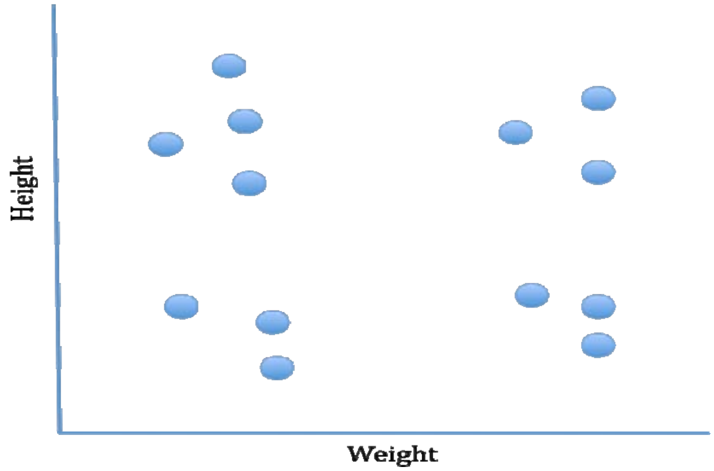
6.00 Handout, Lecture 19
(Not intended to make sense outside of lecture)

```
def dToB(n, numDigits):
    """requires: n is a natural number less than 2**numDigits
      returns a binary string of length numDigits representing the
              the decimal number n."""
    assert type(n)==int and type(numDigits)==int and n >=0 and n <
2**numDigits
    bStr = ''
    while n > 0:
        bStr = str(n % 2) + bStr
        n = n//2
    while numDigits - len(bStr) > 0:
        bStr = '0' + bStr
    return bStr

def genPset(Items):
    """Generate a list of lists representing the power set of Items"""
    numSubsets = 2**len(Items)
    templates = []
    for i in range(numSubsets):
        templates.append(dToB(i, len(Items)))
    pset = []
    for t in templates:
        elem = []
        for j in range(len(t)):
            if t[j] == '1':
                elem.append(Items[j])
        pset.append(elem)
    return pset

def chooseBest(pset, constraint, getVal, getWeight):
    bestVal = 0.0
    bestSet = None
    for Items in pset:
        ItemsVal = 0.0
        ItemsWeight = 0.0
        for item in Items:
            ItemsVal += getVal(item)
            ItemsWeight += getWeight(item)
        if ItemsWeight <= constraint and ItemsVal > bestVal:
            bestVal = ItemsVal
            bestSet = Items
    return (bestSet, bestVal)

def testBest():
    Items = buildItems()
    pset = genPset(Items)
    taken, val = chooseBest(pset, 20, Item.getValue, Item.getWeight)
    print ('Total value of items taken = ' + str(val))
    for item in taken:
        print '  ', item
```



$$\mathrm{var}\,iance(c) = \sum_{x \in c} (mean(c) - x)^2$$

$$badness(C) = \sum_{c \in C} \mathrm{var}\,iance(c)$$

	BOS	NY	CHI	DEN	SF	SEA
BOS	0	206	963	1949	3095	2979
NY		0	802	1771	2934	2815
CHI			0	966	2142	2013
DEN				0	1235	1307
SF					0	808
SEA						0

$$dist(X1, X2, p) = (\sum_{k=1}^{len} abs(X1_k - X2_k)^p)^{1/p}$$