Setu Madhavi Namburu

MSDS 480 (Fall 2019)

Michelangelo: Uber's Machine Learning Platform

Summary:

Uber is increasingly investing in artificial intelligence (AI) and machine learning (ML) platform called Michelangelo, an internal ML-as-a-service platform that democratizes machines learning and makes scaling AI to meet the needs of business as easy as requesting a ride. It is designed to cover the end-to-end ML workflow: manage data, train, evaluate, and deploy models, make predictions, and monitor predictions. It also supports traditional ML models, time series forecasting and deep learning. It has become the de-facto system for machine learning for Uber engineers and data scientists and has been deployed across several Uber data centers.

Motivation behind Michelangelo:

- There were number of challenges related to size and scale of operations to build and deploy machine learning models.
- -There were no systems in place to build reliable, uniform, and reproducible pipelines for creating and managing training and prediction data at scale.
- -There was neither a standard place to store the results of training experiments nor an easy way to compare one experiment to another.
- There was no established path to deploying a model into production–in most cases, the relevant engineering team had to create a custom serving container specific to the project at hand.

Use case: UberEATS estimated time of delivery model:

Predicting meal estimated time of delivery (ETD) is not simple. UberEATS has several models running on Michelangelo, covering meal delivery time predictions, search rankings, search autocomplete, and restaurant rankings. The delivery time models predict how much time a meal will take to prepare and deliver before the order is issued and then again at each stage of the delivery process, this is a complex modeling problem. UberEats data scientists use gradient boosted decision tree regression models to predict this end-

to-end delivery time. These predictions are displayed to UberEATS customers prior to ordering from a restaurant and as their meal is being prepared and delivered.

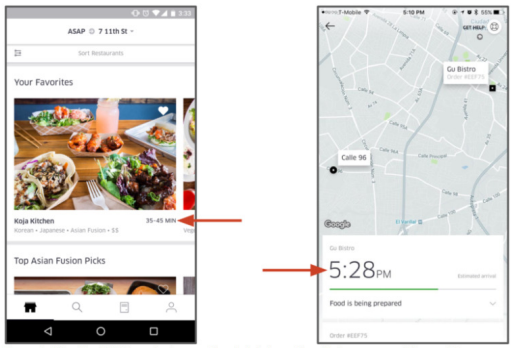

Figure 1: The UberEATS app hosts an estimated delivery time feature powered by machine learning models built on Michelangelo.

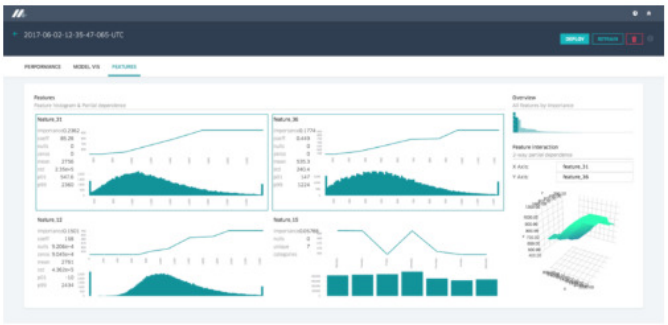

Figure 7: Features, their impact on the model, and their interactions can be explored though a feature report.

System architecture and Machine learning workflow:

Michelangelo consists of a mix of open source systems and components built in-house. The primary open sourced components used are: HDFS, Spark, Samza, Cassandra, MLLib, XGBoost, and TensorFlow. Michelangelo is built on top of Uber's data and compute infrastructure by taking utmost care of the data and infrastructure as described in 2nd article, providing a data lake that stores all of Uber's transactional and logged data, Kafka brokers that aggregate logged messages from all Uber's services, a Samza streaming compute engine, managed Cassandra clusters, and Uber's in-house service provisioning and deployment tools.

Michelangelo specifically designed to provide scalable, reliable, reproducible, easy-to-use, and automated tools to address the following six-step workflow:

Manage data

Train models

Evaluate models

Deploy models

Make predictions

Monitor predictions

Offline, online modules with batch precompute and near real time compute are used to provide up to date predictions to the customers.

One impressive part of this infrastructure is building a centralized feature store in which teams around Uber can create and manage canonical features to be used by their teams and shared with others. At present, there are 10,000 features in feature store that are used to accelerate machine learning projects. The features may need to be reengineered by different teams for their domain specific needs, so DSL (domain specific language) is created for the modelers to select, transform and combine features based on their needs. DSL is implemented as sub-set of Scala by facilitating complete set of commonly used functions and user defined functions. Scalable visualization tools are used to evaluate and monitor performance of the tools through dashboards.

Discussion:

The platforms and the use case presented are compelling and justifying for scalable/reliable data and modeling platforms. Uber being data rich company, the platforms presented in the references demonstrate how these companies take data very seriously and manage and take care of it at every step of business life cycle. The evolution of these platforms to address outgrowing challenges illustrate how they adapt to change and bring innovation to problem solving. While statistician may ask, why do we need so much data to make these decisions, won't sampling do the job? It boils down to list of exhaustive scenarios and time in hand to make real time decisions and the value it can generate. I'm convinced with the way they have these platforms set up where so much transparency is maintained from data to feature engineering to model building to evaluation through visualizations and continuous monitoring. Only one thing that bothers me is, with ever growing no looking back attitude, it may be losing track of fundamental/foundational problems (e.g., the 2nd article is great example where with growing scale they kept adjusting the infrastructure and at the end went back to basics of ETL by accommodating inserts and updates and data modeling to address fundamental needs). So while adjusting to change/demand at exponential growth is very important, keeping track of fundamental/originality of the problems is equally important to avoid unnecessary complexity or reinventing the wheels yet times.

References

1. Meet Michelangelo: Uber's Machine Learning Platform: https://eng.uber.com/michelangelo/

2. Uber's Big Data Platform: 100+ Petabytes with Minute Latency: https://eng.uber.com/uber-big-data-platform/