**OBJECTIVE:**

Due to this pandemic situation, where the majority of the population has been confined to their homes and online procurement of goods for daily use is the new normal, reviews of such goods on the online space plays a huge role and acts as a metric by which users can select products without physically feeling and checking them

**PROBLEM STATEMENT**

Moreover, in the case of online shopping for medicines, reviews become even more important as mining on less descriptive, unverified and bogus reviews may become lethal. Also, due to the absence of certified medical practitioners, most of the users are relying heavily on self-medication. In such a scenario, it becomes very difficult for non-expert users to study reviews of a drug and identify the disease for which the drug is prescribed for

**ABSTRACT**

Drug reviews play a very important role in providing crucial medical care information for both healthcare professionals and consumers. Also, in the absence of an actual practicing healthcare professional, a consumer can look for an online review of drugs before making a purchase. But these reviews are generally unstructured in nature and often do not provide concise information on the disease/nature of the disease, the drugs are prescribed for. In this scenario, a learning model that can be trained to predict the disease/type of disease, when provided with a drug name and its corresponding review, becomes very important. To mitigate the above-mentioned issue, we present and compare various machine learning-based prediction models. Also, the performance of each of the models has been quantified using metrics such as precision, recall, F1-Score, and accuracy.

**INTRODUCTION** Despite genomic and technological advances, drug discovery and development continues to be a time-consuming and costly process. The number of approved new drugs has remained far below expectations, notwithstanding substantial investments in the pharmaceutical and health sciences. Therefore, one attractive option is to reduce the time and cost of drug development by expanding the scope of usage of already approved, known drugs. Given that these drugs have passed stringent approvals by US Food and Drug Administration, there is minimal risk associated with their safety and tolerability. Drug repositioning can dramatically reduce development times and costs from the discovery to the clinical approval stages. Between 20 and 30 scientific papers on the subject of drug repositioning are published each month [1]. Given this data, the importance of repositioned drugs on the market is highlighted by the fact that they account for 30% of new indications per year. Previous efforts to estimate large-scale novel drug indications have focused on the mapping of gene expression profiles [10, 8] and on the recommendation of similar drugs or diseases based on known drug-disease relationships [4]. Machine learning has a significant advantage over other methods, by offering a way in which to optimally combine different drug and disease characteristics into a predictive model. It may also reveal important features that allow for identifying promising drug indications. Machine learning based drug indication prediction studies have used various similarity measures such as chemical structure, sideeffect, protein target information. One such approach is the PREDICT method by [5]. In this method, 5 drug-drug similarity and 2 disease-disease similarity measures were used to train a logistic classifier to predict potential drug-disease association. Zhang and colleagues [23] proposed k-nearest neighbor approach ( Similarity-based LArge-margin learning of Multiple Sources (SLAMS)) to predict novel drug indications by calculating the combined similarity score with the drug data obtained from different sources. Guney [6] developed an open-source software tool for researchers to repeat this work and made it public. Since machine learning generally treats drug indication prediction as a binary classification problem, it is necessary to specify the known drug indications (positive set) and the drug-disease pairs with no indications (negative set). Although the indications in the positive set are usually previously known, the results of clinical trials in which drugs have failed are often not reported. A recent attempt aimed to provide a gold standard database, repoDB [1], that also contains failed drug-indications by retrieving clinical trial records from AACT database 4 . But the number of reported failed indications are far less than number of true indications. In this study, drug and disease features were obtained by querying open data to train our classifier for predicting new drug indications, and the predictive performance of the classifier for different validation schemes was evaluated. We compared our method with previous computational drug indication prediction approaches. We observed that we had better predictive performance than the PREDICT and the SLAMS in disjoint cross-validation settings. Tests and predictions data generated by combining multiple drug indications data sources were evaluated. Finally, we make our work open and freely available so that others can use or extend this methodology Since the arrival of advanced computing, the doctors’ still requires the technology in various possible ways like surgical representation process and x-ray photography, but the technology perceptually stayed behind. The method still requires the doctor’s information and experience due to alternative factors starting from medical records to weather conditions, atmosphere, blood pressure and numerous alternative factors. The huge numbers of variables are consider as entire variables that are required to understand the complete working process itself, however no model has analyzed successfully. To tackle this drawback, Medical decision support systems must be used. This system is able to assist the doctors to make the correct decision. Medical decision support system refers to both the process of attempting to determine or identify possible diseases or disorder and the opinion reached by this process. The diagnostic opinion in the sense, it indicates either degree of abnormality on a continuum or a kind of abnormality in a classification. It’s influenced by non medical factors such as power ethics and financial incentives for patient or doctor. It can be a brief summation or an extensive formulation, even taking the form of story or metaphor. It might be a means of communication such as computer code through which it triggers payment, prescription, notification, information or advice. Indication of medical diagnostic includes knowledge of what is normal and measuring of patient’s current condition. Automated decision support systems are rule based systems that are automatically providing solutions to repetitive management problems. Medical decision could be extremely specialized and difficult job due to alternative factors or incase of rare diseases. The alternative factors include stress; tired misdiagnosis might vary from ignorance of doctors and incomplete information. Standard algorithm may go through the entire variables like prevailing conditions history of medical records, history of family records and various factors relating to the patient records, sheer magnitude of obtainable hidden factors. Differential diagnosis methods can be used to identify the presence of an entity where multiple alternatives are possible and also refers to include the candidate alternatives. This method is needs a process of elimination or obtaining information that shrinks the probability of candidate conditions to negligible levels. It contains four steps: 1) The doctor gather all information about the patients and create a symptoms list.2) The doctor should make a list of all possible causes of symptoms.3) The doctor should prioritize the list by which is the most dangerous possible cause of symptoms put in the top of the list. 4) The doctor should rule out or treat the possible causes beginning with the most urgently dangerous conditions.”Rule Out” in the sense to use the test method or other scientific method. If there will be no such diagnosis means removing the diagnosis from the list and using tests that should have distinct results, depends on which diagnosis is correct. This can be done based on the doctor’s knowledge and experience. This method is very easy to implement.

**LITERATURE REVIEW**

Mauritius, a tropical island in the Indian Ocean, is prone to many communicable diseases such as gastroenteritis and flu among others. Every year, the country faces many cases that the health authorities have to handle and keep under control. There are several applications available worldwide that can allow people to report diseases but no such application allows the health authorities to predict their propagation. A situation may arise where there is a disease outbreak and hospitals might not be prepared to handle the outbreak or may not have the required amount of medicines in stock. The population is currently alerted only after there has been an outbreak and not beforehand. As a result, authorities might be better prepared if they could use latest technologies to predict the propagation of diseases. Technology has played a vital role in healthcare settings during the recent years. Along with technology, crowdsourcing has been very helpful in gathering health related information that can help in solving a medical problem. Crowdsourcing for healthcare refers to the collection or analysis of health and medical research data by anyone who wants to participate [1]. Various crowdsourcing applications for healthcare have been developed with a variety of features. One among the well-known crowdsourcing applications is HealthMap [2]. This application enables user to report outbreaks and allows the real time monitoring as well as propagations of diseases. Machine learning (ML), the study of tools and methods for identifying patterns in data, can help handling health-related data [3]. Machine learning is a natural extension of Artificial Intelligence (AI). The fusion of healthcare data and machine learning in order to identify patterns of interests is known as healthcare informatics [4]. Machine learning can be divided into two broad categories [5], namely supervised learning [6] and unsupervised learning [7]. For unsupervised learning, only input values are used with no desired output. The algorithms discover and identify interesting pattern in data by themselves without any human guidance. Some examples are: K-means and Apriori algorithm. In supervised machine learning, a dataset is used to train the learning algorithm and the learning is stopped upon the achievement of an appropriate performance. Some of these algorithms are: support vector machines, linear regression and artificial neural networks [8]. The primary contribution and motivation of this research is the formulation of a system that uses machine learning, more specifically, artificial neural networks, to predict the propagation of communicable diseases based on current data and forthcoming weather conditions. The proposal is to gather real time disease information through crowdsourcing mobile applications and combine this information with extracted weather forecasts to predict the propagation of diseases. A prototype application has been initially developed and the results are promising with an accuracy of about 90%.

This section presents an overview of the existing healthcare crowdsourcing applications. Founded by a dynamic group of software engineers and researchers at the Boston hospital, HealthMap [2] has been used globally to report any outbreaks that may be happening. The application enables its users to monitor real-time disease reporting and propagations happening on the world map. Additionally, it provides information about emerging diseases and information about local health departments and governments. The main features of HealthMap include: reporting of any human or animal diseases, photo attachment during reporting, locating diseases occurring in the neighbourhood, update on a 24/7/365 basis, information in nine languages and alert of outbreaks near you. Figure 1 [17] is an application designed for users interested in medical cases. Healthcare specialists and medical students use Figure 1 to communicate a medical issue with a photo and a description. Other users of the application can instantly give feedback on the case through comments and they can also follow the case. Through Figure 1, medical students can expand their knowledge on diseases and doctors can share patients’ problems with their colleagues as well as find potential solutions for special cases. This application, amongst other features, allows to browse or comment on cases or give an idea for treatment, post picture and description of patients’ cases and send notification when a comment is added to one’s post. Epidemiologists from Harvard and Boston Children’s hospital created the FluNearYou [18] application in order to use the contribution of the crowd to gather information about influenza and report to the public as well as to researchers. It allows individuals to report symptoms in real-time and provide useful information directly to the public. FluNearYou’s main features are: It locates diseases reported in area through a map, allows reporting of flu with symptoms and vaccination dates, locates nearby vaccines, push notifications to report symptoms and flu news. A mobile application was created in 2011 to allow volunteers to report their symptoms and to identify local sources for vaccines [19]. Their mission was to collect and share user reported data in a free, timely and open way to the public, and to give the earliest alert in case of disease propagation if ever a considerable number of cases are being reported. The creators of the application wished to give users the ability to report diseases in real time and to list additional symptom options. They also wanted to provide a 2-way communication and give alerts to users to remind them to get vaccinated. Additionally, the future works included the analysis from other data sources such as online videos, social media and radio to provide early detection of diseases and alert users about health hazards in their area. SickWeather [20] is an application that uses crowdsourcing and social media monitoring to gather information about illnesses around. The application tracks all the illnesses on a map and provides alert of sickness that may be present in regions of the world along with forecasts. The main features are: alerts about nearby sicknesses, daily forecast on active illness, shares forecast on social networks, enables user to seek medical assistance, allows reporting in any area and shows progress of illness 5 days’ back. Predict and Beat Dengue [21] application was created in order to control and prevent dengue outbreaks. It also employs machine learning, artificial intelligence to provide precautions about dengue for the users. The application allows the reporting of dengue disease in a particular location, submission of photo that could lead to dengue, alert someone if he has entered a dengue prone area and it makes use of artificial intelligence to assists and advice users who are sick. Veta [22] is a simple platform that allows users to report and discover incidents of vector-borne diseases. Information shared on Veta can be used to take necessary precautions and prevent propagation of diseases. The following features are included in this application: Alerts when someone reports a disease, submits report on dengue, shares information about diseases or vaccines and monitors frequently visited locations. CrowdMed [23] is a platform where patients post their problems and medical detectives submit diagnosis with respect to the problems. A study conducted on the reality of crowd-powered healthcare concluded that the application heavily relies on crowdsourcing to solve a patient’s problems. However, the drawbacks are that there is no proper way to verify patient reported data and some patients may also report issues which may be unrealistic. Although several healthcare crowdsourcing applications exist, they are not widely adopted in Mauritius. Moreover, the applications have a severe limitation in the sense that very few can predict the propagation of the diseases. Furthermore, none take into consideration the forthcoming weather forecasts in their prediction. With changes in climatic conditions, it is vital to predict the impact of a forthcoming weather conditions over any existing outbreak of diseases. To our knowledge, there are currently no application in Mauritius that can visually display the status of diseases in real-time on a map and that can predict their propagation. Only conventional ways are being used to get the diseases status and to detect disease outbreaks. Thus, there is a potential for a crowdsourcing application that can display real-time information and also help in predicting the propagation of diseases by taking into account the weather forecasts

**IN “BROWN, A.S., PATEL, C.J.: A STANDARD DATABASE FOR DRUG REPOSITIONING. SCIENTIFIC DATA 4, 170029 (2017)”** Drug repositioning, the process of discovering, validating, and marketing previously approved drugs for new indications, is of growing interest to academia and industry due to reduced time and costs associated with repositioned drugs. Computational methods for repositioning are appealing because they putatively nominate the most promising candidate drugs for a given indication. Comparing the wide array of computational repositioning methods, however, is a challenge due to inconsistencies in method validation in the field. Furthermore, a common simplifying assumption, that all novel predictions are false, is intellectually unsatisfying and hinders reproducibility. We address this assumption by providing a gold standard database, repoDB, that consists of both true positives (approved drugs), and true negatives (failed drugs). We have made the full database and all code used to prepare it publicly available, and have developed a web application that allows users to browse subsets of the data Drug repositioning is the process of discovering, validating, and marketing previously approved drugs for new indications. The drug repositioning field is growing rapidly, due to the promise of reduced costs and expedited approval schedules1 . Unsurprisingly, the number of publications in PubMed with the text ‘drug repositioning’ in their abstracts has ballooned from only 11 articles per year in 2007 to 274 in 2015. Prevalent among repositioning publications are computational methods, which perform in silico experiments to determine the most promising repositioning candidates for further preclinical testing2,3. Computational repositioning methods have been developed that use a variety of direct and indirect evidence for hypothesis generation, including molecular4–7 , literature-derived8–11, and clinical12,13 data. In many computational repositioning methods papers, authors claim that their methods are analytically validated in some way. Authors typically present either case studies, in which they describe a single wellsupported example, or sensitivity- and specificity-based analyses to support their claims. Sensitivity- and specificity-based methods rely on comparing the full spectrum of predictions made by a repositioning method to currently approved or investigational drug-indication pairs14. It is difficult, however, to directly compare and/or independently assess computational methods or reported new repositioning candidates due to the variety of analytic validation methodologies favored by different groups (or impossible if only case studies are provided). Furthermore, studies that claim to use unbiased validation methods like predictive ‘area under the receiver-operator curve’ (AUROC)15 rely on true, approved drug-indication pairs only, and typically assume that all other drug-indication pairs are false. This assumption is unsatisfying because it relies on a database of approved drugs (the choice of which varies widely in the repositioning literature14), and suggests that all novel repositioning predictions are false. To address these concerns, we present repoDB, a database of approved and failed drugs and their indications. repoDB approved indications were drawn from DrugCentral, which contains United Medical Language System (UMLS) indications mapped from free-text mentions in drug labels16,17. The UMLS is a large biomedical thesaurus that contains information about a wide variety of medical concepts16. Failed indications were drawn from the American Association of Clinical Trials Database (the ‘AACT Database’, Clinical Trials Transformation Initiative, 2016), which contains structured records from the National Library of Medicine’s ClinicalTrials.gov database service. Indications in the AACT database are again annotated using medical subject heading (MeSH) terms (a subset of UMLS terms), and represent a mix of investigator-submitted and automatically extracted annotations (see Table 1 for database characteristics and Fig. 1a for an overview of our methodology)18. repoDB spans 1,571 drugs and 2,051 UMLS disease concepts, accounting for 6,677 approved and 4,123 failed drug-indication pairs (see Table 2 and Fig. 1b for trial status breakdown). To further assist investigators, we provide a web application (http://apps. chiragjpgroup.org/repoDB/) that enables browsing of the full repoDB database and allows users to download either the full database (or portions relevant to their work). repoDB will enable investigators to not only benchmark their computational repositioning methods, but also gain insight into trends in the drug discovery field and avenues that have not yet been explored.

**IN “CALLAHAN, A., CRUZ-TOLEDO, J., ANSELL, P., DUMONTIER, M.: BIO2RDF RELEASE 2: IMPROVED COVERAGE, INTEROPERABILITY AND PROVENANCE OF LIFE SCIENCE LINKED DATA. IN: EXTENDED SEMANTIC WEB CONFERENCE. PP. 200–212. SPRINGER (2013)”** Bio2RDF currently provides the largest network of Linked Data for the Life Sciences. Here, we describe a significant update to increase the overall quality of RDFized datasets generated from open scripts powered by an API to generate registry-validated IRIs, dataset provenance and metrics, SPARQL endpoints, downloadable RDF and database files. We demonstrate federated SPARQL queries within and across the Bio2RDF network, including semantic integration using the Semanticscience Integrated Ontology (SIO). This work forms a strong foundation for increased coverage and continuous integration of data in the life sciences. With the advent of the World Wide Web, journals have increasingly augmented their peer-reviewed journal publications with downloadable experimental data. While the increase in data availability should be cause for celebration, the potential for biomedical discovery across all of these data is hampered by access restrictions, incompatible formats, lack of semantic annotation and poor connectivity between datasets [1]. Although organizations such as the National Center for Biotechnology Information (NCBI) and the European Bioinformatics Institute (EBI) have made great strides to extract, capture and integrate data, the lack of formal, machine-understandable semantics results in ambiguity in the data and the relationships between them. With over 1500 biological databases, it becomes necessary to implement a more sophisticated scheme to unify the representation of diverse biomedical data so that it becomes easier to integrate and explore [2]. Importantly, there is a fundamental need to capture the provenance of these data in a manner that will support experimental design and reproducibility in scientific research. Providing data also presents real practical challenges, including ensuring persistence, availability, scalability, and providing the right tools to facilitate data exploration including query formulation. The Resource Description Framework (RDF) provides an excellent foundation to build a unified network of linked data on the emerging Semantic Web. While an increasing number of approaches are being proposed to describe and integrate specific biological data [3-5], it is the lack of coordinated identification, vocabulary overlap and alternative formalizations that challenges the promise of large-scale integration [6]. Formalization of data into ontologies using the Web Ontology Language (OWL) have yielded interesting results for integration, classification, consistency checking and more effective query answering with automated reasoning [7-11]. However, these efforts build the ontology in support of the task and there is little guarantee that the formalization will accommodate future data or support new applications. Alternatively, integration of data may be best facilitated by independent publication of datasets and their descriptions and subsequent coordination into integrative ontologies or community standards. This approach provides maximum flexibility for publishing original datasets with publisher provided descriptors in that they are not constrained by limited standards, but provides a clear avenue for future integration into a number of alternative standards. Bio2RDF is a well-recognized open-source project that provides linked data for the life sciences using Semantic Web technologies. Bio2RDF scripts convert heterogeneously formatted data (e.g. flat-files, tab-delimited files, dataset specific formats, SQL, XML etc.) into a common format – RDF. Bio2RDF follows a set of basic conventions to generate and provide Linked Data which are guided by Tim Berners-Lee’s design principles 1 , the Banff Manifesto 2 and the collective experience of the Bio2RDF community. Entities, their attributes and relationships are named using a simple convention to produce Internationalized Resource Identifiers (IRIs) while statements are articulated using the lightweight semantics of RDF Schema (RDFS) and Dublin Core. Bio2RDF IRIs are resolved through the Bio2RDF Web Application, a servlet that answers Bio2RDF HTTP requests by formulating SPARQL queries against the appropriate SPARQL endpoints. Although several efforts for provisioning linked life data exist such as Neurocommons [12], LinkedLifeData [13], W3C HCLS3 , Chem2Bio2RDF [14] and BioLOD, Bio2RDF stands out for several reasons: i) Bio2RDF is open source and freely available to use, modify or redistribute, ii) it acts on a set of basic guidelines to produce syntactically interoperable linked data across all datasets, iii) does not attempt to marshal data into a single global schema, iv) provides a federated network of SPARQL endpoints and v) provisions the community with an expandable global network of mirrors that host RDF datasets. Thus, Bio2RDF uniquely offers a community-focused resource for creating and enhancing the quality of biomedical data on the Semantic Web. Here, we report on a second coordinated release of Bio2RDF Release 2 (R2), which yields substantial increases in syntactic and semantic interoperability across refactored Bio2RDF datasets. We address the problem of IRI inconsistency arising from independently generated scripts through an API over a dataset registry to generate validated IRIs. We further generate provenance and statistics for each dataset, and provide public SPARQL endpoints, downloadable database files and RDF files. We demonstrate federated SPARQL queries within and across the Bio2RDF network, including queries that make use of the Semanticscience Integrated Ontology (SIO) 4, which provides a simple model with a rich set of relations to coordinate ontologies, data and services.

**EXISTING SYSTEM**

Since the arrival of advanced computing, the doctors’ still requires the technology in various possible ways like surgical representation process and x-ray photography, but the technology perceptually stayed behind. The method still requires the doctor’s information and experience due to alternative factors starting from medical records to weather conditions, atmosphere, blood pressure and numerous alternative factors. The huge numbers of variables are consider as entire variables that are required to understand the complete working process itself, however no model has analyzed successfully. To tackle this drawback, Medical decision support systems must be used. This system is able to assist the doctors to make the correct decision. Medical decision support system refers to both the process of attempting to determine or identify possible diseases or disorder and the opinion reached by this process. The diagnostic opinion in the sense, it indicates either degree of abnormality on a continuum or a kind of abnormality in a classification. It’s influenced by non medical factors such as power ethics and financial incentives for patient or doctor. It can be a brief summation or an extensive formulation, even taking the form of story or metaphor. It might be a means of communication such as computer code through which it triggers payment, prescription, notification, information or advice. Indication of medical diagnostic includes knowledge of what is normal and measuring of patient’s current condition. Automated decision support systems are rule based systems that are automatically providing solutions to repetitive management problems.

**PROPOSED SYSTEM**

Medical decision could be extremely specialized and difficult job due to alternative factors or incase of rare diseases. The alternative factors include stress; tired misdiagnosis might vary from ignorance of doctors and incomplete information. Standard algorithm may go through the entire variables like prevailing conditions history of medical records, history of family records and various factors relating to the patient records, sheer magnitude of obtainable hidden factors. Differential diagnosis methods can be used to identify the presence of an entity where multiple alternatives are possible and also refers to include the candidate alternatives. This method is needs a process of elimination or obtaining information that shrinks the probability of candidate conditions to negligible levels. It contains four steps: 1) The doctor gather all information about the patients and create a symptoms list.2) The doctor should make a list of all possible causes of symptoms.3) The doctor should prioritize the list by which is the most dangerous possible cause of symptoms put in the top of the list. 4) The doctor should rule out or treat the possible causes beginning with the most urgently dangerous conditions.”Rule Out” in the sense to use the test method or other scientific method. If there will be no such diagnosis means removing the diagnosis from the list and using tests that should have distinct results, depends on which diagnosis is correct. This can be done based on the doctor’s knowledge and experience. This method is very easy to implement.To reduce the large number of variables and find the most probable diseases by using the K-Means algorithm. This algorithm is more suitable to cluster the more number of diseases. K-Mean is one of the unsupervised learning algorithms which are used to solve the clustering problem. The main idea is to determine the k centroids, one for each cluster. Different tests performed on the patients will served as a attributes for clustering. By using this algorithm it reduce the number of iterations, boundries of clusters are well define without overlapping, to produce the accurate result for each and every diagnosis. This system uses Service oriented architecture (SOA), anyone can access with internet connections and LAMSTAR Network can be used to calculate the weight, to increase the accuracy of algorithm, overall speed test and produce the better result.

**REQUIREMENT ANALYSIS**

The project involved analyzing the design of few applications so as to make the application more users friendly. To do so, it was really important to keep the navigations from one screen to the other well ordered and at the same time reducing the amount of typing the user needs to do. In order to make the application more accessible, the browser version had to be chosen so that it is compatible with most of the Browsers.

**REQUIREMENT SPECIFICATION**

**FUNCTIONAL REQUIREMENTS**

* Data Collection
* Data Preprocessing
* Training And Testing
* Modiling
* Predicting

**NON FUNCTIONAL REQUIREMENTS**

NON-FUNCTIONAL REQUIREMENT (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system. Example of nonfunctional requirement, “how fast does the website load?” Failing to meet non-functional requirements can result in systems that fail to satisfy user needs. Non- functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are > 10000. Description of non-functional requirements is just as critical as a functional requirement.

* Usability requirement
* Serviceability requirement
* Manageability requirement
* Recoverability requirement
* Security requirement
* Data Integrity requirement
* Capacity requirement
* Availability requirement
* Scalability requirement
* Interoperability requirement
* Reliability requirement
* Maintainability requirement
* Regulatory requirement
* Environmental requirement

Here, are some examples of non-functional requirement:

* Users must upload dataset
* The software should be portable. So moving from one OS to other OS does not create any problem.
* Privacy of information, the export of restricted technologies, intellectual property rights, etc. should be audited.

**ADVANTAGES OF NON-FUNCTIONAL REQUIREMENT**

Benefits/pros of Non-functional testing are:

* The nonfunctional requirements ensure the software system follow legal and compliance rules.
* They ensure the reliability, availability, and performance of the software system
* They ensure good user experience and ease of operating the software.
* They help in formulating security policy of the software system.

**DISADVANTAGES OF NON-FUNCTIONAL REQUIREMENT**

Cons/drawbacks of Non-function requirement are:

* None functional requirement may affect the various high-level software subsystem
* They require special consideration during the software architecture/high-level design phase which increases costs.
* Their implementation does not usually map to the specific software sub-system,
* It is tough to modify non-functional once you pass the architecture phase.

**Functional Requirements**

* Graphical User interface with the User.

**Software Requirements**

For developing the application the following are the Software Requirements:

1. Python
2. Django

**Operating Systems supported**

1. Windows 7
2. Windows XP
3. Windows 8

**Technologies and Languages used to Develop**

1. Python

**Debugger and Emulator**

* Any Browser (Particularly Chrome)

**Hardware Requirements**

For developing the application the following are the Hardware Requirements:

* Processor: Pentium IV or higher
* RAM: 256 MB
* Space on Hard Disk: minimum 512MB

**SYSTEM SPECIFICATION:**

**HARDWARE REQUIREMENTS:**

* **System :** Pentium IV 2.4 GHz.
* **Hard Disk :** 40 GB.
* **Floppy Drive :** 1.44 Mb.
* **Monitor** : 14’ Colour Monitor.
* **Mouse :** Optical Mouse.
* **Ram :** 512 Mb.

**SOFTWARE REQUIREMENTS:**

* **Operating system :** Windows 7 Ultimate.
* **Coding Language :** Python.
* **Front-End :** Python.
* **Designing :** Html,css,javascript.
* **Data Base :** MySQL.

**SYSTEM STUDY**

**FEASIBILITY STUDY**

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

**Three key considerations involved in the feasibility analysis are,**

* **ECONOMICAL FEASIBILITY**
* **TECHNICAL FEASIBILITY**
* **SOCIAL FEASIBILITY**

**ECONOMICAL FEASIBILITY**

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

### TECHNICAL FEASIBILITY

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

**SOCIAL FEASIBILITY**

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

**INPUT AND OUTPUT DESIGN**

**INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**OBJECTIVES**

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the
* Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.

**PYTHON**

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An [interpreted language](https://en.wikipedia.org/wiki/Interpreted_language), Python has a design philosophy that emphasizes code [readability](https://en.wikipedia.org/wiki/Readability) (notably using [whitespace](https://en.wikipedia.org/wiki/Whitespace_character) indentation to delimit [code blocks](https://en.wikipedia.org/wiki/Code_block) rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer [lines of code](https://en.wikipedia.org/wiki/Source_lines_of_code) than might be used in languages such as [C++](https://en.wikipedia.org/wiki/C%2B%2B)or [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many [operating systems](https://en.wikipedia.org/wiki/Operating_system). [CPython](https://en.wikipedia.org/wiki/CPython" \o "CPython), the [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) of Python, is [open source](https://en.wikipedia.org/wiki/Open_source) software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit [Python Software Foundation](https://en.wikipedia.org/wiki/Python_Software_Foundation). Python features a [dynamic type](https://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory management](https://en.wikipedia.org/wiki/Memory_management). It supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [functional](https://en.wikipedia.org/wiki/Functional_programming) and [procedural](https://en.wikipedia.org/wiki/Procedural_programming), and has a large and comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library).

**Interactive Mode Programming**

Invoking the interpreter without passing a script file as a parameter brings up the following prompt −

$ python

Python 2.4.3 (#1, Nov 11 2010, 13:34:43)

[GCC 4.1.2 20080704 (Red Hat 4.1.2-48)] on linux2

Type "help", "copyright", "credits" or "license" for more information.

>>>

Type the following text at the Python prompt and press the Enter −

>>> print "Hello, Python!"

If you are running new version of Python, then you would need to use print statement with parenthesis as in print ("Hello, Python!");. However in Python version 2.4.3, this produces the following result −

Hello, Python!

**Script Mode Programming**

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension .py. Type the following source code in a test.py file −

Live Demo

print "Hello, Python!"

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows −

$ python test.py

This produces the following result −

Hello, Python!

Let us try another way to execute a Python script. Here is the modified test.py file −

Live Demo

#!/usr/bin/python

print "Hello, Python!"

We assume that you have Python interpreter available in /usr/bin directory. Now, try to run this program as follows −

$ chmod +x test.py # This is to make file executable

$./test.py

This produces the following result −

Hello, Python!

**Python Identifiers**

A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore (\_) followed by zero or more letters, underscores and digits (0 to 9).

Python does not allow punctuation characters such as @, $, and % within identifiers. Python is a case sensitive programming language. Thus, Manpower and manpower are two different identifiers in Python.

Here are naming conventions for Python identifiers −

Class names start with an uppercase letter. All other identifiers start with a lowercase letter.

Starting an identifier with a single leading underscore indicates that the identifier is private.

Starting an identifier with two leading underscores indicates a strongly private identifier.

If the identifier also ends with two trailing underscores, the identifier is a language-defined special name.

**Reserved Words**

The following list shows the Python keywords. These are reserved words and you cannot use them as constant or variable or any other identifier names. All the Python keywords contain lowercase letters only.

and exec not

assert finally or

break for pass

class from print

continue global raise

def if return

del import try

elif in while

else is with

except lambda yield

**Lines and Indentation**

Python provides no braces to indicate blocks of code for class and function definitions or flow control. Blocks of code are denoted by line indentation, which is rigidly enforced.

The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount. For example −

if True:

print "True"

else:

print "False"

However, the following block generates an error −

if True:

print "Answer"

print "True"

else:

print "Answer"

print "False"

Thus, in Python all the continuous lines indented with same number of spaces would form a block. The following example has various statement blocks −

Note − Do not try to understand the logic at this point of time. Just make sure you understood various blocks even if they are without braces.

#!/usr/bin/python

import sys

try:

# open file stream

file = open(file\_name, "w")

except IOError:

print "There was an error writing to", file\_name

sys.exit()

print "Enter '", file\_finish,

print "' When finished"

while file\_text != file\_finish:

file\_text = raw\_input("Enter text: ")

if file\_text == file\_finish:

# close the file

file.close

break

file.write(file\_text)

file.write("\n")

file.close()

file\_name = raw\_input("Enter filename: ")

if len(file\_name) == 0:

print "Next time please enter something"

sys.exit()

try:

file = open(file\_name, "r")

except IOError:

print "There was an error reading file"

sys.exit()

file\_text = file.read()

file.close()

print file\_text

Multi-Line Statements

Statements in Python typically end with a new line. Python does, however, allow the use of the line continuation character (\) to denote that the line should continue. For example −

total = item\_one + \

item\_two + \

item\_three

Statements contained within the [], {}, or () brackets do not need to use the line continuation character. For example −

days = ['Monday', 'Tuesday', 'Wednesday',

'Thursday', 'Friday']

Quotation in Python

Python accepts single ('), double (") and triple (''' or """) quotes to denote string literals, as long as the same type of quote starts and ends the string.

The triple quotes are used to span the string across multiple lines. For example, all the following are legal −

word = 'word'

sentence = "This is a sentence."

paragraph = """This is a paragraph. It is

made up of multiple lines and sentences."""

Comments in Python

A hash sign (#) that is not inside a string literal begins a comment. All characters after the # and up to the end of the physical line are part of the comment and the Python interpreter ignores them.

Live Demo

#!/usr/bin/python

# First comment

print "Hello, Python!" # second comment

This produces the following result −

Hello, Python!

You can type a comment on the same line after a statement or expression −

name = "Madisetti" # This is again comment

You can comment multiple lines as follows −

# This is a comment.

# This is a comment, too.

# This is a comment, too.

# I said that already.

Following triple-quoted string is also ignored by Python interpreter and can be used as a multiline comments:

'''

This is a multiline

comment.

'''

Using Blank Lines

A line containing only whitespace, possibly with a comment, is known as a blank line and Python totally ignores it.

In an interactive interpreter session, you must enter an empty physical line to terminate a multiline statement.

Waiting for the User

The following line of the program displays the prompt, the statement saying “Press the enter key to exit”, and waits for the user to take action −

#!/usr/bin/python

raw\_input("\n\nPress the enter key to exit.")

Here, "\n\n" is used to create two new lines before displaying the actual line. Once the user presses the key, the program ends. This is a nice trick to keep a console window open until the user is done with an application.

Multiple Statements on a Single Line

The semicolon ( ; ) allows multiple statements on the single line given that neither statement starts a new code block. Here is a sample snip using the semicolon.

import sys; x = 'foo'; sys.stdout.write(x + '\n')

Multiple Statement Groups as Suites

A group of individual statements, which make a single code block are called suites in Python. Compound or complex statements, such as if, while, def, and class require a header line and a suite.

Header lines begin the statement (with the keyword) and terminate with a colon ( : ) and are followed by one or more lines which make up the suite. For example −

if expression :

suite

elif expression :

suite

else :

suite

**Command Line Arguments**

Many programs can be run to provide you with some basic information about how they should be run. Python enables you to do this with -h −

$ python -h

usage: python [option] ... [-c cmd | -m mod | file | -] [arg] ...

Options and arguments (and corresponding environment variables):

-c cmd : program passed in as string (terminates option list)

-d : debug output from parser (also PYTHONDEBUG=x)

-E : ignore environment variables (such as PYTHONPATH)

-h : print this help message and exit

You can also program your script in such a way that it should accept various options. Command Line Arguments is an advanced topic and should be studied a bit later once you have gone through rest of the Python concepts.

**Python Lists**

The list is a most versatile datatype available in Python which can be written as a list of comma-separated values (items) between square brackets. Important thing about a list is that items in a list need not be of the same type.

Creating a list is as simple as putting different comma-separated values between square brackets. For example −

list1 = ['physics', 'chemistry', 1997, 2000];

list2 = [1, 2, 3, 4, 5 ];

list3 = ["a", "b", "c", "d"]

Similar to string indices, list indices start at 0, and lists can be sliced, concatenated and so on.

A tuple is a sequence of immutable Python objects. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.

Creating a tuple is as simple as putting different comma-separated values. Optionally you can put these comma-separated values between parentheses also. For example −

tup1 = ('physics', 'chemistry', 1997, 2000);

tup2 = (1, 2, 3, 4, 5 );

tup3 = "a", "b", "c", "d";

The empty tuple is written as two parentheses containing nothing −

tup1 = ();

To write a tuple containing a single value you have to include a comma, even though there is only one value −

tup1 = (50,);

Like string indices, tuple indices start at 0, and they can be sliced, concatenated, and so on.

Accessing Values in Tuples

To access values in tuple, use the square brackets for slicing along with the index or indices to obtain value available at that index. For example −

Live Demo

#!/usr/bin/python

tup1 = ('physics', 'chemistry', 1997, 2000);

tup2 = (1, 2, 3, 4, 5, 6, 7 );

print "tup1[0]: ", tup1[0];

print "tup2[1:5]: ", tup2[1:5];

When the above code is executed, it produces the following result −

tup1[0]: physics

tup2[1:5]: [2, 3, 4, 5]

Updating Tuples

Accessing Values in Dictionary

To access dictionary elements, you can use the familiar square brackets along with the key to obtain its value. Following is a simple example −

Live Demo

#!/usr/bin/python

dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

print "dict['Name']: ", dict['Name']

print "dict['Age']: ", dict['Age']

When the above code is executed, it produces the following result −

dict['Name']: Zara

dict['Age']: 7

If we attempt to access a data item with a key, which is not part of the dictionary, we get an error as follows −

Live Demo

#!/usr/bin/python

dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

print "dict['Alice']: ", dict['Alice']

When the above code is executed, it produces the following result −

dict['Alice']:

Traceback (most recent call last):

File "test.py", line 4, in <module>

print "dict['Alice']: ", dict['Alice'];

KeyError: 'Alice'

Updating Dictionary

You can update a dictionary by adding a new entry or a key-value pair, modifying an existing entry, or deleting an existing entry as shown below in the simple example −

Live Demo

#!/usr/bin/python

dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

dict['Age'] = 8; # update existing entry

dict['School'] = "DPS School"; # Add new entry

print "dict['Age']: ", dict['Age']

print "dict['School']: ", dict['School']

When the above code is executed, it produces the following result −

dict['Age']: 8

dict['School']: DPS School

Delete Dictionary Elements

You can either remove individual dictionary elements or clear the entire contents of a dictionary. You can also delete entire dictionary in a single operation.

To explicitly remove an entire dictionary, just use the del statement. Following is a simple example −

Live Demo

#!/usr/bin/python

dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}

del dict['Name']; # remove entry with key 'Name'

dict.clear(); # remove all entries in dict

del dict ; # delete entire dictionary

print "dict['Age']: ", dict['Age']

print "dict['School']: ", dict['School']

This produces the following result. Note that an exception is raised because after del dict dictionary does not exist any more −

dict['Age']:

Traceback (most recent call last):

File "test.py", line 8, in <module>

print "dict['Age']: ", dict['Age'];

TypeError: 'type' object is unsubscriptable

Note − del() method is discussed in subsequent section.

**Properties of Dictionary Keys**

Dictionary values have no restrictions. They can be any arbitrary Python object, either standard objects or user-defined objects. However, same is not true for the keys.

There are two important points to remember about dictionary keys −

(a) More than one entry per key not allowed. Which means no duplicate key is allowed. When duplicate keys encountered during assignment, the last assignment wins. For example −

Live Demo

#!/usr/bin/python

dict = {'Name': 'Zara', 'Age': 7, 'Name': 'Manni'}

print "dict['Name']: ", dict['Name']

When the above code is executed, it produces the following result −

dict['Name']: Manni

(b) Keys must be immutable. Which means you can use strings, numbers or tuples as dictionary keys but something like ['key'] is not allowed. Following is a simple example −

Live Demo

#!/usr/bin/python

dict = {['Name']: 'Zara', 'Age': 7}

print "dict['Name']: ", dict['Name']

When the above code is executed, it produces the following result −

Traceback (most recent call last):

File "test.py", line 3, in <module>

dict = {['Name']: 'Zara', 'Age': 7};

TypeError: unhashable type: 'list'

Tuples are immutable which means you cannot update or change the values of tuple elements. You are able to take portions of existing tuples to create new tuples as the following example demonstrates −

Live Demo

#!/usr/bin/python

tup1 = (12, 34.56);

tup2 = ('abc', 'xyz');

# Following action is not valid for tuples

# tup1[0] = 100;

# So let's create a new tuple as follows

tup3 = tup1 + tup2;

print tup3;

When the above code is executed, it produces the following result −

(12, 34.56, 'abc', 'xyz')

Delete Tuple Elements

Removing individual tuple elements is not possible. There is, of course, nothing wrong with putting together another tuple with the undesired elements discarded.

To explicitly remove an entire tuple, just use the del statement. For example −

Live Demo

#!/usr/bin/python

tup = ('physics', 'chemistry', 1997, 2000);

print tup;

del tup;

print "After deleting tup : ";

print tup;

This produces the following result. Note an exception raised, this is because after del tup tuple does not exist any more −

('physics', 'chemistry', 1997, 2000)

After deleting tup :

Traceback (most recent call last):

File "test.py", line 9, in <module>

print tup;

NameError: name 'tup' is not defined

**DJANGO**

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes [reusability](https://en.wikipedia.org/wiki/Reusability" \o "Reusability)and "pluggability" of components, rapid development, and the principle of [don't repeat yourself](https://en.wikipedia.org/wiki/Don%27t_repeat_yourself). Python is used throughout, even for settings files and data models.



Django also provides an optional administrative [create, read, update and delete](https://en.wikipedia.org/wiki/Create,_read,_update_and_delete) interface that is generated dynamically through [introspection](https://en.wikipedia.org/wiki/Introspection_(computer_science)) and configured via admin models



**IMPLEMENTATION**

Class Diagram



Use Case Diagram



Sequence Diagram



Collaboration Diagram



**ALGORITHMS**

# **DECISION TREE CLASSIFICATION ALGORITHM**

* Decision Tree is a **Supervised learning technique**that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where**internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
* In a Decision tree, there are two nodes, which are the **Decision Node** and**Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
* The decisions or the test are performed on the basis of features of the given dataset.
* It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
* It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
* In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
* A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
* Below diagram explains the general structure of a decision tree:

#### **Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.**



## Why use Decision Trees?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

* Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
* The logic behind the decision tree can be easily understood because it shows a tree-like structure.

## Decision Tree Terminologies

 **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.

 **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

 **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.

 **Branch/Sub Tree:** A tree formed by splitting the tree.

 **Pruning:** Pruning is the process of removing the unwanted branches from the tree.

 **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

**How does the Decision Tree algorithm Work?**

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

* **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
* **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
* **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
* **Step-4:** Generate the decision tree node, which contains the best attribute.
* **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

**Example:** Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:



## Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM.**By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

* **Information Gain**
* **Gini Index**

### 1. Information Gain:

* Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
* It calculates how much information a feature provides us about a class.
* According to the value of information gain, we split the node and build the decision tree.
* A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

1. Information Gain= Entropy(S)- [(Weighted Avg) \*Entropy(each feature)

**Entropy:** Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

**Where,**

* **S= Total number of samples**
* **P(yes)= probability of yes**
* **P(no)= probability of no**

### 2. Gini Index:

* Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
* An attribute with the low Gini index should be preferred as compared to the high Gini index.
* It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
* Gini index can be calculated using the below formula:

Gini Index= 1- ∑jPj2

## Pruning: Getting an Optimal Decision tree

Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.

A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset. Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as Pruning. There are mainly two types of tree **pruning**technology used:

* **Cost Complexity Pruning**
* **Reduced Error Pruning.**

## Advantages of the Decision Tree

* It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
* It can be very useful for solving decision-related problems.
* It helps to think about all the possible outcomes for a problem.
* There is less requirement of data cleaning compared to other algorithms.

## Disadvantages of the Decision Tree

* The decision tree contains lots of layers, which makes it complex.
* It may have an overfitting issue, which can be resolved using the **Random Forest algorithm.**
* For more class labels, the computational complexity of the decision tree may increase.

## Python Implementation of Decision Tree

Now we will implement the Decision tree using Python. For this, we will use the dataset "**user\_data.csv**," which we have used in previous classification models. By using the same dataset, we can compare the Decision tree classifier with other classification models such as [KNN](https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning) [SVM,](https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm) [LogisticRegression,](https://www.javatpoint.com/logistic-regression-in-machine-learning) etc.

Steps will also remain the same, which are given below:

* **Data Pre-processing step**
* **Fitting a Decision-Tree algorithm to the Training set**
* **Predicting the test result**
* **Test accuracy of the result(Creation of Confusion matrix)**
* **Visualizing the test set result.**

**RANDOM FOREST ALGORITHM**

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

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

**The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.**

The below diagram explains the working of the Random Forest algorithm:



Note: To better understand the Random Forest Algorithm, you should have knowledge of the Decision Tree Algorithm.

Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

* There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
* The predictions from each tree must have very low correlations.

Why use Random Forest?

Below are some points that explain why we should use the Random Forest algorithm:

<="" li="">

* It takes less training time as compared to other algorithms.
* It predicts output with high accuracy, even for the large dataset it runs efficiently.
* It can also maintain accuracy when a large proportion of data is missing.

How does Random Forest algorithm work?

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

**Step-1:** Select random K data points from the training set.

**Step-2:** Build the decision trees associated with the selected data points (Subsets).

**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4:** Repeat Step 1 & 2.

**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

The working of the algorithm can be better understood by the below example:

**Example:** Suppose there is a dataset that contains multiple fruit images. So, this dataset is given to the Random forest classifier. The dataset is divided into subsets and given to each decision tree. During the training phase, each decision tree produces a prediction result, and when a new data point occurs, then based on the majority of results, the Random Forest classifier predicts the final decision. Consider the below image:



Applications of Random Forest

There are mainly four sectors where Random forest mostly used:

1. **Banking:** Banking sector mostly uses this algorithm for the identification of loan risk.
2. **Medicine:** With the help of this algorithm, disease trends and risks of the disease can be identified.
3. **Land Use:** We can identify the areas of similar land use by this algorithm.
4. **Marketing:** Marketing trends can be identified using this algorithm.

Advantages of Random Forest

* Random Forest is capable of performing both Classification and Regression tasks.
* It is capable of handling large datasets with high dimensionality.
* It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages of Random Forest

* Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

**NAVIE BAYES**

Naive Bayes classifiers are a collection of classification algorithms based on **Bayes’ Theorem**. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

To start with, let us consider a dataset.

Consider a fictional dataset that describes the weather conditions for playing a game of golf. Given the weather conditions, each tuple classifies the conditions as fit(“Yes”) or unfit(“No”) for plaing golf.

Here is a tabular representation of our dataset.

|  | Outlook | Temperature | Humidity | Windy | Play Golf |
| --- | --- | --- | --- | --- | --- |
| 0 | Rainy | Hot | High | False | No |
| 1 | Rainy | Hot | High | True | No |
| 2 | Overcast | Hot | High | False | Yes |
| 3 | Sunny | Mild | High | False | Yes |
| 4 | Sunny | Cool | Normal | False | Yes |
| 5 | Sunny | Cool | Normal | True | No |
| 6 | Overcast | Cool | Normal | True | Yes |
| 7 | Rainy | Mild | High | False | No |
| 8 | Rainy | Cool | Normal | False | Yes |
| 9 | Sunny | Mild | Normal | False | Yes |
| 10 | Rainy | Mild | Normal | True | Yes |
| 11 | Overcast | Mild | High | True | Yes |
| 12 | Overcast | Hot | Normal | False | Yes |
| 13 | Sunny | Mild | High | True | No |

The dataset is divided into two parts, namely, **feature matrix** and the **response vector**.

* Feature matrix contains all the vectors(rows) of dataset in which each vector consists of the value of **dependent features**. In above dataset, features are ‘Outlook’, ‘Temperature’, ‘Humidity’ and ‘Windy’.
* Response vector contains the value of **class variable**(prediction or output) for each row of feature matrix. In above dataset, the class variable name is ‘Play golf’.

**Assumption:**

The fundamental Naive Bayes assumption is that each feature makes an:

* independent
* equal

contribution to the outcome.

With relation to our dataset, this concept can be understood as:

* We assume that no pair of features are dependent. For example, the temperature being ‘Hot’ has nothing to do with the humidity or the outlook being ‘Rainy’ has no effect on the winds. Hence, the features are assumed to be **independent**.
* Secondly, each feature is given the same weight(or importance). For example, knowing only temperature and humidity alone can’t predict the outcome accuratey. None of the attributes is irrelevant and assumed to be contributing **equally** to the outcome.

**Note:** The assumptions made by Naive Bayes are not generally correct in real-world situations. In-fact, the independence assumption is never correct but often works well in practice.

Now, before moving to the formula for Naive Bayes, it is important to know about Bayes’ theorem.

**Bayes’ Theorem**

Bayes’ Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes’ theorem is stated mathematically as the following equation:



where A and B are events and P(B) ? 0.

* Basically, we are trying to find probability of event A, given the event B is true. Event B is also termed as **evidence**.
* P(A) is the **priori** of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance(here, it is event B).
* P(A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.

Now, with regards to our dataset, we can apply Bayes’ theorem in following way:



where, y is class variable and X is a dependent feature vector (of size *n*) where:



Just to clear, an example of a feature vector and corresponding class variable can be: (refer 1st row of dataset)

X = (Rainy, Hot, High, False)

y = No

So basically, P(y|X) here means, the probability of “Not playing golf” given that the weather conditions are “Rainy outlook”, “Temperature is hot”, “high humidity” and “no wind”.

**Naive assumption**

Now, its time to put a naive assumption to the Bayes’ theorem, which is, **independence** among the features. So now, we split **evidence** into the independent parts.

Now, if any two events A and B are independent, then,

P(A,B) = P(A)P(B)

Hence, we reach to the result:



which can be expressed as:



Now, as the denominator remains constant for a given input, we can remove that term:



Now, we need to create a classifier model. For this, we find the probability of given set of inputs for all possible values of the class variable *y* and pick up the output with maximum probability. This can be expressed mathematically as:



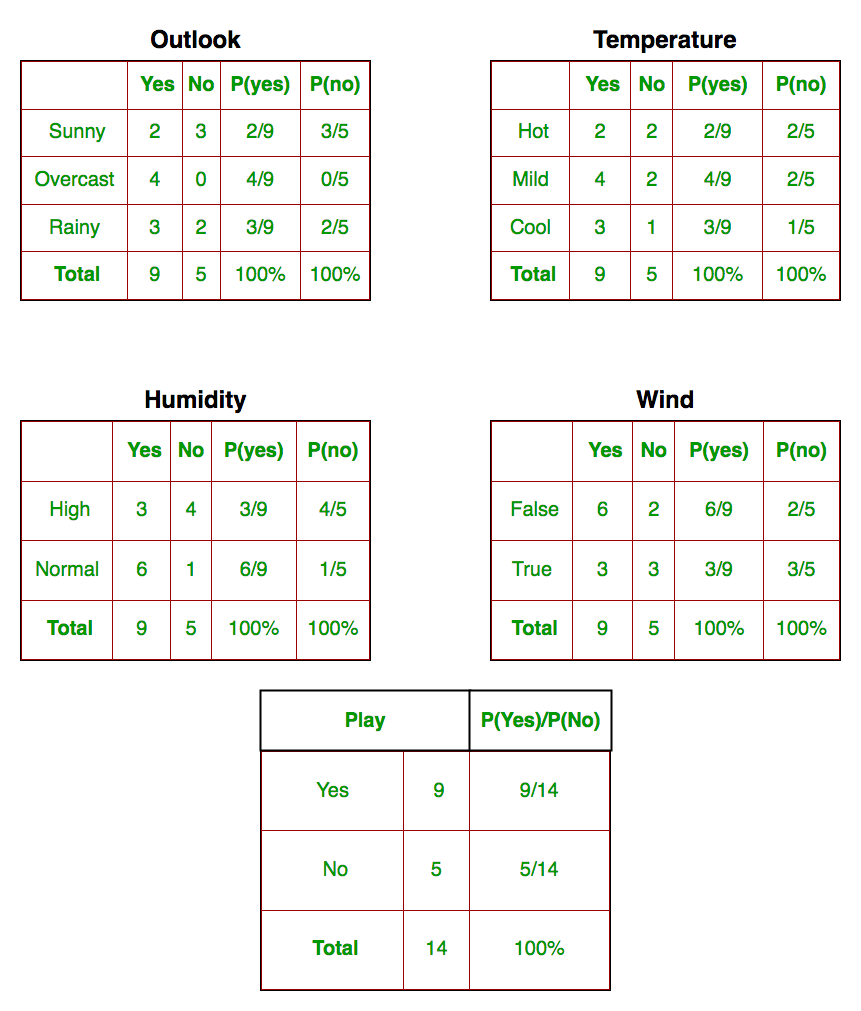
So, finally, we are left with the task of calculating P(y) and P(xi | y).

Please note that P(y) is also called **class probability** and P(xi | y) is called **conditional probability**.

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of P(xi | y).

Let us try to apply the above formula manually on our weather dataset. For this, we need to do some precomputations on our dataset.

We need to find P(xi | yj) for each xi in X and yj in y. All these calculations have been demonstrated in the tables below:



So, in the figure above, we have calculated P(xi | yj) for each xi in X and yj in y manually in

**SYSTEM TEST**

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.

**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.

# 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.

**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.

**RESULTS**

To reduce the large number of variables and find the most probable diseases by using the K-Means algorithm. This algorithm is more suitable to cluster the more number of diseases. K-Mean is one of the unsupervised learning algorithms which are used to solve the clustering problem. The main idea is to determine the k centroids, one for each cluster. Different tests performed on the patients will served as a attributes for clustering. By using this algorithm it reduce the number of iterations, boundries of clusters are well define without overlapping, to produce the accurate result for each and every diagnosis. This system uses Service oriented architecture (SOA), anyone can access with internet connections and LAMSTAR Network can be used to calculate the weight, to increase the accuracy of algorithm, overall speed test and produce the better result.

**DATASET:**

itching,

skin\_rash

nodal\_skin\_eruptions

continuous\_sneezing

shivering

chills

joint\_pain

stomach\_pain

acidity

ulcers\_on\_tongue

muscle\_wasting

vomiting

burning\_micturition

spotting\_ urination

fatigue

weight\_gain

anxiety

cold\_hands\_and\_feets

mood\_swings

weight\_loss

restlessness

lethargy

patches\_in\_throat

irregular\_sugar\_level

cough

high\_fever

sunken\_eyes

breathlessness

sweating

dehydration

indigestion

headache

yellowish\_skin

dark\_urine

nausea

loss\_of\_appetite

pain\_behind\_the\_eyes

back\_pain

constipation

abdominal\_pain

diarrhoea

mild\_fever

yellow\_urine

yellowing\_of\_eyes

acute\_liver\_failure

fluid\_overload

swelling\_of\_stomach

swelled\_lymph\_nodes

malaise

blurred\_and\_distorted\_vision

phlegm

throat\_irritation

redness\_of\_eyes

sinus\_pressure

runny\_nose

congestion

chest\_pain

weakness\_in\_limbs

fast\_heart\_rate

pain\_during\_bowel\_movements

pain\_in\_anal\_region

bloody\_stool

irritation\_in\_anus

neck\_pain

dizziness

cramps

bruising

obesity

swollen\_legs

swollen\_blood\_vessels

puffy\_face\_and\_eyes

enlarged\_thyroid

brittle\_nails

swollen\_extremeties

excessive\_hunger

extra\_marital\_contacts

drying\_and\_tingling\_lips

slurred\_speech

knee\_pain

hip\_joint\_pain

muscle\_weakness

stiff\_neck

swelling\_joints

movement\_stiffness

spinning\_movements

loss\_of\_balance

unsteadiness

weakness\_of\_one\_body\_side

loss\_of\_smell

bladder\_discomfort

foul\_smell\_of urine

continuous\_feel\_of\_urine

passage\_of\_gases

internal\_itching

toxic\_look\_(typhos)

depression

irritability

muscle\_pain

altered\_sensorium

red\_spots\_over\_body

We will predict the disease with the dataset and we have implemented to suggest medicine for diabetes and Hypertension problems.

**SOURCE CODE**

**from tkinter import \***

**import numpy as np**

**import pandas as pd**

**# from gui\_stuff import \***

**l1=['back\_pain','constipation','abdominal\_pain','diarrhoea','mild\_fever','yellow\_urine',**

**'yellowing\_of\_eyes','acute\_liver\_failure','fluid\_overload','swelling\_of\_stomach',**

**'swelled\_lymph\_nodes','malaise','blurred\_and\_distorted\_vision','phlegm','throat\_irritation',**

**'redness\_of\_eyes','sinus\_pressure','runny\_nose','congestion','chest\_pain','weakness\_in\_limbs',**

**'fast\_heart\_rate','pain\_during\_bowel\_movements','pain\_in\_anal\_region','bloody\_stool',**

**'irritation\_in\_anus','neck\_pain','dizziness','cramps','bruising','obesity','swollen\_legs',**

**'swollen\_blood\_vessels','puffy\_face\_and\_eyes','enlarged\_thyroid','brittle\_nails',**

**'swollen\_extremeties','excessive\_hunger','extra\_marital\_contacts','drying\_and\_tingling\_lips',**

**'slurred\_speech','knee\_pain','hip\_joint\_pain','muscle\_weakness','stiff\_neck','swelling\_joints',**

**'movement\_stiffness','spinning\_movements','loss\_of\_balance','unsteadiness',**

**'weakness\_of\_one\_body\_side','loss\_of\_smell','bladder\_discomfort','foul\_smell\_of urine',**

**'continuous\_feel\_of\_urine','passage\_of\_gases','internal\_itching','toxic\_look\_(typhos)',**

**'depression','irritability','muscle\_pain','altered\_sensorium','red\_spots\_over\_body','belly\_pain',**

**'abnormal\_menstruation','dischromic \_patches','watering\_from\_eyes','increased\_appetite','polyuria','family\_history','mucoid\_sputum',**

**'rusty\_sputum','lack\_of\_concentration','visual\_disturbances','receiving\_blood\_transfusion',**

**'receiving\_unsterile\_injections','coma','stomach\_bleeding','distention\_of\_abdomen',**

**'history\_of\_alcohol\_consumption','fluid\_overload','blood\_in\_sputum','prominent\_veins\_on\_calf',**

**'palpitations','painful\_walking','pus\_filled\_pimples','blackheads','scurring','skin\_peeling',**

**'silver\_like\_dusting','small\_dents\_in\_nails','inflammatory\_nails','blister','red\_sore\_around\_nose',**

**'yellow\_crust\_ooze']**

**disease=['Fungal infection-Canesten','Allergy-Desloratadine','GERD-Antacid','Chronic cholestasis-Ursodeoxycholic acid ','Drug Reaction-Antihistamines',**

**'Peptic ulcer diseae-lansoprazole','AIDS-Abacavir','Diabetes-Metformin','Gastroenteritis-Kaopectate','Bronchial Asthma-ProAir HFA','Hypertension-beta-blockers',**

**' Migraine','Cervical spondylosis',**

**'Paralysis (brain hemorrhage)-Aconite napellus','Jaundice-phenobarbital','Malaria-pyrimethamine','Chicken pox-Acyclovir','Dengue-Acetaminophen','Typhoid-Ciprofloxacin','hepatitis A-acetaminophen',**

**'Hepatitis B-tenofovir','Hepatitis C-Daclatasvir','Hepatitis D-Pegylated interferon alpha','Hepatitis E-Ribavirin','Alcoholic hepatitis-Pentoxifylline','Tuberculosis-Isoniazid',**

**'Common Cold-oxymetazoline nasa','Pneumonia-Fluoroquinolones','Dimorphic hemmorhoids(piles)-Scabindon',**

**'Heartattack-Thrombolytics','Varicoseveins-polidocanol','Hypothyroidism-Levo-T','Hyperthyroidism-Levo-T','Hypoglycemia-dextrose','Osteoarthristis-Acetaminophen',**

**'Arthritis- Adalimumab','(vertigo) Paroymsal Positional Vertigo-meclizine','Acne-Benzoyl Peroxide','Urinary tract infection-Fosfomycin','Psoriasis-Retinoids',**

**'Impetigo-mupirocin']**

**l2=[]**

**for x in range(0,len(l1)):**

**l2.append(0)**

**# TESTING DATA df -------------------------------------------------------------------------------------**

**df=pd.read\_csv("Training.csv")**

**df.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,**

**'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,**

**'Migraine':11,'Cervical spondylosis':12,**

**'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,**

**'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,**

**'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,**

**'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,**

**'(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,**

**'Impetigo':40}},inplace=True)**

**# print(df.head())**

**X= df[l1]**

**y = df[["prognosis"]]**

**np.ravel(y)**

**# print(y)**

**# TRAINING DATA tr --------------------------------------------------------------------------------**

**tr=pd.read\_csv("Testing.csv")**

**tr.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Reaction':4,**

**'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,**

**'Migraine':11,'Cervical spondylosis':12,**

**'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid':18,'hepatitis A':19,**

**'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuberculosis':25,**

**'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,**

**'Hyperthyroidism':32,'Hypoglycemia':33,'Osteoarthristis':34,'Arthritis':35,**

**'(vertigo) Paroymsal Positional Vertigo':36,'Acne':37,'Urinary tract infection':38,'Psoriasis':39,**

**'Impetigo':40}},inplace=True)**

**X\_test= tr[l1]**

**y\_test = tr[["prognosis"]]**

**np.ravel(y\_test)**

**# ------------------------------------------------------------------------------------------------------**

**def DecisionTree():**

**from sklearn import tree**

**clf3 = tree.DecisionTreeClassifier() # empty model of the decision tree**

**clf3 = clf3.fit(X,y)**

**# calculating accuracy-------------------------------------------------------------------**

**from sklearn.metrics import accuracy\_score**

**y\_pred=clf3.predict(X\_test)**

**print(accuracy\_score(y\_test, y\_pred))**

**print(accuracy\_score(y\_test, y\_pred,normalize=False))**

**# -----------------------------------------------------**

**psymptoms = [Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get()]**

**for k in range(0,len(l1)):**

**# print (k,)**

**for z in psymptoms:**

**if(z==l1[k]):**

**l2[k]=1**

**inputtest = [l2]**

**predict = clf3.predict(inputtest)**

**predicted=predict[0]**

**h='no'**

**for a in range(0,len(disease)):**

**if(predicted == a):**

**h='yes'**

**break**

**if (h=='yes'):**

**t1.delete("1.0", END)**

**t1.insert(END, disease[a])**

**else:**

**t1.delete("1.0", END)**

**t1.insert(END, "Not Found")**

**def randomforest():**

**from sklearn.ensemble import RandomForestClassifier**

**clf4 = RandomForestClassifier()**

**clf4 = clf4.fit(X,np.ravel(y))**

**# calculating accuracy-------------------------------------------------------------------**

**from sklearn.metrics import accuracy\_score**

**y\_pred=clf4.predict(X\_test)**

**print(accuracy\_score(y\_test, y\_pred))**

**print(accuracy\_score(y\_test, y\_pred,normalize=False))**

**# -----------------------------------------------------**

**psymptoms = [Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get()]**

**for k in range(0,len(l1)):**

**for z in psymptoms:**

**if(z==l1[k]):**

**l2[k]=1**

**inputtest = [l2]**

**predict = clf4.predict(inputtest)**

**predicted=predict[0]**

**h='no'**

**for a in range(0,len(disease)):**

**if(predicted == a):**

**h='yes'**

**break**

**if (h=='yes'):**

**t2.delete("1.0", END)**

**t2.insert(END, disease[a])**

**else:**

**t2.delete("1.0", END)**

**t2.insert(END, "Not Found")**

**def NaiveBayes():**

**from sklearn.naive\_bayes import GaussianNB**

**gnb = GaussianNB()**

**gnb=gnb.fit(X,np.ravel(y))**

**# calculating accuracy-------------------------------------------------------------------**

**from sklearn.metrics import accuracy\_score**

**y\_pred=gnb.predict(X\_test)**

**print(accuracy\_score(y\_test, y\_pred))**

**print(accuracy\_score(y\_test, y\_pred,normalize=False))**

**# -----------------------------------------------------**

**psymptoms = [Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get()]**

**for k in range(0,len(l1)):**

**for z in psymptoms:**

**if(z==l1[k]):**

**l2[k]=1**

**inputtest = [l2]**

**predict = gnb.predict(inputtest)**

**predicted=predict[0]**

**h='no'**

**for a in range(0,len(disease)):**

**if(predicted == a):**

**h='yes'**

**break**

**if (h=='yes'):**

**t3.delete("1.0", END)**

**t3.insert(END, disease[a])**

**else:**

**t3.delete("1.0", END)**

**t3.insert(END, "Not Found")**

**# gui\_stuff------------------------------------------------------------------------------------**

**root = Tk()**

**root.configure(background='blue')**

**# entry variables**

**Symptom1 = StringVar()**

**Symptom1.set(None)**

**Symptom2 = StringVar()**

**Symptom2.set(None)**

**Symptom3 = StringVar()**

**Symptom3.set(None)**

**Symptom4 = StringVar()**

**Symptom4.set(None)**

**Symptom5 = StringVar()**

**Symptom5.set(None)**

**Name = StringVar()**

**# Heading**

**w2 = Label(root, justify=LEFT, text="Disease Predictor using Machine Learning", fg="white", bg="blue")**

**w2.config(font=("Elephant", 30))**

**w2.grid(row=1, column=0, columnspan=2, padx=100)**

**w2 = Label(root, justify=LEFT, text="A Project by ", fg="white", bg="blue")**

**w2.config(font=("Aharoni", 30))**

**w2.grid(row=2, column=0, columnspan=2, padx=100)**

**# labels**

**NameLb = Label(root, text="Name of the Patient", fg="yellow", bg="black")**

**NameLb.grid(row=6, column=0, pady=15, sticky=W)**

**S1Lb = Label(root, text="Symptom 1", fg="yellow", bg="black")**

**S1Lb.grid(row=7, column=0, pady=10, sticky=W)**

**S2Lb = Label(root, text="Symptom 2", fg="yellow", bg="black")**

**S2Lb.grid(row=8, column=0, pady=10, sticky=W)**

**S3Lb = Label(root, text="Symptom 3", fg="yellow", bg="black")**

**S3Lb.grid(row=9, column=0, pady=10, sticky=W)**

**S4Lb = Label(root, text="Symptom 4", fg="yellow", bg="black")**

**S4Lb.grid(row=10, column=0, pady=10, sticky=W)**

**S5Lb = Label(root, text="Symptom 5", fg="yellow", bg="black")**

**S5Lb.grid(row=11, column=0, pady=10, sticky=W)**

**lrLb = Label(root, text="DecisionTree", fg="white", bg="red")**

**lrLb.grid(row=15, column=0, pady=10,sticky=W)**

**destreeLb = Label(root, text="RandomForest", fg="white", bg="red")**

**destreeLb.grid(row=17, column=0, pady=10, sticky=W)**

**ranfLb = Label(root, text="DRUG PREDCITION", fg="white", bg="red")**

**ranfLb.grid(row=19, column=0, pady=10, sticky=W)**

**# entries**

**OPTIONS = sorted(l1)**

**NameEn = Entry(root, textvariable=Name)**

**NameEn.grid(row=6, column=1)**

**S1En = OptionMenu(root, Symptom1,\*OPTIONS)**

**S1En.grid(row=7, column=1)**

**S2En = OptionMenu(root, Symptom2,\*OPTIONS)**

**S2En.grid(row=8, column=1)**

**S3En = OptionMenu(root, Symptom3,\*OPTIONS)**

**S3En.grid(row=9, column=1)**

**S4En = OptionMenu(root, Symptom4,\*OPTIONS)**

**S4En.grid(row=10, column=1)**

**S5En = OptionMenu(root, Symptom5,\*OPTIONS)**

**S5En.grid(row=11, column=1)**

**dst = Button(root, text="prediction of drug and disease by DecisionTree", command=DecisionTree,bg="green",fg="yellow")**

**dst.grid(row=8, column=3,padx=10)**

**rnf = Button(root, text="prediction of drug and disease by Randomforest", command=randomforest,bg="green",fg="yellow")**

**rnf.grid(row=9, column=3,padx=10)**

**lr = Button(root, text="prediction of drug and disease by NaiveBayes", command=NaiveBayes,bg="green",fg="yellow")**

**lr.grid(row=10, column=3,padx=10)**

**#textfileds**

**t1 = Text(root, height=1, width=40,bg="orange",fg="black")**

**t1.grid(row=15, column=1, padx=10)**

**t2 = Text(root, height=1, width=40,bg="orange",fg="black")**

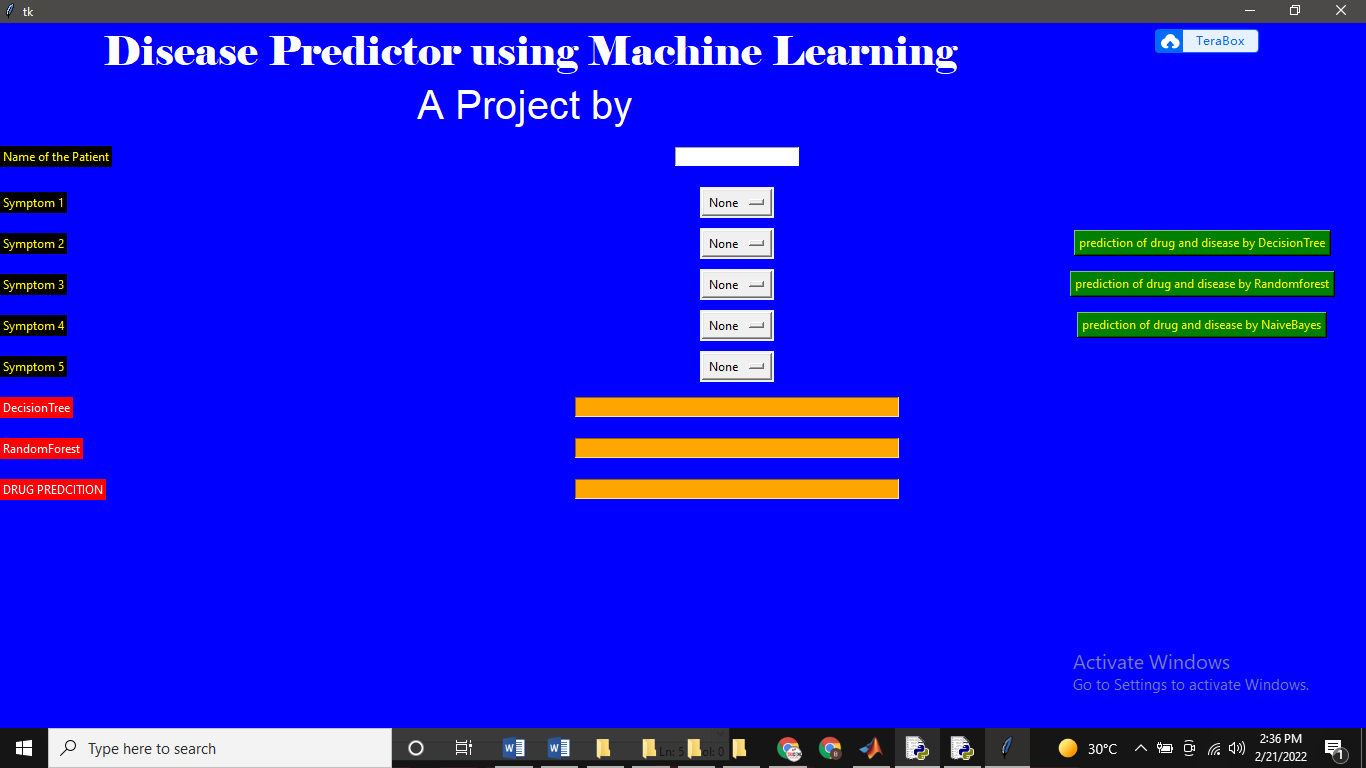
**t2.grid(row=17, column=1 , padx=10)**

**t3 = Text(root, height=1, width=40,bg="orange",fg="black")**

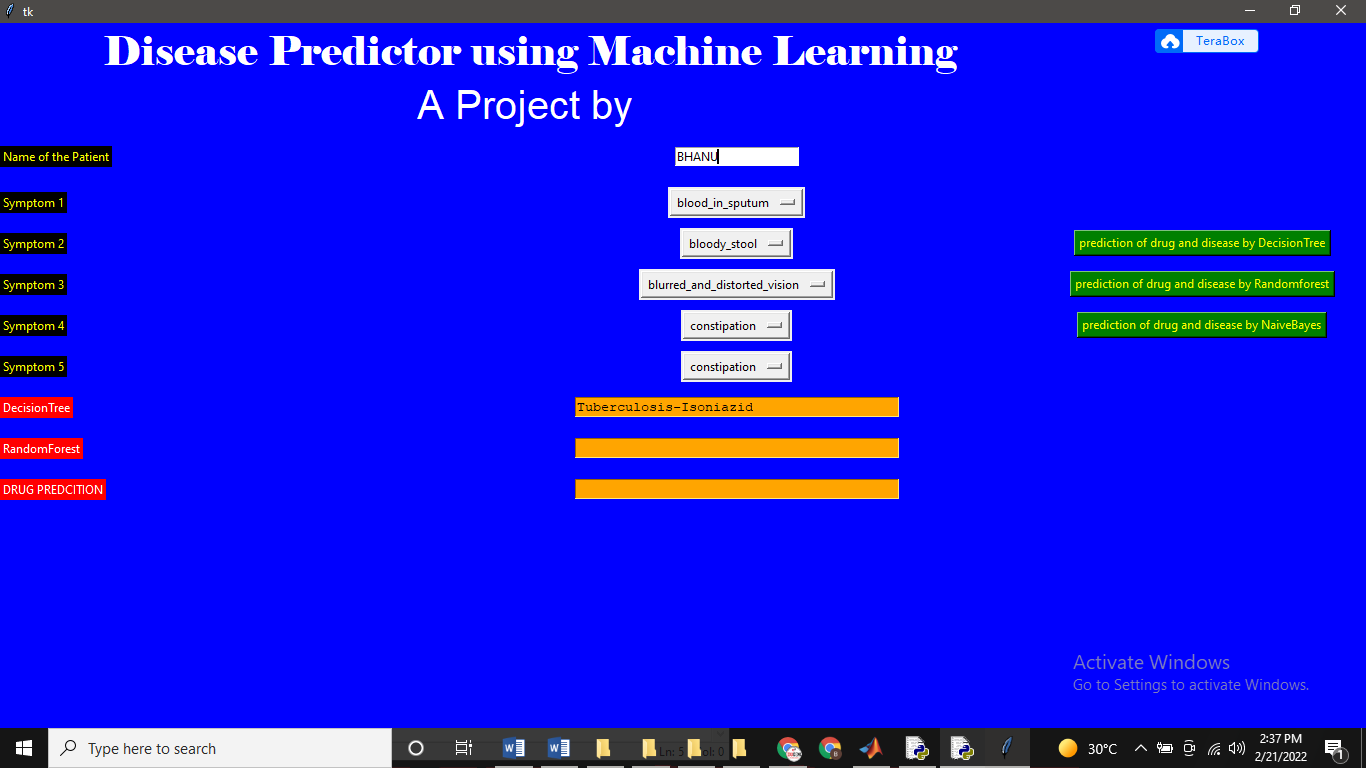
**t3.grid(row=19, column=1 , padx=10)**

**root.mainloop()**

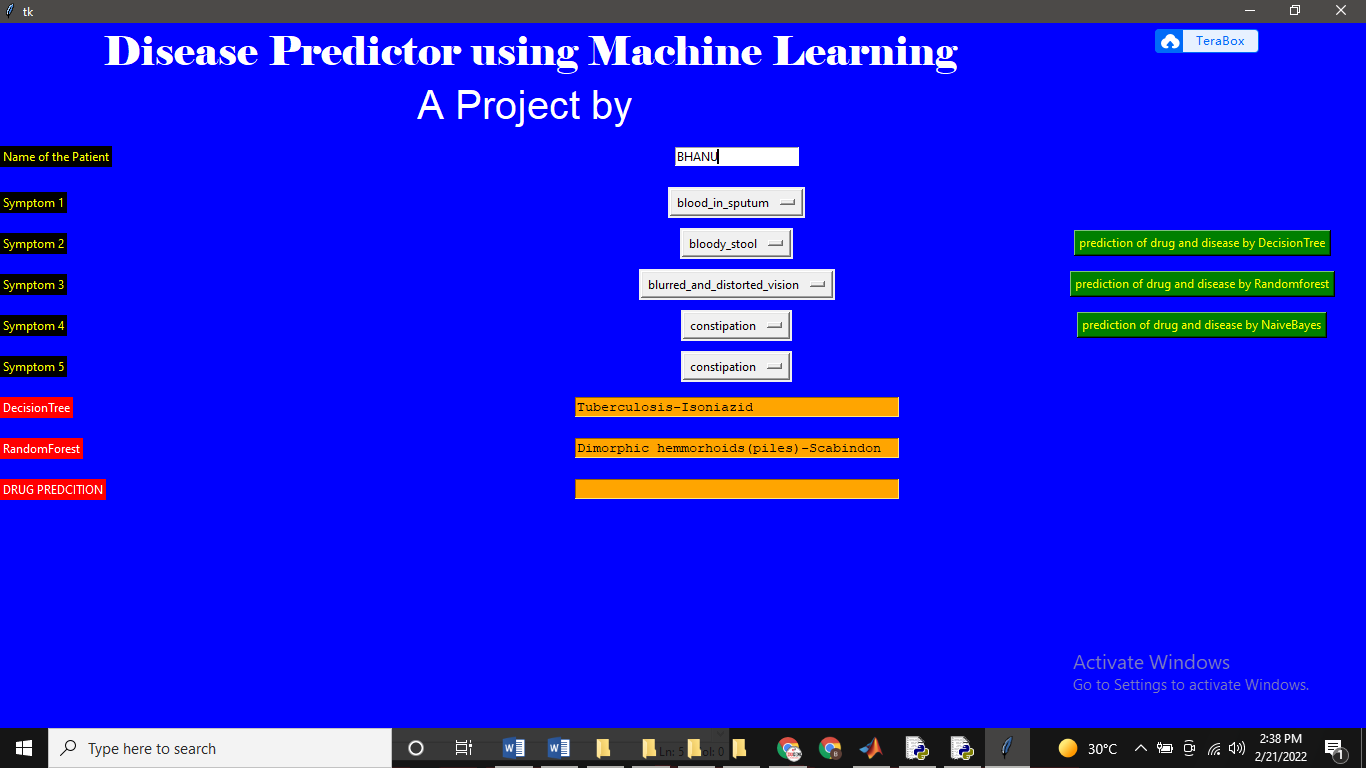
**SCREEN SHOTS**



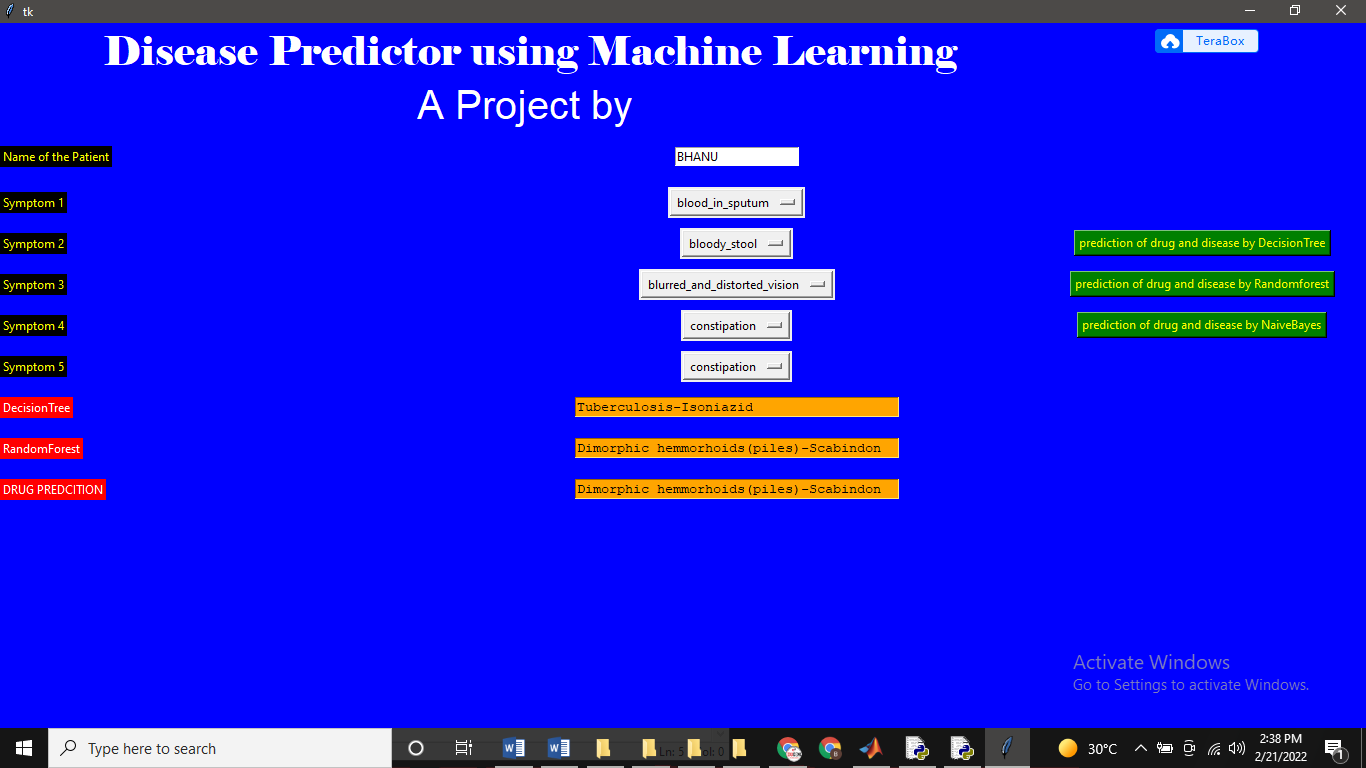
**Prediction of drug and disease by using DecissionTree ALogrithm**



**Prediction of drug and disease by using random forest algorithm**



**Prediction of Drug and Disease By Using NAviebasis algorithm**



**CONCLUSION**

One of the missions of the Ministry of Health and Quality of Life is to transform existing health services into a modern high performing quality health system. Moreover, the ministry wishes to harness the full potential of information technology to empower people to live healthy lives. However, at the same time, Mauritius is prone to many communicable diseases. This paper details the formulation of a system that displays in realtime the disease status over the island. The system consists of three mobile crowdsourcing applications that allows the public, doctors and pharmacies to report diseases and drugs sales in real time. Data regarding reported cases of diseases for the year 2017 were retrieved and augmented with the corresponding daily weather information namely temperature, humidity and wind for that year. The dataset was then used to train an artificial neural network machine learning algorithm which could then predict the propagation of the diseases in the near future using weather forecasts. The accuracy of 90% indicates that such a system is viable and can be a good candidate for an event-based detection system and can definitely help the ministry to achieve its set objectives. The existing system can be enhanced in several ways such as by automatically inferring information from social media posts or by using a chatbot application to retrieve symptoms which would have then been mapped onto the most appropriate disease. The system could also be integrated with the existing information system implemented in pharmacies. The prediction for other common diseases also could have been considered if more datasets were made available. The use of different machine learning algorithms, and consequently their percentage accuracy, could have been investigated. Finally, it is also important to provide incentives to attract users to the crowdsourcing applications and retain them.

REFERENCES [1] B. L. Ranard, Y. P. Ha, Z.F. Meisel, D. A. Asch, S. S. Hill, L. B. Becker, A. K. Seymour and R. M. Merchant, “Crowdsourcing— harnessing the masses to advance health and medicine, a systematic review”, Journal of general internal medicine, 29(1), 2014, pp.187-203. [2] HealthMap (Homepage) 2019. [Online]. Available: http://www.healthmap.org/. [Accessed: 23- Feb- 2019]. [3] J. Wiens and E. S. Shenoy, “Machine learning for healthcare: on the verge of a major shift in healthcare epidemiology”, Clinical Infectious Diseases, 66(1), 2017, pp. 149-153. [4] P. Chowriappa, S. Dua, and Y. Todorov, "Introduction to machine learning in healthcare informatics." In Machine Learning in Healthcare Informatics, pp. 1-23. Springer, Berlin, Heidelberg, 2014. [5] D. T. Schrider and A. D. Kern, "Supervised machine learning for population genetics: a new paradigm." Trends in Genetics 34, no. 4, 2018, pp. 301-312. [6] A. Singh, N. Thakur and A. Sharma, “A review of supervised machine learning algorithms”, In 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), 2016. [7] L. Wang, "Discovering phase transitions with unsupervised learning." Physical Review B 94, no. 19, 2016. [8] A. A. Soofi and A. Awan, "Classification techniques in machine learning: applications and issues", Journal of Basic and Applied Sciences 13, 2017, pp. 459-465. [9] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts and P. Walter, Introduction to Pathogens, Molecular Biology of the Cell, 4th ed., [online] Available: https://www.ncbi.nlm.nih.gov/books/NBK26917/ [Accessed 12 Sep. 2018]. [10] World Health Organization, “The human”, 2019. [Online]. Available: http://www.who.int/denguecontrol/human/en/. [Accessed: 10- Sep2019]. [11] P. K. Praveen, S. Ganguly, R. Wakchaure, P. A. Para, T. Mahajan, K. Qadri,S. Kamble, R. Sharma, S. Shekhar and N. Dalai, “Water-borne diseases and its effect on domestic animals and human health: A Review”, International Journal of Emerging Technology and Advanced Engineering, 6(1), 2016, pp. 242-245. [12] J. Wei and Y. Li, “Airborne spread of infectious agents in the indoor environment”, American Journal of Infection Control, 44(9), 2016, pp. S102-S108. [13] N. Haider, C. Kirkeby, B. Kristensen, L. J. Kjær, J. H. Sørensen and R. Bødker, “Microclimatic temperatures increase the potential for vectorborne disease transmission in the Scandinavian climate”, Scientific reports, 7(1), 8175, 2017. [14] S. L. Shrestha, I. L. Shrestha, N. Shrestha and R. D. Joshi, “Statistical modeling of health effects on climate-sensitive variables and assessment of environmental burden of diseases attributable to climate change in Nepal”, Environmental Modeling & Assessment, 22(5), 2017, pp. 459-472. [15] MedicineNet, 2019. [Online]. Available: https://www.medicinenet.com/. [Accessed: 10- Mar- 2018]. [16] Health Statistics Report 2017, Ministry of Health and Quality of Life , 2018 [Online] Available at: http://health.govmu.org/English/Documents/2018/ANNUAL%20REP ORT%202017%20FOR%20PRINTING.pdf [Accessed 03 Mar 2018]. [17] Figure 1 - Sharpen your clinical skills, Homepage, 2019. [Online]. Available: https://figure1.com/. [Accessed: 01 Mar 2019]. [18] R. Chunara, S. Aman, M. Smolinski and J. S. Brownstein, “Flu near you: an online self-reported influenza surveillance system in the USA”, Online Journal of Public Health Informatics, 5(1), 2013. [19] M. S. Smolinski, A. W. Crawley, K. Baltrusaitis, R. Chunara, J. M. Olsen, O. Wójcik, M. Santillana, A. Nguyen and J. S. Brownstein, “Flu Near You: crowdsourced symptom reporting spanning 2 influenza seasons”, American journal of public health, 105(10), 2015. [20] Sickweather - Sickness Forecasting & Mapping, Homepage, 2019, [Online], Available: http://www.sickweather.com/. [Accessed: 01 Mar 2019]