Data

In [58]:

```
def millionify(lst):
    return list(map(lambda n: n * 1000000, lst))

turnout = [0.64, 0.65, 0.66, 0.74, 0.9]
support = [0.29, 0.46, 0.46, 0.6, 0.64] #brexit
demographics = millionify([7,12,12,10,10])
result = millionify([17.4, 16.1])
mortality = list(map(lambda c: 1/c, [3000, 2000, 500, 130, 40]))

turnout, support, demographics, result, mortality
```

Out[58]:

```
([0.64, 0.65, 0.66, 0.74, 0.9],
 [0.29, 0.46, 0.46, 0.6, 0.64],
 [7000000, 12000000, 12000000, 10000000, 10000000],
 [17400000.0, 16100000.000000002],
 [0.0003333333333333333, 0.0005, 0.002, 0.007692307692307693, 0.02
5])
```

In [59]:

```
import pandas as pd
```

In [60]:

```
keys = ["turnout", "support", "demographics", "mortality"]
raw = [turnout, support, demographics, mortality]
```

In [ ]:

```

```

In [61]:

```
d = []
for i in range(5):
    vals = {}
    for k,r in zip(keys, raw):
        vals[k] = r[i]
    d.append(vals)

data = pd.DataFrame(d)
data
```

Out[61]:

	demographics	mortality	support	turnout
0	7000000	0.000333	0.29	0.64
1	12000000	0.000500	0.46	0.65
2	12000000	0.002000	0.46	0.66
3	10000000	0.007692	0.60	0.74
4	10000000	0.025000	0.64	0.90

In [62]:

```
brexit_vote = (data.demographics * data.turnout * data.support).sum()
brexit_vote
```

Out[62]:

```
18730400.0
```

Only off by about 1.5M... given how rough some of these numbers are that's pretty impressive...

In [63]:

```
remain_vote = (data.demographics * data.turnout * (1 - data.support)).sum()
remain_vote
```

Out[63]:

```
17869600.0
```

In [ ]:

```

```

In [64]:

```
remain_vote / (brexit_vote + remain_vote), brexit_vote / (brexit_vote + remain_v
ote)
```

Out[64]:

```
(0.48824043715846993, 0.51175956284153)
```

Again this is pretty close to real result... could massage a bit but given huge data limitations probably not worth it?

In [ ]:

```
```

Now we start killing people

In [65]:

```
y = 2016

def advance_by_one_year():
    """Functional programmer James is ashamed of this code..."""
    data.demographics = (data.demographics * (1 - data.mortality)).map(int)
    return y + 1
```

Something very odd going on with iPython wouldn't let me year += 1 in this function; I know that is messy but should be valid?

In [66]:

```
y, data
```

Out[66]:

```
(2016,    demographics  mortality  support  turnout
 0        7000000   0.000333     0.29     0.64
 1       12000000   0.000500     0.46     0.65
 2       12000000   0.002000     0.46     0.66
 3       10000000   0.007692     0.60     0.74
 4       10000000   0.025000     0.64     0.90)
```

In [ ]:

```
```

In [68]:

```
def vote_info():
    brexit_vote = (data.demographics * data.turnout * data.support).sum()
    remain_vote = (data.demographics * data.turnout * (1 - data.support)).sum()
    remain_pct = remain_vote / (brexit_vote + remain_vote)
    brexit_pct = brexit_vote / (brexit_vote + remain_vote)
    return { "brexit": brexit_vote, "remain": remain_vote, "brexit_pct": brexit_
pct, "remain_pct": remain_pct, "year": y}
```

In [69]:

```
vote_info()
```

Out[69]:

```
{'brexit': 18730400.0,
 'brexit_pct': 0.51175956284153,
 'remain': 17869600.0,
 'remain_pct': 0.48824043715846993,
 'year': 2016}
```

In [70]:

```
history = []

for i in range(20):
    history.append(vote_info())
    y = advance_by_one_year()
```