Supplement to "FLOWER: Data Flow in ER Diagrams": Experiment Results

1 Results

This document acts as supplementary material to "FLOWER: Data Flow in ER Diagrams." We share the below information as a document, while the actual data is present in the repository it is contained in. For verification purposes, the code may be run to check results.

basic data set:

This code was run in the main directory with the command

```
python -m __init__.py "data_sources/basic/*.py" -v -o "
    ↪ data_sources/basic_output/data.json -e "data_sources/
    ↪ basic_output/diagram"
```

agg.py

Description: A simple aggregation script, reading in "r.csv", "t1.csv" and "t2.csv". It outputs to "t_ka.csv" and "tjp_out.csv" after performing dataframe operations. Notably, r.csv is not used in any State operations leading to outputs, so it should be dropped from consideration in the description.

Results:

```
"inputs": [
    "t2.csv",
    "t1.csv"
],
"outputs": [
    "t_ka.csv",
    "tjp_out.csv"
]
```

gamma.py

Description: A script used in researching medical data. A number of files are input and output, but only two inputs act as ancestors for outputs, so they are the only resources listed for this Entity. This script is a complex pipeline with many transformations and aggregations performed on each state.

Results:

```
"data_sources/basic/gamma.py": {
    "inputs": [
      "/Users/tahsin/Documents/Python_programs/
          ↪ Phe2_05_values_spikes.csv",
      "/Users/tahsin/Documents/Python_programs/
          ↪ Phe2_02_values_spikes_windowsresults_tranposed.csv"
    ],
    "outputs": [
      "/Users/tahsin/Documents/Python_programs/
          ↪ Phe2_02_values_spikes_windowsresults_reducted.csv",
      "/Users/tahsin/Documents/Python_programs/u_05.csv"
    ]
}
```

mini_gradient.py

Description: A script used in experimenting with batch gradient descent. Notably, the script takes as input a string passed from the command line arguments as the resource string to use. This highlights the difficulty in determining certain aspects of a pipeline.

Results:

```
"data_sources/basic/mini_gradient.py": {
    "inputs": [
      "variable[ds]"
    ],
    "outputs": [
      "gammasgdresults_3.csv"
    ]
}
```

toy data set:

This code was run in the main directory with the command

```
python -m __init__.py "data_sources/toy/*.py" -v -o "
    ↪ data_sources/toy_output/data.json -e "data_sources/
    ↪ toy_output/diagram"
```

cat_costs.py

Description: Summarizes product info and aggregates category inventory and total costs.

Results:

```
"data_sources/toy/cat_costs.py": {
    "inputs": [
```

```
      "products.csv",
      "product_categories.csv",
      "product_types.csv"
    ],
    "outputs": [
      "processed__product_info.csv",
      "processed__category_inventory.csv"
    ]
}
```

cat_sales.py

Description: Takes sale information and determines sales by customer and category.

Results:

```
"data_sources/toy/cat_sales.py": {
    "inputs": [
      "sales.csv",
      "processed__product_info.csv"
    ],
    "outputs": [
      "processed__category_sales.csv"
    ]
}
```

recommend_categories.py

Description: Recommends categories for customers based on transaction history and profit potential.

Results:

```
"data_sources/toy/recommend_categories.py": {
    "inputs": [
      "processed__category_inventory.csv",
      "processed__category_sales.csv"
    ],
    "outputs": [
      "processed__category_recs.csv"
    ]
}
```