CA3 PROJECT REPORT

**(**PANCREATIC CANCER DISEASE DETECTION USING ML ALGORITHMS**)**

Submitted by

**ASAM NARASIMHA REDDY REGESTRATION NUMBER: 12103515**

**Course Code: INT234**

**School of Computer Science and Engineering**



# ACKNOWLEDGEMENT

We would like to express our deepest appreciation to all those who have contributed to the successful completion of this computer science project.

First and foremost, we are indebted to our project supervisor, SANGEETA RAI, for their continuous guidance and unwavering support throughout the project. Their expertise and constructive feedback have been invaluable in shaping our ideas and approaches and have been instrumental in ensuring the quality of our work.

We would also like to extend our gratitude to our fellow classmates and colleagues, who have offered their valuable insights and suggestions throughout the course of our research. Their input has been crucial in refining our work and addressing any challenges that arose along the way.

Furthermore, we would like to acknowledge the importance of the resources provided by our institution, including access to cutting-edge software and hardware, as well as various online databases and libraries. These resources have allowed us to conduct thorough research and experimentation and have been crucial in the development of our project.

Finally, we would like to express our appreciation to our friends and families for their constant support and encouragement throughout this project. Their unwavering belief in our abilities has been a great source of motivation and inspiration for us, and we are truly grateful for their love and encouragement.

Once again, we extend our heartfelt thanks to all those who have supported us in this endeavor, and we hope that our work will contribute to the advancement of computer science and technology.

**Introduction:**

Pancreatic cancer is a great challenge in the field of oncology since it is a very severe danger to the health and welfare of people. Pancreatic adenocarcinoma or pancreatic cancer remains one of the most significant challenges in modern medicine, for this is a deadly disease. Since the disease tends to be late in diagnosis when it already has progressed, and many treatment options are nil, the prognosis of this disease is quite grim. This grim prognosis is further underscored by the urgent requirement for more effective early detection and successful treatment plans.The recent convergence of machine learning with healthcare has revolutionized the approach to problem- solving in challenging medical issues. Researchers and physicians have developed new diagnostic and prediction technologies that hold great promise for the pancreatic cancer epidemic by applying complex algorithms and data-driven insights. This emerging field of study opens up new possibilities for tailored treatment plans, early intervention, and finally, better therapeutic outcomes.Our goal, in this research, is to use machine learning to solve this urgent problem of pancreatic cancer prediction and detection. Applying a wide range of various machine learning algorithms: decision trees, k-nearest neighbors, random forests, neural networks, support vector machines, and Naive-Bayes, in developing reliable and accurate models would enable prediction about the existence of pancreatic disease from urine biomarkers. Because these biomarkers play a part in tumor metastasis, pancreatic regeneration, and urinary tract health, they have emerged as potential indicators in the early detection of pancreatic cancer. These biomarkers include LYVE1, REG1B, TFF1, and REG1A.This study does not simply stop at developing predictive models but studies the best reliable and accurate algorithms in predicting pancreatic illness in a detailed comparative study. The study seeks to provide major perspectives that will help healthcare professionals guide the course for the future investigation in this critical field. Finally, in the fight against pancreatic cancer, this work makes use of the best of machine learning algorithms and data- driven wisdom to bring hope. There is the point of delivering improved clinical outcomes, making medical practitioners more powerful, and reducing the immense burden that pancreatic cancer places upon people and the community at large. One hopes that these lay open doors for better treatments, individualized treatment plans, and, one day, a better future for those afflicted by this terrible disease through early detection and accurate prediction.

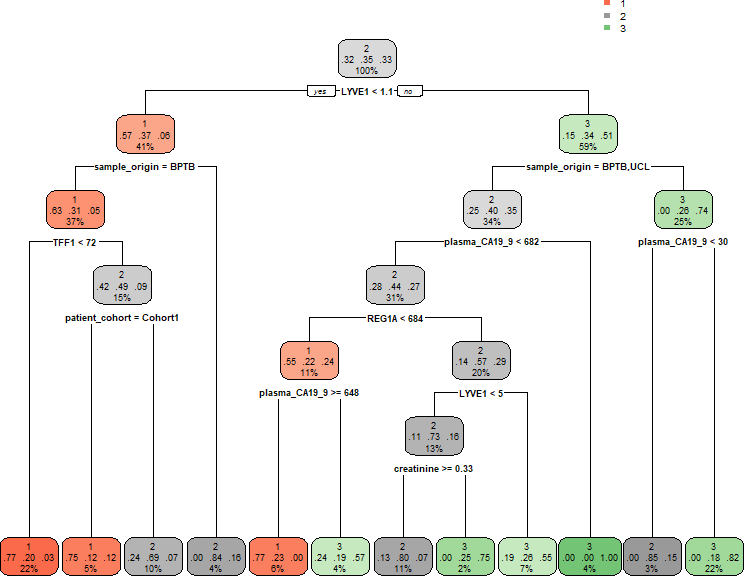
# Implementation of Project:

Dataset Information: this dataset is imported from Kaggle that contain information of potential urinary biomarkers like TFF1, LYVE1, REG1A, REG1B of 590 patients and along with these urinary biomarkers there is a diagnosis column that is a categorical variable in which 1- no tumor 2-non pancreatic condition 3- pancreatic cancer condition along with these there is also an age, gender, and sample origin column which indicates from which location that sample is taken from and stage column which indicates the stage of the pancreatic cancer if the patient has pancreatic cancer.

Decision tree: Decision tree: It is a supervised machine learning algorithm Used for both classification and regression since our dataset needs to classify the patients into pancreatic cancer patients ,non-pancreatic cancer patients so this algorithm is very much suitable it is a tree like structure where each internal node represents feature based split and the branches represents the

outcome of the feature-based split this process continues until we run out of the features to do feature based split or the tree has grown than its pre determined size and the leaf nodes of decision tree represents the class labels and contains all the record of the same class the dataset is imported and then split it into training and testing data subsets and we used the rpart package which contains the rpart function which recursively partitions the dataset until we run out of the features and training dataset is given to this function to perform the same to develop a decision tree model on this dataset





In the above code we have imported rpart which is the shorthand form of recursive partitioning and rpart.plot to plot the decision tree we made after that we read the csv file containing the data using read,csv () function and stored it in the variable df in the form of dataframe and after that we removed first column which is basically record number and sample\_cohort column like record number it does not contribute in predicting the output so we exclude these columns and then we convert all categorical feature columns into categoric like stage of pancreatic cancer, sex, and sample-origin and then we split the dataset into train and test with train dataset containing 80% of the records and test dataset containing 20% records and then we provide rpart function the input and output relationship and training data to develop a decision tree model and then we provide the developed model with testing data and then we take predicted values of the model and actual labels to create confusion matrix and then we calculate accuracy which happened to be 85%.

SVM:Support-vector-machine: Svm is also used for both regression and classification the main of svm is to find the best hyperplane, and hyperplane is nothing but a boundary in the n dimensional feature that exactly divides the data points in the feature space into different labels the best hyper plane is that which accurately classifies the datapoints into their respective categories and the one

that is very much distant from the nearest data-points of all the classes in our model we have imported a package called e1071 which provides us to train our svm classifier with help of svm function to train our model we fed the input output relationship and training data into svm function which then gives us the svm classifier since the svm needs to find the hyperplane that is at the maximum distance from the nearest datapoints of all classes there is a need of normalization of continuous variables for better accuracy



This code contains a number of crucial lines needed to build and assess the SVM classifier constructed with the e1071 package.The relevant libraries are first loaded, e1071 for SVM compilation and caTools for data separation. The CSV file's dataset is then added to the dataframe df. Using the str() command, the structure of the df-typed object is seen both before and after data preprocessing.One of the key components in the machine learning process is thought to be the preprocessing of the data. The dataset is divided into the required columns for the SVM in the code above, and the categorical variables are factorized.The target variable diagnosis is arranged at the end of the dataframe by rearranging it. Next, the sample() function divides the data into two sets: the testing set (ts) and the training set (tr). An 80:20 split ratio is used to further divide the data, setting aside 80% for training and 20% for testing. After splitting the data, the next step is to use the SVM() function from the e1071 package to train an SVM classifier. In order to establish a linear kernel, the function builds the SVM model. The training set, tr, is then used to train the model. Lastly, a prediction is made on the test set ts by the model using the predicted() function, and the result is saved in the predictions variable.The table() method is first coded to create a confusion matrix, which is then used to determine how well our model targets outperforms an existing model. Subsequently, the confusion matrix is utilized to establish the skill evaluation of the model. Next, the accuracy metric—which measures the percentage of correctly classified cases—is estimated based on the total number of scenarios in the testing set.

KNN:

Knn: knn is the acronym for k-nearest neighbors and it is also used for both classification and regression since our data set involves classification of patients into pancreatic cancer and non- pancreatic cancer patients this algorithm is one of best algorithms for classification the main idea behind knn is that similar features have similar labels so when we give the data for training it forms clusters of the data points that have similar features and similar labels so when a new unclassified datapoint is given it tries to find the k-nearest datapoints that have similar features and identify the major label or class that most of these k-data points belong to and classify this new unclassified data point as the datapoint of that major class label ideally it is said that square root of the number of records is best value for k but it can be found through trial and error we have imported the class package split the dataset into training and testing and then used in knn function in the class package to develop the knn model on this dataset and we have gotten best accuracy for k = 1



To begin with the code loads necessary libraries for data manipulation and KNN classification. Then it reads a CSV file which contains the dataset into a data frame named “df”. A first look at the data structure is done in order to get its variables and types. Useless columns are then dropped, leaving only those relevant for analysis. Categorical variables such as "sex", "sample\_origin", "patient\_cohort" and "stage" are converted into numeric format so that they can be analyzed easily. After this step, the data frame is divided again into two parts; “df1” and “df2” where selected columns are kept for each of them respectively. Numerical features in “df2” are scaled using scale() function for normalization purposes after which these scaled features are combined with other selected columns from “df1” to create unified dataset called “df”. The data is now randomly split into training (“tr”) and testing (“ts”) sets with ratio 75:25. Then two different KNN classifiers are trained on training set with k values being 1 and 2 respectively before being tested on testing set afterwards. Finally confusion matrices are produced to assess how well these models performed during training phase together with calculating their accuracy which will tell us whether or not they can be effective when it comes to classification tasks.

Random Forest:

Random-forest: random forest is also one the supervised machine learning algorithms which is also used for both classification and regression Random-forest uses a bunch of decision trees in the background and takes the output from the all this decision trees and then labels the unclassified data-point as the class that is the majority in outputs of the decision trees in our model we have imported a package called random Forest and divided the data-set into training and testing and then used random Forest function to develop the random Forest model by providing input-output relationship and training the data and we can also decide how many trees we want to use so that we can develop random forest using ntree parameter in our model we have set the ntree parameter to 500.



This R code snippet will predict using random forest classifiers from a dataset. At first, it loads the dataset ("clear\_data.csv") and selects particular columns of interest. The categorical variables such as ‘sex’,’ sample\_origin’, ‘patient\_cohort’ and stage are converted to numeric format by use of ‘as.integer(as.factor(...))’ while diagnosis is left as a factor. Further on, dataframe is enhanced where specific columns for modeling are chosen and then divided into two parts (df1 and df2). Numeric columns in ‘df2’ are scaled using ‘scale()’ , combined with original categorical columns from ‘df1’ using ‘cbind()’ to create final preprocessed dataframe (df). 80:20 ratio randomly splits the dataset into training (tr) and testing (ts) sets. A random forest classifier (classifier) is trained on the training data (tr) using randomForest() function; predictor variables (x) excluding diagnosis column while diagnosis column itself serves as target variable(y). Predictions made on test set (ts) by trained model where confusion matrix(cm) is computed so that model’s performance can be evaluated. The accuracy of the model is then calculated based on the confusion matrix to assess the predictive accuracy. This process demonstrates a comprehensive workflow for preparing data, building a random forest model, and evaluating its performance in R using the ‘randomForest’ library.

NaiveBayes: Like knn and decision tree naive bayes is a supervised machine learning algorithm that is used for classification and regression the reason behind calling the bayes classifier naive is the assumption that it makes that all features that helps in predicting output are independent and in addition to that it also assumes the continuous features to be normally distributed and it also views or gives all the features same importance to develop this model we have normalized all the continuous urinary biomarkers because of this classifiers assumptions and then dataset is split into testing and training e1071 package contains naive Bayes function which generates naive Bayes classifier when it is fed with input-output relationship along with the training data

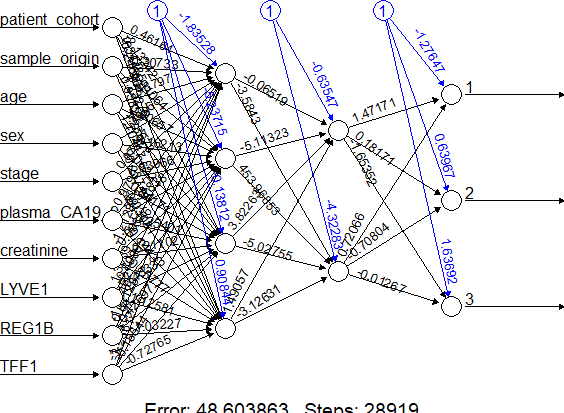
The Code Example demonstrates use of the R e1071 package for creating a Naive Bayes classifier. Loading the required libraries (caTools and e1071) and importation of data from an initial clear data.csv file are also parts of this code. 1. Column selection and conversion of categorical variables to factors which in turn alters the data frame df takes place at this point. Later on it selects only several columns (information hiding). To ensure repeatability/randomness control in results generation, seed number 1234 has been used (set.seed(1234)).To split the data into two sets: training set (tr) and test set (ts), sample function is for division of the data. This specifies the

fraction of data that should be used for training and testing in terms of prob (65% and 35% respectively).In order to train the Naive Bayes classifier (classifier\_cl), we make use of the diagnosis as the target variable using the training set (tr). Once done with that, we can formulate a confusion matrix (cm) so that we may draw a line between projected outcomes against real observations within the testing data. This is usually done by comparing expected labels with what actually happens after completing corresponding tasks within a dataset known as confusion matrix. Then using the test set, the model is used to generate predictions (y\_pred), uses (ts) to give predictions (y\_pred), after which we build the confusion matrix (cm) to compare the predicted values against the actual ones in the test data. Finally, we measure how well cv work through dividing number correct and total cv categorical instances fοr which it produced valid outputs



Neural-network: The main inspiration behind the neural networks is to create a model which imitates the way human brain works in neural networks there are 3 layers input layer, hidden layers and output layer for every feature there is a node in the input layer and after taking input and multiplying it with the weights it sends signal to hidden layer and which in turn processes these inputs from input layer and sends it to the output layer We have imported a module called neural net which contain a function called neuralnet which when provided with the input-output relationship and training data gives an artificial neural network in our 2 layers with 4 neurons

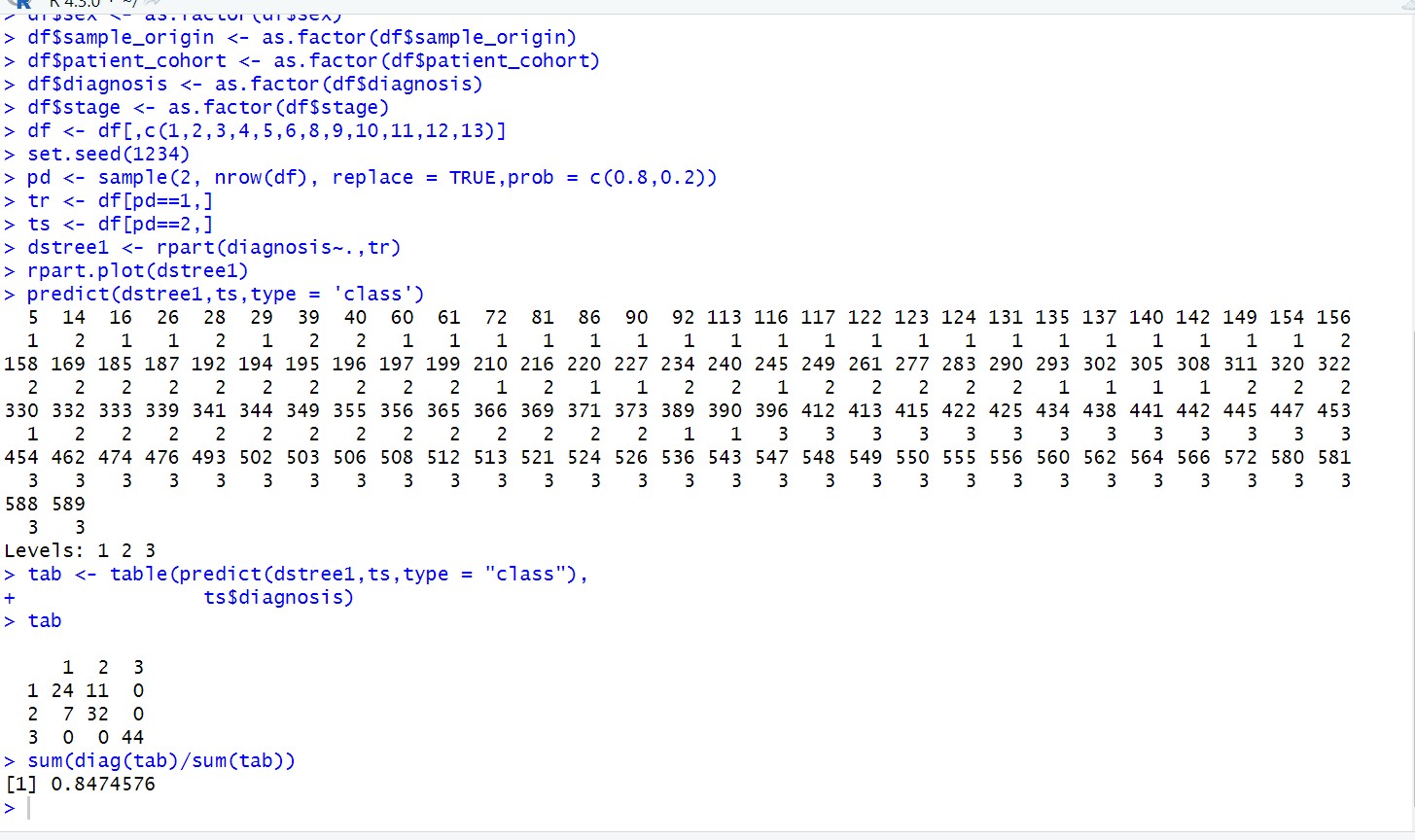




In the above code we have imported a library called neuralnet and then dataset on which we want to develop model is read using read.csv() function in the form of dataframe which is stored in df variable the reason behind importing neuralnet package is it contains the function called neuralnet which can develop a neuralnetwork model neuralnet() function demands all its features to be numeric so we have converted all the categorical features to numeric by first converting them into factors using as.factor() and then we converted the factors into numeric using as.numeric() function and then we used sample() function to split the dataset into train and test and then we provided the neuralnet function with input,output relationship and trainingdata(tr) and the number of layers that should be present in our neural network since we have provided 2 our neural network has 2 layers and then we have want each layers of neural network two have 4 neurons and then we use compute() function to test the developed model on the testing dataset and then compute function gives the predicted values and then we use table() function to create confusion matrix and then we findout accuracy metric from this confusion matrix.

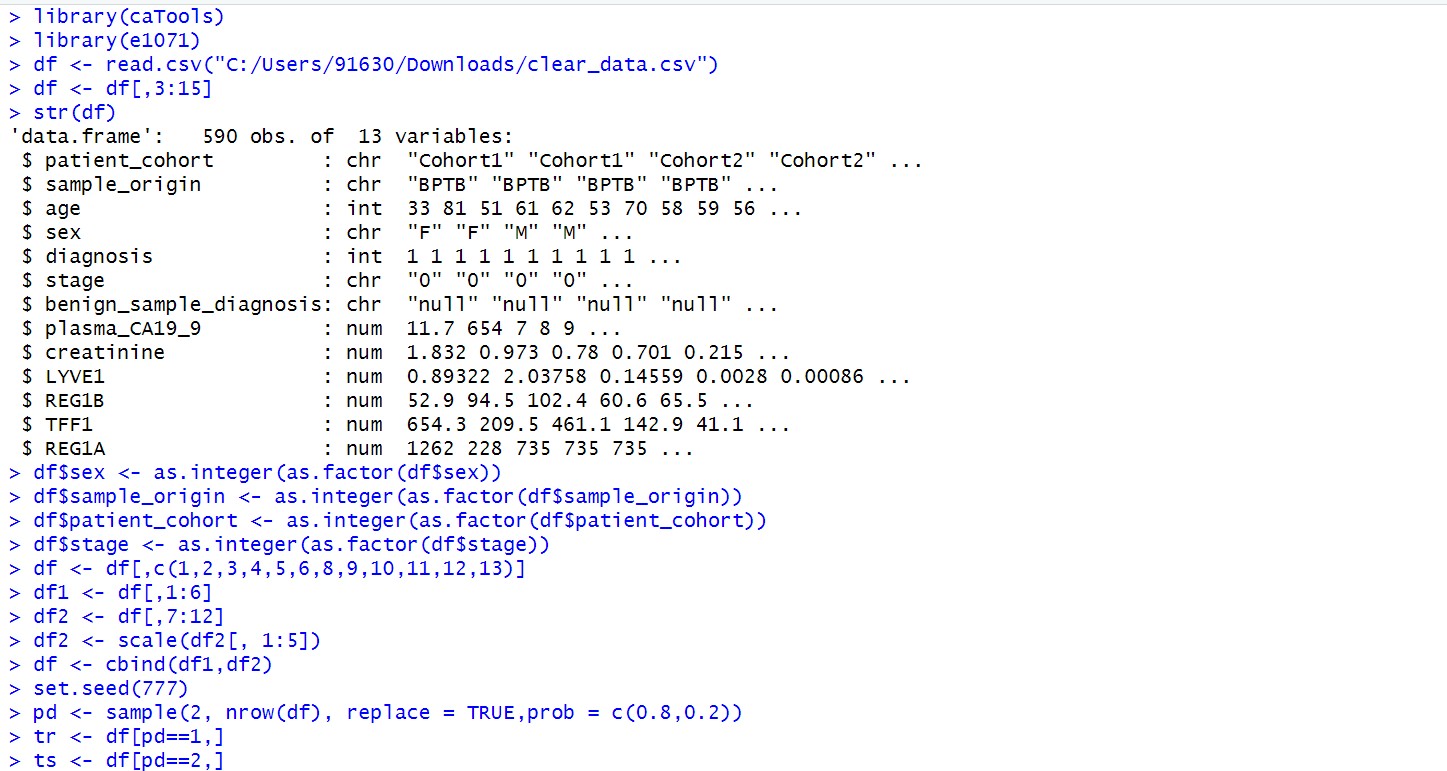
# Results and Discussions:

Outputs for the decision tree:



After executing accuracy is found to be 84.74576 for decision tree and best ratio for split the dataset into train and test for decision is found to be 80% training and 20% testing

Outputs for SVM (Support Vector Machine):





We can see that for svm after executing the accuracy is 86.60% and the best split ratio for svm is found to be 80% training and 20% testing

Output for KNN:



For KNN the accuracy is found to be 88.88% and the best split ratio is found to be 75%training and 25% testing.

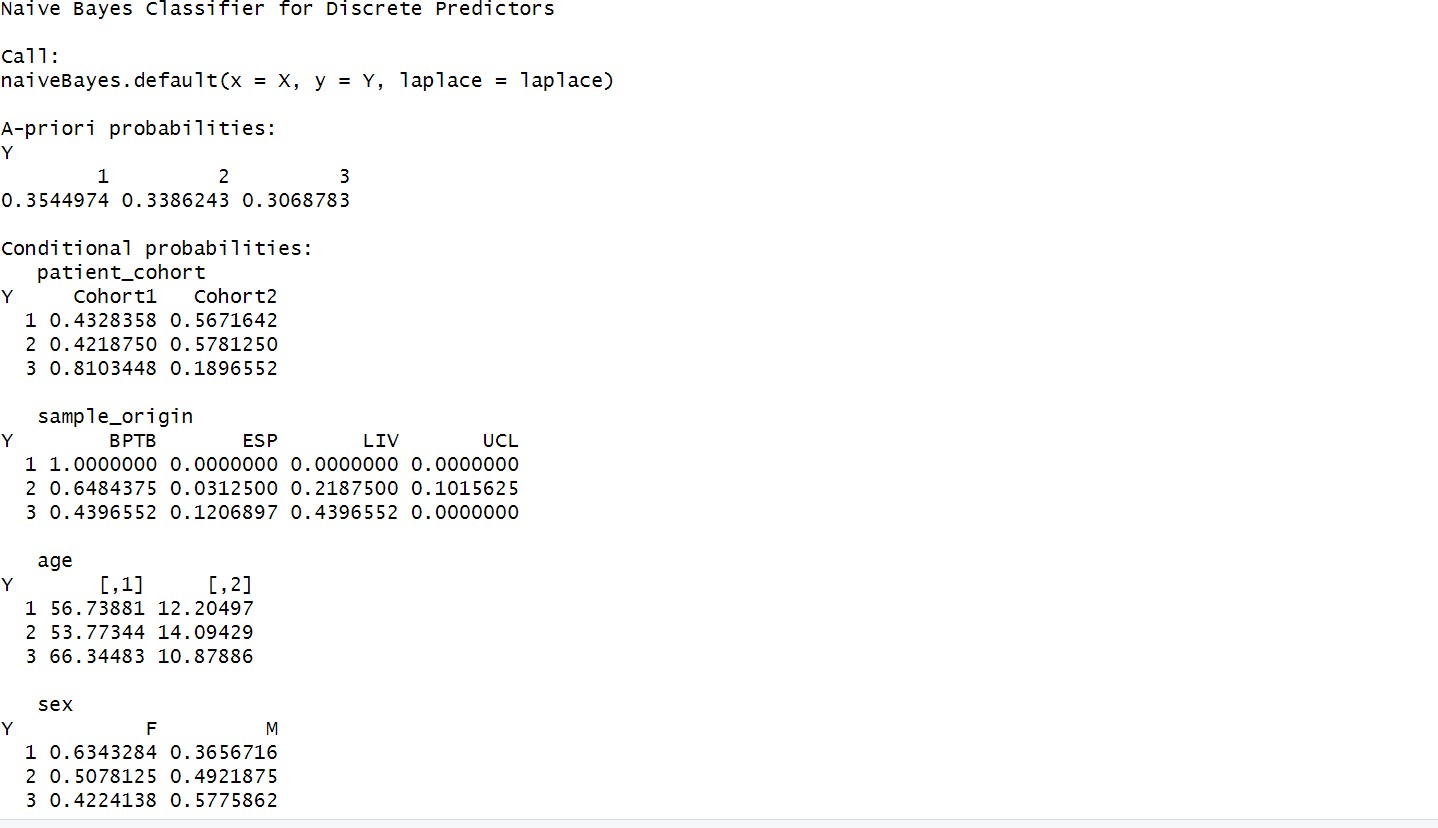
Output for Random Forest:

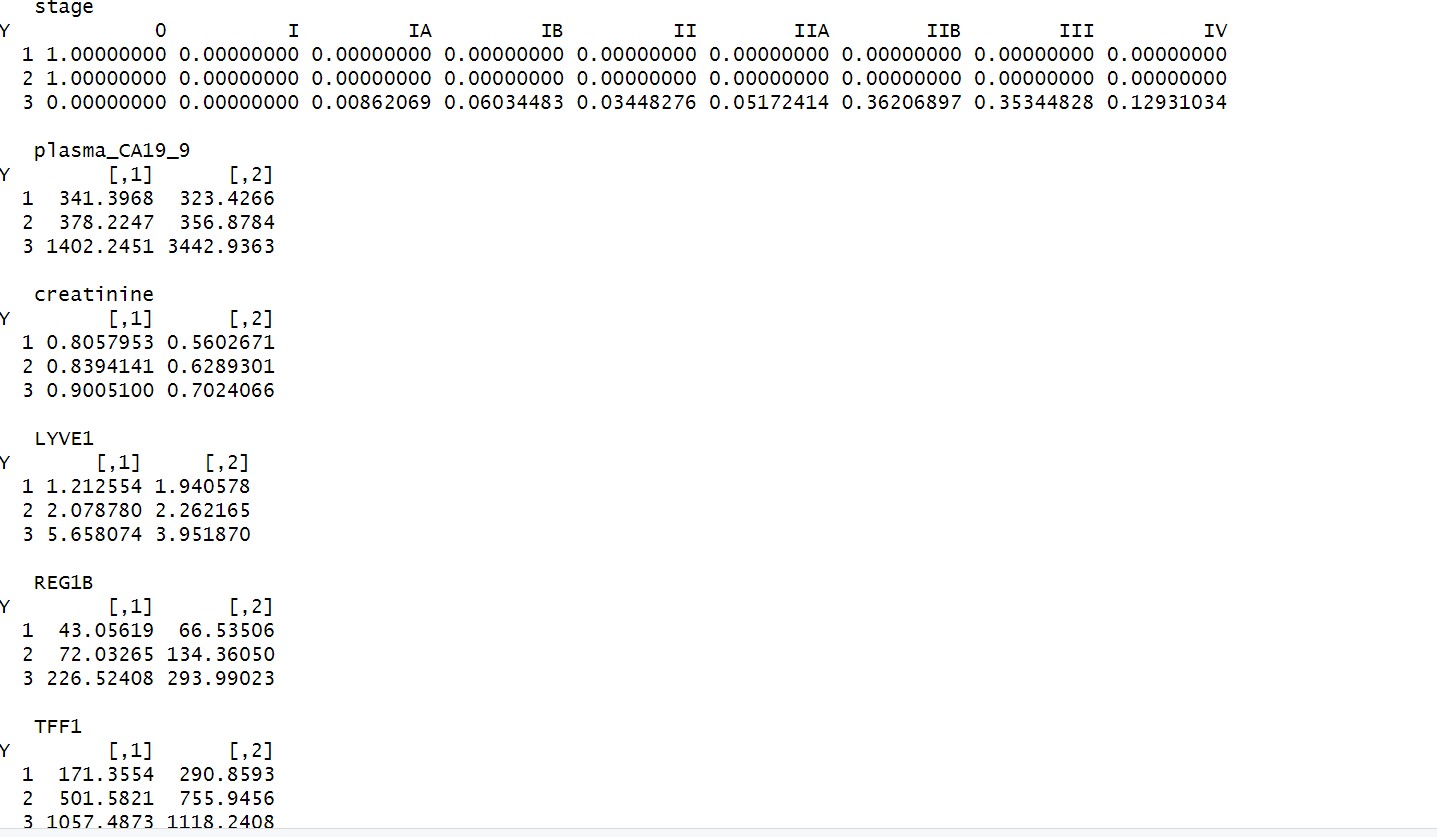


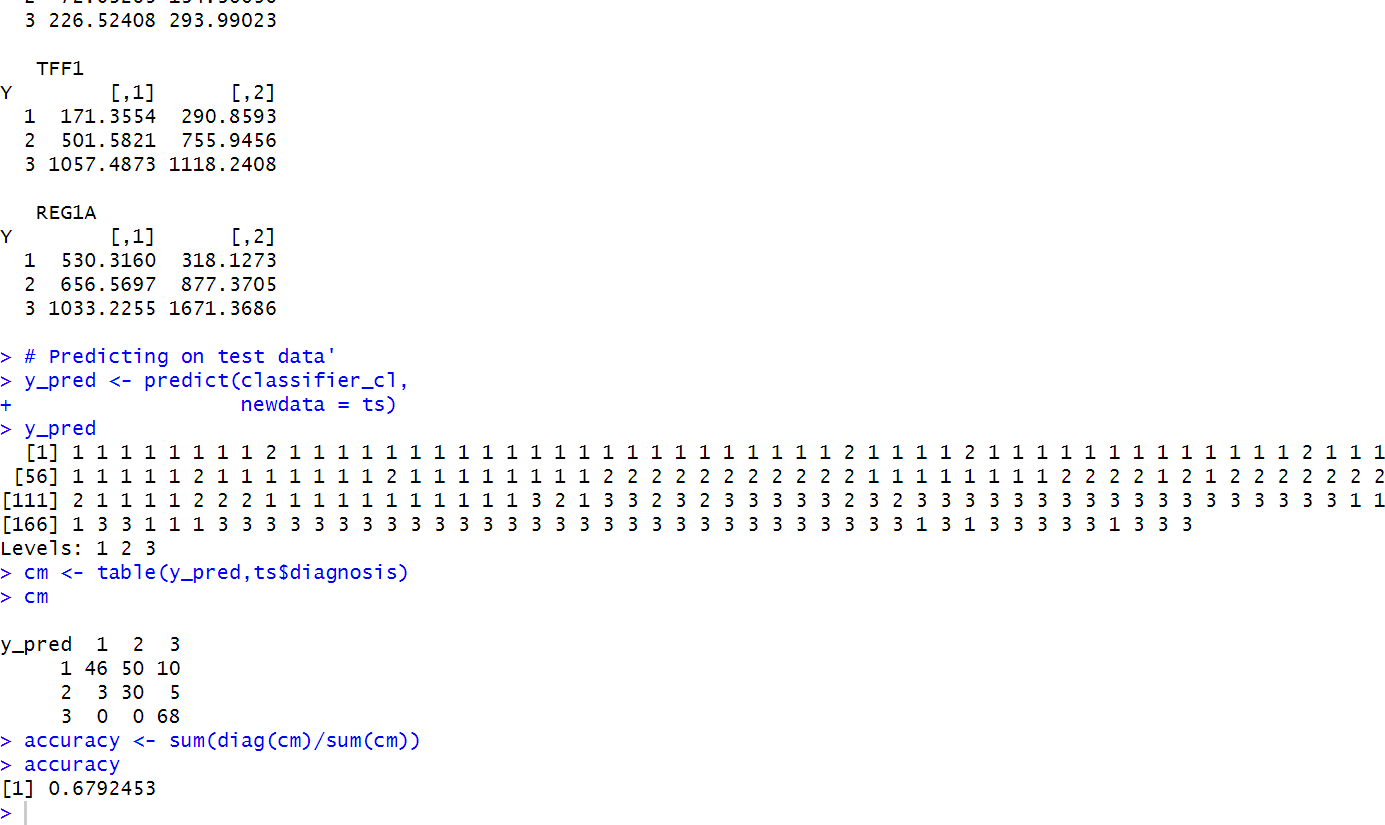
We have found that the accuracy of random forest is 91.07% and the best split Ratio for random forest is found to be 80% training, 20% testing.

Output for NaiveBayes:



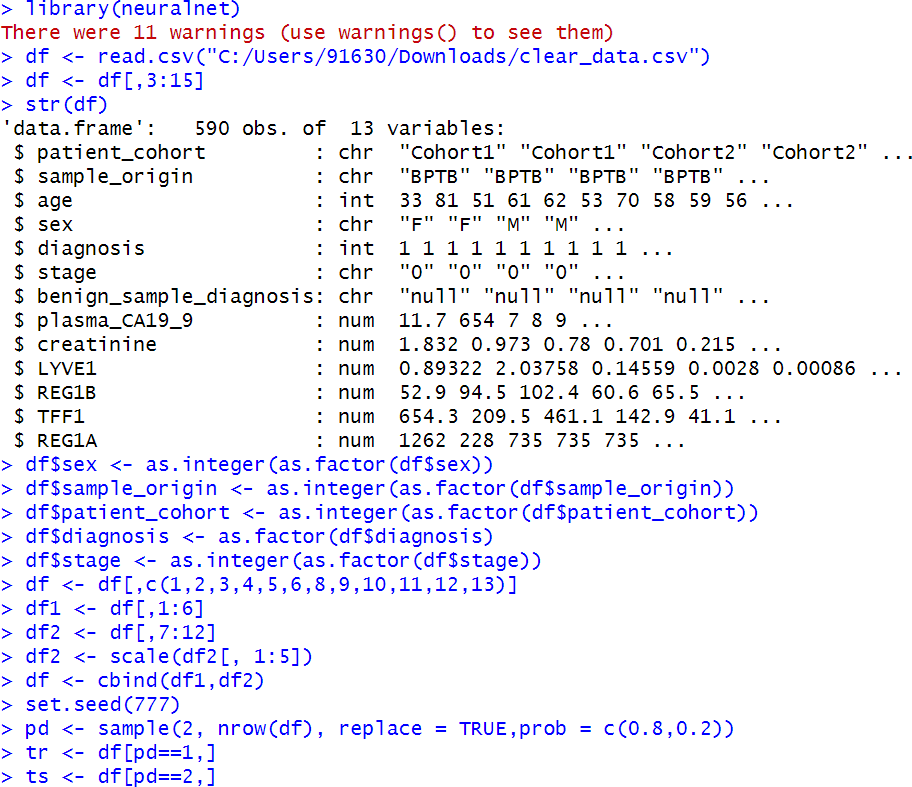


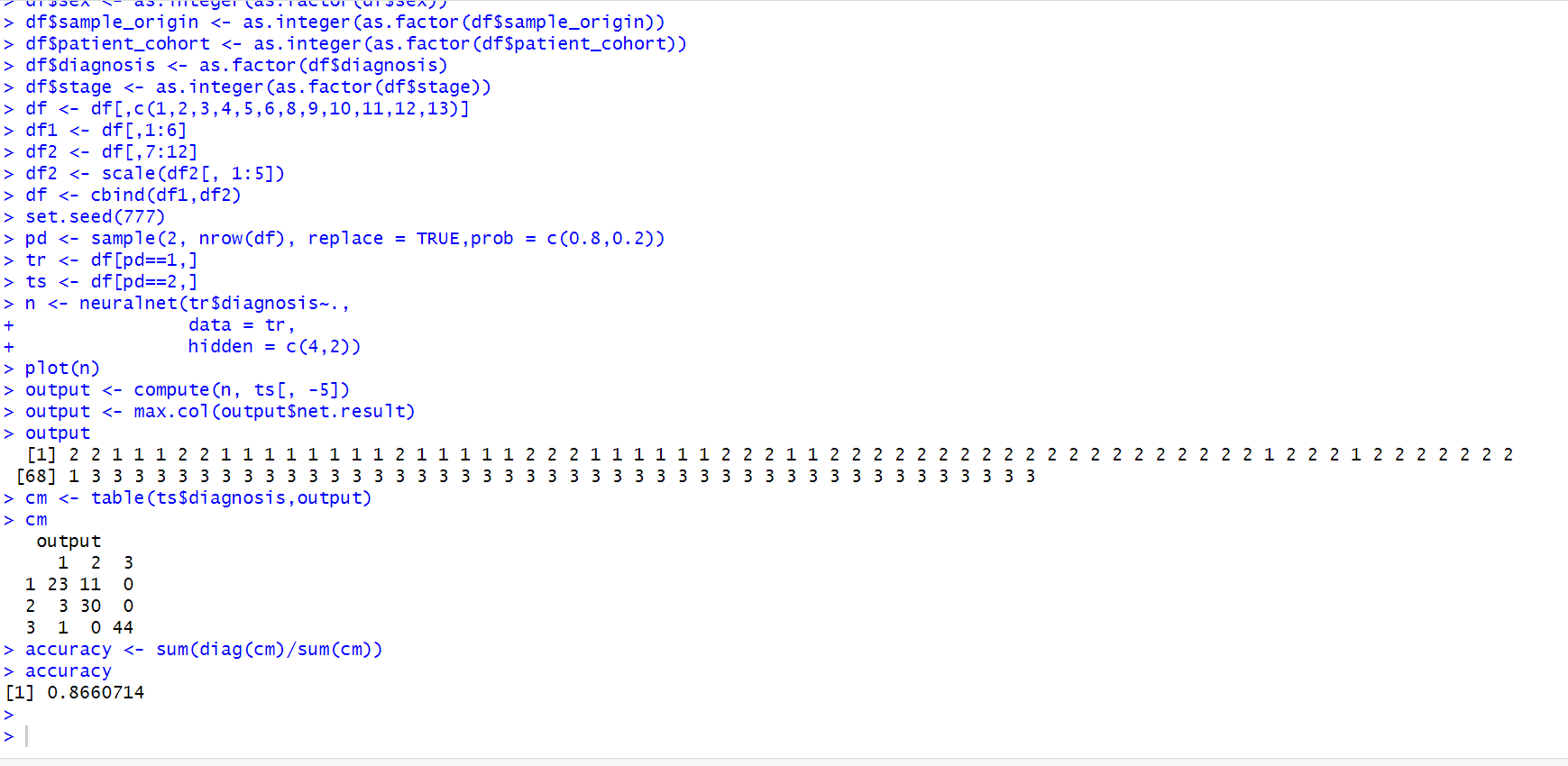




We have found that the accuracy of naivebayes is found to be 67.9% approximately 68% and the best splitRatio for naïve bayes is found to be 65% training and 35% testing.

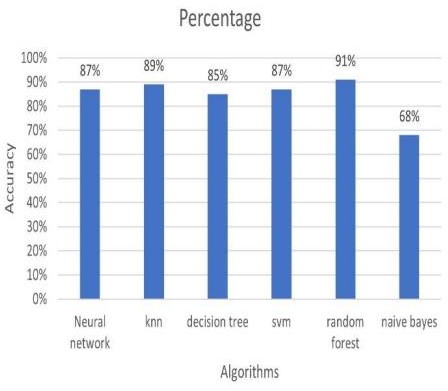
Output for neural Network:





The accuracy for neural network is found to be 86.6% and the best split ratio for neural networks is found to be 80 % training and 20%testing

# Conclusion and future scope



|  |  |
| --- | --- |
| Algorithms | Accuracy |
| Neural Network | 87% |
| KNN | 89% |
| Decision Tree | 85% |
| SVM | 87% |
| Random Forest | 91% |
| Naive Bayes | 68% |

This research explored the potential of various machine learning algorithms for predicting pancreatic cancer the investigation focused on six prominent algorithms Random Forest, nearest neighbors(KNN),Support vector machine (SVM),Decision tree, Neural Network,NaiveBayes The analysis revealed that Random Forest emerged as the most effective algorithm with an accuracy of 91.07% This suggests that Random Forest has the strongest capability to accurately distinguish between pancreatic cancer and noncancerous cases within the given dataset and other algorithms like KNN(89%) and SVM (87%) also demonstrated promising results indicating their potential ability in pancreatic cancer prediction