The tutorial to build shared AI services

--Session 1

Suqiang Song (Jack)
Director & Chapter Leader of Data/AI Engineering @ Mastercard
jackssqcyy@gmail.com
https://www.linkedin.com/in/suqiang-song-72041716/

Agenda

Session 1: Jan. 30th Wed 10am-12pm PT

Module 1: Case study: AI as a Service (30 mins)

- A typical end to end AI Service
- Hidden truths of AI
- Options of AI as a Service
- The journey of AI as a Service
- Challenges of traditional machine learning
- How deep learning can improve
- Enterprise requirements for AI as a Service
- Deep learning approaches evaluation

Module 2: Keras on Spark (30 mins)

- Keras introduction
- Options of Keras on Spark
- Use case for user item propensity model
- Build a User Item Propensity model with deep learning algorithms
- Neural Collaborative Filtering deep learning algorithm

Code Lab 1 (45 mins)

- Build a Docker image and run a Keras on Spark container
- Run the NCF deep learning pipeline for User Item Propensity model

Q & A (15 mins)

Course Prerequisites

- Install Docker at your local laptop
- Download two Docker images from shared drive URL keras-py27-jupyter-cpu.tar and demo-whole.tar

  https://1drv.ms/f/s!AsXKHMXBWUIBiBpaYk9FFjdoUifg

  passcode : jack
- Load images to your Docker environment

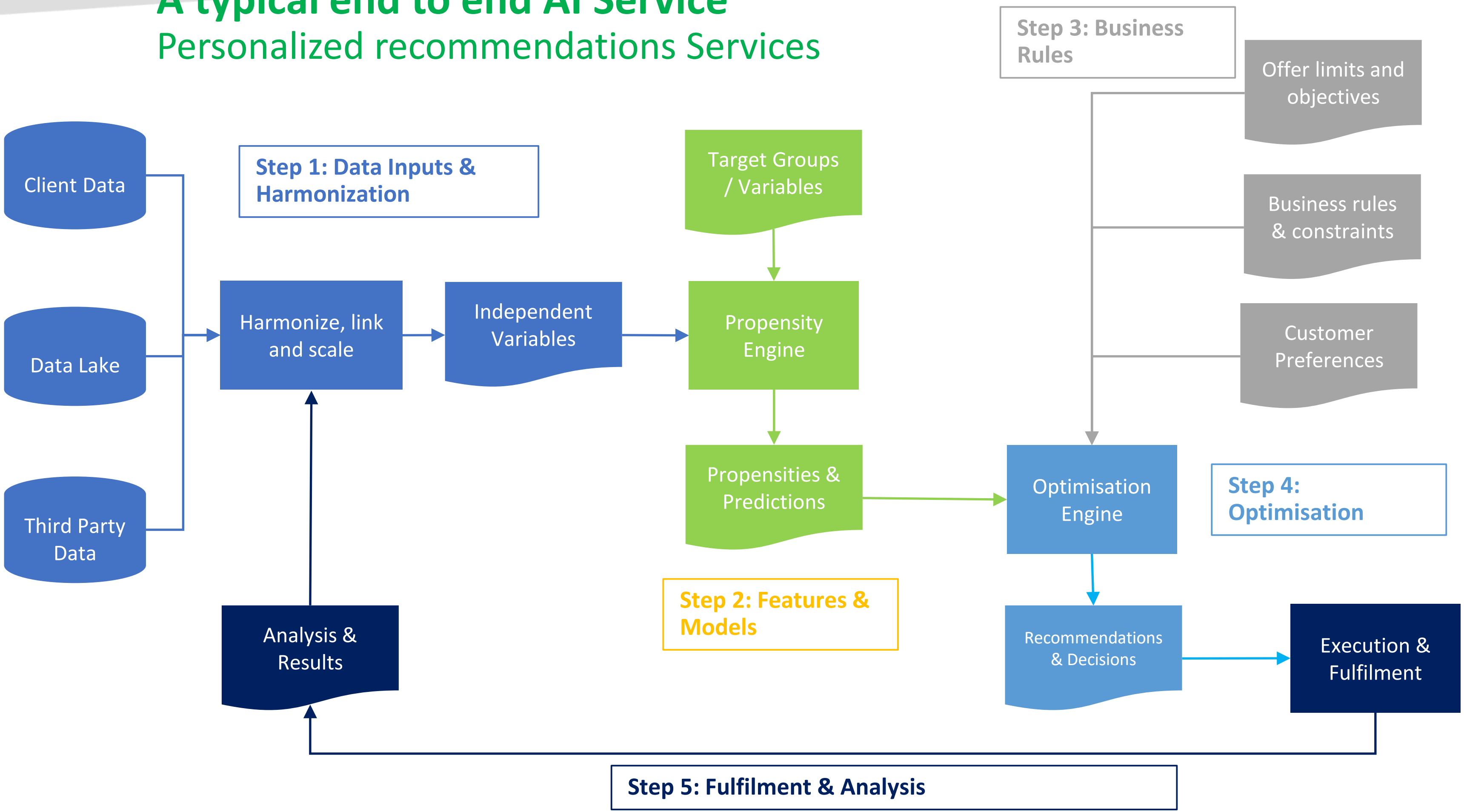
```
$ docker load -i keras-py27-jupyter-cpu.tar

$ docker load -i demo-whole.tar
```

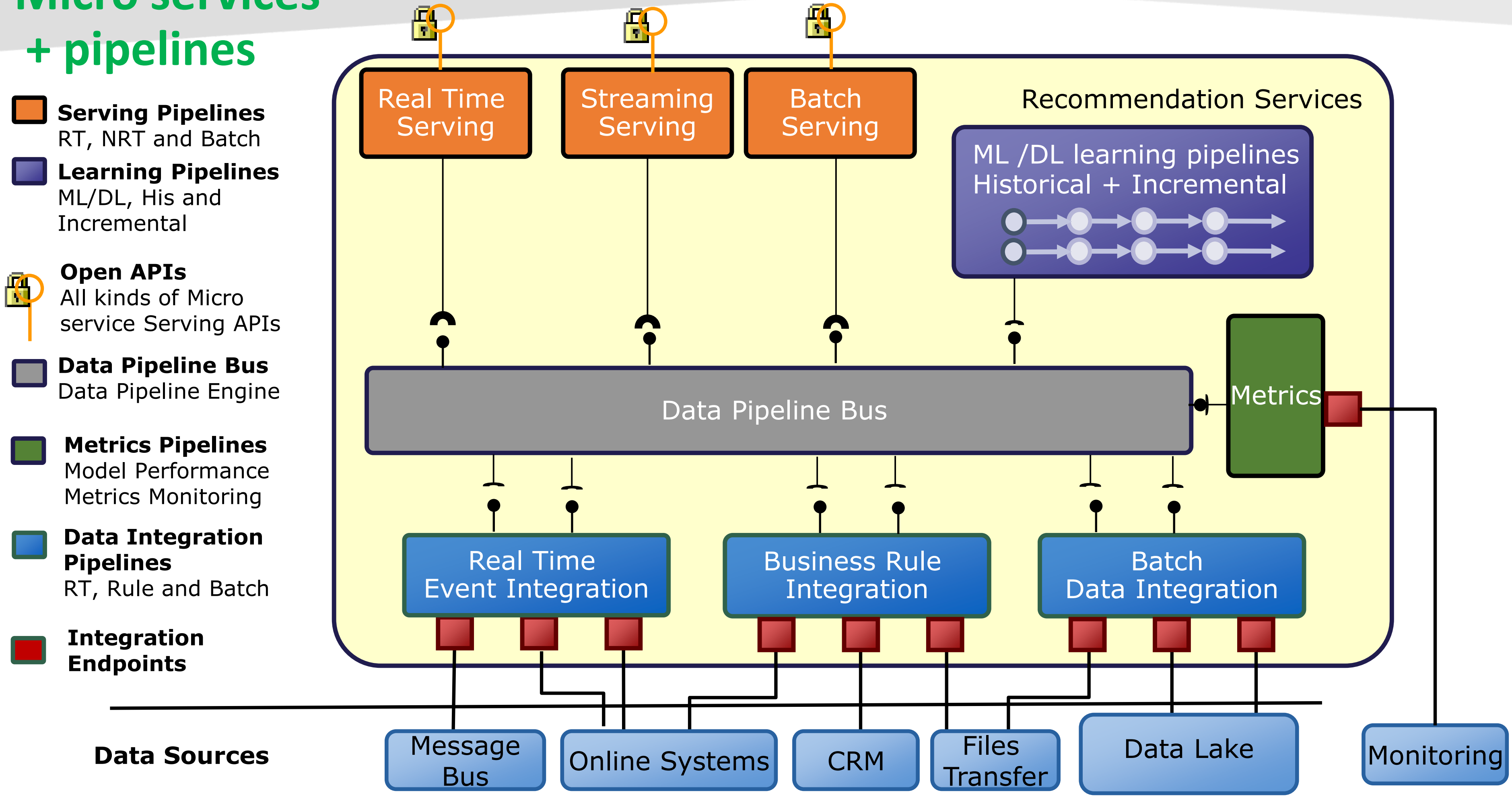
Module 1

A typical end to end AI Service

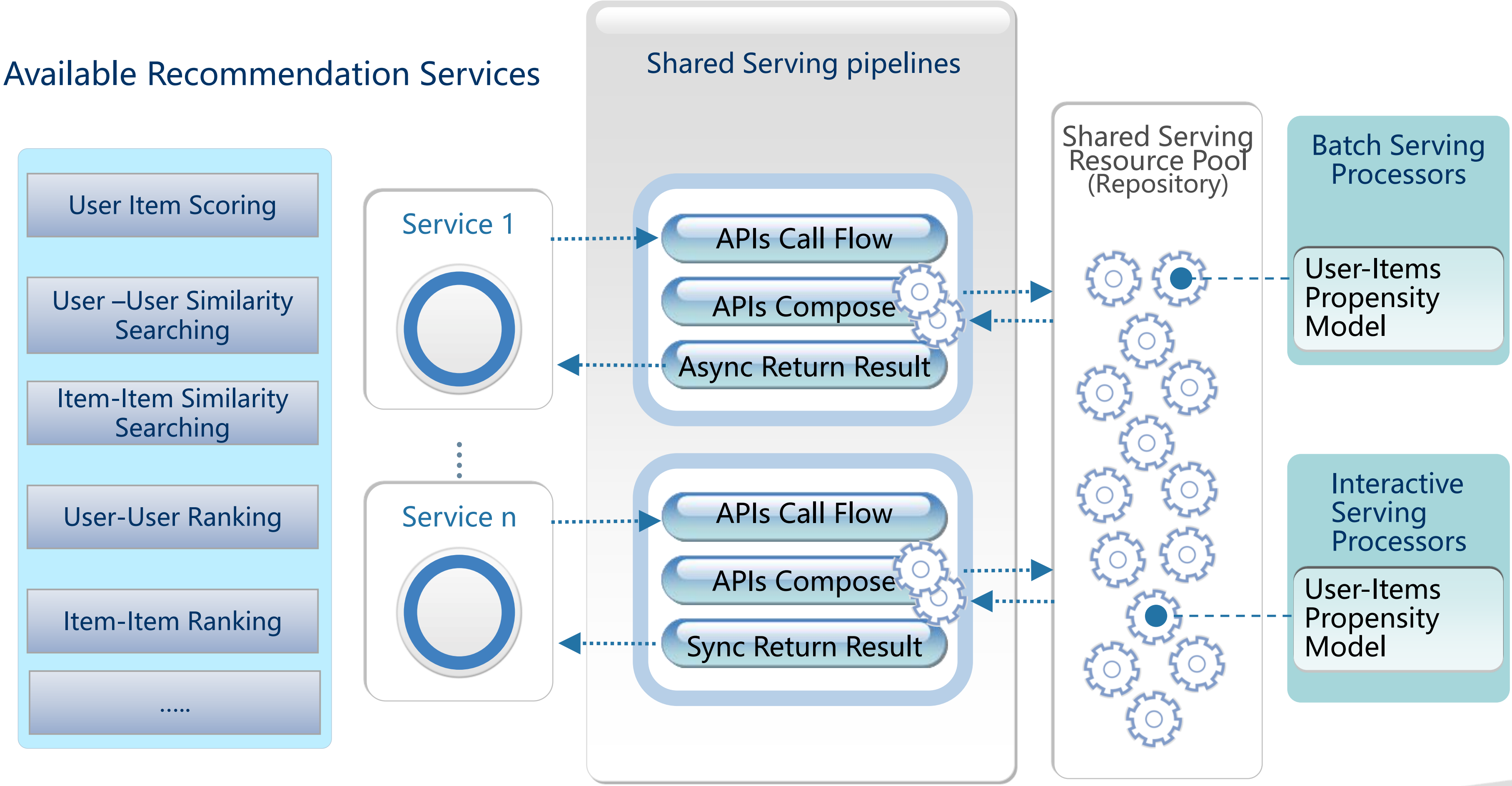
Personalized recommendations Services



Micro services + pipelines



Published Recommendation Services (Open APIs)



Hidden truths for AI

"Hidden Technical Debt in Machine Learning Systems," Google NIPS 2015

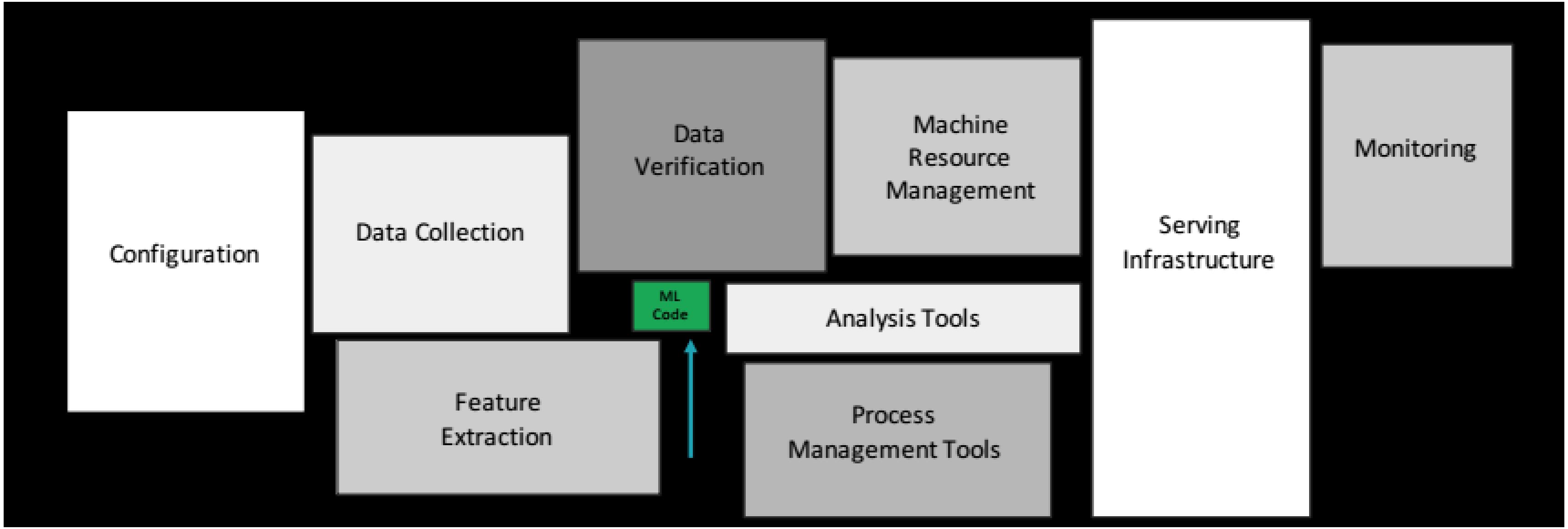
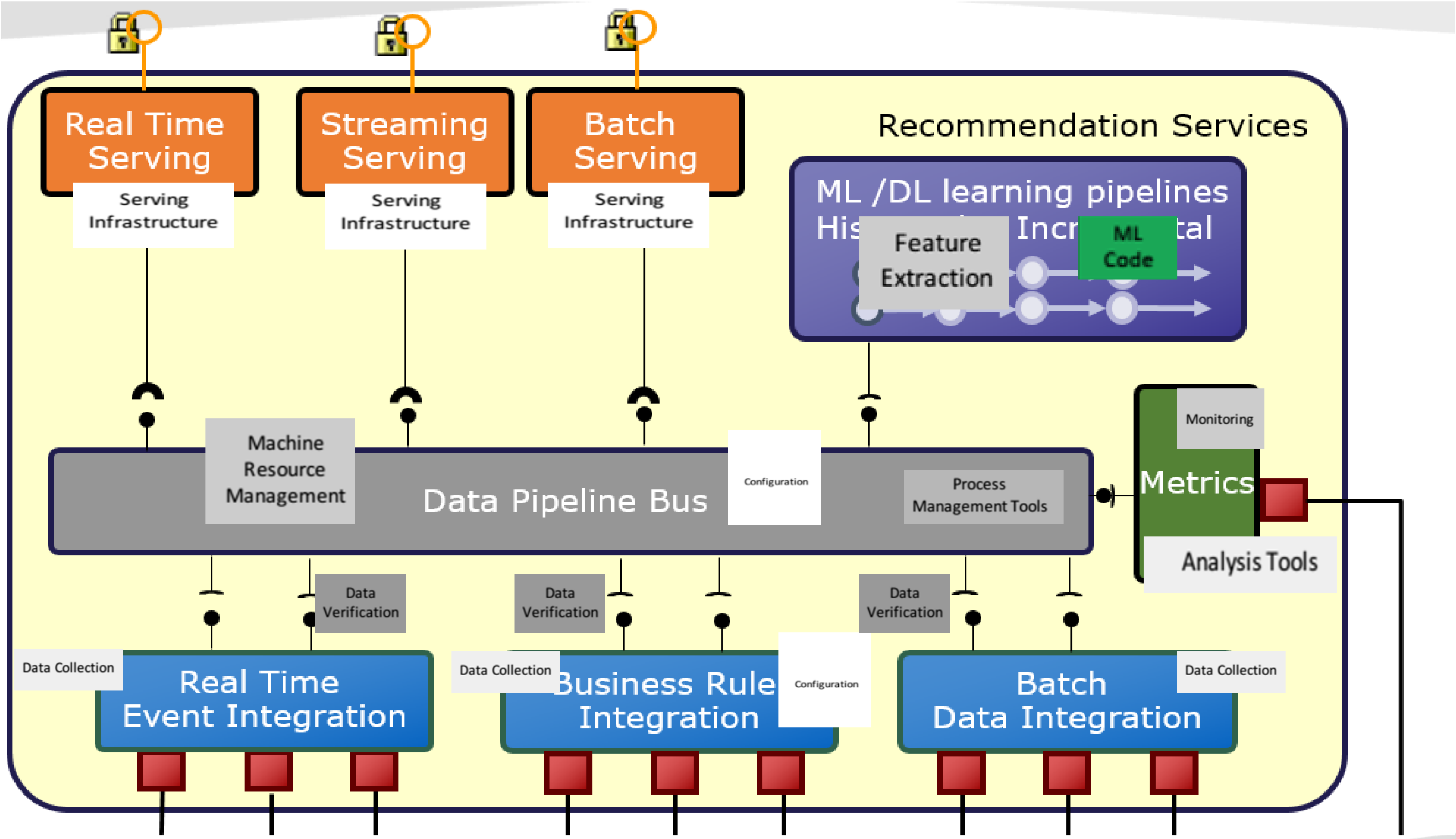


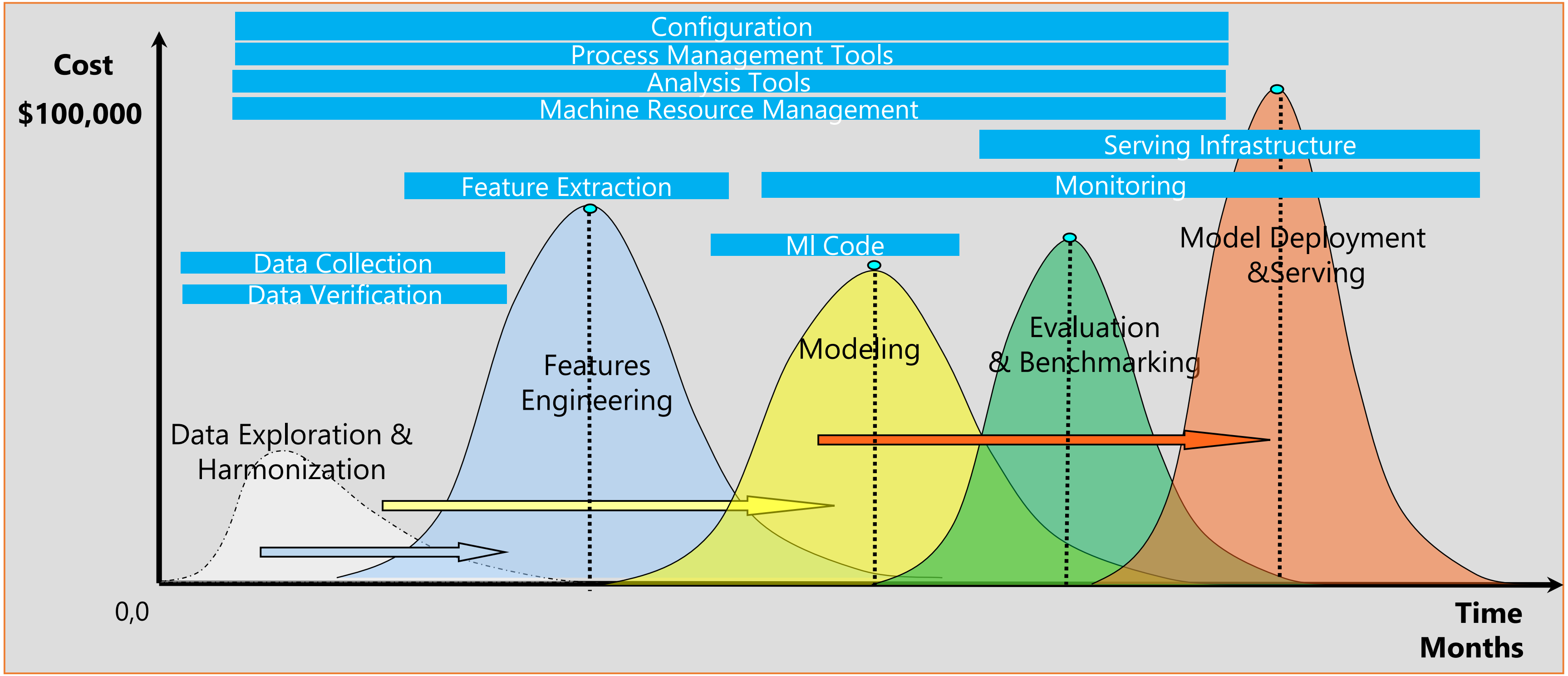
Figure 1: Only a small fraction of real-world ML systems is composed of the ML code. The required surrounding infrastructure is vast and complex. These include boundary erosion, entanglement, hidden feedback loops, undeclared consumers, data dependencies, configuration issues, changes in the external world, and a variety of system-level anti-patterns

Hidden truths for AI – Where the debts ?

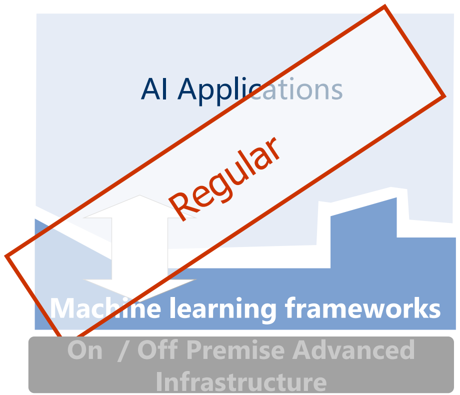


Hidden truths for AI -- continue

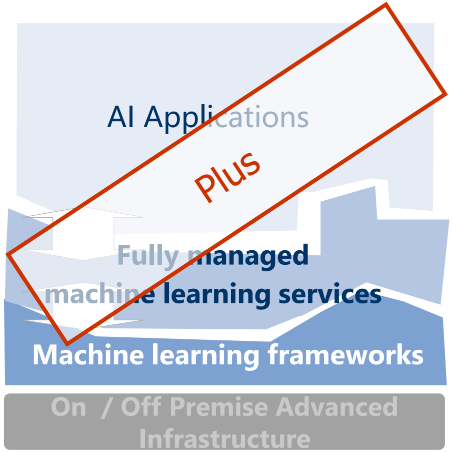
A Long and Expense Journey



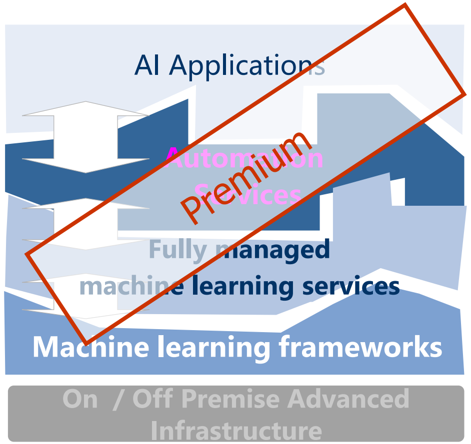
Options of AI as a Service



- Machine learning frameworks: Provide stable and secure environments and consolidate integrated wrappers on top of variable technologies for regular machine learning works
- Applications build silos from scratch



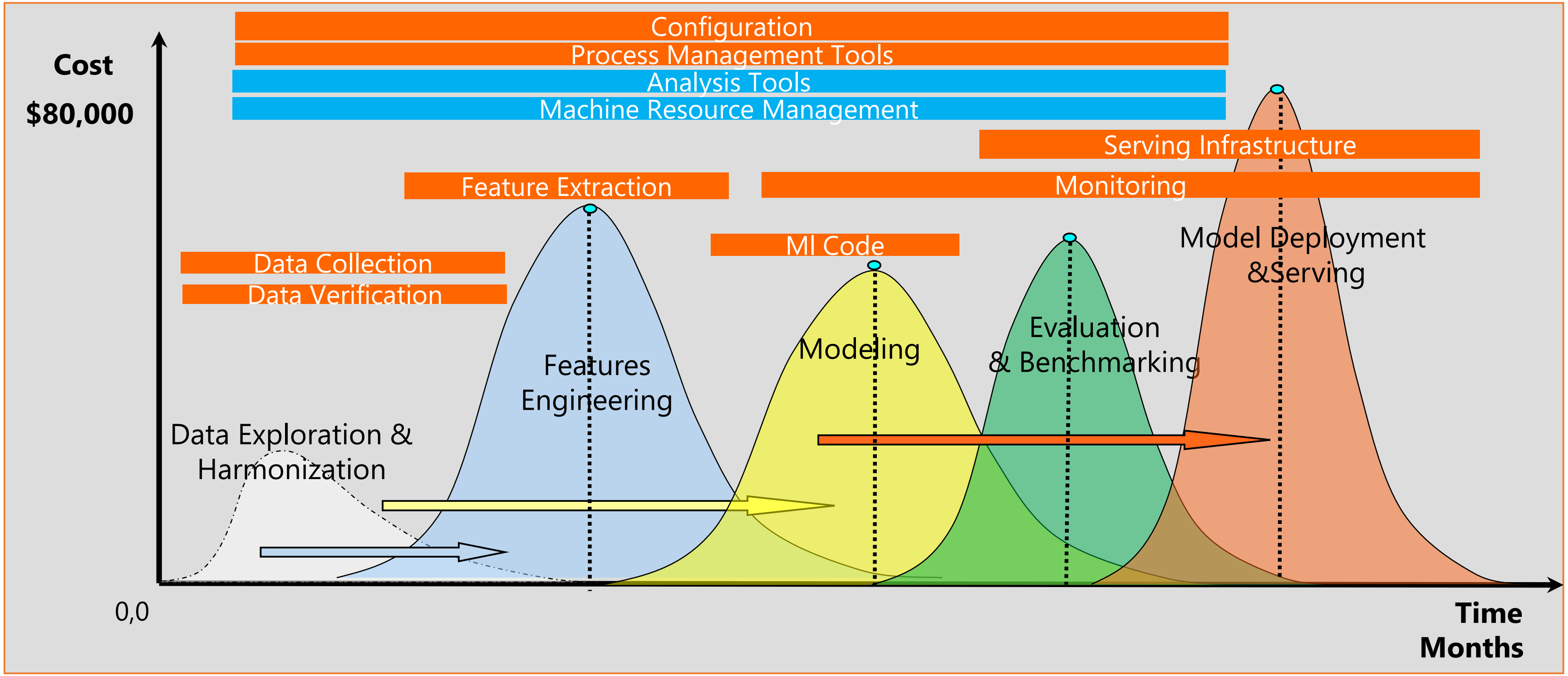
- Fully managed machine learning services use templates, pre-built models and drag-and-drop development tools to simplify and expedite the process of using a machine learning framework
- Applications share templates and pre-built models , assembly and infer them into pipelines or business context



- Automation Services, tasks like exploratory data analysis, pre-processing of data, hyper-parameter tuning, model selection and putting models into production can be automated
- "God's Return to God, Satan's Return to Satan , Math's Return AI, Business's Return Biz"

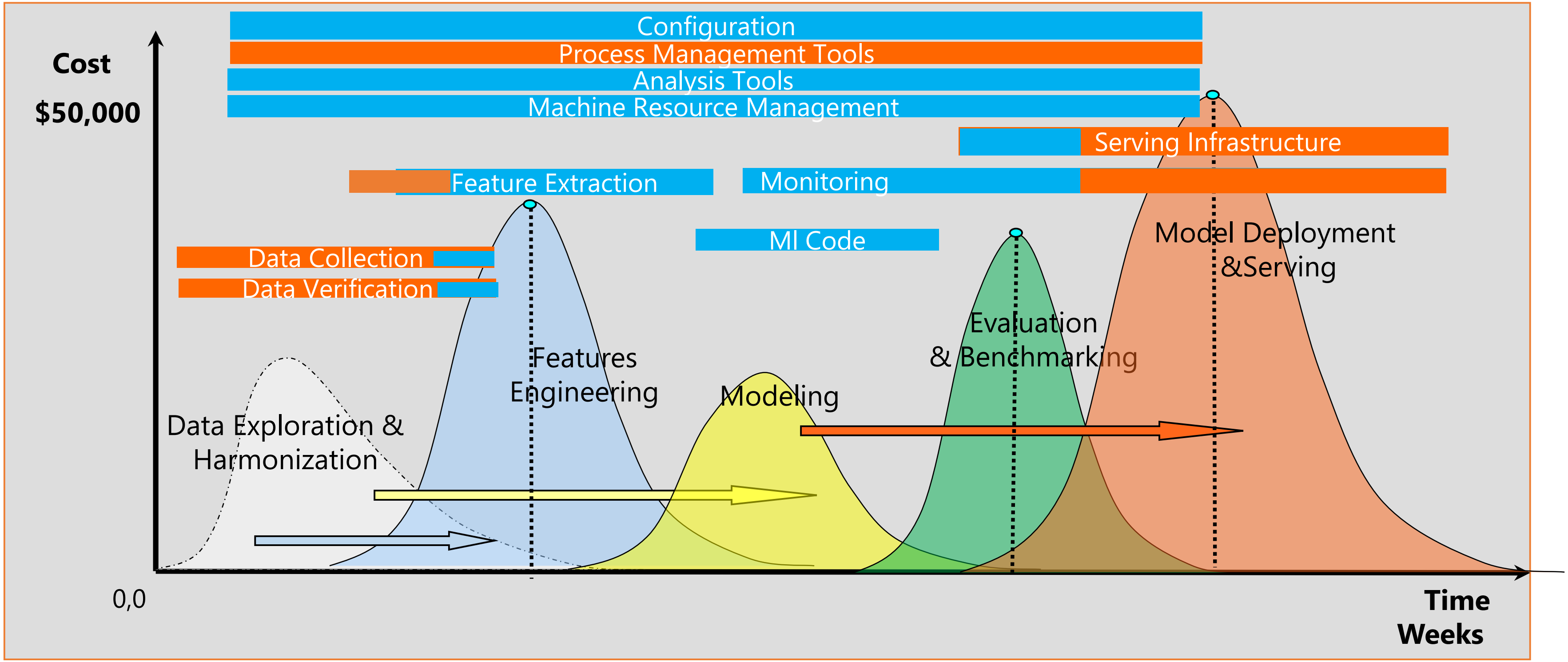
Journey 1:Machine learning frameworks

Example : Machine Learning Sandbox



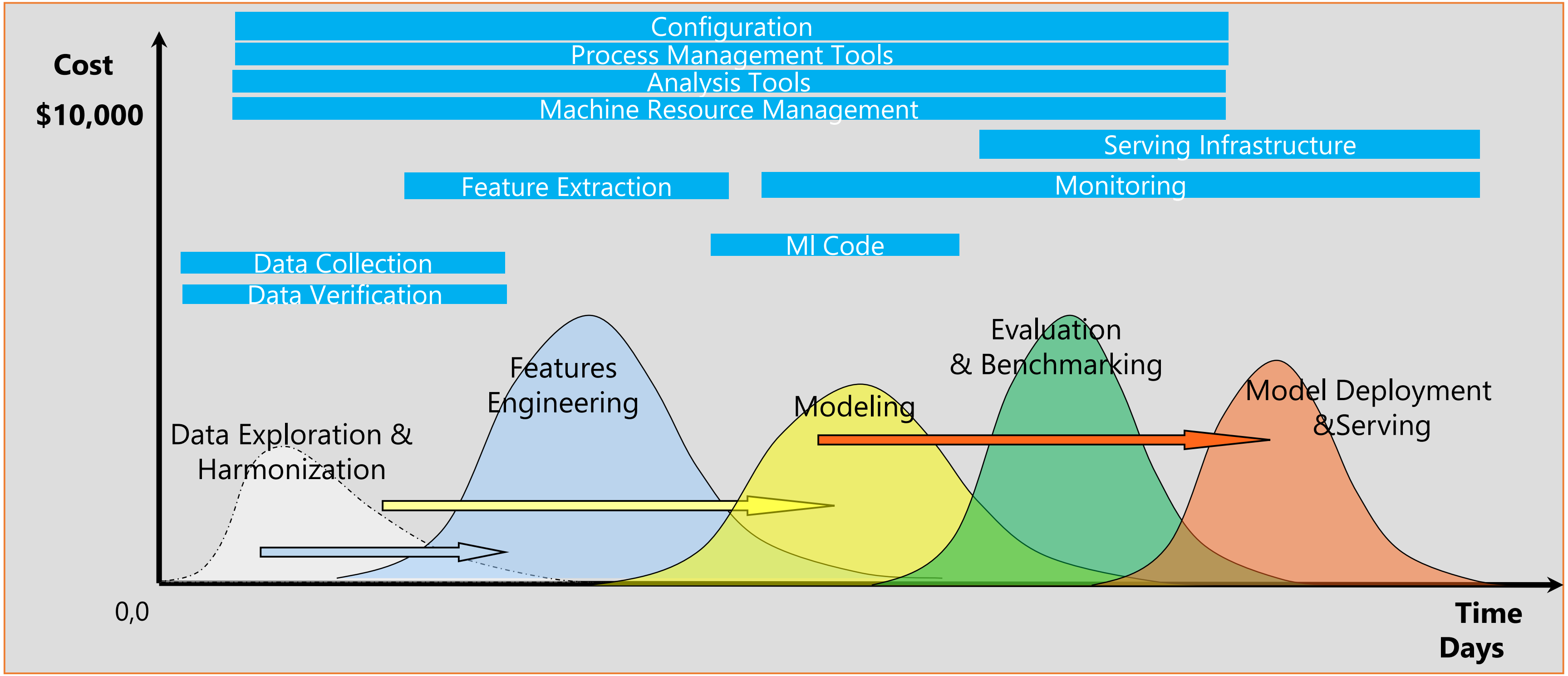
Journey 2: Fully managed machine learning services

Example : Data Science Workbench (Microsoft ML Studio)



Journey 3: Automation Services

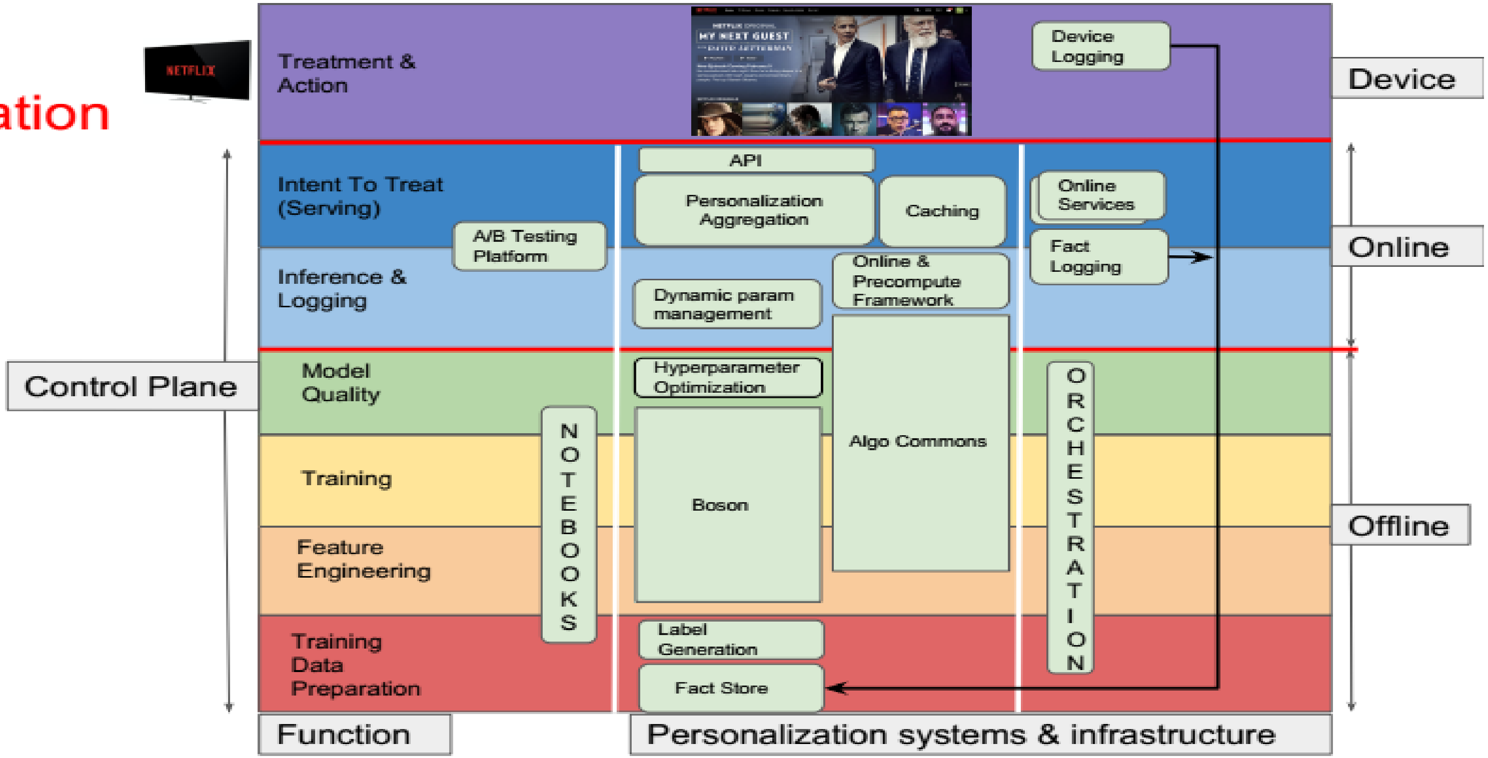
Example : Netflix , Amazon Sage Maker…



What we can learn from Netflix ?

https://www.slideshare.net/FaisalZakariaSiddiqi/ml-infra-for-netflix-recommendations-ai-nextcon-talk

The Personalization Rainbow



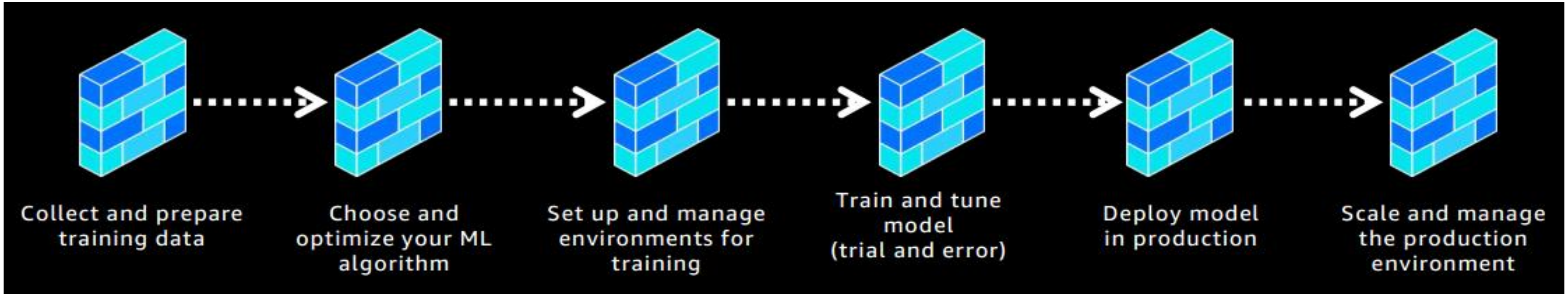
NETFLIX

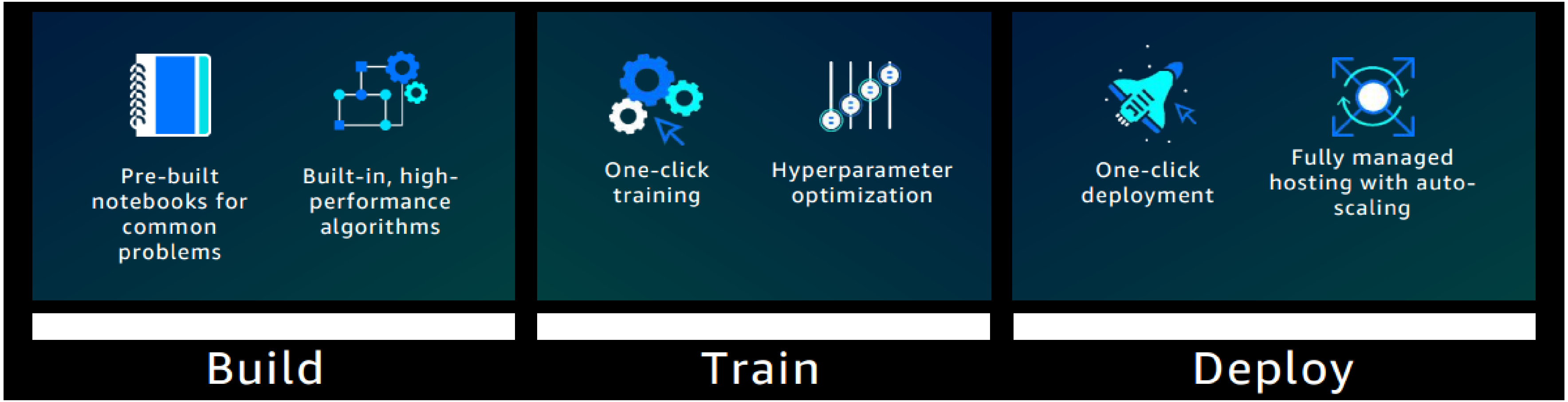
What we can learn from Amazon ?

Amazon SageMaker Components

Building	Training	Hosting
Amazon's fast, scalable algorithms		
Distributed Apache MXNet and TensorFlow		
Bring your own algorithm		
Hyperparameter optimization		

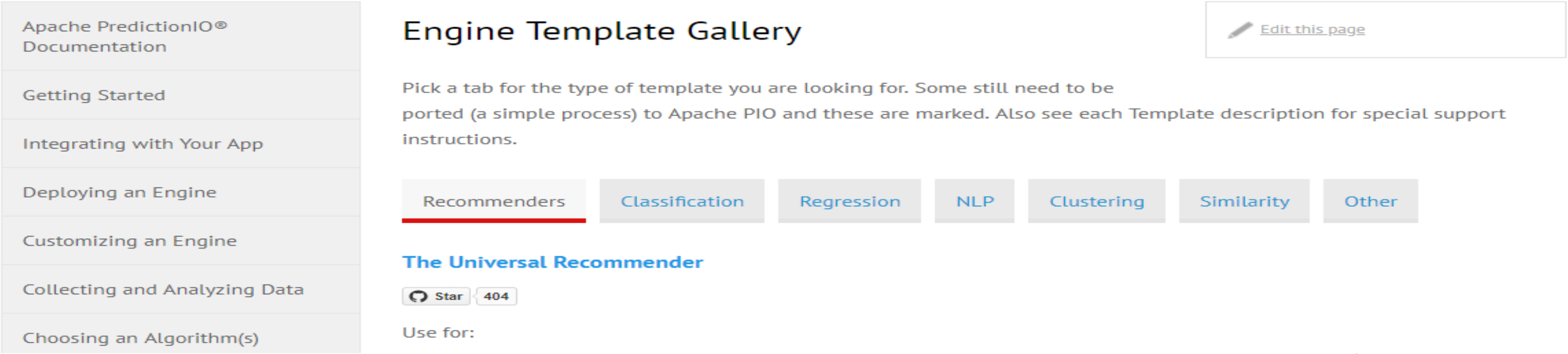
What we can learn from Amazon ? -- Continue

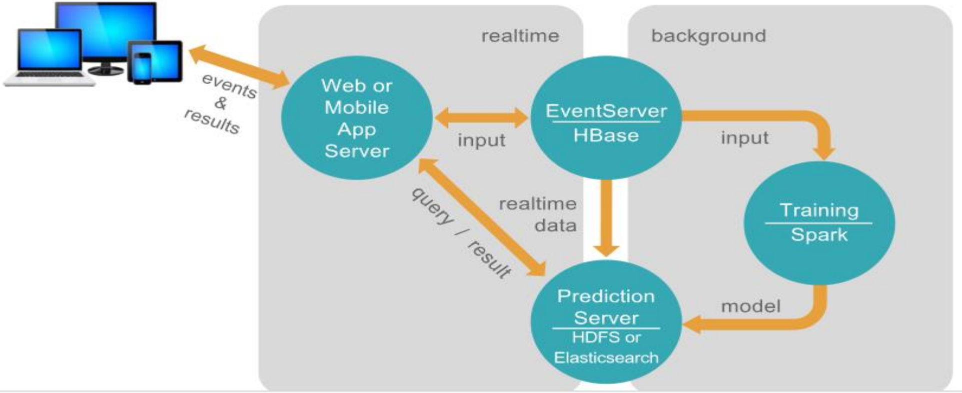




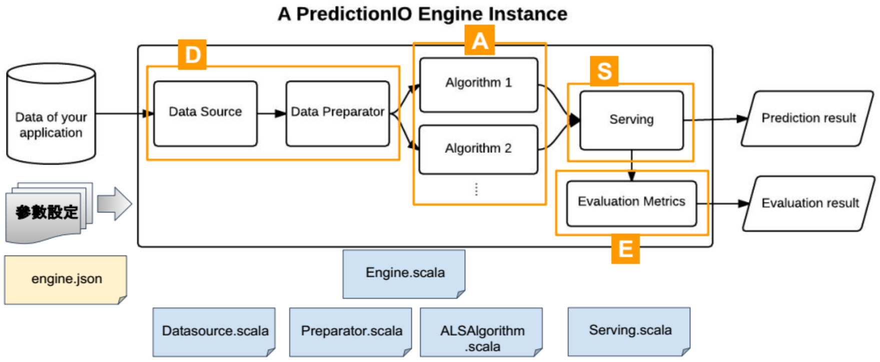
What we can learn from Prediction IO ?

http://predictionio.apache.org/





Customizing your Engine with D-A-S-E



Challenges of traditional machine learning

Learning

1 **Feature engineering bottlenecks**

Pre-calculate hundreds or thousands Long Term Variables take lots of resources and times (greater than 70%)

2 **Model scalability limitations**

Trade-off between automation in parallel and scaling machine learning to ever larger datasets and ever more complicated models

3 **Heavily relies on human machine learning experts**

Relies on human to perform the most of tasks, such as features selection, model selection, model hyper parameters tuning , critically analyze the results obtained...

Serving

4 **Less integration with end to end data pipelines and serving multiple contexts**

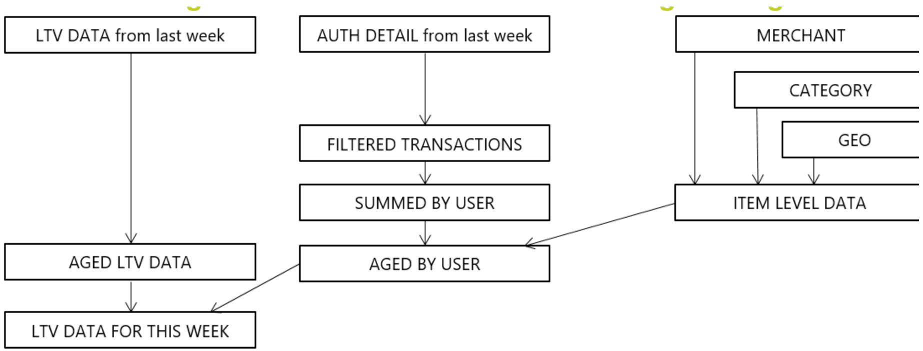
Gap to bring machine learning process into the existing enterprise data pipelines and application contexts including offline, streaming and real-time

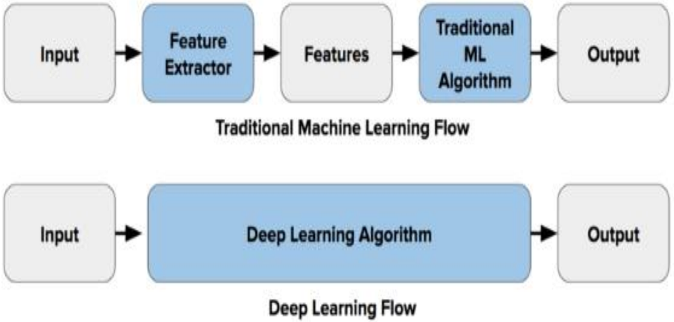
5 **Isolated non-reusable APIs and CI/CD pipelines**

Application teams spent lots of time with data scientist to build high level serving APIs , back and force. Most of time are using different technical stacks and non-reusable pipelines.

Isolated promotions and operation readiness without automate CI/CD

How deep learning can help -- Feature engineering bottlenecks





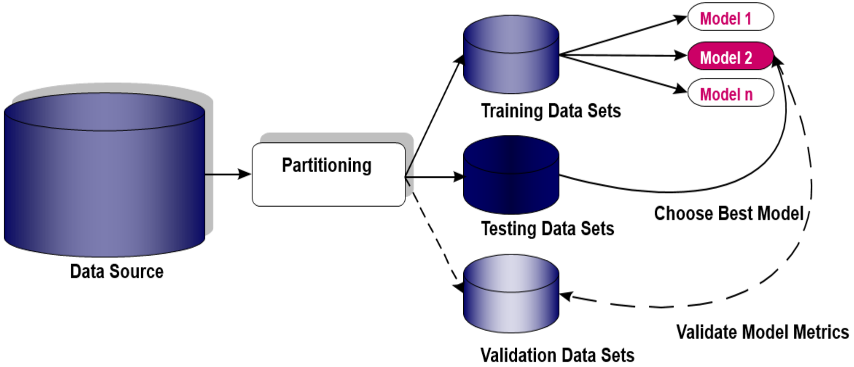
Bottlenecks

- Need to pre-calculate hundreds or thousands Long Term Variables for each user, such as total spends /visits for merchants list, category list divided by week, months and years
- The computation time for LTV features took > 70% of the data processing time for the whole lifecycle and occupied lots of resources which had huge impact to other critical workloads.
- Miss the feature selection optimizations which could save the data engineering efforts a lot

Improvements

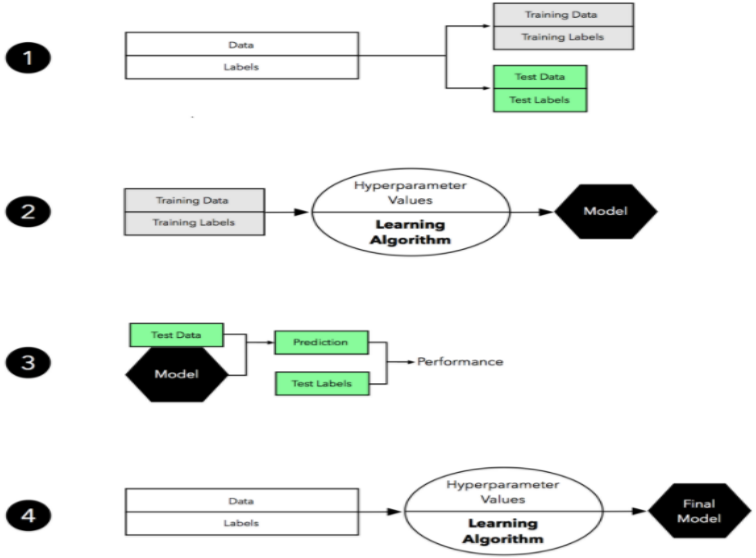
- *When build model , only focus on few pre-defined sliding features and custom overlap features (Users only need to identify the columns names from data source)*
- *Remove most of the LTV pre-calculations works, saved hours time and lots of resources*
- *Deep learning algorithm generates exponential growth of hidden embedding features ,do the internal features selections and optimization automatically when it does cross validation at training stage*

How deep learning can help -- Heavily relies on human machine learning experts



Relies on human to perform the following tasks:

- Select and construct appropriate features.
- Select an appropriate model family.
- Optimize model hyper parameters.
- Post process machine learning models.
- Critically analyze the results obtained.

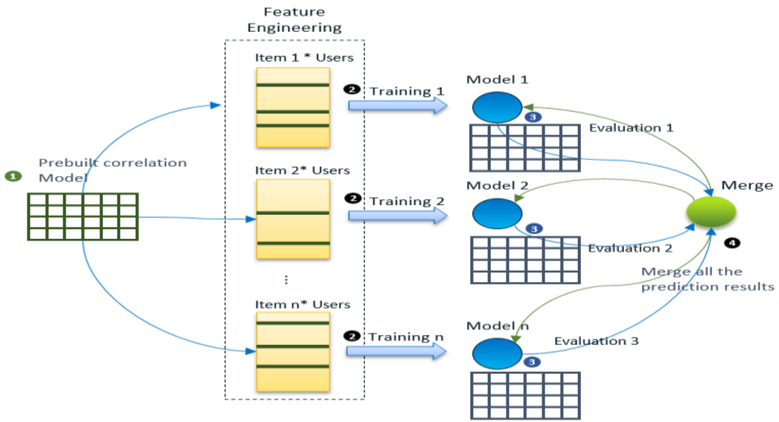


Improvements

- *Common neural network "tricks", including initialization, L2 and dropout regularization, Batch normalization, gradient checking*
- *A variety of optimization algorithms, such as mini-batch gradient descent, Momentum, RMSprop and Adam*
- *Provides optimization-as-a-service using an ensemble of optimization strategies, allowing practitioners to efficiently optimize models faster and cheaper than standard approaches.*

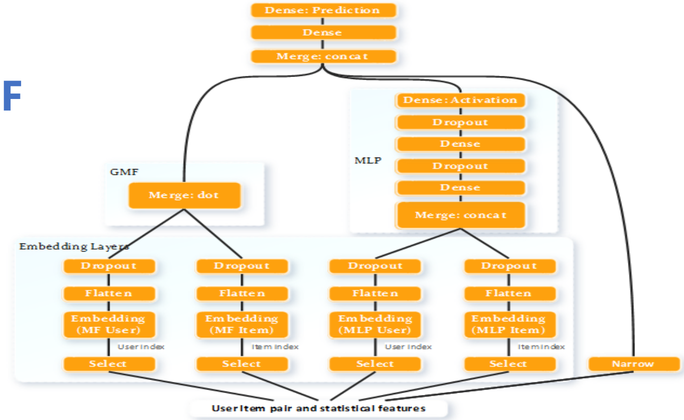
How deep learning can help -- Model scalability

LR

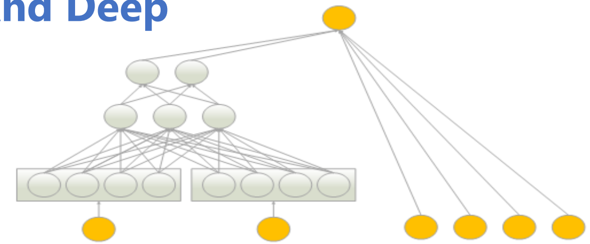


- All the pipelines separated by items and generate one model for each item
- Have to pre-calculate the correlation matrix between items
- Lots of redundant duplications and computations at feature engineering ,training and testing process
- Run items in parallel and occupied most of cluster resources when executed
- Bad metrics for items with few transactions
- It is very hard to scale more items , from hundreds to millions ?

NCF



Wide And Deep



Improvements

- *Scale models in deeper and wider without decreasing metrics*

Enterprise requirements for AI as a Service

Collocated with mass data storage

- Analyze a large amount of data on the same Big Data clusters where the data are stored (HDFS, HBase, Hive, etc.) rather than ***move or duplicate data***

Seamless integration with Products Internal & External

- Add deep learning capabilities to existing Analytic Applications and/or machine learning workflows rather than ***rebuild all of them***

Data governance with restricted Processing

- Follow data privacy, regulation and compliance (such as PCI/PII compliance and GDPR rather than ***operate data in unsecured zones***

Shared infrastructure with Multi-tenant isolated resources

- Leverage existing Big Data clusters and deep learning workloads should be managed and monitored with other workloads (ETL, data warehouse, traditional ML etc..) rather than ***run ML/DL workloads standalone in separate clusters***

Adopt structured sparse data sets

- More challenges also more benefits to adopt structured high dimensional sparse data sets rather than ***non-structured dense data sets***

Automation and easy to go

- End to end automation rather than ***manual efforts***
- Easy API enablement rather than ***big learning curve or depend on special experts***

Deep learning approaches evaluation -- Super Stars

- Examples are good for dense high sample size data sets (But won’t help us)
- Claimed that the GPU computing are better than CPU which requires new hardware infrastructure (very long timeline normally)
- Success requires many engineer-hours (Impossible to Install a Tensor Flow Cluster at STAGE …)
- Low level APIs with steep learning curve (Where is your PhD degree ?)
- Not well integrated with other enterprise tools and need data movements (couldn't leverage the existing ETL, data warehousing and other analytic relevant data pipelines, technologies and tool sets. And it is also a big challenge to make duplicate data pipelines and data copy to the capacity and performance.)
- Tedious and fragile to distribute computations (less monitoring)
- The concerns of Enterprise Maturity and InfoSec (such as use GPU cluster with Tensor Flow from Google Cloud)

…………..

Module 2

Keras introduction
-- Introduce a simple play ground , not just for Keras

https://github.com/ufoym/deepo

Deepo

build passing | docker pulls 49k | license MIT

Deepo is a series of *Docker* images that

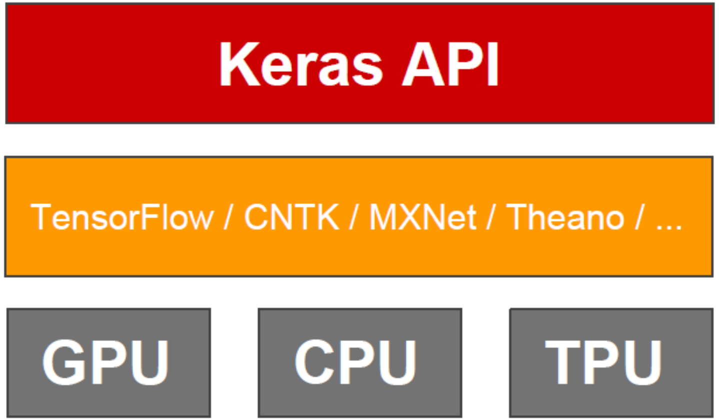
- allows you to quickly set up your deep learning research environment
- supports almost all commonly used deep learning frameworks
- supports GPU acceleration (CUDA and cuDNN included), also works in CPU-only mode
- works on Linux (CPU version/GPU version), Windows (CPU version) and OS X (CPU version)

and their Dockerfile generator that

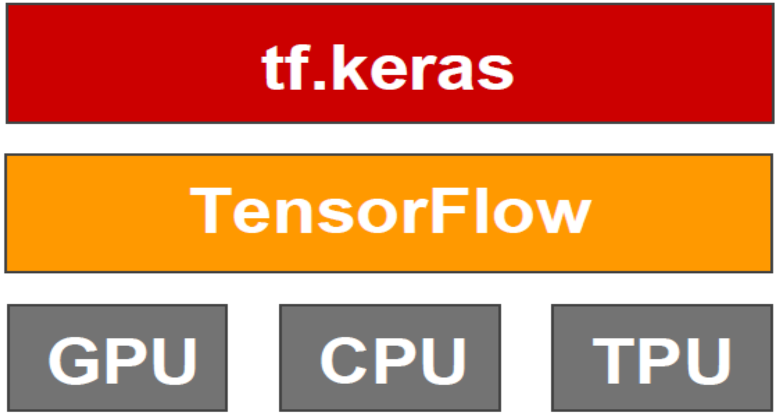
- allows you to customize your own environment with Lego-like modules
- automatically resolves the dependencies for you

Keras introduction
-- Basic concepts

Keras: an API for specifying & training differentiable programs



Keras is the official high-level API of TensorFlow



Keras introduction
-- Three API styles

The Sequential Model

- - Dead simple
- - Only for single-input, single-output, sequential layer stacks
- - Good for 70+% of use cases

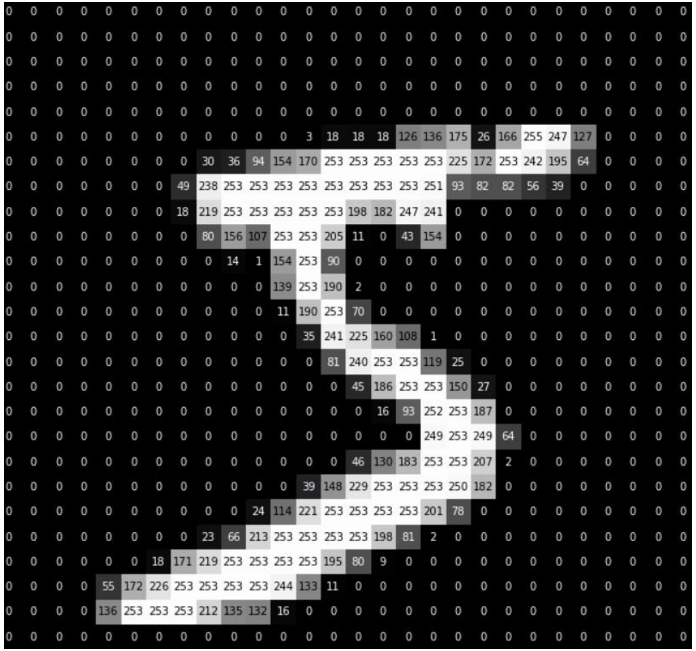
The functional API

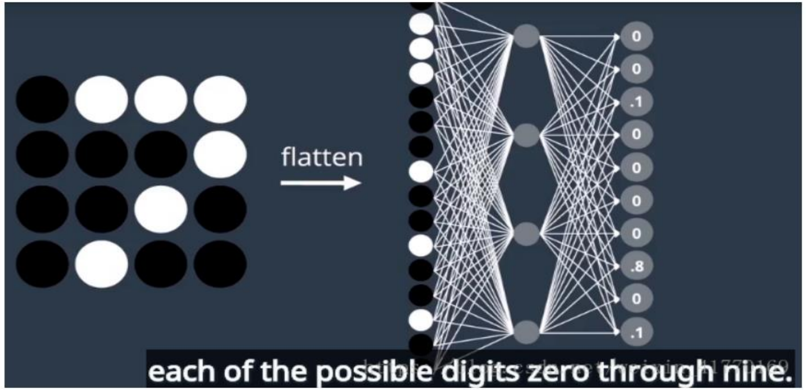
- - Like playing with Lego bricks
- - Multi-input, multi-output, arbitrary static graph topologies
- - Good for 95% of use cases

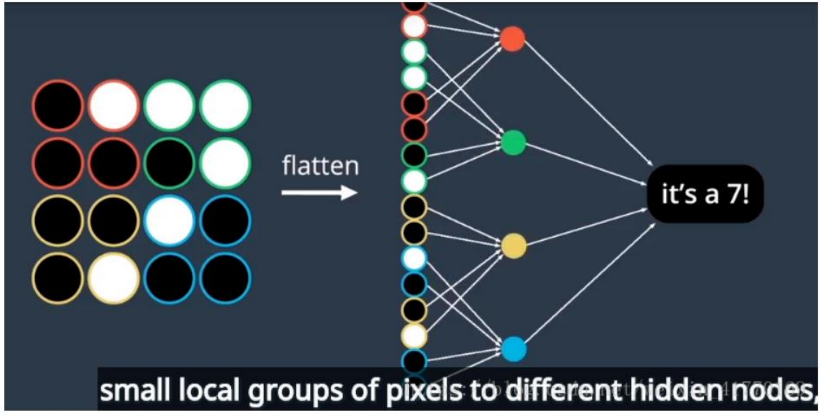
Model subclassing

- - Maximum flexibility
- - Larger potential error surface

Keras introduction
-- Example for entry : MINST by MLP and CNN



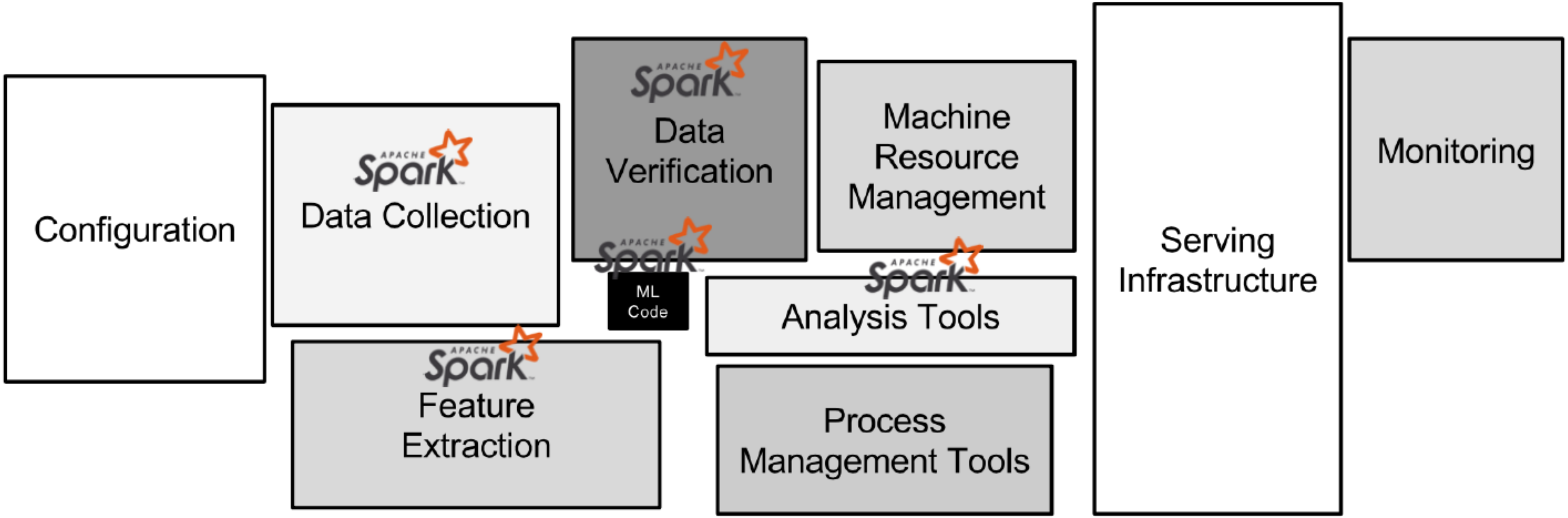




Code time !

```
$ docker run -it -p 8888:8888 --ipc=host ufoym/deepo:all-py27-jupyter-cpu jupyter notebook --no-browser --ip=0.0.0.0 --allow-root --NotebookApp.token="demo" --notebook-dir='/root'
```

Why Spark ?



Configuration
Data Collection
Data Verification
ML Code
Machine Resource Management
Analysis Tools
Serving Infrastructure
Monitoring
Feature Extraction
Process Management Tools

What does Spark offer for deep learning ?

Integrations with existing DL libraries

- Deep Learning Pipelines (from Databricks)
- Caffe (CaffeOnSpark)
- Keras (Elephas)
- mxnet
- Paddle
- TensorFlow (TensorFlow on Spark, TensorFrames)
- CNTK (mmlspark)

Implementations of DL on Spark

- BigDL+ Analytic Zoo
- DeepDist
- DeepLearning4J
- SparkCL
- SparkNet

Options of Keras on Spark

maxpumperla / elephas

Watch 108 | Star 1,122 | Fork 246

Code | Issues 32 | Pull requests 0 | Projects 0 | Wiki | Insights

Distributed Deep learning with Keras & Spark http://maxpumperla.com/elephas/

spark | keras | neural-networks | deep-learning | distributed-computing

309 commits | 3 branches | 5 releases | 16 contributors | MIT

intel-analytics / BigDL

Watch 220 | Star 2,776 | Fork 662

Code | Issues 109 | Pull requests 30 | Projects 0 | Wiki | Insights

BigDL: Distributed Deep Learning Library for Apache Spark https://bigdl-project.github.io/

deep-learning | spark | neural-network | big-data | hadoop | python | scala | keras | ai

2,442 commits | 11 branches | 8 releases | 54 contributors | Apache-2.0

Why BigDL ?

Lenet | Inception | Vgg | Resnet
Fast RCNN | SSD | RNN | LSTM
Deep Speech | Seq2Seq | Auto Encoder | Recommendation

Examples | Documents
Notebook | Tensor Board
Scala | Python

Layers

- Spatial Convolution
- Spatial MaxPooling
- Volumetric Convolution
- Volumetric MaxPooling
- Fully Connected
- ReLu
- LRN
- RNN
- Batch Normalization
- Other Layers

Optimization

- SGD
- Adagrad
- LBFGS
- Adamx
- Adam
- Adadelta
- RMSprop

Criterion

- Cross Entropy
- ClassNLL
- MSE
- Dice Coefficient
- Margin
- Abs

Deep Learning Building Blocks
Layers, Optimizers, Criterion
Deep Learning Models

Scala* and Python* support
Spark ML Pipeline integration
Jupyter* notebook integration
Tensorboard* integration
OpenCV* support
Model Interoperability (Caffe*/TensorFlow*/Keras*)

CONSUMER

CALL CENTER ROUTING
IMAGE SIMILARITY SEARCH
SMART JOB SEARCH

HEALTH

ANALYSIS OF 3D MRI MODELS FOR KNEE DEGRADATION

FINANCE

FRAUD DETECTION
RECOMMENDATION
CUSTOMER/MERCHANT PROPENSITY

RETAIL

IMAGE FEATURE EXTRACTION

MANUFACTURING

STEEL SURFACE DEFECT DETECTION

SCIENTIFIC COMPUTING

WEATHER FORECASTING

Why Analytics Zoo ?

Analytics Zoo -> Unified Analytics + AI Platform for Spark and BigDL

Reference Use Cases	Anomaly detection, sentiment analysis, fraud detection, chatbot, sequence prediction, etc.
Built-In Algorithms and Models	Image classification, object detection, text classification, recommendations, GAN, etc.
Feature Engineering and Transformations	Image, text, speech, 3D imaging, time series, etc.
High-Level Pipeline APIs	DataFrames, ML Pipelines, Autograd, Transfer Learning, etc.
Runtime Environment	Spark, BigDL, Python, etc.

https://github.com/intel-analytics/analytics-zoo

Use case for user item propensity model

- Propensity to buy
- Propensity to use
- Propensity to engage
- Propensity to contract
- Etc.

- Life Insurance
- Auto Insurance
- Homeowner's Insurance
- Mortgage
- Re-Financing
- Credit cards
- Personal Loans
- Investments
- Satellite TV
- Cable
- Netflix
- Online banking
- Online money management
- Voting
- Disease

https://catalog.data.gov/dataset/purchase-card-pcard-fiscal-year-2014

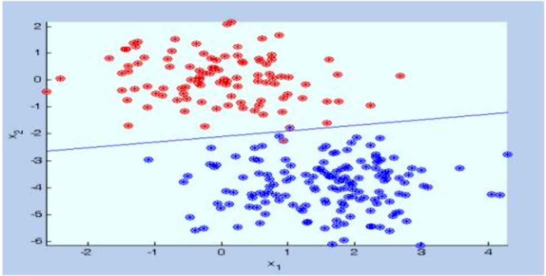
Purchase Card (PCard) Fiscal Year 2014

☐ **Metadata Updated:** September 15, 2016

This dataset contains information on purchases made through the purchase card programs administered by the state and higher ed institutions. The purchase card information will be updated monthly after the end of the month. For example, July information will be added in August.

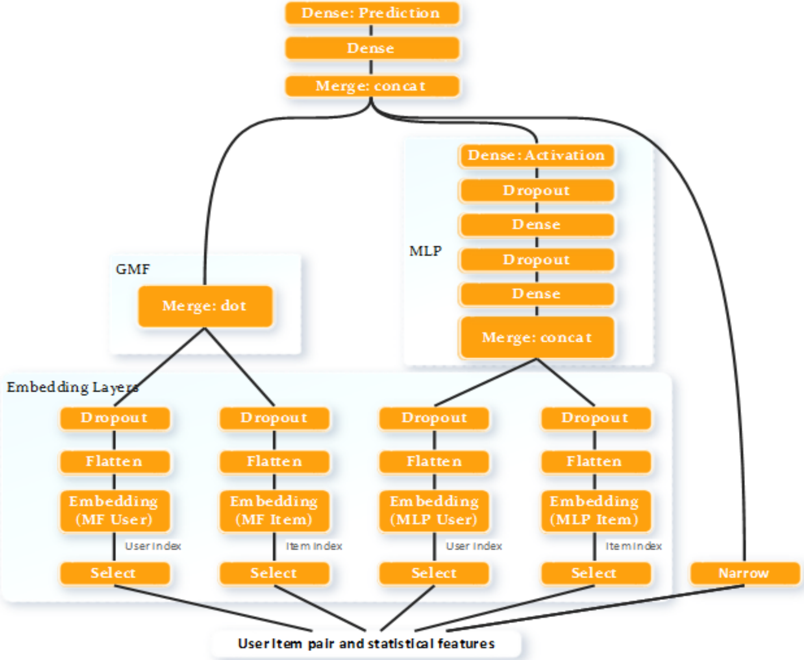
Build a User Item Propensity model with deep learning algorithms

Logistic regression



Collaborative Filtering

	1	2	3
Product A	5☆	5☆	5☆
Product B	3☆	3☆	3☆
Product C	5☆	3☆	4☆



Neural Collaborative Filtering deep learning algorithm

https://arxiv.org/pdf/1708.05031.pdf
https://github.com/hexiangnan/neural_collaborative_filtering

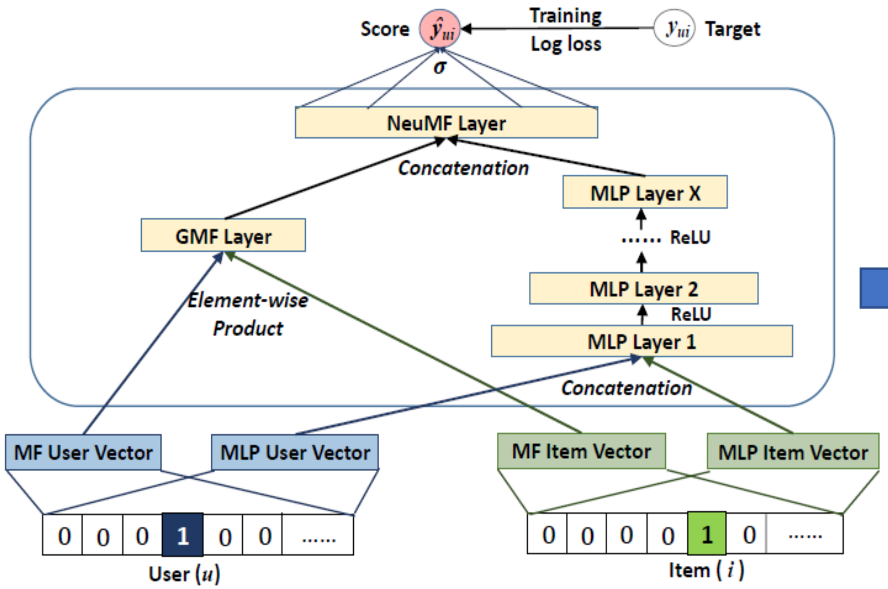
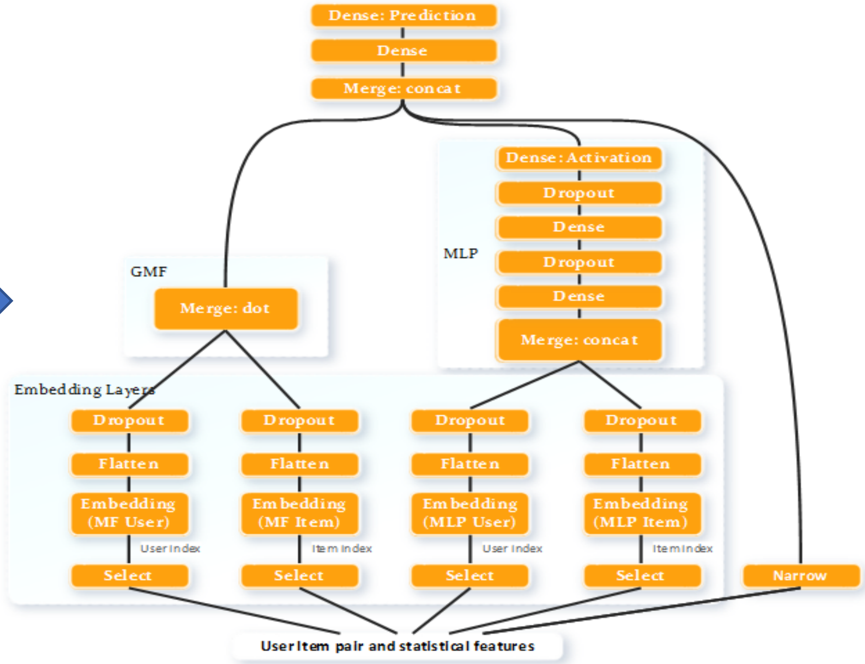


Figure 3: Neural matrix factorization model



Code Lab 1

```
$ docker run -it -p 8080:8080 -p 8443:8443 -p 10000:10000 -p 8998:8998 -p 12345:12345 -p 8088:8088 -p 4040:4040 -p 7077:7077 -e NotebookPort=12345 -e NotebookToken="demo" -e RUNTIME_DRIVER_CORES_ENV=1 -e RUNTIME_DRIVER_MEMORY=2g -e RUNTIME_EXECUTOR_CORES=1 -e RUNTIME_EXECUTOR_MEMORY=4g -e RUNTIME_TOTAL_EXECUTOR_CORES=1 --name demo -h demo demo:latest bash
```

Build a Docker image and run a Keras on Spark container

https://github.com/jack1981/AaaSDemo

Run the NCF deep learning pipeline for User Item Propensity model

https://github.com/jack1981/AaaSDemo

Q & A