Chapter 12. Machine Learning Engineering and MLOps Case Studies

By Noah Gift

> *After accompanying Professor Loewi through his procedure I spent more time in his postop care during which he lectured me further. He signed my copy of his little 62-page book, above his signature in a shaky hand he wrote, "Facts without Theory is chaos, Theory without facts is phantasy.*
>
> —Dr. Joseph Bogen

One of the fundamental problems with technology in the real world is that it is tough to know who to listen to for advice. In particular, a multidisciplinary topic like machine learning is a puzzling challenge. How can you find the right mix of real-world experience, current and relevant skills, and the teaching ability to explain it? This "unicorn" teaching ability is what this chapter aims to do. The goal is to distill these relevant aspects into actionable wisdom for your machine learning projects, as shown in Figure 12-1.

Other domains suffer from the curse of the unbounded complexity that comes with a multidisciplinary field. Examples include Nutritional Science, Climate Science, and Mixed Martial Arts. However, a common thread is the concept of an open system versus a closed system. One toy example of a primarily closed system is an insulated cup. In that example, it is easier to model the behavior of a cold liquid since the environment has minimal effect. But if that same cold liquid goes outside in a regular cup, things get murky quickly. The outside air temperature, humidity, wind, and sun exposure alone create cascading complexity in modeling the behavior of that cold liquid.

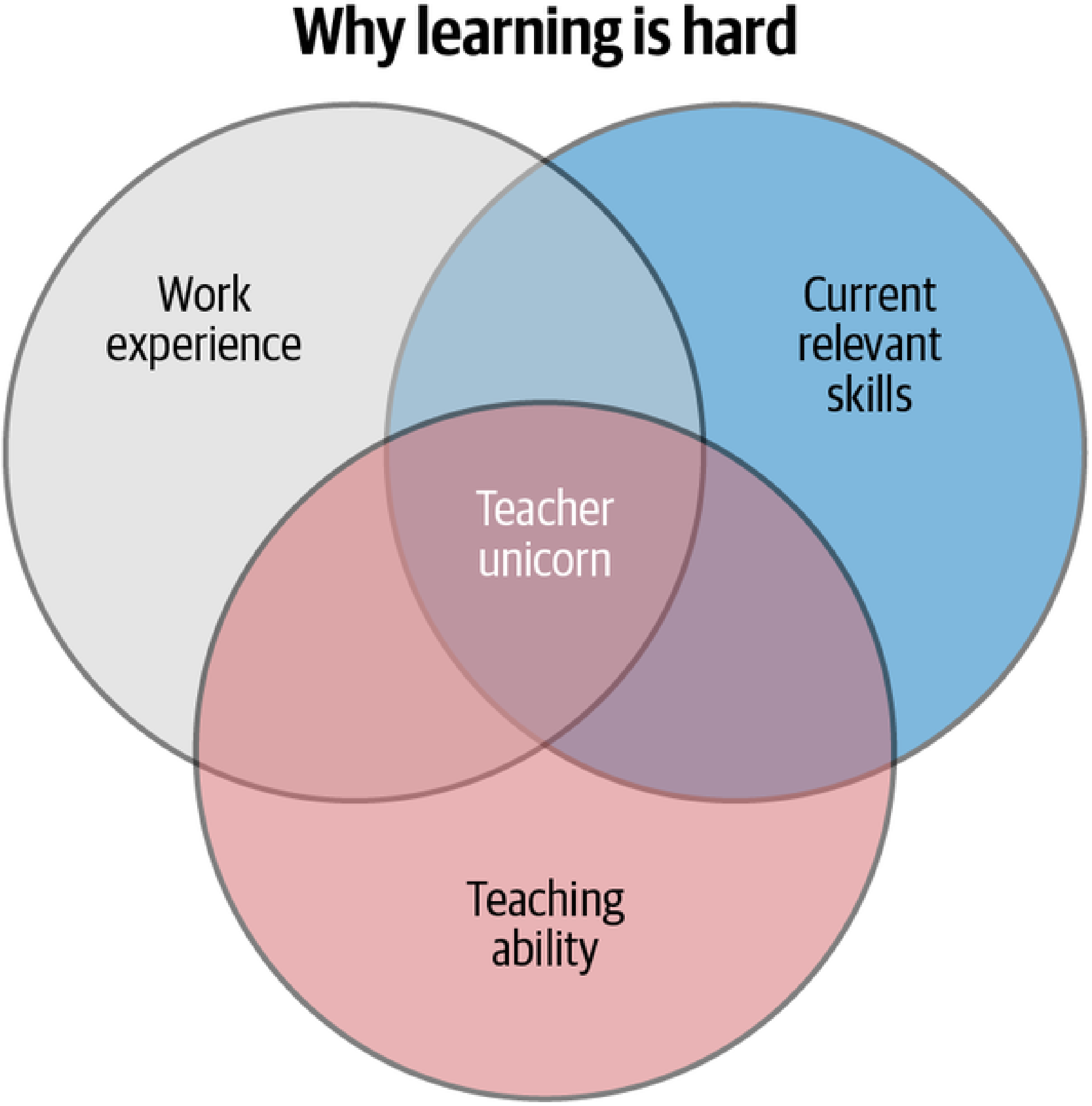


Figure 12-1. Teaching unicorn

This chapter explores how MLOps can leverage learnings from these other domains. It also explores how closed versus open systems influence domain-specific behavior and, ultimately, how to apply this to operationalizing machine learning.

Unlikely Benefits of Ignorance in Building Machine Learning Models

There are many unlikely benefits to ignorance. Ignorance gives you the courage to try something challenging, where if you knew how difficult it was, you would never have done it. Since 2013, ignorance has played a crucial role in two things I have done simultaneously: create a production machine learning model with millions of dollars at stake, including a 100 person company; and learn, train, and compete in Brazilian Jiu-Jitsu with top-ranked professional fighters and Olympians in wrestling and Judo. In some sense, the two things are so intertwined it is hard to separate one from the other in my mind. From 2010–2013 I spent three years taking every statistics, probability, and modeling class I could take at UC Davis in its MBA program while working full-time at startups in San Francisco. Since 2017, I have also taught machine learning and cloud computing at the UC Davis Graduate School of Management. After I graduated, I was ready to take a General Management or Chief Technology Officer position and became one of the first technical employees at a Sports social network as the CTO and General Manager.

Part of the company's culture was that employees would exercise together at a mixed martial arts gym that also did general fitness classes. I accidentally started training in Brazilian Jiu-Jitsu as I got curious watching the pro fighters grapple. Eventually, before I knew it, I worked alongside professional fighters and learned the basics of submissions. A few times, I even accidentally became choked unconscious in sparring. I remember one time thinking, "I probably should tap to this head and arm choke." Later, I wondered where I was; it looked like a gym, and I didn't know what year it was. Was I in my high school gym in Southern California? How old am I? As blood came back into my brain, I realized that, ah, I got choked out, and I am in the martial arts gym in Santa Rosa, California.

In my first few years of training, I also entered two tournaments, winning the first one, a "novice" category, and then losing an "intermediate" class a couple of years later. In hindsight, I honestly didn't understand what I was doing and faced the real risk of serious injury. I observed many people in the tournaments get severe damage, including head butts, broken shoulders, and ripped knee ligaments. In my second tournament as a 40-year old, I competed against a 20-year-old college football player in the 220-pound weight class. In his last match, he got in a real fight, got head-butted, was bleeding from the nose, and was quite angry. I was apprehensive about his emotional state going into our contest, thinking, "What the hell did I sign up for?"

I also counted my blessings that I was ignorant of the actual danger of competing in martial arts in my thirties and forties. I still enjoy training and learning Brazilian Jiu-Jitsu, but it is likely that if I knew what I knew today, I wouldn't have taken the risks I took with as little skill as I had. Ignorance gave me the courage to, frankly, take dumb risks but also learn more quickly.

Similarly, in 2014, I was just as ignorant as I started the journey of building out a machine learning infrastructure for my company. Today this is known as MLOps. At the time, there wasn't a lot of information about how to operationalize machine learning. As with Brazilian Jiu-Jitsu, I was eager yet ignorant of what really would unfold and the risks at play. Later the terror I felt in Brazilian Jiu-Jitsu was nothing compared to the terror I would experience in building prediction models by myself from scratch with the responsibility for millions of dollars a year.

In the first year, 2013, our company built a sports social network and mobile application. But, as many employees at a startup know, building software is only part of a startup's challenge; two other vital challenges are acquiring users and creating revenue. At the start of 2014, we had a platform, yet no users and no income. So we needed to come up with users quickly, or the startup would go bankrupt.

Building traffic generally occurs in two ways for a software platform. One way is to build organic growth through word of mouth. A second way is to purchase advertising. The problem with buying advertising is that it can quickly become a permanent cost allocation. The dream scenario was for our company to get users to come to our platform without buying ads. We had a relationship with a few sports stars, including former NFL quarterback Brett Favre. This initial "superstar" social media influencer gave us tremendous insight into using organic "growth hacking" to grow our platform.

At one point, this machine learning feedback loop got us into the millions of monthly active users. Then, the Facebook legal team sent us a communication that they would "deplatform us" metaphorically. Our "crime" was creating unique, original sports content that had links to our platform. There is an ongoing question about the potential monopolistic power of Big Tech, and the growth hacking power of our influence algorithm got the attention of Facebook. This was another data point about the real-world success of our prediction system. Let's dive into how we did this next.

MLOps Projects at Sqor Sports Social Network

Building a startup from zero—i.e., zero staff, zero uses, and zero revenue—is a stressful pursuit. From 2013–2016 I spent many hours in the park underneath the Transamerica building in San Francisco, directly under my office, plotting these things. In particular, the epistemic risk, the risk I was unaware of, of betting millions of dollars on machine learning predictions was terrifying. But, in many ways, the technical challenges are much more comfortable than the psychological ones.

Here is an overview of how the system worked. In a nutshell, the user posts original content and then we cross-post the content to other social networks like Twitter and Facebook. We then collected the pageviews generated for our site. The pageviews became the target in our ML prediction system. Figure 12-2 shows the MLOps pipeline for our social media company.

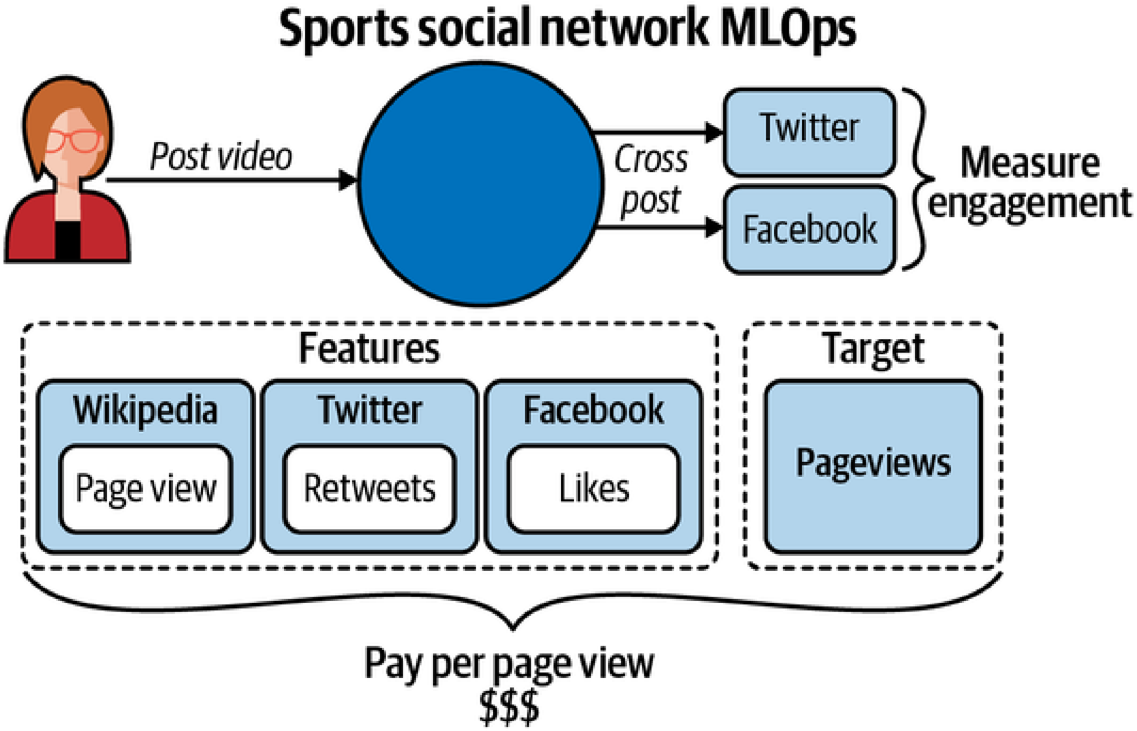


Figure 12-2. MLOps pipeline for sports social network

Later, we collected the social media signals from these users, i.e., their median retweets, median likes, and Wikipedia page views. These became the features and helped us cut through the noise of false data like "fake" followers. We then paid original content creators like Conor McGregor, Brett Favre, Tim McGraw, and Ashlyn Harris by the engagement they generated on our site. It turns out we were way ahead of the game in doing this, and Snapchat pays out millions for users to publish original content on its platform.

The most challenging parts of the problem were reliably collecting the data and then deciding to pay millions of dollars based on the predictions. Both were much more complicated than I initially thought. So let's dive into these systems next.

Mechanical Turk Data Labeling

Initially figuring out that social media signals gave us enough predictive power to "growth hack" our platform was a huge breakthrough. But, unfortunately, the most formidable challenge was yet to come.

We needed to collect the social media handles reliably for thousands of "celebrity" social media users. Unfortunately, it didn't go well initially and ended in complete failure. Our first process looked like Figure 12-3.

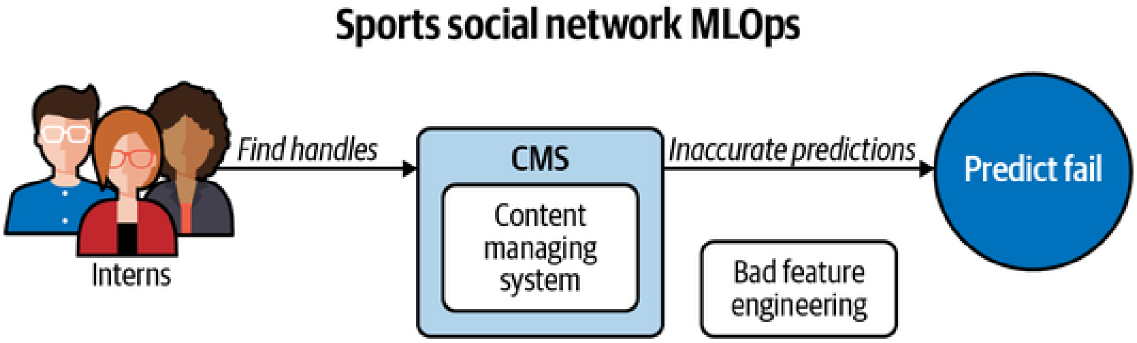


Figure 12-3. Bad feature engineering

Despite some of the interns being future NFL players themselves, the critical problem was that they didn't have the training to input the social media handles reliably. As a result, it was easy to mistake Anthony Davis, the NFL player, with Anthony Davis, the NBA player, and swap Twitter handles. This feature engineering reliability problem would kill the accuracy of our model. We fixed this by adding automation in the form of Amazon Mechanical Turk. We trained a "quorum" of "turkers" to find the social media handles of athletes, and if 7/9 agreed, we found this equated to around 99.9999% accuracy. Figure 12-4 shows Mechanical Turk labeling system for our social media company.

NOTE

Back in 2014, less was known about data engineering and MLOps. Our labeling system project ran through an incredible athlete, programmer, and former UC Davis graduate Purnell Davis. Purnell developed this system from scratch between doing workouts with athletes like NFL player Marshawn Lynch or 300-pound professional MMA fighters, who trained with our company at the same gym during lunch or at the crack of dawn.

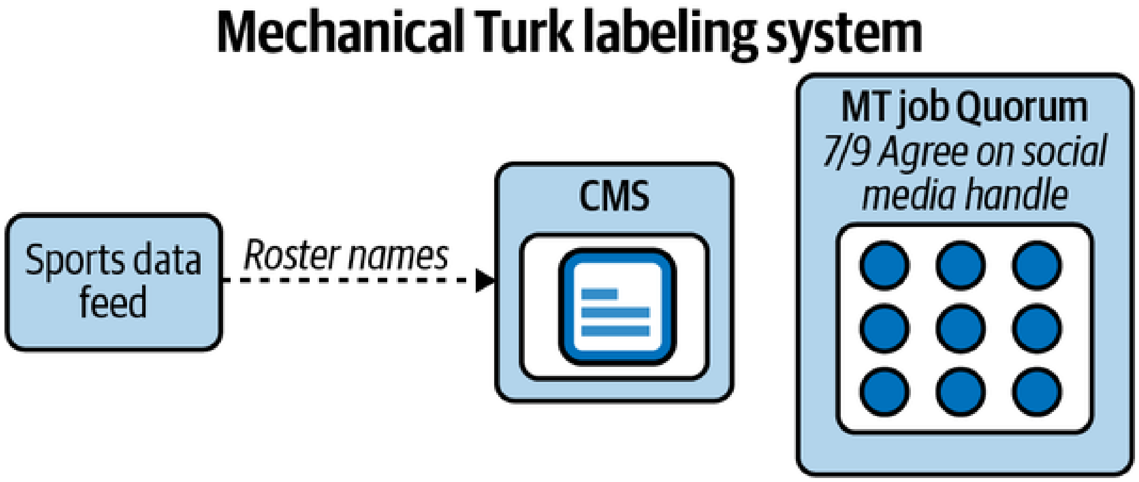

Mechanical Turk labeling system
Sports data feed
Roster names
CMS
MT job Quorum
7/9 Agree on social media handle

Figure 12-4. Mechanical Turk labeling

Influencer Rank

Once we could get the data labeled, we had to collect data from social media APIs. You can find an excellent example of the type of code necessary to bring this into an ML Engine in this repository.

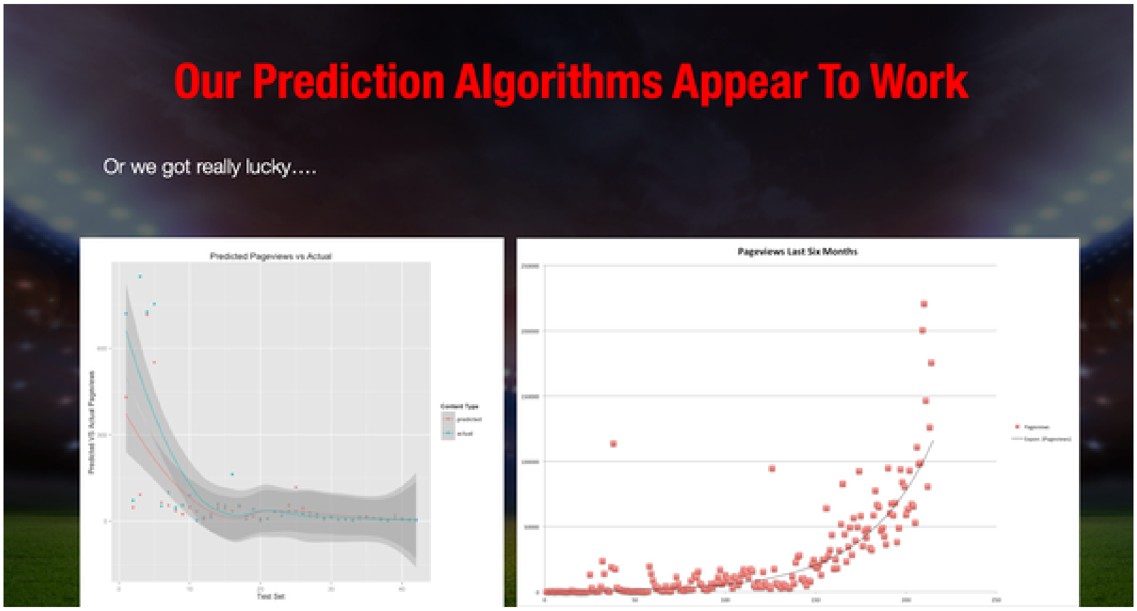
For example, LeBron James's statistics on Twitter look like the following according to our data collection API:

```
Get status on Twitter

        df = stats_df(user="KingJames")
        In [34]: df.describe()
        Out[34]:
         favorite_count retweet_count
        count 200.000000 200.000000
        mean 11680.670000 4970.585000
        std 20694.982228 9230.301069
        min 0.000000 39.000000
        25% 1589.500000 419.750000
        50% 4659.500000 1157.500000
        75% 13217.750000 4881.000000
        max 128614.000000 70601.000000

        In [35]: df.corr()
        Out[35]:
         favorite_count retweet_count
        favorite_count 1.000000 0.904623
        retweet_count 0.904623 1.000000
```

NOTE

If you are familiar with the show *The Shop* on HBO, you will know Maverick Carter and LeBron James. We "almost" officially worked with them, but the talks didn't work out. We ultimately developed a relationship with FC Bayern Munchen instead as a critical partner.

This collection data then fed into a prediction model. A talk I gave at the R meetup in the Bay Area in 2014 shows some of our handiwork. The initial model used the R library Caret to predict the pageviews. In Figure 12-5 the original prediction algorithms for finding influencers was done in R and this ggplot-based chart shows the accuracy of the prediction.


Our Prediction Algorithms Appear To Work

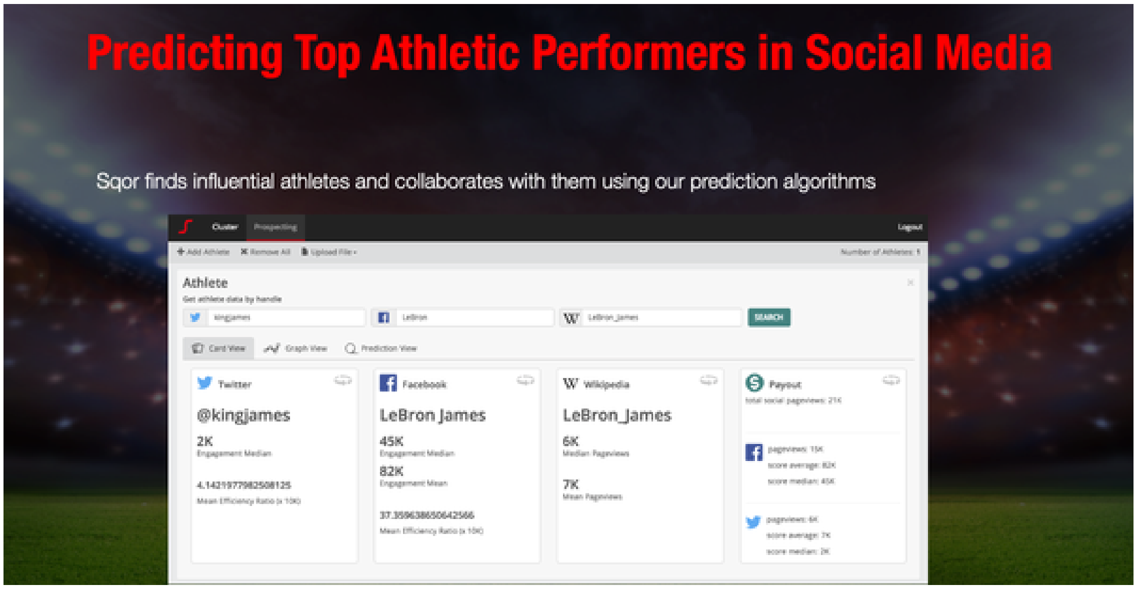
Or we got really lucky....

Figure 12-5. Predicted pageviews from social media influencers versus actual pageviews

We then built a payment system model using the pageviews as the target metric. Ultimately, it led to exponential growth that fueled a quick expansion into millions of pageviews a month. But, as I mentioned earlier, the scary part was the sleepless nights wondering if I was bankrupting the company as we spent millions of dollars on the predictions I helped create.

Athlete Intelligence (AI Product)

With a core MLOps pipeline serving out predictions and fueling our growth without purchasing ads, we started to evolve our product into an AI Product called “Athlete Intelligence.” We had two AI product managers who managed this full time. The general idea of the core product was for “influencers” to understand what they could expect in payment and brands to use this dashboard to partner directly with athletes. In Figure 12-6, we used unsupervised machine learning to categorize different aspects of athletes, including their social media influence.


Predicting Top Athletic Performers in Social Media

Sqor finds influential athletes and collaborates with them using our prediction algorithms

Athlete

Get athlete data by handle

kingjames

LeBron

LeBron_James

Twitter

@kingjames

2K

Engagement Median

4.1421977982508125

Mean Efficiency Ratio (x 10K)

Facebook

LeBron James

45K

Engagement Median

82K

Engagement Mean

37.359638650642566

Mean Efficiency Ratio (x 10K)

Wikipedia

LeBron_James

6K

Median Pageviews

7K

Mean Pageviews

Payout

total social pageviews: 21K

pageviews: 15K

score average: 82K

score median: 45K

pageviews: 6K

score average: 7K

score median: 2K

Figure 12-6. Athlete intelligence

Additional features included unsupervised machine learning that “clustered” similar athlete social profiles. This feature allowed us to package different types of athletes into influencer marketing bundles (Figure 12-7).

Figure 12-7. Athlete intelligence clustering dashboard

Finally, this growth led to two new revenue products for our company. First was a merchandising platform built on top of Shopify that turned into a 500k/year business, and a second was a multimillion dollar influencer marketing business. We directly contacted brands and connected them with the influencers on our platform. An excellent example of this is the “Game of War” commercial we brokered with Machine Zone and Conor McGregor.

In a nutshell, having an effective MLOps pipeline led us to both users and revenue. Without the prediction model, we would have had no users; without the users, we would have had no income; without income, we would have no business. Practical MLOps pay dividends.

Let's talk more about the open versus closed system in the next section. Things seem simple in a controlled environment like academic data science, the martial arts gym in a GI, or theoretical advice on nutrition like calories in versus calories out. Real-world or open systems have many additional influences, though, that are much more challenging to harness.

The Perfect Technique Versus the Real World

Does an armbar (an arm attack that breaks an arm by hyperextending the elbow) by a Brazilian Jiu-Jitsu black belt in the GI (heavy cotton jacket uniform in martial arts) have the same effectiveness against an opponent not wearing a GI? Likewise, in machine learning, what is most important? Is it the accuracy, the technique, or the ability to provide value to a customer or knowledge to a situation?

NOTE

If you are curious about what it looks like to use an armbar in competition, google "Ronda Rousey armbar," and you will see a master technician at work.

What about a street fight versus a grappling match or a UFC match versus a grappling match? Even nonexperts would agree that the more rules involved in a technique, the less chance it has to succeed in an environment with little or no rules, i.e., the real world.

I learned a lesson by traveling to random gyms to train across the United States when I was on business. On more than one occasion, a brown or black belt expert in doing GI Brazilian Jiu-Jitsu would spar with me in NOGI (no uniform) and instantly try to perform an armbar. I have spent years training with professional fighters who practice NOGI. After getting caught perhaps hundreds of times in the submission, I had learned how to easily escape it by slipping my arm out in a specific way. How did I understand this? I just copied what the pro fighters both did and taught me and practiced it over and over again.

Many experts in GI, from black to brown belts to Olympic medalists, said similar things. “You can’t do this in GI,” or “I would have had that in GI,” or “you shouldn’t do this,” etc., as they had a horrified or surprised look on their face. Their “perfect model” didn’t work and their worldview was shocked. Did this mean I was even close to as good as these experts? No, it meant that they had much better technique than I did in a context that didn’t apply to the one we were currently in, grappling without a uniform. Notice in Figure 12-8 how closing the system, or adding rules, to the outside world adds more control yet less realism.

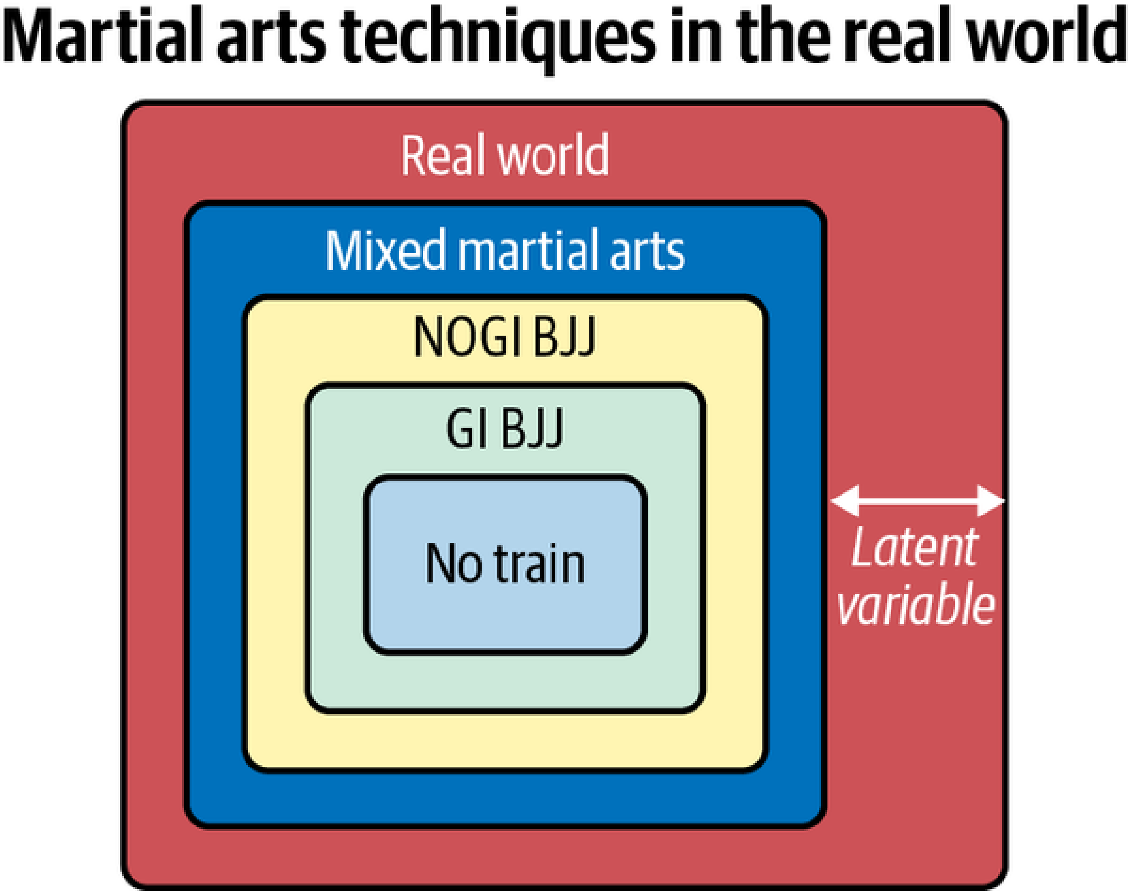


Figure 12-8. Martial arts techniques in the real world

In the book *Critical Thinking* (MIT Press), Jonathan Haber mentions how Ann J. Cahill and Stephan Bloch-Schulman of Elon University make their higher education class like a martial arts studio:

> *In such [martial arts] classes, at each successive level of assessment, students are also required to demonstrate that they have maintained the skills they achieved in previous belt levels. Importantly, a decent sensei does not award a belt based on effort: whether a student has tried hard to master a certain action is not relevant. The question is, can the student throw the punch?*

NOTE

When explaining this same MMA (mixed martial arts) and machine learning theory in a class I teach in Northwestern's data science program, I found out one of my students spent 14 years in the NFL. He then pointed out what he admired about MMA because the best athletes don't have technique agendas. Instead, they wait to see what happens and apply the correct technique for the situation at hand.

INTERVIEW: MARTIAL ARTS AND ML MODELING

I interviewed a top black belt and former mixed martial arts fighter to discuss other real-world martial arts and how it relates to ML modeling:

> *Q: Who are you are and what is your experience in martial arts, sports, and fighting?*
>
> > *A:* My name is Jacob Hardgrove. I have been training in Brazilian Jiu-Jitsu for ~20 years. I have competed in both submission grappling and professional MMA events and have an MMA record of 3-0. Besides the BJJ training, to prepare for the MMA events, I have spent a significant amount of time studying and training other forms of martial arts such as western boxing, muay Thai, freestyle, and catch wrestling.
>
> *Q: Why do some techniques taught very formally, say GI BJJ, not work as well in a real fight, either on the street or in the cage?*
>
> > *A:* My thought process on this question leads me to consider a dichotomy between introducing a concept/technique and the successful implementation of said concept/technique in an uncontrolled environment/situation.
> >
> > To introduce a technique or concept to a beginner, the instructor strips the technique of as many variables as possible to create an easily repeatable framework. This framework should be as stripped down as much as possible and consist of the primary, essential components required to achieve the desired effect. This framework, according to most traditional martial arts, is referred to as "Kata."
> >
> > Now, I see people who end up submerging themselves in the world of Jiu-Jitsu lose sight of the fact that they are practicing martial skills against other martial artists and not against untrained but potentially still viable, violent attackers. Unfortunately, this truth can lead to a severe case of "missing the point."

Much like Jacob, the black belt former MMA fighter, I see similar issues in real-world machine learning. As a professor, I teach machine learning at top universities and also am an industry practitioner. The academic tools like sklearn and pandas and the educational datasets on Kaggle are the "Kata." These datasets and tools are necessary to "strip away" the real-world complexity to teach the material. A critical next step is to make the student aware that machine learning Kata is not real-world. The real world is much more dangerous and complicated.

Critical Challenges in MLOps

Let's discuss some specific challenges in bringing machine learning to production. Three main challenges are ethical and unintended consequences, lack of operational excellence, and focus on prediction accuracy versus the big picture. Operational excellence in implementing machine learning is what matters much more than the technique in the real world. This exact point is clearly articulated in the research paper "Hidden Technical Debt in Machine Learning Systems". The paper's authors found "…it is common to incur massive ongoing maintenance costs in real-world ML systems." Let's discuss some of these issues next.

Ethical and Unintended Consequences

It is easy to get "hand-wavy" or self-righteous about ethics and turn people off. This fact doesn't mean the topic doesn't need discussion. Many companies that deal with social media have created weapons of mass disinformation. According to Tristan Harris, "64% of extremist groups that people joined were due to Facebook's own recommendation system" and YouTube "recommended Alex Jones, InfoWars, conspiracy theory videos 15 billion times. So that's more than the combined traffic of the Washington Post, BBC, Guardian, Fox news combined", and 70% of all watch time on YouTube is recommendations.

The consequences are no longer theoretical. Many social networks and big technology companies are actively changing their approach to machine learning–based enabled systems. These updates range from putting facial recognition systems on hold to evaluating the outcomes of recommendation engines more closely. Ethical considerations in the real world also have to be part of an MLOps solution, not just whether the technique has predictive power.

Already some solutions to the problem of echo chambers appear—for example, the *IEEE Spectrum* in the article "Smart Algorithm Bursts Social Networks *Filter Bubbles*". They go on to say, "one team of researchers in Finland and Denmark has a different vision for how social media platforms could work. They developed a new algorithm that increases the diversity of exposure on social networks, while still ensuring that content is widely shared."

This situation is where ethics come into play. If a company or machine learning engineer is 100% focused on increasing engagement and profit, they may not want to lose 10% of their profit to "save the world." This ethical dilemma is a classic case study in externalities. A nuclear power facility might deliver tremendous energy benefits, but if they dump the nuclear waste into the ocean, other innocent victims pay the price yet reap no reward.

Lack of Operational Excellence

Another antipattern of machine learning engineering is the inability to maintain a machine learning model effectively. A good way to reason about this situation is to consider building a wood bookshelf versus growing a fig tree. If you make a bookshelf, you will probably never modify it again; once designed, that is the end of the project. On the other hand, a fig tree is part of an open system. It needs water, sun, soil, nutrients, wind, trimming of branches, and protection from insects and disease. The fig tree adapts to the environment over time but needs oversight from the person who wants to enjoy the figs later. In other words, the fig tree needs maintenance to produce high-quality fruit. A machine learning system, like any software system, is never finished like a bookshelf. Instead, it requires nurturing like a fig tree.

Next, let's imagine a company that predicts credit limits for each new furniture store customer. Let's say an initial model was developed in January 2019. Since then, a lot has changed. There could be substantial data drift from the features used to create the model. We know that many retail stores in 2020–2021 went out of business, and the mall business model is under serious threat. COVID-19 accelerated a declining trend for physical retail, among other updates in the underlying data. As a result, the data "drifted," i.e., purchase patterns are widely different post-COVID-19, and the original static machine learning model may not work as intended.

A machine learning engineer would be wise to account for constant retraining of the model (nurturing the fig tree) by setting alerts on the data changes when a data drift threshold occurs. Additionally, business metric monitoring could send an alert if late payments exceed a threshold. For example, let's imagine that typically a company selling furniture has 5% of customers that pay 30 days late on their credit. If suddenly 10% of the customers pay 30 days late, then the underlying ML model may need an update.

The concept of alerts and monitoring for machine learning should not be a surprise to traditional software engineers. For example, we monitor CPU load on individual servers, latency for mobile users, and other key performance indicators. With machine learning operations, the same concept applies.

Focus on Prediction Accuracy Versus the Big Picture

As discussed earlier in the chapter, even world-class practitioners can sometimes focus on the technique at the Big Picture's expense. But, like the "perfect armbar," which doesn't quite work in a different situation, a model can be just as brittle. A great example of the tug of war is the Nate Silver versus Nassim Taleb debate. Issac Faber has an excellent breakdown of the critical points in a Medium article. Isaac points out that not all models are perfect replicas of the real world because they cannot tease out the uncertainty of what we do not know, i.e., immeasurable risk. The debate between Nate Silver versus Nassim Taleb comes down to whether it is possible to model elections or whether it is an illusion to think we can model them. The election events of 2020 do seem to put points in favor of Nassim Taleb.

The black-belt Jacob says the same thing when he points out that a Kata is a simplified version of a practice technique. The complexity of the real world and the training exercise is in inherent conflict. The model or technique may be flawless, but the real world may not care. For example, in the case of the 2020 presidential election, even the best modeler didn't anticipate the probability of an insurrection or state officials pressured to throw out ballots. As Issac describes this problem, it is the difference between aleatory and epistemic uncertainty. The aleatory risk you can measure, say, the probability of heads or tails, but the epistemic risk you cannot, like an insurrection that could overturn a presidential election.

Dr. Steven Koonin, whom I worked for at Caltech for several years, says something similar in his book about climate science. Just like election predictions, nutritional science, and other complex systems, the topic of climate science is instantly polarizing. In the book, *Unsettled* (BenBella Books) Koonin says, "Not only can we be fooled by failing to take a 'big picture view of climate over time, but we can also be fooled by failing to take a big picture view of the planet." Further, he goes on to say, "Projections of future climate and weather events rely on models demonstrably unfit for the purpose." Whether you believe his statement or not, it is worth considering the complexity of real-world modeling of a complex system and what can go wrong.

A good summary of this dilemma is to be cautious about the confidence in a prediction or technique. One of the scariest and most talented grapplers I trained with, Dave Terrell, who fought for the UFC championship, told me, don't ever get in a fight with multiple opponents. The more skill and "skin in the game" a practitioner has, the more they become aware of the epistemic risk. Why risk your life in a street fight with multiple people even if you are a world-class martial artist? Likewise, why be more confident than you should when you predict an election, stock prices, or natural systems?

In practice, the best way to approach this problem in production is to limit the technique's complexity and assume a lower knowledge of epistemic uncertainty. This takeaway could mean a traditional machine learning high explainability is better than a complex deep learning model with marginally better accuracy.

INTERVIEW WITH MLOPS PRACTITIONERS: PIERO MOLINO

What is your background, and how did you get involved with operationalizing machine learning?

A: I studied computer science at the University of Bari in Italy, where I got a PhD working on open domain question answering (at the intersection of NLP, ML, and information retrieval). Since when I was a student I was trying to use the research systems I was building for creating products that could be used in the real world as opposed to lab prototypes that was never practically used. This led me to work on applied NLP and ML at a few companies (Yahoo!, IBM Watson, Geometric Intelligence, and Uber AI). In most cases I did both research and applications, that's the intersection where I like to work: new ideas that end up being used by people. In Uber AI in particular I touched many ML applications in the company (customer support, expected time of delivery, food and restaurant recommendation, driver dialogue systems), and I developed Ludwig to make my own life easier in jumping from one project to the next, because it allowed me to avoid reinventing the wheel (data preprocessing, a training loop, evaluation functions, etc.) every single time, but made it possible to create prototypes in minutes.

What are 3–5 of the most important things to be aware of deploying and maintaining machine learning systems at scale?

A: My experience has taught me that the model itself is just a relatively small piece inside a broader system when you productionize it, but at the same time it's both one of the most critical (if the model doesn't return satisfactory predictions, all the rest is kinda pointless), and one of the more complex to get right, because data, infrastructure, monitoring (that are very important) have more established standards and are more engineered, so there’s less uncertainty on the outcome as opposed to the task of model building. That uncertainty needs to be taken into account in the ML development process.

Another important learning is that the reality of deploying an ML system and observing users behavior often breaks our assumptions, and so the process becomes iterative. For instance, live input data the system receives may end up being very different from training data, the distribution of both inputs and outputs may shift over time, and the fact that the system is being used in many cases will impact the distribution of the data that will be collected (think of a recommender system; the data collected after its first adoption will certainly be biased by the suggestions of the recommender itself). For this reason, monitoring becomes extremely important.

Finally, the data collection process is also extremely important. In my experience I worked on both projects where the training data was obtained through labeling and where data was obtained as a byproduct of a process that happened within the organization. In the former case, defining precisely the data collection process, giving exact instructions to annotators, observing their annotations and their agreement and iterating the process was much more involved than simply submitting data to be annotated and getting it back. In the latter case, deeply understanding the process that generates the data was literally the only way one could make sense of the data, and helps identifying outliers, set model expectations correctly, and understand the uncertainty in model predictions. In my experience, this degree of understanding together with the analysis of model predictions led also to improving the data collection process itself, which in turn led to less noisy data that was used to train better models, so there’s a virtuous loop at play here.

What are you most excited about right now with machine learning and why?

A: There are two things that excite me the most. On one hand you have the percolation of ML in other scientific fields, like biology, chemistry, and physics, but also entirely new industries, like graphics and video games (ML for content generation, rendering, and animation, for instance). On the other hand you have what I'm actively working on (I wouldn't work on it if it didn't excite me), which is creating tools and abstractions that make it easier for people without an ML background to use ML for their purposes. The two intersect well; if more people, maybe domain experts in their fields, can access and use ML, the percolation will be faster.

What are the top 3–5 things a person reading this book can do to succeed in their career doing MLOps?

A: Learn to deal with the uncertainty associated with the ML development process, learn to work in cross-functional teams that include people with little to no ML expertise, create repeatable processes, learn to put themselves in the users' shoes, strive to avoid technical debt.

How can people get hold of you and what would you like to share that you are working on?

A: I'm not an avid social media user, but I occasionally post about what I'm working on on Twitter (@w4nderlus7) and on LinkedIn. I'm currently co-running the Stanford MLSys seminar series which I believe most of the readers would be interested in. I also have a personal website that I update with my projects, my publications, and some lectures I give. My current main interest is in building tools and abstractions to make ML more accessible to people with less or no ML expertise, so that they can leverage their domain expertise to train and use models achieve their goals.

What is your background, and how did you get involved with operationalizing machine learning?

A: I have a background in Economics. Before joining Microsoft, I was Research Fellow in Business Economics at Harvard University, where I was performing statistical and econometric analysis within the Technology and Operations Management Unit. At Harvard, I worked on multiple patent, publication, citations data-driven projects to investigate and measure the impact of external knowledge networks on companies' competitiveness and innovation. This experience gave me the unique opportunity to learn how to extract knowledge from big data, build predictive models from scratch, and how to code in R and Python. This was my first step into the fantastic world of machine learning!

A few years later, I started my career at Microsoft as a data scientist: in my role at Microsoft, I was helping companies transform their operations through machine learning algorithms and AI and, in particular, I was in charge of moving their models from development to a production environment in ways that were robust, fast, and repeatable.

What are 3–5 of the most important things to be aware of deploying and maintaining machine learning systems at scale?

A: Many companies see machine learning deployment as a technical practice. However, it is more of a business-driven initiative that starts within the company; in order to become an AI-driven company, it is important that the people who today successfully operate and understand the business collaborate closely with those teams that are responsible for the machine learning deployment workflow. It is very essential to maintain constant interaction to understand the model experimentation process parallel to the model deployment and consumption steps. Most organizations struggle to unlock machine learning's potential to optimize their operational processes and get data scientists, analysts, and business teams speaking the same language. Moreover, machine learning models must be trained on historical data, which demands the creation of a prediction data pipeline, an activity requiring multiple tasks including data processing, feature engineering, and tuning. Each task, down to versions of libraries and handling of missing values, must be exactly duplicated from the development to the production environment. Sometimes, differences in technology used in development and in production contribute to difficulties in deploying machine learning models. Finally, data science languages can be slow. Python is one of the most popular languages for machine learning applications, but complete production models are rarely deployed in those languages for speed reasons. Porting a Python model into a production language like C++ or Java is challenging, and often results in reduced performance of the original, trained model.

What are you most excited about right now with machine learning and why?

A: I am very excited about the extensive adoption of open source frameworks (such as Fairlearn and InterpretML) to support the transparent interpretability and guarantee the fairness of machine learning algorithms. Data scientists know that accuracy is no longer the only concern when developing machine learning models; interpretability and fairness must be considered as well. In order to make sure that machine learning solutions are fair and the value of their predictions easy to understand and explain, it is essential to build open source tools that developers and data scientists can use to assess their machine learning system's fairness and mitigate any observed unfairness issues.

What are the top 3–5 things a person reading this book can do to succeed in their career doing MLOps?

A: They can:

1. Create reproducible ML pipelines.
2. Capture the governance data for the end-to-end ML lifecycle.
3. Monitor ML applications for operational and ML-related issues.

How can people get ahold of you, and what would you like to share that you are working on?

> *A:* You can follow me on Twitter @frlazzeri, LinkedIn, and Medium). I recently wrote a book, *Machine Learning for Time Series Forecasting with Python* (Wiley, 2020), in which you can find real-world examples, resources, and concrete strategies to explore and transform data and develop usable, practical time series forecasts. At Microsoft I lead an international team (in the USA, Canada, UK, and Russia) of engineers and cloud developer advocates, managing a large portfolio of customers. My team is in charge of building technical content and intelligent automated solutions on Azure, utilizing techniques spanning from IoT, time series forecasting, computer vision, natural language processing, and open source frameworks. Currently I also teach "Introduction to AI with Python" at Columbia University. You can read more about my teaching philosophy and experience in my article, "The Importance of Teaching Machine Learning".

Final Recommendations to Implement MLOps

Before we wrap up, we want to walk you through some tips for implementing MLOps in your organization. First, here are a set of final, global recommendations:

- Start with small wins.
- Use the cloud, don't fight the cloud.
- Get you and your team certified on a cloud platform and an ML specialization.
- Automate from the start of a project. An excellent initial automation step is the continuous integration of your project. Another way of putting this is that "if it isn't automated, it is broken."
- Practice Kaizen, i.e., continuous improvement with your pipeline. This method improves the software quality, the data quality, the model

quality, and the customer feedback.
- When dealing with large teams or big data, focus on using platform technology such as AWS SageMaker, Databricks, Amazon EMR, or Azure ML Studio. Let the platform do the heavy lifting for your team.
- Don't focus only on the complexity of techniques, i.e., deep learning versus solving the problem with any tool that works.
- Take data governance and cybersecurity seriously. One way to accomplish this is by using enterprise support for your platform and having regular audits of your architecture and practices.

Finally, there are three laws of automation to consider when thinking about MLOps:

1. Any task that talks about being automated will eventually be automated.
2. If it isn't automated, it's broken.
3. If a human is doing it, a machine eventually will do it better.

Now, let's dig into some tips for dealing with security concerns.

Data Governance and Cybersecurity

There are two seemingly conflicting problems in MLOps: increased cybersecurity concerns and lack of machine learning in production. On the one hand, too many rules, and nothing gets done. On the other hand, ransom attacks on critical infrastructure are increasing, and organizations would be wise to pay attention to how they govern their data resources.

One way to address both issues simultaneously is to have a checklist of best practices. Here is a partial list of best practices for data governance that will improve MLOps productivity and cybersecurity:

- Use PLP (Principle of Least Privilege).
- Encrypt data at rest and in transit.
- Assume systems that are not automated are insecure.
- Use cloud platforms since they have a shared security model.
- Use enterprise support and engage in quarterly architecture and security audits.

- Train staff on platforms used by getting them certified.
- Engage company in quarterly and yearly training on new technology and best practices.
- Create a healthy company culture with standards of excellence, competent employees, and principled leadership.

Next, let's summarize some MLOps Design Patterns that may be useful to you and your organization.

MLOps Design Patterns

The following examples reflect a partial list of recommended MLOps Design Patterns:

CaaS

> Container as a service (CaaS) is a helpful pattern for MLOps because it allows developers to work on an ML microservice on their desktop or in a cloud editor and then shares it with other developers or the public via a `docker pull` command. Further, many cloud platforms offer high-level PaaS (platform as a service) solutions to deploy containerized projects.

MLOps Platform

> All cloud providers have deeply integrated MLOps platforms. AWS has AWS SageMaker, Azure has Azure ML Studio, and Google has Vertex AI. In cases where a large team, a large project, big data, or all of the above come into play, using the MLOps platform on the cloud you are using will save you a tremendous amount of time in the building, deploying, and maintaining of your ML application.

Serverless

> Serverless technology like AWS Lambda is ideal for the rapid development of ML microservices. These microservices can call into cloud AI APIs to do NLP, computer vision, or other tasks or use a pretrained model you developed yourself or one that you downloaded.

Spark-Centric

Many organizations dealing with big data already have experience using Spark. In this situation, it may make sense to use the MLOps capabilities of the managed Spark platform Databricks or managed Spark via the cloud platform, such as AWS EMR.

Kubernetes-Centric

Kubernetes is a “cloud in a box.” If your organization is already using it, it may make sense to use the Kubernetes ML-focused technology like mlflow.

In addition to these recommendations, many additional resources in Appendix B discuss topics ranging from data governance to cloud certification.

Conclusion

This book originally came about from a discussion I had with Tim O’Reilly and Mike Loukides at Foo Camp around making ML go 10X faster. The consensus from the group was, yes, it can go 10X faster! Matt Ridley’s book *How Innovation Works: And Why It Flourishes in Freedom* (Harper) illustrates that the nonintuitive answer is that innovation is a recombination of ideas with better execution. A current colleague and former professor, Andrew Hargadon, first introduced me to these ideas when I was an MBA student at UC Davis. In his book *How Breakthroughs Happen* (Harvard Business Review Press) Andrew mentions that it is the network effect and the recombination of ideas that matters.

For MLOps, this means that the operational excellence of existing ideas is the secret sauce. Companies that want to solve real problems in machine learning and solve these problems quickly can innovate through execution. The world needs this innovation to help us save more lives through preventive medicine like automated higher accuracy cancer screenings, self-driving cars, and clean energy systems that adapt to the environment.

Nothing should be “off the table” to increase operational excellence. If AutoML speeds up rapid prototyping, then use it. If cloud computing increases the velocity of machine learning model deployment, then implement it. As recent events have shown us, with the COVID-19 pandemic and the resulting breakthrough technology innovations like the CRISPR technology and COVID-19 vaccine, we can do incredible things with the proper sense of urgency. MLOps adds rigor to this urgency and allows us to help save the world, one ML model at a time.

Exercises

- Build a machine learning application as quickly as possible that is both continuously trained and continuously deployed.
- Deploy a machine learning model using the Kubernetes stack.
- Deploy the same machine learning model using continuous delivery with AWS, Azure, and GCP.
- Create an automated security scan of the containers for a machine learning project using a cloud native build system.
- Train a model using a cloud-based AutoML system and using a local AutoML system like Create ML or Ludwig.

Critical Thinking Discussion Questions

- How could you build a recommendation engine that didn’t have as many negative externalities as current social media recommendation engines? What would you change, and how?
- What could be done to improve the accuracy and interpretability of modeling complex systems like nutrition, climate, and elections?
- How could operational excellence be the secret ingredient for a company wanting to be a machine learning–related technology leader?
- If operational excellence is a crucial consideration for MLOps, what are your organization’s hiring criteria to identify the right talent?
- Explain the role of operational excellence in machine learning concerning enterprise support for cloud computing? Does it matter, and why?

Support Sign Out