Machine Learning and Data Dependencies: an Impossible Marriage ?

Marie Le Guilly Jean-Marc Petit Marian Scuturici

INSA Lyon, Université de Lyon
LIRIS CNRS (UMR 5205)

May, 9-10th, 2019 – ISIP 2019
Heraklion, Greece

Remake of “Le mariage de la carpe et du lapin” ?

Literally : The marriage of carp and rabbit

(or : A square peg in a round hole)

French expression used to illustrate a union between two different things and by extension, an impossible alliance by nature

Informal talk, ongoing work

Source :
`http://www.expressions-francaises.fr/expressions-l/864-le-mariage-de-la-carpe-et-du-lapin.html`

Machine Learning and Data Dependencies

In the sequel, to keep the presentation simple :

- Machine learning = supervised learning
- Data Dependencies = Functional Dependencies

What would be their lowest common denominator ?

Underlying background

Back to school : function definition (1/2)

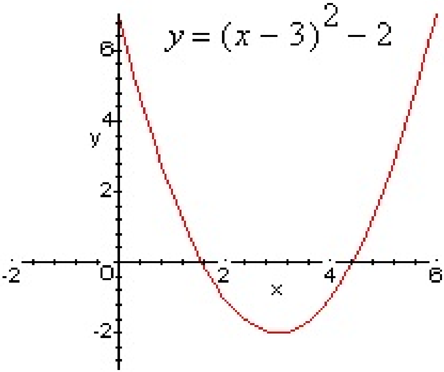
In mathematics, a function was originally the idealization of how a varying quantity depends on another quantity.
For example, the position of a planet is a function of time.

A function is a process or a relation that associates each element x of a set X, the domain of the function, to a single element y of another set Y (possibly the same set), the codomain of the function.

Source :
`https://en.wikipedia.org/wiki/Function_(mathematics)`

Function definition (2/2)

A function is uniquely represented by its graph which is the set of all pairs $(x, f(x))$. When the domain and the codomain are sets of numbers, each such pair may be considered as the Cartesian coordinates of a point in the plane.



Supervised classification and functional dependencies

Let's start the premises of the wedding :-)

Supervised Classification

Learning algorithm definition

Given a set of N training examples of the form $\{(x_1, y_1), ..., (x_N, \ y_N)\}$ such that x_i is the feature vector of the i-th example and y_i is its label (i.e., class)

A **learning algorithm** seeks a function $g : X \rightarrow Y$, where X is the input space and Y is the output space.

The function g is an element of some space of possible functions G, usually called the *hypothesis space*.

It is sometimes convenient to represent g using a scoring function $f : X \times Y \rightarrow \mathbb{R}$ such that g is defined as returning the y value that gives the highest score :

$$g(x) = \arg\max_{y} f(x, y)$$

Source :
https://en.wikipedia.org/wiki/Supervised_learning

Learning algorithm with DB notation

Let $R_0 = \{\overbrace{A_1, \ldots, A_n}^{X}, \underbrace{C}_{Y}\}$ be a relation schema.

Let r_0 be a relation over R_0, i.e. a set of examples (tuples)

A **learning algorithm** seeks a function $g : dom(A_1), \ldots, dom(A_n) \rightarrow dom(C)$, where $dom(A_1) \times \ldots \times dom(A_n)$ is the input space and $dom(C)$ is the output space.

- It learns a function from the examples of the *active domain*,
- The function is expected to generalize well to other (unknown) examples (from the domain)

Such a function could be a polynom, an exponential, an integral, . . . or just a black box (e.g. neural networks, support vector machine) !

Functional dependencies

Functional dependencies (1/2)

Let R be a relation schema and $X, Y \subseteq R$.
A *functional dependency* is an expression of the form $X \rightarrow Y$, satisfied in every possible relation r over R.

$$r \models X \rightarrow Y \text{ iff for all } t_1, t_2 \in r$$

$$\textbf{If} \text{ for all } A \in X, t_1[A] = t_2[A] \textbf{ then} \text{ for all } B \in Y, t_1[B] = t_2[B]$$

Turns out to be a very general notion, related to implications and functions

Functional dependencies (2/2)

FD as implications

a	b	$a \to b$
0	0	1
0	1	1
1	0	0
1	1	1

Many connections with lattice theory, formal concept analysis (Galois connection) and logics (see for ex [**?**])

FD as functions

- $r \models A_1, \ldots, A_n \to C$ **iff** there exists a function from $adom(r, A_1) \times \ldots \times adom(r, A_n)$ to $adom(r, C)$
- $A_1, \ldots, A_n$ is a **key** in $\pi_{A_1,\ldots,A_n,C}(r)$

Main differences between supervised classification and functional dependencies (1/2)

- Data dependencies do not care about the data values themselves : they only care about their **comparisons**
  - `if t1.age = t2.age then ...`
  - `if abs(t1.age - t2.age)`$\leq$ `2 then ...`
- Learning algorithms care about the data values to draw their conclusions
  - `if age`≤ 18 `then ...`

Looks like a bad news

Main differences (2/2)

- A classification model *defines* a function learned from a set of examples
- The satisfaction of a functional dependency in a relation *defines* the existence of a function

For a new (and unseen) feature vector, a classification model predicts a single C-value, while the satisfaction of a FD does not predict anything !

Looks like another bad news !

Synthesis

$\boxed{\text{Existence of a function on one side, identification of a function on the other}}$

One is clearly more difficult than the other !

What does the existence of function mean in a learning context ?
What can we draw ?

A typical data science scenario

Let us consider a simplified supervised classification scenario ;

- Data preprocessing : Experts spend a lot of time to gather their data, to integrate them, to do feature engineering ... At the end, they have a dataset (training/test or k-fold)
- Learning algorithms : Then they apply many of them to build classification models. They pick up the best one wrt robustness.

It might be possible to learn a function in a training dataset ... in which a function does not exist !

A typical data science scenario

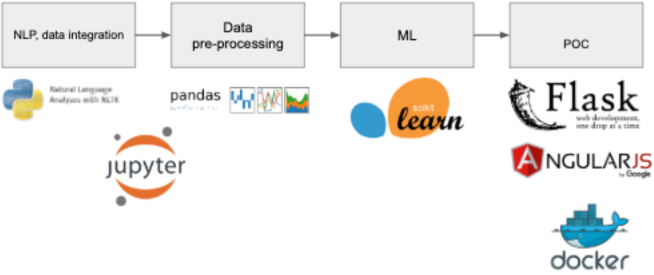
Let us consider a simplified supervised classification scenario ;

- Data preprocessing : Experts spend a lot of time to gather their data, to integrate them, to do feature engineering ... At the end, they have a dataset (training/test or k-fold)
- Learning algorithms : Then they apply many of them to build classification models. They pick up the best one wrt robustness.

It might be possible to learn a function in a training dataset ... in which a function does not exist !

Current technologies for data science

Technological stacks for ML – from integrated platforms such as `Azure`, to more technical stacks – bring to every data scientist the **ability to run this (bad) scenario** . . .



Interest of FD for supervised classification

At some point in a supervised classification scenario, it makes sense to take care about the existence of a function, before trying to identify one of them !

- The data makes it possible to know whether a function exists or not, whatever the form of the function : polynomial, triple integrals, ...

Propositions :

- Data cleansing could be guided by the existence of functions, through the notion of counter-examples
  $\Rightarrow$ very powerful mechanism for interacting with domain experts

Conclusion

The marriage is going to complicated, but still not impossible !

- Seems to be "common sense" to check the existence of a function in data before trying to learn a function from data !
- Not sure at all that data scientists worldwide are aware of this !
- Despite their inherent differences, FDs may help supervised learning

Intimately related to how data is **prepared** for learning, i.e. the data preprocessing step.

Conclusion

The marriage is going to complicated, but still not impossible !

- Seems to be "common sense" to check the existence of a function in data before trying to learn a function from data !
- Not sure at all that data scientists worldwide are aware of this !
- Despite their inherent differences, FDs may help supervised learning

Intimately related to how data is **prepared** for learning, i.e. the data preprocessing step.

Conclusion

The marriage is going to complicated, but still not impossible !

- Seems to be "common sense" to check the existence of a function in data before trying to learn a function from data !
- Not sure at all that data scientists worldwide are aware of this !
- Despite their inherent differences, FDs may help supervised learning

Intimately related to how data is **prepared** for learning, i.e. the data preprocessing step.

Ongoing work

- How to measure the feasibility of ML for a given dataset ?
- How to optimally group together similar raw values such that the existence of a function is guaranteed ?
- Data visualization opportunities to identify counter-examples of a given function (or FD).

Thank you

Merci

σας ευχαριστώ