Nate Boyle
4/24/2023

AB Testing Final Assignment

for the

"Data Wrangling, Analysis, and AB Testing with SQL"

course offered by UC Davis through Coursera

1. Compare the final_assignments_qa table to the assignment events we captured for user_level_testing. Write an answer to the following question: Does this table have everything you need to compute metrics like 30-day view-binary?

You need the date and time the test began and a column of the tests for each item with the result alongside it in a separate column.

```
--We are running an experiment at an item-level, which means all users who visit will see the same page, but the layout of different item pages may differ.
--Compare this table to the assignment events we captured for user_level_testing.
--Does this table have everything you need to compute metrics like 30-day view-binary?

SELECT
  *
FROM
  dsv1069.final_assignments_qa
```

Data | Fields | Source

	item id	test a	test b	test c	test d	test e	test f
1	2512	1	0	1	1	0	1
2	482	0	1	1	1	0	0
3	2446	0	1	1	0	1	0
4	1312	0	0	0	0	0	1
5	3556	1	1	0	1	0	0

2. Write a query and table creation statement to make final_assignments_qa look like the final_assignments table. If you discovered something missing in part 1, you may fill in the value with a place holder of the appropriate data type.

First, I noticed that the result for tests a, b, and c in the final_assignments_qa table matched the results for item_test 1, 2, and 3, respectively, in the final_assignments table but in a transposed manner.

```
--Reformat the final_assignments_qa to look like the final_assignments table, filling in any missing values with a placeholder of the appropriate data type.

SELECT
  *
FROM
  dsv1069.final_assignments_qa
```

Data | Fields | Source

	item id	test a	test b	test c	test d	test e	test f
1	2512	1	0	1	1	0	1
2	482	0	1	1	1	0	0
3	2446	0	1	1	0	1	0
4	1312	0	0	0	0	0	1
5	3556	1	1	0	1	0	0

Please find a screenshot of the query and results of the five item ids above in the final_assignments table on the next page.

```
SELECT
  *
FROM
  dsv1069.final_assignments
WHERE
  item_id IN (2512, 482, 2446, 1312, 3556)
ORDER BY
  item_id, test_number
```

Data | Fields | Source

	item id	test assignment	test number	test start date
1	482	0	item_test_1	2013-01-05 00:00:00
2	482	1	item_test_2	2015-03-14 00:00:00
3	482	1	item_test_3	2016-01-07 00:00:00
4	1312	0	item_test_1	2013-01-05 00:00:00
5	1312	0	item_test_2	2015-03-14 00:00:00
6	1312	0	item_test_3	2016-01-07 00:00:00
7	2446	0	item_test_1	2013-01-05 00:00:00
8	2446	1	item_test_2	2015-03-14 00:00:00
9	2446	1	item_test_3	2016-01-07 00:00:00
10	2512	1	item_test_1	2013-01-05 00:00:00
11	2512	0	item_test_2	2015-03-14 00:00:00
12	2512	1	item_test_3	2016-01-07 00:00:00
13	3556	1	item_test_1	2013-01-05 00:00:00
14	3556	1	item_test_2	2015-03-14 00:00:00
15	3556	0	item_test_3	2016-01-07 00:00:00

Then, I also noticed, and additionally verified through a query, that the test start dates for tests 1, 2, and 3, did not vary between items, i.e., each respective test start date was the same for all items.

```
SELECT
  DISTINCT test_number, test_start_date
FROM
  dsv1069.final_assignments
```

Data | Fields | Source

	test number	test start date
1	item_test_3	2016-01-07 00:00:00
2	item_test_1	2013-01-05 00:00:00
3	item_test_2	2015-03-14 00:00:00

After thinking about it for a while I realized that I could just UNION ALL three different queries of the table together. Each one of the three queries would handle a different test number. Please find a screenshot of this last final_assignments_qa query on the next page and also notice that the results are identical to the results of the final_assignments query at the top of this page, thus solving problem #2.

```sql
/*Reformat the final_assignments_qa to look like the final_assignments table,
filling in any missing values with a placeholder of the appropriate data type.*/
SELECT
  *
FROM
  (SELECT
    item_id, test_a AS test_assignment,
    'item_test_1' AS test_number,
    '2013-01-05 00:00:00' AS test_start_date
  FROM dsv1069.final_assignments_qa table_1
  UNION ALL
  SELECT
    item_id, test_b AS test_assignment,
    'item_test_2' AS test_number,
    '2015-03-14 00:00:00' AS test_start_date
  FROM dsv1069.final_assignments_qa table_2
  UNION ALL
  SELECT
    item_id, test_c AS test_assignment,
    'item_test_3' AS test_number,
    '2016-01-07 00:00:00' AS test_start_date
  FROM dsv1069.final_assignments_qa table_3) final_table
WHERE item_id IN (2512, 482, 2446, 1312, 3556)
ORDER BY item_id, test_number
```

Data | Fields | Source

	item id	test assignment	test number	test start date
1	482	0	item_test_1	2013-01-05 00:00:00
2	482	1	item_test_2	2015-03-14 00:00:00
3	482	1	item_test_3	2016-01-07 00:00:00
4	1312	0	item_test_1	2013-01-05 00:00:00
5	1312	0	item_test_2	2015-03-14 00:00:00
6	1312	0	item_test_3	2016-01-07 00:00:00
7	2446	0	item_test_1	2013-01-05 00:00:00
8	2446	1	item_test_2	2015-03-14 00:00:00
9	2446	1	item_test_3	2016-01-07 00:00:00
10	2512	1	item_test_1	2013-01-05 00:00:00
11	2512	0	item_test_2	2015-03-14 00:00:00
12	2512	1	item_test_3	2016-01-07 00:00:00
13	3556	1	item_test_1	2013-01-05 00:00:00
14	3556	1	item_test_2	2015-03-14 00:00:00
15	3556	0	item_test_3	2016-01-07 00:00:00

3. Use the final_assignments table to calculate the order binary for the 30-day window after the test assignment for item_test_2 (You may include the day the test started).

Fairly straightforward query given what was learned in the course. Create a binary variable that is a success, a '1', when the item was ordered on a date after the date item_test_2 for the item started AND was within thirty days of that test start date. Then, count the number of items for both the control (0) and treatment (1) groups and sum up the number of successes for each group.

```sql
-- Use this table to
-- compute order_binary for the 30 day window after the test_start_date
-- for the test named item_test_2
SELECT
  test_number,
  test_assignment,
  COUNT(item_id) AS items,
  SUM(order_binary_30d) AS orders_binary_30d
FROM
  (SELECT
   fa.item_id,
   test_number,
   test_assignment,
   MAX (CASE WHEN (created_at > test_start_date
   AND DATE_PART('day', created_at - test_start_date) <= 30)
   THEN 1 ELSE 0 END) AS order_binary_30d
  FROM
    dsv1069.final_assignments fa
  LEFT JOIN
    dsv1069.orders o
  ON
    o.item_id = fa.item_id
  WHERE
    test_number = 'item_test_2'
  GROUP BY
   fa.item_id,
   test_number,
   test_assignment) AS order_binary_table
GROUP BY
  test_assignment,
  test_number
```

Data | Fields | Source

	test number	test assignment	items	orders binary 30d
1	item_test_2	0	1130	341
2	item_test_2	1	1068	319

I went ahead and plugged these numbers into the ABBA A/B testing statistics calculator free website (https://thumbtack.github.io/abba/demo/abba.html) and got a -1% lift in the range of -14% to 12%, but also got a p-value of 0.88, meaning that the treatment produced statistically insignificant changes from the control group. Overall, more failures than successes, unlike the View Item Binary results.

4. Use the final_assignments table to calculate the view binary for the 30-day window after the test assignment for item_test_2. (You may include the day the test started)

Basically just the same code from question 3, except using the view_item_events table instead of the orders table, and the event_time date field instead of the date and invoice/order was made.

```sql
-- Use this table to
-- compute view_binary for the 30 day window after the test_start_date
-- for the test named item_test_2
SELECT
  test_number,
  test_assignment,
  COUNT(item_id) AS items,
  SUM(view_binary_30d) AS views_binary_30d
FROM
  (SELECT
   fa.item_id,
   test_number,
   test_assignment,
   MAX (CASE WHEN (event_time > test_start_date
   AND DATE_PART('day', event_time - test_start_date) <= 30)
   THEN 1 ELSE 0 END) AS view_binary_30d
  FROM
    dsv1069.final_assignments fa
  LEFT JOIN
    dsv1069.view_item_events v
  ON
    v.item_id = fa.item_id
  WHERE
    test_number = 'item_test_2'
  GROUP BY
   fa.item_id,
   test_number,
   test_assignment) AS view_binary_table
GROUP BY
  test_assignment,
  test_number
```

Data | Fields | Source

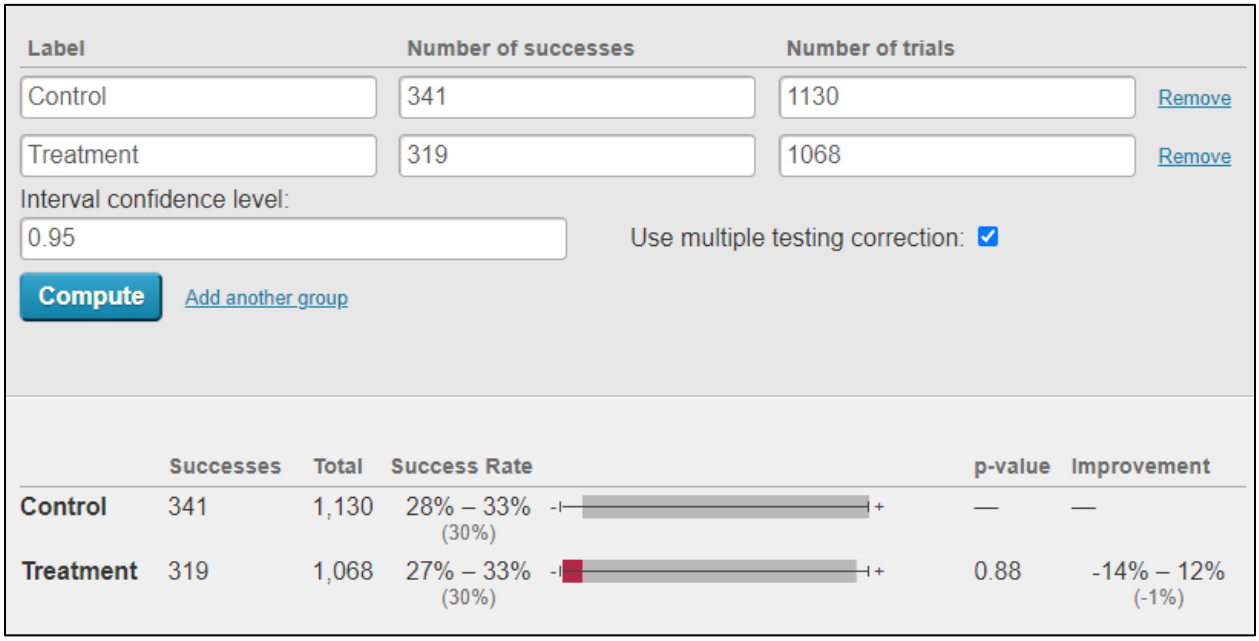
	test number	test assignment	items	views binary 30d
1	item_test_2	0	1130	925
2	item_test_2	1	1068	894

I went ahead and plugged these numbers into the ABBA A/B testing statistics calculator free website (https://thumbtack.github.io/abba/demo/abba.html) and got a lift of 2.3% in the range of -1.6% to 6.1%, but also got a p-value of 0.25, meaning that the treatment produced statistically insignificant changes from the control group. Overall though, more successes than failures, unlike the Order Binary results.

5. Use the https://thumbtack.github.io/abba/demo/abba.html to compute the lifts in metrics and the p-values for the binary metrics (30 day order binary and 30 day view binary) using a interval 95% confidence.

See the bottom of the page for both questions 3 and 4, but screenshots have also been provided below for each test.

Order Binary:



View Item Binary: