**CHRONIC KIDNEY DISEASE ANALYSIS USING DATA MINING CLASSIFICATION TECHNIQUES**

In this paper author is using SVM (Support Vector Machine), Decision Tree Classifier, K-Neighbors Classifier, naïve bayes, Random Forest Classifier, Bagging Classifier, Ada Boost Classifier and MLP Classifier and artificial neural network algorithm.

All the algorithms generate model from train dataset and new data will be applied on train model to predict it class. Random Forest algorithm is giving better prediction accuracy compare to all other algorithm.

**Support vector machine:**

Machine learning involves predicting and classifying data and to do so we employ various machine learning algorithms according to the dataset. SVM or Support Vector Machine is a linear model for classification and regression problems. It can solve linear and non-linear problems and work well for many practical problems. The idea of SVM is simple: The algorithm creates a line or a hyper plane which separates the data into classes. In machine learning, the radial basis function kernel, or RBF kernel, is a popular kernel function used in various kernelized learning algorithms. In particular, it is commonly used in support vector machine classification. As a simple example, for a classification task with only two features (like the image above), you can think of a hyper plane as a line that linearly separates and classifies a set of data.

Intuitively, the further from the hyper plane our data points lie, the more confident we are that they have been correctly classified. We therefore want our data points to be as far away from the hyper plane as possible, while still being on the correct side of it.

So when new testing data is added, whatever side of the hyper plane it lands will decide the class that we assign to it.

How do we find the right hyper plane?

Or, in other words, how do we best segregate the two classes within the data?

The distance between the hyper plane and the nearest data point from either set is known as the margin. The goal is to choose a hyper plane with the greatest possible margin between the hyper plane and any point within the training set, giving a greater chance of new data being classified correctly.

Both algorithms generate model from train dataset and new data will be applied on train model to predict it class. SVM algorithm is giving better prediction accuracy compare to ANN algorithm.

**Naïve Bayes Classifier Algorithm**

It would be difficult and practically impossible to classify a web page, a document, an email or any other lengthy text notes manually. This is where Naïve Bayes Classifier machine learning algorithm comes to the rescue. A classifier is a function that allocates a population’s element value from one of the available categories. For instance, Spam Filtering is a popular application of Naïve Bayes algorithm. Spam filter here, is a classifier that assigns a label “Spam” or “Not Spam” to all the emails.

Naïve Bayes Classifier is amongst the most popular learning method grouped by similarities that works on the popular Bayes Theorem of Probability- to build machine learning models particularly for disease prediction and document classification. It is a simple classification of words based on Bayes Probability Theorem for subjective analysis of content.

**Decision tree:**

A decision tree is a graphical representation that makes use of branching methodology to exemplify all possible outcomes of a decision, based on certain conditions. In a decision tree, the internal node represents a test on the attribute, each branch of the tree represents the outcome of the test and the leaf node represents a particular class label i.e. the decision made after computing all of the attributes.

The classification rules are represented through the path from root to the leaf node.

**Types of Decision Trees**

**Classification Trees-** These are considered as the default kind of decision trees used to separate a dataset into different classes, based on the response variable. These are generally used when the response variable is categorical in nature.

**Regression Trees-**When the response or target variable is continuous or numerical, regression trees are used. These are generally used in predictive type of problems when compared to classification.

Decision trees can also be classified into two types, based on the type of target variable- Continuous Variable Decision Trees and Binary Variable Decision Trees. It is the target variable that helps decide what kind of decision tree would be required for a particular problem.

**Random forest:**

Random Forest is the go to machine learning algorithm that uses a bagging approach to create a bunch of decision trees with random subset of the data. A model is trained several times on random sample of the dataset to achieve good prediction performance from the random forest algorithm. In this ensemble learning method, the output of all the decision trees in the random forest, is combined to make the final prediction. The final prediction of the random forest algorithm is derived by polling the results of each decision tree or just by going with a prediction that appears the most times in the decision trees.

For instance, in the above example - if 5 friends decide that you will like restaurant R but only 2 friends decide that you will not like the restaurant then the final prediction is that, you will like restaurant R as majority always wins.

**K – nearest neighbor:**

***K-nearest* neighbor’s algorithm** (***k*-NN**) is a non parametric method used for classification and regression In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

* In *k-NN classification*, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor.
* In *k-NN regression*, the output is the property value for the object. This value is the average of the values of *k* nearest neighbors.

*K-NN* is a type of instant-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification.

Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/*d*, where *d* is the distance to the neighbor

The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

A peculiarity of the *k*-NN algorithm is that it is sensitive to the local structure of the data.

**Bagging classifier:**

A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction. Such a meta-estimator can typically be used as a way to reduce the variance of a black-box estimator (e.g., a decision tree), by introducing randomization into its construction procedure and then making an ensemble out of it.  
Each base classifier is trained in parallel with a training set which is generated by randomly drawing, with replacement, N examples (or data) from the original training dataset – where N is the size of the original training set. Training set for each of the base classifiers is independent of each other. Many of the original data may be repeated in the resulting training set while others may be left out.

Bagging reduces over fitting (variance) by averaging or voting, however, this leads to an increase in bias, which is compensated by the reduction in variance though.

AdaBoost:

*Adaptive boosting* is a machine learning meat algorithm formulated. It can be used in conjunction with many other types of learning algorithms to improve performance. The output of the other learning algorithms ('weak learners') is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems it can be less susceptible to the over fitting problem than other learning algorithms. The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing, the final model can be proven to converge to a strong learner.

Every learning algorithm tends to suit some problem types better than others, and typically has many different parameters and configurations to adjust before it achieves optimal performance on a dataset, AdaBoost is often referred to as the best out-of-the-box classifier.[[2]](https://en.wikipedia.org/wiki/AdaBoost#cite_note-2) When used with decision tree learning, information gathered at each stage of the AdaBoost algorithm about the relative 'hardness' of each training sample is fed into the tree growing algorithm such that later trees tend to focus on harder-to-classify examples.

**Multilayer perceptron (MLP):**

A **multilayer perceptron**(MLP) is a class of feed forward artificial neural network (ANN). The term MLP is used ambiguously, sometimes loosely to refer to *any* feed forward ANN, sometimes strictly to refer to networks composed of multiple layers of perceptrons (with threshold activation); see § Terminology. Multilayer perceptrons are sometimes colloquially referred to as "vanilla" neural networks, especially when they have a single hidden layer.

An MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called back propagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

**Artificial neuron network (ANN):**

An artificial neuron network (ANN) is a computational model based on the structure and functions of biological neural networks. Information that flows through the network affects the structure of the ANN because a neural network changes - or learns, in a sense - based on that input and output.

ANNs are considered nonlinear statistical data modelling tools where the complex relationships between inputs and outputs are modelled or patterns are found.

ANN is also known as a neural network.

An ANN has several advantages but one of the most recognized of these is the fact that it can actually learn from observing data sets. In this way, ANN is used as a random function approximation tool. These types of tools help estimate the most cost-effective and ideal methods for arriving at solutions while defining computing functions or distributions. ANN takes data samples rather than entire data sets to arrive at solutions, which saves both time and money. ANNs are considered fairly simple mathematical models to enhance existing data analysis technologies.

ANNs have three layers that are interconnected. The first layer consists of input neurons. Those neurons send data on to the second layer, which in turn sends the output neurons to the third layer.

Training an artificial neural network involves choosing from allowed models for which there are several associated algorithms.

To implement above all algorithms we have used python technology and ‘student data’ dataset.This dataset available inside dataset folder which contains test dataset with dataset information file.

Python Packages and Libraries used: Numpy, pandas, tkinter,

|  |  |  |
| --- | --- | --- |
| PyVISA | 1.10.1 | 1.10.1 |
| PyVISA-py | 0.3.1 | 0.3.1 |
| cycler | 0.10.0 | 0.10.0 |
| imutils | 0.5.3 | 0.5.3 |
| joblib | 0.14.1 | 0.14.1 |
| kiwisolver | 1.1.0 | 1.1.0 |
| matplotlib | 3.1.2 | 3.1.2 |
| nltk | 3.4.5 | 3.4.5 |
| numpy | 1.18.1 | 1.18.1 |
| opencv-python | 4.1.2.30 | 4.1.2.30 |
| pandas | 0.25.3 | 0.25.3 |
| pip | 19.0.3 | 20.0.1 |
| pylab | 0.0.2 | 0.0.2 |
| pyparsing | 2.4.6 | 2.4.6 |
| python-dateutil | 2.8.1 | 2.8.1 |
| pytz | 2019.3 | 2019.3 |
| pyusb | 1.0.2 | 1.0.2 |
| scikit-learn | 0.22.1 | 0.22.1 |
| scipy | 1.4.1 | 1.4.1 |
| seaborn | 0.9.0 | 0.9.0 |
| setuptools | 40.8.0 | 45.1.0 |
| six | 1.14.0 | 1.14.0 |
| sklearn | 0.0 | 0.0 |
| style | 1.1.6 | 1.1.6 |
| styled | 0.2.0.post1 | 0.2.0.post1 |

classification report, confusion matrix, accuracy score, train\_test\_split, K-Fold, cross\_val\_score, Grid Search CV, Decision Tree Classifier, K-Neighbors Classifier, SVC, naive\_bayes, Random Forest Classifier, Bagging Classifier, Ada Boost Classifier, MLP Classifier.

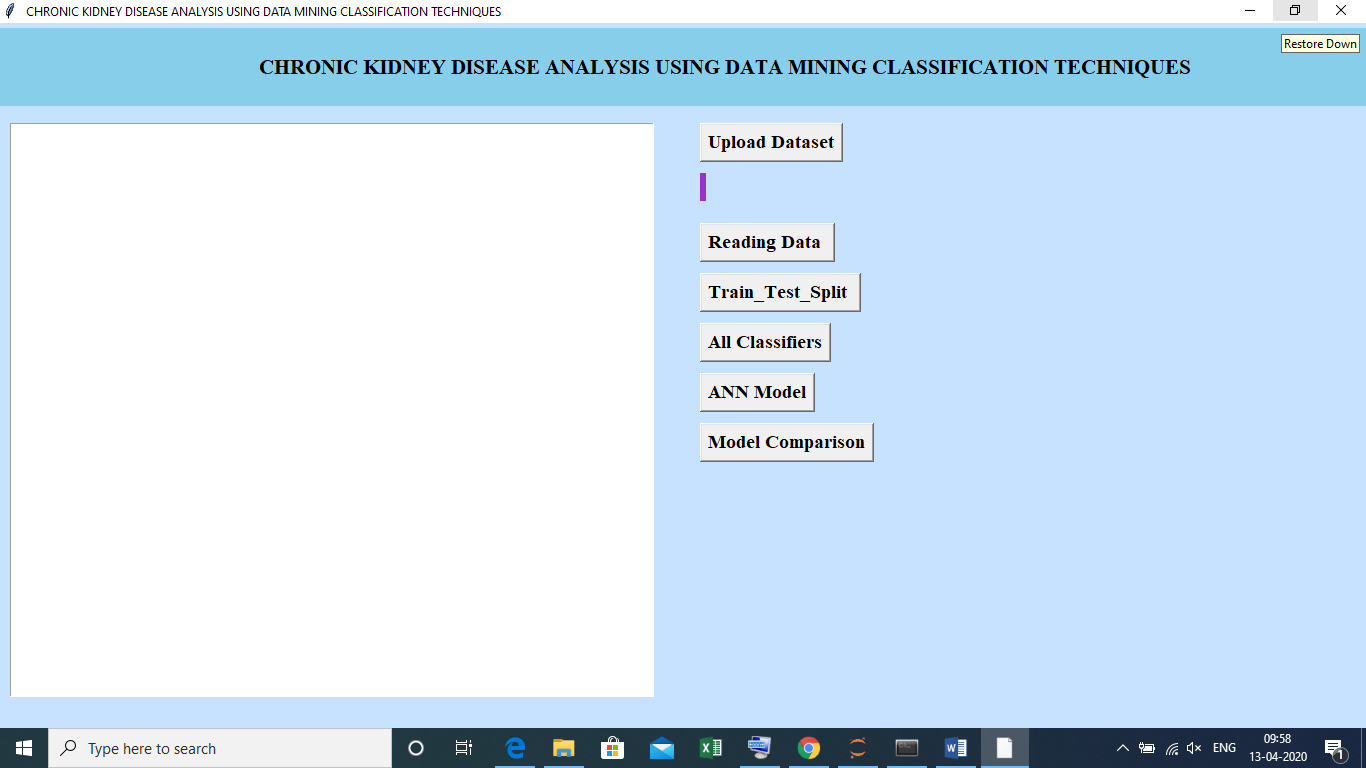
Screen shots

1. Open anaconda prompt

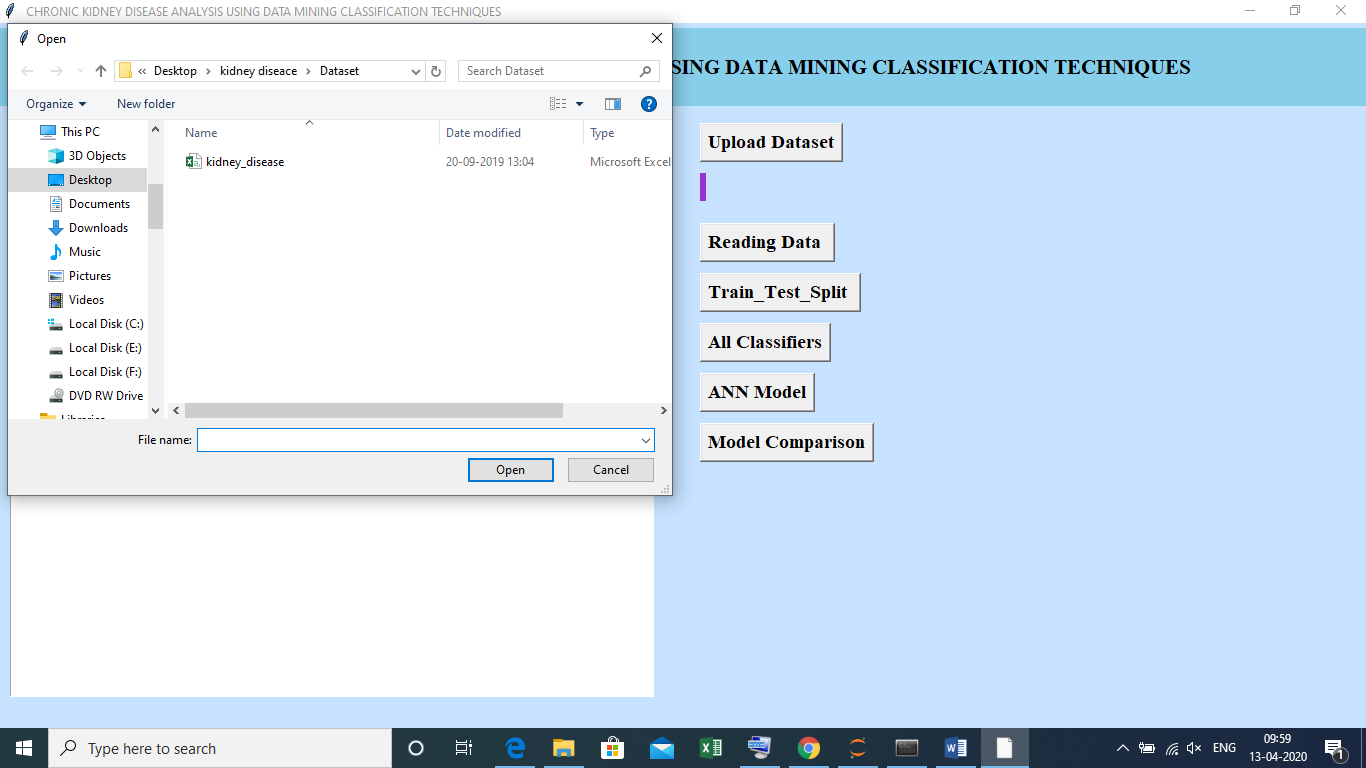
2. Goto project locationi

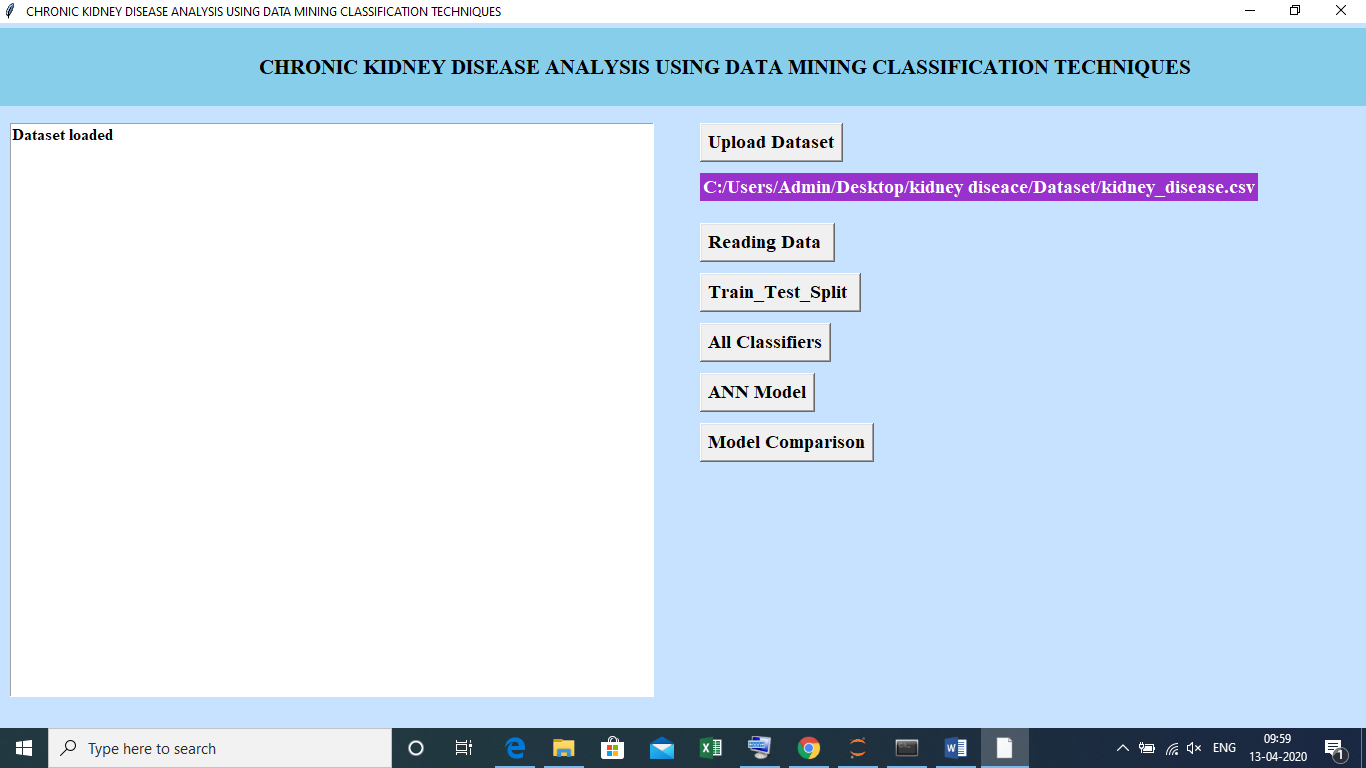
3. run “Python final.py”

When we run the code it displays below window

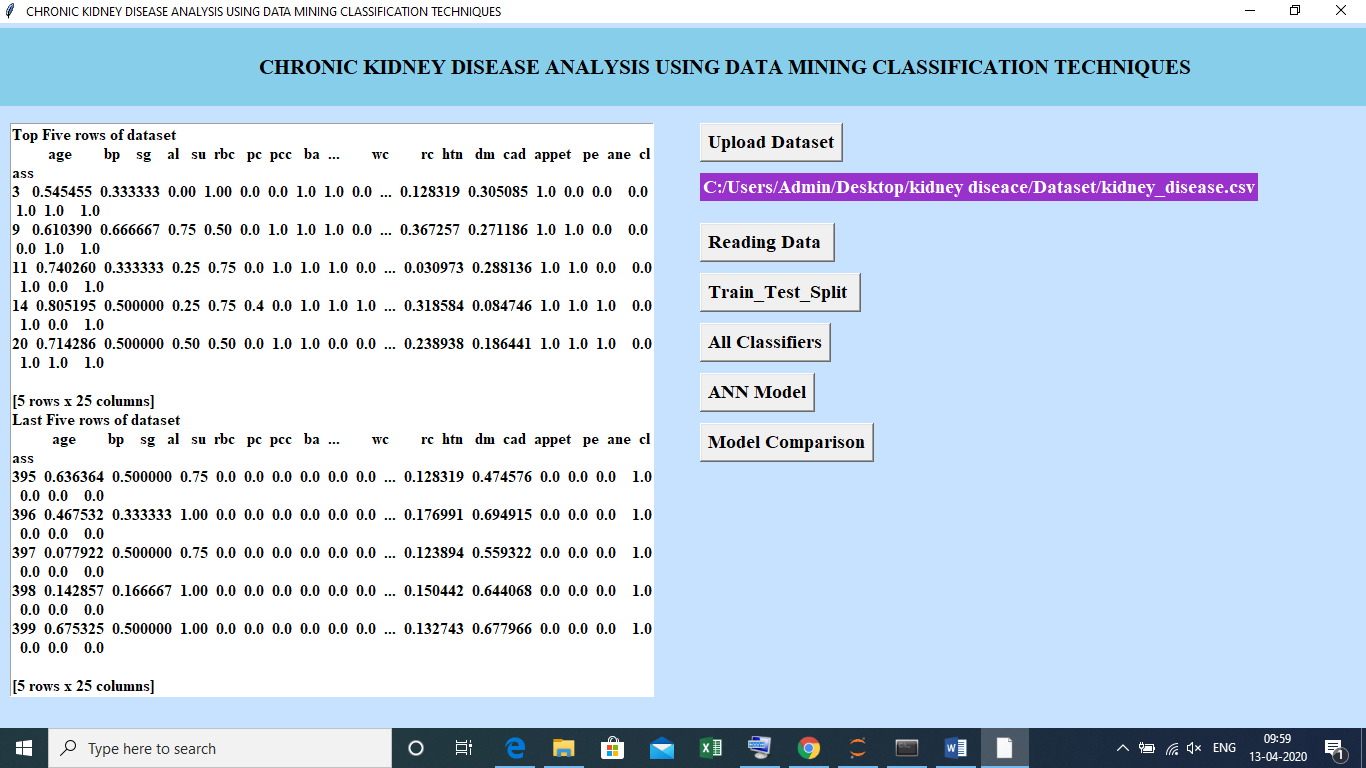


Now click on ‘upload dataset’ to upload the data

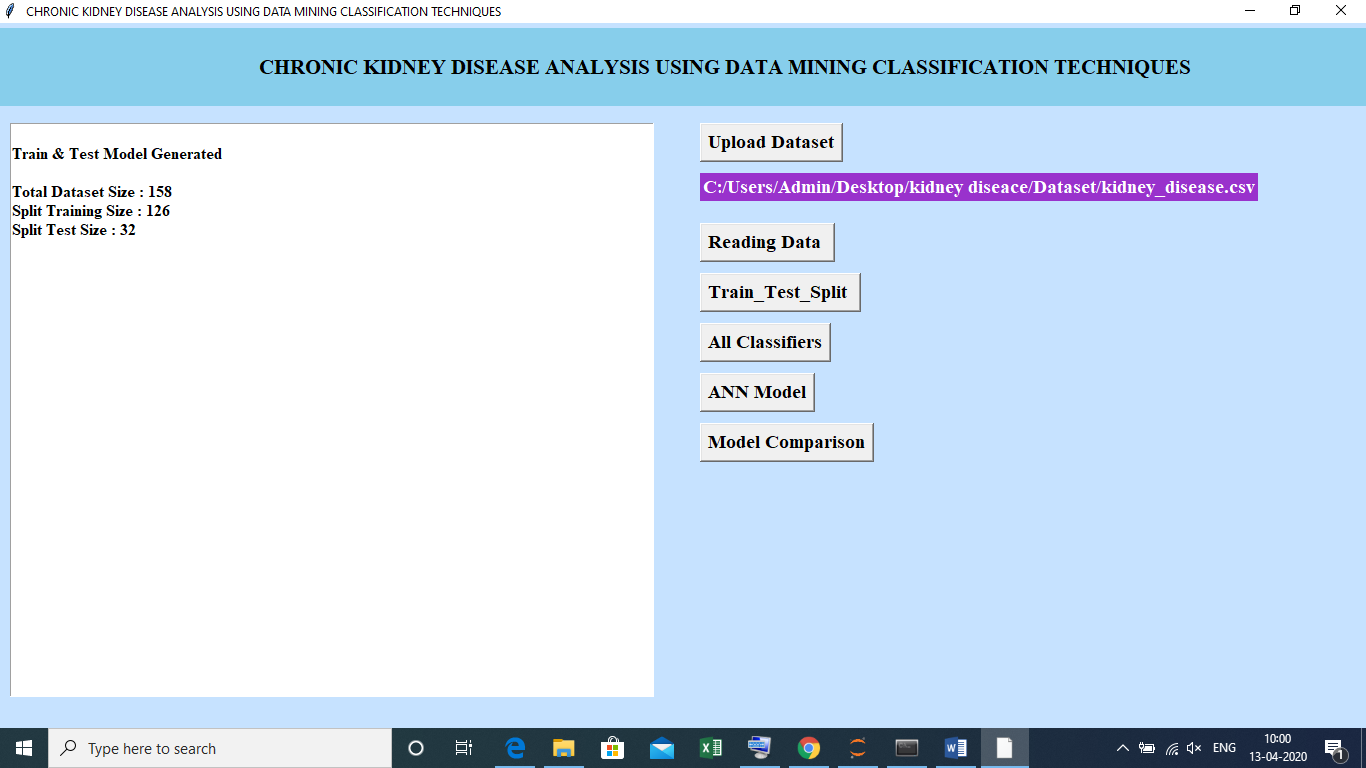




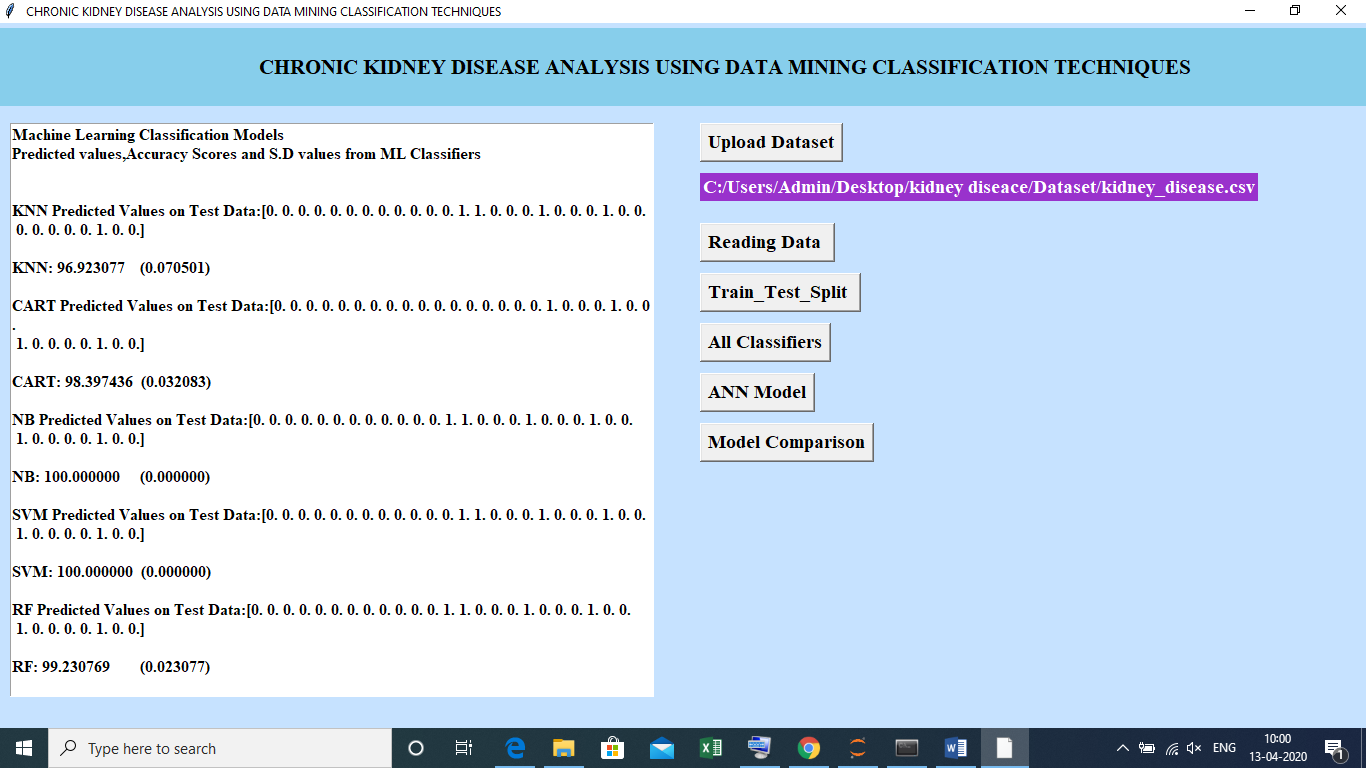
Now click on ‘read data’ it reads the data



Now click on ‘Train\_Test\_split’ to split the data into training and testing



Now click on ‘All classifiers’ to classify the models



KNN Predicted Values on Test Data is 96.92%

CART Predicted Values on Test Data is 98.39%

SVM Predicted Values on Test Data is 100%

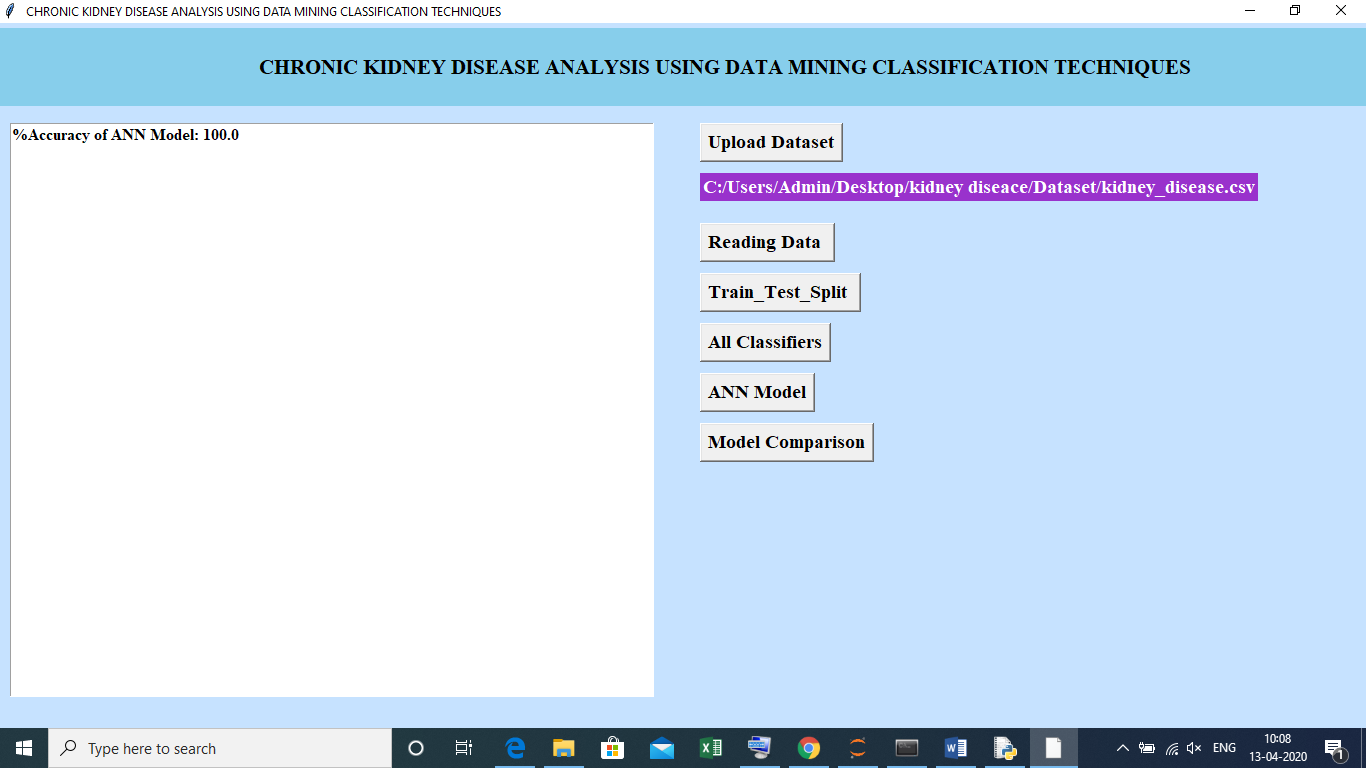
RF Predicted Values on Test Data is 100%

NB Predicted Values on Test Data is 97.41%

bagging Predicted Values on Test Data is 98.39%

MLP Predicted Values on Test Data is 100%

Now click on ‘ANN Model’



Now click on ‘Model comparison’ the comparison between the models

