Neural network approaches to Reality Mining: The Reality Mining dataset

Alessandro Pomes and Domantas Meidus

December 29, 2017

Abstract

The main goal of this project is to find some correlations between the habits of some students and the type of studies their belong. Our ideas is that we could recognize the address of studies(or a type job for a worker) considering in which place the subjects are used to pass their time during a day. More precisely how many time a student can pass at work,at home and elsewhere .
For this type of consideration team has taken the Reality Mining dataset. Below we'll provide a short description how the dataset is composed. The type of classification taken was divided in two ways: one performed with a Multilayer Perceptron Neural Network and one using first Restricted Boltzmann Machines and then again with the same previous Multiltilayer Perceptron. Our thesis is that if a generative learning algorithm is used to reconized pattern among data before the classification, then a better result in term of accuracy could be reach up.

Contents

1 Description of the problem **2**
1.1 Dataset description 2

2 Description of our approach **2**
2.1 Research 2
2.1.1 Feature Extraction 2
2.2 Missing Data 3
2.3 Classifiers 3
2.4 Validation 4

3 Implementation **5**
3.1 Data Handling 5
3.2 Feature Extraction and Selection 6
3.3 Classification 6
3.4 Restricted Bolzman Machine and Multi-layer Perceptron 7

4 Results and Conclusion **7**

1 Description of the problem

1.1 Dataset description

Reality Mining[1] was an experiment started from 2004-2005 at MIT Media Laboratory. It consisted to detect the the following information of 94 subjects:

- Bluetooth devices (proximity of approximately five meters)
- call logs
- cell tower ID
- application usage
- phone status
- self-report relational data

These data were detected with the help of a particular type of cellphone (Nokia 6600) where was installed some additional software who was able to automatically run an application called ContextLog in the background.
This application was in charge to logs all data described above. Therefore for each subject with this dataset might be extract several information during his day. It might also construct how many other people is meeting in a specific time slap. Mining the reality of almost 100 users raises justifiable concerns over privacy. However, the work in this paper is a social science experiment, conducted with human subject approval and consent of the users.

2 Description of our approach

2.1 Research

Many studies[2] [3] [4] were done for routine prediction structure from every day people life. To reach this interesting mining from this dataset some probabilistic tool were used. Mainly Latent Dirichlet Allocation were used[3] [4]. LDA is a powerful mechanism to extract recurrent behaviors and high-level patterns with unsupervised methodology.
These researches devoted themselves to structuring the data with complete information on the student's specific positions (cell tower) and also to the information of how many people met during a day (with the help of bluetooth recording). To build the models different data structure were used, for instance paper from Ferrari, Mamei 4 used an interesting structure way where for each user are recorded several time-frames and the GSM towers where the user was connected.
Principal problem due to handle the dataset was the amount of missing data. Unfortunately when team tried to extract some features in many cases the informations given was too partial to create a robust dataset. So we decided, inspired latter to professor and former from paper of Nathan Eagle & Alex Sandy Pentland 2 to create Multidimentional-Array using the location during the days.
In the Section Feature Extraction is provided the description on how we organize data.

2.1.1 Feature Extraction

As mentioned above we built two types of classification problems with the same goal: provide a way to recognize subjects studies from his routine.

First case of develop was to consider each subject and for each hour in a day the frequencies that a person could have been in a specific place. So for this type of solution we gathered all days information in a single matrix (for more information on the implementation consult Data Handling subsection).
For the second model development we decide to build a much deeper configuration of the data. For reasons due to the structure of RBM and to preserve information of continuous daily routine we decide to keep the structure of a week with binary values.
Inspiring from literature 2 we place a single week in a long array of 840 dimension. This vector has 5 recurrent structure of 168 dimension that belongs on the places we have in the dataset (Home, Elsewhere, Phone is off, No signal). For each 168 sloth that corresponds a single hour we put 1 if the student were in the place, instead of 0 it he was not(implementation detail here).
In term of clarify this structure the week obtained are random weeks divided to type of student: The Sloan Business student and students belong to MIT. Therefore finally we divided our dataset in 2 hypothetical weekly pattern routine, with the goal to detect it.

2.1.1.1 Feature Selection & Restricted Boltzmann Machines

On the second extension of our implementation we used a Generative learning algorithm to provide a sort of feature selection (actually this type of features mapping is not a properly a feature selection, but we are on purpose abusing this term to explain better what is our intention). To show how a eclectic could be the formulation of Neural Network we have adopted Restricted Boltzmann Machines [5] to achieve a better accuracy in classification problem. Restricted Boltzmann machine is a Unsupervised method highlight a set of latent factors. Instead of learning directly on a continuous scale, using RBM we try to discover latent factors that can explain the "activation" in a particular hour that is when a subject is in a precise place; therefore if there is a precise structure in the domain data. So we were trying to collapse the information of a single week in a some meaningful features. To know this we map the original week vectors in some other trained vector that performs the following probability:

$$p(h_j = 1|v) = \sigma(b_j + \sum_i v_i w_{ij}) \ [1]$$

In the final structure of data we put in the RBM 100 hidden neuron unit that can perform a good result in the process of Multi-Layer Perceptron machine learning algorithm training.

2.2 Missing Data

The insight behind processing our missing data is that we handle our dataset and eliminating them directly in order that does not weigh in learning process. In the first classification we figure out that if the frequencies is computed deleting the Nan value we would not have changed the other relevant information because the values frequencies of the other places would not be changed.
Using these data structure allow us to delete Nan values without missing important information.

2.3 Classifiers

As explained above two different types of classification algorithms have been developed:

[1] σ represents sigmoid function (details on Wikipedia: Sigmoid Function)

- Neural Network Multilayer-Perceptron : Supervised Machine learning system made up by a network of layers of neurons and weighs(linked in the neurons) that are updated every steps of the training session. We built three layers whose the first and the last are the input and the output of our supervised (for specific detail of implementation read here).
- RMB & Multilayer-Perceptron : The classification method here are two and so different. RBM is a Generative Models so its scope is to approximate a distribution probability of the input. Is an Unsupervised method and we don't need to put labels in the learning process. Multilayer-Perceptron that we used are the same of the description above.

In this section we are going to answer on the questions proposed for the project:

1. As we can see in this project NN can provide both Supervised and Unsupervised problem with different goal: usually supervised problem are classification problem (for example with a Multilayer Perceptron) that can provide some labels from a given input. For Unsupervised problem(like RBM) we train directly the dataset without labels. Another type of learning could be Reinforcement that could be considered a subsection of Unsupervised Learning. In fact labels are not already given but generated by an agent's interactions with the environment.
2. Architecture of NN could be synthesized:
   - Neurons: nodes of the network that contain input, output and computed values.
   - Weights: are the connection between different layers, the goals are to achieve an optimal configuration depending on the model proposed in a learning rule.
   - Layer : sets of neurons that belongs a precise type of computation from others different layers across connectivity weights (layers could be also input and output)
3. Neural networks was inspired design of a biological neural network that is a series of interconnected neurons whose activation defines a recognizable linear pathway. The interface through which neurons interact with their neighbors usually consists of several axon terminals connected via synapses to dendrites on other neurons. If the sum of the input signals into one neuron surpasses a certain threshold, the neuron sends an action potential at the axon hillock and transmits this electrical signal along the axon (that is almost the same structure of the Percetron Neural Network use in this protect).
4. Information extracted could be reached with many type of activation functions that could perform some transformation of the output value much more useful respect raw data. Activation function could be: density function,Binary step function,Bipolar step function,Sigmoidal function, Binary sigmoidal function,Bipolar sigmoidal function,Ramp function
5. To learn a Neural Network is using gradient descent algorithm. Gradient descent algorithm use back propagation method to update weight and to reduce a possibility loss function. There are many others algorithm like Contrastive Divergence that is a particular type of gradient descent algorithm used in Boltzmann machines.

2.4 Validation

Especially for second procedure in the process of validation we have developed a structure that need some precise steps in order to split data and keep coherence in the procedure. If it will not take this order some side-effects can be produced.

We decide to split week dataset in two parts: one for training and one for testing. We apply first the train part on the two learning procedure and the we use test set to compute the accuracy.
So the two training operation, RMB before and Multilayer-Perceptron after, can rely on two distinct datasets that remain independent and therefore do not change learning information. Some comments in the Jupyter Notebooks can highlight position were dataset is split.
For the first train model we used a simple division in the test and training set before the learning procedure.

3 Implementation

Reality mining dataset has ~57 sub categories with each subject activity data. However it's unable to use all categories for Neural Network - only few categories data was chosen to train and test for learning. List of categories that have been chosen:

1. **my_affil** - this category of dataset contains each subject affilation type. All possible affilation types:
    - 'mlgrad' – Media Lab Graduate Student (not a first year)
    - '1 st yeargrad' – Media Lab First Year Graduate Student
    - 'mlfrosh' – Media Lab First Year Undergraduate Student
    - 'mlstaff' – Media Lab Staff
    - 'mlurop' – Media Lab Undergraduate
    - 'professor' – Media Lab Professor
    - sloan' – Sloan Business School.
2. **data_mat** - Inferred each subject locations at each hour of the day:
    - 1 - Home
    - 2 - Work
    - 3 - Elsewhere
    - 0 - No signal
    - NaN - Phone is off.

From data_mat dataset patterns which indicates each subject locations at each hours of the day subjects are classified to **sloan** or **no sloan**.

3.1 Data Handling

Dataset is stored in the .mat format, in order to extract this data to python data structures, the program uses scipy.io library. After all Reality Mining dataset is stored to python data structures, **my_affilation** and **data_mat** categories are extracted:
affilation = data['my_affil']
data_mat = data['data_mat']
These categories are used for Neural Network training and testing.

Reality mining dataset has a lot of missing data which is handled by excluding all missing data records in my_affil and data_mat datasets. This way of dealing with missing data prevents any error using Machine Learnings algorithm in cost of losing data. Of all missing data in my_affil and data_mat datasets 47 subjects was excluded.

3.2 Feature Extraction and Selection

Two Neaural Network learning algorithms are used: **Restricted Bolzman Machine(RBM)** and **Multi-layer perceptron**. For **Restricted Bolzman Machine** learning algorithm **data_mat** data will be used as features. To extract these features into required 2-nd dimensional vector, the program performs these steps:

1. Each subject activity is stored to the list if subject data_mat data complies with requirements:
   - data_mat data representing subject activity on each hours is not empty;
   - data_mat data should represent at least 7 days of subject activity.

   These conditions complies 69 subjects of 106 subjects. These 69 subjects data_mat data are stored into **features_list** for further data preprocessing.
2. Excluding NaN values from the **features_list**. Data_mat dataset contains NaN values which indicates that subject phone was off at certain hour. These NaN values are converted to Integer data type, in our program NaN values converted to value 4. Our program is not designed to handle non numeric data types, so in order to perform features vector with necessary data, program converts non numeric value to Integer.
3. Creating Features vector for RBM learning algorithm. Features vector is 2-nd dimensional:
   - 1-st dimension data - Subject
   - 2-nd dimension data - Subject activity of each hour in seven days.

Features for RBM algorithm are represented in a shape of **(subject number, observation days * number of hours in a one day * all possible locations conditions)** which in real numbers are **(69, 7 * 24 * 5) = (69, 840)** , explanation:

- subject number - is all subjects which complies data_mat requirements.
- observation days - number of how many days subject activity is stored.
- Number of hours in a one day - since one day has 24 hours, so the number is 24.
- All possible locations conditions - All possible locations conditions are 5 (Home, Elsewhere, Phone is off, No signal).

For each subject, all 7 days of each hour data is stored for different locations conditions.
At the start features vector for each subject is filled with zeros values. Data for each locations conditions is saved in 0,1 numbers representation - all possible locations are iterated and if at the certain hour of the week subject was in that location, the value of that hour of the day changes from 0 to 1.

3.3 Classification

Since **Restricted Bolzman Machine(RBM)** algorithm is unsupervised learning, it will categorize subjects by itself by their mat_data data patterns. Data for **Multi-layer perceptron** supervised learning algorithm are classified into categories: **sloan** and **no sloan**.
For sloan category belongs all subjects who affilation type is equal: *'sloan' or 'sloan_2'*. For no sloan category belongs all subjects who affilation type is equal: *'mlgrad', '1 st yeargrad', 'mlfrosh', 'mlstaff', 'mlurop', 'professor'*.
In total 69 subjects are analyzed and classified in this order:

- Sloans: 20 subjects
- No sloans: 49 subjects

3.4 Restricted Bolzman Machine and Multi-layer Perceptron

Restricted Bolzman Machine learning uses this representation: **BernoulliRBM(n_components=100, verbose=True, learning_rate=0.1, n_iter=50)**, where:

1. n_components - Weight matrix, where features in the number of visible units and components is the number of hidden units.
2. learning_rate - The learning rate for weight updates.
3. n_iter - Number of iterations over the training dataset to perform during training.

After executing this BRM function, results are transformed and saved to list in order to use this data for Multi-layer Perceptron neural network as features.
Training and validating Multi-layer Perceptron neural network:

1. Split labels and features data into testing and validation parts.
2. Define parameters for MLP:
   - inputs - number of training features data.
   - n_hiddens - number of hidden neurons.
   - outputs - number of classifiers. This number is equal 2, because there are two labels: sloan and no_sloan.
   - n_epochs - Number of epochs which is a measure of the number of times all of the training vectors are used once to update the weights.
   - batch_size - defines number of samples that going to be propagated through the network.
3. Define output layer.
4. Define **loss function**. Loss function is a performance metric on how well the Neural Network manages to reach its goal of generating outputs as close as possible to the desired values.
5. Implementing Gradient Descent Optimizer which updates the weights towards less and less global loss function.
6. Learn weights using current batch.
7. Compute accuracies in the training and validation sets using tensorflow.

4 Results and Conclusion

The results obtained from our project have shown too randomness to create strong convictions in trying out the initial proposition in the abstract. The main problem is that the casual division in the to sets(Training and testing) could generate aleatory.
Anyway we can observe some interesting facts. While in the first procedure we can observe an accuracy of 0.6 (not so good as imaged) in the other method the accuracy is floating and can

depend from many influences.
On average the accuracy reach from the second method is in a range of $\approx 0.7 - 0.8$ that is actually a good improvement respect the first method. The amazing thing that can be a source of future developments is that there seems to be a correlation between the quality of RBM learning (considering maximum pseudo-Likehood) and final learning through MLP. In fact, if we try to put a few iterations and to maintain a low Likehood (that is, not training the net) the end result of the perfection decreases and is evaluated in a range of $\approx 0.5 - 0.6$ that is just like to flip a coin. So it can be said that if the RBM network can be useful to identify a probable pattern, but that one can't be considered evident by the presence (perhaps) of a few and noisy data.

Notes

[1]Nathan Eagle, Alex Pentland, and David Lazer. Inferring Social Network Structure using Mobile Phone Data, Proceedings of the National Academy of Sciences (PNAS), 2009, Vol 106 (36), pp. 15274-15278

[2]Eigenbehaviors: identifying structure in routine Nathan Eagle & Alex Sandy Pentland Received: 12 September 2007 / Revised: 24 February 2009 / Accepted: 24 February 2009 / Published online: 7 April 2009 Springer-Verlag 2009

[3]Probabilistic Mining of Socio-Geographic Routines From Mobile Phone Data Katayoun Farrahi, Member, IEEE, and Daniel Gatica-Perez, Member, IEEE

[4]Classification and prediction of whereabouts patterns from the Reality Mining dataset Laura Ferrari, Marco Mamei Dipartimento di Scienze e Metodi dell'Ingegneria, University of Modena and Reggio Emilia, Italy

[5]Geoffrey Hinton 2010.August 2, 2010 UTML TR 2010–003, "A Practical Guide to Training Restricted Boltzmann Machines" Geoffrey Hinton Department of Computer Science, University of Toronto