Baby Derangements

Hello hello everyone and welcome to the new school year and AP Stats! Isn't this so much fun already? I'm currently enjoying morning math a nice bit. In particular, the probability packet that we were given today has quite the interesting problem. I'll briefly reword it for the reader's convenience.

I have not proofread this at all, so forgive any typos :pray:

> We have 4 babies each that correspond to their own mother. As a very funny and legal prank, we give back each baby to each mother in some random order, and we want to know and get an intution for some of the following such questions:
>
> 1. How often do(es) at least (one, two, three, four) mother(s) end up with their correct baby?
> 2. What is the most likely number of correct matches between mother and baby?
> 3. On average, how many correct matches will we find?

In class, I was quite intruiged by the problem, certainly as a start to the statistics class and probability in general, because it isn't a trivial combinatorics argument right off the bat. To someone not familiar, you might not necessarily quickly see how to even interpret this at first look (certainly I couldn't). Although this non-trivial solution is likely to help introduce us to simulations and stuff, I think it's also a great opportunity to apply some fun math.

In order to spice things up just a bit more, a very fun question is to ask how this problem also behaves for not an explicitly specific number of babies, but rather for the general case of n offspring. Let's get into it!

Interpreting the Problem

A good basis to start off with is the fact that we have a finite list of possible combinations that the babies can end up in. Specifically, for n babies there are only $n!$ unique different ways in which they can be handed back to their n mothers. What we're curious of knowing is how many of these babies match their correct parent. Instead of working forwards and trying to determine from a general sequence how many matches are correct, we shall simplify the problem by working backwards and figuring out many of the $n!$ combinations have k correct matches. Knowing then the frequency of the permutations for which have k correct matches, we can then tell which ones are more likely and answer our desired questions.

In particular, fix k babies (where $0 \leqslant k \leqslant n$) to be given to their correct mothers. We then must count the number of permutations where the remaining $n-k$ babies *all do not correspond to their original mothers*. In combinatorics, we have a specific way to call these permutations that do not end up in their original place and, luckily, a decent way of counting them. We call these permutations derangements, and let D_m denote the number of derangements of m elements. There are a few expressions that evaluate to D_m, but we shall focus on the more well known non-recusive relation:

$$D_m = m! \sum_{i=0}^{m} \frac{(-1)^i}{i!}.$$

for all $m \geqslant 1$. In the case of $m = 0$, we define $D_0 = 1$.

With this out of the way, our counting argument is almost done. One needs to also consider the number of ways to choose the k babies that must be fixed in place before permuting the others. As has been somewhat hinted through the use of "choose," we use binomial coefficients to choose k babies out of the set of n total. Putting these two parts together, we

get a very useful expression that helps us better tackle the original question.

Let $P_{n,k}$ denote the number of permutations of the original n babies in which only k get correctly matched to their parents. We have that

$$P_{n,k} = \binom{n}{k} D_{n-k}.$$

Note that it is quite significant that $D_0 = 1$ and $D_1 = 0$ as these work very well with our physical understanding of the problem. If we fix all n children, there is only $D_0 = 1$ way to do this. If we fix only $n - 1$ babies and state that the last one should be incorrectly matched, we know full well that this is impossible, which corresponds to $D_0 = 0$.

One good way to check that this is well and good for all n is to prove that the sum of all these arrangements gets us back to the original $n!$ arrangements. In particular, one must show for all n

$$\begin{aligned} n! &= \sum_{k=0}^{n} P_{n,k} = \sum_{k=0}^{n} \binom{n}{k} D_{n-k} = \sum_{k=0}^{n} \frac{n!}{k!} \sum_{i=0}^{n-k} \frac{(-1)^i}{i!} \\ \implies 1 &= \sum_{k=0}^{n} \frac{1}{k!} \sum_{i=0}^{n-k} \frac{(-1)^i}{i!}. \end{aligned}$$

Perhaps I shall follow up with a proof of this in the future, but it certainly does check out empirically, and we wouldn't like to stray too far from the original objective now would we? ;)

Basically code for "I haven't stared at it long enough."

Speaking of the original problem, let's apply what we have now and look at it again.

A Return to 4 Babies

Using our previously derived formula, let's list out the frequencies for 4 babies in particular, as that's what the problem asks for.

k matches	$P_{4,k}$ possibilities
0	9
1	8
2	6
3	0
4	1

With this it's clear to see that the most frequently appearing case is the $k = 0$ case, with 9 possible ways to hand the babies back to their mothers with the sufficient conditions. In addition, if we wanted to find the average number, or expected value, of the number of mothers on average that have their correctly matched children, this follows directly from the definition of the expected value. Let X denote the random variable representing the number of matched children. We have that

It's interesting to note that it's not always the case that the max is the $k = 0$ case. In fact, it seems that for a given n the maximal case is actually $n \bmod 2$.

$$\begin{aligned} \mathbb{E}[X] &= \sum_{k=0}^{n} k \, \mathbb{P}[X = k] \\ &= \sum_{k=0}^{n} k \cdot \frac{P_{n,k}}{n!} \\ &= \sum_{k=0}^{n} k \cdot \frac{1}{k!} \sum_{i=0}^{n-k} \frac{(-1)^i}{i!} \\ &= \sum_{k=0}^{n} \frac{1}{(k-1)!} \sum_{i=0}^{n-k} \frac{(-1)^i}{i!}, \end{aligned}$$

which also, interestingly enough, seems to be equal to 1 for all n. Because of the linearity

of the expectation operator, we can assume that for any m trials, the expected number of matched children will be $m \cdot 1 = m$ itself.

A Little Simulation

What's a little bit of probability without some fun simulation? Because it's relatively simple to write a computer simulation for this problem, why don't we test it out for $m = 100000$ trials? Below is the code to generate the data:

```
from itertools import permutations
from random import choice

n = 4
m = 100000
arrangements = list(permutations(range(0, n)))

def score(arrangement):
    return sum(1 for i in range(len(arrangement)) if arrangement[i] == i)

frequencies = {0: 0, 1: 0, 2: 0, 3: 0, 4: 0}
total_score = 0

for _ in range(m):
    arrangement = choice(arrangements)
    s = score(arrangement)
    frequencies[s] += 1
    total_score += s

print("Total trials:", m)
print("Frequencies:")
for f in frequencies:
    print(f"  {f}: {frequencies[f]}")
print("Expected Matches:", total_score / m)
```

And its output reasonably checks out empirically with what we calculated earlier. Fun!

```
Total trials: 100000
Frequencies:
  0: 37417
  1: 33319
  2: 25072
  3: 0
  4: 4192
Expected Matches: 1.00231
```

That's all from me. Happy statistic-ing!

Conclusion

Obviously, you don't have to do anything with this, but I thought it'd be fun to show off a little bit of fun combinatorics and probability contained in the class. I'm sure that the class doesn't really require you to be a math nerd to do well in it; I just really like probability.

Hopefully this is right.