Machine Learning Systems Design

Deployment and Monitoring

Lecture 22: Model Maintenance

CE 40959 Spring 2023
Ali Zarezade
SharifMLSD.github.io

Agenda

1. **Continual learning**

1. Continual Learning

Model’s performance degrades in production

- Data distribution shifts
  - Sudden
  - Cyclic
  - Gradual

From Monitoring to Continual Learning

- Monitoring: detect changing data distributions
- Continual learning: continually adapt models to changing data distributions

Continual learning

- Set up infrastructure such that models can continuously learn from new data in production

Continual learning is especially good for

- Natural labels: e.g. user click -> good prediction
- Short feedback loops
- Examples:
  - RecSys
  - Ranking
  - Ads CTR prediction
  - eDiscovery

Data shift and continual learning

- How do we adapt our models to data distribution shifts?
  - *by continually updating our ML models*
- The goal of continual learning is to safely and efficiently automate the update.

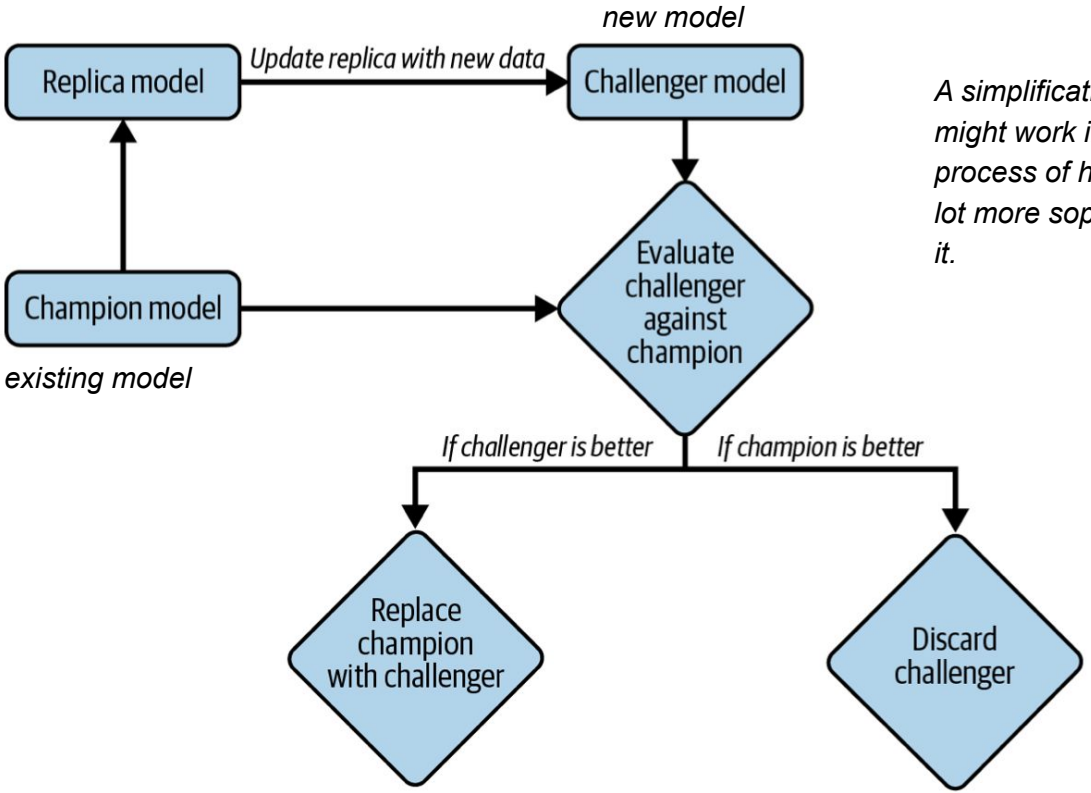
What is continual learning?

- A model updates itself with every incoming sample in production!?
  - Very few companies actually do that.
  - May cause catastrophic forgetting in DNNs.
  - It can make training more expensive

What is continual learning?

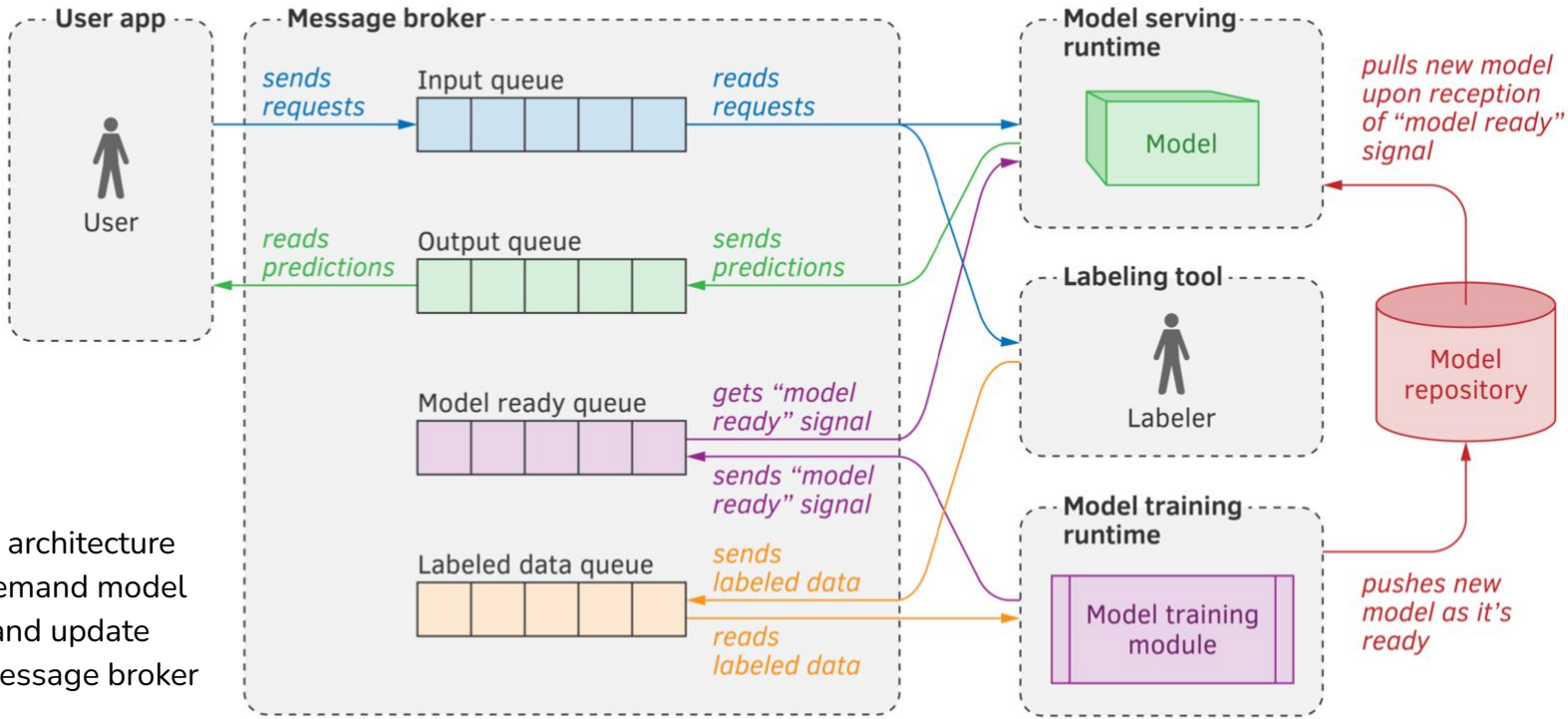
- A model updates itself with every incoming sample in production!?
  - Very few companies actually do that.
  - May cause catastrophic forgetting in DNNs.
  - It can make training more expensive
- Companies that employ continual learning in production update their models in micro-batches.
  - For example, they might update the existing model after every 512 examples (optimal no.?)
- The updated model shouldn’t be deployed until it’s been evaluated.

How continual learning might work in production



A simplification of how continual learning might work in production. In reality, the process of handling the failed challenger is a lot more sophisticated than simply discarding it.

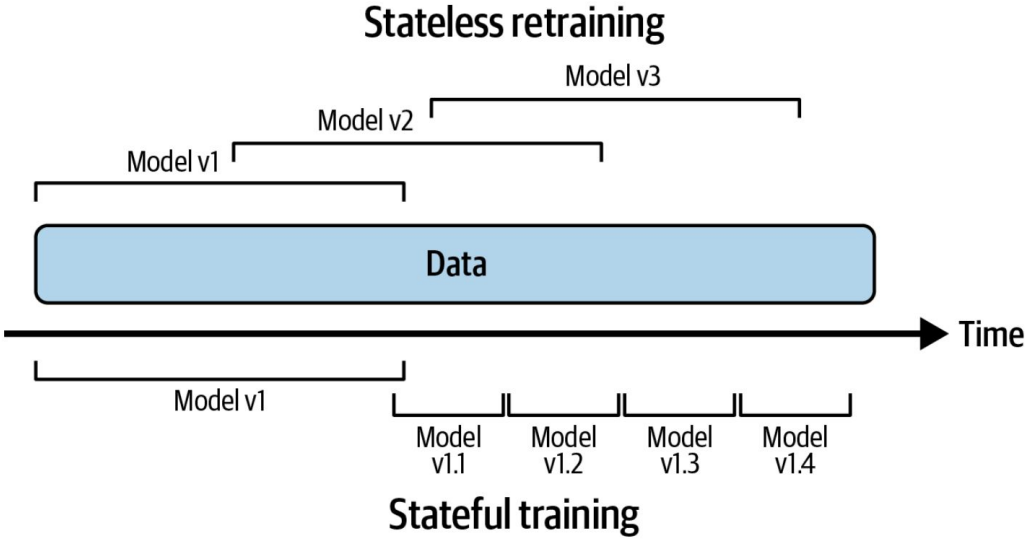
How to update?



Example architecture for on-demand model serving and update with a message broker

Stateless retraining vs stateful training

- **Stateless retraining**: the model is trained from scratch each time.
- **Stateful training**: the model continues training on new data (fine-tuning or incremental learning).



Stateful training

- Update model with less data.
- Models converge faster and require much less compute power.
- It might be possible to avoid storing data altogether! (privacy, big data)
- May occasionally train model from scratch to calibrate it.

Two types of model updates

- **Model iteration**: A new feature is added to an existing model architecture or the model architecture is changed.
- **Data iteration**: The model architecture and features remain the same, but you refresh this model with new data.

Two types of model updates

- **Model iteration**: A new feature is added to an existing model architecture or the model architecture is changed.
- **Data iteration**: The model architecture and features remain the same, but you refresh this model with new data.

As of today, stateful training is mostly applied for data iteration, as changing your model architecture or adding a new feature still requires training the resulting model from scratch.

Model iteration without training from scratch!

There are some research showing that it might be possible to bypass training from scratch for model iteration by using techniques such as:

- Knowledge transfer (Google, 2015)
- Model surgery (OpenAI, 2019)

Surgery transfers trained weights from one network to another after a selection process to determine which sections of the model are unchanged and which must be reinitialized

Terminology Ambiguity

We use the term “continual learning” instead of “online learning”.

- online learning != online education
- online learning := specific setting where a model learns from each incoming new sample
- continual learning != continuous learning (CI/CD/CT)

Why continual learning?

- To tackle data distribution shifts (ads, ride-sharing, recsys)

Why continual learning?

- To tackle data distribution shifts (ads, ride-sharing, recsys)
- To adapt to rare events (to improve performance in black friday, model should learn throughout the day with fresh data.)
    - Christmas/Black Friday/Prime Day shopping

In 2019, Alibaba acquired Data Artisans, for $103 million so that the team could help them adapt Flink for streaming ML use cases. Making better recommendations on Singles Day, a shopping occasion in China similar to Black Friday in the US.

Why continual learning?

- To tackle data distribution shifts (ads, ride-sharing, recsys)
- To adapt to rare events (to improve performance in black friday, model should learn throughout the day with fresh data.)
- To overcome is the continuous cold start problem (make predictions for a new user without any historical data).
  - New users
  - New devices
  - Users not logged in
  - Users rarely logged in

Generalization of the cold start problem. It can happen not just with new users but also with existing users. For example, it can happen because an existing user switches from a laptop to a mobile phone.

It can also happen when a user visits a service so infrequently that whatever historical data the service has about this user is outdated.

Why continual learning?

- To tackle data distribution shifts (ads, ride-sharing, recsys)
- To adapt to rare events (to improve performance in black friday, model should learn throughout the day with fresh data.)
- To overcome is the continuous cold start problem (make predictions for a new user without any historical data).
- TikTok: adapt is recsys within a minutes to users.

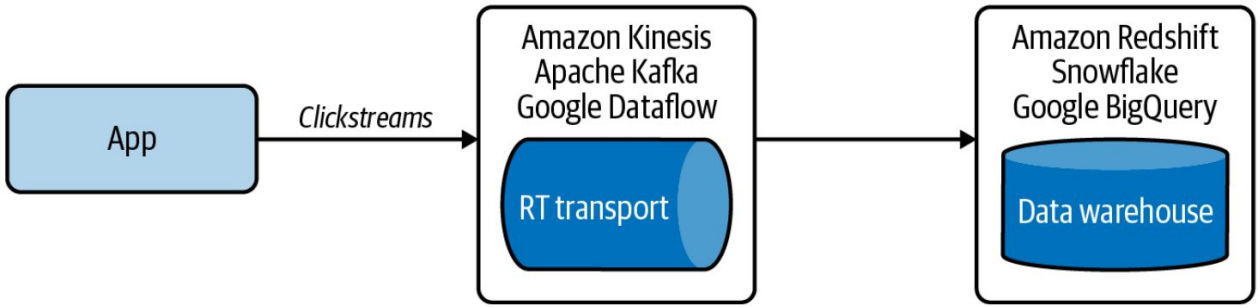
Continual learning challenges

But continual learning still has many challenges:

- Fresh data access challenge
- Evaluation challenge
- Algorithm challenge

Fresh data access challenge

- Many companies pull new training data from their data warehouses.
- The speed at which you can pull data from data warehouses depends on the speed at which this data is deposited into it.
- The speed can be slow, especially if data comes from multiple sources.
- An alternative is to allow pull data before it's deposited into data. warehouses, e.g., directly from real-time transports such as Kafka.



Fresh data access challenge

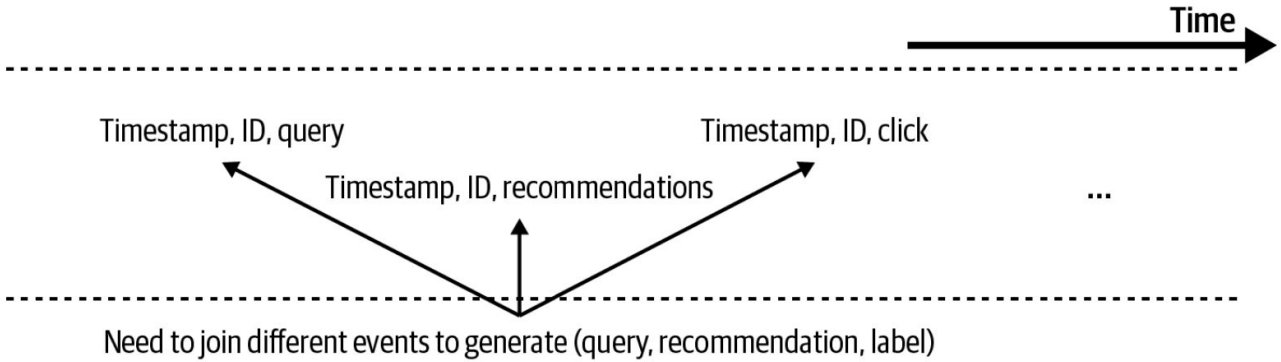
- If your model needs labeled data to update, it will need to be labeled.
- The best candidates for continual learning are tasks where you can get natural labels with short feedback loops (stock, ETA, recsys, dynamic pricing).

Fresh data access challenge

- If your model needs labeled data to update, it will need to be labeled.
- The best candidates for continual learning are tasks where you can get natural labels with short feedback loops (stock, ETA, recsys, dynamic pricing).
- However, natural labels are usually not generated as labels, but rather as behavioral activities that need to be extracted into labels

Label computation can be costly when there are lots of logs.

- Batch (DW)
- Stream (Kafka)



Fresh data access challenge

- The good news is that tooling around streaming is growing fast.
  - Confluent, the platform built on top of Kafka, is a $16 billion company as of October 2021
  - Materialize has raised $100 million to develop a streaming SQL database

Evaluation challenge

- The biggest challenge of continual learning isn’t in writing a function to continually update your model—you can do that by writing a script!
- The biggest challenge is in making sure that this update is good enough to be deployed.

Evaluation challenge

- The more frequently you update your models, the more opportunities there are for updates to fail.
- Continual learning makes your models more susceptible to coordinated manipulation and adversarial attack.

Evaluation challenge

- The more frequently you update your models, the more opportunities there are for updates to fail.
- Continual learning makes your models more susceptible to coordinated manipulation and adversarial attack.

In 2016, Microsoft released Tay, a chatbot capable of learning through “casual and playful conversation” on Twitter. As soon as Tay launched, trolls started tweeting the bot racist and misogynist remarks. Shut downed 16 horse after its launch!

Evaluation challenge

- The more frequently you update your models, the more opportunities there are for updates to fail.
- Continual learning makes your models more susceptible to coordinated manipulation and adversarial attack.
- Test each of your model updates to ensure its performance and safety before deploying the updates to a wider audience.

Evaluation challenge

- The more frequently you update your models, the more opportunities there are for updates to fail.
- Continual learning makes your models more susceptible to coordinated manipulation and adversarial attack.
- Test each of your model updates to ensure its performance and safety before deploying the updates to a wider audience.
- Evaluation takes time, which can be another bottleneck for model update frequency (A/B test against the current model → updates in 2 weeks).

Algorithm challenge

- Neural network can be update with a data batch of any size
- But it is not trivial (or even possible) for some methods like matrix factorization or tree-based algorithms.
  - Hoeffding Tree, incremental learning for tree
  - Incremental matrix factorization
- Not only does the learning algorithm need to work with partial datasets, but the feature extract code has to as well.
  - features using statistics of data
  - you should compute or approximate these statistics incrementally (e.g. Optimal Quantile Approximation in Streams)

Four stages of continual learning

Continual learning isn’t something that companies start out with. The move toward continual learning happens in four stages:

- Stage 1: Manual, stateless retraining
- Stage 2: Automated retraining
- Stage 3: Automated, stateful training
- Stage 4: Continual learning

Stage 1: Manual, stateless retraining

- In the beginning, the ML team often focuses on developing ML models to solve as many business problems as possible.
- So, updating existing models takes a backseat.
- The process of updating a model is manual and ad hoc (from scratch, once in several months)

Stage 2: Automated retraining

- After a few years, you have anywhere between 5 and 10 models in production.
- Your priority is no longer to develop new models, but to maintain and improve existing models.
- You decide to write a script to automatically execute all the retraining steps (might get complicated if there are dependencies among your models).
- The retraining frequency is set based on gut feeling!
- Different models require different retraining schedules: the embedding model might need to be retrained a lot less frequently than the ranking model.

Stage 2: Automated retraining

The feasibility of this stage revolves around the feasibility of writing a script to automate your workflow and configure your infrastructure to automatically:

1. Pull data.
2. Downsample or upsample this data if necessary.
3. Extract features.
4. Process and/or annotate labels to create training data.
5. Kick off the training process.
6. Evaluate the newly trained model.
7. Deploy it.

Stage 2: Automated retraining (requirements)

The three major factors that will affect the feasibility of this script are:

- Scheduler: setup cron, schedulers, or orchestrators (Airflow, Argo).
- Data: setup DW, feature extractor, labeling platform.
- Model store: model versioning, object storage (MinIO, S3)

Stage 3: Automated, stateful training

- In stage 2, each time you retrain your model, you train it from scratch.
- Just re-configure automatic updating script so that, when the model update is kicked off, it first locates the previous checkpoint and loads it into memory before continuing training.
- You need a change in the mindset (stateless → stateful training)
- The main thing you need at this stage is a way to track your data and model lineage:
    - You might want to know how these models evolve over time, which model was used as its base model, and which data was used to update it so that you can reproduce and debug it.

Stage 4: Continual learning

- At stage 3, your models are still updated based on a fixed schedule set out by developers.
- Instead of relying on a fixed schedule, you might want your models to be automatically updated whenever data distributions shift and the model's performance plummets.
- The holy grail is when you combine continual learning with edge deployment.
    - model continually update and adapt to its environment
    - no centralized server cost
    - no need to transfer data
    - better data security and privacy

Stage 4: Continual learning (requirements)

You'll first need a mechanism to trigger model updates.

- Time-based: every five minutes
- Performance-based: whenever model performance plummets
- Volume-based: whenever the total amount of labeled data increases by 5%
- Drift-based: whenever a major data distribution shift is detected

For this trigger mechanism to work, you'll need a solid *monitoring* and *model evaluation* solutions.

How often to update models?

It depends!

How often to update models?

It depends on:

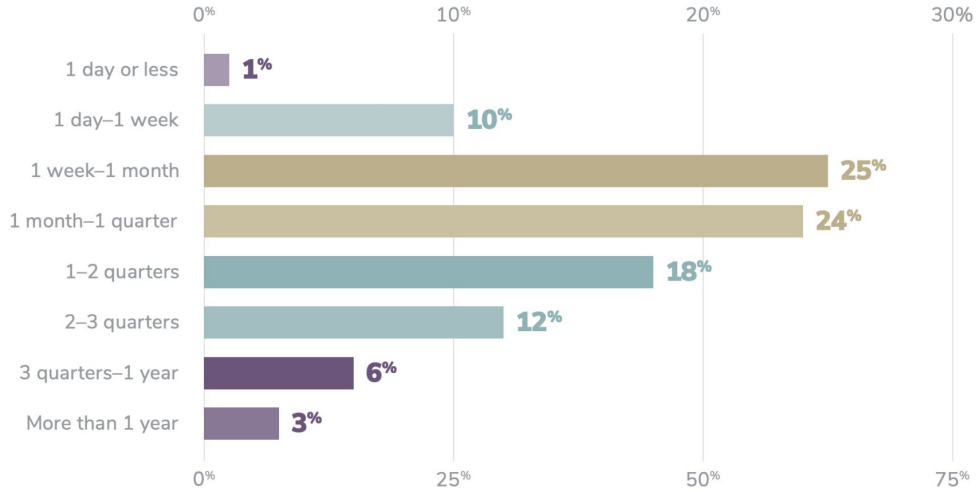
- Business needs and the needs of the user (recsys, ads, google translate)
- Speed of availability of new training data (comments, feedback, labeling, delayed label)
- Update cost: computation cost, and time to train model (speech recognition, logistic regression)
- Rate of model decay: not all models are worth deploying

How frequently should you update your models?

- Very few companies actually update models with each incoming sample
  - Catastrophic forgetting
  - Can get unnecessarily expensive*
- Update models with micro-batches

Iteration cycle: US

Only 11% of organizations can put a model into production within a week, and 64% take a month or longer

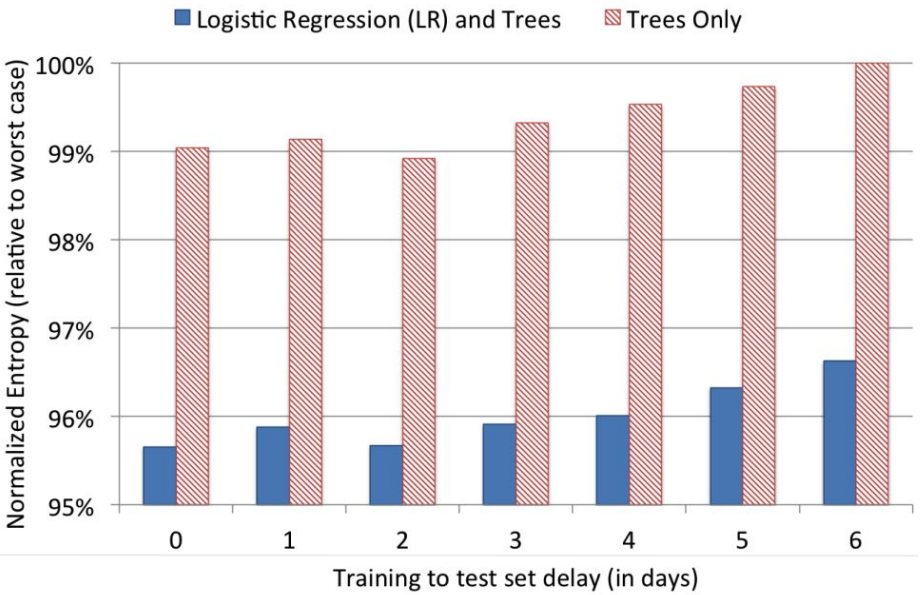


Algorithmia's State of Enterprise Machine Learning (2021)

Quantify the value of data freshness

How much model's performance changes if switch from retraining monthly to weekly to daily to hourly?

- FB: CTR loss can be reduced ~1% going from training weekly to daily

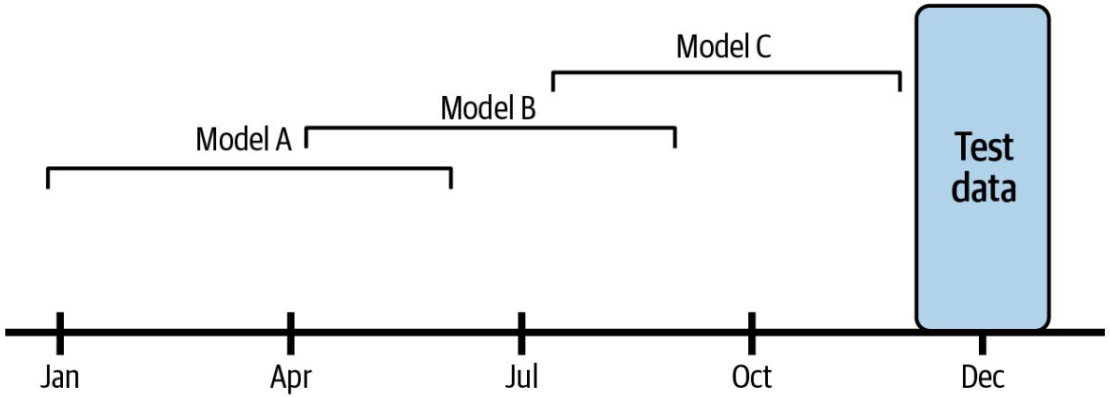


Practical Lessons from Predicting Clicks on Ads at Facebook (2014)

Quantify the value of data freshness

How much model's performance changes if switch from retraining monthly to weekly to daily to hourly?

- FB: CTR loss can be reduced ~1% going from training weekly to daily
- To get a sense of the performance gain you can get from fresher data, train your model on data from different time windows in the past and test on data from today to see how the performance changes

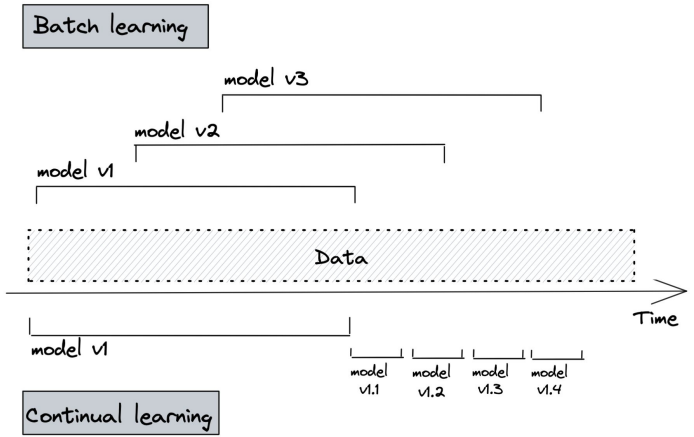


Model iteration vs data iteration

- you might wonder not only how often to update your model, but also what kind of model updates to perform.
- if you find that iterating on your data doesn’t give you much performance gain, then you should spend your resources on finding a better model.
- maybe in the near future, we’ll get more theoretical understanding to know in what situation an approach will work better.

Quantify cloud bill savings

- Train model incrementally each day on fresh data
- Faster convergence → less compute needed



GRUBHUB™ Going from monthly training to daily training gives

45x cost savings and **+20% metrics increase**

Online Learning for Recommendations at Grubhub (Alex Egg, 2021)

Machine Learning Systems Design

Deployment and Monitoring

Next Lecture: Model Online Evaluation

CE 40959 Spring 2023
Ali Zarezade
SharifMLSD.github.io