Davis Zvejnieks
Prof. Billey
Math 480
17 May 2016
Homework 7

1 Describe your strategy for winning Pass the Pigs

Justin and I won both games of Pass the Pigs. This may purely be by chance, but we used a similar strategy for the second game using formalized rules from our probability and estimated value computations.

We first set up an equation for the estimated value incorporating the variable y, which represents the current, un-banked points.

Let

$$\Omega = \text{the sample space, i.e., all valid rolls of the pigs}$$
$$X = \text{The random variable on } \Omega \text{ representing the points received from each roll}$$
$$y = \text{current, un-banked points; } y = 0\text{, on the first roll}$$
$$\begin{aligned} E[X] = &P(X=1)1 + P(X=5)5 + P(X=10)10 + P(X=15)15 + \\ &P(X=20)20 + P(X=40)40 + P(X=60)60 - P(X=0)y \end{aligned}$$

Note that because I do not have access to the sample data we recorded, symbolic notation is being used.

$$E[X] = C - P(X=0)y$$

Where C is some constant.

We then interpreted each subsequent roll as betting the current, un-banked points. We found a value where

$$E[X] > 0$$

Then,

$$E[X] = C - P(X=0)y$$
$$C - P(X=0)y > 0$$
$$P(X=0)y < C$$
$$y < C/P(X=0)$$

We would then continue to roll while $y < C/P(X = 0)$ is true. In our calculation from the sample data we found, we arrived at $C/P(X = 0) \hat{=} 48$. That is we continued to rull until we had more than 48 banked points.

We won using this strategy, but even if the value was reprasentative of the actual probability, we took an unnecessary risk rolling up to 48 points. With two successfull banks of 48 points, we are still short of achieving 100 points. We could have chosen 34 and still had won in 3 attempts. Also, if we had rolled up to 50, then we could have won in two successful banks. So regardless of the accuracy of our probabilities, rolling up to 48 points was sub-optimal.

2 Compute the probabilities of each payout given in the Mega Millions Lottery

First, I created a text document with the data provided by Mega Millions Lottery[1], and imported that to Sage.

```
sage: import numpy
sage: lottoData = numpy.loadtxt("lotto_table",dtype=int)
sage: lottoData
[[        5         1  15000000 258890850]
 [        5         0   1000000  18492204]
 [        4         1      5000    739688]
 [        4         0       500     52835]
 [        3         1        50     10720]
 [        3         0         5       766]
 [        2         1         5       473]
 [        1         1         2        56]
 [        0         1         1        21]]
```

Note that, the jackpot value is variable. For the creation of this table, the jackpot value is set at the minimum of $15,000,000.
Next, let's just create a 2 dimensional array containing just the payout and the denominator of the odds.

```
sage: payoutList=[]
sage: for i in range(len(lottoData)):
....:     tempList=[]
....:     tempList.append(lottoData[i][2])
....:     tempList.append(1.0/(lottoData[i][3]))
....:     payoutList.append(tempList)
```

[1] http://www.megamillions.com/how-to-play

```
payoutList
```

15000000	$3.86263168435655 \times 10^{-9}$
1000000	$5.40768423277182 \times 10^{-8}$
5000	$1.35192135062351 \times 10^{-6}$
500	0.0000189268477335100
50	0.0000932835820895522
5	0.00130548302872063
5	0.00211416490486258
2	0.0178571428571429
1	0.0476190476190476

Let's test that these probabilities provided by Mega Millions is accurate. Starting from the jackpot and moving down the table.

First, let's split the calculations between the two sets of numbers to choose from, 1-75, in the first set, and 1-15 in the second. Let's calculate the size of the sample space for the first set, since this is computationally expensive and we'll be using it frequently.
Choosing five numbers out of the 75 can be represented easily using combinatoric notation.

```
sage: sSize=Combinations(range(75),5).cardinality()
sage: sSize
17259390
```

We can leave the second set alone, since it's easy to compute, which is 15. Then we'll consider two events, picking the 5 numbers out of 75, and picking the 1 number out of 15.

Jackpot
The only way to win the jackpot is to get the same exact 5 numbers as pulled, $\binom{5}{5} = 1$ way, divided by the sample space and then because choosing from the second set of numbers is independent, we multiply the probabilities together.

```
sage: testList=[] ##initialize the list
sage: testList.append((1.0/sSize)*(1.0/15))
None
```

$1,000,000 Prize
The probability within the first sample space is the same, but from the second event, any but the winning number can be selected, hence $\frac{14}{15}$.

```
sage: testList.append((1.0/sSize)*(14.0/15))
None
```

$5,000 Prize
Here, given the 5 winning numbers, we must select 4 of them. There are $\binom{5}{4} = 5$ ways to

do this, as there are 5 ways to leave out a single number. Then, we need to select one more number, it can be any number except for the winning number, or any number we previously chose, hence we have 70 remaining choices. The probability of getting the winning number out of 15 is the same as the jackpot.

```
sage: testList.append( Combinations(range(5),4).cardinality()
   *70/(sSize)*(1.0/15))
None
```

$500 Prize
This is the same as the previous calculation, except we must select any of the 14 non-winning numbers.

```
sage: testList.append( Combinations(range(5),4).cardinality()
   *70/(sSize)*(14.0/15))
None
```

$50 Prize
First, we must pick 3 of the 5 winning numbers, there are $\binom{5}{3} = 10$ ways to do this. Then there are 2 remaining numbers to pick from the first sample space. We cannot pick any of the winning numbers nor numbers we already chose, so there are $\binom{70}{2} = 70 \times 69/2$ ways to do this. Note if we just use 70×69 ways to pick the last two numbers, we overcount by 2 times, hence the multiplication by $1/2$ as a result from combinatorics, e.g., 50,23 and 23,50 are the same in this lottery, as order doesn't matter.

```
sage: testList.append( (Combinations(range(5),3).cardinality()
   *Combinations(range(70),2).cardinality())/(sSize)*(1.0/15))
None
```

$5 Prize (1st Option)
This is the same as the previous calculation, except we choose any of the 14 non-winning numbers from the second sample space.

```
sage: testList.append( (Combinations(range(5),3).cardinality()
   *Combinations(range(70),2).cardinality())/(sSize)*(14.0/15))
None
```

$5 Prize (2nd Option)
This is similar to the $50 prize payout, except we only choose 2 of the winning 5 numbers, and then 3 of the remaining 70 numbers.

```
sage: testList.append( (Combinations(range(5),2).cardinality()
   *Combinations(range(70),3).cardinality())/(sSize)*(1.0/15))
```

```
None
```

$2 Prize

Of the 5 winning numbers, only select 1. Then there are $\binom{5}{1} = 5$ ways. Then from the remaining unselected and non-winning numbers, select 4. Multiply this by the probability of selecting the correct number out of 15.

```
sage: testList.append( (Combinations(range(5),1).cardinality()
   *Combinations(range(70),4).cardinality())/(sSize)*(1.0/15))
None
```

$1 Prize

Here 5 numbers must be selected of the non-winning numbers, but the winning number of the 15 must be selected.

```
sage: testList.append( (Combinations(range(70),5).cardinality
   ())/sSize*(1/15.0))
None
```

```
testList
```

$3.86263168435655 \times 10^{-9}$
$5.40768435809918 \times 10^{-8}$
$1.35192108952479 \times 10^{-6}$
0.0000189268952533471
0.0000932825551772108
0.00130595577248095
0.00211440458401678
0.0177081383911405
0.0467494853526110

Let's examine the differences between our calculated probabilities and the probabilities given by Mega Millions Lottery.

```
sage: for i in range(len(payoutList)):
....:     print payoutList[i][1] - testList[i]
```

0.000000000000000
$-1.25327354864749 \times 10^{-15}$
$2.61098712154746 \times 10^{-13}$
$-4.75198371320887 \times 10^{-11}$
$1.02691234145748 \times 10^{-9}$
$-4.72743760324438 \times 10^{-7}$
$-2.39679154198560 \times 10^{-7}$
0.000149004466002346
0.000869562266436662

We see here the differences are quite small, much past 5 significant digits, however, the \$2 and \$1 prize probabilities diverge greatly. This is likely for simplicity, as it's simpler to understand $\frac{1}{21}$ than 23986927/513094996.

Finally, we'll compute the probability of not winning. We can compute this two ways.

```
sage: sum = 0
sage: for i in range(len(testList)):
....:     sum = sum+testList[i]
sage: 1-sum
0.932008396588755
```

Now let's verify this is correct. We'll be adding these probabilities together, because each event is mutually exclusive:

```
sage: ##Probability that we have two winning numbers from the
   first set of numbers, but no winning number from the second
   set.
sage: part1 = Combinations(range(5),2).cardinality()*
   Combinations(range(70),3).cardinality()/sSize*(14/15.0)
sage: ##Probability that we have 1 winning number from the
   first set, but no winning numbers from the second.
sage: part2 = Combinations(range(5),1).cardinality()*
   Combinations(range(70),4).cardinality()/sSize*(14/15.0)
sage: ##Probability we have no winning numbers from the either
    the first or second set of numbers
sage: part3 = Combinations(range(70),5).cardinality()/sSize
   *(14/15.0)
sage: nowin = part1+part2+part3
sage: nowin
0.932008396588756
```

Here we see that our margin of error is extremely small and is a result of rounding from the sum of all possible ways to win.

Now let's create a payout table using the new accurate probabilities we just calculated.

```
sage: accPayout=[]
sage: for i in range(len(testList)):
....:     tempList=[]
....:     tempList.append(lottoData[i][2])
....:     tempList.append(testList[i])
....:     accPayout.append(tempList)
sage: tempList=[]
sage: tempList.append(0); tempList.append(nowin); accPayout.
    append(tempList)
```

JACKPOT	$3.86263168435655 \times 10^{-9}$
\$1000000	$5.40768435809918 \times 10^{-8}$
\$5000	$1.35192108952479 \times 10^{-6}$
\$500	0.0000189268952533471
\$50	0.0000932825551772108
\$5	0.00130595577248095
\$5	0.00211440458401678
\$2	0.0177081383911405
\$1	0.0467494853526110
\$0	0.932008396588756

3 Calculate expected payouts

Now let's calculate the expected payout at the minimum jackpot value of \$15,000,000, by finding the expected value of this probability distribution. Note that E is initialized to -1, as the cost of playing the lottery.

```
sage: E = -1
sage: for i in range(len(accPayout)):
....:     E = E + (accPayout[i][0]*accPayout[i][1])
sage: E
-0.767828936403
```

Then the expected payout of playing the lottery at the minimum jackpot value is about -\$0.77.

Next let's check the expected payout of the lottery at the maximum value of \$656,000,000.

```
sage: accPayout[0][0] = 656000000
sage: E = -1
sage: for i in range(len(accPayout)):
```

```
....:     E = E + (accPayout[i][0]*accPayout[i][1])
sage: E
1.70811797327
```

Now with a higher jackpot, the expected payout is about \$1.71.