Machine Learning Engineer Nanodegree

Capstone Project

Muhammad Nagy
Jun 1st, 2018

I. Definition

Heart disease is the number one killer according to World Health Organization (WHO) statistics [1]. Millions of people die every year because of heart disease and large population of people suffers from heart disease. Prediction of heart disease early plays a crucial role for the treatment. If heart disease could be predicted before, lots of patient deaths would be prevented and a more accurate and efficient treatment way could be provided. In this work we are trying to early diagnose the heart disease using machine learning methods.

Problem Statement

This is a classification problem. I measure the presence of heart disease in the patient based on the used dataset. Various machine learning approaches are used for addressing heart disease diagnosis issue like found in [2], [3], [4], [5], [6], [7], [8]. And there are more papers in this field.

We will use empirical approach to solve this problem starting with examine data pre-process methods like data transformation and scaling based on the nature of data then we will compare various machine learning algorithms based on the metrics below to evaluate the best model to use.

However, the data has multiple categories for presence of disease but, I will use binary classification to distinguishing absence (value 0) from presence of heart disease (values 1, 2, 3, and 4). The result of prediction should be 0/1 for absence/presence of heart disease.

Metrics

The performance of the proposed system was computed by different metrics like accuracy, precision and recall.

Accuracy is computed dividing number of predictions which are correct by number of all predictions. The obtained result is multiplied by 100 to get value as percentage.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where TP, TN, FP and FN demonstrate in order of the number of True Positives, True Negatives, False Positives and False Negatives. TP demonstrates the number of instances which are sick and diagnosed

accurately. FP demonstrates the number of instances which are healthy and diagnosed wrongly as they are sick. FN demonstrates the number of instances which are sick, but the instances are diagnosed wrongly. TN contains several instances which are healthy, and the instances are diagnosed accurately

Precision denotes the ratio of the instances that are predicted as having heart disease and they have heart disease.

$$Precision = \frac{TP}{TP + FP}$$

Recall denotes the proportion of the instances that are have heart disease are predicted as having heart disease.

$$Recall = \frac{TP}{TP + FN}$$

F_1 score is a measure of a test's accuracy.

$$F_1 = \frac{2\,.\,Precision\,.\;Recall}{Precision + Recall}$$

So, instead of using precision and recall in the output we could use F_1 score.

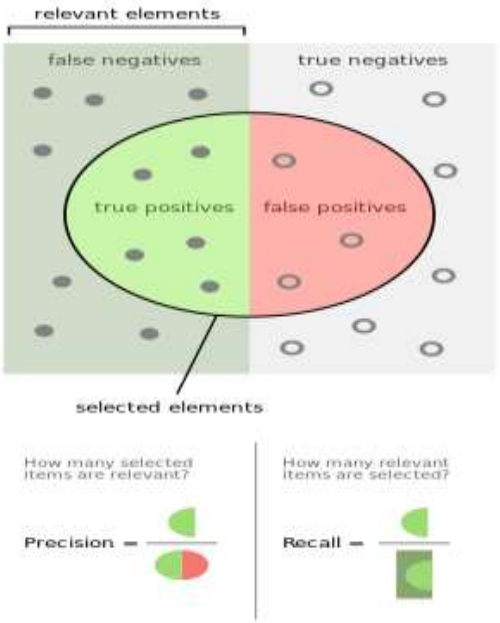

Figure 1 Precision and Recall

II. Analysis

Data Exploration

We are using the heart disease data set available from the [UC Irvine Machine Learning Repository](). This data set dates from 1988 and consists of four databases: Cleveland (303 instances), Hungary (294), Switzerland (123), and Long Beach VA (200). Each database provides 76 attributes, including the predicted attribute. There are many missing attribute values. In addition, the Cleveland data set became corrupted after the loss of a computing node, and the surviving data set contains only 14 attributes per instance. Counting only instances with non-missing values for these 14 attributes, the total for all four databases is 299 instances (297 from Cleveland alone). This is the data set I will be using, and for simplicity I will be referring to it as the Cleveland data set.

#	Attribute	Description	Type
1	age	Age in years	int
2	sex	Female or male	bin
3	cp	Chest pain type (typical angina, atypical angina, non-angina, or asymptomatic angina)	cat
4	trestbps	Resting blood pressure (mm Hg)	con
5	chol	Serum cholesterol (mg/dl)	con
6	fbs	Fasting blood sugar (< 120 mg/dl or > 120 mg/dl)	bin
7	restecg	Resting electrocardiography results (normal, ST-T wave abnormality, or left ventricular hypertrophy)	cat

8	thalach	Max. heart rate achieved during thalium stress test	con
9	exang	Exercise induced angina (yes or no)	bin
10	oldpeak	ST depression induced by exercise relative to rest	con
11	slope	Slope of peak exercise ST segment (upsloping, flat, or downsloping)	cat
12	ca	Number of major vessels colored by fluoroscopy	int
13	thal	Thalium stress test result (normal, fixed defect, or reversible defect)	cat
14	num	Heart disease status: number of major vessels with >50% narrowing (0,1,2,3, or 4)	int

The 14th column will be used as classification result to distinguishing absence (value 0) from presence of heart disease (values 1, 2, 3, and 4). We need to transform the column values to only two values "absence (value 0)" and "disease (values 1)".

Here we could find sample of data

	age	sex	cp	restbp	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num
0	63.0	1.0	1.0	145.0	233.0	1.0	2.0	150.0	0.0	2.3	3.0	0.0	6.0	0.0
1	67.0	1.0	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	2.0	3.0	3.0	2.0
2	67.0	1.0	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	2.0	2.0	7.0	1.0
3	37.0	1.0	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	3.0	0.0	3.0	0.0
4	41.0	0.0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	1.0	0.0	3.0	0.0

Dataset statistics are as follows:

```
Total number of records: 299
Number of infected patients: 139
Number of healthy patients: 160
Percentage of infected patients: 46.49%
```

	age	sex	cp	restbp	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
count	299.000000	299.00000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000
mean	54.521739	0.67893	3.163880	131.715719	246.785953	0.143813	0.989967	149.327759	0.331104	1.058528	1.605351	0.672241	4.745819
std	9.030264	0.46767	0.964069	17.747751	52.532582	0.351488	0.994903	23.121062	0.471399	1.162769	0.616962	0.937438	1.940977
min	29.000000	0.00000	1.000000	94.000000	100.000000	0.000000	0.000000	71.000000	0.000000	0.000000	1.000000	0.000000	3.000000
25%	48.000000	0.00000	3.000000	120.000000	211.000000	0.000000	0.000000	132.500000	0.000000	0.000000	1.000000	0.000000	3.000000
50%	56.000000	1.00000	3.000000	130.000000	242.000000	0.000000	1.000000	152.000000	0.000000	0.800000	2.000000	0.000000	3.000000
75%	61.000000	1.00000	4.000000	140.000000	275.500000	0.000000	2.000000	165.500000	1.000000	1.600000	2.000000	1.000000	7.000000
max	77.000000	1.00000	4.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	3.000000	3.000000	7.000000

Exploratory Visualization

To understand the nature of data, a visualization below shows each feature plot with a comparison between having heart disease and have no disease. We could observe easily that the percentage on ill

and healthy are approximately equal in the dataset which match the percentage we found before (46.49% infected). It looks like it's hard to find a pattern for diagnosis from the chart only.

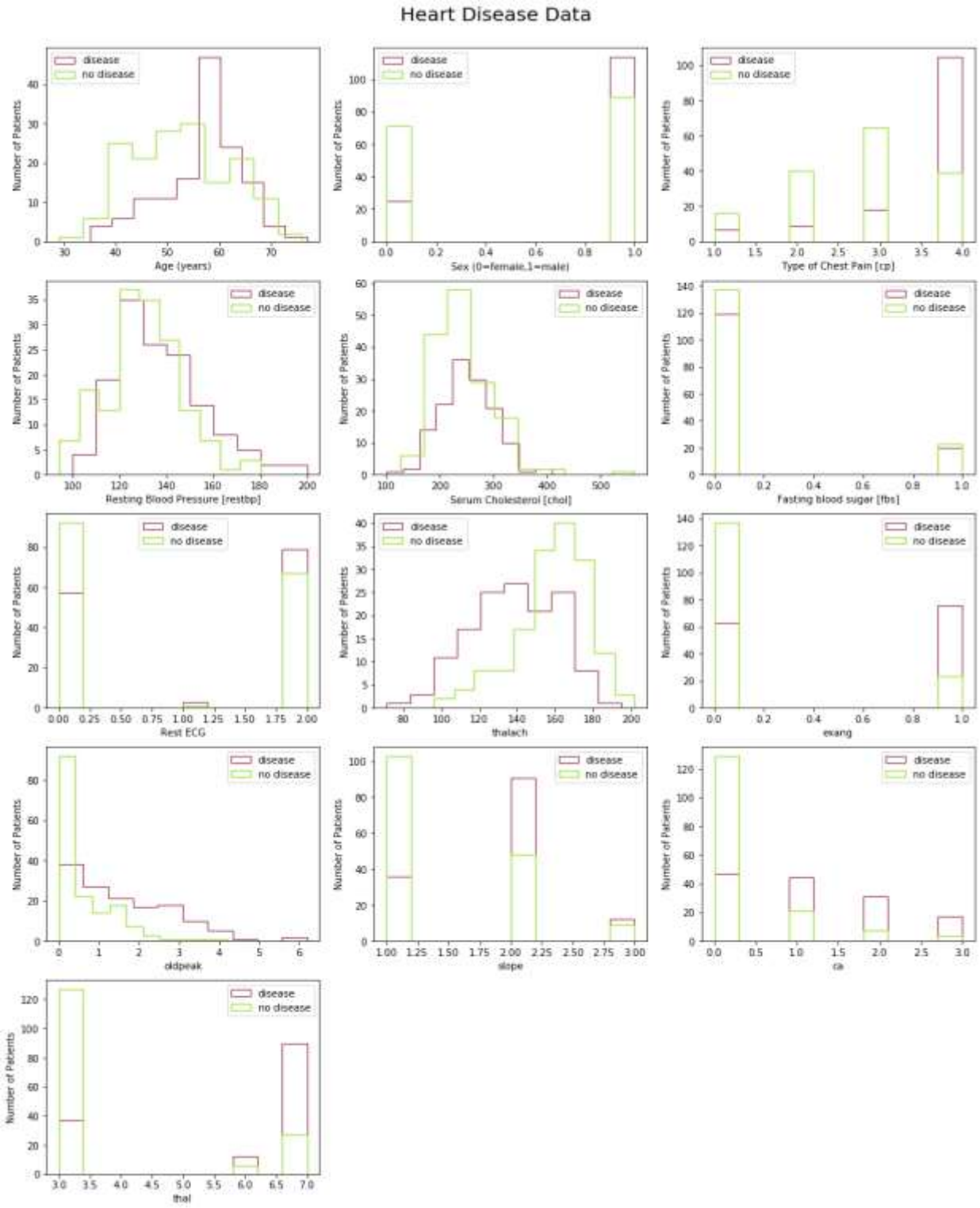

Figure 2 Individual features visualization

Algorithms and Techniques

Our approach is to try many algorithms and compare them to find the best model.

Algorithms used in the project are K-Nearest Neighbor, Random Forest, Gaussian Naïve Bias (GaussianNB), Gradient Boosting, Support Vector Machine(SVM), AdaBoost, stacking ensemble (i.e. Will stack all the previous methods and use logistic regression as meta classifier), and Multilayer Perceptron(MLP).

Then we will use Grid Search method to optimize algorithm's parameters other than MLP.

The parameter used to optimize for each algorithm are as follows:

Algorithm	Parameters
K-Nearest Neighbor	algorithm, n_neighbors, weight
Random Forest	criterion, n_estimators
GaussianNB	N/A
Gradient Boosting	n_estimators, max_depth
SVM	kernerl, C, gamma, probability
AdaBoost	algorithm, learning_rate
Logistic Regression	C
Stacking ensemble	classifiers, use_probas, average_probas, meta_classifier
MLP	activation functions, hidden layer sizes, and epochs

Finally, we compare three ways for dimension reduction and feature selection which are PCA, NMF, and KBest(chi2) to choose one for feature selection to compare full data results to reduced dimension one.

The top algorithms based on results and in compare to benchmark are MLP and Stacking ensemble methods. Let's dig into some details about how we apply them.

Multi-Layer Perceptron(MLP) or Artificial Neural Network is composed of three fully connected layers.

- Input layer: It has dense layer of 22 inputs which are the number of input data after preprocessing. Each input is multiplying with initial weight and we sum up all of them as the input for the next layer. The output of this layer passed by Rectified Linier Unit(ReLU) as activation function.
- Hidden Layer: A set of 8 neurons where each neuron has a weight assigned to it. It takes the input from input layer and does the dot product of inputs and weights, applies activation function, produce the result and pass the data to the next layer.
- Output Layer: Is the final layer that output only one value for our binary classification problem. It uses sigmoid activation function.

Each epoch the neural network refines its weights based on binary cross entropy loss function we used. For discrete p and q, the cross entropy is defined as $H(p,q) = -\sum_x p(x) \,.\, \log q(x)$.

	Propagation
Sigmoid	$y_s = \frac{1}{1+e^{-x_s}}$
Tanh	$y_s = \tanh(x_s)$
ReLu	$y_s = \max(0, x_s)$

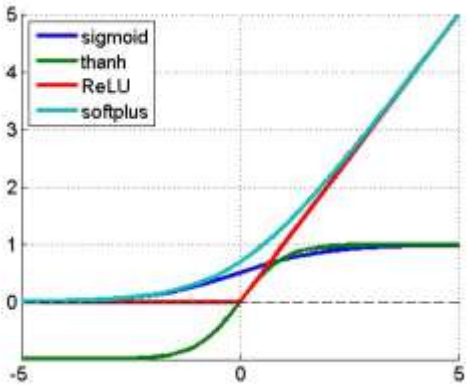

Figure 3 Activation Functions

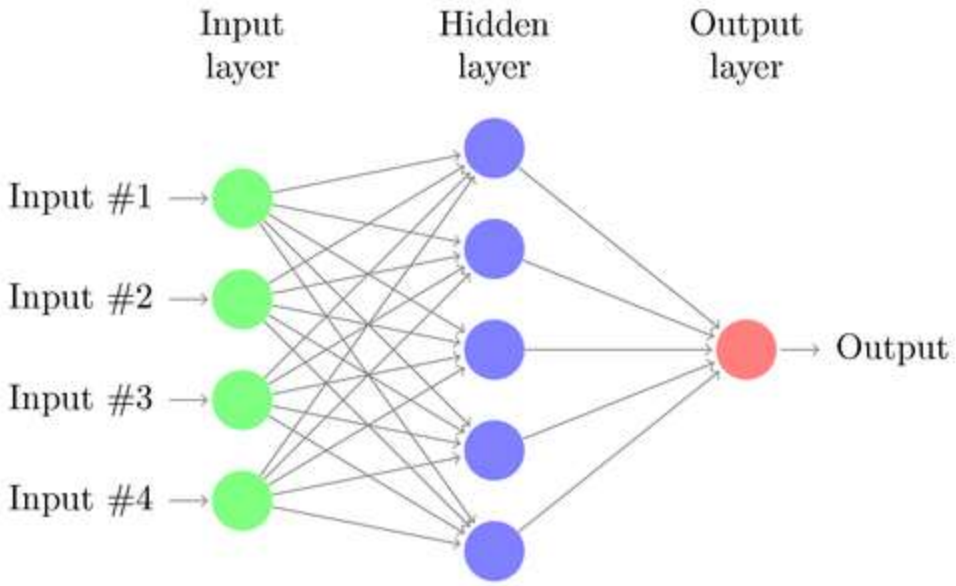

Figure 4 MLP

The Stacking ensemble takes many classifiers as input with a meta-classifier for stacking models as following:

1. For each model i in n selected models
    a. Add metadata M_i as feature to the input data which is the prediction based on model i
2. Fit a new model S (Stacked model) to metadata model found before
3. Used stacked model S to make final prediction on test metadata

Also, we are using AdaBoost algorithm which refer to family of algorithms that convert weak learner (i.e. learner with lower accuracy) to strong learner (i.e. learner with higher accuracy).

How does it work? Assume that the + sign referees to absence of heart disease and – referees to presence of heart disease. Boosting pays higher focus on examples which are mis-classified or have higher errors by preceding weak rules

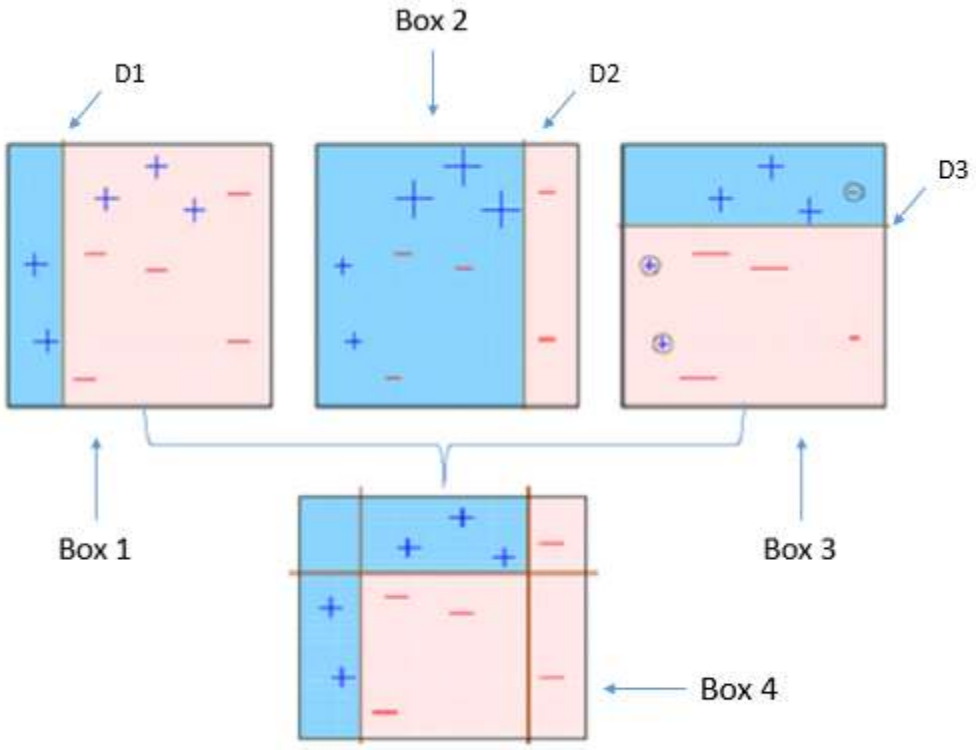

Figure 5 Boosting

1. The base learner takes all the training data we have then output box 1(i.e. weak learner) with decision stamp (D1) that looks like it's not a good classifier yet.
2. The error in box 1 taken into consideration (+ sign with errors magnified in box 2 to demonstrate it takes bigger weight) and we reapply the learner to get D2.
3. We repeat the process many times till the limit reached or accuracy of the learner increases.
4. Combine the result of weak learners to find strong learner in box 4 which improve the prediction power of the model. This is our final model that we can use to predict classification for any unknown input.

What is the base learner? In our case it is decision tree. For example, if we want to decide to go out or not based on weather data. We could represent our given data in decision tree as follows. At the leaf node the decision is taken based on 3 given features (Outlook, Humidity, and wind)

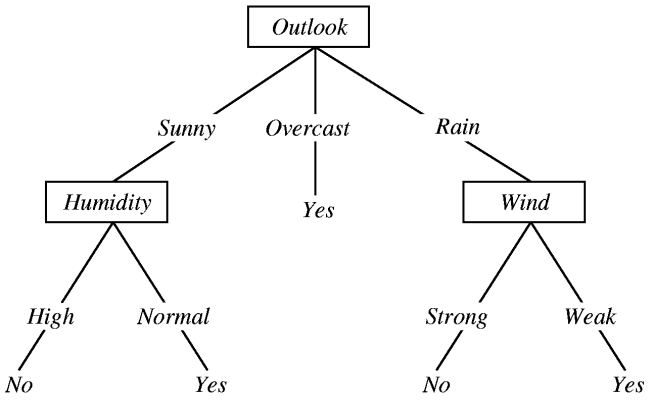

Figure 6 Decision Tree as a Base Learner

Benchmark

For benchmark, we used two approaches first by using linear model second, by comparing the result to papers that work on the same dataset and try to solve the same problem.

Here we can find the results previously found in other papers using the same dataset.

Title	Year	Accuracy
Optimal feature selection using a modified differential evolution algorithm and its effectiveness for prediction of heart disease	2017	0.85
Efficient Heart Disease Prediction System	2016	0.867
Computational intelligence for heart disease diagnosis: A medical knowledge driven approach	2013	.81 ~ .96
Feature selection for medical diagnosis: Evaluation for cardiovascular diseases	2013	0.85
A highly accurate firefly based algorithm for heart disease prediction	2015	0.88
An integrated decision support system based on ANN and Fuzzy_AHP for heart failure risk prediction	2016	0.91
Classifier ensemble reduction using a modified firefly algorithm: An empirical evaluation	2017	0.89
Heart disease Classification using Neural Network and Feature Selection	2011	0.8
Improving the heart disease Diagnosis by evolutionary algorithm of PSO and Feed Forward Neural Network	2016	0.91
Prediction of Heart Disease Using Neural Network	2017	0.95
Prediction of Heart Disease Using Hybrid Technique for Selecting Features	2017	0.84
Efficient Heart Disease Prediction system using Optimization Technique	2017	0.53
Feature selection for medical diagnosis: Evaluation for cardiovascular diseases	2013	0.89

III. Methodology

Data Pre-processing

We have done the following data preprocessing methods

- Feature scaling using minimax scalar for the numerical features 'age', 'restbp', 'chol', 'thalach', 'oldpeak', and 'ca' to be between 0 and 1 this could improve training of many of selected algorithms.
- To detect outliers, we have used interquartile range (IQR) then we detect outliers as the data records detected as outlier in two or more features. After testing we found that when removing outliers, the accuracy decreases so; we rollback this step.
- Transform the ‘num’ column in which we want to predict to only two values “absence (value 0)” and “disease (values 1)” because the data is not big enough to detect accurately the level of disease.

- Using one-hot encoder to convert discrete categorical values that has more than two values which are 'cp', 'restecg', 'slope', and 'thal'.
- Split and shuffling data into training and testing sets

Implementation

First, we implement the linear model using SGD classifier with default parameters to be our base model. Cross validation is used during the training process.

Second, we implement the MLP model using Keras and training the model for 100 epochs. A validation split of 1% is used during the training with shuffling and checkpoints.

Third, we train each classifier of the following list using the default parameters with cross validation. All classifiers in this step also used in stacked ensemble classifier with logistic regression as meta classifier.

- K-Nearest Neighbor
- Random Forest
- GaussianNB
- Gradient Boosting
- SVM
- AdaBoost
- Stacking ensemble

The stacked ensemble classifier in our implementation trains meta-classifier based on predicted probabilities instead of class labels. We found that this approach reaches higher accuracy.

Accuracy and F1 score calculated for each used classifier.

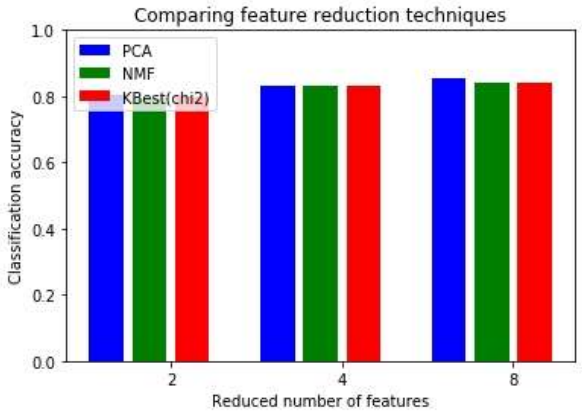

Figure 7 Comparing feature reduction techniques

During the coding process we found the results vary a little between runs however we define a random seed for classifiers when needed. This causes some complication when documenting the results.

We choose PCA from the three tested algorithms for dimension reduction as shown in Figure 7. We choose 8 dimensions based on explained variance of dimensions which reaches 0.0126 in the 8th dimension that's why we settle for 8 dimensions as shown in Figure 8.

In our implementation Non-Negative Matrix Factorization (NMF) finds two non-negative matrices (W, H) whose product approximates the non- negative matrix X. We are using this factorization for dimensionality reduction. The objective function is:

```
0.5 * ||X - WH||_Fro^2
+ alpha * l1_ratio * ||vec(W)||_1
+ alpha * l1_ratio * ||vec(H)||_1
+ 0.5 * alpha * (1 - l1_ratio) * ||W||_Fro^2
```

```
+ 0.5 * alpha * (1 - l1_ratio) * ||H||_Fro^2
```

Where:

```
||A||_Fro^2 = \sum_{i,j} A_{ij}^2 (Frobenius norm)

||vec(A)||_1 = \sum_{i,j} abs(A_{ij}) (Elementwise L1 norm)
```

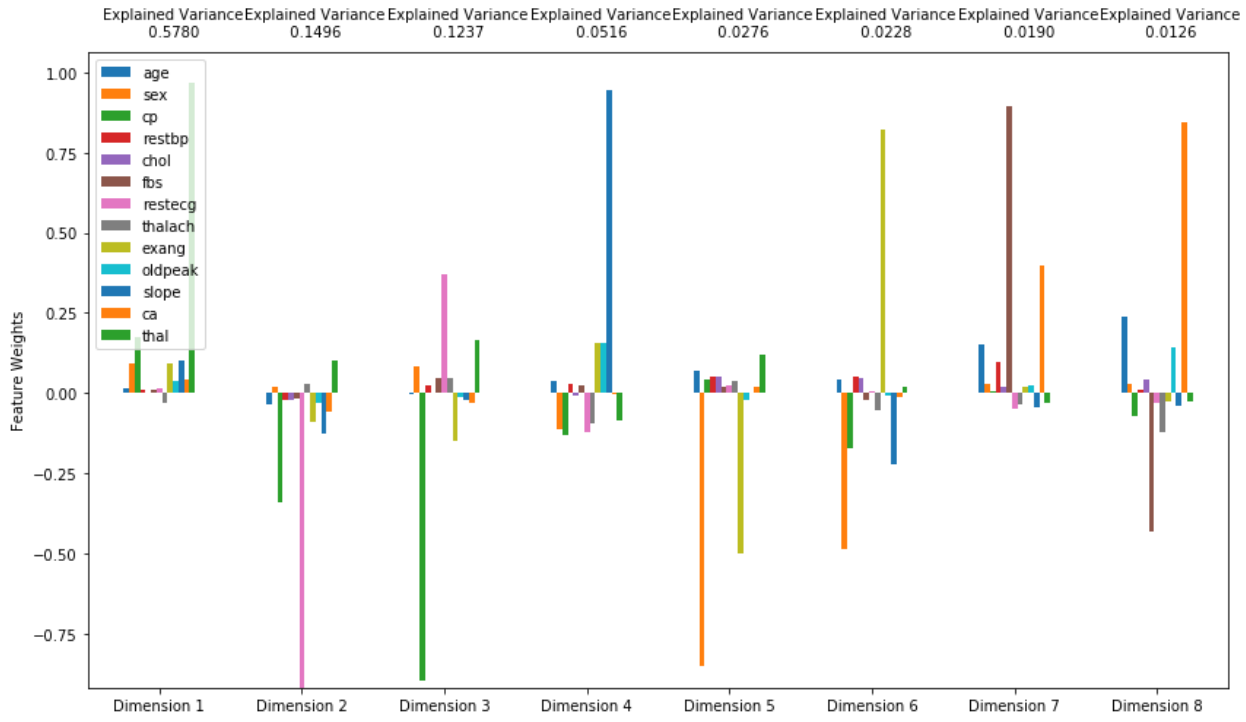

Figure 8 PCA Dimensions

The third method is Select K-Best selects features according to the k highest scores. The score is calculated in our implementation using chi2 function which compute chi-squared stats between each non-negative feature and class.

Refinement

Using grid search on the hyperparameters mentioned in Algorithms and Techniques section we found the best parameters to be as following.

Algorithm	Parameters
K-Nearest Neighbor	algorithm='ball_tree', n_neighbors=5, weights='distance'
Random Forest	criterion='entropy', n_estimators=5
GaussianNB	N/A
Gradient Boosting	n_estimators=50, max_depth=3
SVM	kernel='linear', C = 10, gamma = 0.001, probability=True

AdaBoost	algorithm='SAMME.R', learning_rate=0.1
Logistic Regression	C=10.0
Stacking ensemble	use_probas=True, average_probas=True, meta_classifier=logistic regression, classifiers= [KNN, Random Forest, GaussianNB, Gradient Boosting, SVM, AdaBoost]
MLP	Activation functions = relu for input layer and sigmoid for the output layer, hidden layer size = 8, dropout = 0.3 and epochs = 100, optimizer=rmsprop, loss function = binary cross entropy, metric = accuracy

The MLP layers are as following:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 22)                506
_________________________________________________________________
dropout_1 (Dropout)          (None, 22)                0
_________________________________________________________________
dense_2 (Dense)              (None, 8)                 184
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 9
=================================================================
Total params: 699.0
Trainable params: 699.0
Non-trainable params: 0.0
```

IV. Results

Model Evaluation and Validation

The results for the given models are as follows:

Algorithm	Validation Accuracy	Validation F_1	Test Accuracy	Test F_1
Base linear model	0.77	0.77	0.73	0.50
K-Nearest Neighbor	0.83	0.82	0.90	0.88
Random Forest	0.78	0.77	0.83	0.78
GaussianNB	0.84	0.82	0.90	0.88
Gradient Boosting	0.79	0.78	0.90	0.87
SVM	0.81	0.78	0.90	0.87
AdaBoost	0.82	0.80	0.87	0.83
Stacking ensemble	0.83	0.81	0.90	0.88
MLP	0.92	0.91	0.90	0.88

After dimension reduction the results changed to be as following:

Algorithm	Validation Accuracy	Validation F_1	Test Accuracy	Test F_1
K-Nearest Neighbor	0.82	0.81	0.80	0.77

Random Forest	0.81	0.79	0.87	0.85
GaussianNB	0.81	0.80	0.83	0.81
Gradient Boosting	0.81	0.80	0.83	0.81
SVM	0.83	0.81	0.87	0.83
AdaBoost	0.82	0.79	0.87	0.83
Stacking ensemble	0.85	0.84	0.87	0.85
MLP	0.89	0.88	0.83	0.80

From the above tables we can observe that the MLP is the best model in term of high validation accuracy and testing accuracy. It's also affected by the dimension reduction.

On the other hand, stacking ensemble gives high test accuracy and lower training accuracy than MLP. Its test accuracy less affected after dimension reduction. It could be the result of the improvement of random forest results after dimension reduction.

As a result, MLP model is chosen as the best model with high confidence. We have done further analysis on to validate the robustness of the final Neural Network. K-Fold is used with 10 splits which produce the following results for each fold:

```
Accuracy: 96.43%
Accuracy: 85.71%
Accuracy: 92.59%
Accuracy: 92.59%
Accuracy: 92.59%
Accuracy: 96.30%
Accuracy: 85.19%
Accuracy: 96.15%
Accuracy: 88.46%
Accuracy: 92.31%
Mean Accuracy: 91.83% (+/- 3.93%)
```

The results above show how robust the model is. It has standard deviation of about 4% which measures a low error of deviation from the central tendency.

Justification

Compared to the base linear model the MLP reach a high improvement in term of validation and testing accuracy.

The result found is better than 9 out of 13 papers mentioned before working on the same dataset. The best of all is that our approach is very simple and straight forward as alternative to other complex and hybrid methods. It's even better than stacked classifier which sum up many other classifiers.

V. Conclusion

Free-Form Visualization

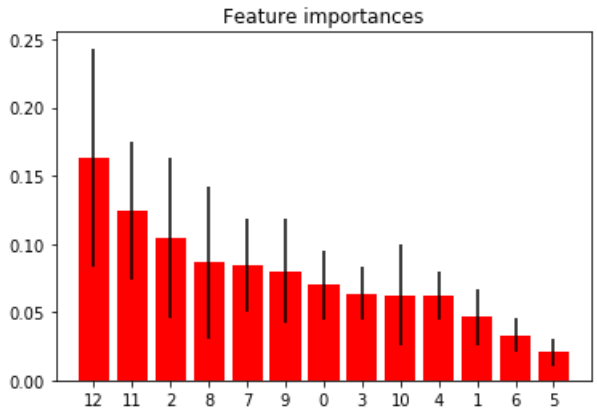

Figure 9 Feature Importance Plot

In Figure 9 we show feature importance plot of our dataset features. We have implement this plot using forests of trees to find which features are important. X-axis shows the index of feature and Y-axis is the importance.

```
Feature ranking:
1. [thal] - feature 12 (0.163345)
2. [ca] - feature 11 (0.124498)
3. [cp] - feature 2 (0.104463)
4. [exang] - feature 8 (0.086379)
5. [thalach] - feature 7 (0.084189)
6. [oldpeak] - feature 9 (0.080116)
7. [age] - feature 0 (0.069942)
8. [restbp] - feature 3 (0.063486)
9. [slope] - feature 10 (0.062225)
10. [chol] - feature 4 (0.061913)
11. [sex] - feature 1 (0.046255)
12. [restecg] - feature 6 (0.032843)
13. [fbs] - feature 5 (0.020346)
```

From the result above we can find that `fbs` for example not an informative feature.

Reflection

In this work we have compared many machine learning techniques for heart diseases classification. We ended up with simple optimized model based on neural network. We also discussed the effect of feature selection and dimension reduction on the classification accuracy.

The most difficult aspect of the project is dataset analysis and trying to map this small dataset to distribution to be able to find the best method for learning. The dataset is found to be non-linear with many features that makes it hard to predict.

Improvement

One of the next steps we could make to improve our results is by working on more complex model by combining neural network with stacked ensemble learning method in multi-stage classifier but, this model could need a bigger dataset or multiple datasets to examine for better results.

VI. References:

Artificial neural networks with Math

Cross entropy

ReLU

Activation Functions: Neural Networks

David H. Wolpert. “Stacked Generalization.” Neural Networks. Volume 5. (1992)

Leo Breiman. “Stacked Regressions.” Machine Learning, 24, 49-64 (1996)

Mark J van der Laan, Eric C Polley, and Alan E Hubbard. “Super Learner.” Journal of the American Statistical Applications in Genetics and Molecular Biology. Volume 6, Issue 1. (September 2007).

LeDell, E. “Scalable Ensemble Learning and Computationally Efficient Variance Estimation” (Doctoral Dissertation). University of California, Berkeley, USA. (2015)

Quick Introduction to Boosting Algorithms in Machine Learning

An Introduction to Machine Learning With Decision Trees

NMF in SKLearn

SelectKBest in SKLearn

Chi2 in SKLearn