Contents

Probability & Monty Hall

In the late 70s and early 80s there was a game show called *Let's Make A Deal that* was hosted by Monty Hall. One of the games is to give the contestant a choice of three doors: Behind one door is 1 million dollars and behind the other two doors are goats. The contestant picks a door, say No. 1, and Monty, who knows what's behind the other doors, always opens another door which has a goat behind it.

The contestant now has two choices:

1. Stay with the original guess
2. Switch doors.

What should you do? This module will generate empirical probabilities to help one determine the best course of action.

Opportunity 1 sets the probability ground-work for the Monty Hall simulation that appears in opportunity two.

Opportunity 2 is the original version the Monty Hall game. There are three doors and the experiment can be run several times (one trial per click) and a running tally is kept.

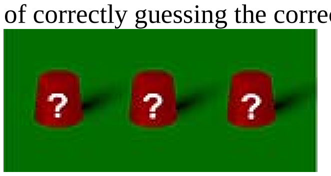
Opportunity 3 modifies the Monty Hall game by allowing the user to choose between 3-10 doors and being able to simulate 1-100 trials per click.

Opportunity 1

Probability for Equally likely outcomes

$$P(A) = \frac{\#\ of\ \ ways\ \ A\ can\ occur}{total\ \#\ of\ \ outcomes}$$

A classic game of chance is to hide an object under one of three cups and attempt to guess under which cup the ball is hidden. If there are three cups, what is the probability of correctly guessing the correct cup? (Ans: 1/3)



What if there are six cups?

(ans: 1/6)

Multiplication principle

If there are n_1 outcomes for one process, n_2, outcomes for another process, … and n_i, outcomes for the i^{th} process, the total number out combined outcomes is

$$n_1 \times n_2 \ldots \times n_i$$

Example:

If you select a person a random, what is the probability that person has a birthday in June? (Assume 365 days)

P(June Birthday) =

$$\frac{30}{365} \approx 0.082 = 8.2\%\ chance$$

P(getting a tail, 1 flip) = ½

P(rolling a 3, one die) = 1/6

The Subway Take-Out Menu, a value meal consists of a sandwich, side (chips, yogurt, cookie, etc), and drink. If there are 18 different sandwiches, 7 different sides (chips, yogurt, cookie, etc), and 10 different drinks on the menu, how many different ways can one order a value meal?

(Ans: 18*7*10= 1260)

How Opportunity 1 is Coded

- This could calculate given probabilities for fixed situations (coin flip, deck of cards, rolling dice, etc).
- The Cup game can easily be modified into the Monty Hall Problem.

To get a random number between two values in ActionScript you can use the following code:

```
function getRandom(start, end) : int {

    var diff:int = end – start;
  return Math.floor(Math.random() * (end – diff)) + start;
}
```

Opportunity 2

There are 3 doors. Pick one and see what P(winning) is when you stay/switch. The experiment can be run several times (one trial per click) and a running tally is kept. Be able to repeat several times and keep a running tally. P(Winning if stay) = 1/3 P(Winning if switch) = 2/3 (see http://www.mste.uiuc.edu/reese/monty/MontyGame5.html for java sample).	Based on the empirical results, what do you think the theoretical probabilities of winning if you stay and winning if you switch are? Ans: P(Winning if stay) = 1/3 P(Winning if switch) = 2/3

Opportunity 3

Allow the user to choose between 3-10 doors and to simulate 1-100 trials per click. Note: After making initial guess out of n doors, n-2 doors are opened always leaving two doors left.	Based on the empirical results, can you come up with a general formula for theoretical probabilities of winning if you stay and winning if you switch for 3 or more doors? Hint: Try several runs with 3,4,5, and 10 doors Ans: P(Winning if stay with n doors) = 1/n P(Winning if switch with n doors) = (n-1)/n

How Opportunity 3 is Coded

You can program the function along the following lines:

```
function runProbabilities(numGames:int) : void {

  var i:int = 0;

  while (i < numGames) {
    var correct:int = Math.floor(Math.random() * numDoors);
    var firstDoor:int = Math.floor(Math.random() * numDoors);

    // If the first door is the correct door, then the second
    // door must be one of the wrong doors
    if (firstDoor == correct) {

      // Update the counts
      SwitchedUpdateScore(0);
      StayedUpdateScore(1);

    } else {
      // Since we didnt pick the correct door initially
      // then when we switch we will be picking the
      // correct doot

      // Update the counts
      SwitchedUpdateScore(1);
      StayedUpdateScore(0);
    }

    i++;
  }
}
```