TRINITY COLLEGE DUBLIN, SCHOOL OF MATHEMATICS

5691 Seminar Report
Multiple Kernel Learning

Gustavo Ramirez

February 13, 2017

1 Introduction: Machine Learning

Machine Learning is a lot of things, and the field is constantly expanding. The boundaries of it are constantly expanding/changing. But it can be almost defined (although, it depends very much on the context). Like Arthur Lee Samuel said:

[Machine Learning is the] field of study that gives computers the ability to learn without being explicitly programmed

From the previous quote, it is obvious that the development and implementation of Machine Learning is a first step towards Artifial Intelligence. On the other hand (and in a less general way), Tom Mitchell (Carnegie Melon University) said:

A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E

To understand the previous quote more explicitly, think of fitting data using a straight line: the experience E can be the data used in obtaining the two parameters for the straight line, the performance P can be the correlation r, and the idea is that Machine Learning (simple linear regression in this very specific example) is useful if the correlation improves when that straight line is used again with another set of data (the task T, of course, is the prediction of results in the specific problem to be implemented).
Two major problems (see [1]) that Machine Learning solves are: regression and classification. The tool presented here (Multiple Kernel Learning) is useful for an algorithm solving any of those two. The *regression* problem is basically a fitting problem, and the *classification* is characteristic of systems where we seek a yes-or-no prediction (e.g. in a group of cells, which are useful and which aren't, for some specific task).
For a better understanding of what Machine Learning really is, it is important to compare it with other branches of data science; figure 1.1 (taken from [4]) accomplishes this. From that figure, it is interesting and important to note that *Databases* has no overlap with Machine Learning.

2 Theory: Multiple Kernel Learning

Multiple Kernel Learning (MKL) can be thought of as a tool, more than as an independent algorithm; it is an algorithm, but it is usually added to regression and classification algorithms, to improve their prediction power.

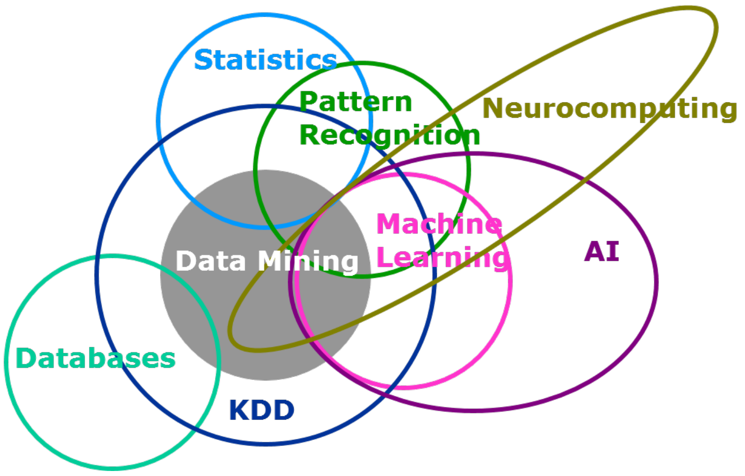

Figure 1.1: Machine Learninng as compared to other areas of data science.

In a technical sense (see [3] for a more extense explanation of MKL), MKL can be defined as a set of machine learning methods that use a predefined set of kernels and learn an optimal linear or non-linear combination of kernels as part of the algorithm. In order to understand the somewhat cryptic previous sentence, MKL will be introduced here with very simple and commonly known tools.

2.1 Support Vector Machines

One way of solving the classification and regression problems, is using what is know as Support Vector Machines (SVM). SVM consists in that, given labeled training data, the algorithm outputs an optimal hyperplane which categorizes new examples. The rest of this subsection tries to explain the previous statement more clearly.
Imagine data as in figure 2.1 (taken from [5]), but now allowing a gap of size ϵ around the fitting line; in this case the SVM algorithm consists of a linear fitting, with the added property of summing the squares of distances only up to the dotted lines, and not up to the continuous (fitting) line. Once the algorithm is implemented, the data points which are on the dotted lines are called *support vectors*, hence the name *Support* Vector Machines.
The specific details of SVM are not important for understanding MKL, but there is a very important resultant property of the SVM algorithm:

the evaluation of the relevant parameters of the model, depend exclusively on the dot products of the data

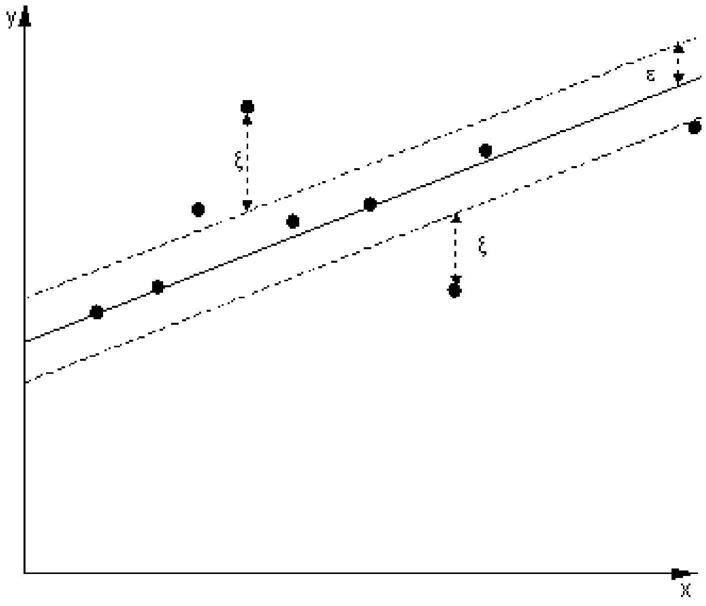

Figure 2.1: SVM for linear regression.

More specifically (and to understand better the previous statement, but without going throught the whole SVM algorithm): if the dependent variable y is to be modeled as a function of the independent variable x, and there are n points of data, then, each of the n points is characterised by a vector of the form $\vec{x}_i = \begin{pmatrix} x_i & y_i \end{pmatrix}$ (working in 2D here). Even more, the previous statement about dot products in SVM, means that the evaluation of the algorithm depends only on the elements in the following matrix (called the *Kernel matrix*) K:

$$X = \begin{bmatrix} \vec{x}_1 \\ \vec{x}_2 \\ \cdots \\ \vec{x}_n \end{bmatrix} \rightarrow K = XX^T \tag{2.1}$$

It is very important to understand that the dependance on only dot products, applies both to linear regression and linear classification, using SVM.

2.2 From SVM to MKL

There are algorithms, such as SVM which, when evaluating the data to find the parameters for the solution of the implementation, depend only on the values in K (as introduced in equation 2.1).

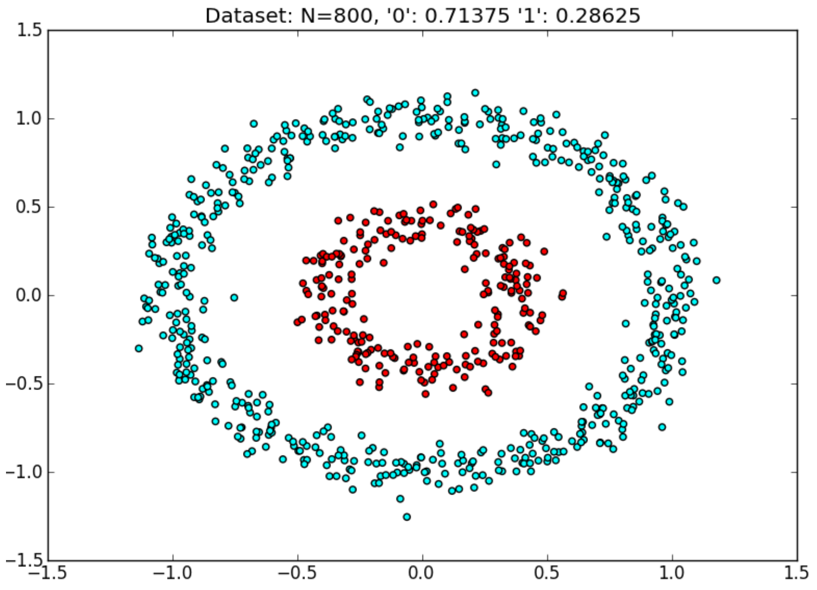

Figure 2.2: data not-linearly separable.

MKL is based on what is called *the Kernel trick*. The Kernel trick is basically taking advantage of the feature mentioned before, that the evaluation of some algorithms depends only on dot products of the data.

2.3 The Kernel trick

If the available data is as in figure 2.2, it is clear that those points are not separable linearly. On the other hand, if the number of dimensions is expanded as in figure 2.3, then a hyperplane can be found, which separates data appropriately. The Kernel trick consists (partly) on performing such a change in dimensions.
The other important feature of the Kernel trick comes from the form of the K matrix. At this point, it is ideal to perform the dimensional change, in such a manner that the K matrix keeps its form (why? Simply because the specific algorithm, SVM for example, can be reused if K keeps its form), i.e. if the transformation is a function ϕ such that: $\phi : \mathbb{R}^2 \to \mathbb{R}^n$, then, K should have the form:

$$K' = \begin{bmatrix} \phi(\vec{x}_1)^T\phi(\vec{x}_1) & \phi(\vec{x}_1)^T\phi(\vec{x}_2) & \dots \\ \phi(\vec{x}_2)^T\phi(\vec{x}_1) & \dots & \dots \\ \dots & \dots & \dots \end{bmatrix}$$

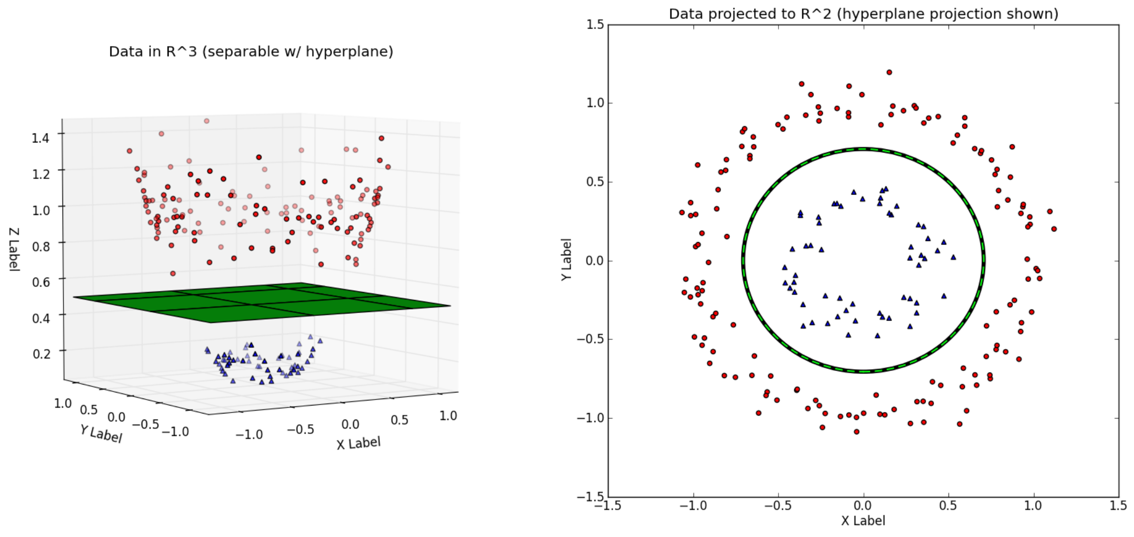

Figure 2.3: separation of data throught the use of a dimensional change, through a hyperplane.

in such a manner that those dot producs (of the form $\phi(\vec{x}_i)^T\phi(\vec{x}_j)$) can be evaluated easily in terms of the dot products of the original K matrix (i.e. terms of the form $\vec{x}_i \cdot \vec{x}_j$). In compact form, the Kernel trick consists in restricting the transformation ϕ by the following:

$$\phi(\vec{x}_i)^T\phi(\vec{x}_j) = f(\vec{x}_i \cdot \vec{x}_j) \tag{2.2}$$

Here is a simple example on using the Kernel trick for an specific transformation from 2D to 3D:

- $\phi : \mathbb{R}^2 \to \mathbb{R}^3$
- transformation ϕ: $(x_1, x_2) \to (z_1, z_2, z_3) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$
- take ($\vec{r}$ and $\vec{s}$ are 3D, $\vec{a}$ and $\vec{b}$ are 2D): $\vec{r} = \phi(\vec{a})$ and $\vec{s} = \phi(\vec{b})$
- $\Rightarrow (\vec{r}\cdot\vec{s})_{3D} = r_1s_1 + r_2s_2 + r_3s_3 = (a_1^2)(b_1^2) + (\sqrt{2}a_1a_2)(\sqrt{2}b_1b_2) + (a_2^2)(b_2^2)$
- $\Rightarrow (\vec{r}\cdot\vec{s})_{3D} = (\vec{a}\cdot\vec{b})^2$

and then, as shown, the dot products in 3D (after the transformation ϕ is taken) can be evaluated in terms of the dot products in 2D, only by squaring them.
As can be seen from the previous example and from equation 2.2, **no explicit knowledge of ϕ is necessary**, but the only thing that matters is the form of f. This is very powerful, as it allows

to make use of extra dimensions, without having to compute the transformations driving the system towards the new dimensionally-extended form; but even more, it is very important, as there are functions $f(x)$ which lead to very complicated transformations $\phi(x)$, or even taking the system to an infinite dimensionality (i.e. $e^{\vec{x}\cdot\vec{y}}$).

2.4 MKL

The idea of MKL is making use of the Kernel trick, but now with a set of different functions: $\{\phi_1, \phi_2, ..., \phi_m\}$.
Usually, in algorithms such as SVM, MKL can be easily plugged in. In simple SVM for linear regression, Lagrange multipliers are used for optimizing and reducing the error; this approach can be extended for integrating MKL into the mechanism, adding one extra Lagrange multiplier for each ϕ_i function.

3 Applications

Machine Learning is used in a lot of different areas (physics, biology, trading, etc.), solving a lot of different problems; the same goes for MKL, and specifically for the Kernel trick.
Extensions of MKL are frequently taken to solve complicated problems. For example, in reference [2], a study on cancer can be found, for which 44 different algorithms were used (and compared) in the prediction of sensitivity of cancer in response to the application of drugs; the algorithm which best performed was an extension of MKL using Bayesian Methods.

4 Conclusions

- Multiple Kernel Learning is useful in algorithms such as SVM, where the evaluation of the parameters of the model makes use only of the dot products of the data vectors.
- The Kernel trick consists in a dimensional extension from d to l, for which the form of the Kernel matrix K (containing the relevant information to solve the system) is preserved, and each element of that matrix is a function of the dot products in d-dimensions.
- Each of the elements of the new matrix K', after the transformation ϕ is taken, is simply the evaluation of the dot product in the original dimension, i.e. no explicit knowledge of ϕ is necessary.

REFERENCES

[1] Christopher M. Bishop *Pattern Recognition and Machine Learning*. 2006: Springer.

[2] Costello *et al. A community effort to assess and improve drug sensitivity prediction algorithms.* Nat Biotechnol. 2014 December; 32(12): 1202-1212.

[3] Mehmet Gönen and Ethem Alpaydin. *Multiple Kernel Learning Algorithms* 2011: Journal of Machine Learning Research 12.

[4] Polly Mitchell-Guthrie *Looking backwards, looking forwards: SAS, data mining, and machine learning* 2014: taken from:
http://blogs.sas.com/content/subconsciousmusings/2014/08/22/looking-backwards-looking-forwards-sas-data-mining-and-machine-learning/.

[5] *Support Vector Machine Regression* Taken from: http://kernelsvm.tripod.com/