CS5542 BIG DATA ANALYTICS AND APPS

Increment -4 Report (04/29/2016)

Project Group -7

By

Abhiram Ampabathina (1)

Harshini Medikonda (14)

Hirenbhai Harshadbhai Shah(27)

Dinesh Reddy (19)

I.INTRODUCTION:

The heart rate of a person depends on age, gender, daily physical activity, mental stress and many other activities/conditions. Furthermore, there is no proper equipment that can keep a track of heart beat rate. We intend to do a system that can collect the person's daily heart rate activity, store it in a database, analyze the heart rate and the activity the person is performing. Moreover, the application can analyze the data and recommend mental or physical activities to be performed by the user to keep the heart rate optimal. It can also suggest the timings of the abnormal heart rate. All these would give a clear idea of the medical condition of the user and the better usage of it can help in a longer life.

II.PROJECT GOAL AND OBJECTIVES:

OVERALL GOAL:

The goal of the project is to build a system that can take care of the user's health. This heart rate system is an android application which he can view even through the smart watch. This application works with the heart rate sensor embedded in the smart watch. It can observe the patterns of the heart rate and determine the health condition. It recommends the user with the necessary physical and mental activity.

SPECIFIC OBJECTIVE:

The objectives that would be achieved are as follows:

- Collect the heart rate and step count of the user
- Store the heart rate in regular intervals
- Get the heart rate onto HDFS per day basis
- Analyze it using machine learning algorithms.
- Notifying the health conditions using smart watch and smart phone
- Recommend the activities to be done by the user.
- Have a medical record, convenient and cost efficient.

SPECIFIC FEATURES:

The specific features designed in the project are:

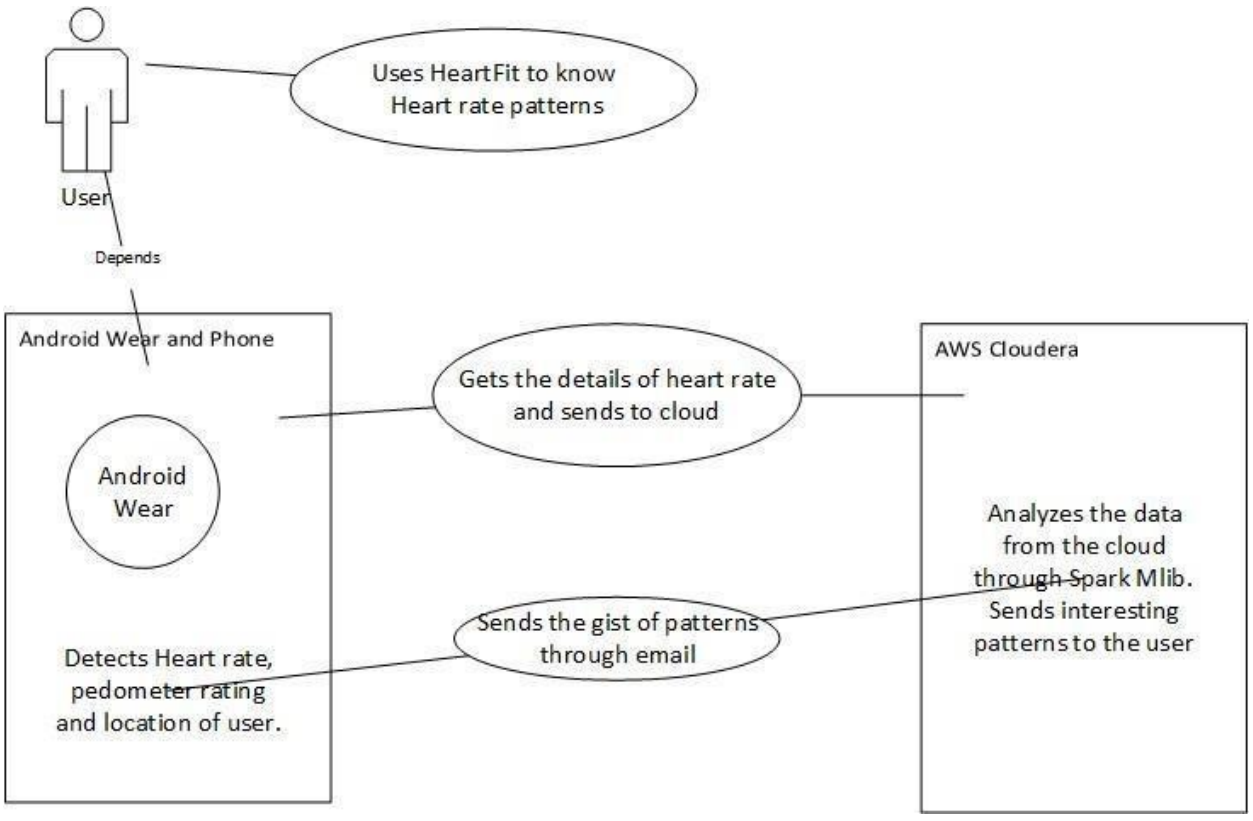
- Heart beat analysis
- Step count analysis
- Notification of current health condition
- Recommendation of health care

SIGNIFICANCE:

The main significance of this application is it is a system that is required for every person in their daily life. It is a trending smart application which makes the life easier. It is beneficial and becomes a part of the life in the upcoming years.

III. PROJECT PLAN:

1. Stories : Scenario & Use case specification


User
Uses HeartFit to know Heart rate patterns
Depends
Android Wear and Phone
Android Wear
Detects Heart rate, pedometer rating and location of user.
Gets the details of heart rate and sends to cloud
Sends the gist of patterns through email
AWS Cloudera
Analyzes the data from the cloud through Spark Mlib. Sends interesting patterns to the user

FEATURE DESIGN:

The application is designed to have the following features

1. Ability to run the application background all the time.

2. Ability to read heart rate accurately.

3. Ability to read the location of the user accurately.

4. Ability to send the information to cloud DB when internet is connected.

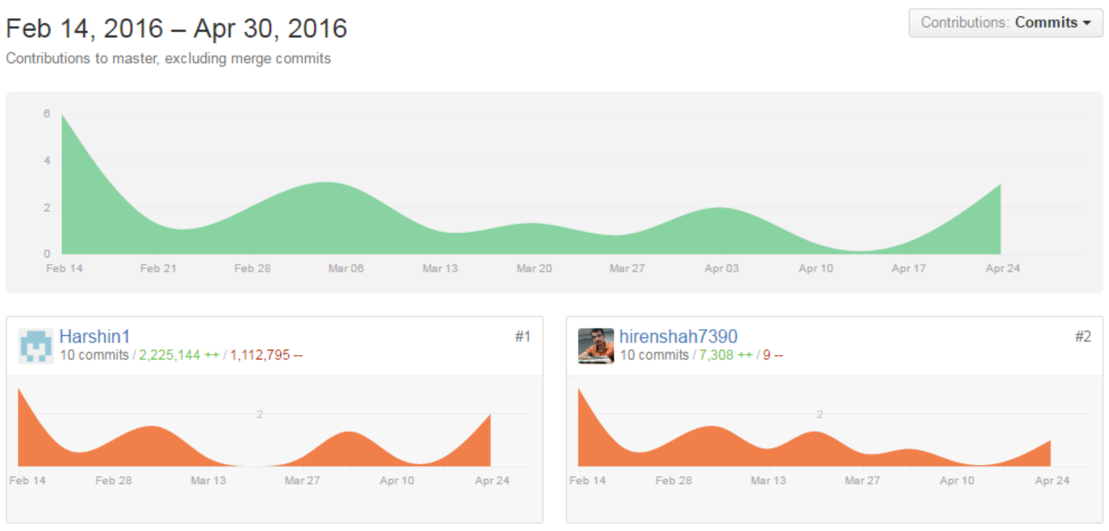
The system performs clustering based on the heart rate data of the user and recommends the user of the exercises to follow to stay healthy.

Each feature is designed accordingly to the specifications it should perform.

FEATURE IMPLEMENTATION:

The application is designed to work continuously in the background. For this we added permission in android activity.xml . So, that the phone or the wear supports to run the application in background. To read user heart rate, we added heart rate sensor as a dependency to the application dependency list. This will accurately detect user's heart rate and return it to mobile. Our application will also send location details of user. For this we included GPS Location dependency plugin to android. This will send us the co-ordinates of the user's location. We also need to make sure the device is connected to internet in order to send the user's details to cloud DB, furtherly to analyze the data in cloudera.

2. PROJECT TIMELINES, MEMBERS, TASK RESOPNSIBILITY:



Feb 14, 2016 – Apr 30, 2016

Contributions to master, excluding merge commits

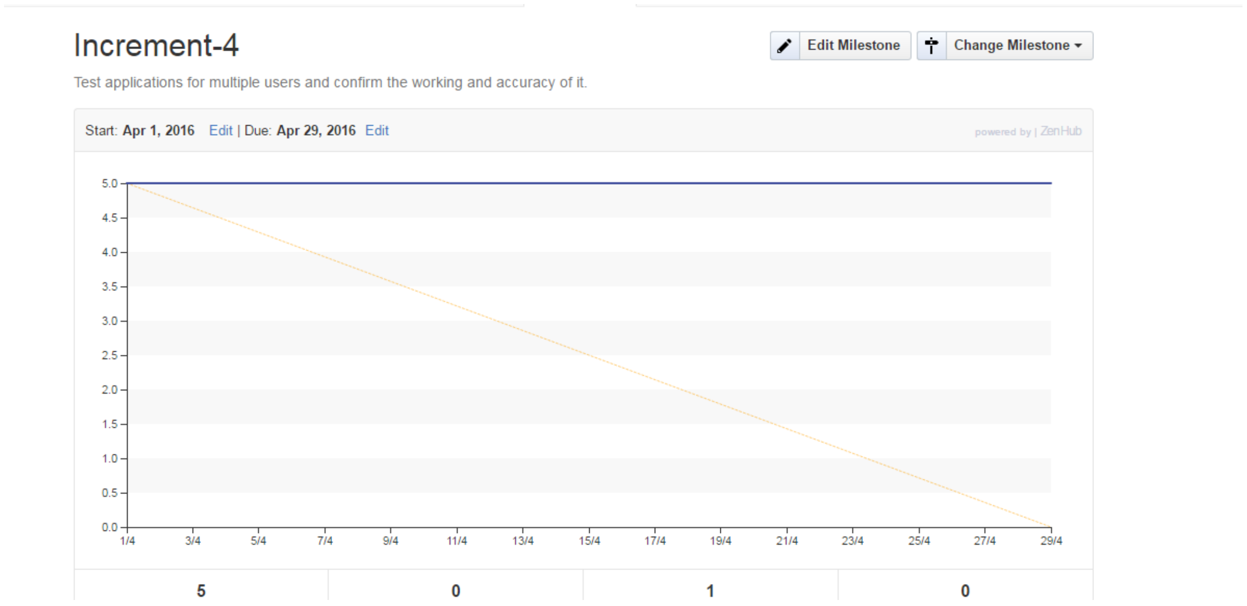
Harshin1 #1

10 commits / 2,225,144 ++ / 1,112,795 --

hirenshah7390 #2

10 commits / 7,308 ++ / 9 --

BURNDOWN CHARTS:



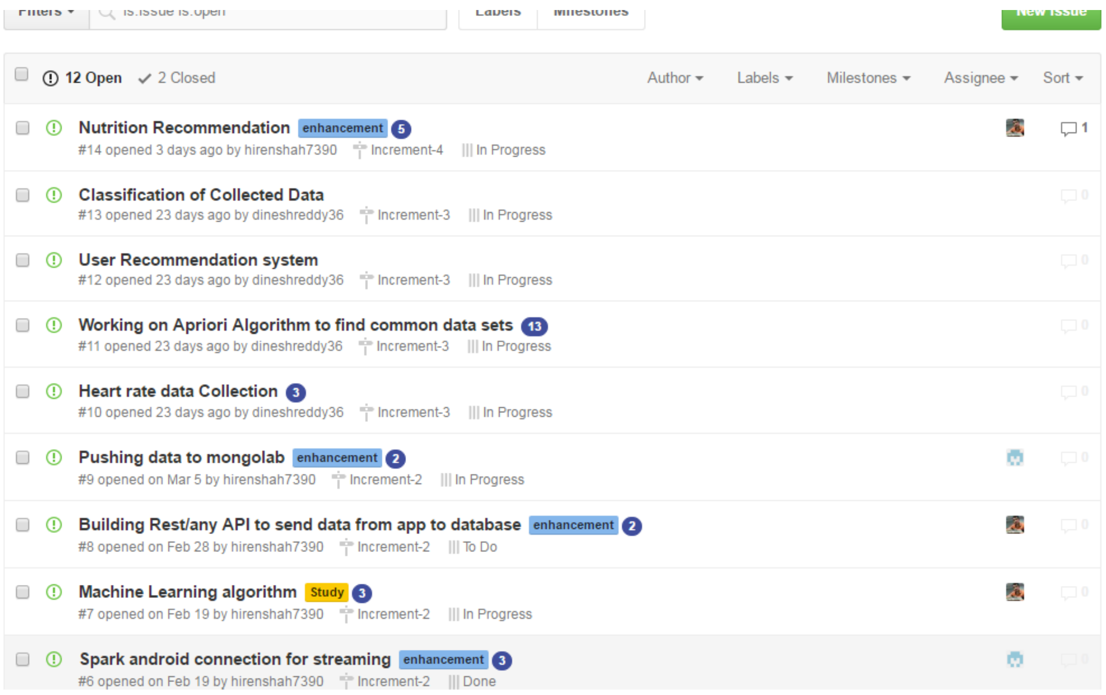
Increment-4

Test applications for multiple users and confirm the working and accuracy of it.

Start: **Apr 1, 2016** Edit | Due: **Apr 29, 2016** Edit

5	0	1	0

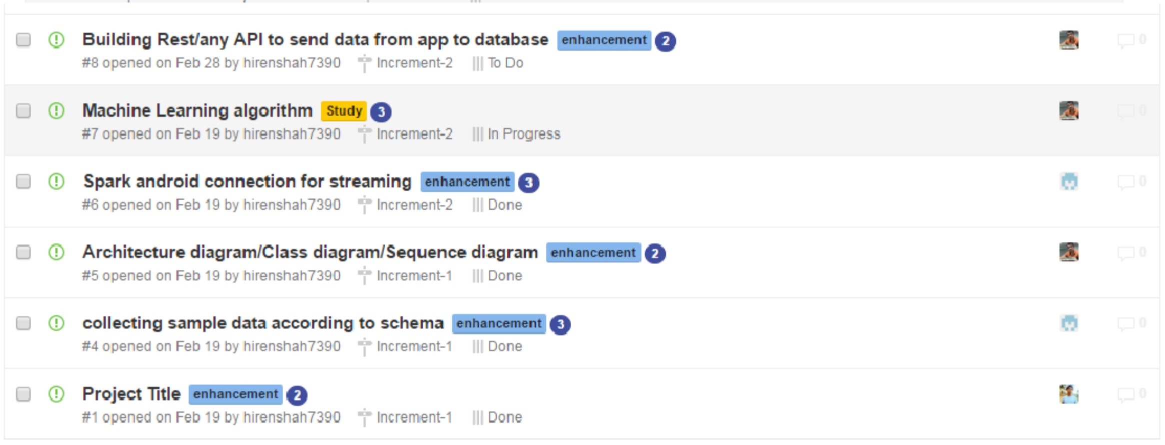
ISSUES:



Filters ▾ is:issue is:open | Labels | Milestones | New issue

12 Open ✓ 2 Closed — Author ▾ Labels ▾ Milestones ▾ Assignee ▾ Sort ▾

- Nutrition Recommendation `enhancement` 5 — 💬 1
  #14 opened 3 days ago by hirenshah7390 — Increment-4 — In Progress
- Classification of Collected Data — 💬 0
  #13 opened 23 days ago by dineshreddy36 — Increment-3 — In Progress
- User Recommendation system — 💬 0
  #12 opened 23 days ago by dineshreddy36 — Increment-3 — In Progress
- Working on Apriori Algorithm to find common data sets 13 — 💬 0
  #11 opened 23 days ago by dineshreddy36 — Increment-3 — In Progress
- Heart rate data Collection 3 — 💬 0
  #10 opened 23 days ago by dineshreddy36 — Increment-3 — In Progress
- Pushing data to mongolab `enhancement` 2 — 💬 0
  #9 opened on Mar 5 by hirenshah7390 — Increment-2 — In Progress
- Building Rest/any API to send data from app to database `enhancement` 2 — 💬 0
  #8 opened on Feb 28 by hirenshah7390 — Increment-2 — To Do
- Machine Learning algorithm `Study` 3 — 💬 0
  #7 opened on Feb 19 by hirenshah7390 — Increment-2 — In Progress
- Spark android connection for streaming `enhancement` 3 — 💬 0
  #6 opened on Feb 19 by hirenshah7390 — Increment-2 — Done



- Building Rest/any API to send data from app to database `enhancement` 2 — 💬 0
  #8 opened on Feb 28 by hirenshah7390 — Increment-2 — To Do
- Machine Learning algorithm `Study` 3 — 💬 0
  #7 opened on Feb 19 by hirenshah7390 — Increment-2 — In Progress
- Spark android connection for streaming `enhancement` 3 — 💬 0
  #6 opened on Feb 19 by hirenshah7390 — Increment-2 — Done

- Architecture diagram/Class diagram/Sequence diagram `enhancement` 2 — 💬 0
  #5 opened on Feb 19 by hirenshah7390 — Increment-1 — Done
- collecting sample data according to schema `enhancement` 3 — 💬 0
  #4 opened on Feb 19 by hirenshah7390 — Increment-1 — Done
- Project Title `enhancement` 2 — 💬 0
  #1 opened on Feb 19 by hirenshah7390 — Increment-1 — Done

IV. FOURTH INCREMENT REPORT:

EXISTING API:

- MongoLab API:

https://api.mongolab.com/api/1/databases/my-db/collections?apiKey=myAPIKey

This API is used to store the heart rate and step count in the database and get the heart rate from the database.

- Heart Rate and Step counter Sensor, Others too

The heart rate and step counter sensors are embedded in the smart watch which can be used to get the data. This data is sent to the Spark HDFS system on a per-day basis.

We also collected accelerometer sensor and other sensors to get the accurate location of the person during that time. We are applying machine learning on the collected sensor data to match the patterns and get the user data.

- Java Mail API

This is used to send an email of the results to the user as an email.

- Twitter Streaming API

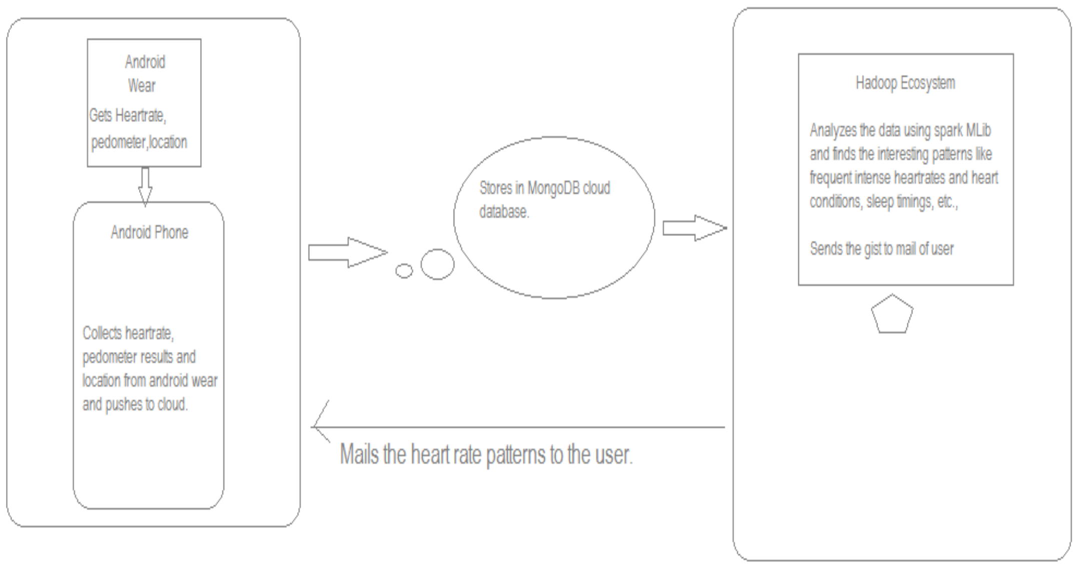
The tweets are collected as per the keywords heart rate, fitness, pulse, health and these are analyzed.

- Nutrients Database

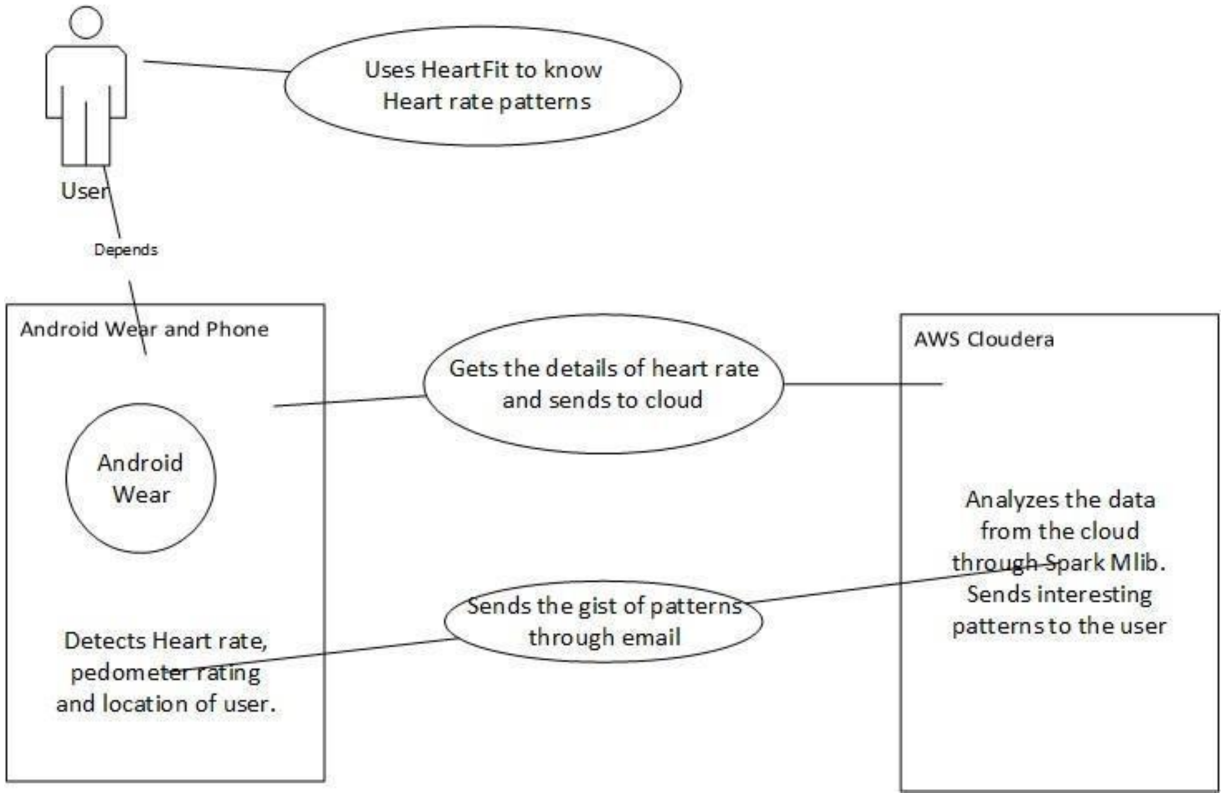
The nutrients data base we have taken for food recommendation has all the food data such as type of food, calories, proteins, nutrients, etc. This is being used to recommend the user to tell him the similar items he can consume based on his preference.

DESIGN OF FEATURES:

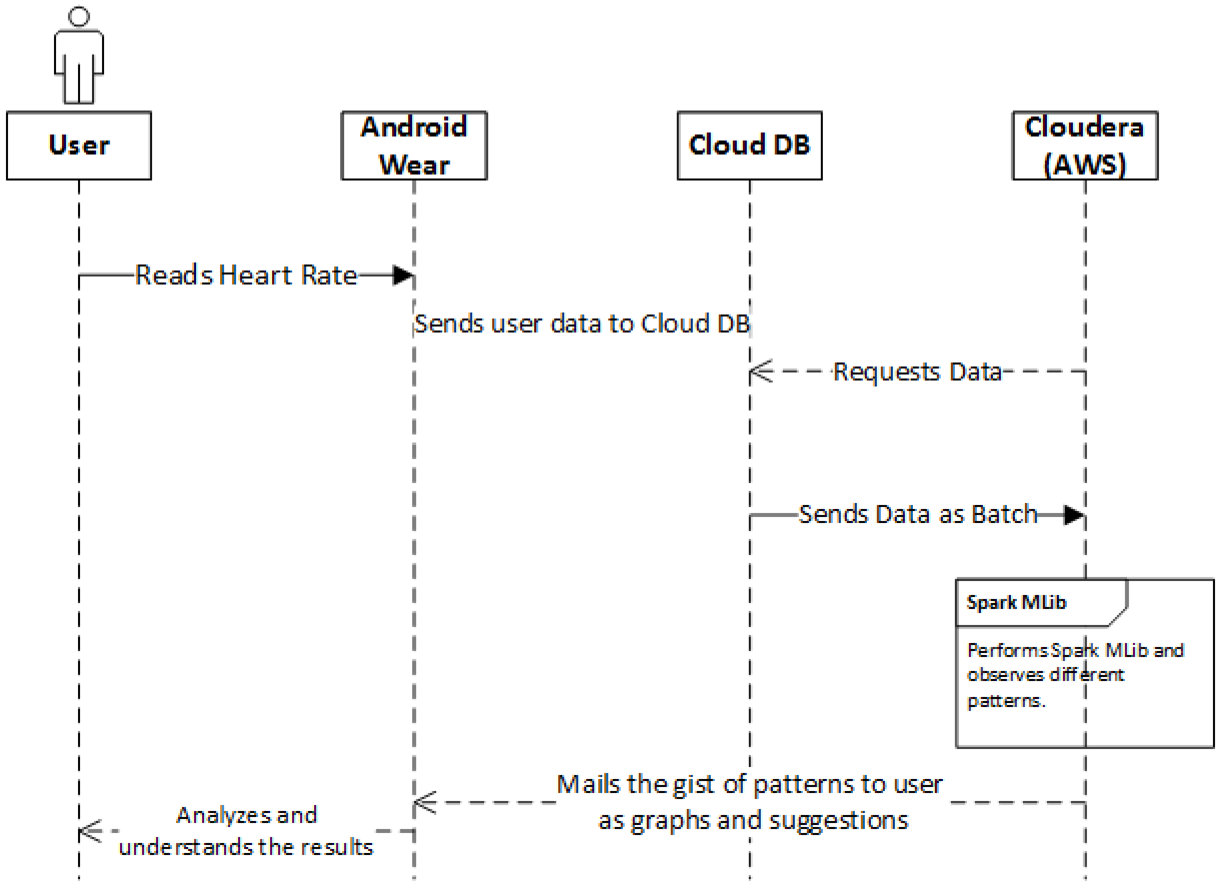
ARCHITECHTURE DIAGRAM:


Android Wear
Gets Heartrate, pedometer,location
Android Phone
Collects heartrate, pedometer results and location from android wear and pushes to cloud.
Stores in MongoDB cloud database.
Hadoop Ecosystem
Analyzes the data using spark MLib and finds the interesting patterns like frequent intense heartrates and heart conditions, sleep timings, etc.,
Sends the gist to mail of user
Mails the heart rate patterns to the user.

CLASS DIAGRAM:


Uses HeartFit to know Heart rate patterns
User
Depends
Android Wear and Phone
Android Wear
Detects Heart rate, pedometer rating and location of user.
Gets the details of heart rate and sends to cloud
Sends the gist of patterns through email
AWS Cloudera
Analyzes the data from the cloud through Spark Mlib. Sends interesting patterns to the user

SEQUENCE DIAGRAM:

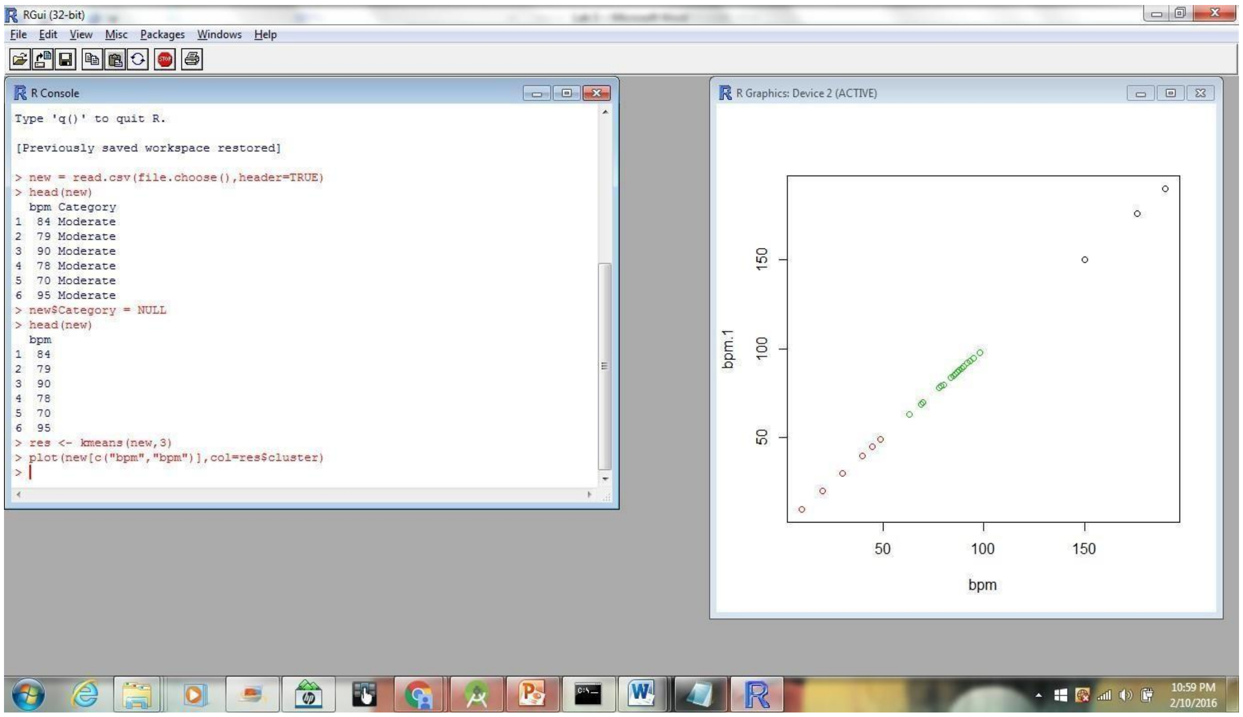

User
Android Wear
Cloud DB
Cloudera (AWS)
Reads Heart Rate
Sends user data to Cloud DB
Requests Data
Sends Data as Batch
Spark MLib
Performs Spark MLib and observes different patterns.
Mails the gist of patterns to user as graphs and suggestions
Analyzes and understands the results

SPARK/MACHINE LEARNING ALGORITHMS:

In this application, we are planning to use the below machine learning algorithms to analyze the data.

1. **K-means Clustering:**

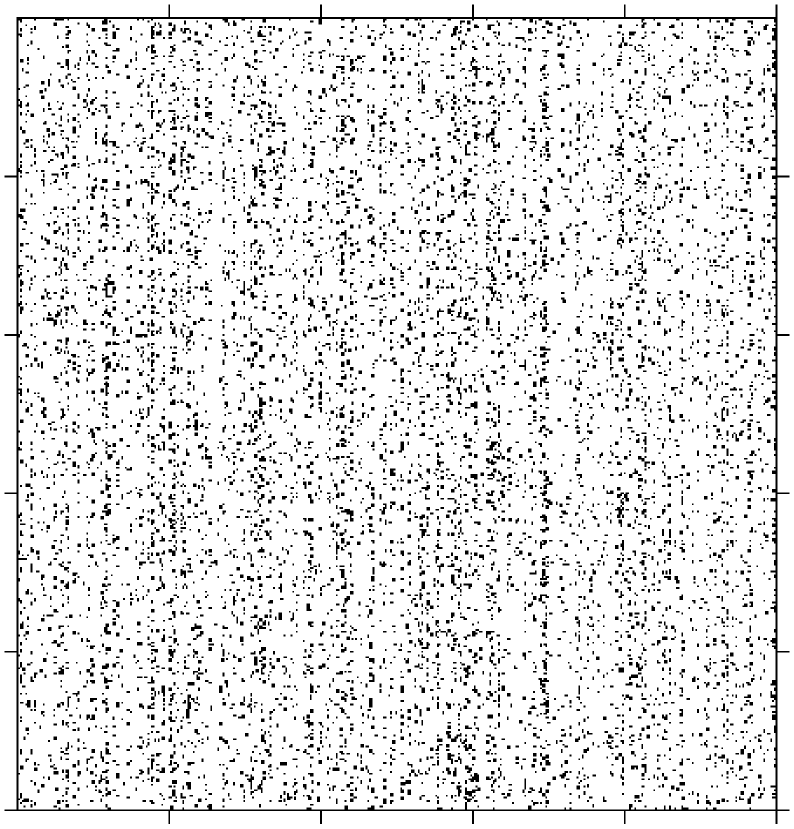
K-means clustering algorithm is a cluster analysis in which k clusters are formed with n observations. The similar observations are clustered determining the centroid. Here the similar heart rates are clustered and the patterns are determined.


RGui (32-bit)
File Edit View Misc Packages Windows Help
R Console
R Graphics: Device 2 (ACTIVE)
bpm.1
150
100
50
50
100
150
bpm
10:59 PM
2/10/2016

```
Type 'q()' to quit R.

[Previously saved workspace restored]

> new = read.csv(file.choose(),header=TRUE)
> head(new)
  bpm Category
1  84 Moderate
2  79 Moderate
3  90 Moderate
4  78 Moderate
5  70 Moderate
6  95 Moderate
> new$Category = NULL
> head(new)
  bpm
1  84
2  79
3  90
4  78
5  70
6  95
> res <- kmeans(new,3)
> plot(new[c("bpm","bpm")],col=res$cluster)
> 
```

2. Apriori Algorithm:

The Apriori algorithm is an algorithm for mining frequent datasets. Since the heart rate of the user when collected it produces similar data and frequent datasets are formed. This enables the use of this algorithm to determine the patterns.



DATASETS:

The datasets in the Heartfit application consists of heart rate data, the steps walked for the day, the timing, the geolocation where it is captured. These datasets are analyzed with the machine learning algorithms and the corresponding patterns are generated.

It would appear as Step Count, Heart rate, time stamp, geolocation. Few more features can be added.

IMPLEMENTATION:

Mobile Client Implementation:

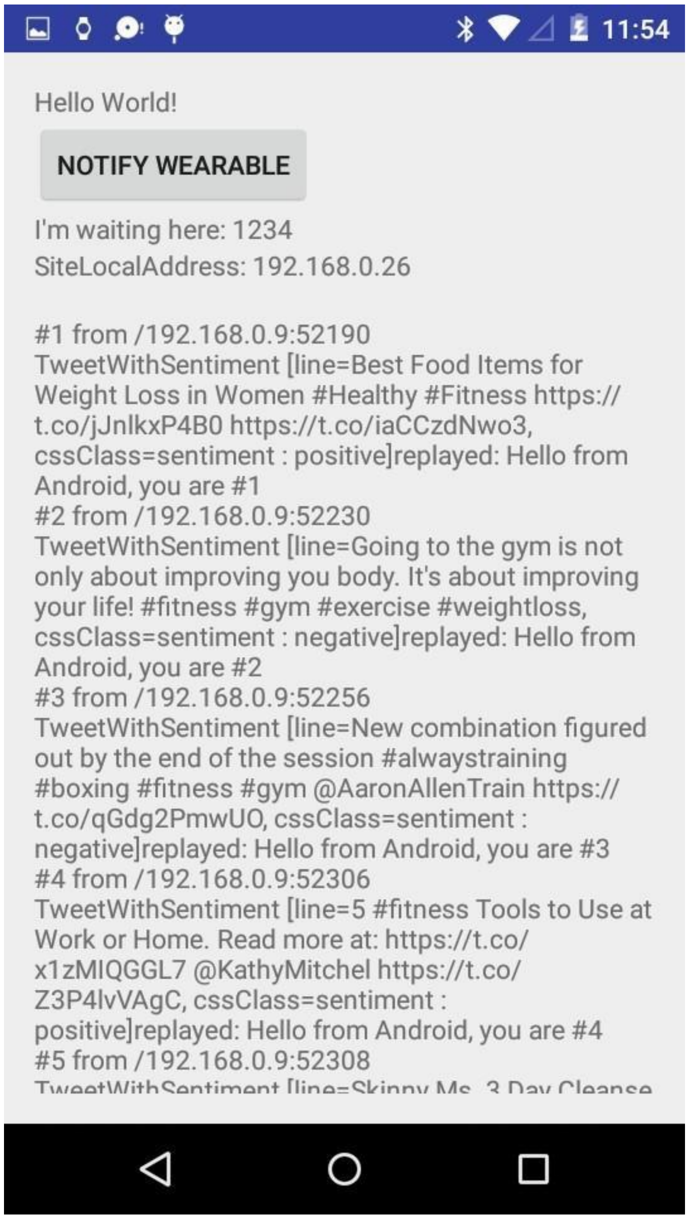
This is smartphone-smartwatch application in which the smart watch senses the data and the data is collected, stored in cloud database. The analysis is performed on the collected data and the results are sent as an email or notification the smartphone. The Spark Mlib and machine learning algorithms are used to perform analysis and determine the patterns from the user health conditions.

Machine Learning Application:

This is the main part of the application, where the machine analyzes and sends the suggests the patterns to user via mail. The machine learning algorithms provide high accuracy of data mining and results. It also results in different patterns that humans cannot determine at the same time. This is an application which is available in hand with the user and keeps a track of the medical history. This medical history can be used in medical field for research purposes as well.

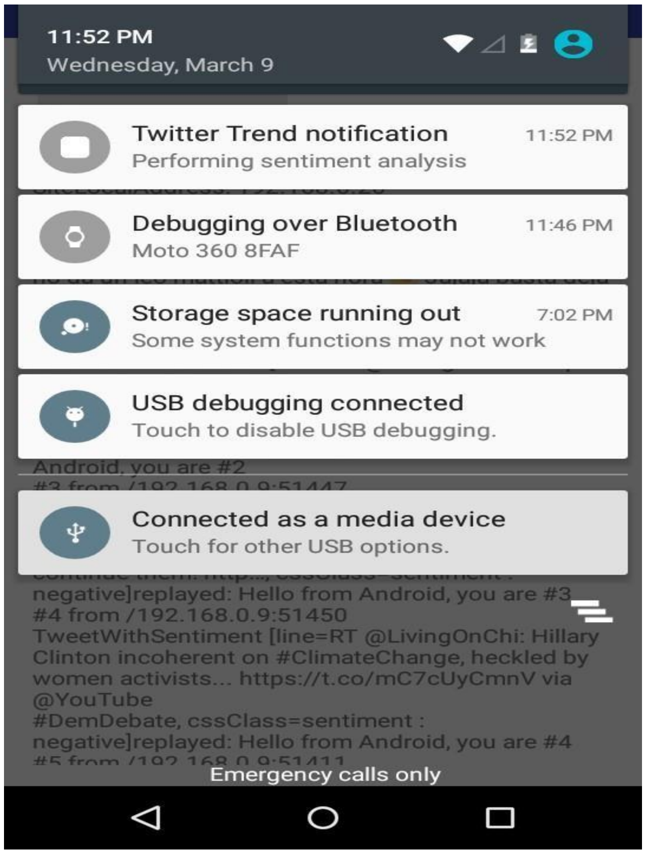
Sentimental Analysis:

The sentimental analysis is performed on the twitter live streaming data and the output is sent to the mobile through the socket connection. This would give us an idea of how the twitter tweets are being posted related to our project.



```
Hello World!

I'm waiting here: 1234
SiteLocalAddress: 192.168.0.26

#1 from /192.168.0.9:52190
TweetWithSentiment [line=Best Food Items for
Weight Loss in Women #Healthy #Fitness https://
t.co/jJnlkxP4B0 https://t.co/iaCCzdNwo3,
cssClass=sentiment : positive]replayed: Hello from
Android, you are #1
#2 from /192.168.0.9:52230
TweetWithSentiment [line=Going to the gym is not
only about improving you body. It's about improving
your life! #fitness #gym #exercise #weightloss,
cssClass=sentiment : negative]replayed: Hello from
Android, you are #2
#3 from /192.168.0.9:52256
TweetWithSentiment [line=New combination figured
out by the end of the session #alwaystraining
#boxing #fitness #gym @AaronAllenTrain https://
t.co/qGdg2PmwUO, cssClass=sentiment :
negative]replayed: Hello from Android, you are #3
#4 from /192.168.0.9:52306
TweetWithSentiment [line=5 #fitness Tools to Use at
Work or Home. Read more at: https://t.co/
x1zMIQGGL7 @KathyMitchel https://t.co/
Z3P4lvVAgC, cssClass=sentiment :
positive]replayed: Hello from Android, you are #4
#5 from /192.168.0.9:52308
TweetWithSentiment [line=Skinny Ms. 3 Day Cleanse
```

It is also sent as notification to the smart watch and also to the smartphone. The screenshots are placed below.



11:52 PM
Wednesday, March 9

Twitter Trend notification 11:52 PM
Performing sentiment analysis

Debugging over Bluetooth 11:46 PM
Moto 360 8FAF

Storage space running out 7:02 PM
Some system functions may not work

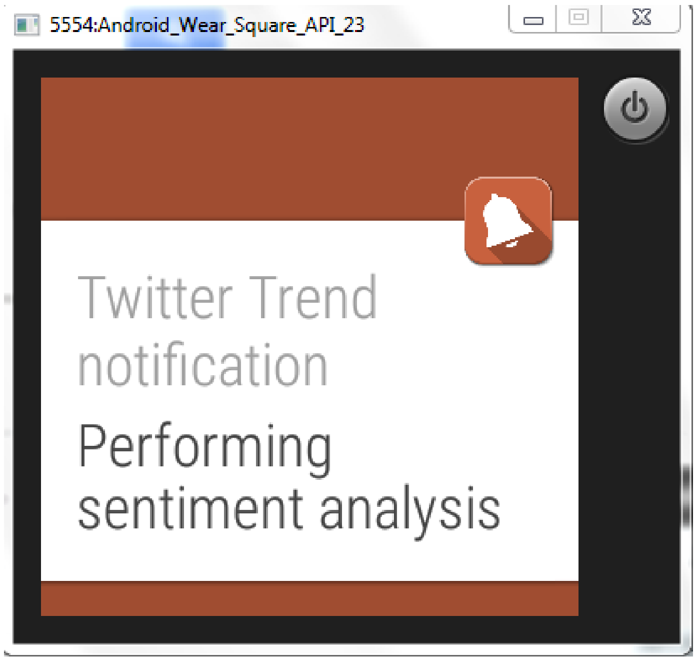
USB debugging connected
Touch to disable USB debugging.

Android, you are #2
#3 from /192.168.0.9:51447

Connected as a media device
Touch for other USB options.

```
negative]replayed: Hello from Android, you are #3
#4 from /192.168.0.9:51450
TweetWithSentiment [line=RT @LivingOnChi: Hillary
Clinton incoherent on #ClimateChange, heckled by
women activists... https://t.co/mC7cUyCmnV via
@YouTube
#DemDebate, cssClass=sentiment :
negative]replayed: Hello from Android, you are #4
```

Emergency calls only



5554:Android_Wear_Square_API_23

Twitter Trend notification

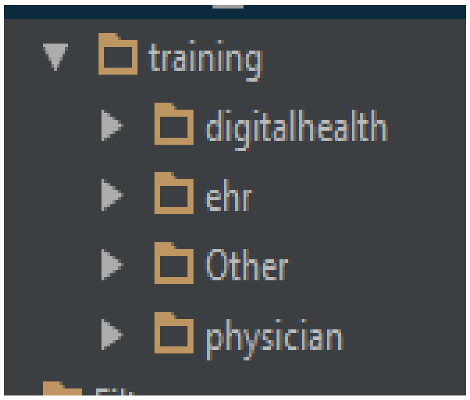
Performing sentiment analysis

CLASSIFICATION:

We are trying to make recommendation system to recommend user some important tweets on basis of some famous medical hashtags. We have predefined our filter hashtags and filtering tweets in streaming. Below is the filter file.

allfilter.txt

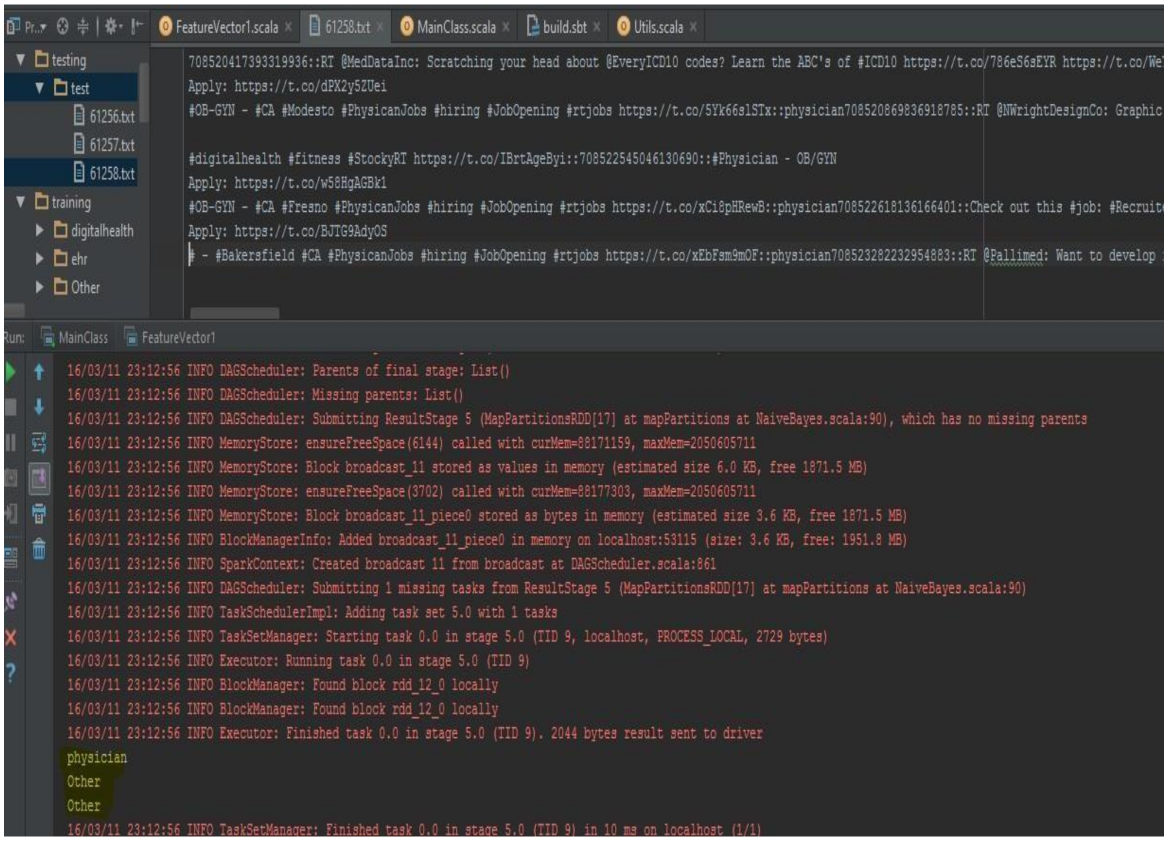
For now we have divided it into below mentioned 4 categories. These categories are on basis of hashtags:



We are training the data for this 4 categories and will test them against the future tweets. For now its just 4 categories but we will include more and plan is to ask user for his interest of tweet and on basis of that we will find appropriate tweet. The selection criteria for choosing tweet from streaming will be on basis of ratings. Program will decide ratings on basis of user’s input and how influenced that tweet is to others people so far. We will include sentiment analysis as well for making sure that positive tweets will reach to user.
Below is the sample predicted results for few tweets.

61257.txt 61258.txt 61256.txt

Above three files are the collected tweets during streaming and below is the predicted output:



```
708520417393319936::RT @MedDataInc: Scratching your head about @EveryICD10 codes? Learn the ABC's of #ICD10 https://t.co/786eS6sEYR https://t.co/We
Apply: https://t.co/dPX2y5ZUei
#OB-GYN - #CA #Modesto #PhysicanJobs #hiring #JobOpening #rtjobs https://t.co/5Yk66slSTx::physician708520869836918785::RT @NWrightDesignCo: Graphic

#digitalhealth #fitness #StockyRT https://t.co/IBrtAgeByi::708522545046130690::#Physician - OB/GYN
Apply: https://t.co/w58HgAGBk1
#OB-GYN - #CA #Fresno #PhysicanJobs #hiring #JobOpening #rtjobs https://t.co/xCi8pHRewB::physician708522618136166401::Check out this #job: #Recruit
Apply: https://t.co/BJTG9AdyOS
# - #Bakersfield #CA #PhysicanJobs #hiring #JobOpening #rtjobs https://t.co/xEbFsm9mOF::physician708523282232954883::RT @Pallimed: Want to develop
```

```
16/03/11 23:12:56 INFO DAGScheduler: Parents of final stage: List()
16/03/11 23:12:56 INFO DAGScheduler: Missing parents: List()
16/03/11 23:12:56 INFO DAGScheduler: Submitting ResultStage 5 (MapPartitionsRDD[17] at mapPartitions at NaiveBayes.scala:90), which has no missing parents
16/03/11 23:12:56 INFO MemoryStore: ensureFreeSpace(6144) called with curMem=88171159, maxMem=2050605711
16/03/11 23:12:56 INFO MemoryStore: Block broadcast_11 stored as values in memory (estimated size 6.0 KB, free 1871.5 MB)
16/03/11 23:12:56 INFO MemoryStore: ensureFreeSpace(3702) called with curMem=88177303, maxMem=2050605711
16/03/11 23:12:56 INFO MemoryStore: Block broadcast_11_piece0 stored as bytes in memory (estimated size 3.6 KB, free 1871.5 MB)
16/03/11 23:12:56 INFO BlockManagerInfo: Added broadcast_11_piece0 in memory on localhost:53115 (size: 3.6 KB, free: 1951.8 MB)
16/03/11 23:12:56 INFO SparkContext: Created broadcast 11 from broadcast at DAGScheduler.scala:861
16/03/11 23:12:56 INFO DAGScheduler: Submitting 1 missing tasks from ResultStage 5 (MapPartitionsRDD[17] at mapPartitions at NaiveBayes.scala:90)
16/03/11 23:12:56 INFO TaskSchedulerImpl: Adding task set 5.0 with 1 tasks
16/03/11 23:12:56 INFO TaskSetManager: Starting task 0.0 in stage 5.0 (TID 9, localhost, PROCESS_LOCAL, 2729 bytes)
16/03/11 23:12:56 INFO Executor: Running task 0.0 in stage 5.0 (TID 9)
16/03/11 23:12:56 INFO BlockManager: Found block rdd_12_0 locally
16/03/11 23:12:56 INFO BlockManager: Found block rdd_12_0 locally
16/03/11 23:12:56 INFO Executor: Finished task 0.0 in stage 5.0 (TID 9). 2044 bytes result sent to driver
physician
Other
Other
16/03/11 23:12:56 INFO TaskSetManager: Finished task 0.0 in stage 5.0 (TID 9) in 10 ms on localhost (1/1)
```

Food recommendation according to nutrients

For recommendation use case we are planning to recommend user food items as per nutrition he is following or meal plan he is taking.

We have collected data from USDA national nutrient data and made it in required format after some clean up. We dumped it to mysql as structured format. We have over 9,000 food items and 25 different nutrients.

We are using cosine similarity function for finding the similarity between users' selected or consumed item and the other items available to the database. We compare individual item with other item to make pair of two rows. We calculate cosine similarity between these two rows and store result as (key, pair) value where key is id of two food items and value is cosine similarity which lies between 0 to 1. Here, we don't have any negative values so it lies between 0 to 1 else it would be between -1 to 1.

Below is the simple example of cosine similarity.

$$cos(\vec{t_1}, \vec{t_2}) = \frac{\vec{t_1} \cdot \vec{t_2}}{\|\vec{t_1}\| \|\vec{t_2}\|}$$

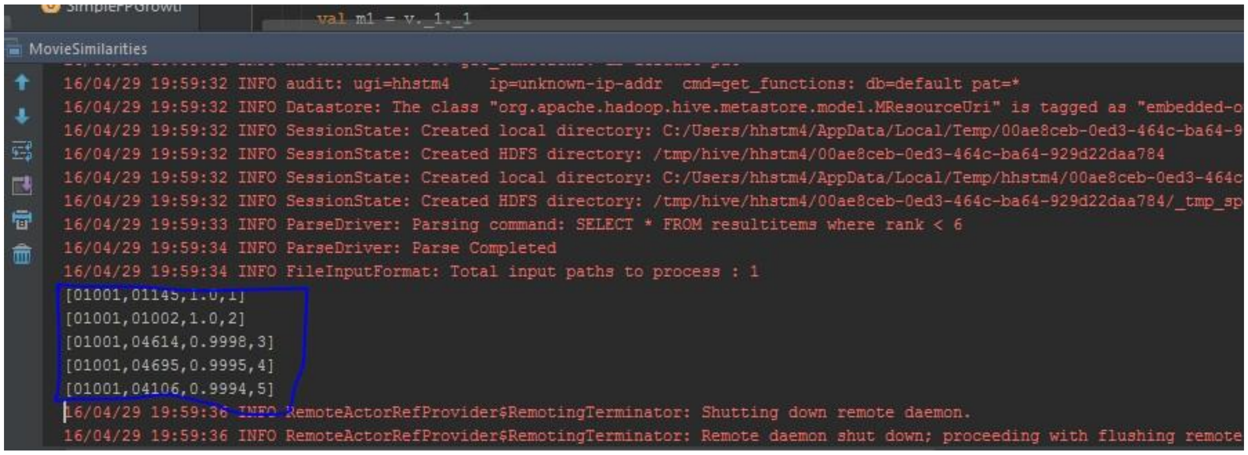
To calculate cosine similarity between two texts t1 and t2, they are transformed in vectors as shown in the Table 1.

For example, a cosine similarity can be computed as below for two texts

$$\frac{1 \cdot 2 + 1 \cdot 0 + 0 \cdot 1 + 1 \cdot 1}{\sqrt{1^2 + 1^2 + 0^2 + 1^2}\sqrt{2^2 + 0^2 + 1^2 + 1^2}} \simeq 0.72$$

	glutathione	homocystine	coa	transhydrogenase
$\vec{t_1}$	1	1	0	1
$\vec{t_2}$	2	0	1	1

We ran this using apache spark core and spark SQL. We used JDBC connecter for java to store result back to the mysql database. Below is the snap shot for one item with its most similar 5 items.



```
val m1 = v._1._1
```

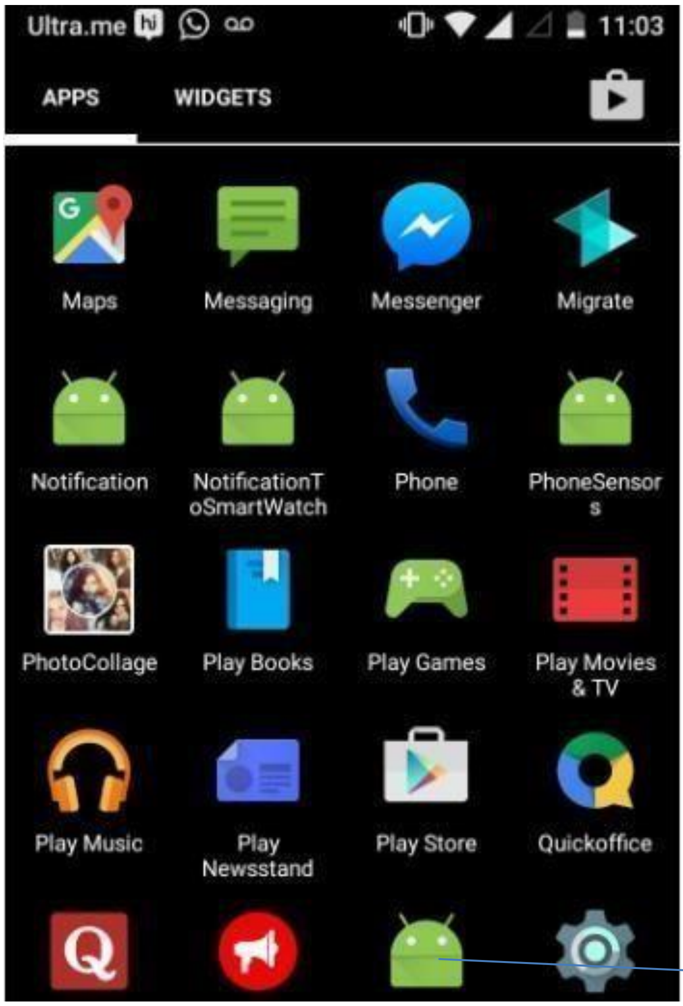
MovieSimilarities

```
16/04/29 19:59:32 INFO audit: ugi=hhstm4    ip=unknown-ip-addr  cmd=get_functions: db=default pat=*
16/04/29 19:59:32 INFO Datastore: The class "org.apache.hadoop.hive.metastore.model.MResourceUri" is tagged as "embedded-o
16/04/29 19:59:32 INFO SessionState: Created local directory: C:/Users/hhstm4/AppData/Local/Temp/00ae8ceb-0ed3-464c-ba64-9
16/04/29 19:59:32 INFO SessionState: Created HDFS directory: /tmp/hive/hhstm4/00ae8ceb-0ed3-464c-ba64-929d22daa784
16/04/29 19:59:32 INFO SessionState: Created local directory: C:/Users/hhstm4/AppData/Local/Temp/hhstm4/00ae8ceb-0ed3-464c
16/04/29 19:59:32 INFO SessionState: Created HDFS directory: /tmp/hive/hhstm4/00ae8ceb-0ed3-464c-ba64-929d22daa784/_tmp_sp
16/04/29 19:59:33 INFO ParseDriver: Parsing command: SELECT * FROM resultitems where rank < 6
16/04/29 19:59:34 INFO ParseDriver: Parse Completed
16/04/29 19:59:34 INFO FileInputFormat: Total input paths to process : 1
[01001,01145,1.0,1]
[01001,01002,1.0,2]
[01001,04614,0.9998,3]
[01001,04695,0.9995,4]
[01001,04106,0.9994,5]
16/04/29 19:59:36 INFO RemoteActorRefProvider$RemotingTerminator: Shutting down remote daemon.
16/04/29 19:59:36 INFO RemoteActorRefProvider$RemotingTerminator: Remote daemon shut down; proceeding with flushing remote
```

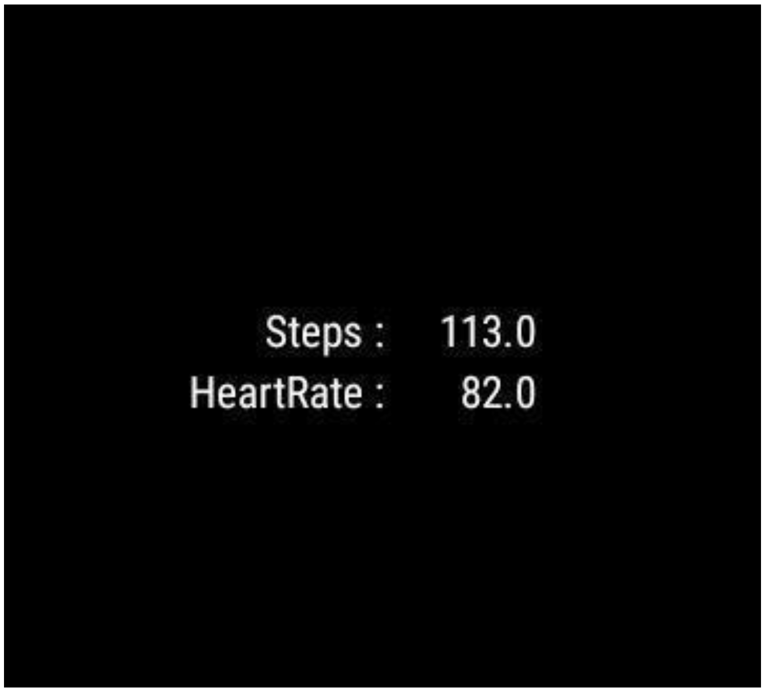
In result, first column is the ID of item for which we are finding the most similar items. Second column is the Id of matching item, 3rd column is similarity and 4th column is rank.

The main idea is that its huge database with bunch of items. User is not able to go through each item and check nutrient contains inside it. So this is the way of providing him matching items of his interest of nutrients to make him healthy.

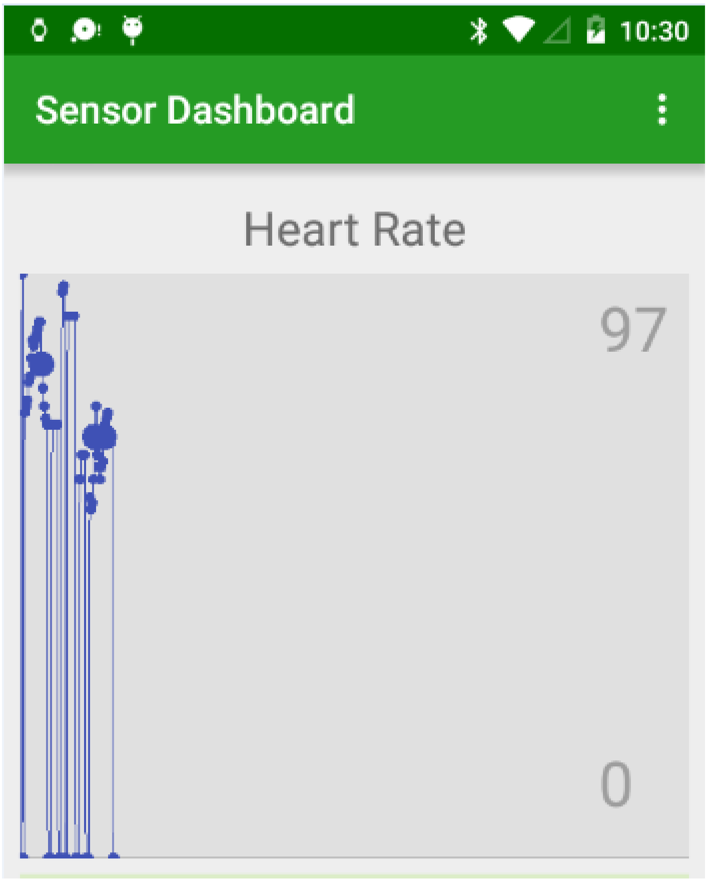
DEPLOYMENT:



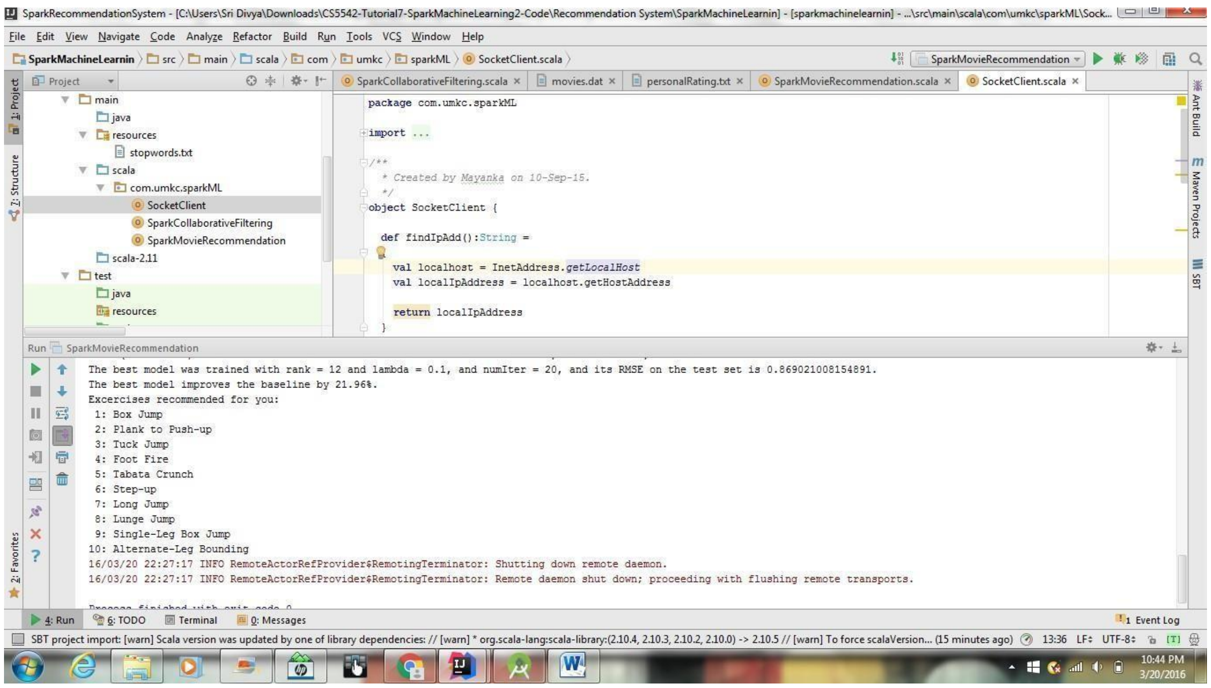
HeartFit application



```
Steps :      113.0
HeartRate :   82.0
```



In our project, we are recommending the users some physical and heart exercises based on their health condition and their daily routine. Based on the personal ratings and ratings of other users followed the recommendations are given.



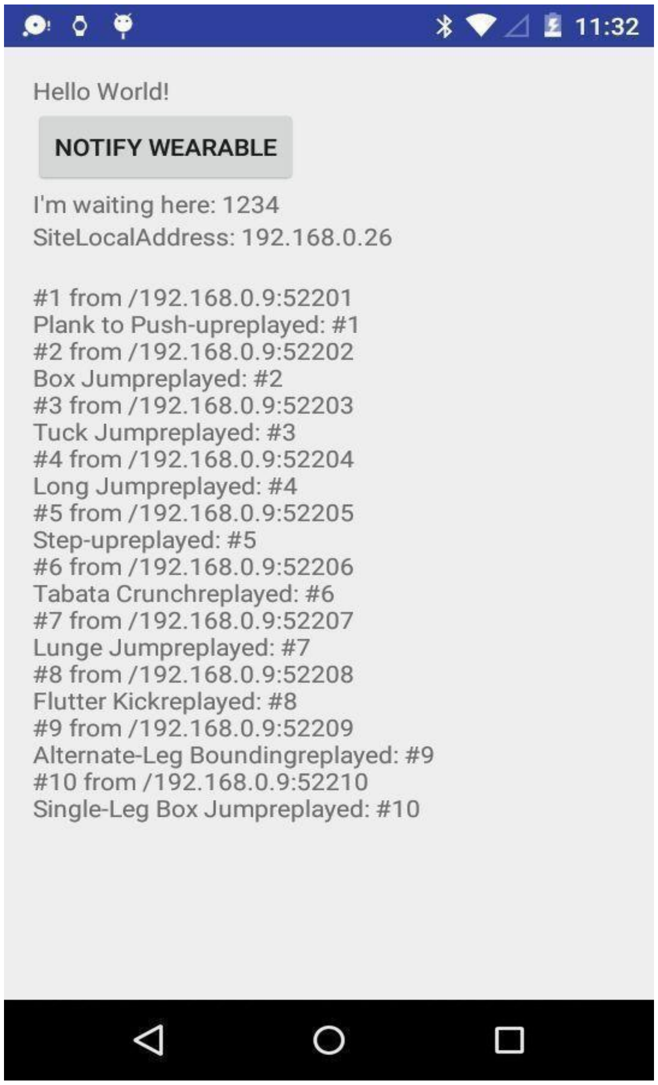
SparkRecommendationSystem - [C:\Users\Sri Divya\Downloads\CS5542-Tutorial7-SparkMachineLearning2-Code\Recommendation System\SparkMachineLearnin] - [sparkmachinelearnin] - ...\src\main\scala\com\umkc\sparkML\Sock...

```scala
package com.umkc.sparkML

import ...

/**
  * Created by Mayanka on 10-Sep-15.
  */
object SocketClient {

  def findIpAdd():String =

    val localhost = InetAddress.getLocalHost
    val localIpAddress = localhost.getHostAddress

    return localIpAddress
  }
```

```
The best model was trained with rank = 12 and lambda = 0.1, and numIter = 20, and its RMSE on the test set is 0.869021008154891.
The best model improves the baseline by 21.96%.
Excercises recommended for you:
 1: Box Jump
 2: Plank to Push-up
 3: Tuck Jump
 4: Foot Fire
 5: Tabata Crunch
 6: Step-up
 7: Long Jump
 8: Lunge Jump
 9: Single-Leg Box Jump
10: Alternate-Leg Bounding
16/03/20 22:27:17 INFO RemoteActorRefProvider$RemotingTerminator: Shutting down remote daemon.
16/03/20 22:27:17 INFO RemoteActorRefProvider$RemotingTerminator: Remote daemon shut down; proceeding with flushing remote transports.
```

SBT project import: [warn] Scala version was updated by one of library dependencies: // [warn] * org.scala-lang:scala-library:(2.10.4, 2.10.3, 2.10.2, 2.10.0) -> 2.10.5 // [warn] To force scalaVersion... (15 minutes ago)

The same is sent to smartwatch/ smart phone.

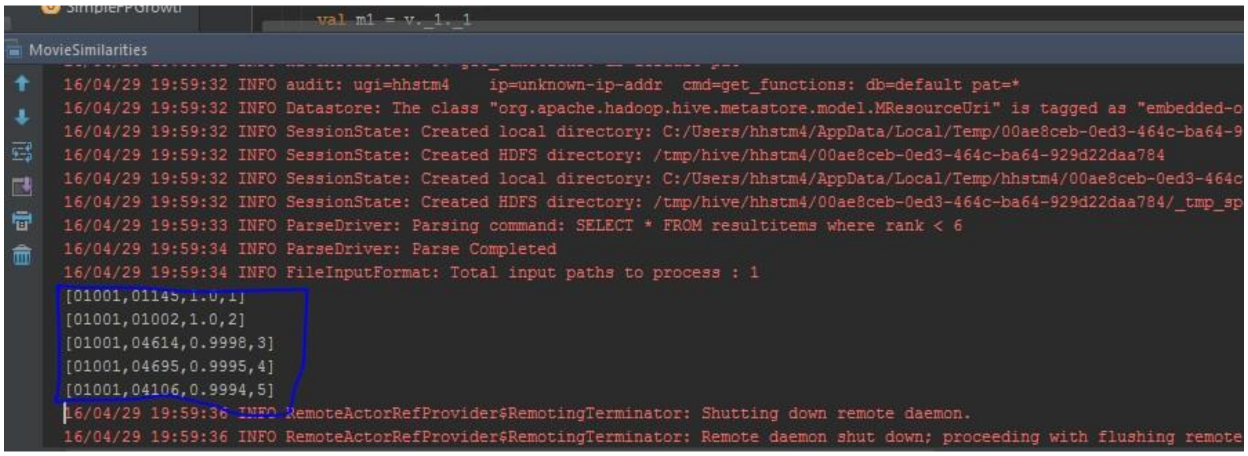


```
Hello World!
I'm waiting here: 1234
SiteLocalAddress: 192.168.0.26

#1 from /192.168.0.9:52201
Plank to Push-upreplayed: #1
#2 from /192.168.0.9:52202
Box Jumpreplayed: #2
#3 from /192.168.0.9:52203
Tuck Jumpreplayed: #3
#4 from /192.168.0.9:52204
Long Jumpreplayed: #4
#5 from /192.168.0.9:52205
Step-upreplayed: #5
#6 from /192.168.0.9:52206
Tabata Crunchreplayed: #6
#7 from /192.168.0.9:52207
Lunge Jumpreplayed: #7
#8 from /192.168.0.9:52208
Flutter Kickreplayed: #8
#9 from /192.168.0.9:52209
Alternate-Leg Boundingreplayed: #9
#10 from /192.168.0.9:52210
Single-Leg Box Jumpreplayed: #10
```

The recommendation system we built from the nutrients database. Here the user can select either lunch, breakfast or dinner. If he tells his preference then we would make a recommendation system on spark and then recommend the user about the foods that has same protein and calories he can take. In this way, this recommendation would benefit the user for intake of various healthy foods with same proteins and nutrients. But we assume that the user goes with the healthy food.

We would also suggest him that a particular food is not healthy and he can change his preference. This helps in maintaining body fitness as well as his heart fit, staying healthy.

The output looks like this when the user searches for an item.



```
val m1 = v._1._1
```

```
16/04/29 19:59:32 INFO audit: ugi=hhstm4    ip=unknown-ip-addr  cmd=get_functions: db=default pat=*
16/04/29 19:59:32 INFO Datastore: The class "org.apache.hadoop.hive.metastore.model.MResourceUri" is tagged as "embedded-o
16/04/29 19:59:32 INFO SessionState: Created local directory: C:/Users/hhstm4/AppData/Local/Temp/00ae8ceb-0ed3-464c-ba64-9
16/04/29 19:59:32 INFO SessionState: Created HDFS directory: /tmp/hive/hhstm4/00ae8ceb-0ed3-464c-ba64-929d22daa784
16/04/29 19:59:32 INFO SessionState: Created local directory: C:/Users/hhstm4/AppData/Local/Temp/hhstm4/00ae8ceb-0ed3-464c
16/04/29 19:59:32 INFO SessionState: Created HDFS directory: /tmp/hive/hhstm4/00ae8ceb-0ed3-464c-ba64-929d22daa784/_tmp_sp
16/04/29 19:59:33 INFO ParseDriver: Parsing command: SELECT * FROM resultitems where rank < 6
16/04/29 19:59:34 INFO ParseDriver: Parse Completed
16/04/29 19:59:34 INFO FileInputFormat: Total input paths to process : 1
[01001,01145,1.0,1]
[01001,01002,1.0,2]
[01001,04614,0.9998,3]
[01001,04695,0.9995,4]
[01001,04106,0.9994,5]
16/04/29 19:59:36 INFO RemoteActorRefProvider$RemotingTerminator: Shutting down remote daemon.
16/04/29 19:59:36 INFO RemoteActorRefProvider$RemotingTerminator: Remote daemon shut down; proceeding with flushing remote
```

Frequent Pattern Mining – Spark - MLib

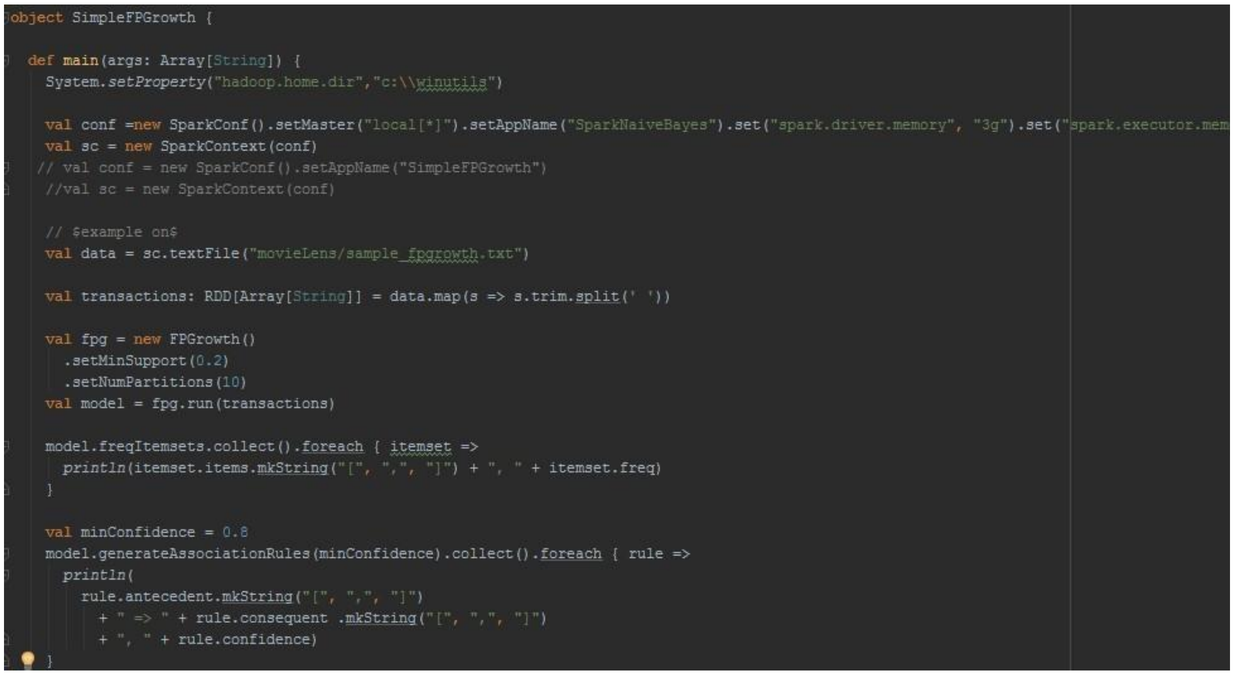
For matching the frequent patterns , we have used FP growth alogorithm.

FP-growth

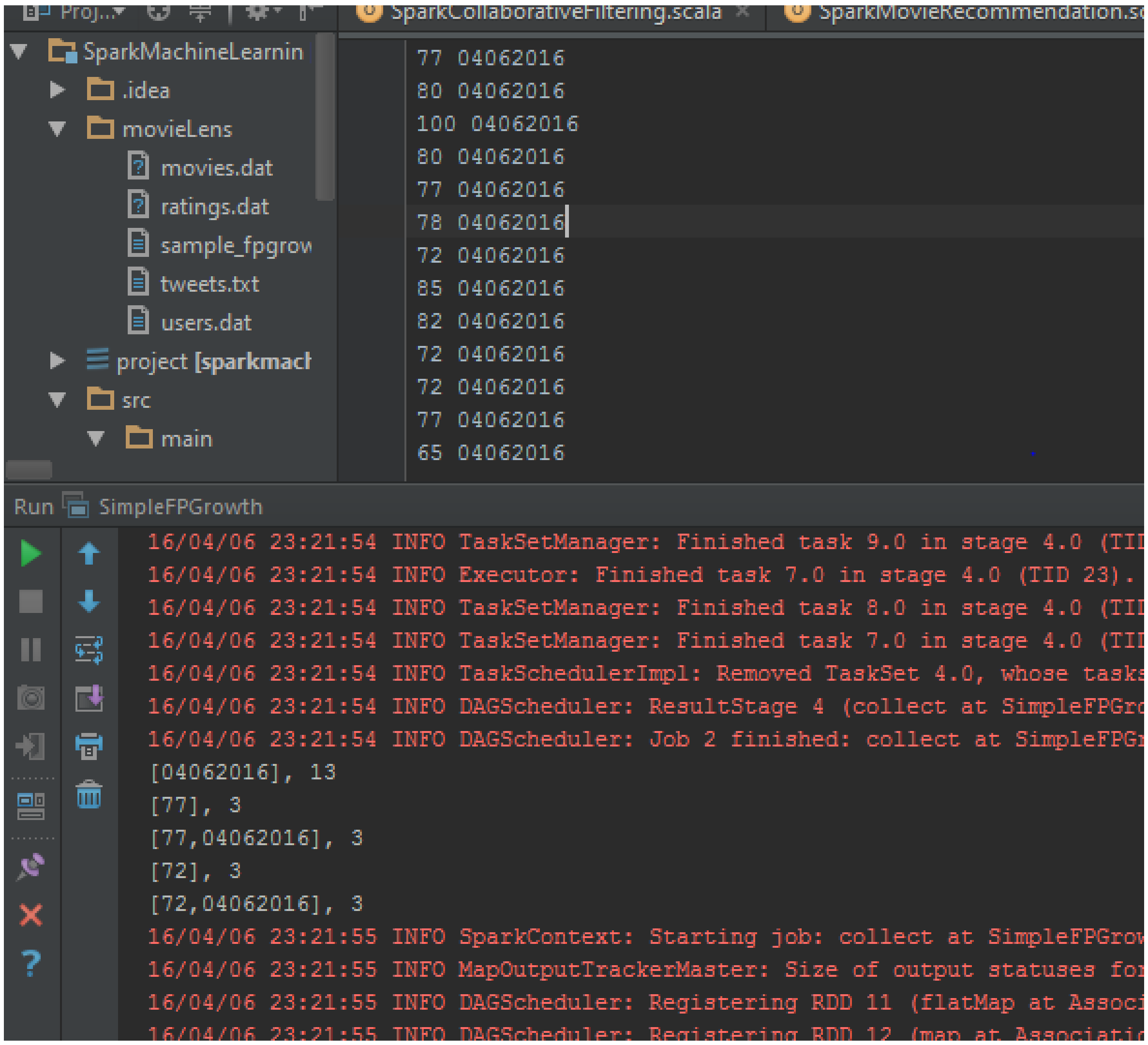
spark.mllib’s FP-growth implementation takes the following (hyper-)parameters:

- minSupport: the minimum support for an itemset to be identified as frequent. For example, if an item appears 3 out of 4 transactions, it has a support of 3/4=0.75.
- numPartitions: the number of partitions that are used to distribute the work.

Below is the implementation of algorithm for our heart bit data to find out the pattern on particular day to check working of heart bit, whether its behaving normal or not. The output for our sample file is included in output file.



```scala
object SimpleFPGrowth {

  def main(args: Array[String]) {
    System.setProperty("hadoop.home.dir","c:\\winutils")

    val conf =new SparkConf().setMaster("local[*]").setAppName("SparkNaiveBayes").set("spark.driver.memory", "3g").set("spark.executor.mem
    val sc = new SparkContext(conf)
   // val conf = new SparkConf().setAppName("SimpleFPGrowth")
    //val sc = new SparkContext(conf)

    // $example on$
    val data = sc.textFile("movieLens/sample_fpgrowth.txt")

    val transactions: RDD[Array[String]] = data.map(s => s.trim.split(' '))

    val fpg = new FPGrowth()
      .setMinSupport(0.2)
      .setNumPartitions(10)
    val model = fpg.run(transactions)

    model.freqItemsets.collect().foreach { itemset =>
      println(itemset.items.mkString("[", ",", "]") + ", " + itemset.freq)
    }

    val minConfidence = 0.8
    model.generateAssociationRules(minConfidence).collect().foreach { rule =>
      println(
        rule.antecedent.mkString("[", ",", "]")
          + " => " + rule.consequent .mkString("[", ",", "]")
          + ", " + rule.confidence)
    }
```



```
77 04062016
80 04062016
100 04062016
80 04062016
77 04062016
78 04062016
72 04062016
85 04062016
82 04062016
72 04062016
72 04062016
77 04062016
65 04062016
```

```
16/04/06 23:21:54 INFO TaskSetManager: Finished task 9.0 in stage 4.0 (TII
16/04/06 23:21:54 INFO Executor: Finished task 7.0 in stage 4.0 (TID 23).
16/04/06 23:21:54 INFO TaskSetManager: Finished task 8.0 in stage 4.0 (TII
16/04/06 23:21:54 INFO TaskSetManager: Finished task 7.0 in stage 4.0 (TII
16/04/06 23:21:54 INFO TaskSchedulerImpl: Removed TaskSet 4.0, whose tasks
16/04/06 23:21:54 INFO DAGScheduler: ResultStage 4 (collect at SimpleFPGro
16/04/06 23:21:54 INFO DAGScheduler: Job 2 finished: collect at SimpleFPGr
[04062016], 13
[77], 3
[77,04062016], 3
[72], 3
[72,04062016], 3
16/04/06 23:21:55 INFO SparkContext: Starting job: collect at SimpleFPGrow
16/04/06 23:21:55 INFO MapOutputTrackerMaster: Size of output statuses for
16/04/06 23:21:55 INFO DAGScheduler: Registering RDD 11 (flatMap at Associ
16/04/06 23:21:55 INFO DAGScheduler: Registering RDD 12 (map at Associatio
```

Github link: https://github.com/hirenshah7390/Bigdata_Project/tree/master/Hiren

PROJECT MANAGEMENT:

Planning

We as a team discussed about the project idea, project flow , features that are to be implemented. Roles and responsibilities are being discussed and given below.

Time: 16 hours

Members Participated: Abhiram, Dinesh Reddy, HirenShah, Harshini

Implementation

The step counter and heart sensor data is collected continuously using the sensors in the smart watch.

The data is sent to the smartphone and graphs are plotted on phone. The collected data is classified and working on recommendation systems.

The machine learning algorithms were implemented and the recommendation systems was implemented. Recommending the user regarding food he can consume and also observing patterns of the user based on his heart rate.

Responsibility: Data collection, Zenhub, FP growth algorithm, Recommendation system.

Time: 90 hours

Participants: Harshini, HirenShah, Abhiram, Dinesh Reddy

Testing

Test cases for all the above designed pages were implemented.

Responsibility: Tried to collect data at different times when sleeping, walking,etc

Time: 20 hours

Participants: HirenShah, Dinesh Reddy, Abhiram, Harshini

BIBILIOGRAPHY:

http://www.r-bloggers.com/association-rule-learning-and-the-apriori-algorithm/

http://www.dreamincode.net/forums/topic/324137-periodically-collect-accelerometer-data-in-android/

https://github.com/pocmo/SensorDashboard

http://www.wareable.com/fitbit/fitbit-70-of-people-ignore-heart-rate-data-1523

https://dev.fitbit.com/docs/heart-rate/

http://www.livescience.com/42132-heart-rate-activity-tracker-useful.html

http://spark.apache.org/docs/latest/mllib-guide.html