Predict Liver Failure based on People's Demographics

1. Introduction

The aim of this document is to explore the idea of being able to build a Machine Learning / Deep Learning model which can be used to predict the possibility of a liver failure in individuals given their demographics and health related information. This report describes this work of trying to learn people's lifestyle and to try and predicts the possibility a Liver failure based on this information.

1.1 Business Problem

The **Liver** is the largest organ in the body of a Human being and also other vertebrates. The function of a liver is that it detoxifies various metabolites, synthesizes proteins, and produces biochemicals necessary for digestion. Its other roles in metabolism include the regulation of glycogen storage, decomposition of red blood cells and the production of hormones.

Considering the critical functions that a liver performs, a liver failure is a life-threatening condition that demands urgent medical care. Liver failure is a loss of one's liver function and that could occur slowly (**Chronic Liver Failure**) or rapidly (**Acute Liver Failure**). The commonly known causes for liver failure are paracetamol overdose, excessive alcohol consumption, viral hepatitis, food poisoning etc.

There is no known way to compensate the absence of the liver function in the long term which makes Liver Failure a serious disease to worry about. However, if detected early enough, supportive medical care can be given to treat the symptoms and to try and see if its effects can be reversed.

Hence being able to predict the possibility of a liver failure in people will go a long way in helping this noble cause. This project is a first and very basic step in trying to see if we can help in predicting the possibility of a liver failure in individuals based on some basic data related to their demographics and health.

2. Data

2.1 Data Requirements

To be able to build and train a prediction model that will help recommend the possibility of a Liver Failure in people based on their demographics and health information, we would first need some historic data about Liver Failures that can be used as the basis to build and train our prediction algorithm.

Now, what kind of data do we need to be able to make predictions? For answering this question let us look at the causes of a liver failure. Based on the causes we understand that we will need some basic health related information which would include people's vitals, information about their habits, physical activities etc.

For what kind of people should this information be collected? We should make sure that we gather this information for a good mix of people i.e. people who have suffered from liver failure and people that have not.

2.2 Data Collection

The demographics and health related data that we were looking for with respect to this project was available to us at Kaggle. This data was collected and made available by the JPAC Center for Health Diagnosis and Control. Since 1990, they have conducted nationwide surveys using trained personnel and have collected a wide variety of demographic and health information using direct interviews, examinations, and blood samples. This dataset consists of selected information from 8,785 adults 20 years of age or older taken from the 2008–2009 and 2014–2015 surveys.

2.3 Data Cleaning

The Liver failure dataset from JPAC Center for Health Diagnosis and Control was downloaded from Kaggle. This data was available as an excel file. After downloading this data file when we started our analysis of the data, we found the following data quality issues:

- Missing features data – There were several feature columns for which data was missing for certain records. This constituted about 8% of the total number of records in our dataset.
- Missing classification data – There were about 1/3rd of the records in the dataset where people were not categorized as either having liver failure or not.

All of the data quality issues were related to missing data as categorized above. Based on the above observations, the data had to be cleaned up before we could continue with our analysis. The following cleanup activities were performed on our dataset to fix the problems observed.

- Missing classification data – The records missing classification data cannot be used for training or validation of our model since this data had not been labeled. Hence we will remove this from our data set.
- Missing features data – This data was dealt with using the following strategies
    - Imputing – Data related to few columns could be assumed or inferred based on its value distribution. The data for such columns was manually updated.

- Filtering – Data related to few columns like people’s Vitals cannot be imputed. Hence such records were removed from our data set.

In addition to the above data quality issues that were addressed, we also observed that there were a few features / columns in the dataset which had no bearing on our label data that we are trying to predict. These features / columns were removed from our data set.

After performing all the above mentioned cleanup we now have a good quality dataset to continue our analysis.

3. Methodology

3.1 Feature Engineering

The cleansed data set now comprised of the features that we will be using in our prediction algorithm to predict the possibility of a liver failure which is our Target column and has a binary value to tell us if the person had a Liver failure or not.

Our Feature set

Age
Gender
Body Mass Index
Waist
Maximum Blood Pressure
Minimum Blood Pressure
Good Cholesterol
Bad Cholesterol
Total Cholesterol
Dyslipidemia
PVD
Physical Activity
Poor Vision
Alcohol Consumption
Hyper Tension
Family Hyper Tension
Diabetes
Family Diabetes
Hepatitis
Family Hepatitis
Chronic Fatigue

After the data cleansing is done, the next step in our process is to feature engineer our dataset. As part of the feature engineering process, we will do the following

- Apply One Hot Encoding to the categorical features to make them Machine Learning friendly by converting them into encoded vectors.
- Merge all the features including the One Hot encoded features into a single feature vector.
- Normalize / scale the features to ensure that all the features were scaled to a value range of 0 to 1.

After feature engineering our dataset for our prediction algorithm, the next step in our process is to identify the most appropriate algorithm for defining our prediction model.

3.2 Model Definition

3.2.1 Understanding our Use-Case and Dataset

As a first step in Model Definition, let us understand our use-case and dataset and then look for Machine Learning algorithms that would best fit our use-case.

In our use case we are trying to predict the possibility of an individual running into a Liver failure. For modeling this use case, we have a dataset generated by JPAC Center for Health Diagnosis and Control in which we have a bunch of features that can be used for our prediction. Our target variable or label which we will be predicting gives us a binary value of 0 or 1 which would tell us the possibility of a liver failure.

Considering our use case and dataset, ours is a case of Binary Classification where we will need to predict a binary value using our feature set. Now let us look at the choices of Machine Learning and Deep Learning Algorithms that are available and best suited for our purpose.

3.2.2 Choosing a Machine Learning / Deep Learning Algorithm for our Model

Now that we have a good understanding of our use-case and the expectation from our Prediction model, let us take a look at the various Machine Learning and Deep Learning algorithms for choosing the most appropriate algorithm for defining our prediction model. As part of this project, we will be defining two models –

1) One Machine Learning model, and
2) One Deep Learning model

3.2.2.1 Choosing our Machine Learning Algorithm

We have already seen that our model is an example of a Binary Classification Model. Hence we will need to choose a **Supervised Machine Learning** algorithm that will be best suited for Binary Classification. The Supervised Machine Learning algorithms that can be used for our classification use case are listed below.

- Naïve Bayes
- Support Vector Machine
- Decision trees
- Random Forest
- Gradient Boosted Trees.

After analyzing all these algorithms, we decided to go use either **Support Vector Machine** or **Gradient Boosted Trees** for modeling our use case. The reason for narrowing down on these is as follows:

- **Support Vector Machine** – This is one of the most popular Supervised Machine Learning algorithms and is well suited for binary classification and regression analysis.
- **Gradient Boosted Trees** – This is also a very powerful Supervised Machine Learning algorithm that can be used for binary classification and regression analysis with powerful techniques for building predictive models.

To choose the best model for our use-case, we will build models using both these algorithms and then choose the one that gives us the best performance between the two.

3.2.2.2 *Choosing our Deep Learning Algorithm*

With respect to Deep Learning Algorithms, there is a wide range of algorithms to choose from. However, based on our use case and dataset, we will use a **Feed Forward Neural Network / Multi-Layer Perceptron** for our model as the Perceptron is a binary linear classifier and meets our needs for this use case.

Now that we have narrowed down on the Algorithms to be used for our prediction model, let us actually start the process of building the model.

3.2.3 Building our Prediction Model

3.2.3.1 *Preparing our Datasets for Training and Validation*

Before we start the process of building our models, let us first prep our data for training and validation of our models. For this purpose we will split our dataset into two and create a training dataset and a validation dataset. We will go with an 80:20 split to create our datasets i.e. 80% training data and 20% validation data.

3.2.3.2 *Building our Supervised Machine Learning Model*

Now let us start building our Supervised Machine Learning Model. In our previous step of defining our Model we had narrowed down on 2 algorithms for our prediction model. Now let us build 2 models using these 2 algorithms and check their performance. Then we will select the model which gives us the best performance and use that as our prediction model.

The **framework** that we have selected for building these models is **Apache Spark Machine Learning** framework. This framework is built on top of Apache Spark which will give us the flexibility of being able to use **parallel processing** in the future if we need to.

Next we built our prediction models and measured the performance of our models. The performance numbers observed are as shown below.

1) Support Vector Machine: Accuracy – 0.77 (77%)
2) Gradient Boosted Trees: Accuracy – 0.96 (96%)

Based on the above number we can clearly say that the Gradient Boosted Trees algorithm gives us a much better performance and hence is better suited for our use-case and data set. Hence we will continue with our prediction model using Gradient Boosted Trees as our Supervised Machine Learning Model for training and evaluation.

3.2.3.3 Building our Deep Learning Model

Now let us start building our Deep Learning Model. In our previous step of defining the model, we had narrowed down on **Feed Forward Neural Networks / Multi-layer Perceptron** as the algorithm for building our prediction model.

In this step of the process, we will actually build our Deep learning model and the measure its performance. The **framework** that we have selected for building these models is **Apache Spark Machine Learning** framework. This framework is built on top of Apache Spark which will give us the flexibility of being able to use **parallel processing** in the future if we need to. Also we will be building our Multi-layer perceptron using **Keras** as our Deep Learning Library.

The performance of our Multi-layer perceptron based prediction model is as shown below.

1) Multi-layer Perceptron: Accuracy – 0.94 (94%)

The above performance number justifies the choice of the Deep Learning algorithm for our model. As next steps we will continue to train and evaluate our Multi-layer perceptron for making predictions.

3.3 Model Training

Now that we have built our models, they are ready to be trained. As mentioned earlier, we have used Apache Spark as the framework for building out models. This give us the flexibility to use the power of parallel processing.

In this step, we will train our Machine Learning and Deep Learning models on the Apache Spark framework, using the training dataset that we had created.

After training the models on the Apache spark framework using our training dataset, we measured the performance of our models. We captured accuracy as the measure of performance for our classification model using the training dataset. The training accuracy measured for our models is as shown below.

1) Machine Learning – Gradient Boosted Trees:
    a. Training Accuracy = 0.96 (96%)
2) Deep Learning – Feed Forward Neural Network / Multi-layer Perceptron:
    a. Training Accuracy = 0.94 (94%)

Both our prediction models are giving us a pretty good accuracy measurement. Now, we will move forward to the next step in our process and evaluate our model.

3.4 Model Evaluation

Now that the training performance of our models is pretty good, let us go ahead and start evaluating our models to see how these models fair in their evaluation and if their validation performance is comparable to or better than their training performances.

Earlier I the process, we had already created a test dataset which can be used for evaluating the performance of our prediction model. This dataset is new to our model and is different from the dataset that was used to train our models. Again in the test dataset for each record, we have labels which classifies each of these as either “having” or “not having” liver failure.

Now, let us go ahead and evaluate out models using the test dataset and then measure the accuracy of our models against the test dataset. As part of our evaluation, we will also try and tune the hyper parameters defined for each of these models to ensure that we get the most optimal results. Example of some of the hyper parameters that we will be tuning are as follows:

1) Hyper Parameters for **Gradient Boosted Trees**
    a. Maximum Number of Iterations
2) Hyper Parameters for **Feed Forward Neural Network**
    a. Activation function
    b. Optimizer
    c. Loss Function
    d. Number of Epochs

The hyper parameters were tuned and the models were evaluated against the test dataset. We then captured the performance measure for the models based on the best hyper parameter configuration. The validation accuracy measured for these models is as shown below.

1) Machine Learning – Gradient Boosted Trees:
    a. Validation Accuracy = 0.92 (92%)
2) Deep Learning – Feed Forward Neural Network / Multi-layer Perceptron:
    a. Validation Accuracy = 0.93 (93%)

Based on the above measurements we can see that the accuracy measured for both our models i.e. the Machine Learning (Gradient Boosted trees) model and the Deep Learning (Multi-layer Perceptron) model is pretty good.

4. Results and Discussion

In the process of creating and evaluating a prediction model for being able to predict the possibility of Liver failure, so far we have defined, built, trained and evaluated two prediction models using the following algorithms.

1) Supervised Machine Learning – Gradient Boosted Trees
2) Deep Learning – Feed Forward Neural Network / Multi-layer Perceptron

The evaluation of these models was done using training and validation datasets and the performance measured using accuracy as a measure of evaluation. The performance numbers measured for our 2 models is as shown below.

1) Supervised Machine Learning – Gradient Boosted Trees
    a. Training Accuracy – 0.96 (96%)
    b. Validation Accuracy – 0.92 (92%)
2) Deep Learning – Feed Forward Neural Network / Multi-layer Perceptron
    a. Training Accuracy – 0.94 (94%)
    b. Validation Accuracy – 0.93 (93%)

Based on the above results we can confirm that the performance of both our prediction models is pretty good. Using these models we will able to predict the possibility of a Liver failure in individuals with almost 93% accuracy.

This is however the first step in building this prediction model. As next steps there is ample scope for enhancing this model in terms of functionality and also try and tune this prediction model to improve its performance levels. Some of the things that can be done as next steps to improve this model are

- Continuously train the model against updated data and tune the model accordingly to improve upon its performance
- Encapsulate the model behind a Rest API for it to be consumed by Data products that will be used for predicting Liver failure in people

5. Conclusion

The purpose of this project was to create prediction models that can be used to predict the possibility of a liver failure in people using basic demographics and health related information. Liver as we have seen earlier if a very critical organ in terms of the functions that it performs. Hence the goal here is to help people understand their possibility of getting a liver failure based on their lifestyle and health. If this can be predicted almost accurately then this will help people a long way in being able to get the right medical support required to either prevent a liver failure or recover from what could be the beginning of a liver failure.

Based on the observed results so far, our model seems to be working towards achieving this goal of trying and helping people and we will continue to work on refining this model to try and achieve this goal of ours completely.