Algorithm for Dimensional Sampling in HoloClean

Step 1 identifies the dimensional attributes which aide to our sampling methodology. It helps us to eliminate training points which generate similar features during featurization and fail to add any new information to the existing model during training.

Algorithm 1 Identify Dimensional Columns

```
procedure get_dim_columns()
    if domain_knowledge is true then
        dim_attr ← import(user_input)
    else
        for a in attributes do
            if a in (list of columns in CD or FD) then
                dim_attr[ ].append(a)
            end if
        end for
    end if
    return dim_attr[ ]
end procedure
```

Step 2 shows the modification made to the domain engine, which iterates through each cell in the dataset to generate the domain. At this stage, we identify which cells should be sampled based on our dimensional attributes. We do not remove any cells here to ensure the initial dataset is entirely available to make the train-test split.

Algorithm 2 Modify Domain Engine to mark sampling cells

```
procedure generate_domain()
    dim_duplicate[ ] ← duplicated (dataset, [group_by = dimensional_attr])
    samp_ind[ ] ← empty_list()
    for t in tuples do
        for a in attributes do
            if (a in dim_attr) AND (t in dim_duplicate) then
                samp_ind.append(0)
            else
                samp_ind.append(1)
            end if
            cell_domain ← <generate domain on each cell with existing code>
            cell_domain.append_attribute(samp_ind[ ])
        end for
    end for
    return cell_domain[ ]
end procedure
```

Step 3 shows the modification made to the data featurizer. Here the training set is generated from the initial input dataset. Existing code filters out cells which are erroneous and for which the weak label is NULL. We add one extra layer to also filter out cells which are not going to aide to the training model, based on our dimensional attributes.

Algorithm 3 Modify Dataset Featurizer to filter training points

```
procedure generate_weak_labels()
    for cd in cell_domain do
        cd.filter(weak_labels is True)
        cd.filter(samp_ind is False)
    end for
    labels ← <Generate training points with cd as input with existing code>
    return labels
end procedure
```