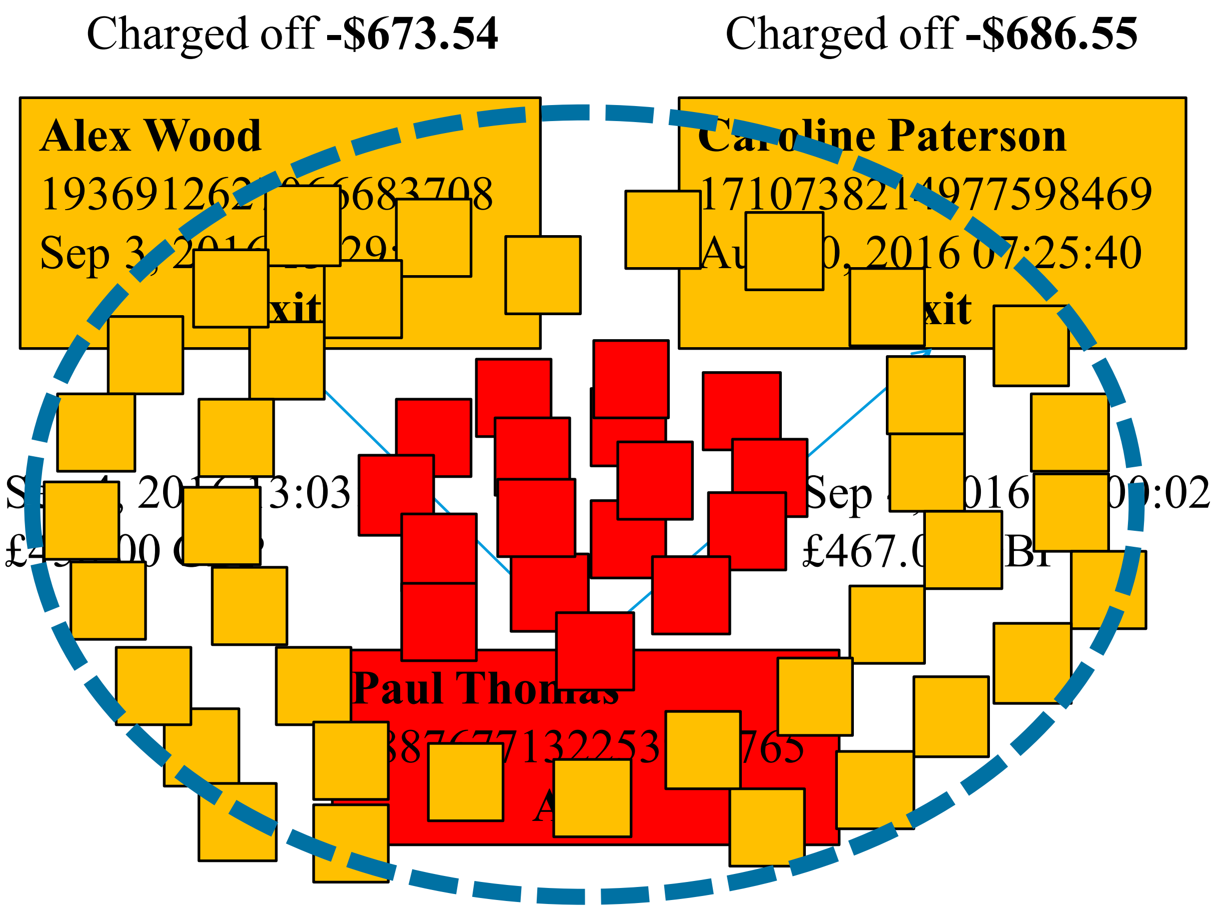
Machine Learning as a Platform at PayPal Risk

Start from a Sophisticated Payment Fraud Case



- ✧The fraudsters scaled the attack by opening many accounts
- ✧The attack cause this loss in just a few days
- ✧It was a clean and sophisticated fraud with no links or velocity

Agenda

- PayPal Risk Overview & Introduction
- An End-to-End Machine Learning Platform
- Deep Learning Inference Framework & Service
- Best Practices & Learnings

Agenda

- PayPal Risk Overview & Introduction
- An End-to-End Machine Learning Platform
- Deep Learning Inference Framework & Service
- Best Practices & Learnings

PayPal Risk: Building Trust in a New World

Industry Trends Redefining the Way PayPal Builds Trust Between Buyers and Sellers

TRANSFORMATION OF MONEY

40% of money is in the form of checks or cash; predicted to go down to 25%[1]

MOBILE PAYMENTS BECOMING MAINSTREAM

Mobile spending projected to rise by roughly $190B over the next 3 years[2]

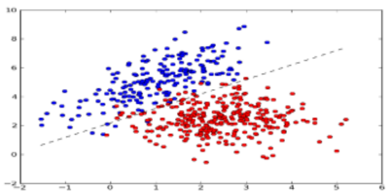
CHIEF RISK OFFICER = CHIEF TRUST OFFICER

500M to 1B identities stolen globally; $32M in U.S. retail fraud losses[3]

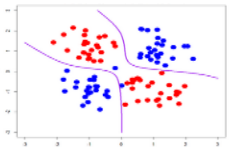
Sources: [1] Nielsen, Dept of Commerce, JP Morgan; [2] PayPal & IPSOS Study; [3] Symantec, Gemalto, LexisNexis

Hybrid Solution of Risk Fraud Detection & New Product Promotion

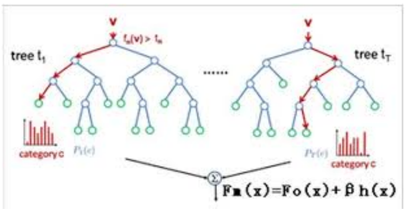
Linear/Logistic Regression



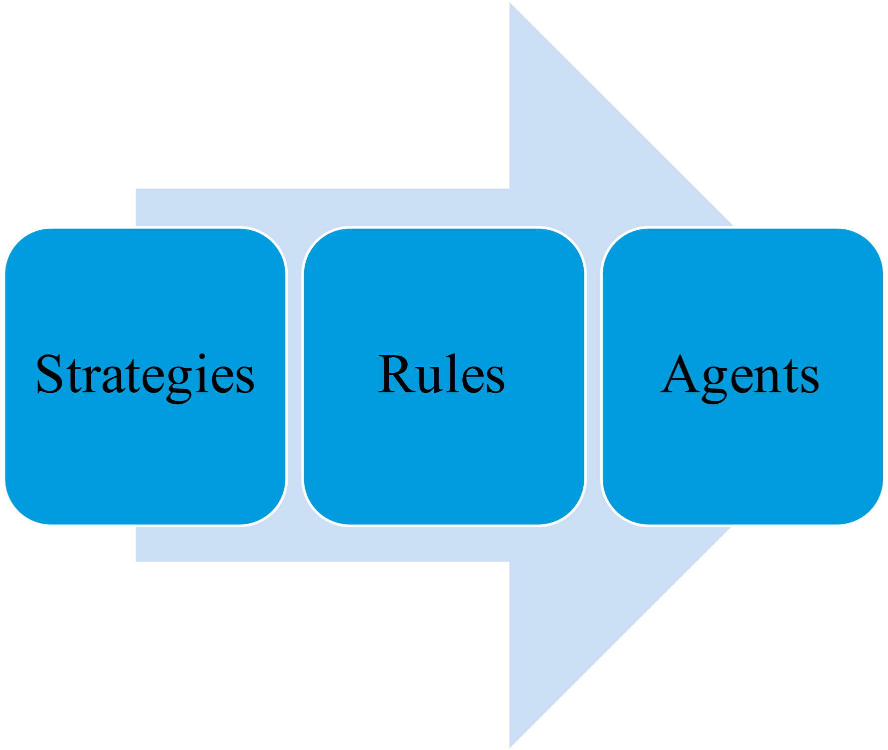
Neural Networks
Ensemble/Embedding Models



Tree Ensemble Models



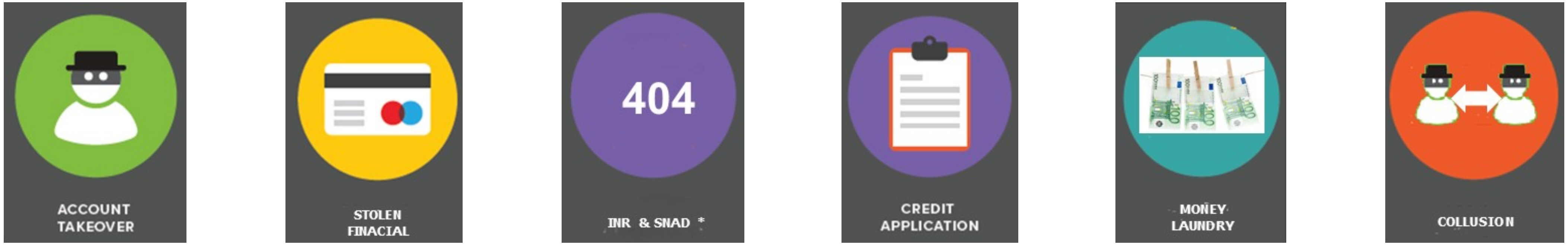
* Different kinds of models adopted in different fraud cases



* Strategies is tree based rules based on machine learning model scores
* Rules for some fraud trend which cannot be reflected in models in time

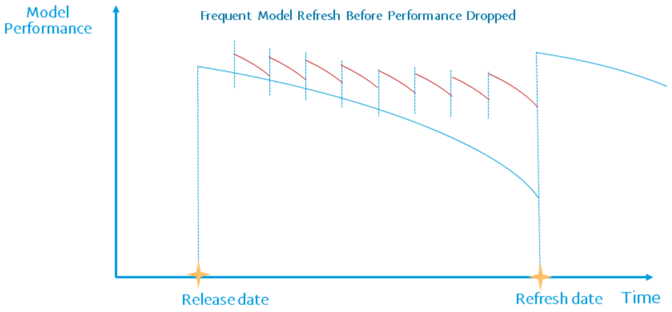
More and More Machine Learning Scenarios at PayPal Risk

More and More Business Cases

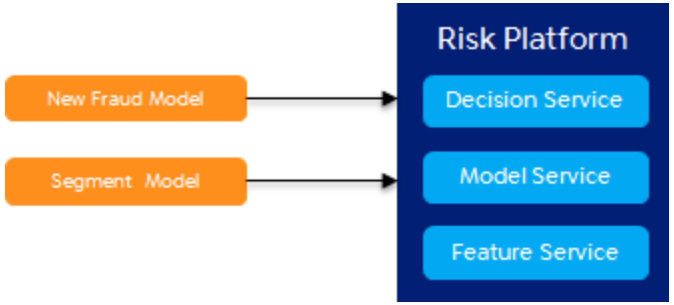


……

Platform Requirements



Fast Model Refresh

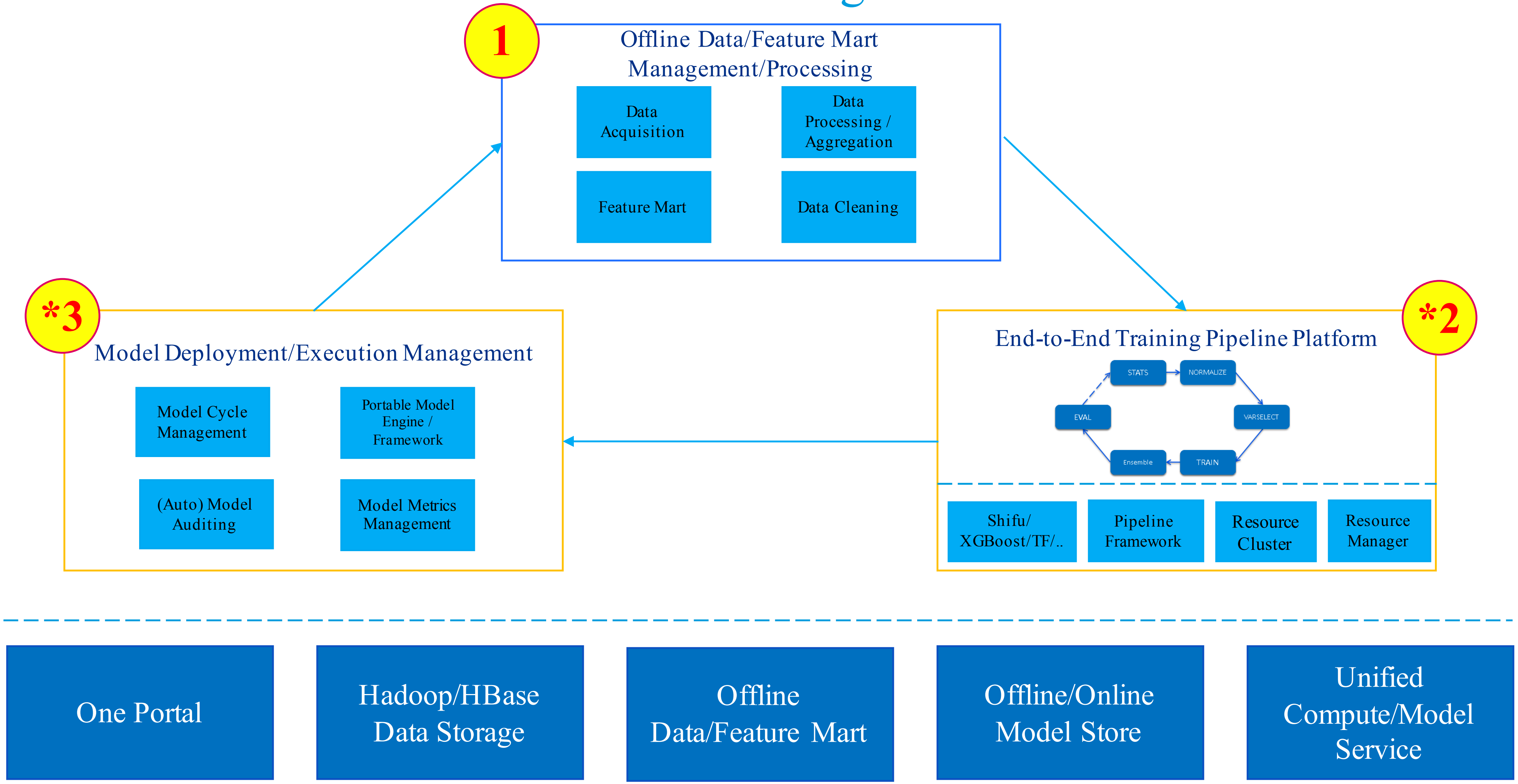


New ML Model On Board

Agenda

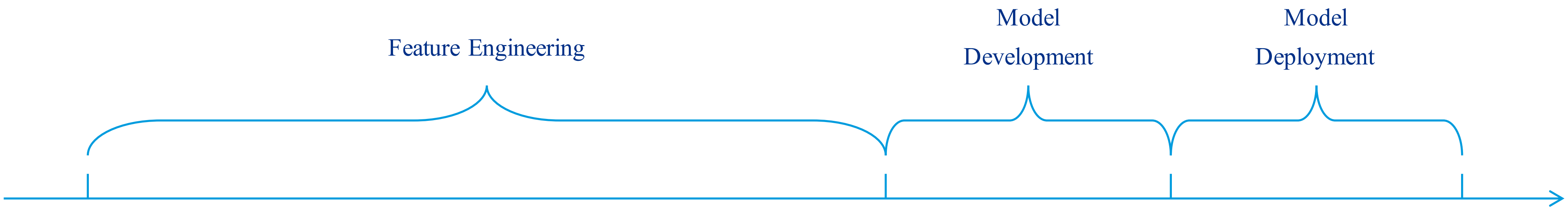
- PayPal Risk Overview & Introduction
- An End-to-End Machine Learning Platform
- Deep Learning Inference Framework & Service
- Best Practices & Learnings

Overview: An End-to-End Machine Learning Platform

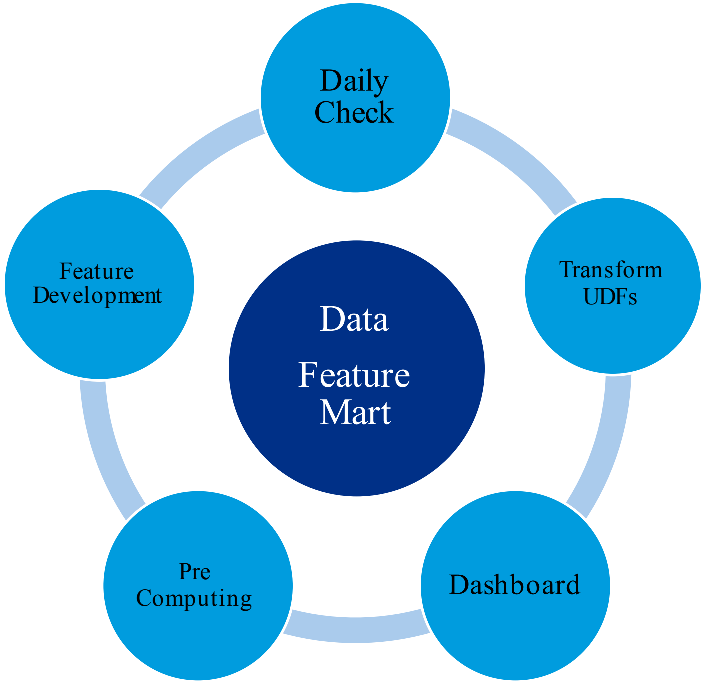


* Components Support Automation

1. Data & Feature Platform



Pain point: > 50% of time is in feature engineering: data preparation, data cleaning, data transforming



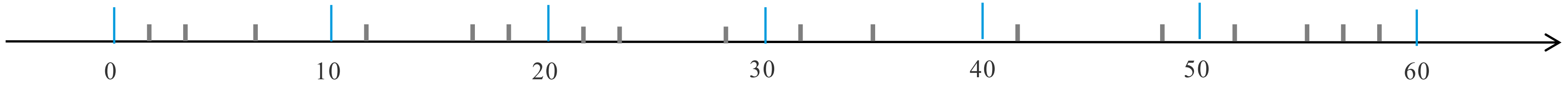
- ✧ Feature data mart is built to solve feature engineering pain point
- ✧ Clean data daily before new data ETL to data mart
- ✧ Dashboard for users to check feature metrics
- ✧ UDF for user easy to do transform
- ✧ Built on Pig/Hive/SparkSQL, unified interface / pipeline

Statistical Features & Complicated/Embedding Features

Variable: traditional variable is profile/behavior based statistical variables like # of transactions in a period.

Example: transaction decay value in last 60 hours

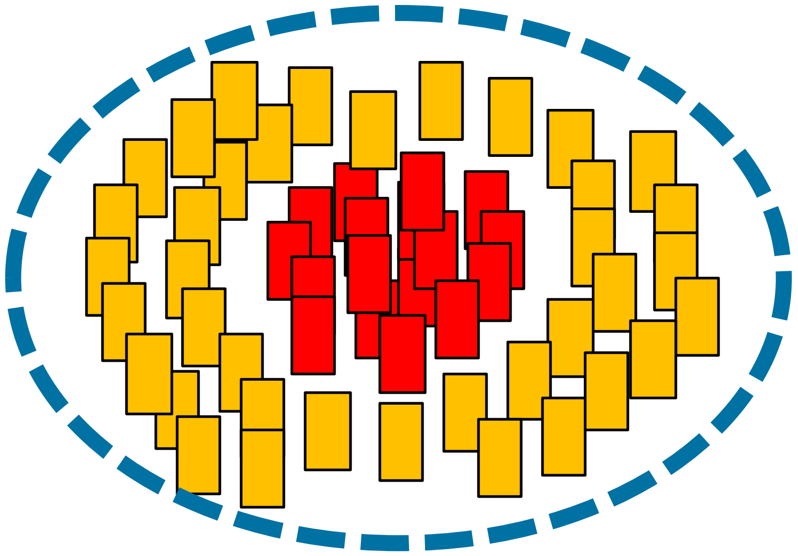
$$decay_{\downarrow}value = \sum_{i=1}^{bin_{\downarrow}cnt} decay(pit) * count$$



Component: complicated variable developed by complicated data mining process like clustering or classifying on specified data set.

Example: fraud networks on clustering

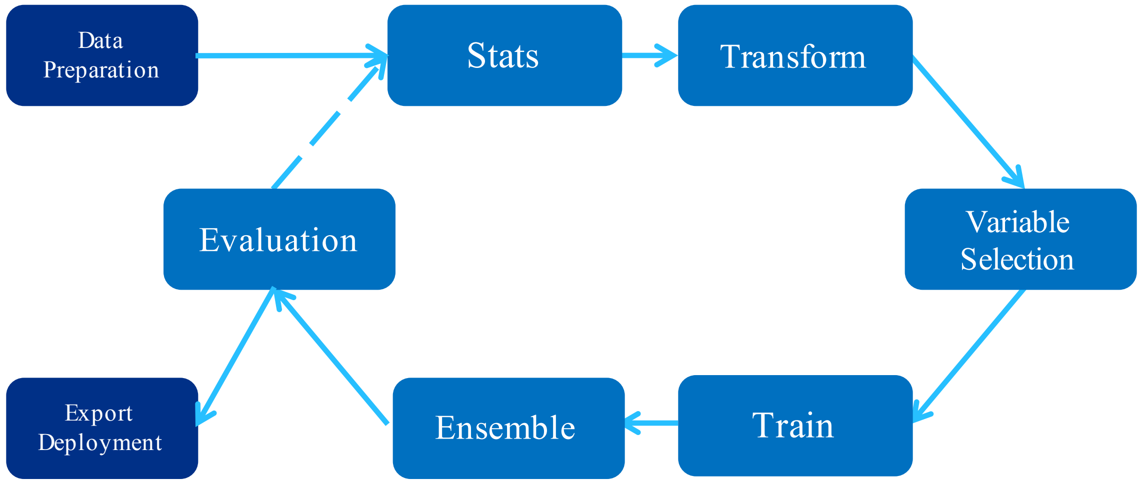
Typical use case: collusion model



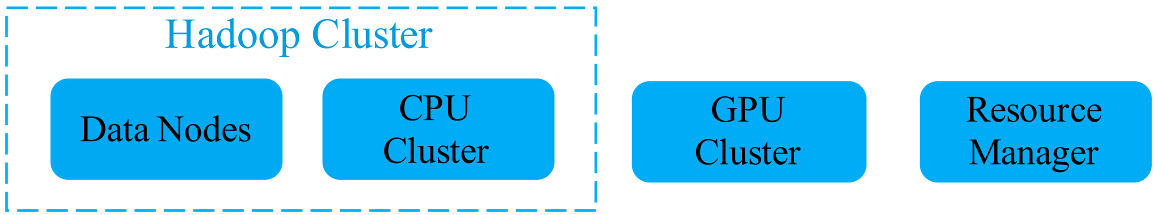
1. The fraudsters scaled the attack by opening many accounts
2. The attack causes this loss in just a few days
3. It was a clean and sophisticated fraud with no links or velocity

2. (Auto) End-to-End Training Platform

Training Pipeline Layer



Resource Management Layer



✧ Training Pipeline Layer

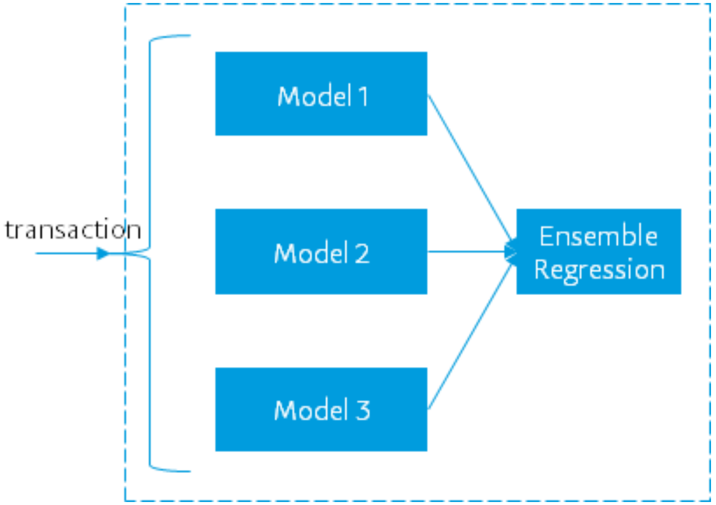
- ✧ Full pipeline support without stepping out
- ✧ Flexible pipeline (restarting from every step)
- ✧ Large scale/high performance for more tries
- ✧ More training frameworks proactively adapted
- ✧ More AI approaches natively support
- ✧ Integrated with offline/online model store

✧ Resource Management Layer

- ✧ Such layer is transparent to front-end users
- ✧ Unified data input layer
- ✧ Multiple tenancy support for resources
- ✧ Scheduler for CPU & GPU resources

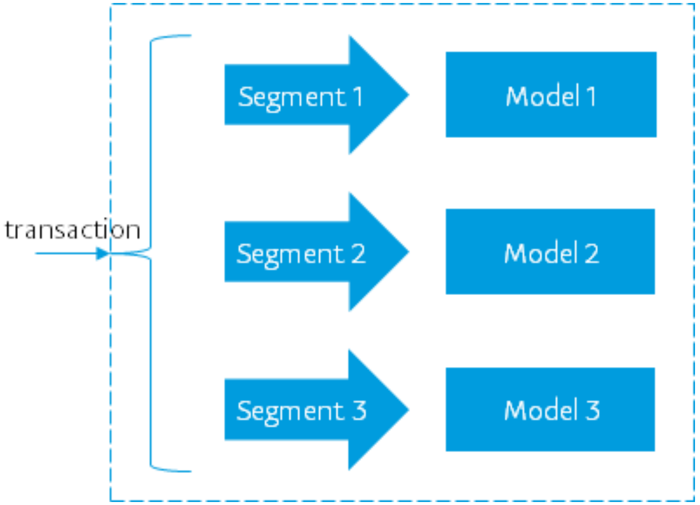
Ensemble/Segment/Embedding Model Native Support

Ensemble Models



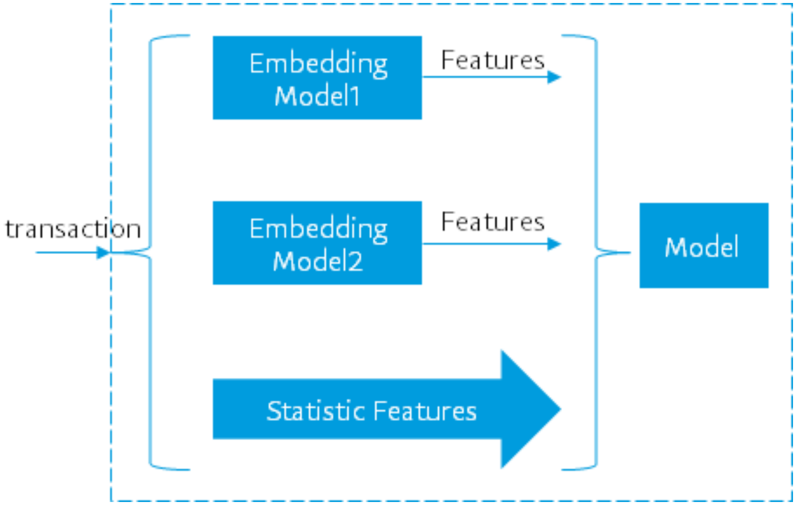
1. Meta model can be LR/NN/GBDT/LSTM …
2. Ensemble model by LR or Poly-Regression by align different model scores into one score
3. Logic under ensemble is each mode has lift, by ensemble, can leverage all lifts

Segment Models



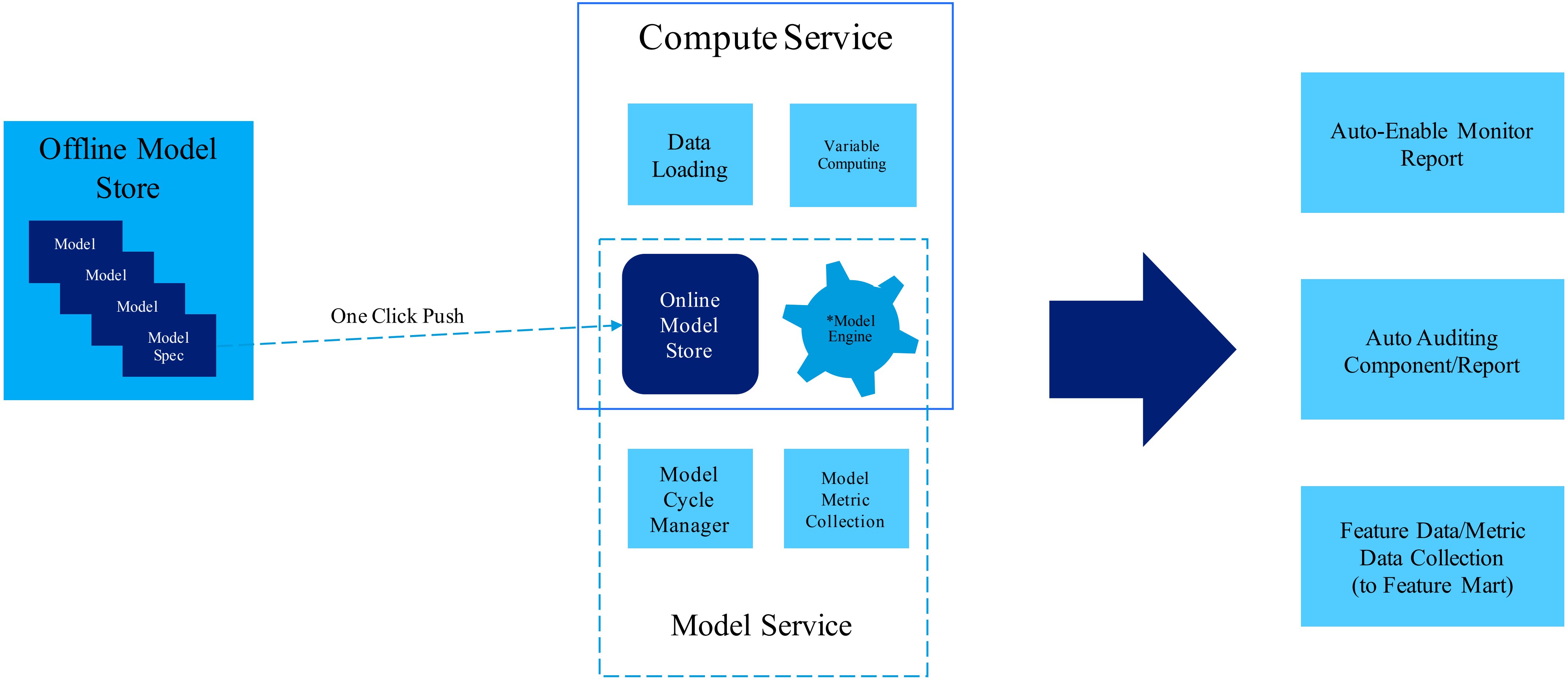
1. Segment is business condition
2. In different segments, models/features can be different
3. Start from a general model, then deep into segments to check if segment model is needed

Embedding Model



1. Embedding is useful for new feature generation
2. Final models leverage raw features and embedded features
3. Model cascading like ensemble models

3. (Auto) Model Deployment & Execution



* Portable model engine support in compute service or model service

Offline & Online Model Cycle Management

Offline Model Cycle Management

- Offline Model Store
  - Store historical models
  - Key checkpoint model storage
  - Link with model sync system for fast model push
- Model Profile Information
  - Modeling platform, version
  - Training data information, variable stats
  - For ensemble, sub model profile information
  - Variable importance
  - Key training parameters
- Model Evaluation Result
  - Evaluation data stats
  - Performance metrics
- ……

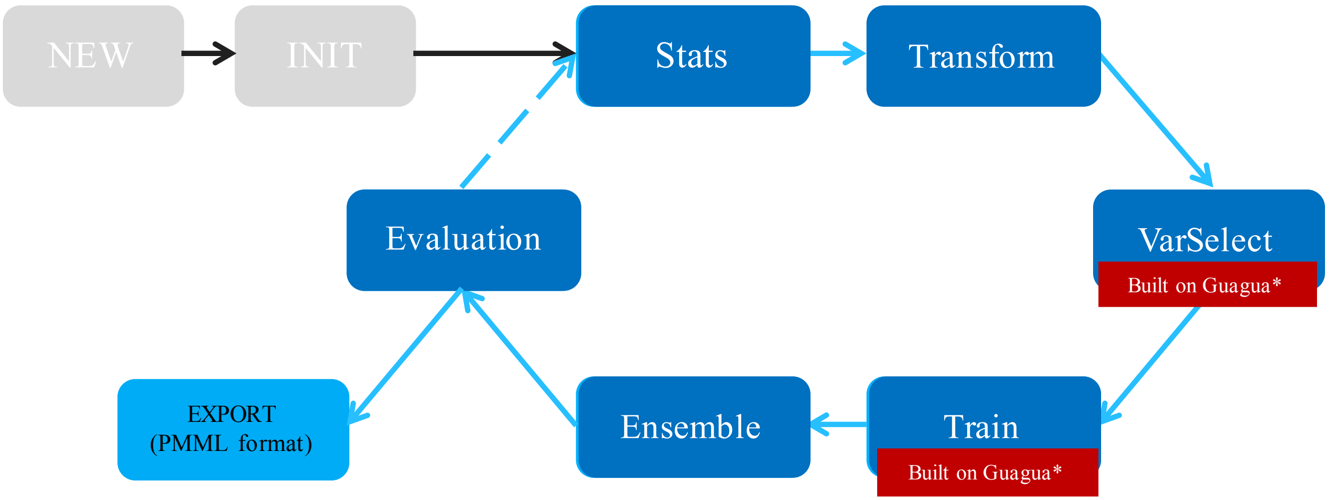
Online Model Cycle Management

- Model State Management
  - Deploy -> Audit -> Serving -> Dead
  - Version management
  - Ensemble/segment model management
- Model Metrics Collection & Monitor
  - Computation cost
  - Memory cost
  - Disk cost
  - Feature cost
- Portable Model Engine / Service
  - Easy to port into compute service/model service/…
  - Isolate CPU with IO, enable CPU optimizations
  - Isolate audit model & production model computation
- ……

Machine Learning Pipeline Framework

Shifu is an open-source, end-to-end machine learning and data mining framework built on top of Hadoop.

- https://github.com/ShifuML/shifu
- 5+ orgs/companies leverage Shifu to train models outside of PayPal
- 5+ contributors for PR outside of PayPal



*Guagua is an iterative computing framework on Hadoop YARN: https://github.com/ShifuML/guagua

Fast & Powerful: Distributed training to handle large dataset.

Standard process and independent tool to build model

Data Scientist + Engineer = More Possible

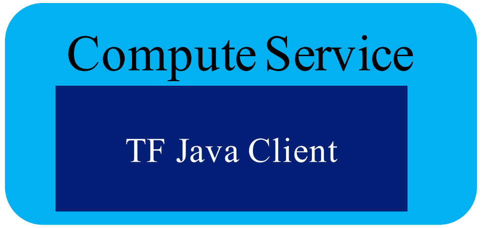
- Variable ReBinning
- Sensitivity Analysis
- Correlation Analysis
- PARETO Variable Selection
- Segments Combine Training

Agenda

- PayPal Risk Overview & Introduction
- An End-to-End Machine Learning Platform
- Deep Learning Inference Framework & Service
- Best Practices & Learnings

Deep Learning Inference Support in Compute Service

Java Inference Client



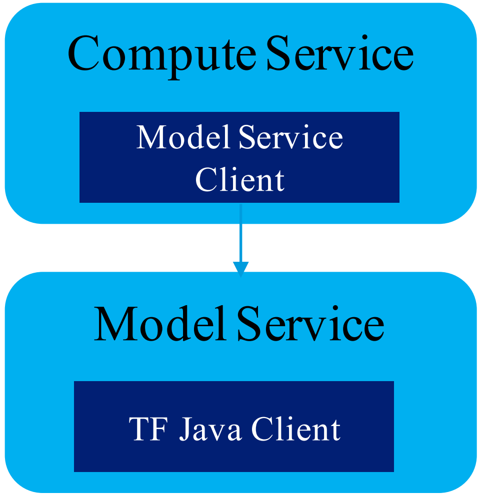
Pros:

DNN/CNN/RNN are All Supported Natively

Cons:

CPU Bound, Not Isolated from Compute Service

Rest DL Inference Service



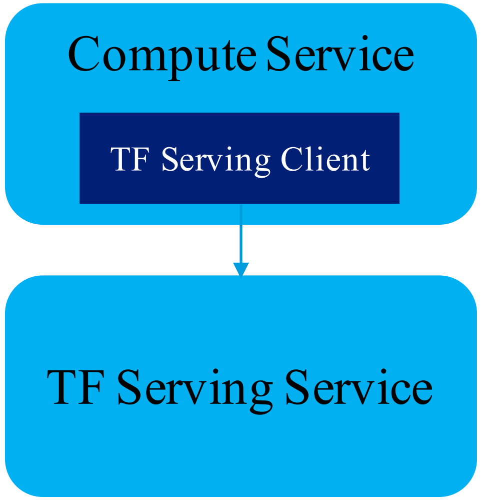
Pros:

Dedicated Model Service

Cons:

Need Extra Resources

TensorFlow Serving



Pros:

TF Serving is Supported by Google

Cons:

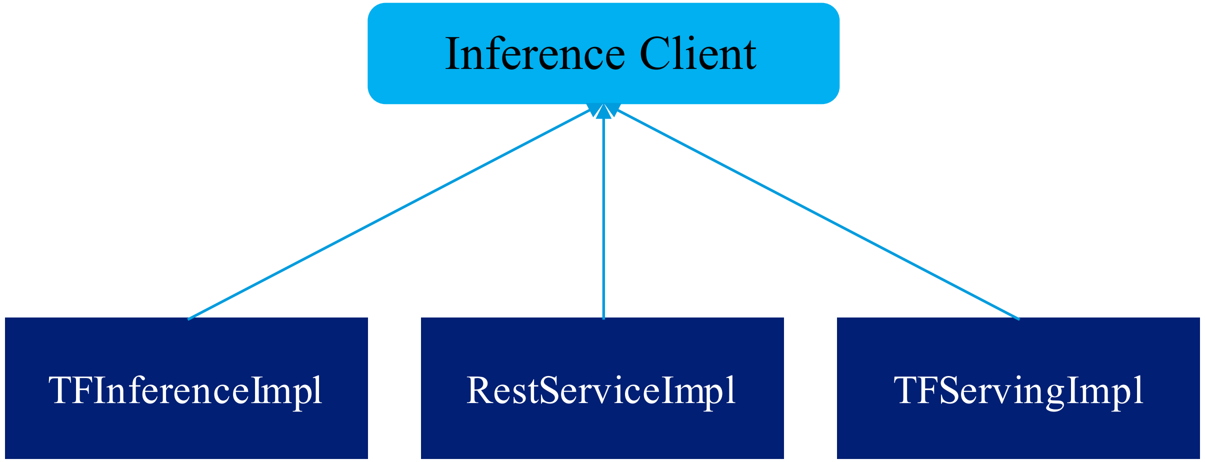
Need Extra Resources

gRPC is http 2.0 based

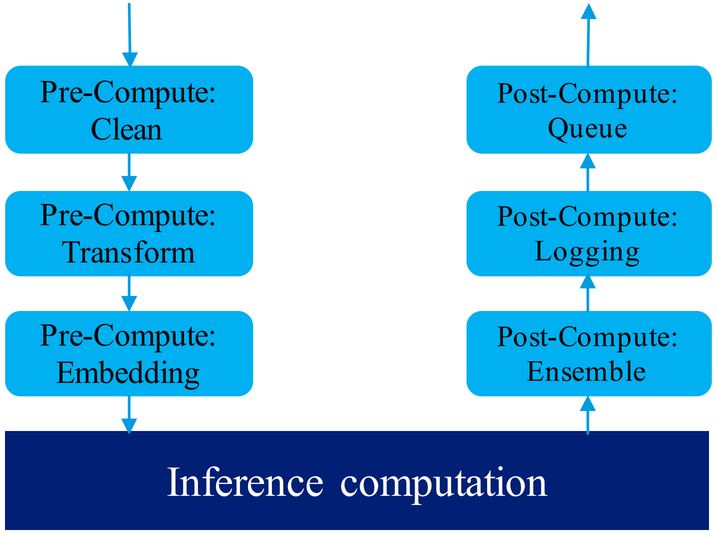
Only TF model spec is supported

Generic Deep Learning Inference Framework

Generic Interface

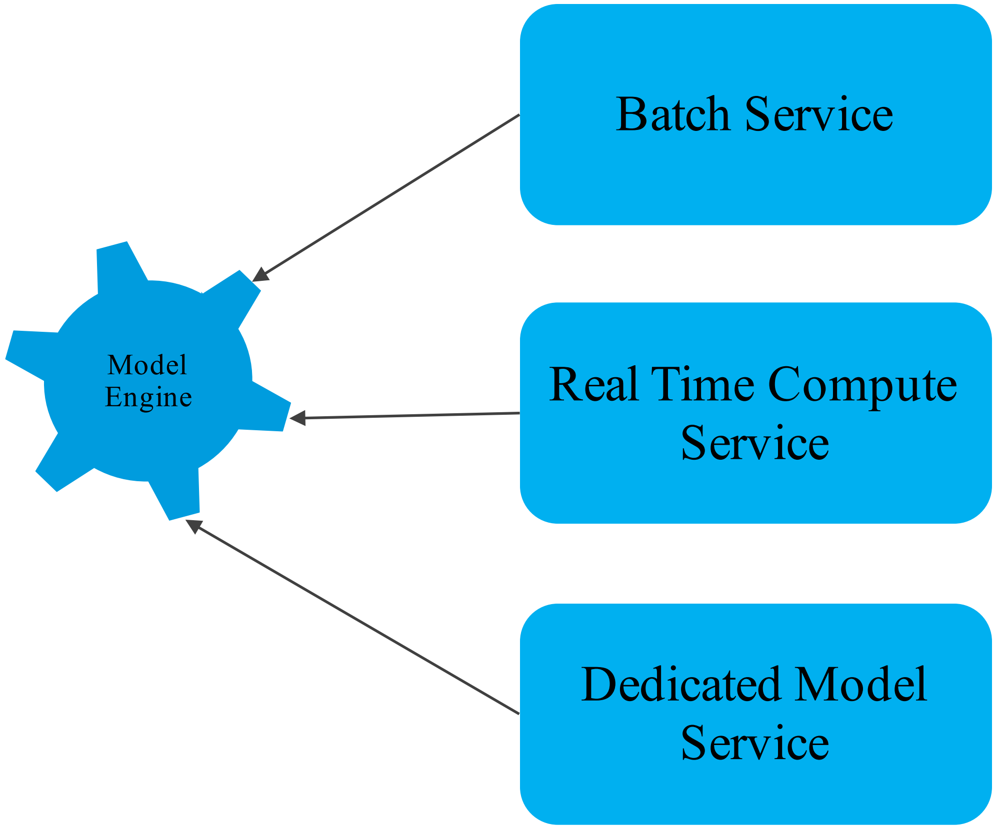


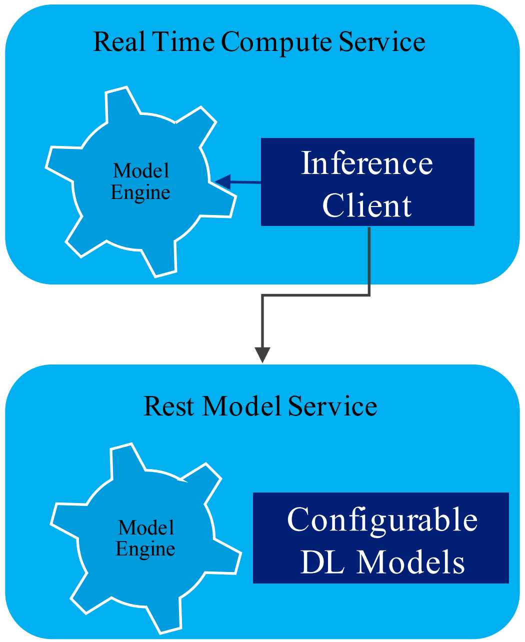
Compute Logic & Interceptors



* All inference implementations can be replaced by using different implementation
* Interceptor mechanism supports logic pre and post inference
* Same interceptor can be configured to different inference implementation

Portable Model Engine & Smart Client



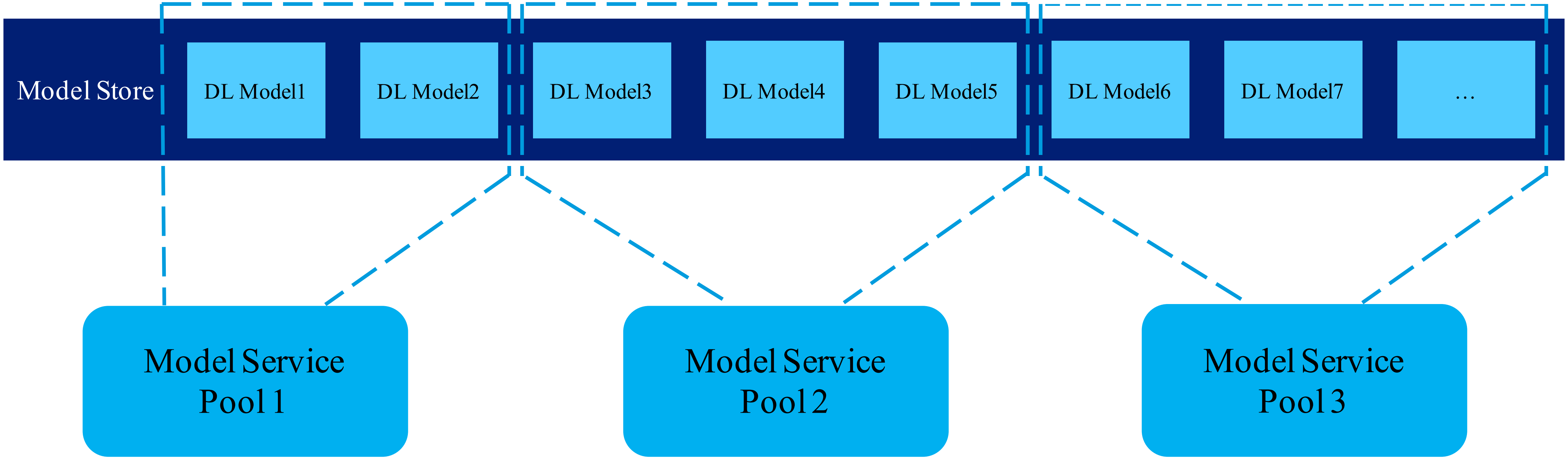


* Models can be run in compute service or dedicated model service
* Portable model engine means such model by dynamic configuring it run in compute service or model service
* Real time compute service including data loading, feature computation and model computation
* Smart client means no code change to call model from local or remote service

Unified/Scalable Deep Learning Model Service

Questions:

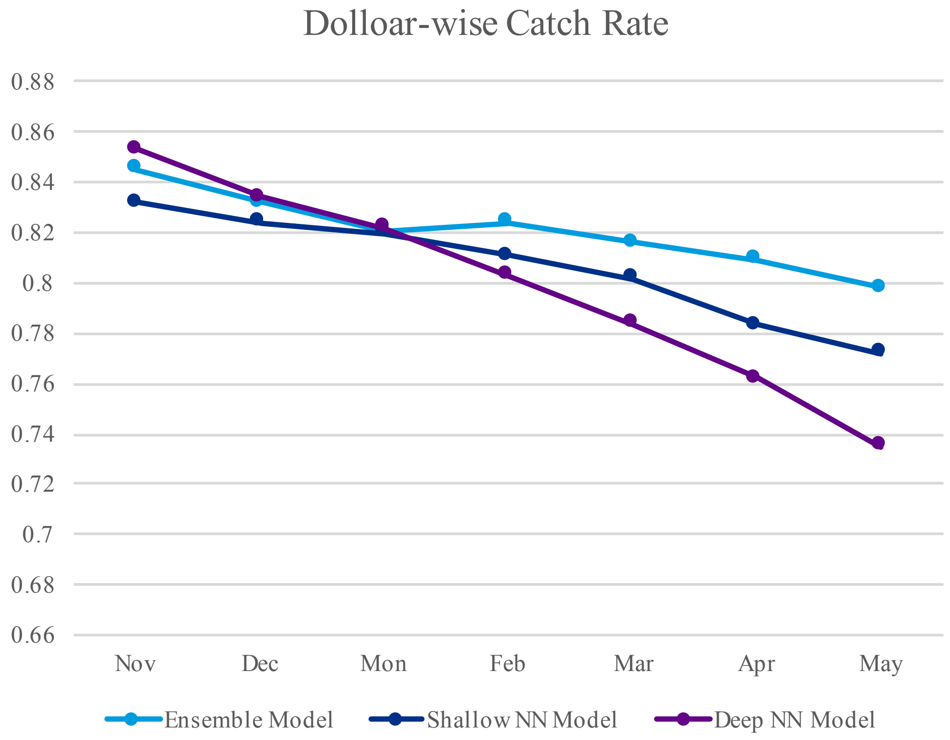
1. How to scale model service to 1000 models level?
2. How to dynamically call multiple models in one request?



Agenda

- PayPal Risk Overview & Introduction
- An End-to-End Machine Learning Platform
- Deep Learning Inference Framework & Service
- **Best Practices & Learnings**

Model Performance: Stable > Accurate

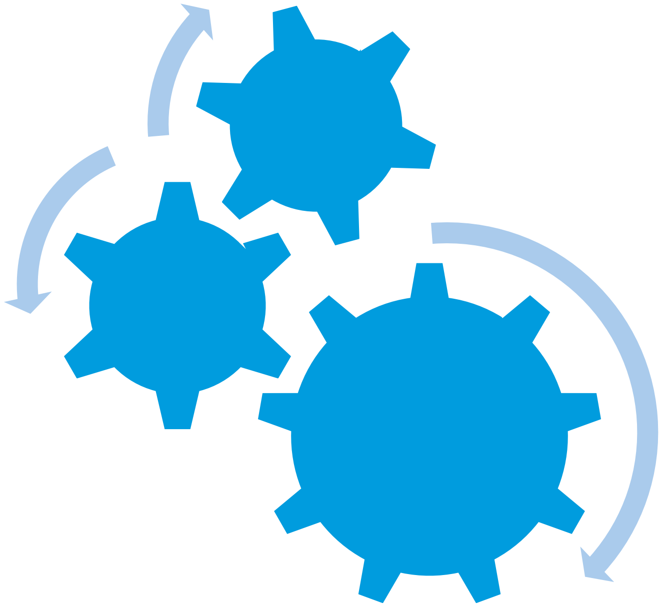


- ✧Deep model is good at first but later worse
- ✧Ensemble & bagging model is the most stable one
- ✧Cost of ensemble model < deep NN model
- ✧Deep model (feature embedding) + ensemble model (stable performance)

More Intelligent Training Platform

Auto Tuning

Auto tune system parameters for run time performance



Auto Diagnose

1. Suggest solutions when failures
2. Auto recovery for some kind of failures

Auto ML

1. Automated parameter tuning
2. Automated algorithm selection
3. Automated feature selection
4. Automated model ensemble

Performance, Stability, Flexibility

Goal of Platform: **Fast** but Less Failures

1. 80% training jobs are finished in 2 hours in one week
2. 94% training jobs running successfully in last one week

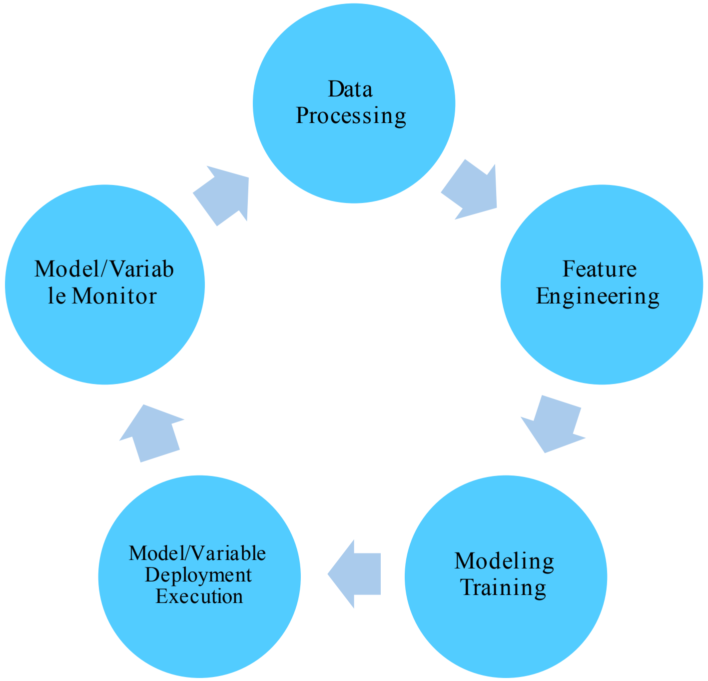
Goal of Platform: **Scalable** but Less Resource Usage

1. # Of workers scaled to maximal 3000; (20T memory)
2. Memory reduction by leveraging float numbers in NN and short in tree-ensemble models

Goal of Platform: **Automated** but Flexible

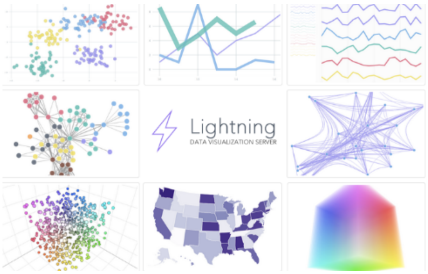
1. Automated pipeline to support fast model refresh case
2. Whole pipeline is flexible and can be integrated into different tools/platforms

Unified Machine Learning System

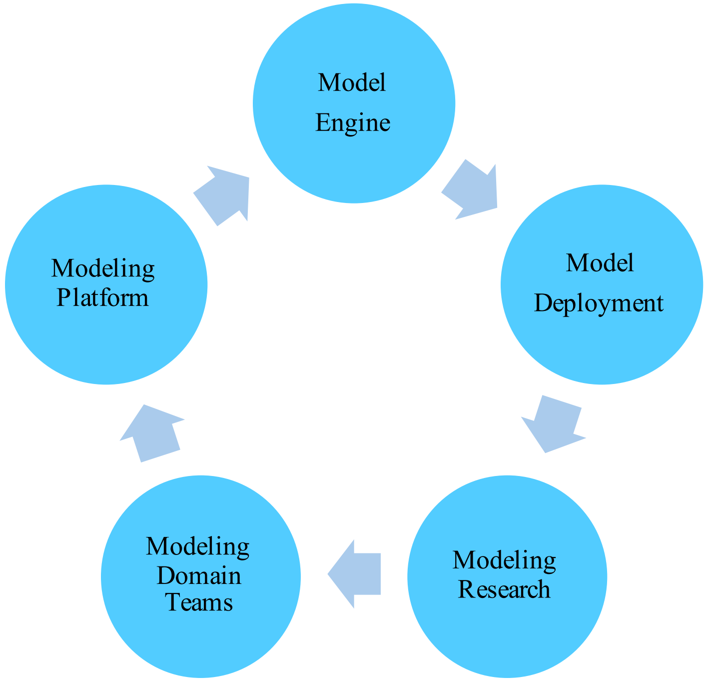


1. Continuous evolvement framework/platform
2. Key is unified as one product
3. More data/feature/model governance

Python notebook/data visualization to enable better eco system



UI is very important!!!



1. Evolved in every domain of modeling
2. Better/quick feeding requests for domain teams
3. Support work for more/better adoptions
4. Collaborations with modeling/data science teams

Thank You!

极客邦科技
企业培训与咨询
Geekbang>.

扫码关注

获取更多培训信息