TRUSTEQ

Cooperative Excellence

Source: www.rawpixel.com licensed under creative commons zero (CC0)

MLOPS IN THE FINANCIAL INDUSTRY

Building a cross-unit ML Cockpit to support your ML governance

TRUSTEQ GmbH
Hansaring 20
50670 Köln

contact@trusteq.de
www.trusteq.de

Monitoring cross-unit ML models is key for organizations

Application teams within the financial industry often use multiple Machine Learning (ML) models in separate development environments and / or with no corporate platform or process in place to holistically monitor them.

This leads to a set of **common challenges** that appear **in cross-unit multi-model management**:

- Missing model and feature transparency between business units.
- Lack of organizing, scheduling, and displaying model drift or degrading application performance.
- Ensuring that same versions of data are always used in overlapping models and that different versions of data do not make one model incompatible with the other.
- Missing reporting on model issues and tracking reported issue status.
- Lack of sufficient concept drifts checks.
- Input data distribution might be different from the one on which the model was trained.
- Insufficient identification of changes related to the model output distribution.
- Tracking changes in feature engineering: changes in the feature extraction for one model often influence the same feature in other models. Without a thorough method of informing and proposing changes, models might use unexpected input for the same feature.

At the same time, it becomes more and more important to have transparency over applied models and their performance to avoid business risks, to ensure auditability and to mitigate reputation risks e. g. due to biased automated decision making. A clear overview on models implemented, features applied and data used helps to make the application of ML more efficient as it unlocks potential for re-use, standardization & industrialization.

Hence it is crucial for companies within the financial industry to have a continuous cross-unit ML model governance process implemented.

A (central) ML Cockpit can be a valuable building block of a governance framework by providing crucial interactive real-time information along three main dimensions:

Model feature monitoring	**Model impact monitoring**	**Model service metadata provision**
Monitor feature changes. Identify most common model inputs and features. Notifications or alerts that are published due to feature changes. Feature / data drift and skew monitoring.	Number of model output subscribers (and link to related pipelines). Summary of actions / decisions taken based on model. Concept drift updates. Provision of model KPIs like: - accuracy, - confusion matrix, - output distribution.	Overview on all ML models. View on model dependencies. Historical statistics (training dataset size, precision, …). Link to documentation, correct git version or model image. Information on parameter definition, model history, current version owner, contact person, model status (dev, QA, prod), last productive run, last model build and deployment.

Increased requirements related to data feeds for ML models

The fast pace of digitalization followed by incorporation of ML and AI into business decision making is a disruptive change for most industries. Daily decisions are more often based on parameters that are derived from data and as data feeds available to the financial industry are expanding from month-to-month there is additional business value in utilizing them as early as possible (e. g. for fraud detection, credit / loan risk assessments, client retention and financial advisory apps).

However, these data must go through set of pre-processing validations and transformations to become valuable & trustable. As soon as a pre-processed data feed is made available in a consolidated way on the corporate data hub, different teams can start utilizing it for their own set of use cases.

In this paper we will focus on the challenges that appear during the management of a diverse ML ecosystem and provide examples of solutions / methods how to solve them.

Managers need to create transparency in a complex environment when they approach ML governance

To be successful, three dimensions need to be managed actively when it comes to ML models:

1. **Clear model ownership definition** (within a complex net of relations & interdependencies)

2. Thorough **transparency over users & producers** (incl. potential model overlap & data flows) of ML models and ML data (meaning input as well as output data of ML models)

3. Setup of a **cross-unit ML ecosystem governance process**

To manage any ML process upgrades or new ML process implementation, it is crucial to have a structured way of monitoring the changes. The IT ecosystems within the financial industry are complex and diverse which leads to different ways of implementing similar processes.

Managers responsible for leading development teams (especially teams that are utilizing ML models) have challenging times when steering and synchronizing processes over multiple departments covering a bunch of stakeholders (see exhibit 1 for a theoretical, but realistic stakeholder diagram below).

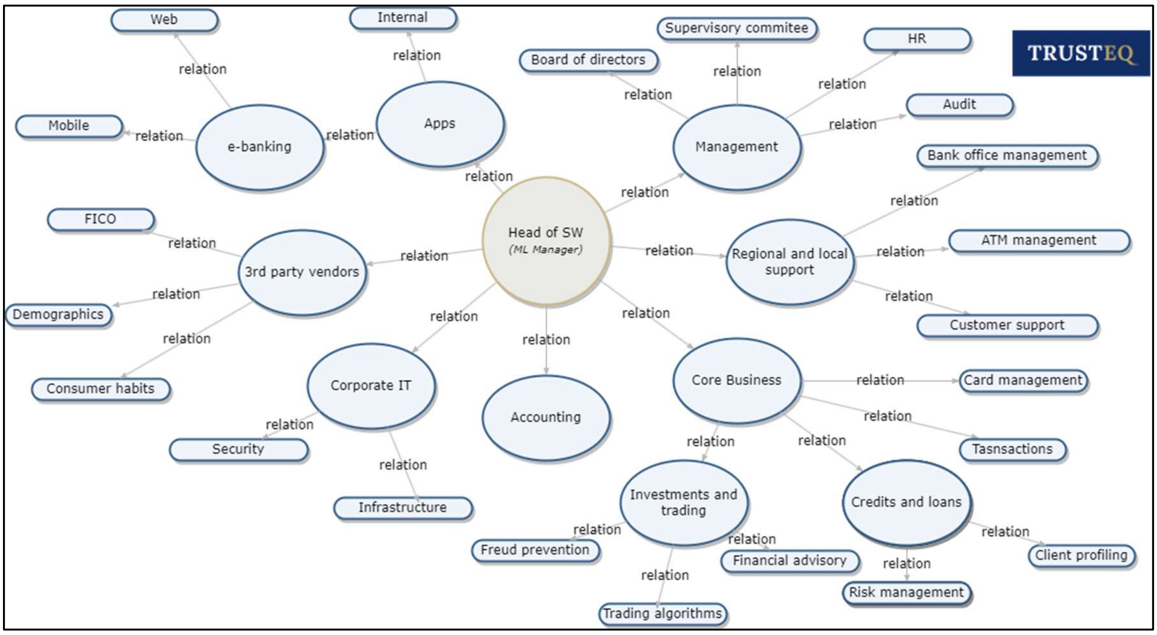


Exhibit 1 Stakeholder diagram from ML manager perspective

Given this complexity, core challenges comprise gathering & structuring information about:

- Utilization of the company data hub by different teams using ML models
- Potential ML model overlaps
- Data flow and interfaces between ML applications

To support this and the overall ML governance process, one option is to include a "ML Governance Layer" between the data hub (referring to any kind of data storage as data warehouses, data lakes etc) that can be used as basis for a ML Cockpit as a starting point for transparency (see exhibit 2).

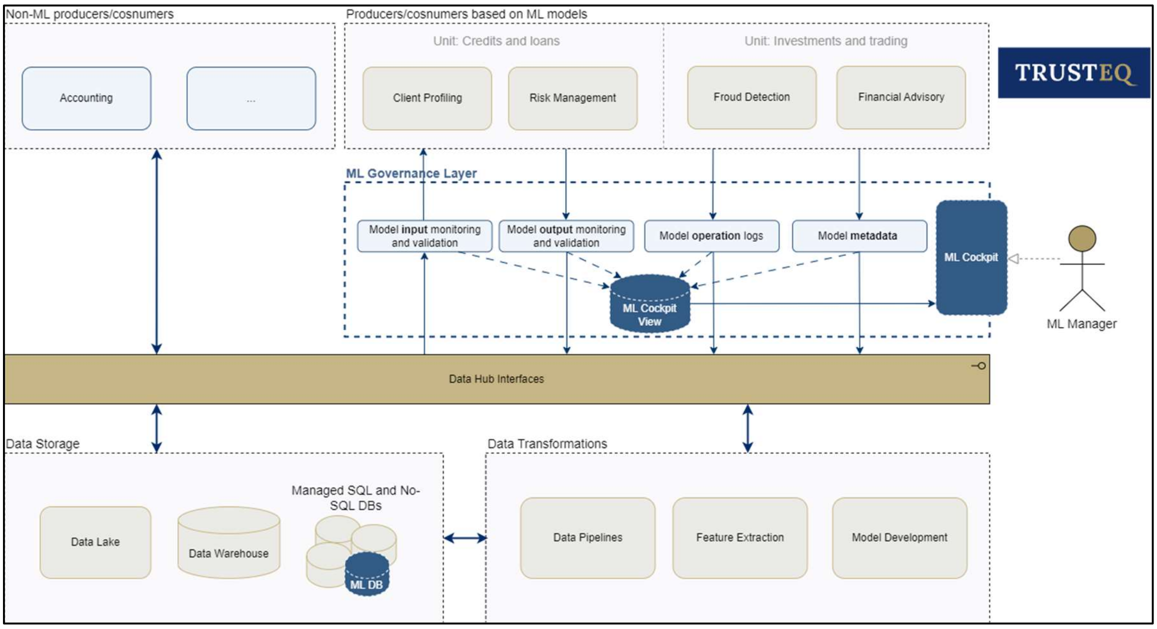


Exhibit 2 High-level ML solution architecture diagram

Continuous management of the cross-unit ML ecosystem provides significant impact to ML risk mitigation & industrialization

Establishing a continuous cross-unit ML governance process supports a synchronized and prompt decision taking throughout the whole company structure.

Compared to rule-based modelling, **ML model development is a continuous process** by heart. Especially since most of the models are constantly re-trained on latest datasets. Each stage of model development is usually done multiple times since deploying the first model version within production. Hence the model development lifecycle consists of several phases (see exhibit 3).

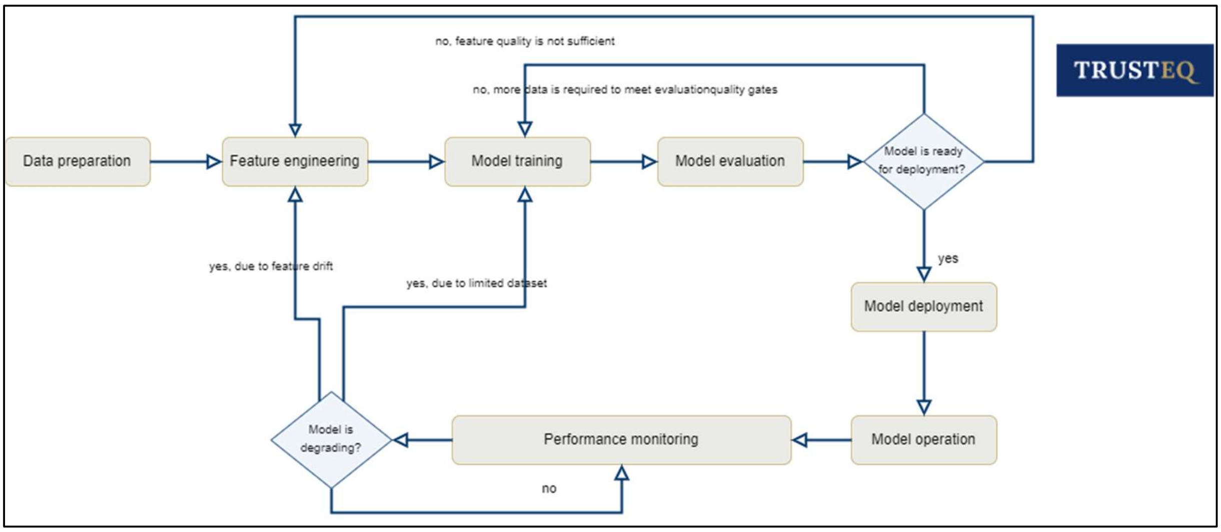

no, feature quality is not sufficient
no, more data is required to meet evaluationquality gates
TRUSTEQ
Data preparation
Feature engineering
Model training
Model evaluation
Model is ready for deployment?
yes
yes, due to feature drift
yes, due to limited dataset
Model deployment
Model is degrading?
Performance monitoring
Model operation
no

Exhibit 3 ML model development lifecycle

Depending on maturity of the lifecycle, models can be classified in different stages such as Design, PoC, Production Ready, In Production and Deprecated which is a crucial information required for ML model governance.

Often the transition from development to production is forced due to business pressure and ML best practices are missing. Consequential potential high-level issues are low process automation and the related risks & costs of maintenance of manual steps. Furthermore, all models degrade over time. To mitigate these risks, an important part of the model lifecycle is a **continuous model performance monitoring**.

Unfortunately, monitoring model performance in production cannot only be based on accuracy metrics. The main issue is that often model accuracy cannot immediately be validated in production. Several common models within the financial industry are exactly fitting in that group: card fraud detection, cross-selling, credit risk prediction, estimated asset / property value, client illness risk etc.

Automating a continuous model monitoring process helps to identify different types of feature / data / model skew and drift before they do harm. Monitoring model skew and drift is a well-known problem in the industry. All major cloud service providers have already implemented additional services and metrices to handle drifts and skews out of the box. Due to that we will discuss how to consolidate and integrate such out of the box solutions into a single ML cockpit rather than focusing on details of performance metrics evaluations algorithms and types. You can find a basic architecture example on how data drift can be detected using AWS Sage Maker on the Amazon webpage if you are interested: https://aws.amazon.com/blogs/architecture/detecting-data-drift-using-amazon-sagemaker/.

In last few years the number of issues is exponentially growing as ML is emerging in the financial industry. Find some **examples and cases where model governance could have helped to mitigate the risk beforehand**:

- 'Sexist' credit card case of a US company
  Gender biased model issue was identified too late. Anonymization of gender and feature importance overview were missing to prevent this scenario.

- Customer classification training dataset lag
  When the customer classification was changed (the business decided to add a new customer group) some of the model owners within the bank were not notified. This resulted in running some models in production that were trained on a deprecated dataset for one of the most significant features: the customer segments.

- Multi-state regulatory compliance
  For US credit models alone, one must make sure to adhere to several regulations like the Fair Housing Act, Consumer Credit Protection Act, Fair Credit Reporting Act, Equal Credit Opportunity Act, Fair and Accurate Credit Transactions Act who might directly or indirectly impact ML model compliance. It’s also possible for an AI model to be deployed in multiple territories where one jurisdiction has more conservative guidelines or completely different ones.

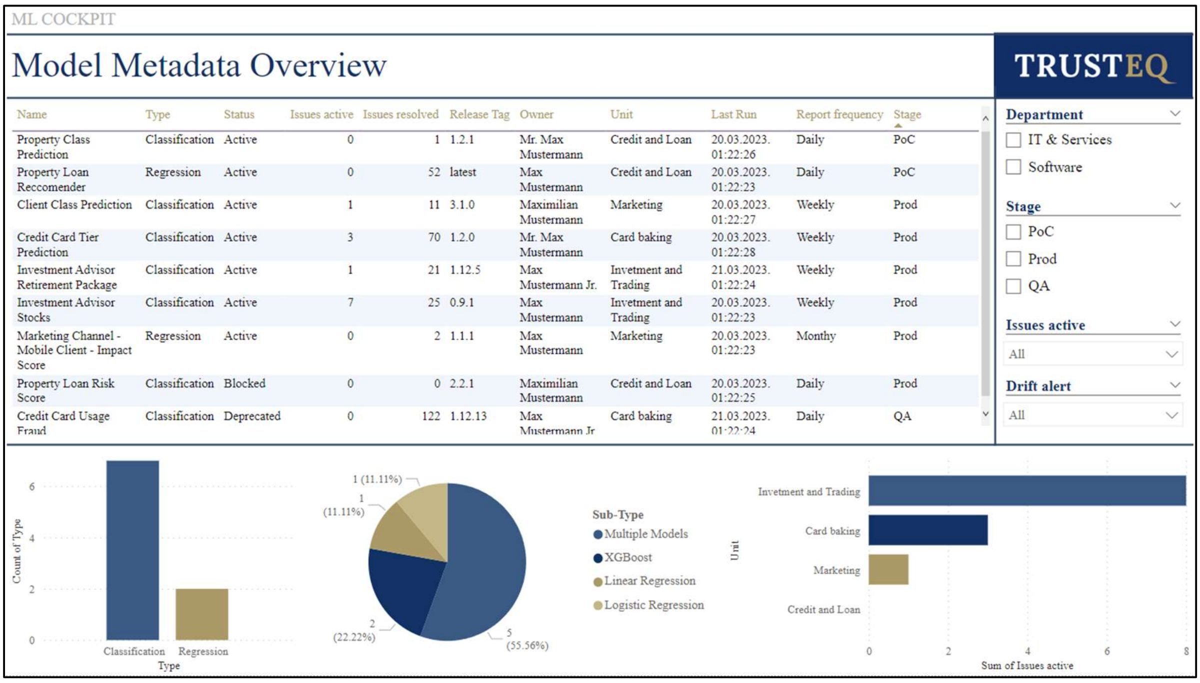
To mitigate these kind of challenges cross-unit multi-model governance requires implementation of additional processes, features and culture that may include:

- Model metadata extraction pipelines
- Well maintained model version documentation
- Unified model log handling
- Unified model validation result messages
- Model classification
- SLA classification
- Notification publishing during feature engineering
- Model deprecation process and many other required overhead features on top of single ML model development

A ML Cockpit can improve transparency significantly

The purpose of the Cockpit is to provide interactive and up-to-date ML monitoring views to anybody holding a ML manager role. It connects directly and only to a ML Cockpit Backend via API or SQL view connection. To cover the relevant requirements the ML Cockpit UI supports with six different dashboard sheets connected with drill down feature:

- Landing page
- Model metadata overview (see exibit 4)
- Feature monitoring
- Alerts and notifications
- Single model overview
- Model history



ML COCKPIT

Model Metadata Overview

Name	Type	Status	Issues active	Issues resolved	Release Tag	Owner	Unit	Last Run	Report frequency	Stage
Property Class Prediction	Classification	Active	0	1	1.2.1	Mr. Max Mustermann	Credit and Loan	20.03.2023. 01:22:26	Daily	PoC
Property Loan Reccomender	Regression	Active	0	52	latest	Max Mustermann	Credit and Loan	20.03.2023. 01:22:23	Daily	PoC
Client Class Prediction	Classification	Active	1	11	3.1.0	Maximilian Mustermann	Marketing	20.03.2023. 01:22:27	Weekly	Prod
Credit Card Tier Prediction	Classification	Active	3	70	1.2.0	Mr. Max Mustermann	Card baking	20.03.2023. 01:22:28	Weekly	Prod
Investment Advisor Retirement Package	Classification	Active	1	21	1.12.5	Max Mustermann Jr.	Invetment and Trading	21.03.2023. 01:22:24	Weekly	Prod
Investment Advisor Stocks	Classification	Active	7	25	0.9.1	Max Mustermann	Invetment and Trading	20.03.2023. 01:22:23	Weekly	Prod
Marketing Channel - Mobile Client - Impact Score	Regression	Active	0	2	1.1.1	Max Mustermann	Marketing	20.03.2023. 01:22:23	Monthy	Prod
Property Loan Risk Score	Classification	Blocked	0	0	2.2.1	Maximilian Mustermann	Credit and Loan	20.03.2023. 01:22:25	Daily	Prod
Credit Card Usage Fraud	Classification	Deprecated	0	122	1.12.13	Max Mustermann Jr	Card baking	21.03.2023. 01:22:24	Daily	QA

Department
□ IT & Services
□ Software

Stage
□ PoC
□ Prod
□ QA

Issues active: All

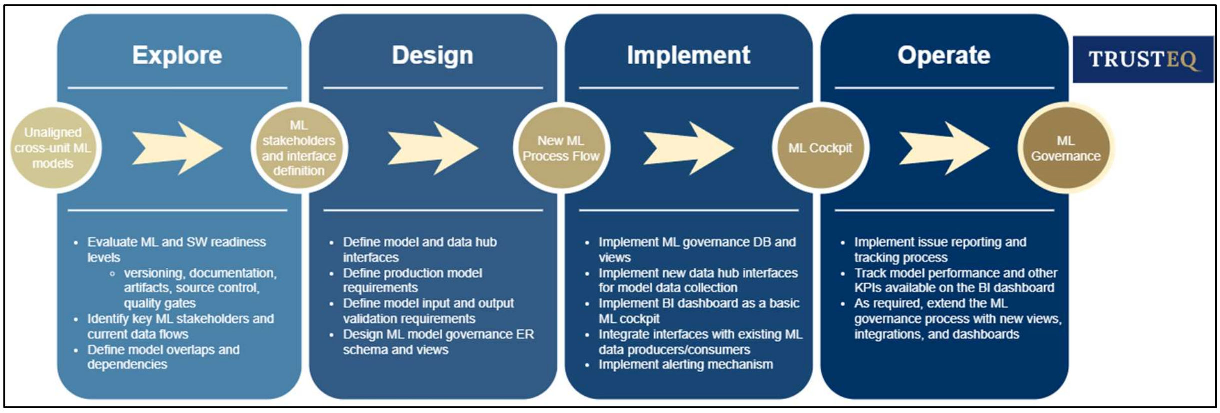
Drift alert: All

Exibit 4: Model Metadata Overview part of a ML Cockpit UI

To Implement the ML model Cockpit, we break the process down in three main phases before we bring it to operation:

- Definition of process steps
- Data flow design and interface integration
- Design of the ML Cockpit UI

It is important to understand that key to continuous delivery of high production quality is to regularly rethink and repeat processes from the first stage. Process automation heavily depends on software readiness level that should be identified in an exploration phase along with stakeholders, ML overlaps & dependencies.



TRUSTEQ
Unaligned cross-unit ML models
ML stakeholders and interface definition
New ML Process Flow
ML Cockpit
ML Governance

Explore

- Evaluate ML and SW readiness levels
  - versioning, documentation, artifacts, source control, quality gates
- Identify key ML stakeholders and current data flows
- Define model overlaps and dependencies

Design

- Define model and data hub interfaces
- Define production model requirements
- Define model input and output validation requirements
- Design ML model governance ER schema and views

Implement

- Implement ML governance DB and views
- Implement new data hub interfaces for model data collection
- Implement BI dashboard as a basic ML cockpit
- Integrate interfaces with existing ML data producers/consumers
- Implement alerting mechanism

Operate

- Implement issue reporting and tracking process
- Track model performance and other KPIs available on the BI dashboard
- As required, extend the ML governance process with new views, integrations, and dashboards

Exhibit 5 Process stages for implementation of ML Governance

After ML governance process steps are aligned between ML stakeholders, it is time to start the process design and implementation. The design step requires collecting solution architecture diagrams from different units to derive common interfaces and data flows. To implement the cross-unit ML governance architecture we introduce a new "ML Governance Layer" between ML applications (also referred as ML producers / consumers) and the data hub (interfaces which ML applications use to access and store data).

For the ML Cockpit backend to serve views utilized by dashboard sheets we needed to create a central ML database which is represented as a part of the data hub. We will not go into details about the ER diagram, but rather explain backend integrations done by the four different ML governance service types shown on exhibit 6:

Input and output services are subscribed on to each ML model input (feature) and output (prediction). Their core business logic covers continuous model performance monitoring and the real-time informing of all required stakeholders via notification / alerting system. This is as well a good place to integrate existing cloud ML or open-source services.

Operation service is representing a ML extension of, in most banking & insurance companies existing, log management and APM monitoring infrastructure.

Metadata service has the task of tracking model metadata, features and statistics. It represents the central service of the ML Governance Layer as it provides information about all cross-unit models in one place, which enables easier dependency tracking and decision making.

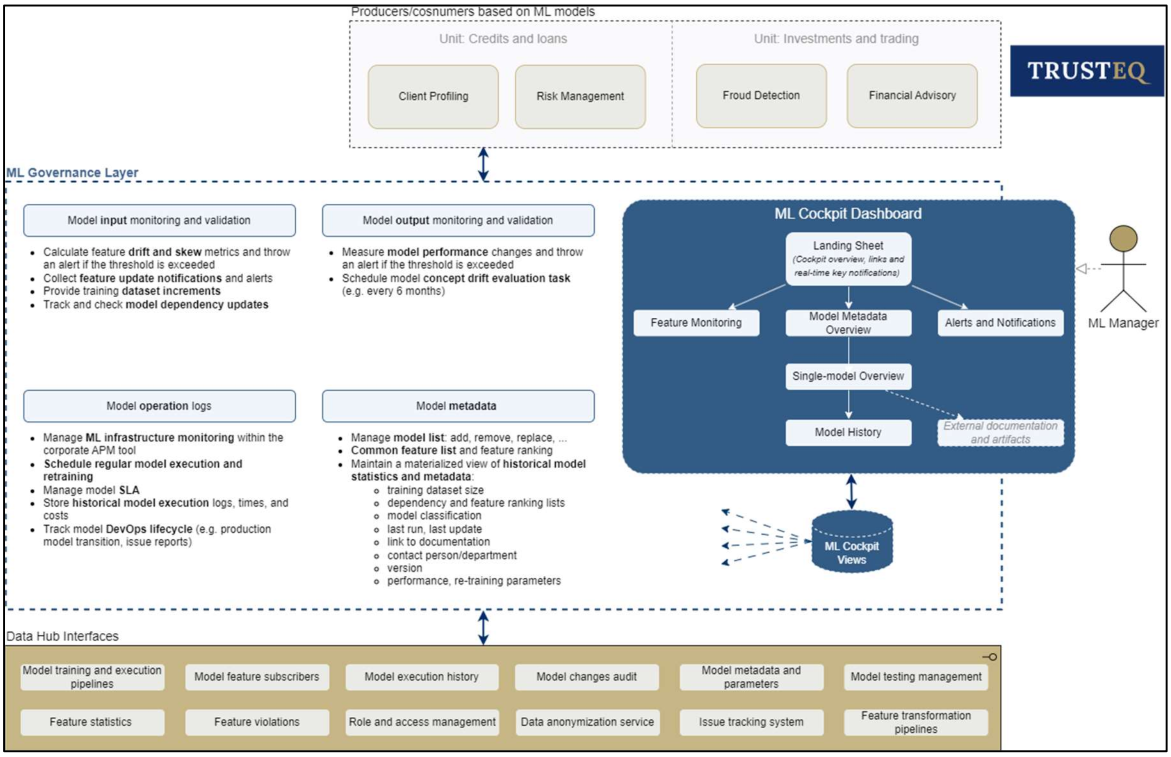

Producers/cosnumers based on ML models
Unit: Credits and loans
Client Profiling
Risk Management
Unit: Investments and trading
Froud Detection
Financial Advisory
TRUSTEQ
ML Governance Layer
Model **input** monitoring and validation
Calculate feature **drift and skew** metrics and throw an alert if the threshold is exceeded
Collect **feature update notifications** and alerts
Provide training **dataset increments**
Track and check **model dependency updates**
Model **output** monitoring and validation
Measure **model performance** changes and throw an alert if the threshold is exceeded
Schedule model **concept drift evaluation task** (e.g. every 6 months)
Model **operation** logs
Manage **ML infrastructure monitoring** within the corporate APM tool
Schedule regular model execution and retraining
Manage model **SLA**
Store **historical model execution** logs, times, and costs
Track model **DevOps lifecycle** (e.g. production model transition, issue reports)
Model **metadata**
Manage **model list**: add, remove, replace, ...
Common feature list and feature ranking
Maintain a materialized view of **historical model statistics and metadata**:
training dataset size
dependency and feature ranking lists
model classification
last run, last update
link to documentation
contact person/department
version
performance, re-training parameters
ML Cockpit Dashboard
Landing Sheet
(Cockpit overview, links and real-time key notifications)
Feature Monitoring
Model Metadata Overview
Alerts and Notifications
Single-model Overview
Model History
External documentation and artifacts
ML Manager
ML Cockpit Views
Data Hub Interfaces
Model training and execution pipelines
Model feature subscribers
Model execution history
Model changes audit
Model metadata and parameters
Model testing management
Feature statistics
Feature violations
Role and access management
Data anonymization service
Issue tracking system
Feature transformation pipelines

Exhibit 6 ML Governance Layer

Summary

In this whitepaper we walked through some main challenges in the enterprise ML model governance process. Since ML model governance covers a wide range of topics, our proposed ML Cockpit solution provides a solid start point for a centralized & holistic view on the usage of ML models within the company.

However, there are other business critical aspects which are not in the focus of this paper as data sharing and anonymizing policies (access management and GDPR), continuous model development and integration, ensuring ML decision reproducibility & accountability and more of which all must be considered when designing ML infrastructure interfaces.

Nevertheless, having a centralized cross-unit model governance enables ML managers to have synchronization and trust into quality of production applications that utile ML models.

We presented a process how a ML Cockpit solution could be developed and integrated into bigger finance IT ecosystem.

Looking long-term, the number of ML model will grow within all units in companies of the financial industry and therefore having a solution like the proposed ML Cockpit in place should be a high priority for all organizations.

For more details get in contact with us

Ante Gojsalic
Lead Data Scientist
ante.gojsalic@trusteq.de

Nils Gilles
Head of Data & Analytics
nils.gilles@trusteq.de