**ABSTRACT**

Since student performance and pass rates in school reflect teaching level of the school and even all education system, it is critical to improve student pass rates and reduce dropout rates. Decision Tree (DT) algorithm and Support Vector Machine(SVM) algorithm in data mining, have been used by researchers to find important student features and predict the student pass rates, however they did not consider the coefficient of initialization, and whether there is a dependency between student features. Therefore, in this study, we propose a new concept: features dependencies, and use the grid search algorithm to optimize DT and SVM, in order to improve the accuracy of the algorithm. Furthermore, we added 10-fold cross-validation to DT and SVM algorithm. The results show the experiment can achieve better results in this work. The purpose of this study is providing assistance to students who have greater difficulties in their studies, and students who are at risk of graduating through data mining techniques.

**Algorithms:**

* Decision Tree (DT) algorithm and
* Support Vector Machine (SVM) algorithm

**1. INTRODUCTION**

The most important application of educational data mining technology is to early warn students about their academic performance and even find the impact of different features on the students. The establishment of a high-accuracy prediction of student achievement or behavior is useful for all educational institutions, because as soon as possible to identify the high failure rate of student learning, poor student achievement and low learning efficiency of students can improve the quality of teaching. Cortez and Silva used the data mining algorithm to model the two data sets (mathematics and Portuguese language) of Portuguese students, mainly for two questions: Is it possible for us to predict student performance in Portuguese school using data mining algorithm? What are the important features that affect student achievement? Among them, the DT algorithm model and the NV algorithm model are higher accurate to predict student behavior. In addition, several factors that are most relevant to student achievement are identified by random forest algorithms. Random forest results show that failure, absences, first period grade and second period grade have a great impact on student achievement.

Up to now, people are still seeking for the key features that affect the student pass rates, and trying to find ways to improve the accuracy of the student pass rates prediction. So, in this study, we used the DT and SVM algorithms optimized by the grid search algorithm to classify Portuguese students into two categories (pass / fail), in order to find the important features that affect student pass rates and predict the student pass rates. In particular, due to the existence of a certain dependencies between student features, we introduce features dependencies and expert guidance in calculating the relevance of student features. In order to initialize the coefficients and make the algorithm converge faster, we propose the initialization coefficient rule. Furthermore, we applied DT and SVM algorithms using 10-fold cross-validation. In this paper, the data set consists of two Comma Separated Values (CSV) files from data of Portuguese students about Mathematics and Portuguese lessons. Furthermore, results show that the method used in the study is effective in predicting student pass rates and can find important features that affect student’ performance.

**2. LITERATURE SURVEY**

**Survey on Data Mining System for Predicting University Students’ Graduation Grades Using ID3 Decision Tree Algorithm:** The desire of every organization is to extract hidden but useful knowledge from their data through data mining tools. Also, the recent decline in the standard of education in most developing countries has necessitated researches that will help proffer solutions to some of the problems. From the literature, different analysis has been carried out on university data, which includes student’s university entrance examination and Ordinary level results but the relationship between these entry results and students’ final graduation grades has been in isolation. Therefore, in this work, a new system that will predict students’ graduation grades based on entry results data using the Iterative Dichotomiser 3 (ID3) decision tree algorithm was developed. ID3 decision tree algorithm was used to train the data of the graduated sets. The knowledge represented by decision trees were extracted and presented in form of IF-THEN rules. The trained data were then used to develop a model for making future prediction of students’ graduation grades. The developed system could be very useful in predicting students’ final graduation grades even from the point of entry into the university. This will help management staff, academic planners to properly counsel students in order to improve their overall performance.

**Survey on Predicting Student Performance in Distance Higher Education Using Active Learning:**

In this era of computerization, education has also revamped itself and is not limited to old lecture method. The regular quest is on to find out new ways to make it more effective and efficient for students. Nowadays, lots of data is collected in educational databases, but it remains unutilized. In order to get required benefits from such a big data, powerful tools are required. Data mining is an emerging powerful tool for analysis and prediction. It is successfully applied in the area of fraud detection, advertising, marketing, loan assessment and prediction. But, it is in nascent stage in the field of education. Considerable amount of work is done in this direction, but still there are many untouched areas. Moreover, there is no unified approach among these researches. This paper presents a comprehensive survey, a travelogue (2002- 2014) towards educational data mining and its scope in future.

**Survey on Choosing the Kernel parameters of Support Vector Machines According to the Inter-cluster Distance:**

This paper proposes using the inter-cluster distance between class means in the feature space to help choose parameters for a kernel function when training a support vector machine (SVM). With the proposed method, the square values of the distance between the two class means of the training data in different feature spaces are calculated. These values are used as the indexes of data separation in the feature space. The experiment results show that the proposed method can choose the parameters close to the best ones. As a result, fewer possible values of the kernel parameters are required to be tested when training an SVM, and thus the training time of total training process can be significantly shortened.

**Survey on Drop out characteristics of Student Data for Academic Performance Using Decision Tree Techniques:**

Students’ academic performance is critical for educational institutions because strategic programs can be planned in improving or maintaining students’ performance during their period of studies in the institutions. The academic performance in this study is measured by their cumulative grade point average (CGPA) upon graduating. This study presents the work of data mining in predicting the drop out feature of students. This study applies decision tree technique to choose the best prediction and analysis. The list of students who are predicted as likely to drop out from college by data mining is then turned over to teachers and management for direct or indirect intervention.

**3. SYSTEM ANALYSIS**

**Existing System:**

In the education, data mining technology is mainly used for student data analysis, auxiliary teachers and managers to make right decisions. Data mining includes many algorithms, such as DT, association rules, SVM, clustering analysis, and so on. However, compared with other data mining algorithms, the DT is easier to be understood and can be constructed a decision tree for data sets with many attributes, and SVM can solve nonlinear problems and solve high-dimensional problems.

**Proposed System:**

In this paper, we use the classification algorithm in data mining technology, also known as supervised learning DT algorithm and SVM algorithm. Particularly, because the grid search algorithm is a method to optimize the model performance by traversing a given combination of parameters, the algorithm can be used to optimize the DT and SVM algorithms to improve the accuracy of the algorithm. In addition, because different student features have different effects on student pass rates and there is some dependency between the various features, we introduce the features dependences and expert guidance.

**Feasibility Study**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* Economical Feasibility
* Technical Feasibility
* Social Feasibility

**Economical Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**Social Feasibility**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**System Configuration**

**Hardware Requirements:**

Processer                     :           Any Update Processer

Ram                             :           Min 4 GB

Hard Disk                   :           Min 100 GB

**Software Requirements:**

Operating System       :           Windows family

Technology                 :           Python 3.6

IDE : PyCharm

**Design of Proposed System:**

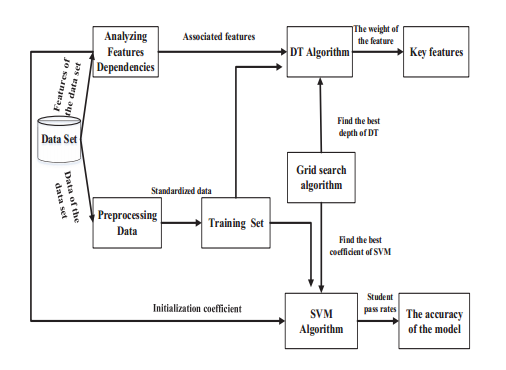
****

Fig. 1. System Model.

In the education, data mining technology is mainly used for student data analysis, auxiliary teachers and managers to make right decisions. Data mining includes many algorithms, such as DT, association rules, SVM, clustering analysis, and so on. However, compared with other data mining algorithms, the DT is easier to be understood and can be constructed a decision tree for data sets with many attributes, and SVM can solve nonlinear problems and solve high-dimensional problems. This work was supported by 2017WJ033, Southwest University 2017 Annual Network and Continuing Education Educational Research Project: Online Education Teacher Evaluation Based on Big Data Analysis. In particular, the overall algorithm flowchart is shown in Figure 1.

**4. SYSTEM IMPLEMENTATION**

**Preprocessing Student Data:**

In the real world, there is much potentially valuable information in the field of education. So the data must be standardized and preprocessed before we use the data mining to dig information. The datasets used in this paper can be downloaded from the machine learning website. In this section, we mainly elaborate on how to deal with different types of data, such as numeric, nominal, binary and etc., mainly dealing with nominal types.

The nominal types of data were transformed into a 1-of-C encoding and all features were standardized to a zero mean and one standard deviation. This means that instead of normalizing between 0 and 1 each category is given its own slot. For example, there are five types of work, teachers, doctor, civil servants (such as administrative or police), at home or other. If the mother is a teacher, you can use the binary array as [1, 0, 0, 0, 0], if the mother’s work is a doctor, you can use the binary array as [0, 1, 0, 0, 0]. Compared with the nominal type, we will directly use the numerical data and binary type of data translated into 1-0.

**Decision Tree method:**

In the field of education, the student pass rates represent the teaching quality of the school to a certain extent, and even the teaching level of the whole educational institution. Different student features have impact on student pass rates in different degrees, and it is important to find key features that affect student pass rates. The index used to measure degree of impurity is Entropy. So the information entropy in the DT algorithm is often used to calculate the weight of student features. DT is a branch structure composed of rules. Leaf nodes represent the results of classification; the root node represents the key features. The process of constructing a branch structure in a DT is called recursive partitioning. This branch structure can be expressed as IF-THEN form, easily understood by human. Since the data set contains features attributes and tag attributes, we can use classes to distinguish.

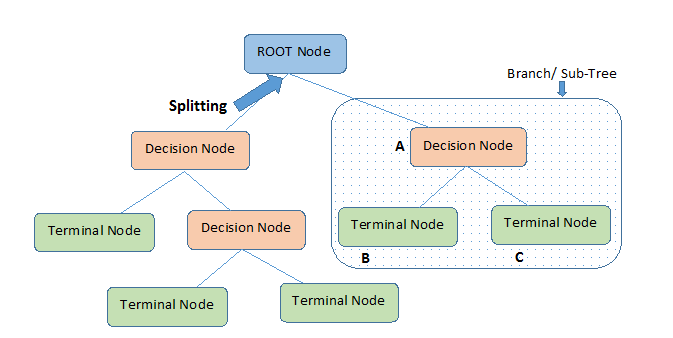


Figure 2: Decision Tree

**Support Vector Machine method:**

For this study, compared with other algorithms, support vector machines are very advantageous for binary classification problems, so we use the SVM algorithm to calculate the student pass rates and the accuracy of the model. SVM is a nonlinear function of supervised learning. The biggest advantage of the SVM algorithm is that when solving the problem, it is irrelevant to the dimension of the sample, and the small sample can still maintain a strong adaptability to the new sample.

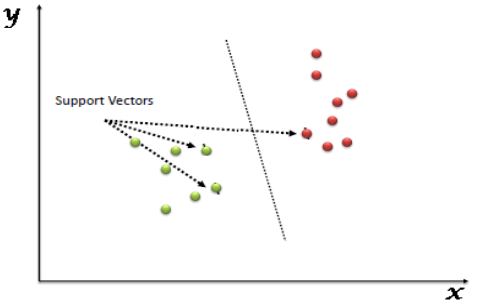
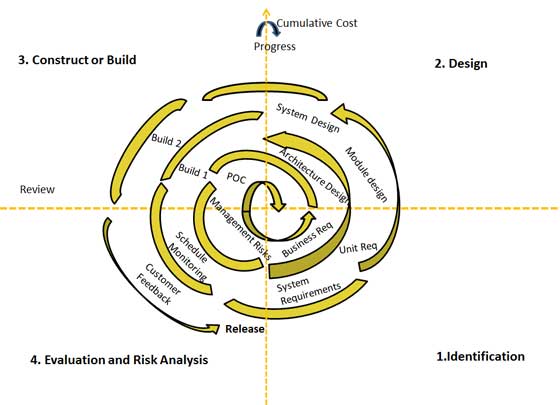


Figure 3: Support Vector Machine

**Process Model Used With Justification**

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**SDLC (Spiral Model):**

****

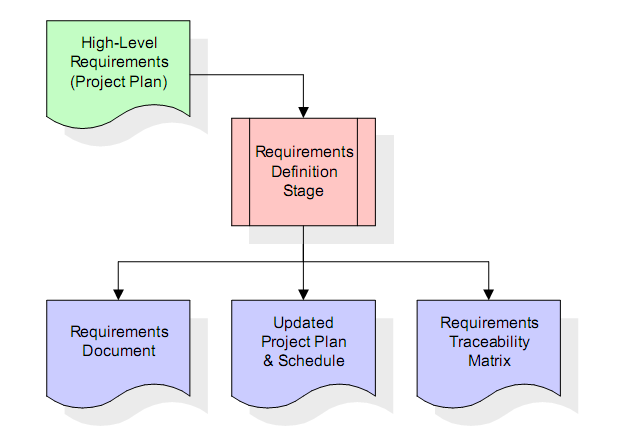
**Stages of SDLC:**

Requirement Gathering and Analysis

* Designing
* Coding
* Testing
* Deployment

**Requirements Definition Stage and Analysis:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.

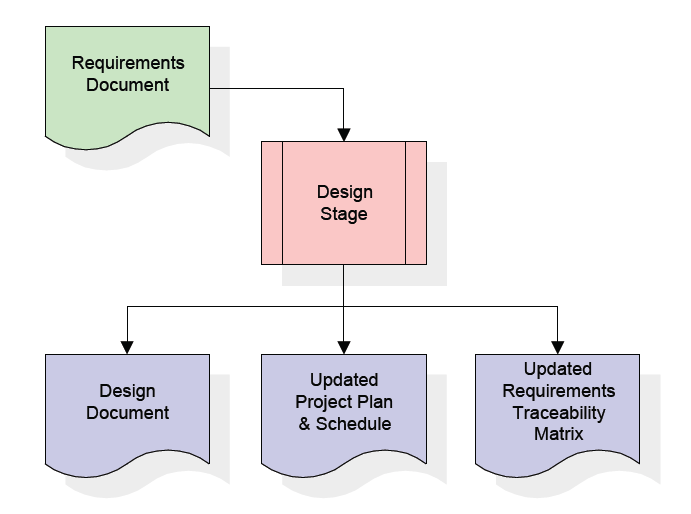


These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). the requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document. The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term *requirements traceability*. The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

**Design Stage:**

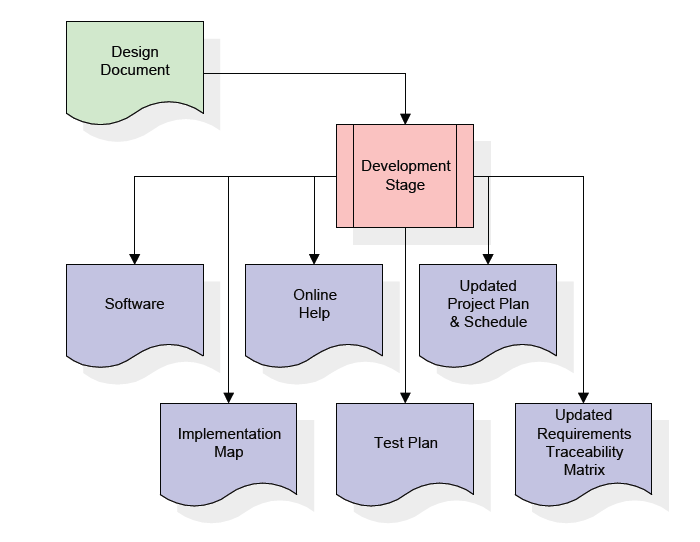
The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.



When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.

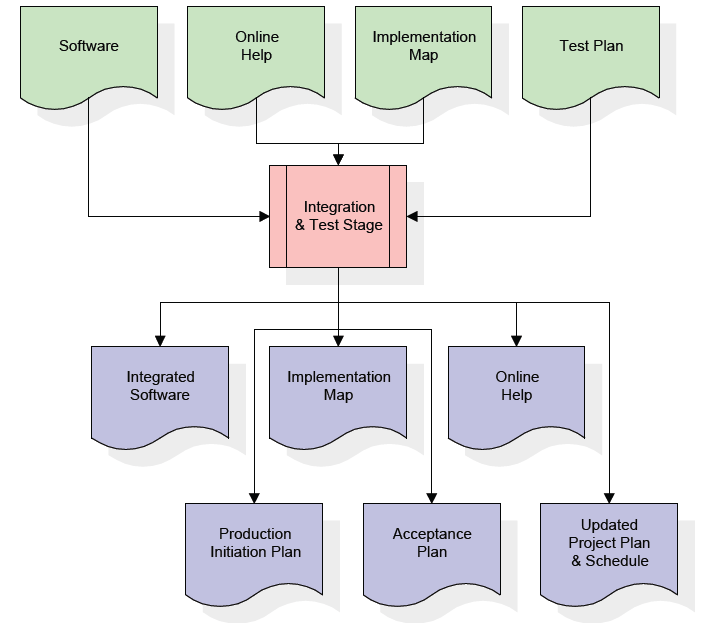


The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability.

During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.

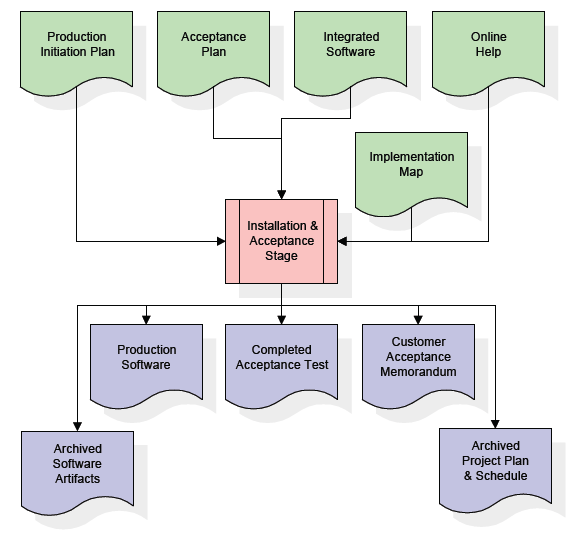


The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

**Installation & Acceptance Stage**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**5. SOFTWARE OVER VIEW**

**History of Python**

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

**Input as CSV File**

Reading data from CSV(comma separated values) is a fundamental necessity in Data Science. Often, we get data from various sources which can get exported to CSV format so that they can be used by other systems. The Panadas library provides features using which we can read the CSV file in full as well as in parts for only a selected group of columns and rows.

The CSV file is a text file in which the values in the columns are separated by a comma. Let's consider the following data present in the file named input.csv. You can create this file using windows notepad by copying and pasting this data. Save the file as input.csv using the save As All files(\*.\*) option in notepad.

import pandas as pd

data = pd.read\_csv('path/input.csv')

print (data)

**Operations using NumPy**

NumPy is a Python package which stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Using NumPy, a developer can perform the following operations −

* Mathematical and logical operations on arrays.
* Fourier transforms and routines for shape manipulation.
* Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

## Key Features of Pandas

* Fast and efficient DataFrame object with default and customized indexing.
* Tools for loading data into in-memory data objects from different file formats.
* Data alignment and integrated handling of missing data.
* Reshaping and pivoting of date sets.
* Label-based slicing, indexing and subsetting of large data sets.
* Columns from a data structure can be deleted or inserted.
* Group by data for aggregation and transformations.
* High performance merging and joining of data.
* Time Series functionality.

**Sample Code:**

**Predict\_DT.py:**

**from** PyQt5 **import** QtCore, QtGui, QtWidgets  
**from** sklearn.tree **import** DecisionTreeClassifier*# DecisionTree Classifier model***import** pandas **as** pd  
**from** sklearn.preprocessing **import** LabelEncoder  
  
**class** PredictDT(object):  
  
 **def** browse\_file(self):  
 fileName, \_ = QtWidgets.QFileDialog.getOpenFileName(**None**, **"Select File"**)  
 print(fileName)  
 self.lineEdit.setText(fileName)  
  
 **def** browse\_file1(self):  
 fileName1, \_ = QtWidgets.QFileDialog.getOpenFileName(**None**, **"Select File"**)  
 print(fileName1)  
 self.lineEdit\_2.setText(fileName1)  
  
 **def** prediction(self):  
 **try**:  
 training\_dataset = self.lineEdit.text()  
 testing\_dataset = self.lineEdit\_2.text()  
 **if** training\_dataset == **"" or** training\_dataset == **"null" or** testing\_dataset == **"" or** testing\_dataset == **"null"**:  
 self.showMessageBox(**"Information"**, **"Please fill out all fields"**)  
 **else**:  
 df = pd.read\_csv(training\_dataset, sep=**";"**)  
 df1 = pd.read\_csv(testing\_dataset, sep=**";"**)  
  
 print(**"\nStudent Performance Prediction"**)  
  
 *# Training Dataset  
 # For each feature, encode to categorical values* class\_le = LabelEncoder()  
 **for** column **in** df[  
 [**"school"**, **"sex"**, **"address"**, **"famsize"**, **"Pstatus"**, **"Mjob"**, **"Fjob"**, **"reason"**, **"guardian"**,  
 **"schoolsup"**, **"famsup"**, **"paid"**, **"activities"**, **"nursery"**, **"higher"**, **"internet"**, **"romantic"**]].columns:  
 df[column] = class\_le.fit\_transform(df[column].values)  
  
  
 *# Encode G1, G2, G3 as pass or fail binary values* **for** i, row **in** df.iterrows():  
 **if** row[**"G1"**] >= 10:  
 df[**"G1"**][i] = 1  
 **else**:  
 df[**"G1"**][i] = 0  
  
 **if** row[**"G2"**] >= 10:  
 df[**"G2"**][i] = 1  
 **else**:  
 df[**"G2"**][i] = 0  
  
 **if** row[**"G3"**] >= 10:  
 df[**"G3"**][i] = 1  
 **else**:  
 df[**"G3"**][i] = 0  
  
 *# Target values are G3* y = df.pop(**"G3"**)  
 *# print("Y=",y)  
 # Feature set is remaining features* X = df  
  
  
 *# Testing Dataset* **for** column **in** df1[  
 [**"school"**, **"sex"**, **"address"**, **"famsize"**, **"Pstatus"**, **"Mjob"**, **"Fjob"**, **"reason"**, **"guardian"**,  
 **"schoolsup"**, **"famsup"**,  
 **"paid"**, **"activities"**, **"nursery"**, **"higher"**, **"internet"**, **"romantic"**]].columns:  
 df1[column] = class\_le.fit\_transform(df1[column].values)  
  
 *# Encode G1, G2, G3 as pass or fail binary values* **for** i, row **in** df1.iterrows():  
 **if** row[**"G1"**] >= 10:  
 df1[**"G1"**][i] = 1  
 **else**:  
 df1[**"G1"**][i] = 0  
  
 **if** row[**"G2"**] >= 10:  
 df1[**"G2"**][i] = 1  
 **else**:  
 df1[**"G2"**][i] = 0  
  
 **if** row[**"G3"**] >= 10:  
 df1[**"G3"**][i] = 1  
 **else**:  
 df1[**"G3"**][i] = 0  
  
 *# Target values are G3* y1 = df1.pop(**"G3"**)  
  
 *# Feature set is remaining features* X1 = df1  
 clf\_dt = DecisionTreeClassifier()  
 clf\_dt.fit(X, y)  
 res = clf\_dt.predict(X1)  
 print(**"res="**, res)  
 **if**(res[0]==1):  
 self.showMessageBox(**"Result"**, **"Prediction Result is : PASS "**)  
 **else**:  
 self.showMessageBox(**"Result"**, **"Prediction Result is : FAIL "**)  
  
 **except** Exception **as** e:  
 print(**"Error="** + e.args[0])  
 tb = sys.exc\_info()[2]  
 print(tb.tb\_lineno)  
 print(e)  
  
 **def** showMessageBox(self, title, message):  
 msgBox = QtWidgets.QMessageBox()  
 msgBox.setIcon(QtWidgets.QMessageBox.Information)  
 msgBox.setWindowTitle(title)  
 msgBox.setText(message)  
 msgBox.setStandardButtons(QtWidgets.QMessageBox.Ok)  
 msgBox.exec\_()  
  
 **def** setupUi(self, Dialog):  
 Dialog.setObjectName(**"Dialog"**)  
 Dialog.resize(712, 535)  
 Dialog.setStyleSheet(**"background-color: rgb(85, 85, 127);"**)  
 self.label = QtWidgets.QLabel(Dialog)  
 self.label.setGeometry(QtCore.QRect(260, 60, 421, 61))  
 self.label.setStyleSheet(**"color: rgb(255, 255, 255);\n"  
"font: 18pt \"Georgia\";"**)  
 self.label.setObjectName(**"label"**)  
 self.label\_2 = QtWidgets.QLabel(Dialog)  
 self.label\_2.setGeometry(QtCore.QRect(150, 150, 191, 41))  
 self.label\_2.setStyleSheet(**"color: rgb(255, 255, 255);\n"  
"font: 14pt \"Georgia\";"**)  
 self.label\_2.setObjectName(**"label\_2"**)  
 self.lineEdit = QtWidgets.QLineEdit(Dialog)  
 self.lineEdit.setGeometry(QtCore.QRect(150, 200, 341, 41))  
 self.lineEdit.setStyleSheet(**"font: 14pt \"Times New Roman\";"**)  
 self.lineEdit.setObjectName(**"lineEdit"**)  
 self.pushButton = QtWidgets.QPushButton(Dialog)  
 self.pushButton.setGeometry(QtCore.QRect(530, 200, 121, 41))  
 self.pushButton.setStyleSheet(**"color: rgb(0, 85, 127);\n"  
"font: 14pt \"Times New Roman\";\n"  
"color: rgb(255, 255, 255);"**)  
 self.pushButton.setObjectName(**"pushButton"**)  
 self.pushButton.clicked.connect(self.browse\_file)  
 self.pushButton\_2 = QtWidgets.QPushButton(Dialog)  
 self.pushButton\_2.setGeometry(QtCore.QRect(260, 420, 131, 41))  
 self.pushButton\_2.setStyleSheet(**"font: 14pt \"Georgia\";\n"  
"background-color: rgb(85, 85, 255);\n"  
"color: rgb(255, 255, 255);"**)  
 self.pushButton\_2.setObjectName(**"pushButton\_2"**)  
 self.pushButton\_2.clicked.connect(self.prediction)  
 self.label\_3 = QtWidgets.QLabel(Dialog)  
 self.label\_3.setGeometry(QtCore.QRect(150, 280, 221, 51))  
 self.label\_3.setStyleSheet(**"color: rgb(255, 255, 255);\n"  
"font: 14pt \"Georgia\";"**)  
 self.label\_3.setObjectName(**"label\_3"**)  
 self.lineEdit\_2 = QtWidgets.QLineEdit(Dialog)  
 self.lineEdit\_2.setGeometry(QtCore.QRect(150, 340, 341, 41))  
 self.lineEdit\_2.setStyleSheet(**"font: 14pt \"Times New Roman\";"**)  
 self.lineEdit\_2.setObjectName(**"lineEdit\_2"**)  
 self.pushButton\_3 = QtWidgets.QPushButton(Dialog)  
 self.pushButton\_3.setGeometry(QtCore.QRect(530, 340, 121, 41))  
 self.pushButton\_3.setStyleSheet(**"color: rgb(0, 85, 127);\n"  
"font: 14pt \"Times New Roman\";\n"  
"color: rgb(255, 255, 255);"**)  
 self.pushButton\_3.setObjectName(**"pushButton\_3"**)  
 self.pushButton\_3.clicked.connect(self.browse\_file1)  
  
 self.retranslateUi(Dialog)  
 QtCore.QMetaObject.connectSlotsByName(Dialog)  
  
 **def** retranslateUi(self, Dialog):  
 \_translate = QtCore.QCoreApplication.translate  
 Dialog.setWindowTitle(\_translate(**"Dialog"**, **"Student Pass Rating Prediction"**))  
 self.label.setText(\_translate(**"Dialog"**, **"Decision Tree"**))  
 self.label\_2.setText(\_translate(**"Dialog"**, **"Load Training Dataset"**))  
 self.pushButton.setText(\_translate(**"Dialog"**, **"Browse"**))  
 self.pushButton\_2.setText(\_translate(**"Dialog"**, **"Predict"**))  
 self.label\_3.setText(\_translate(**"Dialog"**, **"Load Testing Dataset"**))  
 self.pushButton\_3.setText(\_translate(**"Dialog"**, **"Browse"**))  
  
  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 **import** sys  
 app = QtWidgets.QApplication(sys.argv)  
 Dialog = QtWidgets.QDialog()  
 ui = Ui\_Classifications()  
 ui.setupUi(Dialog)  
 Dialog.show()  
 sys.exit(app.exec\_())

**Predict\_SVM:**

**from** PyQt5 **import** QtCore, QtGui, QtWidgets  
**from** sklearn **import** svm *# Support Vector Machine Classifier model***import** pandas **as** pd  
**from** sklearn.preprocessing **import** LabelEncoder  
  
**class** PredictSVM(object):  
  
 **def** browse\_file(self):  
 fileName, \_ = QtWidgets.QFileDialog.getOpenFileName(**None**, **"Select File"**)  
 print(fileName)  
 self.lineEdit.setText(fileName)  
  
 **def** browse\_file1(self):  
 fileName1, \_ = QtWidgets.QFileDialog.getOpenFileName(**None**, **"Select File"**)  
 print(fileName1)  
 self.lineEdit\_2.setText(fileName1)  
  
 **def** prediction(self):  
 **try**:  
 training\_dataset = self.lineEdit.text()  
 testing\_dataset = self.lineEdit\_2.text()  
 **if** training\_dataset == **"" or** training\_dataset == **"null" or** testing\_dataset == **"" or** testing\_dataset == **"null"**:  
 self.showMessageBox(**"Information"**, **"Please fill out all fields"**)  
 **else**:  
 df = pd.read\_csv(training\_dataset, sep=**";"**)  
 df1 = pd.read\_csv(testing\_dataset, sep=**";"**)  
  
 print(**"\nStudent Performance Prediction"**)  
  
 *# Training Dataset  
 # For each feature, encode to categorical values* class\_le = LabelEncoder()  
 **for** column **in** df[  
 [**"school"**, **"sex"**, **"address"**, **"famsize"**, **"Pstatus"**, **"Mjob"**, **"Fjob"**, **"reason"**, **"guardian"**,  
 **"schoolsup"**, **"famsup"**, **"paid"**, **"activities"**, **"nursery"**, **"higher"**, **"internet"**, **"romantic"**]].columns:  
 df[column] = class\_le.fit\_transform(df[column].values)  
  
  
 *# Encode G1, G2, G3 as pass or fail binary values* **for** i, row **in** df.iterrows():  
 **if** row[**"G1"**] >= 10:  
 df[**"G1"**][i] = 1  
 **else**:  
 df[**"G1"**][i] = 0  
  
 **if** row[**"G2"**] >= 10:  
 df[**"G2"**][i] = 1  
 **else**:  
 df[**"G2"**][i] = 0  
  
 **if** row[**"G3"**] >= 10:  
 df[**"G3"**][i] = 1  
 **else**:  
 df[**"G3"**][i] = 0  
  
 *# Target values are G3* y = df.pop(**"G3"**)  
 *# print("Y=",y)  
 # Feature set is remaining features* X = df  
  
  
 *# Testing Dataset* **for** column **in** df1[  
 [**"school"**, **"sex"**, **"address"**, **"famsize"**, **"Pstatus"**, **"Mjob"**, **"Fjob"**, **"reason"**, **"guardian"**,  
 **"schoolsup"**, **"famsup"**,  
 **"paid"**, **"activities"**, **"nursery"**, **"higher"**, **"internet"**, **"romantic"**]].columns:  
 df1[column] = class\_le.fit\_transform(df1[column].values)  
  
 *# Encode G1, G2, G3 as pass or fail binary values* **for** i, row **in** df1.iterrows():  
 **if** row[**"G1"**] >= 10:  
 df1[**"G1"**][i] = 1  
 **else**:  
 df1[**"G1"**][i] = 0  
  
 **if** row[**"G2"**] >= 10:  
 df1[**"G2"**][i] = 1  
 **else**:  
 df1[**"G2"**][i] = 0  
  
 **if** row[**"G3"**] >= 10:  
 df1[**"G3"**][i] = 1  
 **else**:  
 df1[**"G3"**][i] = 0  
  
 *# Target values are G3* y1 = df1.pop(**"G3"**)  
  
 *# Feature set is remaining features* X1 = df1  
 clf = svm.SVC()  
 clf.fit(X, y)  
 res = clf.predict(X1)  
 print(**"res="**, res)  
 **if**(res[0]==1):  
 self.showMessageBox(**"Result"**, **"Prediction Result is : PASS "**)  
 **else**:  
 self.showMessageBox(**"Result"**, **"Prediction Result is : FAIL "**)  
  
  
 **except** Exception **as** e:  
 print(**"Error="** + e.args[0])  
 tb = sys.exc\_info()[2]  
 print(tb.tb\_lineno)  
 print(e)  
  
 **def** showMessageBox(self, title, message):  
 msgBox = QtWidgets.QMessageBox()  
 msgBox.setIcon(QtWidgets.QMessageBox.Information)  
 msgBox.setWindowTitle(title)  
 msgBox.setText(message)  
 msgBox.setStandardButtons(QtWidgets.QMessageBox.Ok)  
 msgBox.exec\_()  
  
 **def** setupUi(self, Dialog):  
 Dialog.setObjectName(**"Dialog"**)  
 Dialog.resize(712, 535)  
 Dialog.setStyleSheet(**"background-color: rgb(170, 85, 0);"**)  
 self.label = QtWidgets.QLabel(Dialog)  
 self.label.setGeometry(QtCore.QRect(260, 60, 421, 61))  
 self.label.setStyleSheet(**"color: rgb(255, 255, 255);\n"  
"font: 18pt \"Georgia\";"**)  
 self.label.setObjectName(**"label"**)  
 self.label\_2 = QtWidgets.QLabel(Dialog)  
 self.label\_2.setGeometry(QtCore.QRect(150, 150, 191, 41))  
 self.label\_2.setStyleSheet(**"color: rgb(255, 255, 255);\n"  
"font: 14pt \"Georgia\";"**)  
 self.label\_2.setObjectName(**"label\_2"**)  
 self.lineEdit = QtWidgets.QLineEdit(Dialog)  
 self.lineEdit.setGeometry(QtCore.QRect(150, 200, 341, 41))  
 self.lineEdit.setStyleSheet(**"font: 14pt \"Times New Roman\";"**)  
 self.lineEdit.setObjectName(**"lineEdit"**)  
 self.pushButton = QtWidgets.QPushButton(Dialog)  
 self.pushButton.setGeometry(QtCore.QRect(530, 200, 121, 41))  
 self.pushButton.setStyleSheet(**"color: rgb(0, 85, 127);\n"  
"font: 14pt \"Times New Roman\";\n"  
"color: rgb(255, 255, 255);"**)  
 self.pushButton.setObjectName(**"pushButton"**)  
 self.pushButton.clicked.connect(self.browse\_file)  
 self.pushButton\_2 = QtWidgets.QPushButton(Dialog)  
 self.pushButton\_2.setGeometry(QtCore.QRect(260, 420, 131, 41))  
 self.pushButton\_2.setStyleSheet(**"font: 14pt \"Georgia\";\n"  
"background-color: rgb(85, 85, 255);\n"  
"color: rgb(255, 255, 255);"**)  
 self.pushButton\_2.setObjectName(**"pushButton\_2"**)  
 self.pushButton\_2.clicked.connect(self.prediction)  
 self.label\_3 = QtWidgets.QLabel(Dialog)  
 self.label\_3.setGeometry(QtCore.QRect(150, 280, 221, 51))  
 self.label\_3.setStyleSheet(**"color: rgb(255, 255, 255);\n"  
"font: 14pt \"Georgia\";"**)  
 self.label\_3.setObjectName(**"label\_3"**)  
 self.lineEdit\_2 = QtWidgets.QLineEdit(Dialog)  
 self.lineEdit\_2.setGeometry(QtCore.QRect(150, 340, 341, 41))  
 self.lineEdit\_2.setStyleSheet(**"font: 14pt \"Times New Roman\";"**)  
 self.lineEdit\_2.setObjectName(**"lineEdit\_2"**)  
 self.pushButton\_3 = QtWidgets.QPushButton(Dialog)  
 self.pushButton\_3.setGeometry(QtCore.QRect(530, 340, 121, 41))  
 self.pushButton\_3.setStyleSheet(**"color: rgb(0, 85, 127);\n"  
"font: 14pt \"Times New Roman\";\n"  
"color: rgb(255, 255, 255);"**)  
 self.pushButton\_3.setObjectName(**"pushButton\_3"**)  
 self.pushButton\_3.clicked.connect(self.browse\_file1)  
  
 self.retranslateUi(Dialog)  
 QtCore.QMetaObject.connectSlotsByName(Dialog)  
  
 **def** retranslateUi(self, Dialog):  
 \_translate = QtCore.QCoreApplication.translate  
 Dialog.setWindowTitle(\_translate(**"Dialog"**, **"Student Pass Rating Prediction"**))  
 self.label.setText(\_translate(**"Dialog"**, **"Support Vector Machine"**))  
 self.label\_2.setText(\_translate(**"Dialog"**, **"Load Training Dataset"**))  
 self.pushButton.setText(\_translate(**"Dialog"**, **"Browse"**))  
 self.pushButton\_2.setText(\_translate(**"Dialog"**, **"Predict"**))  
 self.label\_3.setText(\_translate(**"Dialog"**, **"Load Testing Dataset"**))  
 self.pushButton\_3.setText(\_translate(**"Dialog"**, **"Browse"**))  
  
  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 **import** sys  
 app = QtWidgets.QApplication(sys.argv)  
 Dialog = QtWidgets.QDialog()  
 ui = Ui\_Classifications()  
 ui.setupUi(Dialog)  
 Dialog.show()  
 sys.exit(app.exec\_())

**6. TESTING**

Testing is the debugging program is one of the most critical aspects of the computer programming triggers, without programming that works, the system would never produce an output of which it was designed. Testing is best performed when user development is asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not quantity but quality of the data used the matters of testing. Testing is aimed at ensuring that the system was accurately an efficiently before live operation commands.

**Testing objectives:**

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

A successful test is one that uncovers an as yet undiscovered error.

A good test case is one that has probability of finding an error, if it exists.

The test is inadequate to detect possibly present errors.

The software more or less confirms to the quality and reliable standards.

**Levels of Testing:**

In order to uncover present in different phases we have the concept of levels of testing.

**The basic levels of Testing:**

**Code testing:**

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

**Specification Testing:**

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

**Unit testing:**

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

Each Module can be tested using the following two Strategies:

1. Black Box Testing
2. White Box Testing

**BLACK BOX TESTING**

**What is Black Box Testing?**

Black box testing is a software testing techniques in which **functionality of the software under test (SUT) is tested without looking at the internal code structure**, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

**In Black Box Testing we just focus on inputs and output of the software system** without bothering about internal knowledge of the software program.



The above Black Box can be any software system you want to test. For example : an operating system like Windows, a website like Google ,a database like Oracle or even your own custom application. Under Black Box Testing, you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

**Black box testing - Steps**

Here are the generic steps followed to carry out any type of Black Box Testing.

* Initially requirements and specifications of the system are examined.
* Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
* Tester determines expected outputs for all those inputs.
* Software tester constructs test cases with the selected inputs.
* The test cases are executed.
* Software tester compares the actual outputs with the expected outputs.
* Defects if any are fixed and re-tested.

**Types of Black Box Testing**

There are many types of Black Box Testing but following are the prominent ones -

* **Functional testing** – This black box testing type is related to functional requirements of a system; it is done by software testers.
* **Non-functional testing** – This type of black box testing is not related to testing of a specific functionality, but non-functional requirements  such as performance, scalability, usability.
* **Regression testing** – Regression testing is done  after code fixes , upgrades or any other system maintenance to check the new code has not affected the existing code.

**WHITE BOX TESTING**

White Box Testing is the testing of a software solution's internal coding and infrastructure.It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability.White box testing is also known as **clear, open, structural, and glass box testing**.

It is one of two parts of the **"box testing" approach** of software testing. Its counter-part, blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "whitebox" was used because of the see-through box concept. The clear box or whitebox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested

## What do you verify in White Box Testing ?

White box testing involves the testing of the software code for the following:

* Internal security holes
* Broken or poorly structured paths in the coding processes
* The flow of specific inputs through the code
* Expected output
* The functionality of conditional loops
* Testing of each statement, object and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

**How do you perform White Box Testing?**

To give you a simplified explanation of white box testing, we have divided it into **two basic steps**. This is what testers do when testing an application using the white box testing technique:

**STEP 1) UNDERSTAND THE SOURCE CODE**

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

**Step 2) CREATE TEST CASES AND EXECUTE**

The second basic step to white box testing involves testing the application’s source code for proper flow and structure. One way is by writing more code to test the application’s source code. The tester will develop little tests for each process or series of processes in the application. This method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include manual testing, trial and error testing and the use of testing tools as we will explain further on in this article.

**System testing:**

Once the individual module testing is completed, modules are assembled and integrated to perform as a system. The top down testing, which began from upper level to lower level module, was carried out to check whether the entire system is performing satisfactorily.

There are three main kinds of System testing:

1. Alpha Testing
2. Beta Testing
3. Acceptance Testing

**Alpha Testing:**

This refers to the system testing that is carried out by the test team with the Organization.

**Beta Testing**:

This refers to the system testing that is performed by a selected group of friendly customers

**Acceptance Testing:**

This refers to the system testing that is performed by the customer to determine whether or not to accept the delivery of the system.

**Integration Testing:**

Data can be lost across an interface, one module can have an adverse effort on the other sub functions, when combined, may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. The testing was done with sample data. The developed system has run successfully for this sample data. The need for integrated test is to find the overall system performance.

**Output testing:** After performance of the validation testing, the next step is output testing. The output displayed or generated by the system under consideration is tested by asking the user about the format required by system.

**TEST CASES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID #1 | | Test Case Description - Validations in Upload Dataset | | |
| Entering data in upload form | | | | |
| Step # | Step Details | Expected Results | Actual Results | Pass/Fail/Not Executed/Suspended |
| 1 | User select not matched file type (txt,pdf) | Uploaded successfully | Invalid file selections | Please select valid file type (.csv) |
| 2 | Submitting the form without select the file | Uploaded successfully | Not empty fields | Fill required fields |
| 3 | User select valid file | Uploaded successfully | Success | Uploaded  Successfully |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID #2 | | Test Case Description - Validations in Prediction Form | | |
| Entering data in Prediction form | | | | |
| Step # | Step Details | Expected Results | Actual Results | Pass/Fail/Not Executed/Suspended |
| 1 | User enters wrong input values | Predict Results | invalid input type | Number format  Exceptions |
| 2 | Submitting the form without entering any details | Predict Results | Enter Required  Input values | Fill required fields |
| 3 | User enters valid input values | Predict Results | Enter correct input type values | Result |

**7. SYSTEM DESIGN**

**UML DIAGRAMS**

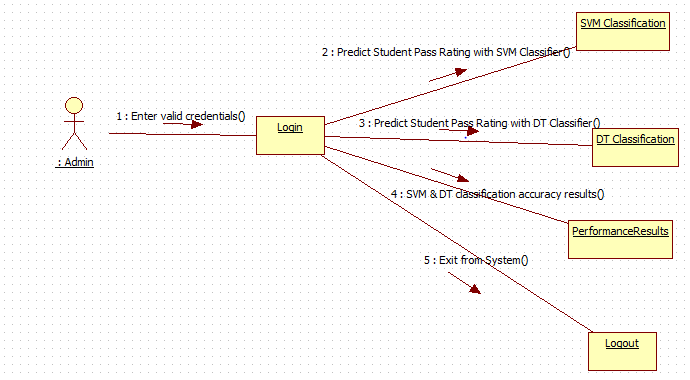
**Use Case**

****

**Sequence**

****

**Collaboration:**

****

**State Chart Diagram:**

****

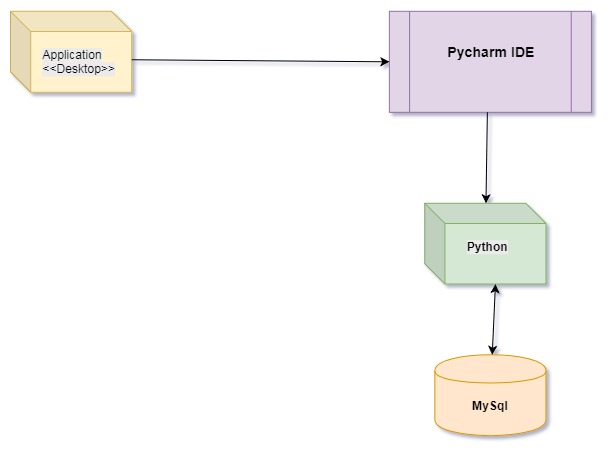
**Activity Diagram:**

****

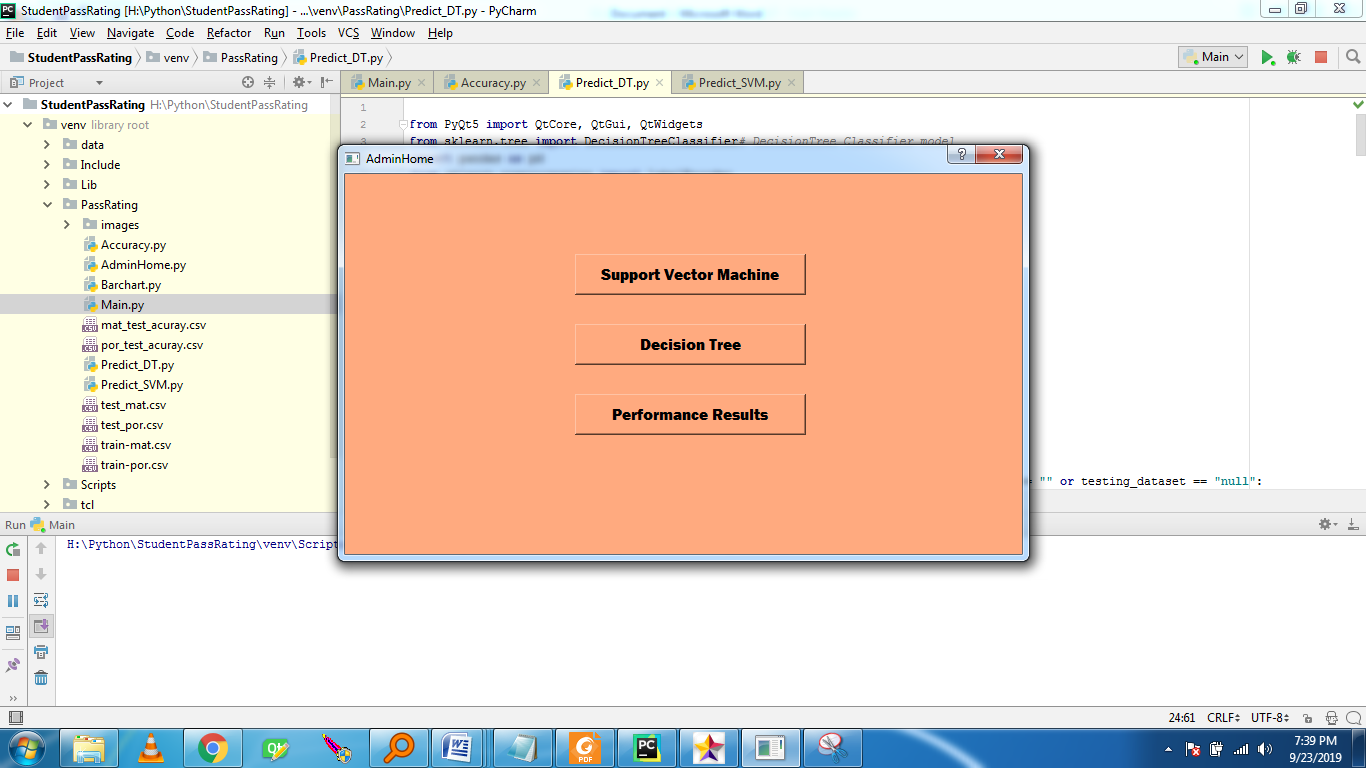
**Component Diagram:**

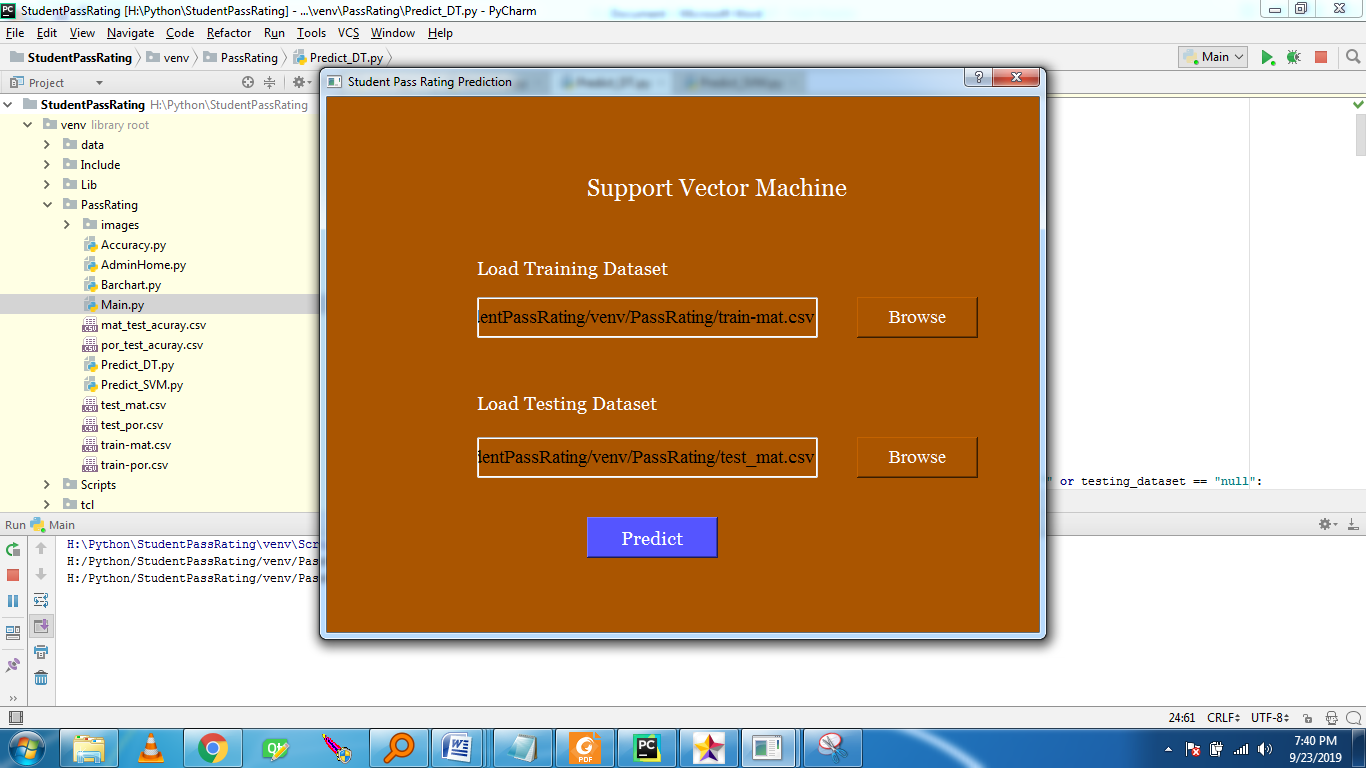
****

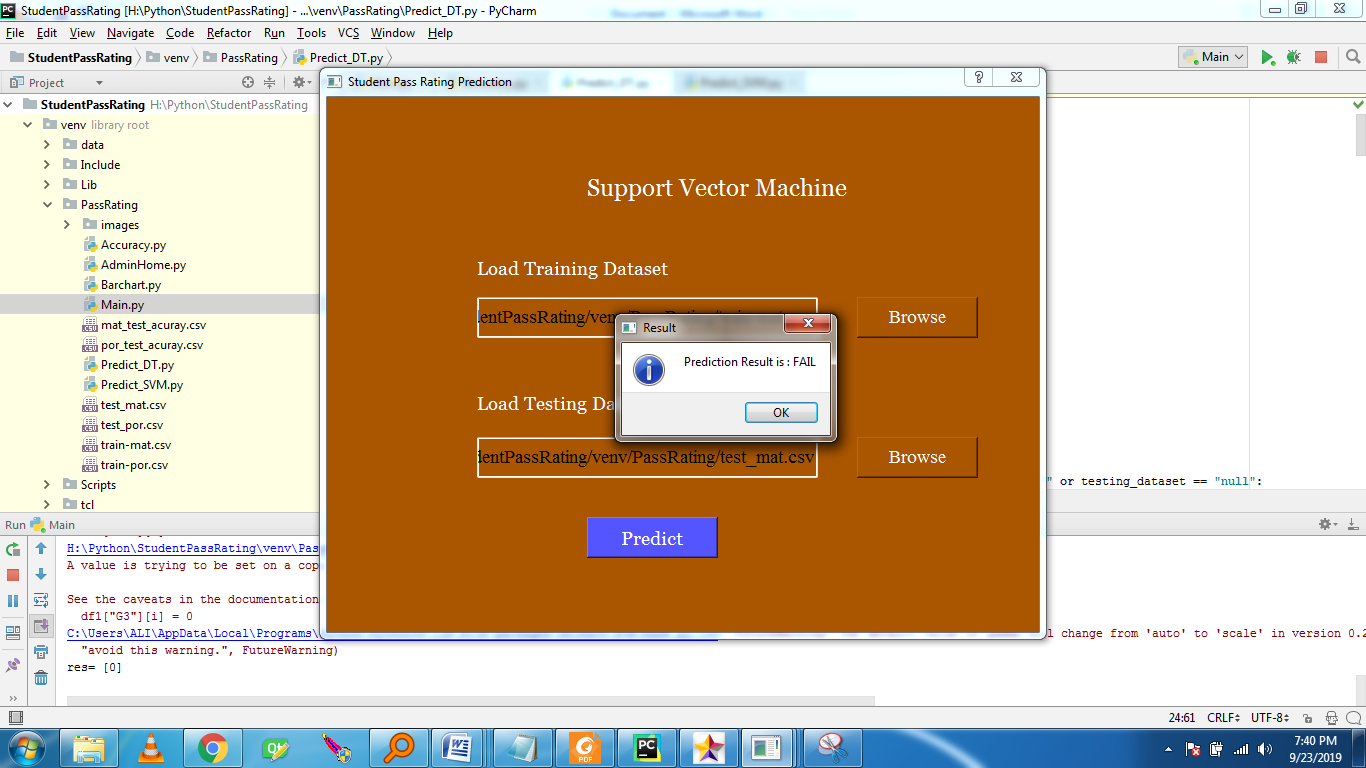
**Deployment Diagram:**

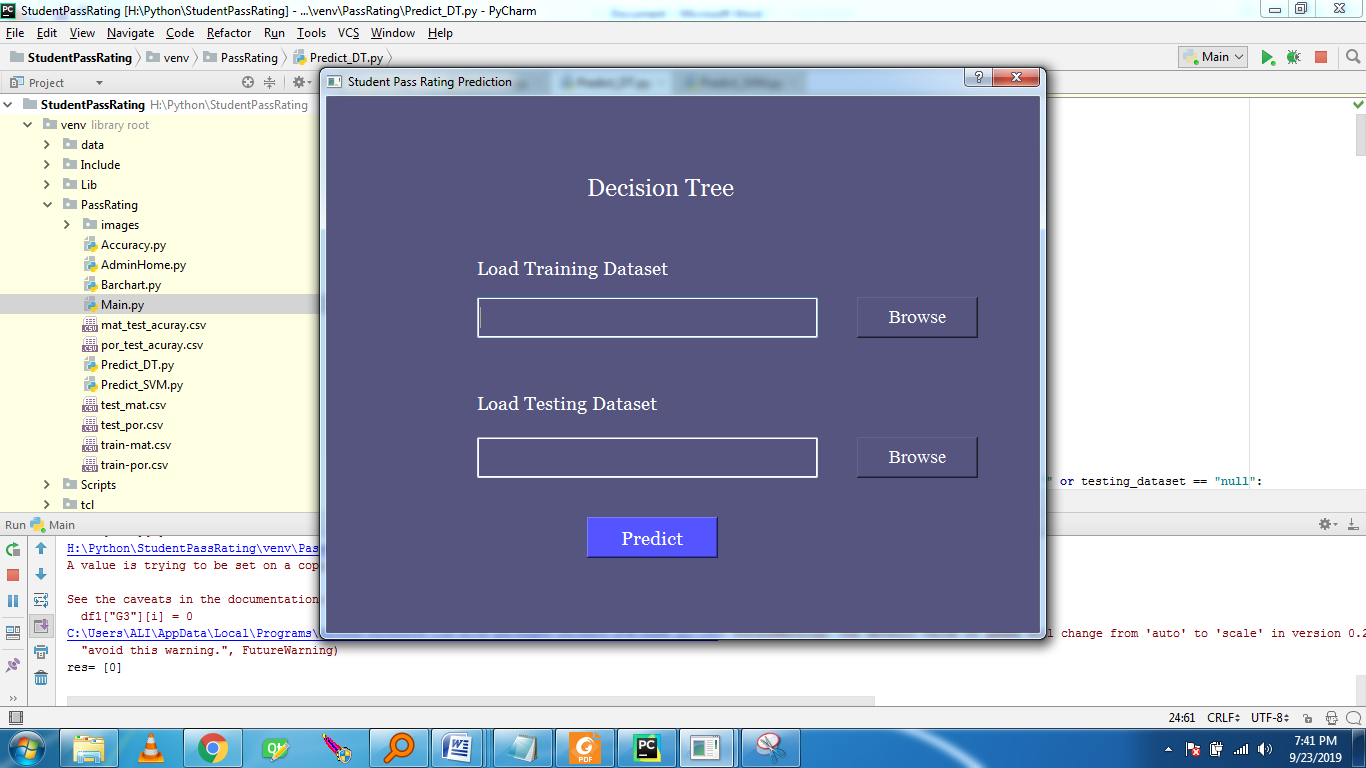
****

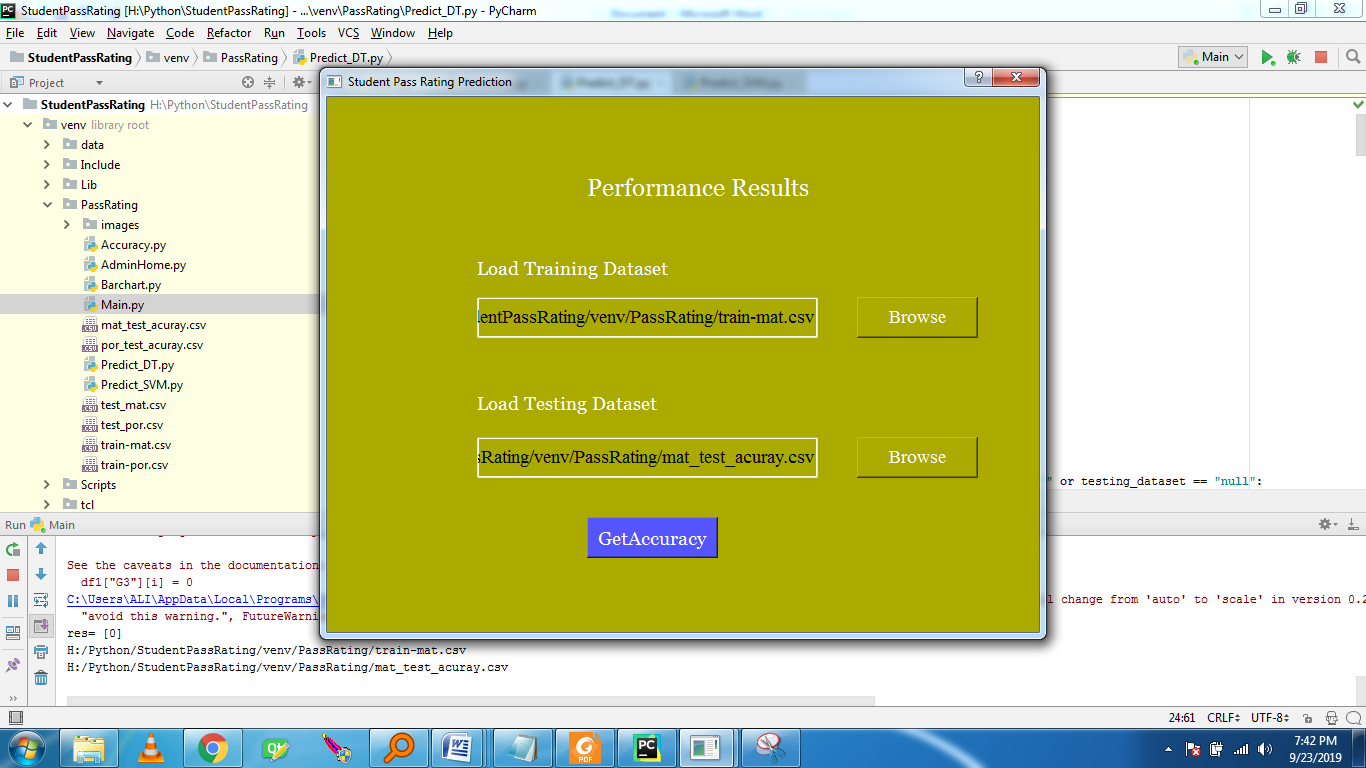
**8. SCREEN SHOTS**

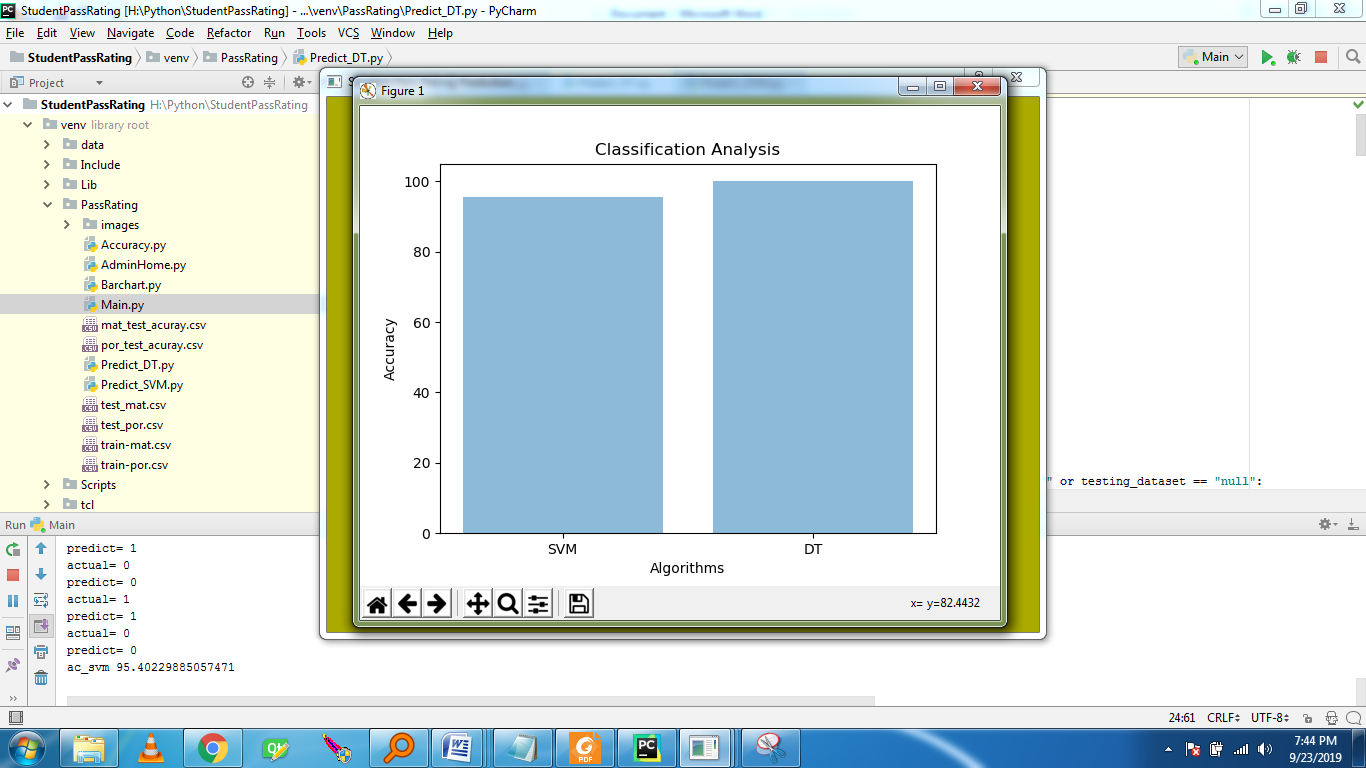
****

****

****

****

****

****

**9. CONCLUSION**

In the education field, many of the previous researchers have shown great interest in student pass rates or even finding important student features. However, they often overlook the links that exist between student features. Different from the previous study, in this study, we consider the dependency between student features, introducing features dependencies and expert guidance to initialize coefficient, so that the algorithm can be converged faster. In particular, in order to further determine the initial values of the coefficients, we propose the initialization coefficients rules. Furthermore, in order to improve the accuracy of the model, we use the SVM algorithm and the DT algorithm optimized by the grid search algorithm to predict the student pass rates and use the information gain to find out the features that have a great influence on the student performance. Experiments show that the algorithm has achieved good result, so we can use the model to identify the important features of students and student pass rates.

**10. REFERENCES**

[1] A.O.Ogunde, and D.A .Ajibade. Data Mining System for Predicting University Students’ Graduation Grades Using ID3 Decision Tree Algorithm. Journal of Computer Science and Information Technology, 2015.

[2] R.S.J.D.Baker. Data mining for education. International encyclopedia of education, 7, 112-118, 2010.

[3] R.S.J.D.Baker, and Kalina Yacef. The state of educational data mining in 2009: A review and future visions. JEDM-Journal of Educational Data Mining, 3-17, 2009.

[4] P.Cortez and A.Silva. Using Data Mining to Predict Secondary School Student Performance, 2008. In A. Brito and J. Teixeira Eds. Proceedings of 5th FUture BUsiness TEChnology Conference,5-12, Porto, Portugal, EUROSIS, ISBN 978-9077381-39-7, 2008.

[5] Mr.M.N.Quadri and Dr.N.V.Kalyankar. Drop Out characteristics of Student Data for Academic Performance Using Decision Tree Techniques. Global Journal of Computer Science and Technology, 2010.

[6] B. K. Bharadwaj and S. Pal. Data Mining: A prediction for performance improvement using classification. International Journal of Computer Science and Information Security (IJCSIS), 136-140, 2011.

[7] Surjeet Kumar Yadav and Saurabh Pal. Data Mining: A Prediction for Performance Improvement of Engineering Students using Classification. World of Computer Science and Information Technology Journal (WCSIT), 2012.

[8] Alaa Khalaf Hamoud. Selection of Best Decision Tree Algorithm for Prediction and Classification of Students’ Action, 2016.

[9] Kenneth David Strang. Beyond engagement analytics: which onlinemixed-data factors predict student learning outcomes, 2017.

[10] Surjeet Kumar Yadav, Saurabh Pal and Yong Liu . An Improved Grid Search Algorithm of SVR Parameters Optimization, 2012.