Setting up Machine Learning Operations for an Organization

Version 0.1 (work in progress)
Aapo Kyrola / Missing Exponent Oy, September 08 2022

This is a checklist of things to consider when building up Machine Learning operations for production purposes in a company. All items are not relevant for every case, and often simple solutions are sufficient. However, it is good to go through this list to ensure you are prepared for the complexity of running ML in the real world.

1. Developing new applications
    a. **Evaluation criteria.** How to determine if a model is good? Collect a hard evaluation dataset, or even better, a randomized train/test/validation data split to abstain statistical distributions for model performance. Beware of accuracy as a metric as it is biased on label distribution, instead use false pos/false neg, PR curves to evaluate. Use multiple metrics to evaluate models to avoid over optimizing against a particular metric.
        i. **Cost of wrong predictions:** weight properly the cost of wrong predictions. In some applications, wrong predictions can be very serious and default fallbacks are better.
    b. **Simple baseline:** develop one or more as-simple-as-possible baselines and run evaluation on them. Always compare to the baselines. Example baselines: random, simple models such as logistic regression and also heuristics if feasible. Can you beat the heuristic?.
    c. **Automate training & evaluation pipeline:** it should be completely automatic to train & evaluate a model in a consistent way. Any manual step can incur mistakes or bias into the process.
    d. **Model understanding:** use tools to analyze feature importance, understand model mistakes, etc. It is often very beneficial to study individual mistake cases.
    e. **Modeling frameworks:** what libraries can one use to develop models? How can one deploy these models into production? Offline/online deployment consideration.
    f. **Hyperparameter sweeps:** it should be easy to automatically sweep model parameters (when applicable). For more advanced use, architecture searches.
    g. **Versioning & Reproducibility:** automatically track versions of models using some version control system or model store.
    h. **Collaboration:** how colleagues can review each others work, reuse and reproduce.
    i. **Feature store:** separate features from models to facilitate reuse of features. Features must be computed exactly the same way in serving and training. Beware of “future leaks”.

    j. **Fairness / bias considerations:** for many applications, it may even be illegal to discriminate based on protected characteristics such as ethnicity or gender. Have a robust process to avoid this data being used and automatic way to analyze biases of trained models. Protected characteristics easily leak via other features.
    k. **Model selection:** choose the simplest model that can achieve sufficiently close to the best evaluation criteria.
    l. **Ownership:** who trains models? How is the final call made to publish these models, validate the data used etc.?

    m. (**Data governance:** retention, restricted access etc.)
2. Deploying new models (online or offline?)
    a. **A/B tests** to validate offline metrics and measure impact on production. It is not uncommon to have a model that outperforms others offline but perform poorly when deployed due to different data distribution or poor evaluation criteria.
    b. **Fallback:** if a model fails to compute a prediction, what is the default action?
    c. **Other metrics:** in addition to business metrics, monitor model latency, memory usage, and such.
    d. **Model versioning:** duplicate to above.
    e. **Ownership:** who deploys the models? Can it be automated as much as possible?)

3. Maintaining applications in production
    a. **Data / feature drift.** World changes over time; patterns of usage of a service change; seasonality; trends: models need to be monitored continuously and sometimes recalibrated or retrained to match new data distribution.
    b. **Operational monitoring.** Software updates, bugs, may cause models to misbehave or go out-of-service. Need to be monitored similarly to any other service components. **Logging, real-time analytics, dashboards and alerts.**
    c. **Legal issues for data retention:** can data be used more than 30 days, 90 days.
    d. **Ownership:** who maintains the models?
4. Sunsetting applications
    a. **Garbage collection:** remove models that are not used in production or perform poorly and don't have resources to maintain. It can be risky to run poor or deteriorating models in production.
    b.