Using Random Fourier Features with Random Forest

Albert Ribes

March 19, 2019

Contents

1 Context

1.1 General Framework

Machine Learning uses statistical models to make predictions or decisions when there is no known formula of feasible procedure to find the correct answer. In the supervised learning sub-field, it uses a collection of data instances and their corresponding answer to predict the correct outcome for new unseen data. This is known as learning.

The theory behind this process has been developed for many time and is quite old. But it has not been possible to use it in real world problems until the recent years, when the computational power of the machines has grown so much that it is able to perform most of the calculations needed to "learn" with a decent level of accuracy.

Moreover, nowadays it is relatively easy to find or produce huge datasets about almost any field, and that makes it possible for many businesses to learn from this data and to make better decisions.

Nevertheless, the current level of learning is not enough. The error rate is still too high for some applications, and it is needed a lot of computation time to train a model. There is still much work to do in this field.

1.2 Into the specifics

The error made by an algorithm may be caused by inherent limitations of the data (such as random noise, lack of information, mistakes in data collection, etc.) or by limitations in the mathematical model behind the algorithm, such as having a lot of bias or variance. One way to reduce this last problem is to use some kind of ensemble of predictors.

Bagging[1] is an ensemble technique which has shown very good results for some algorithms. It is known to substantially reduce the variance of a model when each of the estimators has very little correlation with the others. It achieves that by using bootstrap (a random sampling with replacement), but for most of the models this is not enough. This is why it is rarely used for any model but the Decision Tree. The instability in this algorithm helps each of the estimators to have low correlation.

A kernel function [7] is a function $\mathbb{R}^n \mapsto \mathbb{R}$ between two vectors (or a different structure) which is equivalent to the dot product of some transformation of the vectors: $\kappa(x, y) = \phi(x)\phi(y)$. It is primarily used with Support Vector Machines to "move" the data to a different vectorial space where data is expected to be linearly separable. It allows to solve non-linear problems with a linear method. The implicit function $\phi(\cdot)$ could have infinite dimensionality, so it is not always possible to use it explicitly.

There exists a mapping to transform data from a given space to a different one which is called Random Fourier Features[2]. It can approximate any shift-invariant kernel function with a randomized mapping, and has successfully been used to decrease the error rate of a Neural Network[4].

In this project I try to use this mapping to increase the accuracy that can be achieved by models using the bagging ensemble. It is expected that it will enable other models than Decision Tree to produce uncorrelated estimators and thus reduce the variance of the model.

1.3 State of the Art

Support Vector Machines [3] were quite popular a few years ago. With the use of the kernel trick, they proved to have a big ability to learn and generalize with complex problems.

However, it was soon obvious that the computational cost of training a Support Vector Machine was not acceptable since it is cubic with the number of instances. With time, they lost popularity.

The new big wave is led by Deep Learning. It has showed a huge capability to discover hidden structures and patterns and also to generalize. It scales better to big datasets, but it still is very expensive to train a Deep Neural Network. It has succesfully been used for unsupervised Learning [5].

Random Fourier Features have showed to be a good way to approximate a kernel function without losing too much accuracy, but they haven't been used too much. A recent study used them in conjuntion with Deep Neural Networks in order to learn a hierarchy of layers of nonlinear features, showing very good results [4].

In short, we can say there is a battery of good techniques which are used on their own to solve specific problems, but there is very little effort to combine them together to produce a better model. This is what we will try to do in this project.

1.4 Problem to solve

Machine Learning has two main objectives: be able to learn, and to do it fast. But most of the times trying to increase one will be with a cost of decreasing the other. The problem is to find a good equilibrium between these objectives.

There are two main ways to cope with this. The first one is to try to produce better datasets. A good dataset will contain more relevant information, more precise, cleaner of unknown values or simply with more instances.

The other way is to try to produce mathematical models which are able to generalize better or which require less computation time to learn well.

In this project we work on this second method: we try to decrease the computational time of training a model without decreasing too much the accuracy.

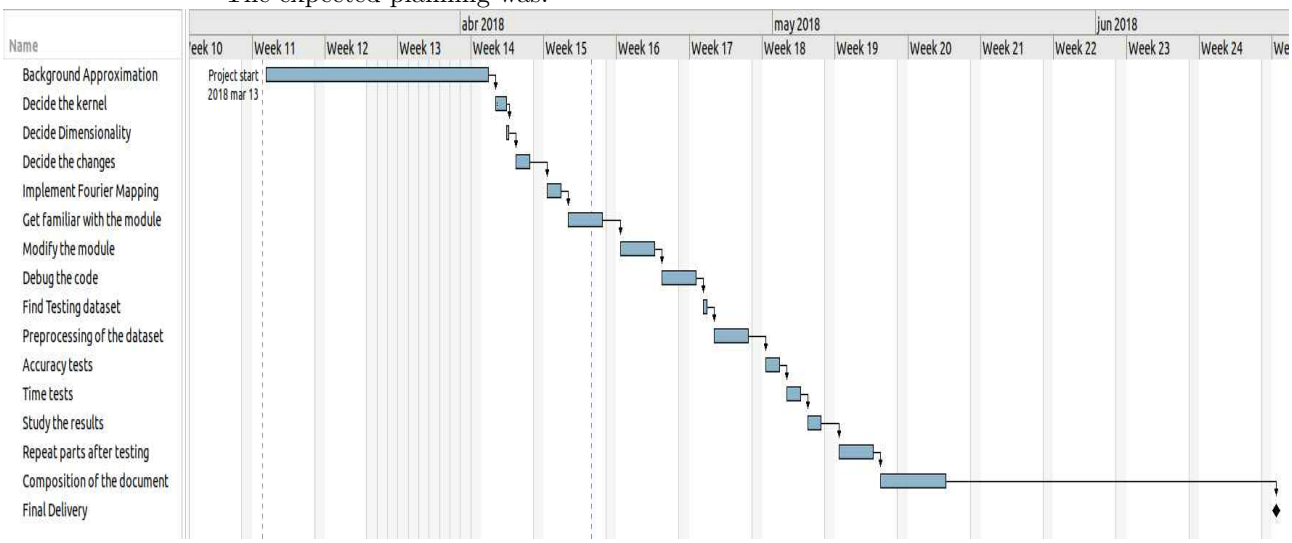
2 Planning

2.1 Original Planning

In the original planning of this project the scope was expected to be limited to the study of Random Fourier Features with the Random Forest algorithm.

After checking that performance could not be improved with this model, it was changed to cover also other models, such as Logistic Regression and Support Vector Machines.

Furthermore, I expected to deliver it on June 2018, and that was not possible. The expected planning was:



2.2 Problems encountered with original planning

I encountered many problems with the original planning, both in the setting and in the execution. The problems were:

- **Very few knowledge of the study field**: in the beginning of the project I didn't have a clear image of how the whole project would be. I lacked much of the knowledge that I would need in the project, and was not able to write a realistic planning.
- **Naive assumption of success**: I expected to find good results on the first experiments I performed and didn't have a plan for when they failed.
- **Programming time underestimated**
- **Meetings with the project director**: I didn't have a good planning on when to meet the director of the project, and time passed without advancing on the work.
- **Communication problems**: for some time it was not possible to meet the director of the project physically, due to a sick leave of 3 months. For this reason the communication was too slow.
- **Lack of initiative**: during the whole project I've been blindly following the path proposed by the director, not knowing where we were going.

Task		Time (h)
Accuracy Tests	Datasets comparison	5
	Black Box vs. White Box	10
	RBF-SVM vs. Best defined model	5
	Bag vs. Ensemble	10
Timing Tests	RBF-SVM vs. Defined models	8
	Random Forest vs. DT best model	7
Finish the project document		50
Unexpected extras		15
Total		**110**

Table 1: Remaining tasks

Thus, I needed constant feedback and was not able to advance without his advices.

- **Lack or rigour in following the planning**: the planning was not checked during the project.

2.3 Proposed new planning

As the contents of the project have changed from the original planning, it is needed to define a new set of tasks to finish the work. The ones which still need to be done are showed in Table 1.

3 Methodology

3.1 Original Proposed Methodology

The original proposed methodology was to define a set of tasks which should be done by the end of the project and a correct arrangement of them to ensure all of them can be done.

3.2 Problems encountered with original methodology

The problems encountered with the original methodology are similar to the ones in the planning.

For the one hand, by the time I defined the set of tasks I didn't have a clear idea of how the project would go. I lacked a lot of knowledge about the study case and planned the tasks assuming that I would obtain some results which I never got. Therefore, most of the tasks defined where useless and other ones should have been there.

By the other hand, there was not a strict monitoring of the tasks that were done and the ones that where missing.

3.3 New methodology

I will continue to use the same methodology, but having fixed the problems I had.

Now I have a clear understanding of the ins and outs of the project, so the new planning is expected to meet the requirements. Moreover, I do now follow the planning and meet the director of the project every week.

4 Alternatives Analysis

4.1 Language for development

Although any programming language could potentially be used for Machine Learning, there are some characteristics that make them more suitable for it.

As there is a lot of trial and error, it is very comfortable if you use an interpreted language, which most of the times can be run in interactive mode.

Besides, it is very helpful if there is a big community which is already using that language for Machine Learning, since a lot of code will already be written and there will be fewer bugs.

The two main options with this characteristics are Python[8] and R[9], and both have pros and cons:

Pros	Cons
Very fast	Machine Learning algorithms could be more robust
Easy to program with it	Doesn't have a graphical interface by default
Very big community	Need to import many modules for simple tasks
I am comfortable with it	Lack of support for a native categorical variable
Open source	Not known by the director of the project
General purpose	

Table 2: Pros and cons of Python

Pros	Cons
Easy to learn	Not intuitive for a programmer coming from another language
Well integrated with graphical interface	Very slow
Very robust and well tested libraries	The majority of the community comes from statistics
Open Source	Many different modules to do the same thing
Well known by the director of the project	There isn't a standardized interface for calling functions
	It is very biased towards statistics
	Hard to use outside of RStudio

Table 3: Pros and cons of R

I have chosen Python as the main development language. The main reason is that it is a language I already know well and that it is much faster than R.

The only drawback is that the Machine Learning module for Python still has some issues with the code.

4.2 Running environment

This is not a very important decision, but it may contribute to make the programming process faster and more comfortable.

The options range from very simple and flexible to some more sophisticated ones. The ones I considered where: **text editor and console, Jupyter Notebook** and **Atom's Hydrogen package**.

Pros	Cons
Very simple	Not very interactive
Can use whatever text editor I want	Graphics integration not enabled by default
Very scalable	

Table 4: Pros and cons of text editor and console

Pros	Cons
Very easy to integrate code, graphs and documentation	A little bit buggy
Easy to show results to the director	The interface for writing code is not comfortable
Open Source	Not scalable for large projects
Very interactive	
Extensions and add-ons to make a GUI	
Very agnostic of the code	

Table 5: Pros and cons of Jupyter Notebook

Pros	Cons
Integrated with my favourite text editor	Not easy to export the results to another format
Nice graphs and programming experience	Makes programming bug-prone

Table 6: Pros and cons of Atom's Hydrogen

I finally chose Jupyter Notebook because of the good programming experience I have with it and the easiness of seeing and showing the obtained results.

Concerning the problems with large projects, I have solved it by using a module architecture and writing the modules in separate files, with a normal text editor, and importing them to Jupyter

4.3 Machine Learning Algorithms

There is a big range of options to test with the ideas of this project. However, it is not possible to test all of them, so I need to get a subset.

The considered algorithms were: **Logistic Regression, Naive Bayes, K-Nearest Neighbours, Support Vector Machines, Decision Tree**.

The final decision had in mind the needed number of hyper-parameters, the training speed and a very opinionated thinking of which of them would show interesting results.

The ones I chose were **Decision Tree, Logistic Regression** and **Support Vector Machine with a Linear Kernel**

5 Knowledge Integration

This project is totally biased towards Machine Learning, so it has required knowledge from many subjects related to it. It addition, I have required programming skills to write and assemble all the code. The following subjects have helped me to develop the project.

APA - Machine Learning
Almost all the knowledge for this project comes from this subject

PRO1 and PRO2 - Programming
For obvious reasons: programming skills were required for this project

IES - Introduction to Software Engineering
When a program becomes very large, it is needed some kind of architecture and program desing to avoid bugs and have clean code

MD - Data Mining
Theory on Decision Trees and Random Forest
Theory on PCA
Development tools like Jupyter Notebook

PE - Probability and Statistics
The foundations of Machine Learning are based on this science, and it uses a lot of methods from it

6 Implication and Decision Making

6.1 Meetings with director

I started to do this project without setting any periodical meeting with the director of the project. As a result of this, it was difficult for me to force myself to work on time.

After realizing of that, I changed strategy and tried hard to meet him each week.

I distracted again for a couple of times and let the time pass without working as expected.

6.2 Goals achievement

Although there haven't been goals defined in the long term, there have been many in the short one. As I was meeting the director each week, there was always some work to have done for the next one, and in general I have been bringing the work which was agreed.

6.3 Rigour in scientific procedures

In general, I think I have been very rigorous in the methods and procedures that I have used during the project. I have been careful to read all the documentation available to check that what I was doing had the expected behaviour, and when that was not possible, I checked directly the code to understand each operation.

I have also read many scientific books in order to confirm and increase the knowledge that I had over a field, and I have been careful to consult each step with the director when I was not sure.

The code that I write is very clean, and I have tried not to use any personal recipe, but the standard and well known methods.

The only point I can think of where I may not have been rigorous is to tend to use the default values of the functions I call in the code for some hyper-parameters. It would have been more correct to perform some kind of cross-validation.

7 Laws and regulations

7.1 My responsibility

Software I have used:

- **Python Programming Language**: very permissive license compatible with GPL[10]
- **Library Scikit-learn**: uses 3-Clause BSD License[11]
- **Jupyter Notebook**: uses 3-Clause BSD Licence[12]
- **Python modules**: Numpy, Pandas, Matplotlib, etc. Permissive licenses

As all the software I use is Open Source, there are very little restrictions on its usage. The only important one is not to use the name of a product for a modification I make on their code. I won't.

7.2 Others responsibility

All the code for this project is publicly available in a GitHub's repository. I've licensed it with the MIT license, which is Open Source, so I give permission to anybody to do pretty much anything they want.

References

[1] Leo Breiman, *Bagging Predictors*, Department of Statistics, University of California at Berkeley 1994.

[2] Ali Rahimi and Benjamin Recht, *Random Features for Large-Scale Kernel Machines*, Advances in Neural Information Processing Systems, 2007.

[3] Vladimir Vapnik, *Support-Vector Networks* 1995

[4] Shuai Zhang, Jianxin Li, Pengtao Xie, Yingchun Zhang, Minglai Shao, Haoyi Zhou, Mengyi Yan, *Stacked Kernel Network*, Beihang University, Carnegie Mellon University 2017.

[5] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, Demis Hassabis. *Mastering the Game of Go without Human Knowledge*, DeepMind, 2016.

[6] Ji Wan, Dayong Wang, Steven Chu Hong Hoi, Pengcheng Wu, Jianke Zhu, Yongdong Zhang, Jintao Li, *Deep Learning for Content-Based Image Retrieval: A Comprehensive Study*, 2014.

[7] Thomas Hofmann, Bernhard Schölkopf, Alexander J. Smola *Kernel Methods in Machine Learning* 2008

[8] https://www.python.org

[9] https://www.r-project.org

[10] https://docs.python.org/3/license.html

[11] https://github.com/scikit-learn/scikit-learn/blob/master/COPYING

[12] https://github.com/jupyter/jupyter/blob/master/LICENSE