Logan Staley

Machine learning term project

04/18/2023

CSCI4336.01 Introduction to machine learning

**Project Overview:**

In this project, we will implement a logistic regression algorithm from scratch using Java. The goal of the project is to predict the probability of a binary outcome based on a set of input features. We will use the gradient descent algorithm to optimize the model parameters and minimize the log loss function. We will use the Heart Failure Prediction dataset from UCI, which contains 12 clinical features related to heart failure, such as age, sex, blood pressure, and serum creatinine. I split the dataset into training and testing sets, and use the training set to train the logistic regression model. I implemented the gradient descent algorithm to optimize the model parameters and minimize the log loss function. We will also normalize the input features to improve the performance of the algorithm.

**Pre-processing-**

For visualization and data cleaning I used jupyter notebook to do such. By displaying the data I first removed the data that was unnecessary such as 'anaemia', 'diabetes', 'high\_blood\_pressure', 'sex', 'smoking', and 'time' using the Drop function. Then after that I constructed a correlation matrix, a heatmap, pair plot, histogram, scatter plot, and implemented logistic regression in the jupyter notebook. Lastly used a feature selection method that showed the importance of each feature. After that I decided to split the dataset in to a 20% for testing and 80% training the logistic regression model. scaling the dataset which was done in the java code with a unit vector normalization function. Inside the jupyter note book I was able to fit the logistic regression model to fir the data, that then was able to show the importance of each feature in deciding if the patient could potentially have another heart failure issue. After looking at the data in the jupyter notebook I decided that the features of creatinine\_phosphokinase,ejection\_fraction,serum\_creatinine,serum\_sodium,DEATH\_EVENT were all important in the processing and direction of the logistic regression model and used these features in the curation of the model.

**Visualization-**

Chart, treemap chart

Description automatically generatedFigure 1: Correlation matrix

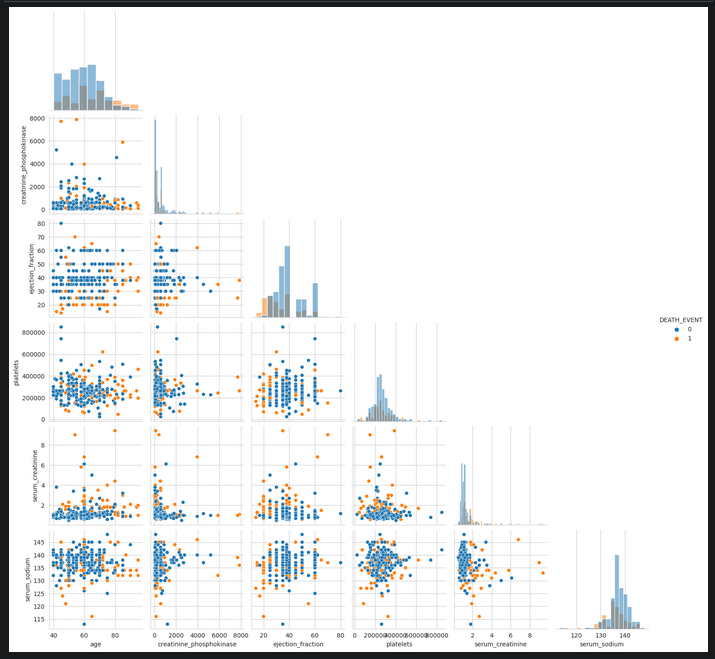


Figure 2: Pair plot

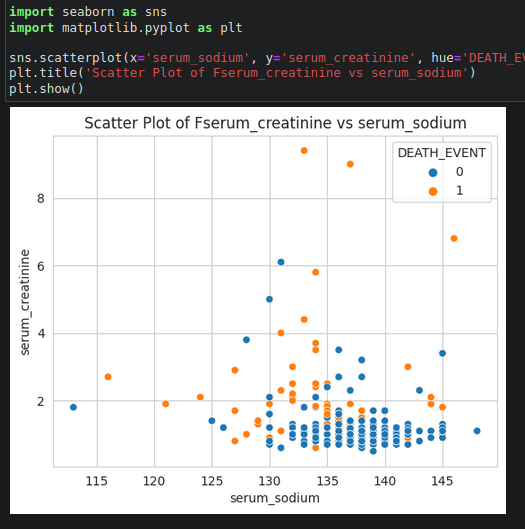


Figure 3: scatter plot



Figure 4: Histogram

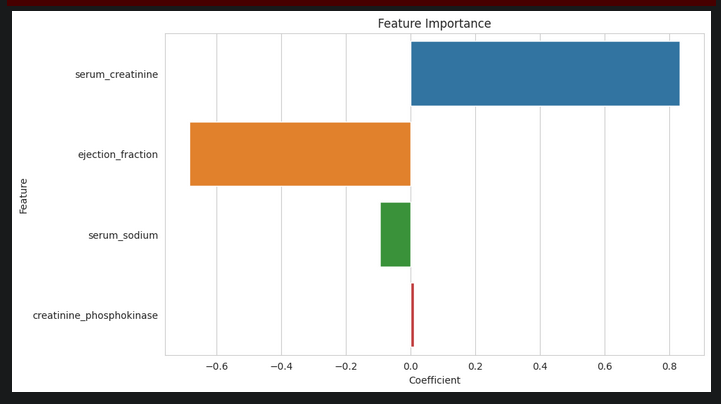


Figure 5: feature importance

**Data-**

The data set that is being used is “Heart failure clinical records data set” which contains 13 attributes which are age: age of the patient (years) anaemia: decrease of red blood cells or hemoglobin (boolean) high blood pressure: if the patient has hypertension (boolean) creatinine phosphokinase (CPK): level of the CPK enzyme in the blood (mcg/L) diabetes: if the patient has diabetes (boolean) ejection fraction: percentage of blood leaving the heart at each contraction (percentage) platelets: platelets in the blood (kiloplatelets/mL) sex: woman or man (binary) serum creatinine: level of serum creatinine in the blood (mg/dL) serum sodium: level of serum sodium in the blood (mEq/L) smoking: if the patient smokes or not (boolean) time: follow-up period (days) [target] death event: if the patient deceased during the follow-up period (boolean). These features were used and reduced in the jupyter notebook, after observing the data with the figures above, I removed the Boolean variables, and out of scope variables. This resulted in using the 3 major features such as Serum\_creantinine, Ejection\_fraciton, Serum\_sodium and Creatinine\_phosphokinase. After analyzing these 4 features they are shown in the 5 figures above. The correlation matrix will compare each data set and is related to the correlation matrix shows that age platelets and creatinine phosphokinase are not necessarily needed. Pair plot, shows the relations ship between two of the more important features such as Serum sodium and Serum creatinine having a few outliers but a few pockets are noticeable. Histogram shows frequency of the data plots and a basic view of each feature, and feature importance the magnitude of each feature and show each impact of each feature. Logistic regression is a good fit for this problem because it can handle binary classification tasks, like predicting death events, and can work well with a mix of continuous and categorical features. Additionally, logistic regression is relatively simple and can provide interpretable results, which can be useful in medical applications.

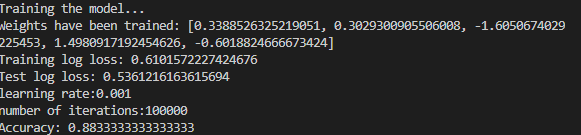
**Implementation-**

For this data set I noticed that logistic regression with gradient descent would be a good fit for the performance of accurately predicting when a patient is at risk of another heart failure. The logistic regression with gradient descent was implemented with java in a VS code IDE, also a short python script that quickly cleaned the data in to a smaller CSV file removing some of the binary features of the CSV file mentioned above.

Once that was done I implemented the cleaned csv file in to the java environment and read the values in to a dynamic list to make the process of handling the large data set smoother. Once imported to the list I incremented through the list placing them in to arrays X for the main features and Y for the Target variable.

I then split the arrays with an 80% Training data, and 20% testing data for the logistic regression model, but after a few runs and manipulating the data, it wasn’t running quite right. But after deciding to normalize the data with a unit vector method, the model is then trained on 80% of the data. For training the data we used logistic regression with gradient descent to optimize weights. I went with an learning rate of .001 and an iteration count of 100000. The weights are calculated to be around Weights have been trained [0.3388526325219051, 0.3029300905506008, -1.6050674029225453, 1.4980917192454626, -0.6018824666673424](This is an example of the weights). After training the weights, I then implemented a Log loss function that finds the values for both the trained and test values. With and outcome of Training log loss: 0.6101572227424676 Test log loss: 0.5361216163615694, with running a linear regression model achieved worse results. Then after computing the log loss I implemented an accuracy test for the model, with resulted in an accuracy around Accuracy: 0.8833333333333333. meaning the performance of the logistic algorithm on detecting patients potential for a heart failure, has an 80% chance to produce the correct prediction for the patient. With this being a medical issue, this is enough to give medical professionals enough information to provide medical assistance and information to patients that are at risk.

**Outcome-**



**Running the program-**

The only changes that should be needed are the directories for the CSV read files along with the data cleaning python program if needed.

**Conclusion –**

During this term project I had a lot of issues getting a good connection with this data and trying to find a way to correlate the features with one another. Especially with this data set, I found that it was difficult to connect features with one another to accurately predict critical heart failure of patients. But this makes sense for when trying to implement a machine learning problem in the future it different projects and project boundaries. So instead of trying to find a different model or data set I decided to put my head down and deal with the issues and try to get the logistic regression model to work with the data. I believe I was able to get correctly, to identify with decent 86% accuracy the ability to notify patients and medical personnel of potential heart failure patterns in patients. The main problems with this project would have been to get a good performance rating and the normalization of the data, this resulted in a few days of headache trying to manipulate the algorithm to work properly. But overall the experience and time spent on this was insightful and opened my eyes to see what else this could be used for, with my internship this summer it will be neat to see if some machine learning models can be used to improve work flow.