Genetic Microarray Data Disease Classes Prediction using Machine Learning Algorithm

Fidelis Aimua [SID: 002-27-9591] Punan Chowdhury [SID: 002-32-3073]

Abstract- Healthcare has become one of the leading fields for the implementation of machine learning in recent times. Research facilities as well as medical institutions strive hard to improve and make more informed and accurate decision for patients' diagnosis and care. Volumes of complex medical dataset can be better analysed and processed using machine learning prediction models resulting in a more informed prognosis on disease diagnosis. This allows medical practitioners to easily identify terminal and other life-threatening disease at an early stage, prompting early treatment before its advanced stage. This research paper looks at how different classifiers can be applied on a genetic dataset obtained from a gene microarray, comparing results obtained.

Keywords- Machine Learning Classifiers, Neural Networks, Genetic Data, Prediction

I. INTRODUCTION

Data mining technique have proven to be a very reliable tool over the years in various clinical systems, aiding in prediction and diagnosis of various diseases and resulting in high accuracy rates. The techniques are highly successful as every disease's genetic makeup occurs naturally, which could be identified and defined base on the collected gene samples. Such prognosis will aid medical practitioners in giving patients accurate diagnosis and warning beforehand. The aim of this project is to train various classifiers on the provided genetic data for disease classification. Different classifiers work well on certain data sets obtaining the best suitable classifier.

This paper focuses mainly on analysing the performance of disease prediction approaches using various supervised machine learning techniques. It involves selecting specific genes of various classes of disease to obtain the best predictive disease modelling accuracy. For this work, the classification model applied utilized training data separated from the gene set. Results from different supervised machine learning algorithms have been compared resulting in a more robust, accurate and less error prone biased results.

II. DATA AND METHODOLOGY

A dataset collected from DNA microarray measuring expression levels for large number of genes simultaneously was used for this project. The details of the dataset to be trained and tested by the prediction model is enumerated as follows.

No of classes(targets): 5 (EPD, JPA, MED, MGL, RHB)
No of features used: 7070 genes expression
No of samples in training dataset: 69
No of samples in test dataset: 23

A. data loading and processing

Python was used. This enables effective data manipulation and improves processing speed. The dataset was split into training and test sets, labelled as 'pp5i_train.gr.csv' and 'pp5i_test.gr.csv' respectively.

B. class label

Pre-processing function from the Scikit library was used to group the 'pp5i_train_class.txt' target class into individual labels.

C. data cleaning

The minimum and maximum threshold value for both the train and test data was limited to between 20 and 16000. Genes with less than 2-fold difference were removed from the train data.

D. Feature selection

The ANOVA F-value was applied for the feature selection. It is a value on the F distribution whose value can be used to determine whether the test is statistically significant. It is obtained by dividing two mean squares and helps to determine the ratio of explained variance to unexplained variance.

For each class, we generate subsets with the 2,4,6,8,10,12,15,20,25 and 30 top genes that have the highest F-value. This was carried out to help find for each class the set of best genes to discriminate it from other classes. We then combined the top genes for each class into a single file for training the models. The Scikit-learn library was used for this purpose.

E. K-fold cross validator

This provides train/test indices to split data in train/test sets. The data set splits into k consecutive folds (without shuffling by default). Each fold is then used once as a validation while the k-1 remaining folds form the training set.
For this paper, the sklearn library with a cross validation of 5-splits is used.

III. MACHINE LEARNING CLASSIFIERS

A classifier in machine learning can be termed as an algorithm that automatically orders data into one or more of a set of classes. The most common example is an email classifier scanning emails to classify them into labels: Spam or Not Spam.

A. Naïve Bayes

These are built on Bayesian classification method. It relies on Bayesian theory, an equation describing the relationship of conditional probabilities of statistical quantities. In Bayesian theory, one is interested in finding the probability of a label given some observed features. This can be represented mathematically,

$$P(L|Features) = \frac{P(Features|L)P(L)}{P(Features)}$$

Where L can be referred to as labels.

There are different types of Naïve Bayes which include Gaussian Naïve, Bernoulli Naïve, Multinomial Naïve Bayes amongst others. For the purpose of this paper, the Gaussian Naïve Bayes will be used.

Gaussian Naïve Bayes classifier: In this classifier, it is assumed that data from each label is drawn from a simple Gaussian distribution. Here the likelihood of the features is assumed to be Gaussian which can be represented mathematically as:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi y\sigma^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Parameters μ_y and σ_y are estimated using maximum likelihood.

This algorithm is fast and can save a lot of time. It is better suited for categorical input variables than numerical variables. Some of its disadvantages include its assumption that all predictors are independent, rarely happening in real life, thus reducing its applicability in real-world use case. Also, the possibility of its estimation being wrong is very high

Library used: GaussianNB from scikit-learn (python)

B. Decision tree

This is a tree-based method in which each path starting from the root is represents a sequence of data splitting until a Boolean outcome is reached at the leaf node. The data is split into nodes until the leaf node of the tree is pure. To avoid overfitting, the depth of the tree is restricted. Its main advantage is its ability to use different feature subsets and decision rules at different stage of classification. It also requires less effort for data preparation during pre-processing. However, a small change in the data can lead to large change in the structure of the decision tree causing instability.
A general decision tree consists of one root node, a number of internal and leaf nodes and branches. Leaf nodes indicate the class to be assigned to sample while each internal node of a tree corresponds to a feature. The branches represent conjunctions of features that lead to classification.
One of the most commonly used Decision tree methods is the C4.5 which is an inductive algorithm developed by Quinlan (1993). It uses an approach which uses information theoretically measured based on ‘gain’ and ‘gain ratio’.

For instance, let S be Current state, P_i as probability of an event i of state S or Percentage of class i in a node of state S. We can therefore say, Entropy is

$$E(S) = \sum_{i=1}^{c} - P_i \ log_2 P_i$$

Also,
If B is the feature, Y its current state. Then gain is:

Information Gain (Y, B) = Entropy (Y) – Entropy (Y, B)

Library used: DecisionTreeClassifier from scikit-learn (Python)

C. K – Nearest Neighbour (KNN)

This is a non-parametric supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. Though it can be applied for regression problems, it is typically used for classification algorithm, working off the assumption that similar points can be found near one another. The KNN classifier groups and predicts using the Euclidean distance 'd' calculated with the labelled trained data. Distance between two points is obtained using

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(y_i} - x_i)^2$$

One of the major advantages of this algorithm is its ability to accept new data seamlessly without it impacting on the accuracy of the algorithm since the KNN algorithm requires no training before prediction. It however does not work well with large dataset since the cost of calculating the distance between new point and existing one is huge thus degrading the performance of the algorithm

Library used: KNeighborsClassifier from scikit-learn (Python)

D. Neural Network- Multi-Layer Perceptron (MLP):

The MLP is a supplement of the feedforward neural network. It consists of three layers, the input layer, output layer and hidden layer. The input layers receive the input signal to be processed while the output layer performs the task of prediction and classification. The MLP true computational engine is the arbitrary number of hidden layers placed between the input and output layers. Data flows in a forward direction from input layer to output layer like the feedforward network. MLP neurons are trained with the back propagation learning algorithm. They are designed to approximate any continuous function and can solve problems which are linearly separable and works well with large input data. Its disadvantage lies in the fact that one cannot tell the extent each independent variable is affected by the dependent variable, hence making computation cumbersome and time sapping. MLP are majorly used in pattern classification, recognition, prediction as well as approximation. Fig.1 shows the schematic representation

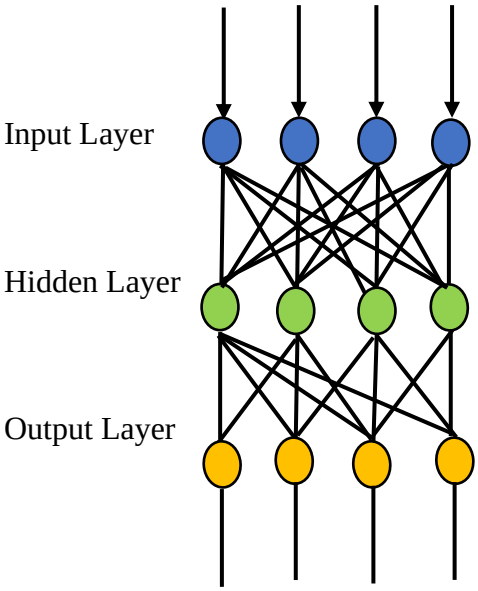

Fig.1. Schematic representation of MLP with one hidden layer

Library used: MLPClassifier from scikit-learn (Python)

E. Random forest:

This is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Knowing that individual decision trees typically exhibit high variance and tend to overfit, the injected randomness in forest yield decision trees with somewhat decoupled prediction errors. Taking an average of those prediction can lead to some errors cancelling out. Random forest achieves a reduced variance by combining diverse trees, at the cost of a slight increase in bias, sometimes. However, the variance reduction is often significant resulting in an overall better model.

Library used: RandomForestClassifier from scikit-learn (Python)

IV. RESULTS AND MODEL COMPARISON

A. Model Accuracy and Error rates of Machine Learning

At the end of the classification, most of the classifiers yielded good prediction accuracies with little error rates. Each of classifier was tested using its default parameters. To optimize the model accuracy, we then tested with different hyperparameters of each classifier using GridSearchCV from scikit learn library.

Amongst the classifiers used, the K neighbours (KNN) predicted the disease class with best accuracy rate of 97.1% for the subset of top gene samples with N value of 25 and number of neighbours as 3 out of the 2,3,4 values used. Results for other classifiers with the hyperparameters yielding best prediction are as follows.

Decision Tree classifier.
Splitting criterion: [entropy, Gini]
Best estimator: Gini
Genes: 15
Accuracy: 91.2%

Gaussian Naïve Bayes classifier
Accuracy: 70.8%
Genes: 30

Random Forest classifier
n_estimator (The number of trees in the forest.): 200,350
Splitting criterion: [entropy, Gini]
Best estimator: (n_estimator: 350, splitting criterion: Gini)
Genes: 30
Accuracy: 94.2%

MLP Classifier

No of neurons: 100,150,200

Hidden layer: 1, 2

Activation function: [‘logistic’, ‘relu’]

Learning rate: 0.01, 0.05

No of epoch: 150,250

Best parameter: Activation function: relu, Learning rate: 0.01, Neurons:150, hidden layer: 2

Accuracy: 92.6%

No of Genes: 30

The line graph of Fig.2 shows the error rate vs the number of genes, while fig.3 shows the accuracy of each classifier obtained.

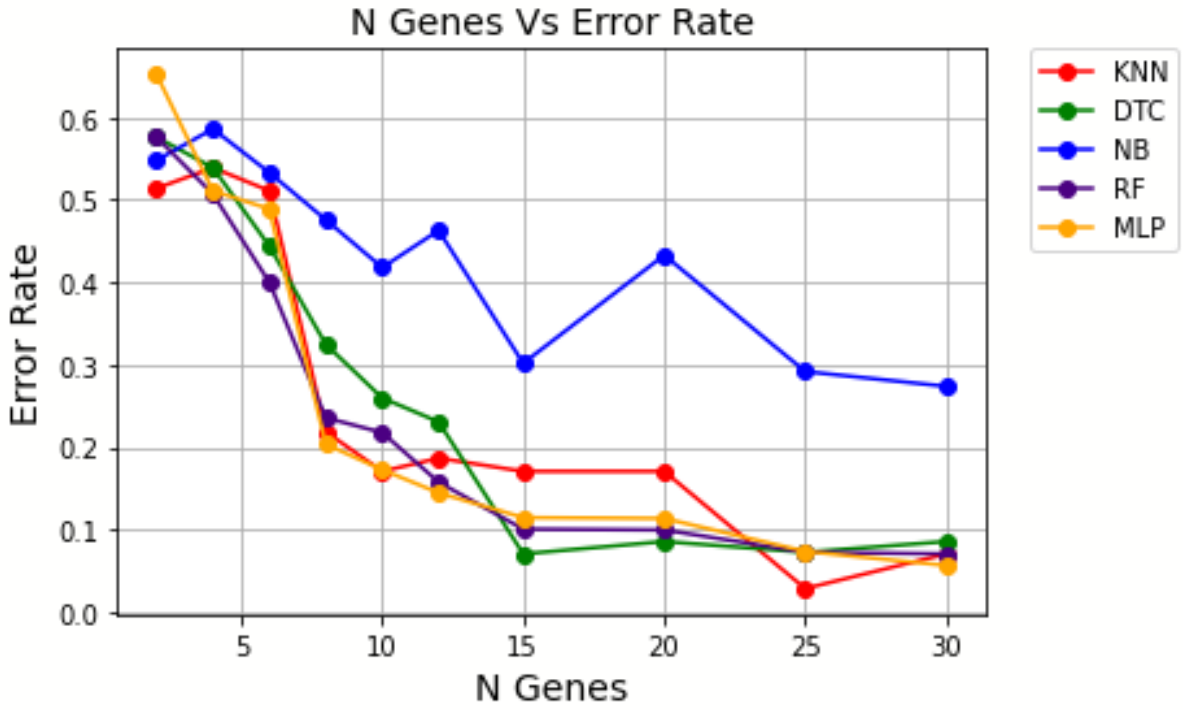

Fig.2 Error Rate vs N Genes

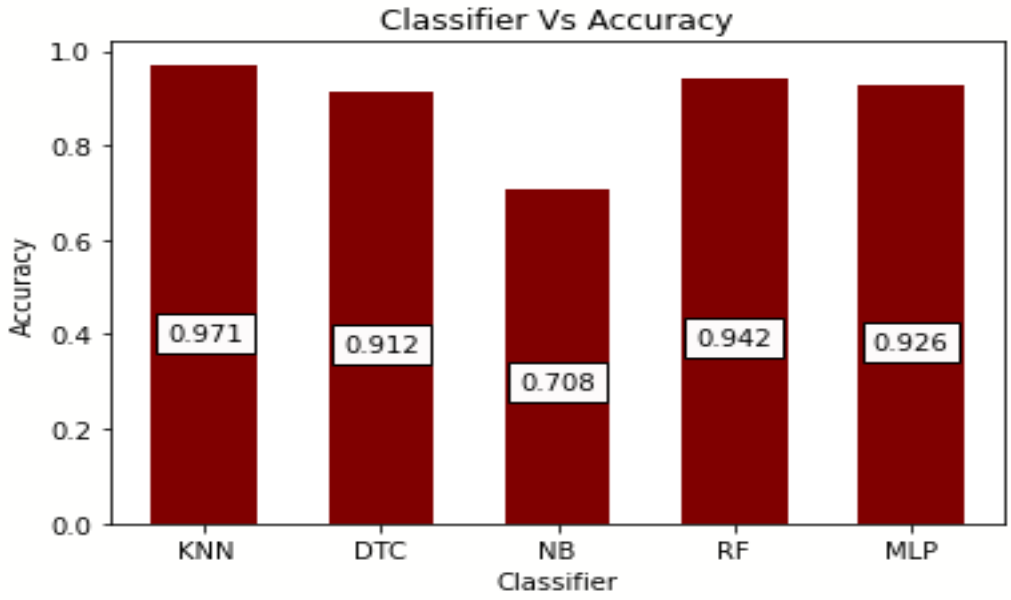

Fig.3 Accuracy vs classifiers used

For the unlabelled test dataset given for prediction, we obtained the below prediction result after being fed to the classifiers. This is shown in fig.4

```
Prediction of test dataset :
 ['MGL' 'EPD' 'MED' 'MED' 'EPD' 'MED' 'MED' 'MED' 'EPD' 'JPA' 'JPA' 'MED'
 'MED' 'MED' 'MED' 'MED' 'MGL' 'MED' 'RHB' 'RHB' 'RHB' 'MED' 'MED']
```

Fig.4 Predicted result of test dataset

B. Adaptive Boosting Technique.

An AdaBoost is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

To further see the performance prediction using boosting technique, we used AdaBoost with base estimator as Random Forest, Decision Tree and GaussianNB. The accuracy remained fairly constant for Decision Tree and gaussian NB (91.2% and 71% respectively) as a base estimator of AdaBoost. However, we could see a good performance boost for the decision tree classifier as the base estimator of AdaBoost with an accuracy of 98.61%. This was the best accuracy and prediction among all the classifiers.

AdaBoost's best hyperparameter that gave the optimum result (learning rate: 0.1, classifier: DecisionTree classifier, max_depth = 5)

Final output for the test set using the AdaBoost is given shown in fig.5

```
Test dataset predictions using Adaboost :
 ['MGL' 'EPD' 'MED' 'MED' 'EPD' 'MED' 'MED' 'MED' 'EPD' 'EPD' 'JPA' 'MED'
 'MED' 'MED' 'MED' 'MED' 'EPD' 'MED' 'MED' 'RHB' 'EPD' 'MED' 'MED']
```

Fig.5 Predicted result of test dataset applying AdaBoost

V. CONCLUSION AND FUTURE WORK

For this paper, machine learning classifiers for predicting disease using data collected from gene microarray has been developed and compared. The labelled training gene samples was used to train the classifiers and predicted on the unlabelled test sample. The K-nearest neighbour classifier (KNN) showed to be more efficient with the best accuracy rate. However, when subjected to a boosting technique (AdaBoost), the Decision Tree Classifier performed best with an accuracy of 98.61%, indicating better predictions can be reached with the use of a boosting technique. The computational time experienced with the MLP classifier was extremely long compared to other classifiers. It was a major challenge experience in the cause of this project. With the proposed classification model, prognosis can be carried out faster and more accurately, assisting medical practitioners in their work and most importantly improving patients' treatment.

As a future work, we recommend more datasets be applied in training the different classification model to improve the performance of the classifiers.

VI. REFERENCES

[1] Computer Vision Technology for Food Quality Evaluation. Cheng-Jin Du, Da-Wen Sun. Science Direct,2008

[2] The Digital Twin Paradigm for Smarter Systems and Environments: The Industry Use Case. S.Abirami, P. Chitra. Science Direct, 2020

[3] Machine Learning and its Application in Microscopic Image Analysis. F.Xing, L.Yang. Machine Learning and Medical Imaging,2016

[4] A Multilayer Perceptron-Based Medical Decision Support System for Heart Disease Diagnosis. Hongmei Yan, Yiangtao Jian, Jun Zheng, Chenglin Peng, Quinghui Li, 2006

[5] https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html?highlight=adaboost#sklearn.ensemble.AdaBoostClassifier