GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Prediction Of Disease By Using Machine Learning

A Report for the Evaluation 3 of Project 2

Submitted by

SHARAD KUMAR

(1613101668 / 16SCSE101610)

in partial fulfillment for the award of the degree of

Bachelor of Technology

IN

Computer Science and Engineering

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Ms. Supriya Khaitan Chandra

Assistant Professor

APRIL / MAY- 2020

ACKNOWLEDGEMENT

I am pleased to present this Project report entitled **Prediction of disease using machine learning**. It is indeed a great pleasure and a moment of immense satisfaction for me to express sense of profound gratitude and indebtedness towards my guide Asst. Prof. **Ms. Supriya khaitan Chandra** whose enthusiasm are the source of inspiration for me. I am extremely thankful for the guidance and untiring attention. I would like to thank the entire Teaching staff who are directly or indirectly involved in the various data collection and software assistance to bring forward this report. I express deep sense of gratitude towards my parents for their sustained cooperation and wishes, which have been a prime source of inspiration to take this project work. Last but not the least, I would like to thank all our B.Tech colleagues for their co-operation and useful suggestion and all those who have directly or indirectly helped us in completion of this project work.

Greater Noida, Uttar Pradesh

Winter 2019-2020

TABLE OF CONTENTS

ABSTRACT

Machine learning (ML) is the technology that allows a computer system to learn from the environment, through re-iterative processes, and improve itself from experience. Recently, machine learning has gained massive attention across numerous fields, and is making it easy to model data extremely well, without the importance of using strong assumptions about the modeled system. The rise of machine learning has proven to better describe data as a result of providing both engineering solutions and an important benchmark. Therefore, in this current research work, we applied three different machine learning algorithms, which were, the Random Forest (RF), Decision Tree, Naive Bayes to predict disease based on a biomedical data. We will look for all other algorithms available to achieve our task. We suggest that machine learning should be adopted and used as an essential and critical tool across the maximum spectrum of answering biomedical questions.

Breast cancer is the second most leading cancer occurring in women compared to all other cancers. Around 1.1 million cases were recorded in 2004. Observed rates of this cancer increase with industrialization and urbanization and also with facilities for early detection. It remains much more common in high-income countries but is now increasing rapidly in middle- and low-income countries including within Africa, much of Asia, and Latin America. Breast cancer is fatal in under half of all cases and is the leading cause of death from cancer in women, accounting for 16% of all cancer deaths worldwide. The objective of this research paper is to present a report on breast cancer where we took advantage of those available technological advancements to develop prediction models for breast cancer survivability. We used three popular data mining algorithms (Naïve Bayes, RBF Network, J48) to develop the prediction models using a large dataset (683 breast cancer cases).The results (based on average accuracy Breast Cancer dataset) indicated that the Naïve Bayes is the best predictor

with 97.36% accuracy on the holdout sample (this prediction accuracy is better than any reported in the literature), RBF Network came out to be the second with 96.77% accuracy, J48 came out third with 93.41% accuracy.

INTRODUCTION

Humans are vulnerable to different types of diseases ,some of them may cause death of the patient if not taken care on time or delayed in treatment .But as we are getting familiar with the disease ,bacteria and virus are evolved day by day and becoming more powerful that we can relate with the today's scenario i.e. covid-19 which is evolved from Sars cov2 which affect almost all the countries in world. As there are various type of diseases present with similar symptoms ,it is difficult for the doctors to prescribe the medicines without testing the patient, there are various types of test done which take more than a week for the test result to be declared, after then only doctors gives meds to their patient but this process is so long that some of disease may cross their initial stage and hard to control or may cause death of the individuals.

What is the Solution, is there any way to reduce the waiting time for test results.

The Idle solution is the to increase the number of labs where testing I performed and we need modern and superfast equipments for testing, but it cannot be possible to increase the number of labs so, we have to look for modern method of predicting the disease by using machine learning, lets see how it helps us in medical field.

Machine learning is a subset of artificial intelligence which has ability to recognize the pattern for finding the best solution of the given problem and has ability to improve itself by comparing it to previous model. It helps in working more efficiently and more creatively

generate the output. This self learning helps in medical field which can predict the disease more efficiently an can be used by doctor to prescribe certain medicine until test report will be out and give idea of the disease that a patient can have. Medical field gets the boost when the topics like data mining and machine learning integrate with it, which gives better prediction , decrease the cost of medicine and saving the lives of people by real time decision. The paper provides you brief detail of algorithms, accuracy and comparison between different algorithms such as Support Vector Machine, Random Forest, Naïve Bayes, Logistic Regression, K-Nearest Neighbors. Before we get into detail of different algorithm lets get started from the bottom.

There are four different types of learning i.e.

i. Supervised Learning
ii. Unsupervised Learning
iii. Semi Supervised Learning
iv. Reinforcement Learning

Supervised Learning :- In the simple words it predicts the next value or output. In supervised learning we label the given data in the form of training set and test set. The model is developed by feeding the the dataset by training set and improvise the model by predicting the output of the test set.each time model improves itself. when it is fully trained then it has given the that has never seen by the model and predicts a good label for it. Spam classification and face recognition are common example.

Unsupervised Learning :-In one word we can define it as a cluster Identifier. It is a data driven learning. It is just opposite to the supervised learning ,we don’t need to feed the label dataset just all data is given to it and it search for the similar pattern, behavior and other

similarities. Based on those similarities it distinguish the dataset into different clusters. For ex:-Recommendation System, Buying habits etc.

Semi Supervised Learning:-It is mixture of both supervised and unsupervised Learning .The model is trained with a small amount of labeled dataset and large amount of unlabeled data to reduce the cost of training. Some of its examples are Speech analysis, Internet content classification.

Reinforcement Learning :-It can be said as behavior driven learning in one word. It learn from its own mistakes each time model evaluates itself and punish or reward so for next time it will not repeat the same mistakes. Over the time it will not make more mistakes. I am not going deeper because it is not our topic of interest. We need a idea of the learnings. Some of its examples are video games ,resource Management. We will look in detail of algorithm used for these learnings and how it can help in medical field.

Being the most frequently occurring cancer in women, breast cancer affects around 10% of women at some point in their life. It is the second leading contributor to women's death after lung cancer. 25% of all cancers in women including 12% of all new cases are caused by breast cancer. Big Data has seen a rise in value due to it being used in derivation of business intelligence, business analytics and data mining to obtain reports and result predictions. Topics like medical science rise rapidly when certain approaches like data mining is applied due to better possibility of prediction of diseases, reducing medicine costs, improving health of patient by revamping the quality of healthcare along with value by saving people's lives through real time decisions. The paper provides you with a analysis of performance and comparison of accuracy in classification between the algorithms such as: Logistic Regression, SVM, Random Forest and Naïve Bayes, being the major influential algorithms of data mining used in the research community. Logistic Regression is used to perform regression analysis

when the dependent variable is binary. It is a predictive analysis similar to all other regression analyses. Naïve Bayes is a powerful classification algorithm from machine learning, it is not a algorithm but a group of algorithm in which all of them have same fundamental principle. In that group of algorithms, they all classify the data independently such that no algorithm provides same classification result or analysis. Naïve Bayes can be seen using in places such as document classification. Document classification is nothing but it is used to classify document based on fields Random forest algorithm basically creates multiple trees and select the best among them by voting methodology. Random Forest algorithm creates multiple tree, each tree having different decision independent to each other. In common words no two tree is similar in Random Forest such that every tree provides implies classification methodology. This can be useful in certain instances like these where we have to classify medical dataset. Random forest is also used in places such as ETM, detection and prediction of an object and in games where it replicates the actions produced by humans in games. SVM is classification and regression model. It can be used to both circumstances. SVM creates a hyperplane which is basically called as threshold limit to classify data. This limit is created by the dataset while training. What if there is an deviation in data, SVM creates soft threshold limit which is close near to the main threshold limit and near to the deviation. The more dataset given to the algorithm the proper the classification will be of hyperplane. The kernel used in this project is linear. The reason to use linear kernel is it is faster and it is preferred when the data can be linearly separable. Places where you can see SVM used is text classification, face detection etc. Face detection is one of the main reasons SVM is preferred widely. Algorithm's efficiency evaluation is the primary objective of this project. Decision tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label. Random Forest

is a collection of decision tree. A tree can be learned by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions. The construction of decision tree classifier does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery. Decision trees can handle high dimensional data. In general decision tree classifier has good accuracy. Decision tree induction is a typical inductive approach to learn knowledge on classification.

OVERALL DESCRIPTION

This section presents the dataset used, the pre-processing of the data and the machine learning algorithms, namely, Random Forest (RF), Support Vector Machines, Decision Tree,and the Artificial Neural Network. The pre-processing of the data and the implementations of the models were achieved by using the Python environment. We used Python 3 as the version in writing the codes, which was achieved through the Jupyter Notebook by installing the Anaconda software.

PURPOSE

In machine learning, cancer classification can be done using benign or malignant cells could significantly improve our ability for prognosis in cancer patients, poor progress has been made for their application in the clinics. However, before gene expression profiling can be used in clinical practice, studies with larger data samples and more adequate validation are needed. And thus, our aim is to develop a prediction system that can predict chances of breast cancer and other diseases on a huge data. Our models could serve as the bases of new cost-

effective and non-invasive tools to inform and prompt screening and immediate and long-term preventative actions with the potential to increase early detection and reduce the incidence of breast cancer, and we can predict various other chronic disease which can take more time to be tested in labs. We need more efficient and accurate way of predicting the disease so we can use proper medication on time to save lives. Humans lives should be prioritized and some disease should be taken care in their initial stages because waiting for the test result may result into serious problem.

MOTIVATION AND SCOPE

To compare the behaviors of LR, NB, SVM and Random Forest, the experiment conducted was focused on the evaluation of the algorithms. Questions raised from researchers were: Which algorithm is more effective? Which algorithm executes more efficiently? Which algorithm is more accurate in classifying?

A. Experiment Environment The sickit learn python libraries were used to conduct all experiments on classifiers explained in this paper. Sklearn is a collection of data mining, machine learning and deep learning algorithms used for classification, regression, data pre-processing and clustering. The sklearn libraries were used to implement machine learning algorithms for various real-world problems. Developers and practitioners can build and evaluate suitable models with this framework. The experiment conducted in which environment is conducted is ANACONDA. It contains various applications in that we have preferred Spyder, which is a development environment that supports python. It is a powerful IDE for python compared to others. It also has introspection features. Since our problem might require those features and also debugging is easier in this platform it is preferred.

B. Breast cancer dataset The UCI machine learning repository consists of The Wisconsin Breast Cancer datasets which is used in this study. There are 569 instances in which 357 are

benign and 212 are malignant. In addition, there are two classes malignant which contributes to 37.2% of dataset a

nd benign 62.8%. The breast cancer dataset is obtained in a csv format from their database.

C. Data Visualization Data visualization is a key aspect of data science. It helps one to comprehend and also convey the data to another person in a meaningful manner. Matplotlib and Seaborn are some of the several python data visualization libraries. It is essential in analysing large amounts of information and to make decisions. It employs the use of pictorial elements such as maps, plots, patterns, graph trends, etc. to provide the user with an easy method of comprehending the data.

EXPECTED RESULTS

After creating predictive model, efficiency can be checked. For this, the models can be compared based on accuracy and time consumed. It was really hard to choose the algorithm which has higher performance, greater accuracy and efficiency, since all of them ended very close in accuracy. The time consumed and accuracy value of the algorithms from machine learning is shown in Table 1.

Table 1: Experimental Results

Algorithm	Accuracy	Time taken
LR	99.06%	0.02s
SVM	98.59%	0.03s
NB	94.83%	0.01s
Random Forest	99.76%	0.02s

LITERATURE SURVEY

Data mining is been applied on medical data of the past and current research papers. Thorough study is done on various base reports. Jacob et al. have compared various classifier algorithms on Wisconsin Breast Cancer diagnosis dataset. They came across that Random Tree and SVM classification algorithm produce best result i.e. 100% accuracy. However they mainly worked on 'Time' feature along with other parameters to predict the outcome of non-recurrence or recurrence of breast cancer among patients. In this paper, "Time" feature has not been relied upon for prediction of recurrence of the disease. Here, prediction is based on "Diagnosis" feature of WBCD dataset. Chih-Lin Chi et al. used the ANN model for Breast Cancer Prognosis on two dataset. They predicted recurrence and non-recurrence based on probability of breast cancer and grouped patients with bad (5 years) prognoses. Delen et al. used the SEER dataset of breast cancer to predict the survivability of a patient using 10-fold cross validation method. The result indicated that the decision tree is the best predictor with 93.6% accuracy on the dataset as compared to ANN and logistic regression model.

PROBLEM STATEMENT

As there are various type of diseases present with similar symptoms ,it is difficult for the doctors to prescribe the medicines without testing the patient, there are various types of test done which take more than a week for the test result to be declared, after then only doctors gives meds to their patient but this process is so long that some of disease may cross their initial stage and hard to control or may cause death of the individuals.

What is the Solution, is there any way to reduce the waiting time for test results.

The Idle solution is the to increase the number of labs where testing I performed and we need modern and superfast equipments for testing, but it cannot be possible to increase the number of labs so, we have to look for modern method of predicting the disease by using machine learning, lets see how it helps us in medical field. Other real lives problems can be seen below.

The accurate prediction of survival rate in patients with breast cancer remains a challenge due to the increasing complexity of cancer, treatment protocols, and various patient population samples. Reliable and well-validated predictions could assist in a better way personalized care and treatment, and improve the control over the cancer development. Usually in good clinical practices, clinicians use data collected from different sources as medical records, clinical laboratory tests, and studies aiming more precise diagnostics, therapy, and disease-development prognosis. There is a definite increase in the use of classification-based approaches in contemporary medical diagnostics. Cancer studies are the major target in using contemporary bioinformatics, statistics, and ML techniques for the purposes of more accurate and rapid diagnostics. In the scope of constantly growing significance of predictive and personalized medicine, there is a rapidly growing demand to apply machine learning-driven models to make predictions and prognosis in cancer studies. At first sight, all these classification-based approaches use various and heterogeneous medical data and can inflate the quality of diagnostics. On the contrary, numerous recent developments in computer science, data science, and ML assist in the decrease of errors in overall diagnostics. The use of artificial intelligence techniques for classification in cancer studies provides more informative knowledge-based background for prediction and prognosis of cancer to be tested more meticulously and rapidly, in a short time. Prediction and prognosis of cancer development are focused on three major domains: risk assessment or prediction of cancer susceptibility, prediction of cancer relapse, and prediction of cancer survival rate. The first

domain comprises prediction of the probability of developing certain cancer prior to the patient diagnostics. The second issue is related to prediction of cancer recurrence in terms of diagnostics and treatment, and the third case is aimed at prediction of several possible parameters characterizing cancer development and treatment after the diagnosis of the disease: survival time, life expectancy, progression, drug sensitivity, etc. The survivability rate and the cancer relapse are dependent very much on the medical treatment and the quality of the diagnosis . About 40% of all ML studies on breast cancer prediction were focused on predicting patient survivability. There is a variety of examples of machine learning-based approaches applied to different datasets. The obvious trend is that all the major studies with clinical data mostly use models related to artificial neural networks (ANN) and support vector machines (SVM), and use statistical methods for validation. In this way, some problems with classification and validation have been overcome. There is an obvious demand to improve the ML impact in survival time prediction studies in breast cancer in the scope of generality, better accuracy, and validation. These challenges are also in the scope of our work

PROPOSED MODEL

We obtained the breast cancer dataset of Wisconsin Breast Cancer diagnosis dataset and used jupyter notebook as the platform for the purpose of coding. Our methodology involves use of supervised learning algorithms and classification technique like Decision Tree, Random Forest and Logistic Regression, with Dimensionality Reduction technique.

1. **Data Processing:**

   Our dataset may be Incomplete or have some missing attribute values, or having only aggregate data. So, there is a need to pre-process our medical dataset which has major

attribute as id, diagnosis and other real valued features which are computed for each cell nucleus like radius, texture, parameter, smoothness, area, etc

2. **Categorical Data**:

Categorical data are variables that contain label values rather than numeric values. So, here we have represented benign cells as value 0 and malignant cells as value 1.

Splitting the dataset: The data we use is usually split into training data and test data. In our project 80% data is trained data and 20% data is test data.

3. Feature Scaling :

Generally, dataset contains features which highly varies in magnitudes, units and range. So there is a need to bring all features to the same level of magnitudes. This can be achieved by scaling.

4. Model Selection:

This is the most important phase where algorithm selection is done for the developing system. Data Scientists use various types of Machine Learning algorithms which can be classified as: supervised learning and unsupervised learning. For this Prediction System, we only need Supervised Learning

4.1 Supervised Learning:

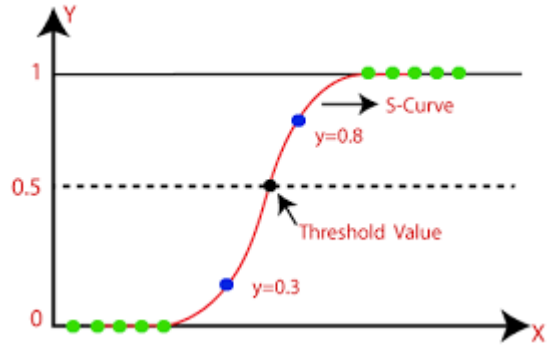
Supervised Learning is a type of system in which both input and desired output data are provided. Input and output data are labelled for classification to provide a learning basis for future data processing. Supervised systems provide the learning algorithms with known quantities to support future judgments.

I. Logistic Regression

Logistic Regression (LR) is one of the fundamental and famous algorithms to solve classification problems. LR is used to obtain odds ratio in the presence of more than one independent variable . LR deals with outliers by using sigmoid function. Therefore, Logistic function is a sigmoid function, which takes any real value between 0 and 1. It is mathematically expressed as:

(1) Considering t as a linear function in a univariate regression model:

(2) Therefore, the Logistic Regression is represented as:



II. Support Vector Machine

Support Vector Machine theory is based on statistics which have a fundamental principle of estimating the optimal linear hyperplane in a feature space that maximally separates the two-mark groups or classes. SVM modeling geometrically finds an optimal hyperplane with the maximal interval to separate two groups or classes. The procedure for solving such a constraint problem is as follows [24,25,26,27]:

Subject to:

x = the feature vectors or the input pattern, w = the direction of the optimal hyperplane,

b = bias

To make rooms for errors, the optimization problem currently becomes:

Subject to:

The Lagrange multiplier method assists us to obtain the two formulas, that is expressed in terms of the variable αi:

Subject to:

For all i = 1, 2, 3…., n ----------eqn (i)

The linear classifier based on a linear discriminant function is then given as:

A non-linear classifier sometimes assists in providing better accuracy in many applications. A fragile way of making a non-linear classifier out of linear classifier is by mapping out data from the input space X to a feature space F based on a non-linear function . Using kernel function, in the space F, the optimization assumes the following:

Subject to:

For all i = 1, 2, 3…., n ----------eqn(ii)

The classifier is given as a SVM-Polynomial, where, if d is large, the kernel still requires n multiplications to compute. So, given

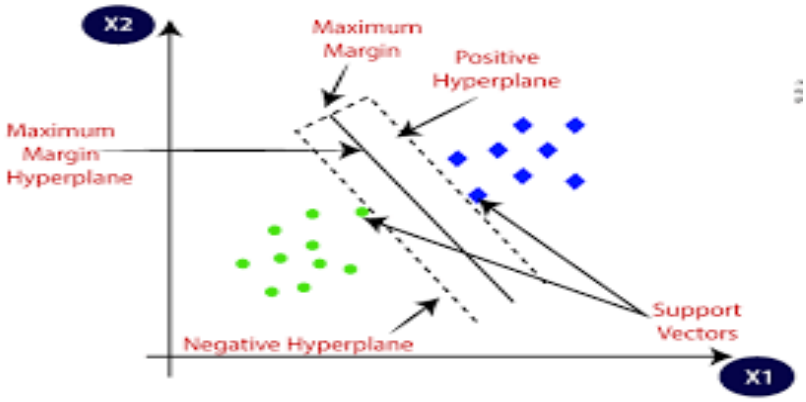
The RBF classifier is given as _____eqn(iii)

In this current research paper, we implemented (i), (ii) and (iii) as the SVM kernels which represent Linear, Polynomial and Radial Basis Function (RBF) respectively. The cost function (C) presents the measure of how wrong a model is in terms of its ability to estimate the relationship between x and y as seen from. The cost parameter was optimized by grid

search in the training dataset. Given a Hypothesis -> H0 (x) = β0 + β1x, then the cost function (C) is represented as:

where N is the number of observations.

In developing the SVM classifiers, the values of the cost function used were (0.1, 1, 10, 100).



III. **Random Forest**

Random Forest was proposed by Breiman as an ensemble classifier or regression tree based on many decision trees. Each of the trees is based on a bootstrap sample from the original training dataset using a tree classification procedure. Bootstrapping is a metric or test that depends on random sampling with replacement. A random selection of the whole variable set is used as variables for splitting the tree nodes. After formation of the forest, a new object which needs to be classified is noted for classification by each of the trees in the forest. A vote is cast by each of the trees to indicate the tree's decision about the group or class of the object. The group or class with the majority of the votes is chosen by the forest.

According to Neural Designer, Random Forest is one of the most famous algorithms used by data scientists. In Random Forest, each tree is influenced by the values of a random vector sampled independently .

Random Forest algorithm follows as:

Select number of trees to grow (ntree)

➢For i = 1 to ntree

- Randomly sample with replacement, same size as original dataset (bootstrap).
- Grow a tree.
- For every split of tree.
- Randomly select mtry predictors.
- Grow trees till stopping criteria is reached.
- Each tree then casts a vote for the most famous class, and the class with most votes wins.

➢end

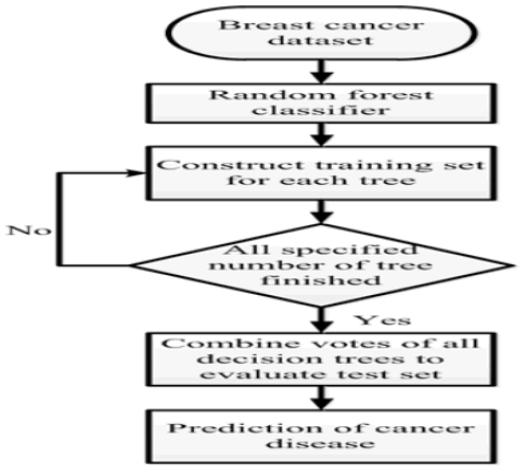


Figure 2: Flow diagram of random forest classifier for prediction of cancer disease.

IV. Naive Bayes Algorithm:

Naive Bayes algorithm is one of the most effective methods in the field of text classification, but only in the large training sample set can it get a more accurate result. The requirement of a large number of samples not only brings heavy work for previous manual classification, but also puts forward a higher request for storage and computing resources during the computer post-processing. Naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem (from Bayesian statistics) with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be "independent feature model". In simple terms, a naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature.

Step 1: Scan the dataset (storage servers)

Step 2: Calculate the probability of each attribute value. [n, n_c, m, p]

Step 3: Apply the formulae P(attributevalue(ai)/subjectvalue(vj))=(n_c + mp)/(n+m) Where:

- n = the number of training examples for which v = vj
- nc = number of examples for which v = vj and a = ai
- p = a priori estimate for P(aijvj)
- m = the equivalent sample size

Step 4: Multiply the probabilities by p

Step 5: Compare the values and classify the attribute values to one of the predefined set of class.

We can express it in mathematical form as :

$$P(A|B)= \frac{P(B|A).P(A)}{P(B)}$$

Here P(A) is the priori or the unknown instance which we are going to predict and P(B) is the evidence P(A|B) is the probability after P(B) has been seen.

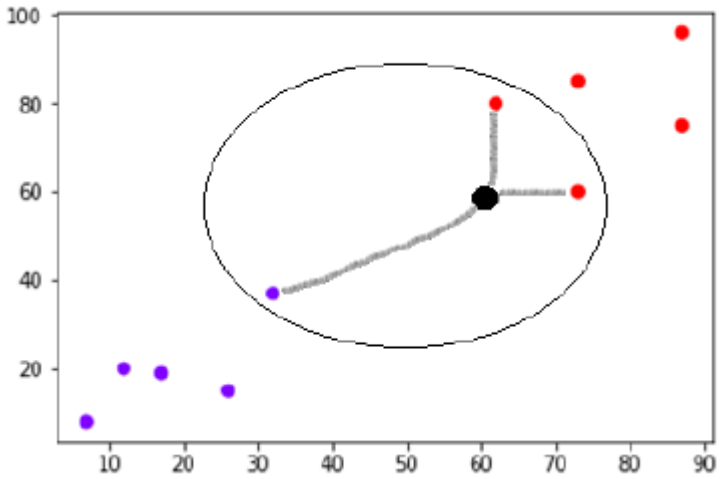
In the case of machine learning we use :-

$$P(Y|x1,x2,\ldots\ldots,xn) = \frac{P(x1|Y).P(x2|Y)\ldots\ldots P(xn|Y).P(Y)}{P(x1),P(x2),\ldots,P(xn)}$$

V. K-nearest neighbour

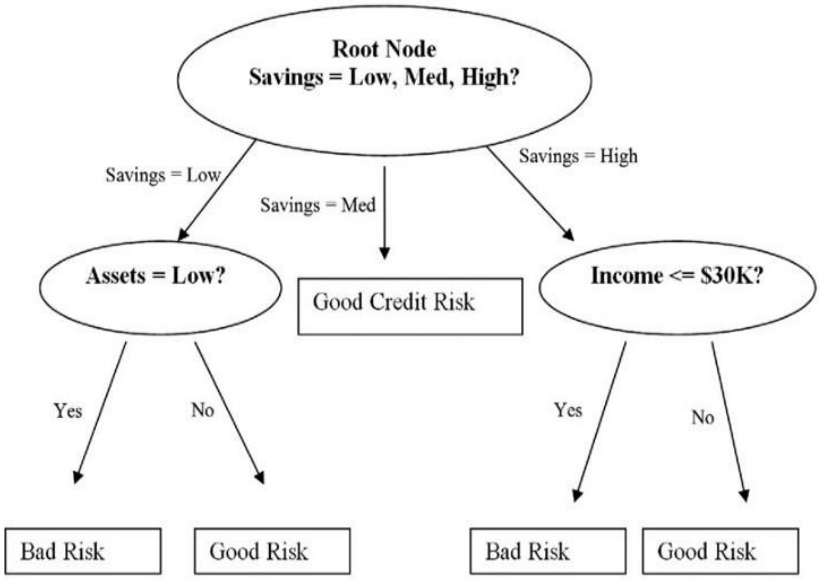
It is an algorithm which can be used for both the classification and regression predictive problem. In the classification problem we classify between or 1 for example if you like cake or not can be trained in model with respect to age can be feed ,and in regression it gives real value as output for example we have to predict the weight of the person the basis of their age. Knn simply follows the technique that similar things are close to each other. This assumption works well in the case of knn, and similarity itself has different meaning with different situations like similarity refers to distance, proximity and closeness which we can calculate by using Euclideans Distance formula.

If k=3 we will choose x=60,y=60.



VI. Decision Tree

As we are moving to our last algorithm last but not the least because their are still various algorithm available. We have seen Random Forest is collection of decision tree and yield better output. Decision tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label. A tree can be learned by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions. The construction of decision tree classifier does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery. Decision trees can handle high dimensional data. In general decision tree classifier has good accuracy. Decision tree induction is a typical inductive approach to learn knowledge on classification. Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute as shown in the figure below. This process is then repeated for the subtree rooted at the new node. Here we made a decision tree for the loan which is used by banks i.e. is it safe to give loan to a person who falls under different criteria and we predict different output some are having low risk and some are having high risk which is totally based on different data or value for example your income your banking history, previous credit scores, and savings etc which helps it developing a tree and also conclude that are you eligible for getting loan or not.



IMPLEMENTATION

Here comes the most interesting part of our project where we are going to implement our thinking with the help of computer. We are going to write code for prediction of disease to understand our code one must have prior knowledge of python and few machine learning libraries so, it will be easy to understand how will our code going to function. We are going to built a model and then train to through various algorithms sounds interesting lets do it.

Things we need before building our model is a system with an ide which is mentioned earlier we can choose either jupyter or spyder. we need python 3 installed to create an gui and a dataset to train our model and we are good to go.

We need to import certain libraries for machine learning

```
from tkinter import *

import numpy as np

import pandas as pd
```

List of the symptoms is listed here in list l1.

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_sputu

m',

'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']

List of Diseases is listed in list disease.

disease=['Fungal infection','Allergy','GERD','Chronic cholestasis','Drug Reaction',

'Peptic ulcer diseae','AIDS','Diabetes','Gastroenteritis','Bronchial Asthma','Hypertension',

' Migraine','Cervical spondylosis',

'Paralysis (brain hemorrhage)','Jaundice','Malaria','Chicken pox','Dengue','Typhoid','hepatitis A',

'Hepatitis B','Hepatitis C','Hepatitis D','Hepatitis E','Alcoholic hepatitis','Tuberculosis',

'Common Cold','Pneumonia','Dimorphic hemmorhoids(piles)',

'Heartattack','Varicoseveins','Hypothyroidism','Hyperthyroidism','Hypoglycemia','Osteoarthristis',

'Arthritis','(vertigo) Paroymsal Positional Vertigo','Acne','Urinary tract infection','Psoriasis',

'Impetigo']

l2=[]

for i in range(0,len(l1)):

```
l2.append(0)
```

Here we are reading our dataset to build a model

```
df=pd.read_csv(r'E:/svm/Prototype.csv',encoding='utf-8')
```

Replace the values in the imported file by pandas by the inbuilt function replace in pandas.

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

'Pepticulcerdiseae':5,'AIDS':6,'Diabetes':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertension ':10,

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

'Paralysis(brainhemorrhage)':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,

'CommonCold':26,'Pneumonia':27,'Dimorphichemmorhoids(piles)':28,'Heart attack':29,'Varicose veins':30,'Hypothyroidism':31,

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

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

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

Lets check the df i.e. dataset

```
print(df.head())

X= df[l1]

print(X)

y = df[["prognosis"]]

np.ravel(y)

print(y)
```

Read a csv named Testing.csv

```
tr=pd.read_csv("Prototype.csv")
```

Use replace method in pandas.

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

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

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

'Paralysis (brainhemorrhage)':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"]]

print(y_test)

np.ravel(y_test)

we are building model with the help of decision tree for which we had created function and uses inbuilt tree function from sklarn library

```
def DecisionTree():
    from sklearn import tree
    clf3 = tree.DecisionTreeClassifier()
    clf3 = clf3.fit(X,y)
    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)):

    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")
```

Here we develop function named random forest and uses random forest function from sklearn.ensemble library

```
def randomforest():
    from sklearn.ensemble import RandomForestClassifier
    clf4 = RandomForestClassifier()
    clf4 = clf4.fit(X,np.ravel(y))
    calculating accuracy for different algorithms
    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")
```

now we uses naive bayes prediction algorithm and fit it into our model using naive bayes library

```
def NaiveBayes():
    from sklearn.naive_bayes import GaussianNB
    gnb = GaussianNB()
    gnb=gnb.fit(X,np.ravel(y))
```

```
    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")
```

now, its time to create a gui for our project which will be going to predict output or disease gui makes more interacting to user and helps us with the exact output

```
root = Tk()
root.configure(background='black')
Symptom1 = StringVar()
Symptom1.set("Select Here")
Symptom2 = StringVar()
Symptom2.set("Select Here")
Symptom3 = StringVar()
Symptom3.set("Select Here")
Symptom4 = StringVar()
Symptom4.set("Select Here")
Symptom5 = StringVar()
Symptom5.set("Select Here")
```

```
Name = StringVar()

w2 = Label(root, justify=LEFT, text="Disease Predictor using Machine Learning", fg="Red",
bg="White")

w2.config(font=("Times",30,"bold italic"))

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

w2 = Label(root, justify=LEFT, text="A Project by Sharad Kumar", fg="Pink", bg="Blue")

w2.config(font=("Times",30,"bold italic"))

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

NameLb = Label(root, text="Name of the Patient", fg="Red", bg="Sky Blue")

NameLb.config(font=("Times",15,"bold italic"))

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

S1Lb = Label(root, text="Symptom 1", fg="Blue", bg="Pink")

S1Lb.config(font=("Times",15,"bold italic"))

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

S2Lb = Label(root, text="Symptom 2", fg="White", bg="Purple")

S2Lb.config(font=("Times",15,"bold italic"))

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

S3Lb = Label(root, text="Symptom 3", fg="Green",bg="white")

S3Lb.config(font=("Times",15,"bold italic"))
```

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

S4Lb = Label(root, text="Symptom 4", fg="blue", bg="Yellow")

S4Lb.config(font=("Times",15,"bold italic"))

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

S5Lb = Label(root, text="Symptom 5", fg="purple", bg="light green")

S5Lb.config(font=("Times",15,"bold italic"))

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

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

lrLb.config(font=("Times",15,"bold italic"))

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

destreeLb = Label(root, text="RandomForest", fg="Red", bg="Orange")

destreeLb.config(font=("Times",15,"bold italic"))

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

ranfLb = Label(root, text="NaiveBayes", fg="White", bg="green")

ranfLb.config(font=("Times",15,"bold italic"))

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

OPTIONS = sorted(l1)

NameEn = Entry(root, textvariable=Name)

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

```
S1 = OptionMenu(root, Symptom1,*OPTIONS)

S1.grid(row=7, column=1)

S2 = OptionMenu(root, Symptom2,*OPTIONS)

S2.grid(row=8, column=1)

S3 = OptionMenu(root, Symptom3,*OPTIONS)

S3.grid(row=9, column=1)

S4 = OptionMenu(root, Symptom4,*OPTIONS)

S4.grid(row=10, column=1)

S5 = OptionMenu(root, Symptom5,*OPTIONS)

S5.grid(row=11, column=1)

dst = Button(root, text="Prediction 1", command=DecisionTree,bg="Red",fg="yellow")

dst.config(font=("Times",15,"bold italic"))

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

rnf = Button(root, text="Prediction 2", command=randomforest,bg="White",fg="green")

rnf.config(font=("Times",15,"bold italic"))

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

lr = Button(root, text="Prediction 3", command=NaiveBayes,bg="Blue",fg="white")

lr.config(font=("Times",15,"bold italic"))

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

t1 = Text(root, height=1, width=40,bg="Light green",fg="red")

t1.config(font=("Times",15,"bold italic"))

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

t2 = Text(root, height=1, width=40,bg="White",fg="Blue")

t2.config(font=("Times",15,"bold italic"))

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

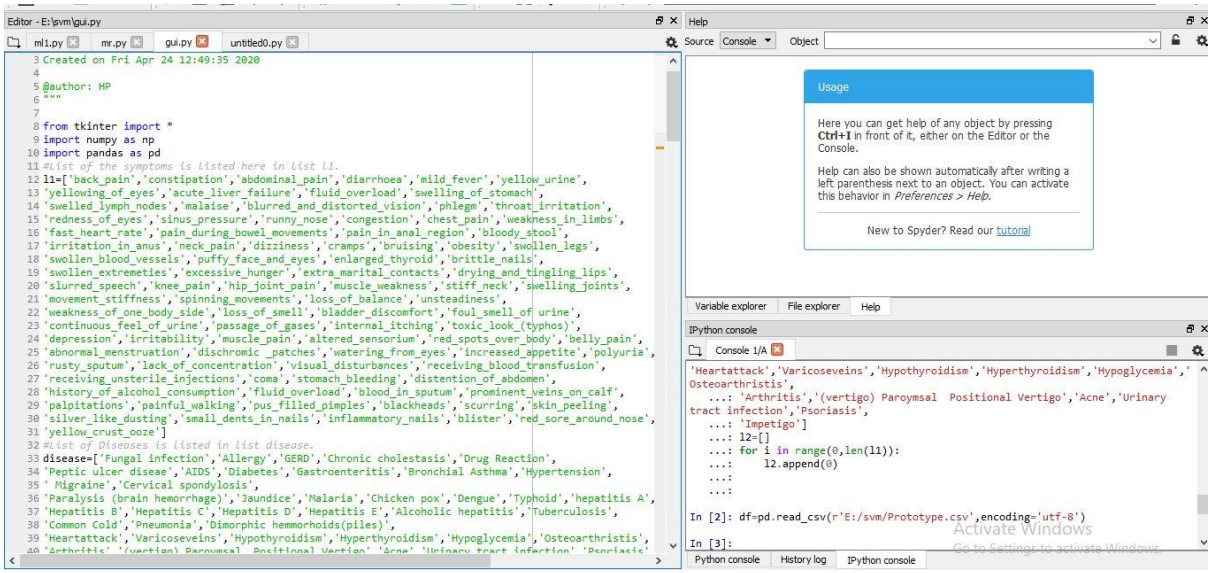
t3 = Text(root, height=1, width=40,bg="red",fg="white")

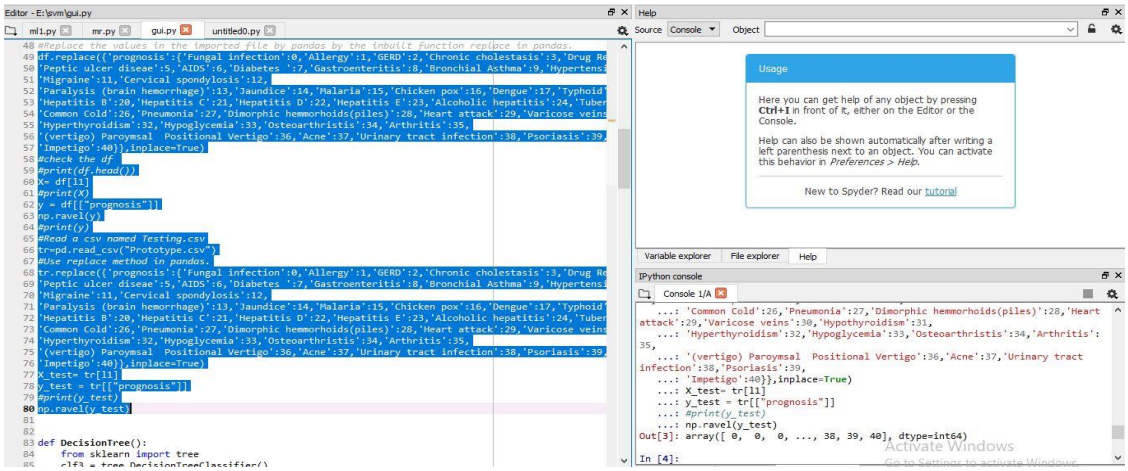
t3.config(font=("Times",15,"bold italic"))

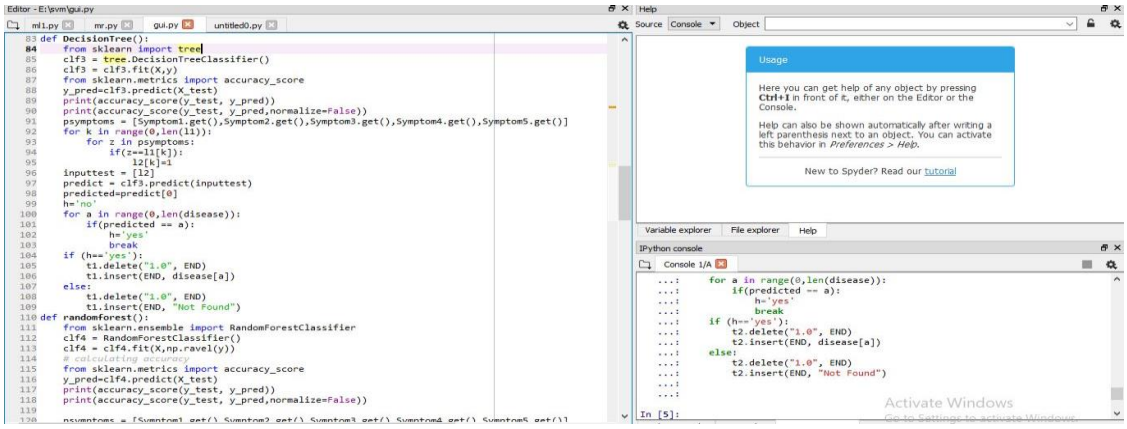
t3.grid(row=19, column=1 , padx=10)

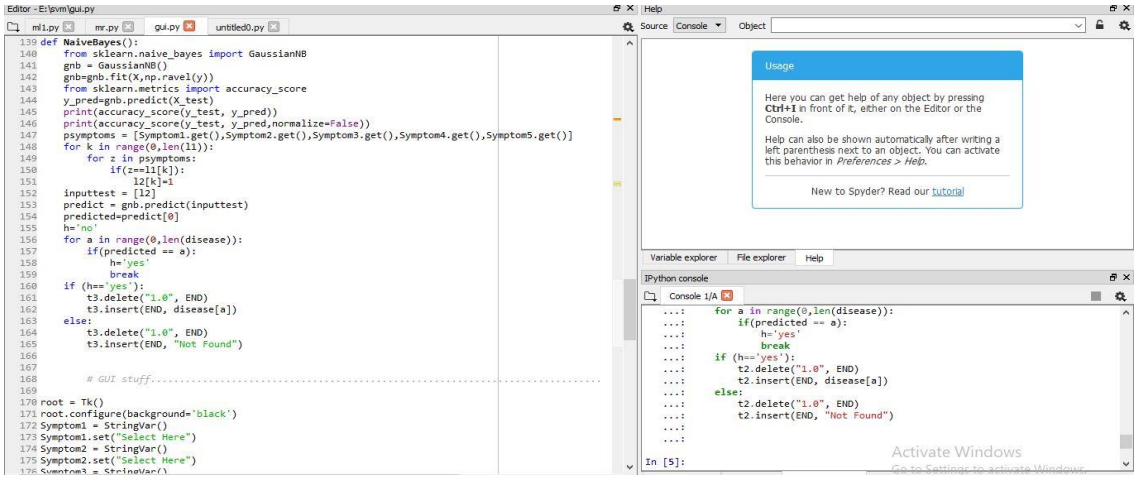
root.mainloop()

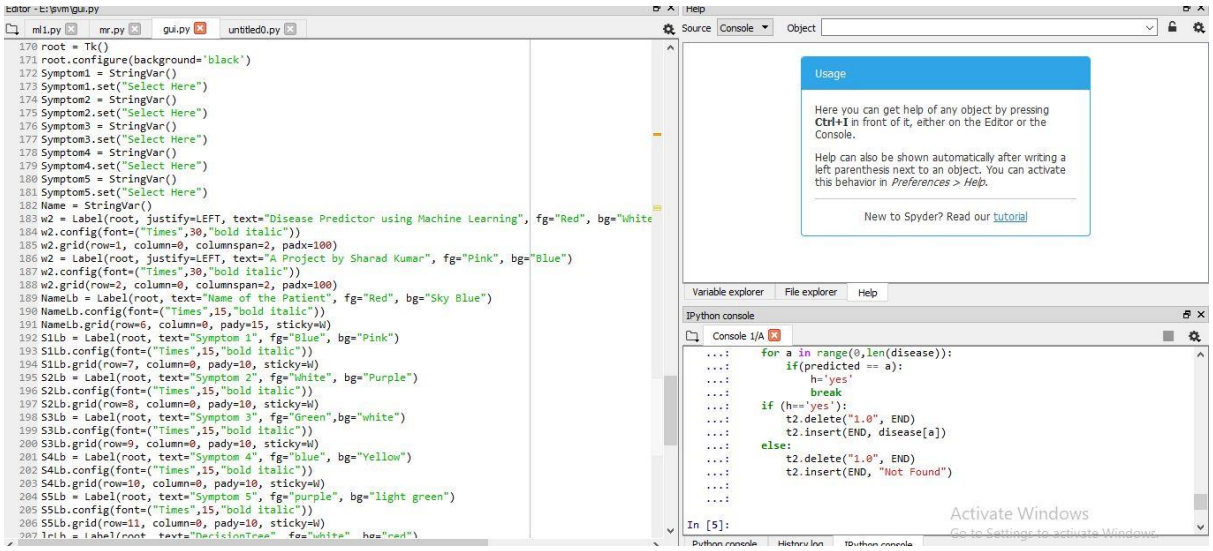
Here are some snap of our code




Editor - E:\svm\gui.py
ml1.py
mr.py
gui.py
untitled0.py
#Replace the values in the imported file by pandas by the inbuilt function replace in pandas.
df.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Re
'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertensi
'Migraine':11,'Cervical spondylosis':12,
'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid'
'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuber
'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose vein
'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)
#check the df
#print(df.head())
X= df[11]
#print(X)
y = df[["prognosis"]]
np.ravel(y)
#print(y)
#Read a csv named Testing.csv
tr=pd.read_csv("Prototype.csv")
#Use replace method in pandas.
tr.replace({'prognosis':{'Fungal infection':0,'Allergy':1,'GERD':2,'Chronic cholestasis':3,'Drug Re
'Peptic ulcer diseae':5,'AIDS':6,'Diabetes ':7,'Gastroenteritis':8,'Bronchial Asthma':9,'Hypertensi
'Migraine':11,'Cervical spondylosis':12,
'Paralysis (brain hemorrhage)':13,'Jaundice':14,'Malaria':15,'Chicken pox':16,'Dengue':17,'Typhoid'
'Hepatitis B':20,'Hepatitis C':21,'Hepatitis D':22,'Hepatitis E':23,'Alcoholic hepatitis':24,'Tuber
'Common Cold':26,'Pneumonia':27,'Dimorphic hemmorhoids(piles)':28,'Heart attack':29,'Varicose vein
'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[11]
y_test = tr[["prognosis"]]
#print(y_test)
np.ravel(y_test)
def DecisionTree():
from sklearn import tree
Help
Source
Console
Object
Usage
Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.
Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.
New to Spyder? Read our tutorial
Variable explorer
File explorer
Help
IPython console
Console 1/A
...: '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[11]
...: y_test = tr[["prognosis"]]
...: #print(y_test)
...: np.ravel(y_test)
Out[3]: array([0, 0, 0, ..., 38, 39, 40], dtype=int64)
In [4]:
Activate Windows
Go to Settings to activate Windows.

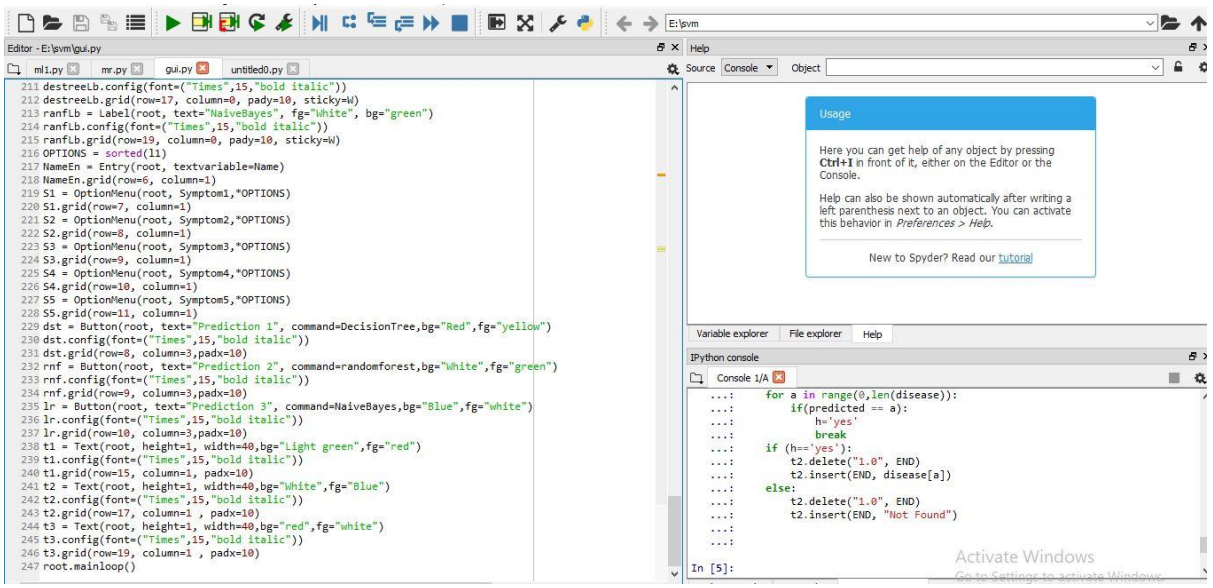

Editor - E:\svm\gui.py
ml1.py
mr.py
gui.py
untitled0.py
def DecisionTree():
from sklearn import tree
clf3 = tree.DecisionTreeClassifier()
clf3 = clf3.fit(X,y)
from sklearn.metrics import accuracy_score
y_pred=clf3.predict(X_test)
print(accuracy_score(y_test, y_pred))
print(accuracy_score(y_test, y_pred,normalize=False))
psymptoms = [Symptom1.get(),Symptom2.get(),Symptom3.get(),Symptom4.get(),Symptom5.get()]
for k in range(0,len(11)):
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()]
Help
Source
Console
Object
Usage
Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.
Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.
New to Spyder? Read our tutorial
Variable explorer
File explorer
Help
IPython console
Console 1/A
...: 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")
...:
...:
In [5]:
Activate Windows
Go to Settings to activate Windows.


Editor - E:\svm\gui.py
ml1.py
mr.py
gui.py
untitled0.py
def NaiveBayes():
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb=gnb.fit(X,np.ravel(y))
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(11)):
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='black')
Symptom1 = StringVar()
Symptom1.set("Select Here")
Symptom2 = StringVar()
Symptom2.set("Select Here")
Help
Source
Console
Object
Usage
Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.
Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.
New to Spyder? Read our tutorial
Variable explorer
File explorer
Help
IPython console
Console 1/A
...: 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")
...:
...:
In [5]:
Activate Windows
Go to Settings to activate Windows.



```
root = Tk()
root.configure(background='black')
Symptom1 = StringVar()
Symptom1.set("Select Here")
Symptom2 = StringVar()
Symptom2.set("Select Here")
Symptom3 = StringVar()
Symptom3.set("Select Here")
Symptom4 = StringVar()
Symptom4.set("Select Here")
Symptom5 = StringVar()
Symptom5.set("Select Here")
Name = StringVar()
w2 = Label(root, justify=LEFT, text="Disease Predictor using Machine Learning", fg="Red", bg="White
w2.config(font=("Times",30,"bold italic"))
w2.grid(row=1, column=0, columnspan=2, padx=100)
w2 = Label(root, justify=LEFT, text="A Project by Sharad Kumar", fg="Pink", bg="Blue")
w2.config(font=("Times",30,"bold italic"))
w2.grid(row=2, column=0, columnspan=2, padx=100)
NameLb = Label(root, text="Name of the Patient", fg="Red", bg="Sky Blue")
NameLb.config(font=("Times",15,"bold italic"))
NameLb.grid(row=6, column=0, pady=15, sticky=W)
S1Lb = Label(root, text="Symptom 1", fg="Blue", bg="Pink")
S1Lb.config(font=("Times",15,"bold italic"))
S1Lb.grid(row=7, column=0, pady=10, sticky=W)
S2Lb = Label(root, text="Symptom 2", fg="White", bg="Purple")
S2Lb.config(font=("Times",15,"bold italic"))
S2Lb.grid(row=8, column=0, pady=10, sticky=W)
S3Lb = Label(root, text="Symptom 3", fg="Green",bg="white")
S3Lb.config(font=("Times",15,"bold italic"))
S3Lb.grid(row=9, column=0, pady=10, sticky=W)
S4Lb = Label(root, text="Symptom 4", fg="blue", bg="Yellow")
S4Lb.config(font=("Times",15,"bold italic"))
S4Lb.grid(row=10, column=0, pady=10, sticky=W)
S5Lb = Label(root, text="Symptom 5", fg="purple", bg="light green")
S5Lb.config(font=("Times",15,"bold italic"))
S5Lb.grid(row=11, column=0, pady=10, sticky=W)
```

```
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")
```

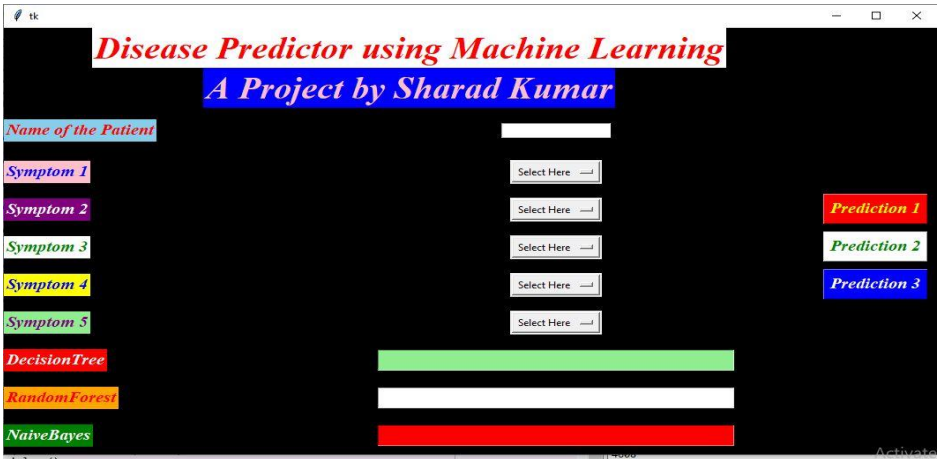


```
destreeLb.config(font=("Times",15,"bold italic"))
destreeLb.grid(row=17, column=0, pady=10, sticky=W)
ranflb = Label(root, text="NaiveBayes", fg="White", bg="green")
ranflb.config(font=("Times",15,"bold italic"))
ranflb.grid(row=19, column=0, pady=10, sticky=W)
OPTIONS = sorted(l1)
NameEn = Entry(root, textvariable=Name)
NameEn.grid(row=6, column=1)
S1 = OptionMenu(root, Symptom1,*OPTIONS)
S1.grid(row=7, column=1)
S2 = OptionMenu(root, Symptom2,*OPTIONS)
S2.grid(row=8, column=1)
S3 = OptionMenu(root, Symptom3,*OPTIONS)
S3.grid(row=9, column=1)
S4 = OptionMenu(root, Symptom4,*OPTIONS)
S4.grid(row=10, column=1)
S5 = OptionMenu(root, Symptom5,*OPTIONS)
S5.grid(row=11, column=1)
dst = Button(root, text="Prediction 1", command=DecisionTree,bg="Red",fg="yellow")
dst.config(font=("Times",15,"bold italic"))
dst.grid(row=8, column=3,padx=10)
rnf = Button(root, text="Prediction 2", command=randomforest,bg="White",fg="green")
rnf.config(font=("Times",15,"bold italic"))
rnf.grid(row=9, column=3,padx=10)
lr = Button(root, text="Prediction 3", command=NaiveBayes,bg="Blue",fg="white")
lr.config(font=("Times",15,"bold italic"))
lr.grid(row=10, column=3,padx=10)
t1 = Text(root, height=1, width=40,bg="Light green",fg="red")
t1.config(font=("Times",15,"bold italic"))
t1.grid(row=15, column=1, padx=10)
t2 = Text(root, height=1, width=40,bg="White",fg="Blue")
t2.config(font=("Times",15,"bold italic"))
t2.grid(row=17, column=1 , padx=10)
t3 = Text(root, height=1, width=40,bg="red",fg="white")
t3.config(font=("Times",15,"bold italic"))
t3.grid(row=19, column=1 , padx=10)
root.mainloop()
```

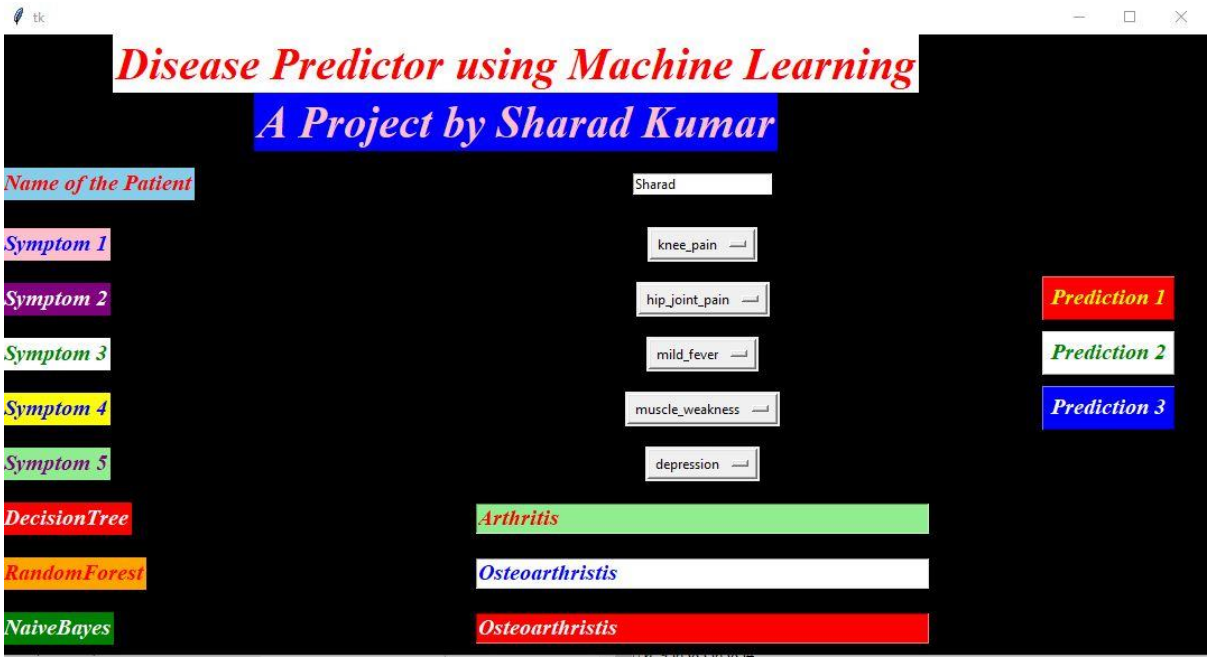
```
for a in range(0,len(disease)):
    if(predicted == a):
        h='yes'
        break
if (h=='yes'):
    t2.delete("1.0", END)
    t2.insert(END, disease[a])
else:
    t2.delete("1.0", END)
    t2.insert(END, "Not Found")
```

OUTPUT

Now its time to see how effective our code is and what it yields. Is it working as we are expected or not. We have done with our implementation section now its time to compile our code and get our result for which we have written so many lines of codes.



In the output we uses some columns to get the input of the patient we require the name of the patient firstly then we choose the symptoms of the disease the patient is suffering from and after the symptoms are inserted we click on the predict button to predict the output we had made different prediction using different algorithms for prediction 1 , prediction 2 and prediction 3 we uses decision tree, random forest and naive bayes respectively. Lets see what result we get after all the prediction.



As we see osteoarthritis is being predicted by our two algorithms that are random forest and naive bayes but decision tree predicted the arthritis. And it is intresting to know that the osteoarthritis is also a coomon form of arthritis in which person suffers from joint pain.

CONCLUSION

Hence we have seen implemented our model and seen output and performace we can even compare their accuracy and time taken for the execution of our problem till resulting an output we can compare each and every scenario hence we can now conclude which algorithm performs better and what is the result on each time we press predict we can get different output because it improvises itself if user is not satisfied with the result.

We have seen that Random Forest works better than other two algorithms with higher efficiency we can trust on its result and this may help in stopping various diseases in their initial stage.

FUTURE ENHANCEMENT

This is a small project which is window based and we will try to implement on various other platforms so all type of user can use it. Second most important thing right now we are only predicting the disease a patient is suffering from but we will going to improve it by adding best and specialist doctors against diseases from which patient is suffering from because many time we don’t even know whom to contact who is the best person, who can help us out. This the first version of our project we will improve it more as per user feedback.

REFERENCES

- Evans DG, Howell A. Breast cancer risk-assessment models. Breast Cancer Res. 2007 Sep 12;9(5):213. pmid:17888188
- U. S. Preventive Services Task Force [Internet]. Final Update Summary: Breast Cancer: Screening; 2019 May [cited 2019 Sep 20]. Available from: https://www.uspreventiveservicestaskforce.org/Page/Document/UpdateSummaryFinal/breast-cancer-screening
- https://towardsdatascience.com/building-a-simplemachine-learning-model-on-breast-cancer-dataeca4b3b99fa3
- https://www.ijitee.org/wpcontent/uploads/papers/v 8i6/F3384048619.pdf
- https://ieeexplore.ieee.org/document/7943207