TOPIC :

CUSTOMER COMPLAINTS ANALYSIS(Banking).

BY :

VISHAL PANDEY

(ROLL NUMBER : 22)

UNDER THE GUIDANCE OF :

PROF. ZAINAB KHAN

INDEX :

INTRODUCTION

Consumer-generated content on the Internet continues to grow and impact the banking industry. The so-called big data analytics approach emphasizes and lever-ages the capacity to collect and analyze data with an unprecedented breadth, depth, and scale to solve real-life problems. In the banking field, there is a growing interest in utilizing user-generated data to gain insights into redressal problems that have not been well understood by conventional methods.

Indeed, big data analytics opens the door to numerous opportunities to develop new knowledge to reshape our understanding of the field and to support decision making in the banking industry. However, while a handful of studies have employed new data sources to tackle important research problems, they were conducted on an ad hoc basis and the application of the big data analytics approach in banking is yet to be well developed and established. The goal of this study is to explore and demonstrate the utility of big data analytics by using it to study core banking management variables and propose a good option for redressal mechanism for customer/consumer complaints and redressals.Previous studies define a complaint as a conflict between a consumer and a business organization in which the fairness of the resolution procedures, the interpersonal communication and behavior, and the outcome of the complaint resolution process are the principal evaluative criteria used by the customer.

In our opinion, a complaint is not necessarily a conflict, however, it can create a conflict between a customer and a business organization, when the answer to the consumer’s complaint is not satisfactory.The consumer complaint dataset handles all complaints according to states, products, zip codes, company and sub product categories. Research questions are formulated with the focus on using online customer reviews to enrich our understanding of these constructs. The methodology section details data collection and the text analytical approach utilized to answer the research questions. Findings are then presented and discussed. Finally, the study’s contributions to literature and practice as well as directions for future research are discussed.

IMPLEMENTATION DETAILS

This block of our project basically deals with a core or base software and then the dependencies associated with it so we have divided it in a two block flowchart as follows:

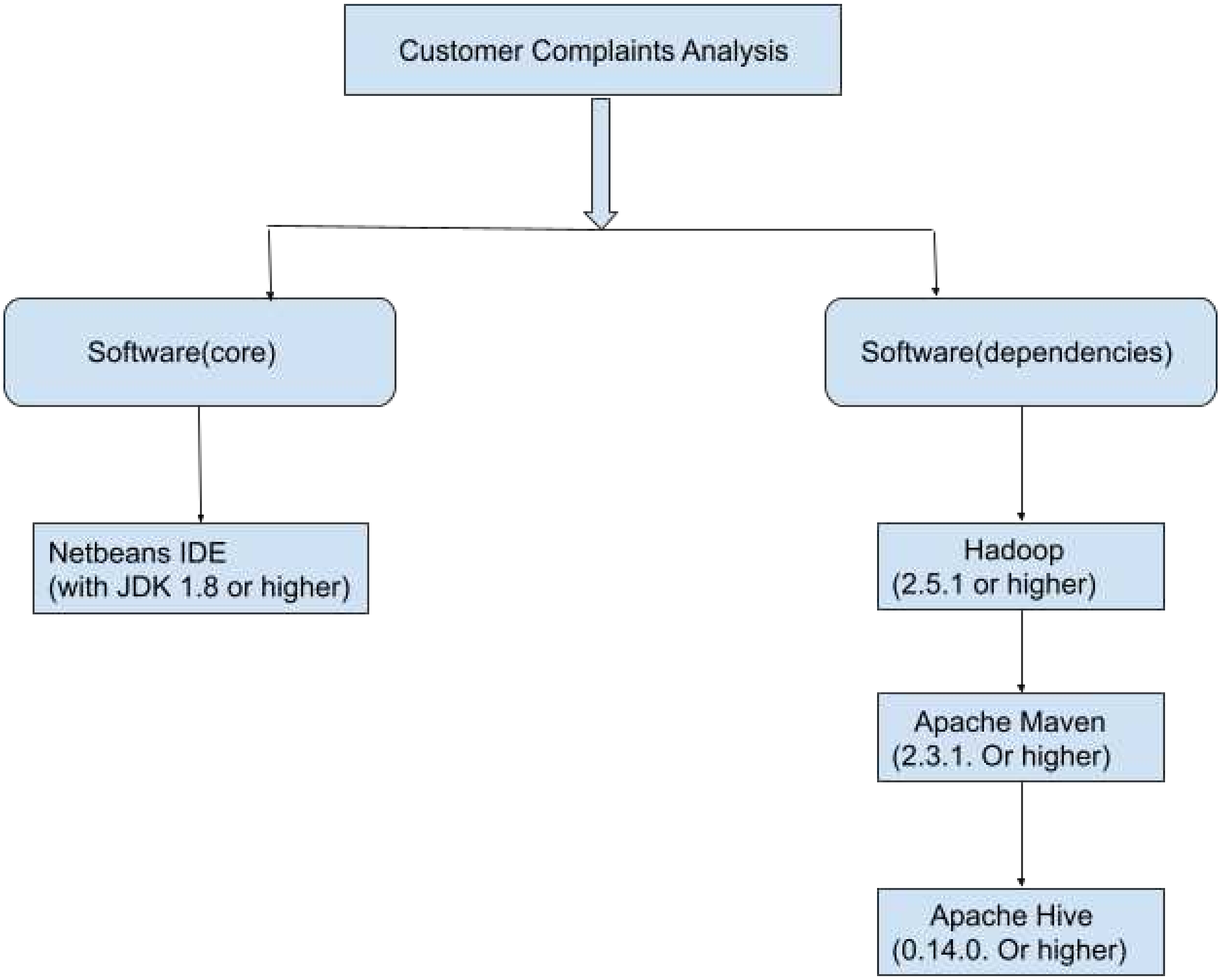


Fig: Flowchart for implementation of CCA

Now further we would like to tell you what exactly needs to be done for smooth running of our project module that is Customer complaints analysis(CCA) in banking industry:

- First and foremost comes a basic hardware requirement to run Netbeans IDE with a version of JDK (java development kit) 1.8 or higher version which requires:

- Any windows version 7 or above
- RAM : 128MB
- Disk Space : 124MB for JRE; 2MB for Java update
- Processor : Minimum pentium 2 266 MHz processor
- Browser : Internet Explorer 9 and above ; Firefox
- NetBeans IDE 8.2 or higher version
- Operating System: Windows 10 / Window 8 / Windows 7 / XP / Vista / Linux
- Memory (RAM): 512MB (for good results)
- Hard Disk: 900MB
- Processor: Intel Pentium IV or above.

- Secondly we have to install all the dependencies mentioned above such as

    - Apache Hadoop (2.5.1 or higher) and installing it requires basic requirements such as RAM : 1GB, HDD 20GB etc
    - Apache Maven (2.3.1 or higher) and installing it has prerequisites such as follows JDK : require JDK 1.7 or above to execute - they still allows you to build against 1.3 and other JDK versions , Disk : Approximately 10MB is required for the Maven installation itself In addition to that, additional disk space will be used for your local Maven repository. The size of your local repository will vary depending on usage but expect at least 500MB.
    - Apache Hive (0.14.0. Or higher) also has some prerequisites that need to be taken care of such as Apache Maven .

Next we have to ensure that Hadoop (2.5.1 or higher) , and Maven are downloaded and installed as they are mandatory for build. Also the sequence files will be sorted in the by default file system provided by Hadoop which is HDFS.After all this is done we need to run our main project file which has the back end code for running the three modules of our project such as classification , metrics performance and analysis in the last. The back end code is written in java and will run on Netbeans IDE so we need to start it from there after that we have to run some path environment settings to set up java_home , maven home , hadoop home and hdfs so that classification is done properly on the data provided.

Simultaneously there are some .CSV files (Comma-separated values) to test our modules for extensive analysis and classification. These .csv files are taken from the internet with due permission and are used in our module , these are available with us on the webpage link provided. Lastly we need to run maven repositories and their POM (project object model). The Classification takes a bit of time as the hardware we tested it on has some limitations and the speed of execution can be improved with scalability and availability of resources.

Also if the classification and analysis is done once on a particular machine that machine responds faster enough hence improving the performance metrics.The .CSV files would be in a jar format also the links would be provided to pre download it. Next we need to execute some mahout naive bytes classification steps and using mahout and performance matrix analysis too. Also we would require the .CSV files from the CSVReader jar file(the link would be provided as it would act as our dataset). Next we would need Hive to be installed and Hadoop for analysis , The analysis would be shown throughout using geographic maps , graphs,pie charts and bar graphs. These analysis will be with respect to the dataset of the customer complaints and their form of complaints. For example which states have more number of X complaints or which company's complaints are more in Y department of the bank .

EXPERIMENTAL SETUP AND RESULTS

For the experimental set up the required hardware and software prerequisites have been talked about in the implementation details previously but just to highlight some really important aspects for the projects implementation are JDK installation and any idea that can run java applications or programs for that matter. Also we need Apache maven , Apache hive and Apache mahout for experimenting our project and give viable results and last but not least take care of the environmental variables that you set for different paths in your system as there are some of the most common errors people come across and get frustrated with.

After installation of the required hardware and software prerequisites we need to follow a plan for the experiment which is are divided into two parts and are as follows

- Compilation Instructions : These include a pre defined set path for smoothing data compilation experience and basically it deals with the aspects of getting your repositories cloned , installed and compiled.

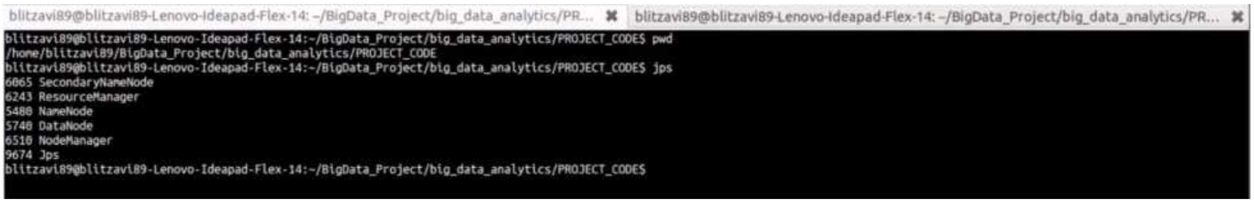
- Execution instructions : In this section of experiment we deal with specific queries to get the desired results on the datasets and for that matter CSV (Comma separated values)files which are shot at various terminals such as for hive, hadoop etc.

- [NOTE: Please download the required CsvReader jar file from the following link http://javacsv.sourceforge.net/com/csvreader/CsvReader.html]

COMPILATION INSTRUCTIONS :

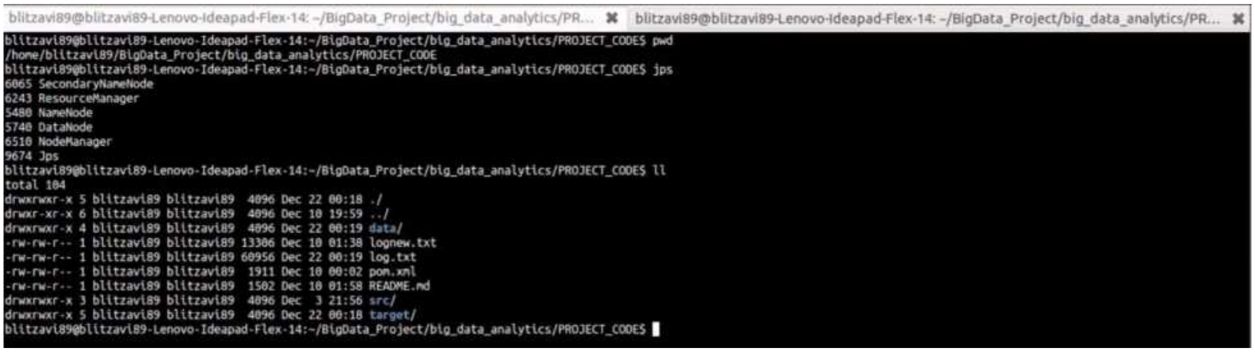
- Project code needs to be cloned from the data repository .Clone the directory and navigate to PROJECT_CODE.
- Install maven if you don't have it.
    - mvn clean install (Do this after navigating to PROJECT_CODE folder)
- The jar file compiled goes into location ./target/*.jar
    - CONTENTS OF target/*.jar
        - com/
        - com/bigdata/
        - com/bigdata/complaintanalysis/
        - com/bigdata/complaintanalysis/ColumnReducer.class
        - com/bigdata/complaintanalysis/ClassificationAutomator.class
        - com/bigdata/complaintanalysis/StripColumn.class
        - com/bigdata/complaintanalysis/ComplaintsCSVtoSeq.class
        - com/bigdata/complaintanalysis/StatewiseSorter.class

- All required processes should be running for hadoop check using jps command


blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14: ~/BigData_Project/big_data_analytics/PR...
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14: ~/BigData_Project/big_data_analytics/PR...

```
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ pwd
/home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ jps
6065 SecondaryNameNode
6243 ResourceManager
5480 NameNode
5748 DataNode
6510 NodeManager
9674 Jps
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$
```

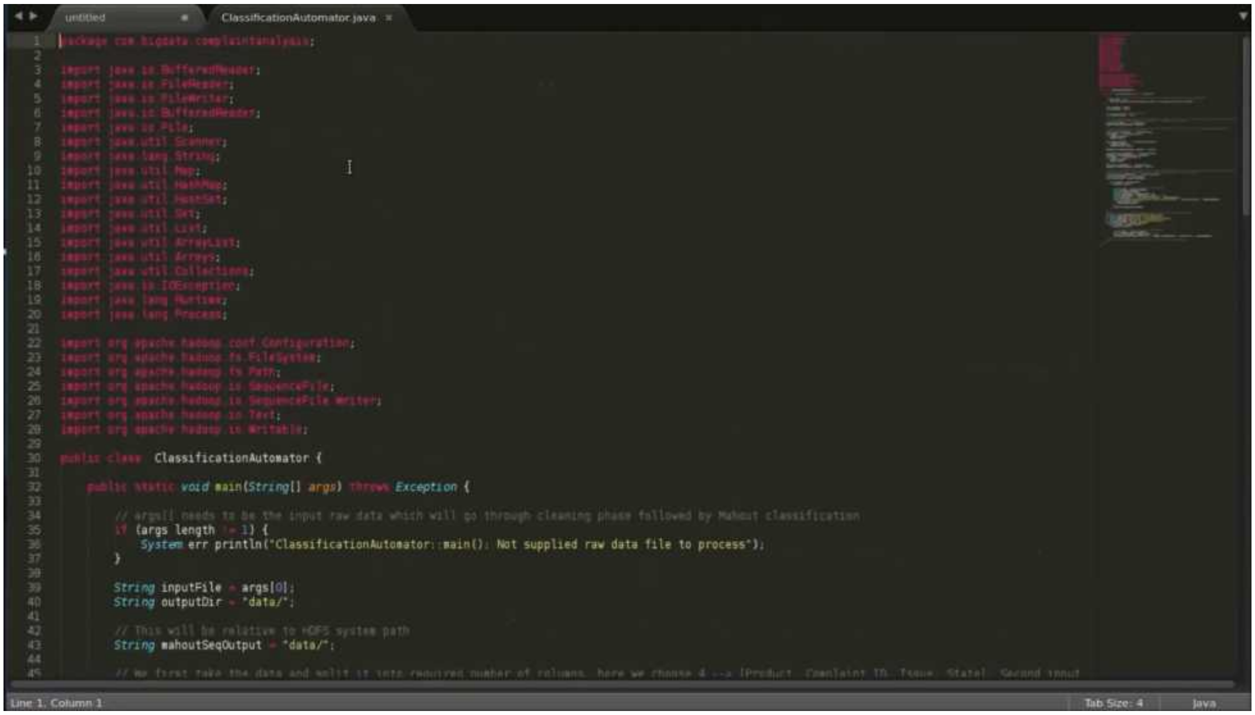
- Check root directory file content for POM a XML which has all the dependencies that are needed to be compiled



blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14: ~/BigData_Project/big_data_analytics/PR...
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14: ~/BigData_Project/big_data_analytics/PR...

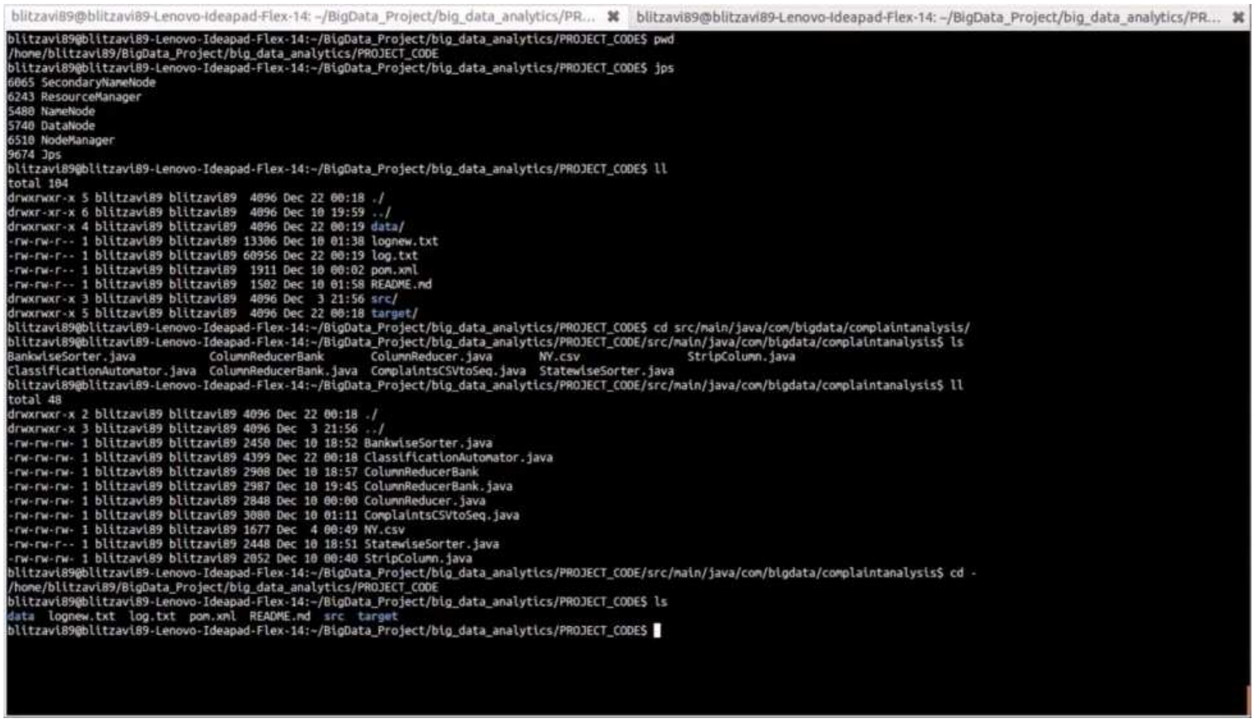
```
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ pwd
/home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ jps
6065 SecondaryNameNode
6243 ResourceManager
5480 NameNode
5740 DataNode
6510 NodeManager
9674 Jps
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ll
total 104
drwxrwxr-x 5 blitzavi89 blitzavi89  4096 Dec 22 00:18 ./
drwxr-xr-x 6 blitzavi89 blitzavi89  4096 Dec 10 19:59 ../
drwxrwxr-x 4 blitzavi89 blitzavi89  4096 Dec 22 00:19 data/
-rw-rw-r-- 1 blitzavi89 blitzavi89 13306 Dec 10 01:38 lognew.txt
-rw-rw-r-- 1 blitzavi89 blitzavi89 60956 Dec 22 00:19 log.txt
-rw-rw-r-- 1 blitzavi89 blitzavi89  1911 Dec 10 00:02 pom.xml
-rw-rw-r-- 1 blitzavi89 blitzavi89  1502 Dec 10 01:58 README.md
drwxrwxr-x 3 blitzavi89 blitzavi89  4096 Dec  3 21:56 src/
drwxrwxr-x 5 blitzavi89 blitzavi89  4096 Dec 22 00:18 target/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$
```

- Classification automator main file which basically calls other java class files inside the source directory



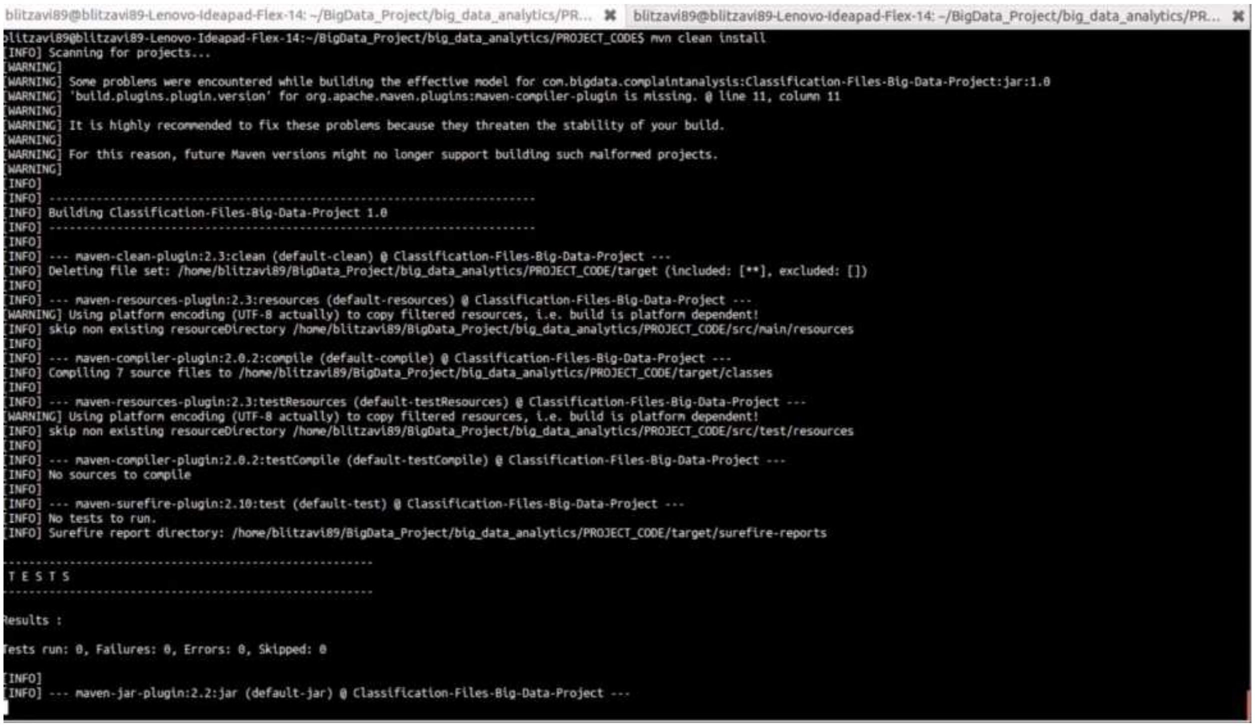
```java
package com.bigdata.complaintanalysis;

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.BufferedReader;
import java.io.File;
import java.util.Scanner;
import java.lang.String;
import java.util.Map;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Set;
import java.util.List;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.io.IOException;
import java.lang.Runtime;
import java.lang.Process;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.SequenceFile;
import org.apache.hadoop.io.SequenceFile.Writer;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.Writable;

public class  ClassificationAutomator {

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

        // args[] needs to be the input raw data which will go through cleaning phase followed by Mahout classification
        if (args.length != 1) {
            System.err.println("ClassificationAutomator::main(): Not supplied raw data file to process");
        }

        String inputFile = args[0];
        String outputDir = "data/";

        // This will be relative to HDFS system path
        String mahoutSeqOutput = "data/";

        // We first take the data and split it into required number of columns, here we choose 4 --> [Product, Complaint ID, Issue, State], Second input
```

- After going into source java directory we can see a list of java files which are built by maven because they are dependent on each other and required for compilation and execution


blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14: ~/BigData_Project/big_data_analytics/PR...
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14: ~/BigData_Project/big_data_analytics/PR...

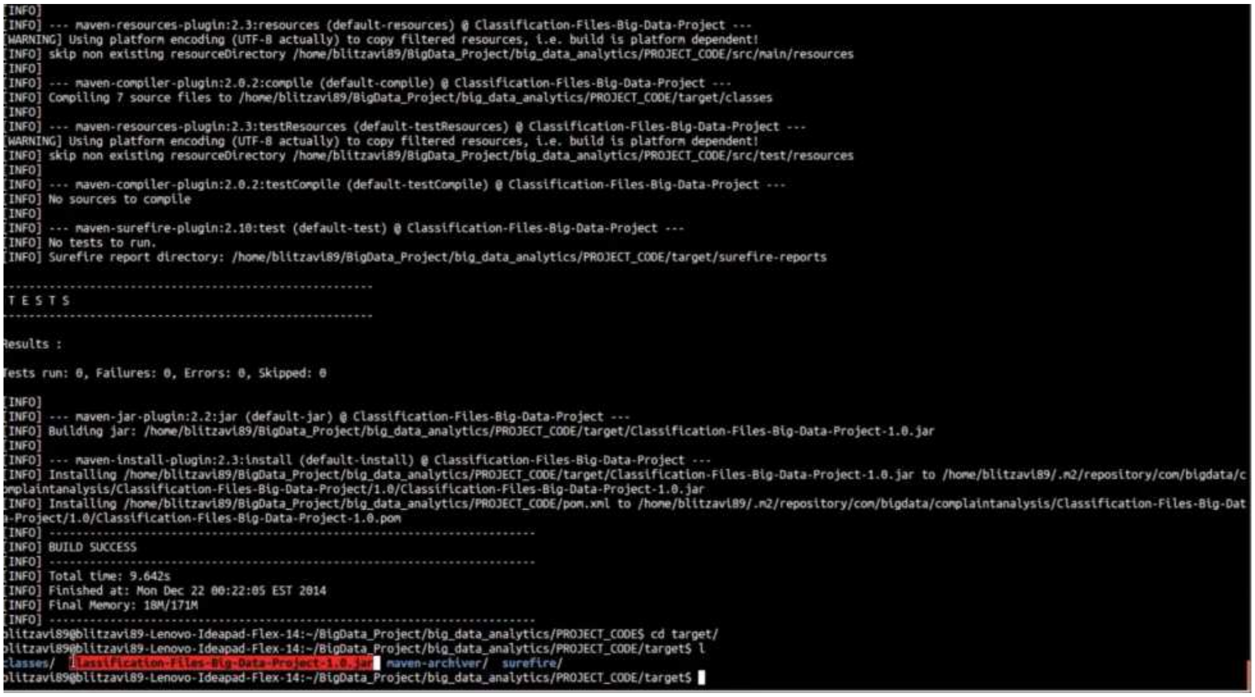
```
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ pwd
/home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ jps
6065 SecondaryNameNode
6243 ResourceManager
5480 NameNode
5740 DataNode
6510 NodeManager
9674 Jps
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ll
total 104
drwxrwxr-x 5 blitzavi89 blitzavi89  4096 Dec 22 00:18 ./
drwxr-xr-x 6 blitzavi89 blitzavi89  4096 Dec 10 19:59 ../
drwxrwxr-x 4 blitzavi89 blitzavi89  4096 Dec 22 00:19 data/
-rw-rw-r-- 1 blitzavi89 blitzavi89 13306 Dec 10 01:38 lognew.txt
-rw-rw-r-- 1 blitzavi89 blitzavi89 60956 Dec 22 00:19 log.txt
-rw-rw-r-- 1 blitzavi89 blitzavi89  1911 Dec 10 00:02 pom.xml
-rw-rw-r-- 1 blitzavi89 blitzavi89  1502 Dec 10 01:58 README.md
drwxrwxr-x 3 blitzavi89 blitzavi89  4096 Dec  3 21:56 src/
drwxrwxr-x 5 blitzavi89 blitzavi89  4096 Dec 22 00:18 target/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ cd src/main/java/com/bigdata/complaintanalysis/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/src/main/java/com/bigdata/complaintanalysis$ ls
BankwiseSorter.java          ColumnReducerBank       ColumnReducer.java       NY.csv                StripColumn.java
ClassificationAutomator.java  ColumnReducerBank.java  ComplaintsCSVtoSeq.java  StatewiseSorter.java
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/src/main/java/com/bigdata/complaintanalysis$ ll
total 48
drwxrwxr-x 2 blitzavi89 blitzavi89 4096 Dec 22 00:18 ./
drwxrwxr-x 3 blitzavi89 blitzavi89 4096 Dec  3 21:56 ../
-rw-rw-rw- 1 blitzavi89 blitzavi89 2450 Dec 10 18:52 BankwiseSorter.java
-rw-rw-rw- 1 blitzavi89 blitzavi89 4399 Dec 22 00:18 ClassificationAutomator.java
-rw-rw-rw- 1 blitzavi89 blitzavi89 2908 Dec 10 18:57 ColumnReducerBank
-rw-rw-rw- 1 blitzavi89 blitzavi89 2987 Dec 10 19:45 ColumnReducerBank.java
-rw-rw-rw- 1 blitzavi89 blitzavi89 2848 Dec 10 00:00 ColumnReducer.java
-rw-rw-rw- 1 blitzavi89 blitzavi89 3080 Dec 10 01:11 ComplaintsCSVtoSeq.java
-rw-rw-rw- 1 blitzavi89 blitzavi89 1677 Dec  4 00:49 NY.csv
-rw-rw-r-- 1 blitzavi89 blitzavi89 2448 Dec 10 18:51 StatewiseSorter.java
-rw-rw-rw- 1 blitzavi89 blitzavi89 2052 Dec 10 00:40 StripColumn.java
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/src/main/java/com/bigdata/complaintanalysis$ cd -
/home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ls
data  lognew.txt  log.txt  pom.xml  README.md  src  target
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ 
```

- Now do a maven built now by a clean install


blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14: ~/BigData_Project/big_data_analytics/PR...
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14: ~/BigData_Project/big_data_analytics/PR...

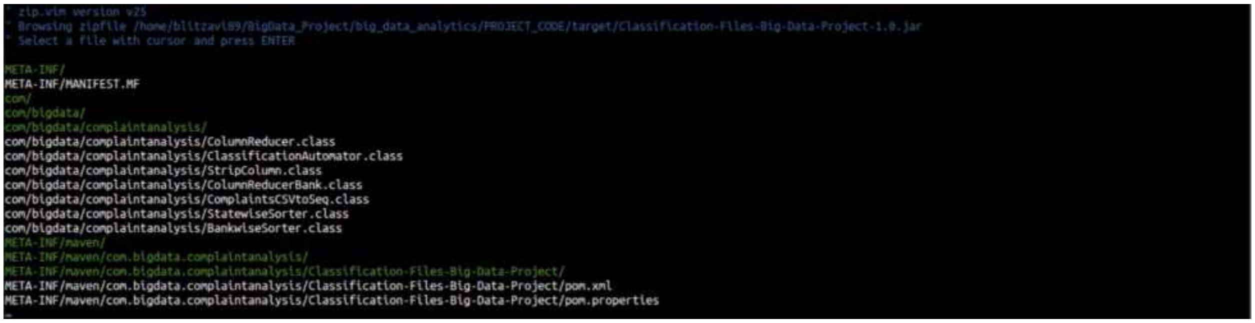
```
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ mvn clean install
[INFO] Scanning for projects...
[WARNING]
[WARNING] Some problems were encountered while building the effective model for com.bigdata.complaintanalysis:Classification-Files-Big-Data-Project:jar:1.0
[WARNING] 'build.plugins.plugin.version' for org.apache.maven.plugins:maven-compiler-plugin is missing. @ line 11, column 11
[WARNING]
[WARNING] It is highly recommended to fix these problems because they threaten the stability of your build.
[WARNING]
[WARNING] For this reason, future Maven versions might no longer support building such malformed projects.
[WARNING]
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building Classification-Files-Big-Data-Project 1.0
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.3:clean (default-clean) @ Classification-Files-Big-Data-Project ---
[INFO] Deleting file set: /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target (included: [**], excluded: [])
[INFO]
[INFO] --- maven-resources-plugin:2.3:resources (default-resources) @ Classification-Files-Big-Data-Project ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:2.0.2:compile (default-compile) @ Classification-Files-Big-Data-Project ---
[INFO] Compiling 7 source files to /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.3:testResources (default-testResources) @ Classification-Files-Big-Data-Project ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:2.0.2:testCompile (default-testCompile) @ Classification-Files-Big-Data-Project ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-surefire-plugin:2.10:test (default-test) @ Classification-Files-Big-Data-Project ---
[INFO] No tests to run.
[INFO] Surefire report directory: /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:2.2:jar (default-jar) @ Classification-Files-Big-Data-Project ---
```

- Maven will basically install all the necessities which we require on our system



```
[INFO]
[INFO] --- maven-resources-plugin:2.3:resources (default-resources) @ Classification-Files-Big-Data-Project ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:2.0.2:compile (default-compile) @ Classification-Files-Big-Data-Project ---
[INFO] Compiling 7 source files to /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.3:testResources (default-testResources) @ Classification-Files-Big-Data-Project ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:2.0.2:testCompile (default-testCompile) @ Classification-Files-Big-Data-Project ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-surefire-plugin:2.10:test (default-test) @ Classification-Files-Big-Data-Project ---
[INFO] No tests to run.
[INFO] Surefire report directory: /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:2.2:jar (default-jar) @ Classification-Files-Big-Data-Project ---
[INFO] Building jar: /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target/Classification-Files-Big-Data-Project-1.0.jar
[INFO]
[INFO] --- maven-install-plugin:2.3:install (default-install) @ Classification-Files-Big-Data-Project ---
[INFO] Installing /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target/Classification-Files-Big-Data-Project-1.0.jar to /home/blitzavi89/.m2/repository/com/bigdata/c
omplaintanalysis/Classification-Files-Big-Data-Project/1.0/Classification-Files-Big-Data-Project-1.0.jar
[INFO] Installing /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/pom.xml to /home/blitzavi89/.m2/repository/com/bigdata/complaintanalysis/Classification-Files-Big-Dat
a-Project/1.0/Classification-Files-Big-Data-Project-1.0.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 9.642s
[INFO] Finished at: Mon Dec 22 00:22:05 EST 2014
[INFO] Final Memory: 18M/171M
[INFO] ------------------------------------------------------------------------
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ cd target/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/target$ l
classes/  Classification-Files-Big-Data-Project-1.0.jar  maven-archiver/  surefire/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/target$ 
```

- We will use hadoop jar command to compile and we will need hadoop packages also and then



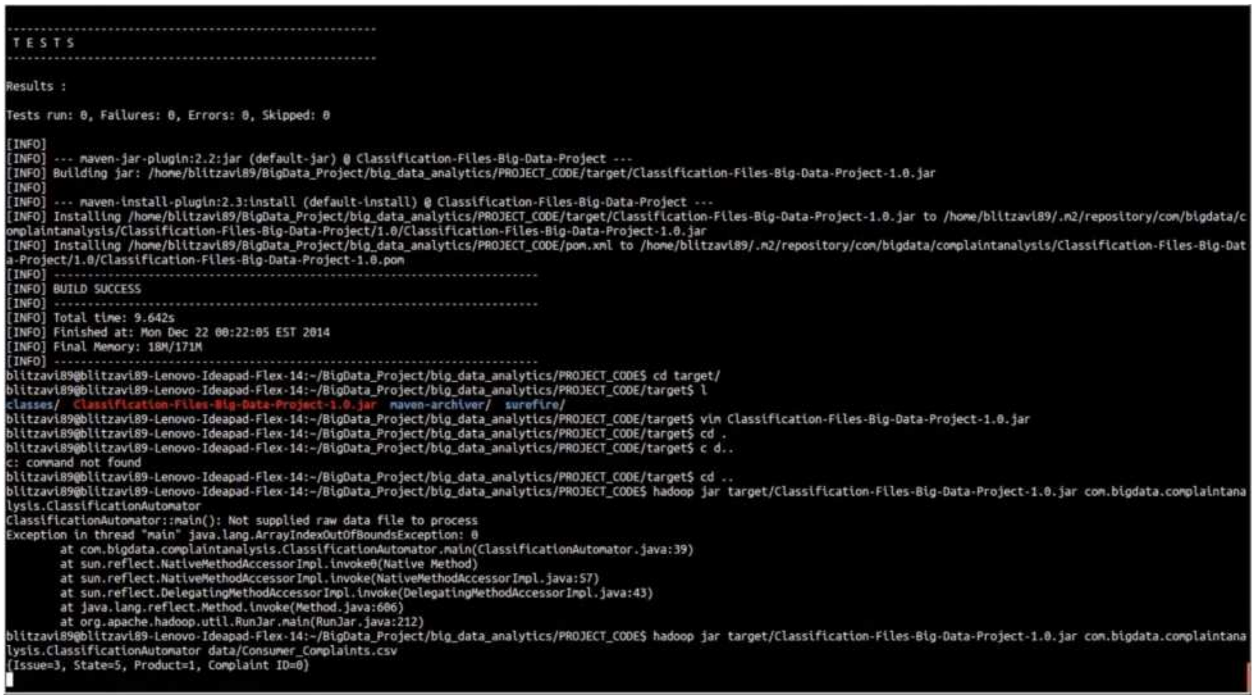
```
" zip.vim version v25
" Browsing zipfile /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target/Classification-Files-Big-Data-Project-1.0.jar
" Select a file with cursor and press ENTER

META-INF/
META-INF/MANIFEST.MF
com/
com/bigdata/
com/bigdata/complaintanalysis/
com/bigdata/complaintanalysis/ColumnReducer.class
com/bigdata/complaintanalysis/ClassificationAutomator.class
com/bigdata/complaintanalysis/StripColumn.class
com/bigdata/complaintanalysis/ColumnReducerBank.class
com/bigdata/complaintanalysis/ComplaintsCSVtoSeq.class
com/bigdata/complaintanalysis/StatewiseSorter.class
com/bigdata/complaintanalysis/BankwiseSorter.class
META-INF/maven/
META-INF/maven/com.bigdata.complaintanalysis/
META-INF/maven/com.bigdata.complaintanalysis/Classification-Files-Big-Data-Project/
META-INF/maven/com.bigdata.complaintanalysis/Classification-Files-Big-Data-Project/pom.xml
META-INF/maven/com.bigdata.complaintanalysis/Classification-Files-Big-Data-Project/pom.properties
```

This is all in the compilation section of our module, The next section is a continuation of our modules execution after what we have successfully done till now.

EXECUTION INSTRUCTIONS

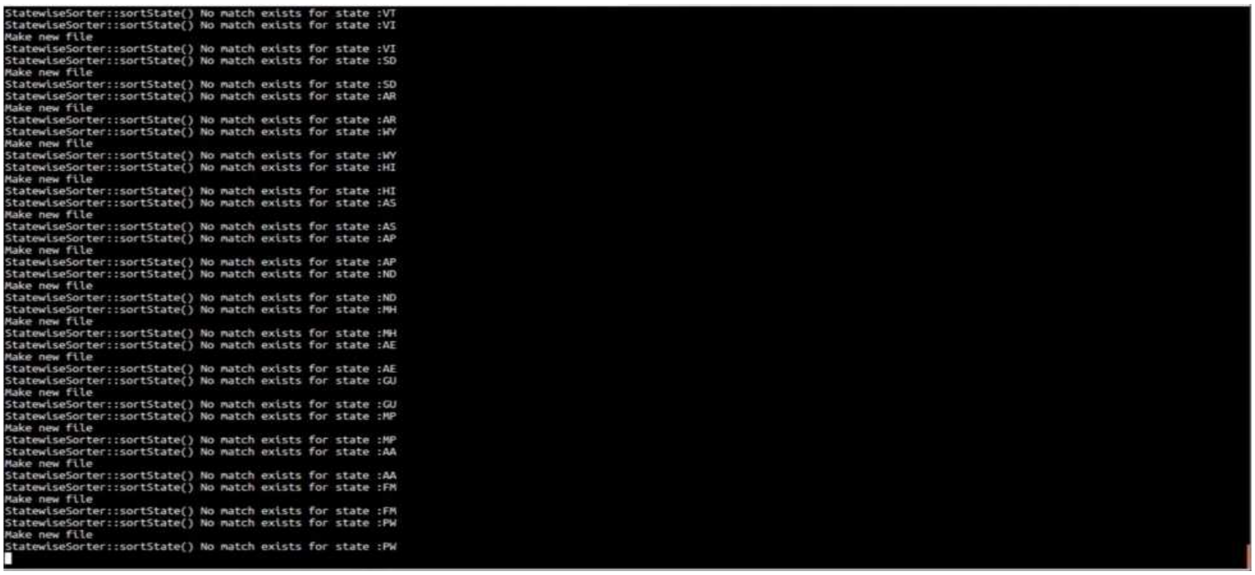
In the execution section of the module we are going to deal with the execution of our main classes and running CSV files into our hdfs system for better naive bayes classification and its related confusion matrix in a glance.

- So we can now execute our class file, we can use our main class file as input and also supply an input which is a csv file which is a cleaned dataset.



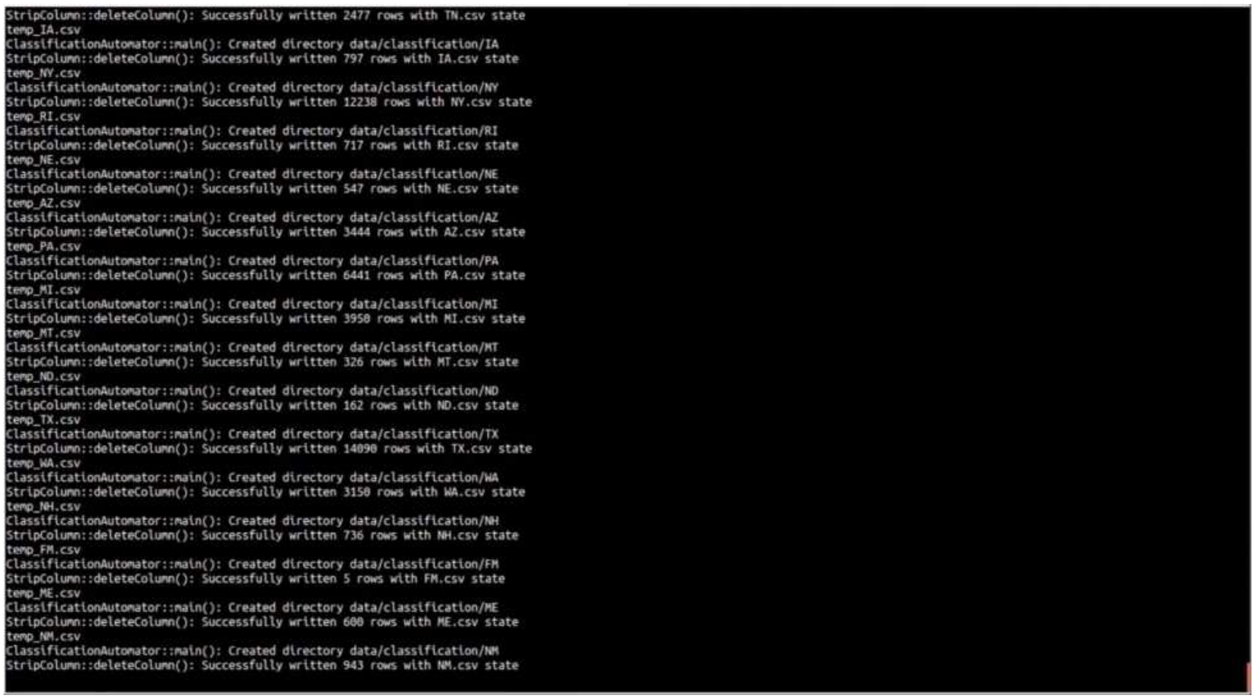
```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:2.2:jar (default-jar) @ Classification-Files-Big-Data-Project ---
[INFO] Building jar: /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target/Classification-Files-Big-Data-Project-1.0.jar
[INFO]
[INFO] --- maven-install-plugin:2.3:install (default-install) @ Classification-Files-Big-Data-Project ---
[INFO] Installing /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/target/Classification-Files-Big-Data-Project-1.0.jar to /home/blitzavi89/.m2/repository/com/bigdata/c
omplaintanalysis/Classification-Files-Big-Data-Project/1.0/Classification-Files-Big-Data-Project-1.0.jar
[INFO] Installing /home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/pom.xml to /home/blitzavi89/.m2/repository/com/bigdata/complaintanalysis/Classification-Files-Big-Dat
a-Project/1.0/Classification-Files-Big-Data-Project-1.0.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 9.642s
[INFO] Finished at: Mon Dec 22 00:22:05 EST 2014
[INFO] Final Memory: 18M/171M
[INFO] ------------------------------------------------------------------------
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ cd target/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/target$ l
classes/  Classification-Files-Big-Data-Project-1.0.jar  maven-archiver/  surefire/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/target$ vim Classification-Files-Big-Data-Project-1.0.jar
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/target$ cd .
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/target$ c d..
c: command not found
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/target$ cd ..
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ hadoop jar target/Classification-Files-Big-Data-Project-1.0.jar com.bigdata.complaintana
lysis.ClassificationAutomator
ClassificationAutomator::main(): Not supplied raw data file to process
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 0
        at com.bigdata.complaintanalysis.ClassificationAutomator.main(ClassificationAutomator.java:39)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:606)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:212)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ hadoop jar target/Classification-Files-Big-Data-Project-1.0.jar com.bigdata.complaintana
lysis.ClassificationAutomator data/Consumer_Complaints.csv
{Issue=3, State=5, Product=1, Complaint ID=0}
```

- This actually creates a lot of log files in the background and the states and banks are split. We also have the sequence files being generated by ComplaintstoCSVsequence generator.



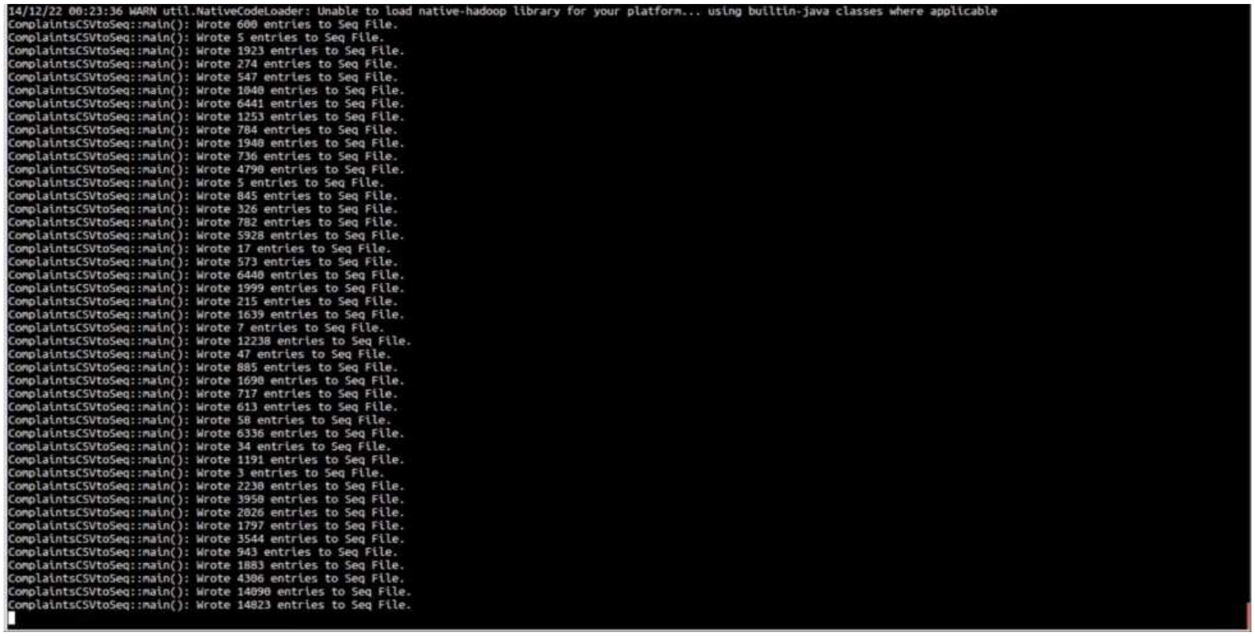
```
StatewiseSorter::sortState() No match exists for state :VT
StatewiseSorter::sortState() No match exists for state :VI
Make new file
StatewiseSorter::sortState() No match exists for state :VI
StatewiseSorter::sortState() No match exists for state :SD
Make new file
StatewiseSorter::sortState() No match exists for state :SD
StatewiseSorter::sortState() No match exists for state :AR
Make new file
StatewiseSorter::sortState() No match exists for state :AR
StatewiseSorter::sortState() No match exists for state :WY
Make new file
StatewiseSorter::sortState() No match exists for state :WY
StatewiseSorter::sortState() No match exists for state :HI
Make new file
StatewiseSorter::sortState() No match exists for state :HI
StatewiseSorter::sortState() No match exists for state :AS
Make new file
StatewiseSorter::sortState() No match exists for state :AS
StatewiseSorter::sortState() No match exists for state :AP
Make new file
StatewiseSorter::sortState() No match exists for state :AP
StatewiseSorter::sortState() No match exists for state :ND
Make new file
StatewiseSorter::sortState() No match exists for state :ND
StatewiseSorter::sortState() No match exists for state :MH
Make new file
StatewiseSorter::sortState() No match exists for state :MH
StatewiseSorter::sortState() No match exists for state :AE
Make new file
StatewiseSorter::sortState() No match exists for state :AE
StatewiseSorter::sortState() No match exists for state :GU
Make new file
StatewiseSorter::sortState() No match exists for state :GU
StatewiseSorter::sortState() No match exists for state :MP
Make new file
StatewiseSorter::sortState() No match exists for state :MP
StatewiseSorter::sortState() No match exists for state :AA
Make new file
StatewiseSorter::sortState() No match exists for state :AA
StatewiseSorter::sortState() No match exists for state :FM
Make new file
StatewiseSorter::sortState() No match exists for state :FM
StatewiseSorter::sortState() No match exists for state :PW
Make new file
StatewiseSorter::sortState() No match exists for state :PW
```

- These Sequence files which are generated by the ComplaintstoCSVsequence generator will then be automatically pushed to our hdfs(hadoop distributed file system).



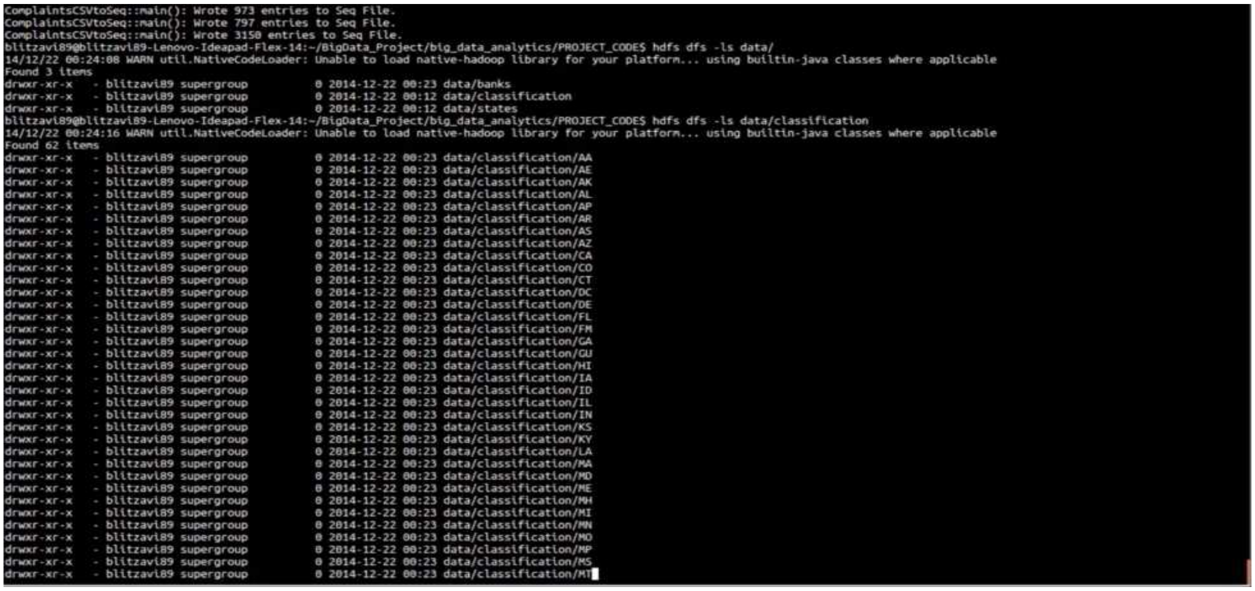
```
StripColumn::deleteColumn(): Successfully written 2477 rows with TN.csv state
temp_IA.csv
ClassificationAutomator::main(): Created directory data/classification/IA
StripColumn::deleteColumn(): Successfully written 797 rows with IA.csv state
temp_NY.csv
ClassificationAutomator::main(): Created directory data/classification/NY
StripColumn::deleteColumn(): Successfully written 12238 rows with NY.csv state
temp_RI.csv
ClassificationAutomator::main(): Created directory data/classification/RI
StripColumn::deleteColumn(): Successfully written 717 rows with RI.csv state
temp_NE.csv
ClassificationAutomator::main(): Created directory data/classification/NE
StripColumn::deleteColumn(): Successfully written 547 rows with NE.csv state
temp_AZ.csv
ClassificationAutomator::main(): Created directory data/classification/AZ
StripColumn::deleteColumn(): Successfully written 3444 rows with AZ.csv state
temp_PA.csv
ClassificationAutomator::main(): Created directory data/classification/PA
StripColumn::deleteColumn(): Successfully written 6441 rows with PA.csv state
temp_MI.csv
ClassificationAutomator::main(): Created directory data/classification/MI
StripColumn::deleteColumn(): Successfully written 3958 rows with MI.csv state
temp_MT.csv
ClassificationAutomator::main(): Created directory data/classification/MT
StripColumn::deleteColumn(): Successfully written 326 rows with MT.csv state
temp_ND.csv
ClassificationAutomator::main(): Created directory data/classification/ND
StripColumn::deleteColumn(): Successfully written 162 rows with ND.csv state
temp_TX.csv
ClassificationAutomator::main(): Created directory data/classification/TX
StripColumn::deleteColumn(): Successfully written 14090 rows with TX.csv state
temp_WA.csv
ClassificationAutomator::main(): Created directory data/classification/WA
StripColumn::deleteColumn(): Successfully written 3150 rows with WA.csv state
temp_NH.csv
ClassificationAutomator::main(): Created directory data/classification/NH
StripColumn::deleteColumn(): Successfully written 736 rows with NH.csv state
temp_FM.csv
ClassificationAutomator::main(): Created directory data/classification/FM
StripColumn::deleteColumn(): Successfully written 5 rows with FM.csv state
temp_ME.csv
ClassificationAutomator::main(): Created directory data/classification/ME
StripColumn::deleteColumn(): Successfully written 600 rows with ME.csv state
temp_NM.csv
ClassificationAutomator::main(): Created directory data/classification/NM
StripColumn::deleteColumn(): Successfully written 943 rows with NM.csv state
```

- Now hdfs (hadoop distributed file system) will be called and files are pushed into hadoop, Also these sequence files are hexadecimal encoded.



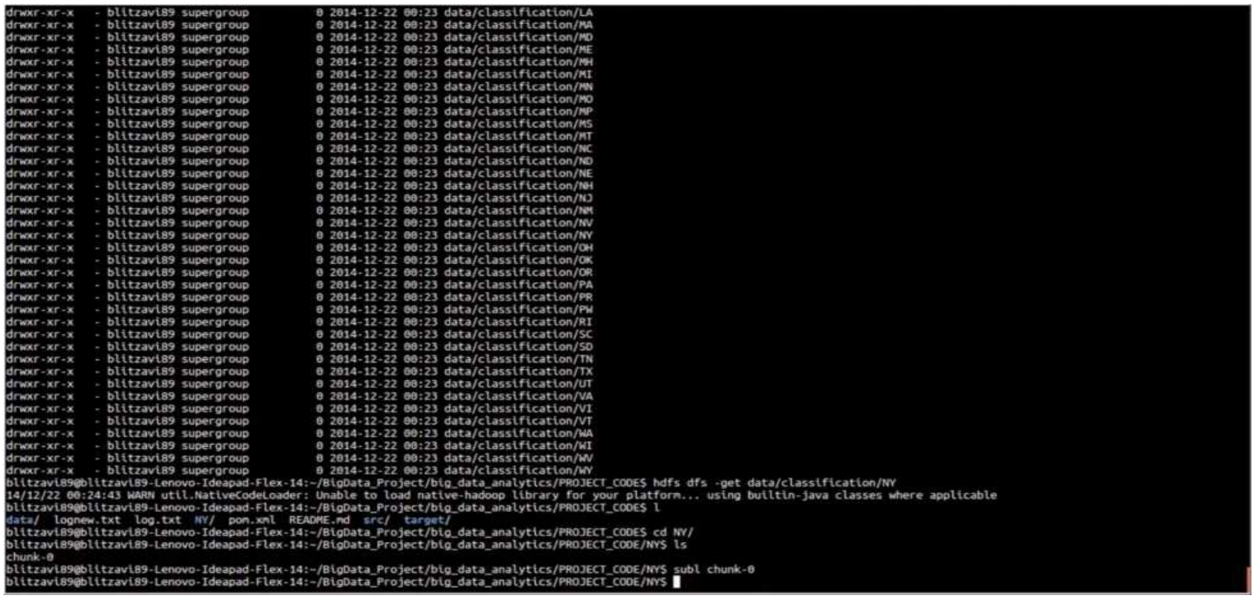
```
14/12/22 00:23:36 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
ComplaintsCSVtoSeq::main(): Wrote 600 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 5 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1923 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 274 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 547 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1040 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 6441 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1253 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 784 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1940 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 736 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 4790 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 5 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 845 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 326 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 782 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 5928 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 17 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 573 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 6440 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1999 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 215 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1639 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 7 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 12238 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 47 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 885 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1690 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 717 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 613 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 58 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 6336 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 34 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1191 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 3 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 2230 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 3950 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 2026 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1797 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 3544 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 943 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 1883 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 4306 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 14090 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 14823 entries to Seq File.
```

- After the above operation is done successfully we now have a lot of useful directories inside the hdfs such as states, classification and bank directory. And inside the classification directory we will have a list of states and each state directory will have its chunk-0 file which basically is a sequence file for every state's CSV.



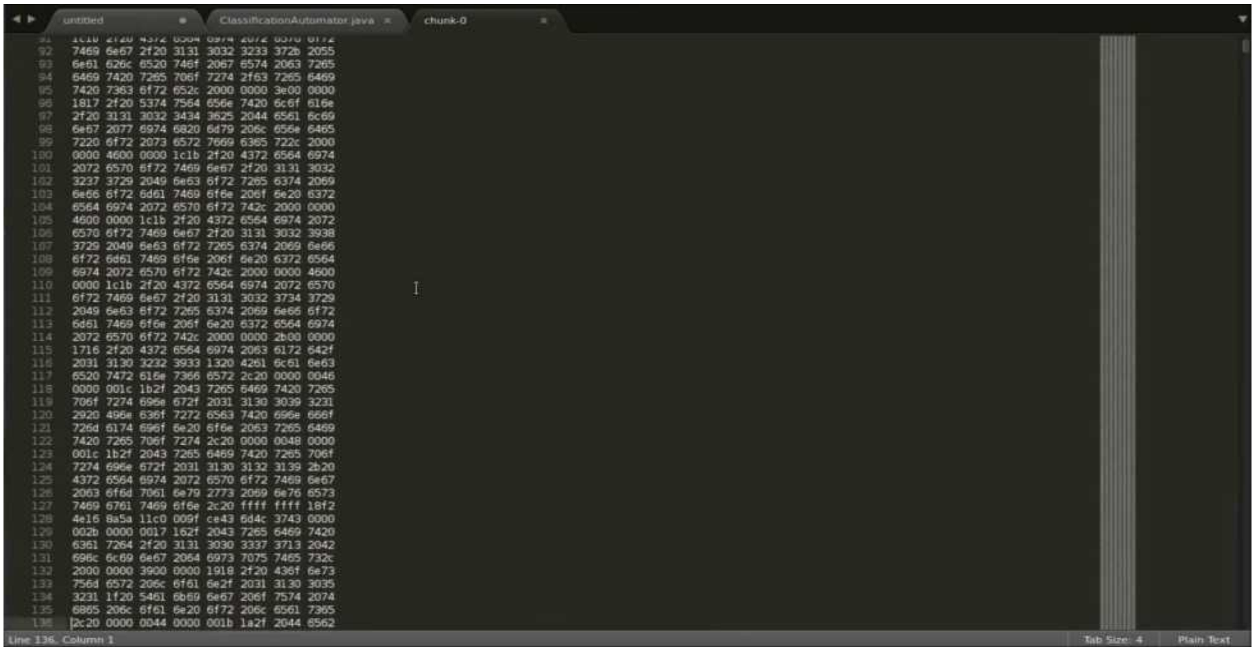
```
ComplaintsCSVtoSeq::main(): Wrote 973 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 797 entries to Seq File.
ComplaintsCSVtoSeq::main(): Wrote 3150 entries to Seq File.
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODES$ hdfs dfs -ls data/
14/12/22 00:24:08 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 3 items
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/banks
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:12 data/classification
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:12 data/states
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODES$ hdfs dfs -ls data/classification
14/12/22 00:24:16 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 62 items
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/AA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/AE
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/AK
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/AL
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/AP
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/AR
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/AS
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/AZ
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/CA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/CO
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/CT
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/DC
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/DE
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/FL
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/FM
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/GA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/GU
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/HI
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/IA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/ID
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/IL
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/IN
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/KS
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/KY
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/LA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MD
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/ME
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MH
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MI
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MN
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MO
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MP
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MS
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MT
```

- For example: Let us take New York state (NY) state sequence file which looks hexadecimally encoded and we will use this to generate vectors



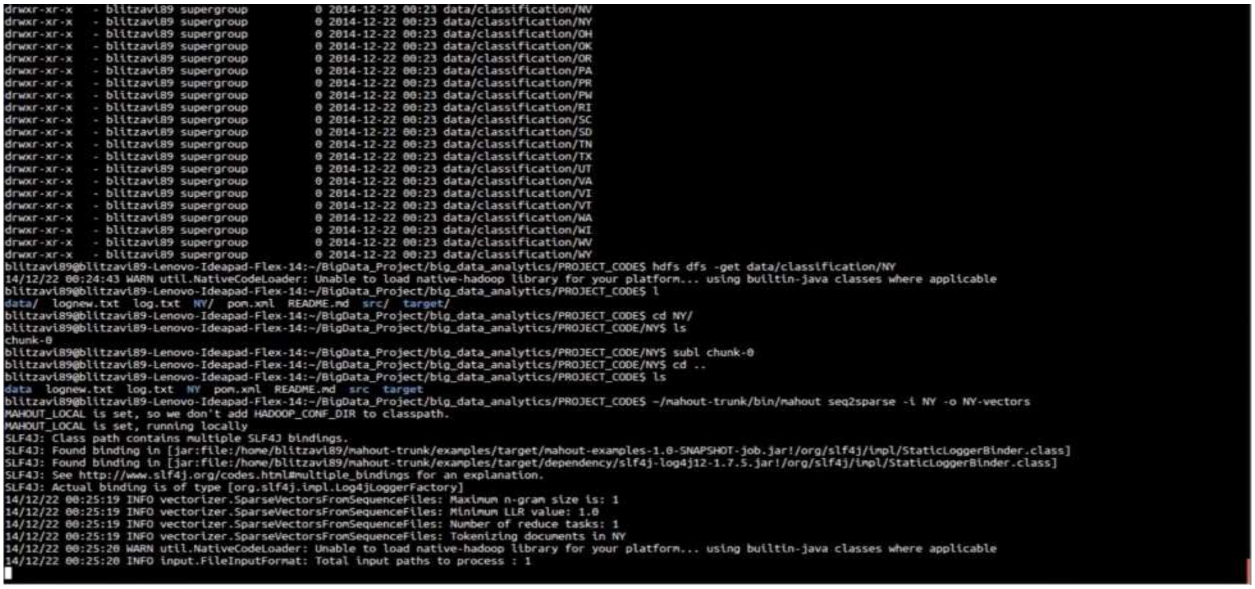
```
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/LA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MD
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/ME
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MH
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MI
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MN
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MO
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MP
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MS
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/MT
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/NC
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/ND
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/NE
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/NH
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/NJ
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/NM
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/NV
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/NY
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/OH
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/OK
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/OR
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/PA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/PR
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/PW
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/RI
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/SC
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/SD
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/TN
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/TX
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/UT
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/VA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/VI
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/VT
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/WA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/WI
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/WV
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/WY
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODES$ hdfs dfs -get data/classification/NY
14/12/22 00:24:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODES$ l
data/  lognew.txt  log.txt  NY/  pom.xml  README.md  src/  target/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODES$ cd NY/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/NY$ ls
chunk-0
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/NY$ subl chunk-0
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/NY$ 
```

- Hexadecimal encoding of the sequence file of the particular state such as New York (NY) as our example is as follows



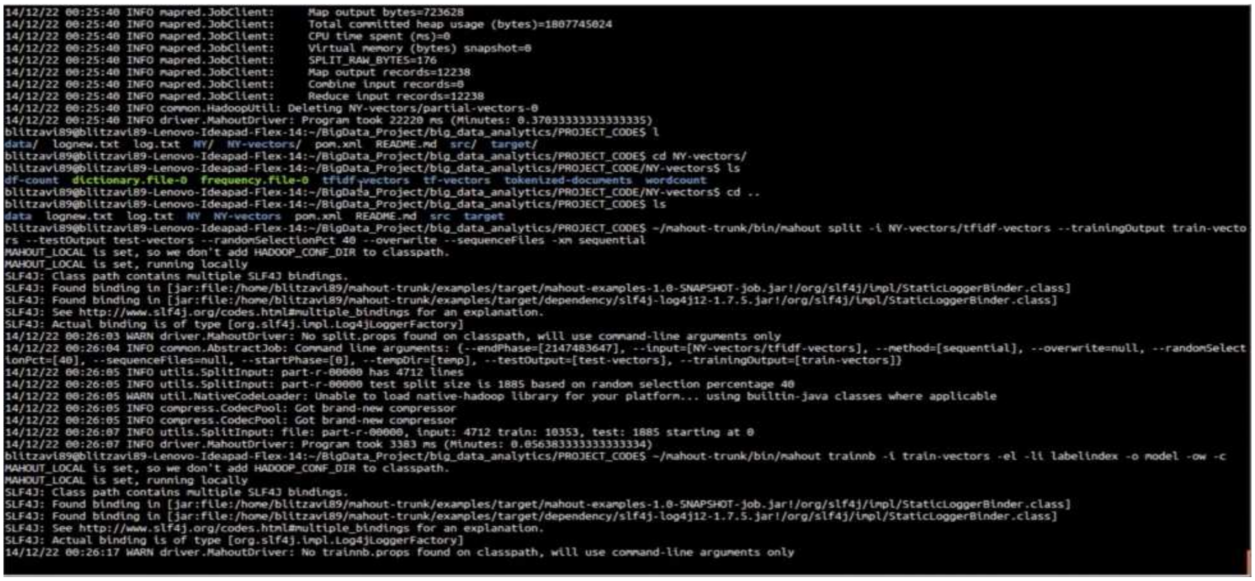
```
1c1b 2f20 4372 6564 6974 2072 6570 6f72
7469 6e67 2f20 3131 3032 3233 372b 2055
6e61 626c 6520 746f 2067 6574 2063 7265
6469 7420 7265 706f 7274 2f63 7265 6469
7420 7363 6f72 652c 2000 0000 3e00 0000
1817 2f20 5374 7564 656e 7420 6c6f 616e
2f20 3131 3032 3434 3625 2044 6561 6c69
6e67 2077 6974 6820 6d79 206c 656e 6465
7220 6f72 2073 6572 7669 6365 722c 2000
0000 4600 0000 1c1b 2f20 4372 6564 6974
2072 6570 6f72 7469 6e67 2f20 3131 3032
3237 3729 2049 6e63 6f72 7265 6374 2069
6e66 6f72 6d61 7469 6f6e 206f 6e20 6372
6564 6974 2072 6570 6f72 742c 2000 0000
4600 0000 1c1b 2f20 4372 6564 6974 2072
6570 6f72 7469 6e67 2f20 3131 3032 3938
3729 2049 6e63 6f72 7265 6374 2069 6e66
6f72 6d61 7469 6f6e 206f 6e20 6372 6564
6974 2072 6570 6f72 742c 2000 0000 4600
0000 1c1b 2f20 4372 6564 6974 2072 6570
6f72 7469 6e67 2f20 3131 3032 3734 3729
2049 6e63 6f72 7265 6374 2069 6e66 6f72
6d61 7469 6f6e 206f 6e20 6372 6564 6974
2072 6570 6f72 742c 2000 0000 2b00 0000
1716 2f20 4372 6564 6974 2063 6172 642f
2031 3130 3232 3933 1320 4261 6c61 6e63
6520 7472 616e 7366 6572 2c20 0000 0046
0000 001c 1b2f 2043 7265 6469 7420 7265
706f 7274 696e 672f 2031 3130 3039 3231
2920 496e 636f 7272 6563 7420 696e 666f
726d 6174 696f 6e20 6f6e 2063 7265 6469
7420 7265 706f 7274 2c20 0000 0048 0000
001c 1b2f 2043 7265 6469 7420 7265 706f
7274 696e 672f 2031 3130 3132 3139 2b20
4372 6564 6974 2072 6570 6f72 7469 6e67
2063 6f6d 7061 6e79 2773 2069 6e76 6573
7469 6761 7469 6f6e 2c20 ffff ffff 18f2
4e16 8a5a 11c0 009f ce43 6d4c 3743 0000
002b 0000 0017 162f 2043 7265 6469 7420
6361 7264 2f20 3131 3030 3337 3713 2042
696c 6c69 6e67 2064 6973 7075 7465 732c
2000 0000 3900 0000 1918 2f20 436f 6e73
756d 6572 206c 6f61 6e2f 2031 3130 3035
3231 1f20 5461 6b69 6e67 206f 7574 2074
6865 206c 6f61 6e20 6f72 206c 6561 7365
2c20 0000 0044 0000 001b 1a2f 2044 6562
```

- Now we need vectors so in Apache mahout now we will generate the vectors, also once the vectorization is generated we will use tfidf vectors to create testing and training datasets



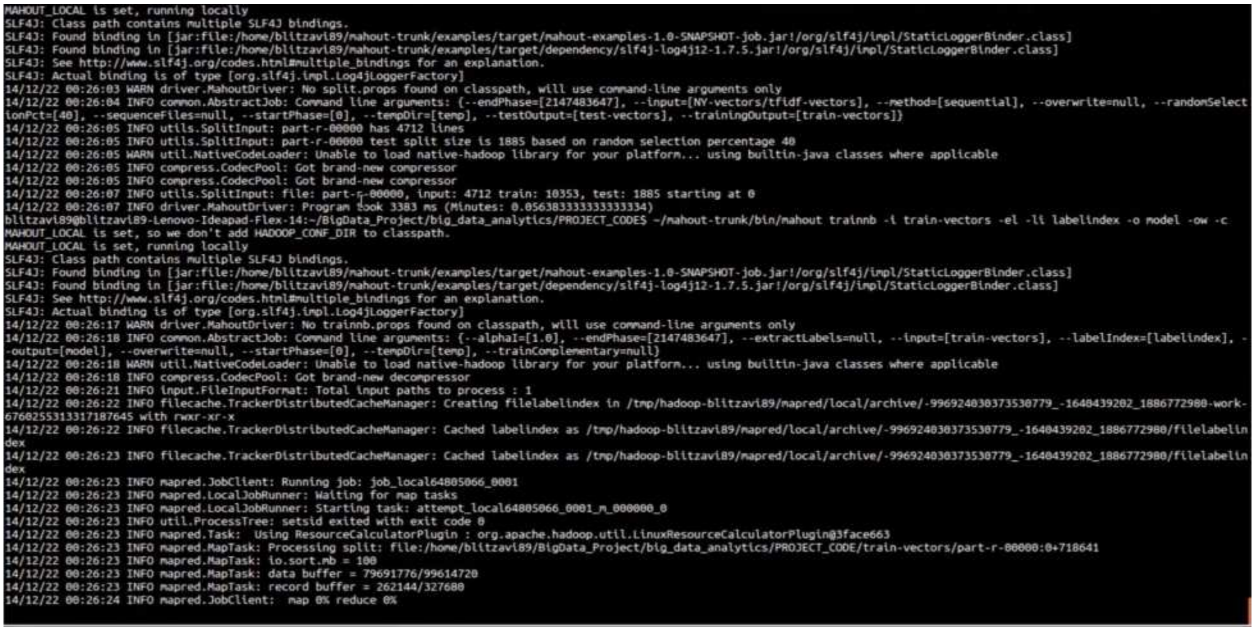
```
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/NV
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/NY
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/OH
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/OK
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/OR
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/PA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/PR
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/PW
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/RI
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/SC
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/SD
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/TN
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/TX
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/UT
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/VA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/VI
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/VT
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/WA
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/WI
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/WV
drwxr-xr-x   - blitzavi89 supergroup          0 2014-12-22 00:23 data/classification/WY
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ hdfs dfs -get data/classification/NY
14/12/22 00:24:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ l
data/  lognew.txt  log.txt  NY/  pom.xml  README.md  src/  target/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ cd NY/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/NY$ ls
chunk-0
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/NY$ subl chunk-0
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/NY$ cd ..
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ls
data  lognew.txt  log.txt  NY  pom.xml  README.md  src  target
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ~/mahout-trunk/bin/mahout seq2sparse -i NY -o NY-vectors
MAHOUT_LOCAL is set, so we don't add HADOOP_CONF_DIR to classpath.
MAHOUT_LOCAL is set, running locally
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/mahout-examples-1.0-SNAPSHOT-job.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/dependency/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
14/12/22 00:25:19 INFO vectorizer.SparseVectorsFromSequenceFiles: Maximum n-gram size is: 1
14/12/22 00:25:19 INFO vectorizer.SparseVectorsFromSequenceFiles: Minimum LLR value: 1.0
14/12/22 00:25:19 INFO vectorizer.SparseVectorsFromSequenceFiles: Number of reduce tasks: 1
14/12/22 00:25:19 INFO vectorizer.SparseVectorsFromSequenceFiles: Tokenizing documents in NY
14/12/22 00:25:20 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/12/22 00:25:20 INFO input.FileInputFormat: Total input paths to process : 1
```

- Basically we have to split the two vectors,datasets and test accordingly so the following command helps us achieving the Split, after the split is done we will do training of the dataset as follows



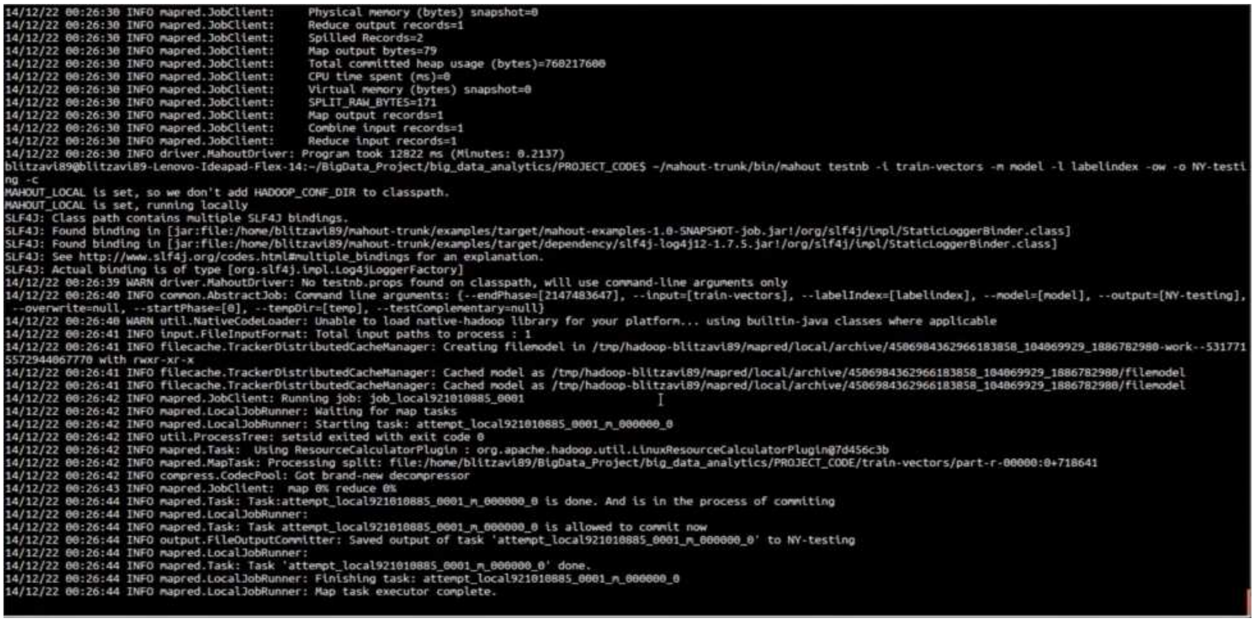
```
14/12/22 00:25:40 INFO mapred.JobClient:     Map output bytes=723628
14/12/22 00:25:40 INFO mapred.JobClient:     Total committed heap usage (bytes)=1807745024
14/12/22 00:25:40 INFO mapred.JobClient:     CPU time spent (ms)=0
14/12/22 00:25:40 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=0
14/12/22 00:25:40 INFO mapred.JobClient:     SPLIT_RAW_BYTES=176
14/12/22 00:25:40 INFO mapred.JobClient:     Map output records=12238
14/12/22 00:25:40 INFO mapred.JobClient:     Combine input records=0
14/12/22 00:25:40 INFO mapred.JobClient:     Reduce input records=12238
14/12/22 00:25:40 INFO common.HadoopUtil: Deleting NY-vectors/partial-vectors-0
14/12/22 00:25:40 INFO driver.MahoutDriver: Program took 22220 ms (Minutes: 0.37033333333333335)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ l
data/  lognew.txt  log.txt  NY/  NY-vectors/  pom.xml  README.md  src/  target/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ cd NY-vectors/
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/NY-vectors$ ls
df-count  dictionary.file-0  frequency.file-0  tfidf-vectors  tf-vectors  tokenized-documents  wordcount
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE/NY-vectors$ cd ..
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ls
data  lognew.txt  log.txt  NY  NY-vectors  pom.xml  README.md  src  target
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ~/mahout-trunk/bin/mahout split -i NY-vectors/tfidf-vectors --trainingOutput train-vecto
rs --testOutput test-vectors --randomSelectionPct 40 --overwrite --sequenceFiles -xm sequential
MAHOUT_LOCAL is set, so we don't add HADOOP_CONF_DIR to classpath.
MAHOUT_LOCAL is set, running locally
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/mahout-examples-1.0-SNAPSHOT-job.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/dependency/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
14/12/22 00:26:03 WARN driver.MahoutDriver: No split.props found on classpath, will use command-line arguments only
14/12/22 00:26:04 INFO common.AbstractJob: Command line arguments: {--endPhase=[2147483647], --input=[NY-vectors/tfidf-vectors], --method=[sequential], --overwrite=null, --randomSelect
ionPct=[40], --sequenceFiles=null, --startPhase=[0], --tempDir=[temp], --testOutput=[test-vectors], --trainingOutput=[train-vectors]}
14/12/22 00:26:05 INFO utils.SplitInput: part-r-00000 has 4712 lines
14/12/22 00:26:05 INFO utils.SplitInput: part-r-00000 test split size is 1885 based on random selection percentage 40
14/12/22 00:26:05 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/12/22 00:26:05 INFO compress.CodecPool: Got brand-new compressor
14/12/22 00:26:05 INFO compress.CodecPool: Got brand-new compressor
14/12/22 00:26:07 INFO utils.SplitInput: file: part-r-00000, input: 4712 train: 10353, test: 1885 starting at 0
14/12/22 00:26:07 INFO driver.MahoutDriver: Program took 3383 ms (Minutes: 0.05638333333333334)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ~/mahout-trunk/bin/mahout trainnb -i train-vectors -el -li labelindex -o model -ow -c
MAHOUT_LOCAL is set, so we don't add HADOOP_CONF_DIR to classpath.
MAHOUT_LOCAL is set, running locally
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/mahout-examples-1.0-SNAPSHOT-job.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/dependency/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
14/12/22 00:26:17 WARN driver.MahoutDriver: No trainnb.props found on classpath, will use command-line arguments only
```

- This is training the dataset with the use of vectors,training the module; We basically have to first train the model and secondly test it, So the following one is the training dataset



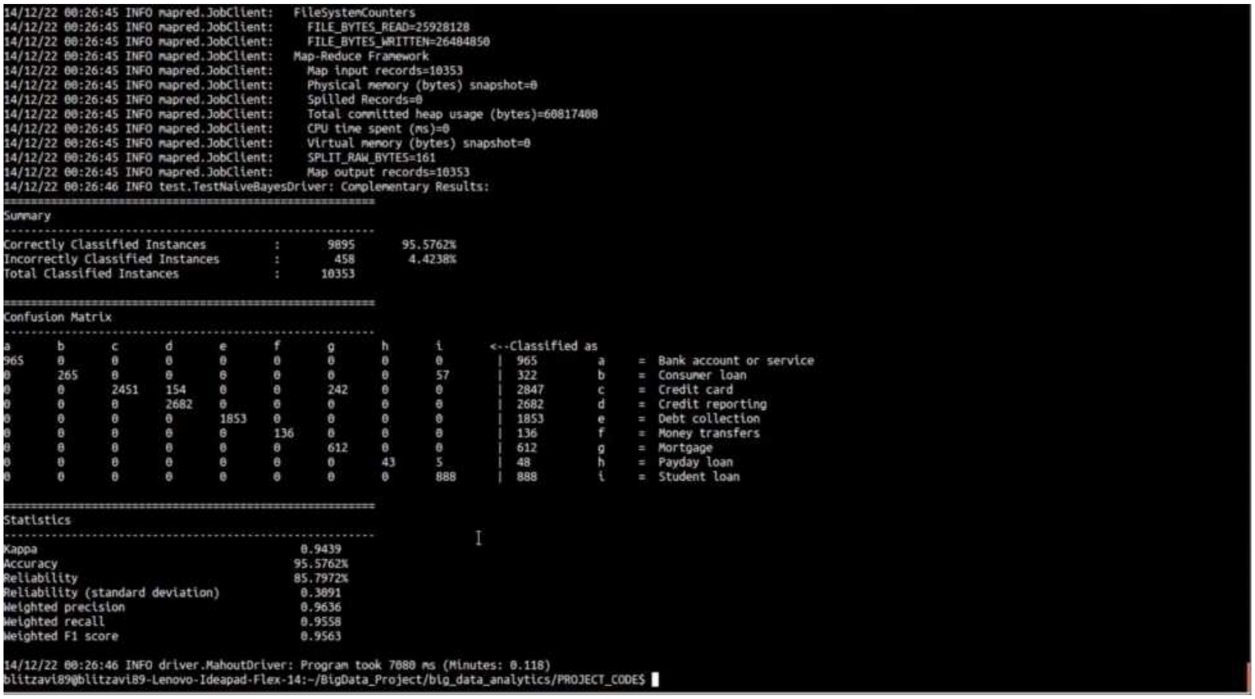
```
MAHOUT_LOCAL is set, running locally
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/mahout-examples-1.0-SNAPSHOT-job.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/dependency/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
14/12/22 00:26:03 WARN driver.MahoutDriver: No split.props found on classpath, will use command-line arguments only
14/12/22 00:26:04 INFO common.AbstractJob: Command line arguments: {--endPhase=[2147483647], --input=[NY-vectors/tfidf-vectors], --method=[sequential], --overwrite=null, --randomSelect
ionPct=[40], --sequenceFiles=null, --startPhase=[0], --tempDir=[temp], --testOutput=[test-vectors], --trainingOutput=[train-vectors]}
14/12/22 00:26:05 INFO utils.SplitInput: part-r-00000 has 4712 lines
14/12/22 00:26:05 INFO utils.SplitInput: part-r-00000 test split size is 1885 based on random selection percentage 40
14/12/22 00:26:05 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/12/22 00:26:05 INFO compress.CodecPool: Got brand-new compressor
14/12/22 00:26:05 INFO compress.CodecPool: Got brand-new compressor
14/12/22 00:26:07 INFO utils.SplitInput: file: part-r-00000, input: 4712 train: 10353, test: 1885 starting at 0
14/12/22 00:26:07 INFO driver.MahoutDriver: Program took 3383 ms (Minutes: 0.05638333333333334)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ~/mahout-trunk/bin/mahout trainnb -i train-vectors -el -li labelindex -o model -ow -c
MAHOUT_LOCAL is set, so we don't add HADOOP_CONF_DIR to classpath.
MAHOUT_LOCAL is set, running locally
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/mahout-examples-1.0-SNAPSHOT-job.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/dependency/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
14/12/22 00:26:17 WARN driver.MahoutDriver: No trainnb.props found on classpath, will use command-line arguments only
14/12/22 00:26:18 INFO common.AbstractJob: Command line arguments: {--alphaI=[1.0], --endPhase=[2147483647], --extractLabels=null, --input=[train-vectors], --labelIndex=[labelindex], -
-output=[model], --overwrite=null, --startPhase=[0], --tempDir=[temp], --trainComplementary=null}
14/12/22 00:26:18 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/12/22 00:26:18 INFO compress.CodecPool: Got brand-new decompressor
14/12/22 00:26:21 INFO input.FileInputFormat: Total input paths to process : 1
14/12/22 00:26:22 INFO filecache.TrackerDistributedCacheManager: Creating filelabelindex in /tmp/hadoop-blitzavi89/mapred/local/archive/-996924030373530779_-1640439202_1886772980-work-
6760255313317187645 with rwxr-xr-x
14/12/22 00:26:22 INFO filecache.TrackerDistributedCacheManager: Cached labelindex as /tmp/hadoop-blitzavi89/mapred/local/archive/-996924030373530779_-1640439202_1886772980/filelabelin
dex
14/12/22 00:26:23 INFO filecache.TrackerDistributedCacheManager: Cached labelindex as /tmp/hadoop-blitzavi89/mapred/local/archive/-996924030373530779_-1640439202_1886772980/filelabelin
dex
14/12/22 00:26:23 INFO mapred.JobClient: Running job: job_local64805066_0001
14/12/22 00:26:23 INFO mapred.LocalJobRunner: Waiting for map tasks
14/12/22 00:26:23 INFO mapred.LocalJobRunner: Starting task: attempt_local64805066_0001_m_000000_0
14/12/22 00:26:23 INFO util.ProcessTree: setsid exited with exit code 0
14/12/22 00:26:23 INFO mapred.Task:  Using ResourceCalculatorPlugin : org.apache.hadoop.util.LinuxResourceCalculatorPlugin@3face663
14/12/22 00:26:23 INFO mapred.MapTask: Processing split: file:/home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/train-vectors/part-r-00000:0+718641
14/12/22 00:26:23 INFO mapred.MapTask: io.sort.mb = 100
14/12/22 00:26:23 INFO mapred.MapTask: data buffer = 79691776/99614720
14/12/22 00:26:23 INFO mapred.MapTask: record buffer = 262144/327680
14/12/22 00:26:24 INFO mapred.JobClient:  map 0% reduce 0%
```

- The next step is to run the vectors and test the dataset which we have after running the training dataset, So now we will be testing the dataset as follows



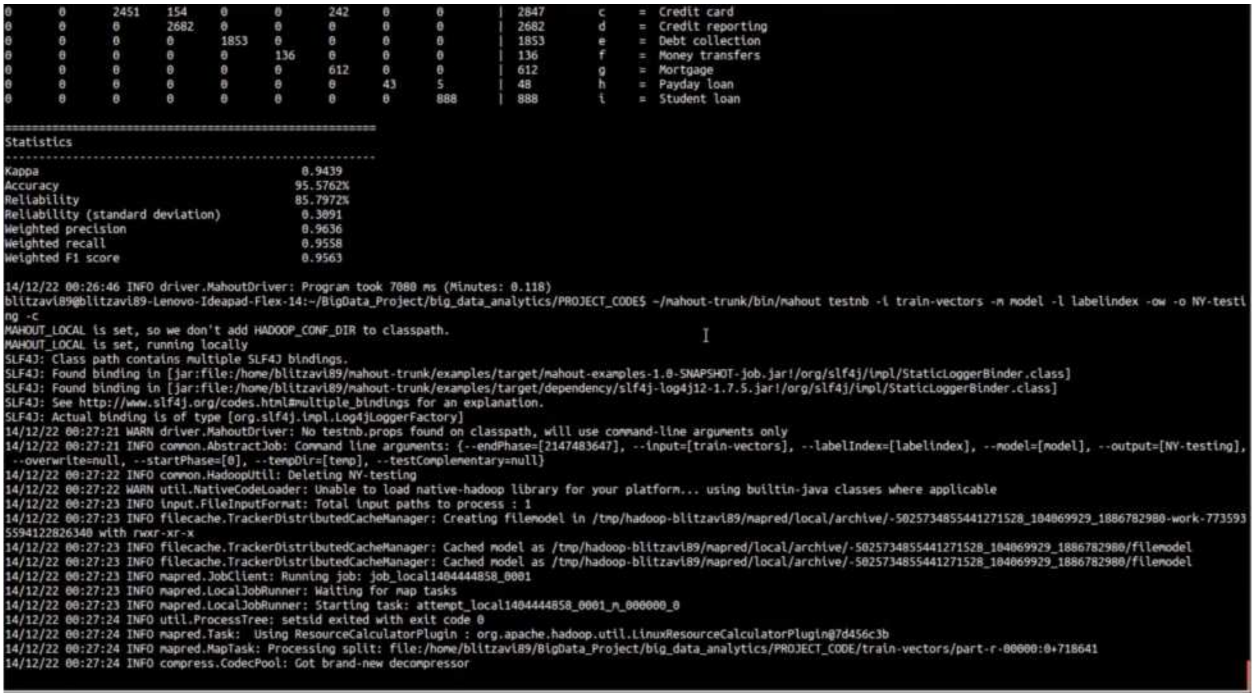
```
14/12/22 00:26:30 INFO mapred.JobClient:     Physical memory (bytes) snapshot=0
14/12/22 00:26:30 INFO mapred.JobClient:     Reduce output records=1
14/12/22 00:26:30 INFO mapred.JobClient:     Spilled Records=2
14/12/22 00:26:30 INFO mapred.JobClient:     Map output bytes=79
14/12/22 00:26:30 INFO mapred.JobClient:     Total committed heap usage (bytes)=760217600
14/12/22 00:26:30 INFO mapred.JobClient:     CPU time spent (ms)=0
14/12/22 00:26:30 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=0
14/12/22 00:26:30 INFO mapred.JobClient:     SPLIT_RAW_BYTES=171
14/12/22 00:26:30 INFO mapred.JobClient:     Map output records=1
14/12/22 00:26:30 INFO mapred.JobClient:     Combine input records=1
14/12/22 00:26:30 INFO mapred.JobClient:     Reduce input records=1
14/12/22 00:26:30 INFO driver.MahoutDriver: Program took 12822 ms (Minutes: 0.2137)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ~/mahout-trunk/bin/mahout testnb -i train-vectors -m model -l labelindex -ow -o NY-testi
ng -c
MAHOUT_LOCAL is set, so we don't add HADOOP_CONF_DIR to classpath.
MAHOUT_LOCAL is set, running locally
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/mahout-examples-1.0-SNAPSHOT-job.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/dependency/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
14/12/22 00:26:39 WARN driver.MahoutDriver: No testnb.props found on classpath, will use command-line arguments only
14/12/22 00:26:40 INFO common.AbstractJob: Command line arguments: {--endPhase=[2147483647], --input=[train-vectors], --labelIndex=[labelindex], --model=[model], --output=[NY-testing],
 --overwrite=null, --startPhase=[0], --tempDir=[temp], --testComplementary=null}
14/12/22 00:26:40 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/12/22 00:26:41 INFO input.FileInputFormat: Total input paths to process : 1
14/12/22 00:26:41 INFO filecache.TrackerDistributedCacheManager: Creating filemodel in /tmp/hadoop-blitzavi89/mapred/local/archive/4506984362966183858_104069929_1886782980-work--531771
5572944067770 with rwxr-xr-x
14/12/22 00:26:41 INFO filecache.TrackerDistributedCacheManager: Cached model as /tmp/hadoop-blitzavi89/mapred/local/archive/4506984362966183858_104069929_1886782980/filemodel
14/12/22 00:26:41 INFO filecache.TrackerDistributedCacheManager: Cached model as /tmp/hadoop-blitzavi89/mapred/local/archive/4506984362966183858_104069929_1886782980/filemodel
14/12/22 00:26:42 INFO mapred.JobClient: Running job: job_local921010885_0001
14/12/22 00:26:42 INFO mapred.LocalJobRunner: Waiting for map tasks
14/12/22 00:26:42 INFO mapred.LocalJobRunner: Starting task: attempt_local921010885_0001_m_000000_0
14/12/22 00:26:42 INFO util.ProcessTree: setsid exited with exit code 0
14/12/22 00:26:42 INFO mapred.Task:  Using ResourceCalculatorPlugin : org.apache.hadoop.util.LinuxResourceCalculatorPlugin@7d456c3b
14/12/22 00:26:42 INFO mapred.MapTask: Processing split: file:/home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/train-vectors/part-r-00000:0+718641
14/12/22 00:26:42 INFO compress.CodecPool: Got brand-new decompressor
14/12/22 00:26:43 INFO mapred.JobClient:  map 0% reduce 0%
14/12/22 00:26:44 INFO mapred.Task: Task:attempt_local921010885_0001_m_000000_0 is done. And is in the process of commiting
14/12/22 00:26:44 INFO mapred.LocalJobRunner:
14/12/22 00:26:44 INFO mapred.Task: Task attempt_local921010885_0001_m_000000_0 is allowed to commit now
14/12/22 00:26:44 INFO output.FileOutputCommitter: Saved output of task 'attempt_local921010885_0001_m_000000_0' to NY-testing
14/12/22 00:26:44 INFO mapred.LocalJobRunner:
14/12/22 00:26:44 INFO mapred.Task: Task 'attempt_local921010885_0001_m_000000_0' done.
14/12/22 00:26:44 INFO mapred.LocalJobRunner: Finishing task: attempt_local921010885_0001_m_000000_0
14/12/22 00:26:44 INFO mapred.LocalJobRunner: Map task executor complete.
```

- After training now we will generate confusion matrix by doing a naive bayes classification on the train set as well as the test dataset which goes as follows



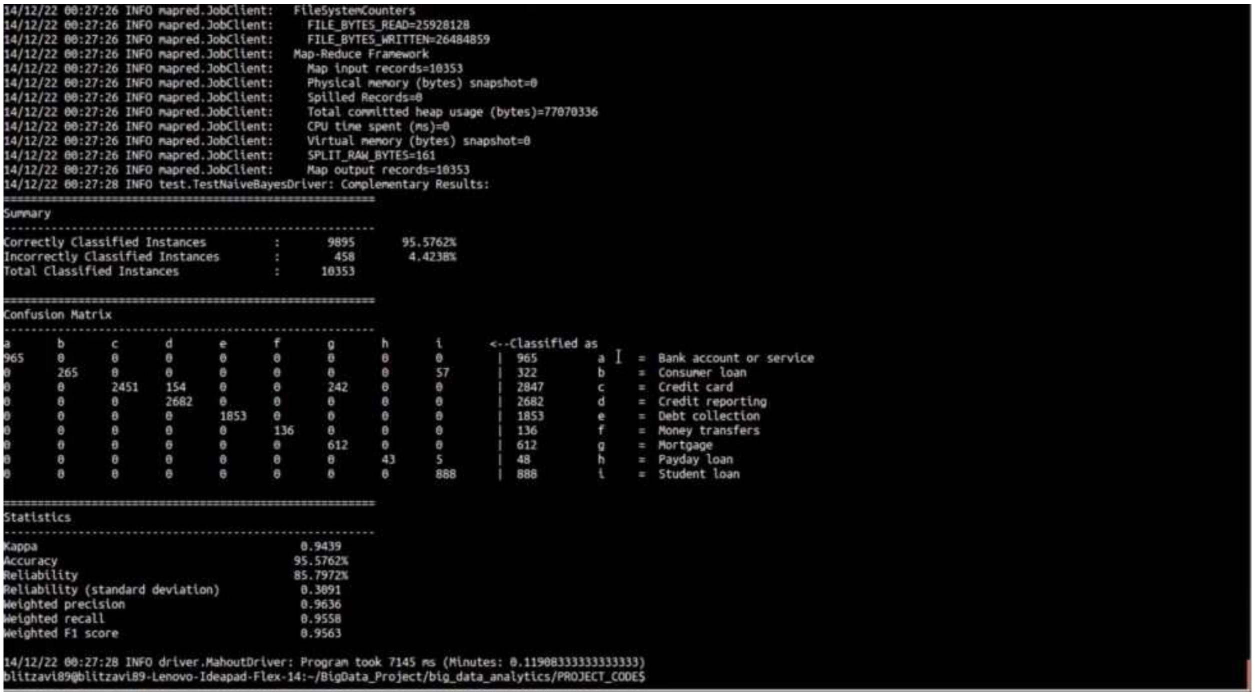
```
14/12/22 00:26:45 INFO mapred.JobClient:   FileSystemCounters
14/12/22 00:26:45 INFO mapred.JobClient:     FILE_BYTES_READ=25928128
14/12/22 00:26:45 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=26484850
14/12/22 00:26:45 INFO mapred.JobClient:   Map-Reduce Framework
14/12/22 00:26:45 INFO mapred.JobClient:     Map input records=10353
14/12/22 00:26:45 INFO mapred.JobClient:     Physical memory (bytes) snapshot=0
14/12/22 00:26:45 INFO mapred.JobClient:     Spilled Records=0
14/12/22 00:26:45 INFO mapred.JobClient:     Total committed heap usage (bytes)=60817408
14/12/22 00:26:45 INFO mapred.JobClient:     CPU time spent (ms)=0
14/12/22 00:26:45 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=0
14/12/22 00:26:45 INFO mapred.JobClient:     SPLIT_RAW_BYTES=161
14/12/22 00:26:45 INFO mapred.JobClient:     Map output records=10353
14/12/22 00:26:46 INFO test.TestNaiveBayesDriver: Complementary Results:
=======================================================
Summary
-------------------------------------------------------
Correctly Classified Instances          :       9895       95.5762%
Incorrectly Classified Instances        :        458        4.4238%
Total Classified Instances              :      10353

=======================================================
Confusion Matrix
-------------------------------------------------------
a       b       c       d       e       f       g       h       i       <--Classified as
965     0       0       0       0       0       0       0       0        |  965        a     =  Bank account or service
0       265     0       0       0       0       0       0       57       |  322        b     =  Consumer loan
0       0       2451    154     0       0       242     0       0        |  2847       c     =  Credit card
0       0       0       2682    0       0       0       0       0        |  2682       d     =  Credit reporting
0       0       0       0       1853    0       0       0       0        |  1853       e     =  Debt collection
0       0       0       0       0       136     0       0       0        |  136        f     =  Money transfers
0       0       0       0       0       0       612     0       0        |  612        g     =  Mortgage
0       0       0       0       0       0       0       43      5        |  48         h     =  Payday loan
0       0       0       0       0       0       0       0       888      |  888        i     =  Student loan

=======================================================
Statistics
-------------------------------------------------------
Kappa                                       0.9439
Accuracy                                   95.5762%
Reliability                                85.7972%
Reliability (standard deviation)            0.3091
Weighted precision                          0.9636
Weighted recall                             0.9558
Weighted F1 score                           0.9563

14/12/22 00:26:46 INFO driver.MahoutDriver: Program took 7080 ms (Minutes: 0.118)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ 
```

- Using Apache Mahout we will do the classification which is also called as the Naive Bayes classification , In Naive bayes classification we will train the training dataset, which usually gives good results and also provides a higher accuracy of around 95 to 100% but once we run it on the testing dataset it gives clearly low accuracy between 85 to 90% which is expected and measured.



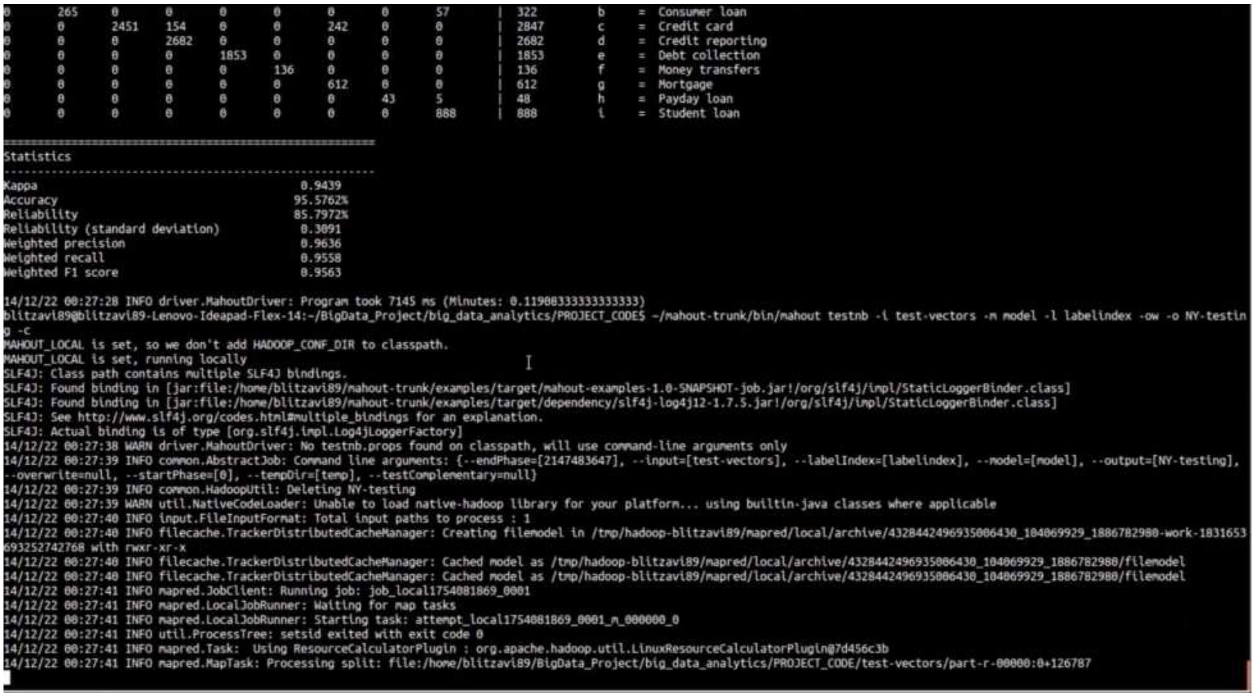
```
0       0       2451    154     0       0       242     0       0        |  2847       c     =  Credit card
0       0       0       2682    0       0       0       0       0        |  2682       d     =  Credit reporting
0       0       0       0       1853    0       0       0       0        |  1853       e     =  Debt collection
0       0       0       0       0       136     0       0       0        |  136        f     =  Money transfers
0       0       0       0       0       0       612     0       0        |  612        g     =  Mortgage
0       0       0       0       0       0       0       43      5        |  48         h     =  Payday loan
0       0       0       0       0       0       0       0       888      |  888        i     =  Student loan

=======================================================
Statistics
-------------------------------------------------------
Kappa                                       0.9439
Accuracy                                   95.5762%
Reliability                                85.7972%
Reliability (standard deviation)            0.3091
Weighted precision                          0.9636
Weighted recall                             0.9558
Weighted F1 score                           0.9563

14/12/22 00:26:46 INFO driver.MahoutDriver: Program took 7080 ms (Minutes: 0.118)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ~/mahout-trunk/bin/mahout testnb -l train-vectors -m model -l labelindex -ow -o NY-testi
ng -c
MAHOUT_LOCAL is set, so we don't add HADOOP_CONF_DIR to classpath.
MAHOUT_LOCAL is set, running locally
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/mahout-examples-1.0-SNAPSHOT-job.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/dependency/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
14/12/22 00:27:21 WARN driver.MahoutDriver: No testnb.props found on classpath, will use command-line arguments only
14/12/22 00:27:21 INFO common.AbstractJob: Command line arguments: {--endPhase=[2147483647], --input=[train-vectors], --labelIndex=[labelindex], --model=[model], --output=[NY-testing],
 --overwrite=null, --startPhase=[0], --tempDir=[temp], --testComplementary=null}
14/12/22 00:27:22 INFO common.HadoopUtil: Deleting NY-testing
14/12/22 00:27:22 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/12/22 00:27:23 INFO input.FileInputFormat: Total input paths to process : 1
14/12/22 00:27:23 INFO filecache.TrackerDistributedCacheManager: Creating filemodel in /tmp/hadoop-blitzavi89/mapred/local/archive/-5025734855441271528_104069929_1886782980-work-773593
5594122826340 with rwxr-xr-x
14/12/22 00:27:23 INFO filecache.TrackerDistributedCacheManager: Cached model as /tmp/hadoop-blitzavi89/mapred/local/archive/-5025734855441271528_104069929_1886782980/filemodel
14/12/22 00:27:23 INFO filecache.TrackerDistributedCacheManager: Cached model as /tmp/hadoop-blitzavi89/mapred/local/archive/-5025734855441271528_104069929_1886782980/filemodel
14/12/22 00:27:23 INFO mapred.JobClient: Running job: job_local1404444858_0001
14/12/22 00:27:23 INFO mapred.LocalJobRunner: Waiting for map tasks
14/12/22 00:27:23 INFO mapred.LocalJobRunner: Starting task: attempt_local1404444858_0001_m_000000_0
14/12/22 00:27:24 INFO util.ProcessTree: setsid exited with exit code 0
14/12/22 00:27:24 INFO mapred.Task:  Using ResourceCalculatorPlugin : org.apache.hadoop.util.LinuxResourceCalculatorPlugin@7d456c3b
14/12/22 00:27:24 INFO mapred.MapTask: Processing split: file:/home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/train-vectors/part-r-00000:0+718641
14/12/22 00:27:24 INFO compress.CodecPool: Got brand-new decompressor
```

- Now we will get to the interesting part of our module which is the Confusion matrix , Confusion matrix actually has classifiers such as credit card, debit card, mortgage, loan, etc which are basically some of the functions of the banking industry and has a lot of complaints regarding it. So basically it simplifies and classifies and we can see those trends by the classifiers it has for ex: complaints related to credit card, loan,etc with respect to our project and input.



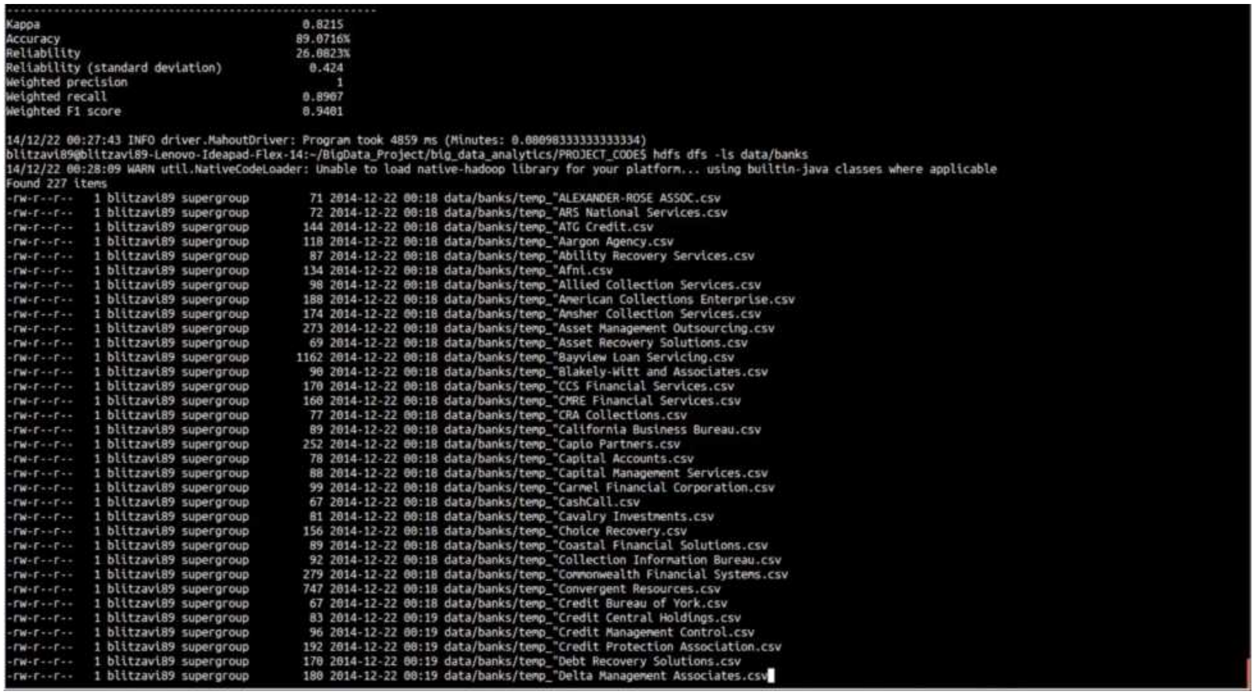
```
14/12/22 00:27:26 INFO mapred.JobClient:   FileSystemCounters
14/12/22 00:27:26 INFO mapred.JobClient:     FILE_BYTES_READ=25928128
14/12/22 00:27:26 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=26484859
14/12/22 00:27:26 INFO mapred.JobClient:   Map-Reduce Framework
14/12/22 00:27:26 INFO mapred.JobClient:     Map input records=10353
14/12/22 00:27:26 INFO mapred.JobClient:     Physical memory (bytes) snapshot=0
14/12/22 00:27:26 INFO mapred.JobClient:     Spilled Records=0
14/12/22 00:27:26 INFO mapred.JobClient:     Total committed heap usage (bytes)=77070336
14/12/22 00:27:26 INFO mapred.JobClient:     CPU time spent (ms)=0
14/12/22 00:27:26 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=0
14/12/22 00:27:26 INFO mapred.JobClient:     SPLIT_RAW_BYTES=161
14/12/22 00:27:26 INFO mapred.JobClient:     Map output records=10353
14/12/22 00:27:28 INFO test.TestNaiveBayesDriver: Complementary Results:
=======================================================
Summary
-------------------------------------------------------
Correctly Classified Instances          :       9895       95.5762%
Incorrectly Classified Instances        :        458        4.4238%
Total Classified Instances              :      10353

=======================================================
Confusion Matrix
-------------------------------------------------------
a       b       c       d       e       f       g       h       i       <--Classified as
965     0       0       0       0       0       0       0       0        |  965         a     =  Bank account or service
0       265     0       0       0       0       0       0       57       |  322         b     =  Consumer loan
0       0       2451    154     0       0       242     0       0        |  2847        c     =  Credit card
0       0       0       2682    0       0       0       0       0        |  2682        d     =  Credit reporting
0       0       0       0       1853    0       0       0       0        |  1853        e     =  Debt collection
0       0       0       0       0       136     0       0       0        |  136         f     =  Money transfers
0       0       0       0       0       0       612     0       0        |  612         g     =  Mortgage
0       0       0       0       0       0       0       43      5        |  48          h     =  Payday loan
0       0       0       0       0       0       0       0       888      |  888         i     =  Student loan

=======================================================
Statistics
-------------------------------------------------------
Kappa                                       0.9439
Accuracy                                   95.5762%
Reliability                                85.7972%
Reliability (standard deviation)            0.3091
Weighted precision                          0.9636
Weighted recall                             0.9558
Weighted F1 score                           0.9563

14/12/22 00:27:28 INFO driver.MahoutDriver: Program took 7145 ms (Minutes: 0.11908333333333333)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$
```

- Now we will repeat the analysis on the testing dataset to validate the results for our module and we can see the confusion matrix for our classification performed on the testing dataset.



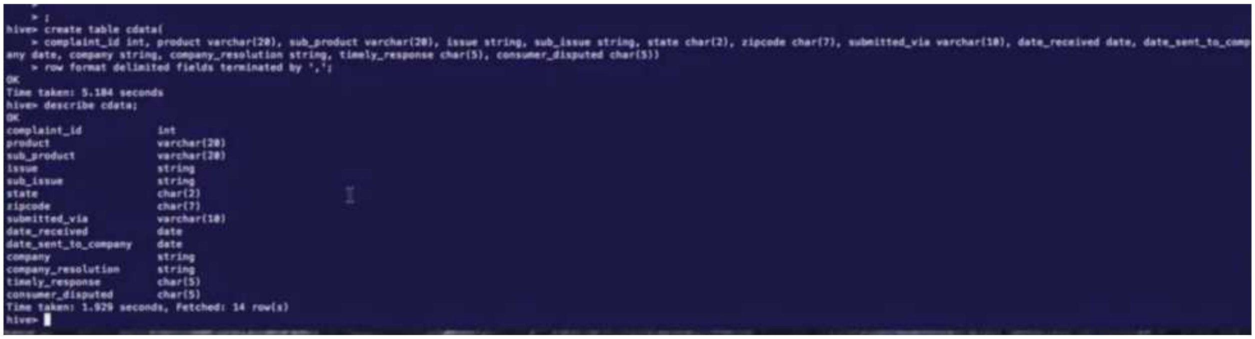
```
0       265     0       0       0       0       0       0       57       |  322         b     =  Consumer loan
0       0       2451    154     0       0       242     0       0        |  2847        c     =  Credit card
0       0       0       2682    0       0       0       0       0        |  2682        d     =  Credit reporting
0       0       0       0       1853    0       0       0       0        |  1853        e     =  Debt collection
0       0       0       0       0       136     0       0       0        |  136         f     =  Money transfers
0       0       0       0       0       0       612     0       0        |  612         g     =  Mortgage
0       0       0       0       0       0       0       43      5        |  48          h     =  Payday loan
0       0       0       0       0       0       0       0       888      |  888         i     =  Student loan

=======================================================
Statistics
-------------------------------------------------------
Kappa                                       0.9439
Accuracy                                   95.5762%
Reliability                                85.7972%
Reliability (standard deviation)            0.3091
Weighted precision                          0.9636
Weighted recall                             0.9558
Weighted F1 score                           0.9563

14/12/22 00:27:28 INFO driver.MahoutDriver: Program took 7145 ms (Minutes: 0.11908333333333333)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ ~/mahout-trunk/bin/mahout testnb -i test-vectors -m model -l labelindex -ow -o NY-testin
g -c
MAHOUT_LOCAL is set, so we don't add HADOOP_CONF_DIR to classpath.
MAHOUT_LOCAL is set, running locally
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/mahout-examples-1.0-SNAPSHOT-job.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/blitzavi89/mahout-trunk/examples/target/dependency/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
14/12/22 00:27:38 WARN driver.MahoutDriver: No testnb.props found on classpath, will use command-line arguments only
14/12/22 00:27:39 INFO common.AbstractJob: Command line arguments: {--endPhase=[2147483647], --input=[test-vectors], --labelIndex=[labelindex], --model=[model], --output=[NY-testing],
--overwrite=null, --startPhase=[0], --tempDir=[temp], --testComplementary=null}
14/12/22 00:27:39 INFO common.HadoopUtil: Deleting NY-testing
14/12/22 00:27:39 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/12/22 00:27:40 INFO input.FileInputFormat: Total input paths to process : 1
14/12/22 00:27:40 INFO filecache.TrackerDistributedCacheManager: Creating filemodel in /tmp/hadoop-blitzavi89/mapred/local/archive/4328442496935006430_104069929_1886782980-work-1831653
693252742768 with rwxr-xr-x
14/12/22 00:27:40 INFO filecache.TrackerDistributedCacheManager: Cached model as /tmp/hadoop-blitzavi89/mapred/local/archive/4328442496935006430_104069929_1886782980/filemodel
14/12/22 00:27:40 INFO filecache.TrackerDistributedCacheManager: Cached model as /tmp/hadoop-blitzavi89/mapred/local/archive/4328442496935006430_104069929_1886782980/filemodel
14/12/22 00:27:41 INFO mapred.JobClient: Running job: job_local1754081869_0001
14/12/22 00:27:41 INFO mapred.LocalJobRunner: Waiting for map tasks
14/12/22 00:27:41 INFO mapred.LocalJobRunner: Starting task: attempt_local1754081869_0001_m_000000_0
14/12/22 00:27:41 INFO util.ProcessTree: setsid exited with exit code 0
14/12/22 00:27:41 INFO mapred.Task:  Using ResourceCalculatorPlugin : org.apache.hadoop.util.LinuxResourceCalculatorPlugin@7d456c3b
14/12/22 00:27:41 INFO mapred.MapTask: Processing split: file:/home/blitzavi89/BigData_Project/big_data_analytics/PROJECT_CODE/test-vectors/part-r-00000:0+126787
```

- Similarly we can do the classification for banks which measures priorities for different products such as credit card, loans, etc. So banks can use these results in near future in order to calculate the priority and importance for tackling specific issues raised by customers. The classification for banks is shown below for example



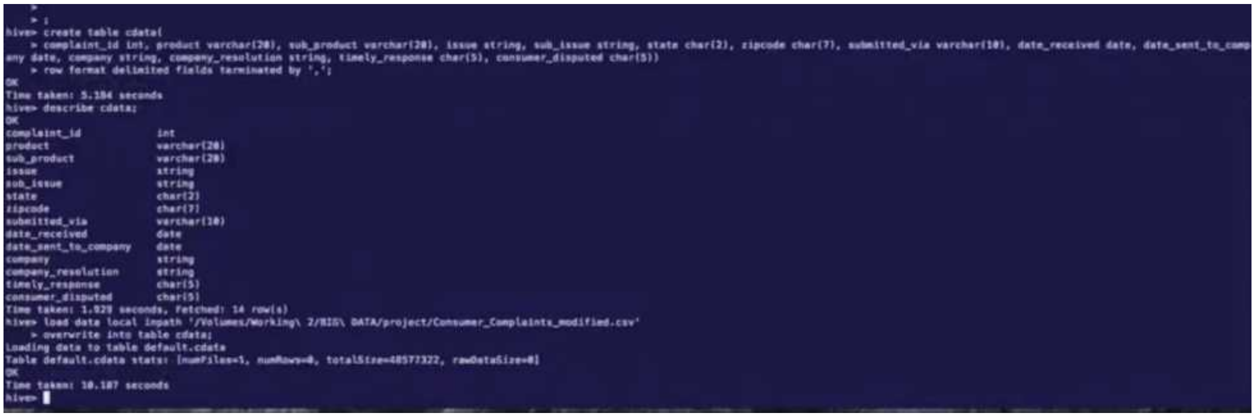
```
-------------------------------------------------------
Kappa                                       0.8215
Accuracy                                   89.0716%
Reliability                                26.0823%
Reliability (standard deviation)            0.424
Weighted precision                              1
Weighted recall                             0.8907
Weighted F1 score                           0.9401

14/12/22 00:27:43 INFO driver.MahoutDriver: Program took 4859 ms (Minutes: 0.08098333333333334)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$ hdfs dfs -ls data/banks
14/12/22 00:28:09 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 227 items
-rw-r--r--   1 blitzavi89 supergroup         71 2014-12-22 00:18 data/banks/temp_"ALEXANDER-ROSE ASSOC.csv
-rw-r--r--   1 blitzavi89 supergroup         72 2014-12-22 00:18 data/banks/temp_"ARS National Services.csv
-rw-r--r--   1 blitzavi89 supergroup        144 2014-12-22 00:18 data/banks/temp_"ATG Credit.csv
-rw-r--r--   1 blitzavi89 supergroup        118 2014-12-22 00:18 data/banks/temp_"Aargon Agency.csv
-rw-r--r--   1 blitzavi89 supergroup         87 2014-12-22 00:18 data/banks/temp_"Ability Recovery Services.csv
-rw-r--r--   1 blitzavi89 supergroup        134 2014-12-22 00:18 data/banks/temp_"Afni.csv
-rw-r--r--   1 blitzavi89 supergroup         98 2014-12-22 00:18 data/banks/temp_"Allied Collection Services.csv
-rw-r--r--   1 blitzavi89 supergroup        188 2014-12-22 00:18 data/banks/temp_"American Collections Enterprise.csv
-rw-r--r--   1 blitzavi89 supergroup        174 2014-12-22 00:18 data/banks/temp_"Amsher Collection Services.csv
-rw-r--r--   1 blitzavi89 supergroup        273 2014-12-22 00:18 data/banks/temp_"Asset Management Outsourcing.csv
-rw-r--r--   1 blitzavi89 supergroup         69 2014-12-22 00:18 data/banks/temp_"Asset Recovery Solutions.csv
-rw-r--r--   1 blitzavi89 supergroup       1162 2014-12-22 00:18 data/banks/temp_"Bayview Loan Servicing.csv
-rw-r--r--   1 blitzavi89 supergroup         90 2014-12-22 00:18 data/banks/temp_"Blakely-Witt and Associates.csv
-rw-r--r--   1 blitzavi89 supergroup        170 2014-12-22 00:18 data/banks/temp_"CCS Financial Services.csv
-rw-r--r--   1 blitzavi89 supergroup        160 2014-12-22 00:18 data/banks/temp_"CMRE Financial Services.csv
-rw-r--r--   1 blitzavi89 supergroup         77 2014-12-22 00:18 data/banks/temp_"CRA Collections.csv
-rw-r--r--   1 blitzavi89 supergroup         89 2014-12-22 00:18 data/banks/temp_"California Business Bureau.csv
-rw-r--r--   1 blitzavi89 supergroup        252 2014-12-22 00:18 data/banks/temp_"Caplo Partners.csv
-rw-r--r--   1 blitzavi89 supergroup         78 2014-12-22 00:18 data/banks/temp_"Capital Accounts.csv
-rw-r--r--   1 blitzavi89 supergroup         88 2014-12-22 00:18 data/banks/temp_"Capital Management Services.csv
-rw-r--r--   1 blitzavi89 supergroup         99 2014-12-22 00:18 data/banks/temp_"Carmel Financial Corporation.csv
-rw-r--r--   1 blitzavi89 supergroup         67 2014-12-22 00:18 data/banks/temp_"CashCall.csv
-rw-r--r--   1 blitzavi89 supergroup         81 2014-12-22 00:18 data/banks/temp_"Cavalry Investments.csv
-rw-r--r--   1 blitzavi89 supergroup        156 2014-12-22 00:18 data/banks/temp_"Choice Recovery.csv
-rw-r--r--   1 blitzavi89 supergroup         89 2014-12-22 00:18 data/banks/temp_"Coastal Financial Solutions.csv
-rw-r--r--   1 blitzavi89 supergroup         92 2014-12-22 00:18 data/banks/temp_"Collection Information Bureau.csv
-rw-r--r--   1 blitzavi89 supergroup        279 2014-12-22 00:18 data/banks/temp_"Commonwealth Financial Systems.csv
-rw-r--r--   1 blitzavi89 supergroup        747 2014-12-22 00:18 data/banks/temp_"Convergent Resources.csv
-rw-r--r--   1 blitzavi89 supergroup         67 2014-12-22 00:18 data/banks/temp_"Credit Bureau of York.csv
-rw-r--r--   1 blitzavi89 supergroup         83 2014-12-22 00:19 data/banks/temp_"Credit Central Holdings.csv
-rw-r--r--   1 blitzavi89 supergroup         96 2014-12-22 00:19 data/banks/temp_"Credit Management Control.csv
-rw-r--r--   1 blitzavi89 supergroup        192 2014-12-22 00:19 data/banks/temp_"Credit Protection Association.csv
-rw-r--r--   1 blitzavi89 supergroup        170 2014-12-22 00:19 data/banks/temp_"Debt Recovery Solutions.csv
-rw-r--r--   1 blitzavi89 supergroup        180 2014-12-22 00:19 data/banks/temp_"Delta Management Associates.csv
```

- Apache hive was used in order to extract meaningful information from a given dataset in hive, But first we need to create an empty table Cdata for complaint data and fire describe tablename query which gives schema of the table as below in hive



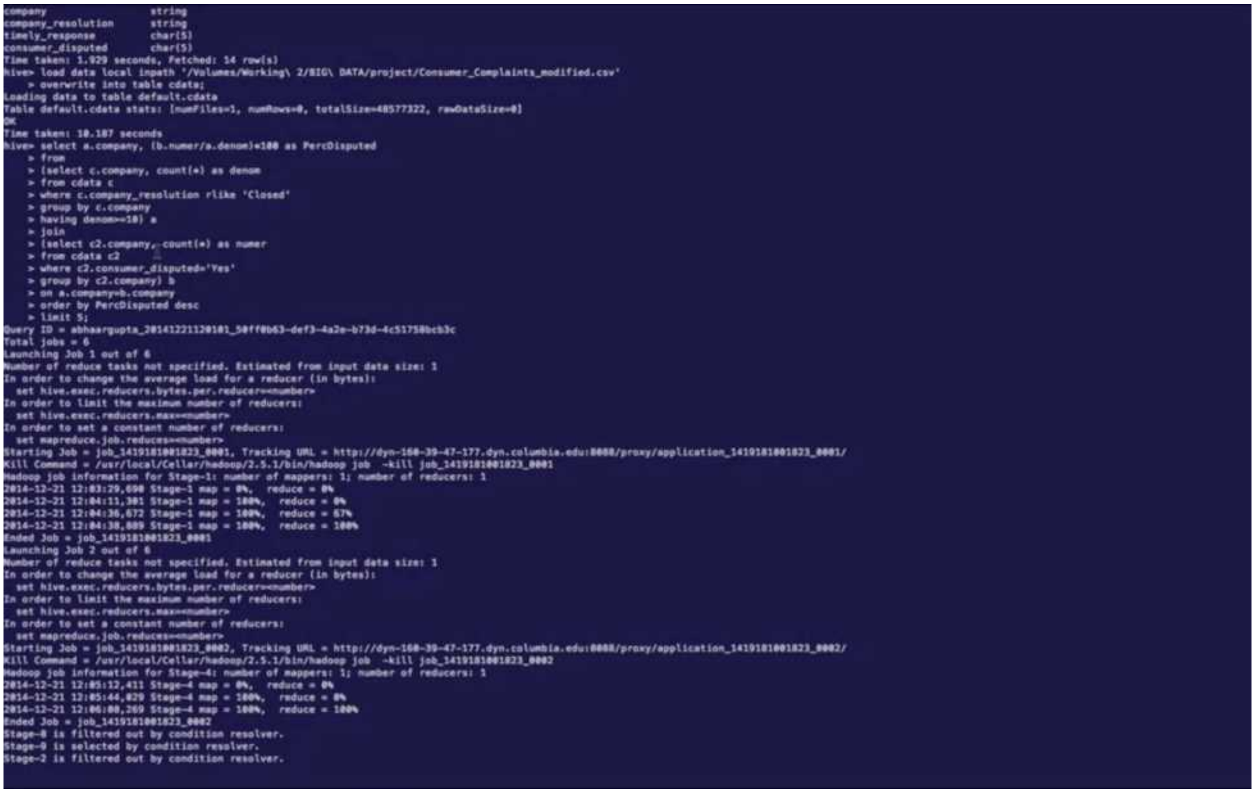
```
    > ;
hive> create table cdata(
    > complaint_id int, product varchar(20), sub_product varchar(20), issue string, sub_issue string, state char(2), zipcode char(7), submitted_via varchar(10), date_received date, date_sent_to_comp
any date, company string, company_resolution string, timely_response char(5), consumer_disputed char(5))
    > row format delimited fields terminated by ',';
OK
Time taken: 5.184 seconds
hive> describe cdata;
OK
complaint_id            int
product                 varchar(20)
sub_product             varchar(20)
issue                   string
sub_issue               string
state                   char(2)
zipcode                 char(7)
submitted_via           varchar(10)
date_received           date
date_sent_to_company    date
company                 string
company_resolution      string
timely_response         char(5)
consumer_disputed       char(5)
Time taken: 1.929 seconds, Fetched: 14 row(s)
hive> 
```

- Now lets insert data into the table, as data is stored locally in a CSV file we will simply instruct hive to copy that local file into the hive table



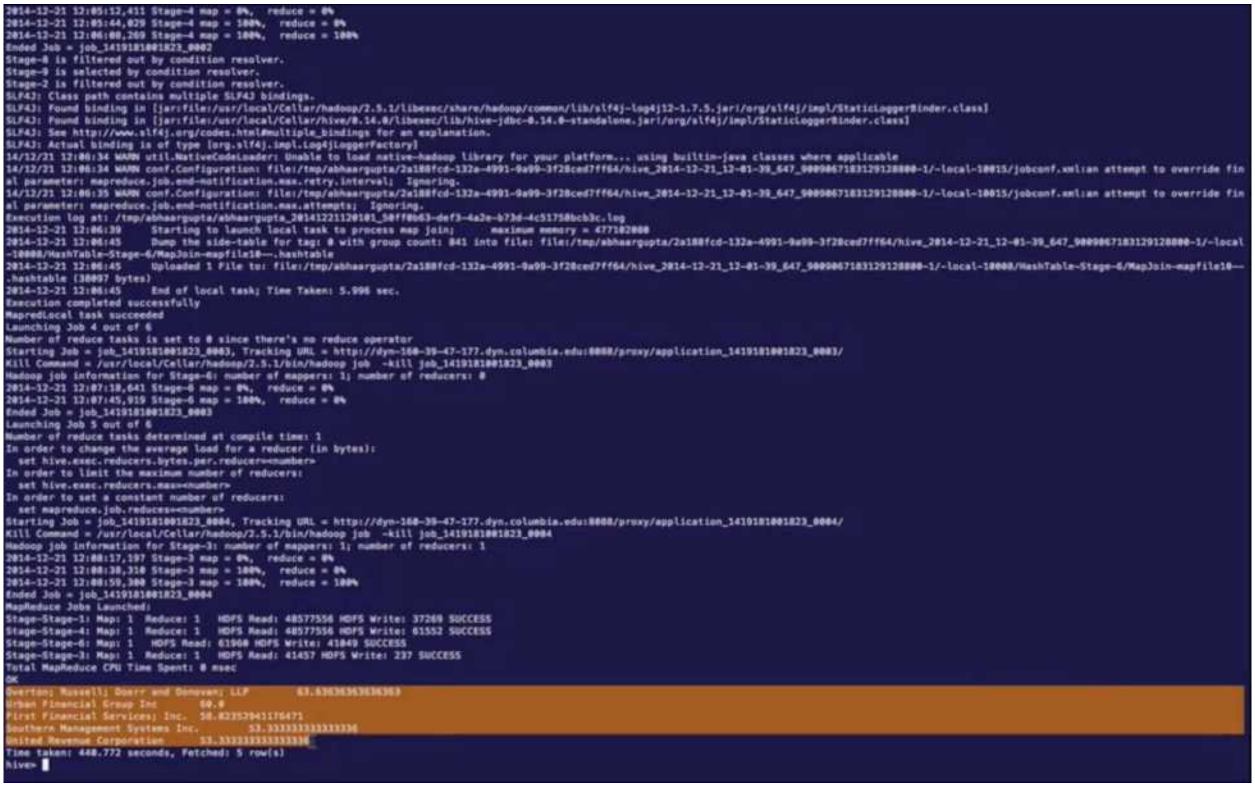
```
    >
    > ;
hive> create table cdata(
    > complaint_id int, product varchar(20), sub_product varchar(20), issue string, sub_issue string, state char(2), zipcode char(7), submitted_via varchar(10), date_received date, date_sent_to_comp
any date, company string, company_resolution string, timely_response char(5), consumer_disputed char(5))
    > row format delimited fields terminated by ',';
OK
Time taken: 5.184 seconds
hive> describe cdata;
OK
complaint_id            int
product                 varchar(20)
sub_product             varchar(20)
issue                   string
sub_issue               string
state                   char(2)
zipcode                 char(7)
submitted_via           varchar(10)
date_received           date
date_sent_to_company    date
company                 string
company_resolution      string
timely_response         char(5)
consumer_disputed       char(5)
Time taken: 1.929 seconds, Fetched: 14 row(s)
hive> load data local inpath '/Volumes/Working\ 2/BIG\ DATA/project/Consumer_Complaints_modified.csv'
    > overwrite into table cdata;
Loading data to table default.cdata
Table default.cdata stats: [numFiles=1, numRows=0, totalSize=48577322, rawDataSize=0]
OK
Time taken: 10.187 seconds
hive>
```

- Now the table is ready to accept queries so query the table with a query which asks top 5 companies who have highest percentage of closed complaints by customers and have at least 10 closed complaints we have also exported the data in a csv file for future purposes



```
company                 string
company_resolution      string
timely_response         char(5)
consumer_disputed       char(5)
Time taken: 1.929 seconds, Fetched: 14 row(s)
hive> load data local inpath '/Volumes/Working\ 2/BIG\ DATA/project/Consumer_Complaints_modified.csv'
    > overwrite into table cdata;
Loading data to table default.cdata
Table default.cdata stats: [numFiles=1, numRows=0, totalSize=48577322, rawDataSize=0]
OK
Time taken: 10.187 seconds
hive> select a.company, (b.numer/a.denom)*100 as PercDisputed
    > from
    > (select c.company, count(*) as denom
    > from cdata c
    > where c.company_resolution rlike 'Closed'
    > group by c.company
    > having denom>=10) a
    > join
    > (select c2.company, count(*) as numer
    > from cdata c2
    > where c2.consumer_disputed='Yes'
    > group by c2.company) b
    > on a.company=b.company
    > order by PercDisputed desc
    > limit 5;
Query ID = abhaargupta_20141221120101_50ff0b63-def3-4a2e-b73d-4c51750bcb3c
Total jobs = 6
Launching Job 1 out of 6
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1419181001823_0001, Tracking URL = http://dyn-160-39-47-177.dyn.columbia.edu:8088/proxy/application_1419181001823_0001/
Kill Command = /usr/local/Cellar/hadoop/2.5.1/bin/hadoop job  -kill job_1419181001823_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2014-12-21 12:03:29,690 Stage-1 map = 0%,  reduce = 0%
2014-12-21 12:04:11,301 Stage-1 map = 100%,  reduce = 0%
2014-12-21 12:04:36,672 Stage-1 map = 100%,  reduce = 67%
2014-12-21 12:04:38,809 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_1419181001823_0001
Launching Job 2 out of 6
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1419181001823_0002, Tracking URL = http://dyn-160-39-47-177.dyn.columbia.edu:8088/proxy/application_1419181001823_0002/
Kill Command = /usr/local/Cellar/hadoop/2.5.1/bin/hadoop job  -kill job_1419181001823_0002
Hadoop job information for Stage-4: number of mappers: 1; number of reducers: 1
2014-12-21 12:05:12,411 Stage-4 map = 0%,  reduce = 0%
2014-12-21 12:05:44,029 Stage-4 map = 100%,  reduce = 0%
2014-12-21 12:06:00,269 Stage-4 map = 100%,  reduce = 100%
Ended Job = job_1419181001823_0002
Stage-8 is filtered out by condition resolver.
Stage-9 is selected by condition resolver.
Stage-2 is filtered out by condition resolver.
```

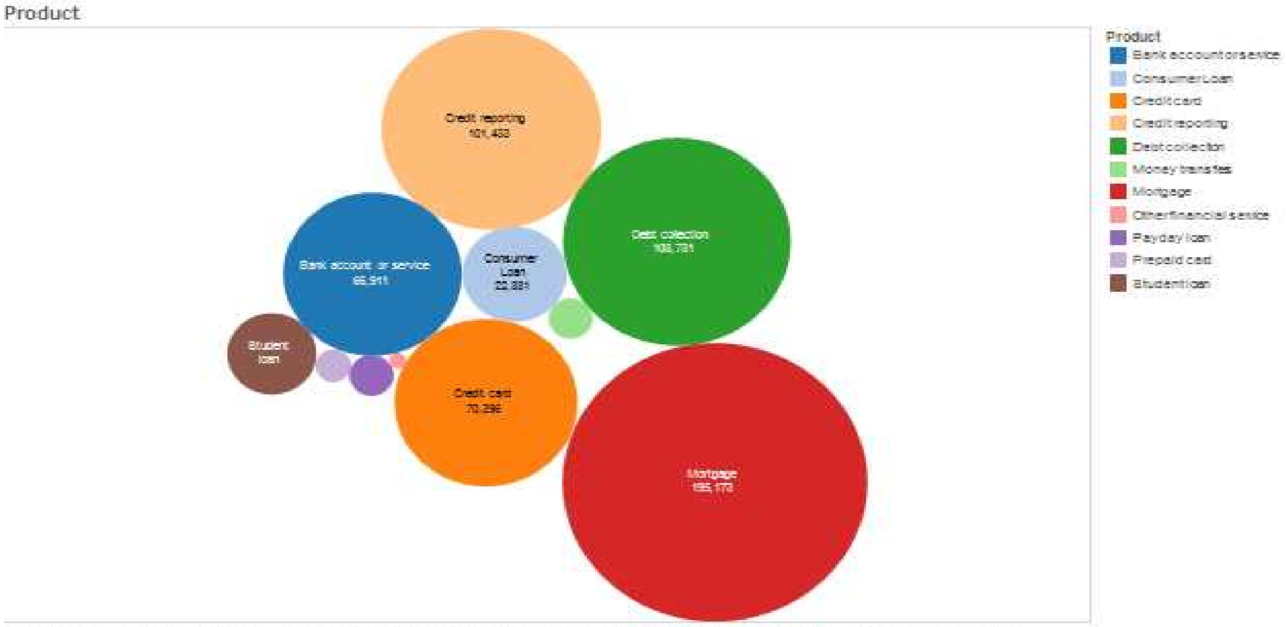
- The working of the above query goes as the sub query A gives the company name and total number of complaints closed by the company after that we join the table with the result of sub query B which gives company name and number of closed complaints disputed by the customer dividing the result of second with the result of the first we get the percentage of the closed complaints that were disputed ,Results here gives out top 5 companies which have maximum percentage of customers dis-satisfied by their solutions. These results thus are very important from a company perspective as well because it helps them to judge their performance.



```
2014-12-21 12:05:12,411 Stage-4 map = 0%,  reduce = 0%
2014-12-21 12:05:44,029 Stage-4 map = 100%,  reduce = 0%
2014-12-21 12:06:00,269 Stage-4 map = 100%,  reduce = 100%
Ended Job = job_1419181001823_0002
Stage-8 is filtered out by condition resolver.
Stage-9 is selected by condition resolver.
Stage-2 is filtered out by condition resolver.
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/Cellar/hadoop/2.5.1/libexec/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/Cellar/hive/0.14.0/libexec/lib/hive-jdbc-0.14.0-standalone.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
14/12/21 12:06:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
14/12/21 12:06:34 WARN conf.Configuration: file:/tmp/abhaargupta/2a180fcd-132a-4991-9a99-3f20ced7ff64/hive_2014-12-21_12-01-39_647_9009067183129128000-1/-local-10015/jobconf.xml:an attempt to override fin
al parameter: mapreduce.job.end-notification.max.retry.interval;  Ignoring.
14/12/21 12:06:35 WARN conf.Configuration: file:/tmp/abhaargupta/2a180fcd-132a-4991-9a99-3f20ced7ff64/hive_2014-12-21_12-01-39_647_9009067183129128000-1/-local-10015/jobconf.xml:an attempt to override fin
al parameter: mapreduce.job.end-notification.max.attempts;  Ignoring.
Execution log at: /tmp/abhaargupta/abhaargupta_20141221120101_50ff0b63-def3-4a2e-b73d-4c51750bcb3c.log
2014-12-21 12:06:39     Starting to launch local task to process map join;      maximum memory = 477102080
2014-12-21 12:06:45     Dump the side-table for tag: 0 with group count: 841 into file: file:/tmp/abhaargupta/2a180fcd-132a-4991-9a99-3f20ced7ff64/hive_2014-12-21_12-01-39_647_9009067183129128000-1/-local
-10008/HashTable-Stage-6/MapJoin-mapfile10--.hashtable
2014-12-21 12:06:45     Uploaded 1 File to: file:/tmp/abhaargupta/2a180fcd-132a-4991-9a99-3f20ced7ff64/hive_2014-12-21_12-01-39_647_9009067183129128000-1/-local-10008/HashTable-Stage-6/MapJoin-mapfile10--
.hashtable (38097 bytes)
2014-12-21 12:06:45     End of local task; Time Taken: 5.996 sec.
Execution completed successfully
MapredLocal task succeeded
Launching Job 4 out of 6
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1419181001823_0003, Tracking URL = http://dyn-160-39-47-177.dyn.columbia.edu:8088/proxy/application_1419181001823_0003/
Kill Command = /usr/local/Cellar/hadoop/2.5.1/bin/hadoop job  -kill job_1419181001823_0003
Hadoop job information for Stage-6: number of mappers: 1; number of reducers: 0
2014-12-21 12:07:18,641 Stage-6 map = 0%,  reduce = 0%
2014-12-21 12:07:45,919 Stage-6 map = 100%,  reduce = 0%
Ended Job = job_1419181001823_0003
Launching Job 5 out of 6
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1419181001823_0004, Tracking URL = http://dyn-160-39-47-177.dyn.columbia.edu:8088/proxy/application_1419181001823_0004/
Kill Command = /usr/local/Cellar/hadoop/2.5.1/bin/hadoop job  -kill job_1419181001823_0004
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 1
2014-12-21 12:08:17,197 Stage-3 map = 0%,  reduce = 0%
2014-12-21 12:08:38,318 Stage-3 map = 100%,  reduce = 0%
2014-12-21 12:08:59,300 Stage-3 map = 100%,  reduce = 100%
Ended Job = job_1419181001823_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   HDFS Read: 48577556 HDFS Write: 37269 SUCCESS
Stage-Stage-4: Map: 1  Reduce: 1   HDFS Read: 48577556 HDFS Write: 61552 SUCCESS
Stage-Stage-6: Map: 1   HDFS Read: 61960 HDFS Write: 41049 SUCCESS
Stage-Stage-3: Map: 1  Reduce: 1   HDFS Read: 41457 HDFS Write: 237 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Overton; Russell; Doerr and Donovan; LLP        63.63636363636363
Urban Financial Group Inc       60.0
First Financial Services; Inc.  58.82352941176471
Southern Management Systems Inc.        53.333333333333336
United Revenue Corporation      53.333333333333336
Time taken: 440.772 seconds, Fetched: 5 row(s)
hive>
```

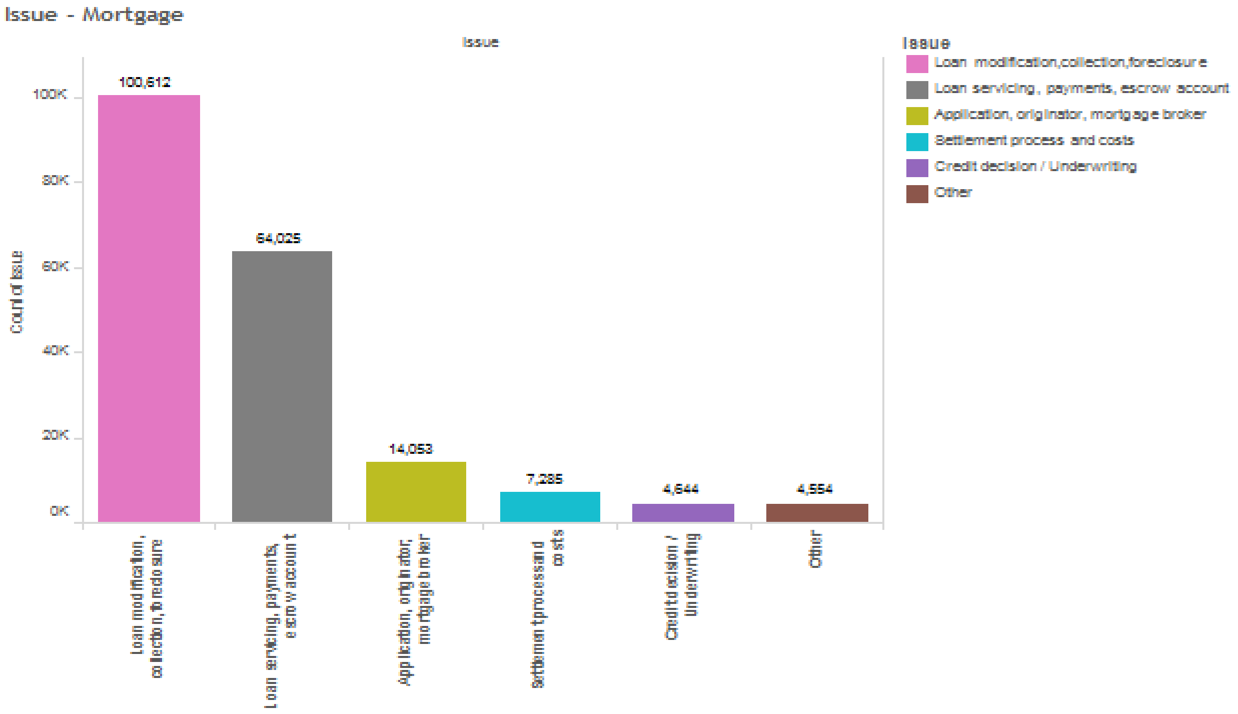
- Some additional analysis done via tableau software which gives insights in a more representative manner

- Product-wise analysis



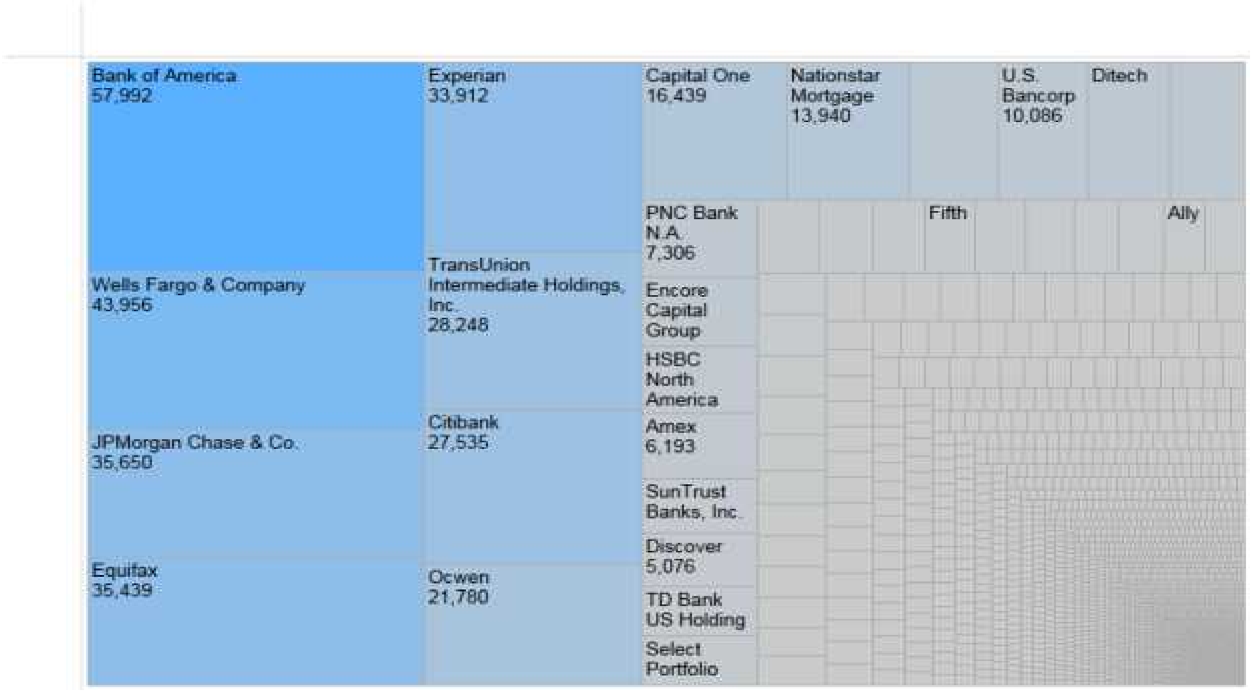
Product and count of Product. Color shows details about Product. Size shows count of Product. The marks are labeled by Product and count of Product.

- Mortgage-issue analysis

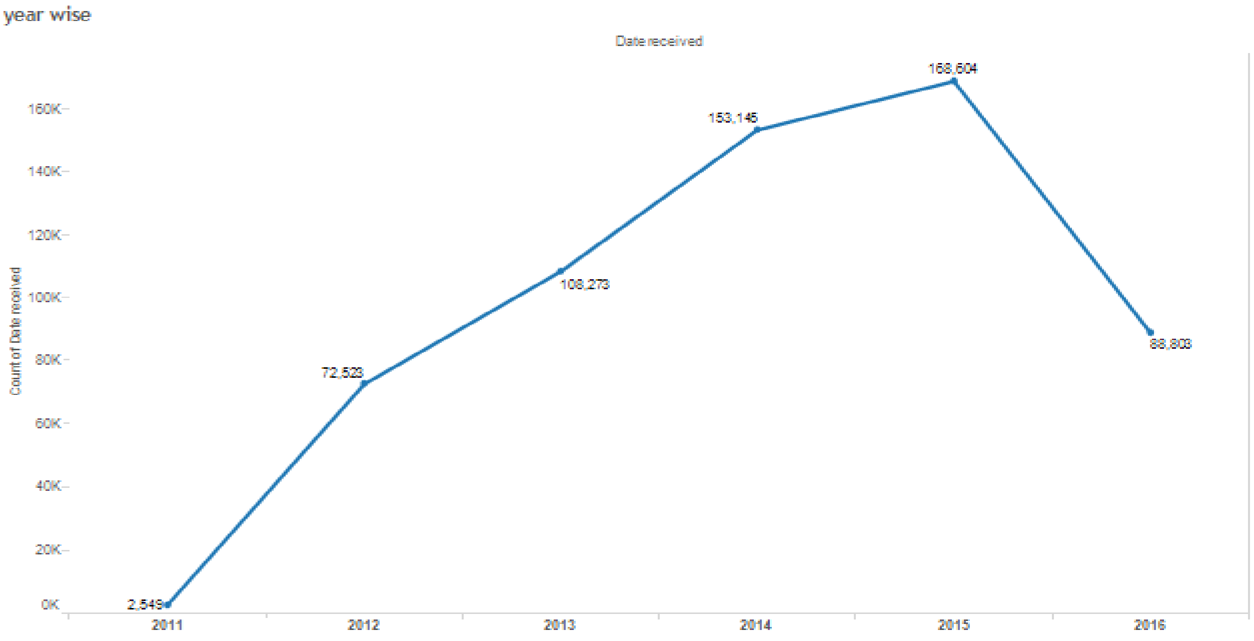


Count of Issue for each Issue. Color shows details about Issue. The data is filtered on Product, which keeps Mortgage.

- Company wise analysis



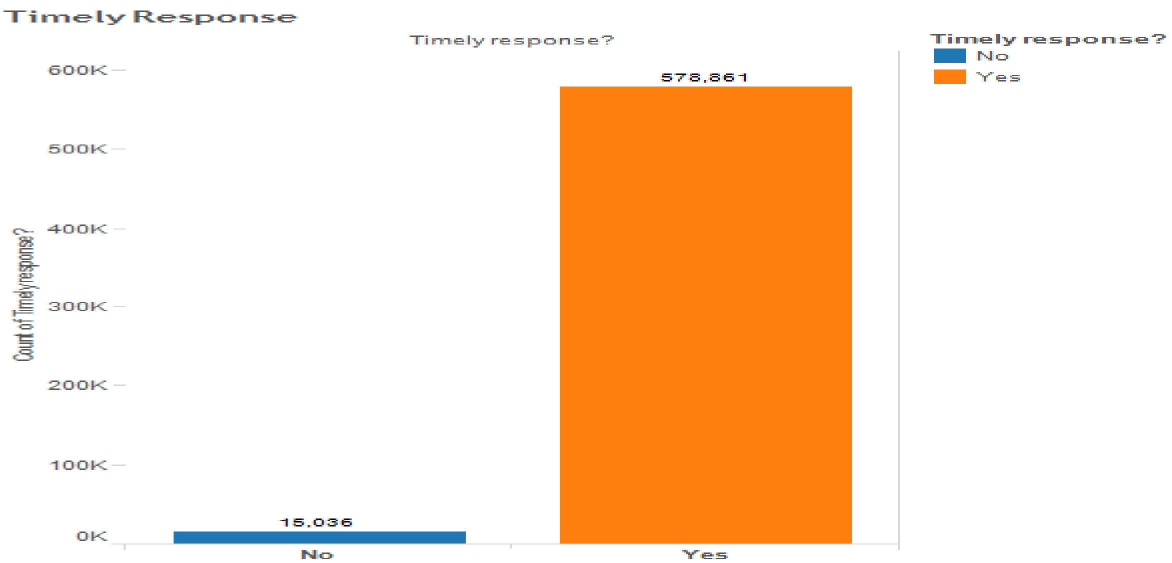
- Year wise analysis



The trend of count of Date received for Date received Year.

- Timely response

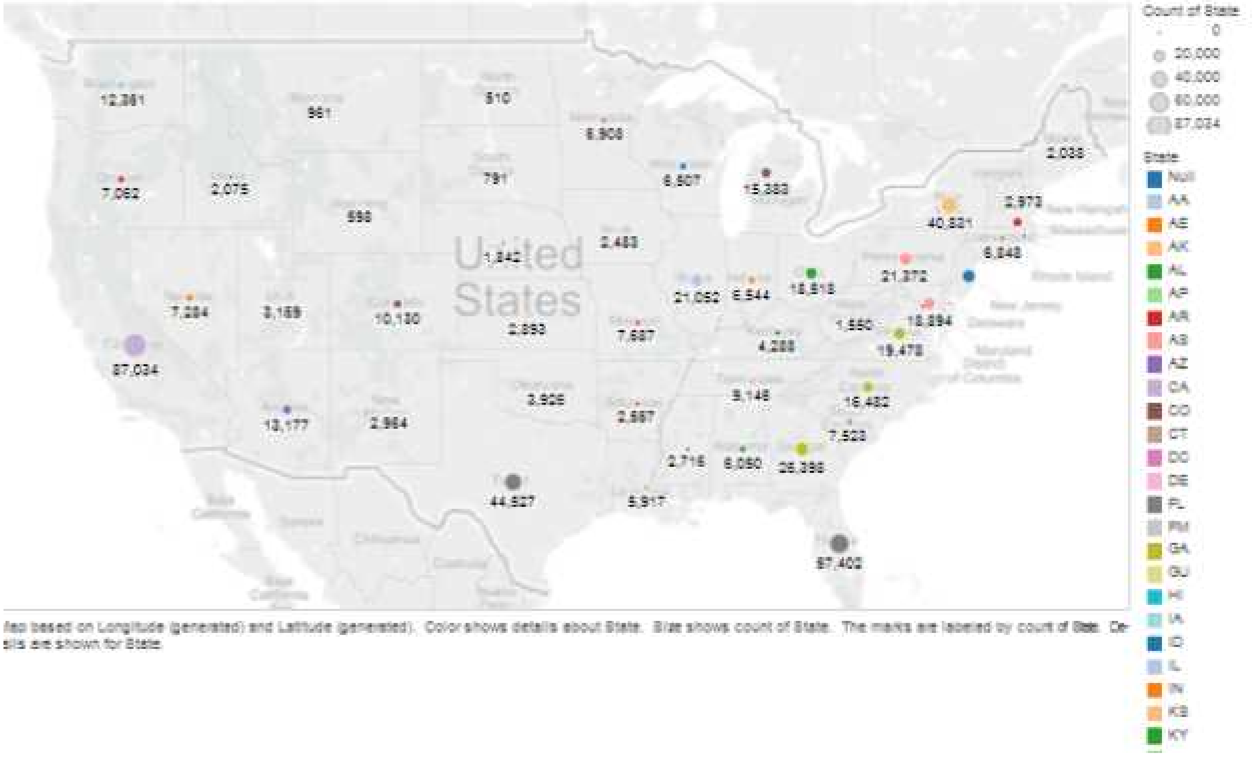
Timely Response



Count of Timely response? for each Timely response?. Color shows details about Timely response?.

- Maximum issues registered state wise

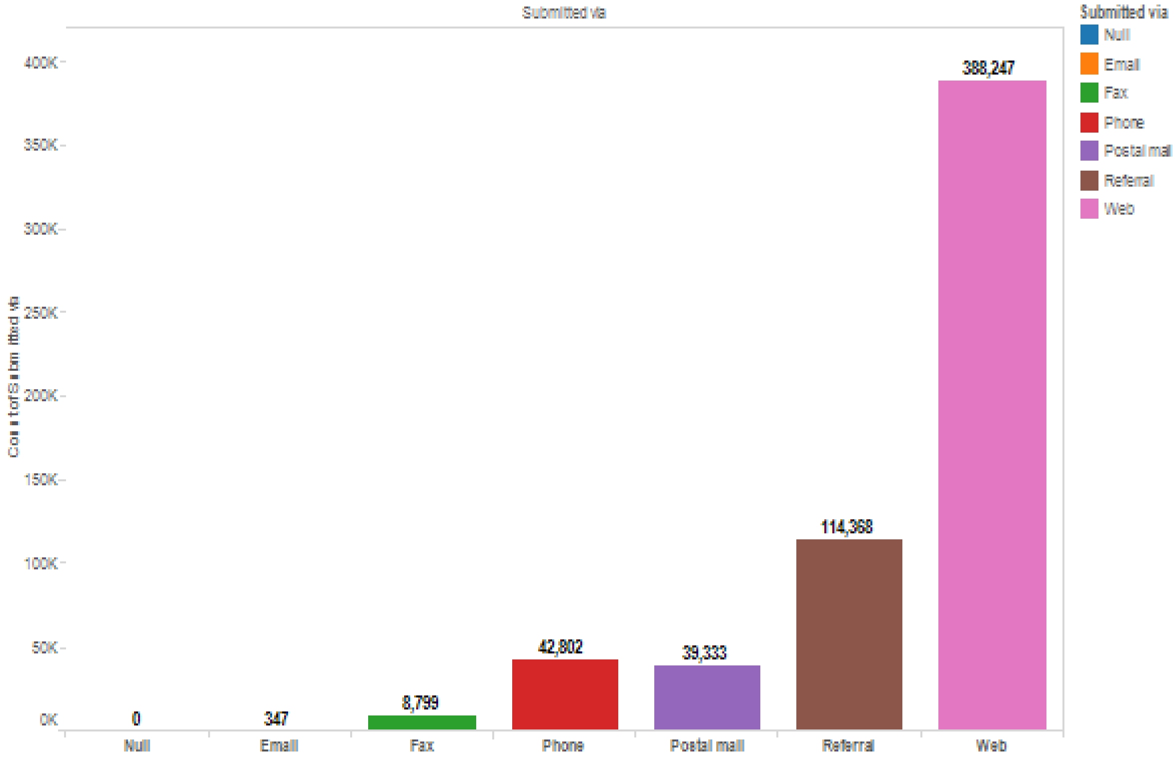
lap Of Statewise Count



lap based on Longitude (generated) and Latitude (generated). Color shows details about State. Size shows count of State. The marks are labeled by count of State. Details are shown for State.

- Complaints submitted via

submitted via

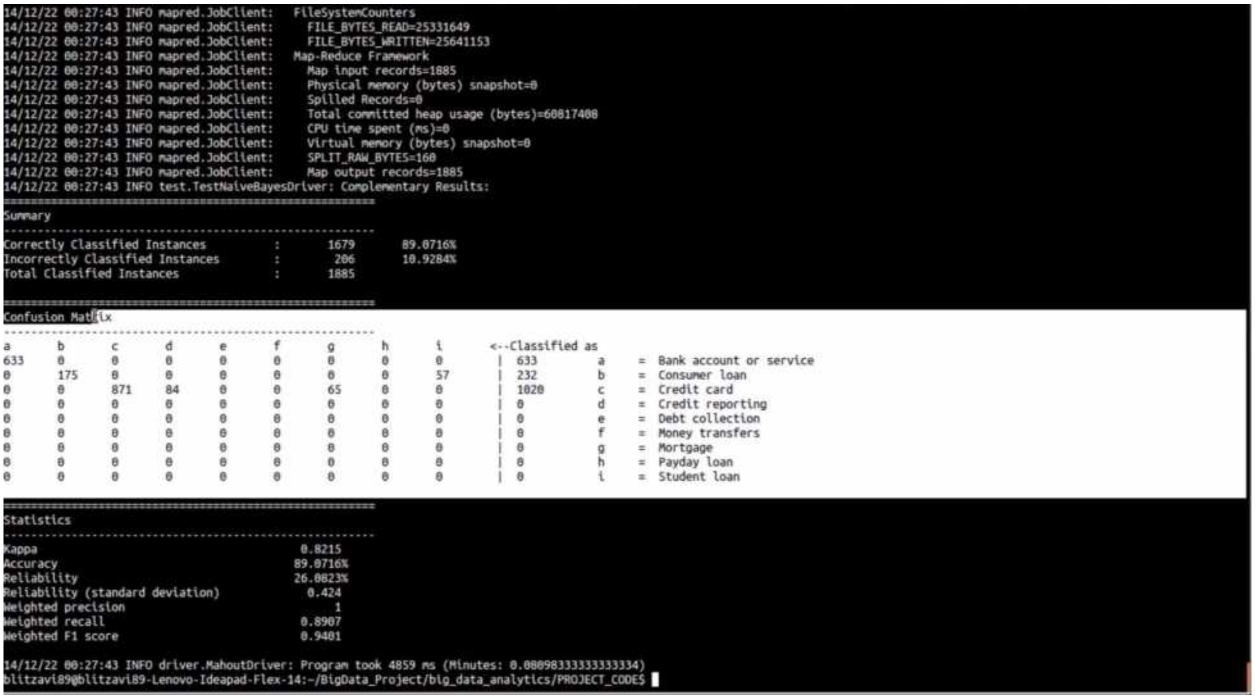


Count of Submitted via for each Submitted via. Color shows details about Submitted via. Details are shown for Submitted via.

ANALYSIS OF THE RESULTS

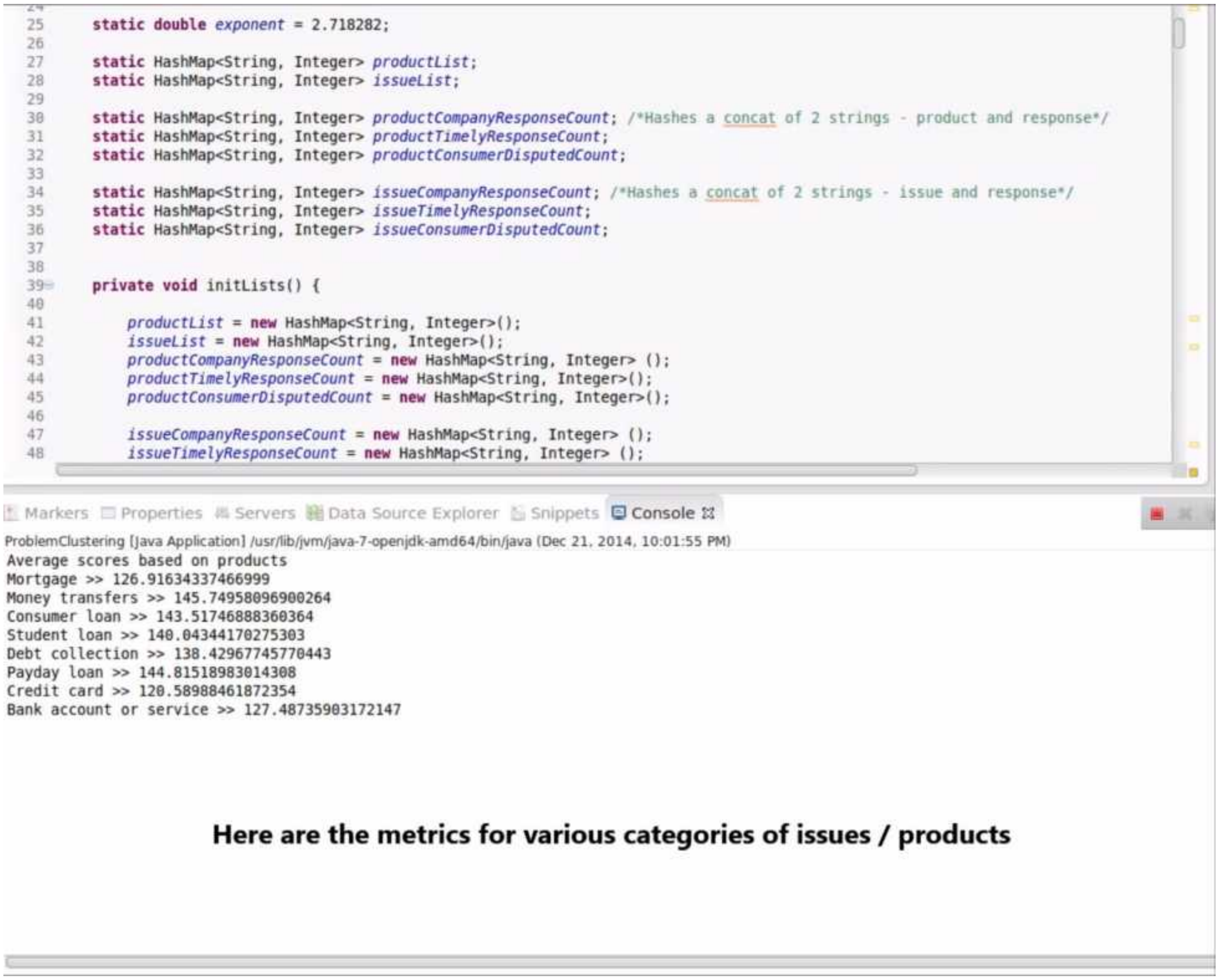
- Analysis of the results we had are extremely encouraging as we took the datasets of the Consumer Financial Protection Bureau (CFPB) of the states and made it valuable by classifying it with regards to specific inputs and outputs which can help the current scenario as according to us there are more frustrated people with the current redressal mechanism adopted by the banking industry .

- So talking about the analysis of the results we had with respect to classification done on Apache Mahout with Naive bayes classification we found out that the module had a higher accuracy of about 95 to 100% with the training dataset and the accuracy of our module classification with the testing dataset had an accuracy of 85 to 90% which is shown below and was predicted which in turn tells us about the analysis was good.



```
14/12/22 00:27:43 INFO mapred.JobClient:   FileSystemCounters
14/12/22 00:27:43 INFO mapred.JobClient:     FILE_BYTES_READ=25331649
14/12/22 00:27:43 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=25641153
14/12/22 00:27:43 INFO mapred.JobClient:   Map-Reduce Framework
14/12/22 00:27:43 INFO mapred.JobClient:     Map input records=1885
14/12/22 00:27:43 INFO mapred.JobClient:     Physical memory (bytes) snapshot=0
14/12/22 00:27:43 INFO mapred.JobClient:     Spilled Records=0
14/12/22 00:27:43 INFO mapred.JobClient:     Total committed heap usage (bytes)=60817408
14/12/22 00:27:43 INFO mapred.JobClient:     CPU time spent (ms)=0
14/12/22 00:27:43 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=0
14/12/22 00:27:43 INFO mapred.JobClient:     SPLIT_RAW_BYTES=160
14/12/22 00:27:43 INFO mapred.JobClient:     Map output records=1885
14/12/22 00:27:43 INFO test.TestNaiveBayesDriver: Complementary Results:
=======================================================
Summary
-------------------------------------------------------
Correctly Classified Instances          :       1679       89.0716%
Incorrectly Classified Instances        :        206       10.9284%
Total Classified Instances              :       1885

=======================================================
Confusion Matrix
-------------------------------------------------------
a       b       c       d       e       f       g       h       i        <--Classified as
633     0       0       0       0       0       0       0       0         |  633         a     =  Bank account or service
0       175     0       0       0       0       0       0       57        |  232         b     =  Consumer loan
0       0       871     84      0       0       65      0       0         |  1020        c     =  Credit card
0       0       0       0       0       0       0       0       0         |  0           d     =  Credit reporting
0       0       0       0       0       0       0       0       0         |  0           e     =  Debt collection
0       0       0       0       0       0       0       0       0         |  0           f     =  Money transfers
0       0       0       0       0       0       0       0       0         |  0           g     =  Mortgage
0       0       0       0       0       0       0       0       0         |  0           h     =  Payday loan
0       0       0       0       0       0       0       0       0         |  0           i     =  Student loan

=======================================================
Statistics
-------------------------------------------------------
Kappa                                       0.8215
Accuracy                                   89.0716%
Reliability                                26.0823%
Reliability (standard deviation)            0.424
Weighted precision                              1
Weighted recall                             0.8907
Weighted F1 score                           0.9401

14/12/22 00:27:43 INFO driver.MahoutDriver: Program took 4859 ms (Minutes: 0.08098333333333334)
blitzavi89@blitzavi89-Lenovo-Ideapad-Flex-14:~/BigData_Project/big_data_analytics/PROJECT_CODE$
```

- Performance metric analysis calculation for various categories of issues that are actually entered by the customer are parameters that help us better the calculator and up the game of redressal mechanism. When a customer is entering a specific complaint with the database, he can have issues with credit card, debit card, etc so based on the priority values that are being calculated by the code we can figure out which issue is supposed to be given a higher priority while a customer actually enters his specific complaint allowing him to a better redressal experience. Here in this performance analysis everything ran as predicted keeping the resources in mind.

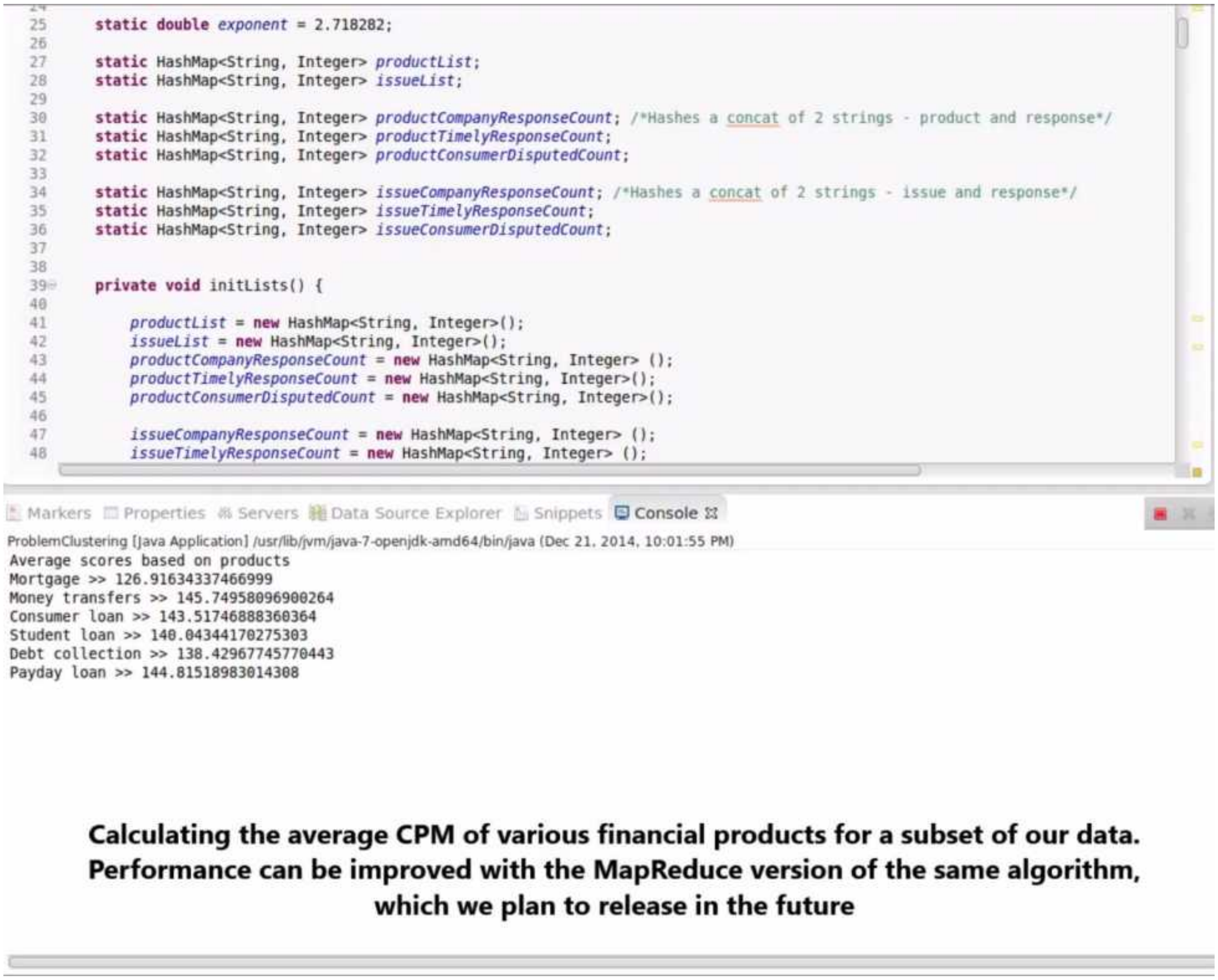


```
    static double exponent = 2.718282;

    static HashMap<String, Integer> productList;
    static HashMap<String, Integer> issueList;

    static HashMap<String, Integer> productCompanyResponseCount; /*Hashes a concat of 2 strings - product and response*/
    static HashMap<String, Integer> productTimelyResponseCount;
    static HashMap<String, Integer> productConsumerDisputedCount;

    static HashMap<String, Integer> issueCompanyResponseCount; /*Hashes a concat of 2 strings - issue and response*/
    static HashMap<String, Integer> issueTimelyResponseCount;
    static HashMap<String, Integer> issueConsumerDisputedCount;


    private void initLists() {

        productList = new HashMap<String, Integer>();
        issueList = new HashMap<String, Integer>();
        productCompanyResponseCount = new HashMap<String, Integer> ();
        productTimelyResponseCount = new HashMap<String, Integer>();
        productConsumerDisputedCount = new HashMap<String, Integer>();

        issueCompanyResponseCount = new HashMap<String, Integer> ();
        issueTimelyResponseCount = new HashMap<String, Integer> ();
```

```
ProblemClustering [Java Application] /usr/lib/jvm/java-7-openjdk-amd64/bin/java (Dec 21, 2014, 10:01:55 PM)
Average scores based on products
Mortgage >> 126.91634337466999
Money transfers >> 145.74958096900264
Consumer loan >> 143.51746888360364
Student loan >> 140.04344170275303
Debt collection >> 138.42967745770443
Payday loan >> 144.81518983014308
Credit card >> 120.58988461872354
Bank account or service >> 127.48735903172147
```

Here are the metrics for various categories of issues / products

- The performance metric analysis calculator which did calculation for the current module can of course be further improvised by using map reduce version of it. It will also help in reducing response time of the query, better availability and seamless experience with the whole complaint to resolution cycle reduced to its limit.



```java
    static double exponent = 2.718282;

    static HashMap<String, Integer> productList;
    static HashMap<String, Integer> issueList;

    static HashMap<String, Integer> productCompanyResponseCount; /*Hashes a concat of 2 strings - product and response*/
    static HashMap<String, Integer> productTimelyResponseCount;
    static HashMap<String, Integer> productConsumerDisputedCount;

    static HashMap<String, Integer> issueCompanyResponseCount; /*Hashes a concat of 2 strings - issue and response*/
    static HashMap<String, Integer> issueTimelyResponseCount;
    static HashMap<String, Integer> issueConsumerDisputedCount;


    private void initLists() {

        productList = new HashMap<String, Integer>();
        issueList = new HashMap<String, Integer>();
        productCompanyResponseCount = new HashMap<String, Integer> ();
        productTimelyResponseCount = new HashMap<String, Integer>();
        productConsumerDisputedCount = new HashMap<String, Integer>();

        issueCompanyResponseCount = new HashMap<String, Integer> ();
        issueTimelyResponseCount = new HashMap<String, Integer> ();
```

```
ProblemClustering [Java Application] /usr/lib/jvm/java-7-openjdk-amd64/bin/java (Dec 21, 2014, 10:01:55 PM)
Average scores based on products
Mortgage >> 126.91634337466999
Money transfers >> 145.74958096900264
Consumer loan >> 143.51746888360364
Student loan >> 140.04344170275303
Debt collection >> 138.42967745770443
Payday loan >> 144.81518983014308
```

Calculating the average CPM of various financial products for a subset of our data. Performance can be improved with the MapReduce version of the same algorithm, which we plan to release in the future

CONCLUSION

- From the above mentioned analysis, it is seen that the credit card complaints are increasing since the last 5 years. From the business point of view, the home loan objections can be decreased to a degree if the banks' records are frequently checked by the powers and the banks ought to be advised to deliver and submit reports of their exercises all the time barring the clients' private and classified information.

- Steps should be taken to verify all documents and avoid customers dissatisfaction. During this project we learned and improved our analytical skills. Working on Hadoop Distributed File System helped us gain knowledge on Hive Query Language.

- Meanwhile we were able to represent queries into graphical form. In our detailed analysis, we were able to analyze and represent the maximum number of complaints in California with the majority of complaints being of credit card Complaints. Also, through our analysis, we found which banks are complained about the most and via which web are the complaints recorded. Also it came to our notice that all complaints were timely responded.

- If talking about very specific analysis trend and details drawn from it we can also conclude the following
    - Mortgage product and other mortgage sub-product complaints are maximum in number.
    - California has the highest number of complaints.
    - Most of the complaints are registered with Bank of america.
    - Consumer complaints are increasing year by year.

- This brought us to conclusion providing useful aspects in regards to recurrence of complaints, complaints from different places, about diverse products, sub-products and companies which make us aware and take necessary steps to avoid them.

- With our new metric system, banks can relatively prioritize the complaints to resolve as we proposed a system to evaluate a good redressal mechanism for banking customers/consumers complaints based on the analysis.

- By analysing and processing the text, the system can make the reviews into a graph that is easy to read and understand. Performed data analysis on the data sets, to give a detailed overview of the banks performance from a customer sentiment perspective.

- Developed a novel metric system that assigns priorities to customers complaints. This helps banks prioritize customers problems on specific constraints such as response time, etc.

FUTURE ENHANCEMENT

- Resolution Methodology Recommender.Build a recommender engine that can derive the best “first response” for a complaint.

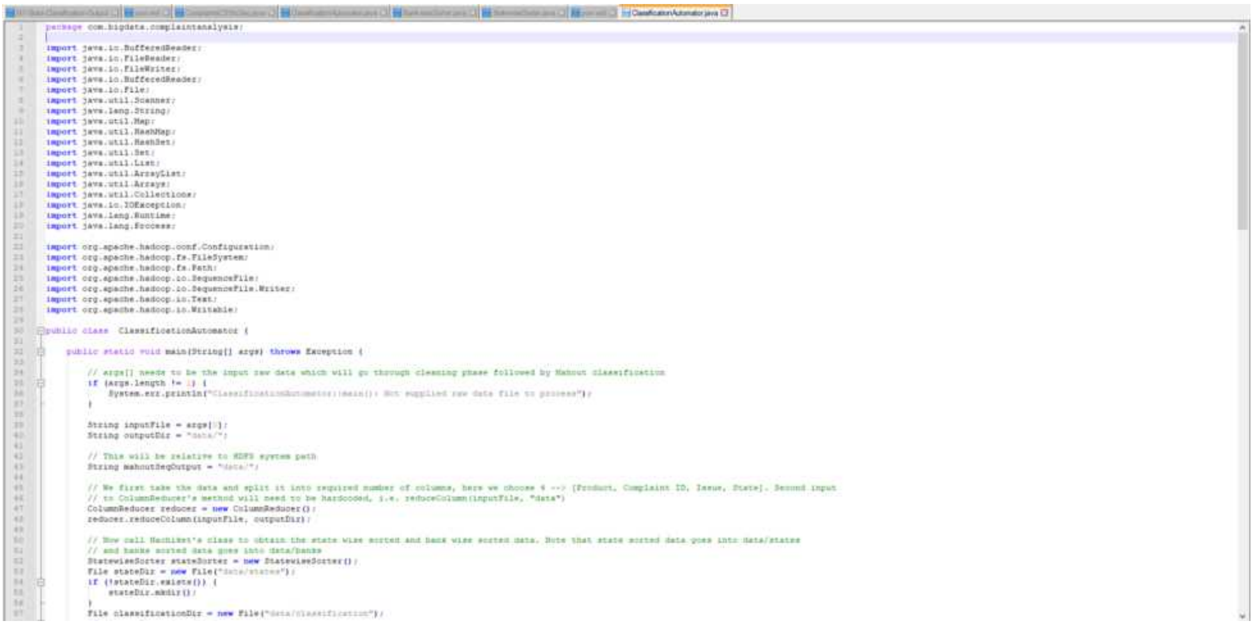
- More rigorous data analysis and research into complaint resolution methodologies required.

- Quantitatively gauge impact of various classes of complaints on various products to gain insights into customer outlook towards a specific product.

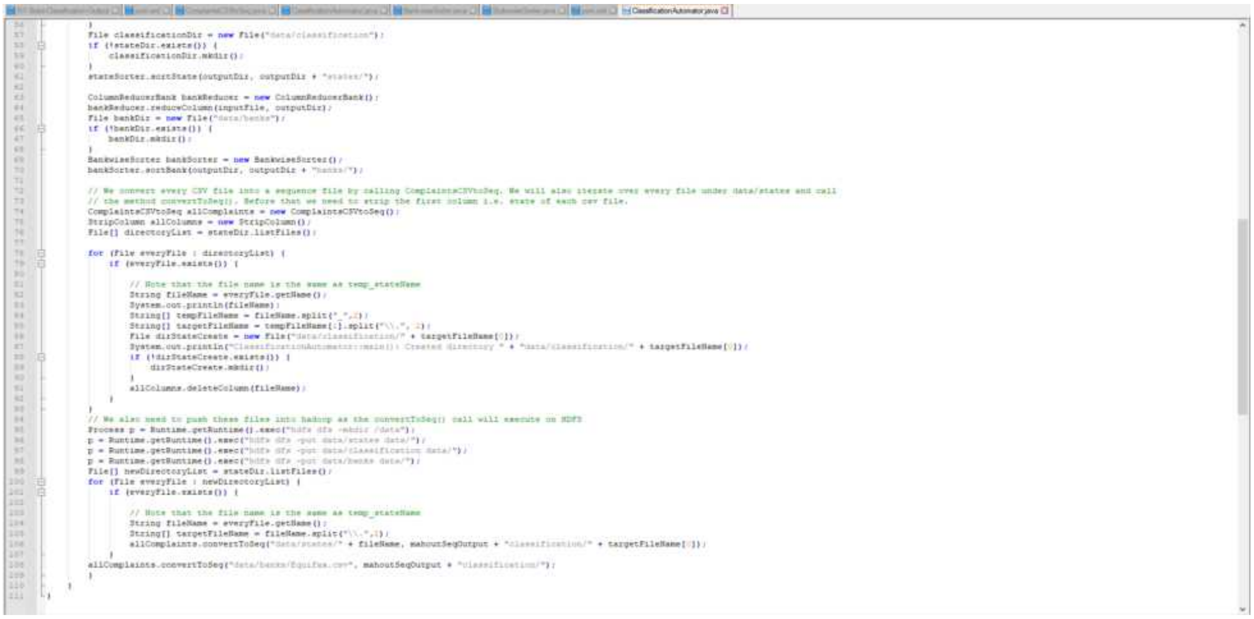
- Seek to make a better performance metric analysis calculator with the map reduce version of what was used in this module of customer complaints analysis with respect to banking

PROGRAM CODE

- Program code of this project has some major files coded in java and xml both as all the main class files are being called by a file called ClassificationAutomator which basically calls all the useful and dependent java class files and the main class too.

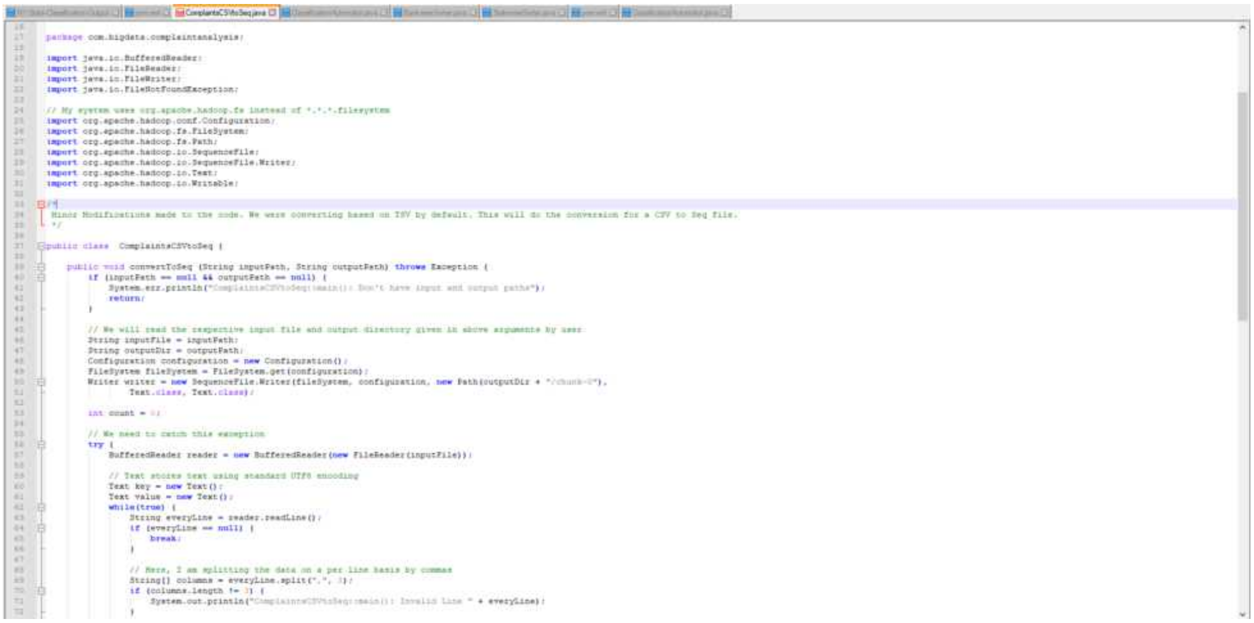


```
package com.bigdata.complaintanalysis;

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.BufferedReader;
import java.io.File;
import java.util.Scanner;
import java.lang.String;
import java.util.Map;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Set;
import java.util.List;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.io.IOException;
import java.lang.Runtime;
import java.lang.Process;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.SequenceFile;
import org.apache.hadoop.io.SequenceFile.Writer;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.Writable;

public class  ClassificationAutomator {

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

        // args[] needs to be the input raw data which will go through cleaning phase followed by Mahout classification
        if (args.length != 1) {
            System.err.println("ClassificationAutomator::main(): Not supplied raw data file to process");
        }

        String inputFile = args[0];
        String outputDir = "data/";

        // This will be relative to HDFS system path
        String mahoutSeqOutput = "data/";

        // We first take the data and split it into required number of columns, here we choose 4 --> [Product, Complaint ID, Issue, State]. Second input
        // to ColumnReducer's method will need to be hardcoded, i.e. reduceColumn(inputFile, "data")
        ColumnReducer reducer = new ColumnReducer();
        reducer.reduceColumn(inputFile, outputDir);

        // Now call Hachiket's class to obtain the state wise sorted and bank wise sorted data. Note that state sorted data goes into data/states
        // and banks sorted data goes into data/banks
        StatewiseSorter stateSorter = new StatewiseSorter();
        File stateDir = new File("data/states");
        if (!stateDir.exists()) {
            stateDir.mkdir();
        }
        File classificationDir = new File("data/classification");
```

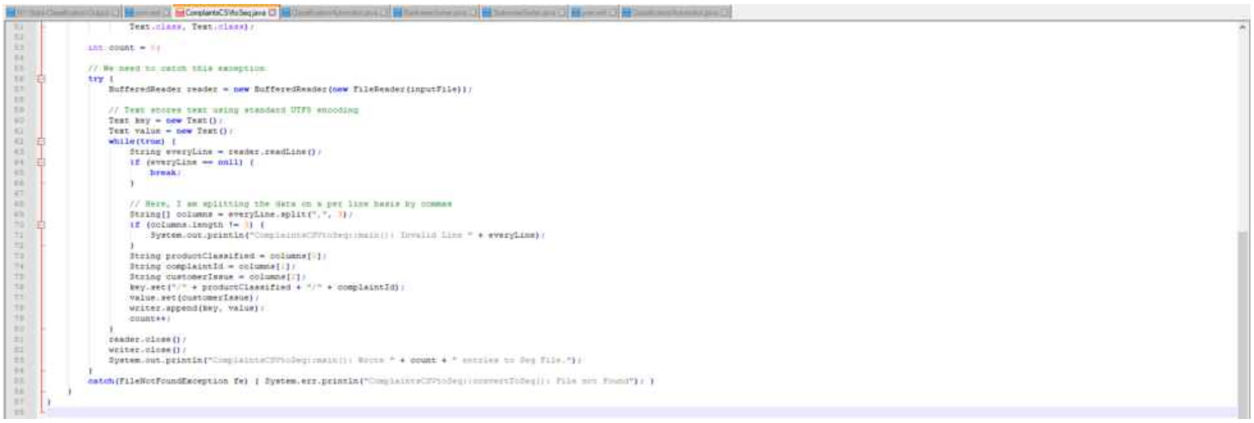


```
        }
        File classificationDir = new File("data/classification");
        if (!stateDir.exists()) {
            classificationDir.mkdir();
        }
        stateSorter.sortState(outputDir, outputDir + "states/");

        ColumnReducerBank bankReducer = new ColumnReducerBank();
        bankReducer.reduceColumn(inputFile, outputDir);
        File bankDir = new File("data/banks");
        if (!bankDir.exists()) {
            bankDir.mkdir();
        }
        BankwiseSorter bankSorter = new BankwiseSorter();
        bankSorter.sortBank(outputDir, outputDir + "banks/");

        // We convert every CSV file into a sequence file by calling ComplaintsCSVtoSeq. We will also iterate over every file under data/states and call
        // the method convertToSeq(). Before that we need to strip the first column i.e. state of each csv file.
        ComplaintsCSVtoSeq allComplaints = new ComplaintsCSVtoSeq();
        StripColumn allColumns = new StripColumn();
        File[] directoryList = stateDir.listFiles();

        for (File everyFile : directoryList) {
            if (everyFile.exists()) {

                // Note that the file name is the same as temp_stateName
                String fileName = everyFile.getName();
                System.out.println(fileName);
                String[] tempFileName = fileName.split("_",2);
                String[] targetFileName = tempFileName[1].split("\\.", 2);
                File dirStateCreate = new File("data/classification/" + targetFileName[0]);
                System.out.println("ClassificationAutomator::main(): Created directory " + "data/classification/" + targetFileName[0]);
                if (!dirStateCreate.exists()) {
                    dirStateCreate.mkdir();
                }
                allColumns.deleteColumn(fileName);
            }
        }
        // We also need to push these files into hadoop as the convertToSeq() call will execute on HDFS
        Process p = Runtime.getRuntime().exec("hdfs dfs -mkdir /data");
        p = Runtime.getRuntime().exec("hdfs dfs -put data/states data/");
        p = Runtime.getRuntime().exec("hdfs dfs -put data/classification data/");
        p = Runtime.getRuntime().exec("hdfs dfs -put data/banks data/");
        File[] newDirectoryList = stateDir.listFiles();
        for (File everyFile : newDirectoryList) {
            if (everyFile.exists()) {

                // Note that the file name is the same as temp_stateName
                String fileName = everyFile.getName();
                String[] targetFileName = fileName.split("\\.",2);
                allComplaints.convertToSeq("data/states/" + fileName, mahoutSeqOutput + "classification/" + targetFileName[0]);
            }
        allComplaints.convertToSeq("data/banks/Equifax.csv", mahoutSeqOutput + "classification/");
        }
    }
}
```

- Then comes one of the main java files named ComplaintsCSVtoSeq java file it is the heart of our project as it generates sequence files and pushes it automatically to the hdfs.



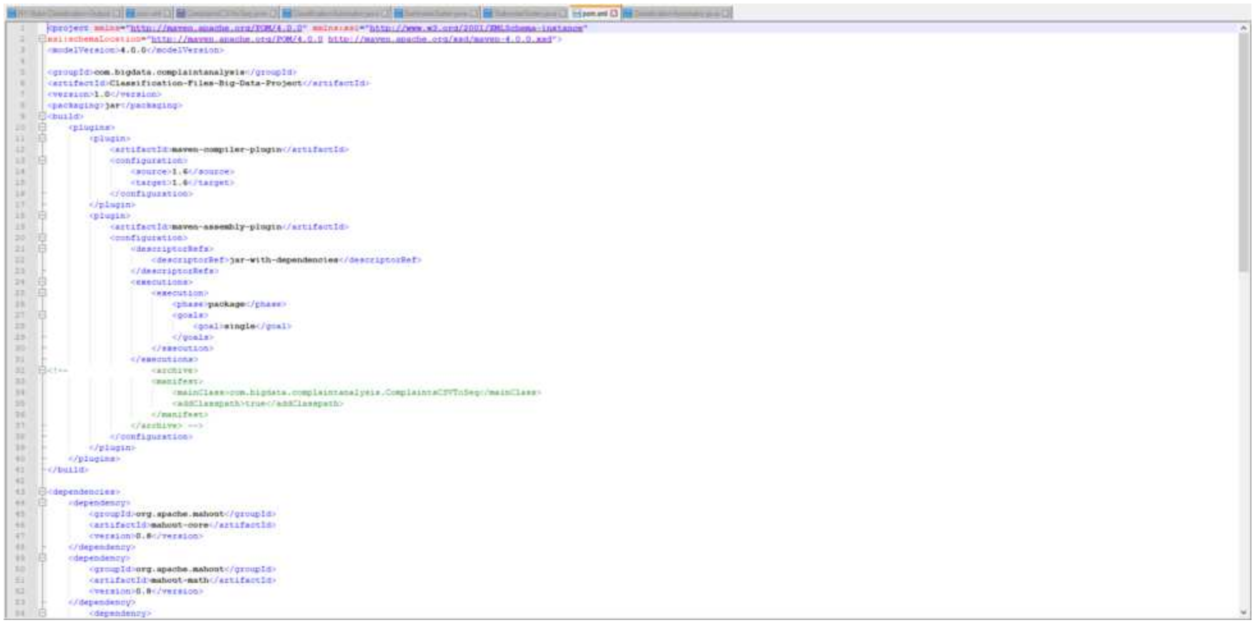
```java
package com.bigdata.complaintanalysis;

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.FileNotFoundException;

// My system uses org.apache.hadoop.fs instead of *.*.*.filesystem
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.SequenceFile;
import org.apache.hadoop.io.SequenceFile.Writer;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.Writable;

/*
 Minor Modifications made to the code. We were converting based on TSV by default. This will do the conversion for a CSV to Seq file.
 */

public class  ComplaintsCSVtoSeq {

    public void convertToSeq (String inputPath, String outputPath) throws Exception {
        if (inputPath == null && outputPath == null) {
            System.err.println("ComplaintsCSVtoSeq::main(): Don't have input and output paths");
            return;
        }

        // We will read the respective input file and output directory given in above arguments by user
        String inputFile = inputPath;
        String outputDir = outputPath;
        Configuration configuration = new Configuration();
        FileSystem fileSystem = FileSystem.get(configuration);
        Writer writer = new SequenceFile.Writer(fileSystem, configuration, new Path(outputDir + "/chunk-0"),
                Text.class, Text.class);

        int count = 0;

        // We need to catch this exception
        try {
            BufferedReader reader = new BufferedReader(new FileReader(inputFile));

            // Text stores text using standard UTF8 encoding
            Text key = new Text();
            Text value = new Text();
            while(true) {
                String everyLine = reader.readLine();
                if (everyLine == null) {
                    break;
                }

                // Here, I am splitting the data on a per line basis by commas
                String[] columns = everyLine.split(",", 3);
                if (columns.length != 3) {
                    System.out.println("ComplaintsCSVtoSeq::main(): Invalid Line " + everyLine);
                }
```



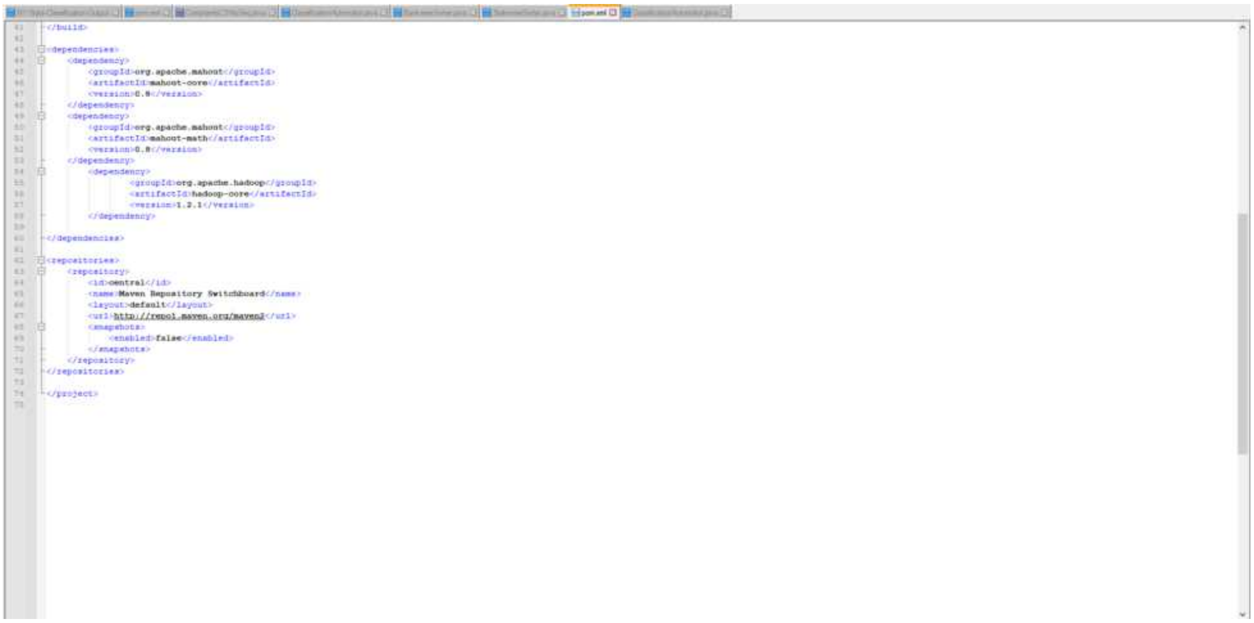
```java
                Text.class, Text.class);

        int count = 0;

        // We need to catch this exception
        try {
            BufferedReader reader = new BufferedReader(new FileReader(inputFile));

            // Text stores text using standard UTF8 encoding
            Text key = new Text();
            Text value = new Text();
            while(true) {
                String everyLine = reader.readLine();
                if (everyLine == null) {
                    break;
                }

                // Here, I am splitting the data on a per line basis by commas
                String[] columns = everyLine.split(",", 3);
                if (columns.length != 3) {
                    System.out.println("ComplaintsCSVtoSeq::main(): Invalid Line " + everyLine);
                }
                String productClassified = columns[0];
                String complaintId = columns[1];
                String customerIssue = columns[2];
                key.set("/" + productClassified + "/" + complaintId);
                value.set(customerIssue);
                writer.append(key, value);
                count++;
            }
            reader.close();
            writer.close();
            System.out.println("ComplaintsCSVtoSeq::main(): Wrote " + count + " entries to Seq File.");
        }
        catch(FileNotFoundException fe) { System.err.println("ComplaintsCSVtoSeq::convertToSeq(): File not Found"); }
    }
}
```

- A Project Object Model or POM is the fundamental unit of work in Maven. It is an XML file that contains information about the project and configuration details used by Maven to build the project. It contains default values for most projects. Examples for this are the build directory, which is target; the source directory, which is src/main/java;

the test source directory, which is src/test/java; and so on. When executing a task or goal, Maven looks for the POM in the current directory. It reads the POM, gets the needed configuration information, then executes the goal.



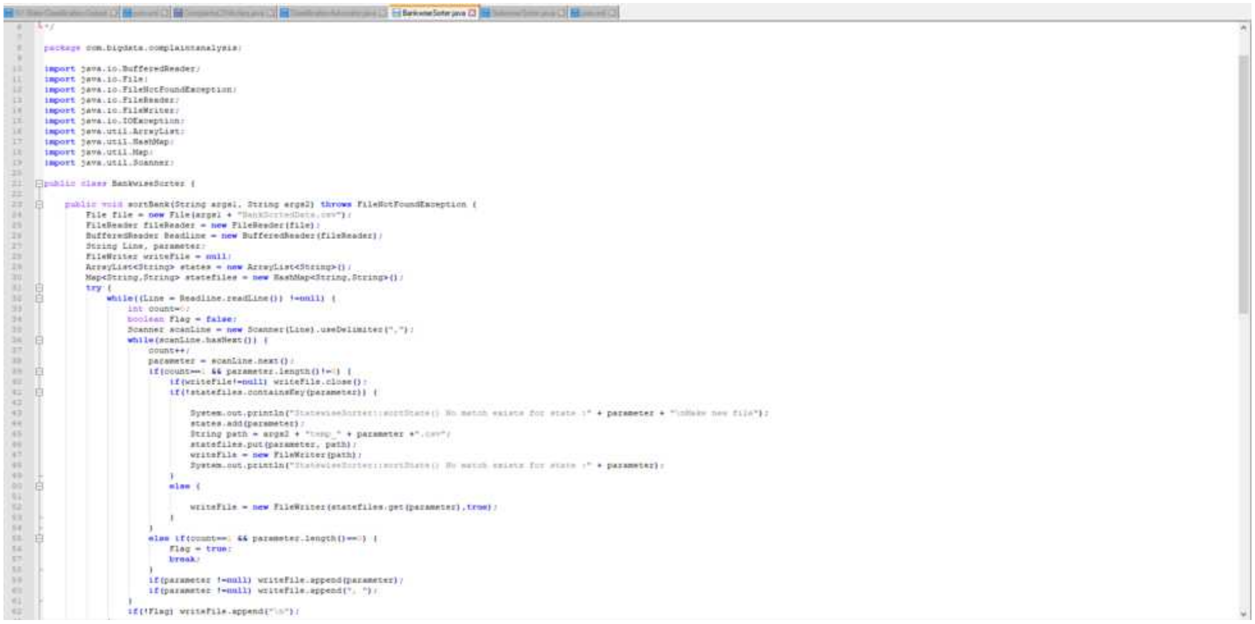
```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>

<groupId>com.bigdata.complaintanalysis</groupId>
<artifactId>Classification-Files-Big-Data-Project</artifactId>
<version>1.0</version>
<packaging>jar</packaging>
<build>
    <plugins>
        <plugin>
            <artifactId>maven-compiler-plugin</artifactId>
            <configuration>
                <source>1.6</source>
                <target>1.6</target>
            </configuration>
        </plugin>
        <plugin>
            <artifactId>maven-assembly-plugin</artifactId>
            <configuration>
                <descriptorRefs>
                    <descriptorRef>jar-with-dependencies</descriptorRef>
                </descriptorRefs>
                <executions>
                    <execution>
                        <phase>package</phase>
                        <goals>
                            <goal>single</goal>
                        </goals>
                    </execution>
                </executions>
<!--                <archive>
                    <manifest>
                        <mainClass>com.bigdata.complaintanalysis.ComplaintsCSVToSeq</mainClass>
                        <addClasspath>true</addClasspath>
                    </manifest>
                </archive> -->
            </configuration>
        </plugin>
    </plugins>
</build>

<dependencies>
    <dependency>
        <groupId>org.apache.mahout</groupId>
        <artifactId>mahout-core</artifactId>
        <version>0.8</version>
    </dependency>
    <dependency>
        <groupId>org.apache.mahout</groupId>
        <artifactId>mahout-math</artifactId>
        <version>0.8</version>
    </dependency>
        <dependency>
```



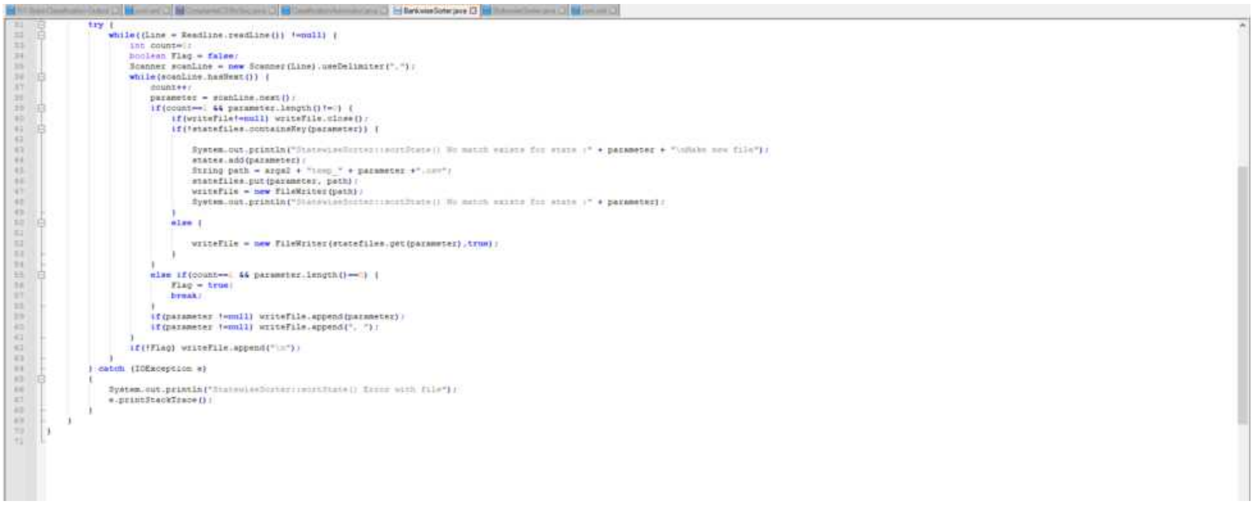
```
</build>

<dependencies>
    <dependency>
        <groupId>org.apache.mahout</groupId>
        <artifactId>mahout-core</artifactId>
        <version>0.8</version>
    </dependency>
    <dependency>
        <groupId>org.apache.mahout</groupId>
        <artifactId>mahout-math</artifactId>
        <version>0.8</version>
    </dependency>
        <dependency>
                <groupId>org.apache.hadoop</groupId>
                <artifactId>hadoop-core</artifactId>
                <version>1.2.1</version>
        </dependency>

</dependencies>

<repositories>
    <repository>
        <id>central</id>
        <name>Maven Repository Switchboard</name>
        <layout>default</layout>
        <url>http://repo1.maven.org/maven2</url>
        <snapshots>
            <enabled>false</enabled>
        </snapshots>
    </repository>
</repositories>

</project>
```

- Then come the secondary files mandatory for smooth execution such as vectors and bank and states sorter files also we have not uploaded all the files for all files we have drop our github repository link where you can find the whole project code and some more valuable information.

  Following is the code for Bankwise Sorting

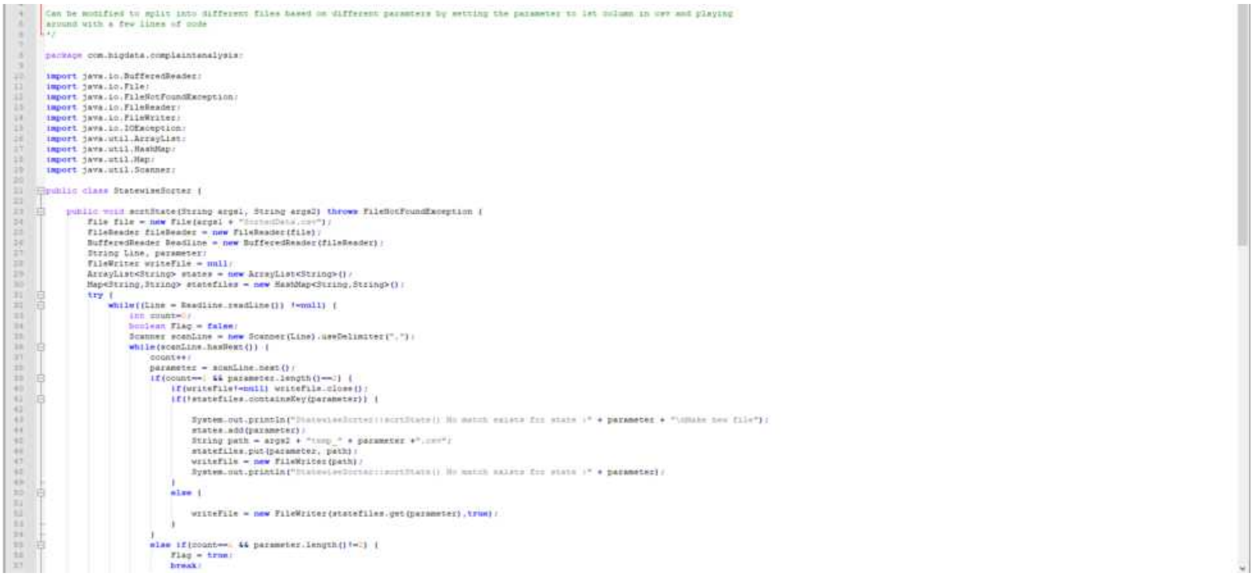


```
 */

package com.bigdata.complaintanalysis;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class BankwiseSorter {

    public void sortBank(String args1, String args2) throws FileNotFoundException {
        File file = new File(args1 + "BankSortedData.csv");
        FileReader fileReader = new FileReader(file);
        BufferedReader Readline = new BufferedReader(fileReader);
        String Line, parameter;
        FileWriter writeFile = null;
        ArrayList<String> states = new ArrayList<String>();
        Map<String,String> statefiles = new HashMap<String,String>();
        try {
            while((Line = Readline.readLine()) !=null) {
                int count=0;
                boolean Flag = false;
                Scanner scanLine = new Scanner(Line).useDelimiter(",");
                while(scanLine.hasNext()) {
                    count++;
                    parameter = scanLine.next();
                    if(count==1 && parameter.length()!=0) {
                        if(writeFile!=null) writeFile.close();
                        if(!statefiles.containsKey(parameter)) {

                            System.out.println("StatewiseSorter::sortState() No match exists for state :" + parameter + "\nMake new file");
                            states.add(parameter);
                            String path = args2 + "temp_" + parameter +".csv";
                            statefiles.put(parameter, path);
                            writeFile = new FileWriter(path);
                            System.out.println("StatewiseSorter::sortState() No match exists for state :" + parameter);
                        }
                        else {

                            writeFile = new FileWriter(statefiles.get(parameter),true);
                        }
                    }
                    else if(count==1 && parameter.length()==0) {
                        Flag = true;
                        break;
                    }
                    if(parameter !=null) writeFile.append(parameter);
                    if(parameter !=null) writeFile.append(", ");
                }
                if(!Flag) writeFile.append("\n");
```

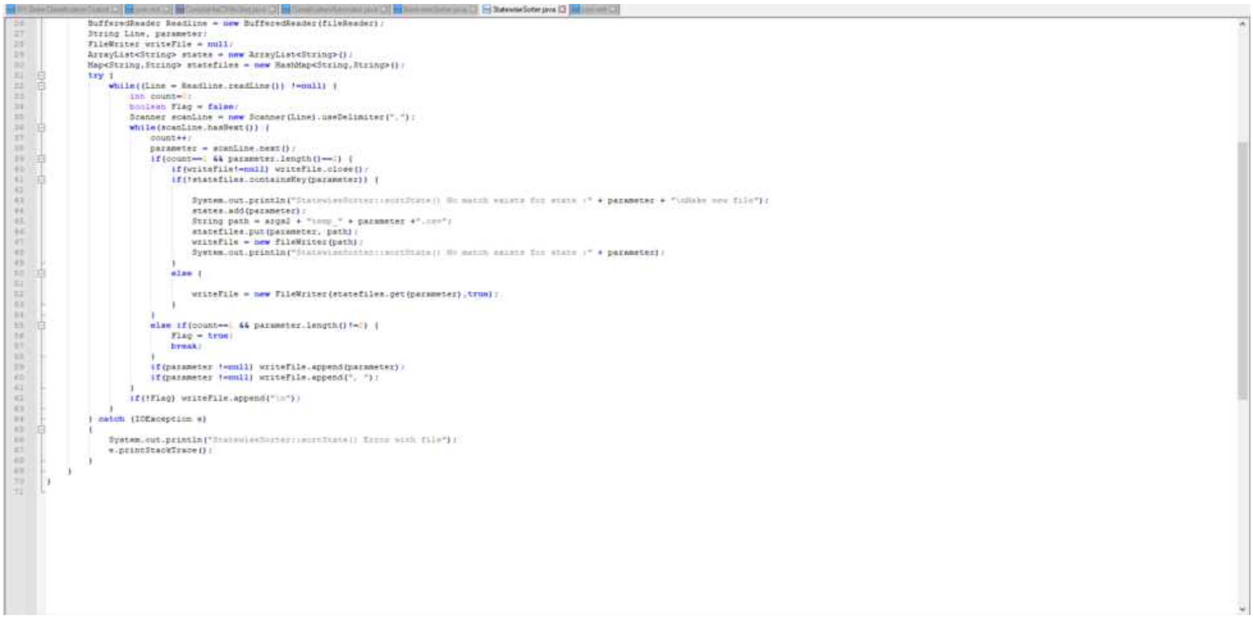


```
        try {
            while((Line = Readline.readLine()) !=null) {
                int count=0;
                boolean Flag = false;
                Scanner scanLine = new Scanner(Line).useDelimiter(",");
                while(scanLine.hasNext()) {
                    count++;
                    parameter = scanLine.next();
                    if(count==1 && parameter.length()!=0) {
                        if(writeFile!=null) writeFile.close();
                        if(!statefiles.containsKey(parameter)) {

                            System.out.println("StatewiseSorter::sortState() No match exists for state :" + parameter + "\nMake new file");
                            states.add(parameter);
                            String path = args2 + "temp_" + parameter +".csv";
                            statefiles.put(parameter, path);
                            writeFile = new FileWriter(path);
                            System.out.println("StatewiseSorter::sortState() No match exists for state :" + parameter);
                        }
                        else {

                            writeFile = new FileWriter(statefiles.get(parameter),true);
                        }
                    }
                    else if(count==1 && parameter.length()==0) {
                        Flag = true;
                        break;
                    }
                    if(parameter !=null) writeFile.append(parameter);
                    if(parameter !=null) writeFile.append(", ");
                }
                if(!Flag) writeFile.append("\n");
            }
        } catch (IOException e)
        {
            System.out.println("StatewiseSorter::sortState() Error with file");
            e.printStackTrace();
        }
    }
}
```

Next is the State wise Sorter file



```
Can be modified to split into different files based on different parameters by setting the parameter to 1st column in csv and playing
around with a few lines of code
*/

package com.bigdata.complaintanalysis;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class StatewiseSorter {

    public void sortState(String args1, String args2) throws FileNotFoundException {
        File file = new File(args1 + "SortedData.csv");
        FileReader fileReader = new FileReader(file);
        BufferedReader Readline = new BufferedReader(fileReader);
        String Line, parameter;
        FileWriter writeFile = null;
        ArrayList<String> states = new ArrayList<String>();
        Map<String,String> statefiles = new HashMap<String,String>();
        try {
            while((Line = Readline.readLine()) !=null) {
                int count=0;
                boolean Flag = false;
                Scanner scanLine = new Scanner(Line).useDelimiter(",");
                while(scanLine.hasNext()) {
                    count++;
                    parameter = scanLine.next();
                    if(count==1 && parameter.length()==2) {
                        if(writeFile!=null) writeFile.close();
                        if(!statefiles.containsKey(parameter)) {

                            System.out.println("StatewiseSorter::sortState() No match exists for state :" + parameter + "\nMake new file");
                            states.add(parameter);
                            String path = args2 + "temp_" + parameter +".csv";
                            statefiles.put(parameter, path);
                            writeFile = new FileWriter(path);
                            System.out.println("StatewiseSorter::sortState() No match exists for state :" + parameter);
                        }
                        else {

                            writeFile = new FileWriter(statefiles.get(parameter),true);
                        }
                    }
                    else if(count==1 && parameter.length()!=2) {
                        Flag = true;
                        break;
```



```
        BufferedReader Readline = new BufferedReader(fileReader);
        String Line, parameter;
        FileWriter writeFile = null;
        ArrayList<String> states = new ArrayList<String>();
        Map<String,String> statefiles = new HashMap<String,String>();
        try {
            while((Line = Readline.readLine()) !=null) {
                int count=0;
                boolean Flag = false;
                Scanner scanLine = new Scanner(Line).useDelimiter(",");
                while(scanLine.hasNext()) {
                    count++;
                    parameter = scanLine.next();
                    if(count==1 && parameter.length()==2) {
                        if(writeFile!=null) writeFile.close();
                        if(!statefiles.containsKey(parameter)) {

                            System.out.println("StatewiseSorter::sortState() No match exists for state :" + parameter + "\nMake new file");
                            states.add(parameter);
                            String path = args2 + "temp_" + parameter +".csv";
                            statefiles.put(parameter, path);
                            writeFile = new FileWriter(path);
                            System.out.println("StatewiseSorter::sortState() No match exists for state :" + parameter);
                        }
                        else {

                            writeFile = new FileWriter(statefiles.get(parameter),true);
                        }
                    }
                    else if(count==1 && parameter.length()!=2) {
                        Flag = true;
                        break;
                    }
                    if(parameter !=null) writeFile.append(parameter);
                    if(parameter !=null) writeFile.append(", ");
                }
                if(!Flag) writeFile.append("\n");
            }
        } catch (IOException e)
        {
            System.out.println("StatewiseSorter::sortState() Error with file");
            e.printStackTrace();
        }
    }
}
```

- The link for the whole project code and some more files related to analysis and execution can be found at

  [https://github.com/Vshal-P/CCA]