Approach to Generalization

This document contains advice on how to modify the code in this repository to be used in other use case scenarios. Our solution was designed to take specific advantage of correlations between attributes in the Chicago Taxi Rides dataset.

To generalize our solution, you will need to identify which attributes in your dataset are similar in type to the ones in the challenge dataset, and identify any potential correlations between them. We recommend using an entropy (**Theil's U)** heatmap of the relations between the variables to identify potential derived attributes.

Types of Attributes

We can express all attributes of the datafiles as one a member of one of 3 categories:
1) Archetype Attributes that are determined in archetype generation (clustering) and counting.
2) Derived Attributes that are determined using the archetypes and the public data.
3) Independent Attributes that are determined independently from the private data.

After investigating the data set and target utility measure, you could create an "extra_parameters.json" file which would allow you to specify which category each attribute belongs to. This file can also contain how to determine the derived attributes and how to determine the values of the independent attributes.

How to Generate Archetypes

Archetypes in our solution represent groups of similar individuals in the dataset, when grouped by the distributions of a subset of the attributes. It is possible to select these attributes such that the archetype has semantic meaning. For example, using the Chicago Taxi Rides dataset, we can group individuals by their pickup location, dropoff location and shift distribution (the temporal and spatial attributes). Then, the archetypes of individuals tend to represent groups of real-world taxi drivers, such as *drivers based out of the airport*, or *predominately weekend drivers*.

To identify these archetypes in a more general way, we can use clustering techniques on the dataset, and use the clusters as the archetypes. In our solution, we use a Gaussian Mixture Model (GMM) approach to identify these clusters, as we believe the archetype attributes are normally distributed. You do not have to use this approach in your implementation; you could use another general clustering approach like k-means, or form the archetypes using your prior domain knowledge.

Dealing with Derived Attributes

Synthetic data for derived attributes are determined entirely from existing data, either explicitly public or differentially private releases. This process can be done in essentially any way the analyst chooses, as long as it does not interact directly with the private data. An attribute is a good candidate to be derived if it can be predicted well from other information the analyst already has and the analyst does not expect the relationship between attributes to change significantly between their public and private data.

A very simple example would be attributes that are explicitly hierarchical. If we have location data from individuals in the USA and have already created synthetic data for zip code, we can fully derive state without looking at the private data again. In this case, our derived attribute is predicted perfectly (at least nearly) from another attribute.

One example we used in our solution was the proposed relationship between pickup/dropoff location and other properties of the taxi trip (e.g. miles traveled and fare). We argue that, if you know pickup/dropoff location, you ought to be able to predict these other attributes effectively in the sense the distribution of each conditional on pickup/dropoff is unlikely to change between our public and private data. We perform this prediction by sampling from the public data distribution of the attribute conditional on pickup/dropoff, but this prediction could be done using a more complicated model, domain knowledge, etc.

Now, say we had background knowledge that Chicago was undergoing major repairs on main roads during the collection of our public data, which had been completed by the time of the private data collection. We argue that an analyst should now be wary of deriving properties like miles traveled and fare from pickup/dropoff, as the relationship between these sets of attributes has likely changed.

Dealing with Independent Attributes

Ideally, one will be able to assign all attributes to either be part of the archetype or derived from them, however some attributes may be independent of the archetypes. In the Chicago Taxi Rides dataset, we found that Company_ID for the taxis was not correlated with the archetypes and could not be derived well from individual taxi trips. To address this, we assigned each taxi to a single Company_ID and used a private histogram query, using a portion of the privacy budget, to determine its distribution in the private dataset.

Treating the attribute as an independent attribute removes the correlations that do exist between it and the other attributes, which may lead to worse performance on k-marginal metrics. One could improve the synthetic data by using domain knowledge to assign the independent values to the synthetic individuals in a way that better preserves the structure. It could also be appropriate to ignore the independent attribute and assign the values at random, if the attribute is not significant.

How to parameterize the code

Hyperparameters in our code:
Step 0:

- num_clusters
- The attributes that are part of the archetype

Step 1:

- epsilon
- delta
- The attributes that are independent
- Proportions of budget assigned between the archetype counting and the independent attribute counting

Step 2:

- Max number of records for the synthetic data set