Chris Piech
CS 109

Section 6

1. **Binary Tree**: Consider the following function for constructing binary trees:

```
def random_binary_tree(p):
    """
␣␣␣␣Returns␣a␣dictionary␣representing␣a␣random␣binary␣tree␣structure.
␣␣␣␣The␣dictionary␣can␣have␣two␣keys,␣"left"␣and␣"right".
␣␣␣␣"""
    if random_bernoulli(p):   # returns true with probability p
        new_node = {}  # initialize one new node
        new_node["left"] = random_binary_tree(p)
        new_node["right"] = random_binary_tree(p)
        return new_node
    else:
        return None
```

   The `if` branch is taken with probability p (and the `else` branch with probability $1 - p$). A tree with no nodes is represented by `None`; so a tree node with no left child has `None` for the `left` field (and the same for the right child).

   Let X be the number of nodes in a tree returned by `random_binary_tree(p)`. You can assume $0 < p < 0.5$. What is $E[X]$, in terms of p?

2. **Entropy & Name2Age**

   See the Colab notebook at: `https://web.stanford.edu/class/cs109/section/6/`

3. **Choosing a Diagnostic Test with Information Theory**

   A doctor is deciding which diagnostic test to administer to a patient. There are nine mutually exclusive possibilities: diseases A, B, C, D, E, F, G, H, or having *no disease* (labeled None).

   You are given:

   - `prior`: A prior distribution $P(x)$ over all diseases $x \in A, B, C, D, E, F, G, H$, None stored in a dictionary called prior.
   - A function `prob_pos_given_disease(test, x)` that returns the probability that a given test yields a positive result if the patient truly has disease x. (For the "None" case, this value represents the false-positive rate.)
   - `tests`: A list of available tests, stored as `tests`.

   When a test is run, it will return either a + or − result. However, the probability that a test is positive can vary depending on which disease the patient actually has. For example, a test designed to detect disease A may also occasionally return a positive result if the patient has disease B (a cross-reactivity), even though the true disease is not A. So while we only run one test at a time and get one result, that result provides evidence that updates our belief about *all* diseases.

The goal is to determine which test to run by choosing the one that is expected to reduce our uncertainty about the patients condition the most. To do this, write code to compute the expected uncertainty for each test.

4. **Variance of Height among Island Corgis (Optional)**

   This problem is great if you want more practice with bootstrapping but we aren't going to do it in section this week.

   A colleague has collected samples of heights of corgis that live on two different islands, A and B. The colleague collects 50 samples from each island.

   The sample mean is the same for both groups: 10 inches. However, island B has a **sample variance** that is 3.1 in^2 **greater** than island A. The colleague wants to make the claim that island B corgis have a significantly higher spread of heights than island A corgis. You are skeptical. It's possible that heights are identically distributed across both islands, and the observed difference in variance is just a result of chance and small sample size, i.e. the **null hypothesis**.

   Write code that uses **bootstrapping** to calculate the probability of the null hypothesis. Here is the data. Each number is the height, in inches, of an independently sampled corgi:

   Island A Corgi Heights ($S^2 = 6$): [13, 12, 7, 16, 9, 11, 7, 10, 9, 8, 9, 7, 16, 7, 9, 8, 13, 10, 11, 9, 13, 13, 10, 10, 9, 7, 7, 6, 7, 8, 12, 13, 9, 6, 9, 11, 10, 8, 12, 10, 9, 10, 8, 14, 13, 13, 10, 11, 12, 9]

   Island B Corgi Heights ($S^2 = 9.1$): [8, 8, 16, 16, 9, 13, 14, 13, 10, 12, 10, 7, 14, 8, 13, 14, 7, 13, 7, 9, 4, 11, 7, 12, 8, 9, 12, 8, 11, 10, 12, 6, 10, 15, 11, 12, 3, 8, 11, 10, 10, 8, 12, 9, 11, 6, 7, 10, 9, 7]