Coronary Heart Disease Dataset

Hyperparameters

```
layers of the network : 3
number of neurons in each layer : (9, 32, 2)
epochs = 12
mini_batch_size = 10
eta(learning rate) = 1
```

Result

```
Initial performance : 58 / 100
Epoch 0 : 72 / 100
Epoch 1 : 71 / 100
Epoch 2 : 78 / 100
Epoch 3 : 76 / 100
Epoch 4 : 71 / 100
Epoch 5 : 75 / 100
Epoch 6 : 75 / 100
Epoch 7 : 63 / 100
Epoch 8 : 62 / 100
Epoch 9 : 78 / 100
Epoch 10 : 75 / 100
Epoch 11 : 76 / 100

the time required to train the net : 0.25 s
```

Data process

```
HeartData = []

### first I load data to a list
with open(filename, newline='') as f:
    reader = csv.reader(f)
    for row in reader : HeartData.append(row[1:])

### divide header and data
Header = HeartData[0]
HeartData = HeartData[1:]

### convert famhist to bool type
### convert string to int/float
for item in HeartData:
    for i in range(len(Header)):
        if Header[i] == 'famhist':
            item[i] = 1 if item[i] == 'Present' else 0

        elif Header[i] == 'chd':
            item[i] = int(item[i])
        else:
            item[i] = float(item[i])
```

```
### Get the len of dataset
n = len(HeartData)

### Convert list to np.array
HeartData = np.array(HeartData)

### divide dataset to features and labels
features = HeartData[:, 0 : -1]
labels = HeartData[:, -1]

### rescale the age to have maximum value 1.
features[:, 8] = features[:, 8] / features[:, 8].max()

### convert other variables to z-scores
for i in range(9):
    if i == 4 or i == 8 : continue
    features[:, i] = standardize(features[:, i], features[:, i].mean(),
features.std())
```

Change & Reason

```
I change the number of neurons of hidden layer and learning rate and epochs.
Since this data set is small, I adjusted the learning rate to 1, which allows the
network to quickly converge to its optimal parameters.
Using more hidden layer neurons can improve the learning ability of the neural
network so I change the number of hidden layer.
```