Manifold

Model-agnostic visual debugging tool for ML

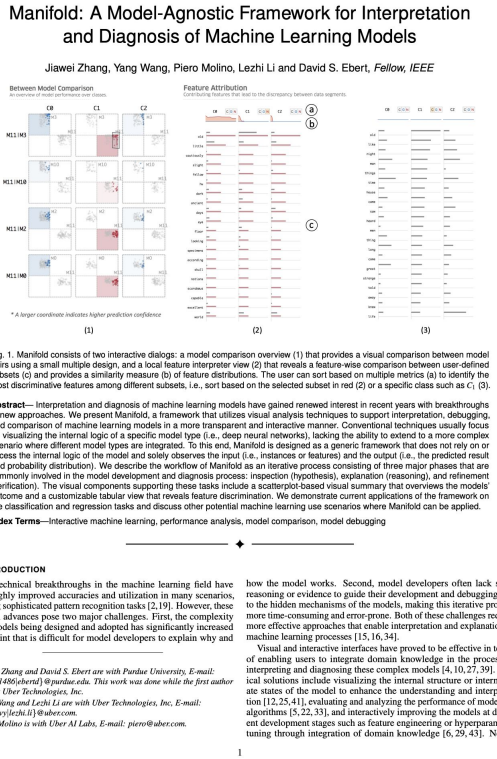
Lezhi Li, Yunfeng Bai, Yang Wang

OpML'19 May 20, 2019

Uber



Manifold: model-agnostic visual debugging tool for ML



Manifold: A Model-Agnostic Framework for Interpretation and Diagnosis of Machine Learning Models

Jiawei Zhang, Yang Wang, Piero Molino, Lezhi Li and David S. Ebert, *Fellow, IEEE*

Between Model Comparison

Feature Attribution

(1) (2) (3)

Fig. 1. Manifold consists of two interactive dialogs: a model comparison overview (1) that provides a visual comparison between model pairs using a small multiple design, and a local feature interpreter view (2) that reveals a feature-wise comparison between user-defined subsets (c) and provides a similarity measure (b) of feature distributions. The user can sort based on multiple metrics (a) to identify the most discriminative features among different subsets, i.e., sort based on the selected subset in red (2) or a specific class such as C_1 (3).

Abstract— Interpretation and diagnosis of machine learning models have gained renewed interest in recent years with breakthroughs in new approaches. We present Manifold, a framework that utilizes visual analysis techniques to support interpretation, debugging, and comparison of machine learning models in a more transparent and interactive manner. Conventional techniques usually focus on visualizing the internal logic of a specific model type (i.e., deep neural networks), lacking the ability to extend to a more complex scenario where different model types are integrated. To this end, Manifold is designed as a generic framework that does not rely on or access the internal logic of the model and solely observes the input (i.e., instances or features) and the output (i.e., the predicted result and probability distribution). We describe the workflow of Manifold as an iterative process consisting of three major phases that are commonly involved in the model development and diagnosis process: inspection (hypothesis), explanation (reasoning), and refinement (verification). The visual components supporting these tasks include a scatterplot-based visual summary that overviews the models' outcome and a customizable tabular view that reveals feature discrimination. We demonstrate current applications of the framework on the classification and regression tasks and discuss other potential machine learning use scenarios where Manifold can be applied.

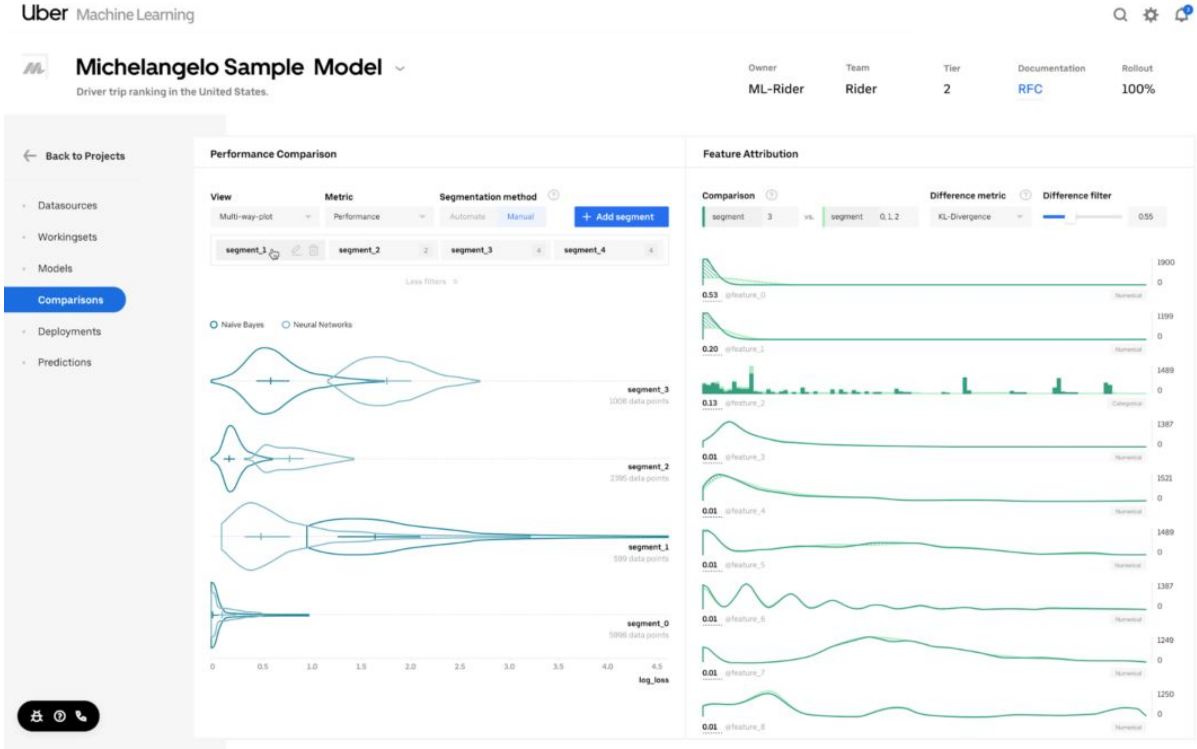
Index Terms—Interactive machine learning, performance analysis, model comparison, model debugging

1 Introduction

Recent technical breakthroughs in the machine learning field have led to highly improved accuracies and utilization in many scenarios, including sophisticated pattern recognition tasks [2, 19]. However, these technical advances pose two major challenges. First, the complexity of the models being designed and adopted has significantly increased to the point that is difficult for model developers to explain why and how the model works. Second, model developers often lack solid reasoning or evidence to guide their development and debugging due to the hidden mechanisms of the models, making this iterative process more time-consuming and error-prone. Both of these challenges require more effective approaches that enable interpretation and explanation of machine learning processes [15, 16, 34].

Visual and interactive interfaces have proved to be effective in terms of enabling users to integrate domain knowledge in the process of interpreting and diagnosing these complex models [4, 10, 27, 39]. Typical solutions include visualizing the internal structure or intermediate states of the model to enhance the understanding and interpretation [12, 25, 41], evaluating and analyzing the performance of models or algorithms [5, 22, 33], and interactively improving the models at different development stages such as feature engineering or hyperparameter tuning through integration of domain knowledge [6, 29, 43]. Never-

- *Jiawei Zhang and David S. Ebert are with Purdue University, E-mail: {zhan1495|ebertd}@purdue.edu. This work was done while the first author was at Uber Technologies, Inc.*
- *Yang Wang and Lezhi Li are with Uber Technologies, Inc, E-mail: {gnavyl|lezhi.li}@uber.com.*
- *Piero Molino is with Uber AI Labs, E-mail: piero@uber.com.*



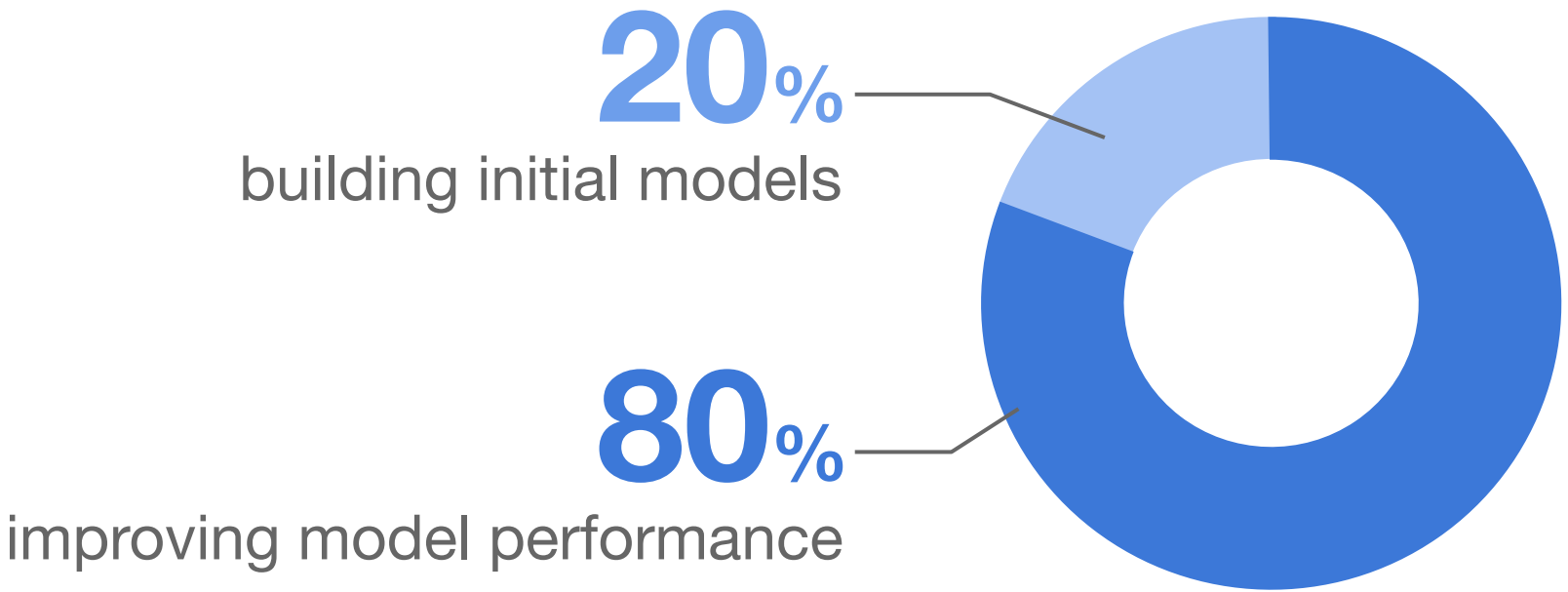
Agenda

01 Motivation
02 Workflow
03 Integration

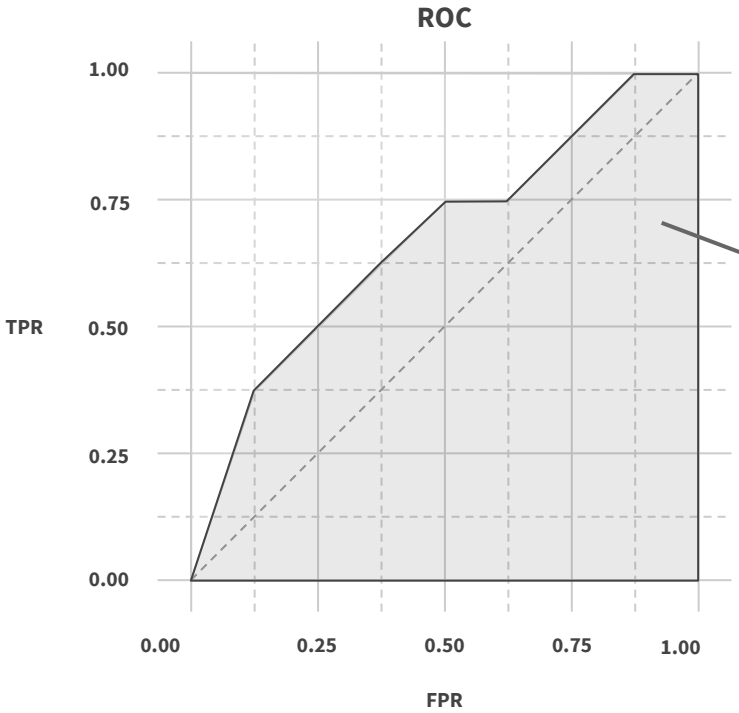
Manifold
https://eng.uber.com/manifold

Michelangelo
https://eng.uber.com/scaling-michelangelo

Importance of Model Debugging



Inadequacy of performance metrics



AUC: 0.81

Great! but what's Next?

not actionable...

Model Interpretability?

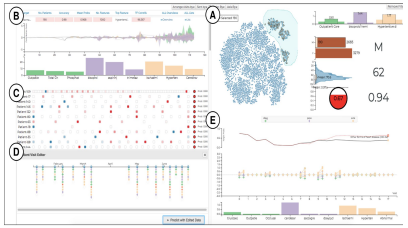
Interpretable Machine Learning

Internal Structure
Intermediate State
Mimic Learning
Interactive Hyperparameter tuning
Model Performance Visualization
…

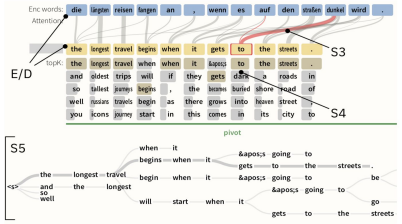
Focus on single (family of) model(s)
Too much love for deep models
: /



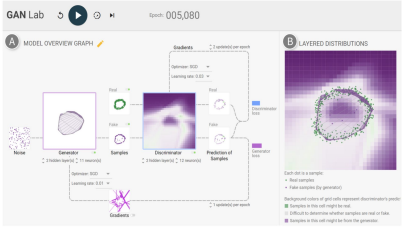
DQNViz [Wang 2019]



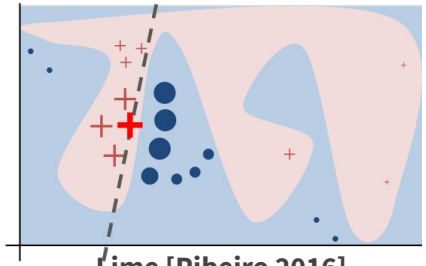
RetinaVis [Kwon 2019]



Seq2Seq-Vis [Strobelt 2019]



GAN Lab [Kahng 2019]



Lime [Ribeiro 2016]



Shap [Lundberg 2017]

Model-agnostic?

Interpretable Machine Learning

Internal Structure
Intermediate State
Mimic Learning
Interactive Hyperparameter tuning
Model Performance Visualization
…

Model-specific interpretability

will not scale

from the operational perspective

Machine Learning at Uber

Predicting Supply & Demand
One-click Chat
Restaurant Recommendation
Sensor Intelligence and Location
Autonomous Vehicles
Support Ticket Routing
Incentive Optimizing
Fraud Detection
Financial Planning
…

ML Revisited

$$y = f^{*}(x) + \varepsilon$$

Model space => Data space

What went wrong with the model?

Which data **subset** did the **model(s)** make mistake?

Why the model made such mistake?

Which **feature** has contributed to the mistakes?

Manifold workflow

1. Compare

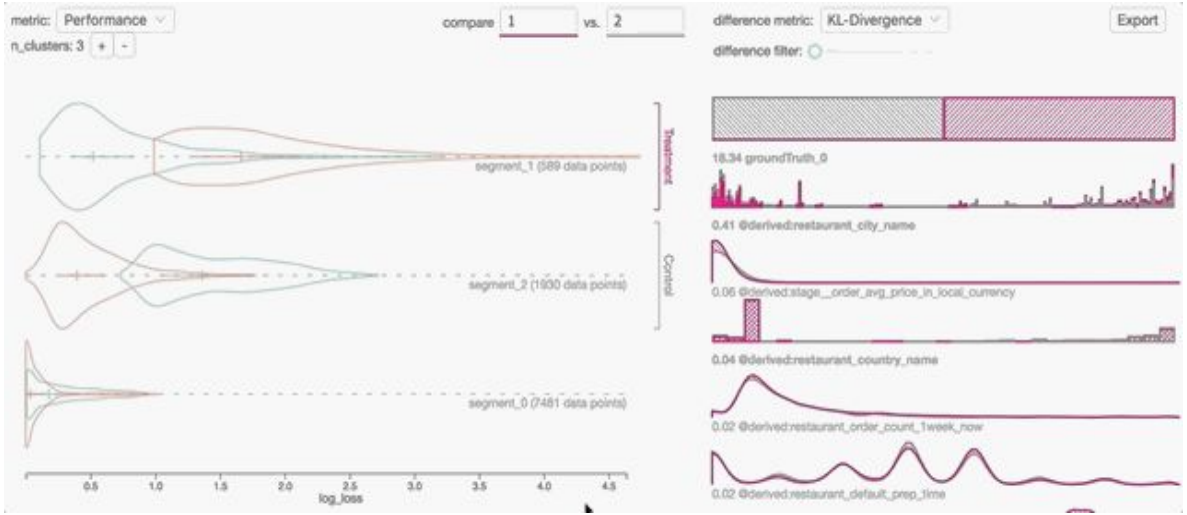
Given a dataset with the output from one or more ML models, Manifold compares and highlights performance differences across **models** or **data subsets**

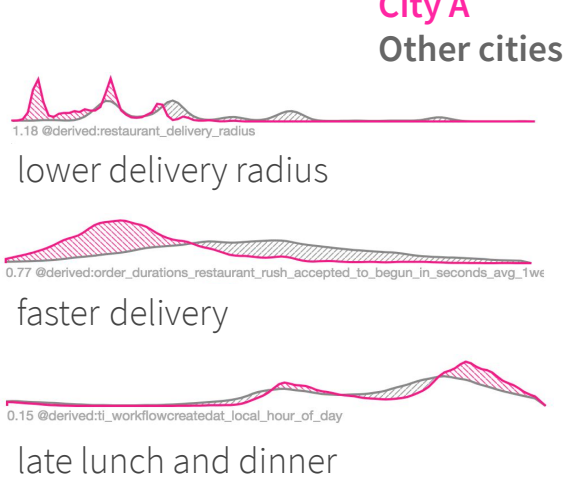
2. Slice

Users can select data **subsets of interest** based on model performance for further inspection

3. Attribute

Manifold then highlights **feature distribution** differences between the selected data subsets, helping users find the reasons behind the performance outcomes





Manifold workflow

1. Compare

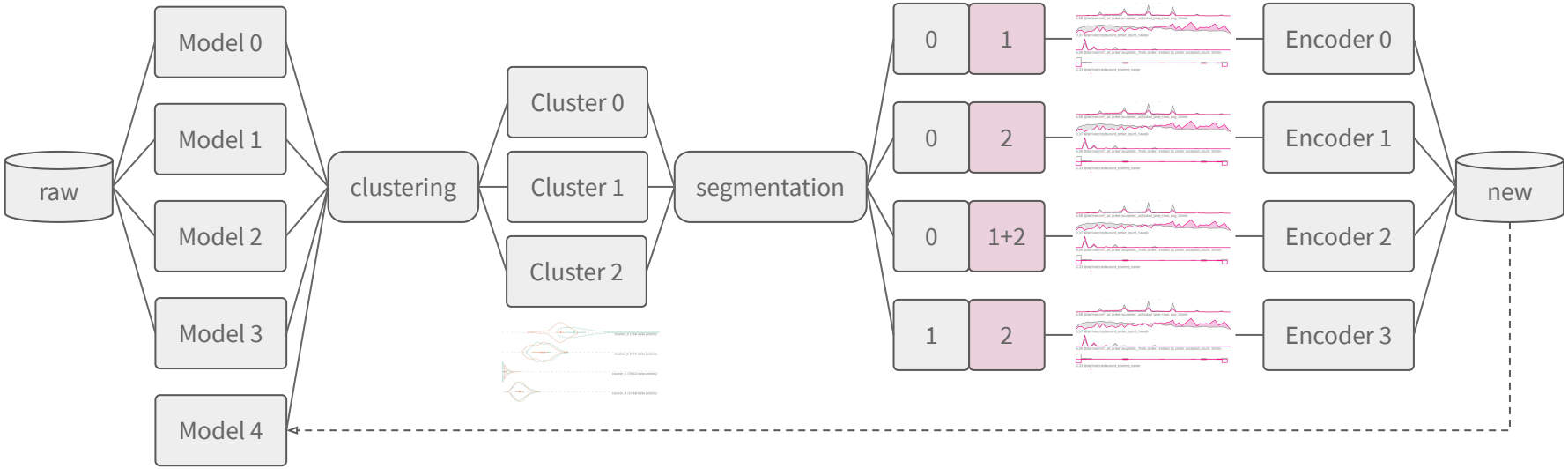
Given a dataset with the output from one or more ML models, Manifold compares and highlights performance differences across **models** or **data subsets**

2. Slice

Users can select data **subsets of interest** based on model performance for further inspection

3. Attribute

Manifold then highlights **feature distribution** differences between the selected data subsets, helping users find the reasons behind the performance outcomes



Challenges

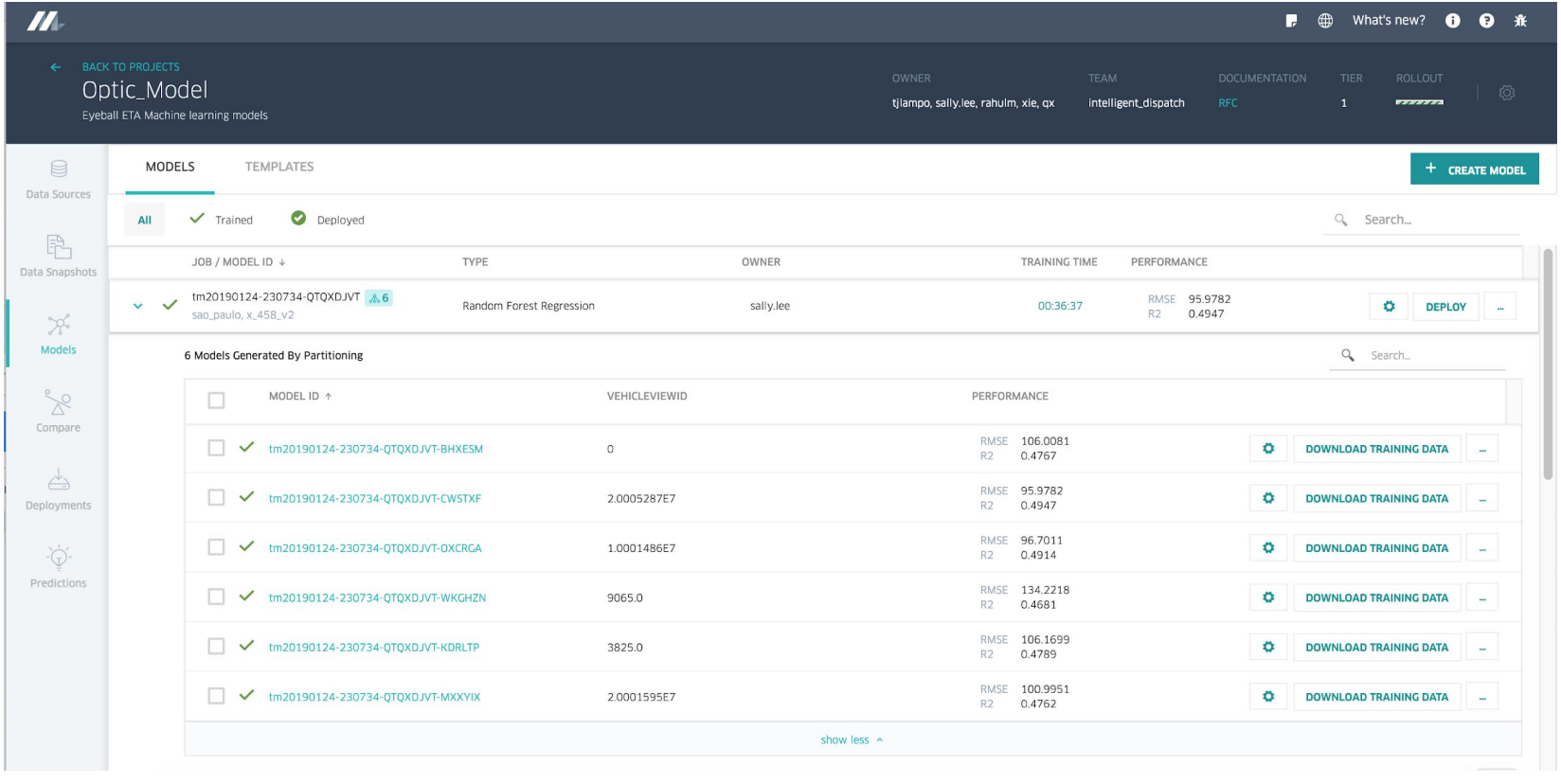
01 Interactivity √

02 Scalability ?

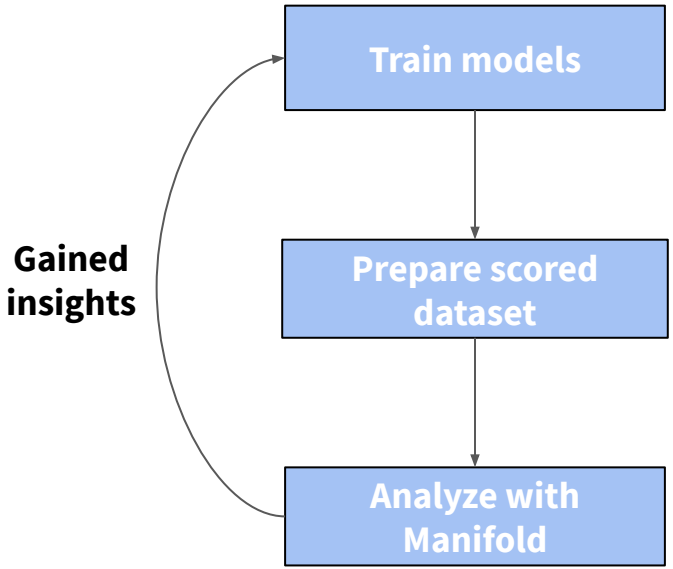
Operational
Computational

=> Michelangelo

Integration with Michelangelo



Integration with Michelangelo



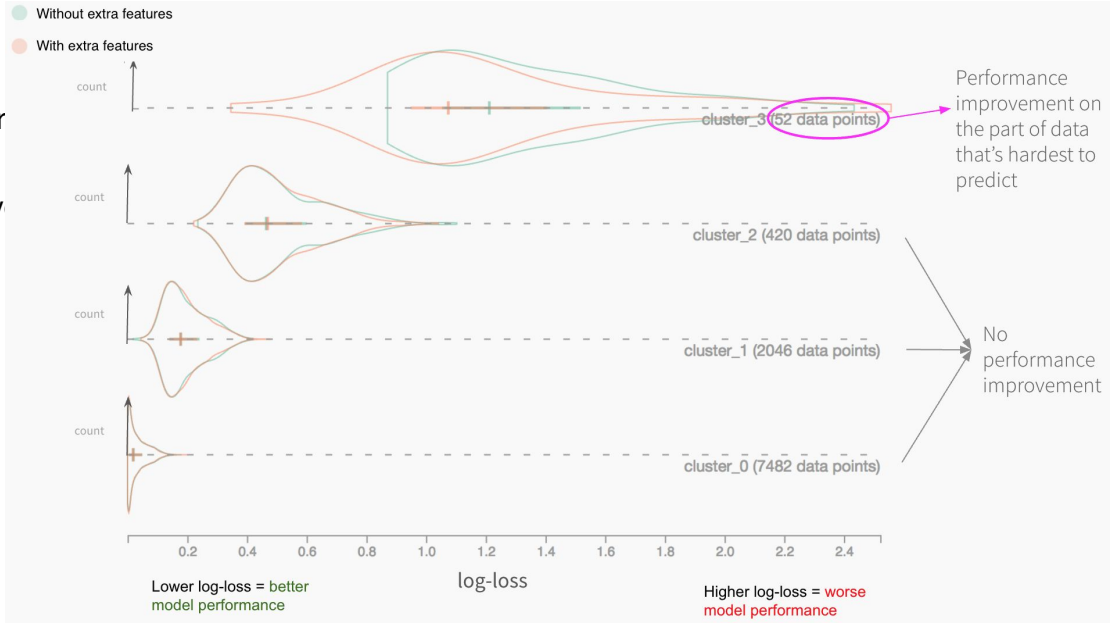
Integration with Michelangelo

Case Study:

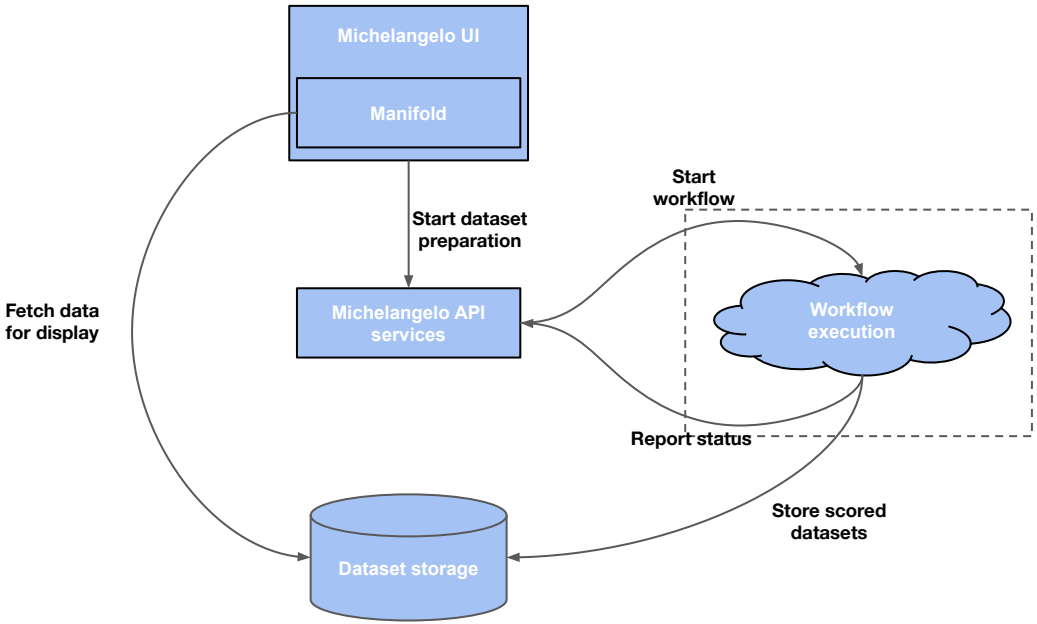
Identify useful features

- A team at Uber is evaluating the value of an extr features
- Adding new features did not change model's ov performance
- Are the new features still worth adding?

- **Yes, improvement on hard-to-predict data**

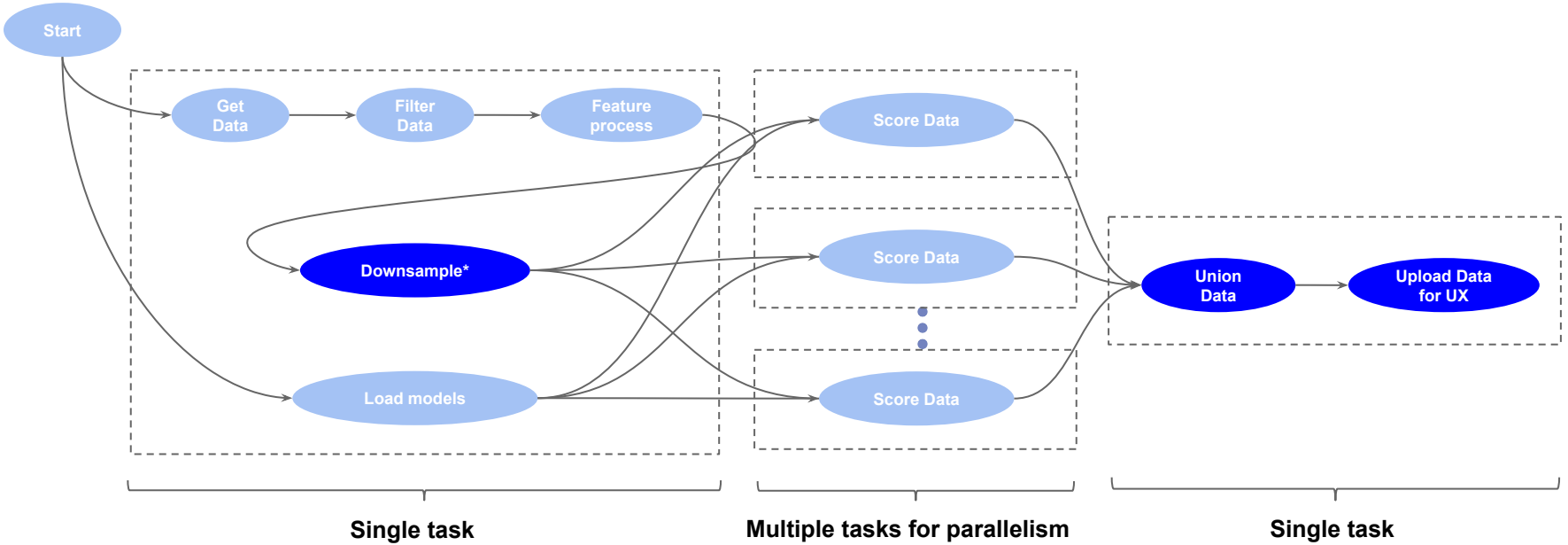


Architecture



Manifold Workflow

- Expressing Manifold comparison dataset generation with MA physical workflow



*: consistent sampling

Thank You!

Questions?

Consistent Sampling

How to consistently downsample rows across Dataframes?

- No native support for consistent sampling in Spark
- Filter based on hash value

