HACETTEPE UNIVERSITY
ENGINEERING FACULTY
DEPARTMENT OF COMPUTER ENGINEERING

BBM 407 - Fuzzy Logic

Adaptive Neuro Fuzzy Inference System

Sena Çakırlı
2200356074

TABLE OF CONTENTS

Libraries

```
import os
import glob
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import missingno as msno
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,f1_score,recall_score,precision_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay
from sklearn.metrics import r2_score
from sklearn.preprocessing import StandardScaler
import tensorflow.compat.v1 as tf
tf.disable_v2_behavior()
```

Data Set

I took 500 person datasets from the sepsis file and 500 person datasets from the no sepsis file, then took mean in every column for all person datasets and combined them in a single dataset with 1000 rows. Then I checked all the features with any missing value and switched them with the means of the columns. Lastly, I splitted the dataset to train and test sets using sklearn.model_selection.train_test_split() according to 5 fold. Since I shuffled the dataset randomly before, how much sepsis and no sepsis people data does train and test sets have was also random. After checking we see the results as 402 no sepsis, 398 sepsis people data for the train set and 98 no sepsis, 102 sepsis people data for the testset.

```
print('X_train.shape:  ', X_train.shape)
print('y_train.shape:  ', y_train.shape)
unique, counts = np.unique(y_train, return_counts=True)
print("Train:  ", dict(zip(unique, counts)))

print('X_test.shape:  ', X_test.shape)
print('y_test.shape:  ', y_test.shape)
unique, counts = np.unique(y_test, return_counts=True)
print("Test:  ", dict(zip(unique, counts)))
```

```
X_train.shape:   (800, 7)
y_train.shape:   (800,)
Train:   {0.0: 402, 1.0: 398}
X_test.shape:   (200, 7)
y_test.shape:   (200,)
Test:   {0.0: 98, 1.0: 102}
```

Coefficients Between Features

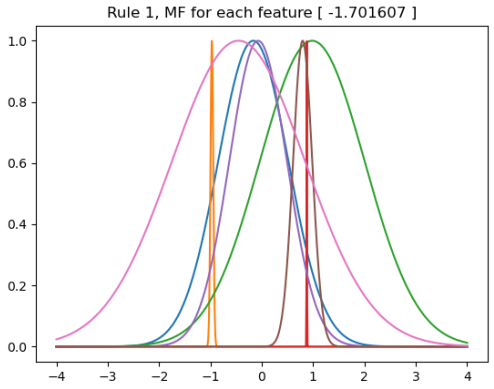
```
# R-square Values Between Features

features = ['Heart Rate', 'MAP', 'Bilirubin', 'Creatinine', 'Lactate', 'Platelets', 'GCS']

for first in range(X.shape[1]):
    for second in range(X.shape[1]):
        R_square = r2_score(X[first], X[second])
        print('Coefficient of Determination Between', features[first], 'and', features[second], ':   ', R_square)
```

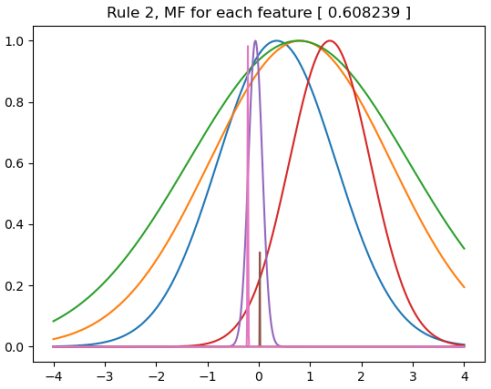
```
Coefficient of Determination Between Heart Rate and Heart Rate :    1.0
Coefficient of Determination Between Heart Rate and MAP :    -1.7087243943520627
Coefficient of Determination Between Heart Rate and Bilirubin :    -0.7169613962215802
Coefficient of Determination Between Heart Rate and Creatinine :    -3.9175218684603728
Coefficient of Determination Between Heart Rate and Lactate :    -1.110752716029785
Coefficient of Determination Between Heart Rate and Platelets :    -1.3727054008596582
Coefficient of Determination Between Heart Rate and GCS :    -0.6448865921167295
Coefficient of Determination Between MAP and Heart Rate :    -4.329751389589482
Coefficient of Determination Between MAP and MAP :    1.0
Coefficient of Determination Between MAP and Bilirubin :    -1.6068933899341609
Coefficient of Determination Between MAP and Creatinine :    -6.023456619995701
Coefficient of Determination Between MAP and Lactate :    -0.39064405609906316
Coefficient of Determination Between MAP and Platelets :    -1.7294410162032197
Coefficient of Determination Between MAP and GCS :    -1.7818552866116497

Coefficient of Determination Between Bilirubin and Heart Rate :    -3.545378603460467
Coefficient of Determination Between Bilirubin and MAP :    -2.5074433950478525
Coefficient of Determination Between Bilirubin and Bilirubin :    1.0
Coefficient of Determination Between Bilirubin and Creatinine :    -9.130109311435447
Coefficient of Determination Between Bilirubin and Lactate :    -4.195220342981038
Coefficient of Determination Between Bilirubin and Platelets :    -2.699131209483387
Coefficient of Determination Between Bilirubin and GCS :    -4.6799500380927945
Coefficient of Determination Between Creatinine and Heart Rate :    -0.8075563263197711
Coefficient of Determination Between Creatinine and MAP :    -0.3120618603165237
Coefficient of Determination Between Creatinine and Bilirubin :    -0.4065336889435107
Coefficient of Determination Between Creatinine and Creatinine :    1.0
Coefficient of Determination Between Creatinine and Lactate :    0.19902146234805107
Coefficient of Determination Between Creatinine and Platelets :    0.0019685123493482504
Coefficient of Determination Between Creatinine and GCS :    -1.0596947460743733
Coefficient of Determination Between Lactate and Heart Rate :    -2.004723860222525
Coefficient of Determination Between Lactate and MAP :    -0.0060997943783511754
Coefficient of Determination Between Lactate and Bilirubin :    -1.7935834439504807
Coefficient of Determination Between Lactate and Creatinine :    -2.1020053936085143
Coefficient of Determination Between Lactate and Lactate :    1.0
Coefficient of Determination Between Lactate and Platelets :    -0.48180576764125727
Coefficient of Determination Between Lactate and GCS :    -0.8141867945501609

Coefficient of Determination Between Platelets and Heart Rate :    -1.3207631515804836
Coefficient of Determination Between Platelets and MAP :    -0.35680858366092627
Coefficient of Determination Between Platelets and Bilirubin :    -0.3667126211995142
Coefficient of Determination Between Platelets and Creatinine :    -1.6557409197401807
Coefficient of Determination Between Platelets and Lactate :    -0.018148377145474015
Coefficient of Determination Between Platelets and Platelets :    1.0
Coefficient of Determination Between Platelets and GCS :    -2.1221207592953966
Coefficient of Determination Between GCS and Heart Rate :    -0.36446992217078367
Coefficient of Determination Between GCS and MAP :    -0.1727904142467489
Coefficient of Determination Between GCS and Bilirubin :    -0.7797663411198275
Coefficient of Determination Between GCS and Creatinine :    -3.6482055804932028
Coefficient of Determination Between GCS and Lactate :    -0.05716505421419371
Coefficient of Determination Between GCS and Platelets :    -1.6478336778122022
Coefficient of Determination Between GCS and GCS :    1.0
```

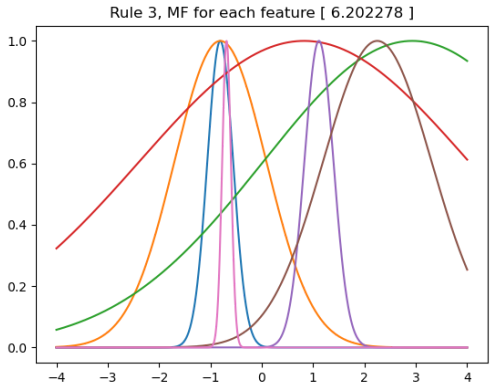
The ANFIS Model

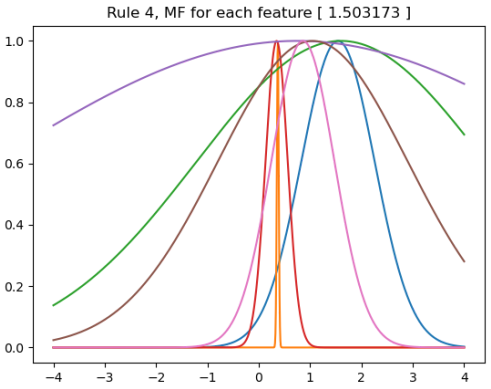
For training and testing the dataset I used an ANFIS model I found online designed for diabetes prediction[1]. I set the total number of fuzzy rules used in the model to 14. I used binary cross entropy for the loss function, adam optimizer with learning parameter=0.01 for the loss function and gaussian membership function for each fuzzy rule. I set the number of epochs to 1000 and printed out the results at every 200 epochs.

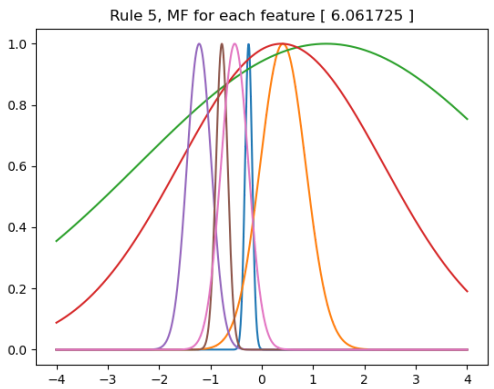
Rules

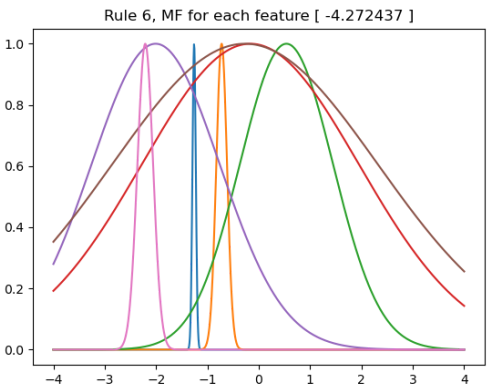

Rule 1, MF for each feature [-1.701607]

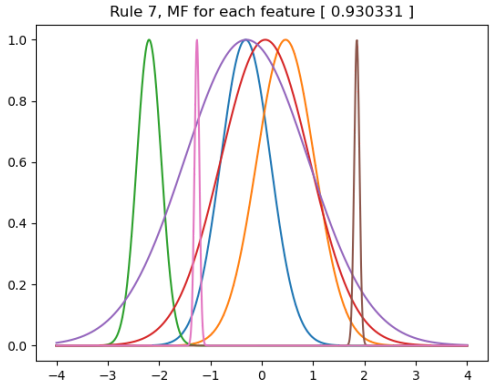

Rule 2, MF for each feature [0.608239]

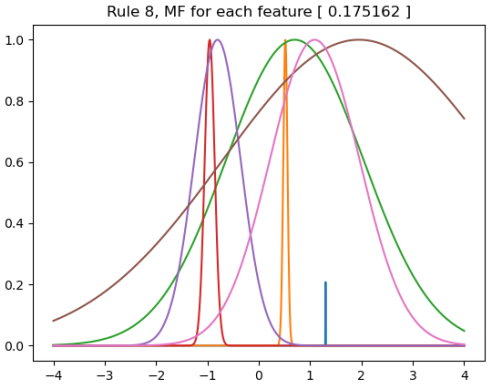

Rule 3, MF for each feature [6.202278]

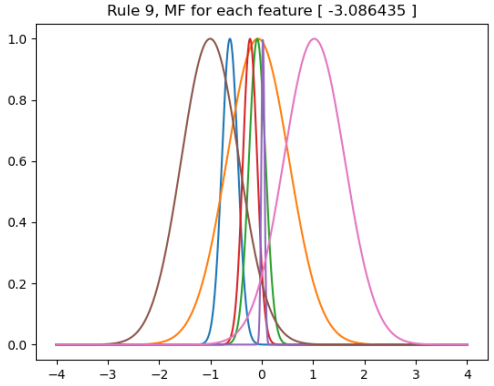

Rule 4, MF for each feature [1.503173]

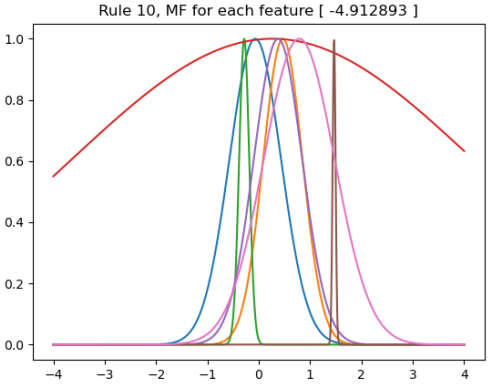

Rule 5, MF for each feature [6.061725]

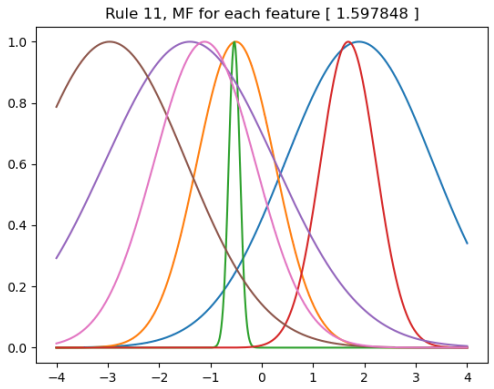

Rule 6, MF for each feature [-4.272437]

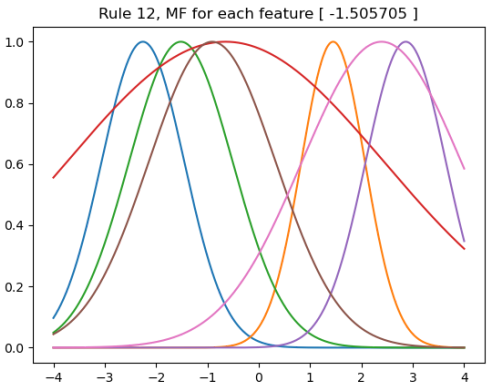

Rule 7, MF for each feature [0.930331]

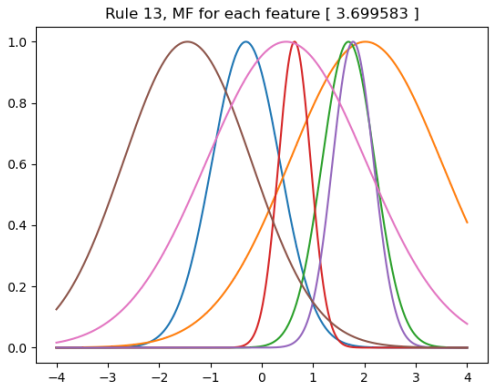

Rule 8, MF for each feature [0.175162]

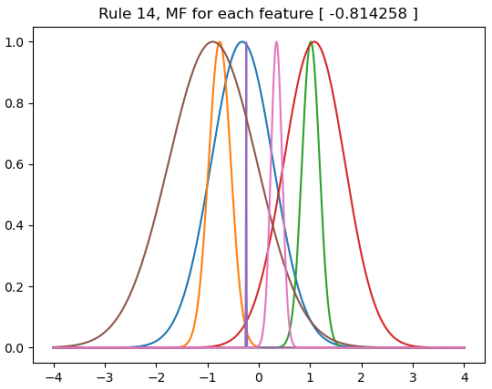

Rule 9, MF for each feature [-3.086435]


Rule 10, MF for each feature [-4.912893]

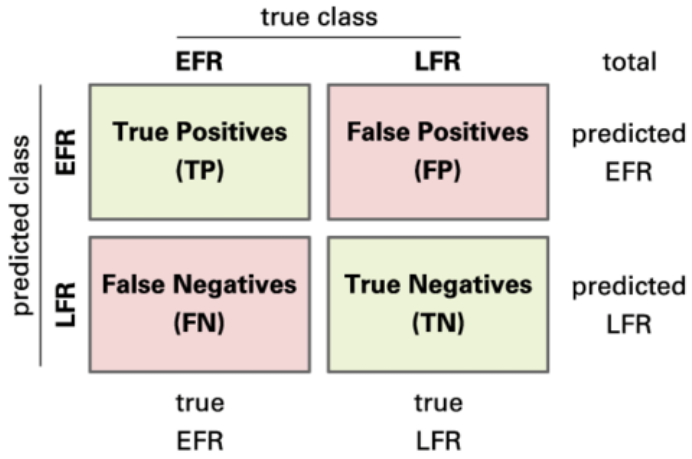

Rule 11, MF for each feature [1.597848]


Rule 12, MF for each feature [-1.505705]


Rule 13, MF for each feature [3.699583]


Rule 14, MF for each feature [-0.814258]

Results



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

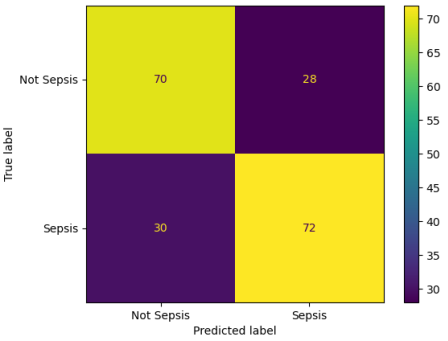
$$RE = \frac{TP}{TP+FN}$$

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

$$F_1 = \frac{2TP}{2TP+FP+FN}$$

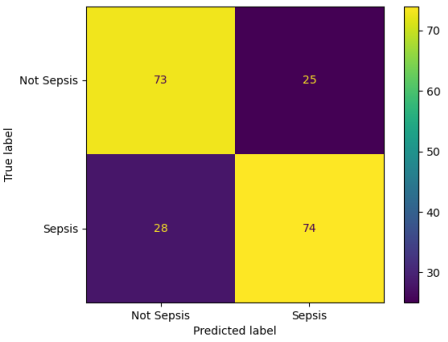
Epoch 200

```
Epoch  200 >>>>>>>>>>>>>
loss       >>> test: 0.59845376           train: 0.53007364
accuracy   >>> test: 0.71         train: 0.7525
f1-score   >>> test: 0.712871287128713    train: 0.736
precision >>> test: 0.72         train: 0.7840909090909091
recall     >>> test: 0.7058823529411765   train: 0.6934673366834171
```



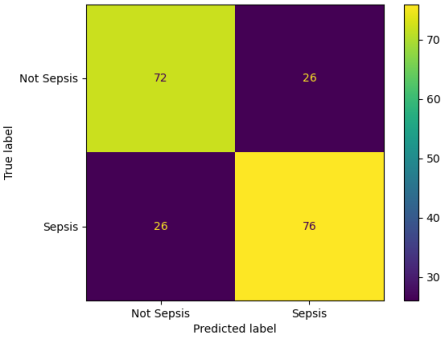
Epoch 400

```
Epoch  400 >>>>>>>>>>>>>
loss       >>> test: 0.626671             train: 0.4858874
accuracy   >>> test: 0.735        train: 0.76125
f1-score   >>> test: 0.736318407960199    train: 0.7490144546649148
precision >>> test: 0.7474747474747475   train: 0.7851239669421488
recall     >>> test: 0.7254901960784313   train: 0.7160804020100503
```



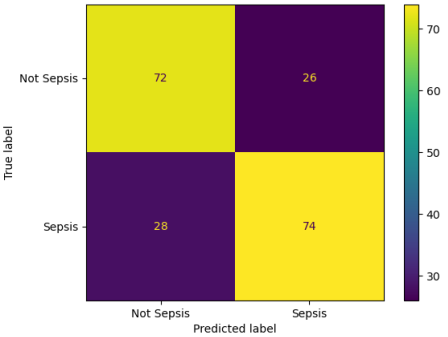
Epoch 600

```
Epoch  600 >>>>>>>>>>>>>>
loss       >>> test: 0.6521392            train: 0.4739498
accuracy   >>> test: 0.74          train: 0.775
f1-score   >>> test: 0.7450980392156863   train: 0.7662337662337662
precision >>> test: 0.7450980392156863   train: 0.793010752688172
recall     >>> test: 0.7450980392156863   train: 0.7412060301507538
```



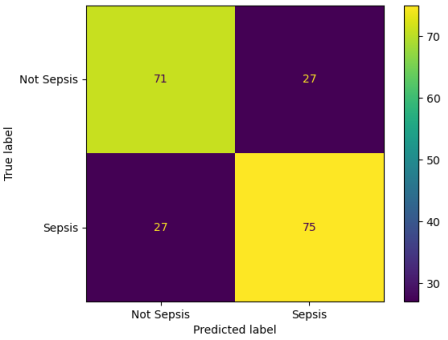
Epoch 800

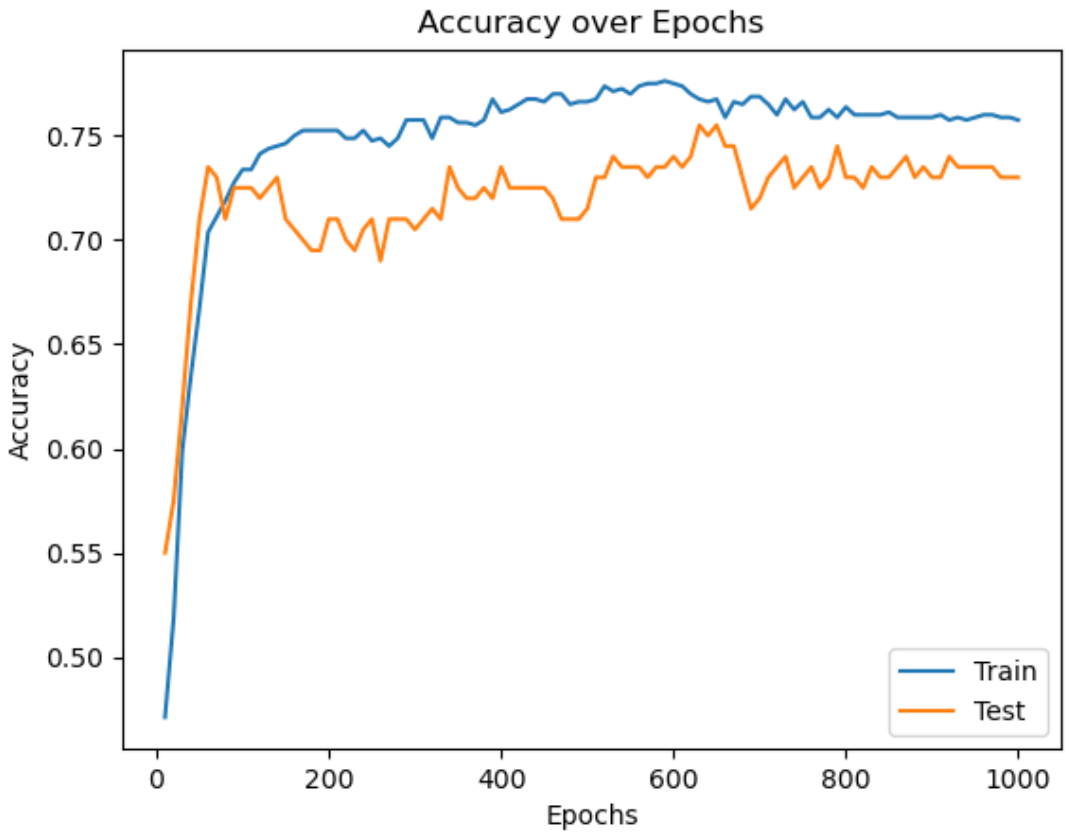
```
Epoch  800 >>>>>>>>>>>>>>
loss       >>> test: 0.6677343            train: 0.45597136
accuracy   >>> test: 0.73          train: 0.76375
f1-score   >>> test: 0.7326732673267328   train: 0.7573812580231066
precision >>> test: 0.74          train: 0.7742782152230971
recall     >>> test: 0.7254901960784313   train: 0.7412060301507538
```

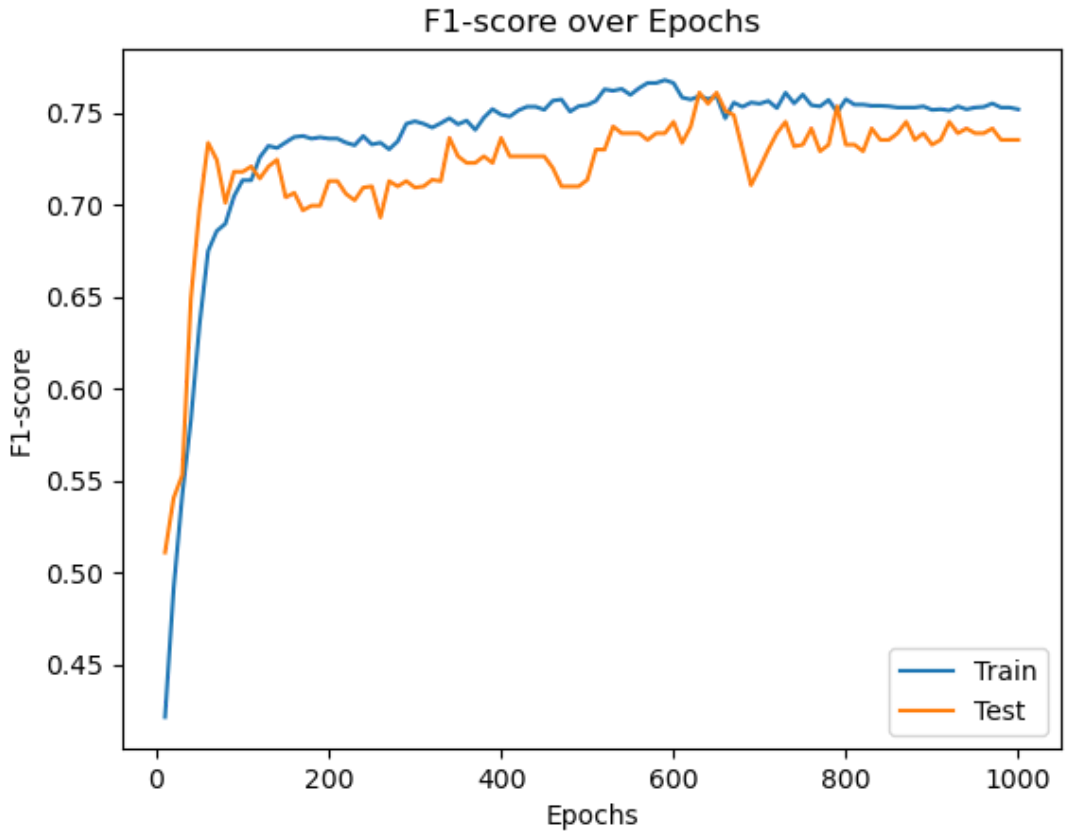


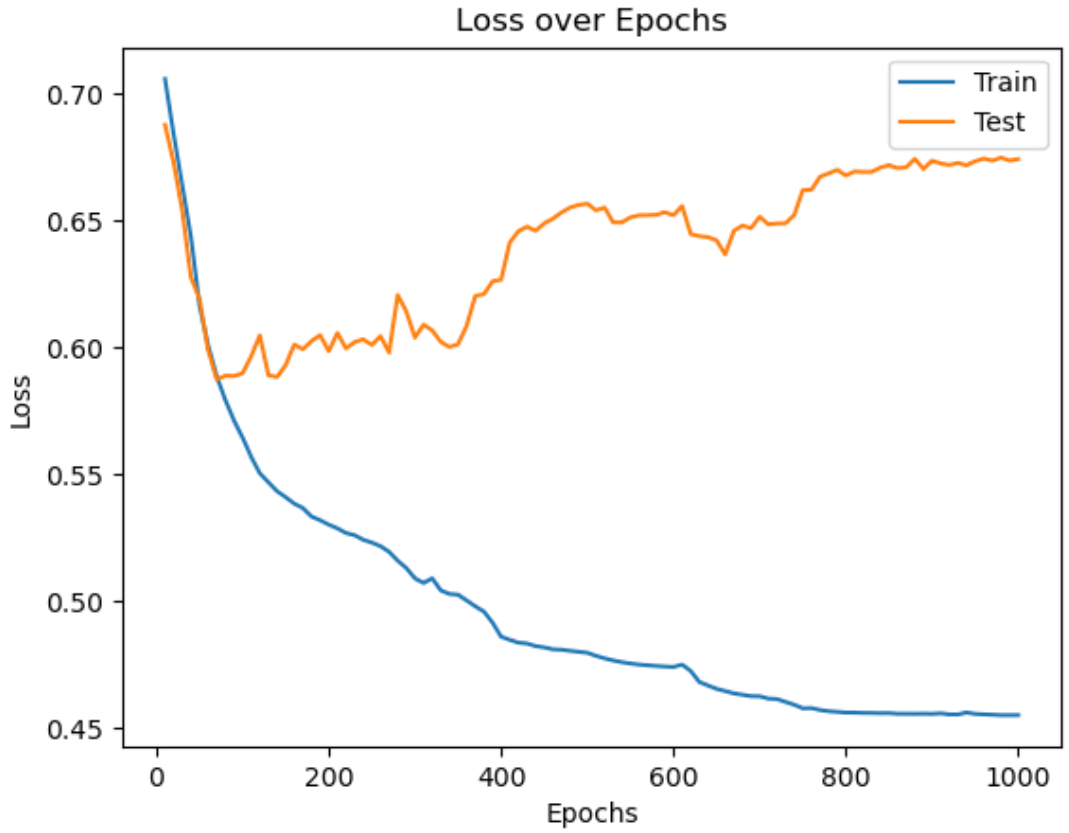
Epoch 1000

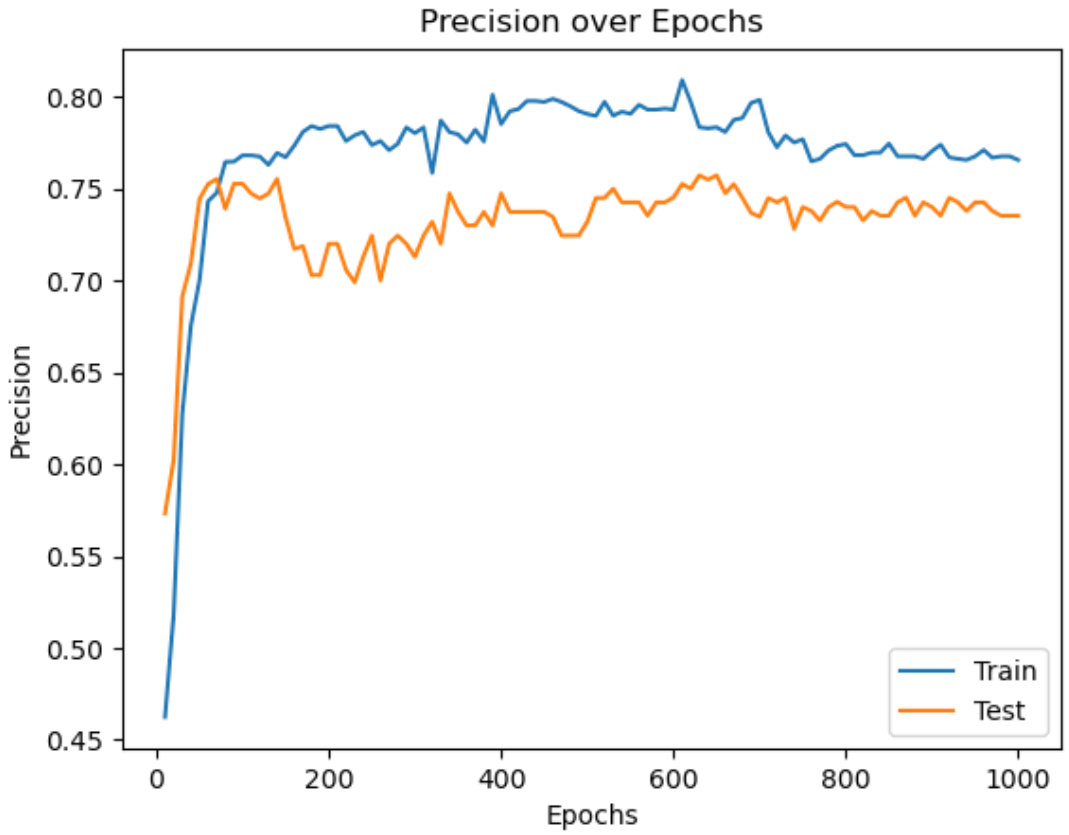
```
Epoch  1000 >>>>>>>>>>>>>>
loss       >>> test: 0.67423433           train: 0.45495486
accuracy   >>> test: 0.73          train: 0.7575
f1-score   >>> test: 0.735294117647059    train: 0.751918158567775
precision >>> test: 0.7352941176470589   train: 0.765625
recall     >>> test: 0.7352941176470589   train: 0.7386934673366834
```

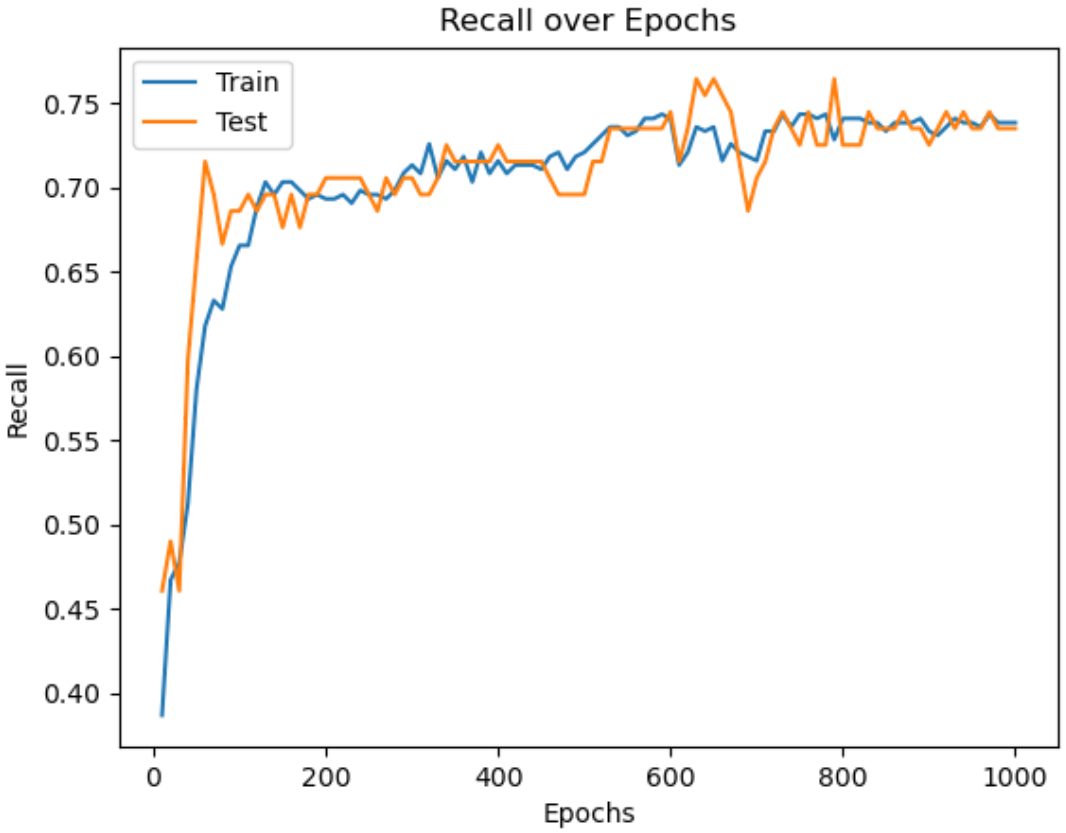



Accuracy over Epochs
Accuracy
0.75
0.70
0.65
0.60
0.55
0.50
0
200
400
600
800
1000
Epochs
Train
Test


F1-score over Epochs
F1-score
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0
200
400
600
800
1000
Epochs
Train
Test


Loss over Epochs
Train
Test
0.70
0.65
0.60
0.55
0.50
0.45
Loss
0
200
400
600
800
1000
Epochs


Precision over Epochs
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45
Precision
Train
Test
0
200
400
600
800
1000
Epochs



References

[1] “ANFIS Model”. https://github.com/AgHarsh/ANFIS-Diabetes-Predictor.