BIG DATA PROJECT REPORT – YELP DATASET

Link to dataset : http://www.yelp.com/dataset\_challenge

AWS CLUSTER-EMR

Created EMR cluster, with 3 nodes , 1 master and 2 slaves. Accessed

The nodes via PUTTY. Implemented Map reduce, hive, pig and hbase on the cloud.

JAVA HADOOP MAP-REDUCE:

1. Year wise reviews count:

package test;

import java.io.File;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.conf.Configured;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

import org.json.JSONException;

import org.json.JSONObject;

/\*\*

\*

\* @author Nirav

\*/

public class ReviewDateCount extends Configured implements Tool {

public static class IPcountMapper

//extends Mapper<LongWritable, Text, Text, IntWritable> {

extends Mapper<LongWritable, Text, IntWritable, IntWritable> {

private static IntWritable one = new IntWritable(1);

private Text wordText = new Text();

public void map(LongWritable key, Text line, Context context) throws IOException, InterruptedException {

//String []words=line.toString().split("\\s");//this will split the words by spaces and tabs[^\w] instead

//for (String word : words) {

try {

// wordText.set(words[0]);

// context.write(wordText, one);

JSONObject jsonObj = new JSONObject(line.toString());

//parse the input data with JSONObject

String businessID = (String)jsonObj.get("business\_id");

//Date value fetch

String date = (String)jsonObj.get("date");

int index = date.indexOf('-');

String yearValue = date.substring(0, index);

int year = Integer.parseInt(yearValue);

IntWritable yearWR = new IntWritable();

yearWR.set(year);

yearWR.set(yearWR.get());

//Business count

StringBuilder key2 = new StringBuilder();

key2.append(businessID);

//emitting directory structure as key2 and input record as value.

context.write(yearWR,one);

} catch (JSONException ex) {

System.out.println("JSON ERROR!!!"+ex);

}

}

}

public static class IPcountReducer extends Reducer<IntWritable, IntWritable, IntWritable, IntWritable> {

private IntWritable result = new IntWritable();

public void reduce(IntWritable key, Iterable<IntWritable> values, Context context)

throws IOException, InterruptedException {

int total = 0;

for (IntWritable value : values) {

total += value.get();

}

result.set(total);

context.write(key, result);

}

}

public int run(String[] args) throws Exception {

Configuration conf = getConf();

Job job = new Job(conf, "usefulVoteCount");

job.setJarByClass(ReviewDateCount.class);

final File f = new File(ReviewDateCount.class.getProtectionDomain().getCodeSource().getLocation().getPath());

String inFiles = f.getAbsolutePath().replace("/build/classes", "") + "/src/inFiles/review.json";

String outFiles = f.getAbsolutePath().replace("/build/classes", "") + "/src/outFiles/reviewDatesCount";

System.out.println(inFiles);

System.out.println(outFiles);

Path in = new Path(inFiles);

Path out = new Path(outFiles);

FileInputFormat.setInputPaths(job, in);

FileOutputFormat.setOutputPath(job, out);

job.setMapperClass(IPcountMapper.class);

job.setCombinerClass(IPcountReducer.class);

job.setReducerClass(IPcountReducer.class);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

job.setOutputKeyClass(IntWritable.class);

job.setOutputValueClass(IntWritable.class);

System.exit(job.waitForCompletion(true) ? 0 : 1);

return 0;

}

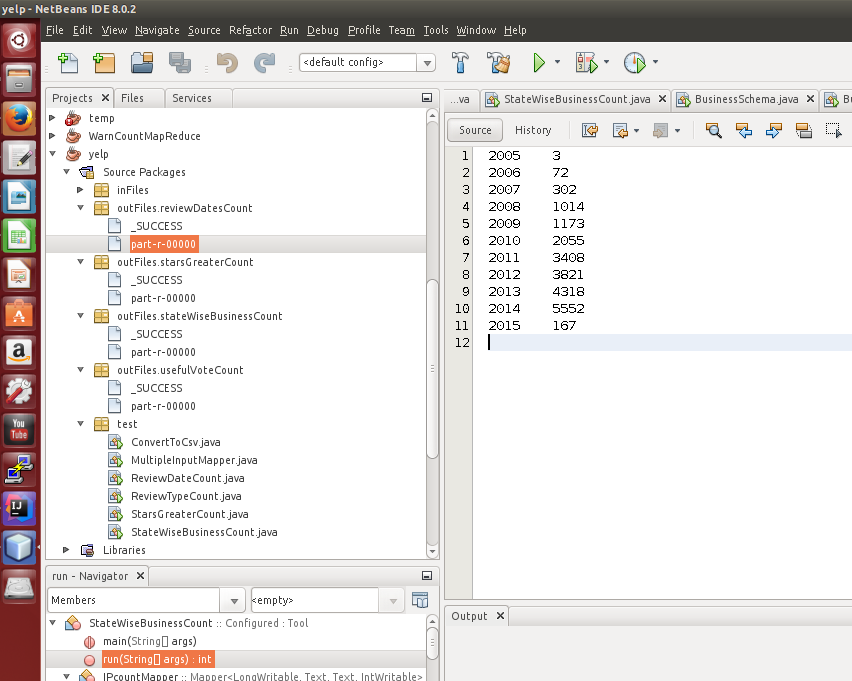
public static void main(String[] args) throws Exception {

int res = ToolRunner.run(new Configuration(), new ReviewDateCount(), args);

System.exit(res);

}

}



2. Review type count (how many useful votes a business has got )

package test;

import java.io.File;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.conf.Configured;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

import org.json.JSONException;

import org.json.JSONObject;

/\*\*

\*

\* @author Nirav

\*/

public class ReviewTypeCount extends Configured implements Tool {

public static class IPcountMapper

extends Mapper<LongWritable, Text, Text, IntWritable> {

private static IntWritable one = new IntWritable(1);

private Text wordText = new Text();

public void map(LongWritable key, Text line, Context context) throws IOException, InterruptedException {

try {

JSONObject jsonObj = new JSONObject(line.toString());

//parse the input data with JSONObject

String businessID = (String)jsonObj.get("business\_id");

JSONObject votesArray= (JSONObject) jsonObj.get("votes");

//int funnyVoteObj = (int) votesArray.get("funny");

int usefulVoteObj = (int) votesArray.get("useful");

//int coolVoteObj = (int) votesArray.get("cool");

IntWritable voteWR = new IntWritable();

voteWR.set(usefulVoteObj);

voteWR.set(voteWR.get());

//Business emit

StringBuilder key2 = new StringBuilder();

key2.append(businessID);

//emitting directory structure as key2 and input record as value.

context.write(new Text(key2.toString()), voteWR);

} catch (JSONException ex) {

System.out.println("JSON ERROR!!!"+ex);

}

}

}

//public static class IPcountReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

public static class IPcountReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

private IntWritable voteReult = new IntWritable();

public void reduce(Text key, Iterable<IntWritable> values, Context context)

throws IOException, InterruptedException {

for (IntWritable value : values) {

if (value.get() > 0){

voteReult.set(value.get());

}

}

context.write(key, voteReult);

}

}

public int run(String[] args) throws Exception {

Configuration conf = getConf();

Job job = new Job(conf, "usefulVoteCount");

job.setJarByClass(ReviewTypeCount.class);

final File f = new File(ReviewTypeCount.class.getProtectionDomain().getCodeSource().getLocation().getPath());

String inFiles = f.getAbsolutePath().replace("/build/classes", "") + "/src/inFiles/review.json";

String outFiles = f.getAbsolutePath().replace("/build/classes", "") + "/src/outFiles/usefulVoteCount";

System.out.println(inFiles);

System.out.println(outFiles);

Path in = new Path(inFiles);

Path out = new Path(outFiles);

FileInputFormat.setInputPaths(job, in);

FileOutputFormat.setOutputPath(job, out);

job.setMapperClass(IPcountMapper.class);

job.setCombinerClass(IPcountReducer.class);

job.setReducerClass(IPcountReducer.class);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

job.setOutputKeyClass(Text.class);

//job.setOutputValueClass(IntWritable.class);

job.setOutputValueClass(IntWritable.class);

System.exit(job.waitForCompletion(true) ? 0 : 1);

return 0;

}

public static void main(String[] args) throws Exception {

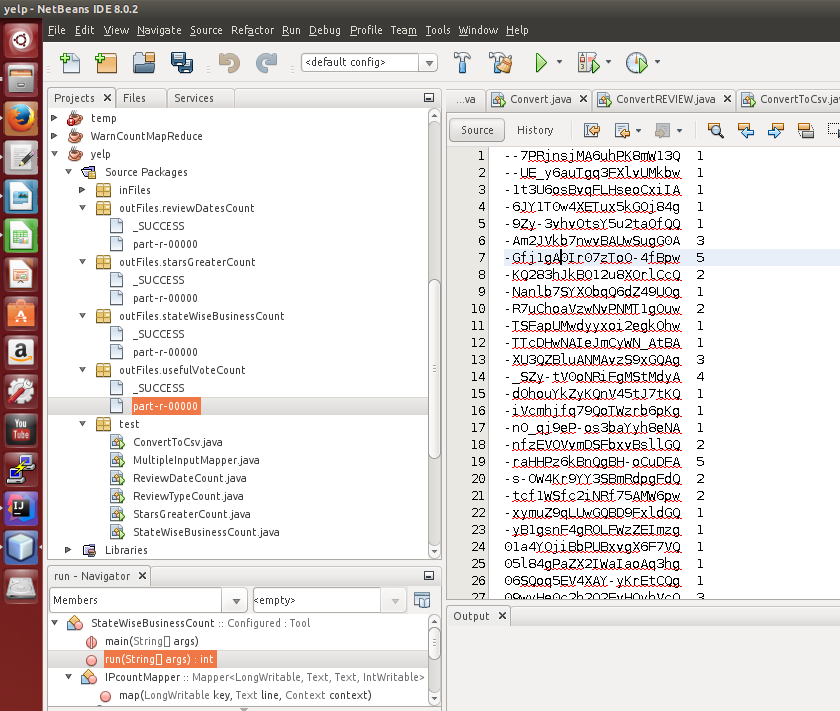
int res = ToolRunner.run(new Configuration(), new ReviewTypeCount(), args);

System.exit(res);

}

}

Output of 2:



3. business with Stars greater than a value (example : > 3.5)

package test;

import java.io.File;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.conf.Configured;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.DoubleWritable;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

import org.json.JSONException;

import org.json.JSONObject;

/\*\*

\*

\* @author Nirav

\*/

public class StarsGreaterCount extends Configured implements Tool {

public static class IPcountMapper

extends Mapper<LongWritable, Text, Text, DoubleWritable> {

private static IntWritable one = new IntWritable(1);

private Text wordText = new Text();

public void map(LongWritable key, Text line, Context context) throws IOException, InterruptedException {

try {

JSONObject jsonObj = new JSONObject(line.toString());

//parse the input data with JSONObject

String name = (String)jsonObj.get("name");

String state = (String)jsonObj.get("state");

String businessID = (String)jsonObj.get("business\_id");

String fullAddress = (String)jsonObj.get("full\_address");

double stars = (double)jsonObj.get("stars");

DoubleWritable starsWR = new DoubleWritable();

starsWR.set(stars);

//Business count

StringBuilder key2 = new StringBuilder();

//key2.append(businessID);

key2.append(name);

//Stars count

context.write(new Text(key2.toString()), starsWR);

} catch (JSONException ex) {

System.out.println("JSON ERROR!!!"+ex);

}

//}

}

}

public static class IPcountReducer extends Reducer<Text, DoubleWritable, Text, DoubleWritable> {

private IntWritable result = new IntWritable();

//stars variables

private DoubleWritable stars = new DoubleWritable();

private DoubleWritable starReult = new DoubleWritable();

public void reduce(Text key, Iterable<DoubleWritable> values, Context context)

throws IOException, InterruptedException {

for (DoubleWritable value : values) {

if (value.get() > 3.0){

starReult.set(value.get());

}

}

context.write(key, starReult);

}

}

public int run(String[] args) throws Exception {

Configuration conf = getConf();

Job job = new Job(conf, "IPcount");

job.setJarByClass(StarsGreaterCount.class);

final File f = new File(StarsGreaterCount.class.getProtectionDomain().getCodeSource().getLocation().getPath());

String inFiles = f.getAbsolutePath().replace("/build/classes", "") + "/src/inFiles/business.json";

String outFiles = f.getAbsolutePath().replace("/build/classes", "") + "/src/outFiles/starsGreaterCount2";

System.out.println(inFiles);

System.out.println(outFiles);

Path in = new Path(inFiles);

Path out = new Path(outFiles);

FileInputFormat.setInputPaths(job, in);

FileOutputFormat.setOutputPath(job, out);

job.setMapperClass(IPcountMapper.class);

job.setCombinerClass(IPcountReducer.class);

job.setReducerClass(IPcountReducer.class);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

job.setOutputKeyClass(Text.class);

//job.setOutputValueClass(IntWritable.class);

job.setOutputValueClass(DoubleWritable.class);

System.exit(job.waitForCompletion(true) ? 0 : 1);

return 0;

}

public static void main(String[] args) throws Exception {

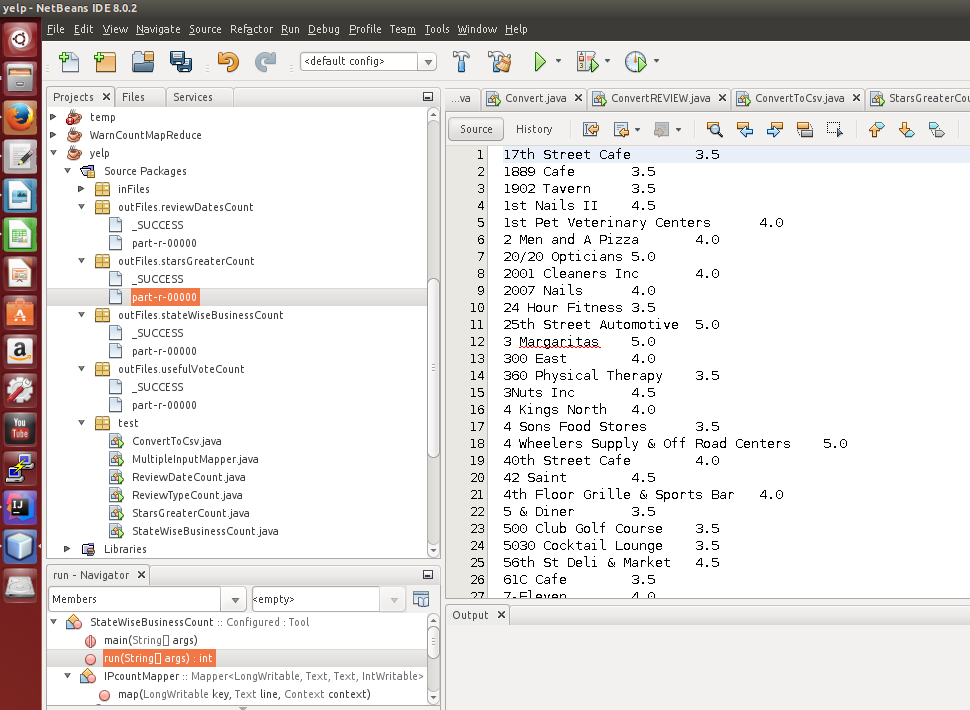
int res = ToolRunner.run(new Configuration(), new StarsGreaterCount(), args);

System.exit(res);

}

}

Output sample:



4. State wise count of business ( how many businesses are there in a state : for visualization)

package test;

import java.io.File;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.conf.Configured;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.DoubleWritable;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

import org.apache.hadoop.util.Tool;

import org.apache.hadoop.util.ToolRunner;

import org.json.JSONException;

import org.json.JSONObject;

/\*\*

\*

\* @author Nirav

\*/

public class StateWiseBusinessCount extends Configured implements Tool {

public static class IPcountMapper

extends Mapper<LongWritable, Text, Text, IntWritable> {

private static IntWritable one = new IntWritable(1);

private Text wordText = new Text();

public void map(LongWritable key, Text line, Context context) throws IOException, InterruptedException {

try {

JSONObject jsonObj = new JSONObject(line.toString());

//parse the input data with JSONObject

String name = (String)jsonObj.get("name");

String state = (String)jsonObj.get("state");

String businessID = (String)jsonObj.get("business\_id");

String fullAddress = (String)jsonObj.get("full\_address");

double stars = (double)jsonObj.get("stars");

DoubleWritable starsWR = new DoubleWritable();

starsWR.set(stars);

//State wise business count

StringBuilder key2 = new StringBuilder();

key2.append(state);

//emitting directory structure as key2 and input record as value.

context.write(new Text(key2.toString()), one);

} catch (JSONException ex) {

System.out.println("JSON ERROR!!!"+ex);

}

//}

}

}

public static class IPcountReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

private IntWritable result = new IntWritable();

public void reduce(Text key, Iterable<IntWritable> values, Context context)

throws IOException, InterruptedException {

int total = 0;

for (IntWritable value : values) {

total += value.get();

}

result.set(total);

context.write(key, result);

}

}

public int run(String[] args) throws Exception {

Configuration conf = getConf();

Job job = new Job(conf, "IPcount");

job.setJarByClass(StateWiseBusinessCount.class);

final File f = new File(StateWiseBusinessCount.class.getProtectionDomain().getCodeSource().getLocation().getPath());

String inFiles = f.getAbsolutePath().replace("/build/classes", "") + "/src/inFiles/business.json";

String outFiles = f.getAbsolutePath().replace("/build/classes", "") + "/src/outFiles/stateCount";

System.out.println(inFiles);

System.out.println(outFiles);

Path in = new Path(inFiles);

Path out = new Path(outFiles);

FileInputFormat.setInputPaths(job, in);

FileOutputFormat.setOutputPath(job, out);

job.setMapperClass(IPcountMapper.class);

job.setCombinerClass(IPcountReducer.class);

job.setReducerClass(IPcountReducer.class);

job.setInputFormatClass(TextInputFormat.class);

job.setOutputFormatClass(TextOutputFormat.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

System.exit(job.waitForCompletion(true) ? 0 : 1);

return 0;

}

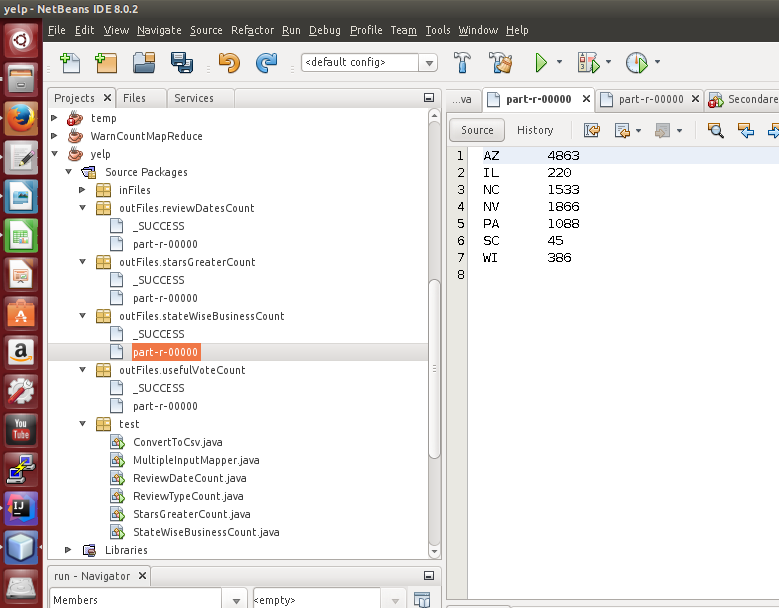
public static void main(String[] args) throws Exception {

int res = ToolRunner.run(new Configuration(), new StateWiseBusinessCount(), args);

System.exit(res);

}

}



PIG COMMANDS:

/////// Business and user relative count

A = Load 'hdfs://localhost:9000/user/yelp\_csv/business.csv' using PigStorage(',') as (business\_id:chararray, name:chararray, state:chararray, city:chararray, stars:double, latitude:long, longitude:long, review\_count:int);

B = Load 'hdfs://localhost:9000/user/yelp\_csv/reviews.csv' using PigStorage(',') as (business\_id:chararray, user\_id:chararray, stars:int, date:datetime, funnyVoteCount:int, usefulVoteCount:int, coolVoteCount:int);

A2 = FOREACH B GENERATE business\_id, user\_id;

B = Group A2 by business\_id;

C = FOREACH B Generate group as business\_id, COUNT(A2.user\_id) as business\_review\_count;

M2 = FOREACH A GENERATE business\_id, name, state;

joind = Join C by business\_id, M2 by business\_id;

nJoind = FOREACH joind GENERATE name, state, business\_review\_count;

O = order nJoind by business\_review\_count DESC;

////// count of reviews given by each user to a business /////////

B = Load 'hdfs://localhost:9000/user/yelp\_csv/reviews.csv' using PigStorage(',') as (business\_id:chararray, user\_id:chararray, stars:int, date:datetime, funnyVoteCount:int, usefulVoteCount:int, coolVoteCount:int);

A2 = FOREACH B GENERATE business\_id, user\_id;

B = Group A2 by business\_id;

C = FOREACH B Generate group as business\_id, COUNT(A2.user\_id) as business\_review\_count;

D = FOREACH C Generate business\_id, business\_review\_count;

///

AWS COMMANDS:

/////// SSH to aws

ssh hadoop@ec2-54-213-123-126.us-west-2.compute.amazonaws.com -i myawsbigdata.pem

////// copy from local to aws

scp -i myawsbigdata.pem /home/nirav/Desktop/yelp.jar hadoop@ec2-54-213-123-126.us-west-2.compute.amazonaws.com:~

/////// Copy from aws system to local

scp -i myawsbigdata.pem hadoop@ec2-54-213-123-126.us-west-2.compute.amazonaws.com:~/starsOutput/starsOutput/\* HdfsOutput/

scp -i myawsbigdata.pem hadoop@ec2-54-213-123-126.us-west-2.compute.amazonaws.com:~/hive\_out5 HdfsOutput/

/////////// hadoop java program run ( take input from S3) and (load output to S3)

hadoop jar yelpUsefulVoteCount.jar com.me.usefulVoteCount.ReviewTypeCount s3n://myawsbigdata/input/review.json s3n://myawsbigdata/usefulVoteCount/

HIVE COMMANDS:

Query:1

Julian Serrano,Las Vegas,-115.175766,36.10846,990

U.S. Egg,Phoenix,-112.06845,33.637367,99

Five Guys Burgers and Fries,Mesa,-111.75459,33.38057,99

Sauce Pizza & Wine,Scottsdale,-111.92627,33.61746,99

Yama Asian Fusion,Charlotte,-80.82398,35.157032,99

California Pizza Kitchen,Las Vegas,-115.15054,36.083168,99

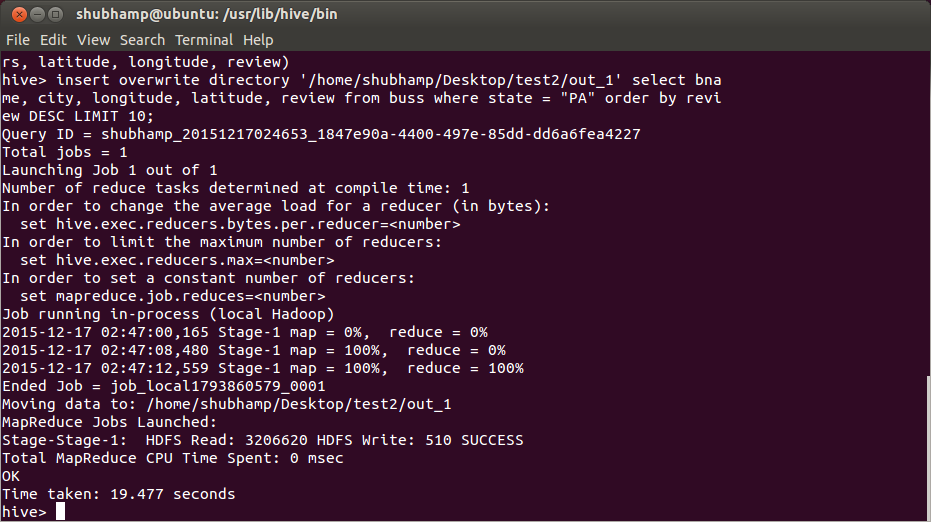
Nogales Hot Dogs,Phoenix,-112.03936,33.49437,99

Smeeks,Phoenix,-112.02721,33.51021,99

Fran's Filling Station,Charlotte,-80.851906,35.197716,99

Benjarong Authentic Thai Cuisine,Las Vegas,-115.27956,36.053383,99

insert overwrite directory 'hdfs:///hive\_out1/' select bname, city, longitute, latitude, total\_reviews from business where state = "PA" order by total\_reviews DESC LIMIT 10;



Query : 2

Earl of Sandwich,3984

Serendipity 3,3007

ARIA Hotel & Casino,2440

XS Nightclub,2302

Hash House A Go Go,2110

Planet Hollywood Las Vegas Resort & Casino,1714

InterContinental Alliance Resorts THE PALAZZO,1507

Vdara Hotel,1469

Studio B,1428

Cirque du Soleil - The Beatles LOVE,1397

SUSHISAMBA - Las Vegas,1264

Encore,1211

Hot N Juicy Crawfish,1142

The Buffet at ARIA,1125

Trump Hotel,1086

LAVO Italian Restaurant & Lounge,1069

Mr. Mama's,1065

Joe's Seafood Prime Steak & Stone Crab,1023

Julian Serrano,990

Charlotte Douglas International Airport,983

MIX,933

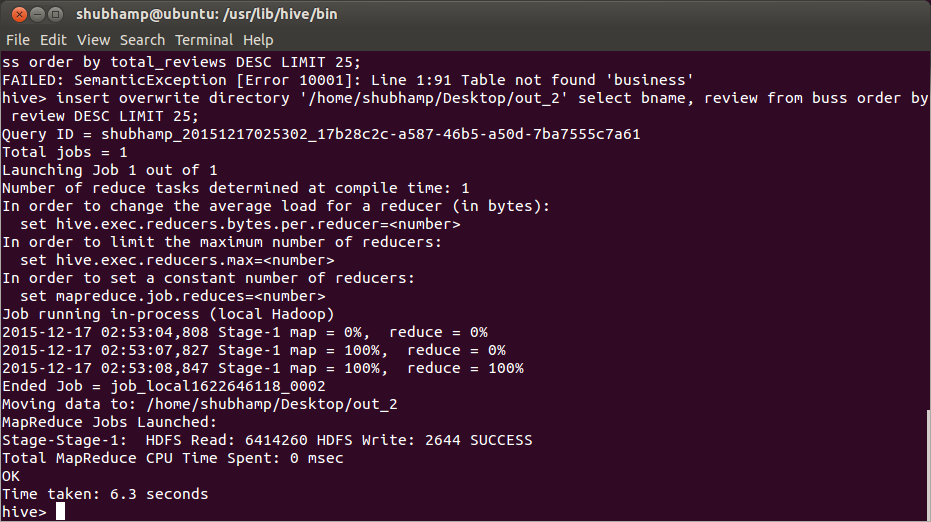
Gallo Blanco,924

The Mission,908

BLT Burger,908

Texas de Brazil,885

insert overwrite directory 'hdfs:///hive\_out2/' select bname, total\_reviews from business order by total\_reviews DESC LIMIT 25;



Query : 3

Earl of Sandwich,NV,3984

Serendipity 3,NV,3007

ARIA Hotel & Casino,NV,2440

XS Nightclub,NV,2302

Hash House A Go Go,NV,2110

Planet Hollywood Las Vegas Resort & Casino,NV,1714

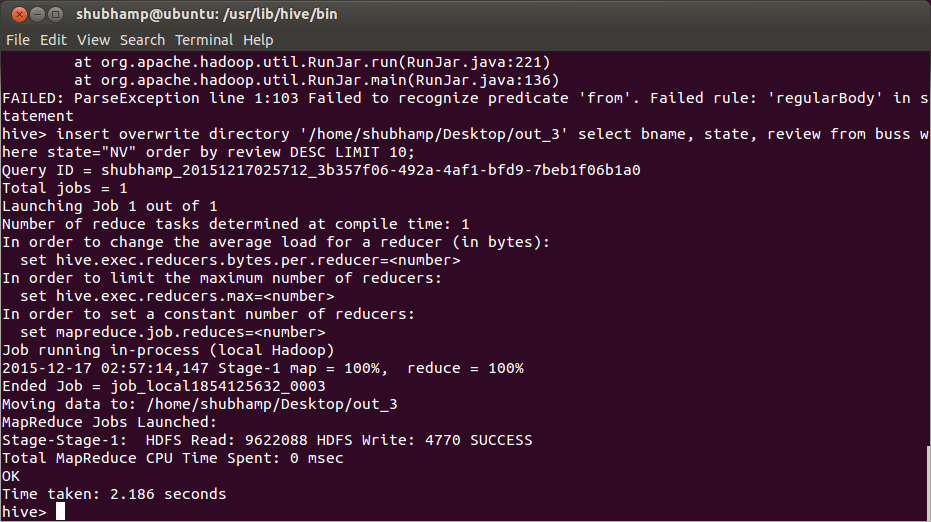
InterContinental Alliance Resorts THE PALAZZO,NV,1507

Vdara Hotel,NV,1469

Studio B,NV,1428

Cirque du Soleil - The Beatles LOVE,NV,1397

insert overwrite directory 'hdfs:///hive\_out3/' select bname, state, total\_reviews from business where state="NV" order by total\_reviews DESC LIMIT 10;



Query : 4

Ken,15144

Phuong,13630

Fancypants,9839

Gourmet,9485

Ann,8956

Keiko,8218

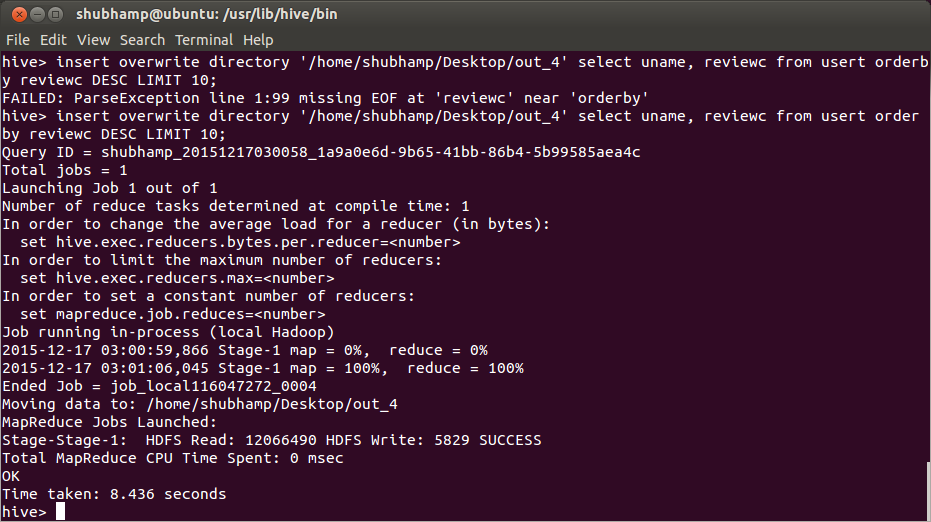
Jacob,6951

Kathleen,6947

Fenny,6876

Santi,6764

insert overwrite directory 'hdfs:///hive\_out7/' select uname, review\_count from usert orderby review\_count DESC LIMIT 10;



Query:5

megan,215,1,3,8

Caroline,168,5,2,2

Caroline,168,3,2,3

Caroline,168,3,4,2

Caroline,168,9,8,8

Jordan,137,3,1,0

Dee,111,2,1,0

Dee,111,2,1,0

Cassie,95,1,1,0

Cassie,95,1,0,0

Alexa,78,2,2,1

Lisa,74,1,0,0

Shanna,69,3,1,3

Shanna,69,1,2,0

Shanna,69,2,2,1

Chad,68,6,1,0

Chad,68,2,2,0

Chad,68,0,0,0

Chad,68,2,2,0

Chad,68,0,1,0

Chad,68,0,0,0

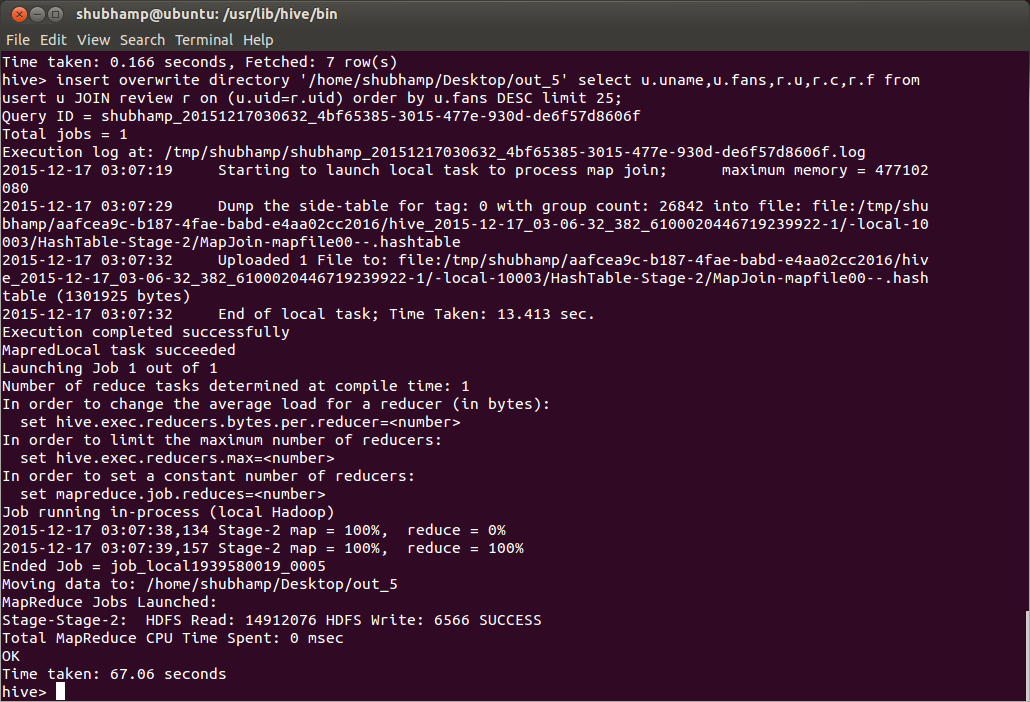
Chad,68,1,0,0

Chad,68,2,0,1

Chad,68,5,1,1

Chad,68,1,0,0

insert overwrite directory 'hdfs:///hive\_out17/' select u.uname,u.fans,r.uc,r.cv,r.fv from usert u JOIN reviews r on (u.uid=r.uid) order by u.fans DESC limit 25;



Query : 6

zriel,21,59,26

zoie,5,61,5

zanna,4,21,17

z,0,0,0

yummy,0,0,0

your,3,20,0

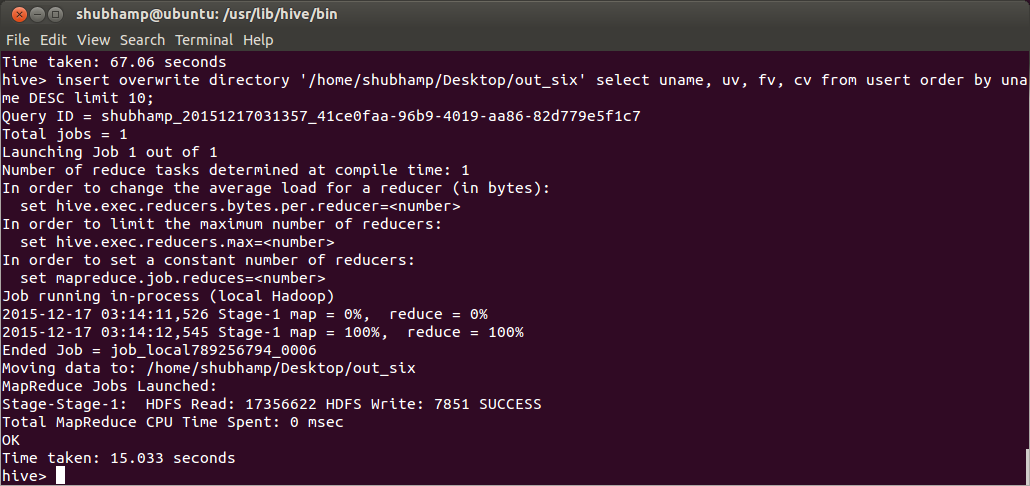
yoko,7,50,37

yeonhee,0,0,0

xtina,42,113,50

xoxx,4,5,4

insert overwrite directory select uname, uv, fv, cv from usert order by uname DESC limit 10;



Query :7

Sadie,3.51,0

Chris,3.51,1

Holly,3.51,1

Holly,3.51,1

Charles,3.51,0

Andrew,3.52,0

Gerald,3.52,0

Gerald,3.52,1

Andrew,3.52,0

Andrew,3.52,0

MScott,3.52,1

Gerald,3.52,1

Gerald,3.52,1

Gerald,3.52,0

Tri,3.52,1

Gerald,3.52,0

Saleem,3.52,0

Arielle,3.52,0

Gerald,3.52,0

Gerald,3.52,1

Gerald,3.52,0

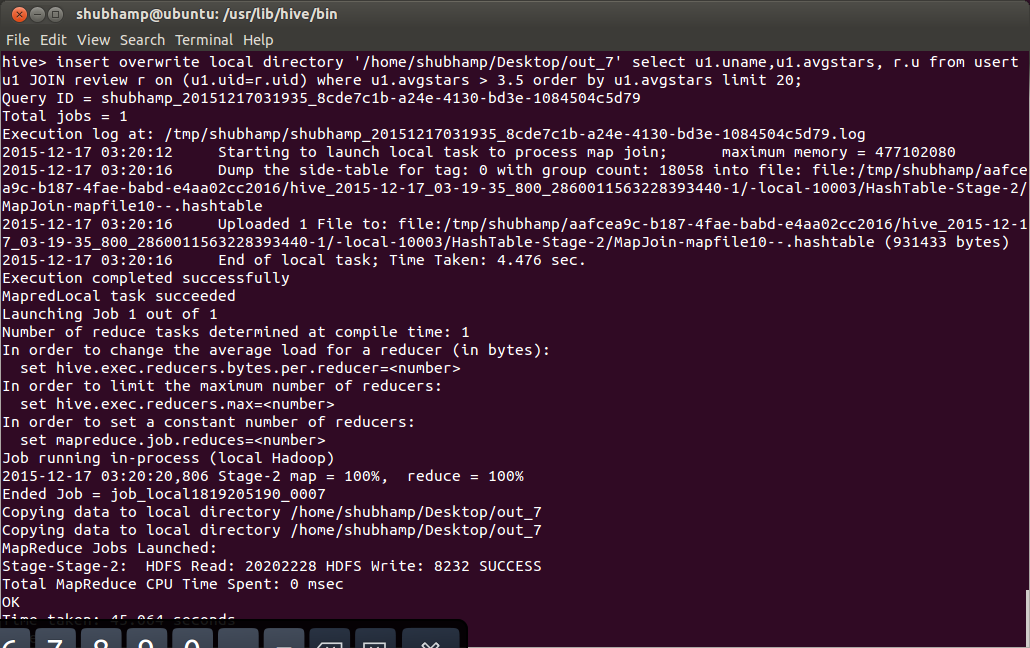
Lima Beans,3.53,0

Lima Beans,3.53,0

Lima Beans,3.53,3

Morgan And Brian,3.53,0

insert overwrite local directory 'hdfs:///hive\_out15/' select u.uname,u.avgstars, r.uv from usert u JOIN reviews r on (u.uid=r.uid) where u.avgstars>3.5 orderby avgstars limit 20;



Query: 8

Caroline.9

Graham.8

Jennifer.8

Kieran.8

Chad.7

Chad.7

Rob.7

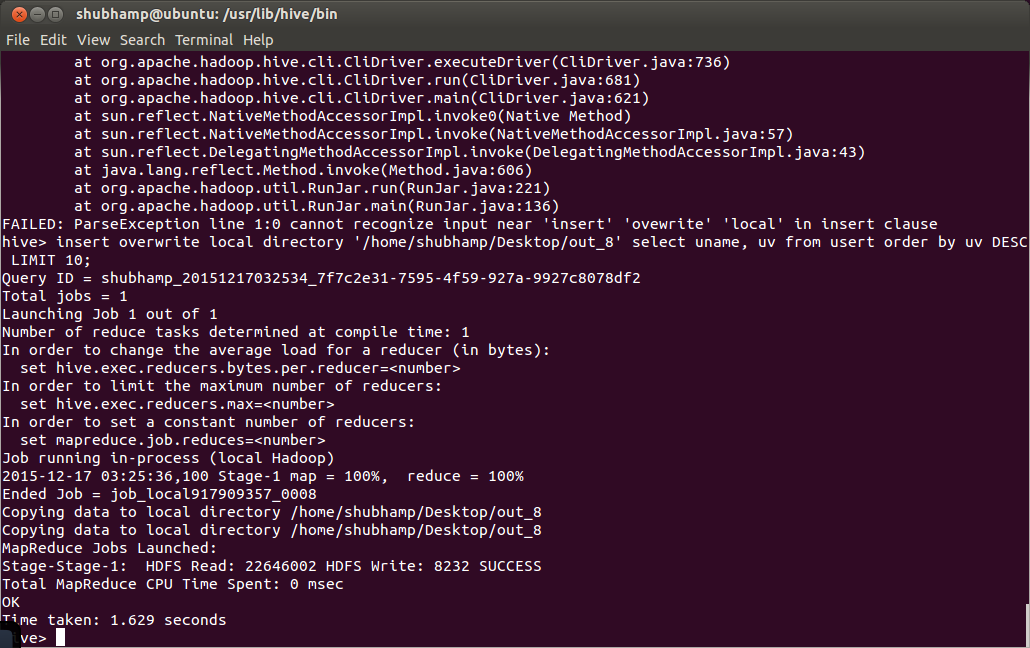
Kerri.7

Chad.7

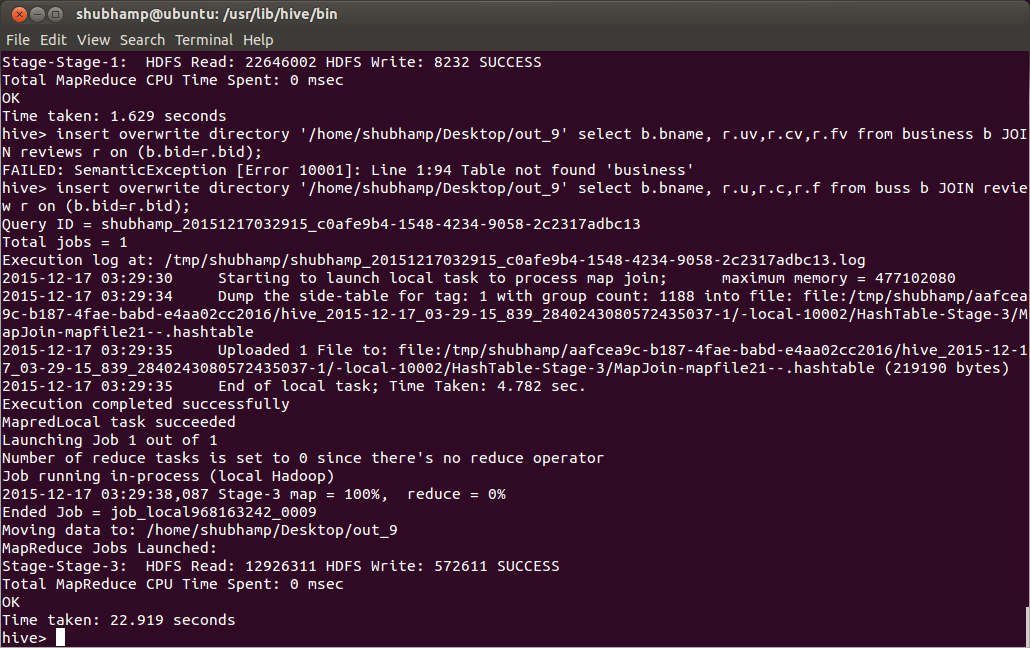
Tricia.6

insert overwrite directory 'hdfs:///hive\_out10/' select uname, uv from usert order by uv DESC LIMIT 10;

insert overwrite directory 'hdfs:///hive\_out11/' select b.bname, r.uv,r.cv,r.fv from business b JOIN reviews r on (b.bid=r.bid);



Query : 9



MAHOUT

Recommendation Analysis:

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package recommendation;

import java.io.File;

import java.io.IOException;

import java.util.List;

import org.apache.mahout.cf.taste.common.TasteException;

import org.apache.mahout.cf.taste.impl.model.file.FileDataModel;

import org.apache.mahout.cf.taste.impl.neighborhood.ThresholdUserNeighborhood;

import org.apache.mahout.cf.taste.impl.recommender.GenericUserBasedRecommender;

import org.apache.mahout.cf.taste.impl.similarity.PearsonCorrelationSimilarity;

import org.apache.mahout.cf.taste.model.DataModel;

import org.apache.mahout.cf.taste.neighborhood.UserNeighborhood;

import org.apache.mahout.cf.taste.recommender.RecommendedItem;

import org.apache.mahout.cf.taste.recommender.UserBasedRecommender;

import org.apache.mahout.cf.taste.similarity.UserSimilarity;

/\*\*

\*

\* @author Hitesh

\*/

public class RecommendationUser {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) throws IOException, TasteException {

// TODO code application logic here

DataModel model = new FileDataModel(new File ("C:\\Users\\Hitesh\\Desktop\\YELP\_FILES\\reviews\_new.csv"));

UserSimilarity similarity = new PearsonCorrelationSimilarity(model);

UserNeighborhood neighborhood = new ThresholdUserNeighborhood(3.0, similarity, model);

UserBasedRecommender recommender = new GenericUserBasedRecommender(model, neighborhood, similarity);

List<RecommendedItem> recommendations = recommender.recommend(2, 3);

for (RecommendedItem recommendation : recommendations) {

System.out.println(recommendation);

}

}

}

Visualization Using D3 and Tableau

