# CIT 594 Group Project Report

Yuyuan Lin(linyuy@seas.upenn.edu), Zhong Liu(zhongliu@seas.upenn.edu)

## Additional Feature

The topic of the additional feature in our final project is the relationship between the Total Fines Per Capita and the Total Residential Market Value Per Capita. In statistics, we can use the cross-correlation coefficient to quantify the relationship. There are several types of correlation coefficient formulas. One of the most used is the Pearson's correlation coefficient formula, which is shown below.

$$CC = \frac{\sum_i (I_1(i) - \overline{I_1}) \cdot (I_2(i) - \overline{I_2})}{\sqrt{\sum_i (I_1(i) - \overline{I_1})^2 \sum_i (I_2(i) - \overline{I_2})^2}}$$

In the equation, let us use $I_1(i)$ as the Total Fines Per Capita in each zip code, and $I_2(i)$ as the Total Residential Market Value Per Capita in each zip code. $\overline{I_1}$ is the average Total Fines Per Capita in all zip codes of interest. And $\overline{I_2}$ is the average Total Residential Market Value Per Capita in all the zip codes of interest. Therefore, the entries of $I_1$ have used the datasets of population and parking and those of $I_2$ have used the datasets of population and property.

In the implementation, for simplicity, we have used a derived form of the Pearson's correlation coefficient formula, which is shown below:

$$CC = \frac{cov(I_1, I_2)}{\sigma_{I_1} \sigma_{I_2}}$$

, where $cov(I_1, I_2)$ is the covariance, $\sigma_{I_1}$ and $\sigma_{I_2}$ are the standard deviations. Essentially the calculation of covariance, standard deviation, variance is similar. Hence using this equation avoided writing similar codes.

Now let us explain the physical meaning of the cross-correlation coefficient. The range of the coefficient is [-1, 1]. If the coefficient is close to 1, then the two features are positively correlated, i.e. a larger value of feature 1 results in a larger value of feature 2. If the coefficient is close to -1, then the two features are negatively correlated, i.e. a smaller value of feature 1 results in a larger value of feature 2. If the value of the coefficient is close to 0, then the two features are almost independent.

We verified the results of our implementation with a few online Pearson's correlation coefficient calculators with some real and artificial data sets. The results are the same. For example, using the artificial data below:

| Zip Code | Total Fines Per Capita | Total Residential Market Value Per Capita |
|---|---|---|
| 19103 | 0.9 | 62.5 |
| 19104 | 1.69 | 40 |
| 19106 | 0.72 | 20 |
| 19107 | 3.6 | 600 |
| 19123 | 1.8 | 250 |

Intuitively we can see that the larger value of total residential market value per capita results in a larger total fine per capita, there must be a positive correlation. Using the on-line correlation coefficient calculator[1], the coefficient is 0.944. In our implementation, we have 0.944.

The final result using the provided input data sets is shown below:

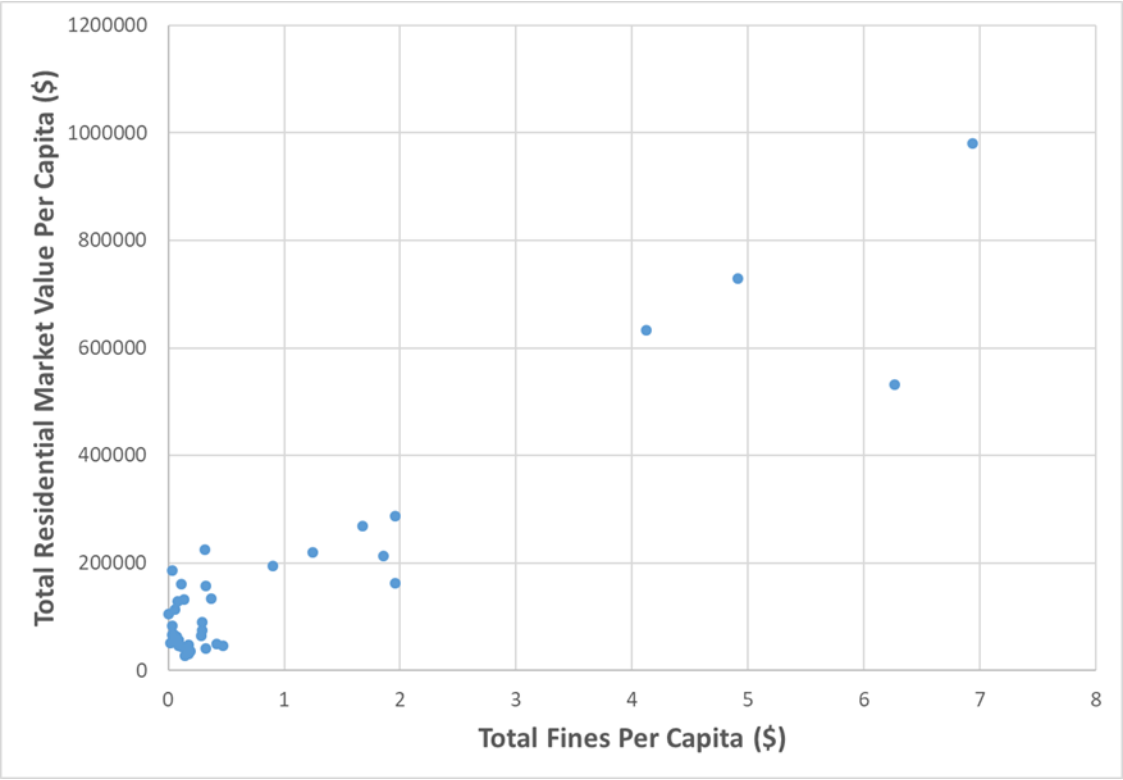


Figure 1

As shown in the figure, there is a positive correlation between the Total Fines Per Capita and the Total Residential Market Value Per Capita. Indeed, the calculated correlation coefficient is 0.9392, which indicates a strong positive correlation. However, the data points are not distributed uniformly. Without the largest 10 high market value per capita areas, the correlation is low.

By looking at the data points with large residential market values per capita, such as areas in the zip code 19102 and 19103, they are all located in the downtown areas, which usually has high residential market value, limited parking space, high demand of parking, and highly restricted parking rules. However, the residential population of those areas is not necessarily high. Indeed, the population of 19102 and 19103 is 4705 and 21908 respectively. On the other hand, in the other areas such as in 19153, although there is a similar population (12259 in 19153), the total fine per capita is only $0.0325. The region of zip code 19153 is located near the airport, where residential parking usually is not an issue.

Therefore, if we consider all the zip codes in the data set, there is a strong positive correlation between the Total Fines Per Capita and Total Residential Market Value Per Capita. This may be attributed to the high property value and high parking violation numbers in the downtown areas. Other than these areas, the correlation is low.

## Use of Data Structures

### Array

We have used the array data structure for a temporary storage of the data line entries for csv type files in the reader classes. We have used this data structure for the following reasons:

(1) Array is associated with the string split() method in java API. So it is straightforward and convenient.
(2) We only need a data structure for simple temporary data storage and quick access to the data in it. We can quickly access to the data of interest in array with the O(1) time complexity. There is no need of other operations such as insertion, search, and deletion after the data of interest of retrieved, as the data of interest is saved in the data object.
(3) Our input file has a relatively fixed format. And the meaning of each column in the csv files is fixed. So we have already known the index of data. There is no need of looping over the data structure again to find a value after the split() method is done. The other data structures may be helpful if we need the additional functionalities, for example, a linkedlist with O(1) time complexity in insertion and deletion, compared to the O(n) complexity using array. But these functionalities are not used here. We only need a data structure with quick access, which can be provided by the array with the O(1) time complexity.

### HashMap

We have used HashMap to store and update the values associated with certain keys. For examples, in the property reader class, we have used the zip codes as the keys and the total market values as the values. The total market values are updated for each new entry. The memorization or cache of answers to several of the six problems also uses HashMap, where we cache calculation results by each zip code as a key and the corresponding result as the value. See **UserInterface** class for details.

We have used the HashMap data structure for the following reasons:

(1) We need a data structure that can store and update key-value pairs. And the order of the key-value pairs is not of interest.
(2) We need to do access and insertion for many times. The time complexity is extremely important. The HashMap has complexity of O(1) for insertion and lookup, which is perfect for our purposes.
(3) We can use LinkedHashMap for the similar time complexity in terms of the insertion and lookup. The LinkedHashMap will keep the order in which key-value pairs are inserted. But this is not necessary.

### TreeMap

We have used tree map for question 2 -- Total Fines Per Capita in the ParkingViolationProcessor class. The information must be displayed on screen in the ascending order. And the information displayed is a key-value pair.

We have used the data structure for the following reasons:

(1) We need a data structure that can store and update key-value pairs. And the order of the key-value pairs is of interest.

(2) We need to do access and insertion for many times. The time complexity is extremely important. TreeMap has complexity of O(logN) for insertion and lookup. So it is not very efficient if we have a dataset with a size of half a million. However, the HashMap has complexity of O(1) for insertion and lookup. Therefore, we have decided to use the HashMap to create the complete set of key-value pairs with quick accessibility first. At the end, the HashMap only have a size of about 50 key-value pairs. And then we convert the HashMap to TreeMap by looping over the entire data structure. The TreeMap is used for display in the end.

**Reference:**

[1] https://www.socscistatistics.com/tests/pearson/