A Quick Introduction to Machine Learning (Scaling)

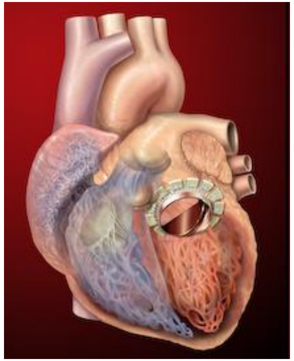
Lecturer: John Guttag

An Example

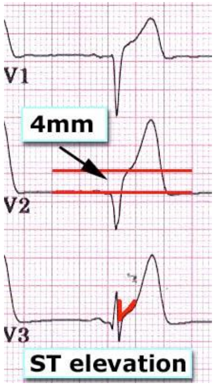
OGL Image Courtesy of
LA(Phot) Stuart Hill/MOD

1000 patients with 4 features each

Heart rate in beats per minute
Number of past heart attacks
ST segment elevation (binary)
Age



(modified) CC-BY Image Courtesy of Patrick J. Lynch



Binary outcome based on features

Probabilistic, not deterministic
Roughly 31% positive

A Sampling of Examples

```
P0755:[ 48.   1.   0.   62.]:1
P0383:[ 103.  1.   1.   99.]:1
P0849:[ 42.   1.   1.   92.]:1
P0188:[ 71.   2.   0.   58.]:0
P0061:[ 87.   1.   0.   79.]:0
P0196:[ 52.   0.   0.   85.]:0
P0280:[ 78.   0.   0.   81.]:0
P0178:[ 50.   1.   0.   59.]:1
P0497:[ 80.   0.   0.   58.]:0
P0742:[ 78.   2.   0.   72.]:1
P0527:[ 78.   1.   0.   60.]:0
P0915:[ 60.   2.   0.   57.]:1
```

Fraction of positives in total population of 1000 was 0.312

Cluster Using K-means (k = 3)

Fraction of positives in population = 0.312

Ran k-means 100 times and chose best clustering

Cluster of size 354 with fraction of positives = 0.338 (1.08x)
Cluster of size 322 with fraction of positives = 0.315 (1.01x)
Cluster of size 324 with fraction of positives = 0.281 (0.90x)

What Happened?

Features have very different means and variance

- Heart rate in beats per minute ($\mu = 70$, $\sigma = 15$)
- Number of past heart attacks ($\mu = 0.25$, $\sigma = 1.0$)
- ST segment elevation (binary)
- Age ($\mu = 65$, $\sigma = 15$)

HR and age have higher means & greater dynamic range

- Euclidean distance will be biased towards them

Does this seem like a good idea?

“No, No, a Thousand Times No”

Rescale Features

Each variable has same mean and variance

$$x' = \frac{x - \mu_x}{\sigma_x}$$

```
def scaleFeatures(vals):
    vals = pylab.array(vals)
    mean = sum(vals)/float(len(vals))
    sd = stdDev(vals)
    vals = vals - mean
    return vals/sd
```

What is the new mean?
What is the new standard deviation?

Testing Scaling

```
def testScaling(n, mean, std):
    vals = []
    for i in range(n):
        vals.append(int(random.gauss(mean, std)))
    print 'original values', vals
    sVals = scaleAttrs(vals)
    print '\n', 'scaled values', sVals
    print '\n', 'new mean =', sum(sVals/len(vals))
    print '\n', 'new sd =', stdDev(sVals)

testScaling(10, 25, 3)
```

Testing Scaling

```
original values [22, 24, 21, 26, 18, 26, 22, 26,
27, 21]

scaled values [-0.46517657  0.25047969 -0.82300471
0.96613596 -1.89648911  0.96613596
 -0.46517657  0.96613596  1.32396409 -0.82300471]

new mean = -2.22044604925e-16

new sd = 1.0
```

The Real Test

Fraction of positives = 0.312

Clustering with unscaled features

Cluster of size 354 with fraction of positives = 0.338 (1.08x)
Cluster of size 322 with fraction of positives = 0.315 (1.01x)
Cluster of size 324 with fraction of positives = 0.281 (0.90x)

Clustering with scaled features

Cluster of size 324 with fraction of positives = 0.055 (0.18x)
Cluster of size 108 with fraction of positives = 0.335 (1.07x)
Cluster of size 568 with fraction of positives = 0.454 (1.45x)