Einführung in MLOps

22 Feature Stores und Plattformen

Tobias Mérinat
teaching2025@fsck.ch

Lucerne University of
Applied Sciences and Arts
HOCHSCHULE
LUZERN

Department of Information Technology
Lucerne University of Applied Sciences and Arts
6343 Rotkreuz, Switzerland

14. und 15. Februar 2025

Recap: Training-Serving Skew und Data Leakage

Gründe für Training Serving Skew:

- Nicht repräsentative Trainingsdaten
- Falsche Cross Validation (Zeitreihen, Normalisierung über Validierungsdaten hinweg, Grouped Data)
- Daten-Drift
- Fehler bei der Neuimplementierung von Feature Pipelines
- Fehler beim Joinen von Features

Feature Store Hauptaufgaben

Drei Hauptaufgaben:

1. Latenz bei der Bereitstellung während Inferenz zu verringern
2. Persistieren, um historische Daten für das Training zur Verfügung zu stellen
3. Bereitstellung eines Katalogs, um Features durchsuchbar zu machen

Feature Store Funktionalität

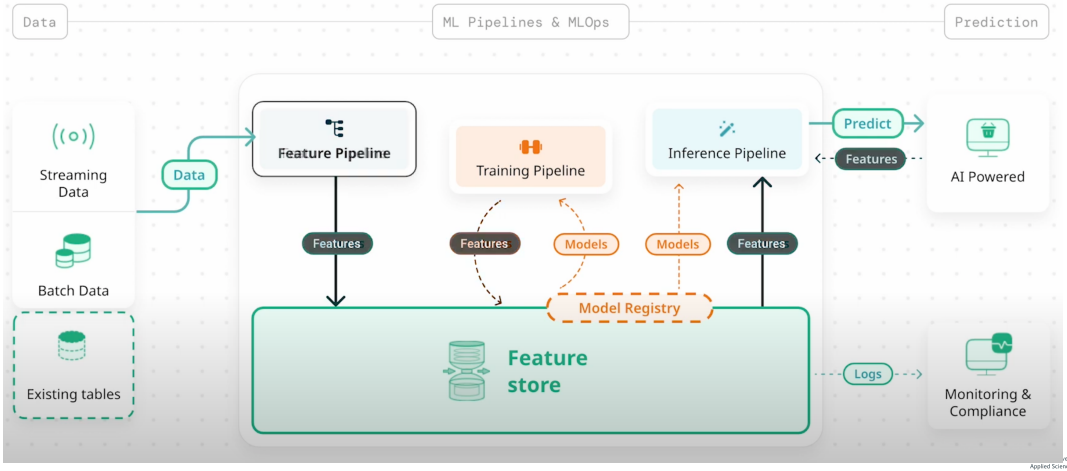
Ein Feature Store muss demnach

- Features via Batch- und Streaming APIs entgegennehmen
- Features effizient speichern
- Feature Management und Discovery ermöglichen
- Features für Batch- und Online Inference zur Verfügung stellen
- Features für Training zur Verfügung stellen

Feature Plattform

- Erlaubt zusätzlich die Berechnung von Features
- Ein Feature Store ist demnach ein Teil einer Feature Plattform
- Feature Plattformen decken die ganze Data Engineering Pipeline bis zum Zeitpunkt der Inferenz ab
- Feature Plattformen als Service, Produkt oder Open Source Software sind aktuell noch wenig verbreitet

Architektur mit Feature Store (hopsworks)



Feature Store für Inferenz

Für die Nutzung während der Inferenz bietet ein Feature Store zwei (bzw. drei) Zugriffsmöglichkeiten bzw. Datenbanken:

- ein Online-Store mit tiefer Latenz (Online-Prediction)
- ein Offline-Store für grosse Mengen historische Daten (Batch-Prediction)
- optional eine Vektor-Datenbank zum Speichern und Finden von Embeddings

Feature Store für Training

- Backfilling und Point-in-Time Correctness
- Feature Katalog mit Discovery
- Generell: Zentrales Repository für Features mit APIs zum Schreiben, Managen und Verwenden

Log-and-wait Ansatz

Einfacher Ansatz, um für Streaming-Daten korrekte Training-Sets zu erstellen:

- Zum Zeitpunkt der Prediction die dann aktuellen Features niederschreiben
- Können leicht mit einem auch später anfallenden Label kombiniert werden
- Nachteil: Braucht Zeit, bis eine für ein Trainingsset brauchbare Menge an Samples geschrieben wurde
- Nachteil: Niedergeschriebenen Features sind Use-Case-spezifisch, sie verfallen, sollte sich die Feature-Logik einmal ändern

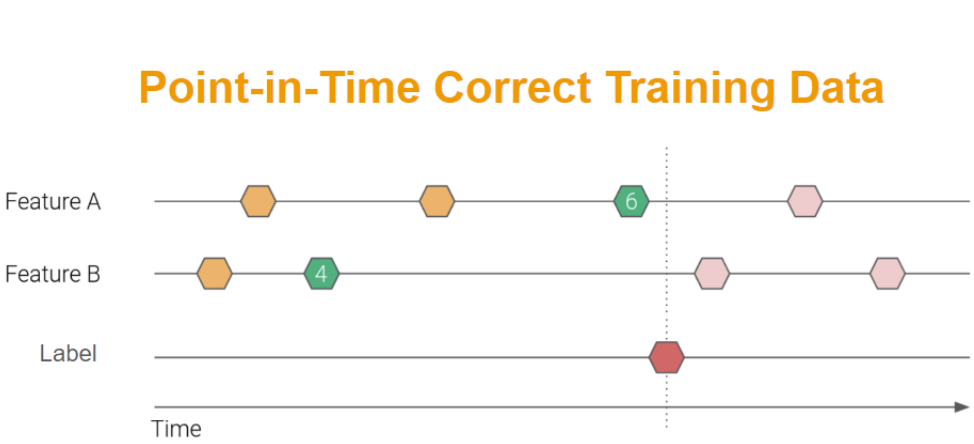
Backfilling

- Der Vorgang, aus historischen Rohdaten ein (Feature-)Datenset zu berechnen
- Also eine Feature Pipeline mit *Von/Bis* Angaben laufen zu lassen, um Features zu generieren
- Sowohl Batch-wie auch Streaming Pipelines sollten Backfilling unterstützen
- Notwendig, um Trainingdaten zu generieren (Retraining oder neues Modell)
- Notwendig, wenn ein neues Feature entwickelt oder ein bestehendes angepasst wird
- Die *gleiche* Feature Pipeline sollte für *live* Daten und für Backfilling verwendet werden
- Ohne Backfilling muss der *log and wait* Ansatz gefahren werden

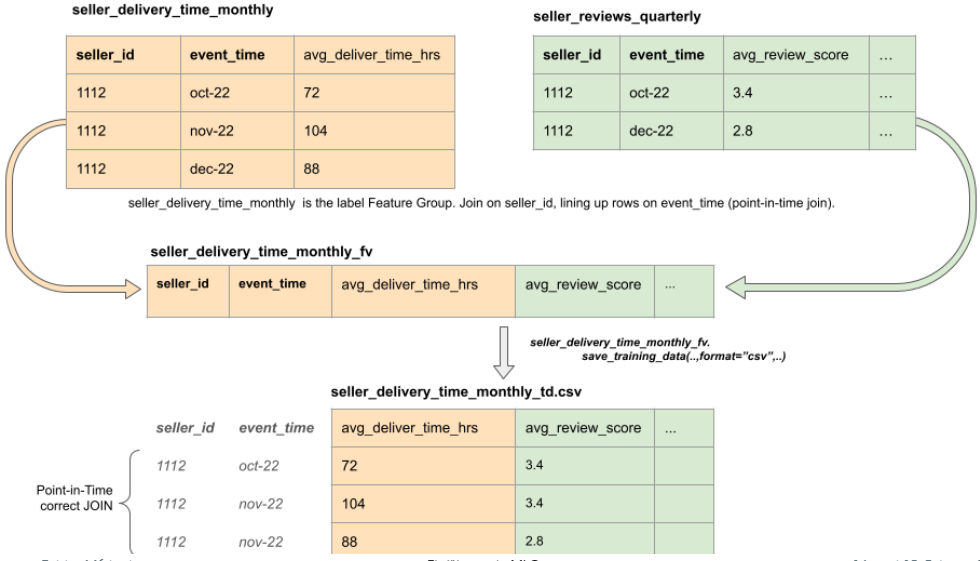
Point-in-Time korrekte Trainingsdaten

- **Point-in-time Correctness**: Die Fähigkeit eines Systems, eine Berechnung genau so durchzuführen, wie sie zu einem beliebigen Zeitpunkt in der Vergangenheit stattgefunden hätte.
- **Point-in-time Correct Join**: Ein temporaler Join zwischen Tabellen, dessen Resultat den Stand beider Tabellen zu einem gegebenen Zeitpunkt widerspiegelt. Die linke Tabelle mit den Labels gibt den Zeitpunkt vor.
- Mit Point-in-time Correct Joins werden beim Zusammenführen von Tabellen keine Daten geleakt.

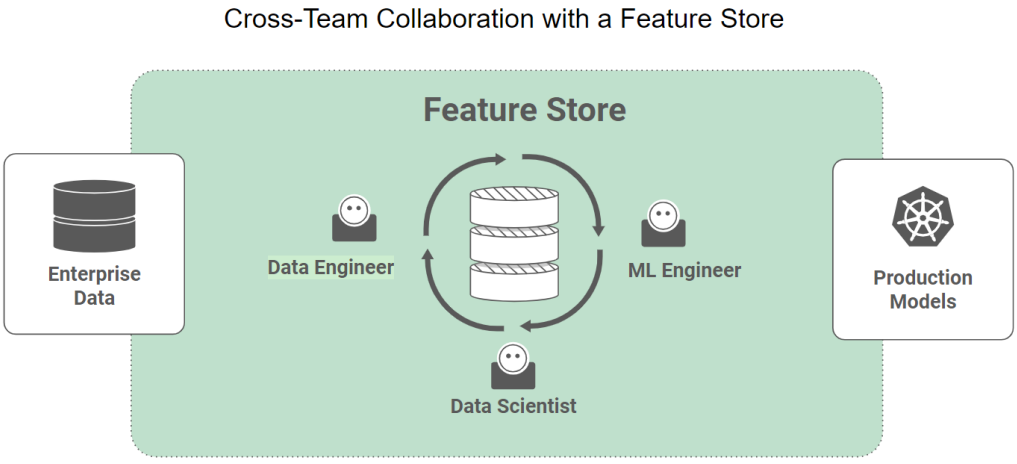
Point-in-time Illustration



Point-in-time Beispiel



Feature Store für kollaborative Entwicklung



Zusammenfassung der Aufgaben eine Feature Store

- Feature-Repository (Offline-Store):
  - API für Verwendung von Features (einfügen, Validierung, Inferenz, Training)
- Online-Store: Ermöglicht Anwendungsfälle, wo Inferenz-Latenz tief sein muss
- Feature-Catalog:
  - Zentrales Repository für Features, Suche, unterstützt Wiederverwendbarkeit
  - Gibt Standards für Benennung von Features, Metadaten und Dokumentation vor
- Point-in-time Correctness:
  - Das Erstellen von Trainingssets wird einfacher und weniger fehleranfällig
  - Schnellere Iterationen durch automatisiertes Backfilling (vs. Log-and-wait Ansatz)
- Data Governance:
  - Feature Lineage und Backfilling hilft bei der Reproduzierbarkeit
  - Zugriffsbeschränkungen (rollenbasierter Zugang, automatisiertes Masking/Anonymisierung, Propagieren von Beschränkungen auf abgeleitete Features)

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN

Feature Store Implementationen

- Hopsworks Featurestore
- Feast
- Feature Form

- Grosse Liste und Vergleiche auf featurestore.org