~\Documents\documents_general\structured_courses\math564\evaluations\projects \p04\solve4.py

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
script for solving the liver disease project
Version: Aug  14, 2023
Author: Tom Asaki
"""

import numpy as np
import optimize as opt
import pandas as pd               # type: ignore

# read in raw data, drop rows with any missing data
B=pd.read_csv('liver.csv').dropna().to_numpy()
r,c=B.shape
# set 'Male' to 0 and 'Female' to 1
for k in range(r):
    B[k,1]=(len(B[k,1])-4)/2
A=np.zeros((r,c))
for k in range(c):
    A[:,k]=B[:,k]


# data class vector
data_class=A[:,c-1]-1

# normalize data values to be 0 <= val <= 1
data_values=A[:,0:c-1]
maxval=np.array(list(max(data_values[:,k]) for k in range(c-1)))
minval=np.array(list(min(data_values[:,k]) for k in range(c-1)))
data_values=(data_values-minval)/(maxval-minval)

# separate into training and testing data
numtrain=200
idx=np.random.permutation(r)
idx=np.arange(200)
train_data=data_values[idx[0:numtrain],]
train_class=np.array([data_class[idx[0:numtrain]]]).reshape(-1,1)
test_data=data_values[idx[numtrain:r],]
test_class=np.array([data_class[idx[numtrain:r]]]).reshape(-1,1)

# set up optimization problem
#  training data  corresponding class           layer sizes     task
p=[train_data.T , (train_class.T)*(1/3)+(1/3), [10,10,10,1] , 'train']

# set up initial weights
sz=np.sqrt(p[2][0])
NumWeights=0
for k in range(len(p[2])-1):
    NumWeights+=p[2][k]*p[2][k+1]
x0=sz*np.random.randn(NumWeights).reshape((NumWeights,1))
```

```python
from objective import nnloss as obj

alg=dict(obj        = obj,
         x0         = x0,
         params     = p,
         method     = 'LBFGS',
         maxiter    = 20000,
         ngtol      = 1E-10,
         dftol      = 1E-10,
         dxtol      = 1E-10,
         Lambda     = 1,
         Lambdamax  = 100,
         linesearch = 'StrongWolfe',
         c1         = 0.0001,
         c2         = 0.9,
         progress   = 1000
         )

res=opt.minimize(alg)

alg['params'][3]='classify'
fitclass=obj(res['x'][:,[-1,]],p,1)

comp=np.abs(np.round(fitclass)-train_class.T)
print('Training Misclassification Rate = %3.1f%%' % (np.sum(comp)/np.size(comp)*100))
```