Technical Assessment Report

Part II: Action Plan

Prepared by: Li Lok Hang, Alan

Date: 27/05/2021

Methodology Outline

In part (I), I have created a model to characterize thousands of user features into clusters. Now, the clustering algorithm of historical data forms a *supervised* classification problem for new data points. In other words, we can treat the clustering results as the ground truth label. Given a new user, with unseen records of transactions, accounts and budgets, we can classify this user into clusters defined by past records of other users.

Common classification methods can be tested from the Python library sklearn, such as k nearest neighbors (kNN) and decision trees classification. We can then train our classification model using a train-test split of historical data, and verify the testing data from the ground truth. Then, the classification model score can be calculated by the method of contingency matrices. We can define the score as common metrics, such as hit rate (HR), false alarm rate (FAR) or F1 score.

After training the classification model, we can apply it to unseen records to obtain the cluster label of that record. Then, for each cluster, we can create a corresponding time series forecasting model. We can test different algorithms beginning from simple ones, such as multivariate linear regression models (MLR) to more complex ones, such as random forest regressors. Using ordinary least squares models as a basic example:

$$y_j = \sum_{i=0}^{n} a_{ij} x_i$$

where: y_j = response variable (net-worth) of cluster j

a_{ij} = fitted parameter for explanatory variable i of cluster j from ordinary least squares in the linear regression model

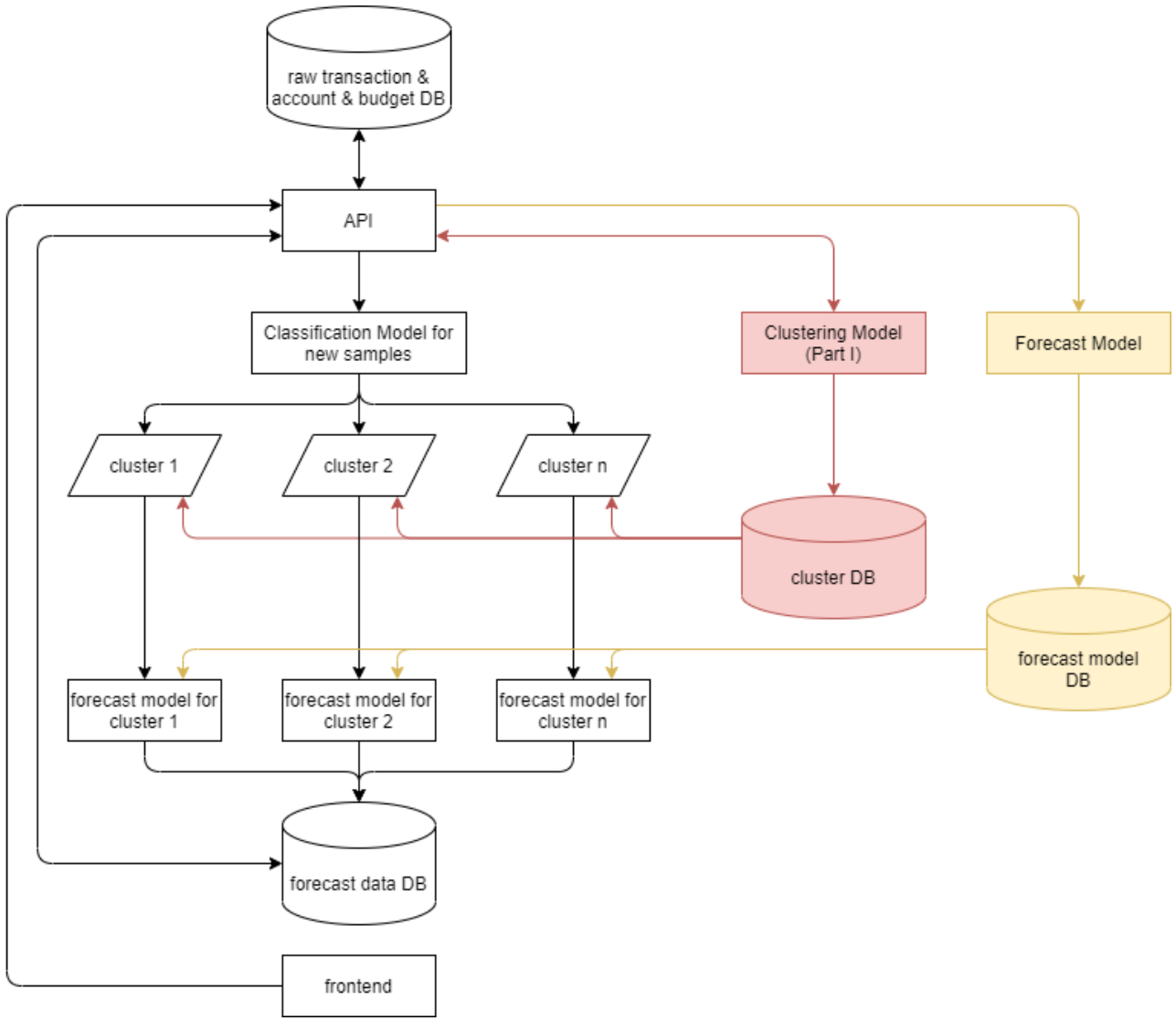
x_i = explanatory variable i

To get the unknown parameters a_{ij}, we need to get historical (x, y) pairs from the database and apply the principle of least squares. This is known as training our forecast model.

Since the dataset is variable in time, we need to redo clustering regularly to keep track of the effect of new data. In other words, we need to schedule the clustering script. Clusters are dynamically loaded from DB in the classification model to synchronize the scheduled change.

Simplified Flow Chart of the Algorithm

The following shows the simplified flow chart of the aforementioned algorithm.



Detailed Action Plan

Action		**Person responsible**	**Time required**	**Details**
Database related	MongoDB setup	Backend developer	1~2 wk	Create database and collections
	API design			Write API functions for CRUD operations to DB, including (etc): - transactions & budgets & accounts - cluster statistics
Forecast Development	Data query	Backend developer	2 wks	Get historical net-worth time series data through API
	Exploratory data analysis	Intern		Use data visualization tools and basic statistic tools, such as *statsmodels* to intuitively understand net-worth trends
	Dynamic clusters	Backend developer		Code to automatically updates cluster labels when new data is added to the database
	Classification model	Intern and I		Use clustered historical data as training data set labels Trial and error to test and evaluate performance of different classification algorithms
	Time series forecast model	Intern and I	2 wks	Trial and error to test and evaluate performance of different forecast algorithms

Post Forecast	API design	Backend	1~2 wks	Write API functions to pass forecast results to frontend
	Cybersecurity checks			Check whether there are possible vulnerabilities, such as injection attacks or data breach vulnerabilities on the frontend. This is of paramount importance to safeguard our user's data privacy.
QA	Unit test	Intern and I	continuous	Using the *pytests* framework to write small tests for code quality assurance. Examples of tests include (etc): - API functions tests - whether a point from a cluster is successfully classified - whether an outlier is successfully removed and not classified to any cluster