Capstone Project

Machine Learning Engineer Nanodegree

Georgios Alexandris

January, 2020

Definition

Project Overview

As is mentioned in MATH 2319 Machine Learning Applied Project Phase I[1], heart disease is currently the leading cause of death across the globe. It is anticipated that the development of computation methods that can predict the presence of heart disease will significantly reduce heart disease caused mortalities while early detection could lead to substantial reduction in health care costs.
In this project, I created a binary classifier, which takes as input the attributes of a patient, and predicts whether this person has cardiovascular disease (value of 1) or not (value of 0). The classifier was trained using the UCI Machine Learning Repository[2] datasets. The project was inspired by Heart Diseases Detection Using Naive Bayes Algorithm[3] article.

Problem Statement

The goal is to create a classifier running on AWS platform; the tasks involved are the following:

1. Download and preprocess UCI datasets for heart disease
2. Train a classifier to predict the presence of heart disease
3. Make the classifier run on AWS platform
4. Test and improve the classifier by tuning model hyperparameters

Metrics

Recall is a common metric in sick patient detection; it actually calculates how many of the Actual Positives our model capture through labeling them as Positives (True Positives).

$$\textbf{\textit{recall}} = \frac{\textbf{\textit{true positives}}}{\textbf{\textit{true positives + false negatives}}}$$

This metric was used when evaluating the classifier because there is a high cost associated with False Negatives:

- ❖ If a sick patient (Actual Positive) goes through the test and predicted as not sick (Predicted Negative), the cost associated with False Negative will be extremely high if the sickness is contagious

[1] Charles Galea. "MATH 2319 Machine Learning Applied Project Phase I." 8 Apr. 2018. https://rstudio-pubs-static.s3.amazonaws.com/396380_639e2f68b09e41a0b05f97b5dc8eb3f2.html.

[2] David W. Aha. "Machine Learning Repository." UCI Machine Learning Repository. 14 Aug. 1991. http://archive.ics.uci.edu/ml/datasets/heart+disease.

[3] K.Vembandasamy. "Heart Diseases Detection Using Naive Bayes Algorithm." IJISET. 9 Sep. 2015. http://ijiset.com/vol2/v2s9/IJISET_V2_I9_54.pdf.

Analysis

Data Exploration

The UCI Heart Disease DataSets consist of 4 databases: Cleveland, Hungary, Switzerland, and the VA Long Beach, with a total of 920 records. Each record contains 14 attributes:

1. age: age in years
2. sex: (1 = male; 0 = female)
3. cp: chest pain type
   - ❖ 1 = typical angina
   - ❖ 2 = atypical angina
   - ❖ 3 = non-anginal pain
   - ❖ 4 = asymptomatic
4. trestbps: resting blood pressure (in mm Hg on admission to the hospital)
5. chol: serum cholesterol in mg/dl
6. fbs: fasting blood sugar > 120 mg/dl
   - ❖ 1 = true
   - ❖ 0 = false
7. restecg: resting electrocardiographic results
   - ❖ 0 = normal
   - ❖ 1 = having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
   - ❖ 2 = showing probable or definite left ventricular hypertrophy by Estes' criteria
8. thalach: maximum heart rate achieved
9. exang: exercise induced angina
   - ❖ 1 = yes
   - ❖ 0 = no
10. oldpeak: ST depression induced by exercise relative to rest
11. slope: the slope of the peak exercise ST segment
    - ❖ 1 = upsloping
    - ❖ 2 = flat
    - ❖ 3 = downsloping
12. ca: number of major vessels (0-3) colored by flourosopy
13. thal: thallium test
    - ❖ 3 = normal
    - ❖ 6 = fixed defect
    - ❖ 7 = reversable defect
14. num (target feature): number of major vessels with >50% narrowing (0,1,2,3, or 4)

As it can be suspected based on the above features, some records may have anomalies, which means they must be discarded during data cleaning.

Exploratory Visualization

The plot below shows that 10 of the 14 features contain missing values. Notably, the features: slope, **ca** and **thal** have 33.6%, 66.4% and 52.8% missing values, respectively.

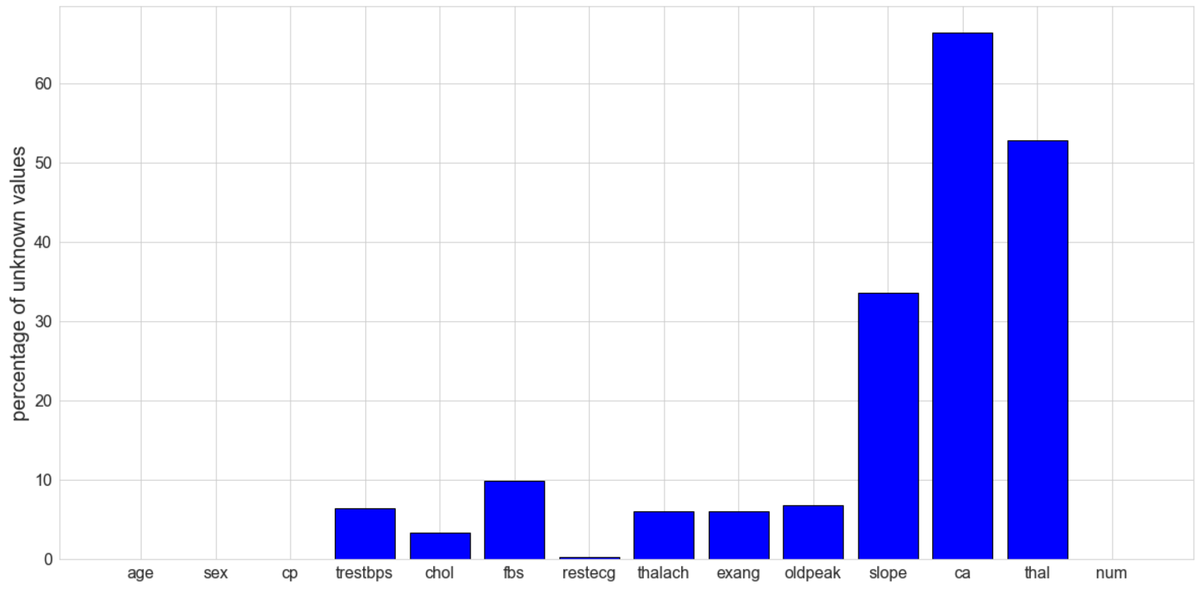


Fig. 1 Percentage of **missing values** (?) for each feature.

Plots Fig2 and Fig3 show that **trestbps** (resting blood pressure) and **chol** (serum cholestrol) contain several data entries with values of 0 which are not possible for these diagnostic tests(4).

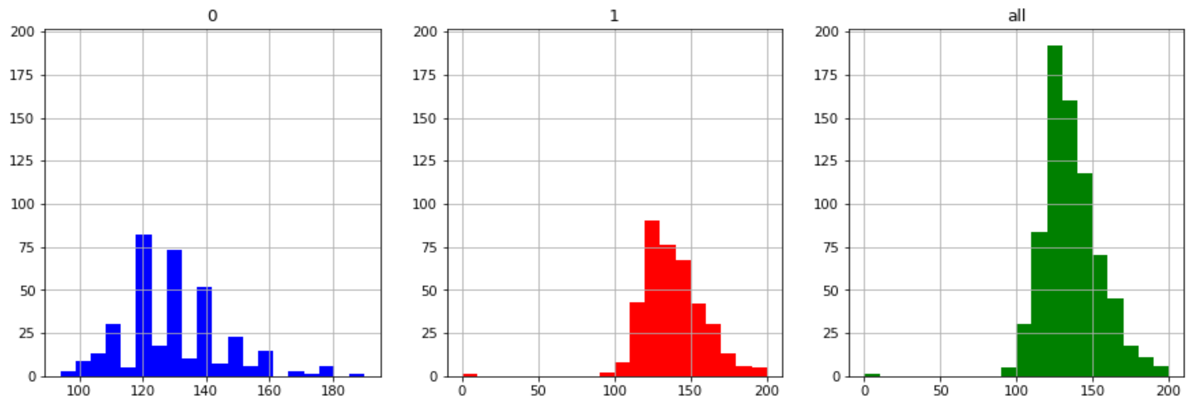


Fig. 2 **Trestbps** histogram for healthy (0), unhealthy(1) and all persons.

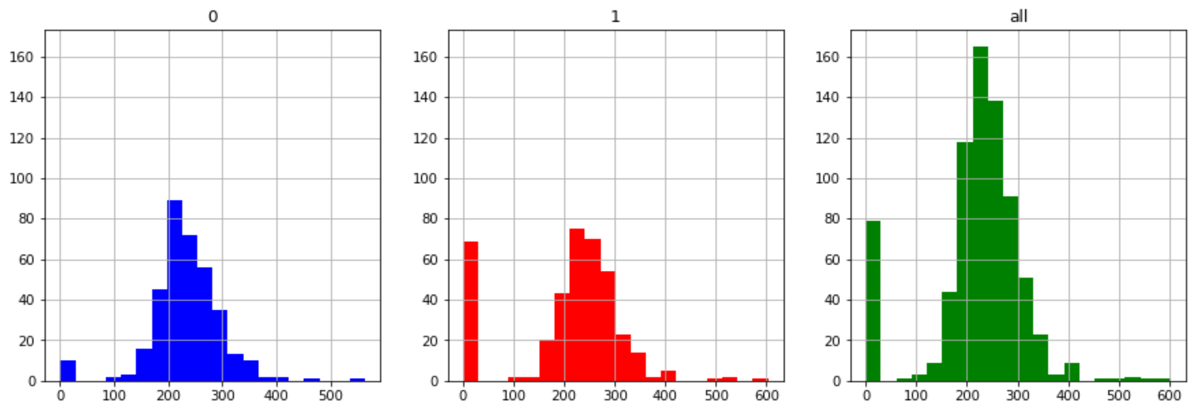


Fig. 3 **Chol** histogram for healthy (0), unhealthy(1) and all persons.

4. Charles Galea. "MATH 2319 Machine Learning Applied Project Phase I." 8 Apr. 2018. https://rstudio-pubs-static.s3.amazonaws.com/396380_639e2f68b09e41a0b05f97b5dc8eb3f2.html.

Finally Fig4 shows that there is no significant feature correlation.

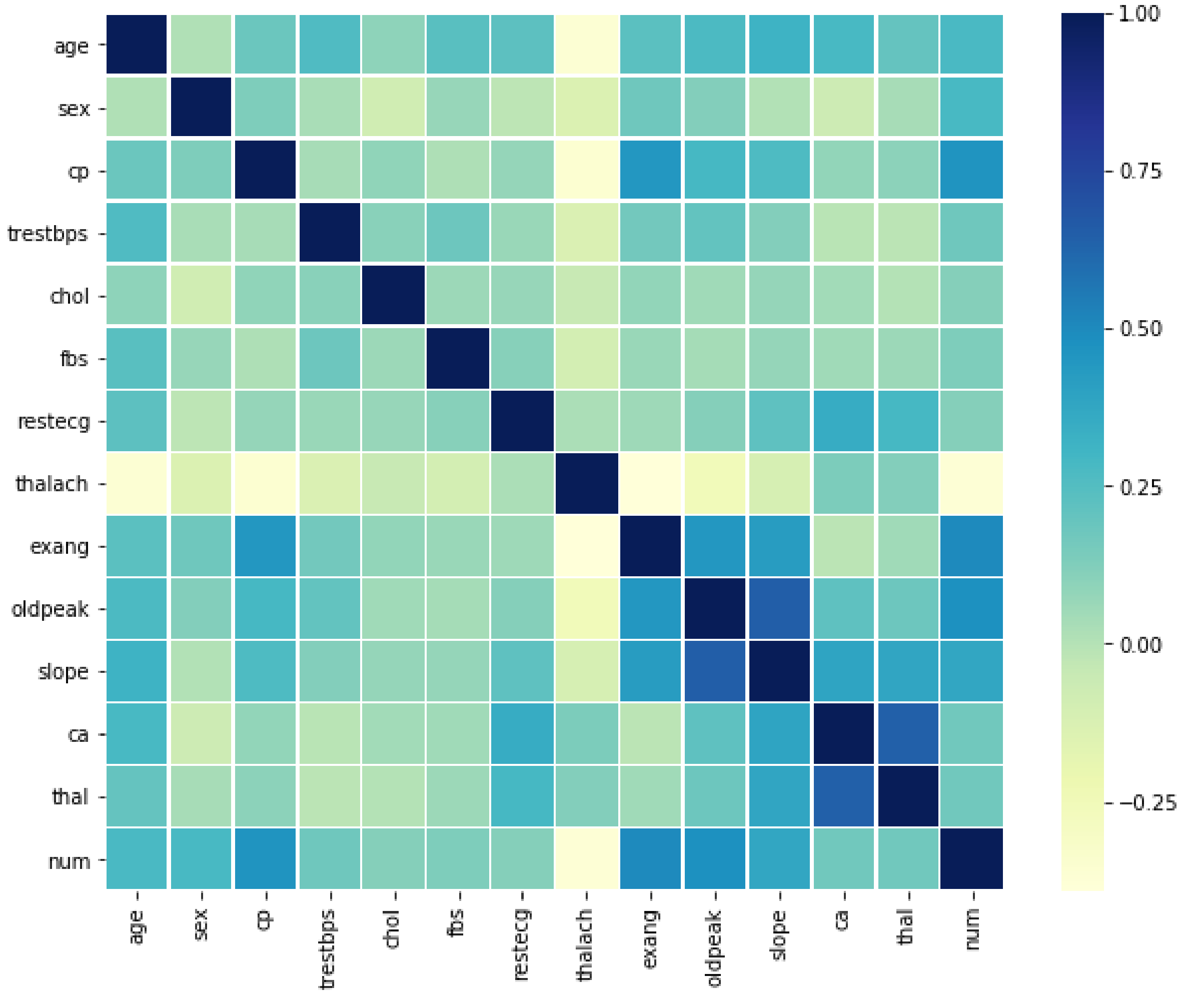


Fig. 4 Dataset features heatmap

Algorithms and Techniques

The classifier is `LinearLearner`(5), an aws Sagemaker's supervised learning algorithm, used for solving either classification or regression problems, with which you can explore discrete training objectives such as F1 measure, precision, recall, or accuracy.

The linear learner algorithm requires a data matrix, with rows representing the observations, and columns representing the dimensions of the features. It also requires an additional column that contains the labels that match the data points. At a minimum, it requires you to specify output data location, and objective type (classification or regression) as arguments.

You can specify additional parameters in the HyperParameters string map of the request body. These parameters control the optimization procedure and can be tuned to optimize the classifier. For example:

- epochs: The maximum number of passes over the training data
- positive_example_weight_mult: The weight assigned to positive examples when training a binary classifier. If you want the algorithm to choose a weight so that errors in classifying negative vs. positive examples have equal impact on training loss, specify *balanced*

[5] Amazon Web Services. "Linear Learner Algorithm." https://docs.aws.amazon.com/sagemaker/latest/dg/linear-learner.html

- binary_classifier_model_selection_criteria: The model evaluation criteria for the training dataset. Criteria include:
  - accuracy—The model with the highest accuracy (default value).
  - f_beta—The model with the highest F1 score. The default is F1.
  - precision_at_target_recall—The model with the highest precision at a given recall target.
  - recall_at_target_precision—The model with the highest recall at a given precision target.
- target_recall: The target recall

Benchmark

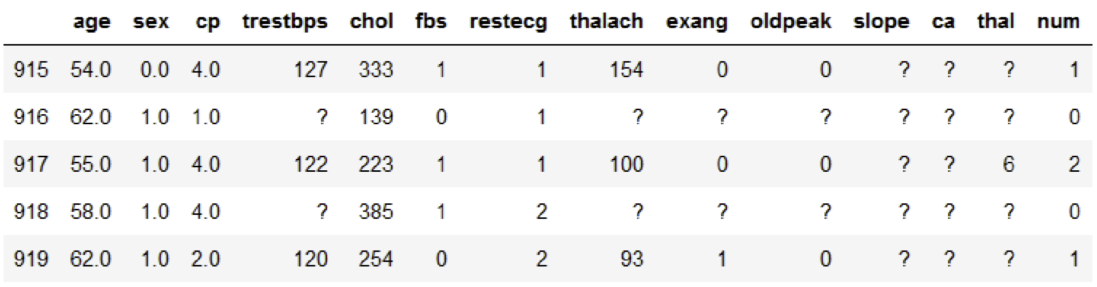
Heart Diseases Detection Using Naive Bayes Algorithm[6] article presented a model, which used the same dataset and achieved average recall of 74% for the test set.
This recall will be used to benchmark the classifier.

Methodology

Data Preprocessing

The preprocessing done in the "Heart Disease Prediction" notebook consists of the following steps:

1. Datasets concatenation:
   The 4 databases: Cleveland, Hungary, Switzerland, and the VA Long Beach, are concatenated to one dataset with a total of 920 records. We observe that some feature values are missing for some patients (denoted with the ? symbol)



	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num
915	54.0	0.0	4.0	127	333	1	1	154	0	0	?	?	?	1
916	62.0	1.0	1.0	?	139	0	1	?	?	?	?	?	?	0
917	55.0	1.0	4.0	122	223	1	1	100	0	0	?	?	6	2
918	58.0	1.0	4.0	?	385	1	2	?	?	?	?	?	?	0
919	62.0	1.0	2.0	120	254	0	2	93	1	0	?	?	?	1

2. Calculation of the percentage of missing values for each feature:

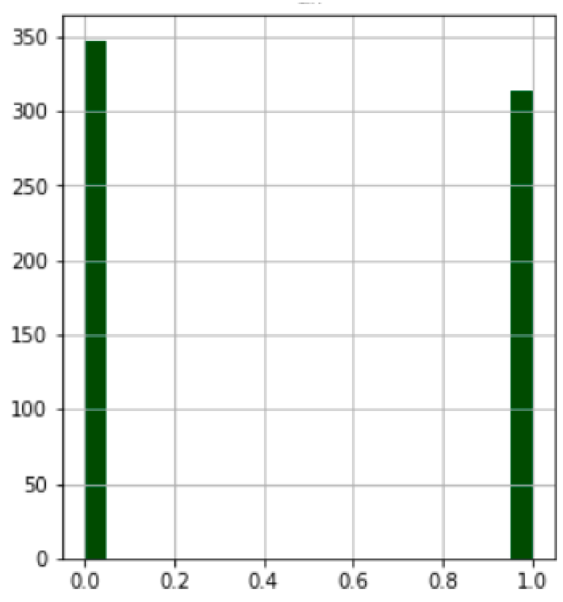
```
trestbps      6.413043
chol          3.260870
fbs           9.782609
restecg       0.217391
thalach       5.978261
exang         5.978261
oldpeak       6.739130
slope        33.586957
ca           66.413043
thal         52.826087
```

3. Removal of rows containing features with percentage of missing values < 10%:
   Rows containing features with percentage of missing values < 10% ('trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang' and 'oldpeak') are removed since they represent a small amount of the overall dataset.

4. Replacement of the remaining missing values with the number -1:
   Rows containing features with percentage of missing values > 10% (slope, 'ca and 'thal') represent a large amount of the dataset and cannot be removed. So we replace the missing values with the number -1.

[6] K.Vembandasamy. "Heart Diseases Detection Using Naive Bayes Algorithm." IJISET. 9 Sep. 2015. http://ijiset.com/vol2/v2s9/IJISET_V2_I9_54.pdf.

5. Binary classification of the dataset:
   The 'num' feature is used for data classification. We assume that a num value of 0 means 'heart is okay', and that num values of 1,2,3 or 4 mean 'heart disease'. That's why we replace num values of 2,3 or 4 with 1 and now our dataset is binary classified.
6. Removal of rows with (trestbps or chol = 0):
   trestbps (resting blood pressure) and chol (serum cholesterol) features contain several data entries with values of 0 which are not possible for these diagnostic tests. So we remove these rows.
7. Calculation of deleted rows and of class balance:
   After all the above data cleaning the remaining dataset has 661 records (percentage of deleted rows equal to 28%) and is quite balanced (52.5 % healthy and 47.5% with heart disease)



Implementation

During the implementation process, the classifier is trained on the preprocessed training data. This is done in a Jupyter notebook (titled "Heart Disease Prediction"), running in an Amazon SageMaker notebook instance (a machine learning Amazon EC2 compute instance) and can be further divided into the following steps:

1. Get the 4 databases from the UCI server and store them locally
2. Preprocess and visualize them as described in the previous sections
3. Split the preprocessed training data into training (67%) and test (33%) matrices
4. Define the LinearLearner estimator (the Sagemaker built-in algorithm) using the following hyperparameters:
   - role: a role that specifies the operations that we permit to estimator
   - train_instance_count: 1
   - train_instance_type: ml.c4.xlarge
   - predictor_type: binary_classifier
   - output_path: a path in our Amazon S3 bucket
   - sagemaker_session: the notebook's current Sagemaker session
   - epochs: 15
5. Create the linear estimator
6. Use the estimator to convert the preprocessed training data to a RecordSet
7. Train the linear estimation using the above RecordSet

The training process achieves the following metrics:

```
[01/21/2020 18:21:34 INFO 139877227620160] #quality_metric: host=algo-1, train
        binary_classification_accuracy <score>=0.809954751131
[01/21/2020 18:21:34 INFO 139877227620160] #quality_metric: host=algo-1, train precision
        <score>=0.785714285714
[01/21/2020 18:21:34 INFO 139877227620160] #quality_metric: host=algo-1, train recall
        <score>=0.830188679245
```

and saves the trained estimator to our S3 bucket.

Refinement

As mentioned in the Metrics section, recall is very crucial in sick patient detection, because we want an estimator that has as few false negatives as possible.
In order to improve the above default linear estimator model, LinearLearner offers the hyperparameter 'binary_classifier_model_selection_criteria', which is the model evaluation criteria for the training dataset. To set recall as the model evaluation criteria we must give 'precision_at_target_recall' as its value.
We also have to further set the exact value of recall we want to aim for, as value to the 'target_recall' hyperparameter.
Finally in order to tune positive examples to have equal impact on training as the negative ones, we can set the value: balanced to 'positive_example_weight_mult' hyperparameter.
The refinement process includes the repetition of steps 4, 5 and 7 of the implementation:

1. Define the tuned LinearLearner estimator using the following additional hyperparameters:
    - ❖ binary_classifier_model_selection_criteria: precision_at_target_recall
    - ❖ target_recall: 0.90
    - ❖ positive_example_weight_mult: balanced
2. Create the tuned linear estimator
3. Train the linear estimation using the above created training RecordSet

The tuned training process achieves the following metrics:

```
[01/21/2020 18:32:16 INFO 139972385855296] #quality_metric: host=algo-1, train
        binary_classification_accuracy <score>=0.782805429864
[01/21/2020 18:32:16 INFO 139972385855296] #quality_metric: host=algo-1, train precision
        <score>=0.718045112782
[01/21/2020 18:32:16 INFO 139972385855296] #quality_metric: host=algo-1, train recall
        <score>=0.900943396226
```

and saves the tuned trained estimator to our S3 bucket.

Results

Model Evaluation and Validation

During development and refinement, a test dataset was used to evaluate the default and the tuned model respectively.

The steps for the **default** model evaluation are the following:

- ❖ Deploy the trained model and create a predictor
- ❖ Use the predictor to make predictions for the test dataset
- ❖ Get metrics from these predictions

and these are the achieved metrics:

```
prediction (col)  0.0  1.0
actual (row)
0.0                88   29
1.0                15   87

Recall:     0.853
Precision:  0.750
Accuracy:   0.799
```

The steps for the **tuned** model evaluation are the following:

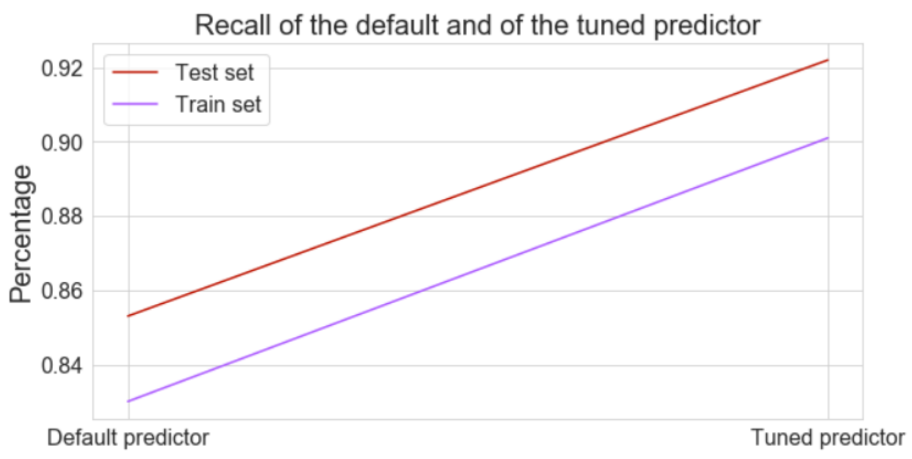
- ❖ Deploy the tuned trained model and create a predictor
- ❖ Use the tuned predictor to make predictions for the test dataset
- ❖ Get metrics from these predictions

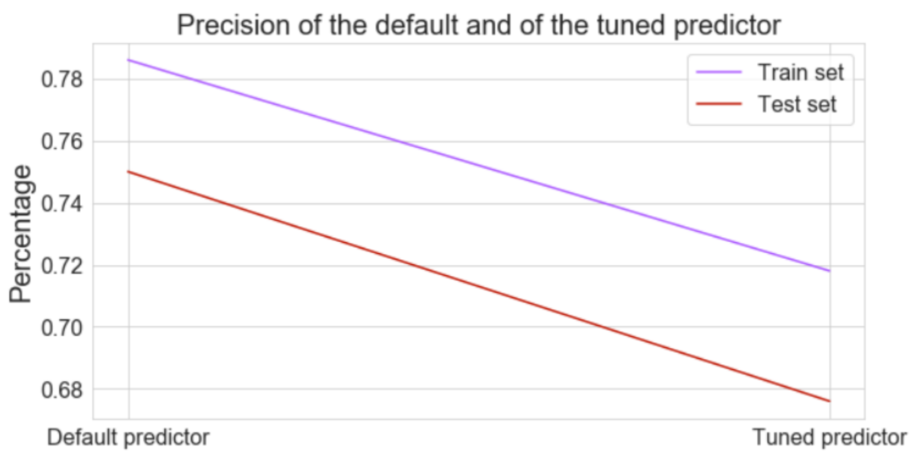
and these are the achieved metrics:

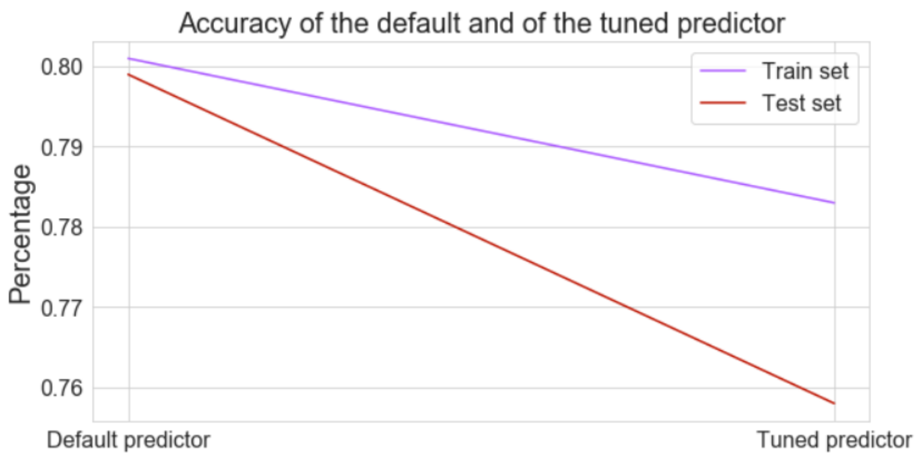
```
prediction (col)  0.0  1.0
actual (row)
0.0                72   45
1.0                 8   94

Recall:     0.922
Precision:  0.676
Accuracy:   0.758
```

The above metrics of the two models are displayed in the charts bellow:







Justification

When we compare the above data we make the following observations:

- ❖ Both models have much better recall from that of the benchmark.
- ❖ The tuned model seems to be slightly overfitted to the train dataset, but achieves the highest possible recall.

We finally chose the tuned model as our final solution, because we prefer to have some more healthy people predicted as diseased and drive them do some extra medical exams, than having more diseased people predicted as healthy and loose the chance to deal with their problem as soon as possible.