Efficient Liver Disease Prediction using Classification Algorithms (December 2017)

Leeladhar Beeranahalli Kumaraswamy, Hitesh Bhatia, Salman Alsalman, Namratha Prabhu, Sanchit Arora
Computer Science Department
San Diego State University

***Abstract*— In this modern society where large number of humans are continuously increasing the consumption of alcohol, inhale of harmful gases, intake of contaminated food, pickles and drugs. This has increased the number of patients with Liver Diseases. The computers at the hospitals of the healthcare industries are used to collect huge amounts of information regarding the patients and their ailments. This huge repository of information contains wealth of knowledge. The hidden patterns and relationships in the data is mostly overlooked. Automatic classification tools may reduce burden on doctors. The algorithms considered here are Logistic Regression, K Nearest Neighbor (K-NN), Random Forest, Decision Tree, Naive Bayes, Neural Network, and Support vector machine (SVM). These algorithms are selected in order to find the highest accuracy of the Liver Disease based on the features provided. The data source we have used for experimental testing are commonly used and considered as a de facto standard for liver disease prediction reliability ranking.**

Index Terms— Liver Disease, Classification, Prediction, SVM, Decision Tree, Log Regression, Neural Network, etc.

I. INTRODUCTION

For many years, research in the area of liver disease prediction has been done in order to reduce towards the time taken by the specialists in the field. This will help doctors to detect if the patient is suffering from a liver disease and what is the severity of it. With the advent of machine learning techniques, public data availability, there has been an exponential increase in the creation of prediction algorithms and using them in order to train the machines for easier and earlier detection of the disease. Using powerful learning models geared for feature and symptom analysis, one can create predictive models that may be used to identify the severity of the disease at an early stage.

Task description

The present research work proposes the use of multiple classification algorithms that will be used with a common aim of classifying and predicting the disease severity at the earliest possible time. For this purpose, the following research objectives were formulated.

1. To develop a data cleaning algorithm that cleans the available dataset, by removing unwanted data, or denoising it.
2. To explore and enhance clustering algorithms to identify severity of disease based on the various features.
3. To explore and enhance classification algorithms to predict occurrence of disease at an early stage.

Challenges

As a continuation of the research done for Indian Liver Disease data, this project aims to higher the accuracy of the prediction of the disease. The data available is not that large which proved troublesome at times. If the dataset was too large, we used data mining methods. Our study has four major steps:

- Classify different features for classification.
- Identify the algorithms optimal for the problem.
- Optimize the algorithm to increase the accuracy of the system,
- Compare the outputs to determine highly efficient algorithm for this problem.

II. EXPERIMENT

Dataset Description: The dataset was collected from the UCI machine learning repository. It is called Indian Liver Patient Dataset (ILPD) [1]. This data set contains 416 liver patient records and 167 non-liver patient records. The data set was collected from north east of Andhra Pradesh, India. LiverResult is a class label used to divide into groups (liver patient or not). This data set contains 441 male patient records and 142 female patient records.

Indian Liver Patient Dataset (ILPD) :
https://archive.ics.uci.edu/ml/datasets/ILPD+(Indian+Liver+Patient+Dataset)

Any patient whose age exceeded 89 is listed as being of age "90". This data set has 10 variables that are Age, Gender, Total Bilirubin (TB), Direct Bilirubin (DB), Total Proteins (TP), Albumin (ALB), A/G Ratio, SGPT, SGOT and Alkphos.

Attribute	Datatype
Age	Real number
Gender	Categorical
TB	Real number
DB	Real number
Alkphos	Real number
Sgpt	Integer
Sgot	Integer
TP	Real number
ALB	Integer
A/G	Real number
LiverResult	Integer

Fig. 1. ILPD Attributes Datatype

The following figure shows a snapshot of the first five entries of our dataset before preprocessing has been done on it.

	Age	Gender	TB	DB	Alkphos	Sgpt	Sgot	TP	ALB	A/G	LiverResult
0	65	Female	0.7	0.1	187	16	18	6.8	3.3	0.90	1
1	62	Male	10.9	5.5	699	64	100	7.5	3.2	0.74	1
2	62	Male	7.3	4.1	490	60	68	7.0	3.3	0.89	1
3	58	Male	1.0	0.4	182	14	20	6.8	3.4	1.00	1
4	72	Male	3.9	2.0	195	27	59	7.3	2.4	0.40	1

Fig. 2. First 5 records of Dataset

1. Data Labeling

Preprocessing: DeNoise

The data set contains invalid data which will reduce the efficiency of the system, this invalid data is noisy data for the system. Since the dataset is quite small, we can manually remove these noisy data from the dataset. To make data clean:

- Add column names so whenever creating graphs, the axes are understandable.
- Remove the rows 209, 241, 253, 312 due to them having null value for column "ratio_albumin_and_globulin_ratio".

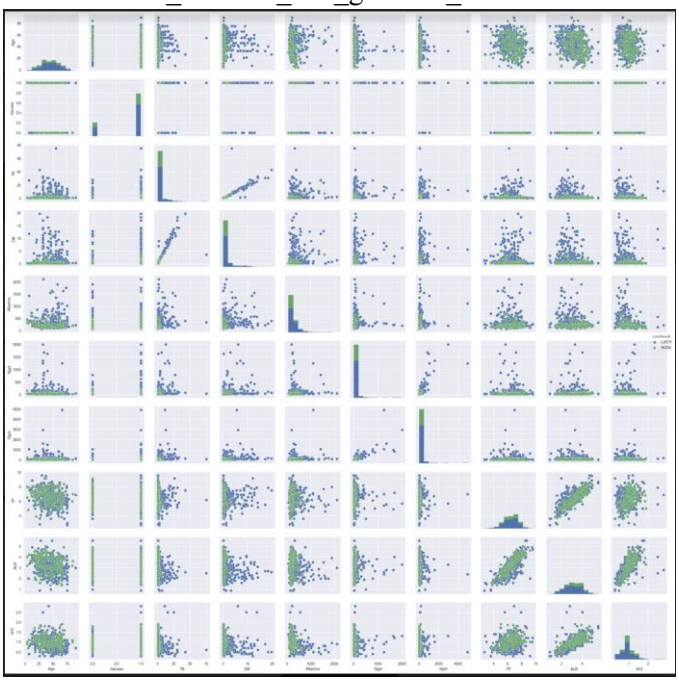

Fig. 3. Feature Correlation Graph

2. Machine Learning Models

2.1 Random Forest

Random forests also called as random decision forests are an ensemble learning method for classification, regression and other tasks. This algorithm operates as follows:

- Constructing a multitude of decision trees at training time
- Outputting the class that is the mode of the

classes (classification) or mean prediction (regression) of the individual trees.

- Random decision forests correct for decision trees' habit of overfitting to their training set.

At each split point (m), number of features are searched and applied as a parameter to the algorithm. Using cross-validation techniques, we can try multiple values.

For classification, a good default is: m = sqrt(p)
For regression, a good default is: m = p

Where m is the number of randomly selected features that can be searched at a split point and p is the number of input variables.

2.2 Logistic regression

Logistic regression, a statistical method used to analyze a dataset containing one or more independent variable to determine an outcome. It gives two possible outcomes, which is measured with dichotomous variable.

In logistic regression, coded as 1 (TRUE), or 0 (FALSE).

The goal is described as best fitting relationship model between the dependent variable and a set of independent variables.

$$logit(p) = b_0 + b_1X_1 + b_2X_2 + b_3X_3 + \ldots + b_kX_k$$

where p is the probability of presence of the characteristic of interest.

$$odds = \frac{p}{1-p} = \frac{probability\ of\ presence\ of\ characteristic}{probability\ of\ absence\ of\ characteristic}$$

where odds is the logit transformations. Can also be written as :

$$logit(p) = \ln\left(\frac{p}{1-p}\right)$$

2.3 Naive Bayes

The naive Bayes algorithms implemented by BernoulliNB for data that is distributed according to multiple features but assuming each one to be a binary-valued (Bernoulli, boolean) variable. Therefore, binary-valued feature vectors samples are required by this class. The algorithm inarized the any other kind of data for it's input.

Bernoulli naive Bayes's Decision rule is based on:

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i)$$

which differs from multinomial NB's rule in that it explicitly penalizes the non-occurrence of a feature i that is an indicator for class y, where the multinomial variant would simply ignore a non-occurring feature.

In the case of text classification, word occurrence vectors (rather than word count vectors) may be used to train and use this classifier. BernoulliNB might perform better on some datasets, especially those with shorter documents. It is advisable to evaluate both models, if time permits.[7]

2.4 Neural Network

Artificial neural networks are a learning algorithm, usually called "neural network" (NN), that is inspired by the structure and functional aspects 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 modeling tools where the complex relationships between inputs and outputs are modeled or patterns are found.
An ANN has several advantages but one of the most recognized of these is the fact that it can 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 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. An artificial neural network is consisting of many neurons and hidden layers. The following figure illustrate the concept of it.

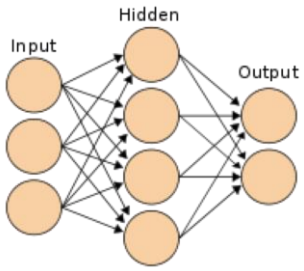

Fig. 4. NN Neurons and Layers Illustration [5]

Choosing how deep is the network is critical for the performance of the model. In our model, we have 10 features. So, we decided to make our NN model to be with 3 neurons of 10 deep for each.

2.5 Decision Tree Learning

Decision tree learning uses a decision tree (as a predictive model) to go from observations about an item to conclusions about the item's projected value [4].

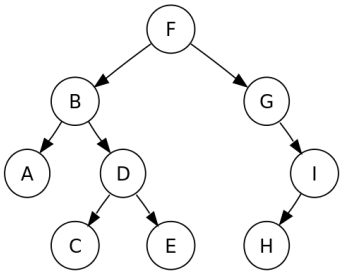

Fig. 5. Decision Tree Iteration

There are many parameters that can be tuned to optimize Decision Tree algorithm's performance. First and foremost is the number "min_samples_split" which is the constant number that is required for each code split internally. The next tunable parameter is "random_state", which is the random number generator. Because the dataset and the features are small, I guess the range between 90 and 99 should be enough. In case the optimal one turns out to be 99 after "searching", I will increase the range to [90, 105], and keep trying.

Given the training dataset, it takes 0.003 second/fit (a fit = one specific set of parameters, e.g.: (min_samples_split=20, random_state=99) runs against one "fold" of the training data. Using these parameters, make a final Decision Tree Classifier, fit the (training) data, use the model to predict the testing data. We retrieve the "Confusion Matrix":

For 60-40 data set:
[[136 31],
[45 22]]

For 70-30 data set"
[[99, 23],
[36, 17]]

It seems when one truly has the disease, they are mostly diagnosed as "having". But when one doesn't, 2/3 of them are still diagnosed as "having". For the best result, the percentage is about 0.67.

2.6 SVM

Support Vector Machine; is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.

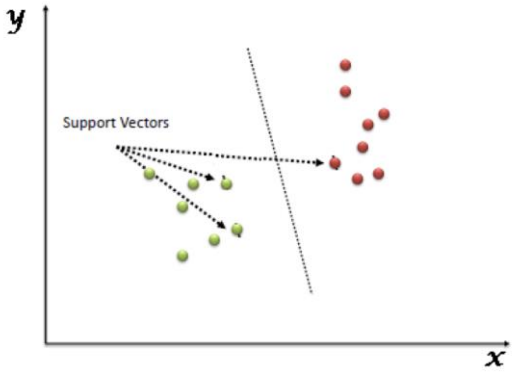

Fig. 6. A simple illustration of the SVM (hyper-plane/ line) [2]

Therefore, Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).
SVM uses kernels or internal functions that shape the procedure of creating the hyper-plane and segregates the data. In some references, they may call them models.

These models are closely related to classical multilayer perceptron neural networks. By using a kernel function, these are an alternative training method for polynomial, radial basis function and multi-layer perceptron classifiers in which the weights of the network are found by solving a quadratic programming problem with linear constraints, rather than by solving a non-convex, unconstrained minimization problem as in standard neural network training. [3]
In the SVM literature, a predictor variable which is called an attribute and a transformed attribute that is used to define the hyper plane is called a feature [4]. In practice and in our model on this project, we use an RBF-SVM kernel. The reason for that because looking to the features correlation figure, we can see that the features are not linearly correlated to each other. It means the separable line will not be linear. Therefore, choosing the RBF kernel is more useful for non-linear hyper-plane. In this situation, our kernel is called Radial Kernel SVM that follows the following formula:

Where gamma is a parameter that must be specified to the learning algorithm. A good default value for gamma is 0.1, where gamma is often 0 < gamma < 1. The radial kernel is very local and can create complex regions within the feature space, like closed polygons in two-dimensional space.

2.7 KNeighborsClassifier

k-NN classification, works on the class membership, by assigning it to the output. A feature is classified by a maximum support from its neighbors. This allows the selected feature to be assigned to the class, with most common among its k nearest neighbors (k is a positive integer, typically small). The object is simply assigned to the class of the single nearest neighbor, when k = 1 and hence it is not recommended, as there would be classification among the data set entered.

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. Though there is no explicit training step in this particular classification algorithm, the above mentioned criterion can be considered as the training set. We have chosen the number of neighbors to use as 3.

Input distance measure is used between the selected feature and the neighbors, to determine which of the K instances in the training dataset are most similar. Euclidean Distance is one of the most popular measures that is considered for real-valued input variables. It is calculated by square rooting the sum of the squared differences between a new point (x) and an existing point (xi) across all input attributes j.

Euclidean Distance: (x, xi) = sqrt(sum((xj – xij)^2))

We have used fit(X, Y). Where it will fit the model using X as training data and Y as target values.

III. PERFORMANCE EVALUATION

In this section, we will evaluate the results of different classification models in terms of accuracy and execution time to see how well that they predict a patient may have a liver disease or not. We have applied seven different classification models such as Logistic Regression, K Nearest Neighbor, Random Forest, Decision Tree, Naive Bayes, Neural Network, and Support vector machine. We have also plotted their confusion matrix and tabulated the accuracy and execution time for each model. Also, we have done the testing on two different splitting methods. One for 70% as training dataset and 30% as testing dataset, and the other test for 60% as training dataset and 40% as testing dataset.

1. Environment Setup

Python as a programming language and Jupyter python Notebook have been used to implement the preprocessing of data and to apply different classification models. We have used few python libraries for achieving different tasks as part of our implementation. Pandas library have been used for loading and handling the data from input csv data file. Sickit Learner library have been used for preprocessing modules and implementing classification model. Numpy and Matpyplot have been used for plotting the heat map and confusion matrix for each learning models.

Indian Liver Patient, (ILPD)[1], input dataset is used as input file for all our classification model. It contains 11 features. and total number of records 583. We have applied 7 different classification models which uses many features from the input file for prediction model as needed for each model.

Parameter	Values
Input data file	Indian Liver Patient Dataset (ILPD).csv
Number of features	11
Number of Records	583
Classification Models	Logistic Regression K Nearest Neighbor Random Forest Decision Tree Naive Bayes Neural Network Support vector machine

Fig. 5. Parameters of the System

2. Classification Models with Accuracy

Each model takes some unique parameters along the with the input data. The input data is preprocessed and contains only the relevance features for the learning models. The following table and figures show the model with the resulting accuracy:

Classification Model	Accuracy (Percent) (70~30) % Train~Test Dataset	Accuracy (Percent) (60~40) % Train~Test Dataset
Log Regression	69.71%	69.23%
Random Forest	68.00%	71.36%
SVM	70.00%	71.79%
Naive bays	69.71%	71.37%
KNeighborsClassifier	68.00%	67.95%
Decision Tree	66.29%	67.52%
Neural Network	58.29%	70.51%

Fig. 7. Accuracy Chart

Fig. 8. Accuracy Graph

As results too, we have produced the following figure that shows the confusion matrix of each classification model:

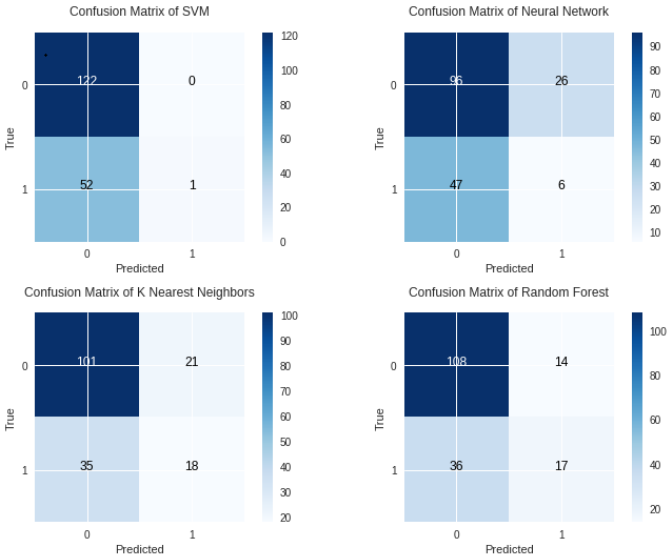

Fig. 9a. Confusion Matrix

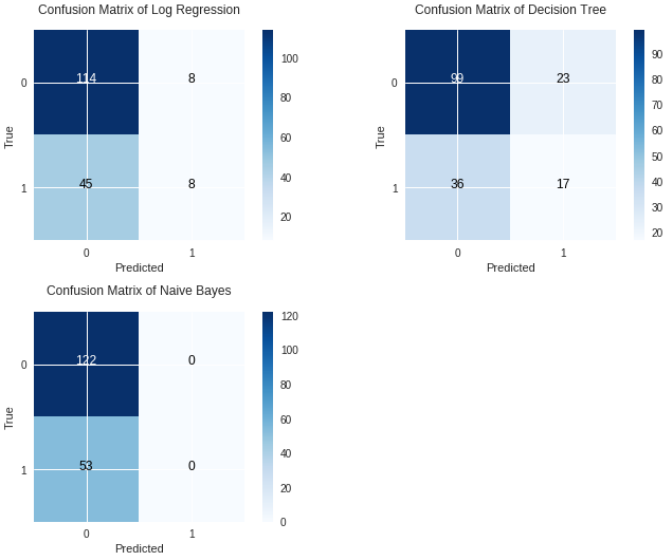

Fig. 9b. Confusion Matrix

Based on True Positive and False positive data for the algorithms we have tested on the dataset. We got definite data which showed SVM and Naive Bayes performed better w.r.t true positive rate. This accounts for the higher accuracy when contrasted with other algorithms.

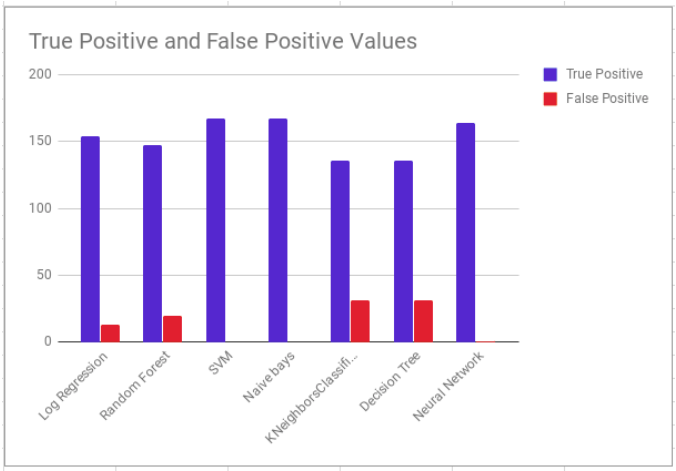

Fig 10. TP and FP values for 60% Training Data

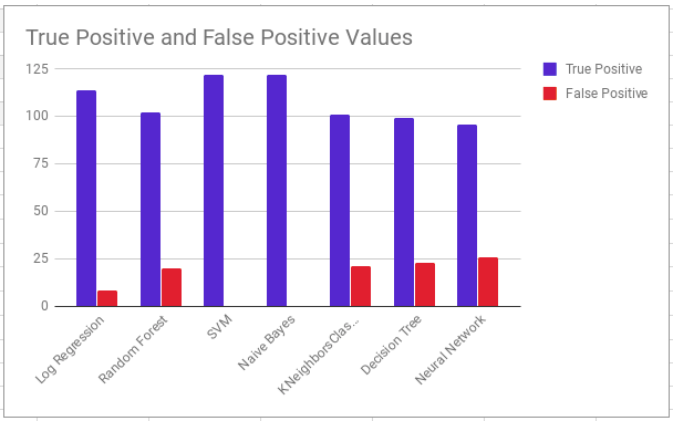

Fig 11. TP and FP values for 70% Training Data

3. Classification Models with Execution Time

Each model takes some time to process the input data. The process here means the learning of data, followed by providing the output to test data. The input data is preprocessed and contains only the relevance features for the learning models. The following table and figures show the model with the resulting time taken to process the test data and provide output:

Algorithm	Running Time (sec) (70 - 30)% Train - Test Dataset	Running Time (sec) (70 - 30)% Train - Test Dataset
Log Regression	0.03145957	0.035113096
Random Forest	0.258708477	0.263366461
SVM	0.021265745	0.024335146
Naive bays	0.008899212	0.005567312
KNeighborsClassifier	0.01088357	0.004552364
Decision Tree	0.003212929	0.00396204
Neural Network	0.126742601	0.417187214

Fig. 12. Running Time Chart

Fig. 13 . Running Time Graph

IV. FURTHER STUDY

Conclusion

This research work focus on predicting liver disease among the Indian population. Due to diversity of population, we tested the dataset on seven algorithms. Based on the observation, we able to achieve an accuracy about 0.71. Among all the seven algorithms, SVM and Naive Bayes provided more accurate true positive results and Random Forest provided more accurate true negative results.

Furthermore, we want to study other model/methods. We hope that our study would provide access to an low cost liver disease prediction system and eventually, reduce cost of healthcare.

Acknowledgement

This paper describes research done at SDSU in the department of Computer Science. We are thankful to Dr. Xiaobai Liu for his guidance and cooperation.

References:

[1]A Critical Study of Selected Classification Algorithms for Liver Disease Diagnosis : http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.205.9572&rep=rep1&type=pdf

[2]A Critical Comparative Study of Liver Patients from USA and INDIA: An Exploratory Analysis : http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.401.9235&rep=rep1&type=pdf

[3]https://towardsdatascience.com/decision-trees-in-

machine-learning-641b9c4e8052
[4]http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
[5] Michael J. Sorich, John O. Miners, Ross A. McKinnon, David A. Winkler, Frank R. Burden,| and Paul A. Smith. Comparison of linear and nonlinear classification algorithms for the prediction of drug and chemical metabolism by human UDP-Glucuronosyltransferase Isoforms.
[6] Tali Soroker - Understanding Stock Market Prediction Using Artificial Neural Networks and Their Adaptations https://iknowfirst.com/understanding-stock-market-prediction-using-artificial-neural-networks-and-their-adaptation
[7]http://scikit-learn.org/stable/modules/naive_bayes.html