CS 590 Data Mining Project
Predict disease classes using genetic microarray data
Full names: Yuhao Zhang
Students id: 002313313

1. Abstract

The goal of this experiment is to develop a classifier to predict disease class base on gene dataset we have. Gene data will be cleaned on certain rules and filter by ANOVA test and sorted a set of numbers of genes from different classes. Performances of the classifier is measured by cross validate score on training data. Predictions are generated by the chosen classifier using adaptive boosting for further study.

2. Document each step.

Step 1. Data Cleaning

The data set contains Training dataset: pp5i_train.gr.csv, Training data classes: pp5i_train_class.txt, Test dataset: pp5i_test.gr.csv, with 7070 genes for 69 samples, and train class for each sample which is MED, RHB, MGL, EPD, JPA. Test data with 23 unlabeled samples and same genes. The gene data in both train and test dataset are limited to a value of 20 to 16000.

Step 2. Selecting top genes by class

The data genes with fold difference which is calculated by a ratio between maximum and minimum values (Max/Min) across samples less than 2 is removed. To extract the top genes from the dataset, ANOVA test is performed to Compute the ANOVA F-value for the provided sample. The rule of total variance, on which the ANOVA is based, divides the observed variance in a given variable into components owing to various causes of variation. ANOVA generalises the t-test beyond two means by offering a statistical test to determine if two or more population means are equal. In other words, the ANOVA is employed to determine if two or more means differ from one another. The dataset is sorted and filter the top 2,4,6,8,10,12,15,20,25, and 30 top genes from each class and remove duplicates genes for further experiment.

Step 3. Find the best classifier/best gene set combination

In this step, we will try each classifier in different N of gene and use cross_val_score from scikit-learn to calculate the error rate and try different parameter for better result.

- Naïve Bayes

Gaussian Naive Bayes (sklearn.naive_bayes.GaussianNB) form scikit-learn 1.1.1

Bayes Theorem can be used to calculate conditional probability. The Bayes Theorem is the foundation of Naive Bayes Classifiers. [1]Naive Bayes Classifiers are learned well in supervised learning environments. A modest amount of training data is required for naive Bayed classifiers in order to estimate the classification parameters. Naive Bayes Classifiers are easy to create and use, and they may be used in a variety of real-world scenarios.

$$P(A|B) = \frac{P(A \bigcap B)}{P(B)} = \frac{P(A) \cdot P(B|A)}{P(B)}$$

where:

$P(A) =$ The probability of A occurring

$P(B) =$ The probability of B occurring

$P(A|B) =$ The probability of A given B

$P(B|A) =$ The probability of B given A

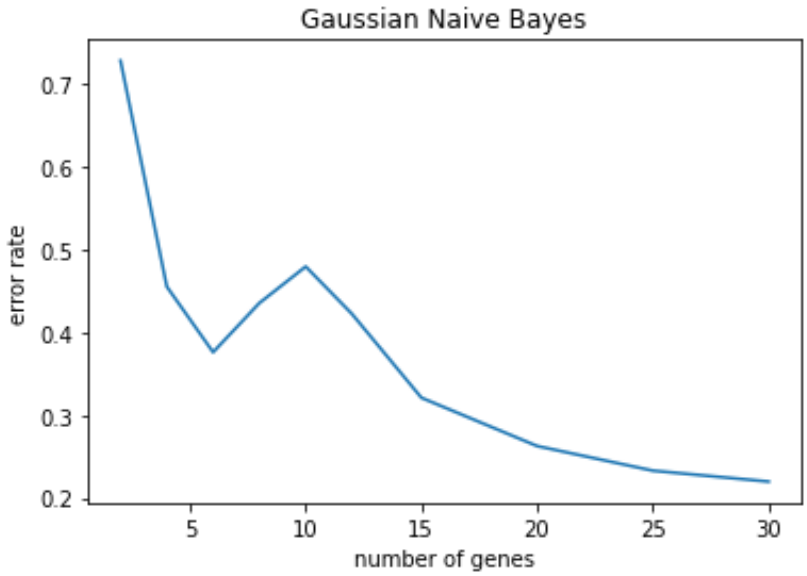
$P\left(A \bigcap B\right)) =$ The probability of both A and B occurring

Gaussian Naive Bayes the assumption that the continuous values associated with each class are distributed according to a normal distribution is frequently made when working with continuous data. The characteristics' probability is predicated by this

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

formula-

Parameter used: priors=None, var_smoothing=1e-09

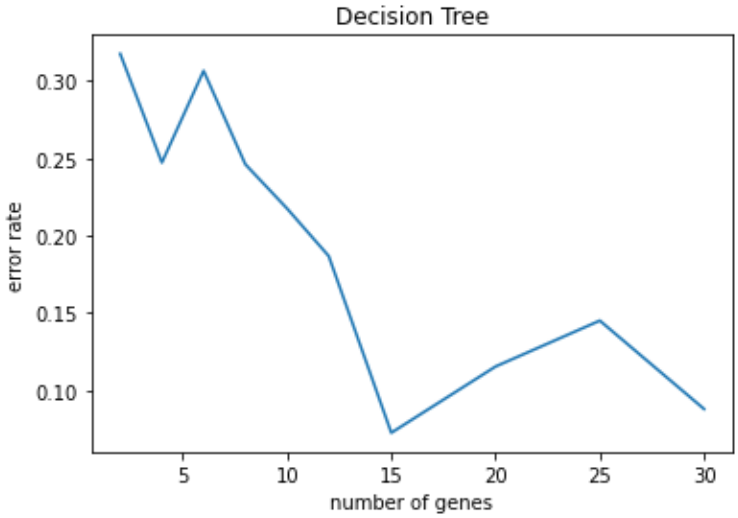


- Decision tree (J48 algorithm)

DecisionTreeClassifier form scikit-learn 1.1.1

The Univariate Decision Tree technique is implemented using the J48 algorithm, and its results are reviewed. [2] The multivariate technique is described as a linear machine approach that uses thermal perceptron rules and absolute error correction. A common method for supervised categorization is the decision tree, particularly when the output is read by a human.

Parameter used: DecisionTreeClassifier(class_weight='balanced',max_depth=30)



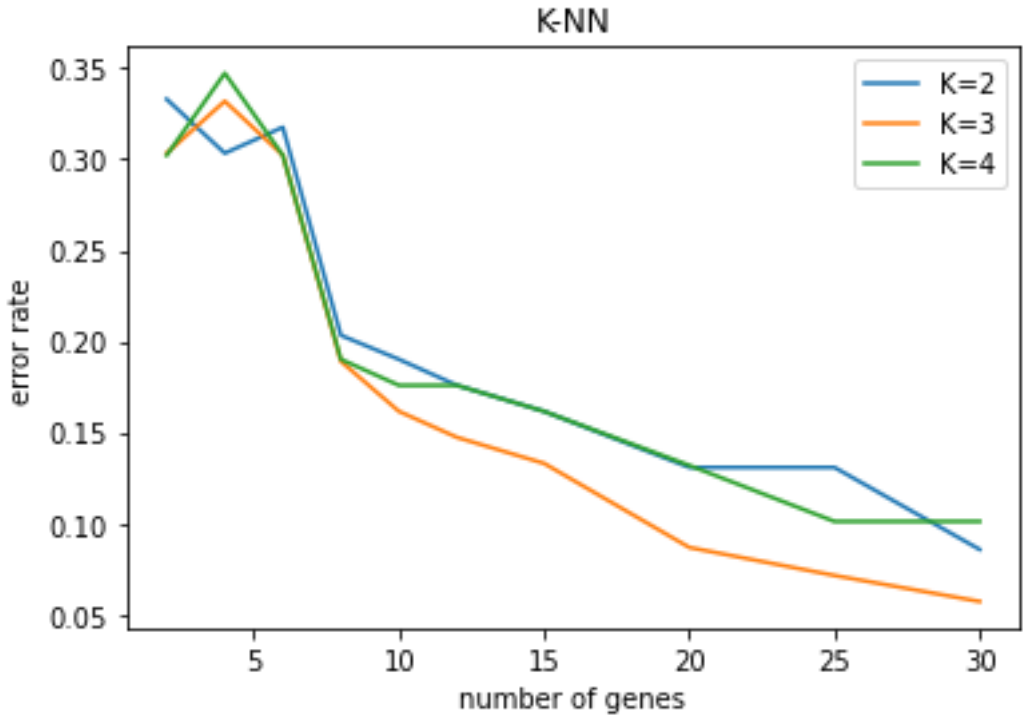
- K-NN (IBk algorithm)

Classifier implementing the k-nearest neighbors vote form scikit-learn 1.1.1
Following the discovery of the k nearest neighbors when utilizing the kNN technique, a number of approaches might be used to infer the category of a test text based on them. However, regardless of the various distributions of the classes, a fixed k value is often used across the board in these techniques. [3]Two of the often-used methods of this kind are Equations (1) and (2) below.

$$y(d_i) = \arg\max_k \sum_{x_j \in kNN} y(x_j, c_k) \quad \textbf{(1)}$$

$$y(d_i) = \arg\max_k \sum_{x_j \in kNN} Sim(d_i, x_j) y(x_j, c_k) \; . \quad \textbf{(2)}$$

Equation (1) predicts that the class with the greatest number of members among its k nearest neighbours will prevail, but equation (2) predicts that the class with the greatest sum of similarity will do so. The latter is considered superior to the former and is utilised more frequently.
Parameter used: (k=3)



- A neural network

Multi-layer Perceptron classifier form scikit-learn 1.1.1

A supervised learning system called a multi-layer perceptron (MLP) train on a dataset to learn a function, where is the number of input dimensions and is the number of output dimensions. [4] It can learn a non-linear function approximator for either classification or regression given a collection of features and a goal. There may be one or more non-linear layers, known as hidden layers, between the input layer and the output layer, which distinguishes it from logistic regression. A single hidden layer MLP with scalar output is shown in Figure below.

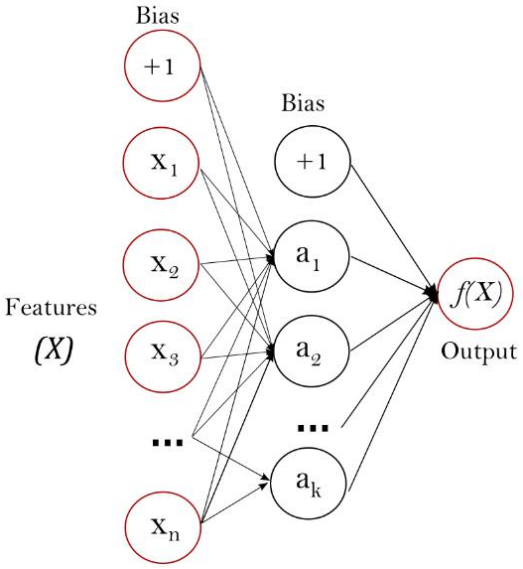
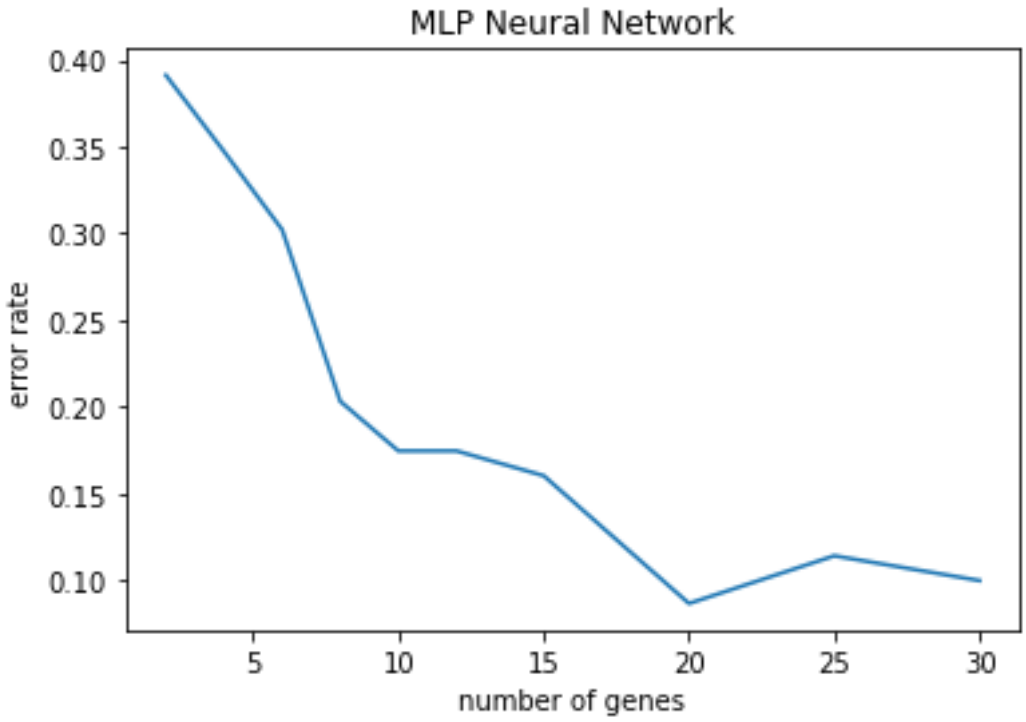


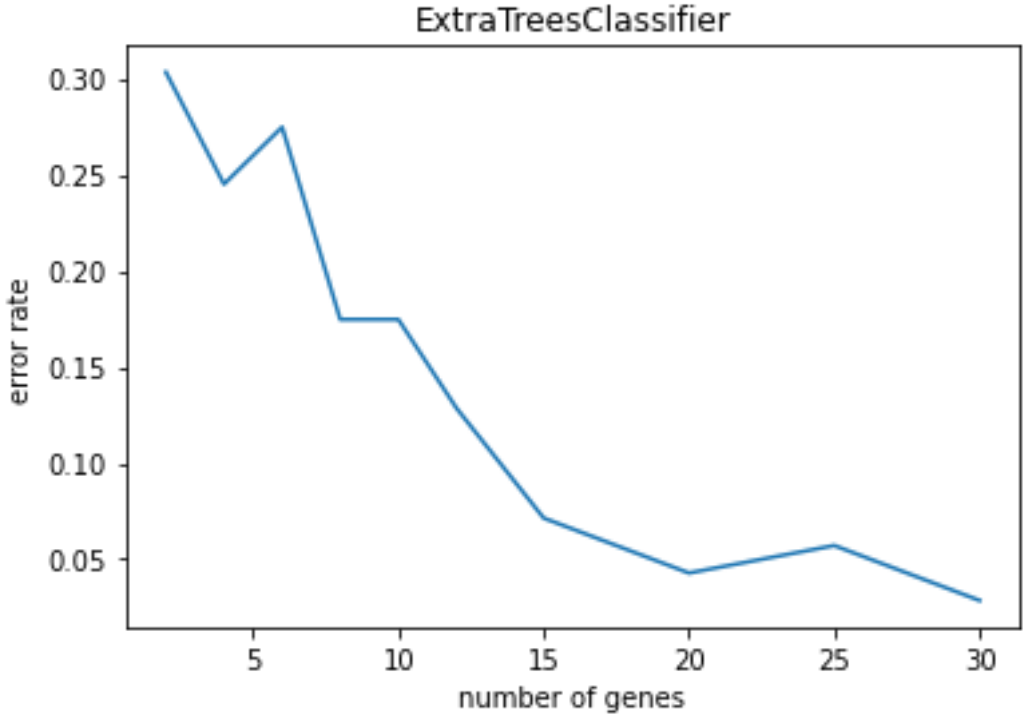
Figure 1 one hidden layer MLP

Parameter used: hidden_layer_sizes=100,activation='logistic'



- extra-trees classifier.

extra-trees classifier form scikit-learn 1.1.1
This class implements a meta estimator that employs averaging to increase prediction accuracy and reduce overfitting. [5] The meta estimator fits a number of randomised decision trees (also known as extra-trees) on different sub-samples of the dataset.
Parameter used: n_estimators=200



Step 4. Generate predictions for the test set
Result:(MGL,EPD,MED,MED,EPD,MED,MED,MED,EPD,JPA,JPA,MED,MED,ME

D,MED,MED,MGL,MED,RHB,RHB,RHB,MED,MED)
Step 5. Generate a prediction using Adaptive Boosting
AdaBoost classifier form scikit-learn 1.1.1
The AdaBoost classifier is a meta-estimator that starts by fitting a classifier on the initial dataset and then fits additional copies of the classifier on the same dataset with the weights of instances that were incorrectly classified being changed so that subsequent classifiers concentrate more on challenging cases.
Parameter used:
AdaBoostClassifier(base_estimator=ExtraTreesClassifier(n_estimators=200))
Result:(MGL,EPD,MED,MED,EPD,MED,MED,MED,EPD,JPA,JPA,MED,MED,MED,MED,MGL,MED,MED,RHB,RHB,MED,MED)

3. Conclusion

For all result from experiment on different classifier, ExtraTreesClassifier gain the best result of accuracy of 0.97 for top 30 of genes used. In this experiment we clean the dataset and extract the key information from the dataset, with different classifier in training data we got cross validated error to identify which classifier works the best for this dataset. Based on the chosen classifier we use adaptive boosting to enhance the performance and generate a prediction for test dataset. In future study for this project, we can try using more genes for the prediction or combine different classifier together and test if one classifier is more accurate for certain disease class.

REFERENCES

[1] Ontivero-Ortega, Marlis, et al. "Fast Gaussian Naïve Bayes for searchlight classification analysis." *Neuroimage* 163 (2017): 471-479.
[2]Bhargava, Neeraj, et al. "Decision tree analysis on j48 algorithm for data mining." *Proceedings of international journal of advanced research in computer science and software engineering* 3.6 (2013).
[3] Li, Baoli, Shiwen Yu, and Qin Lu. "An improved k-nearest neighbor algorithm for text categorization." *arXiv preprint cs/0306099* (2003).
[4] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." *the Journal of machine Learning research* 12 (2011): 2825-2830.
[5] Sharaff, Aakanksha, and Harshil Gupta. "Extra-tree classifier with metaheuristics approach for email classification." *Advances in computer communication and computational sciences*. Springer, Singapore, 2019. 189-197.