CAPS Case - Lecture 3
Algorithm Selection Under Uncertainty

Leo Klenner, Henry Fung, Cory Combs

Last updated: 11/9/2019

Goal

This is the case we will discuss in the third lecture. The goal of this case is to familiarize you with selecting the right algorithm for a given task based on end-user feedback and consideration of trade-offs. In the previous case, we've learned about selecting the right data. The next step in an applied machine learning workflow is selecting the right algorithm that can compute over this data, which is what this case is about.

The case has a background section that introduces, at a high level, the core trade-offs that determine algorithm selection. Once you have familiarized yourself with these trade-offs, your task is to make recommendation about algorithm selection to an end-user based on fuzzy feedback from the user.

Don't worry if not all of the concepts introduced in the background section are immediately clear as we will dive into them in the lecture. For the case, focus on your intuition and be prepared to explain your thought process. Thought process > right results.

If you have questions, please contact us at capsseminar@gmail.com

Background

The trade-offs are based on five characteristics that define and differentiate algorithms: **bias**, **variance**, **interpretability**, **time**, and **complexity**. We'll review these characteristics in more detail in the lecture but for the case, here's a brief definition of the properties:

Characteristic	Definition
Bias	How closely the model follows the training data and delivers accurate results for the test data?
Variance	How easily do the algorithm's results change based on changes in the training data?
Interpretability	How human understandable is the process by which the algorithm arrived at the results?
Complexity	How computationally simple or complex is the algorithm?
Time	How long does it take for the algorithm to arrive at results?

These characteristics are related to each other in ways that gives rise to trade-offs that we need to consider when selecting an algorithm. For the case, we need to have a broad understanding of the relationship between these characteristics. So, let's make some assumptions:

- **High bias comes with low variance** - The results of the model do not capture the underlying pattern of the training or test data. The model does not follow the training data at all and hence the results don't change when the training data changes. We refer to this as *underfitting*.
- **Low bias comes with high variance** - The results of the model capture the noise along with underlying pattern in the data. The model follows the training data too closely and hence the results change heavily when the training data changes. We refer to this as *overfitting*.
- **High bias comes with high interpretability** - The pattern found by closely following the training data is oversimplified and therefore human understandable.
- **Low bias comes with low interpretability** - The pattern found by following the training data less closely is more general but less human understandable.
- **High interpretability comes with low complexity** - If the model is easy to interpret, it isn't computationally complex.
- **Low interpretability comes with high complexity** - If the model is difficult to interpret, it is computationally complex.
- **High complexity comes with high time** - If the model is computationally complex, it takes a relatively long amount of time to return results.
- **Low complexity comes with low time** - If the model is computationally simple, it takes a relatively short amount of time to return results.

Note, that these relationships are simplified assumptions but suffice for the case. We'll give a more detailed review of these trade-offs in the lecture.

Tasks

Your organization has shipped a supervised learning model to a client operating in a remote area. After sending the code, you receive an email from your client outlining a problem with the model. Your tasks is to determine how an algorithm with different trade-offs might be selected to resolve the problem. You only need to specify the trade-offs, don't worry about thinking what the name of the algorithm might be.

Note, that the client has limited technical understanding. You should reflect on whether the client frames their problem correctly in the email, before you proceed to finding a solution.

For each problem below, answer the following questions:

1. What question would you ask or want to know the answer to before you proceed?
2. What characteristic or trade-off is at the core of the problem?
3. Is the problem in the email framed correctly?
4. What trade-offs might arise for the client if the model is updated as requested?
5. Can the new trade-offs potentially outweigh the old ones?
6. What's your recommendation for how you and the client should proceed?

We provide you with three different versions of the email, so there are three different problems. The first part of the email is the same for each of them.

Email head

> Thanks for sending us the code of the trained model, which we have deployed here. Unfortunately, it has been reported to me that the model does not work for our purposes.
>
> ...

Problem 1

> The inference we're trying to compute between insurgent attack patterns and organizational structure does not seem to be supported by the model as it is currently specified. We've been communicating with operators on the ground but they're unwilling to send out personnel based on the model's results. Given that these are domain experts who understand the environment we're engaging, could you send over an updated model fast?

Problem 2

> We've explained the model to the personnel on the ground and while we have now established that the technical aspects are clear, the results returned by the model seem disappointing. As we're trying to predict changes in the organizational structure of the insurgents here, we're operating in a time-sensitive environment. So we've had problems with getting the right information to our people in the right time and therefore we have not been able to make use of the models results at all. When can you send us a new model that fixes this problem?

Problem 3

> There was some issue with the code, it seems that you in fact shipped an untrained model. So, we retrained the model based on the data you provided. However, we don't see the same results as referenced in your documentation of the model under "Examples". From what I can see, it's the same attack dataset that you sent us and we followed the steps outlined in the documentation, split the data, then train, etc. Look, this is obviously a problem as we'll be collecting more data here and will retrain the model locally in the future. How can you fix this?