# TURBOML

Product Brief

## About

TurboML is a machine learning platform reinvented for real-time. Imagine having all the capabilities of an ML platform - from creating data sources, features, models, deployments and metrics to continuously maintaining and improving them - while simultaneously being able to use the freshest real-time data .

**How quickly would your team be able to iterate if they could directly test their hypotheses on the live production data? How effective and relevant would your models be if they could learn from real-time data?**

For example, in an ETA prediction use-case, a ride completion event, or in a fraud detection use-case, a chargeback can be used to update the ML model, evaluate and compare different models, update real-time features that are passed on to the model, and more, all in real-time.

Our team has expertise at the intersection of streaming systems and production machine learning. Leveraging our research background, as well as our experience building large-scale systems at Google and AWS, we are on a mission to democratize real-time machine learning.

## Real-Time ML Evolution

### Real-time Predictions

Get outputs from your model in an on-demand, synchronous manner!

- Fresh Predictions with Online Inference
- Supports Low Latency
- Simple On-Demand Features

**Want to make ML more responsive**

Most ML frameworks and platforms out there support real-time predictions via REST and RPC based methods.

### Real-time Features

Transformations on recent data passed as context for your model!

**All the goodness of Real-time Predictions, and:**

- Fresh Features w/ streaming computation
- Easier Feature Experimentation
- Complex Aggregation-based features

**Want to leverage dynamic context**

Companies like TurboML, Tecton, Fennel, Hopsworks, and Kaskada offer feature platforms to help define and enable real-time features. Alternatively, you can set up your own streaming pipelines to compute these features, with a feature store supporting online serving.

### TurboML

Use the freshest data to update your ML models

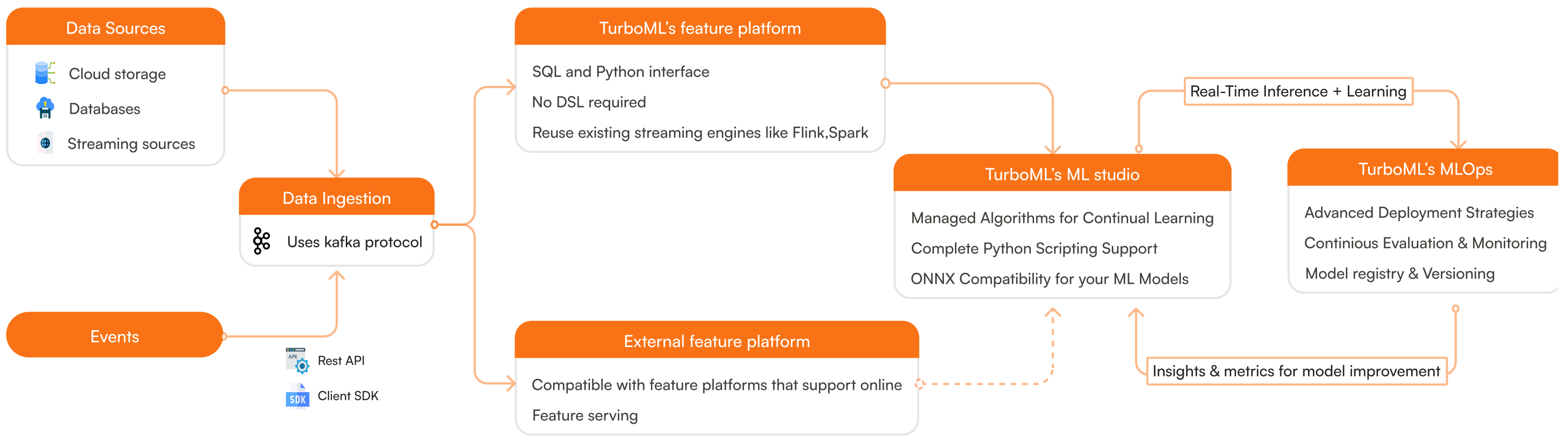
**All the greatness of Real-time Features, and:**

- Fresh Models with Continual Learning
- Live experimentation
- Handle cold-start, rare-events & long-tails

**Want to stay fresh**

Packages like River and Vowpal Wabbit can be great starting points for incremental ML. However, a complete production deployment of continual learning is much more nuanced. Get in touch with us at TurboML to understand how you can get started.

# Architecture


Data Sources
Cloud storage
Databases
Streaming sources
Data Ingestion
Uses kafka protocol
Events
Rest API
Client SDK
TurboML's feature platform
SQL and Python interface
No DSL required
Reuse existing streaming engines like Flink,Spark
External feature platform
Compatible with feature platforms that support online Feature serving
TurboML's ML studio
Managed Algorithms for Continual Learning
Complete Python Scripting Support
ONNX Compatibility for your ML Models
Real-Time Inference + Learning
TurboML's MLOps
Advanced Deployment Strategies
Continious Evaluation & Monitoring
Model registry & Versioning
Insights & metrics for model improvement

## Interaction

| | |
|---|---|
| **Platform** | Frontend UI, Jupyter server wth Python SDK |
| **Models** | REST APIs (Request-Response), Kafka (Event-Driven |
| **Outputs** | Grafana (Dashboard), Real-time OLAP (Analytics) |

## Deployment

| | |
|---|---|
| **Installing dependencies** | Completely containerized with a self-sufficient environment |
| **Deployment model** | Completely on your premises or your VPC |
| **Infra requirements** | Scale-dependent, single VM (e.g. c6a.8xlarge) should suffice for PoC |
| **Data Privacy** | Completely private to you, isolated to your own internal network |

01

## Data Ingestion 📊

- Pull-based ingestion: Readily available connectors that can connect and continuously ingest data from any of the following data sources:
  - Data lakes like Snowflake or Databricks
  - Cloud storage solutions like S3
  - Databases like RDS or MongoDB
  - Streaming sources like Kafka or Kinesis
- Push-based ingestion: To push the data to us on a per-event basis
  - REST API endpoints
  - Performant client SDKs in multiple languages
- Support for handling sensitive data (e.g. masking)

02

## Feature Engineering ⚙️

- Flexibility of SQL and arbitrary Python to define features
- Jupyter environment to supplement python interface for quick experimentation
- Optimized implementations of common feature transformations, such as complex time-aggregations
- Simple no-code UI
- (Optional) Bring in your own feature platform with online serving capabilities

03

## Modelling Workflow 🔄

- ML algorithms published in top ML conferences, proven effective in online and continual learning scenarios, optimized for real-time use-cases
- Latency as low as tens of microseconds
- Dynamically adapt to drifting data distributions:
  - Automatic
  - Trigger-based (performance, volume, time, or drift)
- Explainable
- Native python support to write your own algorithms
- Bring your own models from any of the popular ML frameworks:
  - PyTorch
  - TensorFlow
  - Keras
  - Scikit-Learn

04

## Consuming Output 📤

- Event-driven downstream application:
  - Push all model outputs to a Kafka topic, which could be subscribed by your downstream application, or directly ingested into another data source
  - Support webhooks with custom logic for the outputs
  - APIs to include the outputs in your existing monitoring/ops dashboards, e.g. Grafana
- Request-response downstream application:
  - Production-ready model-serving REST API endpoints for high-throughput, low-latency inference
- OLAP interface to run real-time analytical queries on the outputs

# Ease of Use

Ease of use is of utmost importance for us at TurboML. Our platform is customized to your needs, whether you want

- To use batch or streaming data sources
- To use our algorithms or bring your own
- To use our feature platform or bring your own
- To use python or SQL or a no-code interface
- Us to manage it, or manage it yourself
- Synchronous or asynchronous ML inference
- To host on the cloud, fully private VPC or on-premises

... and the list goes on. This helps us maintain a zero learning curve, resulting in quicker and higher returns. We're battle-tested for the enterprise, with a horizontally-scalable architecture providing single-digit millisecond latencies and 250,000 QPS (async) on a single instance.

# Why do you need TurboML to optimize your

## Costs

Make ML fast enough to increasingly shift towards on-demand compute, thereby reducing redundancies.

Optimize costs by processing only new incoming data, thus eliminating the need to repeatedly crunch through previously processed information.

Case studies from Grubhub and Etsy show direct cost savings of up to 45 times!

## Revenue

Keeping your models fresh directly correlates with improvements in your business use cases.

Benefits include faster fraud detection, tailored recommendations, improved user engagement, increased click-through rates, and precise forecasts.

Case studies showing improvements across use-cases from TikTok, LinkedIn, Pinterest, Instacart, Airbnb, and many others!

## ROI

How long does it take you to validate a new model or test a new feature, from conception to deployment? How long does it take you to use the data collected today to update their ML models?

Modern ML is data-driven. If you're not able to leverage real-time data in your ML pipelines, you're not making the most out of your data.

## Metrics

Tech giants like Google and Meta have emphasized, time and again, the importance of mitigating drifts through retraining, along with continuous evaluation and model selection.

Your models are only as good as the data on which they are trained. So, the only question is: if you could power your ML with the freshest real-time data, why wouldn't you?

## Productivity

Move from ideation to results in seconds. Define and Compare new models and features immediately.

No worrying about engineering and infra concerns such as serving latency, training-inference skew,complex MLOps etc. Lesser engineering, more ML.

Your team is most productive when they focus on what really matters: your specific vertical business use-case.

## UX

Continuous evaluation to safely experiment with multiple models on live data in shadow mode.

Automagical routing to use the model expected to perform best for that request

Leverage user feedback, implicit and explicit, to continuously improve ML models.

## FAQs

### Is TurboML meant to replace the data scientists in my team?

Absolutely not! On the contrary, our aim is to become the platform that your data scientists love, integrating it more and more into their daily workflows. In fact, all our abstractions are designed to simplify usage for data scientists: no DSLs, minimal learning curve, batch-like semantics, abstractions for all streaming operations, full interoperability with Python, and the ability to bring your own feature stores,models, algorithms, etc.

### If I've already built an ML platform in-house, do you expect me to discard it entirely?

Absolutely not! Interoperability has been our priority from the very beginning, and is heavily incorporated in our design. All components of TurboML are loosely coupled and can be substituted by other services. For instance, if you've built an in-house solution for model predictions or feature engineering, TurboML can be deployed to continue using your implementations for those services while supplementing your solution with capabilities like real-time learning, integrated streaming MLOps, etc.

### If I receive labels or feedback with delays, or not at all, is real-time learning not for me?

Absolutely not! As counter-intuitive as it might sound, instantaneous feedback is not a prerequisite for real-time learning. Consider scenarios like predicting return-to-origin (RTO) for e-commerce or chargebacks for credit cards. In both cases, obtaining labels for individual data points can take days to weeks. However, that doesn't mean we must wait weeks before updating our models. Real-time learning is a continuous process. Even today, we can leverage new information about past predictions to figure out which models were more accurate under what circumstances and further enhance performance.