Designing better algorithms: 5 case studies

- Posted by Vincent Granville on October 31, 2016 at 8:30pm
- Send Message View Blog

In this article, using a few examples and solutions, I show that the "best" algorithm is many times not what data scientists or management think it is. As a result, too many times, misfit algorithms are implemented. Not that they are bad or simplistic. To the contrary, they are usually too complicated, but the biggest drawback is that they do not address the key problems. Sometimes they lack robustness, sometimes they are not properly maintained (for instance they rely on outdated lookup tables), sometimes they are unstable (they rely on a multi-million rule system), sometimes the data is not properly filtered or inaccurate, and sometimes they are based on poor metrics that are easy to manipulate by a third party seeking some advantage (for instance, click counts are easy to fake.) The solution usually consists in choosing a different approach and a very different, simple algorithm - or no algorithm at all in some cases.

Five Case Studies

Here I provide a few examples, as well as an easy, low-cost, robust fix in each case.

Many times, the problem is caused by data scientists lacking business understanding (they use generic techniques, and lack domain expertise) combined with management lacking basic understanding of analytical, automated, optimized (semi-intelligent, self-learning) decision systems based on data processing. The solution consists of educating both groups, or using hybrid data scientists (I sometimes call them business scientists) who might not design the most sophisticated algorithms, but instead the most efficient ones given the problems at stake - even if sometimes it means creating ad-hoc solutions. This may result in simpler, less costly, more robust, more adaptive, easier to maintain, and generally speaking, better suited solutions.

- **Click fraud detection**: This is an old problem, yet publishers using affiliates to generate traffic (including Google and its network of partners) still deliver vast amounts of fraudulent clicks. While these companies have become much smarter about pricing (very lowly) these worthless clicks, better and easier solutions exist. For instance, pricing per keyword per day rather than per click, or targeting specific people/audiences to make it difficult to create fake traffic (and at the same time increasing relevancy and thus ROI both for the advertiser and the ad network.) For instance, both Facebook and Twitter allow you to target friends of friends, or profiles similar to pre-specified people. The issue with the pay-per-click algorithms (specifically, fraud detection) is not so much the fact that the algorithms miss a lot of fraud, but rather caused by the business model which is flawed by design. My solution consists of changing the business model.

- **Ad matching or relevancy algorithms**: We've all seen too many times ads that are irrelevant to us - it is a waste of money for the advertiser and the ad network. Ad matching algorithms (or generally speaking, relevancy algorithms) aim at optimally serving the right ads (or content) to the right user on the right page at the right time. When several ads compete for a spot and can all be displayed, they need to be displayed in the right order (on a specific page, to a specific user.) Such algorithms can greatly be improved by assigning categories both to pages, users and ads, in order to optimize the match, even in the absence of a search query. This is described in this article, and modern techniques rely on tagging or indexation algorithms. Such indexation algorithms are conceptually very simple and robust, and can quickly create taxonomies on big data, to help solve the problem. Another search engine algorithm that can benefit from substantial improvements is content attribution: assign the content to the original source (by displaying it at the top in search results), rather than to an authorized copy or worse, to a plagiarist. Click here for details; the solution might be as easy as pre-sorting index entries (for a same keyword and identical content) by time stamps. More on search engine technology here.

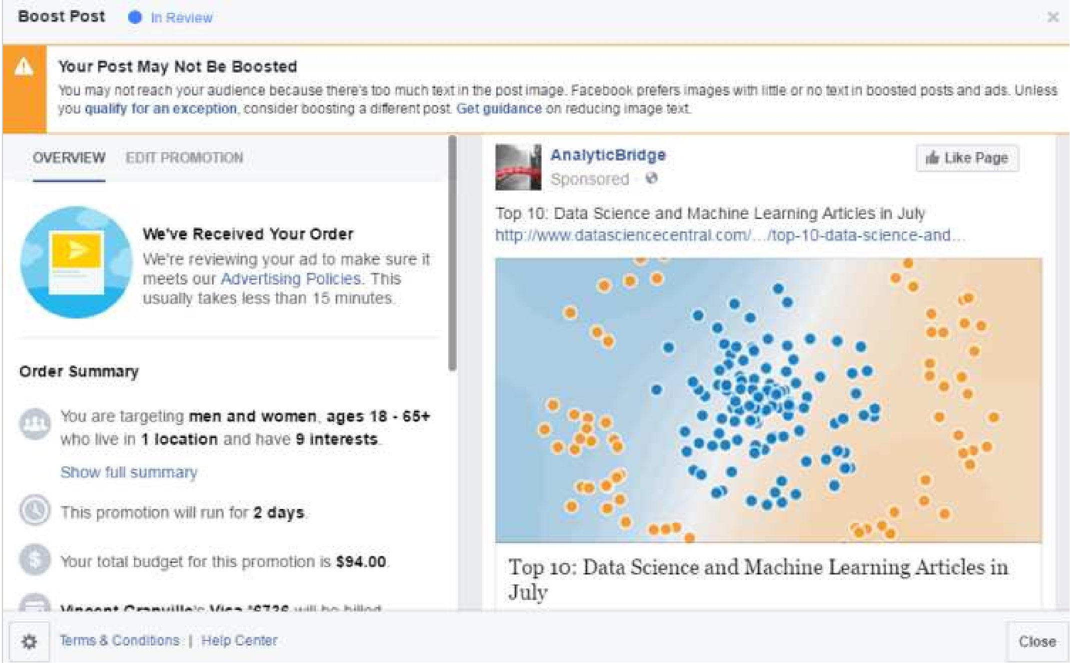
- **Optimum pricing**: Just like using the same drug for all patients (to cure a specific ailment) is a poor strategy, using the same price (for a specific product) for all customers may not be the best solution. Optimum pricing varies based on time, sales channel, and customer. I described this concept in an article on hotel rooms pricing.

- **Fake reviews detection**: Product and book reviews are notoriously biased as authors get reviews from friends, and blackmailers try to get your money to post good reviews about your product, or otherwise will write bad reviews. Read this article for details. The bulk production of fake reviews is indeed a striving business in its own (if executed properly with the right algorithms as described here.) It negatively impacts all websites (such as Amazon) that rely on product reviews to increase sales and attract users: it is a trust issue. The concept itself is subject to conflicts of interests - good reviews supposedly increasing sales, are thus encouraged or given more weight. So here we are facing a business flaw rather than poor detection algorithms. The solution is having professionals write the reviews and then the problem of fake reviews almost disappears - no need for an algorithm to handle it. If however you really want to implement user-based product reviews on your e-store, here is the way to do it right: as in the relevancy algorithm described above, assign categories to each user, each product and each reviewer. When there is a strong match (the user, the reviewer and the product categories all match) assign a high score to the product review in question. Eliminate reviews that are too short. First-time reviewers might be assigned a lower score. Then compute a weight for each star rating assigned to a product, by summing up all the individual scores for the star rating in question, possibly putting more emphasis on recent ratings. The global rating is the weighted sum of star ratings, for the product in question. This is far better than a flat average of the star ratings regardless of the quality of the review or reviewer, which is what Amazon is still doing.

- **Image recognition (Facebook ads)**: This is indeed a funny algorithm. As a Facebook advertiser promoting data science articles, most images in my ads are charts and do not contain text. For whatever business reason (probably an archaic rule invented long ago and never revisited) Facebook does not like postings (ads in particular) in which the image contains text. Such ads get penalized: they are displayed less frequently, and cost more per click; sometimes they are just rejected. Most of my ads are erroneously flagged as containing text, see Fig 1 for a typical example. Note that the Facebook algorithm (to detect text in images) processes a large number of ads in near real time, thus it must be rudimentary enough to make decisions very fast -- although it is very easy to use a distributed, Map-Reduce architecture to process these ads. The solution to this issue: get rid of your algorithm entirely, instead use a much better relevancy metric (rather than whether or not the image contains text): click-through rate. The computation is straightforward, though you might need an algorithm to detect and filter out fraudulent or robotic clicks. More on the text detection algorithm in the section below, where a simple, efficient solution is offered to advertisers facing this problem. Of course you could tell me to put arbitrary, irrelevant pictures of people, mountains, vegetables, or lakes in all my ads, to pass muster, but that is not the point -- it might backfire and data scientists are genuinely interested in ... charts. Read the next section for more details. Another faulty algorithm that I will analyze in a future article is the one used to detect posts (on Twitter or Facebook) that violate editorial policies against hate speech, bullying or raunchy language. This algorithm is so bad that it caused Walt Disney to pass on buying Twitter. I wouldn't be surprised if it relies on the Naive Bayes technique - still currently in use in many (poor) spam detection algorithms.

More about the Facebook ad processing system

The first four cases have been discussed in various articles highlighted above, so here I focus on the last example: the image recognition algorithm used by Facebook to detect whether an image contains text or not, to assess ad relevancy -- and best illustrated in figure 1. This algorithm eventually controls to a large extent, the cost and relevancy associated with a specific ad. I will also briefly discuss a related algorithm used by Facebook, that also needs significant improvement. I offer solutions both for Facebook (to nicely boost its revenue yet boost ROI for advertisers at the same time), as well as solutions for advertisers facing this problem, assuming Facebook sticks with its faulty algorithms.



Boost Post ● In Review

Your Post May Not Be Boosted

You may not reach your audience because there's too much text in the post image. Facebook prefers images with little or no text in boosted posts and ads. Unless you qualify for an exception, consider boosting a different post. Get guidance on reducing image text.

OVERVIEW EDIT PROMOTION

We've Received Your Order

We're reviewing your ad to make sure it meets our Advertising Policies. This usually takes less than 15 minutes.

Order Summary

You are targeting **men and women**, **ages 18 - 65+** who live in **1 location** and have **9 interests**.

Show full summary

This promotion will run for **2 days**.

Your total budget for this promotion is **$94.00**.

Vincent Granville's Visa *6736 will be billed

AnalyticBridge

Sponsored

Like Page

Top 10: Data Science and Machine Learning Articles in July

http://www.datasciencecentral.com/.../top-10-data-science-and...

Top 10: Data Science and Machine Learning Articles in July

Terms & Conditions | Help Center

Close

Figure 1: Facebook image recognition algorithm thinks the above image contains text!

Solution for advertisers

For each article that you want to promote on Facebook, starts with a small budget, maybe as small as $10 spread over 7 days, and target a specific audience. Do it for dozens of articles each day, adding new articles all the time. Regularly check articles that exhausted their ad spend; boost (that is, add more dollars of ad spend) to those that perform well. Performance is measured as the number of clicks per dollar of ad spend. All this can probably be automated using an API.

Solution for Facebook

Eliminate the algorithm that is supposed to detect text in images associated with ads. Instead focus on click-though rate (CTR) like other advertising platforms (Google, Twitter.) Correctly measure impressions and clicks to eliminate non-human traffic, to compute an accurate CTR.

Predicting reach based on ad spend

Facebook provides statistics to help you predict the reach for a specific budget (ad spend) and audience, but again, the algorithm doing this forecast is faulty, especially when you try to "add budget" prior to submitting your ad. Google also provides forecasts that in my experience, are significantly off. I believe the problem is that for small buckets of traffic, the strength of this forecast is very weak. While confidence intervals are provided, they are essentially meaningless. The solution to this problem is to either provide the strength of the forecast (I call it predictive power), or *not* provide a forecast at all: the advertiser can use the solution offered in the previous paragraph to optimize her ad spend. And if Facebook or Google really want to provide confidence intervals for their forecasts, they should consider this model-free technique that does not rely on the normal distribution: it is especially fit for small buckets of data that have arbitrary, chaotic behavior.

Top DSC Resources

- Article: What is Data Science? 24 Fundamental Articles Answering This Question
- Article: Hitchhiker's Guide to Data Science, Machine Learning, R, Python
- Tutorial: Data Science Cheat Sheet
- Tutorial: How to Become a Data Scientist - On Your Own
- Categories: Data Science - Machine Learning - AI - IoT - Deep Learning
- Tools: Hadoop - DataViZ - Python - R - SQL - Excel
- Techniques: Clustering - Regression - SVM - Neural Nets - Ensembles - Decision Trees
- Links: Cheat Sheets - Books - Events - Webinars - Tutorials - Training - News - Jobs
- Links: Announcements - Salary Surveys - Data Sets - Certification - RSS Feeds - About Us
- Newsletter: Sign-up - Past Editions - Members-Only Section - Content Search - For Bloggers
- DSC on: Ning - Twitter - LinkedIn - Facebook - GooglePlus

Follow us on Twitter: @DataScienceCtrl | @AnalyticBridge

Views: 7940

Like
2 members like this

Share Tweet G+ Facebook

Like 0

- < Previous Post
- Next Post >