sudo code

```
# define: 1) each face image would be 60*70 (X*Y)
          2) each digit image would be 28*28 (X*Y)
          3) we divide each face image into 42*(6*7) small regions (call it o)
          4) labels[i]: 1 or 0, indicates face or not face for the image[i]
          5) data_regions[i][j]: integer, indicates number of '#' in the region j(0-41) for
                    the image[i]
```

perceptron sudo code:

```
perceptron(g, w, f, bias):
# a loop multiplying each weight with corresponding o value
for (i in range 42)
        f = f + w[i] * g[i]

# add bias after the loop
f = f + bias

return
end of perceptron



main:
# g is a list holds number of '#' in a given region of an image
g = []

# w is the weight for corresponding g value
# initially we assign random number to each w
w = []
for(i in range 42)
        w[i] = uniform(-1, 1)

# f(x), indicator of our prediction. < 0 means our model predict it is not a face while >= 0
# means our model predict it is face
f = 0

# bias
bias = uniform(-1, 1)

# gather input indicators
type, percent, algorithm

# gather the wanted data set
labels = training_labels(type, float(percent))
data_regions = training_data(type, float(percent))
```

```
# loop
for (i in range (every single image in the percentage of data we want to use) )
        # the range would just be len(labels)
        g = data_region[i]
        perceptron(g, w, f, bias)

        # if we predict it right, move on. Otherwise do the penalty to w
        if f >= 0 and label[i] == 0:
           for(i in range 42)
              w[i] = w[i] - g[i]
              bias = bias - 1

        elif f < 0 and label[i] == 1:
           for(i in range 42)
              w[i] = w[i] + g[i]
              bias = bias + 1
```

end of main