Metabolomic Data Analysis

Problem Definition:-

With the help of various machine learning classifiers we are trying to predict whether the given samples are healthy or diseased. We are also finding features which are important in predicting the sample.

Importance:-

On successful training and prediction using the machine learning algorithm we will be able to classify samples as either healthy or defective and hence this is very much useful.Instead of looking into various parameters we find the most significant parameters for testing sample (in our case we have 3000 parameters and testing and collecting data would then become tedious task).So we will pick out only the significant parameters out of them.

Motivation:-

Our motivation to do this project is to gain experience in data analysis and machine learning using python

Implementation:-

Background:-

We have a protein dataset which contains 12 samples (rows) , 3009 features(protein ids) and 1 class ('Target') . The zeros in the data of protein ids represent NAN values (experimental errors.). In the ‘Target’ column, ‘0’ represents healthy samples and ‘1’ represents diseased samples.

Reading data frame:-

After reading data in csv format we have to edit the data frame,and make few changes.Unlike conventional way samples were in columns,so we changed samples to rows and then changed the column names.Then we converted all entries into numeric form,Healthy as 1 and Diseased as 0.Then we renamed few rows and columns.Now our data frame is properly read

Code for missing values:-

There are many zeros in our data frame and hence we have to delete columns which have uncertainty within it.Having so much of unmeasured data will not give accurate result.So we have to delete columns which has zeroes beyond a limit.

We removed protein ids having more than 30% zero values. We wrote a function ‘remove_zero_values(df,k)’ to achieve this functionality. After this step our dataset contained 1363 features.

Replacing zeros with mean of the feature:-

Even after removing most of the zeros,there will be few zeroes left and these be assigned some appropriate value as a method of filling data.In our case we replaced zeroes with mean of that respective column.

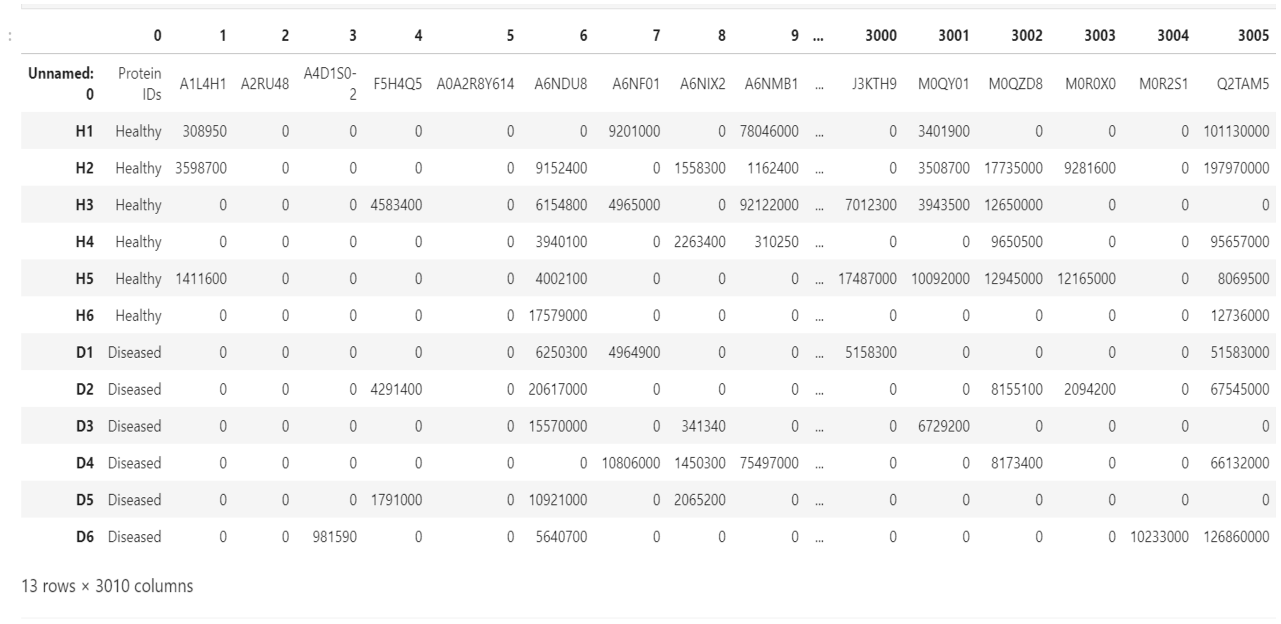
Zeros in the remaining features were replaced by their respective mean. We wrote a function “ replace_zero_mean(df) ” to achieve this functionality.We calculate mean of each column and replace every zero with corresponding mean.

<u>Correlation matrix</u>:-

We plotted correlation matrix for the samples using matplotlib and seaborn library.This helps in identifying if samples are perfectly read.In our case sample 5 was behaving comparatively odd with rest other sample.However we still included that in training our model.

From the correlation matrix we conclude that sample 5 is not correlated with other samples since it’s values were close to the zero. This means the experimenter has to redo the experiment for this sample.Heatmap is plotted by importing seaborn so that visualization can be done easily.

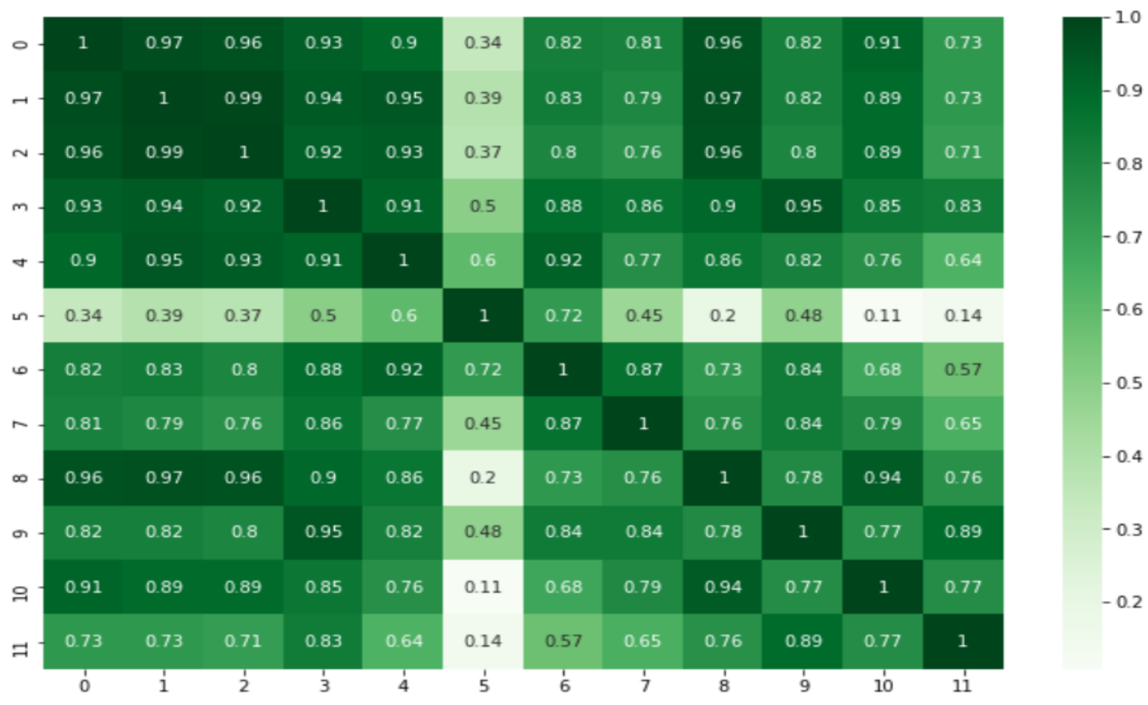
(Data frame we are working with)



	0	1	2	3	4	5	6	7	8	9	...	3000	3001	3002	3003	3004	3005
Unnamed: 0	Protein IDs	A1L4H1	A2RU48	A4D1S0-2	F5H4Q5	A0A2R8Y614	A6NDU8	A6NF01	A6NIX2	A6NMB1	...	J3KTH9	M0QY01	M0QZD8	M0R0X0	M0R2S1	Q2TAM5
H1	Healthy	308950	0	0	0	0	0	9201000	0	78046000	...	0	3401900	0	0	0	101130000
H2	Healthy	3598700	0	0	0	0	9152400	0	1558300	1162400	...	0	3508700	17735000	9281600	0	197970000
H3	Healthy	0	0	0	4583400	0	6154800	4965000	0	92122000	...	7012300	3943500	12650000	0	0	0
H4	Healthy	0	0	0	0	0	3940100	0	2263400	310250	...	0	0	9650500	0	0	95657000
H5	Healthy	1411600	0	0	0	0	4002100	0	0	0	...	17487000	10092000	12945000	12165000	0	8069500
H6	Healthy	0	0	0	0	0	17579000	0	0	0	...	0	0	0	0	0	12736000
D1	Diseased	0	0	0	0	0	6250300	4964900	0	0	...	5158300	0	0	0	0	51583000
D2	Diseased	0	0	0	4291400	0	20617000	0	0	0	...	0	0	8155100	2094200	0	67545000
D3	Diseased	0	0	0	0	0	15570000	0	341340	0	...	0	6729200	0	0	0	0
D4	Diseased	0	0	0	0	0	0	10806000	1450300	75497000	...	0	0	8173400	0	0	66132000
D5	Diseased	0	0	0	1791000	0	10921000	0	2065200	0	...	0	0	0	0	0	0
D6	Diseased	0	0	981590	0	0	5640700	0	0	0	...	0	0	0	0	10233000	126860000

13 rows × 3010 columns

(correlation heat map)

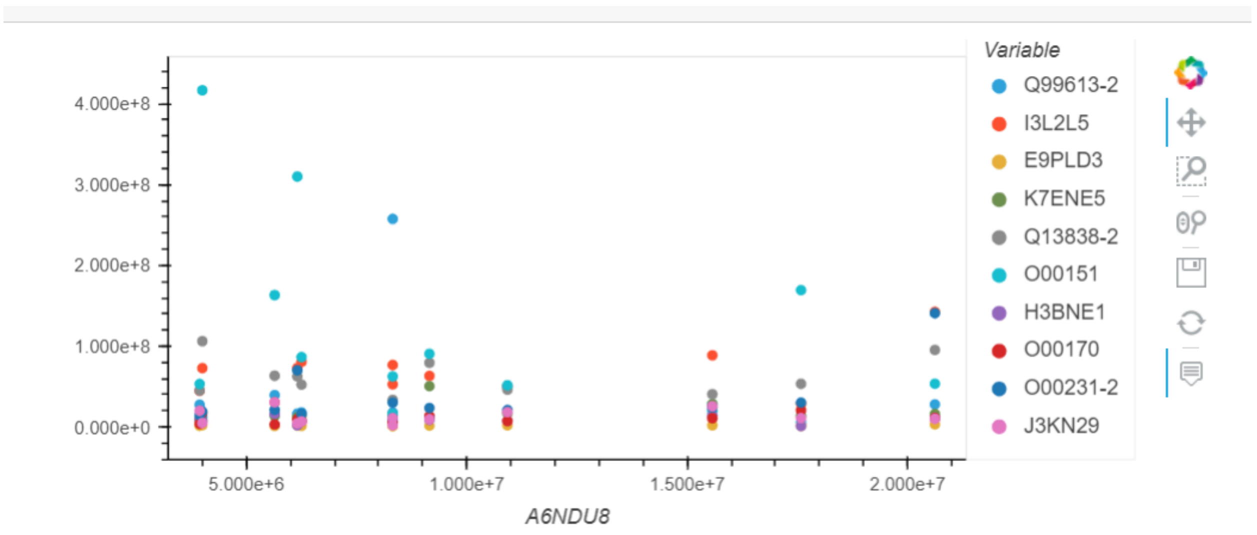


Plotting a scatter plot:-

In order to visualise the data we have plotted the

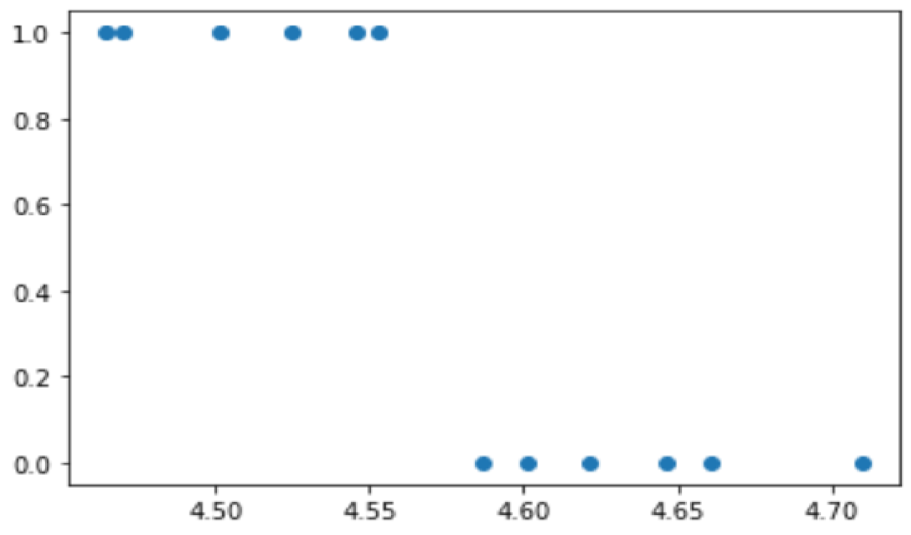
Scatter plot. The scatter plot shows that most of the data points were close to zero as there were some outliers. To solve the problem we use rescaling techniques.

Here we included only a few parameters while plotting the scatter plot as considering all parameters would be difficult.



After observing correlation of result with other parameters we found significant parameters amongst all others. Below is the scatter plot between the one of significant parameter "P02760" and the result.It could be observed that result is now clearly distinguishable and we can classify between healthy and diseased.

(scatter plot between result and one significant parameter["P02760"])



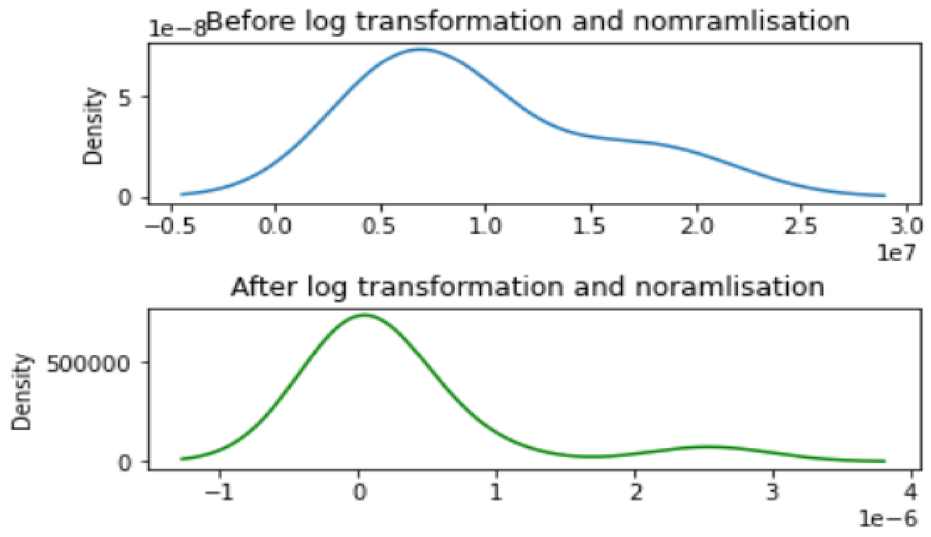
Log transformation and normalisation:-

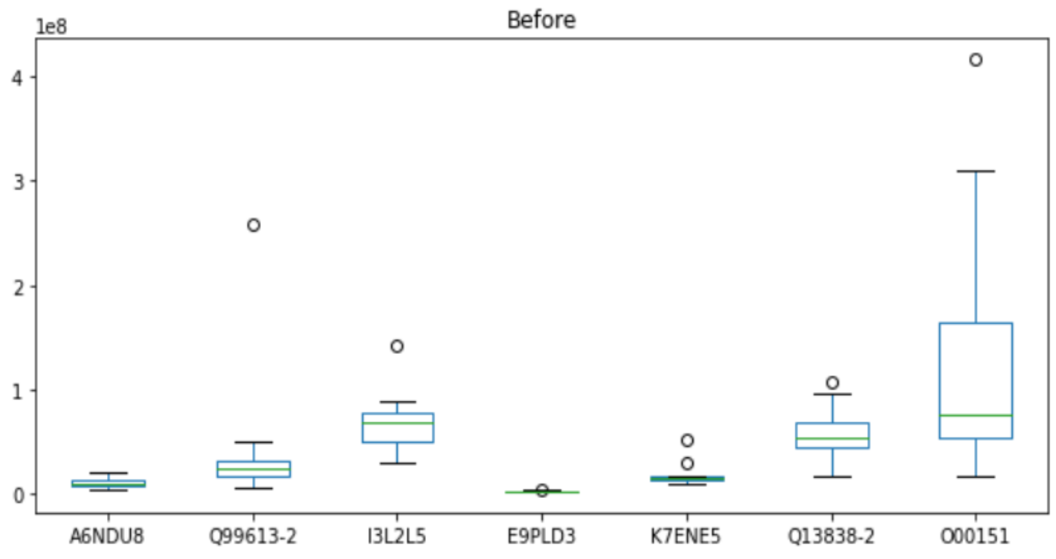
We applied log2 transformation to rescale the data. For this we used numpy library and numpy's log2 function.In our data frame entries were not in a particular range and there was huge difference between few entries and hence we applied log transformation

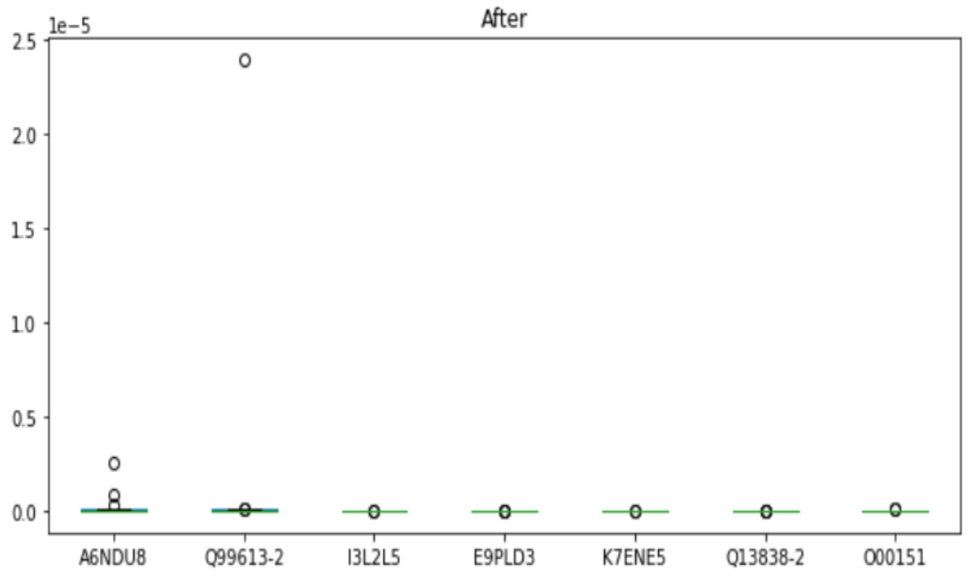
Then we did normalisation of the data. For this we wrote a function 'normalisation(dataframe)'.We actually applied to kinds of normalization techniques.In standard normalization we subtracted mean and divided by standard deviation for each column whereas in minmax normalization we subtracted by minimum value of respective column and divided by (maximum-minimum) value of that respective column

Plotting box plot and density plot:-

In order to visualise the effects of log transformation and normalization, we plotted the box plot and normal plot for some features before and after transformation and normalisation







Feature selection:-

Since our dataset contains a lot of features,and it may also be considering all parameters might lead to overfitting of data.Few parameters might not be even correlated with the result which may lead to errors in prediction . So we tried to reduce the features and take only important features. To achieve this functionality we use

ExtraTreesClassifier. With the help of this classifier we chose top 50 significant features.

<u>KMeans cluster:-</u>

Main reason to use KMeans cluster is to assure that the 50 features which are filtered as significant were actually significant. To validate this we use KMeans cluster. KMeans cluster is able to divide our data in two clusters .After looking how the clustering is done we could judge if the filtered parameters were actually significant or not.And in our case features were significant and we applied other few models to train data sets and predicted the test data set

<u>Predict the labels:-</u>

We splitted the data into train and test data sets ,using train test split method and trained data set and predicted the other data set

We used a <u>logistic regression</u> classifier to train the data and predict the labels. Logistic regression gave accuracy of 66%.
We also used <u>SVM</u>(support vector machines). SVM gave accuracy of whooping 100%.

The machine learning models were implemented by importing sklearn library.

Conclusion:-

Lack of availability of lot many samples was one drawback, however considering limitations we have done pretty good work.By the end of project we figured out which proteins were important in classifying the sample out of total number of 3000 proteins and measuring for all 3000 proteins is waste of time and may overfit the data.Results obtained through logistic regression and SVM models were promising.SVM predicted all the test samples correctly whereas logistic regression model had accuracy of (0.66).Finally we analyzed the data and after transformation of data and training machine learning models we could now classify samples as either healthy or diseased.

By-

Ashish Kondra
Deepak Sanjay Thorat
Shubhi Garg