [0]
```

Fig 12

In Fig 12 we have the results of the prediction and accuracy. We got 93% accuracy with the GPC algorithm.

7.3.2 QDA Classifier

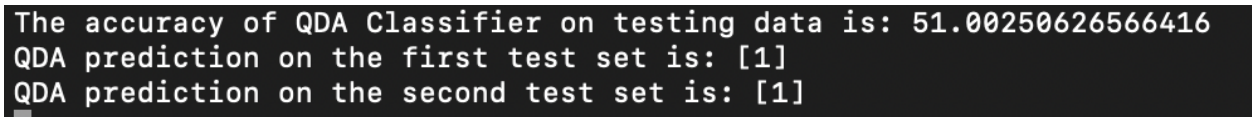
Predictions & Accuracy:



```
accuracy = 100.0 * accuracy_score(testing_outputs, predictions)
print ("The accuracy of QDA Classifier on testing data is: " + str(accuracy))
testSet = [[0,0,1,0,0,0,1,0,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('QDA prediction on the first test set is:',predictions)
testSet = [[0,0,1,0,1,1,1,1,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('QDA prediction on the second test set is:',predictions)
```

Fig 13

In Fig 13, we implemented Quadratic Discriminant Analysis on our dataset and predicted outcomes of the testing sets as shown.



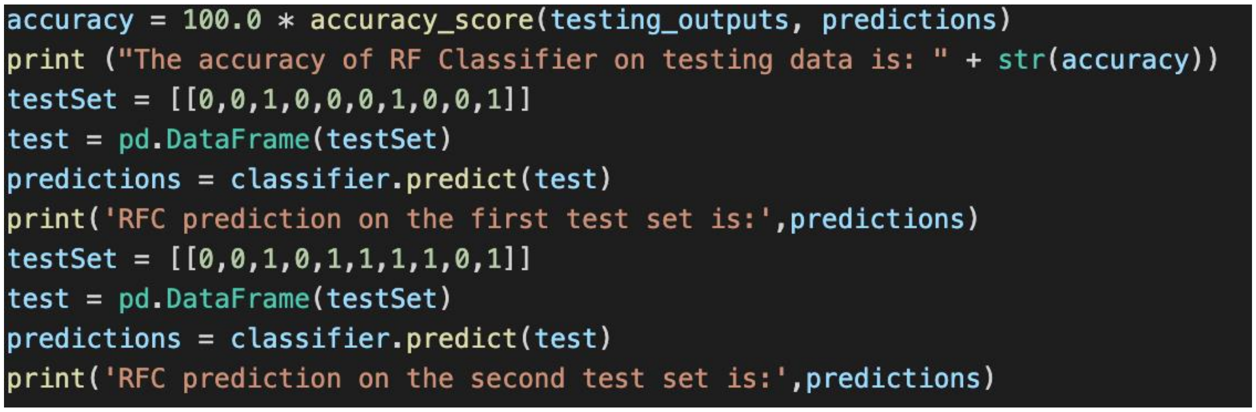
```
The accuracy of QDA Classifier on testing data is: 51.00250626566416
QDA prediction on the first test set is: [1]
QDA prediction on the second test set is: [1]
```

Fig 14

In Fig 14 we have the results of the prediction and accuracy. We got 51% accuracy with the QDA algorithm.

7.3.3 Random Forest Classifier

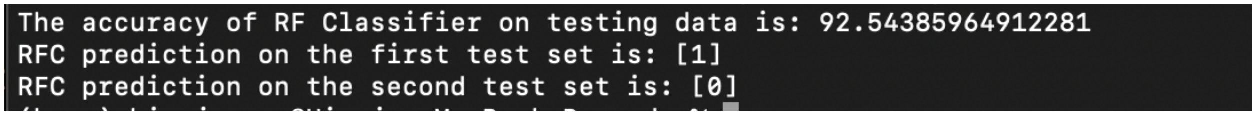
Predictions & Accuracy:



```python
accuracy = 100.0 * accuracy_score(testing_outputs, predictions)
print ("The accuracy of RF Classifier on testing data is: " + str(accuracy))
testSet = [[0,0,1,0,0,0,1,0,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('RFC prediction on the first test set is:',predictions)
testSet = [[0,0,1,0,1,1,1,1,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('RFC prediction on the second test set is:',predictions)
```

Fig 15

In Fig 15, we implemented Random Forest Classifier on our dataset and predicted outcomes of the testing sets as shown.



```
The accuracy of RF Classifier on testing data is: 92.54385964912281
RFC prediction on the first test set is: [1]
RFC prediction on the second test set is: [0]
```

Fig 16

In Fig 16 we have the results of the prediction and accuracy. We got 92% accuracy with the RFC algorithm.

7.3.4 Decision Tree Classifier

Predictions & Accuracy

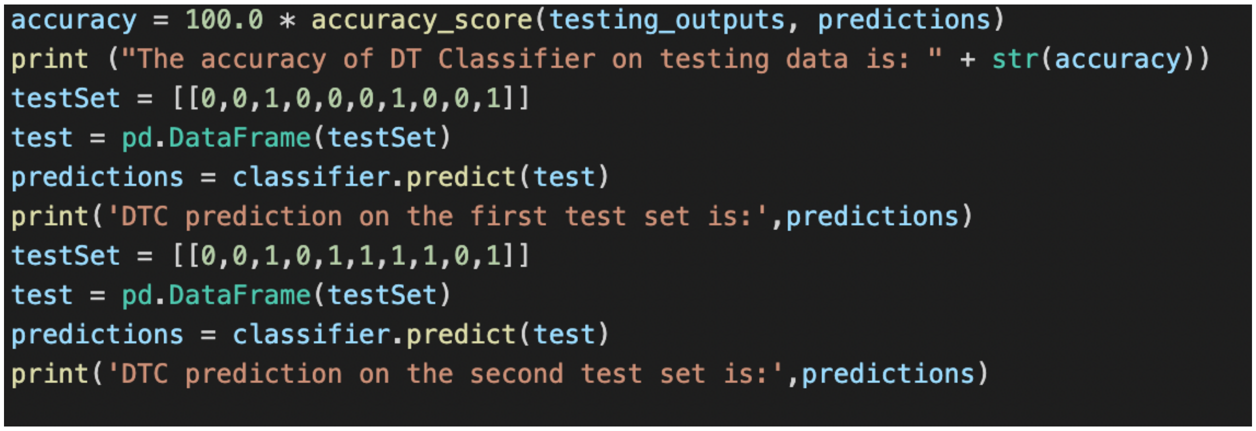

```
accuracy = 100.0 * accuracy_score(testing_outputs, predictions)
print ("The accuracy of DT Classifier on testing data is: " + str(accuracy))
testSet = [[0,0,1,0,0,0,1,0,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('DTC prediction on the first test set is:',predictions)
testSet = [[0,0,1,0,1,1,1,1,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('DTC prediction on the second test set is:',predictions)
```

Fig 17

In Fig 17, we implemented Decision Tree Classifier on our dataset and predicted outcomes of the testing sets as shown.

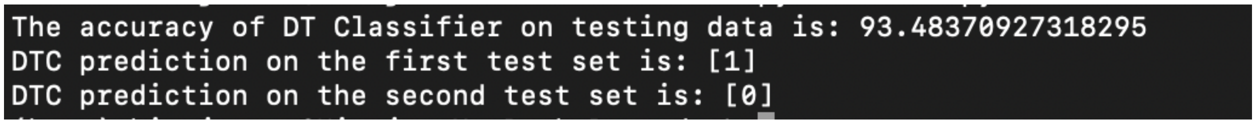

```
The accuracy of DT Classifier on testing data is: 93.48370927318295
DTC prediction on the first test set is: [1]
DTC prediction on the second test set is: [0]
```

Fig 18

In Fig 18 we have the results of the prediction and accuracy. We got 93% accuracy with the DTC algorithm.

7.3.5 Precision, Recall, F1 score, Support

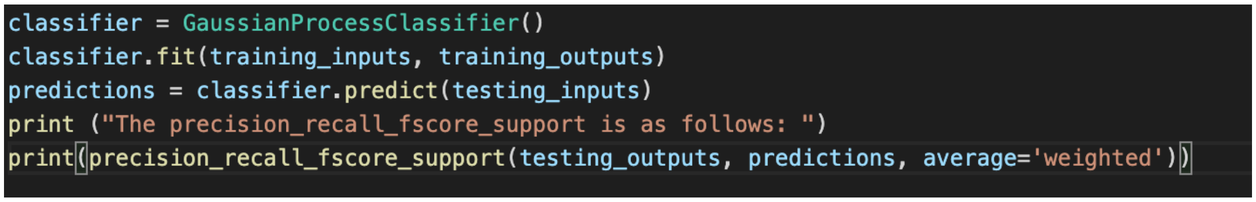

```
classifier = GaussianProcessClassifier()
classifier.fit(training_inputs, training_outputs)
predictions = classifier.predict(testing_inputs)
print ("The precision_recall_fscore_support is as follows: ")
print(precision_recall_fscore_support(testing_outputs, predictions, average='weighted'))
```

Fig 19

In Fig 19, we different metrics which are Precision, Recall, F1score, Support on our dataset and predicted outcomes of the testing sets as shown.

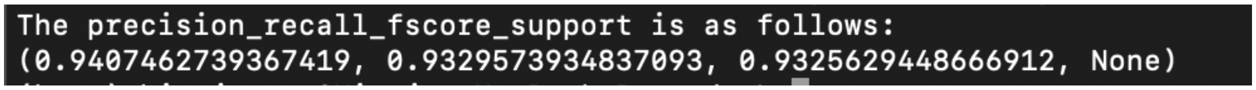

```
The precision_recall_fscore_support is as follows:
(0.9407462739367419, 0.9329573934837093, 0.9325629448666912, None)
```

Fig 20

In Fig 20, we got our respective values for the metrics.

7.3.6 RBF Kernel

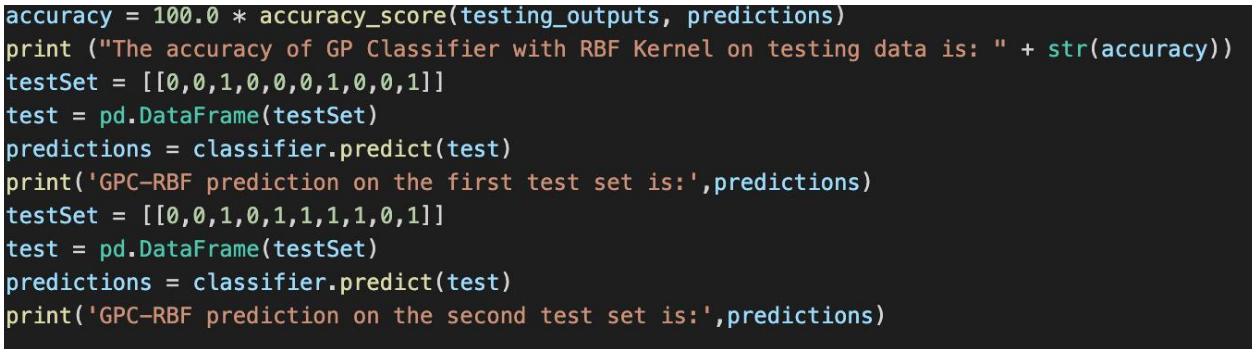

```
accuracy = 100.0 * accuracy_score(testing_outputs, predictions)
print ("The accuracy of GP Classifier with RBF Kernel on testing data is: " + str(accuracy))
testSet = [[0,0,1,0,0,0,1,0,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('GPC-RBF prediction on the first test set is:',predictions)
testSet = [[0,0,1,0,1,1,1,1,0,1]]
test = pd.DataFrame(testSet)
predictions = classifier.predict(test)
print('GPC-RBF prediction on the second test set is:',predictions)
```

Fig 21

In Fig 21, we implemented RBF Kernel under Gaussian Process Classifier on our dataset and predicted outcomes of the testing sets as shown. We made the Kernel work under 100% efficiency.

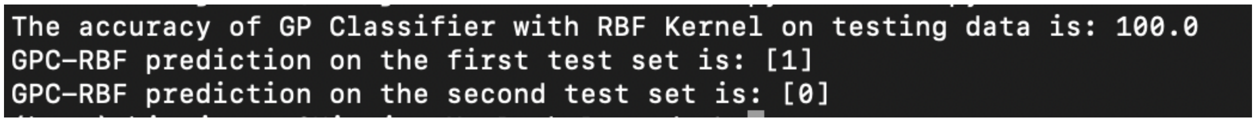

```
The accuracy of GP Classifier with RBF Kernel on testing data is: 100.0
GPC-RBF prediction on the first test set is: [1]
GPC-RBF prediction on the second test set is: [0]
```

Fig 22

In Fig 22, we got our predictions and the accuracy of the kernel is 100%.

7.3.7 Confusion Matrix

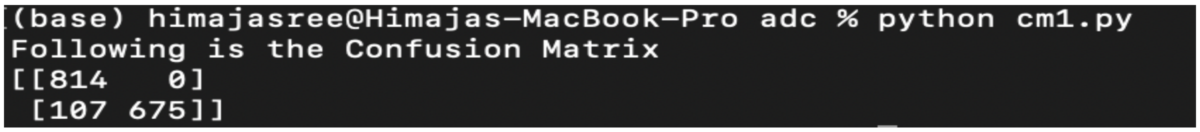

```
(base) himajasree@Himajas-MacBook-Pro adc % python cm1.py
Following is the Confusion Matrix
[[814   0]
 [107 675]]
```

Fig 23

Fig 23 represents the Confusion Matrix of our outcome ‘Effective’ and ‘InEffective’ data.

8. RESULT ANALYSIS

8.1 Training vs Testing Data Plot

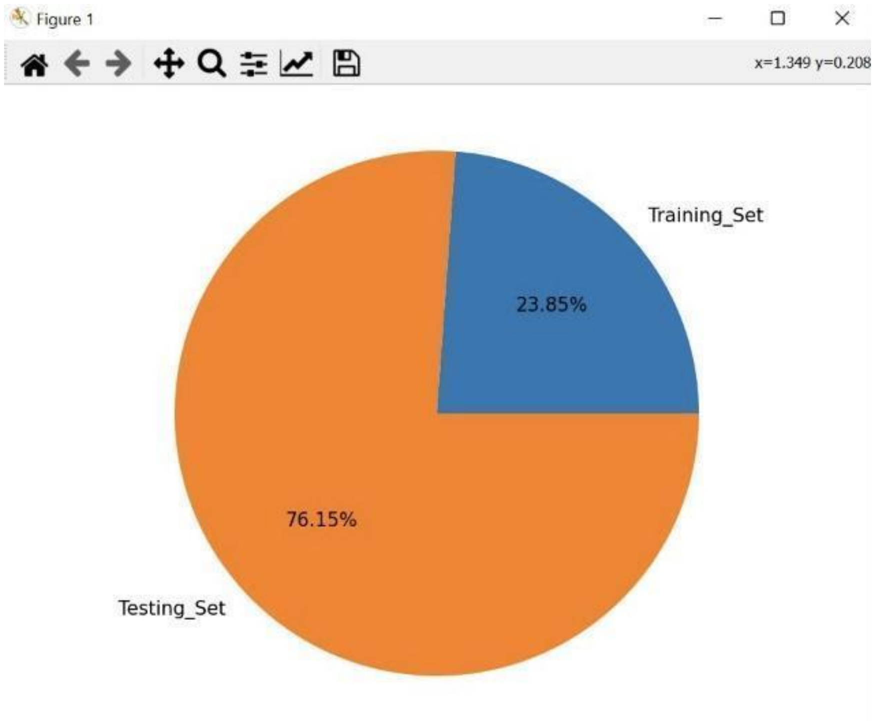


Fig 24

In Fig 24 we are plotting Training vs testing data which we have taken as a pie chart to compare them.

8.2 Comparison plot of the Metrics

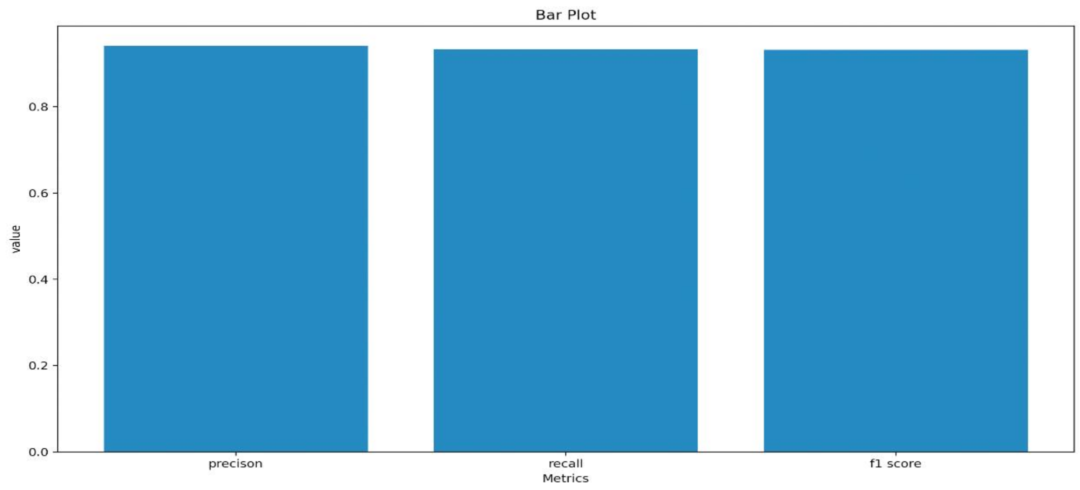


Fig 25

From our testing we got the outcome as following:

Precision- 0.9407

Recall- 0.93295

F1score- 0.9325

Support- None

In Fig 25, we plotted a graph between precision, recall and f1score. In the x-axis we have the metrics and, in the y-axis, we have the values range.

8.3 Comparison plot of the Accuracies

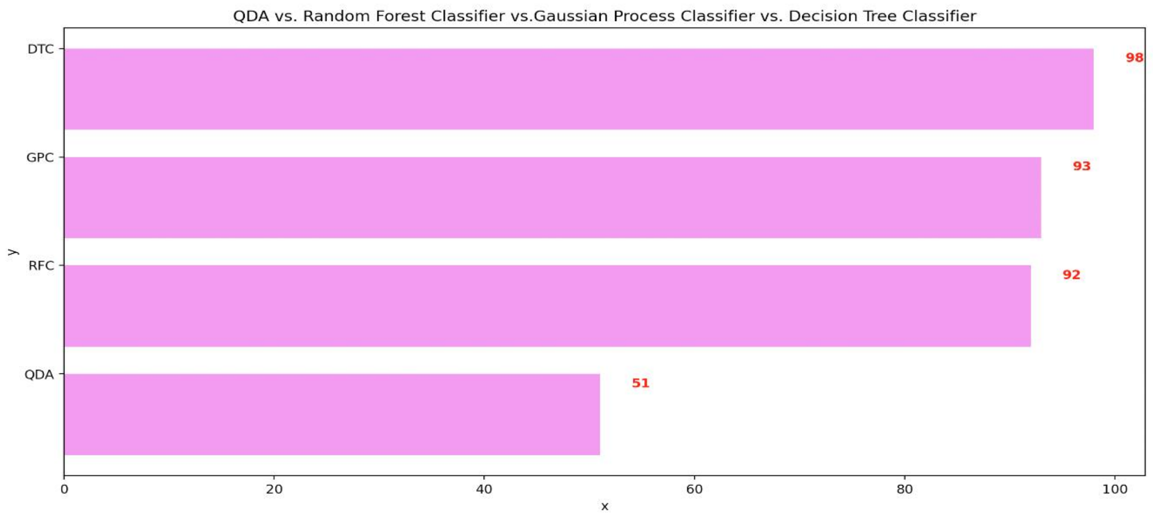


Fig 26

In Fig 26, we can see the comparison of all the accuracies we got. From our testing cases we got **Gaussian process classifier accuracy: 93%, Decision Tree Classifier: 98%, Random Forest Classifier: 92%, Quadratic Discriminant Analysis: 51%.**

From this observation we can say that, for our dataset **Decision Tree Classifier is more suitable**.

9. CONCLUSION & FUTURE SCOPE

CONCLUSION:

This project entitled **“Algorithmic management in a work context by predicting the effectiveness of the work centers using Machine Learning”** is useful to the work center management to determine the effectiveness of algorithmic management. The project is useful to the data analysts to understand more about quadratic discriminant analysis and Gaussian process analysis. The project finally leads to the improvement of the effectiveness of the work center.

FUTURE SCOPE:

As of now, the project is implemented by using machine learning techniques. It can be further enhanced by using deep learning techniques.

10. REFERENCES

[1] Mohammad Hossein Jarrahi, Gemma Newlands, Min Kyung Lee, Christine T. Wolf, Eliscia Kinder and Will Sutherland: Algorithmic management in a work context, Big Data & Society

[2]] Abraham M, Niessen C, Schnabel C, et al.Electronic monitoring at work: The role of attitudes, functions, and perceived control for the acceptance of tracking technologies. Human Resource Management Journal 29(4):657–675.

[3] Bader V and Kaiser S: Algorithmic decision-making? The user interface and its role for human involvement in decisions supported by artificial intelligence. Organization 26(5): 655–672.

[4] https://www.python.org/

[5]https://github.com/baoboa/pyqt5/blob/master/pyuic/uic/pyuic.py

[6] https://www.numpy.org/

[7] https://riverbankcomputing.com/software/pyqt/intro