**Farmer Market Search!**

**A Case Study in Data Cleaning and Provenance**

**Submitted By: Pragya Mishra**

**(**CS-513 Theory and Practice of Data Cleaning)



1. **Introduction and Overview**

This data cleaning project will aim cleaning U.S. Farmer Markets dataset publicly available at USFarmersMarkets(<https://www.ams.usda.gov/local-food-directories/farmersmarkets>).

The Farmers Market Directory lists markets that feature two or more farm vendors selling agricultural products directly to customers at a common, recurrent physical location. Maintained by the Agricultural Marketing Service, the Directory is designed to provide customers with convenient access to information about farmers market listings to include market locations, directions, operating times, product offerings, accepted forms of payment, and more.

**Use Case**

One use case of this could be: **“*Give me all the clusters (group of close by cities near me) where there is a farmers market which provides both seafood and vegetables?”*** The user will want to minimize his transportation time and in the same time get all the products he needs.

**Data Cleaning Goals**

As of now, the way to fetch the desired data is **to filter all the records that have ‘Y’ for Seafood and Vegetables and are currently functioning now** (Select the current season date and time and check if the date and time are current) and then estimate the distance between the customer’s current location and the market location based on latitude and longitude and Haversine formula. **The query should account the time it will take the user to arrive in the desired location and select only the closest markets which will be open at least 30 minutes after ETA of the user.**

In order the query to be optimal the date sparsity should not be like it is in its current condition. For instance, the latitude and longitude should be accurate and able to form a cluster based on users’ location and trying to minimize the radius of the cluster. There are Geo Spatial and proximity search algorithms which can be applied here. Also, another difficulty to query data is the date and time the market is opened. Having the concepts of seasons, it is hard to determine the boundaries of each season so that the query will know which column to check for datetime to match the user’s desired datetime. Basically, the query should go over all the 8 season/date/time columns and look for this time interval. Furthermore, the values in date and time columns follow 2-3 different formats which will make it difficult to select and filter without loading all the data into memory, converting these column’s values into a consistent format and then apply the filtering.

**Are there use cases for which the dataset is already clean enough?**

A use case which will require very less or no cleaning or data preprocessing will be answering the following question: *“****Find all the markets for given zip code where I can use credit card?”***

Both zip and Credit columns are consistent and queries running against them can be fast without additional data manipulation.

**Conversely, are there use cases for which the dataset will never be good enough?**

**One use case in which the data will never be good enough is if someone want to export the prices of all the goods a market** provides by web crawling their website or social media link. The reason for this is that **most of the links are not valid or not present** which makes it impossible to be able to access all of them successfully and scrape some information of their web sites

**2. Initial Assessment of the Dataset**

**Structure and content of the dataset and quality issues that are apparent from an initial inspection**

The farmers market dataset contains 59 columns. Broadly classifying columns as below for better understanding:

1. **FMID** is identifier for each farmer’s market and contains 7 digits number.

2. **Market name** which is free text column and allows special characters.

3. **MediaSites**: Next five columns (Website, Facebook, Twitter, YouTube, OtherMedia) are supposed to contain information that will uniquely identify the market in one of the world wide web media sites. Most of the times this information is an URL to their page but there are no restrictions or validation so that sometimes the value in these columns is free text. All the five columns support blank or null values.

4. **Market Address :** Next five columns (street, city, country, state, zip) are supposed to provide the address that will uniquely identify the market on the geographic map. All of them are free text columns, even the ZIP code column contains some special characters like ‘-‘. However, one **quality issue** is that some of the **names are lower case, some capital case, other have whitespaces before or after the string**. This will cause issues if the user wants to do direct string comparison. Also, a good way to avoid that and establish unification over these names of countries, cities and streets can be for example to give each country a unique identifier and map the row that should belong to that country to the identifier instead.

**5.Season Date Time:** The following eight columns containing the date and time for every season most likely represent the periods when the market is opened. The initial assessment clearly shows that these **columns also are not consistent in terms of format**. There are occurrences where the period is stated as month A to month B and in the same time other occurrences where period is more granular for example from date to another date.

The **X and Y columns represent latitude and longitude**, although the **names of these columns are not meaningful enough** and the only way for a user to understand that is to see the column values. Another not so meaningful name is ‘**location’**. The understanding is that it will contain map/geographic specific information, however it looks like it is more of **a description of the place** where the market is held.

6. **Market characteristic:** The next **35** columns **are Boolean columns** containing **Y-true and N-false** for given characteristic of given market. A easily **distinguishable data issue** is that sometimes they have a **third or fourth value option like ‘-‘ or empty string** which most likely indicates these tuples do not have neither false or true values for these columns.

7. **UpdateTime** : The last column (updateTime) representing date and time when the record has been updated also does **not** follow **consistent** **datetime** format. **There are records with only year**, records **with** **full date time** and records where **month is present with a word instead of a digit**.

Some of the major issues in this dataset are **the inconsistency of the format of data values** for most of the columns. This is caused because of all **the columns are free text columns** and probably the software which inserts the data in this data source does not have any validations. The consumer of this data cannot rely on any kind of consistency and operations like comparison will always be tricky unless the consumer does not have any logic to convert the current column format to one of the ISO standardization formats. Moreover, some of the values are meant to mean the same thing but are written a little differently missing a letter or having special characters or having different casing. The data does not violate **First or Second Normalization database forms because the FMID is unique and there are no duplicates of this key having different values as attributes.** However there is still a lot of **redundancy in terms of the way different types of products are associated with a market.**

1. **Data Cleaning methods and process with Open Refine**

Given all the data present in .csv format here are some of the columns and the transformation that were done on them. OpenRefine has been used mainly to **remove bad/special characters,** **clustering based on similarity function**, **splitting the data based on some criteria** and **converting data to specific ISO format for consistency.**

The consolidated **farmersmarkets.csv was split into 11 sub csv files** which contain the same data having some relations between the different tuples. This was done to **reduce the data repetitions.** Now counties, cities, states etc. have their separate data csv files. On top of this the following several data cleaning operations were performed on some of the columns. The **provenance information is saved in “provenance.json”** file which is in the “2. OpenRefine” directory.

**Transformations over FMID:**

**Verify FMID contains only unique values:**

\***Sort** the values in FMID.

\***Blank down** and verify that 0 cells were blanked down.

**Transformations over City, Country, State:**

**\*Remove the special characters** –

\*%@#!\[]()?'\* using value.replace(/%@#!\\\[\]\(\)\?/,"")

A screenshot of a cell phone

Description automatically generated

**\*Trim leading and trailing white spaces and collapse consecutive white spaces.**

**\*Make a facet and perform the cluster operation** using the \*key-collision\* method and \*fingerprint\* function. Merge the relevant clusters.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

**Transformations over Season1Date:**

**\*Remove the special characters** –

\*%@#!\[]()?'\* using value.replace(/%@#!\\\[\]\(\)\?/,"")

\***Split the column** values using \* to \* as separator, I got 2 columns as a result.

A screenshot of a cell phone

Description automatically generated

**Rename** the first column as \*Season1DateStart\* and second column as \*Season1DateEnd\*.

\***Convert the values in the column to ISO** 'd/M/y' format using value.toDate('d/M/y').

A screenshot of a social media post

Description automatically generated

**Repeat** 1, 2 and 3 for \*Season2Date\*, \*Season3Date\* and \*Season4Date\*.

**Transformations over Season1Time:**

**\*Remove the special characters** - \*% @ # ! \ [ ] ( ) ? '\* using value.replace(/%@#!\\\[\]\(\)\?/,"")

\***Split the column values using \*;\*** as separator. Each of the new columns will represent values for given day of the week.

\***Repeat** 1 for \*Season2Time\*, \*Season3Time\* and \*Season4Time\*.

**Transformations over updateTime:**

\***Convert** the values in the column to ISO 'd/M/y H:m:s' format using value.toDate('d/M/y H:m:s').

**4. Data Cleaning Results**

**Schema:**

ER diagram given below represents the data structure after cleaning with OpenRefine and importing the data into SQLlite database. In order my database schema to satisfy the fourth normal form a lot of manipulations for removing the data redundancy and improving the data reusability were done. Having the schema below it allows adaptability, expandability and high performance once the data becomes part of a real system which interacts with it. The main idea behind the division is to use the concept of dimensions and facts.

A screenshot of a cell phone

Description automatically generated

**Integrity Constraints Check**

**latitude and longitude should be in the interval [0,90], [-180,180]**

A screenshot of a cell phone

Description automatically generated

**There are no inactive markets. There should be at least one datetime per market in seasondates. And the datefrom and dateto values should not be empty**

A screenshot of a cell phone

Description automatically generated

**Select all the markets that have seafood and vegetables and are opened in November**

A screenshot of a cell phone

Description automatically generated

**All the markets should have non-empty city, state and county**

A screenshot of a cell phone

Description automatically generated

**No empty names for cities, states, counties**

A screenshot of a cell phone

Description automatically generated

**There should not be two markets that are in the same location during the same time**

A screenshot of a cell phone

Description automatically generated

All of the scripts related to the SQL part of the project were included in sqllite-commands.txt under “3. SQL” directory. The scripts were run in a terminal to create schema, import the data from csv and export it to SQL based insert queries. “farmersmarket\_schema.sql” is the file containing the plain schema implementing the relations between the tables including foreign and primary keys and other constraints like field types.

**Creating A Workflow Model With YesWorkflow**

I used the YesWorkflow on-line editor to create the workflow graph for whole process. Firstly, the data root is the farmersmarkets.csv which got split into multiple files and cleaning using OpenRefine was performed over each of these files. After the cleaning the diagram represents how each of the files got imported into corresponding SQL table and constraints check was executed over all the tables to verify whether the data has the needed quality to satisfy the use cases mentioned in the Initial Assessment document.

A screenshot of a cell phone

Description automatically generated

1. **Challenges**

The most challenging problem within data cleaning remains the correction of inconsistent values and invalid entries. In many cases, the available information on such anomalies is limited and insufficient to determine the necessary transformations or corrections, leaving the deletion of data, though, leads to loss of information; this loss can be particularly costly if there is a large amount of deleted data.

**6. Conclusion**

Data Cleaning or Data Wrangling is not only an effective tool for removing the unwanted data “dirty” data, but also the medium to make data in our system selective, concise and appropriate in order to perform the better analysis. Based on our experiences , I can say Data Cleaning is one of the most time-consuming steps in any Data Analysis.

I tried to clean the data using OpenRefine and SQLLite. It took me significant time to clean the data using OpenRefine, specially because of the way it enables its usage, i.e. through GUI- with manual steps.

Tools like OpenRefine have their limitations in terms of how much data can they handle; and that’s why more people are inclined towards scription methods with Python, R etc. but they do not provide the visibility that OpenRefine does. We are long way from one perfect solution to Data Cleaning; but with advancement in tools. l a lot of tedious task busden can be transferred to machines.

1. **Next Steps- Visualizing Data**

Data Visualization is one of the most effective techniques for gaining insights from data.

