Global Asset Management
Data Science Test Report

Philip Ndikum, HSBC Data Scientist

Liquidity Risk

- This tasks centers around predicting the cost of selling an asset
- Goal: Construct a liquidity model for bids based on time and volume
- Interpret this model using a variety of statistical techniques

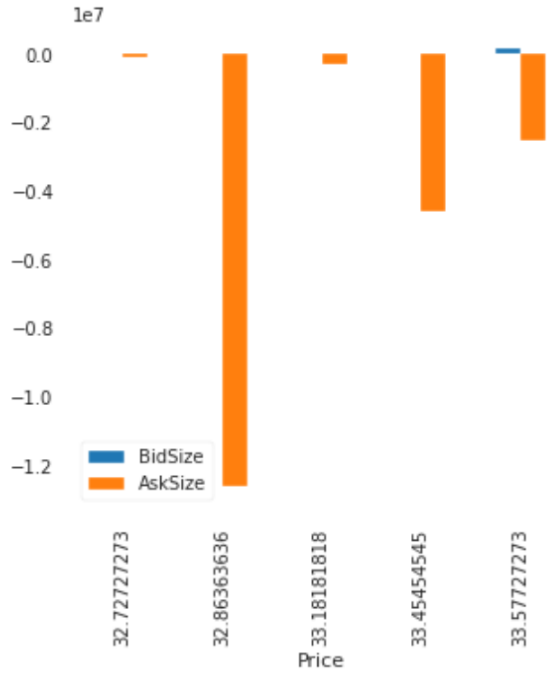
Remark: Whilst we were focusing on bid transactions the software contains models and Visualizations for both bid and ask transactions from the LOB. In this presentation we Only focus on bid transactions.

Exploratory Data Analysis

See Jupyter Notebook with Pandas Profiling for detailed EDA

Key takeaways

- No Null/NaN values – dataset was likely cleaned prior
- All price metrics were correlated as expected (high, low etc)
- Average volume for each bid price seems to be much higher on the ask side vs bid size suggesting the supply greatly outweighs demand



Computing Liquidity Cost from Level 2 Data

We are not given the liquidity costs so we must first try to predict this. In a quick research phase a variety of models were found:

Reference	Features	Implementation notes
Macquarie, Stewart (2014)	Bid-offer spread, Duration, Turnover	**Model could not pick up any signals**
Stefano Pasquali MD Blackrock (2019)	Implementation shortfall	Not implemented
Knight and Satchell (2007)	Slippage, Spot Price, Weighted Spot Price, Quote Type	Not implemented
Amaya, Rochen et al (2015)	Empirical model based on bid price and volume from Limit Order Book (LOB)	**This was implemented**

Computing Liquidity Cost from Level 2 Data

Amaya, Rochen et al (2015) paper *"Distilling liquidity costs from limit order books"* presented a model which could be implemented with the data provided.

In a short the paper gives the following:

1. First, we must compute the average price per share in each interval using a volume weighted average of the (volume x price) of the asset on each level

$$X_i = \sum_{j=1}^{i} x_j,$$

$$\hat{S}_t(X_i) = \frac{\sum_{j=1}^{i} p_j x_j}{X_i}.$$

2. Finally it is argued that in order to compute the marginal liquidity cost one must undertake a linear regression on the average share price on the volume at each interval

$$\hat{S}_t(X_n) = \hat{\alpha}_t + \hat{\lambda}_t X_n + \epsilon_t, \quad n = -N, \ldots - 1, 1, \ldots, N. \tag{7}$$

The coefficient $\hat{\lambda}_t$ represents the estimate of the marginal liquidity cost per share from an order limit book.

Remark: This can be seen analogous to calculating the Beta in CAPM equations with rolling linear regressions.

Computing Liquidity Cost from Level 2 Data

Amaya, Rochen et al (2015) paper *"Distilling liquidity costs from limit order books"* presented a model which could be implemented with the data provided.

Important remark: The author's note that the model presented in the previous page works well for short term time intervals (intra-day LOB transactions) but may not work for long term intervals and proposes other methods may be required. This was reflected in the model analysis and so two data-sets were built for a frequency of f=30 min intervals and f=daily intervals.

$$X_i = \sum_{j=1}^{i} x_j,$$

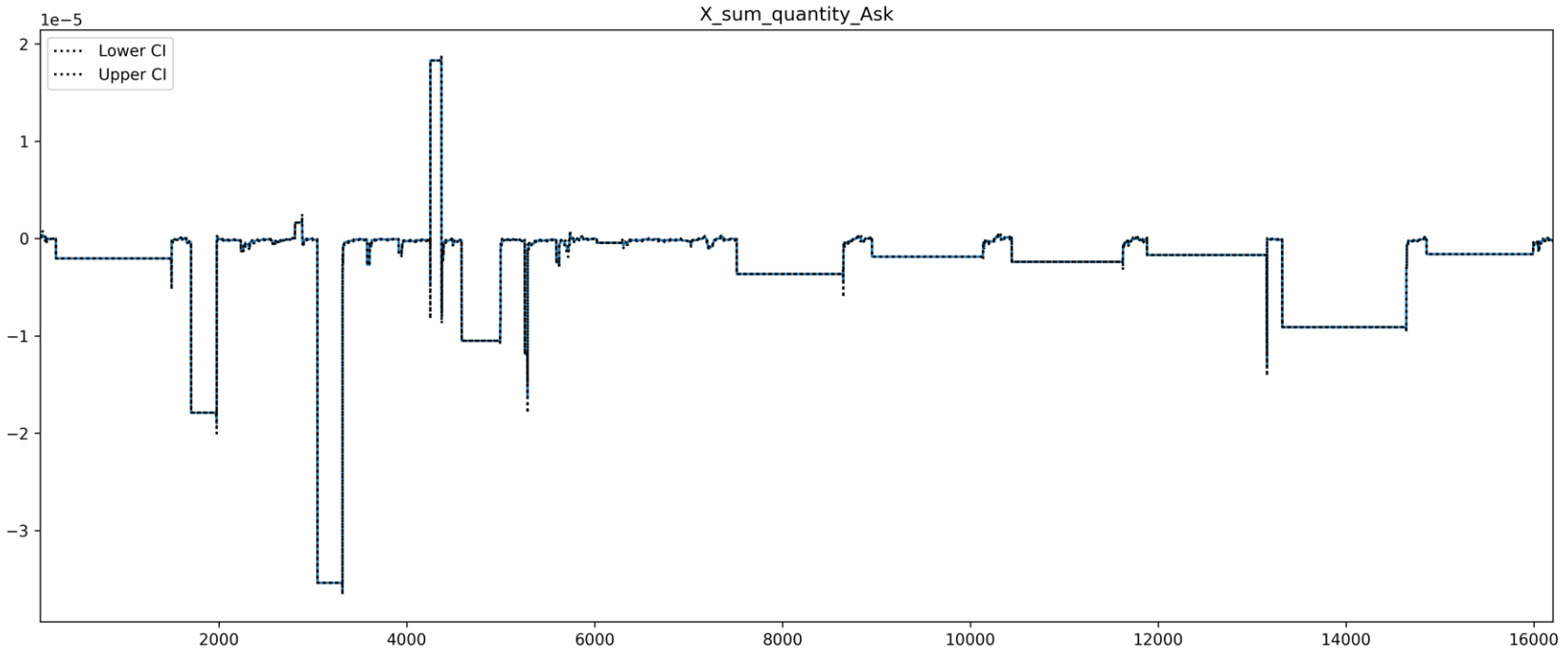
$$\hat{S}_t(X_i) = \frac{\sum_{j=1}^{i} p_j x_j}{X_i}.$$

$$\hat{S}_t(X_n) = \hat{\alpha}_t + \hat{\lambda}_t X_n + \epsilon_t, \quad n = -N, ... - 1, 1, ..., N. \tag{7}$$

The coefficient $\hat{\lambda}_t$ represents the estimate of the marginal liquidity cost per share from an order limit book.

Computing Liquidity Cost from Level 2 Data

Example plot of rolling liquidity cost for bids after regression analysis using StatsModels



Remark: A bayesian rolling regression analysis was tried for a more accurate computation of liquidity costs analysis. This did not work but could be debugged with more time.

HSBC

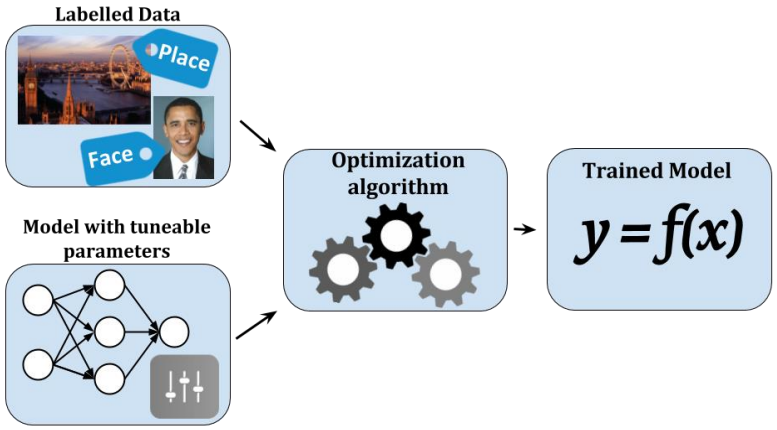
Philip Ndikum, HSBC Data Scientist

Model Training – Time Series Cross Validation

After the liquidity cost was computed this had to be joined onto the level 1 market data on the time columns. The final *merged* data-sets can be described as follows:

Short time Data (30 min intervals)	Long time Data (Daily intervals)
• Frequency: 30 min intervals • Rows: 1014	• Frequency: Daily (24 hour) time intervals • Rows: 21 • Remarks:There is a risk of massive overfitting here due to the small data-size. With more time one could explore if the merging technique could be improved to extract more rows.

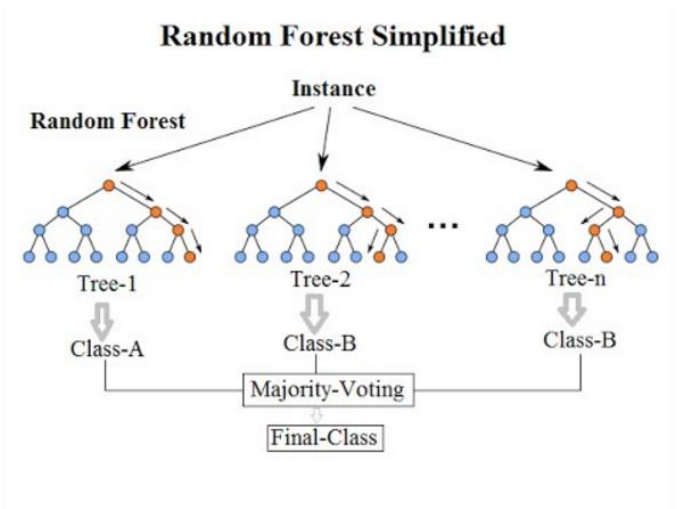
With labels we can now do predictive modelling. Our label is the liquidity cost and the input data is the level 1 market data.



- **Remark:** For interpretability of the liquidity cost variable y was scaled such that y in [-100, 100]

Model Training – Time Series Cross Validation

- Feature Leakage: To prevent feature leakage – time series cross validation was used.
- Over 300 models were trained and assessed
- The results were saved to the "cross-validation" data folder



- Feature Leakage: To prevent feature leakage – time series cross validation (TSCV) was used.
- There are a variety of TSCV schemes that can be used prevent overfitting on time series data

HSBC

Philip Ndikum, HSBC Data Scientist

Model Evaluation

See Cross Validation Folder for full

Model Results

Model Evaluation

In order to build a model from our cross validation we did the following

1. Extracted the model with the best score and good variance score relative to the other models (to ensure the model is not overfitting).
2. Train a model on a 77:33 train-test split (because the time is in ascending order this will not cause any feature leakage)
3. Use SHAP and sci-kit learn partial dependence plots to build a liquidity surface plot

Model Evaluation – 30 min interval model (MSE)

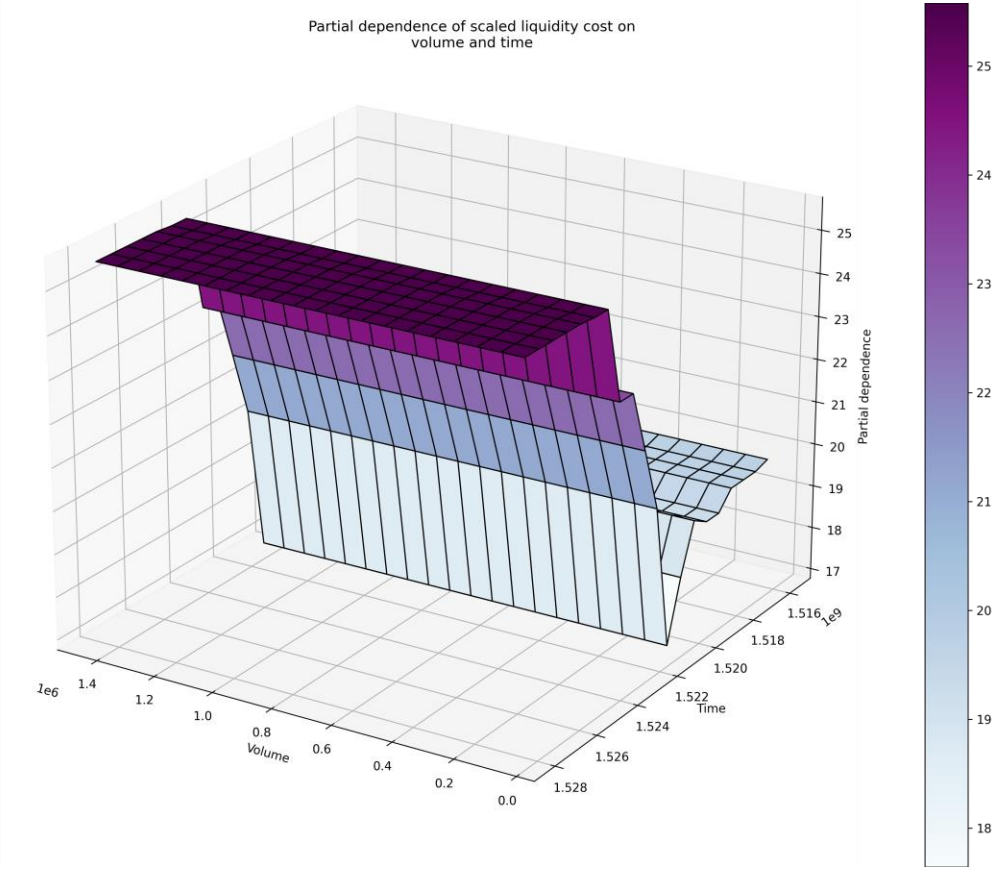
Top 5 Models (Bids)

estimator	min_score	mean_score	max_score	std_score	n_estimators	max_depth
NGBRegressor	2261.41	671.89	21.142	-843.32	16	N/A
ExtraTreesRegressor	2601.71	1169.22	23.43	-1170.14	16	3
ExtraTreesRegressor	2808.37	1187.551	24.09	-1229.93	32	3
ExtraTreesRegressor	3526.15	1499.11	24.49	-1616.80	16	4
ExtraTreesRegressor	2634.94	1136.29	24.96	-1184.39	150	3

Model Evaluation – 30 min interval model

Partial Dependent Plot analysis:

1. As suggested by research a short-time horizon may not enable you to capture complex volume dynamics.
2. Our PDP seems to show the longer the greater the time the higher the liquidity

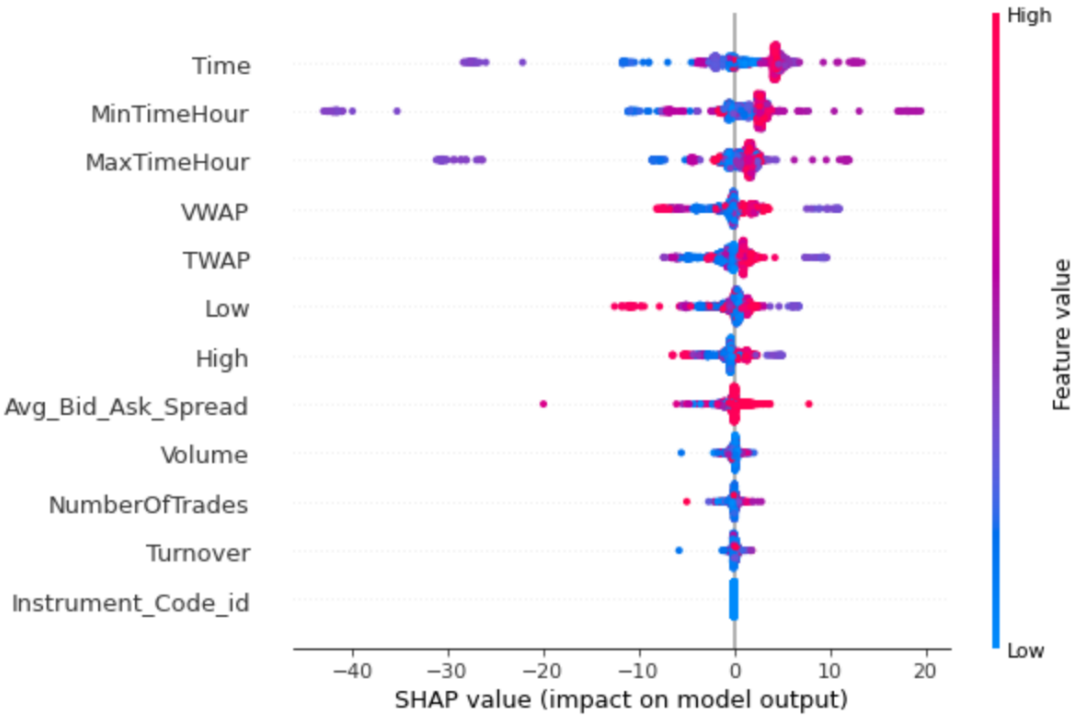


Philip Ndikum, HSBC Data Scientist

Model Evaluation – 30 min interval model

SHAP feature importance plot insights:

1. Time seems to have the greatest impact on liquidity cost.
2. Asset price also seems to be linearly proportional to liquidity cost



Philip Ndikum, HSBC Data Scientist

Model Evaluation – 30 min interval model (MSE)

Top 5 Models (Bids)

estimator	min_score	mean_score	max_score	std_score	n_estimators
NGBRegressor	1687.47	527.73	7.50	-671.55	16
NGBRegressor	1696.46	530.86	7.88	-675.09	32
AdaBoostRegressor	1579.48	604.16	9.53	-580.36	100
BaggingRegressor	1652.37	559.97	11.04	-586.23	32
BaggingRegressor	1715.85	581.88	14.62	-614.68	100

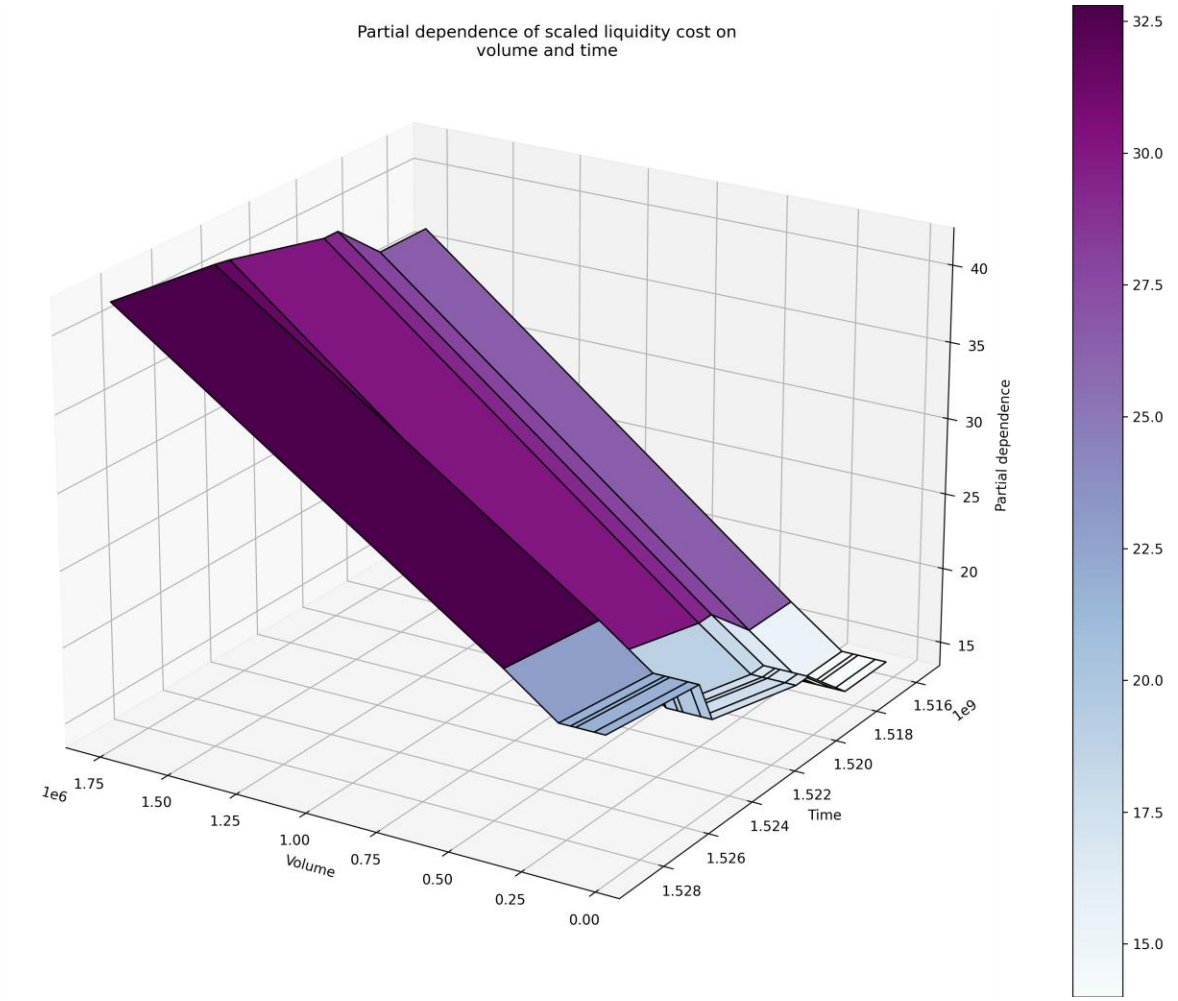
Remark: Some models built using the automated feature extraction package TSFEL (2020) provided more accurate results the trade-off being interpretability. For interpretable results we have provided the results of the best explainable models above. Other metrics such as Mean Absolute Percentage Error (MAPE) could be used for better interpretability.

HSBC

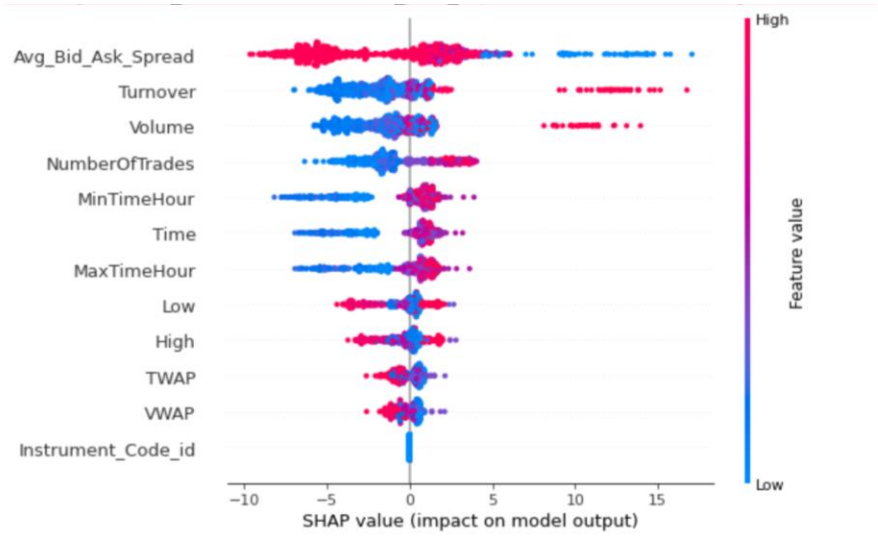
Model Evaluation – Daily Model

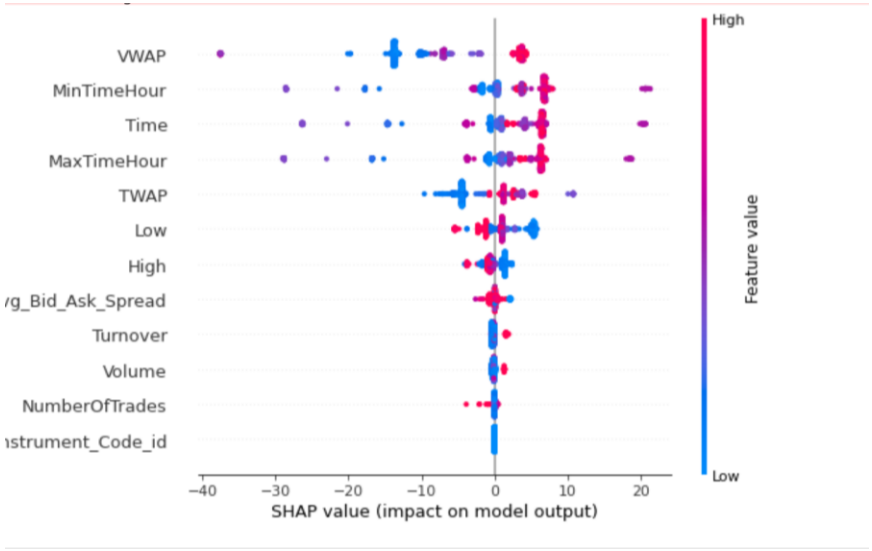
Partial Dependent Plot analysis:

1. Whilst there is little data this plot would suggest that the greater the volume the greater the liquidity costs



Model Evaluation – Daily Model





SHAP feature importance plot insights for CatBoost

1. Time seems to have the greatest impact on liquidity cost.
2. Asset price also seems to be linearly proportional to liquidity cost

SHAP feature importance plot insights for NGBoost

1. The VWAP seems to have the highest impact on the liquidity cost
2. Time also has an impact on the liquidity cost
3. Volume doesn't seem to make a big impact for the NGBoost model

HSBC

Philip Ndikum, HSBC Data Scientist

Q&A

Questions – Due to time constraints of challenge short answers are provided:

1. Assumed liquidity model was correct – from research papers the result from the paper seemed mathematically sound. The results also reflected the results I saw in the literature review that volume and time were proportional to liquidity costs.
2. Generalized liquidity model: Depending on the asset I would segment down the universe of the asset by it's underlying characteristics. For example create a group of large cap U.S equities and cluster those into a group and build a model. This approach could be applied to other asset classes.
3. Present through presentation and code. With more time I would heavily refactored the code. Due to time contraints I focused on solving the problem and omitted unit tests. In a production-grade package I would have heavily reduced the lines of code (LOC) and eliminated any wasteful or ad-hoc method. I would also improve the general code structue for re-usability.
4. AWS: S3 bucket or SQL database to store level 1 and level 2 market data. Sagemaker to train and productionize models into REST API architecture. GCP: Cloud SQL to store level 1 and level 2 market data. ML Engine to train and productionize models. If regulation requires auditing models of models I would also explore a tool like DVC to version control models and write methods to automatically store feature importance of models into a database.
5. Larger data-set – use DASK to parallelize feature engineering over CPU's in local machine. Use TSFresh instead of the new package used to parrellize feature engineering over CPU's in local machine. For larger computation use spark or hadoop for feature engineering – for cross validation use GCP/AWS tools to train over multiple cloud servers. For neural networks use cloud or on-premise GPU's for model training.
6. Training on production environment: Dockerize the code to enable it to be run on servers using Docker Swarm or Kubernetes. Use cloud API architecture for model serving or for on-premise severs use FastAPI/Django/Flask. See Question (3) for code improvements.

Appendix

Notes:

1. EDA:
   - If time permitted, I would have done data analysis in greater detail
2. Model choices:
   - Neural networks: These would be hard to train on local machine and would require more boiler point code for hyperparameter optimization and model training (normalization and standardization of features) so this was avoided due to time constraints.
   - Support Vector Machines were also slow on local machines. DPSVM provides distributed SVM models on larger machines.
3. Package installation and instructions are provided in the README.md file.
4. References:
   - All code used and papers are cited throughout the code