Custom ML Workflow

Treasure Data Academy

October 15-16, 2024

Speakers

Dilyan Kovachev

Head of Machine Learning
& Analytics Engineering

TREASURE
DATA

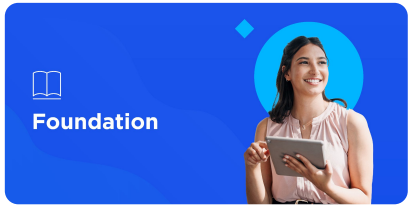
Yish Lim

Senior Machine Learning
Engineer

TREASURE
DATA

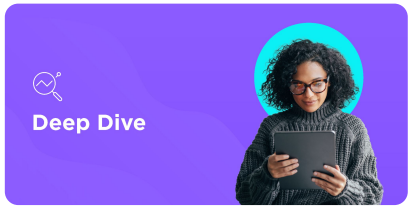
AI/ML Specialist Certification

Complete 1 of 3 exam prerequisites today!



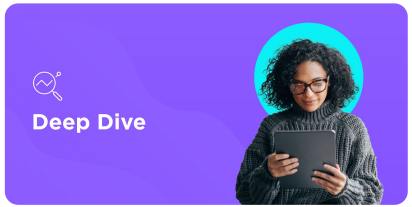
Audience Studio: Predictive Scoring

Mandatory | EN | E-Learning | 12m



Building a Custom ML Workflow

Mandatory | EN | E-Learning | 25m



ML for Marketing: Use Cases & Solutions

Mandatory | EN | E-Learning | 35m

Certification on Demand: “TD Expert: AI/ML Specialist”

Mandatory | EN | E-Learning | 45m

Exam

The Lifecycle of Productizing Custom ML Workflow in TD

Overview of Custom ML Workflow

This diagram illustrates one of the many frameworks that exist for building and deploying Machine Learning models. For best practices, an ML framework should include these processes.

1. DATA PREPARATION › **2. MODEL TRAINING** › **3. MODEL EVALUATION** › **4. DEPLOYMENT**

This includes Exploratory Data Analysis (EDA), **feature engineering** – which includes creating new features, and scaling, as well as **feature selection** – deciding what features to include in the model.

In TD, there are many options for modeling. We'll discuss the best practices for different use-cases to optimize your modeling.

This includes storing and tracking your model's evaluation metrics to either **tune and iterate** on your model while training, or to **keep track of your model's performance**.

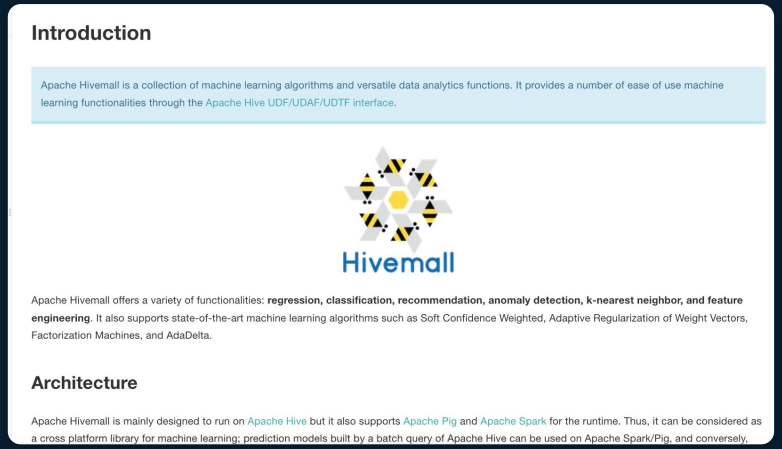
This includes making **predictions** on future data, as well as model **monitoring** to flag when there is data or model drift (which may require re-training of the model).

Types of ML Models Supported by TD

There are two main ways to run ML models in TD:

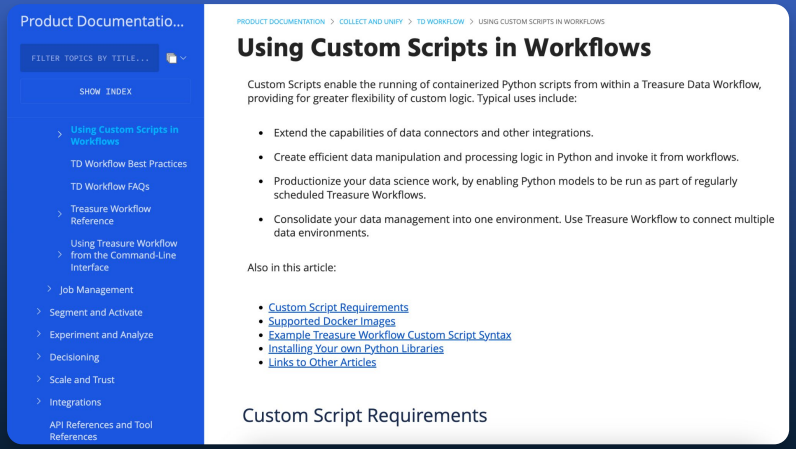
1

Through Hivemall, a library with SQL-like implementation of ML models. https://hivemall.github.io/.



2

Through Python, via our Python Custom Scripting tool. Barring any **memory limitations**, you can run any Python libraries' models and save outputs to Treasure Data. TD Documentation Link



Data Preparation (Presto processes)

Some of the most common data preparation steps we take before modeling include feature engineering and feature selection:

Feature Selection

You want to think about what data you want to feed into your model. This data is likely to come from multiple sources, different tables or different databases. For most models, you want to combine the features you want to include into one cohesive table.

Data grouping and aggregation

If we have a raw orders table, we might want to perform a GROUP BY and get the number of orders and average order value per customer.

Time filters

If you are building a recommendation system, you might not need to train on purchases over 3-5 years old

After creating features, you might want to do additional data cleaning, including:

Dealing with null values

You might want to use COALESCE(feature, 0) to fill numerical columns with zeros, or impute your categorical data with the most common value. You might also consider dropping the rows that have significant proportions of null values.

Dealing with outliers

A common formula to detect outliers is by flagging what values fall outside (1.5*IQR) where IQR (interquartile range) is difference between the 75th and 25th percentile value. You might remove, impute with custom min/max values, or just be aware of outliers in your data.

Scaling data

Following Data Science standards, some models (K-nearest neighbors, K-means clustering) might perform better with feature scaling.

Exploratory Data Analysis

	1	2	3	4	5
	Comparing the CV (Coefficient of Variation) ratio of your training data and your testing data.	**For regression models (which predict numerical values), calculate correlation coefficients.**	**Analyze the distribution of your target variable.**	**For clustering/segmentation models, examine the distribution of predicted clusters.**	**For recommendation engines, compare top ordered products with top recommended products.**
Approach	The CV ratio measures the variability of data relative to its mean.	Correlation coefficients measure how strongly variables are related, ranging from -1 to 1.	The distribution shows how values are spread out in your data.	This shows how many items are assigned to each group by your model.	This compares what's actually popular with what your model thinks should be popular.
How to Compute	For each feature, calculate the CV by dividing the standard deviation by the mean, then multiply by 100. Do this for both training and testing data. The difference between these ratios should be less than 3.	Use Treasure Data's SQL functions or statistical libraries to calculate Pearson's correlation between each input variable and your target variable.	Create a histogram using SQL queries in Treasure Data. Group your data into ranges and count how many fall into each range.	After running your clustering model, count how many items are in each cluster using a simple SQL query.	Use SQL queries to find the most frequently ordered products, then compare this list with the products most frequently suggested by your recommendation model.
Why it's useful	This ensures your data split is fair and representative.	It helps identify which variables are most strongly related to what you're trying to predict.	It helps you understand the nature of what you're predicting and how well your model is performing.	It helps you understand if your groupings are balanced and meaningful.	It helps validate if your recommendations align with actual customer behavior.

Modeling & Recommended Data Sizes

- **Hivemall models** run much faster and are much more memory efficient, but you are limited to the models that exist in the library: https://hivemall.github.io/
- **Python models** are limited to the amount of Custom Scripting your account has, but you are able to use any Python library you want!

ML Language	Custom Scripting Tier	Table Size for Model Training	Prediction Method + Limitations	Algorithms Available
Hivemall	N/A	100M +	Score via Hivemall Code. 100M+ users	Anything except Ensemble Models, Clustering or Deep Learning / LLMs
Python	**Tier1:** Max CPU: 1 vCPU, Max Memory: 10 GB, Max Disk: 10 GB	500K-1M rows (depends on algorithm and has to be tested on your data)	Pickle model and store in S3 Bucket. Load model in Docker and score new users in batches 500K-1M users	Any Python Library
Python	**Tier2:** Max CPU: 4 vCPU, Max Memory: 30 GB, Max Disk: 10 GB	1M-5M rows (depends on algorithm and has to be tested on your data)	Pickle model and store in S3 Bucket. Load model in Docker and score new users in batches 1M-5M users.	Any Python Library

Predictions

Hivemall models have built-in prediction code.

Python models are able to be pickled and saved to an external source (S3 bucket) to be reused later to score new users (lookup boto3 library). Predictions on the train/test data can be done in the same process as model training from your Custom Scripts.

Model Validation

To keep track of your model's performance, there should be several model validation processes in place. This includes:

Model metrics

These can be calculated in either Python (with the modeling script) or in Presto if the model's outputs are output in a TD database. For example, R-squared and RMSE (root-mean-squared error) for regression models, accuracy/recall/F1 scores for classification models. A good practice is to insert these metrics with the session ID of the WF run to keep track of performance across iterations.

Analysis of predictions

This is used to compare model fit (over or underfitting) and also to make sure that predictions make sense in the context of the use-case. You could get the distribution of predictions (mean/median/mode or distribution of categorical outputs) and output that to a TD database. Again, this can be inserted with session ID to a table for tracking.

Model monitoring and data drift

Hivemall has a function for calculating the drift of any model's averages and standard deviations over multiple runs. This outputs the KL-divergence coefficient, that you want to make sure does not spike. A spike indicates possible data error, or that the model needs to be retrained.

How to Use ML Model Outputs for CDP Use Cases

There are typically two main uses of the outputs of an ML model built in TD:

Adding model outputs
to Parent Segment & Audience Studio as customer attributes to be used for segmentation

Writing model outputs
to external systems (ADL, SFMC, S3, BI Tools etc.) to be used for campaign setup or building dashboards with useful business insights

This diagram summarizes the processes that our CLTV workflow uses.