Theory and Practice of Data Cleaning CS-513

Olga Scrivner, Fall 2018

https://github.com/obscrivn/finalProject

Data Provenance

Farmers Market Directory is a dataset provided by the Agricultural Marketing Service. This data describes nation-wide farmers markets that adheres to the following conditions i) the market must have two or more farm vendors, ii) the vendors must sell agricultural products directly, iii) the markets must be at a regular location. The aim of the dataset is to provide customers with a variety of information about markets, including location, products, and form of payments, among others (Agricultural Marketing Service, 2018). The website allows for an excel export and API access. The web site does not provide a dictionary and does not specify how data was entered, whether it was transformed or contains issues.

The provided csv dataset has a timestamp of July 17, 2017 with the size of 2.6 MB.

Assessment of Data Quality Dimensions

The initial dataset assessment is based on the six primary data quality dimensions (Ashham et al., 2013) using python.

Uniqueness. The dataset consists of 8687 rows and 59 columns with the following characteristics: Farmers Market ID is represented as integers, columns x (latitude) and y (longitude) are numeric types, and the remaining 56 columns are characters. Farmers ID has 8687 unique values and no null value, which shows that this column can be safely used as unique records id for SQL.

Columns representing type of payments have only two unique values (‘Y’, ’N’), while columns describing food items have three unique values (‘Y’, ’N’ and ‘nan’ or ‘-‘). While there are no null values among payments columns, food items have the same amount of null values - 2933 across all columns.

```
import pandas as pd
dataset = pd.read_csv('farmersmarkets.csv’)
print('N of row:',len(dataset),'; N of columns',len(dataset.columns))
dataset.isnull().sum()
for i in range(23, 58): # payment and food values
        print(dataset.iloc[:,i].unique())
```

Timeliness. Each record has a timestamp entry that varies in its format from 4-digit year representation (e.g., 2017) to a full time stamp (e.g., 6/20/2017 10:43:57 PM). The format information will need to be further reviewed in OpenRefine and normalized for reporting the statistics of how many markets were entered per each year.

Validity. Season Information (Dates, Times) will require further evaluation and normalization, as the current format includes strings (July to November), dates (06/14/2017 to 08/30/2017), and a list of days of the week and time. The new format should be developed so that the user can select a specific time of the year or day of the week to find the market.

Accuracy. Zip represent a 5-digit value with 944 records represented by null values. State column has 53 unique values and zero null values, whereas county has 515 and city has 39 null values. Market names do not have zero values however the accuracy of labeling has to be verified.

Open Refine

The first step consists of reviewing Markets Name and adding pre-processing steps to eliminates cases of spelling inconsistency: i) small vs capital letters, 2) apostrophe as in farmers' vs farmer's. Common transform is used to removed inconsistent spaces, trailing and transforming to lower cases. In addition, grel script was applied to remove apostrophe. This transformation reduced the number of clusters to 16 based on the key collision method and fingerprint function, which was manually checked and applied.

The second step is to review and normalize columns representing timestamps (updateTime, Season1Date, Season1Time, Season2Date, Season2Time). In their current format, the values cannot be converted to timeline facet. By using TextFacet, it is clear that there is a variation in the value types (string values for months - May 26 2010; numeric values - 6/6/2017) and the 4-digit year format (2013) versus date-time format (2/9/2015 6:48:25 PM). In order to normalize this structure and make it useful for a descriptive statistics in reports, all the variations are converted into a 4-digit format (e.g., 2013). First, the string values must be converted to numeric (Apr, Jul, Jun, May): value.replace("Apr ","4/") - 129 cells; value.replace("Jul “,"7/") - 1 cell; value.replace("May ","5/") - 124; value.replace("Jun ","6/") - 49. Note a space after the month. To preserve time values, first the cells are split by space (\s+) and date and timestamps are placed into two new columns. Finally, the date is reformatted into a 4-digit format by using grel. For example, to remove days/month, the following regular expression was used: value.replace(/[0-9]+\/[0-9]+\//,"").

The next transformation is done on Season and Date columns, facilitating the user's search for a specific time of the year when the market is open. Under the assumption that the records represent the existing markets, the value for season does not need an year information (e.g. 04/02/2014 to 11/30/2014), that is, the user wants to know about the season. Some values however contain even less information - only months (July to November). To normalize season values for the simplicity of usage, the string values for months will replace the other formatting. For example, the following expression replaces any string with 6 or 06 by June: value.replace(/0?6\\/[0-9]+\\/[0-9]{4}/,\"June\").

While the exact dates, when available, will be lost, the dataset only exhibits the accurate date information for the year, when the record is inserted. The data integrity will need to be checked for this column, as it is still possible it contains other values that were not cleaned in

OpenRefine.

The remaining columns cleaning included a transformation to a title case for City and County columns, and the replacement of "-" value to Null value for Organic column to match the remaining food columns values. Furthermore, the columns FMID, x, y, zip and Update were transformed to number type. Finally, the single apostrophe was replaced by double single quote (value.replace("'","''"): Website - 5; Facebook - 53; Twitter - 4; Youtube - 2; OtherMedia - 4; street - 72; city - 11; County - 30;)

The history is exported from OpenRefine (http://guides.library.illinois.edu/openrefine/exporting).

Database

The modified farmers market data is exported using SQL exporter. The schema to create a table for database is shown in Table 1. ID is a primary unique key with the constraint for null value and a length of 10. The column x and y are INTEGER, the remaining columns VARCHAR.

Table 1. SQL Schema

```
CREATE TABLE "farmers" (
ID varchar(10) UNIQUE PRIMARY KEY NOT NULL,
MarketName VARCHAR(255) NULL,
Website VARCHAR(255) NULL,
Facebook VARCHAR(255) NULL,
Twitter VARCHAR(255) NULL,
Youtube VARCHAR(255) NULL,
OtherMedia VARCHAR(255) NULL,
street VARCHAR(255) NULL,
city VARCHAR(255) NULL,
County VARCHAR(255) NULL,
State VARCHAR(255) NULL,
zip VARCHAR(255) NULL,
Season1Date VARCHAR(255) NULL,
Season1Time VARCHAR(255) NULL,
Season2Date VARCHAR(255) NULL,
Season2Time VARCHAR(255) NULL,
Season3Date VARCHAR(255) NULL,
Season3Time VARCHAR(255) NULL,
Season4Date VARCHAR(255) NULL,
Season4Time VARCHAR(255) NULL,
x INTEGER NULL,
y INTEGER NULL,
Location VARCHAR(255) NULL,
Credit VARCHAR(255) NULL,
WIC VARCHAR(255) NULL,
WICcash VARCHAR(255) NULL,
SFMNP VARCHAR(255) NULL,
SNAP VARCHAR(255) NULL,
Organic VARCHAR(255) NULL,
Bakedgoods VARCHAR(255) NULL,
Cheese VARCHAR(255) NULL,
Crafts VARCHAR(255) NULL,
Flowers VARCHAR(255) NULL,
Eggs VARCHAR(255) NULL,
Seafood VARCHAR(255) NULL,
Herbs VARCHAR(255) NULL,
Vegetables VARCHAR(255) NULL,
Honey VARCHAR(255) NULL,
Jams VARCHAR(255) NULL,
Maple VARCHAR(255) NULL,
Meat VARCHAR(255) NULL,
Nursery VARCHAR(255) NULL,
Nuts VARCHAR(255) NULL,
Plants VARCHAR(255) NULL,
Poultry VARCHAR(255) NULL,
Prepared VARCHAR(255) NULL,
Soap VARCHAR(255) NULL,
Trees VARCHAR(255) NULL,
Wine VARCHAR(255) NULL,
Coffee VARCHAR(255) NULL,
Beans VARCHAR(255) NULL,
Fruits VARCHAR(255) NULL,
Grains VARCHAR(255) NULL,
Juices VARCHAR(255) NULL,
Mushrooms VARCHAR(255) NULL,
PetFood VARCHAR(255) NULL,
Tofu VARCHAR(255) NULL,
WildHarvested VARCHAR(255) NULL,
updateYear VARCHAR(255) NULL,
updateTime2 VARCHAR(255) NULL
);
```

The data is loaded with the execution time of 2 seconds. Several constraints are checked using sql scripts to test duplicate, non-null values and the length of id characters. None of the constraints are violated:

```
select id, count(id) from public.farmers group by id having count(*)>1;
select id from public.farmers where id is null;
select id, length(id) from public.farmers where length(id)>10;
```

Despite the OpenRefine transformations, SeasonDate and UpdateYear columns still require some formatting cleaning.
First, the character length check shows that UpdateYear has 69 records with length < 4 instead of 4-digit year value and 234 records have backslash (e.g., 4/25). There records are replaced with null values and the table is updated.

```
update farmers SET updateyear = 'NA' where length(updateyear)<4;
update farmers SET updateyear = 'NA' where updateyear ILIKE '%/%';
```

These two modifications allow for a better statistical overview of data entry records:

Table 2. Data Entry Records

updateyear	count
2009	1606
2011	698
2012	648
2013	955
2014	1506
2015	866
2016	1373
2017	732
NA	303

Similarly, the column Season1Date has a number of formatting inconsistencies:

```
select season1Date, count(season1Date) from farmers group by season1Date;
```

For example, not every record supplies information indicating the end of season or the date is shown as a month or day, month and year, as illustrated in Table 3.

Table 3. Sample of Inconsistent Formatting

December	1
June 25, 2011 to September 24, 2011	1
March to November	24
August to October	14

Two steps are undertaken: removal of all non-alphabetic characters and splitting column to startSeason and endSeason, where null values can occur.

```
update farmers SET season1Date = regexp_replace(season1Date, '[^a-zA-Z]', ' ',
'g’);
alter table farmers add column startseason varchar, add column endseason
```

```
varchar;
update farmers SET startseason = split_part(season1Date, 'to', 1), endseason =
split_part(season1Date, 'to', 2);
update farmers SET endseason = regexp_replace(endseason, 'Oc', 'October',
'g');
```

Note: the last replacement is needed to repair the output from splitting by ‘to’.

Workflow

The overall workflow is demonstrated in the Figure 1 and workflow with parameters is shown in Figure 2:

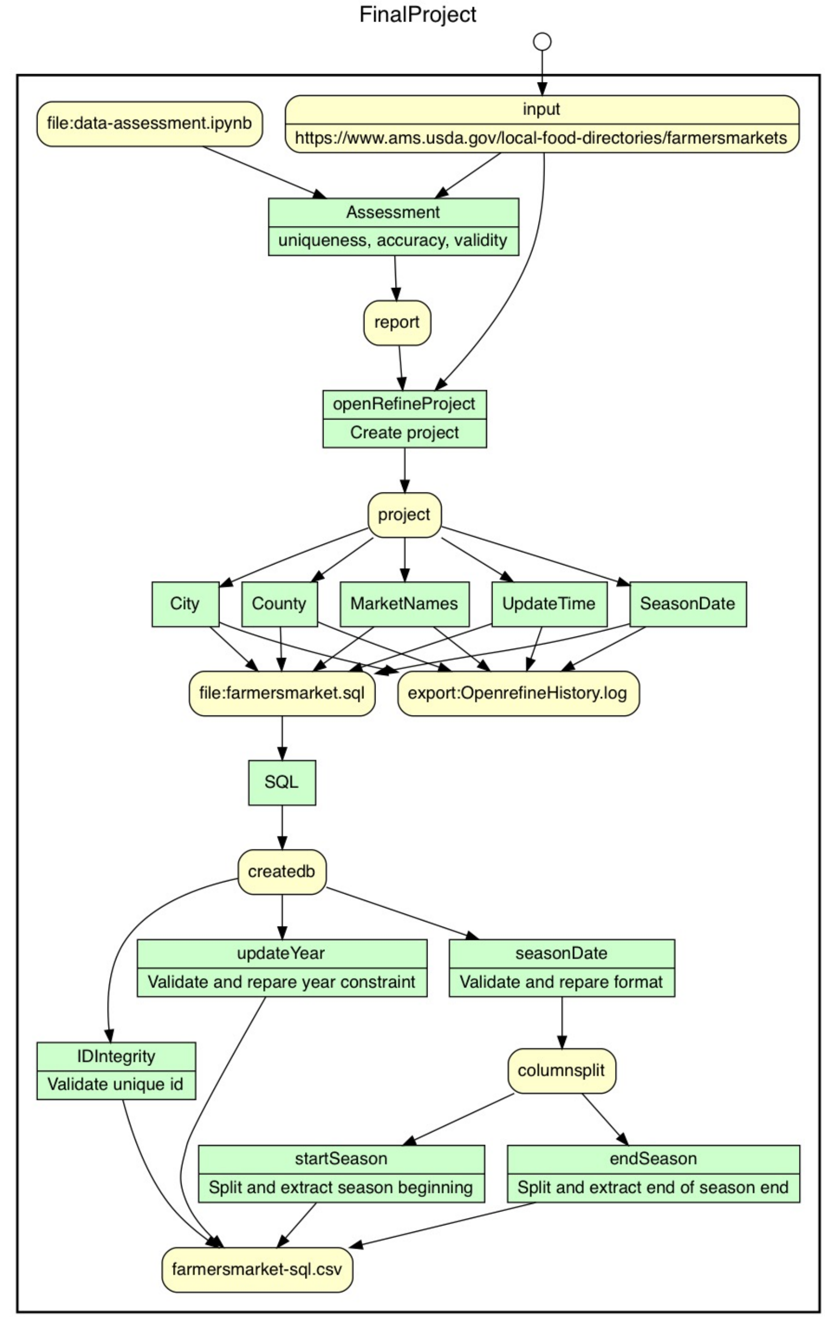
FinalProject
input
https://www.ams.usda.gov/local-food-directories/farmersmarkets
file:data-assessment.ipynb
Assessment
uniqueness, accuracy, validity
report
openRefineProject
Create project
project
City
County
MarketNames
UpdateTime
SeasonDate
file:farmersmarket.sql
export:OpenrefineHistory.log
SQL
createdb
updateYear
Validate and repare year constraint
seasonDate
Validate and repare format
IDIntegrity
Validate unique id
columnsplit
startSeason
Split and extract season beginning
endSeason
Split and extract end of season end
farmersmarket-sql.csv

Figure 1. Farmersmarket workflow

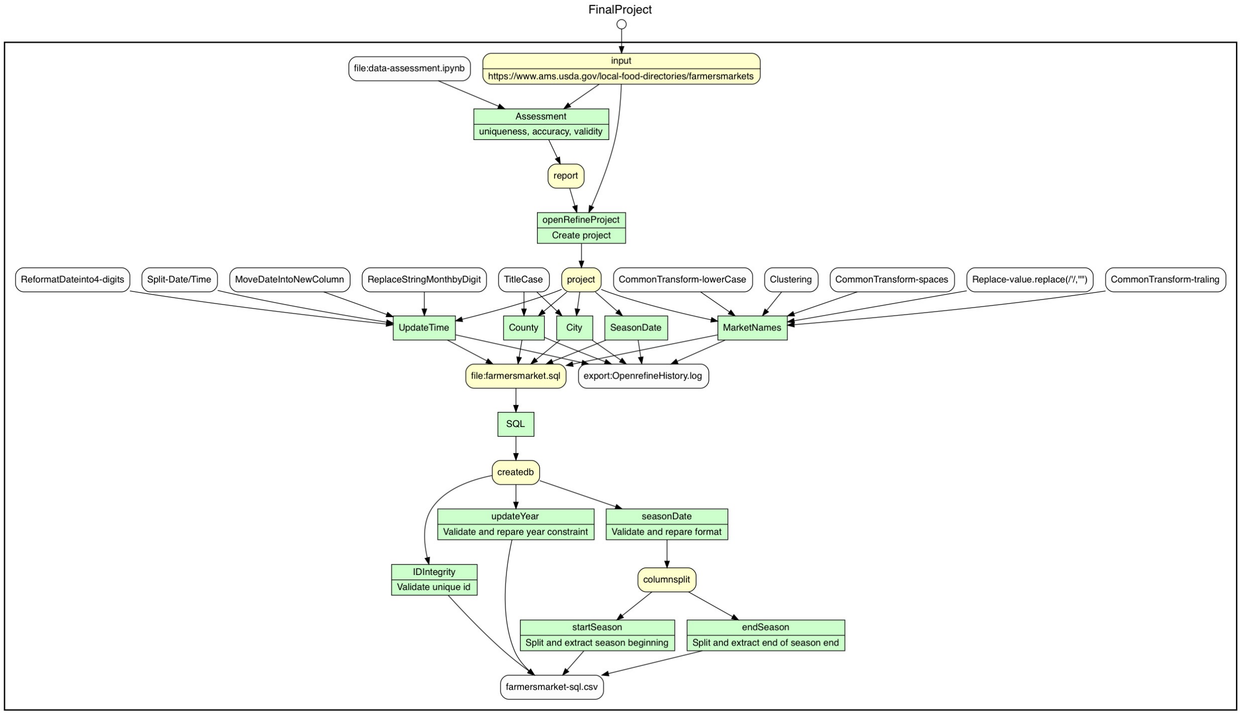

Figure 2. Farmersmarket workflow with all parameters

Conclusion

Data assessment, data cleaning, and transformation were performed using python, OpenRefine, and Aqua Fold Studio tools. As a result, the data is appropriate for users who are interested in 1) viewing the records history (e.g., how many new records are added every year), 2) selecting seasons by months, 3) querying existing unique markets for payment/food categories. On the other hand, the current data is not complete in terms of location (missing zip numbers) and none of the social media links are verified if they still exists.

References

Agricultural Marketing Service. United States Department of Agriculture. (2018). Local Food Directories: National Farmers Market Directory | Agricultural Marketing Service. Retrieved November 30, 2018, from https://www.ams.usda.gov/local-food-directories/farmersmarkets
Askham, N., Cook, D., Doyle, M., Fereday, H., Gibson, M., Landbeck, U., … Schwarzenbach, J. (2013). *The six primary dimensions for data quality assessment. Defining Data Quality Dimensions*. Retrieved November 30, 2018,
from https://www.whitepapers.em360tech.com/wp-content/files_mf/1407250286DAMAUKDQDimensionsWhitePaperR37.pdf
report - https://blog.ouseful.info/2015/09/04/converting-spreadsheet-rows-to-text-based-summary-reports-using-openrefine/
https://www.datacamp.com/community/tutorials/python-data-profiling
https://guides.library.illinois.edu/openrefine/textdiscovery
Tools:
http://try.yesworkflow.org/