SDSC 2022 London

Powering Cloud-based Spatial Analytics For Retail with CARTO

Context

Identifying an optimal location for a new store is not always an easy task, and we often do not have enough data at our disposal to build a solid model to predict potential revenues across an entire territory. In these cases, managers rely on different business criteria in order to make a sound decision for their expansion strategy. For example, they rely on defining their target market and segmenting population groups accordingly in order to locate the store closer to where the target market lives (e.g. areas with a great presence of youngsters).

In this example, we are going to use CARTO's Analytics Toolbox for BigQuery to explore good locations to open a new Pizza Hut restaurant in Honolulu, Hawaii. To do that, we will run a couple of different spatial analyses.

Area of study

We will start by defining an area of interest for our study. For that, we define a buffer of 5 km around downtown Honolulu.

```
DECLARE honolulu_buffer GEOGRAPHY;
-- We use the ST_BUFFER to define a 5 km buffer centered in Honolulu
SET honolulu_buffer = ST_BUFFER(ST_GEOGPOINT(-157.852587, 21.304390), 5000);
```

Functions used:

- ST_BUFFER

Find and visualize all Pizza Huts in Honolulu

Next, we will get all Pizza Hut restaurants in Honolulu. For that, we use OSM POI data, which is publicly in CARTO's Spatial Data Catalog.

```
DECLARE honolulu_buffer GEOGRAPHY;
-- We use the ST_BUFFER to define a 5 km buffer centered in Honolulu
SET honolulu_buffer = ST_BUFFER(ST_GEOGPOINT(-157.852587, 21.304390), 5000);
SELECT
tag.value AS brand, geometry,
FROM
`cartobq.docs.sdsc_honolulu_osm_planet_nodes` d,
--
`carto-do-public-data.openstreetmap.pointsofinterest_nodes_usa_latlon_v1_quarterly_v1` d,
UNNEST(all_tags) as tag
WHERE ST_CONTAINS(honolulu_buffer, geometry)
AND ((tag.value in ("Pizza Hut") AND tag.key = 'brand'))
```

Note we are using the table `cartobq.docs.sdsc_honolulu_osm_planet_nodes` that contains all POIs in the area of study. We are using this table so that:

- Attendees don't need to create a new connection.
- Queries run faster.

Get the right data

Gridify area of study. Polyfill

Our customer is interested in looking for new areas based on:

- Demographics
- Points of Interest (POI)
- Distance to existing Pizza Hut restaurants

Therefore, our next step is to get this data using the enrichment capabilities of CARTO's Analytics Toolbox as well as other core spatial functions.

We will use public data so that attendees can replicate the entire guide once they create a connection to their data warehouse.

First, we **create an empty table** with the area of study polyfilled with h3 cells resolution 10. For more information on h3 and other spatial indices, visit our documentation.

```
DECLARE honolulu_buffer GEOGRAPHY;
-- We use the ST_BUFFER to define a 5 km buffer centered in Honolulu
SET honolulu_buffer = ST_BUFFER(ST_GEOGPOINT(-157.852587, 21.304390), 5000);

CREATE OR REPLACE TABLE `<project>.<dataset>.<origin_table>` AS (
   SELECT h3
   FROM UNNEST(`carto-un`.carto.H3_POLYFILL(honolulu_buffer, 10)) h3
)
```

Functions used:

- H3_POLYFILL

Enrich grid using the Analytics Toolbox

Next, we enrich that table with ACS sociodemographics variables. We need to be subscribed first.

Here you can check the documentation of all functions used in this section.

Subscription

Here, follow the steps in your CARTO3 account to subscribe to the corresponding ACS dataset.

We'll take a quick look at ACS data. This visualization of large amounts of data is possible thanks to CARTO BigQuery Tiler.

Table: `carto-demo-data.demo_tilesets.sociodemographics_usa_blockgroup`

1. *Check subscriptions*

Once subscribed, we can check which datasets we are subscribed to.

```
CALL `carto-un`.carto.DATAOBS_SUBSCRIPTIONS('<project>.<dataset>','')
```

In particular, we can check which datasets from ACS we are subscribed to.

```
CALL
`carto-un`.carto.DATAOBS_SUBSCRIPTIONS('<project>.<dataset>','dataset_provider="Ame
```

```
rican Community Survey"')
```

We are interested in the dataset with slug name `acs_sociodemogr_95c726f9` because it has the most recent data at the finest resolution available (census block groups).

Functions used:

- DATAOBS_SUBSCRIPTIONS

2. *Check available variables*

Once we know the dataset we'd like to use, let's select variables. In this case, we're interested in population by age, income, type of families, and unemployment information.

In particular, we're interested in identifying areas with large populations between 15 and 34 and families with kids under 6 because they're more likely to eat at a fast food restaurant.

Let's first take a look at all available variables for this dataset.

```
CALL
`carto-un`.carto.DATAOBS_SUBSCRIPTION_VARIABLES('<project>.<dataset>','dataset_slug
= "acs_sociodemogr_95c726f9"')
```

Let's now focus only on population variables by age and gender adding a filter.

```
CALL
`carto-un`.carto.DATAOBS_SUBSCRIPTION_VARIABLES('<project>.<dataset>','dataset_slug
= "acs_sociodemogr_95c726f9" AND (variable_name LIKE "female_%" OR variable_name
LIKE "male_%")')
```

Functions used:

- DATAOBS_SUBSCRIPTION_VARIABLES

3. *Enrichment*

Finally, we enrich our table.

Note here we need to have every variable's slug and the aggregation method we'd like to apply.

```
DROP TABLE IF EXISTS `<project>.<dataset>.<output_table_name>`;
CALL `carto-un`.carto.DATAOBS_ENRICH_GRID
(
-- grid_type
'h3',
-- input_query
R'''
SELECT * from `<project>.<dataset>.<input_table_name>`
''',
-- input_index_column
'h3',
-- variables
[('total_pop_3409f36f', 'sum'), ('female_15_to_17_eb1658f1', 'sum'),
('female_18_to_19_6d791436', 'sum'), ('female_20_f727dc', 'sum'),
('female_21_77f0174a', 'sum'), ('female_22_to_24_121a63e5', 'sum'),
('female_25_to_29_a90c21d6', 'sum'), ('female_30_to_34_50344313', 'sum'),
('male_15_to_17_8dd9d9ac', 'sum'), ('male_18_to_19_bb6956b', 'sum'),
('male_20_5264b51', 'sum'), ('male_21_72217bc7', 'sum'),
('male_22_to_24_74d5e2b8', 'sum'), ('male_25_to_29_cfc3a08b', 'sum'),
('male_30_to_34_36fbc24e', 'sum'), ('income_per_capi_bfb55c80', 'avg'),
('households_d7d24db5', 'sum'), ('families_with_y_5c7faf3a', 'sum'),
('unemployed_pop_e1bb4940', 'sum'), ('nonfamily_house_588da9f9', 'sum')],
-- filters
NULL,
-- output
[``<project>.<dataset>.<output_table_name>``],
-- source
'carto-data.ac_7xhfwyml2')
```

Functions used:

- DATAOBS_ENRICH_GRID

4. *Process enriched data*

We would like to have a single variable with the total population aged 15-34 and change the naming.

```
CREATE OR REPLACE TABLE `<project>.<dataset>.honolulu_pizza_aos_enriched_sdsc_abs`
AS
SELECT h3, total_pop_3409f36f_sum AS total_pop,
female_15_to_17_eb1658f1_sum+female_18_to_19_6d791436_sum+female_20_f727dc_sum+fema
le_21_77f0174a_sum+female_22_to_24_121a63e5_sum+female_25_to_29_a90c21d6_sum+female
_30_to_34_50344313_sum+male_15_to_17_8dd9d9ac_sum+male_18_to_19_bb6956b_sum+male_20
_5264b51_sum+male_21_72217bc7_sum+male_22_to_24_74d5e2b8_sum+male_25_to_29_cfc3a08b
_sum+male_30_to_34_36fbc24e_sum AS pop_15_34,
income_per_capi_bfb55c80_avg AS income_per_capita, households_d7d24db5_sum AS
total_hh, families_with_y_5c7faf3a_sum AS families_w_kids,
unemployed_pop_e1bb4940_sum AS unemployed_pop, nonfamily_house_588da9f9_sum AS
nonfamily_hh
FROM `<project>.<dataset>.honolulu_pizza_aos_enriched_sdsc`
```

In addition, we compute relative amounts.

```
CREATE OR REPLACE TABLE `<project>.<dataset>.honolulu_pizza_aos_enriched_sdsc_rel`
AS
SELECT *, 100*pop_15_34/total_pop AS pop_15_34_pct, 100*families_w_kids/total_hh AS
families_w_kids_pct,
100*unemployed_pop/total_pop AS unemployed_pop_pct, 100*nonfamily_hh/total_hh AS
nonfamily_hh_pct
FROM `<project>.<dataset>.honolulu_pizza_aos_enriched_sdsc_abs`
WHERE total_pop > 0
```

Addtional enrichments

POI counts

We compute the number of POIs per cell.

```
-- CREATE OR REPLACE TABLE `cartobq.docs.sdsc_honolulu_aos_enriched` AS
WITH poi_agg_t AS (
 WITH t1 AS (
   SELECT *, `carto-un`.carto.H3_FROMGEOGPOINT(geometry, 10) AS h3
   FROM `cartobq.docs.sdsc_honolulu_osm_planet_nodes`
 )
 SELECT h3, COUNT(id) as n_pois
 FROM t1
 GROUP BY h3
)
SELECT acs.*, IFNULL(poi_agg_t.n_pois, 0) AS n_pois
FROM `cartobq.docs.sdsc_honolulu_aos_enriched_sociodemo` acs
LEFT JOIN poi_agg_t
USING(h3)
```

We compute the number of competitors and the percentage they represent with regards to total POIs per cell.

```
-- CREATE OR REPLACE TABLE `cartobq.docs.sdsc_honolulu_aos_enriched` AS
WITH competitors AS (
 SELECT CAST(id AS STRING) AS id , tag.value, geometry as geom
 FROM `cartobq.docs.sdsc_honolulu_osm_planet_nodes` d,
 UNNEST(all_tags) as tag
 WHERE (tag.value in ('fast_food', 'restaurant') AND tag.key = 'amenity')
),
comp_agg_t AS (
 WITH t1 AS (
 SELECT *, `carto-un`.carto.H3_FROMGEOGPOINT(geom, 10) AS h3
 FROM competitors
 )
 SELECT h3, COUNT(id) as n_pois
```

```
 FROM t1
 GROUP BY h3
)
SELECT acs.*, IFNULL(comp_agg_t.n_pois, 0) AS n_competitors, IF(acs.n_pois>0,
100*IFNULL(comp_agg_t.n_pois, 0)/acs.n_pois, 0) AS n_competitors_pct
FROM `cartobq.docs.sdsc_honolulu_aos_enriched` acs
LEFT JOIN comp_agg_t
USING(h3)
```

Functions used:

- H3_FROMGEOGPOINT

Note this data transformation and computation can be easily done using CARTO's AT function ENRICH_GRID.

Distance to closest

In addition to the population, Pizza Hut would like to consider distance to the closest existing Pizza Hut restaurant because they'd like to avoid cannibalization between their own restaurants.

In order to compute distances, we use kring distances because it is very straight forward.

```
DECLARE honolulu_buffer GEOGRAPHY;
SET honolulu_buffer = ST_BUFFER(ST_GEOGPOINT(-157.852587, 21.304390), 5000);
-- CREATE OR REPLACE TABLE `cartobq.docs.sdsc_honolulu_aos_enriched` AS
-- WITH h3dist AS (
WITH t1 AS (
  SELECT `carto-un`.carto.H3_FROMGEOGPOINT(geometry, 10) as h3,
  FROM `cartobq.docs.sdsc_honolulu_osm_planet_nodes` d,
  UNNEST(all_tags) as tag
  WHERE ST_CONTAINS(honolulu_buffer, geometry)
  AND ((tag.value in ("Pizza Hut") AND tag.key = 'brand'))
),
t2 AS (
```

```
  SELECT * FROM `cartobq.docs.sdsc_honolulu_aos_enriched`
)
SELECT t2.h3, MIN(`carto-un`.carto.H3_DISTANCE(t2.h3, t1.h3)) AS dist
FROM t1
CROSS JOIN t2
GROUP BY t2.h3
-- )
-- SELECT *
-- FROM `cartobq.docs.sdsc_honolulu_aos_enriched`
-- LEFT JOIN h3dist
-- USING(h3)
```

Functions used:

- H3_FROMGEOGPOINT
- H3_DISTANCE

Analysis

Moran's I

We start by running some descriptive analyses.

We are interested in knowing if the variables selected show spatial autocorrelation. This can be a very powerful analysis because if our variable of interest shows high spatial autocorrelation, that means:

- We can find homogeneous local areas more easily
- We have more flexibility for selecting the exact location for the new store

```
SELECT `carto-un`.carto.MORANS_I_H3(input_data, 2, 'exponential')
FROM (
  SELECT ARRAY_AGG(STRUCT(h3, pop_15_34_pct)) AS input_data
  FROM `cartobq.docs.sdsc_honolulu_aos_enriched`
)
```

Functions used:

- MORANS_I_H3

Geographically Weighted Regression

Next, we'd like to know how income per capita is related to the other sociodemographics variables.

For Pizza Hut, higher income does not necessarily translate into higher sales.

```
CALL `carto-un`.carto.GWR_GRID(
  -- input_table
  'cartobq.docs.sdsc_honolulu_aos_enriched',
  -- features_columns
  ['total_pop', 'pop_15_34_pct', 'families_w_kids_pct', 'unemployed_pop_pct',
'nonfamily_hh_pct', 'n_pois'],
  -- label_column
  'income_per_capita',
  -- cell_column
  'h3',
  -- cell_type
  'h3',
  -- kring_distance
  3,
  -- kernel_function
  'gaussian',
  -- fit_intercept
  TRUE,
  -- output_table
  NULL)
```

Functions used:

- GWR_GRID

Commercial hotspots

Finally, we would like to identify areas that meet Pizza Hut requirements, i.e., locations with large populations aged 15-34 and far from existing Pizza Hut restaurants.

In order to identify these locations, we use the Commercial hotspots functionality available in the AT. This functionality identifies areas with values that are significantly higher than the average.

```
CALL `carto-un`.carto.COMMERCIAL_HOTSPOTS(
 -- input
 'cartobq.docs.sdsc_honolulu_aos_enriched',
 -- output
 NULL,
 -- index_column
 'h3',
 -- index_type
 'h3',
 -- variable_columns
 ['pop_15_34','dist'],
 -- variable_weights
 [0.7, 0.3],
 -- kring
  2,
  -- pvalue_thresh
  0.01
)
```

Functions used:

- COMMERCIAL_HOTSPOTS

Local outlier factor

Finally, we visualize Pizza Hut competitors and compute the local outlier factor to identify those that are very close to one another and those far from the other, to visualize where it would be more interesting to open a new restaurant.

Visualize competitors

```
DECLARE honolulu_buffer GEOGRAPHY;
SET honolulu_buffer = ST_BUFFER(ST_GEOGPOINT(-157.852587, 21.304390), 5000);

SELECT CAST(id AS STRING) AS id , tag.value, geometry as geom
FROM `cartobq.docs.honolulu_planet_nodes` d,
UNNEST(all_tags) as tag
WHERE ST_CONTAINS(honolulu_buffer, geometry)
AND ((tag.value in ('fast_food', 'restaurant') AND tag.key = 'amenity'))
```

Compute LOF

```
DECLARE honolulu_buffer GEOGRAPHY;
SET honolulu_buffer = ST_BUFFER(ST_GEOGPOINT(-157.852587, 21.304390), 5000);

-- We get all amenities tagged as restaurants or fast_food POIS in Honolulu
WITH fast_food AS (
 SELECT CAST(id AS STRING) AS id , tag.value, geometry as geom
 FROM `cartobq.docs.honolulu_planet_nodes` d,
 UNNEST(all_tags) as tag
 WHERE ST_CONTAINS(honolulu_buffer, geometry)
 AND ((tag.value in ('fast_food', 'restaurant') AND tag.key = 'amenity'))
),

-- We calculate the Local Outlier Factor in order to identify restaurants without
competition.
lof_output as (
SELECT `carto-un`.carto.LOF(ARRAY_AGG(STRUCT(id,geom)), 5) as lof FROM fast_food
)
SELECT lof.* FROM lof_output, UNNEST(lof_output.lof) AS lof
```

Functions used:

- LOF

Twin Areas

Finally, we'd like to identify areas in the city that are similar to the best performing store located in cell `8a464b96b817fff`.

We'll run this analysis using the twin areas functionality that given an origin location (best performing store in this case) and a set of variables (external variables we have used to characterized the grid), identifies areas that are similar to the origin. For further detail, take a look at this blog post.

```
CALL `carto-un`.carto.FIND_TWIN_AREAS(
   -- origin_query
   R'''
     SELECT *
     FROM `cartobq.docs.sdsc_honolulu_aos_enriched`
     WHERE h3 in ('8a464b96b817fff')
   ''',
   -- target_query
   '''SELECT * FROM `cartobq.docs.sdsc_honolulu_aos_enriched`''',
   -- index_column
   'h3',
   -- pca_explained_variance_ratio
   0.90,
   -- max_results
   250,
   -- output_prefix
   '<project>.<dataset>.<output_table_name>')
```

Functions used:

- FIND_TWIN_AREAS