Heart Attack Prediction Neural Network Design

Pengfei Ma

Boston University

CS 767

Prof. Heather

Nov.25th 2021

Abstract

Machine learning and artificial intelligence, as the general trend of future development, have gradually shown irreplaceable roles in many fields. The most well-known ones are computer science, financial models, and robotics. With the development of machine learning, more fields can obtain more efficient and accurate predictions through machine learning. For example, the verification code needs to be filled in every time you log in. These verification codes require users to select sidewalks, traffic lights, buses, etc. in order to provide sufficient training sets for artificial intelligence. Based on this principle, machine learning has a greater advantage in the healthcare field, because each patient will enrich the training set and increase the accuracy of the model's prediction. In the case of tight medical resources, an efficient machine learning model can provide disease analysis for some acute diseases such as heart attack at the fastest speed and provide emergency treatment recommendations for nurses until the arrival of professional doctors.

This project would focus on designing a neural network prediction model for heart attacks. The data sets used in this project were collected from *UCI Machine Learning Open Data Websites*. Processed_cleveland.data, heart.csv, and reprocessed_hungarian.data are three data sets that were used in this project. The data pre-processing step would merge these three data set as one large dataset. At first, data visualization by matplotlib and seaborn will be applied to the merged data set to describe the age, sex, trestbps, chol, and thalach values. After visualization, train and test set split will be 70/30 by train_test_split from sk-learn package. First 12 columns are features and the last column is the label.

Except multilayer perceptron, three classifiers will be used for the same data set to find out the advantage and disadvantage of the neural networks. At the end, model evaluations will be applied to rank the accuracy from the test set from highest to the lowest and rolled out the comparison of classifiers and neural networks.

Multi-layers Perceptron has four layers with 200 neurons each and 0.1 validation set. This project will use grid and random search to test different number of epochs: 100, 200, 300; and four different optimizers: adam, nadam, sgd, rmsprop. After finding the best model, evaluated and visualized the changes of the accuracy of the model by each epoch at the end to summarize the entire progress.

Logistic regression, Naïve Bayes, and Gaussian SVM will be used to compare the accuracy at the end. All three classifiers will use the built model from sk-learn

Keywords: visualization, min max scale, sk-learn, Multi-layer Perceptron, Grid Search, Logistic regression, Gaussian Naïve Bayes, Gaussian SVM.

Research scenario

This project will implement multi-layer perceptron and logistic regression, naïve bayes, and gaussian svm to find out the accuracy of the model to predict the heart attack rate. Found out the best MLP model that have the highest prediction accuracy and evaluate the model by the test set. Compared and ranked four accuracies and conclude the advantages of the neural networks.

The main research question is that under what conditions will the neural network perform better than the classifiers.

Project map

Data preparation

I. Pick the data set

Based on the choices of the data set, this project decided to choose *UCI Machine Learning Open Data Websites* (http://archive.ics.uci.edu/ml/datasets/Heart+Disease) as the resource of data website. Three data files were collected: Processed_cleveland.data, heart.csv, and reprocessed_hungarian.data. These three data sets will be processed and merged for the future use in this project. Columns from 1st to 13th are the features of a patient and the 14th column (target) is the label prediction attributes. X and Y set will be split in the later of this project.

II. Import data set

Import all three data sets into Pandas data frames and set the column names as "age", "sex", "cp", "trestbps", "chol", "fbs", "restecg", "thalach", "exang", "oldpeak", "slope", "ca", "thal", "target". Target is the label.

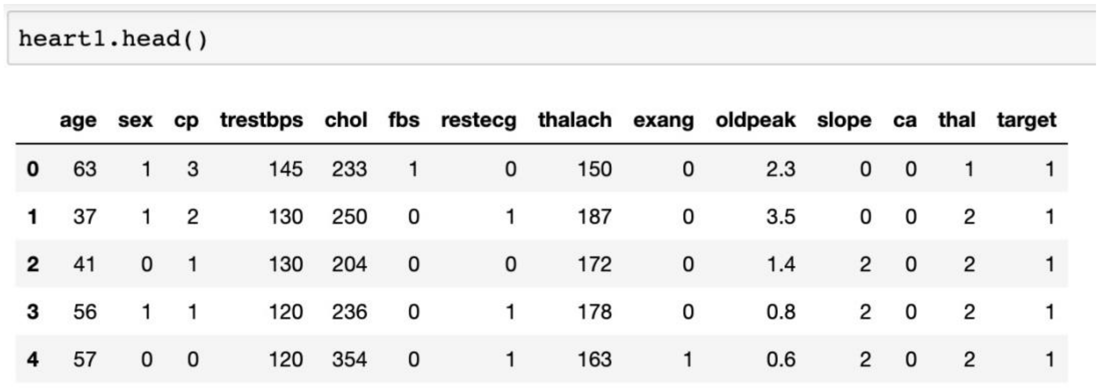
Here is the data description of the data set.

Column name	**Description**
age	Age of the patient in years
sex	Sex of the patient • 1 = male; • 0 = female
cp	Chest Pain type • Value 1: typical angina; • Value 2: atypical angina; • Value 3: non-anginal pain; • Value 4: asymptomatic
trestbps	Resting blood pressure (in mm Hg)
chol	Serum cholesterol in mg/dl

fbs	Fasting blood sugar > 120 mg/dl • 1 = true; • 0 = false
restecg	Resting electrocardiographic results • Value 0: normal; • Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV); • Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
thalach	Maximum heart rate achieved
exang	Exercise induced angina • 1 = yes; • 0 = no
oldpeak	ST depression induced by exercise relative to rest
slope	The slope of the peak exercise ST segment • Value 1: upsloping; • Value 2: flat; • Value 3: downsloping
ca	Number of major vessels (0-3) colored by flourosopy
thal	Thal rate • 3 = normal; • 6 = fixed defect; • 7 = reversable defect; • 9 = unclaimed
target	Diagnosis of heart disease (angiographic disease status) • Value 0: < 20% diameter narrowing; • Value 1: >= 20% and < 40% diameter narrowing; • Value 2: >= 40% and < 60% diameter narrowing; • Value 3: >= 60% and < 80% diameter narrowing; • Value 4: > 80% diameter narrowing

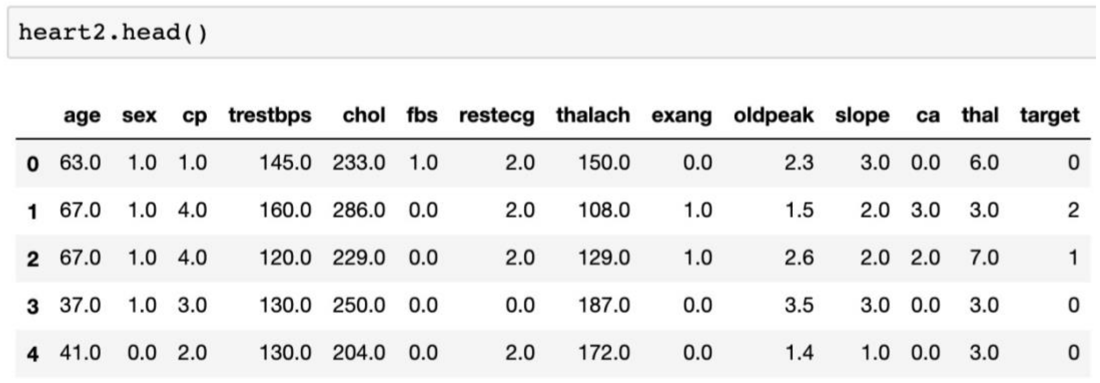
III. Process the data set

After importing the data set, this is the screenshot of the data set.



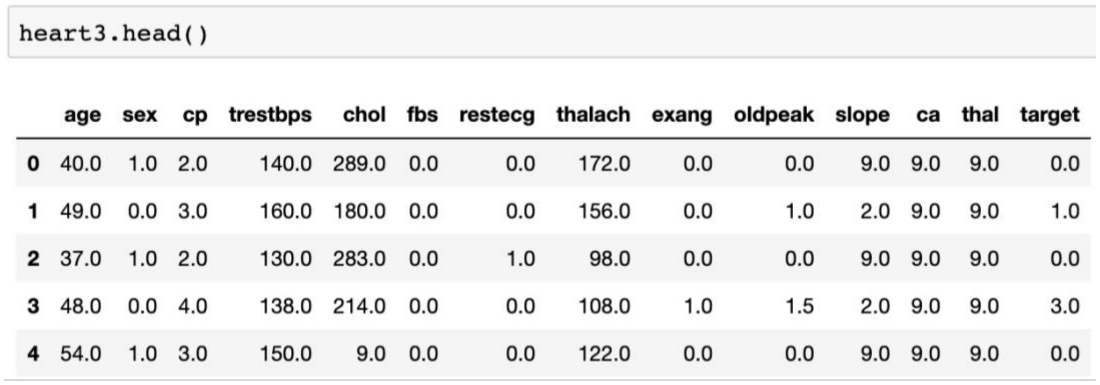
```
heart1.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1



```
heart2.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
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
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
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
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
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



```
heart3.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	40.0	1.0	2.0	140.0	289.0	0.0	0.0	172.0	0.0	0.0	9.0	9.0	9.0	0.0
1	49.0	0.0	3.0	160.0	180.0	0.0	0.0	156.0	0.0	1.0	2.0	9.0	9.0	1.0
2	37.0	1.0	2.0	130.0	283.0	0.0	1.0	98.0	0.0	0.0	9.0	9.0	9.0	0.0
3	48.0	0.0	4.0	138.0	214.0	0.0	0.0	108.0	1.0	1.5	2.0	9.0	9.0	3.0
4	54.0	1.0	3.0	150.0	9.0	0.0	0.0	122.0	0.0	0.0	9.0	9.0	9.0	0.0

Next, merging three data sets into one large data set for train and test set split using pandas.concat() with ignored index. Converted the data frame to numpy array.

Setting X for the first 12 columns and all rows of these columns as the features and setting Y for the 13th column as the label. Then imported preprocessing from sklearn. Used min_max_scaler to scale the X set. Named the scaled feature set to X_scale.

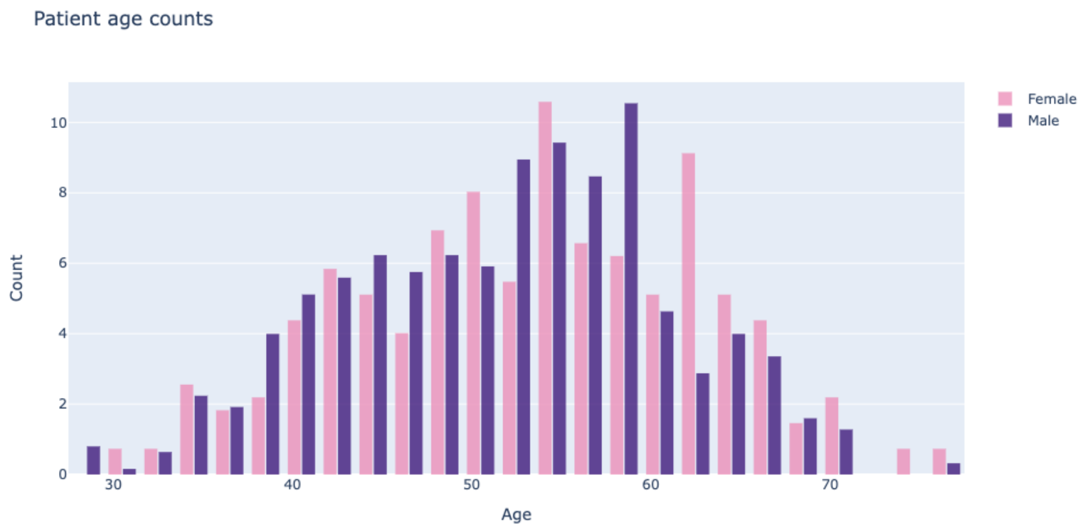
Imported train_test_split from sklearn.model_selection to split X_scale and Y into X_train, X_test, Y_train, Y_test with test_size = 0.3.

Data visualization

I. Age and sex

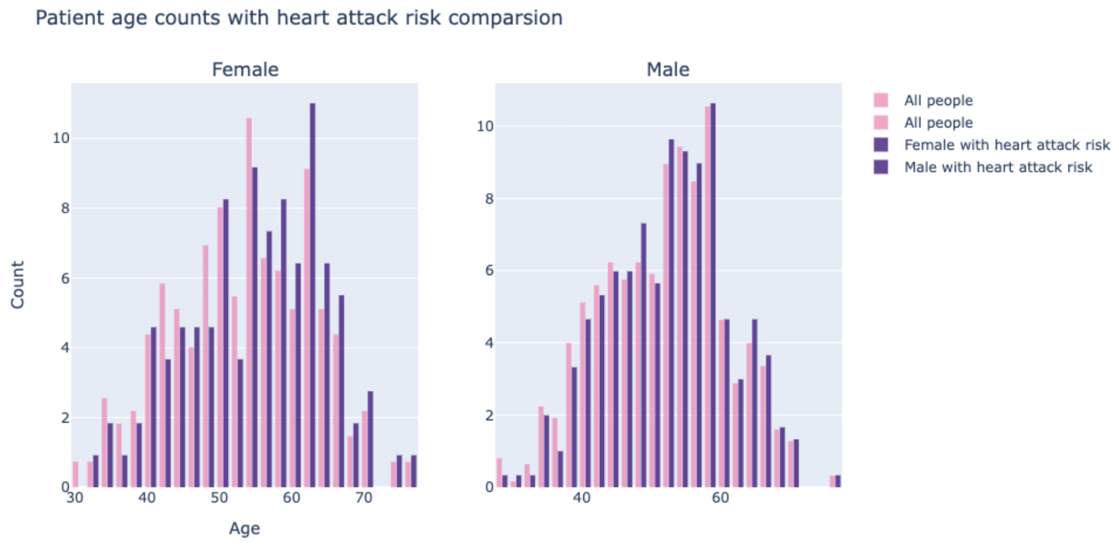
1. Histogram

By histogram from plotly, the main age of the patients in this project is between 40 and 60 years old.



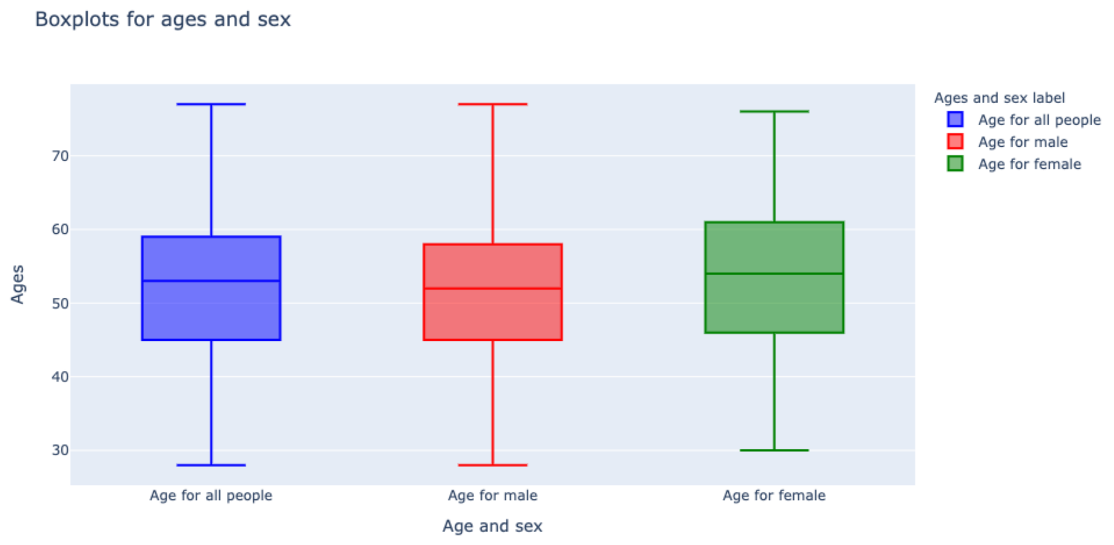
2. Comparison with target

By comparing the total number of people of different genders and the number of people at risk of heart attack. At different ages, women have a higher risk of heart attack than men.



3. Boxplot of ages

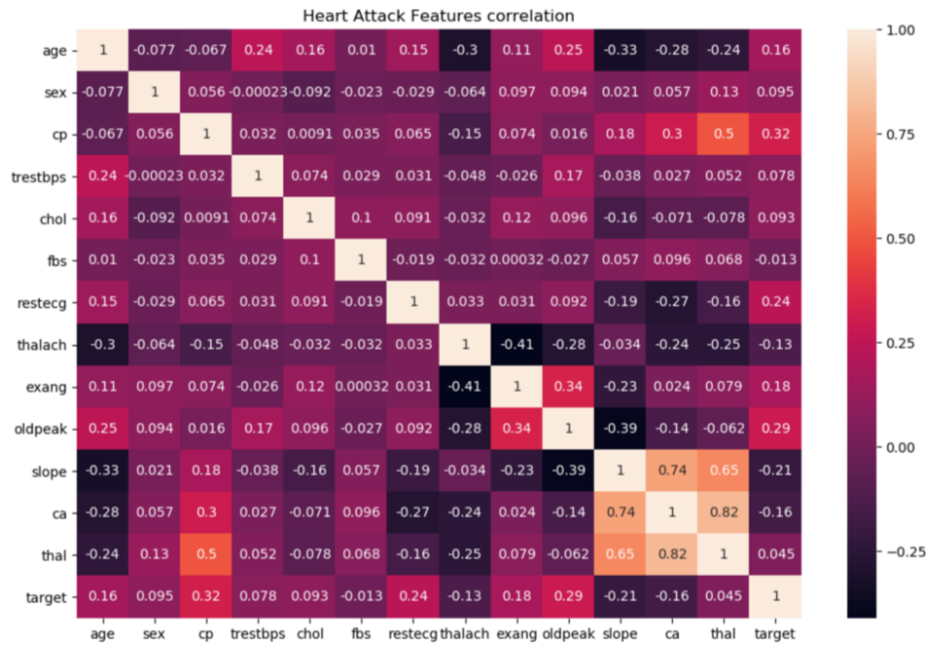
It is clear that the range of age for all patients is from 28 to 77 years old, the median is 53 years old. For male patients, the range is from 28 to 77 years old and the median is 52. For female patients, the range is from 30 to 76 years old and the median is 54.



II. Correlation Heatmap

Created the correlation heatmap by seaborn to find out that the three features that are most related to the heart attack risk (target) is CP, oldpeak, and restecg.

The three features that are least related to the heart attack risk is fbs, thal, and trestbps. Since their absolute values are close to 0.



Multi-layers Perceptron

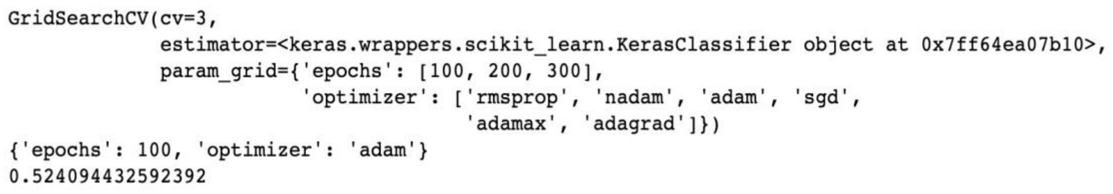
I. Model design

Created a MLP with four hidden layers. Defined the MaxNormDense by partial function with ‘selu’ activation function, ‘lecun_normal’ kernel_initializer. The model is designed having 200 neurons each hidden layers with ‘relu’ activation function and (12, none) input shape because there are 12 features. The last layer is the output layer with 4 outputs because there are 0-4 as a total of 5 labels and ‘sigmoid’ activation function.

Complied the model with binary_crossentropy for the loss and report the accuracy. Built the neural network by KerasClassifiers with the model. Then the model is ready for the grid search.

II. Grid search

Imported GridSearchCV from sklearn.model_selection. Sat the list of epochs 100, 200, 300 and 6 different optimizers: rmsprop, nadam, adam, sgd, adamax, and adagrad. Applied grid search with cv = 3. The model that provided the highest accuracy is the model by ‘adam’ optimizer and 100 epochs. The highest accuracy is 0.5241.

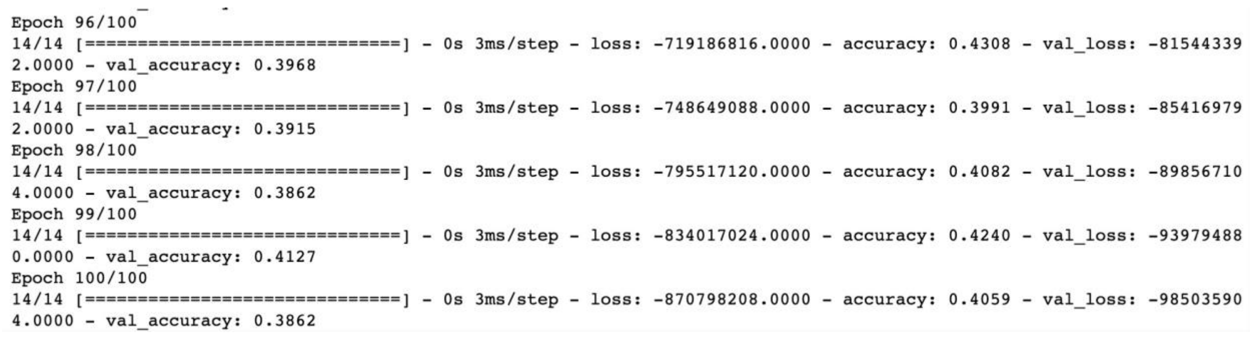


```
GridSearchCV(cv=3,
             estimator=<keras.wrappers.scikit_learn.KerasClassifier object at 0x7ff64ea07b10>,
             param_grid={'epochs': [100, 200, 300],
                         'optimizer': ['rmsprop', 'nadam', 'adam', 'sgd',
                                       'adamax', 'adagrad']})
{'epochs': 100, 'optimizer': 'adam'}
0.524094432592392
```

III. Best model collection

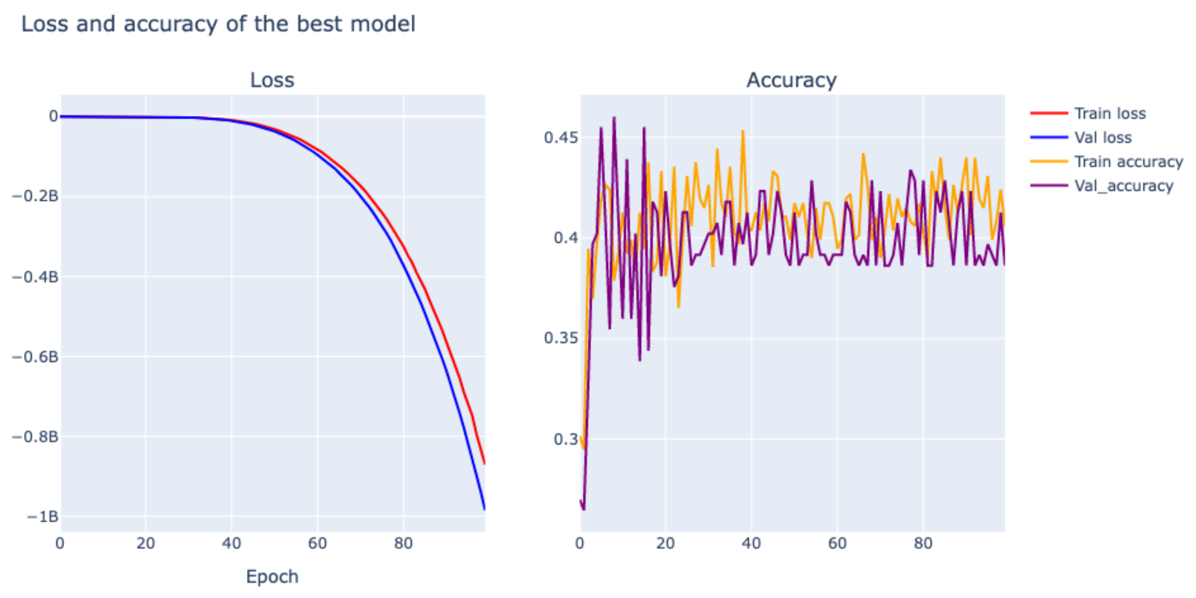
Based on the results from grid search, rebuild the best model that has four hidden layers and 200 neurons each. Input shape is (12, none) and the activation function for the hidden layers is ‘relu’. Complied the model by ‘adam’ optimizers, binary_crossentropy loss function, and accuracy for the metrics. Fit the X_train and Y_train to the model with batch size = 32, epochs = 100 and validation split = 0.3.

The last five epochs are shown the following.



```
Epoch 96/100
14/14 [==============================] - 0s 3ms/step - loss: -719186816.0000 - accuracy: 0.4308 - val_loss: -81544339
2.0000 - val_accuracy: 0.3968
Epoch 97/100
14/14 [==============================] - 0s 3ms/step - loss: -748649088.0000 - accuracy: 0.3991 - val_loss: -85416979
2.0000 - val_accuracy: 0.3915
Epoch 98/100
14/14 [==============================] - 0s 3ms/step - loss: -795517120.0000 - accuracy: 0.4082 - val_loss: -89856710
4.0000 - val_accuracy: 0.3862
Epoch 99/100
14/14 [==============================] - 0s 3ms/step - loss: -834017024.0000 - accuracy: 0.4240 - val_loss: -93979488
0.0000 - val_accuracy: 0.4127
Epoch 100/100
14/14 [==============================] - 0s 3ms/step - loss: -870798208.0000 - accuracy: 0.4059 - val_loss: -98503590
4.0000 - val_accuracy: 0.3862
```

Visualized the loss and the accuracy by plotly.



From above diagram, it is clearly to see that the loss of the model is significantly decreasing after 40 epochs. However, the accuracy went up after the first 5 epochs then changed to unstable. The possible reasons for the unstable accuracy are small dataset and missing values. The model is likely to provide stable result with large dataset and well-done binary data. In this case, using MLP to predict a disease risk need more well-prepared and larger dataset.

IV. Model evaluation

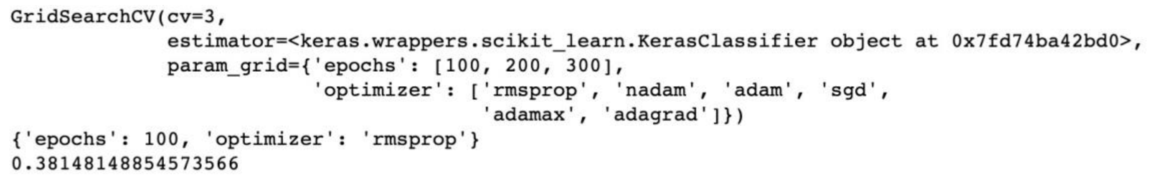
Now, evaluated the model by the test set and received 0.3838 accuracy.

V. MLP with 10 neurons each layer

Next, the same MLP model with 4 hidden layers and have 10 neurons each layer will be used to test the accuracy.

VI. Grid search 2

Applied Grid search for the new model to find out the best model among different number of epochs and different optimizers.



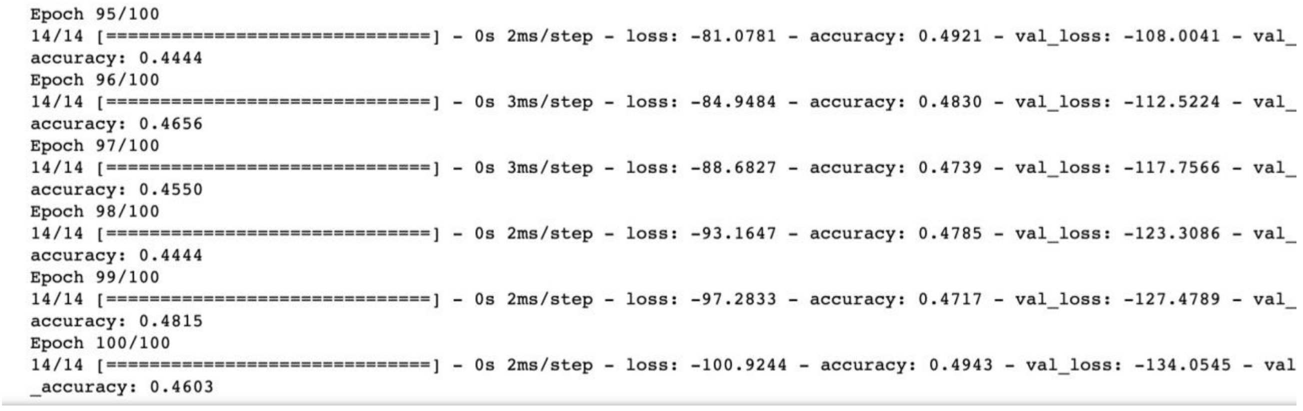
```
GridSearchCV(cv=3,
             estimator=<keras.wrappers.scikit_learn.KerasClassifier object at 0x7fd74ba42bd0>,
             param_grid={'epochs': [100, 200, 300],
                         'optimizer': ['rmsprop', 'nadam', 'adam', 'sgd',
                                       'adamax', 'adagrad']})
{'epochs': 100, 'optimizer': 'rmsprop'}
0.38148148854573566
```

In this case, the best model has four layers and 10 neurons each layer respectively. 100 epochs and rmsprop will give the highest accuracy with was 0.3815.

VII. Best model collection (10 neurons)

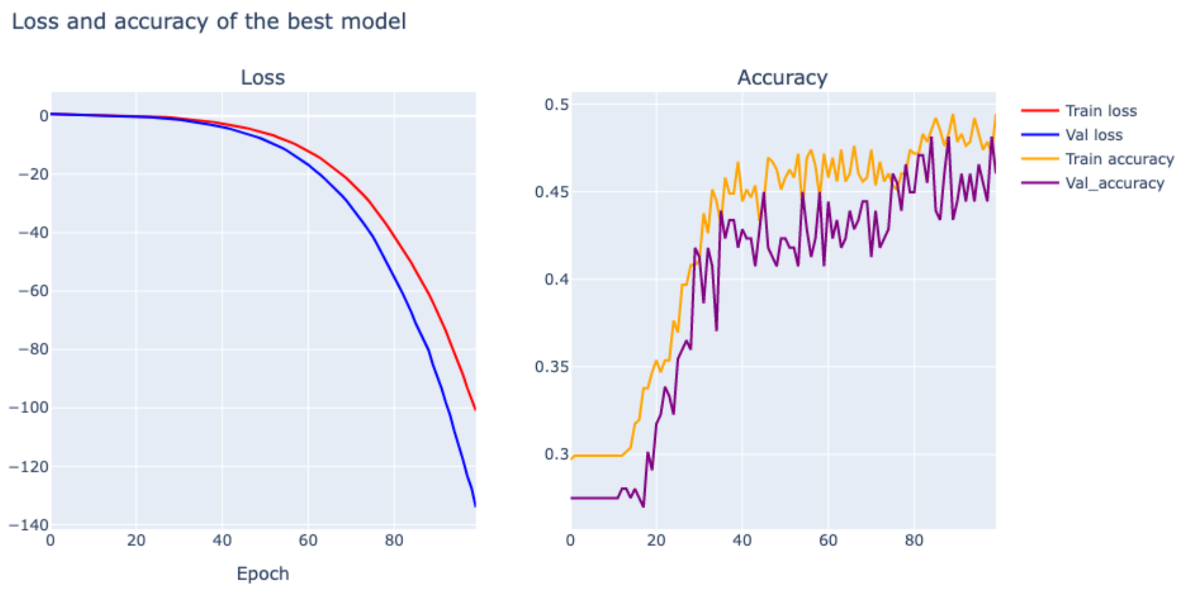
Based on the results from grid search, rebuild the best model that has four hidden layers and 10 neurons each. Input shape is (12, none) and the activation function for the hidden layers is 'relu'. Complied the model by 'rmsprop optimizers, binary_crossentropy loss function, and accuracy for the metrics. Fit the X_train and Y_train to the model with batch size = 32, epochs = 100 and validation split = 0.3.

The last five epochs are shown the following.



```
Epoch 95/100
14/14 [==============================] - 0s 2ms/step - loss: -81.0781 - accuracy: 0.4921 - val_loss: -108.0041 - val_
accuracy: 0.4444
Epoch 96/100
14/14 [==============================] - 0s 3ms/step - loss: -84.9484 - accuracy: 0.4830 - val_loss: -112.5224 - val_
accuracy: 0.4656
Epoch 97/100
14/14 [==============================] - 0s 3ms/step - loss: -88.6827 - accuracy: 0.4739 - val_loss: -117.7566 - val_
accuracy: 0.4550
Epoch 98/100
14/14 [==============================] - 0s 2ms/step - loss: -93.1647 - accuracy: 0.4785 - val_loss: -123.3086 - val_
accuracy: 0.4444
Epoch 99/100
14/14 [==============================] - 0s 2ms/step - loss: -97.2833 - accuracy: 0.4717 - val_loss: -127.4789 - val_
accuracy: 0.4815
Epoch 100/100
14/14 [==============================] - 0s 2ms/step - loss: -100.9244 - accuracy: 0.4943 - val_loss: -134.0545 - val
_accuracy: 0.4603
```

Visualized the loss and the accuracy by plotly.



Changing the neurons of each layer from 200 to 10 rolled out a big different. The loss of the model remained the same but the accuracy is stably going up. Not like the model with 200 neurons. MLP with 10 neurons performed very stable for the prediction. On the other hand, if the data set got larger and prepared better. MLP with 10 neurons each layer is more reliable than MLP with 200 neurons because the accuracy performance is stable.

VIII. Model evaluation

Now, evaluated the model by the test set and received 0.4111 accuracy.

Classifiers

I. Logistic regression

Imported logistic regression model from sklearn.linear_model. Fitted the model for the training set and compute the correct rate.

The logistic regression model rolled out 0.4481.

II. Gaussian Naïve Bayes

Imported GaussianNB from sklearn.naive_bayes. Fitted the model for the training set and compute the correct rate.

The Gaussian Naïve Bayes model rolled out 0.4667.

III. Gaussian SVM

Imported svm from sklearn. Fitted the training set and compute the correct rate.

The Gaussian Naïve Bayes model rolled out 0.5148.

Conclusion

After 100 epochs, MLP provided 38.38% accuracy. Logistic regression provided 44.81%, Gaussian Naïve Bayes provided 46.67%, and Gaussian SVM provided 51.48% accuracy. It is clear to see that Gaussian SVM provided the highest accuracy and MLP neural network provided the least accuracy. However, it does not mean that neural network is not as reliable as classifiers.

There are two possible reasons why MLP did not perform better than classifiers. First of all, the data is linear. ANN have the ability to learn and model non-linear and complex relationship and neural networks are good to model with nonlinear data with large number of inputs. On the other hand, regression models and classifiers would perform better on linear data since these models are based on statistical model.

Secondly, the data set is not high-dimensional. Neural networks are best for situations where the data is high-dimensional like images. The data set used is a regular linear data set. So, the shape of the data set is perfectly fit the regression models and classifiers.

In conclusion, back to the research question, two conditions would be the best for neural network. Neural network would perform the best for the high dimensional data or non-linear data.

Reference:

Data set:
http://archive.ics.uci.edu/ml/datasets/Heart+Disease

The authors of the databases have requested that any publications resulting from the use of the data include the names of the principal investigator responsible for the data collection at each institution. They would be:

1. Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D.
2. University Hospital, Zurich, Switzerland: William Steinbrunn, M.D.
3. University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D.
4. V.A. Medical Center, Long Beach and Cleveland Clinic Foundation:Robert Detrano, M.D., Ph.D.

https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781788397872/1/ch01lvl1sec27/pros-and-cons-of-neural-networks#:~:text=Neural%20networks%20are%20good%20to,layered%20network%20of%20simpler%20elements.