Big Data Project 1 by David Kim and Katherine Holotko

Introduction

In this assignment, teams were given an opportunity to display the knowledge and skills gathered and learned about datastore technologies. For this project we began by trying to use the NYC metro data set, but because the data was very disparate and there would be design/formatting issues within these datastore technologies, we decided to go with one dataset that would be more accessible to work with and formulating statistical analyses with it. We decided to go with median housing sales prices aggregated into months over about a 10 year period from Zillow from all the cities that Zillow was located in.

To import the data, we designed an R script (TransposeCSV.R) that would alter the data slightly, and then imported the csv to our database as table uscities (SQL tables are under Katherine's database, keh384 as well as csv/R/sql files on IBM cloud under /home/2019/nyu/spring/6513/keh384/P1 directory).

Linear Regression

One of the questions that our team first posed about the dataset was "Does the date that Zillow came into the market in a specific city affect the housing prices in that city 2, 5, etc years down the line?" This was a good question to ask, however a better question, which was a bit more specific that came up was, "How does the linear regression slope of US compare to different states' LR slope?" We thought that using the US as a baseline could effectively show whether or not zillow COULD have had an effect in those markets comparatively. We were not concerned with the y intercept as we were looking for the change comparisons.

Because our dataset was incomplete for many of the cities, we decided to only look for the time frame in which all the data was present for every city so that we could get a more complete overview of the city data scope from a more data fidelity perspective. We found that the time frame was to be handled between December 2016 to December 2017. We decided to handle the calculations directly from MySQL as it was an easier way to handle the import of data and manipulation and we required ACID for this sort of data type.

In order to implement Simple Linear Regression, we utilized the Least Squares Method, while normalizing the data to be able to handle this equation. Because of the time series dataset we were working with was difficult, as the year/month data type isn't a discrete variable, we decided to convert it to be discrete. We gave each of the months we were working with a value from the start date (0, 1, 2, 3… 13). We then created multiple tables to hold the manipulated data along the process of figuring out the slopes for organization purposes, which is sort of inspired by how Mongo stored its data in documents. We treated the different tables as aggregates of the data into different documents and outputting them to different tables and performing calculations from there.

We held the price (y) and time (x) data in table uscities, the x - avg(x) and y - avg(y) in linearregressionxyminxybar table, the multiplication of those values and the squares of the time data in linearregressionfinalsteps, and then the final output in a different table for ease of access to the linear regression slopes in finallinearregression. We finally were able to query the linear regression slopes for each of the cities that Zillow was available in between Dec 2016 to Dec 2017 and from our analysis we found that 201 of the cities that Zillow was in outperformed the U.S. for that time frame and 278 underperformed. This could be from a plethora of factors including the time frame itself and analyses in to other factors could prove useful for future analyses.

All of our queries related to Simple LInear Regression was associated with the file LinearRegression.sql which can be run from MySQL providing the command line with the source of the file. This is with the assumption that the uscities table is active with the data present. In the R file, there are comment headings that explain the usage of certain portions of the script that were used to create these extremely long SQL queries and pasted into the LinearRegression.sql script.

Standard deviation of Larger versus Smaller cities

Another question our team asked is whether the price fluctuation over time would be greatest in a large or small city (see query in StandardDeviation.sql). Our intuition said the price fluctuation would be smaller in a small city, because there are less people trying to live there, and therefore less market fluctuations with supply and demand. However, the data revealed a different conclusion:

Big city avg standard deviation = $6208.8416
Small city avg standard deviation = $16748.8464

Because the results were over multiple cities, our data is more reliable and resists some of the influences such as local policy change that could affect housing prices. This result could be reflective that because there are more housing options in a bigger city, any fluctuations tend to have a smaller overall effect, whereas in a smaller city any one fluctuation has a bigger effect. The data could also indicate stronger regulations in bigger cities that prevent higher prices for example, or that prices in bigger cities could not increase or decrease dramatically while remaining profitable.

In order to arrive at average standard deviation of bigger vs. smaller cities, we ran a query across the uscities table which was imported as described above. The query found the standard deviation across the 13 months of interest for the top 10 largest cities. Within the same query, we calculated the average of each of those sets.

Problems

There were a few problems we encountered with this project, which included the following:

1. Deciding on the usage of datastore technology
2. Realizing the need for certain concepts like ACID as opposed to CAP for our specific dataset and analyses
3. “Moving the algorithms to the data” and the inefficiency of our procedures
4. Data that wasn’t two dimensional

When we were deciding on data sets to use, we believed in the idea that we had to use a certain type of data set for the datastore technology we were going to use, namely Mongo. We wanted to find the dataset that would fit Mongo better, but later on we realized that the point of why these technologies exist is to utilize the power of these datastore technologies and find what type works for the data we will use and what is needed for our data.

Which brings us to the next problem we had was that we had primarily focused on utilizing Mongo for our analyses. We put a significant amount of work into Mongo, however the way that Mongo was presenting our data was placing our data in separate fields that could not be of the best usage for us as there wasn’t a way to do operations across different fields; Mongo’s assumptions was to have consistent data across the fields and perform calculations within them and present them from the aggregates in different documents. While Mongo certainly was interesting and could have been powerful in a different sort of analysis, we decided the best steps forward to implement ACID was to use MySQL as our datastore tech.

Some things that our team had trouble with and had disagreements on was the way that we would structure our database so that it could perform calculations, or if it was possible to perform these sorts of statistical analyses from a RDBMS. As the dataset we had was not perfect (none are), we had to come up with a design for our system that would be able to perform the sorts of calculations that we had in mind, even if it meant the cost of space efficiency. This was especially difficult because our data was similarly shaped in terms of columns and rows, so there was not a clear best way to approach the problem. Though our method/approach was a bit inefficient in those regards, it was a proof of concept that it was possible to perform the sorts of statistical analyses from within the datastore technologies, “moving the algorithms to the data”, and not needing to rely on other software/packages.

We also spent a sizeable amount of time trying to answer the question of whether or not the size of city correlated with the average sales price in that area. However, using the formula for correlation proved to be difficult, requiring a lot of data manipulation (as can be seen in the Correlate.sql file). For example, the size of city was listed along with the pricing data as if it were the same, and we had to find ways to separate it by listing as the column name or through SQL filters. In addition, since doing aggregations along many columns in SQL is not very feasible, we

had to transpose the data to put it in row format, in order for aggregation formulas such as SUM to work. Since the maximum amount of columns that can be displayed on the shell is 255, we looked at every 4th city in size, such as the 1st largest, the 5th largest, etc. until the 477th largest.

By the end of the correlation calculation, even though it appears the steps were followed correctly, the data returned ended up being meaningless, as correlation coefficient should be between -1 and 1, and we were getting data outside that range. We believe this is because the correlation is calculated as a combination of 2 variables, but our case is actually a combination of 3 variables (time, size, and pricing), meaning some simplifications/assumptions are probably needed in order to reduce the problem.

Conclusion

During this project, we learned the following:

1. The power of different datastore technologies such as relational and non-relational databases, and that it should be how each tool is used to fit the data and not the other way around.
2. With the tools that we are given, if they cannot do the sorts of analyses that we want, we could engineer solutions with them.
3. Doing data analysis only in SQL without any additional systems can be tricky as well as redundant.
4. We also learned that more important than the implementation, design was the key to the success of this project. Being able to design the database, how we planned to implement these statistical measures, and coming up with a plan was how we successfully produced our results.

Contributions

David Kim: Wrote R helper scripts and a SQL script for the Linear Regression analysis that was performed for each of the cities in the database. Contributed to report for Linear Regression, problems, and Introduction. Contributed to the resolution of a plethora of misconfigurations that existed in getting our database set up and running.

Katherine Holotko: Wrote some R scripts to dynamically read in and create some raw Create/Import SQL statements for SQL. Wrote R scripts to transpose data and print to csv. Helped develop SQL scripts for Correlation and Standard Deviation. Contributed to report for Problems and Standard Deviation section. Contributed to solutions for misconfiguration issues (there were many).