DEPARTMENT

OF

COMPUTER SCIENCE AND ENGINEERING

GITAM SCHOOL OF TECHNOLOGY
GITAM
(Deemed to be University)

MULTICLASS PREDICTION MODEL FOR STUDENT GRADE PREDICTION USING MACHINE LEARNING

A Project Report submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Aarsh varma ,221910313002
Rohith venkatesh, 221910313007
Yenugupalli snigdha, 221910313053
Pusapati geetanjali, 221910313058

Under the esteemed guidance of

Name of the Supervisor

PRAVIN MUNDHE
Designation

GUIDE

DEPARTMENT OFCOMPUTER SCIENCE & ENGINEERING

GITAM

(Deemed to be University)

HYDERABAD

MONTH YEAR

November 2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM SCHOOL OF TECHNOLOGY

GITAM

(Deemed to be University)

DECLARATION

I/We, hereby declare that the project report entitled "**MULTICLASS PREDICTION MODEL FOR STUDENT GRADE PREDICTION USING MACHINE LEARNING "**is an original work done in the Department of Computer Science and Engineering, GITAM Institute 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:

Registration No(s).	**Name(s)**	**Signature(s)**
221910313002	Aarsh varma	
221910313007	Rohith venkatesh	
221910313053	Yenugupalli snigdha	
221910313058	Pusapati geetanjali	

Sample Page 3

PROJECT COMPLETION CERTIFICATE FROM THE ORGANISATION/ COMPANY
(OPTIONAL)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM SCHOOL OF TECHNOLOGY

GITAM

(Deemed to be University)

CERTIFICATE

This is to certify that the project report entitled **"MULTICLASS PREDICTION MODEL FOR STUDENTGRADE PREDICTION USING MACHINE LEARNING "**is a bonafide record of work carried out by

Aarsh varma(221910313002)
Rohithvenkatesh(221910313007)
Yenugupalli snigdha(221910313053)
Pusapati geetanjali(221910313058)

students submitted in partial fulfillment of requirement for the award of degree of Bachelors of Technology in Computer Science and Engineering.

Project Guide

PRAVIN MUNDHE

<Name of the Faculty>

GUIDE

<Designation>

Head of the Department

PHANI KUMAR

<Name of the HoD>

Professor

TABLE OF CONTENTS

MULTICLASS PREDICTION MODEL FOR STUDENT GRADE PREDICTION USING MACHINE LEARNING

221910313002- AARSH VARMA

221910313007-ROHITH VENKATESH

221910313053-YENUGUPALLI SNIGDHA

221910313058-PUSAPATI GEETANJALI

ABSTRACT:

One of the most challenging tasks in the education sector in India is to predict student's academic performance due to a huge volume of student data. In the Indian context, we don't have any existing system by which analysing and monitoring can be done to check the progress and performance of the student mostly in Higher education system. Every institution has their own criteria for analysing the performance of the students. The reason for this happing is due to the lack of study on existing prediction techniques and hence to find the best prediction methodology for predicting the student academics progress and performance. Another important reason is the lack in investigating the suitable factors which affect the academic performance and achievement of the student in particular course. We proposed a multiclass prediction model to reduce the overfitting and misclassification results caused by imbalanced multi-classification based on oversampling Synthetic Minority Oversampling Technique (SMOTE) with features selection methods. This proposed model indicates the comparable and promising results that can enhance the prediction performance model for imbalanced multi-classification for student grade prediction.

Keywords: - Machine learning, predictive model, imbalanced problem, student grade prediction, multi-class classification

INTRODUCTION

Today, predictive analytics applications became an urgent desire in higher educational institutions. Predictive analytics used advanced analytics that encompasses machine learning implementation to derive high-quality performance and meaningful information for all education levels. Mostly know that student grade is one of the key performance indicators that can help educators monitor their academic performance. During the past decade, researchers have proposed many variants of machine learning techniques in education domains. However, there are severe challenges in handling imbalanced datasets for enhancing the performance of predicting student grades. Therefore, this project presents a comprehensive analysis of machine learning techniques to predict the final student grades in the first semester courses by improving the performance of predictive accuracy. First, we compare the accuracy performance of six well-known machine learning techniques namely Decision Tree, Support Vector Machine, Naïve Bayes, K-Nearest Neighbours, Logistic Regression and Random Forest using student's course grade dataset. Second, we proposed a multiclass prediction model to reduce the overfitting and misclassification results caused by imbalanced multi-classification based on oversampling Synthetic Minority Oversampling Technique (SMOTE) with features selection methods. This proposed model indicates the comparable and promising results that can enhance the prediction performance model for imbalanced multi-classification for student grade prediction

LITERATURE SURVEY

1. A Review on Predicting Student's Performance using Data Mining Techniques

By Amirah Mohamed Shahiria, Wahidah Husaina , Nur'aini Abdul Rashida ,

Student's performance is an essential part in higher learning institutions. This is because one of the criteria for a high quality university is based on its excellent record of academic achievements. There are a lot of definitions on students' performance based on the previous literature. Usamah et al. (2013) stated that student's performance can be obtained by measuring the learning assessment and co-curriculum. However, most of the studies mentioned about graduation being the measure of student's success. Generally, most of higher learning institutions in Malaysia used the final grades to evaluate students' performance. Final grades are based on course structure, assessment mark, final exam score and also extracurricular activities. The evaluation is important to maintain students' performances and the effectiveness of learning process. By analysing students' performance, a strategic program can be well planned during their period of studies in an institution [3]. Currently, there are many techniques being proposed to evaluate students' performance.

Data mining is one of the most popular techniques to analyse students' performance. Data mining has been widely applied in educational Peer-review under responsibility of organizing committee of Information Systems International Conference (ISICO2015) Amirah Mohamed Shahiri et al. / Procedia Computer Science 72 (2015) 414 – 422 415 area recently. It is called educational data mining. Educational data mining is a process used to extract useful information and patterns from a huge educational database. The useful information and patterns can be used in predicting students' performance. As a result, it would assist the educators in providing an effective teaching approach. Besides, educators could also monitor their students' achievements. Students could improve their learning activities, allowing the administration to improve the systems performance. Thus, the application of data mining techniques can be focused on specific needs with different entities. In order to encounter the problems, a systematically review is proposed. The proposed systematically review is to support the objectives of this study, which are: 1. to study and identify the gaps in existing prediction methods. 2. To study and identify the variables used in analysing students' performance. 3. To study the existing prediction methods for predicting students' performance.

Summary

Predicting students' performance is mostly useful to help the educators and learners improving their learning and teaching process. This paper has reviewed previous studies on predicting students'

performance with various analytical methods. Most of the researchers have used cumulative grade point average (CGPA) and internal assessment as data sets. While for prediction techniques, the classification method is frequently used in educational data mining area. Under the classification techniques, Neural Network and Decision Tree are the two methods highly used by the researchers for predicting students' performance. In conclusion, the meta-analysis on predicting students' performance has motivated us to carry out further research to be applied in our environment. It will help the educational system to monitor the students' performance in a systematic way.

2. **Predicting Academic Performance**: **By Arto Hellas: University of Helsinki, Petri Ihantola: University of Helsinki, Andrew Petersen: University of Toronto Mississauga**

The ability to predict student performance in a course or program creates opportunities to improve educational outcomes. With effective performance prediction approaches, instructors can allocate resources and instruction more accurately. Research in this area seeks to identify features that can be used to make predictions, to identify algorithms that can improve predictions, and to quantify aspects of student performance. Moreover, research in predicting student performance seeks to determine interrelated features and to identify the underlying reasons why certain features work better than others. This working group report presents a systematic literature review of

work in the area of predicting student performance. Our analysis shows a clearly increasing amount of research in this area, as well as an increasing variety of techniques used. At the same time, the review uncovered a number of issues with research quality that drives a need for the community to provide more detailed reporting of methods and results and to increase efforts to validate and replicate work.

Summary

Overall, we saw a broad range of surveys in terms of quality, area and amount of the literature covered, and focus. We see an opportunity to provide a higher-level view of the methods being used and to survey the literature over a longer period of time than the existing reviews. We also believe there is an unified need for analysis that relates the methods and features used to the context in which they were applied.

3. Predicting Students' Academic Performance at Secondary and Intermediate Level Using Machine Learning By Shah Hussain & Muhammad Qasim Khan

Forecasting academic performance of student has been a substantial research inquest in the Educational Data-Mining that utilizes Machine-learning (ML) procedures to probe the data of educational setups. Quantifying student academic performance is challenging

because academic performance of students hinges on several factors. The in hand research work focuses on students' grade and marks prediction utilizing supervised ML approaches. The data-set utilized in this research work has been obtained from the Board of Intermediate & Secondary Education (B.I.S.E) Peshawar, Khyber Pakhtunkhwa. There are 7 areas in BISEP i.e., Peshawar, FR-Peshawar, Charsadda, Khyber, Mohmand and Upper and Lower Chitral. This paper aims to examine the quality of education that is closely related to the aims of sustainability. The system has created an abundance of data which needs to be properly analysed so that most useful information should be obtained for planning and future development. Grade and marks forecasting of students with their historical educational record is a renowned and valuable application in the EDM. It becomes an incredible information source that could be utilized in various ways to enhance the standard of education nationwide. Relevant research study reveals that numerous methods for academic performance forecasting are built to carryout improvements in administrative and teaching staff of academic organizations. In the put forwarded approach, the acquired data-set is pre-processed to purify the data quality, the labelled academic historical data of student (30 optimum attributes) is utilized to train regression model and DT-classifier. The regression will forecast marks, while grade will be forecasted by classification system, eventually analysed the results obtained by the models. The results

obtained show that machine learning technology is efficient and relevant for predicting students' performance.

Summary

An approach is proposed in this article for observing and forecasting the student's marks and grades in an automated way. This research study aims to gain better accuracy for the classification and low root means square error. This study also led us to make groups of students who have same education historic record, for instance, students have taken the same subjects in the same academic session. This job is not simple and easy, fact that intermediate & secondary grade students do not have the same conduct while studying in the same group. Thus to attain reliable forecasting outcomes it is essential to choose students' of the same academic section and group. The student marks and grade were analyzed in this study by knowledge areas. It can be justified that a grade from one subject can be utilized to predict from the grade of a student who took the exam in the previous academic session. The proposed GA based decision tree classifier and regression achieved remarkable results, as presented in Figs 3 and 4, for grade prediction, the classification accuracy is 96.64% whereas marks predicting system based on regression has an Root Means Square Error of 5.34.

4. **Predicting academic success in higher education By Eyman Alyahyan & Dilek Düştegör**

Computers have become ubiquitous, especially in the last three decades, and are significantly widespread. This has led to the collection of vast volumes of heterogeneous data, which can be utilized for discovering unknown patterns and trends, as well as hidden relationships, using data mining techniques and tools. The analysis methods of data mining can be roughly categorized as: 1) classical statistics methods (e.g. regression analysis, discriminant analysis, and cluster analysis 2) artificial intelligence (e.g. genetic algorithms, neural computing, and fuzzy logic), and 3) machine learning (e.g. neural networks, symbolic learning, and swarm optimization). The latter consists of a combination of advanced statistical methods and AI heuristics. These techniques can benefit various fields through different objectives, such as extracting patterns, predicting behaviour, or describing trends. A standard data mining process starts by integrating raw data – from different data sources – which is cleaned to remove noise, duplicated or inconsistent data. After that, the cleaned data is transformed into a concise format that can be understood by data mining tools, through filtering and aggregation techniques. Then, the analysis step identifies the existing interesting patterns, which can be displayed for a better visualization standard data mining process. Recently data mining has been applied to various fields like healthcare, business, and also education Indeed, the development of educational database management systems created a large number of educational databases, which enabled the application of data mining to extract useful information from this

data. This led to the emergence of Education Data Mining (EDM) as an independent research field. Nowadays, EDM plays a significant role in discovering patterns of knowledge about educational phenomena and the learning process, including understanding performance. Especially, data mining has been used for predicting a variety of crucial educational outcomes, like performance, retention, success, satisfaction, achievement, and dropout rate. The process of EDM is an iterative knowledge discovery process that consists of hypothesis formulation, testing, and refinement. Despite many publications, including case studies, on educational data mining, it is still difficult for educators – especially if they are a novice to the field of data mining – to effectively apply these techniques to their specific academic problems. Knowledge discovery process in educational institutions This study aims to fill the described gap, by providing a complete guideline, providing easier access to data mining techniques and enabling all the potential of their application to the field of education. In this study, we specifically focus on the problem of predicting the academic success of students in higher education. For this, the state-of-the-art has been compiled into a systematic process, where all related decisions and parameters are comprehensively covered and explained along with arguments.

Summary

Early student performance prediction can help universities to provide timely actions, like planning for appropriate training to improve students' success rate. Exploring educational data can certainly help

in achieving the desired educational goals. By applying EDM techniques, it is possible to develop prediction models to improve student success. However, using data mining techniques can be daunting and challenging for non-technical persons. Despite the many dedicated software's, this is still not a straightforward process, involving many decisions. This study presents a clear set of guidelines to follow for using EDM for success prediction. The study was limited to undergraduate level, however the same principles can be easily adapted to graduate level. It has been prepared for those people who are novice in data mining, machine learning or artificial intelligence.A variety of factors have been investigated in the literature related to its impact on predicting students 'academic success which was measured as academic achievement, as our investigation showed that prior-academic achievement, student demographics, e-learning activity, psychological attributes, are the most common factors reported. In terms of prediction techniques, many algorithms have been applied to predict student success under the classification technique. Moreover, a six stages framework is proposed, and each stage is presented in detail. While technical background is kept to a minimum, as this not the scope of this study, all possible design and implementation decisions are covered, along with best practices compiled from the relevant literature. It is an important implication of this review that educators and non-proficient users are encouraged to applied EDM techniques for undergraduate students from any discipline (e.g. social sciences). While reported

findings are based on the literature (e.g. potential definition of academic success, features to measure it, important factors), any available additional data can easily be included in the analysis, including faculty data (e.g. competence, criteria of recruitment, academic qualifications) may be to discover new determinants.

5. **Student Academic Performance Using Hybrid Deep Neural Network By Bashir Khan Yousafzai 1, Sher Afzal Khan 1, Taj Rahman 2, Inayat Khan 3, Inam Ullah, Ateeq Ur Rehman, Mohammed Baz, Habib Hamam and Omar Cheikhrouhou**

In the Sustainable Development Goals, quality education plays an important role
which is approved by the United Nations [1], this is also an important and basic challenge
for supporting sustainable development globally. The gradual increase in education data is due to the continuous generation of such data from different sources, such as e-learning, learning management systems, admission Sustainability systems, and student feedback analysis systems. The student data acquired from the aforementioned sources are used for making a simple query-level decision, whereas a huge bulk of data remains unused due to the complex and noisy nature of datasets. Student related educational data have received considerable attention from researchers in the field of Educational Data Mining (EDP) for finding useful information, such as the

prediction of student performance [2]. Therefore, it is an essential task to investigate and apply state-of-the-art deep learning techniques in the domain of Educational Data Mining (EDM)for efficient prediction of performance from students' historical data. The assessment of student performance from historical data has been investigated by different researchers by employing EDM techniques. The main emphasis of these works is on the early prediction of student performance in terms of marks, grades, and pass/fail.

However, prediction of student academic performance from noisy and large datasets is a

Challenging task due to the following major limitations associated with the existing works:

(i) Poor selection of predictor variables describing the student performance, and (ii) use of

Machine learning techniques, based on feature classical representation schemes followed

by a classifier.

The authors studied the machine learning-based technique for the prediction of student performance using historical data. In the baseline study, different ML classifiers

are used to predict student performance in terms of binary classes (pass/fail). However,

Prediction of performance in terms of pass/fail does not provide a deeper insight into

student’s academic assessment. Another major drawback of their technique is that it is
Deficient in terms of evaluating the overall dependencies pertaining to predictor variables
in the student data. Therefore, the classical machine learning classifiers do not provide an
efficient mechanism for predicting student performance from academic data.
To overcome the drawbacks associated with the baseline study, we employ an improved feature selection technique followed by a deep neural network model, which has
successfully been used in different applications such as rumour detection, extremist affiliation detection, and other domains. We propose to employ a Chi-Square test for feature
selection and the attention-based BiLSTM model for student grade prediction. It works as
follows: (i) in the feature selection module, the Chi-Square test extracts the most appropriate high ranked features having a significant role in student grade prediction, and (ii) the
Bidirectional Long Short-Term Memory (BiLSTM) considers the contextual information of
the past as well as the future, and (iii) the attention mechanism has also been introduced to
capture the most significant features from the given student data. Therefore, the proposed

technique takes advantage of the functionalities of both improved feature selection and
BiLSTM, along with the attention layer, to predict students' final grades on the basis of
their historical academic performance.
The prediction of grades (performance) from students' historical academic data faces
different challenges, such as poor selection of predictor variables, the small size of the
dataset labelled with binary classes (pass/fail). Additionally, the machine learning technique is applied to predict student grades. To overcome these issues, we take the task of
student grade prediction from historical data as the multi-label classification problem, in
which, from the given input student data, final grade G3 is predicted as "A1", "A", "B", "C",
"D", "E", or "F". A sequence of student's performance training data D = [td1, td2, tdn]
is taken as input by the deep learning method to predict the final grade G3e [0 6], i.e.,
0 if A1, 1 for A, etc., and 6 for F. Our goal is to build an automatic technique that learns
from the given data and labels, to accurately predict the respective class labels (student

grade). The aim of the study is to build a computation model based on improved feature
selection and hybrid deep neural network in the domain of EDM, which is trained over
student academic performance historical data, and which can predict student final grade.
Our aim at the development of a deep neural network model is based on improved
feature selection. Firstly, a statistical technique, namely the Chi-Square test, is applied to
select the most appropriate predictor variables for student grade prediction. In the next
step, the Bidirectional Long Short-Term Memory (BiLSTM) is applied, where the forward

LSTM keeps track of future information and backward LSTM manipulates past information.
Finally, the attention layer is introduced to implement an attention mechanism to capture
the most significant features from the given student data.

Summary

Educational data generated through various platforms such as e-learning, e-admission systems, and automated result management systems can be effectively processed through educational data mining techniques in order to gather highly useful insights into students'

performance. The prediction of student performance from historical academic data is a highly desirable application of educational data mining. In this regard, there is an urgent need to develop an automated technique for student performance prediction. Existing studies on student performance prediction primarily focus on utilizing the conventional feature representation schemes, where extracted features are fed to a classifier. In recent years, deep learning has enabled researchers to automatically extract high-level features from raw data. Such advanced feature representation schemes enable superior performance in challenging tasks. In this work, we examine the deep neural network model, namely, the attention based Bidirectional Long Short-Term Memory (BiLSTM) network to efficiently predict student performance (grades) from historical data. In this article, we have used the most advanced BiLSTM combined with an attention mechanism model by analysing existing research problems, which are based on advanced feature classification and prediction. This work is really vital for academicians, universities, and government departments to early predict the performance. The superior sequence learning capabilities of BiLSTM combined with attention mechanism yield superior performance compared to the existing state-of-the-art. The proposed method has achieved a prediction accuracy of 90.16%.

6. CS1 Student Grade Prediction: Unconscious Optimism vs. Insecurity by Sónia Rolland Sobral

CS1 (computer science 1) is the designation widely used for introduction to programming courses in computer science major since ACM's 1978 Computing Curricula. The course belongs to the first semester and is where many students start using programming language to run small programs. Teaching how to program is a task that proves to be complicated since although some students love it and can succeed easily, others feel that it is almost impossible to be able to pass the course, which often leads them to abandon it . Failure and dropout rates are traditionally very high , which makes any professional responsible and motivated to carry out all tasks in order to help students develop the necessary skills in such a unit, Some methods and strategies have been extensively studied and tried to motivate students who do not have the internal strength to succeed: active methodologies, project based learning [10], agile methodology such as SCRUM, pair programming and many others. Some researchers suggest the use of self-assessment as a way to rethink their own work and commit students to tasks , and to have greater awareness of their own behaviour, Manuscript received Mar 30, 2020 Sónia Rolland Sobral is with REMIT, Universidad Portucalense, Porto, Portugal (e-mail: sonia@upt.pt). Motivation and cognition. This article tries to assess the possibility of using self-assessment as one of the strategies in teaching programming to freshman university students. As an exploratory technique, an experiment is made to propose to students that after the exam and after the correction are

presented by the teacher, students should do a prediction of the grade they will obtain. The curricular units foresee two tests and two group projects. The grades expected by the students are also indicated by them after making the presentation and discussing the project. There are experiments that use grade prediction at the beginning of the semester, during the semester, before the exam, others at the end of the exam. The particularity is that this article is associated with a course of introduction to programming in which the results are almost objective and not as subjective as the experiences reported in the literature, usually in psychology or economics. This experience is only experimental and the grades that students say they deserve are not used in the formula for calculating the final classification of the course. This strategy is used very cautiously because the literature indicates that there is a clear tendency for students to be too optimistic and to classify themselves with grades that are not at all what they deserve. This article is divided into a literature review on grade prediction by students, methodology (course organization, survey methodology and data collection), results, discussion and conclusions.

Summary

The difficulties of many students in introductory programming courses and the consequent failure and drop out make it necessary to look for motivation strategies for them to be successful. One of the strategies that is touted in the literature is self-assessment to compromise and motivate students. As we had doubts about the possibility of this strategy, we did an experiment and asked the

students to predict the grades of the two tests and the two projects during a semester. Even knowing the correction grid and exercises that involve programming languages, which shows the result to the programmer, we found that the students' forecasts were not very accurate. In the first test we found that the worst students said they were going to get reasonable grades and much better than reality, while the best students thought they had worse grades than they actually had. The other moments of evaluation did not have as severe results, but forecasts continued to be inaccurate. We did tests by gender, by age, for being a freshman or not, for having taken a computer course in high school and for previous knowledge of programming languages: none of these variables proved to be as significant as the students' grades and their corresponding insecurity-fear or optimism-unconscious.

OBJECTIVE:

The main objective of this application is to propose a way to handle imbalanced datasets for enhancing the performance of predicting student grades.

METHODOLOGY AND ALGORITHMS:

1. Logistic Regression

Logistic regression is used to deal a classification problem. It gives the binomial outcome as it gives the probability if a 37 event will

occur or not (in terms of 0 and 1) based on values of input variables. For example, predicting if a tumour is malignant or benign or an e-mail is classified as spam or not are the instances which can be considered as binomial outcome of Logistic Regression. There can be multinomial outcome of Logistic Regression as well e.g., prediction of type of cuisine preferred: Chinese, Italian, Mexican etc. There can be ordinal outcome as well like: product rating 1 to 5 etc. So Logistic Regression deals with prediction of target variable which is categorical. Whereas Linear Regression deals with prediction of values of continuous variable egg. prediction of real estate price over a span of 3 years. Logistic Regression has the following advantages:

simplicity of implementation, computational efficiency, efficiency from training perspective, ease of regularization. No scaling is required for input features.

This algorithm is predominantly used to solve problems of industry scale. As the output of Logistic Regression is a probability score so to apply it for solving business problem it is required to specify customized performance metrics so as to obtain a cut-off which can be used to do the classification of the target. Also, logistic regression is not affected by small noise in the data and multicollinearity.

Logistic Regression has the following disadvantages: inability to solve non-linear problem as its decision surface is linear, prone to over fitting, will not work out well unless all independent variables are identified. Some examples of practical application of Logistic Regression are: predicting the risk of developing a given disease, cancer diagnosis, predicting mortality of injured patients and in engineering for predicting probability of failure of a given process, system or product.

2. Decision Tree

Decision Tree is a Supervised Machine Learning approach to solve classification and regression problems by continuously splitting data based on a certain parameter. The decisions are in the leaves and the data is split in the nodes. In Classification Tree the decision variable is categorical (outcome in the form of Yes/No) and in Regression tree

the decision variable is continuous. Decision Tree has the following advantages: it is suitable for regression as well as classification problem, ease in interpretation, ease of handling categorical and quantitative values, capable of filling missing values in attributes with the most probable value, high performance due to efficiency of tree traversal algorithm.

Decision Tree might encounter the problem of over-fitting for which Random Forest is the solution which is based on ensemble modelling approach. Disadvantages of decision tree is that it can be unstable, it may be difficult to control size of tree, it may be prone to sampling error and it gives a locally optimal solution- not globally optimal solution. Decision Trees can be used in applications like predicting future use of library books and tumour prognosis problems.

3. Naive Bayes

NAÏVE BAYES This algorithm is simple and is based on conditional probability. In this approach there is a probability table which is the model and through training data it is updated. The "probability table" is based on its feature values where one needs to look up the class probabilities for predicting a new observation. The basic assumption is of conditional independence and that is why it is called "naive". In real world context the assumption that all input features are independent from one another can hardly hold true. Naïve Bayes (NB) have the following advantages: implementation is easy, gives good performance, works with less training data, scales linearly with

number of predictors and data points, handles continuous and discrete data, can handle binary and multi-class classification problems, make probabilistic predictions.

It handles continuous and discrete data. It is not sensitive to irrelevant features. Naïve Bayes has the following disadvantages: Models which are trained and tuned properly often outperform NB models as they are too simple. If there is a need to have one of the features as “continuous variable” (like time) then it is difficult to apply Naive Bayes directly, even though one can make “buckets” for “continuous variables” it’s not 100% correct. There is no true online variant for Naive Bayes, so all data need to be kept for retraining the model. It won’t scale when the number of classes are too high, like > 100K. Even for prediction it takes more runtime memory compared to SVM or simple logistic regression. It is computationally intensive specially for models involving many variables. Naïve Bayes can be used in applications such as Recommendation System and forecasting of cancer relapse or progression after Radiotherapy.

4. K Nearest Neighbour

K Nearest Neighbour (KNN) Algorithm is a classification algorithm It uses a database which is having data points grouped into several classes and the algorithm tries to classify the sample data point given to it as a classification problem. KNN does not assume any underlying data distribution and so it is called non-parametric. Advantages of KNN algorithm are the following: it is simple

technique that is easily implemented. Building the model is cheap. It is extremely flexible classification scheme and well suited for Multi-modal classes. Records are with multiple class labels. Error rate is at most twice that of Bayes error rate. It can sometimes be the best method. KNN outperformed SVM for protein function prediction using expression profiles. Disadvantages of KNN are the following: classifying unknown records are relatively expensive. It requires distance computation of k-nearest neighbours.

With the growth in training set size the algorithm gets computationally intensive, Noisy / irrelevant features will result in degradation of accuracy. It is lazy learner; it computes distance over k neighbours. It does not do any generalization on the training data and keeps all of them. It handles large data sets and hence expensive calculation. Higher dimensional data will result in decline in accuracy of regions. KNN can be used in Recommendation system, in medical diagnosis of multiple diseases showing similar symptoms, credit rating using feature similarity, handwriting detection, analysis done by financial institutions before sanctioning loans, video recognition, forecasting votes for different political parties and image recognition.

5. Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class

selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

6. Synthetic Minority Oversampling Technique (SMOTE)

Imbalanced classification involves developing predictive models on classification datasets that have a severe class imbalance. The challenge of working with imbalanced datasets is that most machine learning techniques will ignore, and in turn have poor performance on, the minority class, although typically it is performance on the minority class that is most important. One approach to addressing imbalanced datasets is to oversample the minority class. The simplest approach involves duplicating examples in the minority class, although these examples don’t add any new information to the model. Instead, new examples can be synthesized from the existing examples. This is a type of data augmentation for the minority class and is referred to as the Synthetic Minority Oversampling Technique, or SMOTE for short.

7. Feature Selection

In machine learning and statistics, feature selection, also known as variable selection, attribute selection or variable subset selection, is the process of selecting a subset of relevant features (variables, predictors) for use in model construction. Feature selection techniques are used for several reasons:

- simplification of models to make them easier to interpret by researchers/users
- shorter training times
- to avoid the curse of dimensionality
- improve data's compatibility with a learning model class
- encode inherent symmetries present in the input space.

The central premise when using a feature selection technique is that the data contains some features that are either redundant or irrelevant, and can thus be removed without incurring much loss of information. Redundant and irrelevant are two distinct notions, since one relevant feature may be redundant in the presence of another relevant feature with which it is strongly correlated.

Feature selection techniques should be distinguished from feature extraction. Feature extraction creates new features from functions of the original features, whereas feature selection returns a subset of the features. Feature selection techniques are often used in domains where there are many features and comparatively few samples (or data points). Archetypal cases for the application of feature selection

include the analysis of written texts and DNA microarray data, where there are many thousands of features, and a few tens to hundreds of samples.

SOFTWARE DEVELOPMENT LIFE CYCLE – SDLC:

In our project we use waterfall model as our software development cycle because of its step-by-step procedure while implementing.

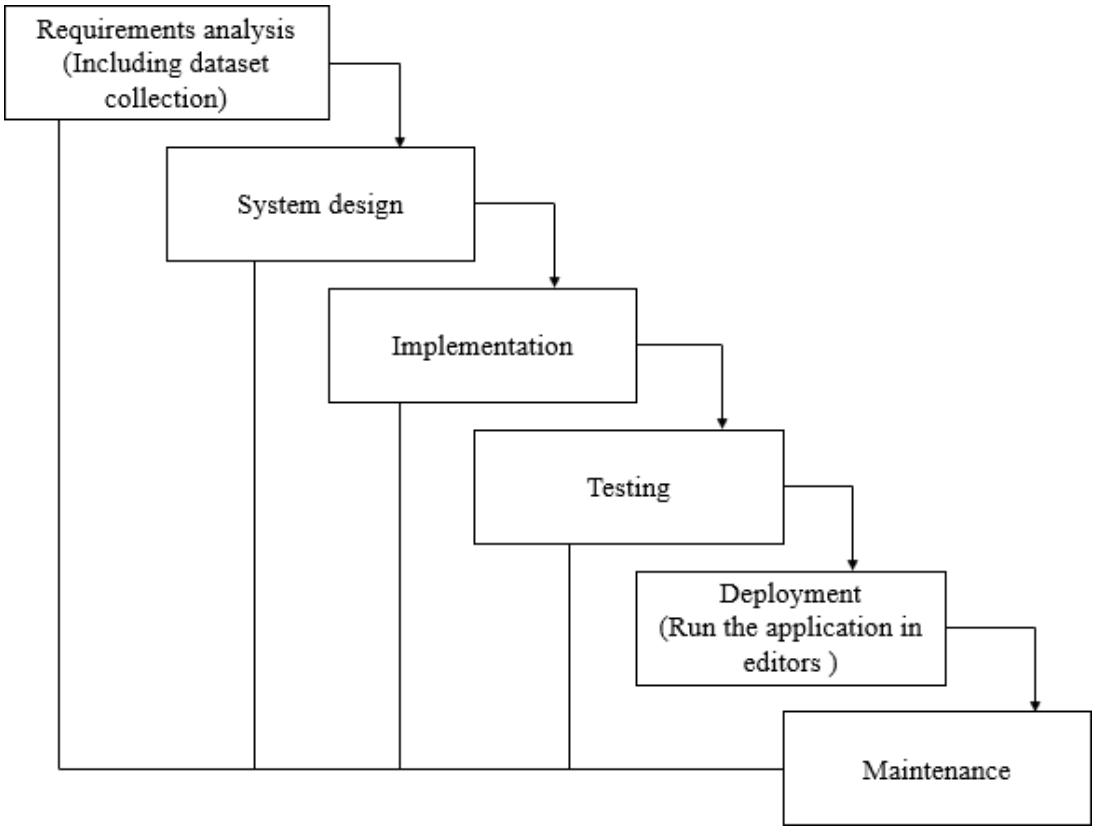


Fig1: Waterfall Model

- **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released.

Maintenance is done to deliver these changes in the customer environment.

FEASIBILITY STUDY

The feasibility of the project is analysed 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

Economic 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 REQUIREMENTS SPECIFICATION

Functional and non-functional requirements:

Requirement's analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

Functional Requirements: These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

1) Authentication of user whenever he/she logs into the system
2) System shutdown in case of a cyber-attack
3) A verification email is sent to user whenever he/she register for the first time on some software system.

Non-functional requirements: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:

- Portability
- Security

- Maintainability
- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

Examples of non-functional requirements:

1) Emails should be sent with a latency of no greater than 12 hours from such an activity.
2) The processing of each request should be done within 10 seconds
3) The site should load in 3 seconds whenever of simultaneous users are > 10000

SOFTWARE AND HARDWARE REQUIREMENTS:

Operating system	: Windows 7 or 7+
Ram	: 8 GB
Hard disc or SSD	: More than 500 GB
Processor	: Intel 3rd generation or high or Ryzen with 8GB Ram
Software's	: Python 3.6 or high version, Visual studio,PyCharm.

SYSTEM DESIGN:

Input Design:

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well-designed input forms and screens have following properties −

- It should serve specific purpose effectively such as storing, recording, and retrieving the information.
- It ensures proper completion with accuracy.
- It should be easy to fill and straightforward.
- It should focus on user's attention, consistency, and simplicity.
- All these objectives are obtained using the knowledge of basic design principles regarding −
    - What are the inputs needed for the system?
    - How end users respond to different elements of forms and screens.

Objectives for Input Design:

The objectives of input design are −

- To design data entry and input procedures
- To reduce input volume

- To design source documents for data capture or devise other data capture methods
- To design input data records, data entry screens, user interface screens, etc.
- To use validation checks and develop effective input controls.

Output Design:

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

Objectives of Output Design:

The objectives of input design are:

1. To develop output design that serves the intended purpose and eliminates the production of unwanted output.
2. To develop the output design that meets the end user's requirements.
3. To deliver the appropriate quantity of output.
4. To form the output in appropriate format and direct it to the right person.

5. To make the output available on time for making good decisions.

MODULES:

1. **User**:

1.1 Data gathering:

Needs to gather the information or data from the open source, this will be use in the train the models.

1.2 Pre-processing:

Data need to be pre-processed according the models it helps to increase the accuracy of the model and better information about the data.

1.3 Feature Engineering:

In this step features are selected based on the priority of the column data, by this we can reduce the time investing on many columns.

1.4 Model Building

To get the final result model building for the dataset is an important step. Based on the dataset we build the model for classification and regression.

1.5 View Results

User view's the generated results from the model.

2. **System**

2.1 Model Checking

System checks model accuracy and it takes of the necessary for the model building

2.2 Generate Results

System takes the input data from the users and produces the output.

UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

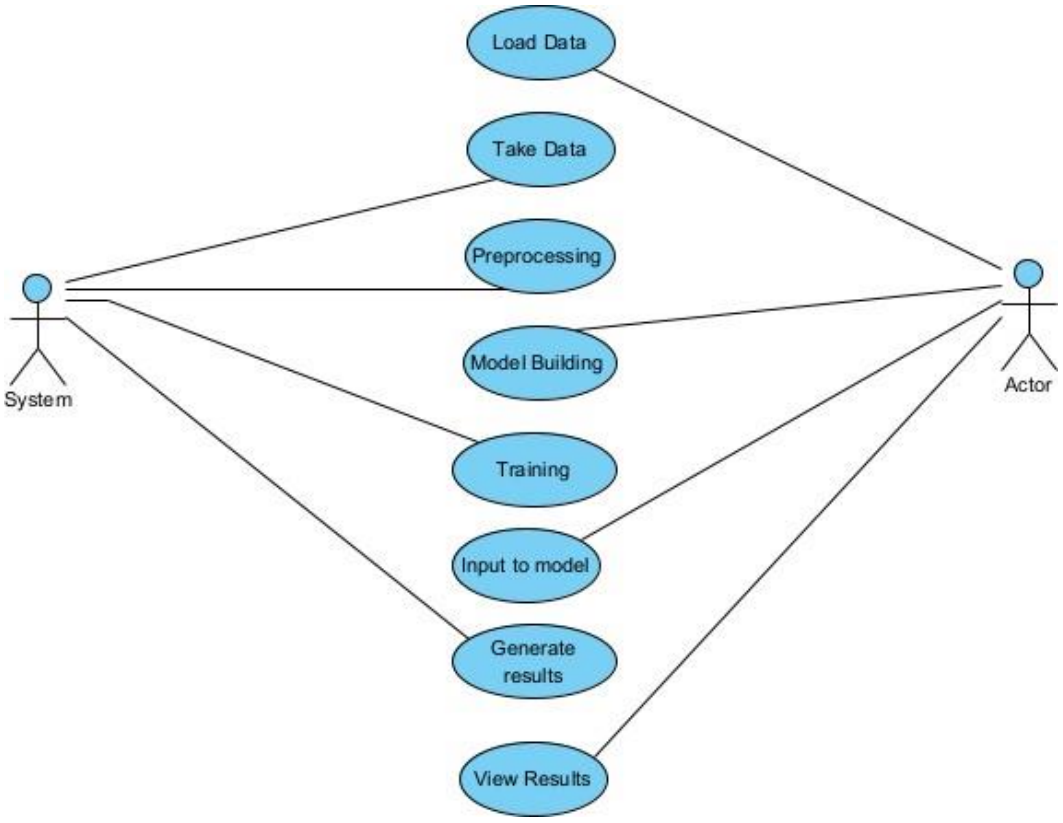
1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM

- A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.

- Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

► The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



CLASS DIAGRAM

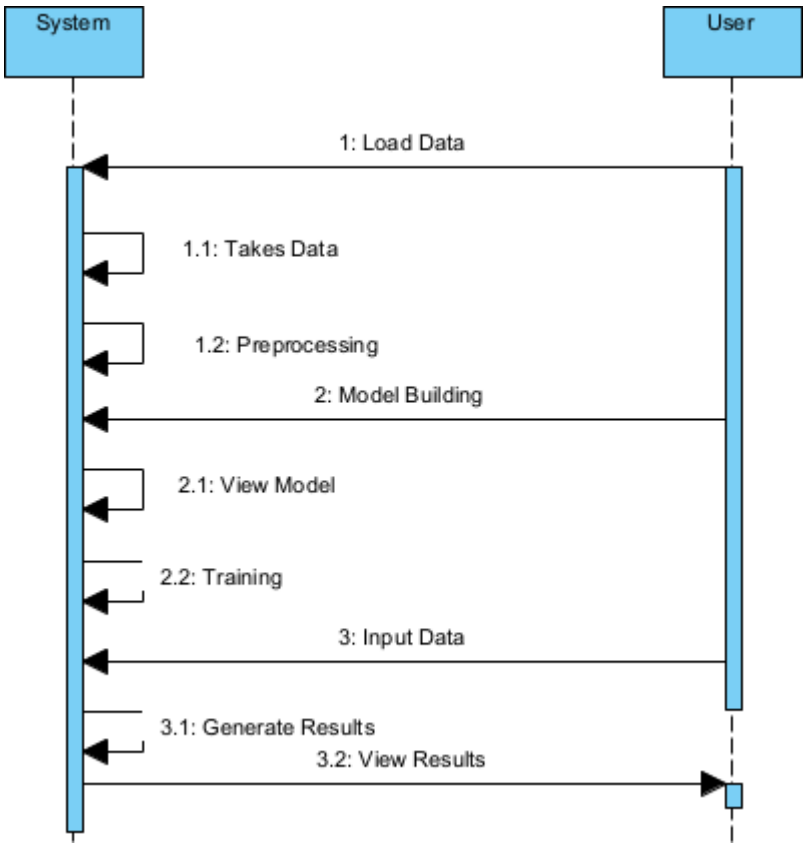
In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information

\

SEQUENCE DIAGRAM

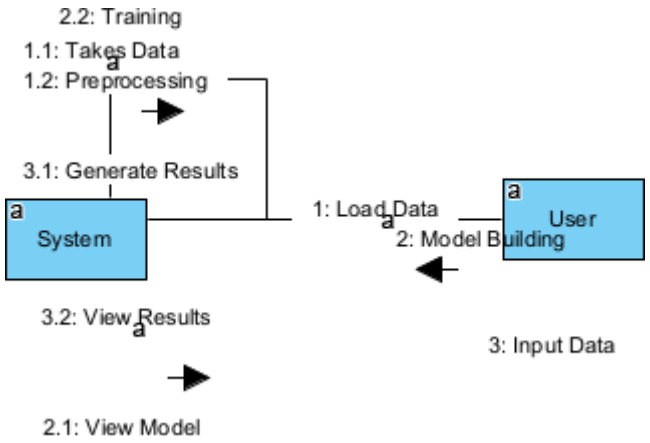
- A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.

- It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams



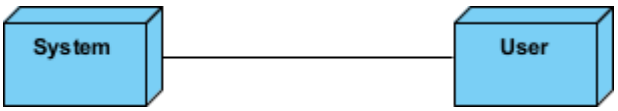
COLLABORATION DIAGRAM:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



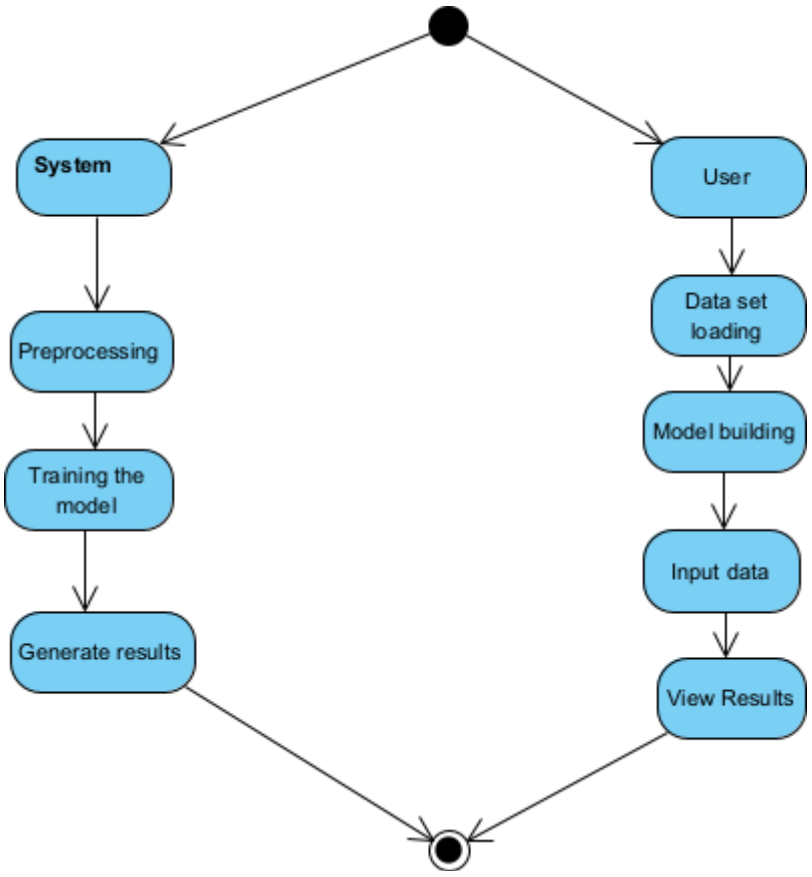
DEPLOYMENT DIAGRAM

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.



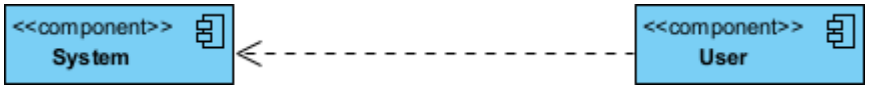
ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development.

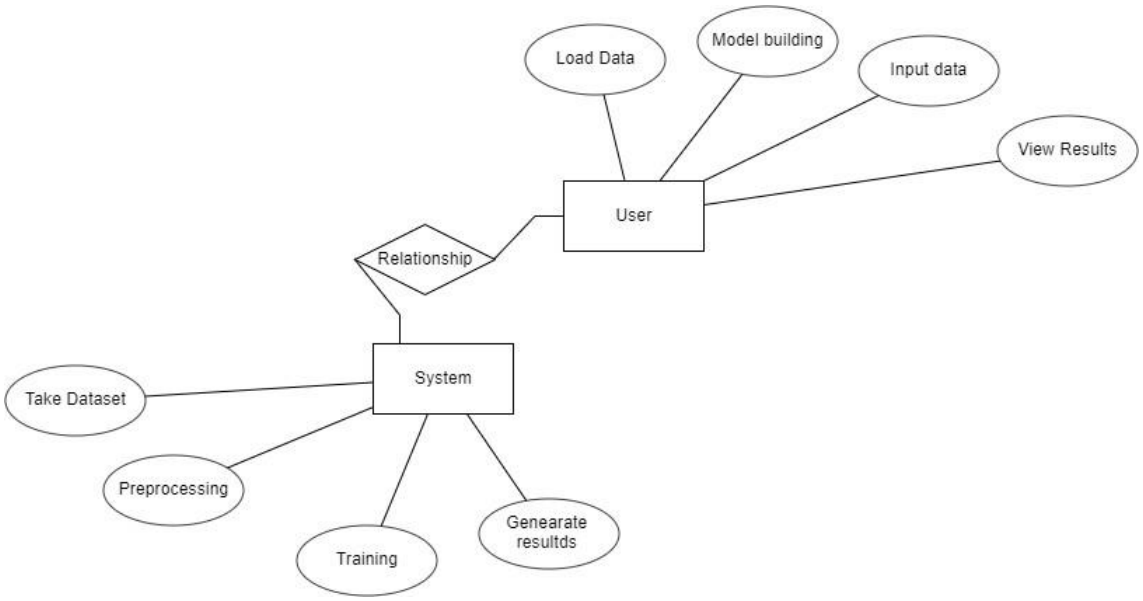


ER DIAGRAM:

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

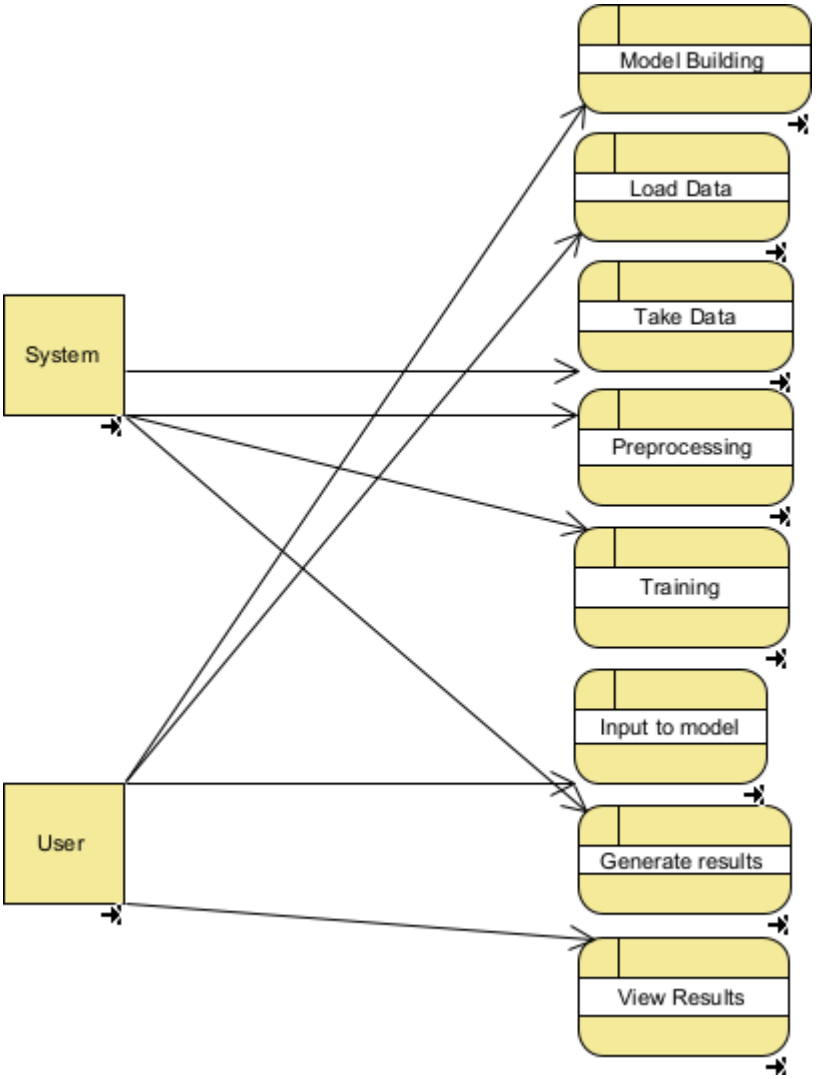
An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let’s have a look at a simple ER diagram to understand this concept.

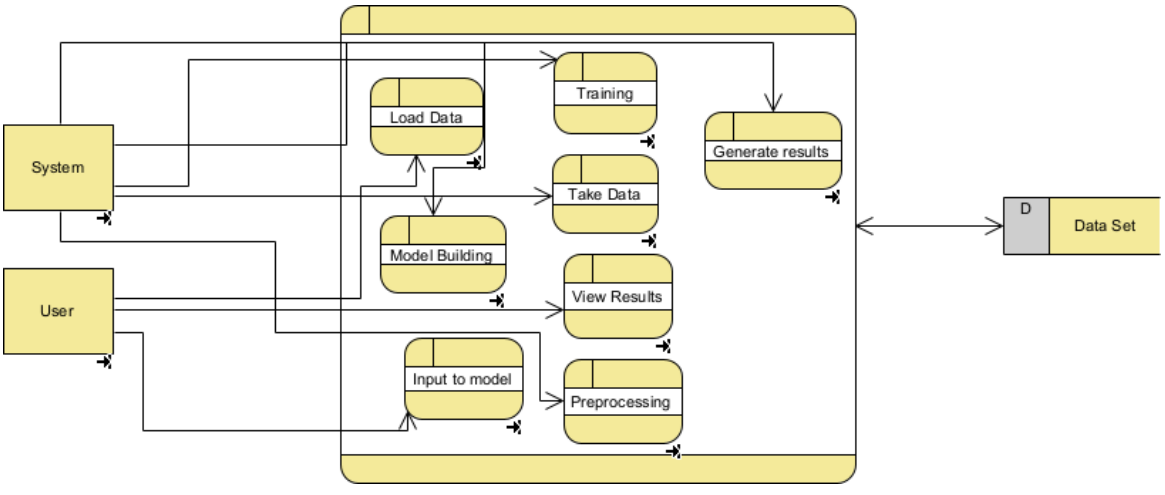
`



DFD DIAGRAM:

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.


Model Building
Load Data
Take Data
System
Preprocessing
Training
Input to model
User
Generate results
View Results


Training
Load Data
Generate results
System
Take Data
D
Data Set
Model Building
User
View Results
Input to model
Preprocessing

SYSTEM ANALYSIS & FEASIBILITY STUDY

Existing Method:

In the existing system, acquiring perfectly balanced and highly related dataset is almost impossible. Although large quantities of data are available but still extracting relevant data is a complex job. To overcome all this, we use machine learning packages available in the scikit-learn library to extract useful data.

Disadvantages:

- High complexity.

- Time consuming

Proposed System:

To overcome the difficulties facing in the existing we use inbuilt packages available in the machine learning libraries. Pre-processing techniques and feature selection techniques are used to improve the accuracy of the model, by using this we can perform easily for any kind of dataset. Feature engineering is important in the proposed system because of if number of columns are in dataset, we can easily find out the important columns, this reduces the execution time and time complexity.

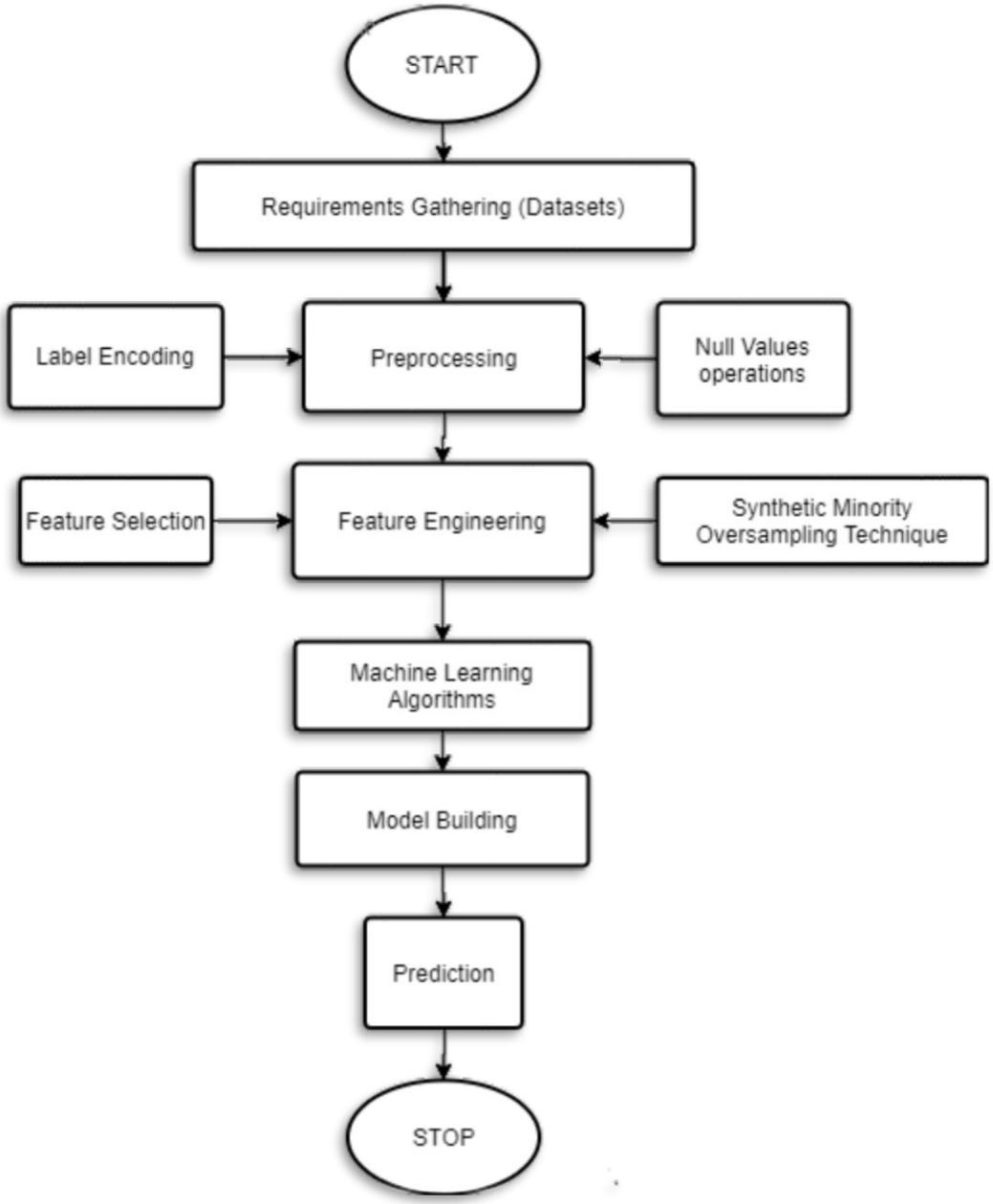


Fig: Block Diagram

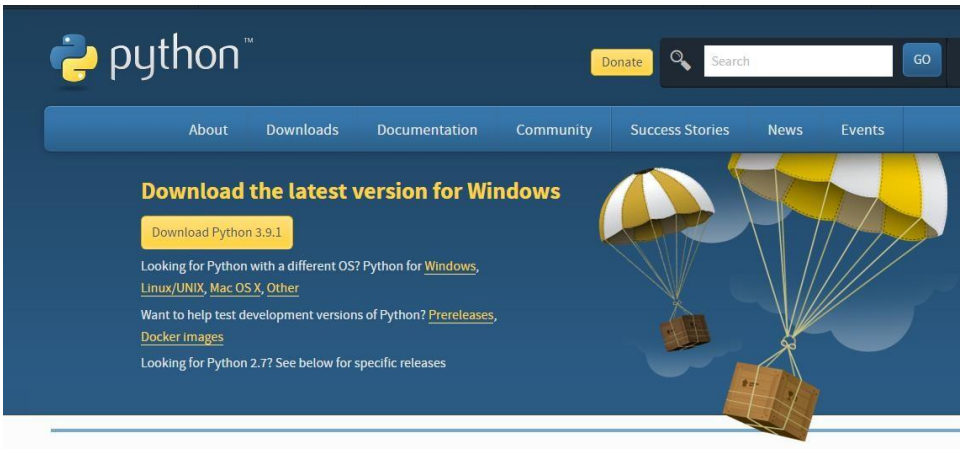
Advantages:

- It increases the accuracy.
- It reduces the time complexity.
- It provides better information on features

SOFTWARE INSTALLATION FOR MACHINE LEARNING PROJECTS:

Installing Python:

1. To download and install Python visit the official website of Python https://www.python.org/downloads/ and choose your version.



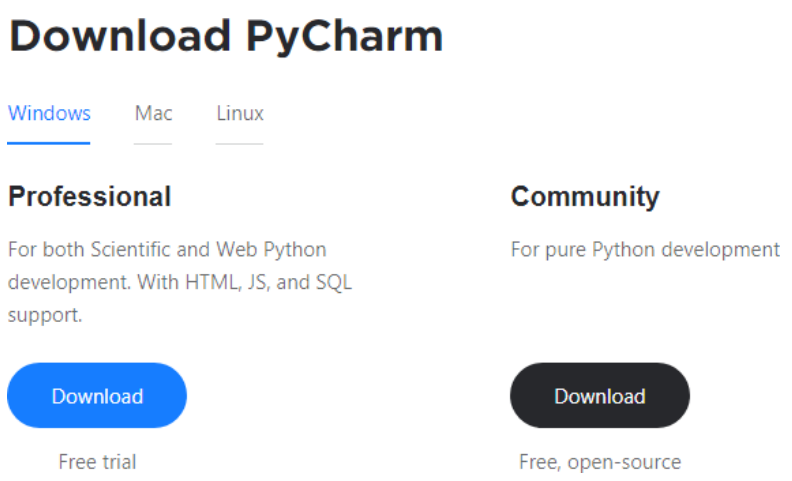
2. Once the download is complete, run the exe for install Python. Now click on Install Now.

3. You can see Python installing at this point.

4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

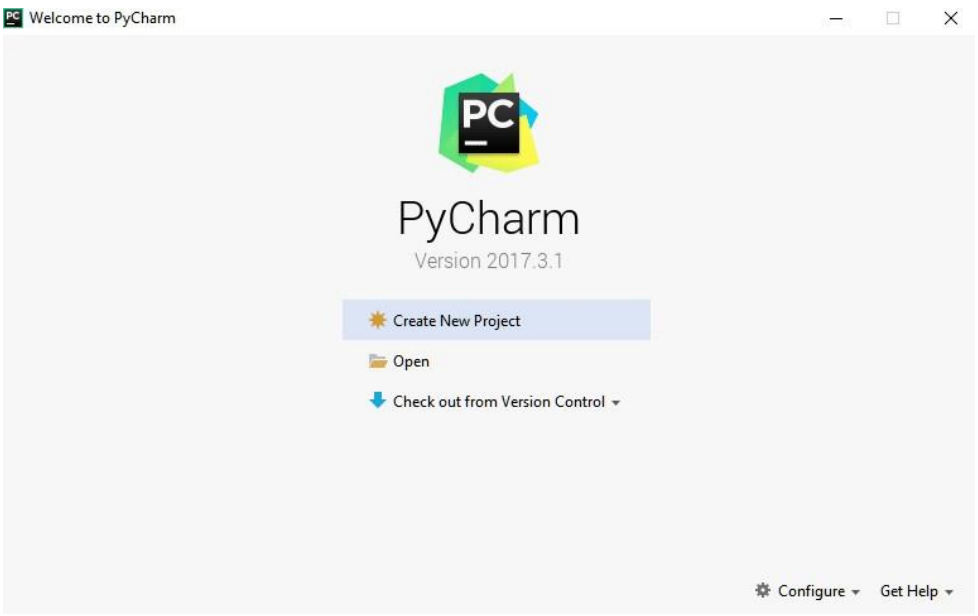
Installing PyCharm:

1. To download PyCharm visit the website https://www.jetbrains.com/pycharm/download/ and click the "DOWNLOAD" link under the Community Section.



2. Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click “Next”.
3. On the next screen, Change the installation path if required. Click “Next”.
4. On the next screen, you can create a desktop shortcut if you want and click on “Next”.
5. Choose the start menu folder. Keep selected JetBrains and click on “Install”.
6. Wait for the installation to finish.

7. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.
8. After you click on "Finish," the Following screen will appear.

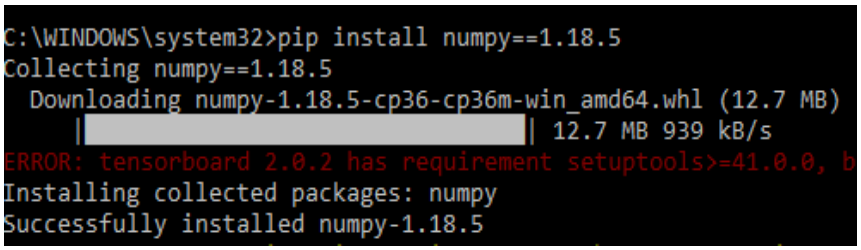


9. You need to install some packages to execute your project in a proper way.

10. Open the command prompt/ anaconda prompt or terminal as administrator.

11. The prompt will get open, with specified path, type “pip install package name” which you want to install (like NumPy, pandas, sea born, scikit-learn, Matplotlib, Pyplot)

Ex: Pip install NumPy



INTRODUCTION TO PYTHON

✓ Python

What Is a Script?

Up to this point, I have concentrated on the interactive programming capability of Python. This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode

Scripts are reusable

Basically, a script is a text file containing the statements that comprise a Python program. Once you have created the script, you can execute it over and over without having to retype it each time.

Scripts are editable

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor. Then you can execute each of the individual versions. In

this way, it is easy to create different programs with a minimum amount of typing.

You will need a text editor

Just about any text editor will suffice for creating Python script files. You can use *Microsoft Notepad, Microsoft WordPad, Microsoft Word,* or just about any word processor if you want to.

Difference between a script and a program

Script:

Scripts are distinct from the core code of the application, which is usually written in a different language, and are often created or at least modified by the end-user. Scripts are often interpreted from source code or byte code, whereas the applications they control are traditionally compiled to native machine code.

Program:

The program has an executable form that the computer can use directly to execute the instructions.

The same program in its human-readable source code form, from which executable programs are derived (e.g., compiled)

Python

What is Python? Chances you are asking yourself this. You may have found this book because you want to learn to program but don't know anything about programming languages. Or you may have heard of

programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to “big name” languages. Hopefully I can explain it for you.

Python concepts

If you’re not interested in the how’s and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it’s a great one to start programming with.

• Open-source general-purpose language.

• Object Oriented, Procedural, Functional

• Easy to interface with C/ObjC/Java/Fortran

• Easy-is to interface with C++ (via SWIG)

• Great interactive environment

• Great interactive environment

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

- Python is Object-Oriented − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- Python is a Beginner's Language − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Possum 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 Possum still holds a vital role in directing its progress.

Python Features

Python's features include −

- Easy-to-learn − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

- Easy-to-read − Python code is more clearly defined and visible to the eyes.
- Easy-to-maintain − Python's source code is fairly easy-to-maintained.
- A broad standard library − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- Interactive Mode − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- Extendable − you can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- Databases − Python provides interfaces to all major commercial databases.
- GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- Scalable − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Dynamic vs. Static

Types Python is a dynamic-typed language. Many other languages are static typed, such as C/C++ and Java. A static typed language requires the programmer to explicitly tell the computer what type of "thing" each data value is.

For example, in C if you had a variable that was to contain the price of something, you would have to declare the variable as a "float" type.

This tells the compiler that the only data that can be used for that variable must be a floating point number, i.e. a number with a decimal point.

If any other data value was assigned to that variable, the compiler would give an error when trying to compile the program.

Python, however, doesn't require this. You simply give your variables names and assign values to them. The interpreter takes care

of keeping track of what kinds of objects your program is using. This also means that you can change the size of the values as you develop the program. Say you have another decimal number (a.k.a. a floating point number) you need in your program.

With a static typed language, you have to decide the memory size the variable can take when you first initialize that variable. A double is a floating point value that can handle a much larger number than a normal float (the actual memory sizes depend on the operating environment).

If you declare a variable to be a float but later on assign a value that is too big to it, your program will fail; you will have to go back and change that variable to be a double.

With Python, it doesn't matter. You simply give it whatever number you want and Python will take care of manipulating it as needed. It even works for derived values.

For example, say you are dividing two numbers. One is a floating point number and one is an integer. Python realizes that it's more accurate to keep track of decimals so it automatically calculates the result as a floating point number

Variables

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by

assigning different data types to variables, you can store integers, decimals or characters in these variables.

Standard Data Types

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types −

- Numbers
- String
- List
- Tuple
- Dictionary

Python Numbers

Number data types store numeric values. Number objects are created when you assign a value to them

Python Strings

Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes. Subsets of strings can be taken using the slice

operator ([ ] and [:]) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

Python Lists

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type.

The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (*) is the repetition operator.

Python Tuples

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.

The main differences between lists and tuples are: Lists are enclosed in brackets ([ ]) and their elements and size can be changed, while tuples are enclosed in parentheses (( )) and cannot be updated. Tuples can be thought of as read-only lists.

Python Dictionary

Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object.

Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).

Different modes in python

Python has two basic modes: normal and interactive.

The normal mode is the mode where the scripted and finished .pie files are run in the Python interpreter.

Interactive mode is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole

20 Python libraries

1. Requests. The most famous http library written by Kenneth remits. It’s a must have for every python developer.

2. Scrappy. If you are involved in web scraping then this is a must have library for you. After using this library you won’t use any other.

3. Python. A guy toolkit for python. I have primarily used it in place of tinder. You will really love it.

4. Pillow. A friendly fork of PIL (Python Imaging Library). It is more user friendly than PIL and is a must have for anyone who works with images.

5. SQL Alchemy. A database library. Many love it and many hate it. The choice is yours.

6. Beautiful Soup. I know it’s slow but this xml and html parsing library is very useful for beginners.

7. Twisted. The most important tool for any network application developer. It has a very beautiful ape and is used by a lot of famous python developers.

8. Numbly. How can we leave this very important library? It provides some advance math functionalities to python.

9. Skippy. When we talk about numbly then we have to talk about spicy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.

10. Matplotlib. A numerical plotting library. It is very useful for any data scientist or any data analyser.

11. Pygmy. Which developer does not like to play games and develop them? This library will help you achieve your goal of 2d game development.

12. Piglet. A 3d animation and game creation engine. This is the engine in which the famous python port of mine craft was made

13. Pit. A GUI toolkit for python. It is my second choice after python for developing GUI's for my python scripts.
14. Pit. Another python GUI library. It is the same library in which the famous Bit torrent client is created.
15. Scaly. A packet sniffer and analyser for python made in python.
16. Pywin32. A python library which provides some useful methods and classes for interacting with windows.
17. Notch. Natural Language Toolkit – I realize most people won't be using this one, but it's generic enough. It is a very useful library if you want to manipulate strings. But its capacity is beyond that. Do check it out.
18. Nose. A testing framework for python. It is used by millions of python developers. It is a must have if you do test driven development.
19. Simply. Simply can-do algebraic evaluation, differentiation, expansion, complex numbers, etc. It is contained in a pure Python distribution.
20. I Python. I just can't stress enough how useful this tool is. It is a python prompt on steroids. It has completion, history, shell capabilities, and a lot more. Make sure that you take a look at it.

NumPy

Humpy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed

by a tuple of positive integers. In numbly dimensions are called *axes*. The number of axes is *rank.*

• Offers Matlab-ish capabilities within Python

• Fast array operations

• 2D arrays, multi-D arrays, linear algebra etc.

Matplotlib

• High quality plotting library.

Python class and objects

These are the building blocks of OOP. Class creates a new object. This object can be anything, whether an abstract data concept or a model of a physical object, e.g. a chair. Each class has individual characteristics unique to that class, including variables and methods. Classes are very powerful and currently “the big thing” in most programming languages. Hence, there are several chapters dedicated to OOP later in the book.

The class is the most basic component of object-oriented programming. Previously, you learned how to use functions to make your program do something.

Now will move into the big, scary world of Object-Oriented Programming (OOP). To be honest, it took me several months to get a handle on objects.

When I first learned C and C++, I did great; functions just made sense for me.

Having messed around with BASIC in the early ’90s, I realized functions were just like subroutines so there wasn’t much new to learn.

However, when my C++ course started talking about objects, classes, and all the new features of OOP, my grades definitely suffered.

Once you learn OOP, you’ll realize that it’s actually a pretty powerful tool. Plus many Python libraries and APIs use classes, so you should at least be able to understand what the code is doing.

One thing to note about Python and OOP: it’s not mandatory to use objects in your code in a way that works best; maybe you don’t need to have a full-blown class with initialization code and methods to just return a calculation. With Python, you can get as technical as you want.

As you’ve already seen, Python can do just fine with functions. Unlike languages such as Java, you aren’t tied down to a single way of doing things; you can mix functions and classes as necessary in the same program. This lets you build the code

Objects are an encapsulation of variables and functions into a single entity. Objects get their variables and functions from classes. Classes are essentially a template to create your objects.

Here’s a brief list of Python OOP ideas:

• The class statement creates a class object and gives it a name. This creates a new namespace.

• Assignments within the class create class attributes. These attributes are accessed by qualifying the name using dot syntax: ClassName.Attribute.

• Class attributes export the state of an object and its associated behaviour. These attributes are shared by all instances of a class.

• Calling a class (just like a function) creates a new instance of the class.

This is where the multiple copy's part comes in.

• Each instance gets ("inherits") the default class attributes and gets its own namespace. This prevents instance objects from overlapping and confusing the program.

• Using the term self identifies a particular instance, allowing for per-instance attributes. This allows items such as variables to be associated with a particular instance.

Inheritance

First off, classes allow you to modify a program without really making changes to it.

To elaborate, by sub classing a class, you can change the behaviour of the program by simply adding new components to it rather than rewriting the existing components.

As we've seen, an instance of a class inherits the attributes of that class.

However, classes can also inherit attributes from other classes. Hence, a subclass inherits from a superclass allowing you to make a generic superclass that is specialized via subclasses.

The subclasses can override the logic in a superclass, allowing you to change the behavior of your classes without changing the superclass at all.

Operator Overloads

Operator overloading simply means that objects that you create from classes can respond to actions (operations) that are already defined within Python, such as addition, slicing, printing, etc.

Even though these actions can be implemented via class methods, using overloading ties the behavior closer to Python's object model and the object interfaces are more consistent to Python's built-in objects, hence overloading is easier to learn and use.

User-made classes can override nearly all of Python's built-in operation methods

Exceptions

I've talked about exceptions before but now I will talk about them in depth. Essentially, exceptions are events that modify program's flow, either intentionally or due to errors.

They are special events that can occur due to an error, e.g. trying to open a file that doesn't exist, or when the program reaches a marker, such as the completion of a loop.

Exceptions, by definition, don’t occur very often; hence, they are the "exception to the rule" and a special class has been created for them. Exceptions are everywhere in Python.

Virtually every module in the standard Python library uses them, and Python itself will raise them in a lot of different circumstances.

Here are just a few examples:

• Accessing a non−existent dictionary key will raise a Key Error exception.

• Searching a list for a non−existent value will raise a Value Error exception

. • Calling a non−existent method will raise an Attribute Error exception.

• Referencing a non−existent variable will raise a Name Error exception.

• Mixing data types without coercion will raise a Type Error exception.

One use of exceptions is to catch a fault and allow the program to continue working; we have seen this before when we talked about files.

This is the most common way to use exceptions. When programming with the Python command line interpreter, you don’t need to worry about catching exceptions.

Your program is usually short enough to not be hurt too much if an exception occurs.

Plus, having the exception occur at the command line is a quick and easy way to tell if your code logic has a problem.

However, if the same error occurred in your real program, it will fail and stop working. Exceptions can be created manually in the code by raising an exception.

It operates exactly as a system-caused exceptions, except that the programmer is doing it on purpose. This can be for a number of reasons. One of the benefits of using exceptions is that, by their nature, they don't put any overhead on the code processing.

Because exceptions aren't supposed to happen very often, they aren't processed until they occur.

Exceptions can be thought of as a special form of the if/elf statements. You can realistically do the same thing with if blocks as you can with exceptions.

However, as already mentioned, exceptions aren't processed until they occur; if blocks are processed all the time.

Proper use of exceptions can help the performance of your program.

The more infrequent the error might occur, the better off you are to use exceptions; using if blocks requires Python to always test extra conditions before continuing.

Exceptions also make code management easier: if your programming logic is mixed in with error-handling if statements, it can be difficult to read, modify, and debug your program.

User-Defined Exceptions

I won’t spend too much time talking about this, but Python does allow for a programmer to create his own exceptions.

You probably won’t have to do this very often but it’s nice to have the option when necessary.

However, before making your own exceptions, make sure there isn’t one of the built-in exceptions that will work for you.

They have been "tested by fire" over the years and not only work effectively, they have been optimized for performance and are bug-free.

Making your own exceptions involves object-oriented programming, which will be covered in the next chapter

. To make a custom exception, the programmer determines which base exception to use as the class to inherit from, e.g. making an exception for negative numbers or one for imaginary numbers would probably fall under the Arithmetic Error exception class.

To make a custom exception, simply inherit the base exception and define what it will do.

Python modules

Python allows us to store our code in files (also called modules). This is very useful for more serious programming, where we do not want to retype a long function definition from the very beginning just to change one mistake. In doing this, we are essentially defining our own modules, just like the modules defined already in the Python library.

To support this, Python has a way to put definitions in a file and use them in a script or in an interactive instance of the interpreter. Such a

file is called a *module*; definitions from a module can be *imported* into other modules or into the *main* module.

Testing code

As indicated above, code is usually developed in a file using an editor.

To test the code, import it into a Python session and try to run it.

Usually there is an error, so you go back to the file, make a correction, and test again.

This process is repeated until you are satisfied that the code works. T

His entire process is known as the development cycle.

There are two types of errors that you will encounter. Syntax errors occur when the form of some command is invalid.

This happens when you make typing errors such as misspellings, or call something by the wrong name, and for many other reasons. Python will always give an error message for a syntax error.

Functions in Python

It is possible, and very useful, to define our own functions in Python. Generally speaking, if you need to do a calculation only once, then use the interpreter. But when you or others have need to perform a certain type of calculation many times, then define a function.

You use functions in programming to bundle a set of instructions that you want to use repeatedly or that, because of their complexity, are better self-contained in a sub-program and called

when needed. That means that a function is a piece of code written to carry out a specified task.

To carry out that specific task, the function might or might not need multiple inputs. When the task is carved out, the function can or cannot return one or more values.

There are three types of functions in python:

Help (), min (), print ().

Namespaces in Python are implemented as Python dictionaries, this means it is a mapping from names (keys) to objects (values). The user doesn't have to know this to write a Python program and when using namespaces.

Some namespaces in Python:

- global names of a module
- local names in a function or method invocation
- built-in names: this namespace contains built-in functions (e.g. abs(), camp(), ...) and built-in exception names

Garbage Collection

Garbage Collector exposes the underlying memory management mechanism of Python, the automatic garbage collector. The module includes functions for controlling how the collector operates and to examine the objects known to the system, either pending collection or stuck in reference cycles and unable to be freed.

Python XML Parser

XML is a portable, open source language that allows programmers to develop applications that can be read by other applications, regardless of operating system and/or developmental language.

What is XML? The Extensible Markup Language XML is a markup language much like HTML or SGML.

This is recommended by the World Wide Web Consortium and available as an open standard.

XML is extremely useful for keeping track of small to medium amounts of data without requiring a SQL-based backbone.

XML Parser Architectures and APIs the Python standard library provides a minimal but useful set of interfaces to work with XML.

The two most basic and broadly used APIs to XML data are the SAX and DOM interfaces.

Simple API for XML SAX: Here, you register callbacks for events of interest and then let the parser proceed through the document.

This is useful when your documents are large or you have memory limitations, it parses the file as it reads it from disk and the entire file is never stored in memory.

Document Object Model DOM API : This is a World Wide Web Consortium recommendation wherein the entire file is read into memory and stored in a hierarchical tree − based form to represent all the features of an XML document.

SAX obviously cannot process information as fast as DOM can when working with large files. On the other hand, using DOM exclusively can really kill your resources, especially if used on a lot of small files.

SAX is read-only, while DOM allows changes to the XML file. Since these two different APIs literally complement each other, there is no reason why you cannot use them both for large projects.

Python Web Frameworks

A web framework is a code library that makes a developer's life easier when building reliable, scalable and maintainable web applications.

Why are web frameworks useful?

Web frameworks encapsulate what developers have learned over the past twenty years while programming sites and applications for the web. Frameworks make it easier to reuse code for common HTTP operations and to structure projects so other developers with knowledge of the framework can quickly build and maintain the application.

Common web framework functionality

Frameworks provide functionality in their code or through extensions to perform common operations required to run web applications. These common operations include:

1. URL routing
2. HTML, XML, JSON, and other output format tinplating
3. Database manipulation
4. Security against Cross-site request forgery (CSRF) and other attacks
5. Session storage and retrieval

Not all web frameworks include code for all of the above functionality. Frameworks fall on the spectrum from executing a single use case to providing every known web framework feature to every developer. Some frameworks take the "batteries-included" approach where everything possible comes bundled with the framework while others have a minimal core package that is amenable to extensions provided by other packages.

Comparing web frameworks

There is also a repository called compare-python-web-frameworks where the same web application is being coded with varying Python web frameworks, tinplating engines and object.

Web framework resources

- When you are learning how to use one or more web frameworks it's helpful to have an idea of what the code under the covers is doing.
- Frameworks is a really well done short video that explains how to choose between web frameworks. The author has some particular opinions about what should be in a framework. For the most part I agree although I've found sessions and database ORMs to be a helpful part of a framework when done well.
- What is a web framework? Is an in-depth explanation of what web frameworks are and their relation to web servers?
- Jingo vs. Flash vs. Pyramid: Choosing a Python web framework contains background information and code comparisons

for similar web applications built in these three big Python frameworks.

- This fascinating blog post takes a look at the code complexity of several Python web frameworks by providing visualizations based on their code bases.
- Python’s web frameworks benchmarks is a test of the responsiveness of a framework with encoding an object to JSON and returning it as a response as well as retrieving data from the database and rendering it in a template. There were no conclusive results but the output is fun to read about nonetheless.
- What web frameworks do you use and why are they awesome? Is a language agnostic Reedit discussion on web frameworks? It's interesting to see what programmers in other languages like and dislike about their suite of web frameworks compared to the main Python frameworks.
- This user-voted question & answer site asked "What are the best general purpose Python web frameworks usable in production?” The votes aren't as important as the list of the many frameworks that are available to Python developers.

Web frameworks learning checklist

1. Choose a major Python web framework (Jingo or Flask are recommended) and stick with it. When you're just starting it's best to learn one framework first instead of bouncing around trying to understand every framework.

2. Work through a detailed tutorial found within the resources links on the framework's page.
3. Study open source examples built with your framework of choice so you can take parts of those projects and reuse the code in your application.
4. Build the first simple iteration of your web application then go to the deployment section to make it accessible on the web.

2. SYSTEM STUDY

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units

of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

SYSTEM_TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the

software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

6.1 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format

- No duplicate entries should be allowed
- All links should take the user to the correct page.

6.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.3 Acceptance Testing

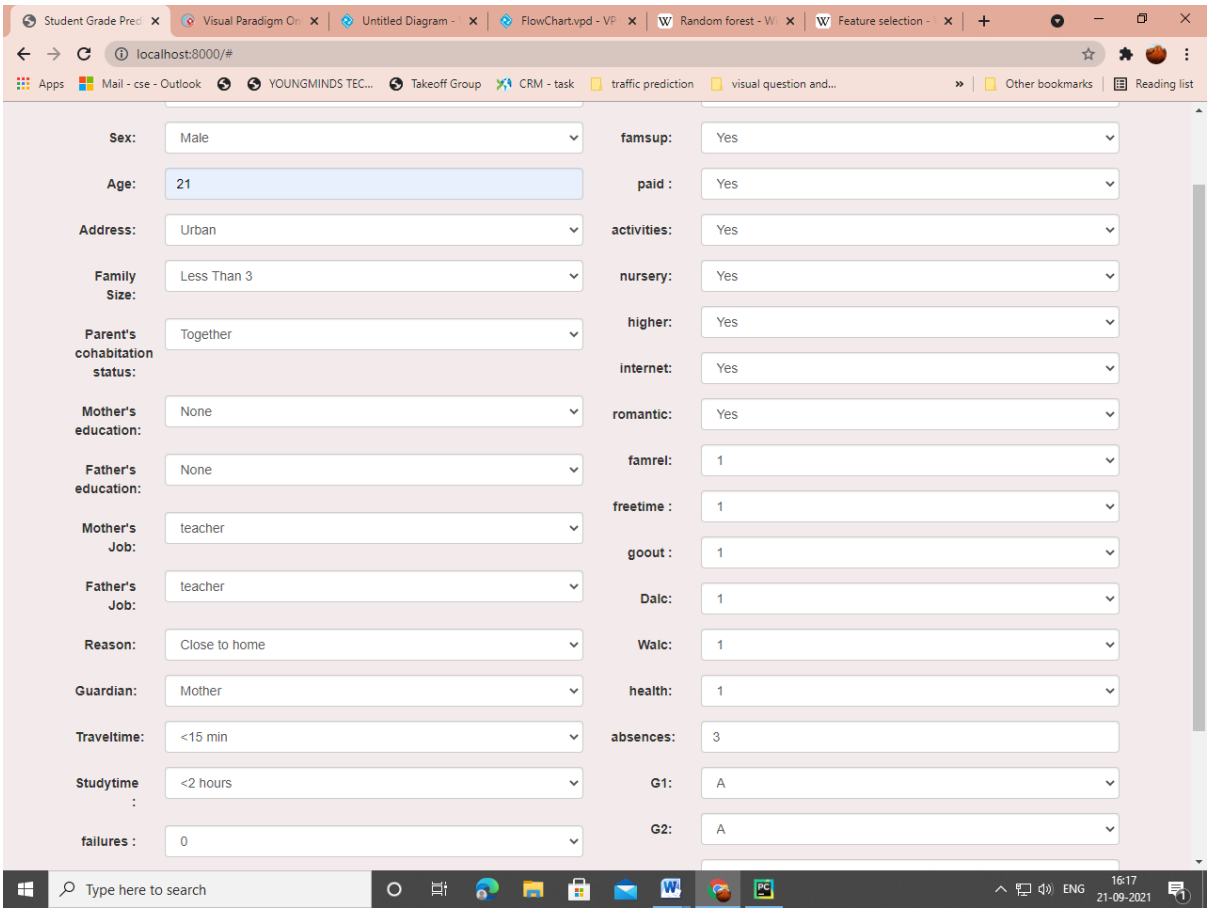
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

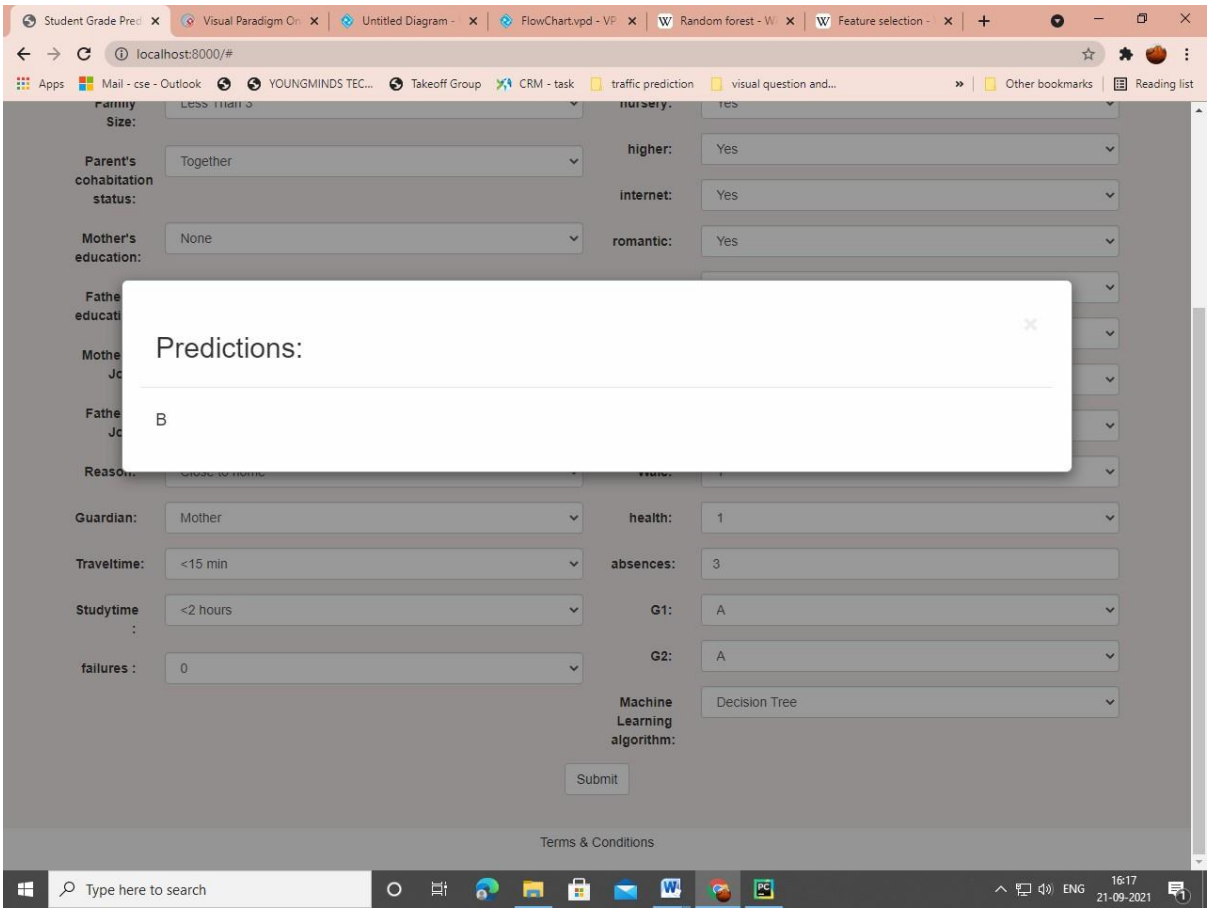
Test Results: All the test cases mentioned above passed successfully. No defects encountered.

OUTPUT SCREEN SHOTS WITH DESCRIPTION

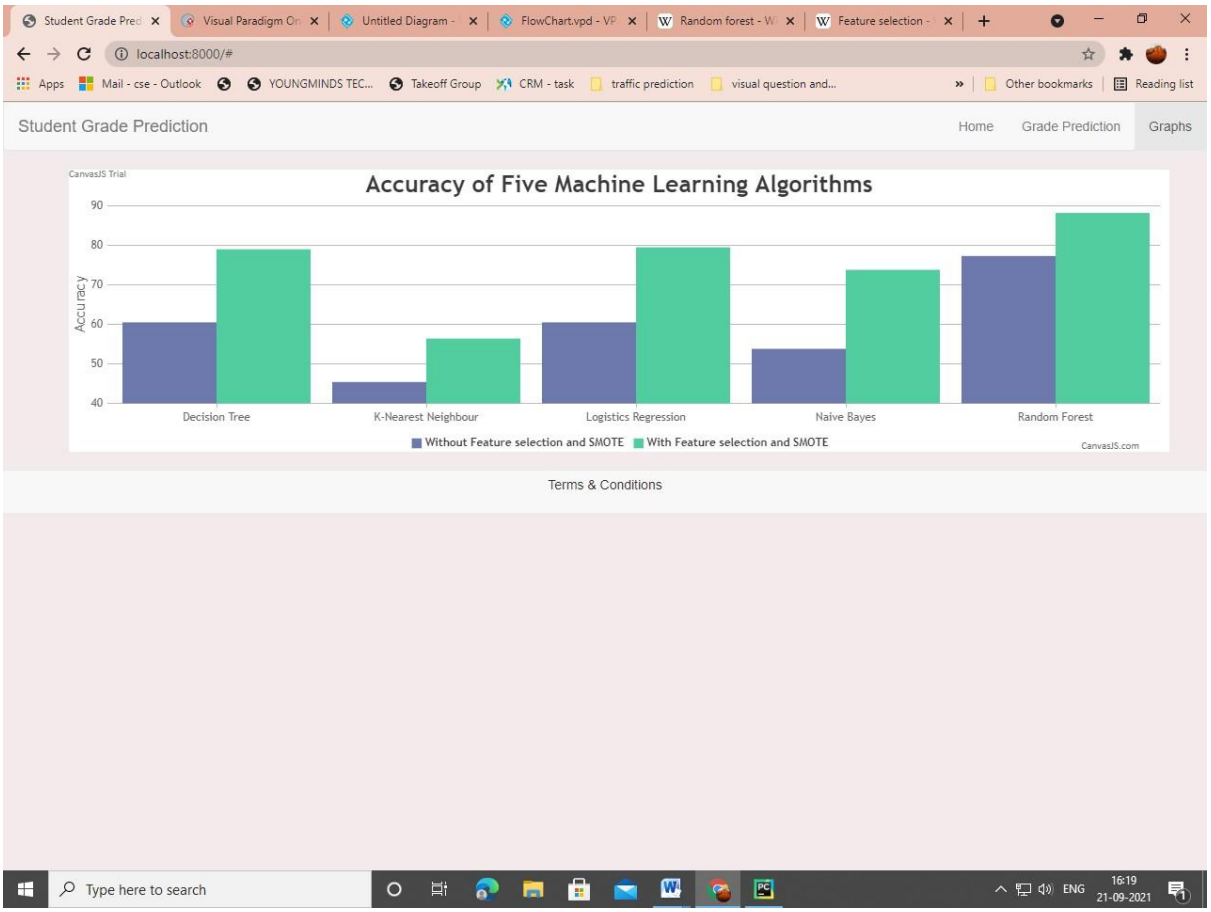
Student Grade Prediction:

This page describes the predicted grade of the given input features.





This Page describes the accuracy of models



Accuracy without SMOTE and Feature Selection

- Decision Tree: 60.5%
- K-Nearest Neighbours: 45.4%
- Logistic Regression: 60.5%
- Naïve Bayes: 53.8%
- Random Forest: 77.3%

Accuracy with SMOTE and Feature Selection

- Decision Tree: 79%

- K-Nearest Neighbours: 56.4%
- Logistic Regression: 79.5%
- Naïve Bayes: 73.8%
- Random Forest: 88.2%

TEST CASES

Input	Output	Result
Input features	Tested for different features given by user on the different model.	Success
Student Grade classification	Tested for different input features given by the user on different features from the models are created using the different algorithms and data.	Success
Grade Prediction	Grade prediction will be performed using the different models build from the algorithms.	Success

Test cases Model building:

S.NO	Test cases	I/O	Expected O/T	Actual O/T	P/F
1	Read the datasets.	Dataset's path.	Datasets need to read successfully.	Datasets fetched successfully.	It produced P. If this not F will come
2	Verifying the features and generates result.	Input features as input, for Student Grade classification	Output is classified as different Grades	Output is classified as different Grades	It produced P. If this is not, it will undergo F
3	Verifying the features and generates result	Input features as input for Student Grade prediction	Need to predict the student Grade.	Model successfully predicted Student Grade	It produced P. If this is not, it will undergo F

4	Verifying the features and generates result	Input features as input for Student Grade prediction	Need to predict the student Grade with DT	Model successfully predicted Student Grade	It produced P. If this is not, it will undergo F
5	Verifying the features and generates result	Input features as input for Student Grade prediction	Need to predict the student Grade with KNN	Model successfully predicted Student Grade	It produced P. If this is not, it will undergo F
6	Verifying the features and generates result	Input features as input for Student Grade prediction	Need to predict the student Grade with LR	Model successfully predicted Student Grade	It produced P. If this is not, it will undergo F
7	Verifyin	Input	Need to	Model	It

	g the features and generates result	features as input for Student Grade prediction	predict the student Grade with NB	successfully predicted Student Grade	produced P. If this is not, it will undergo F
8	Verifying the features and generates result	Input features as input for Student Grade prediction	Need to predict the student Grade with RF	Model successfully predicted Student Grade	It produced P. If this is not, it will undergo F
9	Verifying the features and generates result	Input features as input for Student Grade prediction	Need to predict the student Grade with DT with SMOTE and Feature Selection	Model successfully predicted Student Grade	It produced P. If this is not, it will undergo F
10	Verifying the	Input features as	Need to predict the	Model successfull	It produce

	features and generates result	input for Student Grade prediction	student Grade with KNN with SMOTE and Feature Selection	y predicted Student Grade	d P. If this is not, it will undergo F
11	Verifying the features and generates result	Input features as input for Student Grade prediction	Need to predict the student Grade with LR with SMOTE and Feature Selection	Model successfully predicted Student Grade	It produced P. If this is not, it will undergo F
12	Verifying the features and generates result	Input features as input for Student Grade prediction	Need to predict the student Grade with NB with SMOTE and Feature Selection	Model successfully predicted Student Grade	It produced P. If this is not, it will undergo F
13	Verifying the features	Input features as input for	Need to predict the student	Model successfully predicted	It produced P. If

	and generates result	Student Grade prediction	Grade with RF with SMOTE and Feature Selection	Student Grade	this is not, it will undergo F

CONCLUSION:

Predicting student grades is one of the key performance indicators that can help educators monitor their academic performance. Therefore, it is important to have a predictive model that can reduce the level of uncertainty in the outcome for an imbalanced dataset. This project proposes a multiclass prediction model with six predictive models to predict final student's grades based on the previous student final examination result of the first-semester course. Specifically, we have done a comparative analysis of combining oversampling SMOTE with FS methods to evaluate the performance accuracy of student grade prediction. We also have shown that the explored oversampling SMOTE is overall improved consistently with using FS with all predictive models.

FUTURE SCOPE

As for future works, further investigation on the use of appropriate emerging predictive techniques in such advanced machine learning algorithms [40] and more ensemble algorithms are recommended to optimize the result for predicting student grades. It is also essential to select several multi-class imbalanced datasets to be analysed with appropriate sampling techniques and different evaluation metrics which suitable for the imbalanced multi-class domain such as Kappa, Weighted Accuracy and other measures. Thus, using machine learning in higher learning institutions for student grade prediction will ultimately enhance the decision support system to improve their student academic performance in the future.

REFERENCES:

1. X. Zhang, R. Xue, B. Liu, W. Lu, and Y. Zhang, ‘‘Grade prediction of student academic performance with multiple classification models,’’.
2. S. T. Jishan, R. I. Rashu, N. Haque, and R. M. Rahman, ‘‘Improving accuracy of students’ final grade prediction model using optimal equal width binning and synthetic minority over-sampling technique,’’
3. A. Polyzou and G. Karypis, ‘‘Grade prediction with models specific to students and courses,’’
4. Z. Iqbal, J. Qadir, A. N. Mian, and F. Kamiran, ‘‘Machine learning based student grade prediction

5. I. Khan, A. Al Sadiri, A. R. Ahmad, and N. Jabeur, ‘‘Tracking student performance in introductory programming by Means of machine learning,’’
6. M. A. Al-Barrak and M. Al-Razgan, ‘‘Predicting students final GPA using decision trees:
7. E. C. Abana, ‘‘A decision tree approach for predicting student grades in research project using WEKA,
8. F. Ahmad, N. H. Ismail, and A. A. Aziz, ‘‘The prediction of students’ academic performance using classification data mining techniques,’’
9. T. Anderson and R. Anderson, ‘‘Applications of machine learning to student grade prediction in quantitative business courses,’’
10. S. Hussain, N. A. Dahan, F. M. Ba-Alwib, and N. Ribata, ‘‘Educational data mining and analysis of students’ academic performance using WEKA,’’
11. A. Verma, ‘‘Evaluation of classification algorithms with solutions to class imbalance problem on bank marketing dataset using WEKA,
12. D. Berrar, ‘‘Cross-validation,’’ Comput. Biol.,
13. M. Hussain, W. Zhu, W. Zhang, S. M. R. Abidi, and S. Ali, ‘‘Using machine learning to predict student difficulties from learning session data,
14. B. Predić, G. Dimić, D. Rančić, P. Štrbac, N. Maček, and P. Spalević, ‘‘Improving final grade prediction accuracy in blended learning environment using voting ensembles,’’