High Level Pipeline for Application Level Project development in AI/ML

"If you want a guarantee, buy a toaster" - Clint Eastwood

(Konstantin Burlachenko, burlachenkok@gmail.com, 31MAY2020)

Introduction

Happy 90th Birthday to Clint Eastwood. He was born May 31, 1930 in San Francisco, California. As you will see his quote has deep relation to AI/ML.

The real world is very complicated. Whatever concepts we define really does not matter. At the end of the day, we need:

- To write some code
- Build some hardware
- Exploit available data
- Append priors which is not expressed as data

People in ML state that they train models only in from data. Even it sounds neat, but it's only partially true. Once you have fixed function class from which you're pulling function, you already reduce your function class. This note is dedicated to describe that for use AI/ML methods in the production there is in fact a whole pipeline.

In my opinion, this a true and honest pipeline. If you will skip some part of it, you can be in big-big trouble. Especially, if you're working in startup projects or in company where such infrastructure has not been set yet.

But company want use AI/ML type things because it's very fashion or they have real problems and they decide to solve them in some nice way with using modern technologies from AI umbrella.

Catalog of Approaches from AI

Even it seems catalog of problems can vary and you can introduce your own problems too with predicting some new things (e.g., the graph) but classical catalog of tasks in AI include the following:

- **Unsupervised learning is useful** when there are a lot of data and answer for questions of clustering and projecting your data in some other space.
- **State-Based models** are very useful for the search for answers even in huge state space without noise.
- **Reinforcement Learning** can be used to understand how control, when there are no first principles and there is actually a noise (due to lack of modeling or whatever) which brings to some state in which you actually do not want to move, but you have moved.
- **Game Models** are useful when there an adversarial part with opponents . And your dynamic model is not alone in the world. Now there is not only a nature and some lack of modeling, but also an opponent, who probably want to kill you.
- **Logic-Based models, Bayesian Networks -** is useful to have deep reasoning, but they also require constructing let's say framework for this deep reasoning.
- **Supervised models** are very popular so it has a form that real dependency is X,Y->Z and we approximate it with X->Z with some method which is very popular.

Modern vision is that some aspects of the model should be tuned from data, not only from First-Principles so even you have first principles maybe you still need data.

Pipeline how people very often(but not always) do ML/AI type things

Now let's go step-by-step in what to do.

1. Select a Problem

The problem should be sound feasible. You cannot predict exchange currency by temperature.

If your boss asked it and press on you solve impossible task – *probably it's time to change your boss*.

Second, you should understand that AI is ok to your problem. Even "AI solves everything", but typically provide very small guarantees into quality of a solution. Except very special cases what is constructed are powerful approximations.

If boss asked: Let's check you prediction, if it will not correct classify my single image – I will kill you.

In this case it's better change your boss, or change the problem which allow to be sloppy.

Except very special cases you can construct good approximation, but due to deep problems – you can not guarantee in strong sense for a lot of things. The problem is that AI/ML connects apply math with real-life, there are too much things which will crack your system.

2. Get Data

First-order message is that "*data is needed*" for data driven approach.

Please not give up with small data, but small data is a challenge.

So if you have no data, but need a complete project to be honest – *you're in trouble*.

3. Design Model

If you want to hire somebody to design models then it can be or people who have domain knowledge.

Hire experts or person who is interested in the domain. It's better to hire experts, but if no way it's

possible to hire non-experts, but at least people who can dig into the field of study, it's jargon, etc..

Not all of the richness of the real world can be captured, and therefore there is an art of modeling.

Experts will do it better. What we are doing are very often various approximations during modeling.

Models should be robust to noise (if noise is possible).

4. Train or Learning stage of the Model

Find parameters (ML word) or variables (Math Optimization world) that describes the model. In fact, there are non-parametric models too. In that case you do not need to find parameters of the model by itself.

To make solution robust and have small variance there is a principle with empirical risk minimization on validation set.

There are also some variations of how samples come into the system:

Online Learning – when train samples come as a stream to your system.

Offline Learning – situation when you train on trained sett. In this case it's fine even if algorithm jams - you just restart it. In particular it implies – it's completely fine even math optimization algorithms jams.

Active Learning – when train samples come in such way that you should make prediction and maybe you obtain feedback. And in any case you will know is your prediction correct in *runtime*.

>

5. Test Model
Here there are three ways. Test models on test set or hire experts who will assess the model. Option 3 is used assessing method from your field. Maybe in your field there is a simulator or something like this.

6. Deploy Model
After model is deployed, it's possible perform **inference** is to answer questions with respect to the model

7.Maintain Model.
It's needed due to some limitation of ML with which researchers try to live, but everything deep is very hard.

References and credits
Theoretical knowledge I have been obtained from several courses:

[CS221] AI Designs and Principles with P.Liang, Stanford University

[CS229] Machine Learning with A.Ng, Stanford University

[CS230] Deep Learning with A.Ng, Stanford University.

In which I was a student and my employer NVIDIA paid for it due to NVIDIA SCPD program.

https://online.stanford.edu/company/nvidia

Practical knowledge I have obtained from several companies where I worked on this type of thing.