Big Data Final Project

Cook County Mortgages and Deaths

Brian Whitehouse

Overview:Cook County has many diverse neighborhoods (represented by zip code). In these neighborhoods population shift and change are constantly occuring certain

groups are getting older and dying off (Ie. the original immigrant communities that made up little italy, ukrainian village are no longer alive and living there). Certain areas

are getting more dangerous and have lots of violent crime. My goal is to see if the Cook County medical examiners report can show a correlation between average mortgage

value and the percent of deaths by cause (HOMICIDE, SUICIDE, NATURAL, ACCIDENT, UNDETERMINDED).

I began by working the mortgage data and then with the medical examiners data to create to seperate Hive tables that I would join based on zipcode in hive and then upload to HBase for my serving layer.

Batch for mortgage

-Original data set, "Cook\_County\_Recorder\_2013\_through\_March\_27\_2015", contained: mortgages, corrected mortgages, quit claim deeds and lis pendends foreclosure ("forclosure"). This data set had however had the final field for location on multiple lines this made it impossible to read in using the CSVReader later on. In order to correct this and purge some of the data (ie. missing zip codes or zip codes of 00000-0000) that had missing fields I wrote a python script, "filter\_csv.py", that can be run from the command line that takes in the location for the input csv file and output csv file. The file returned I named "Filtered\_CC\_Mortgage\_2013\_through\_March\_27\_2015.csv". However, during this process of filtering out empty I realized that when Cook County records a forclosure it does not submit an amount for said forclosure. In addition, forclosures make up minority of the data set (after cleaning the data there were only 103 out of 300k+ ) and 9 quit claim deeds. So I removed quit claim deed and forerclosures from the data set leaving mortgages and corrected mortgages. For the reasons I filtered down the original data set from 511,167 values to 317,082 values and after removing zipcodes with 00000, forclosures and quit claim deeds got that down to 398,309 data points. Now my app will simply track new mortgages and corrected mortgages.

Instructions:

-Run python script to remove data with missing fields. Return cleaned dataset.

-From local terminal run the following command:

scp /Users/btwhi/documents/big\_data/final\_project/batch/Filtered\_CC\_Mortgage\_2013\_through\_March\_27\_2015.csv emr:/home/hadoop/bwhitehouse/mortgage

-Now your .csv file is in Hadoop

-To load .csv onto master node in order for hive to work we have to load data onto the worker nodes with a put statement, while logged on hadoop:

hdfs dfs -put Filtered\_CC\_Mortgage\_2013\_through\_March\_27\_2015.csv /bwhitehouse/mortgage/

-To check to make sure your file is uploaded correctly

find $PWD -type f | grep "Filtered\_CC\_Mortgage\_2013\_through\_March\_27\_2015.csv"

hdfs dfs -ls /bwhitehouse/mortgage/

//now need to creat the correct corresponding tables

/log into hive do the following

create external table CookCountymortgage\_csv(

Pin string,

DocNumber BIGINT,

DocType string,

DateRecord string,

DateExecute string,

Amount double,

Street string,

City string,

State string,

ZipCode BIGINT,

Location string)

row format serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde'

WITH SERDEPROPERTIES (

"separatorChar" = "\,",

"quoteChar" = "\""

)

STORED AS TEXTFILE

LOCATION "/bwhitehouse/mortgage/"

TBLPROPERTIES ("skip.header.line.count"="1");

-- Run a test query to make sure the above worked correctly

select \* from cookcountymortgage\_csv limit 10;

-- Create an ORC table for forclosure data (Note "stored as ORC" at the end)

create table CookCountymortgage(

Pin string,

DocNumber BIGINT,

DocType string,

DateRecord string,

DateExecute string,

Amount double,

Street string,

City string,

State string,

ZipCode BIGINT,

Location string)

stored as orc;

-- Copy the CSV table to the ORC table

insert overwrite table cookcountymortgage select \* from cookcountymortgage\_csv

where amount is not null and location is not null;

-- Run a test query to make sure the above worked correctly

select \* from cookcountymortgage limit 10;

///create an external table that tracks number of homes and total amount spent by zip code

//refrence flights delays

//how to overwrite to sum the amounts for sum amount

//what table to import from orc or hive??

//map reduce the information in order to put into hbase

//create another blank orc table

create table summortgage(

Zipcode BIGINT,

sum\_mortgages double,

num\_homes BIGINT)

stored as orc;

//now insert while map and reducing the values

//will create 3 columns zipcode, sum\_mortgages, num\_homes

//this will allow for us to determine the average home cost per a zip code

insert into table summortgage

select zipcode, sum(amount),count(1)

from cookcountymortgage where NOT (zipcode IS NULL)

group by zipcode;

-- Run a test query to make sure the above worked correctly

select \* from summortgage;

Batch for Medical Examiner

-In general this data set is missing a lot of values for the purpose of creating our ground truth most of the data is fine. Since we are focused on zip code and cause of death

I made the decision to only exclude data that had an entirely blanks line (if any existed), and all 'PENDING' causes since these reports have not been finalized yet.

I added 5 columns based on cause of death to make later HQLs streamlined for a sum and assist with data joins later.

This took our total data set from 60229 values to 58857.

-From local terminal run the following command:

scp /Users/btwhi/documents/big\_data/final\_project/batch/Filtered\_Medical\_Examiner.csv emr:/home/hadoop/bwhitehouse/medical

-Now your .csv file is in Hadoop

-To load .csv onto master node in order for hive to work we have to load data onto the worker nodes with a put statement, while logged on hadoop:

hdfs dfs -put Filtered\_Medical\_Examiner.csv /bwhitehouse/medical/

-create the directory on hdfs not just local

hdfs dfs -mkdir /bwhitehouse/medical/

-To check to make sure your file is uploaded correctly

find $PWD -type f | grep "Filtered\_Medical\_Examiner.csv"

hdfs dfs -ls /bwhitehouse/medical/

//now need to creat the correct corresponding tables

/log into hive do the following

create external table CookCountyMedical\_csv(

accident BIGINT,

homicide BIGINT,

natural BIGINT,

suicide BIGINT,

undetermined BIGINT,

casenum string,

dateIncident string,

dateDeath string,

age TINYINT,

gender string,

race string,

latino boolean,

mannerDeath string,

primaryCause string,

primaryLineA string,

primaryLineB string,

primaryLineC string,

secondaryCause string,

gunRelated boolean,

opiodRelated boolean,

coldRelated boolean,

heatRelated boolean,

district string,

incidentAddress string,

incidentCity string,

incidentZipCode BIGINT,

long string,

latt string,

loc string,

residentCity string,

residentZipCode BIGINT,

objectID BIGINT,

ward TINYINT,

community string)

row format serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde'

WITH SERDEPROPERTIES (

"separatorChar" = "\;",

"quoteChar" = "\""

)

STORED AS TEXTFILE

LOCATION "/bwhitehouse/medical/"

TBLPROPERTIES ("skip.header.line.count"="1");

-- Run a test query to make sure the above worked correctly

select \* from cookcountymedical\_csv limit 10;

-- Create an ORC table for forclosure data (Note "stored as ORC" at the end)

create table CookCountyMedical(

accident BIGINT,

homicide BIGINT,

natural BIGINT,

suicide BIGINT,

undetermined BIGINT,

casenum string,

dateIncident string,

dateDeath string,

age TINYINT,

gender string,

race string,

latino boolean,

mannerDeath string,

primaryCause string,

primaryLineA string,

primaryLineB string,

primaryLineC string,

secondaryCause string,

gunRelated boolean,

opiodRelated boolean,

coldRelated boolean,

heatRelated boolean,

district string,

incidentAddress string,

incidentCity string,

incidentZipCode BIGINT,

long string,

latt string,

loc string,

residentCity string,

residentZipCode BIGINT,

objectID BIGINT,

ward TINYINT,

community string)

stored as orc;

-- Copy the CSV table to the ORC table

insert overwrite table cookcountymedical select \* from cookcountymedical\_csv

where incidentZipCode is not null and mannerDeath is not null;

-- Run a test query to make sure the above worked correctly

select \* from cookcountymedical limit 10;

//map reduce the information in order to put into hbase

//create another blank orc table

create table summedical(

Zipcode BIGINT,

sum\_accident BIGINT,

sum\_homicide BIGINT,

sum\_natural BIGINT,

sum\_suicide BIGINT,

sum\_undetermined BIGINT,

sum\_deaths BIGINT)

stored as orc;

//now insert while map and reducing the values

//will create 7 columns zipcode, sum\_accident, sum\_homicide, sum\_natural, sum\_suicide, sum\_undetermined, sum\_deaths

//this will give us the number of people who have died and cause by zip code. Even though I selected the incident zipcode a lot of zipcodes

//do not fall into cook county this table has 513 rows. A lot of these might be the result of human error, however more than likely the person filling out the report

//filled in residentZipCode into incidentZipCode otherwise I am not sure how a person who died in NYC is being examined in Chicago?

insert into table summedical

select incidentZipCode, sum(accident), sum(homicide), sum(natural), sum(suicide), sum(undetermined), count(1)

from cookcountymedical where NOT (incidentZipCode IS NULL or incidentZipCode <= 9999 or incidentZipCode >=100000)

group by incidentZipCode;

//test your new table in hive

select \* from summedical;

//create table that will save the join to

create table cookcombined(

Zipcode BIGINT, sum\_mortgages double,

num\_homes BIGINT, Zipcode2 BIGINT,

sum\_accident BIGINT, sum\_homicide BIGINT,

sum\_natural BIGINT, sum\_suicide BIGINT,

sum\_undetermined BIGINT, sum\_deaths BIGINT)

stored as orc;

//using an inner join I have reduced the total number of rows for summmortage = 200 and summedical =500+ to 185

//joining two tables with deaths and mortgage costs

insert into table cookcombined

SELECT \*

FROM summortgage

INNER JOIN summedical

ON summortgage.zipcode=summedical.zipcode;

//test the combined table

select \* from cookcombined;

Serving

-Make table in HBase

create "CCmortgage\_v3", "zipcode"

-In hive push table from hive to HBASE with the following two commands:

create external table CCmortgage\_v3 (

zipcode bigint,

sum\_mortgages double,

num\_homes BIGINT, sum\_accident BIGINT,

sum\_homicide BIGINT, sum\_natural BIGINT,

sum\_suicide BIGINT, sum\_undetermined BIGINT,

sum\_deaths BIGINT)

STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'

WITH SERDEPROPERTIES ('hbase.columns.mapping' = ':key,zipcode:sum\_mortgages,zipcode:num\_homes,zipcode:sum\_accident,zipcode:sum\_homicide,

zipcode:sum\_natural,zipcode:sum\_suicide,zipcode:sum\_undetermined,zipcode:sum\_deaths')

TBLPROPERTIES ('hbase.table.name' = 'CCmortgage\_v3');

-The next command:

insert overwrite table CCmortgage\_v3

select zipcode,

sum\_mortgages, num\_homes,

sum\_accident, sum\_homicide,

sum\_natural, sum\_suicide,

sum\_undetermined, sum\_deaths

from cookcombined;

-test table in hive:

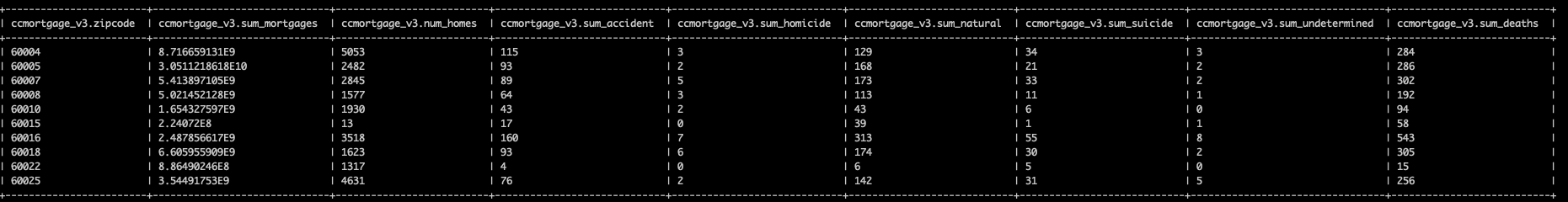
select \* from CCmortgage\_v3 limit 10;

-test table in hbase:

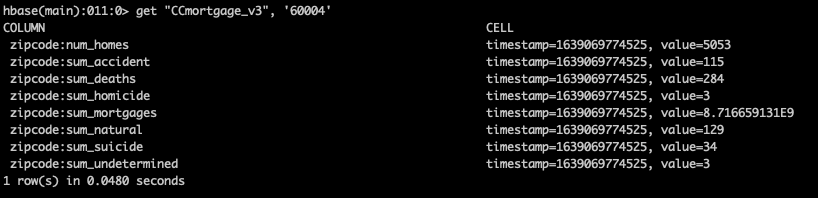
get "CCmortgage\_v3", '60004'

Pictures:

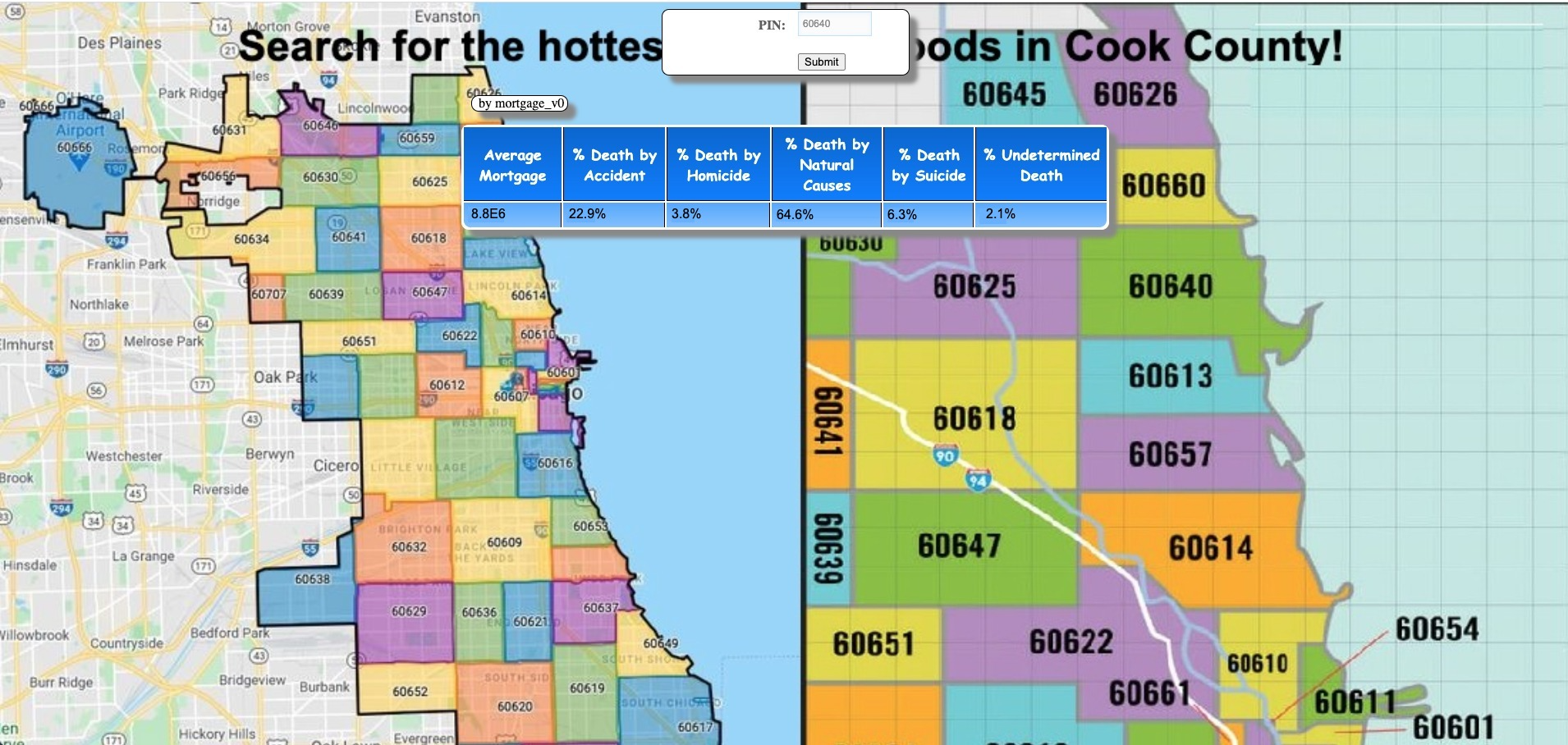
Hive CCmortgage\_v3



HBASE CCmortgage\_v3



Serving layer website results



Speed layer, wrote out still having issues with receiving the KAFKA information and submitting.

