Self-Assessment 3

Signal Data Science

We'll be having another self-assessment. As before,

- Type your answers in a new R script file with comments indicating where the answer to each question begins.
- Write down your starting time. When you finish, mail `signaldatascience@gmail.com` with your R script attached along with the amount of time you spent on the self assessment.
- Work individually. You can however consult R documentation, look at old assignments, use the Internet, etc., but don't copy and paste code verbatim.
- Make your code as clear, compact, and efficient as possible. Use everything that you've learned! **Please comment and organize your code so we can easily tell how parts of your R script correspond to specific problems.**

Packages you may find useful: `ggplot2`.

Part 1: Logistic regression

Part 2: Probability

In Part 2, we'll look at computational approaches to a variety of probability-based interview questions.

Hashmap collisions

From *120 Interview Questions*, we have the following question:

> Your hash function assigns each object to a number between 1:10, each with equal probability. With 10 objects, what is the probability of a hash collision? What is the expected number of hash collisions? What is the expected number of hashes that are unused?

Use `sample(..., replace=TRUE)` to give estimates of all three.

(*Clarification:* If n objects are assigned to the same hash, that counts as $n - 1$ hash collisions.)

Rolling the dice

Here's a nice question from one of the first cohort's students:

> Given a fair, 6-sided dice, what's the *expected number of rolls* you have to make before each number (1, 2, . . . , 6) shows up at least once?

Write code to estimate the answer.

Bobo the Amoeba

Lastly, here's a problem commonly found in quantitative finance interviews, an easier version of which sometimes appears in data science interviews:

> Bobo the amoeba can divide into 0, 1, 2, or 3 amoebas with equal probability. (Dividing into 0 means that Bobo dies.) Each of Bobo's descendants have the same probabilities. What's the probability that Bobo's lineage eventually dies out?

To solve this problem, we're going to simultaneously simulate a large number of amoeba lineages and incrementally step forward in time to determine how the probability of total extinction changes as we keep iterating forward.

- Write a function `next_gen(n)` that takes in an initial number of amoebas `n`, determines how many amoebas are in the next generation according to the probability above, and returns that value. Sanity check: `next_gen(1)` should return 0, 1, 2, or 3 with equal probability.
- Note the enormous computation time required for `next_gen(n)` when `n` is very large. If there are a *large* of amoebas, we can assume (with reasonable confidence) that the lineage isn't going to die out. Pick a reasonably large value of `n`, like 500 – let's call it `N` – and modify `next_gen(n)` to just return `N+1` when `n > N`. (This is fine because we just want to know if the lineage will *die out* or not – how huge the population can get in cases where it doesn't don't really matter to us.)
- You're going to simulate `num_lineages` lineages for `n_gens` generations, so set `num_lineages = 10000` and `n_gens = 30`.
- Initialize a matrix of appropriate size and dimensions, where each column represents a single lineage of amoebas and every row represents a different generation. Next, set the initial generation to a population of 1.

- Iterate over the number of generations. For each iteration, apply `next_gen(n)` to the population of the most recent generation to get the population for the next generation, filling in the values of your population matrix.
- Turn your population matrix into a matrix of 1s and 0s corresponding to whether the lineage was still alive or died out at each step of the simulation.
- Calculate the probabilities of lineage extinction from your population matrix. *Hint:* Take the `rowSums()` of the matrix.
- `qplot()` the time evolution of the extinction probability. What do you think it is? Give a numerical estimate using your calculations.

Now, use WolframAlpha to solve the cubic equation $p = \frac{1}{4} + \frac{1}{4}p + \frac{1}{4}p^2 + \frac{1}{4}p^3$. One of the solutions will numerically correspond to your calculated probability. How does that polynomial relate to the problem?

Part 3: SQL