RFC - Upgrading Transaction Enrichment

Author: Alex Spanos
Date: 04/12/24

Executive summary

This RFC proposes a basic hybrid rules-based and Machine Learning enabled system categorisation service to improve the value it provides to Monarch's users. The scope is constrained to improving system categorisation capabilities. Rather than a comprehensive solution, the RFC avoids premature optimisations and spending too many innovation tokens .

A link to the accompanying exploratory data analysis can be seen here.
The relevant Github repo can be found here.

Problem statement

The quality of Monarch's system categorisation currently depends entirely on upstream aggregator categorisation quality, which has been shown to be unsatisfactory and prone to fluctuations over time (non-robust). To provide Monarch's users with a better experience, and for other reasons (unit economics, extensibility, even company valuation) it is crucial for Monarch to develop an in-house categorisation process that leverages its potentially powerful transaction data asset, and which does not depend singularly on third party categorisation.

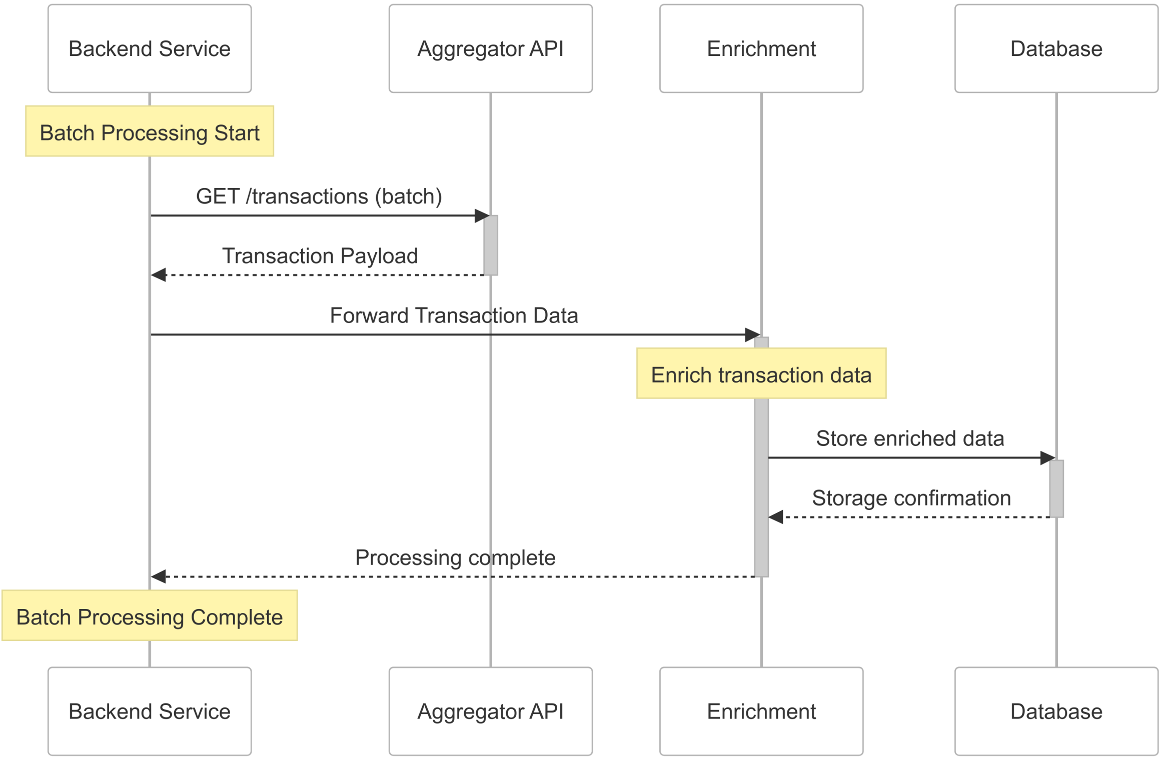
In summary, the problem statement reduces to the following: improve system categorisation accuracy within the latency and batch-size constraints of the current transaction ingestion process.

Existing categorisation process

In summary:

- Takes place during household transaction batch ingestion process (small batches of <=20 transactions)
- Operates on a single-transaction basis (non-contextual)
- Calculated via mapping aggregator category against Monarch taxonomy
- Quality measurement is not implemented

Below is an oversimplified diagram of the ingestion & enrichment process



Open questions & assumptions

- Considering the task focus on categorisation, the scope of the RFC is limited to improving the categorisation process *only*, at a first stage.
- There are no current quality metrics benchmarks stated, or ground truth datasets provided, or specific categories that are higher priority to categorise correctly. Hence the RFC focuses also on this crucial aspect of data-driven product development.

- There are no explicit latency targets defined. Considering the requirement to not affect the ingestion process overall latency, and the "zero-to-one" nature of the problem, a low-complexity solution will be proposed.
- The size and quality of the underlying dataset is not specified, hence the assumption is that it is presently not appropriate for some types of data-intensive algorithms.
- Data exploration does not consider different account types (credit card, bank account, etc) and treats negative amounts as debits and positive as credits (which may not hold in reality).

Quantifying current performance

For the purposes of this RFC, it will be useful to have a gauge of the performance of the current categorisation process, using the provided datasets.

A basic data analysis was performed, and can be viewed in a dedicated Streamlit app.

After some basic preprocessing, the dataset is seen to contain ~7k account transactions for 10 households. The transaction dataset contains duplicates, in cases where users have recategorised or added their own categories.

In the absence of ground truth (although some cases of user re-categorisation could be ground truth candidates), we will use *coverage*, hereby defined as the percentage of transactions that receive *any* system category.

Overall coverage was measured at **~42%**, a particularly low figure. Coverage seems to vary significantly across households.

Investigation of uncategorised transactions thankfully reveals that there exist a number of high-cardinality groups that can be easily categorised, and which can drive the overall coverage metric upwards.

Meanwhile, a qualitative review of system-assigned categories reveals mixed results, implying that the overall accuracy is even lower. (Note: accuracy may not be the best metric to adopt for reporting purposes)

Proposal

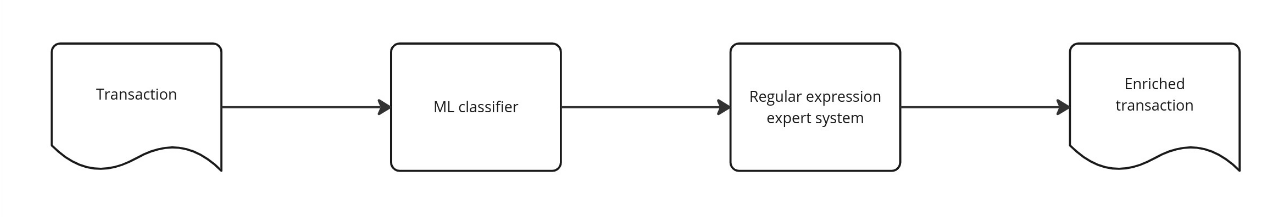
This proposal takes into account the following requirements:

- Improved accuracy
- "Low" latency contribution to ingestion
- Extensibility
- Explainability
- Low complexity

Furthermore, it takes into account:

- The lack of large volume annotated ground truth datasets, that prove particularly useful in developing more advanced methodologies.
- The possible lack of experience within the organisation of deploying ML models

Basic ML+rules-based categorisation system proposal



Regex component

A regular expression table can be used to hardcode specific categories. This should be used sparingly, and potentially as an override.

Regex	Category	Example
^ATM.*	Financial, Cash & ATM	ATM WITHDR 7832

ML pipeline component

A simple feature-based ML pipeline can be leveraged as a first step.
The following features can be used:

- Aggregator category
- User feedback
- Amount
- Description (using basic statistical NLP feature engineering such as tfidf)
  The choice of classifier is secondary, and should rely on experimentation results; prime candidates would include SVM, or tree-based methods (although not the best direct choice for the high dimensional feature space that tfidf produces).

Training details

The above system can be calibrated and trained by leveraging relevant ground truth datasets (see below).
For the ML model training a training CLI would be developed, alongside with a metrics CLI that can be used to evaluate the model on arbitrary hold-out datasets.

The computational requirements for training a traditional feature-based ML pipeline will not be significant and will not require GPUs.

The dataset size, ensuring adequate representation per category, is not known a priori. But a "finger-in-the-air" estimate of 500 per category for ~50 categories gives a rather manageable 25k dataset size. Emphasis on the quality rather than the quantity of the training data will be the biggest factor for ensuring accuracy. Costs are therefore expected to be low, regardless of retraining frequency.

Deployment and maintenance

The adoption of an ML pipeline in production necessitates introducing basic MLOps practices:

- Safe model updates (invest in robust CI/CD processes that compare performance of candidate models with incumbents, and appropriate deployment patterns such as canary deployment)
- Observability (aside from usual functional metrics, should include predicted category distributions and feature statistics)

Prerequisite: ground truth (gold) datasets

To train the ML model, and even create the regular expression table, ground truth datasets are required.
Building these require investment in dataset creation/curation processes, and may require dedicated personnel.
In the absence of this, preliminary ground truth datasets can be constructed through:

- User feedback consensus mechanisms (this can also inform taxonomy evolution for Product)
- Programmatic annotation
  - Rules
  - Locally hosted LLMs

Limitations & constraints

- Performing enrichment during a small-batch ingestion process prevents the ability to use the household (account) context during categorisation. For example, using recurrence of similar transactions can be highly valuable for identifying salary/rent/wages, and other important recurring categories. Therefore, in this architecture a non-contextual categorisation system that operates on a single transaction basis is more appropriate.

- As a consequence, performance on non-merchant transaction types, or where the description itself contains low information, will be suboptimal.
- Monarch's taxonomy contains both merchant and non-merchant categories (such as Transfers, Cash & ATM, Taxes, and others). The transaction descriptions, amounts and dates are not necessarily sufficient for predicting the above categories reliably. It will be important to leverage a transaction type field from the aggregator payload to improve overall categorisation (see Plaid for example)
- Additional data points that could be useful are MCC , aggregator ID and bank ID.
- The accuracy of this approach will likely plateau at some point, at which, more sophisticated models should be developed, using more training data.
- A hierarchical classification approach, predicting first category group and then category may provide superior user experience.

Alternatives considered

1. Transaction enrichment at Monarch currently consists of adding both merchant and category fields. For merchant-related transactions, these 2 fields are clearly interrelated; knowing that a merchant is "Pizza Hut" enables a deterministic mapping to the category of `[Food & Dining, Restaurants & Bars]`.
   Therefore developing a proprietary merchant database and using it as the predominant merchant enrichment process can provide more control and tangible company IP. Entities detected in transaction descriptions can be used for performing stochastic lookups on the merchant tables.

2. Context-aware categorisation will be superior for categorising non-merchant transactions and can be adopted as a general solution - especially when considering Recurrent Neural Network algorithms for this purpose. Such approaches are data-intensive and require more context than what is currently provided in the 20-transaction batch.