**INSTITUTE OF HEALTH INFORMATICS**

Graduate Programme in Health Data Science

# Assessed Coursework Submission

| | |
|---|---|
| **Student candidate number:** | VRHJ5 |
| **Module:** | CHME0016: Machine Learning in Healthcare and Biomedicine |
| **Date due:** | 04 May 2022, 17:00 BST |
| **Word count:** (excluding references, diagrams and appendices) | 2498 |
| **Disability or other medical condition** for which UCL has granted special examination arrangements: | N/A |
| **Formative feedback:** | Please address in formative feedback: |
| | |
| | Please ignore in formative feedback: |
| | Apologise for not adding much comments to code. |

# Introduction

**1.1 Background:** Liver failure is one of most common deaths in India. Every one person out of 5 is affected by liver disease. According to WHO 2018 data, 3% of total deaths in India is due to liver disease. 2 million people die worldwide from liver disease every year. According to NHS, most types of liver diseases do not cause any symptoms in early stages. The liver is already damaged once the patients start having symptoms (aka cirrhosis). Cirrhosis is uncurable but can be treated and mainly caused due to obesity, alcohol misuse and hepatitis.

**1.2 Problem:** It is hard to detect liver disease as it does not cause any symptoms in early stages. However, the diagnosis of liver disease at every stage can help in preventing the disease using proper measure. The aim is to predict the liver disease in patients with patient's liver related extracted information/measurement using machine learning.

**1.3 Dataset:** The dataset selected for this coursework is Indian Liver Patient Dataset (ILPD). This data set contains 10 variables that are age, gender, total bilirubin, direct bilirubin, total proteins, albumin, A/G ratio, SGPT, SGOT and Alkphos.

- This dataset contains 416 liver patient records and 167 non liver patient records. It is highly unbalanced with 71.4% liver patients (Class = 1) and 28.6% patients with no liver disease (Class = 0).
- There are 76% males and 24% females in the dataset.
- A/G variable contains 4 missing values.
- Except for Gender and Patient class, all features are numerical.
- The detailed statistics of dataset is provided in the table 1.

*Table1: Dataset description and descriptive statistics*

| Feature | Description | Type | Total | Statistics | | |
|---|---|---|---|---|---|---|
| | | | | Mean | Standard D | Range (min-max) |
| Age | Patient's age | Numerical | 583 | 44.75 | 16.19 | 4 - 90 |
| Gender | Patient's sex | Nominal | 583 | - | - | 0 or 1 |
| TB | Total Bilirubin | Numerical | 583 | 3.30 | 6.21 | 0.4 - 75 |
| DB | Direct Bilirubin | Numerical | 583 | 1.49 | 2.81 | 0.1 - 19.7 |
| Alkphos | Alkaline Phosphatase | Numerical | 583 | 290.58 | 242.94 | 63 - 2110 |
| Sgpt | Alamine Aminotransferase | Numerical | 583 | 80.71 | 182.62 | 10 - 2000 |
| Sgot | Aspartate Aminotransferase | Numerical | 583 | 110.01 | 288.92 | 2.7 - 9.6 |

| | | | | | | |
|---|---|---|---|---|---|---|
| TP | Total Proteins | Numerical | 583 | 6.48 | 1.08 | 0.9 - 5.5 |
| ALB | Albumin | Numerical | 583 | 3.14 | 0.79 | 0.3 - 2.8 |
| A/G | Albumin and Globulin Ratio | Numerical | 579 | 0.95 | 0.32 | 0.3 - 2.8 |
| Class | Liver patient or not | Numerical | 583 | - | - | 0 or 1 |

**Both Gender and Class are coded as 0 and 1 in the code

### 1.4 Features characteristics:

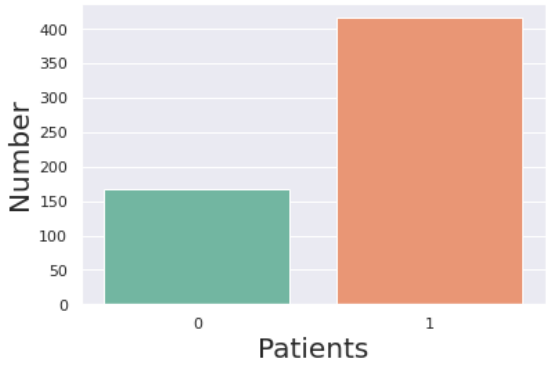


**Figure1**

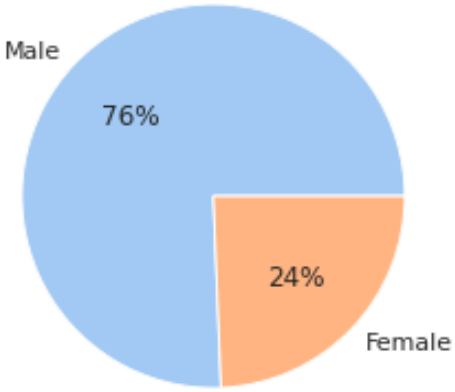


**Figure2**

The number of individuals with liver disease is higher than individuals with no liver problem (Figure 1). It is noticed that number of males in data are higher than no. of females (Figure2). Figure 3 also shows that number of males with liver disease is high that can be hypothesised that males have high chance of liver disease than female and the chance is maximum for male of age around 35 years. Figure 4 shows that the high amount of TB (Total bilirubin) indicates the liver infection. Similarly, TP, Sgpt and Sgot also show strong effectiveness for males with liver disease figure 4,5.


**Figure3**


**Figure4**


**Figure5**

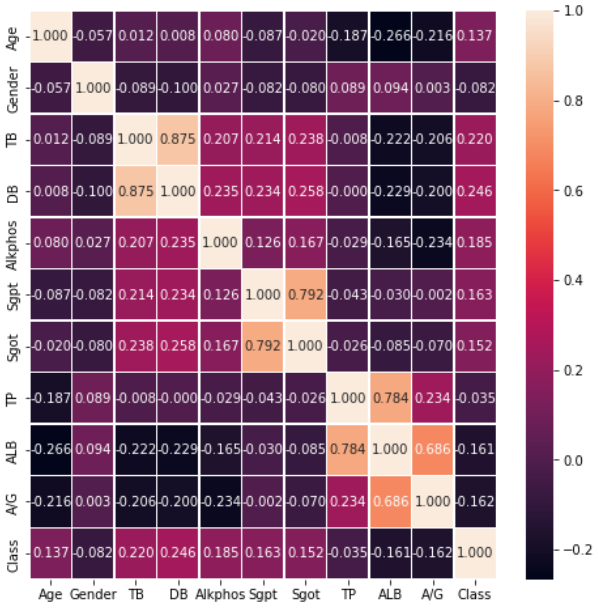


**Figure 6**

Based on Correlation matrix figure6, strong association between the features is observed which can be seen in Table 2. TB and DB can be represented with a straight line and show the highest correlation with 87.46%.

**Table2: Correlation between features**

| **Feature I** | **Feature II** | **Correlation %** |
|---|---|---|
| TB | DB | 87.46 |
| Sgot | Sgpt | 79.20 |
| ALB | TP | 78.40 |
| ALB | A/G | 68.63 |
| AGE | ALB | 26.59 |

As the Class variable is binary, the problem is binary classification. The decision boundary can be defined using Sigmoid/Softmax function where output > 0.5 is 1 else zero (0 = Not liver patient, 1 = Liver Patient). Figure 7

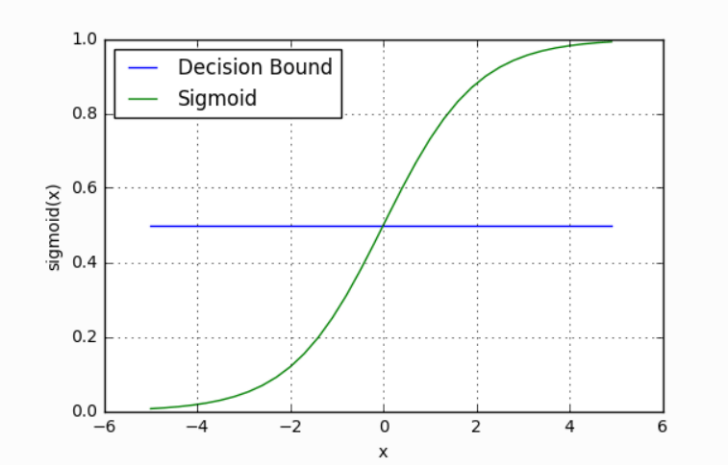


Figure7

# Methodology:

**2.1Pre-processing:** Various pre-processing techniques are performed:

1. **Data Cleaning:**

- Irregular cardinality-The nominal variable Gender is coded is 0 for male and 1 for female.
- Missing data - A/G feature with 4 missing values is filled with the mean ratio of all the other individuals.
- Outliers-Some outliers are observed in Alkphos, sgot and sgpt with 2 exceptionally different outliers from sgot feature are attempted to remove and set to lower maximum value (sgot = 2500).
- Class is relabelled as 0 and 1 instead for 1 and 2 for easy interpretation where 0 is non liver disease patients and 1 is patients with liver disease.

2. **Feature Selection:**

Using Correlation: The features are assumed to be independent and since they are highly correlated, they can be removed as both variables would conclude to impact the same information. Association of TB and DB can be represented with straight line (Figure8). Therefore, we can consider removing one of them (DB). Similarly, Sgot & Sgpt and ALB & TP also show a strong positive association.

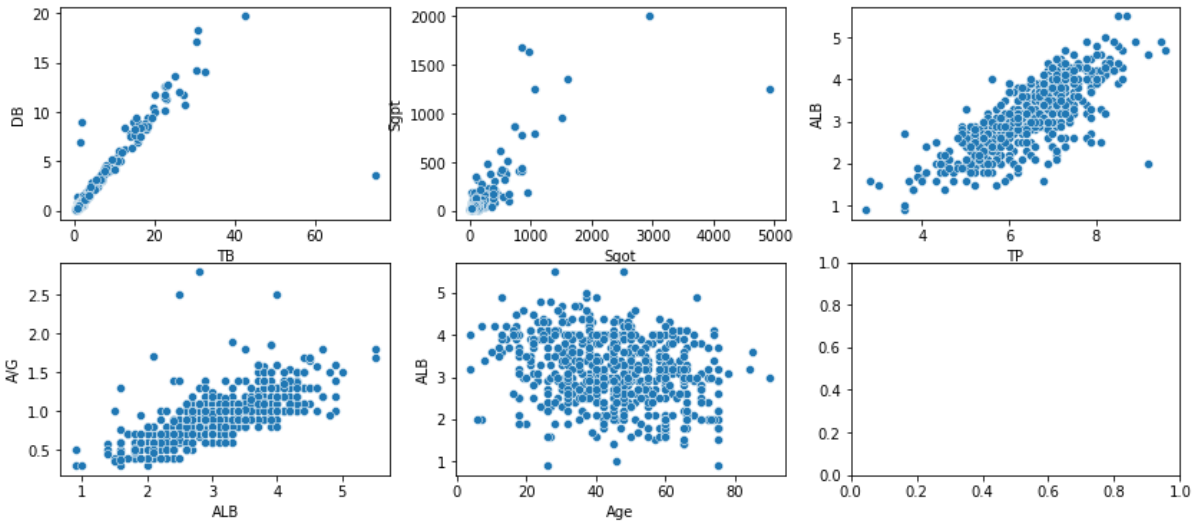


Figure8

SBS and Randomforest are used to select and filter more relevant features. Figure9 suggest taking 8 features (remove sgot, ALB) for KNN whereas figure10 considers Gender as irrelevant to Decision tree.

In conclusion, remove 6 features out the 10.

Final Features = `['Age', 'TB', 'Alkphos', 'Sgpt', 'TP', 'A/G']`

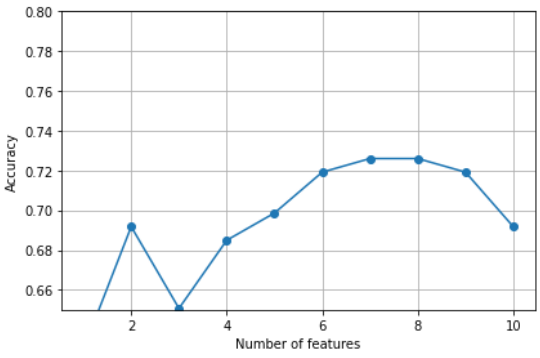


Figure9

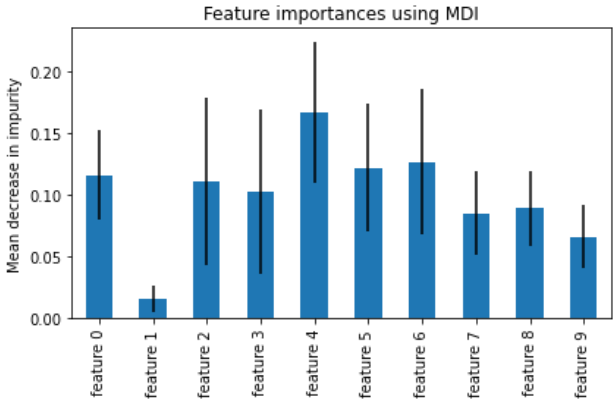


Figure10

3. **Feature Extraction:** Gender is spitted into two separate features Male and Female before feature selection to check if it improves model. A decrease in accuracy was noted.
   In figure 11, two outliers from Sgot were reduced to the value of 2500 but no removed due to small number of instances.

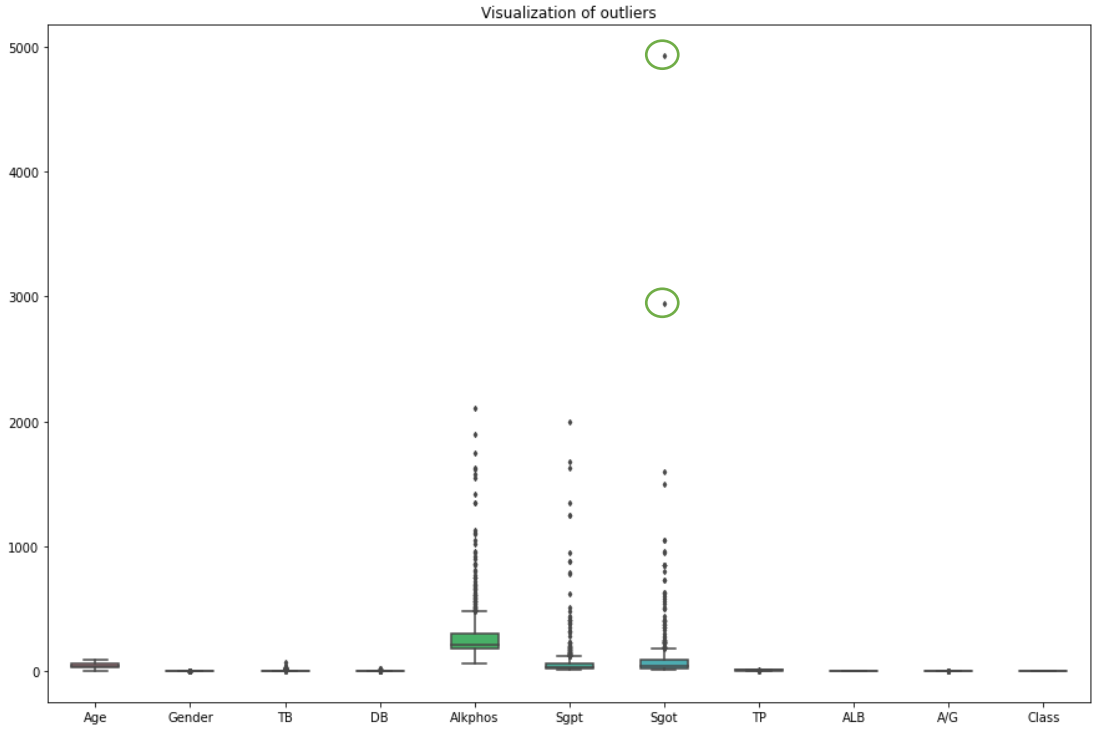


Figure11

### 2.2 Feature Pre-processing:

- Normalization is performed to transform the data in same scale. However, for decision tree it did not make a difference as it works on if-else decision logic.
- Balancing data: As the data was highly imbalanced, more samples were for non-liver patient were added to the dataset. Total samples were increase to 832 from 583.
- Train-test-Split: The data was spitted into 70% training and 30% testing.
- Data Leakage- No data leakage is observed. Besides, we did Min-max scaling of data for train dataset not for the overall data as it can cause data leakage by considering mean of test dataset.

### 2.3 Supervised models:

Two supervised models selected for the following task are: One model clearly explain the reasons of predictions (White-box model) whereas the other model is black-box with no information of prediction.

## 1. Decision Tree:

Decision tree is a graph that uses branching technique to provide every possible outcome of a decision. The root node represents input data (X) and leaf nodes contain output variable (y) that predict the final outcome. As there are no clear Patient/non patient criteria in our data, the nodes are considered as impure. To determine the best separation by measuring this impurity, Gini Impurity or Entropy are used. In classification, Gini cost function evaluates the splits in the dataset by calculating the sum of squared probabilities for each class from 1.

$$Gini = 1 - \sum_{i=1}^{C}(p_i)^2$$

The spitting is done using cost function and best split points are decided based on the minimum cost using greedy approach. Higher Gini value means higher heterogeneity.

```
#------- Some "Helper" functions ---------------
# Segregating out instances that take a particular value
# attributearray is an N x 1 array.
def segregate(attributearray, value):
    outlist = []
    for i = 1 to length(attributearray):
        if (attributearray[i] == value):
            outlist = [outlist, i]  #  Append "i" to outlist
    return outlist


# Assuming labels take values 1..M.
def computeEntropy(labels):
    entropy = 0
    for i = 1 to M:
        probability_i = length(segregate(labels, i)) / length(labels)
        entropy -= probability_i * log(probability_i)
    return entropy

# Find most frequent value. Assuming labels take values 1..M
def mostFrequentlyOccurringValue(labels):
    bestCount = -inf
    bestId = none
    for i = 1 to M:
        count_i = length(segregate(label,i))
        if (count_i > bestCount):
            bestCount = count_i
            bestId = i
    return bestId

#-------- The Dtree code ------------

#Here "attributes" is an Num-instance x Num-attributes matrix. Each row is
#one training instance.
#"labels" is a Num-instance x 1 array of class labels for the training
instances

# Note, we're storing a number of seemingly unnecessary variables, but
# we'll use them later for counting and pruning
class  dtree:
    float @nodeGainRatio
    float @nodeInformationGain
    boolean @isLeaf
    integer @majorityClass
    integer @bestAttribute
    dtree[] @children
    dtree   @parent

    init(attributes, labels):
        @parent = null
        buildTree (attributes, labels, self)

    buildTree (attributes, labels, self):
        numInstances = length(labels)
        nodeInformation = numInstances * computeEntropy(labels)
        @majorityClass = mostFrequentlyOccurringValue(labels)

        if (nodeinformation == 0):  # This is a "pure" node
```

```
            @isLeaf = True
            return

        # First find the best attribute for this node
        bestAttribute = none
        bestInformationGain = -inf
        bestGainRatio = -inf
        for each attribute X:
            conditionalInfo = 0
            attributeEntropy = 0
            for each attributevalue Y:
                ids = segregate(attributes[][X], Y) # get ids of all
instances
                                                    # for which attribute X
== Y

                attributeCount[Y] = length(ids)
                conditionalInfo += attributeCount[Y] *
computeEntropy(labels(ids));
            attributeInformationGain =  nodeInformation - conditionalInfo
            gainRatio = attributeInformationGain /
computeEntropy(attributeCount)
            if (gainRatio > bestGainRatio):
                bestInformationGain = attributeInformationGain
                bestGainRatio = gainRatio
                bestAttribute = X

        #If no attribute provides andy gain, this node cannot be split
further
        if (bestGainRatio == 0):
            @isLeaf = True
            return

        # Otherwise split by the best attribute
        @bestAttribute = bestAttribute
        @nodeGainRatio = bestGainRatio
        @nodeInformationGain = bestInformationGain
        for each attributevalue Y:
            ids = segregate(attributes[][bestAttribute], Y)
            @children[Y] = dtree(attributes[ids], labels[ids])
            @children[Y].@parent = self
        return
```

**Supervised Learning:** The model finds the best decision based if-else tree that splits the training data. In 582 training instances figure12, X[1] i.e., Total bilirubin is greater than 0.015 in 177 samples and gini = 0.5 indicating adequate inequality. Similarly, the tree is constructed with same logic.

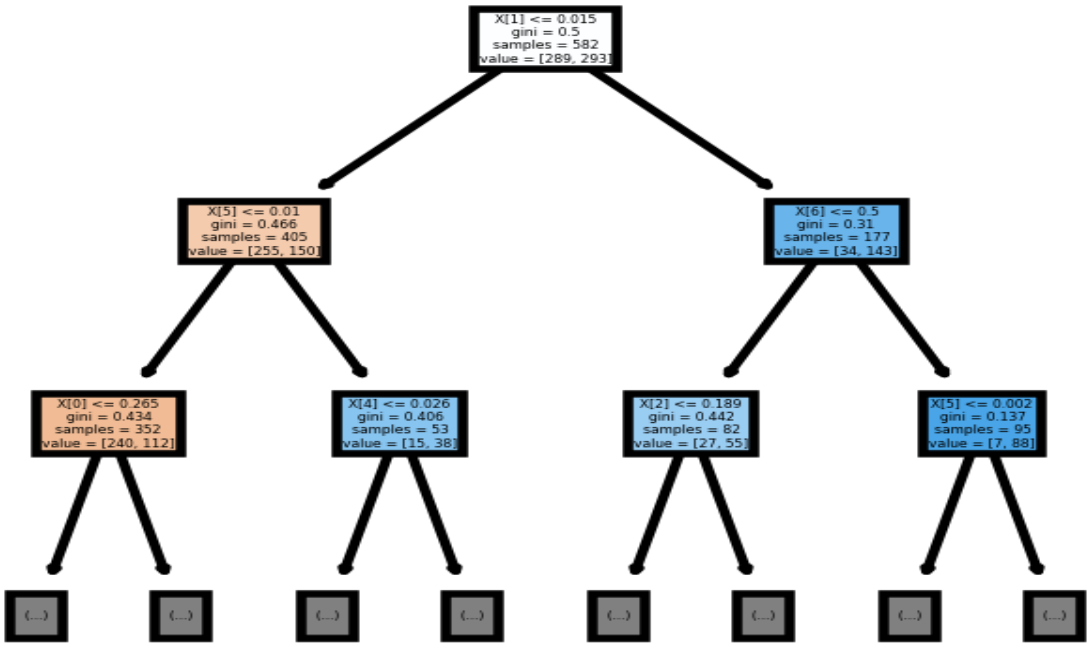


Figure12

**Optimization:** Decision tree adapts to training data very closely and can lead to overfitting. Therefore, regularization hyperparameters is prefers to restrict the independence of DT. Parameters like max_depth, (=None by default), max_leaf_nodes, min_sample_split, min_weight_fraction_leaf are controlled to reduce overfitting risk. Pruning can also be performed to delete unnecessary nodes.

**Model Evaluation:** For classification problems, confusion matrix is best to evaluate the performance measure of our classifier. To have more concise matrix, we will have a look into precision and recall that can be visualized using AUC/ROC plot.

## 2. Artificial Neural Networks

NN is an oriented/directed graph organized in layers. It is composed of one input layer, one or more hidden layers and a final output layer. Layers are made up with interconnected nodes and contain activation function to learn complex patterns. Each layer contains a bias neuron and is fully connected to next layer.

**Supervised Learning:** The model feeds each training instance to the network and makes the prediction of every neuron in each consecutive layer(Forward propagation). It measures the output error and then goes backwards through each layer to evaluate the error contribution from each connection(Back propagation). The weights are changed accordingly using error gradients calculated in back propagation to reduce the error(Gradient descent). The loss function use by our model is binary cross-entropy.

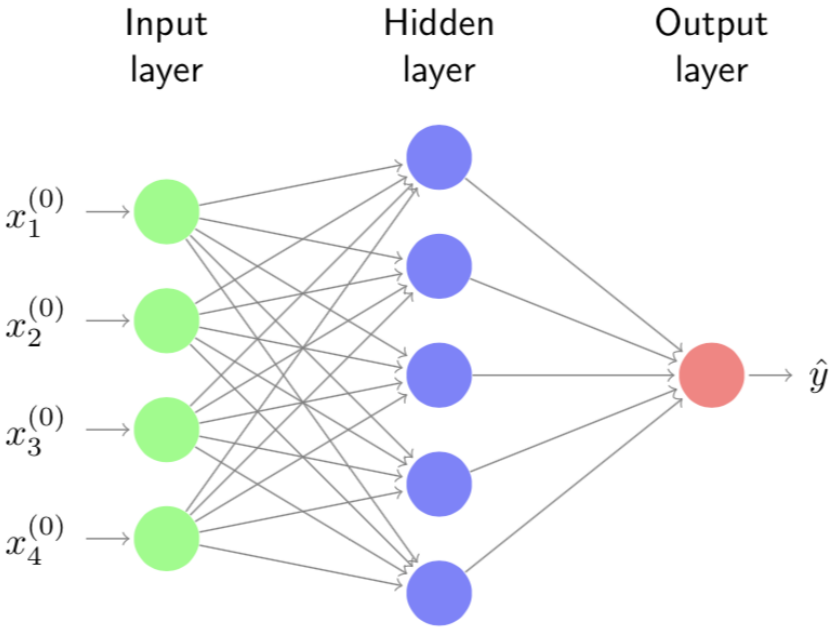


**Figure13**

**Optimization:** Neural networks being a complex model have many hyperparameters for optimization. The number of layers, type of activation function in each layer, initialized weights are some of parameters we can fine-tune. Tanh and ReLU were be used seperately for 2 layers neural network and Adam as an optimizer to see that performance.

**Model Evaluation:** The output for classification corresponds to a binary class (Liver patient or not). The output layers used Softmax function(same as sigmoid but used for multiclass and will work as binary for our dataset.) to estimate probability of the class. [Evaluation similar to decision tree evaluation]

```
function BACK-PROP-LEARNING(examples, network) returns a neural network
  inputs examples, a set of examples, each with input vector x and output
vector y
       network, a multilayer network with L layers, weights wi,j, activation
function g
  local variables: Δ, a vector of errors, indexed by network node

  repeat
     for each weight wi,j in network do
        wi,j ← a small random number
     for each example (x, y) in examples do
        /* Propagate the inputs forward to compute the outputs */
        for each node i in the input layer do
           ai ← xi
        for l = 2 to L do
           for each node j in layer l do
              inj ← Σi wi,j ai
              aj ← g(inj)
        /* Propagate deltas backward from output layer to input layer */
        for each node j in the output layer do
           Δ[j] ← g'(inj) × (yi − aj)
        for l = L − 1 to 1 do
           for each node i in layer l do
              Δ[i] ← g'(ini) Σj wi,j Δ[j]
        /* Update every weight in network using deltas */
        for each weight wi,j in network do
```

```
            wi,j ← wi,j + α × ai × Δ[j]
    until some stopping criterion is satisfied
    return network
```

# Results:

The table3,4 below show the model performance when data is used without any pre-processing (Raw), after normalization, then balancing the instances, picking selected features and modifying outliers. Finally, the last column is for Hyperparameter optimization.

**Decision Tree:**

- The impact of Data Pre-processing on the model can be seen in Table3. It is seen that after scaling no change in accuracy is noticed. It can be justified by the fact that decision tree works on if-else and scaling has no effect on that. After balancing the dataset, accuracy of the models had been improved by large percentage. Moreover, feature selection seems to decrease model efficiency. An improvement is observed after outlier engineering.

  **Table3**

| Accuracy | Raw | Scaling | Balancing | Outlier | Selection | Optimized |
|---|---|---|---|---|---|---|
| Training | 100% | 100% | 100% | 100% | 100% | 89.18% |
| Testing | 64.00% | 64.00% | 78.40% | 77.60% | 81.20% | 82.80% |

- The hyperparameters chosen for optimization are: Criterion(Gini or entropy), Spiller(random or best), Max_depth, Mini_samples_split and Mini_sample_leaf.
  GridSearch with 5-fold cross-validation was selected and optimized for accuracy.
  The results after hyperparameter tuning were optimized with the goal of reducing overfitting as 100% accuracy was shown for training data. As shown, the results for training reduced from 100% to 90.89 and testing increased to 82.08%.
  **Best parameters:**

```
{'criterion':'entropy',

      'max_depth':15,
      'min_samples_leaf':1,
      'min_samples_split':2,
'splitter':'random'}
```

- Based on parameters, entropy is better than Gini in our tree with 15 depth.

- Figure14 shows total impurity of leaves vs effective alphas of pruned trees so the nodes with smallest effective alpha are pruned first.
  Figure shows the ROC curve above the threshold line indicating a good model with area under curel 0.81.

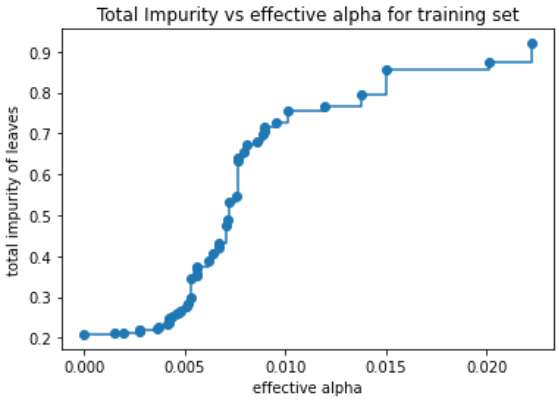


Figure14

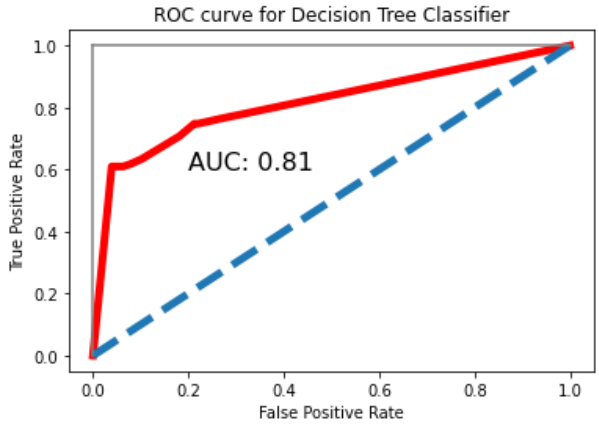


Figure15

- It is shown is figure16, that all the selected features we discussed in feature selection highly contributes to the model accuracy.

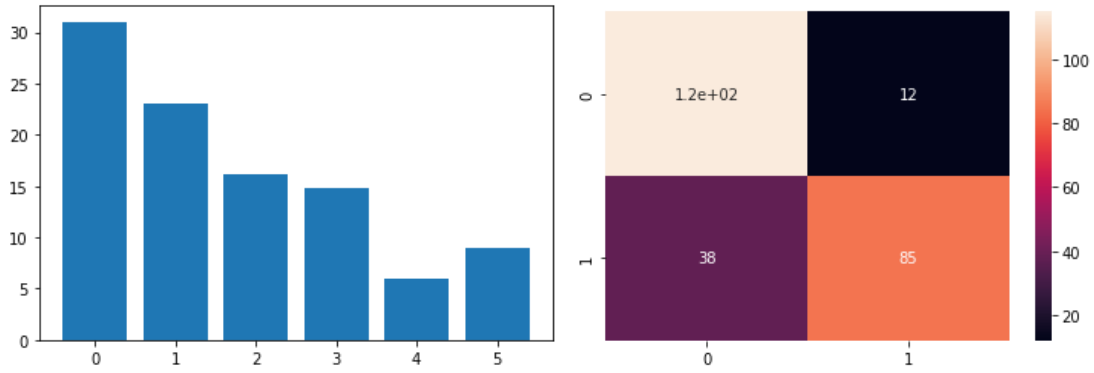


Figure16

**Neural Network:**

- For Neural network, normalization has decreased the model accuracy. However, after balancing the accuracy is back to previous 72% for testing. The feature selection has kept the accuracy for to 74.40% with modified outliers.

**Table4**

| | Raw% | Scaling% | Balancing% | Outlier% | Selection% | Optimized% |
|---|---|---|---|---|---|---|
| Training | 73.77 | 73.77 | 71.31 | 71.65 | 70.62 | 70.00 |
| Testing | 72.00 | 68.57 | 72.00 | 70.00 | 74.40 | 72.40 |

- For fine-tuning the model, RandomSearch was chosen with 10 maximum trials and improved for accuracy. The hyperparameters optimized for our neural networks are: 1$^{st}$, 2$^{nd}$ and 3$^{rd}$ layer where the optimal weights were calculated and learning rate for Adam optimizer. The accuracy of the model changed from 74.40% to 72.40% for testing data. The results could be improved if the

epochs and layers were increased with hyperparameters. The figure17 and figure18 show the model accuracy and loss at each epoch and it is clear that high optimization was noticed during initial epochs.

**Best Parameters:**

```
Layer 1 units 256
Layer 2 units 256
Layer 3 units 256
learning_rate for Adam Optimizer 0.001
```

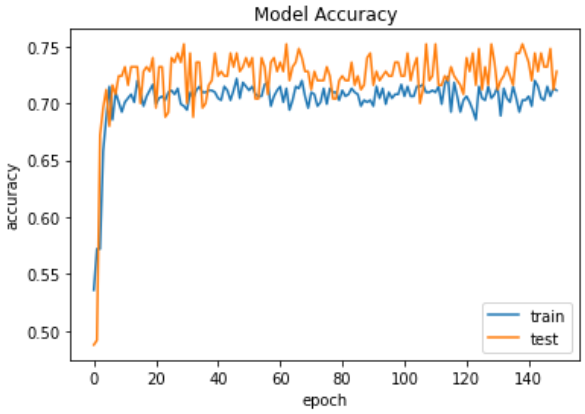


Figure17

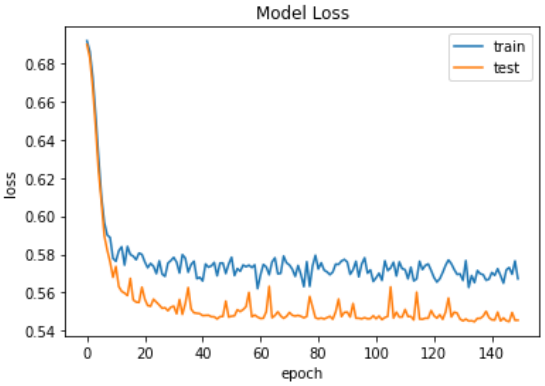


Figure18

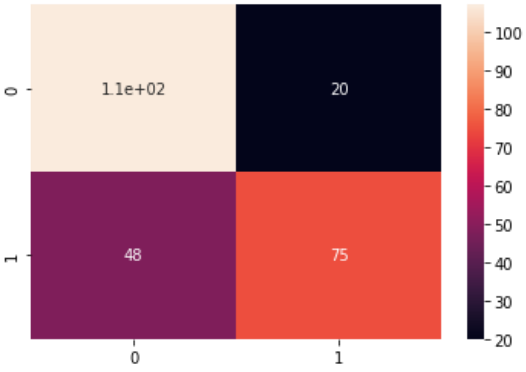


Figure19

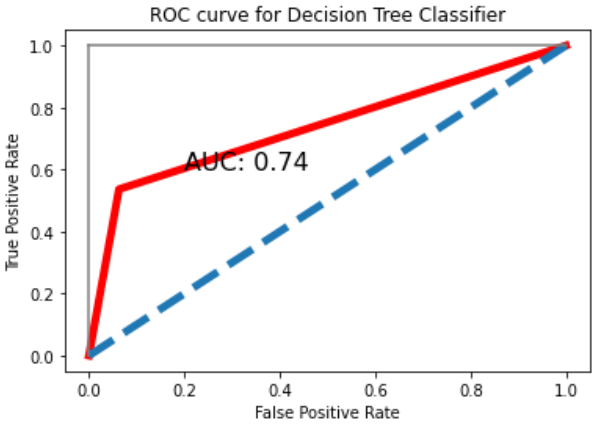


Figure20

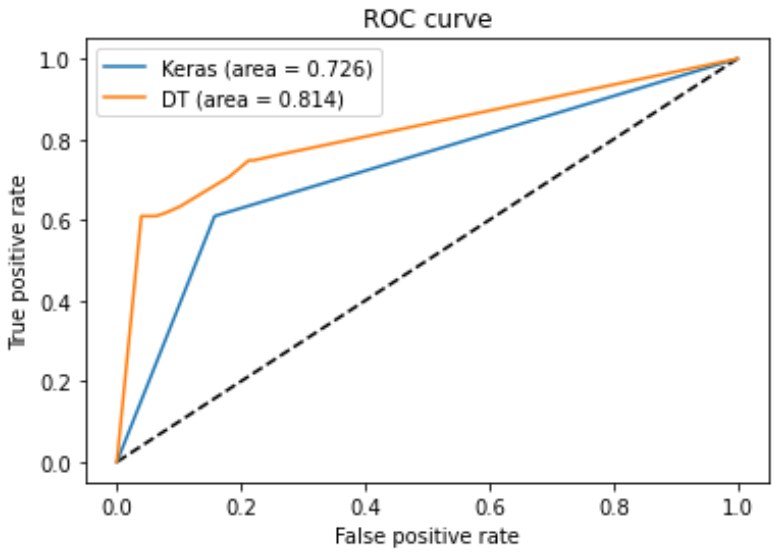


Figure21

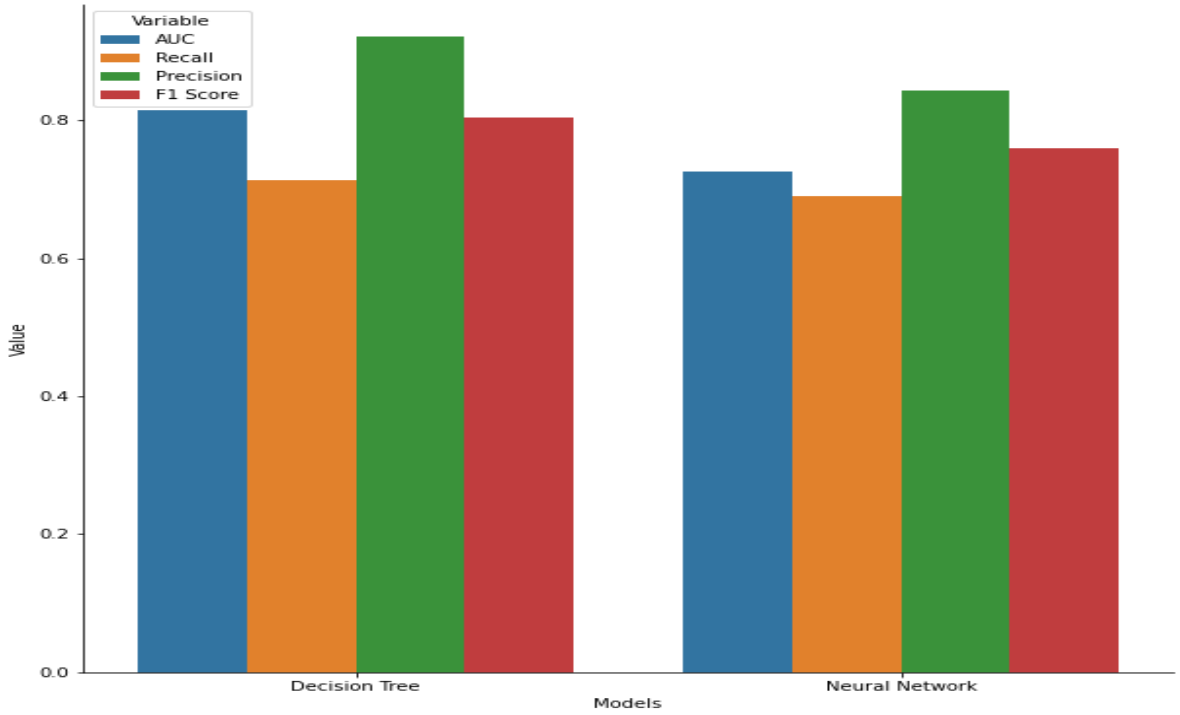


Figure22

As shown is the figure19, 29, Decision tree is clearly outperforming Neural Network. F1 score and accuracy are better for Decision tree. Precision is high for both the models with good F1 score. Recall(sensitivity) can be increase if model requires to cover false positive score. Although, high precision is good for patient prediction.

# Discussion and Conclusion

**Finding and Methodology improvements:**

- Multiple machine learning models are produced to the prediction of Indian liver patients. It was interesting to observe that KNN, SVM and Neural Network were showed with high accuracy on the dataset. In our case, decision tree outperformed the Neural Network model with better accuracy (80.00%) than (Gajendran and Varadharajan, 2020).However, a hybrid model Neuro-SVM performs extraordinary with 98.83% accuracy (Nagaraj and Sridhar, 2015). Our decision tree performs better than (Venkata Ramana, Babu and Venkateswarlu, 2011) Random forest. Including all features showed the increase in performance.
- In (Gajendran and Varadharajan, 2020) it was shown that the accuracy of the model increases when feature selection is performed although the features picked for the training were not described. Based on the algorithmic feature selection and correlation, the model performance seems to decrease by a little for Decision Tree but increase for Neural Network in our case which indicates it's better to pick features based on model. As said in (Nagaraj and Sridhar, 2015) the feature selected are also decreasing the accuracy of the model. Moreover, Gender can be opted out as not much performance change was noticed after removing it.(Venkata Ramana, Babu and Venkateswarlu, 2011) showed that PCA improves the performance of the models.
- Interestingly nobody handled *outliers* and it improved accuracy for our models.
- Balancing the dataset definitely improved the efficiency of the model indicating the importance of having equal number of liver and non-liver patients in the dataset. The dataset should be considered populated with more non-liver patient's data from real world.
- (Venkata Ramana, Babu and Venkateswarlu, 2011) and (Nagaraj and Sridhar, 2015) used **HYBRID MODELS** that surely contributed to increase in performance of the model as features selected by one model can be used efficiently by other. Although in our can we used feature selection method and classification algorithm together which was similar to (Venkata Ramana, Babu and Venkateswarlu, 2011).

**Future Applications:**

Liver function test (LFT) can come abnormal not just because of liver disease but due to congestive heart failure, malignancy or inflammatory and infection conditions(Sørensen *et al.*, 1991; Kim *et al.*, 2004; Roderick, 2004; Schalk *et al.*, 2006). Therefore, it becomes tough to identify the liver disease as no symptoms are observed in patient. The further low risk then can be performed for the investigation of the disease The problem becomes harder with less machines for testing and high number of patients in places such as like India. Therefore, the system could be beneficial predict the liver disease in patient using basic liver parameter and also reduce the burden from hospital staff by making fast and efficient prediction. In India, where the liver related deaths are high. The lives can be saved by predicting the liver disease at early stage as it can be treated if not cured.

## References


Gajendran, G. and Varadharajan, R. (2020) "Classification of Indian liver patients data set using MAMFFN," in *1ST INTERNATIONAL CONFERENCE ON MATHEMATICAL TECHNIQUES AND APPLICATIONS: ICMTA2020*. *1ST INTERNATIONAL CONFERENCE ON MATHEMATICAL TECHNIQUES AND APPLICATIONS: ICMTA2020*, AIP Publishing. doi: 10.1063/5.0025395.

Kim, H. C. *et al.* (2004) "Normal serum aminotransferase concentration and risk of mortality from liver diseases: prospective cohort study," *BMJ (Clinical research ed.)*. BMJ, 328(7446), p. 983.

Nagaraj, K. and Sridhar, A. (2015) "NeuroSVM: A graphical user interface for identification of liver patients," *arXiv [cs.LG]*. Available at: http://arxiv.org/abs/1502.05534.

Roderick, P. (2004) "Commentary: liver function tests: defining what's normal," *BMJ*, 328.

Schalk, B. W. M. *et al.* (2006) "Change of serum albumin and risk of cardiovascular disease and all-cause mortality: Longitudinal Aging Study Amsterdam," *American journal of epidemiology*. Oxford University Press (OUP), 164(10), pp. 969–977.

Sørensen, H. T. *et al.* (1991) "Epidemiology of abnormal liver function tests in general practice in a defined population in Denmark," *Danish medical bulletin*, 38(5), pp. 420–422.

Venkata Ramana, B., Babu, M. S. P. and Venkateswarlu, N. B. (2011) "A critical study of selected classification algorithms for liver disease diagnosis," *International journal of database management systems*. Academy and Industry Research Collaboration Center (AIRCC), 3(2), pp. 101–114.