**Day 1: 16/02/2015**

* Read HR docs
* Get familiarized with the working environment
* Prepared the working folders
* Got in touch with the project
* Saw the dataset
* Monday morning: weekly meeting with Fabien

**Day 2: 17/02/2015**

* Local git repository is home/dehajjik/workspace
* Remote git repository is /speech/misc/com/git/students/dehajjik/appmining.git
* Datamanagement code transferred in the workspace
* Read previous work done
* **Directories:**
  + /home/: coding directory -> backup but small space
  + /speech/dbwork/mul/students/dehajji/: big data there -> no backup but big space
* **Servers:**
  + 24: old, slow, free
  + 30: quite good, free
  + 47,48: fast, high ram, not big memory, free -> recommended
  + 44, 46: quite fast, big memory, usually full -> use if big processing
* top: command to see the jobs in the current server
* **Ideas**:
  + The time, the place, the activity or any combination of those is what determines the behavior of a user (which app he is going to open)
  + Each feature could represent a distribution of opening app over it´s possible values and each person has a distribution over the different features
    - Learn for each person the prob to select a feature
    - The probability of a person to select a feature is not constant and may itself depend on another feature!! Ex: at time 10 A.M, the user will either open one app of the set of apps A is he is at home or one app of the set B if he is at work!! -> Idea: AT ANY CONFIGURATION OF FEATURES; TRY TO FIND THE FEATURES THAT WILL DERTERMINE THE ACTION OF THE USER!!
    - Explore the idea of leaning joint feature probabilities instead of independent ones. Ex: Pr(feature=time|t=10A.M, X=home)
    - The behavior of a person may differ depending on if we are on a week day or a week end. -> THINK OF SPLITTING TIME IN WEEK DAYS AND WEEK ENDS
* Agreed with Fabien to first explore the data with looking in parallel in some documentation

**Day 3: 18/02/2015**

* <https://wiki.stc.eu.sony.com/SpeechAndSound/HowToPullPhoneLogData> contains information about the features of the data and the extraction scripts (the information about the features is not exhaustive, there is some features missing)
* Understood the main parts of the data extraction code
* The data features may mostly come from android
* The original data is stored in json directory, one directory for each user. Each directory contains a file per entry (daily ? one app launch? Don´t know). There is also a directory all/ that contains one file that regroups all the json and another file that contains a validated (keeps only good entries) of the all-json file.
* We can get the duration that a user spent in an application
* The new data generated **records only the actions when a user clicks on an icon** to open it. It does not for example record when an application is opened through another application. -> This is good because that´s what we want exactly to predict.
* **The time is calculated in a unit circle using sinus and cosinus coordinates**. This is to be able to detect that 23 AM and 0AM are close.

**Day 4: 19/02/2015**

* Completing the Application Launch Feature file. It describes all the features that we have in the data and also the structure of the data. Links to the original documentation in google written.
* Really a huge number of features that might be interesting during the classification.
* **Ideas:**
  + When the network of the user is disconnected from the internet, he may access to settings. Or will not open apps that needs internet.

Another example is if the battery is very low, he will not open a game even if he used to do so at this time.

* + - Thus the features may be more or less important depending on the context and the choice of the user might be influenced by some features in some context and by others in another context. NEED TO FIND A WAY TO MODEL THAT
  + Application launch is only a feature like the others, and it might be more or less important depending on the context.

So the records of data that are not registered at an application launch might give us a lot of information regarding the next application to open for the user. It is then important to take them into account and to take profit of such data.

**Day 5: 20/02/2015**

* **Ideas:**
  + The features that influence the behavior of a user may be different depending on the person. For that reason, the model should be able to do an automated feature selection to determine for each user the features that will be taken into account.
  + Each entry (feature) can be represented as an event and then all the features can be represented in a time axis. For example events will be app\_launch, bluetooth\_pairing, battery\_record, activity\_recognition. By doing this, we only need to take the vector of events and their corresponding times and compress this information to predict the event that corresponds to the next application to be opened.
* We can have access to all the features just before the opening of the application. The feature wifiApps can reveal to us information about the place where the user is. Is can be more accurate than simply the GPS coordinates in the sense that we can have information about the place of a user even inside a building (meeting room, dining room, ect…).

**Day 6: 23/02/2015**

* **Project goal:** The project is no more centered in the application prediction. We want to do something more general that can find much more applications.
  + Starting from the data we want to discover and **describe the user behaviors** to better understand the users. It is not necessary related to the applications opening.
* **Approach:** we will try to **discover patterns that repeat in the user behavior**. For example, we could derive that on each Monday the user uses the Bluetooth when he is at work. Another example is when the user plugs his headphones, he usually runs and opens a sport application.
* **Method:**
  + Use the clustering to try to find clusters of features for a specific user. Finding clusters of features implies that we found groups of features that usually occur together. It thus describes the behavior of the user.
  + With clustering we cannot represent sequences (ex: opening app 1 after opening app2). For this, one solution is to code this information in a new feature and to append it to our vector of features.
* SVD and PCA are a good way to explore and clusterize the data. SVD seems to be a good way to represent features on common concepts.
* To be able to catch the meaning of features with SVD **an extended and binary representation of the original vectors** may be a good idea. For example the feature “Bluetooth” can be extended to two features “Bluetooth\_on” and “Bluetooth\_off”. This is useful because it will enable us to see that the fact that the bluetooth is activated is important in one concept which is not the case where the bluetooth is off.
* A solution should be found in the representation of data of the missing fields.
* For the next Monday the goal is to have a good data representation and having tried for example SVD.
* **Idea:**
  + **The probabilistic models** can be a good idea and a better fit than SVD to model the clusters. PLSI, LDA, ect…

**Day 7: 24/02/2015**

* **Feature selection and representation:**
  + We decided to represent the features as a binary vector. This is as representing a corpus of documents and words. We do so because we believe that by doing so, SVD can help us to extract some groups of correlated features.
  + If some features are unknown or inexistent, we decided to put the corresponding values to -1
  + For Battery, decided to put Health (that can be good, cold, dead or unknown) rather than putting the Battery level. The latter could be an option because it enable us to do a different and more precise splitting of the battery. Depending on the user, this could be useful.
  + In Battery we represent the information “plugged or good” as a feature because it is possible that the behavior of the user be the same when the battery is good and when it is plugged (even if it´s plugged and low battery). This is because he can assume in both cases that he can consume as much as he wants from the battery.
  + For “Bluetooth”, even if we don´t have information about it, this doesn´t mean that we don´t know if the device is connected or is detecting some devices but it means that we know that the device is not connected and is not seeing any device. That´s why for that case we don´t put -1 but we simply put 0.
  + For “Bluetooth”, we added one feature called “not\_connected\_with\_any\_device” that takes 1 if the device is not seeing any device or if it is seeing some devices but not connected with any. This is because we could think that if the user is not connected to any device it means that he is not using Bluetooth even if his Bluetooth is activated.
  + For “Bluetooth”, we also combined “bounded” and “bounding” in one feature “bounded or bounding” because we assume that those two events are so close that is will not influence the behavior of the user. If his device is already bounded or is bounding, he will probably in both cases have a behavior of someone that is going to use Bluetooth.
  + Some features in “network info” may be redundant or too local. Keep in mind that some features there could be removed.
  + For the “time” we decided to represent the redundancy as follows: the hour (between 0 and 23), the day (between 1 and 7), weekend or weekday (because the user may have a similar behavior in weekends and another behavior in week days), an entry if the event occurred between 8A.M and 6P.M, another if it has occurred between 6P.M and 0A.M and finally one last if it is occurred between 0A.M and 8A.M (This dividing is quite arbitrary but we believe that users may show different behaviors in the day, the evening and the night). We will also represent the combination between the week end and week days and the three day categories (in week-ends at night may be different that weekdays at night).
* In src (in workspace), created a
  + json\_function folder that will contain some utility functions about json format.
  + Data\_exploration folder that will contain all the stuff about data exploration as getting some primitive statistics and some preliminary results

**Day 8: 25/02/2015**

* Installed anaconda distribution to use python
* Started to code some functions about log writing files and features rates
* Getting used with python
* It may be good to introduce a **2 hours periodicity (3hours ?)** in the features to be able to detect some behaviors that repeats with-in a frame of 2 hours

**Day 9: 26/02/2015**

* Coded the code that outputs a statistic about how often each feature is present for each user.
* The results are in logs/26022015171333features\_presence\_rate
* The features **presence rate is too low for most of the features and most of the users** -> means that in the most cases we do not have information about the feature! Very sparse data

**Day 10: 27/02/2015**

* Coded the period of observation and number of records function
* The results are in the log directory
* There is a lot of samples for each user (Around 500 samples per day) and the period of observation is around 6 months:
  + This is good because the data is really dense, we have a lot of data
  + This means that most of the users are using the monster logger and that automatic records are done
  + The number of records and the presence rate of the app\_launch feature seems to concord with Fabian data -> there is 1% of app\_launch and around 400000 records for user 358240050409564. Fabian recorded around 4000 app\_launches for that particular user.
  + For that user (**358240050409564**) it seems to be that the observation period is anormally long (around 1 year) and **could be that the first period corresponds to non-reliable data** (data collected during the testing period of the launcher)
* According to Fabien, **SVD works well for corpus clustering when the rate between documents and words is around 50000 words and 1M documents**. Check that the feature vector stays in that order.

**Day 11: 02/03/2015**

* Coded the features co-occurences
* Results are in the log directory

**Day 12: 03/03/2015**

* Coded the number of records by day
* Plots in the log directory
  + A lot of variations in the plot
  + We can see a sort of increase in the general mean of points (monster logger) but not that much
  + However the points still fluctuate -> **Need to understand why**

**Day 13: 04/03/2015**

* Factored the code and created a DataExtractor
* Extended the features by day plot so that we can see even the 0 records days
* TMM presentation made: presented the first statistics of the data
  + Slides are in the desktop, folder TMM
  + For the TMM the presentations should be put in **/speech/misc/com/doc/TMM**

**Day 14: 05/03/2015**

* To understand the fluctuations in the number of records, decided to represent the number of records of the futures per day for each user. The goal is to find what feature makes the recorder working more in some days and less in the others.
* Coded the feature\_by\_day code.
* Plots in the log directory
* **Explanation of the fluctuations**:
  + There is one big jump for all the users that corresponds to the use of the monster logger (it was initially used and then removed). It is around the end of August 2014 and the beginning of September 2014. This is because we can see that at that period all the features records increase.
  + Then there is other jumps that are only due to the big number of notifications recorder. In fact, all the feature stays somehow constant during that period except the notifications that really explode at some days. –> **Need to see why**

**Day 15: 09/03/2015**

* Have the number of features by hour
  + There is big fluctuations in some hours
* Printed the hours of congestion and opened them in a json reader
  + The notification event is the one that causes problems
  + There is a lot of notification events (each 2 seconds) and most of them seems to come from the system.
* Need to filter that.

**Day 16: 10/03/2015**

* **What events cause the records number to explode ?**
  + There is a lot of redundant records due to the events launcher\_on, launcher\_off, screen\_off, screen\_on (a record each one second).

Example is the user 2 in 12-08-2014 from 6:06:14 to 6:11:37 where there is around 14 records of launcher\_on, off or sreen\_on, off

* + There is some system notifications that make the system to register a record.
  + There is some notifications, where they appear with multiple post times with a difference of few milliseconds between each post time and that causes the system to make a record each two milliseconds. Ex: clock alarm (maybe because it rings a lot of times each millisecond.

Example is the user 2 in 12-08-2014 from 6:14:15 to 6:59:08 where the is more than one record per second all pointing to the notification com.sonyericsson.organizer with different posttimes

* Printed the different types of notifications sorted by their occurrence number. The notification types occurs in the importance order as described below in the notification meanings. The complete file with the number of occurrences is in the log dir.
* **Notifications meaning:**
  + **com.google.android.gms: google mobile services (gmail, googlechrome, googlemap), need to understand what´s this**
  + com.sonymobile.genericuploader**: do blacklist**
  + com.google.android.googlequicksearchbox: **blacklisted**
  + com.google.android.gm**:** gmail
  + com.sonymobile.lifelog: info about user´s daily life
  + jp.gocro.smartnews.android: news application from android
  + com.android.vending: **filtering**
  + jp.co.yahoo.android.news: yahoo news
  + **android**: need to understand
  + com.sony.voyagent.mixs.mixswidget: **blacklisted**
  + **com.sony.tvsideview.phone**: understand
  + com.gunosy.android: Japanese app
  + com.sony.drbd.reader.other.jp: bd reader japenese app
  + com.latedroid.juicedefender: saving battery app
  + com.facebook.katana: facebook app
  + com.facebook.orca: facebook messenger
  + com.newspicks: news japenese app
  + jp.naver.line.android: free call app as viber
  + com.sony.voyagent.mixs.rawdatalogger: **blacklisted**
  + jp.co.yahoo.android.weather.type1: Japanese yahoo whether
  + com.android.providers.downloads: downloads
  + **com.sonymobile.app.powerpredict**: need to understand
  + com.news.rssfeedreader: rss app
  + com.android.phone: phone part of the smartphone
  + com.android.calendar: calendar part of the smartphone
  + com.moneyforward.android.app: japenese app
  + wni.weathernewsTouch.jp: japanese weather app
  + com.tonchidot.tab: Japanese cooking app
  + com.sonyericsson.updatecenter: app that handles the updates of the softwares
  + jp.co.yahoo.android.locationlog: Japanese location app of yahoo
  + com.sonyericsson.organizer: clock app of Sony
  + **com.sony.sensing.has.activitypug**: need to understand
* There is a black list of notifications (system notifications) in the appvalidator code that is not applied on the data (need to apply it)
* The filtering of the notification will be done as following:
  + Blacklisted notifications: notifications that are blacklisted
  + Filtering with the priority: notifications that are not displayed should be removed
* After having slected the blacklisted notifications and filtering the apps with the priority level, get a new version of the data with:
  + Removing the blacklisted notifications
  + Keeping only the new added notifications in each record (and not an array)
  + Regrouping the same duplicate notifications that occurs in some intervals of few milliseconds
  + If there is no new notification, remove the feature notification from the record
  + When removing the feature notification, there is no other feature than event, then remove the record
* After doing that, one problem still remains; the records of a lot of features that were caused by the explosion of notifications will remain even if the notification feature was removed. Thus, we´ll still see some picks around the period of September and the concerned records will contain a lot of replicated information (because we were taking one record each 1). To treat that problem there is two solutions:
  + Remove those records anyway so that the data becomes sampled in the same manner. The good thing is that the data will be consistent. However we lose some information that we could have took profit from.
  + Change the way of representing the data; represent each feature in a row, with each time the value of the feature and the time corresponding to that value. This way we will only keep the non-redundant information for each feature and will not lose any useful data. However, in that case need to find a way on how to analyze the data to find clusters.
* Decided to go for the second solution because it is the way that makes the most sense to represent the data. Our goal is to find clusters in user behavior, and not in user records. That´s why it makes more sense to eliminate the concept of record and to represent the user behavior on a time basis. After that, it is a choice to see how close two behavior needs to be to be considered as co-occurring. Moreover, By splitting the records, we will araise all the duplicate information. The way to analyze that could be representing an hour frame (or week day as a document.). Can even find better ways.

**Day 17: 11/03/2015**

* Filtering the notifications according to the following filters:
  + There is some blacklisted notifications
  + Removing the duplicates:
    - We consider a duplicate a notification that occurred less that **5 seconds before**: the choice of 5 seconds is arbitrary and because it will remove the all the duplicates with removing only a little few samples. Another argument is that two successive notifications of the same app will not necessarily modify the behavior of the user (two successive what’s app or only one is nearly the same)
  + Removing the app with low priority because they are most surely not displayed
  + Added on event type: duplicate\_notification which means that the record where done due to a duplicated notification

**Day 18: 12/03/2015**

* Correcting the filtering code
* **The value Nan is the Json is present in the json data but is not readable by the json editors** (it causes an error)
* **Filtering notification idea:** To be sure what notifications have a meaning and what are the ones that do not have a good idea could be to keep notifications for which we have app launches.
* **The json data is not read in the same order than written**
* The filtered notifications data are in **/speech/dbwork/mul/students/dehajjik/notifications\_filtered/**
* The class Data\_Utils, plot\_features\_by\_day, plot\_features\_by\_hour and specific\_zoom need to be reviewed because they were modified to deal with the filtered data
* The number of notifications really decreased and is becoming more realistic
* **Clean Data Version:**
  + The idea is that from the noisy data, we output a clean json data with keeping all the information of the original version. This means any of the features is removed and any of the values are removed.
  + Cleaning the data means:
    - Filtering the notifications
    - Removing the redundancy of the data. For example if there is a lot of successive records containing the same location place, the information about the location is redundant and should be collapsed
  + The goal is to find a smart way to represent this data in a good json format that satisfies the upper requirements and that is convenient to use.
  + One way is to do the following:
    - Each feature represent a key. And then for each feature we represent all the times where this feature was recorded and we represent the value of the feature at the corresponding time. We only represent an entry if it is different from the last.
    - We still need to find a way on how to represent the bleuetooth and wifi-apps for example

**Day 19: 12/03/2015**

* User1: For the app com.sony.drbd.reader.other.jp we are observing that it generates a lot of notifications in short periods of times. A rate of one sample per 10 seconds during one hour.
* User4: for the app jp.co.yahoo.android.locationlog we are observing the same problem with 1 record per minute
* User 5: for the jp.co.yahoo.android.weather.type1 same problem.
* The app gms are frequent for users 1 and 4 and completely absent fot the others and there is no app with their package name (they do not come directly from app)
* The app android is frequent for all the users
* Is it logical to consider that if the last notifications is the same than the actual one and if the last event is also a notification then we collapse the two notifications? If I receive n successive notifications from the same app before doing any action, then I will behave as if I received only one.
  + For now decided to let the notifications like that but need to keep this **problem in mind**

**Day 20: 13/03/2015**

* **we are loosing some samples when we filter the notifications which should not be the case.**
* Printed the notifications per user and the events per user.

**Day 21: 16/03/2015**

* Wrote the 1st report for Thiran. Can be found in monthyreports directory
* Completed the data\_documentation\_file

**Day 22: 17/03/2015**

* Data loss corrected! Problem was in the same milliseconds records that had the same key.
* A lot of decisions taken in how to represent the clean version of the data. All those decisions are detailed and explained in the file clean\_data\_documentation in the workspace in folder doc
* Begun to code of transforming the new data

**Day 23: 18/03/2015**

* Nearly finished coding the clean data representation
* Low\_priority notifications may be seen by the user in some cases

**Day 24: 19/03/2015**

* Debugging the clean data representation code

**Day 25: 20/03/2015**

* Still debugging the clean data representation code

**Day 26: 23/03/2015**

* Reading Documentation
* **Application idea:** 
  + **LifeLog:** extracting some specific informations about user activities upon user’s request. Ex: how many times went to work with velo this month (need a previous analysis to determine when is it work). Here, note that we have a missing data (we don’t record all the times when the user went to the job on a bicycle)
* Clean data code debugged and cleaned. Data in the repository:
  + **/speech/dbwork/mul/students/dehajjik/notifications\_filtered/**
* Realized that there is two features that we have not noticed:
  + **appOperatingTime**
  + **currentWeatherInfo**

**Day 27: 24/03/2015**

* Investiguate appOperatingTime and currentWeatherInfo, did not added them yet
* Plot statistics:
  + Number of realizations by feature for each user (by day). In log dir
* Started to think to the presentation

**Day 28: 25/03/2015**

* Did the slides presentation
* **Two directions for the contextualization of a behavior**:
  + Either I want to say which events are co-occuring. That means that the user is in a certain state where there is event e1 , e2 and en that are occurring. -> Define what cooccuring means
  + Or say which events are successive, user use to do e1 then e2 then e3…
* Did google doc where we can share ideas with all people for what applications can we do with this dataset (<https://docs.google.com/document/d/1SxdLaiYGln3_W84_4a6kM2qsH7JKin2p8zx01qpBea0/edit#heading=h.yjlpxhl7534c> )

**Day 29: 26/03/2015**

* **Interesting papers:**
  + A Habit Mining Approach for Discovering Similar Mobile Users (2012): They used contextual mobile data to try to regroup users by similar behaviors
    - Similar dataset than ours
    - Tried to capture user’s behaviors and that’s what we plan to do
  + Exploiting Enriched Contextual Information for Mobile App Classification(2013): They tried to classify mobile apps by combining user mobile data, with some web data about the apps
    - a part of the project is dealing with a similar dataset than our
    - Algorithm that they found heavily outperforms the existing ones so may be can find a good idea in their algorithm that could be applied to our problem
  + Mining Personal Context-Aware Preferences for Mobile Users(2012): They are discovering users behaviours or preferences by combining the users records
    - same dataset
    - good results, maybe useful to see what they have done
  + Social Media Mining and Social Network Analysis: Emerging Research Guandong Xu (University of Technology, Sydney) and Lin Li (Wuhan University of Technology): A book released in 2013 that is cited in the three papers above. It contains a whole chapter explaining how to deal with the emergence of the new mobile data and how to extract similar user behaviors from this mobile data.
    - Need to purchase it, no free version found
  + An Effective Approach for Mining Mobile User Habits (2010): Excatly tackles the same problem than ours.
  + Context-Aware Role Mining for Mobile Service Recommendation(2012): Tries to regroup users by same context behavior to be able to recommend services to them. Example is if you know that user listens to pop music in the mornings, recommend a known pop music app to him.
  + Mining frequent patterns without candidate generation (2000): search for this one, seems to be interesting as the starting art (cited in many papers)
  + Discovering significant places from mobile phones(2009) (see day 31)
  + context-aware query classification(2009) (see day 31)
  + Linearly constrained Bayesian matrix factorization(2009) (see day 31)

**Day 30: 30/03/2015**

* An Effective Approach for Mining Mobile User Habits (2010)**:**
  + Define an interaction as an action that the user is doing w.r.t his smartphone. A context is a context of metadata elements such as time and place. The goal is trying to guess a context that implies an interaction.
  + This is done by association rule, where all the possible contexts are enumerated and the counts of each context with each iteration are computed.
  + Critic:
    - Only the context leading to a user-phone interactions are learned. This means than the context leading to another user behavior than the user-phone interaction (as activity) are not learned in function of the context.
    - The past event can influence the current user-phone interaction. For example, a user can be used to open Facebook app and then a news app. This is not taken into account in this paper.
* **Reflection & ideas:**
  + The above paper was selecting a target (which is an interaction) and the goal was to learn the context that implies this target. In our work, we don’t have a target because we don’t want to only learn users’ interactions with the phone, but we want to be able to learn all the features. This means that all the features are in the same time contexts and targets. We are for example interested in knowing on Sundays morning. We can use this information to tell him in advance that in Sunday it is raining so that he can schedule his running time in another moment.
  + This makes our problem different and a little harder to state. In fact we cannot use the association rule method because we do not have a specific target on which we learn those association rules. We need to detect features realizations that are correlated. The first idea is to use the SVD or PCA, the second is to use a similar approach than used in this paper using the entropy to select the important features and clusters.
  + **SVD & PCA (The following representation of SVD & PCA is new and may be better than the older proposed representations):** each feature can be represented in a dimension of a vector having the size of the different features (Location, App\_Launch, Notification, Time, Activity…). The values taken by the vector belongs to a finite set of integers where each integer maps a realization of the concerned feature. To avoid different scales of features (where a feature integer can go until 100, whereas the other has only 2 different realizations) we map the integers into the set [0,1] by equally dividing the space [0,1] through the size of the support of the feature.
  + **Entropy:**  if some realizations of feature X influence the realizations of feature Y, then the distribution of X and Y will be more predictable than the distribution of Y. We use this observation to detect the features that influences each other. We select a first feature F1 and compute it’s entropy H(F1), then add a second feature F2 that minimizes H(F1,F2). Then verify that all of the current features taken into account improves the entropy (it can happen that after adding F3, F1 does not change the entropy, in this case remove it). Do this until any feature improves the Entropy. Thus we end with a set of the features that really influences the behavior of the user. To find a cluster, we could simply take the most predictable realizations of (F1, …, Fn), remove the onces which their values do not change the probability and end up with sets of cluster. However need to verify that the result is deterministic (which means that every time the same set of features is selected)

**Day 31: 31/03/2015**

* **There is a need to reduce the scarcity of the data:**
  + Decide on which metric will decide the location (GPS, base station, combination): only keep one metric that represents the location
  + Decide on the other features (what to keep, what each one represents)
* Taking into account the **SVD/PCA (matrix representation)** implementation, want to **make a note:**
  + **There are features where the different values taken reflect the variation of those features**. For example, for the feature “time\_of\_the\_day” if we have two realizations one that is 0.2 and the other that is 0.3 this represents how much the two times are close**. For other features, the values taken represent categories and do not reflect the variation on that feature**. For example for the feature “activity”, the value 0.2 could represent RUNNING and 0.3 DRIVING A CAR. This means that the closeness between the values in those features do not represent the closeness of the realizations. **For a matrix representation**, either we use the SVD, PCA or a probabilistic model, **the values taken by the features represents the importance or the amount of that feature in a given record**. So that it will be considered that two contexts that have “activity” respectively 0.2 and 0.3 will be more clustered together (considered closer) than two contexts that have “activity” values 0.2 and 0.9, which do not reflect the reality.
  + This problem may influence or not the results we have, need to keep it in mind in case we want to improve the results.
  + Need also to keep it in mind in the moment of the discretization of the features to try to give the indices in the best possible way.
* A Habit Mining Approach for Discovering Similar Mobile Users (2012):
  + From the users smartphone logs, they define a similarity metric between the users.
  + First, they use their previous work “An Effective Approach for Mining Mobile User Habits (2010)” as an input to their current problem. Each user has represented its lists of behaviors. Then taking into account this list of behaviors they try to cluster users.
  + The behaviors are too spare between the users so first they label some features such as locations that are labeled in Home and Work and also the interactions as social, games, telephony ect…
  + Second they use latent variable to catch super behaviors
  + Critic & remarks:
    - They use a work already done to automatically label home place and work place of users: **Discovering significant places from mobile phones(2009),** another (not used) resource is **adaptive on-device location recognition(2004)**
    - To classify the category of interactions they use an automatic labelling method that takes as input a seed (which is a subset of labeled data) and using the web knowledge will automatically label the others: **context-aware query classification (2009).**
    - The interaction categories comes from the nokia taxonomy ovi store (**www.ovi.com**)
    - They use the Bayesian matrix factorization to discover the super-behaviors from the hidden parameters. This method decomposes the matrix X into 2 matrices H and W. However this method gives a probability distribution of the realization of H and W. Moreover this method enables the computation of matrix factorization under the presence of certain constrains (inequalities). **Need to have a deeper look into this method**, it’s advantages and disadvantages. The paper that talk about it is **Linearly constrained Bayesian matrix factorization(2009).**

**Day 32: 01/04/2015**

* Decided with Fabien to go for an SVD as a first approach

**Day 33: 02/04/2015**

* In the wifiConnected there is a strange wifi with a name 0x a mac address 00:00:00:00:00 and a link speed of 4294967295 Mbps (impossible!) which occurs to be the same value as its networked. This is for the test user.
* Thinking on how combine the GSM location info, the GPS info and the WIFI to extract the locations in the best possible way to increase the accuracy of the location and the time coverage.

**Day 34: 07/04/2015**

* Wrote a detailed plan and pseudocode on how to combine the different informations between the WifiConneedAp, the WifiApps, the Location and the GSM\_base\_station\_location
* Saw in some users that the WifiConnectedApp stays multiple hours non changed (closer than half a day). Maybe need to add a rule in the cleaned version dataset where we are able to collapse 2 non changed feature realizations when the difference between the two do not exceed t time (t could be 1 hour)
* Think on how to collapse the AndroidActivity and SonyActivity:
  + First extract statistics to see how much they diverge

**Day 35: 08/04/2015**

* Saw that in the original dataset there is a lot of realizations that occurs are the same and that occurs after a long time. We were used to collapse them and to consider that in the meantime the realization value is the same, but this is not realistic.
* We created a new version of the dataset where we collapse only if the meantime between two realizations is less than 2 hours.
* Moreover, there is cases where record r1 that occurs before r2 has some realizations that occurs after r2. We faced also this problem

**Day 36: 08/04/2015**

* Thought about the selected values and features for the SVD
  + The indices of the different attributes that will form a vector
  + The values and the meaning of of each value that could be taken by this vector (see notebook): Without details, each vector entry can take 1 if it is present, -1 if it is absent and 0 if it is unknown
* Thought about a new version of the data: the categorized data version from which an automatic matrix can be extracted.
  + For each feature, the attributes will take an integer representing the indice of the vector that should be equal to 1. A -1 means that the value is unknown.
  + Each feature contains metadata information that indicates for each attributes the meaning of each indice

**Day 37: 09/04/2015**

* Prepared the framework for the Categorized data extraction as a Notifier-Observer design pattern where each observer is responsible for extracting one feature.
* Will be dealing with the location one.

**Day 38: 10/04/2015**

* Started Coding the LocationTransformer Class.

**Day 39: 13/04/2015**

* Finished writing the LocationTransformer code: the code responsible for extracting the location.
* Filtered the bad values in the cleaning data version (so as the 0,0 gps coordinates, the “” ssid) and prepared a framework to be able to blacklist some values easially
* Starting debugging it

**Day 40: 14/04/2015**

* Debugged the location extraction code; should be clean now
  + **A lot of problems due the passing per reference objects in Python!!**
  + **Need to check if there are such problems in the cleaning version of data**
* Observing the results thought **about integrating the unique gps points clusters to bigger ones.**

**Day 41: 15/04/2015**

* Discovered another bug in the code coming from the data:
  + **The assumption that each sequence ID defines a record is false:** There is some different records that have the same sequence id
  + **Either take into account the times** => remove the sequence id from the cleaned version of the data
  + **or modify the sequence id so that** 
    - **the sequence numbers are unique for each record**
    - **a record that appears after the other must have a strictly bigger sequence number**

**Day 42: 16/04/2015**

* Will go for modifying the sequence id in the cleaned version of the data. Even if this will result in a bigger change in the code and will result in going through all the data one time more, it is a good way for making us verifying that the cleaned and the categorized version of the data are done well.
* Did that but realized that it may occur that two features are in the same record and that they are delayed by more than 1 hour.
  + Computed the distribution of the time variance of the records.
    - A lot of records have big variance (time diff between 2 of their features is very big -> more than 1 hour)
    - There is no a specific feature that causes this, but all of them do that randomly
  + Consequence: the sequence number can not guarantee that a realization that have a seq number smaller than the other must have been occurred before.
    - * **Remove seq entry from the data**

**Day 43: 17/04/2015**

* Seq entry removed
* The wifiApps and the Bluetooth that appear together in the same record have a have a very small variance in the time-> we can assume that all wifi that appear in the same time time have the same date, the same for Bluetooth
* We have the following problem in the location clustering;
  + Imagine the following scenario; we have WifiApps at t1 contains w1 and w2 and at t1+deltat WifiApps contains also w1 (and w3 for example) and between t1 and t1+deltat we have gps1.
    - The w1 and gps1 will not be clustered together (which should be the case).
    - **Need to solve this problem to gain in time coverage**.
    - WifiConnectedApp could be a solution but a look to the feature presence show us that in many times we have the feature WifiApps and not WifiConnectedApp

**Day 43: 20/04/2015**

* Think on Fabian’s case for clusters=> it is okay
* All clusters all well clustered, only problem remaining is that of some gps’s that come only ones without any wifi and station location.

**Day 46: 23/04/2015**

* **Some wifis bssid appear with gps location (station not included yet) that are far by hundreds of kilometers (600 kilometers)**
* Tried with MAC address instead of bssid, we still have the same problem
* Outputted some examples to see where the error comes from:
  + The dates of the locations selected and the wifi overlaps => error do not come from the selection method of samples
  + In the seen wifis, we assign the date of one of the seen wifis to all of them. Verified in the example that the date of the concerned wifi is the same than assigned => error do not come from the assumption: the wifis that appears together in the same record have the same appearance date.
  + The wrong assumption : a wifi is indicative of a place –> supposition: those examples may concern mobile wifis : as lans or sharing 3G connection
  + Any feature allow us to make the difference between those wifis and the others
* A strange wifi with a mac address 00:00…:00 appears in this kind of wifi, should be removed, but it is not the unique cause of this problem.
* Observed a lot of wifis that cluster appeared in many of different close points a lot of times => those are the wifis that are really representative of the location of the user
* The wifis that apprear in very far locations are wifis that were not visited in a lot of number of places (either they contain a location information a lot number of times but in very small time frame range of they just contain a small number of locations)
* For the wifis that contain a small number of locations (1) that were only visited a small number of times, we do not know if they are representative of the location or not => if we observe them without location can we suppose that the location is the one observed or not.
* The only thing we can do is to have an amount of certitude on the wifi that really represents the locations. For that two parameters are to be taken into account:
  + The number of times we have information about the location when observing a wifi => frequence of observation
  + The time variability on which we observed those locations => time duration
  + The distance that separates the different locations observed with a certain wifi
    - * If for a given WIFI, we observed a lot of times locations associated with it, spread during a long time interval, and in small distance range, then it means that this wifi is representative of the time.
      * Otherwise, the WIFI is either not representative of the place, either we don’t have enough information to decide:
        + If the distance is big –> wifi not representative of the place
        + If a lot of points observed during a small amount of time-> it could be that it is a mobile wifi used during a small amount of time and during that amount of time a lot of locations were retrieved -> not sure
        + If a big time range with a small points observed-> we could have observed a mobile wifi only twice separated by a big range of time, and in those two periods we were at the same location -> not sure
* Tried to see if the wifi that regroup gps points that are very far have outliers: i.e if the most of the regrouped gpss are close and a minority only is far => this is not the case, all the gps points are far from each others. It means that the problem is not coming from the recording time or the location but most certainly from the fact that the WIFI is really mobile. This confirms this hypothesis.

**Day 65: 27/05/2015**

There is some different combinations of features or attributes that should be tested by the SVD matrix. Here we list them:

* **Bluetooth:**
  + A Bluetooth device can be defined by:
    - Combination of MAC and name
    - Name
    - MAC
      * In fact the relation between MAC and Name is not one to one; there is the same name with different MACs and the same MAC with different names
  + The Bluetooth devices that can be represented should be:
    - Only the ones seen and paired
    - Only the ones paired
    - Only the ones seen but not paired
    - Any combination of those
* **Battery:**
  + The type of the Battery could be ignored or not
* **Location:**
  + A network can be defined by:
    - MAC
    - Network name

**Day 66: 28/05/2015**

* After having observed the presence of the different number of sony activities (in logs dir) we decided to:
  + Merge Elevator\_up, Elevator\_down, Escalator\_up, Escalator\_down, Stairs\_up, Stairs\_down in one class called moving\_up\_or\_down
* In activity, if we have walking, then still during 10 seconds then walking again, it is probable that the user is walking, stopped during 10 seconds then walked again, so the whole activity is walking. Think on integrating this kind of knowledge in the data

**09,10, 11/06/2015**

* SVD observation:
  + The concepts separates the features; for example app\_launches in concept 1, notifications in concept 2 ect…
  + The importances are too sparse (very close to each other), there is no features that is much more important than the others in one concept.
  + There is one important concept and the others negligable
* First explication: The dimensions are too close: 2000 features/5000 samples
  + The rate should be nb\_features\*20 = samples -> corpus must be much bigger than words
  + Decrease the feature space
* Second explication: the features are regrouped toghether because there is a strong correlation between the fact that a realization take a 1 and all the others take -1
  + Replace the -1 by 0. We consider that a feature that is absent is the same than if unknown
  + **Done, Interpretation**:
    - The importance of concepts is very spread (with 5 concepts, catch 69% of the energy)
    - The important features in each concept are different-> features no more separated by concepts
    - -> Seems that the hypothesis of the strong correlation is right,
    - -> Seems that it improves the results
    - -> but still the clusters do not seem to be meaningful
    - -> Seems that the very frequent realizations appear with a high importance in all the topics. For example the android notification is very important in all the topics, and this is because it is always present in all of the vector records-> need to compensate that (try Inverse-doc frequency)
* **Dimentions reduced -> done, interpretation:**
  + No remarquable change in results change, only in performance. The features regrouped toghether are still not meaningful.
  + Explanation: we reduced the dimentions by regrouping the less frequent realizazions in the same one (for example we represented the top-20 locations and represented the others in one dimention). The less frequent features will almost have 0 everywhere and will not have any contribution in the SVD. In that sense the results of the SVD do not really change.
* Try the masked array
* Try compensation of the very frequent realizations (inverse doc frequency .log(n/ni)):
  + Tried that, the frequent realizations are completely absent, and this is due to the penalization that seems to be big. Log(5000/4999) = 0.000086, log(5000/2)= 3.3. Moreover do not forget that our realizations take only one so that their score is equal to their idfscore, whereas in the doc, words example the very frequent words are more likely to be frequent inside the same document, which smooth more the idf score of the frequent and the less frequent words.
  + Tried another penalized more softer log(n+1-ni). We capture more energy and we have the frequent features that appear in some concepts (not as idf) but not in all (not as without any transformation). Moreover, we capture 43% of the variance with idc whereas we did 29% in idf (44% in without transformation). This means that the dimensions of the data are more separable with the idc comparing to the idf
* More specifically, the square of each singular value is proportional to the variance explained by each singular vector
* Put to 0 the time features for the vectors where all the other vectors was to 0. This improved the scores:
  + Tested with a small number of features that must be correlated (Location, time and Bleutooth) and observed an improvement
* Think of integrating the following :
  + A very frequent notification has more chance to describe the current behavior of the user that a very rare place. Thus in a record, should give more importance to frequent notification than to a rare place.
* See how to interpret this

**Todo list**

**Week Target:**

* Matrix representation of the data
* Do SVD/PCA
* Start thinking about the entropy approach