Healthcare Analytic with Hadoop Ecosystem

Evan Tang
Northwestern University
tagkk@gmail.com

Hadoop, an open source software framework for distributed storage and distributed processing of very large data sets on computer clusters built from commodity hardware, is the new enterprise data operating system. The key advantage of using Hadoop is its fault tolerance; data is replicated to other nodes in the cluster, which means in the event of failure there is another copy available for use. It is also a more cost effective storage solutions equipped with future scale-out ability in compare to RDBMS which is extremely costly to expand beyond the initial setup. Hadoop is designed to run on a cluster of machines, additional storage and compute power can easily carry out by appending more nodes to the Hadoop cluster. This project is to demonstrate the value of Big Data versus traditional technologies.

The data selected for this project come from the Hospital Compare web site (http://hospitalcompare.hhs.gov) run by the U.S. Department of Health and Human Services. The purpose of the website is to provide data and information about the quality of care across 4,000 Medicare-certified hospitals which essentially cover all major hospitals in the United States. This dataset serves different purposes for various stakeholders, the government uses it to determine whether hospitals should be fined for not providing high quality care to patients; individuals can leverages this dataset to select the best hospital for a particular illness and healthcare search engines can use this dataset to build a recommendation engine. The Hospital Compare web site contains a lot of data, for this project we only look at a small subset from the outcome-of-care-measures.csv. It contains information about 30-day mortality and readmission rates for heart attacks, heart failure, and pneumonia for over 4,000 hospitals.

Hive is a system for managing and querying structured data built on top of Hadoop, the combo of Hive and Hadoop are more efficient than traditional technologies because:

a) SQL as a familiar data warehousing tool
b) Extensibility – easy to plug in custom mapper/reducer code; JDBC/ODBC drivers allow 3rd party applications to pull Hive data for reporting/browsing
c) Scalability and performance – load data in near real-time
d) Data growth – scale out rather than scale up by adding more boxes to the cluster

e) Schema flexibility – schemas are stored in RDBMS so users don’t have to specify them at execution time; column types could be complex types such as map, struct

Curious individuals can create tables in Hive and perform quick ad hoc analyses such as the ones below.

1. Find the best hospital for heart attack in every state (Figure 1)
   Best hospital is defined as having the lowest 30-day mortality for the specified outcome. Hospitals that do not have data on a particular outcome are excluded from the set of hospitals when deciding the rankings.

   select out1.hospitalname, out1.state, 
   out1.hospital30daydeathmortalityratesfromheartattack from outcome out1 join 
   (select state, MIN (hospital30daydeathmortalityratesfromheartattack) as minMR 
    from outcome where 
    hospital30daydeathmortalityratesfromheartattack is not NULL 
    group by state) out2 
   on (out1.state = out2.state and 
    out1.hospital30daydeathmortalityratesfromheartattack = out2.minMR);

2. Find the worst hospital for heart failure in every state (Figure 2)
   Worst hospital is defined as having the highest 30-day mortality for the specified outcome. Hospitals that do not have data on a particular outcome are excluded from the set of hospitals when deciding the rankings.

   select out1.hospitalname, out1.state, 
   out1.hospital30daydeathmortalityratesfromheartfailure from outcome out1 join 
   (select state, MAX(hospital30daydeathmortalityratesfromheartfailure) as maxMR 
    from outcome 
    where 
    hospital30daydeathmortalityratesfromheartfailure is not NULL 
    group by state) out2 
   on (out1.state = out2.state and 
    out1.hospital30daydeathmortalityratesfromheartfailure = out2.maxMR);

3. Find the top 10 busiest hospital in the United States (Figure 3)
   Busiest hospital is defined as having the highest total number of patients of Medicare patients treated for heart attacks, heart failure, and pneumonia. Hospitals that do not have data on a particular outcome are excluded from the set of hospitals when deciding the rankings.

   select hospitalname, state, 
   (COALESCE(numberofpatientshospital30daydeathmortalityratesfromheartattack, 
   CAST(0 AS BIGINT)) + 
   COALESCE(numberofpatientshospital30daydeathmortalityratesfromheartfailure, 
   CAST(0 AS BIGINT)) + 
   COALESCE(numberofpatientshospital30daydeathmortalityratesfrompneumonia, 
   CAST(0 AS BIGINT))) total_patients 
   from outcome 
   order by total_patients desc 
   limit 10;
Figure 1: HiveSQL for best hospital for heart attack in every state and result

Figure 2: HiveSQL for worst hospital for heart failure in every state and result
Figure 3: HiveSQL for top 10 busiest hospital in the United States and result

Figure 4: Metastore Manager to create new table from a file in Hive
Hue is an open source web interface that aggregates the most common Apache Hadoop components into a single point: browse files in HDFS, work with Hive, Pig and Impala queries, MapReduce jobs, and Oozie workflows etc. It aims to provide exceptional “just use” Hadoop user experience without worrying its underlying complexity or the trouble of using a command line. Through Hue individuals have the ability to put outcome-of-care-measures.csv into HDFS, create new table using the file on HDFS through its web interface, this is a big plus for individuals who are none technical.

Pig is a high-level platform for creating MapReduce programs used in Hadoop, it also enables executing data flows in parallel using Pig Latin. Pig Latin includes many of the traditional data operators (join, sort, filter, etc.), as well as the ability to develop customized functions for reading, processing, and writing data. Pig is more suitable for constructing dataflow which typically involve schedule job to periodically crunch the massive data from HDFS and to transfer the summarized data into a relational database for reporting, dashboarding, and ad-hoc analyses. For government analysts who need to perform more complex analytics, they can write time consuming queries as Pig Latin scripts, run them in the background and check the results upon job completion. Some of the Pig use cases for the outcome-of-care-measures dataset include:

1. Find the number of hospital in each state sorted by the highest number of hospital (Figure 5)

```pig
state_group = GROUP A BY state;

hospital_by_state = FOREACH state_group
                    GENERATE group, COUNT(A) as cnt;

sorted = ORDER hospital_by_state BY cnt DESC;
STORE sorted into './hospital_by_state';
```

2. Find the 5 best hospitals for heart attack in United States (Figure 6)

```pig
valid = FILTER A BY
        hospital30daydeathmortalityratesfromheartattack > 0;

data = foreach valid generate hospitalname, state,
      hospital30daydeathmortalityratesfromheartattack;

sortHA = ORDER data BY
         hospital30daydeathmortalityratesfromheartattack;

best5HA = LIMIT sortHA 5;
STORE best5HA into './best5HAHospital';
DUMP best5HA;
```

3. Find the 5 worst hospitals for heart failure in United States (Figure 7)

```pig
valid = FILTER A BY
        hospital30daydeathmortalityratesfromheartattack > 0;

data = foreach valid generate hospitalname, state,
      hospital30daydeathmortalityratesfromheartattack;

sortHF = ORDER data BY
         hospital30daydeathmortalityratesfromheartattack DESC;

worst5HF = LIMIT sortHF 5;
STORE worst5HF into './worst5HFHospital';
DUMP worst5HF;
```

![Figure 5: Pig result for the number of hospital in each state sorted by the highest number of hospital](image-url)
Figure 6: Pig result for the 5 best hospitals for heart attack in United States

Figure 7: Pig result for the 5 worst hospitals for heart failure in United States

Pig is more suitable for experienced analysts because there is no easy-to-use import data web interface. Data loading have to be done through script and prior knowledge of the data type is needed. The Pig Latin script to import the outcome-of-care-measures dataset is below:

```
A = LOAD './outcome-of-care-measures.txt' USING PigStorage('utf8') AS (  
    providernumber: CHARARRAY,  
    hospitalname: CHARARRAY,  
    address1: CHARARRAY,  
    address2: CHARARRAY,  
    address3: CHARARRAY,  
    city: CHARARRAY,  
    state: CHARARRAY,  
    zipcode: CHARARRAY,  
    countyname: CHARARRAY,  
    phonenumber: CHARARRAY,  
    hospital30daydeathmortalityratesfromheartattack: FLOAT,  
    comparisontousratehospital30daydeathmortalityratesfromheartattack: CHARARRAY,  
    lowermortalityestimatehospital30daydeathmortalityratefromheartattack: FLOAT,  
    uppermortalityestimatehospital30daydeathmortalityratefromheartattack: FLOAT,  
    numberofpatientshospital30daydeathmortalityratesfromheartattack: INT,  
    footnotehospital30daydeathmortalityratesfromheartattack: CHARARRAY,  
    hospital30daydeathmortalityratesfromheartfailure: FLOAT,  
    comparisontousratehospital30daydeathmortalityratesfromheartfailure: CHARARRAY,  
    lowermortalityestimatehospital30daydeathmortalityratefromheartfailure: FLOAT,  
    uppermortalityestimatehospital30daydeathmortalityratefromheartfailure: FLOAT,  
    numberofpatientshospital30daydeathmortalityratesfromheartfailure: INT,  
    footnotehospital30daydeathmortalityratesfromheartfailure: CHARARRAY,  
    hospital30daydeathmortalityratesfrompneumonia: FLOAT,  
    comparisontousratehospital30daydeathmortalityratesfrompneumonia: CHARARRAY,  
    lowermortalityestimatehospital30daydeathmortalityratefrompneumonia: FLOAT,  
    uppermortalityestimatehospital30daydeathmortalityratefrompneumonia: FLOAT,  
    numberofpatientshospital30daydeathmortalityratesfrompneumonia: INT,  
    footnotehospital30daydeathmortalityratesfrompneumonia: CHARARRAY,  
    hospital30dayreadmissionratesfromheartattack: CHARARRAY,  
    comparisontousratehospital30dayreadmissionratesfromheartattack: CHARARRAY,  
    lowerreadmissionestimatehospital30dayreadmissionratesfromheartattack: CHARARRAY,  
    upperreadmissionestimatehospital30dayreadmissionratesfromheartattack: CHARARRAY,  
    numberofpatientshospital30dayreadmissionratesfromheartattack: INT,  
    footnotehospital30dayreadmissionratesfromheartattack: CHARARRAY,  
    hospital30dayreadmissionratesfromheartfailure: FLOAT,  
    comparisontousratehospital30dayreadmissionratesfromheartfailure: CHARARRAY,  
    lowerreadmissionestimatehospital30dayreadmissionratesfromheartfailure: FLOAT,  
    upperreadmissionestimatehospital30dayreadmissionratesfromheartfailure: FLOAT,  
    numberofpatientshospital30dayreadmissionratesfromheartfailure: INT,  
    footnotehospital30dayreadmissionratesfromheartfailure: CHARARRAY,  
    hospital30dayreadmissionratesfrompneumonia: FLOAT,  
    comparisontousratehospital30dayreadmissionratesfrompneumonia: CHARARRAY,  
    lowerreadmissionestimatehospital30dayreadmissionratesfrompneumonia: CHARARRAY,  
    upperreadmissionestimatehospital30dayreadmissionratesfrompneumonia: CHARARRAY,  
    numberofpatientshospital30dayreadmissionratesfrompneumonia: INT,  
    footnotehospital30dayreadmissionratesfrompneumonia: CHARARRAY);  
```

Both Hive and Pig provide limited capabilities in data manipulation, statistical computing and machine learning. For data scientists who need to explore and examine data from multiple disparate sources, apply modeling, statistics analytic to discover hidden insights R is a powerful tool. R and its libraries implement a wide variety of statistical and graphical techniques including linear and nonlinear modeling, classical statistical tests, time-series analysis, classification, clustering, etc. R uses a command line interface; Rstudio is a popular open source graphical user interface for R. Using RStudio with Apache Hadoop enables organizations to take advantage of the scale-out storage and processing capabilities of Hadoop along with the powerful analytic and visualization capabilities of R. Following are three use cases for the outcome-of-care-measures dataset operating in R on Hadoop:

1. Plot the distribution of hospital 30 days pneumonia mortality rate (Figure 8)

2. Plot the distribution of number of Medicare patients treated for Heart Attack by the hospital (Figure 9)

3. To prepare data for our next task – using Mahout to build a recommendation engine, we wrote a simple R script to convert the outcome-of-care-measures dataset into Mahout’s recommender expected input format: hospitalID, diseaseID, value. Here hospitalID and outcomeID refer to a particular hospital and a particular outcome (heart attack = 1, heart failure = 2, pneumonia = 3), and value donates the rating given to a hospital (Better than U.S. National Average = 1, No Different than U.S. National Average = 0, Worse than U.S. National Average = -1, Number of Cases too Small = no rating). The resulted matrix is output to file HospitalRatings.txt.

Using the HospitalRatings.txt file created in R from the previous step, data scientist can create a user-based recommender in merely a few minutes in Apache Mahout. Mahout is a machine learning application programming interface built on Hadoop, it contains machine learning algorithms in the following area: clustering, pattern mining, classification, regression, recommenders/collaborative filtering. Recommendation involves the prediction of what new items a user would like or dislike based on preferences of or associations to previous items. The way it works for the outcome-of-care-measures dataset is going to be different, since outcome is our item and there are only three outcomes (heart attack, heart failure and pneumonia) our recommendation engine will give the predicted comparison to U.S. Rate value for specified outcome for certain hospitals. Below is the code to create the hospital-disease recommender in Mahout:

```
wget http://photo.etangkk.com/HospitalRatings.txt
hadoop fs -put HospitalRatings.txt HRatings.txt
mahout recommenditembased --input HRatings.txt --output Hrecommendation --numRecommendations 1 --outputPathForSimilarityMatrix similarity-matrix --similarityClassname SIMILARITY_COSINE --tempDir FinalProject
hadoop fs -ls Hrecommendation
hadoop fs -cat Hrecommendation/part-r-00000 | head
hadoop fs -rm -r FinalProject
hadoop fs -rm HRatings.txt
```

![Figure 8: Distribution of hospital 30 days pneumonia mortality rate](image.png)
Figure 9: Distribution of number of Medicare patients treated for heart attack by the hospital

Figure 10: R code to generate Manhout’s recommender expected input data

Using the HospitalRatings.txt file created in R from the previous step, data scientist can create a user-based recommender in merely a few minutes in Apache Mahout. Mahout is a machine learning application programming interface built on Hadoop, it contains machine learning algorithms in the following area: clustering, pattern mining, classification, regression, recommenders/collaborative filtering. Recommendation involves the prediction of what new items a user would like or dislike based on preferences of or associations to previous items. The way it works for the outcome-of-care-measures dataset is going to be different, since outcome is our item and there are only three outcomes (heart attack, heart failure and pneumonia) our recommendation engine will give the predicted comparison to U.S. Rate value for specified outcome for certain hospitals. Below is the code to create the hospital-disease recommender in Mahout:

```
wget http://photo.etangkk.com/HospitalRatings.txt
hadoop fs -put HospitalRatings.txt
HRatings.txt
mahout recommenditembased --input
HRatings.txt --output Hrecommendation -
numRecommendations 1 --
outputPathForSimilarityMatrix similarity-
matrix --similarityClassname
SIMILARITY_COSINE --tempDir
FinalProject
hadoop fs -ls Hrecommendation
hadoop fs -cat Hrecommendation/part-r-
00000 | head
hadoop fs -rm -r FinalProject
hadoop fs -rm HRatings.txt
```

Figure 11: Mahout’s recommender result
<table>
<thead>
<tr>
<th>Hospital ID</th>
<th>(DiseaseID : Recommendation Strength) Tuples</th>
</tr>
</thead>
<tbody>
<tr>
<td>153</td>
<td>1:1.0</td>
</tr>
<tr>
<td>281</td>
<td>1:1.0</td>
</tr>
<tr>
<td>330</td>
<td>1:1.0</td>
</tr>
<tr>
<td>361</td>
<td>2:1.0</td>
</tr>
<tr>
<td>406</td>
<td>2:1.0</td>
</tr>
<tr>
<td>476</td>
<td>1:1.0</td>
</tr>
<tr>
<td>485</td>
<td>1:1.0</td>
</tr>
<tr>
<td>500</td>
<td>1:1.0</td>
</tr>
<tr>
<td>502</td>
<td>1:1.0</td>
</tr>
<tr>
<td>550</td>
<td>1:1.0</td>
</tr>
<tr>
<td>2980</td>
<td>1:1.0</td>
</tr>
<tr>
<td>3155</td>
<td>1:1.0</td>
</tr>
<tr>
<td>3187</td>
<td>1:1.0</td>
</tr>
<tr>
<td>3252</td>
<td>1:1.0</td>
</tr>
<tr>
<td>3388</td>
<td>1:1.0</td>
</tr>
<tr>
<td>3447</td>
<td>1:1.0</td>
</tr>
<tr>
<td>3545</td>
<td>3:1.0</td>
</tr>
<tr>
<td>3922</td>
<td>1:1.0</td>
</tr>
<tr>
<td>4085</td>
<td>1:1.0</td>
</tr>
<tr>
<td>4349</td>
<td>3:1.0</td>
</tr>
</tbody>
</table>

We configured the recommendation engine to give one disease recommendations to each hospital. From the result we can see all disease recommendation to each hospital are at 100% (Better than U.S. National Average) strength.

Our project demonstrated that Hadoop emerges new technologies like Hadoop Distributed File System (HDFS), MapReduce, Pig, Hive, Hue, Mahout and other software packages that can be installed on the base modules. Leveraging different analytics tools on the market (R in our project), we turned Hadoop into a powerhouse for analytics for a wide variety of stakeholders.

**Reference**


