Introducing Databricks

This puzzle is going to be a little different.

For this puzzle, I find running the simulation easiest to code in Python, but performing the permutations and set based comparisons to be easier in SQL.

So the answer is Databricks, where we can use both Python and SQL in a single notebook to arrive at our answer.

To switch between Python and SQL you simply use magic commands (%python or %sql) to move between the two languages. Along the way you need to know a few tricks for getting a Spark Dataframe into a table, assigning field names and such, but it's all easy enough to figure out with the vast amount of information on the internet.

Enjoy!

Door Prizes

A producer of a TV game show is creating a new game where 5 different prizes are presented to a contestant, which are then hidden behind 5 doors and randomized. The goal is for the contestant to properly guess the prize hidden behind each door.

During gameplay, the contestant will guess the prize hidden behind the first door, and then the prize is reveled. The contestant then guesses the contents behind the next door, the prize is revealed, and so on. The contestant can duplicate any of their previous guesses.

The producer wants to know the following:

For each scenario, what is the probability distribution of the number of correct guesses?

What are the total permutations possible given a binary outcome (correct/incorrect) with 5 doors?

Are all permutations possible given the game play scenario?

Run the simulation

The following runs a simulation 1 million times in Python and outputs the data to the list: lst_result_final.

The output is each guess and if it was correct (True) or incorrect (False).

Remember, we want both the count of correct answers and all the permutations of answers possible.

```
%python
import random
import pandas as pd

x = 1
lst_result_final = []
simulations = 1_000_000

while x in range(simulations):
  i = 0
  lst_doors = random.sample(range(1, 6), 5)
  lst_guess = [1,2,3,4,5]
  lst_result = []

  while lst_guess:
    random_guess = random.choice(lst_guess)

    if lst_doors[i] == random_guess:
      lst_result.append(True)
    else:
      lst_result.append(False)

    lst_guess.remove(lst_doors[i])

    i = i + 1

  lst_result_final.append(lst_result)
  x = x + 1
```

Create a Spark Dataframe

Below creates a Pandas dataframe (df) and then converts the dataframe to a Spark dataframe (f). We use the StructType to define the column headers.

```
%python
from pyspark.sql.types import *
df = pd.DataFrame(lst_result_final)

schema = StructType([
   StructField("d1", StringType(), True),
   StructField("d2", StringType(), True),
   StructField("d3", StringType(), True),
   StructField("d4", StringType(), True),
   StructField("d5", StringType(), True)])

f = spark.createDataFrame(df,schema)
```

Create an SQL table

The following creates an SQL table.

```
%python
temp_table_name = "simulation_results"
f.createOrReplaceTempView(temp_table_name)
```

Create a new SQL table

For simplicity, we convert the True/False values to 1/0.

```
%sql
CREATE OR REPLACE TEMPORARY VIEW temp_true_false_integers AS
SELECT
 (CASE d1 WHEN 'false' THEN 0 ELSE 1 END) AS d1
,(CASE d2 WHEN 'false' THEN 0 ELSE 1 END) AS d2
,(CASE d3 WHEN 'false' THEN 0 ELSE 1 END) AS d3
,(CASE d4 WHEN 'false' THEN 0 ELSE 1 END) AS d4
,(CASE d5 WHEN 'false' THEN 0 ELSE 1 END) AS d5
FROM simulation_results;
```

OK

Determine Total Permutations

Determine all permutations possible where each of the 5 fields (doors) could be either True (1) or False (0).

There are a total of 32 different permutations.

```
%sql
--select * from temp_table;
CREATE OR REPLACE TEMPORARY VIEW temp_true_false AS
(SELECT 0 AS f)
UNION
(SELECT 1);

CREATE OR REPLACE TEMPORARY VIEW temp_all_permutations AS
SELECT a.f AS d1
       ,b.f AS d2
       ,c.f AS d3
       ,d.f AS d4
       ,e.f AS d5
FROM temp_true_false a CROSS JOIN
     temp_true_false b CROSS JOIN
     temp_true_false c CROSS JOIN
     temp_true_false d CROSS JOIN
     temp_true_false e;


SELECT * FROM temp_all_permutations;
```

	d1	d2	d3	d4	d5
1	0	0	0	0	0
2	0	0	0	0	1
3	0	0	0	1	0
4	0	0	0	1	1
5	0	0	1	0	0
6	0	0	1	0	1
7	0	0	1	1	0

Showing all 32 rows.

Determine Counts

Determine the values for the fields [total count] and [total correct guesses].

```
%sql
CREATE OR REPLACE TEMPORARY VIEW temp_results AS
SELECT a.d1,a.d2,a.d3,a.d4,a.d5
      ,a.d1 + a.d2 + a.d3 + a.d4 + a.d5 AS total_correct_guesses
      ,count(b.d1) AS total_count
FROM temp_all_permutations a LEFT OUTER JOIN
    temp_true_false_integers b ON
    a.d1 = b.d1 AND
    a.d2 = b.d2 AND
    a.d3 = b.d3 AND
    a.d4 = b.d4 AND
    a.d5 = b.d5
GROUP BY a.d1,a.d2,a.d3,a.d4,a.d5;
```

OK

Graph the Distribution

There is 0% chance of getting none of the guesses correct, as you will always know the prize behind the last door.

There is less than 1% chance of getting all 5 guesses correct.

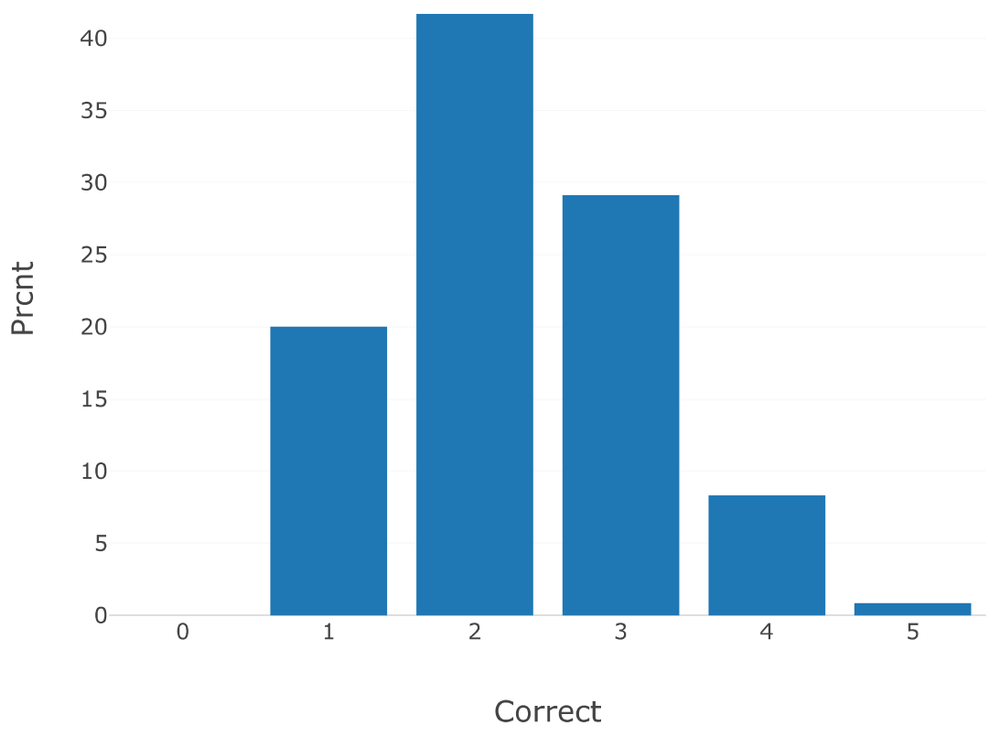
There is a 41% chance you will get 5 correct guesses correct.

```
%sql
SELECT total_correct_guesses AS Correct
        --sum(total_count)/1000000 AS Prcnt --(SELECT SUM(TOTAL_COUNT) FROM
temp_results)) AS Prcnt
        ,(sum(total_count)/(SELECT SUM(TOTAL_COUNT) FROM temp_results)) * 100
AS Prcnt
FROM temp_results
GROUP BY total_correct_guesses
ORDER BY 1;
```

	Correct	Prcnt
1	0	0
2	1	20.01422001422001
3	2	41.68604168604168
4	3	29.13212913212913
5	4	8.332308332308331
6	5	0.8353008353008353

Showing all 6 rows.

```
%sql
SELECT total_correct_guesses AS Correct
        --sum(total_count)/1000000 AS Prcnt --(SELECT SUM(TOTAL_COUNT) FROM
temp_results)) AS Prcnt
        ,(sum(total_count)/(SELECT SUM(TOTAL_COUNT) FROM temp_results)) * 100
AS Prcnt
FROM temp_results
GROUP BY total_correct_guesses
ORDER BY 1;
```



Impossible Permutations

The following lists the 16 permutations you would not find in the simulation. During the game, you will always know the prize behind the last door; the permutations not possible all have a false value for door #5.

```
SELECT * FROM temp_results WHERE total_count = 0 ORDER BY 6,1,2,3,4,5
```

	d1	d2	d3	d4	d5	total_corr
1	0	0	0	0	0	0
2	0	0	0	1	0	1
3	0	0	1	0	0	1
4	0	1	0	0	0	1
5	1	0	0	0	0	1
6	0	0	1	1	0	2
7	0	1	0	1	0	2

Showing all 16 rows.

Possible Permutations

The following lists the 16 permutations you would find in the simulation. During the game, you will always know the prize behind the last door; the permutations possible all have a true value for door #5.

```
SELECT * FROM temp_results WHERE total_count > 0 ORDER BY 7 DESC ,6,1,2,3,4,5
```

	d1	d2	d3	d4	d5	total_corr
1	0	0	0	0	1	1
2	0	0	0	1	1	2
3	0	0	1	0	1	2
4	0	0	1	1	1	3
5	0	1	0	0	1	2
6	0	1	0	1	1	3
7	1	0	0	1	1	3

Showing all 16 rows.

The End