```
W = 0
M = 1


trainData = [ ((170, 57, 32), W),
((190, 95, 28), M),
((150, 45, 35),  W),
((168, 65, 29), M),
((175, 78, 26), M),
((185, 90, 32), M),
((171, 65, 28), W),
((155, 48, 31), W),
((165, 60, 27), W),
((182, 80, 30), M),
((175, 69, 28), W),
((178, 80, 27), M),
((160, 50, 31), W),
((170, 72, 30), M), ]

testX = [((155,40,35), W),
 ((170,70,32), M),
 ((175,70,35), W),
 ((180,90,20),M)]
```

```
from random import randrange
from math import exp
from random import seed
import pdb

PRECISION = 1000

import pprint
from pprint import pprint
# Find the min and max values for each column
def dataset_minmax(dataset):
  minmax = list()
  for col in range(len(dataset[0])):
    col_values = [dataset[row][col] for row in range(len(dataset))]
    value_min = min(col_values)
    value_max = max(col_values)
    minmax.append([value_min, value_max])
  return minmax

# Rescale dataset columns to the range 0-1
def normalize_dataset(dataset, minmax):
  ndataset = list()
  for row in dataset:
```

```
    nrow = list()
    for i in range(len(row)):
      new_i = (row[i] - minmax[i][0]) / (minmax[i][1] - minmax[i][0])
      if new_i < 0.01: new_i = 0.01
      nrow.append(new_i)
    ndataset.append(nrow)
  #pdb.set_trace()
  return ndataset

def get_acc(groundtruth, prediction):
  correct = 0
  for i in range(len(groundtruth)):
    if groundtruth[i] == prediction[i]:
      correct += 1
  return correct / float(len(groundtruth)) * 100.0

def sgd(X, Y, alpha, n):
  theta = [0.0 for i in range(len(X[0]))]
  for _ in range(n):
    for i in range(len(X)):
      Y_ = hypothesis(X[i], theta)
      #pdb.set_trace()
      e = Y[i] - Y_
      theta[0] = theta[0] + alpha * e * Y_ * (1.0 - Y_)
      for m in range(len(X[i])-1):
        #pdb.set_trace()
        theta[m + 1] = theta[m + 1] + alpha * e * Y_ * (1.0 - Y_) *
X[i][m]
  return theta

def hypothesis(X, theta):
  y_ = theta[0]
  for i in range(len(X)-1):
    y_ += theta[i+1] * X[i]
  return softmax(y_)

def softmax(y_):
  return 1.0/(1.0 + exp(-y_))

def rsig(y_):
  return round(y_ * PRECISION)/PRECISION

def prepare_data(dataset):
  X = list()
  Y = list()
  for xy in dataset:
    X.append(list(xy[0]))
    Y.append(xy[1])
  for i in range(len(Y)):
    if Y[i] == M:
      Y[i] = 1
    else: Y[i] = 0
  return X, Y
```

```
if __name__ == '__main__':

  n = 100
  alpha = .1
  X, Y = prepare_data(trainData)
  X = normalize_dataset(X, dataset_minmax(X))
  Xt, Yt  = prepare_data(testX)
  predictions = list()
  theta = sgd(X, Y, alpha, n)

  for x in Xt:
    y_ = hypothesis(x, theta)
    y_ = rsig(y_)
    predictions.append(y_)

  acc = get_acc(Y,y_)
  print("acc: ", acc)
```