1. <u>Semana 1</u>

- Model serving é utilizar um servidor para fornecer dados para um modelo, isso pode ser online (com o mdelo sendo atualizado a quase todo instante) ou offline, deve se considerar coisa como delay entre ações do usuário, latência, throughput, etc. Ainda mais em casos como carros autonomos ou recomendações em sites, latência é bastante importante, deve-se prestar atenção no maximizar throughput x minimizar latência
- A infraestrutura deve ter um balanço entre o custo e a velocidade, levando em consideração coisas como acuracia, mas sem esquecer de latência e coisas o tipo



throughput



TPUs for complex models and large batches



Hardware choices impact cost



Balancing complexity and hardware choices



Choices made at organizational level

- Há diversas formas de se dar deploy no modelo, seja em centros de dados ou no próprio dispositivo do usuário (deve-se considerar coisas como tamanho, uso de cpu, etc, usuários não vao intalar algo pesado), existem diversas ferramentas para lidar com isso, como clipper, tensorflow serving. Essas ferramentas permitem uma série de ganhos, como:

Advantages of Serving with a Managed Service



Realtime endpoint for low-latency predictions on massive batches



Deployment of models trained on premises or on the Google Cloud Platform



Scale automatically based on traffic



Use GPU/TPU for faster predictions

2. <u>Semana 2</u>

ML infraestructure:

ML Infrastructure

On Prem



- Train and deploy on your own hardware infrastructure
- Manually procure hardware GPUs, CPUs etc
- Profitable for large companies running ML projects for longer time

On Cloud



- Train and deploy on cloud choosing from several service providers
 - Amazon Web Services, Google Cloud Platform, Microsoft Azure, etc
- Alguns servidores bastante utilizados>

TensorFlow Serving



TensorFlow

Supports many servables

TF Models

Non TF Models

Word Embeddings

Vocabularies

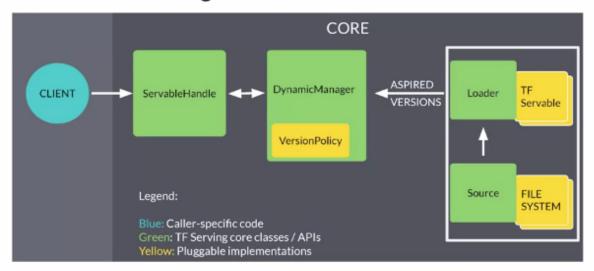
Feature Transformations

Out of the box integration with TensorFlow Models

Batch and Real-time Inference Multi-Model Serving

Exposes gRPC and REST endpoints

TensorFlow Serving Architecture



NVIDIA Triton Inference Server

- Simplifies deployment of AI models at scale in production.
- Open source inference serving software
 - Deploy trained models from any framework: TRITON INFERENCE SERVER
 - TensorFlow, TensorRT, PyTorch, ONNX Runtime, or a custom framework
- Models can be stored on:
 - Local storage, AWS S3, GCP, Any CPU-GPU Architecture (cloud, data centre or edge)

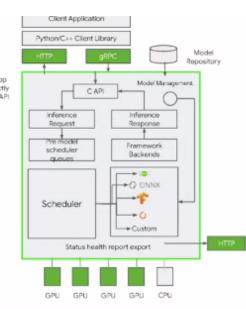
HTTP REST or gRPC endpoints are supported.

Architecture

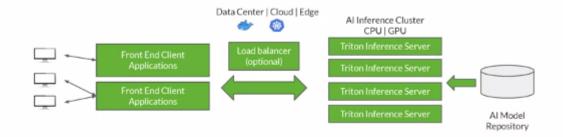
Triton Inference Server Architecture supports: Gand directly Bink to CAPI

- Single GPU for multiple models from same or different frameworks
- Multi-GPU for same model
 - Can run instances of model on multiple GPUs for increased inference performance.

Supports model ensembles.



Designed for Scalability

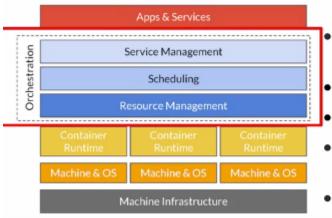


Can integrate with KubeFlow pipelines for end to end Al workflow

- Escalabilidade:

-

² Enter Container Orchestration

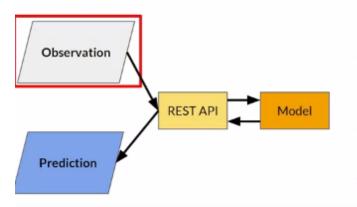


Manages life cycle of containers in production environments

- Manages scaling of containers
 - Ensures reliability of containers
- Distributes resources between containers.
- Monitors health of containers
- Kubernets: um sistema de código aberto para automação, implantação, dimensionamento e gerenciamento de aplicativos em contêineres. Os contêineres de um grupo que compõem um aplicativo em unidades lógicas para fácil gerenciamento e descoberta.
- Kuberflow:
 - Dedicated to making deployments of machine learning (ML) workflows on Kubernetes simple, portable and scalable.
- Kubeflow
- Anywhere you are running Kubernetes, you should be able to run Kubeflow.
- Can be run on premise or on Kubernetes engine on cloud offerings AWS, GCP, Azure etc.,
- Online inference: processo de gerar predições em tempo real dado uma requisição, predições são geradas em uma unica observação de dados no

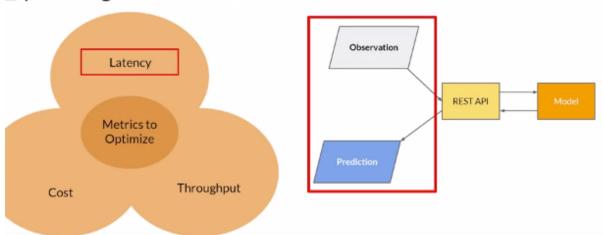
runtime, são gerados em qualquer hora de demanda do dia

²Online Inference



- Process of generating machine learning predictions in real time upon request.
- Predictions are generated on a single observation of data at runtime.
- Can be generated at any time of the day on demand

Optimising ML Inference



 Antes de fazer uma inferência pelo modelo, geralmente se precisa pré processar esses dados e também pos processar as vezes

² Preprocessing Operations Needed Before Inference

Data Cleansing

 Correcting invalid values in incoming data

Feature Tuning

- Normalization
- Clipping outliers
- Imputing Missing Values

Feature Construction

- · Combine inputs
- Feature crossing
- Polynomial expansion

Representation Transformation

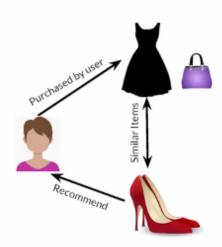
- · Change data format for the model
- · One-hot encoding
- Vectorization

Feature Selection

 Apply same feature selection done during training on incoming data and features fetched from cache

Batch inference gera predições a partir de um batch de predições, são geralmente recorrentes, reduz o custo do sistema, pode ter problemas em sistemas real time,

^{2.}Use Case - Product Recommendations

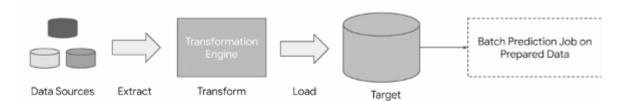


- E-commerce sites: new recommendations on a recurring schedule
- Cache these for easy retrieval
- Enables use of more predictors to train more complex models.
 - Helps personalization to a greater degree, but with delayed data

Data Processing - Batch and Streaming

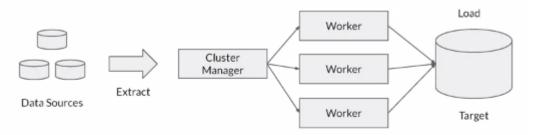
- Data can be of different types based on the source.
- Batch Data
 - Batch processing can be done on data available in huge volumes in data lakes, from csv files, log files etc.,
- Streaming Data
 - Real-time streaming data, like data from sensors.

ETL Pipelines



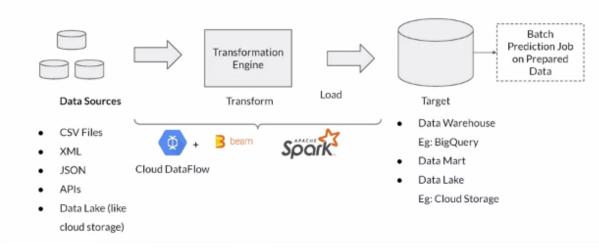
- Set of processes for
 - extracting data from data sources
 - Transforming data
 - Loading into an output destination like data warehouse
- From there data can be consumed for training or making predictions using ML models,
- ETL Extrack track load

Distributed Processing



- ETL can be performed huge volumes of data in distributed manner.
- Data is split into chunks and parallely processed by multiple workers.
- The results of the ETL workflow are stored in a database.
- Results in lower latency and higher throughput of data processing.

² ETL Pipeline components Batch Processing



3. Semana 3

- Experiment tracking: fazer o rastreio dos experimentos é algo muito importante, seja por consumo, seja por tempo, o importante é entender tudo que vai dentro de cada experimento, os hyperparametros, etc.
- notebooks são feramentas interessantes, entretanto não podem ser tão util para produção, existem diversas ferramentas que transformam / convertem um notebook para um .py, por exemplo
- Deve se pensar em um código modular, ainda mais em time, além de usar git.
- Usar config é interessante, além de command line, etc
- Além de controlar versionamento do modelo, também deve se controlar o versionamento do dados, como neptune, pachyderm, delta lake, git lfs (imagems, videos)
- Tensorboard é bem interessante para controlar os parâmetros e o treinamento do modelo conforme os experimentos acontecem.

-