CS 6375
Project Report

Names of students in your group:

Xinyang Zhu
Ye Yao

Number of free late days used: **2**

Note: You are allowed a **total** of 4 free late days for the **entire semester**. You can use at most 2 for each assignment. After that, there will be a penalty of 10% for each late day.

Please list clearly all the sources/references that you have used in this assignment.

1. Introduction

By research, heart disease is most deadly illness to humankind worldwide. On the other hand, it's not easy for doctors to predict heart disease for patient in early stage of the disease. In this project, this paper utilize the one of the very popular heart disease dataset[1] from UCI which spreaded for years, and analyse the data from this very limited dataset, then predict the probability of one person has or doesn't have heart disease, based on this person's physical and health condition. This project use several machine learning approaches to create models, or predict directly from training data, and latter analyse the different result from these methods. This project is also a competition of DrivenData.org[2]. And finally, this project get a quite good result of prediction.

2. Problem Definition and Algorithm

2.1 Task Definition

The goal is to predict the binary class 'heart_desease_present', which represents whether or not a patient has heart disease.

- 0 represents no heart disease present
- 1 represents heart disease present

The training dataset contains 180 instances and the testing dataset contains 90 instances. There are no missing values. There are 13 features in the dataset. They are explained as below:

Feature name	**Data type**	**Description**	**Example**
slope_of_peak_exercise_st_segment	int	the slope of the peak exercise ST segment, an electrocardiography read out indicating quality of blood flow to the heart	2
thal	categorical	results of thallium stress test measuring blood flow to the heart, with possible values normal, fixed_defect, reversible_defect	normal
resting_blood_pressure	int	resting blood pressure	125
chest_pain_type	int	chest pain type (4 values)	3

num_major_vessels	int	number of major vessels (0-3) colored by flourosopy	0
fasting_blood_sugar_gt_120_mg_per_dl	binary	fasting blood sugar > 120 mg/dl	1
resting_ekg_results	int	resting electrocardiographic results (values 0,1,2)	2
serum_cholesterol_mg_per_dl	int	serum cholestoral in mg/dl	245
oldpeak_eq_st_depression	float	oldpeak = ST depression induced by exercise relative to rest, a measure of abnormality in electrocardiograms	2.4
sex	binary	0: female, 1: male	1
age	int	age in years	51
max_heart_rate_achieved	int	maximum heart rate achieved (beats per minute)	166
exercise_induced_angina	binary	exercise-induced chest pain (0: False, 1: True)	0

The performance metric is logarithmic loss, or log loss, which uses the probabilities of class predictions and the true class labels to generate a number that is closer to zero for better models, and exactly zero for a perfect model.

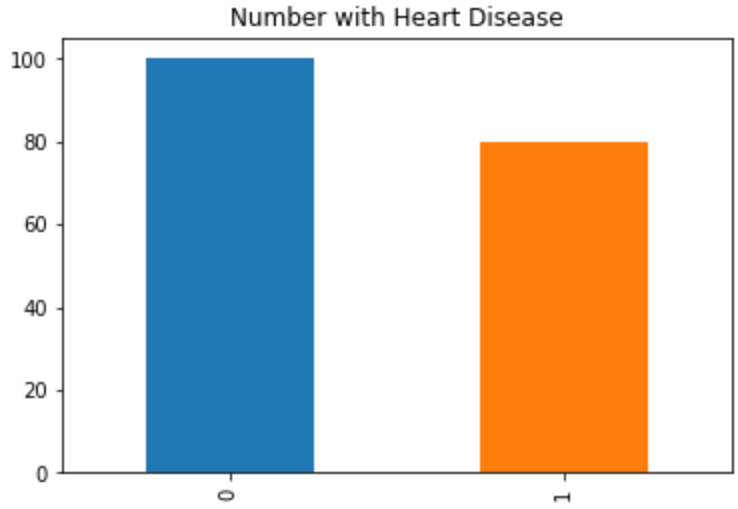
The output for one set of prediction result has two columns with the “patient_id” and “heart_disease_present”. The “heart_disease_present” values are the probabilities that a patient has heart disease (not the binary label).

Here is an example of the output:

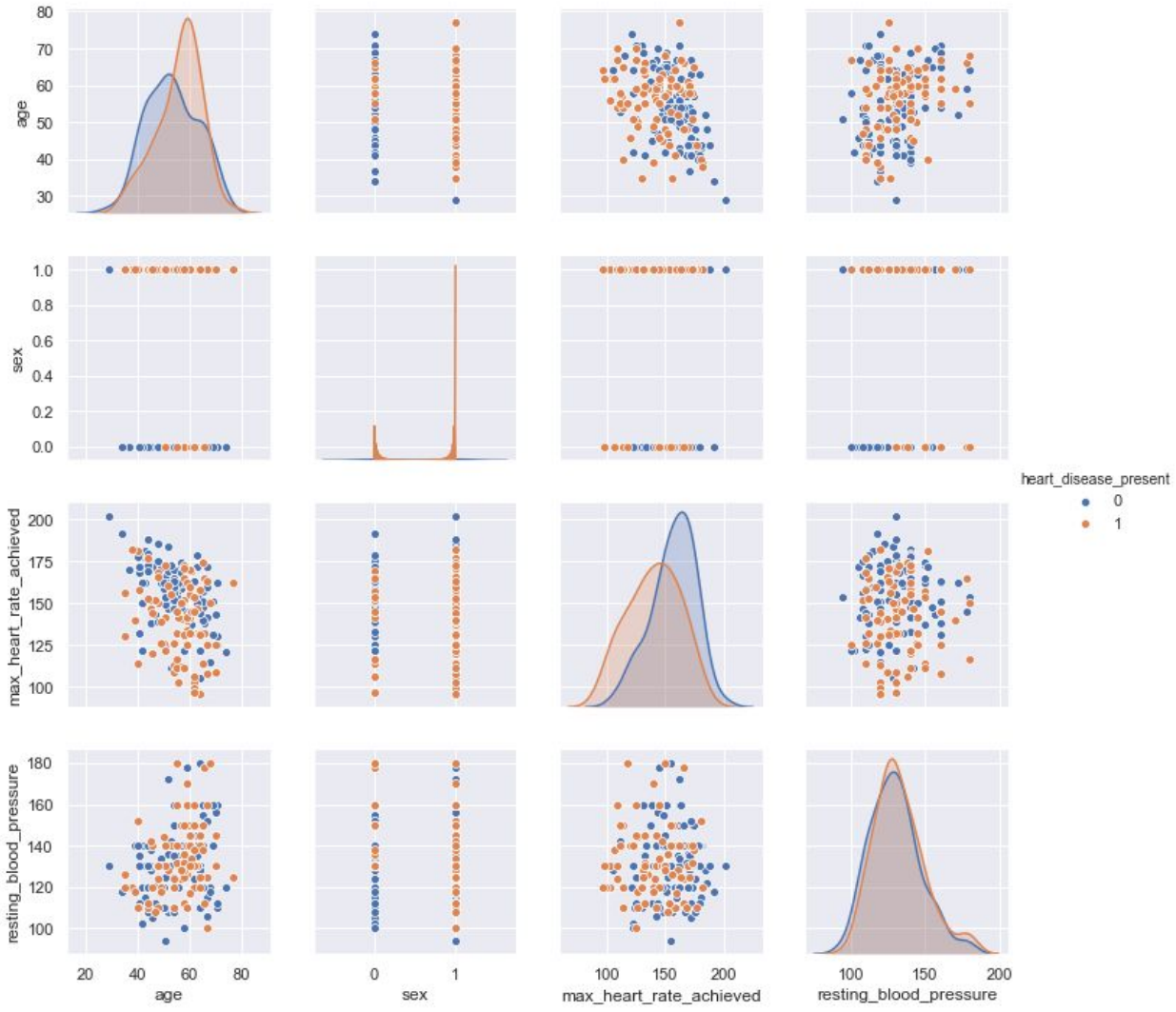
patient_id	**heart_disease_present**
lj5zrq	0.5
i79t3w	0.5
tgpy9u	0.5

46dlca	0.5
c7olxr	0.5
...	...
v5fsfs	0.5
5bbknr	0.5
r4hsar	0.5

In the diagram below, it's been shown that the amount of both class is balanced, thus, no equalization process has been taken here.



Then use the Seaborn to show the correlation for different features.



2.2 Algorithm Definition

One-hot encoding: transforming each categorical feature with n possible values into n binary features, with one of them 1, and all others 0, where n is the number of categories. [3]

Normalization: the process of scaling individual samples to have unit norm. [3]

Overfitting: the production of an analysis that corresponds too closely or exactly to the training data, and may therefore fail to fit the testing data. [3]

Cross-validation: holding out part of the available data as a test set when performing a (supervised) machine learning experiment. [3] It is a common way to avoid overfitting.

Grid search: exhaustive search over specified parameter values for an estimator. [3]

Log loss: also called logistic regression loss or cross-entropy loss, is defined on probability estimates. It is commonly used in (multinomial) logistic regression and neural networks, as well as in some variants of expectation-maximization, and can be used to evaluate the probability outputs (predict_proba) of a classifier instead of its discrete predictions. For binary classification with a true label of 0 or 1 and a probability estimate p = Prob(y=1) (as in this problem), the log loss per sample is the negative log-likelihood of the classifier given the true label:

$$L_{log}(y, p) = -(y \log(p) + (1-y) \log(1-p)) \quad [3]$$

Decision Trees (DT): a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. [3]

Logistic regression: a linear model for classification rather than regression. Logistic regression is also known in the literature as logit regression, maximum-entropy classification (MaxEnt) or the log-linear classifier. In this model, the probabilities describing the possible outcomes of a single trial are modeled using a logistic function. [3]

Multi-layer Perceptron (MLP): is a supervised learning algorithm that learns a function f: R^m -> R^o by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output. Given a set of features X = x1, x2, …, xm and a target y, it can learn a non-linear function approximator for either classification or regression. [3]

Support Vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. [4]

Naive Bayes classifiers: a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features. [4]

Gaussian Naive Bayes classifier: a Naive Bayes classifier with the likelihood of the features assumed to be Gaussian. [3]

K nearest neighbor classification: Classification is computed from a simple majority vote of the k nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the nearest neighbors of the point. [3]

Bagging classifier: an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction. [3]

Random forest: 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. [3]

AdaBoost classifier: 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. [3]

Gradient Boosting: building an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage “n_classes” regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function, where “n_classes” is the number of classes. [3]

XGBoost: an optimized distributed gradient boosting library that implements machine learning algorithms under the Gradient Boosting framework. [5]

Receiver operating characteristic (ROC): a graphical plot which illustrates the performance of a binary classifier system as its discrimination threshold is varied. It is created by plotting the fraction of true positives out of the positives (TPR = true positive rate) vs. the fraction of false positives out of the negatives (FPR = false positive rate), at various threshold settings. [4]

3. Experimental Evaluation

3.1 Methodology

3.1.1 Python and libraries

The project is programmed in Python language. The libraries used are numpy, pandas, sklearn, and matplotlib. Numpy and pandas are used for data processing purpose. Sklearn is used for implementing machine learning algorithms. Matplotlib is used for plotting ROC curve.

3.1.2 Data representations

The input and output datasets are in the form of comma-separated values. The data processed in the program has various types, including Python list, numpy ndarray, pandas DataFrame, 2D plot, and basic Python data types.

3.1.3 Encoding, normalizing, pre-processing

The raw data are encoded, normalized, and pre-processed before training and testing. OneHotEncoder, Normalizer, and Pipeline in sklearn are used here.

The program use one-hot technique to encode non-numerical feature values. As a result, the feature “thal” which is categorical is encoded into one-hot representation and other features which are numerical are not encoded. This is achieved by using sklearn’s ColumnTransformer which runs models only on selected columns.

Normalization is performed over the features values because scaling inputs to unit norms is a common operation for text classification or clustering for instance.

Pipeline is utilized to apply two transformers, i.e. ColumnTransformer and Normalizer, in our case. It fits over the test data and will transform both the test data and the train data. An ending estimator is absent in this pipeline. Instead, we will do a grid search cross validation with various machine learning estimators in a later step.

In the pre-processing, the raw dataset is splitted into train data and test data by the ratio of 0.85 to 0.15. (This raw dataset does not include the “test” dataset that will be used for competition evaluation and the “test” dataset will be called “evaluation data/dataset” in this report.)

3.1.4 Models

Eleven machine learning classification models are trained and tested in the project. They are:

- Decision tree classifier
- Logistic regression
- Neural network or multi-layer perceptron classifier
- SVC
- Gaussian Naive Bayes classifier
- K-neighbors classifier
- Bagging classifier
- Random forest classifier
- AdaBoost classifier
- Gradient boosting classifier
- XGBoost classifier

The training and testing process is based on sklearn’s grid search cross validation (GridSearchCV). For most classifiers, different parameter choices are provided as a parameter grid. GridSearchCV model fits and predicts on each classifier through different combinations of parameters exhaustively and find the best set of parameters. With each classifier and each

parameter combination, GridSearchCV model also utilize the cross validation algorithm, in which way overfitting is largely avoided.

The best set of parameters for each classifier is then selected to print along with the performance metrics. The performance metrics is the log loss in this problem.

ROC curves are drawn for the best prediction for each classifier and ROC curve areas are printed.

A small log loss and/or a large ROC curve area indicates a better prediction.

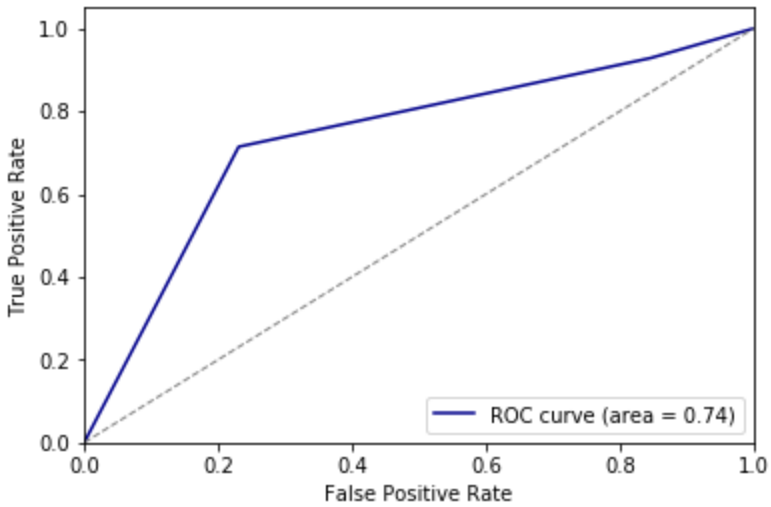
3.1.5 Evaluation

The GridSearchCV model makes a prediction on the evaluation dataset with the best set of parameters for each classifiers. Then the program outputs the results in the form of comma-separated values.

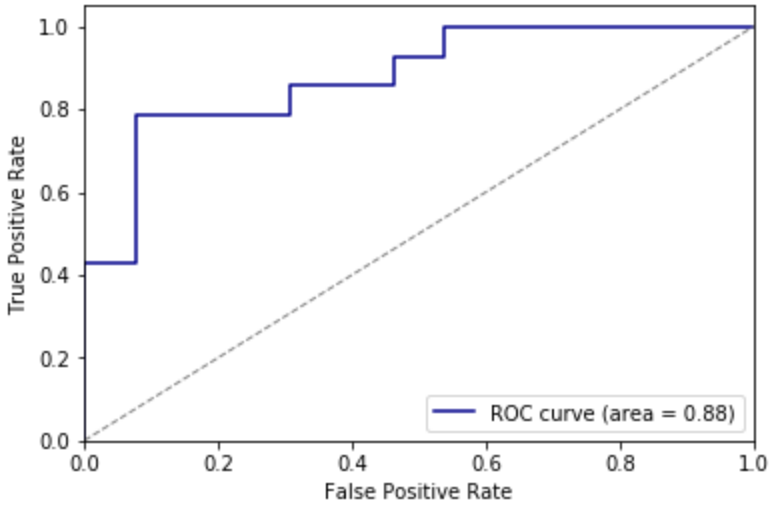
Based on the performance metrics and ROC curves in step 3.1.4, selected outputs are submitted to the DrivenData competition for evaluation.

3.2 Results

0 DT
DT 1.80 {'max_depth': 1, 'max_leaf_nodes': 4, 'min_samples_leaf': 3, 'min_samples_split': 2}

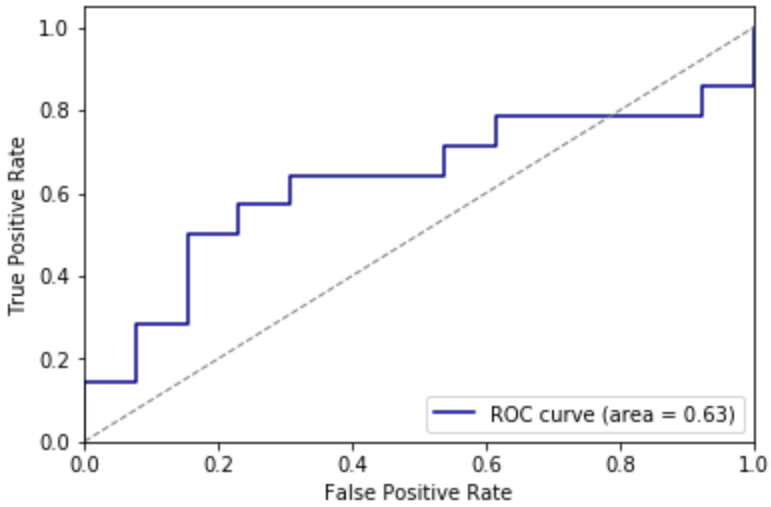


1 MLP
MLP 0.43 {'activation': 'tanh', 'early_stopping': False, 'hidden_layer_sizes': (40, 20, 10), 'learning_rate': 'invscaling', 'max_iter': 2000}



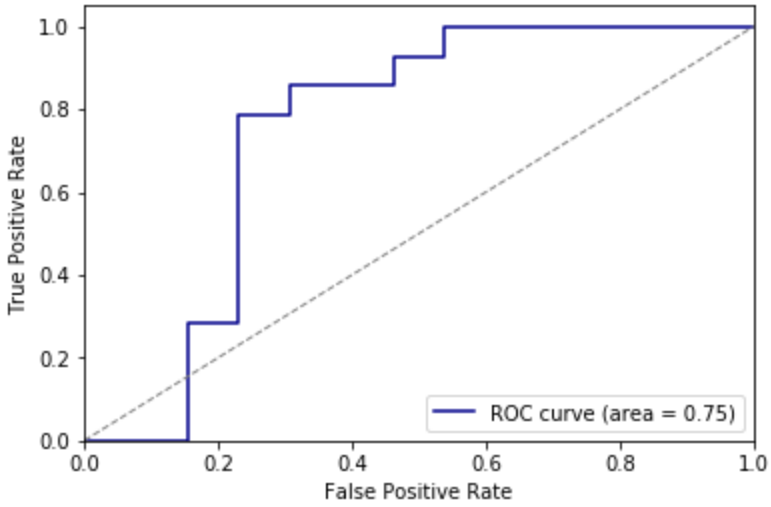
2 SVC

SVC 0.69 {'coef0': 0.0, 'gamma': 'scale', 'kernel': 'sigmoid', 'probability': True, 'shrinking': False}



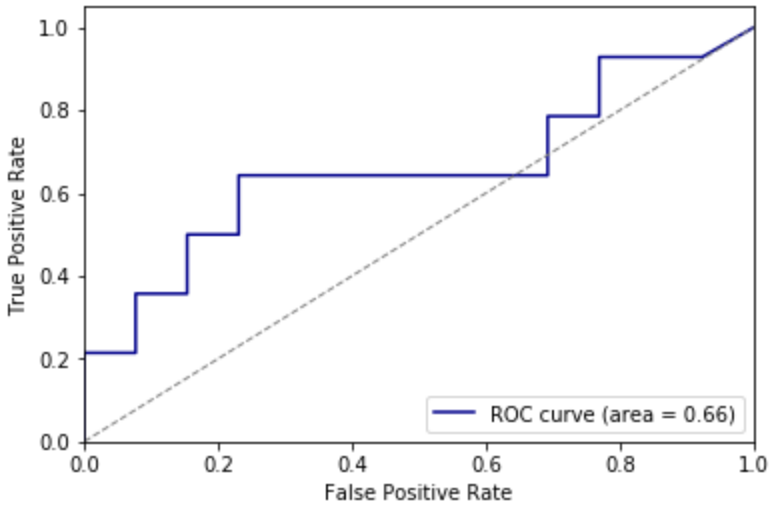
3 NB

NB 2.01 {}



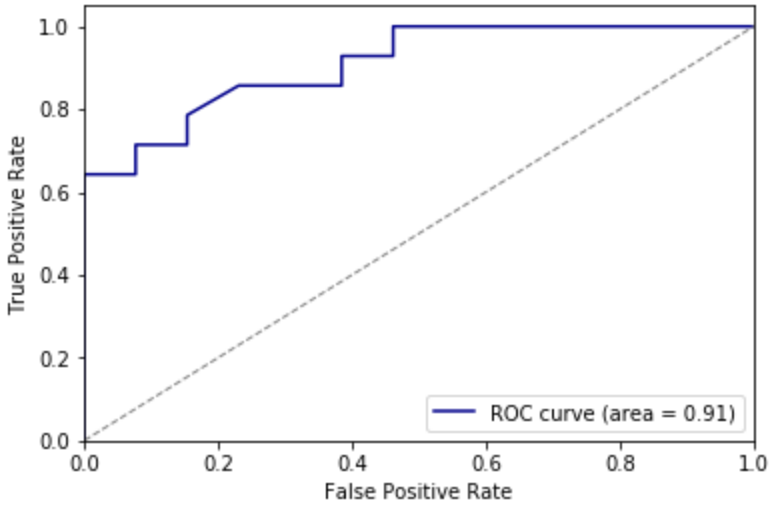
4 KNN

KNN 1.90 {'algorithm': 'auto', 'n_neighbors': 7, 'p': 1, 'weights': 'distance'}



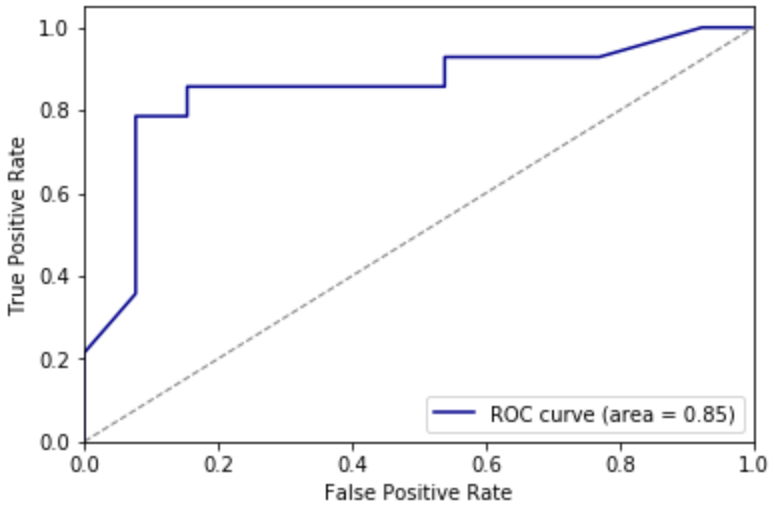
5 Bagging

Bagging 0.43 {'max_features': 1.0, 'max_samples': 0.5, 'n_estimators': 200}



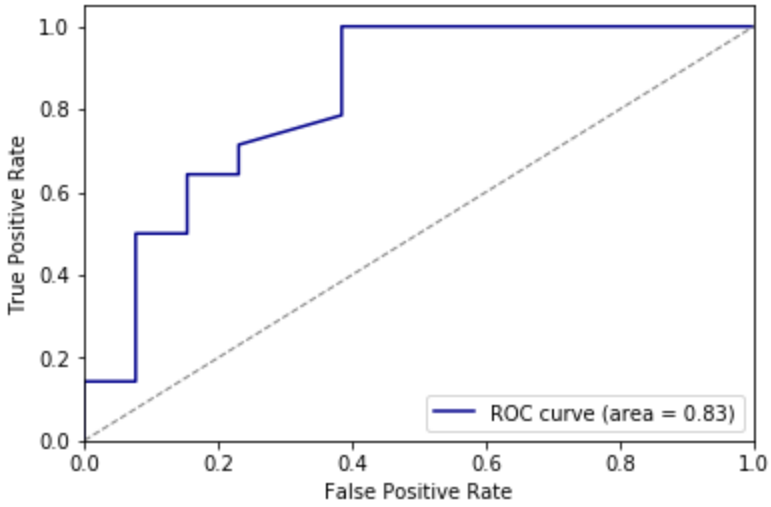
6 RF

RF 0.48 {'criterion': 'entropy', 'max_depth': 10, 'max_features': 'log2', 'n_estimators': 20}



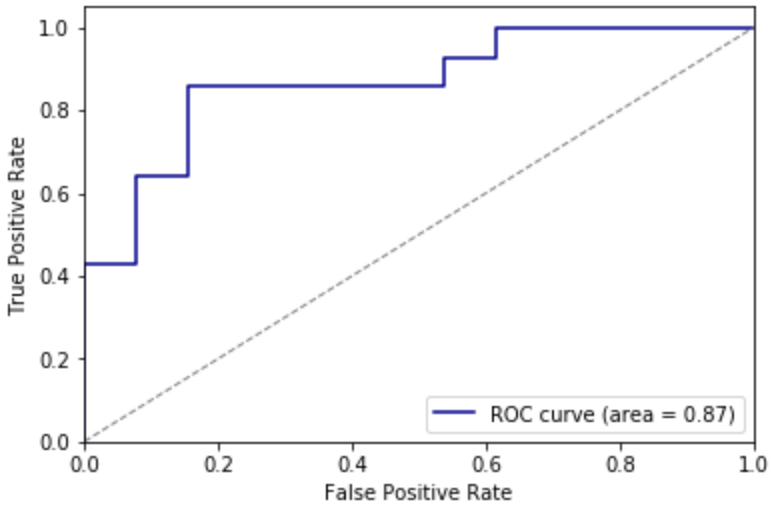
7 AdaBoost

AdaBoost 0.60 {'algorithm': 'SAMME.R', 'learning_rate': 0.1, 'n_estimators': 10}



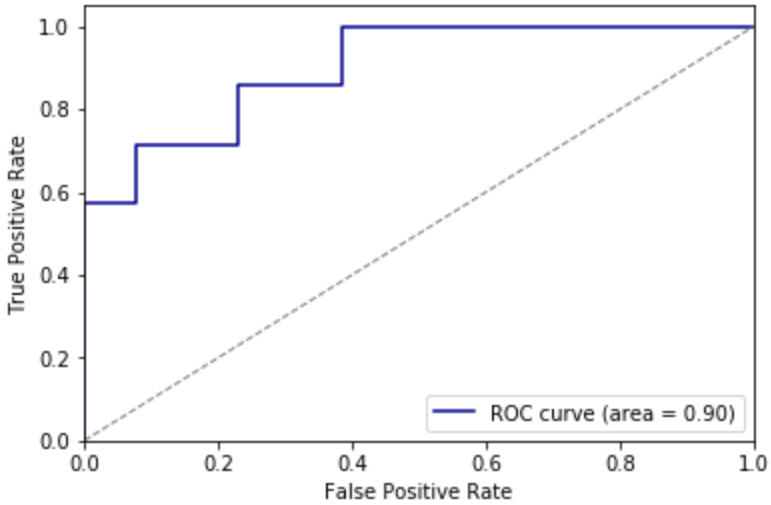
8 GB

GB 0.47 {'learning_rate': 0.05, 'max_depth': 5, 'max_features': 'log2', 'n_estimators': 50}



9 XGB

XGB 0.42 {'booster': 'gbtree', 'learning_rate': 1, 'min_child_weight': 5, 'n_estimators': 10}



3.3 Discussion

Algorithm	log_loss	ROC area	parameter
Decision Tree	1.80	0.74	'max_depth': 1, 'max_leaf_nodes': 4, 'min_samples_leaf': 3, 'min_samples_split': 2
Multi-layer Perceptron (Neural Network)	0.43	0.88	'activation': 'tanh', 'early_stopping': False, 'hidden_layer_sizes': (40, 20, 10), 'learning_rate': 'invscaling', 'max_iter': 2000
SVC	0.69	0.63	'coef0': 0.0, 'gamma': 'scale', 'kernel': 'sigmoid', 'probability': True, 'shrinking': False
NB	2.01	0.75	
KNN	1.90	0.66	'algorithm': 'auto', 'n_neighbors': 7, 'p': 1, 'weights': 'distance'
Bagging	0.43	0.91	'max_features': 1.0, 'max_samples': 0.5, 'n_estimators': 200
Random Forest	0.48	0.85	'criterion': 'entropy', 'max_depth': 10, 'max_features': 'log2', 'n_estimators': 20
AdaBoost	0.60	0.83	'algorithm': 'SAMME.R', 'learning_rate': 0.1, 'n_estimators': 10
Gaussian Bayes	0.47	0.87	'learning_rate': 0.05, 'max_depth': 5, 'max_features': 'log2', 'n_estimators': 50
XGB	0.42	0.90	'booster': 'gbtree', 'learning_rate': 1, 'min_child_weight': 5, 'n_estimators': 10

By the result shown above, basically, ensemble classifiers works better than fundamental classifiers like Decision Tree, Naive Bayes and KNN. Neural Network can work great with well design layer and node structure, and work faster than ensemble classifiers.

4. Related Work

Using machine learning to increase the diagnostic accuracy of disease like heart disease has been researching and developing from long time ago. In 1997, MatjažKukar et al.[6] used features like diagnostic levels such as ECG at rest and at exercise, symptoms of disease, myocardial scintigraphy, and applied to various learning algorithms and achieved some great result, and they also used ROC curve to analyzed their result. In 2008, Sellappan Palaniappan and Rafiah Awang[7] used data mining techniques to design a heart disease prediction system. They utilized Decision Tree, Naive Bayes and Neural Network methods to get several good result of prediction. Their system can also answer some complex problems proposed by users. In 2010, Asha Rajkumar M.phil and G.Sophia Reena[8] used Tanagra tool to data mining the heart disease data and used supervised learning and cross validation to create model.

5. Future Work

One of the major problem in this project is the insufficient amount of data. To avoid this problem, we have implemented the cross validation method, and use bagging and random forest classifier to bootstrap data. However, it's still the top of the shortcoming. In the future, we will focusing on the preprocessing of the data, and train a better model based on a very good understanding of the dataset itself.

6. Conclusion

This project involves a warm-up problem in the machine learning world. It uses several major classifiers to do the one of very important and popular area, classification problem. And as a online competition, we also get a fair good result. By the calculation of log-loss, we get a result of 0.37 which ranks 150^{th} out of all results from 864 competition groups.

BEST	CURRENT RANK	# COMPETITORS
0.37412	150	864

Although it may not encourage or have a impact to future research, it do encourage the project team to learn machine learning better, not only use API provided by library like scikit-learn, but better understand and dig into the theory of machine learning algorithms and statistical theory.

Bibliography

[1] Aha, D., and Dennis Kibler. "Instance-based prediction of heart-disease presence with the Cleveland database." University of California 3.1 (1988): 3-2.

[2] Warm Up: Machine Learning with a Heart hosted by DrivenData
https://www.drivendata.org/competitions/54/machine-learning-with-a-heart/page/107/

[3] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

[4] Wikipedia
https://www.wikipedia.org

[5] XGBoost Documentation
https://xgboost.readthedocs.io/en/latest/#

[6] Kukar, M., Kononenko, I., Grošelj, C., Kralj, K., & Fettich, J. (n.d.). Analysing and improving the diagnosis of ischaemic heart disease with machine learning. *Artificial Intelligence in Medicine.*, *16*(1), 25–50. https://doi.org/10.1016/S0933-3657(98)00063-3

[7] S. Palaniappan and R. Awang, "Intelligent heart disease prediction system using data mining techniques," *2008 IEEE/ACS International Conference on Computer Systems and Applications*, Doha, 2008, pp. 108-115.

[8] RAJKUMAR M.PHIL (COMPUTER SCIENCE), G.SOPHIA REENA (HOD OF BCA DEPARTMENT), Asha. Diagnosis of Heart Disease using Datamining Algorithm. Global Journal of Computer Science and Technology, [S.l.], sep. 2010. ISSN 0975-4172.