**BIG DATA ANALYTICS USING HADOOP**

### A PROJECT REPORT

Submitted in partial fulfillment for the award of the degree of

**B.TECH**

***in***

**Information Technology**

***By***

**N Nagamani 12BIT0258**

**Roop Sai Krishna 12BIT0156**

**T Prashanth Reddy 12BIT0077**

**Under the Guidance of**

#### Prof.Hari Ram Vishwakarma

****

## School of Information Technology & Engineering

April - 2015

**DECLARATION BY THE CANDIDATE**

We here by declare that the project report entitled “**BIG DATA ANALYTICS USING HADOOP”** submitted by us to Vellore Institute of Technology University, Vellore in partial fulfillment of the requirement for the award of the degree of **B.Tech.(Information Technology)** is a record of bonafide project work carried out by us under the guidance of **Prof.Hari Ram Vishwakarma**.We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore Signature of the Candidate

Date: 29-04-2015 Roop Sai Krishna

N Nagamani

T Prashanth Reddy

## vitlogo (2)

## School of Information Technology & Engineering [SITE]

###### CERTIFICATE

This is to certify that the project report entitled “**BIG DATA ANALYTICS USING HADOOP”** submitted by **Roop Sai Krishna (12BIT0156),N Nagamani(12BIT0258),T Prashanth Reddy(12BIT0077)** to Vellore Institute of Technology University, Vellore in partial fulfillment of the requirement for the award of the degree of **B.Tech.(Information Technology)** is a record of bonafide work carried out by them under my guidance. The project fulfills the requirements as per the regulations of this Institute and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

**Prof.Hari Ram Vishwakarma**

**SUPERVISOR**

**Senior Professor, SITE**

The Project Report is **Satisfactory / unsatisfactory**

**Name & Signature of the Examiners**

**ACKNOWLEDGEMENT**

We would like to extend our sincere thanks to Chancellor **Dr. G. Viswanathan**, Vice Chancellor **Dr.V.Raju**, Pro-Vice Chancellor **Dr. S. Narayanan**, Dean-SITE **Dr. K.Ganesan**, Program Chair–**Prof. R. K. Nadesh** Year Coordinator – **Prof. P. Prabhavathy** & **Prof. K. Arivuselvan** for their generous guidance, continuous encouragement, help and useful suggestions.

We are highly indebted to **Prof.Hari Ram Vishwakarma** for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his support in completing the project.

We also wish to extend our thanks to all colleagues who have helped us in successful completion of this project.

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| Chapter No | Contents | Page No |
|  | LIST OF FIGURES | I |
| 1 | Introduction | 1 |
| 1.1 | Background | 1 |
| 1.2 | Problem Statement | 1 |
| 1.3 | Importance | 1 |
| 2 | Overview and Planning | 1 |
| 2.1 | Proposed System Overview | 1 |
| 2.2 | Challenges | 2 |
| 2.3 | Assumptions | 2 |
| 2.4 | Architecture Specifications | 2 |
| 2.5 | Hardware Requirements (Optimum requirements) | 3 |
| 2.6 | Software Requirements | 3 |
| 2.7 | Project Schedule (Gantt chart) | 4 |
| 2.7 | Work Breakdown Structure | 5 |
| 3 | Literature Survey and Review | 6 |
| 3.1 | Literature Survey | 6 |
| 3.2 | Literature Summary | 7 |
| 4 | System Design | 8 |
| 4.1 | High-Level Design | 8 |
| 4.2 | Low-Level Design | 9 |
| 5. | System Implementation | 10 |
| 5.1 | Code | 10 |
| 6. | Results and Discussion | 23 |
| 6.1 | Output | 23 |
| 6.2 | Results Analysis | 28 |
| 6.3 | Discussion | 28 |
| 7 | Conclusion and Future Work | 29 |
| 7.1 | Conclusion | 29 |
| 7.2 | Scope of Future Work | 29 |
| 8 | References | 30 |

**LIST OF FIGURES**

**Figure No Title Page No.**

Fig. 6.1.1 Scatterplot of average salary 23

Fig. 6.1.2 Scatter plot of Loan amount 24

Fig. 6.1.3 Interpreting mined patterns 25

Fig. 6.1.4 Interpreting mined patterns with labels 26

Fig. 6.1.5 Clustered Customers 27

I

**1. INTRODUCTION**

**1.1 BACKGROUND**

Big Data “refers to datasets whose sizes are beyond the ability of typical software tools to capture, store, manage, and analyze them.” The term Big Data Analytics implies methodologies and tools for processing and analyzing the data to produce useful results that cannot be inferred or calculated using other methods in an efficient manner. There are many tools and algorithms available to carry out big data analysis. These methodologies and tools are useful to extract valuable information from large amount of data.

**1.2 PROBLEM STATEMENT**

Every organization or a company collects large amounts of data from various resources. This data collected is used for organizational support and decision making. But this type of complex data is very difficult to process using traditional data processing applications. It needs advanced methods to process this type of data.

**1.3 IMPORTANCE**

Big Data Analytics is very useful for large organizations with tons and tons of data being collected daily. These organizations can implement various big data methodologies and tools to process the enormous amount of data. It can be used by any organization to get valuable information from the data collected which is used for support and decision making.

**2. OVERVIEW AND PLANNING**

**2.1 PROPOSED SYSTEM OVERVIEW**

The proposed system consists of a system which acts as a Master and number of other systems which act as Slaves. The Master system controls all the slave systems and it assigns work to the slave systems. The data collected is stored in the slave systems and various methodologies are implemented by the Master system on the slave systems. The slave systems process the data and produce the results which will be stored in the Master system. This results are displayed by the Master system which are useful for decision making.

**2.2 CHALLENGES**

This system has many challenges which include analysis, capture, storage and visualization of data.

**2.3 ASSUMPTIONS**

The data you collect comes entirely from the past. We can analyze what happened in the past and try to draw trends between actions and decision points and their consequences, based on the data, and we might use that to guess that under similar circumstances, if a similar decision were made, similar outcomes would occur as a result.

**2.4 ARCHITECTURE SPECIFICATIONS**

Hadoop Distributed File System Hadoop is a master/slave based architecture. It primarily consists of a single NameNode, Resource Manager or Yarn, DataNodes and TaskTrackers. In general, the NameNode and Yarn run on the master node where as DataNode and TaskTracker runs on all slave nodes. We also have the provision on creating a separate NameNode and Yarn in case the cluster is huge and load on NameNode is high. NameNode is a master server that manages all the file directories.

NameNode stores metadata related to each file, the metadata mainly consists of file name, location, number of replications etc. NameNode also manages clients and applications access to these files. The DataNode are created on all the slaves usually one per node in the cluster.

DataNodes are responsible for managing storage on the node to which it is attached to. Hadoop Distributed File System exposes a file system namespace and allows user to store their files. Internally, each of these files is divided into several equi-sized blocks, replicated for fixed number of times and then are evenly distributed over the cluster. These blocks are stored in DataNodes with their location stored as a metadata on NameNode. The NameNode executes file system namespace operations like opening files, closing files etc. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system’s clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode. The system design is unique as it prevents any data from being stored on NameNode.

HDFS Architecture Hadoop’s Distributed File System has the traditional hierarchical file organization which is quite similar to most of the existing file systems. A user or application can very well create or delete directories and store files inside it, move file from one director to other or rename it. NameNode maintains a file system namespace, any changes to this namespace is recorded by NameNode. We can specify number of replicas of a file to be maintained in HDFS, this helps in high fault tolerance and reliability.

**2.5 HARDWARE REQUIREMENTS (OPTIMUM REQUIREMENTS)**

A cluster of systems with single Master and many slave systems.

Systems must be connected through LAN

2.13 GHz

100 GB of Disk Drive

4 GB

**2.6 SOFTWARE REQUIREMENTS**

Ubuntu Operating system (Linux-based)

Apache Hadoop

R studio

Java JDK 6 or higher

**2.7 PROJECT SCHEDULE (GANTT CHART)**

**2.8 WORK BREAKDOWN STRUCTURE**

**3. LITERATURE SURVEY AND REVIEW**

**3.1 LITERATURE SURVEY**

The process of the research into complex data basically concerned with the revealing of hidden patterns. **Sagiroglu, S.; Sinanc, D. (20-24 May 2013),”Big Data: A Review”** describe the big data content, its scope, methods, samples, advantages and challenges of Data. The critical issue about the Big data is the privacy and security. Big data samples describe the review about the atmosphere, biological science and research. Life sciences etc .By this paper, we can conclude that any organization in any industry having big data can take the benefit from its careful analysis for the problem solving purpose. Using Knowledge Discovery from the Big data easy to get the information from the complicated data sets. The overall Evaluation describe that the data is increasing and becoming complex. The challenge is not only to collect and manage the data also how to extract the useful information from that collected data. According to the Intel IT Center, there are many challenges related to Big Data which are data growth, data infrastructure, data variety, data visualization, data velocity.

**Mukherjee, A.; Datta, J.; Jorapur, R.; Singhvi, R.; Haloi, S.; Akram, W. (18-22 Dec. 2012) “Shared disk big data analytics with Apache Hadoop”** Big data analytics define the analysis of large amount of data to get the useful information and uncover the hidden patterns. Big data analytics refers to the Mapreduce Framework which is developed by the Google. Apache Hadoop is the open source platform which is used for the purpose of implementation of Google’s Mapreduce Model . In this the performance of SF-CFS is compared with the HDFS using the SWIM by the facebook job traces .SWIM contains the workloads of thousands of jobs with complex data arrival and computation patterns.

**Aditya B. Patel, Manashvi Birla, Ushma Nair (*6-8 Dec. 2012) “*Addressing Big Data Problem Using Hadoop and Map Reduce” reports** the experimental work on the Big data problems. It describe the optimal solutions using Hadoop cluster, Hadoop Distributed File System (HDFS) for storage and Map Reduce programming framework for parallel processing to process large data sets.

**Kanungo, Tapas ; Mount, D.M. ; Netanyahu, N.S. ; Piatko, C.D. ; Silverman, R. ; Wu, A.Y. ,2002.** “**An efficient k-means clustering algorithm: analysis and implementation”.**

This study reveals an efficient implementation of Lloyd's k-means clustering algorithm, called the filtering algorithm. The algorithm is easy to implement and only requires that a k d-tree be built once for the given data points. Efficiency is achieved because the data points do not vary throughout the computation and, hence, this data structure does not need to be recomputed at each stage.

**Zakrzewska, D.; Murlewski, J.Intelligent Systems Design andApplications, 2005.** “**Clustering Algorithms for Bank Customer Segmentation”**

This paper considered three algorithms of cluster analysis: **k-means**, two phase clustering and DBSCAN in bank customer segmentation. The tests showed that all the algorithms have their shortcomings and advantages. **K-means algorithm is very efficient for large multidimensional datasets**, however depends strongly on the choice of input parameter k. It is not recommended in the case of data sets with noise. Two-phase clustering algorithm has a very good performance for data with noise and small amount ofdimensions. In DBSCAN algorithm, wrong choice of input parameters, may resulted in a bad quality.

**3.2 LITERATURE SUMMARY**

Study reveals that K-Means Algorithm always has K clusters. There is always at least one item in each cluster. The clusters are non-hierarchical and they do not overlap. Every member of a cluster is closer to its cluster than any other cluster because closeness does not always involve the ‘center’ of clusters. K-means clustering algorithm is easy to implement and apply on large datasets. It has been successfully used in various fields, including market segmentation, computer vision, geo-statistics, astronomy and agriculture. It often is used as a preprocessing step for other algorithms, for example to find a starting configuration.

**4. SYSTEM DESIGN**

**4.1 HIGH-LEVEL DESIGN**

K-Means

HDFS Input

Map

Split 0

Split 1

K-Means

HDFS Output

Map

Part 0

Sort and merge

Sort and Merge

Sort/merge

Split 2

Reduce

Split 3

Split 4

K-Means

Split 5

Map

**4.2 LOW-LEVEL DESIGN**

Transaction data

**5. SYSTEM IMPLEMENTATION**

Applying K-means algorithm

Required data

Conversion of HDFS to CSV format

Generating reports

Grouping of customers

Using PIG script

Data Cleaning

HDFS

**5.1 CODE**

Data cleaning

sed 's/"//g' account.asc | sed -e '1d' |sed 's/;/,/g'> acccount.csv

Moving the data to HDFS

hadoop dfs -put bank\_project /user/nagamani

Writing PIG script

Loan data: -

Loading and picking only required fields

loan = load '/user/itsupport/data\_berka/loan.csv' using PigStorage(',');

loan\_fields = foreach loan generate $1 as ac\_id,$0 as loan\_id,$3 as amount,$6 as status;

Grouping on account Id

grp\_loan\_ac\_id = group loan\_fields by $0;

Breaking nested bag to store the fields as tuple

grp\_loan\_ac\_id\_flatten = foreach grp\_loan\_ac\_id generate FLATTEN(loan\_fields);

Removing the header

filtered\_grp = filter grp\_loan\_ac\_id\_flatten by $3 != 'status';

Storing the short file back to HDFS

store filtered\_grp into '/bank\_project/loan\_required\_out' using PigStorage(',');

Client data:

Loading and picking only required fields

client = load '/user/itsupport/data\_berka/client.csv' using PigStorage(',') AS (Client\_id:int,dob:chararray,dist\_id:int) ;

client\_fields = foreach client generate $0 as client\_id,$2 as district\_id,$1 as birthday\_n\_sex;

Grouping on client\_id

grp\_client\_id = group client\_fields by $0;

Breaking nested bag to store the fields as tuple

grp\_client\_flat = FOREACH grp\_client\_id GENERATE FLATTEN(client\_fields);

Removing the header

B = filter grp\_client\_flat by $2 != 'birth\_number;

Calling Java function to parse the date and finding age and sex of the customer

age = foreach B generate $0,$1, bank.Age\_calculator(birthday\_n\_sex) ;

store age into '/bank\_project/age\_required\_out' using PigStorage(',');

Transaction data:

Loading and picking only required fields

transaction = load '/user/itsupport/data\_berka/transaction.csv' using PigStorage(',') as (trans\_id:int,ac\_id:int,date:chararray,type:chararray,operation:chararray,amount:int,bal:int,k\_sym:chararray,bank:int,account:int);

transaction\_fields = foreach transaction generate $1 as ac\_id,$2 as date\_of\_transaction,$3 as transaction\_type,$5 as amount,$6 as bal\_post\_trnsaction;

Picking only transaction of last one year

filtered\_trans = filter transaction\_fields by (int)SUBSTRING($1,0,2) > 97;

grp\_ac = group filtered\_trans by $0;

Summing up the transaction carried out by the user in last one year

MAX\_grp\_ac = FOREACH grp\_ac GENERATE group, SUM(filtered\_trans.$3),SUM(filtered\_trans.$4);

store MAX\_grp\_ac into '/bank\_project/transaction\_left\_bal\_required\_out' using PigStorage(',');

Card details data:-

card = load '/user/itsupport/data\_berka/card.csv' using PigStorage(',') ;

card\_fields = foreach card generate $1 as disposition\_id,$2 as card\_type;

grp\_card\_disp\_id = group card\_fields by $0;

flatten\_card = foreach grp\_card\_disp\_id generate FLATTEN(card\_fields);

filtered\_card = filter flatten\_card by card\_type != 'type';

store filtered\_card into '/bank\_project/card\_required\_out' using PigStorage(',');

District data:

REGISTER alljars/pig\_substractjar.jar

district = load '/user/itsupport/data\_berka/district.csv' using PigStorage(',') AS (dist\_id:int,dist\_name:chararray,region:chararray,no\_inhabs:long,mun\_499:int,mun\_1999:int,mun\_10k:int,mun\_more:int,no\_of\_cities:int,no\_of\_urban\_inhabs:double,avg\_sal:int,unemp\_95:double,unemp\_96:double,entre\_ratio:int);

district\_fields = foreach district generate $0 as district\_id,$1 as district\_name,$2 as region,$10 as avg\_salary,$11 as unemp\_rate\_95,$12 as unemp\_rate\_96,$13 as entrepreneur\_per\_1000;

grp\_dist\_id = group district\_fields by $0;

MAX\_grp\_dist = FOREACH grp\_dist\_id GENERATE group,FLATTEN(district\_fields);

B = filter MAX\_grp\_dist by unemp\_rate\_95 > 0.0 AND unemp\_rate\_96 > 0.0;

unem\_percentage = foreach B generate $1, district\_name,avg\_salary,bank.substract(unemp\_rate\_95,unemp\_rate\_96),entrepreneur\_per\_1000 ;

store unem\_percentage into '/bank\_project/district\_required\_out' using PigStorage(',');

Disposition data:

disposition = load '/user/itsupport/data\_berka/disposition.csv' using PigStorage(',') ;

disposition\_fields = foreach disposition generate $2 as ac\_id,$0 as disposition\_id,$3 as disposition\_type,$1 as client\_id;

grp\_disposition\_disp\_id = group disposition\_fields by $1;

flatten\_disposition\_disp\_id = foreach grp\_disposition\_disp\_id generate FLATTEN(disposition\_fields);

filtered\_disposition\_disp\_id = filter flatten\_disposition\_disp\_id by disposition\_type !=

'type';

Joining all the data :-

Client\_age = load '/bank\_project/age\_required\_out' using PigStorage(',') AS (client\_id:int,dist\_id:int,age:double,sex:chararray);

card\_type = load '/bank\_project/card\_required\_out' using PigStorage(',') AS (disp\_id:int,type:chararray);

transaction\_sum = load '/bank\_project/transaction\_left\_bal\_required\_out' using PigStorage(',') AS (ac\_id:int,trans\_sum:long,bal\_sum:long);

loan\_status = load '/bank\_project/loan\_required\_out' using PigStorage(',') AS (ac\_id:int,loan\_id:int,amount:int,status:chararray);

district\_info = load '/bank\_project/district\_required\_out' using PigStorage(',') AS (district\_id:int,dist\_name:chararray,avg\_sal:int,unemprate:double,entrepreneur:int);

join\_disp\_client = join filtered\_disposition\_disp\_id by $3,Client\_age by $0;

join\_disp\_client\_card = join join\_disp\_client by $1,card\_type by $0;

join\_disp\_client\_card\_district = join join\_disp\_client\_card by $5,district\_info by $0;

join\_disp\_client\_card\_district\_trans\_loan = join join\_disp\_client\_card\_district by $0,transaction\_sum by $0,loan\_status by $0;

pick\_fields = foreach join\_disp\_client\_card\_district\_trans\_loan generate $0 as ac\_id,$2 as disp\_type,$6 as age,$7 as sex,$9 as card\_type,$11 as dist\_name,$12 as avg\_sal,$13 as unemp\_rate,$14 as no\_of\_entre,$16 as transaction\_sum,$20 as loan\_amount,$21 as loan\_status;

store pick\_fields into '/bank\_project/combined\_out' using PigStorage(',');

R Commands:

Using the intermediate output of Hadoop i.e. output file “/bank\_project/combined\_out” and build clusters on the variable using R.

Setting Hadoop variables for Hadoop in R environment

Sys.setenv(JAVA\_HOME="/home/nagamani/java")

Sys.setenv(HADOOP\_HOME="/home/nagamani/hadoop")

Sys.setenv(HADOOP\_CMD="/home/nagamani/hadoop/bin/hadoop")

Sys.setenv(HADOOP\_STREAMING="/home/nagamani/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.2.0.jar")

Loading RHadoop packages

library(rmr2)

library(rhdfs)

hdfs.init()

Setting Hadoop root path and reading files from HDFS

hdfs.root <- '/bank\_project'

hdfs.data <- file.path(hdfs.root, 'combined\_out/part-r-00000')

final\_bank\_data <- hdfs.read.text.file(hdfs.data)

content<-hdfs.read.text.file(hdfs.data)

clickpath<-read.table(textConnection(content),sep=",")

Naming all the columns fetched from HDFS

colnames(clickpath) <-c("ac\_id","disposal\_type","age","sex","card\_type","dist","avg\_sal","unemp\_rate","entrepreneur\_no","trans\_sum","loan\_amount","loan\_status")

Data cleaning and pre-processing

Checking structure of the fetched data

str(clickpath)

summary(clickpath)

List of rows with missing values

clickpath[!complete.cases(clickpath),]

List of columns with missing values

clickpath[,!complete.cases(clickpath)]

Omitting missing values

clickpath <- na.omit(clickpath,na.action=TRUE)

Data reduction and projection

selecting only numerical data and removing ac\_id column

mydata <- clickpath[,c(3,7:11)]

First check the complete set of components for outliers

boxplot(mydata)

Outlier in avg\_sal

boxplot(mydata[,c(2)])

plot(mydata[,c(2)])

Outlier in unemp\_rate

boxplot(mydata[,c(3)])

Defining function to replace outliers

library(data.table)

outlierReplace = function(dataframe, cols, rows, newValue = NA) {

if (any(rows)) {

set(dataframe, rows, cols, newValue)

}

}

outlierReplace(clickpath, "unemp\_rate", which(mydata$unemp\_rate > 1.5), 1.5)

fivenum(mydata$unemp\_rate)

Outlier in loan\_amount

boxplot(mydata[,c(6)])

plot((mydata[,c(6)]))

mydata <- scale(mydata[,1:6])

Calculating variance and storing at the first index in wss

wss <- (nrow(mydata)-1)\*sum(apply(mydata,2,var))

Choosing the data mining algorithm(s)

k-means algorithm for this clustering is used.

for(i in 2:15)wss[i]<- sum(fit=kmeans(mydata,centers=i,15)$withinss

Plot each iteration to display the elbow graph

plot(1:15,wss,type="b",main="15 clusters",xlab="no. of cluster",ylab="with clsuter sum of squares")

Searching for patterns of interest in a particular representational form

fit <- kmeans(mydata,3)

fit$withinss

checking betweenss i.e. the inter cluster distance between cluster

fit$betweenss

fit$size

Interpreting mined patterns

plot(mydata,col=fit$cluster,pch=15)

points(fit$centers,col=1:8,pch=3)

library(cluster)

library(fpc)

plotcluster(mydata,fit$cluster)

points(fit$centers,col=1:8,pch=16)

clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE, labels=2, lines=0)

checking mean for each object in each cluster

mydata <- clickpath[,c(3,7:12)]

mydata <- data.frame(mydata,fit$cluster)

cluster\_mean <- aggregate(mydata[,1:8],by = list(fit$cluster),FUN = mean)

Java source code:

The date of birth (DOB) information is in the format yymm+50dd(for female)yymmdd(for male).

1. For calculating the age from the above date format a UDF is written(in java) and its jar is registered with PIG execution engine.

package AgeCalc;

import java.io.DataInput;

import java.io.DataOutput;

import java.io.IOException;

import java.util.Iterator;

import java.util.List;

import java.util.Scanner;

import java.lang.\*;

import org.apache.pig.EvalFunc;

import org.apache.pig.backend.executionengine.ExecException;

import org.apache.pig.data.Tuple;

import org.apache.pig.builtin.\*;

import org.apache.pig.data.TupleFactory;

import org.apache.pig.impl.util.\*;

import org.joda.time.LocalDate;

import org.joda.time.Period;

import org.joda.time.PeriodType;

public class AgeCalculator extends EvalFunc<Tuple>{

private static final PeriodType PeriodType = null;

Tuple output=TupleFactory.getInstance().newTuple(2);

@Override

public Tuple exec(Tuple input) throws IOException {

try

{

String dob=(String)input.get(0);

String s[]=new String[3];

int k=0;

for(int i=0;i<6;i=i+2)

{

s[k]=dob.substring(i, i+2);

k++;

}

System.out.println("s[0]"+s[0]);

int year=Integer.parseInt(s[0]);

int mon=Integer.parseInt(s[1]);

int day=Integer.parseInt(s[2]);

String gender=null;

int month=mon;

if(mon>12)

{

gender="F";

month=mon-50;

}

else

{

gender="M";

}

s[0]="19"+s[0];

year=Integer.parseInt(s[0]);

LocalDate birthdate = new LocalDate (year,month,day);//Birth date

System.out.println("birthdate"+birthdate);

LocalDate CompDate = new LocalDate(1999,1,1); //Today's date

//System.out.println("birthdate"+now);

Period period = new Period(birthdate, CompDate, PeriodType.yearMonthDay());

//Now access the values as below

//System.out.println(

double age=period.getYears();//+" Years");

//System.out.println(period.getMonths()+" months");

//System.out.println(period.getDays()+" days");

System.out.println("gender:"+gender);

output.set(0,age);

output.set(1,gender);

}catch(Exception e)

{

e.printStackTrace();

}

return output;

}

}

2. For separating the age from the tuple obtained by executing the Agecalc.jar following jar file is written and executed.

import java.util.Scanner;

import java.lang.\*;

import org.apache.pig.EvalFunc;

import org.apache.pig.data.DataType;

import org.apache.pig.backend.executionengine.ExecException;

import org.apache.pig.data.Tuple;

import org.apache.pig.builtin.\*;

import org.apache.pig.data.TupleFactory;

import org.apache.pig.impl.util.\*;

public class AgeExtract extends EvalFunc<Double> {

double f1;

//Tuple output=TupleFactory.getInstance().newTuple(2);

public Double exec(Tuple input) throws IOException {

// TODO Auto-generated method stub

try

{

f1 = (double)input.get(0);

//Character f2 = (Character)input.get(1);

//res = f2 - f1;

//output.set(0, res);

}

catch(Exception e){

e.printStackTrace();

}

return f1;

}

3. For separating the gender information from the tuple obtained by executing the Agecalc.jar following jar file is written and executed.

import org.apache.pig.EvalFunc;

import org.apache.pig.data.DataType;

import org.apache.pig.backend.executionengine.ExecException;

import org.apache.pig.data.Tuple;

import org.apache.pig.builtin.\*;

import org.apache.pig.data.TupleFactory;

import org.apache.pig.impl.util.\*;

public class GenderExtract extends EvalFunc<Character> {

Character f1;

//Tuple output=TupleFactory.getInstance().newTuple(2);

public Character exec(Tuple input) throws IOException {

// TODO Auto-generated method stub

try

{

f1 = (Character)input.get(1);

//Character f2 = (Character)input.get(1);

//res = f2 - f1;

//output.set(0, res);

}

catch(Exception e){

e.printStackTrace();

}

return f1;

}

}

4. For calculating the unemployment rate difference between the two successive years(1995 and 1996) this file is written and executed.

import org.apache.pig.EvalFunc;

import org.apache.pig.data.DataType;

import org.apache.pig.backend.executionengine.ExecException;

import org.apache.pig.data.Tuple;

import org.apache.pig.builtin.\*;

import org.apache.pig.data.TupleFactory;

import org.apache.pig.impl.util.\*;

public class UnempDiff1 extends EvalFunc<Double> {

double res;

//Tuple output=TupleFactory.getInstance().newTuple(2);

public Double exec(Tuple input) throws IOException {

// TODO Auto-generated method stub

try

{

double f1 = (double)input.get(0);

double f2 = (double)input.get(1);

res = f2 - f1;

//output.set(0, res);

}

catch(Exception e){

e.printStackTrace();

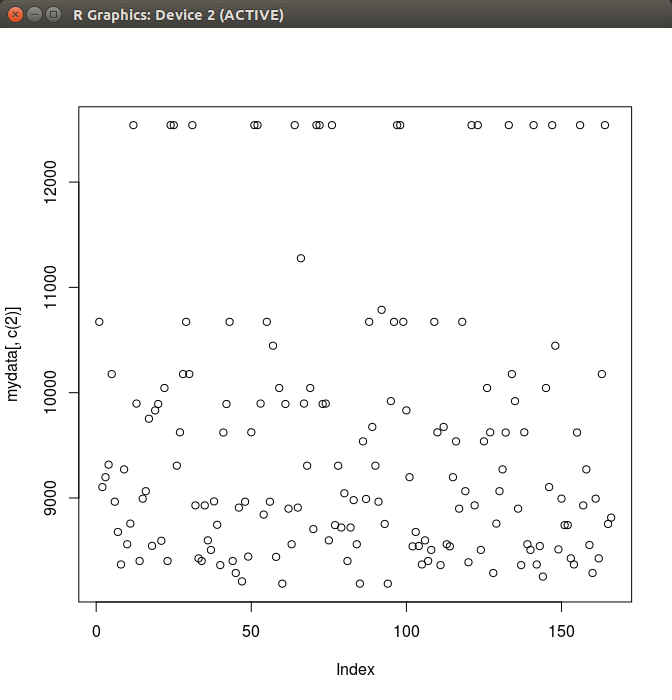
}

return res;

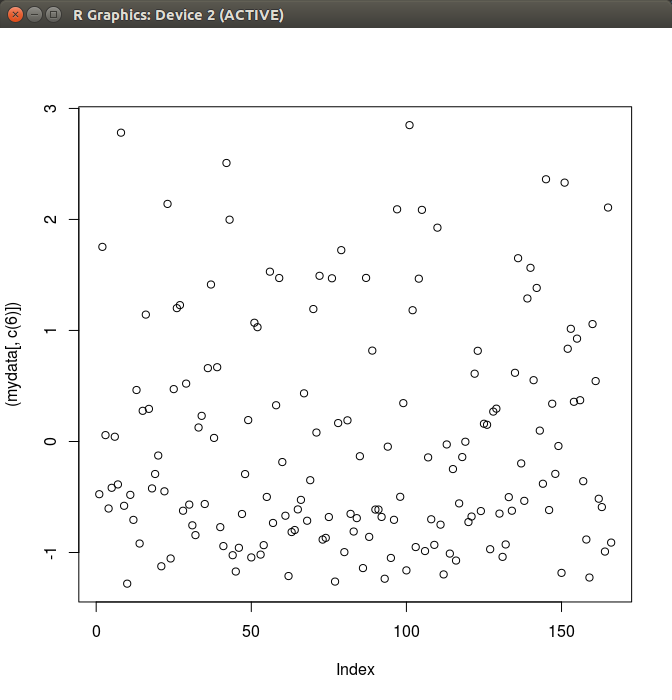
}

**6. RESULTS AND DISCUSSION**

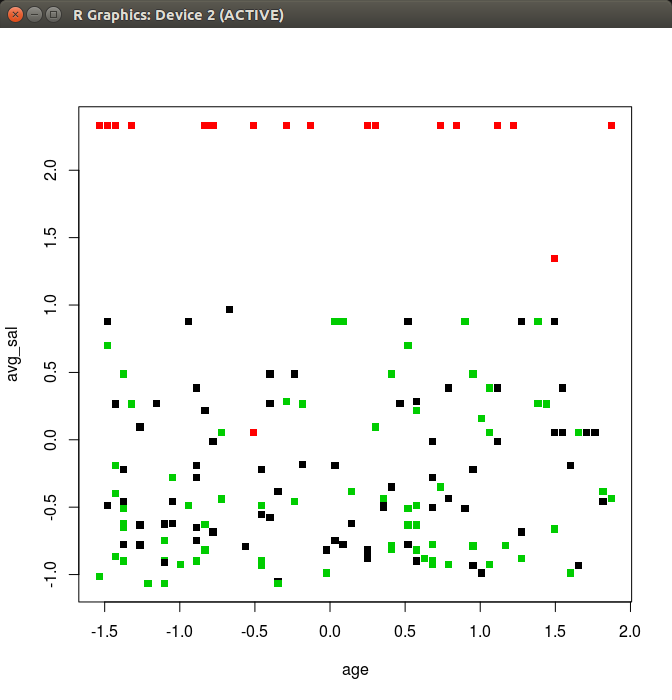
**6.1 OUTPUT**



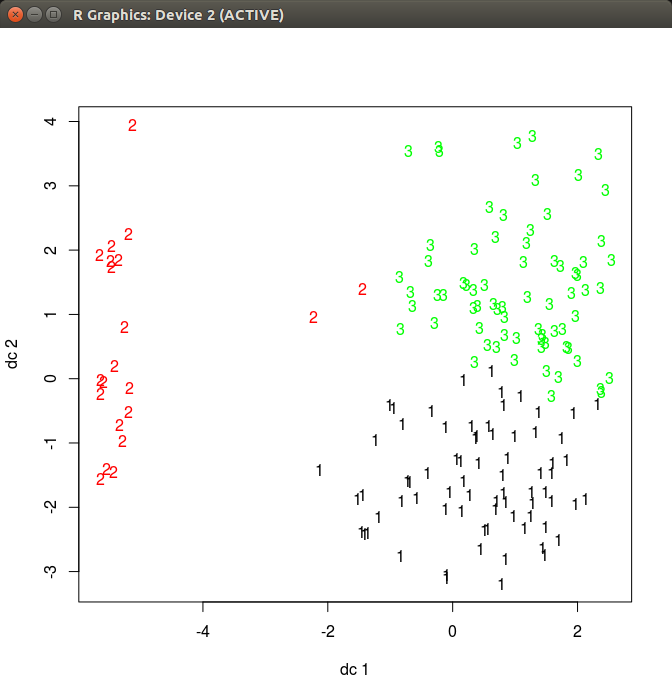
*Figure 1: Scatterplot of average salary*



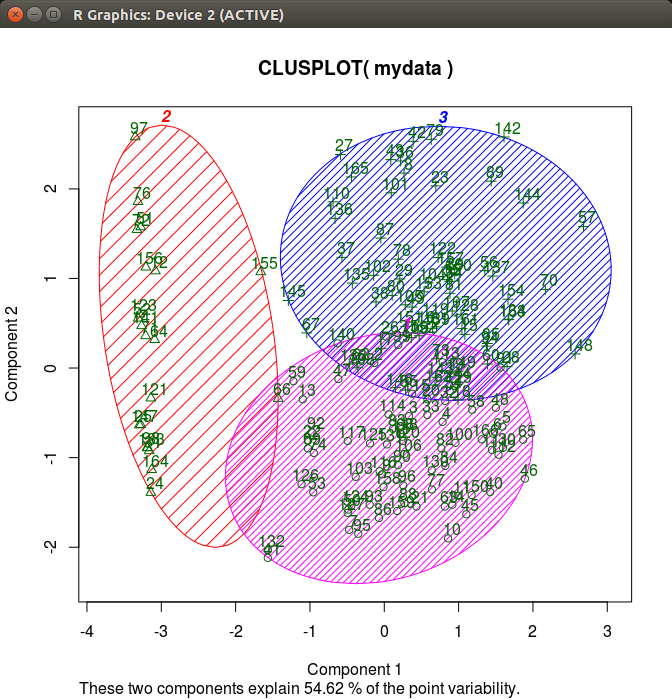
*Figure 2: Scatter plot of Loan amount*



*Figure 3: Interpreting mined patterns*



*Figure 4: Interpreting mined patterns with labels*



*Figure 5: Clustered Customers*

**6.2 RESULTS ANALYSIS**

Now, we have three categories of customers.

The bank can prepare separate plans for three categories as per their business goals.

**6.3 DISCUSSION**

The two key features of k-means which make it efficient are often regarded as its biggest drawbacks:

The number of clusters k is an input parameter: an inappropriate choice of k may yield poor results. That is why, when performing k-means, it is important to run diagnostic checks for determining the number of clusters in the data set.

Convergence to a local minimum may produce counterintuitive ("wrong") results.

Applications

k- Means clustering in particular when using heuristics such as Lloyd's algorithm is rather easy to implement and apply even on large data sets. As such, it has been successfully used in various topics, including market segmentation, computer vision, geo statistics, astronomy and agriculture. It often is used as a preprocessing step for other algorithms, for example to find a starting configuration.

Vector quantization

k- Means originates from signal processing, and still finds use in this domain. For example in computer graphics, color quantization is the task of reducing the color palette of an image to a fixed number of colors k. The k-means algorithm can easily be used for this task and produces competitive results. Other uses of vector quantization include non-random sampling, as k-means can easily be used to choose k different but prototypical objects from a large data set for further analysis.

Feature learning

k- Means clustering has been used as a feature learning (or dictionary learning) step, in either (semi-)supervised learning or unsupervised learning. The basic approach is first to train a k-means clustering representation, using the input training data (which need not be labelled). Then, to project any input datum into the new feature space.

This use of k-means has been successfully combined with simple, linear classifiers for semi-supervised learning in NLP (specifically for named entity recognition) and in computer vision. On an object recognition task, it was found to exhibit comparable performance with more sophisticated feature learning approaches such as auto encoders and restricted Boltzmann machines.

**7. CONCLUSION AND FUTURE WORK**

**7.1 CONCLUSION**

This project analyses the Berka bank customer transaction data sets to predict the customer quality and categorize a customer into three categories:

**Excellent:** Customers whose record is good with the bank

**Good:** Customers who have average earning with a good record till now

**Risky:** Customers who are under debt of bank or who has not paid the loan on time

**7.2 SCOPE OF FUTURE WORK**

The new applications are generating vast amount of data in structured and unstructured form. Big data is able to process and store that data and probably in more amounts in near future. Hopefully, Hadoop will get better. New technologies and tools that have ability to record, monitor measure and combine all kinds of data around us, are going to be introduced soon. We will need new technologies and tools for anonymizing data, analysis, tracking and auditing information, sharing and managing, our own personal data in future. So many aspects of life health, education, telecommunication, marketing, sports and business etc that manages big data world need to be polished in future.

**8. REFERENCES**

Conference Papers

Sagiroglu, S.; Sinanc, D. (20-24 May 2013),”Big Data: A Review”.

Mukherjee, A.; Datta, J.; Jorapur, R.; Singhvi, R.; Haloi, S.; Akram, W. (18-22 Dec. 2012) “Shared disk big data analytics with Apache Hadoop”.

Aditya B. Patel, Manashvi Birla, Ushma Nair (6-8 Dec. 2012) “Addressing Big Data Problem Using Hadoop and Map Reduce”.

Zakrzewska, D.; Murlewski, J.Intelligent Systems Design and Applications, 2005. “Clustering Algorithms for Bank Customer Segmentation”.

Kanungo, Tapas ; Mount, D.M. ; Netanyahu, N.S. ; Piatko, C.D. ; Silverman, R. ; Wu, A.Y. ,2002. “An efficient k-means clustering algorithm: analysis and implementation”.

Web

<http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6567202>

<http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6507520>

<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1578784>

<http://www.ijarcsse.com/docs/papers/Volume_3/10_October2013/V3I10-0415.pdf>

<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1017616>