भारतीय प्रौद्योगिकी संस्थान जोधपुर
Indian Institute of Technology Jodhpur
॥ त्वं ज्ञानमयो विज्ञानमयोऽसि ॥

Trading with Machine Learning and Big Data

Leveraging AI to Optimize Trade Execution

AI for Fintech (MSL 71560)

PRESENTED BY

Navin Kumar

TABLE OF CONTENT

Machine Learning in Trade Execution Support

1 — 2 — 3

Context & Motivation	**Benefits of Machine Learning**	**Challenges to Implementation**
❑ Advances in Technology ❑ Industry Demand	❑ Competitive Edge ❑ Enhanced Decision-Making	❑ Data Quality ❑ Skill Requirements ❑ Liquidity Constraints ❑ Model Transparency

Real-Life Applications of Machine Learning in Trade Execution

- ❏ Use Case 1: Feature Importance
- ❏ Use Cases 2 and 3: Transaction Cost Analysis (TCA).
- ❏ Use Case 4: Normalized and Unbiased Algo Trading.
- ❏ Use Case 5: Trading Strategy Recommendation Model.

Use Case 1 - Feature Importance in TCA for Auto-Order Routing

Objective:
Utilize Feature Importance in ML to identify key factors affecting trading costs, helping traders make data-driven decisions and automate less critical trades.

Why Use Feature Importance?

- ❏ Reduces complexity for traders.
- ❏ Highlights the most impactful factors in trading costs (e.g., order size, time).
- ❏ Supports automation by identifying routine trades to autoroute.

Steps Involved

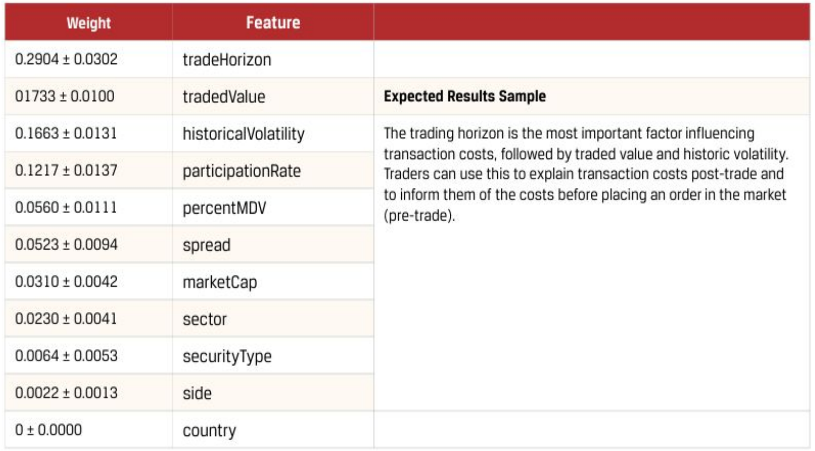
1. Data Cleaning & Normalization
2. Chunking Data
3. Data Labeling & Splitting
4. Selecting Relevant Features
5. Model Selection & Training
6. Feature Importance Methods

Use Case 1: Feature Importance in TCA for Auto-Order Routing

Methods for Calculating Feature Importance

- ➢ Default scikit-learn
  - ○ Quick, but can be biased.
- ➢ Permutation Importance
  - ○ More accurate, but computationally expensive.
- ➢ Drop Column Importance
  - ○ Intuitive but requires model retraining.
- ➢ SHAP Values (Shapley Additive Explanation)
  - ○ Provides detailed insights with decision plots,
  - ○ But computationally intensive.

Exhibit 1. Selecting Feature Importance



Weight	Feature	
0.2904 ± 0.0302	tradeHorizon	
01733 ± 0.0100	tradedValue	**Expected Results Sample**
0.1663 ± 0.0131	historicalVolatility	The trading horizon is the most important factor influencing transaction costs, followed by traded value and historic volatility. Traders can use this to explain transaction costs post-trade and to inform them of the costs before placing an order in the market (pre-trade).
0.1217 ± 0.0137	participationRate	
0.0560 ± 0.0111	percentMDV	
0.0523 ± 0.0094	spread	
0.0310 ± 0.0042	marketCap	
0.0230 ± 0.0041	sector	
0.0064 ± 0.0053	securityType	
0.0022 ± 0.0013	side	
0 ± 0.0000	country	

Insights

The most critical factors in transaction costs include Trade Horizon and Traded Value. Traders can leverage this to anticipate costs pre-trade and better explain post-trade expenses.

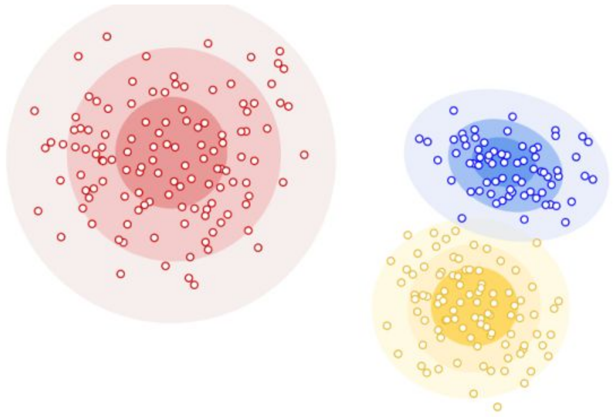
Use Case 2- Semi-Supervised Learning for Clustering Similar Orders

Introduction

In trading, semi-supervised learning addresses challenges where datasets lack clear labels.

Objective: Identify patterns within untagged or partially tagged data, such as commission types or order types.

Key Challenge: Large amounts of untagged or inconsistent data require advanced techniques for clustering and identifying similarities.



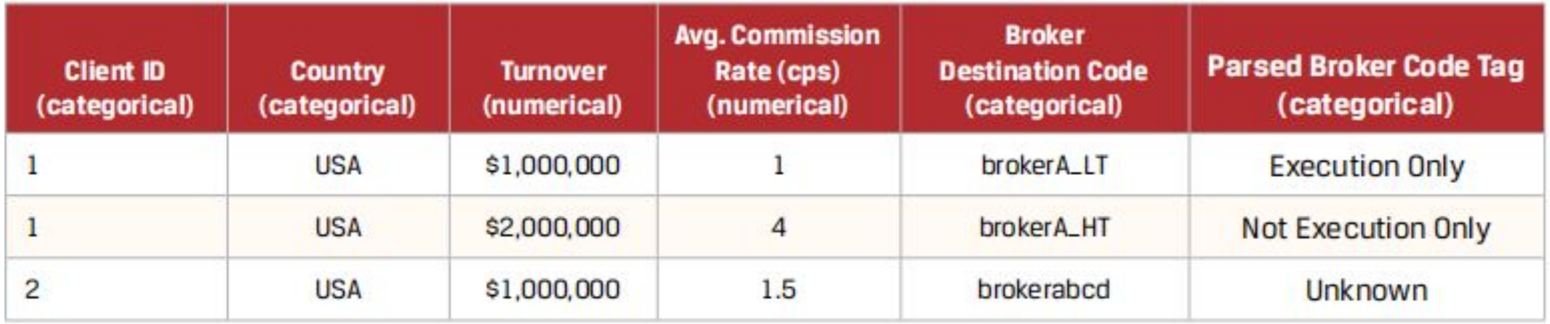
https://neptune.ai/blog/clustering-algorithms

Use Case 2- Semi-Supervised Learning for Clustering Similar Orders

Problem: Missing or inconsistent data tags in trading datasets, particularly for attributes like commission types

Solution: Use semi-supervised learning to leverage both labeled and unlabeled data, filling gaps where tags are missing.

Example: Predicting commission types for trades with incomplete tags by extrapolating from partially tagged data.



Client ID (categorical)	Country (categorical)	Turnover (numerical)	Avg. Commission Rate (cps) (numerical)	Broker Destination Code (categorical)	Parsed Broker Code Tag (categorical)
1	USA	$1,000,000	1	brokerA_LT	Execution Only
1	USA	$2,000,000	4	brokerA_HT	Not Execution Only
2	USA	$1,000,000	1.5	brokerabcd	Unknown

Use Case 2- Semi-Supervised Learning for Clustering Similar Orders

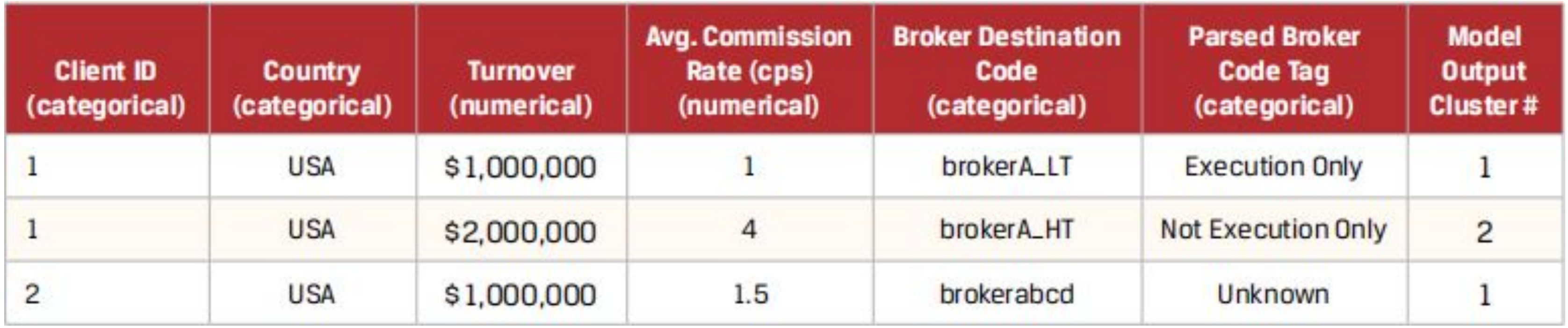
- **Data Preparation**: Organize and preprocess data, including:
  - Parsing broker codes.
  - Categorizing relevant numerical and categorical features.
- **Model Selection**: Choose an appropriate model:
  - K-Means: For numerical data clustering.
  - K-Modes: For categorical data clustering.
  - K-Prototype: Combines K-Means and K-Modes for mixed data types.
- **Training**: Use labeled data to train the model, then apply it to predict missing tags in unlabeled data.

Use Case 2- Semi-Supervised Learning for Clustering Similar Orders

Model Output: Clustering results that identify potential commission types or order types.

Example Output: Distinguish between "Execution Only" and "Not Execution Only" trades by clustering based on commission data.

Outcome: Traders gain insights into execution costs and commission structures, enabling improved decision-making.



Client ID (categorical)	Country (categorical)	Turnover (numerical)	Avg. Commission Rate (cps) (numerical)	Broker Destination Code (categorical)	Parsed Broker Code Tag (categorical)	Model Output Cluster #
1	USA	$1,000,000	1	brokerA_LT	Execution Only	1
1	USA	$2,000,000	4	brokerA_HT	Not Execution Only	2
2	USA	$1,000,000	1.5	brokerabcd	Unknown	1

Use Case 2- Semi-Supervised Learning for Clustering Similar Orders

Benefits and Challenges

Pros	Cons
❑ Automates the tagging of unstructured or incomplete data. ❑ Reduces manual labeling workload and enhances data quality.	❑ Requires significant preprocessing and data cleaning. ❑ Model accuracy depends on the quality and quantity of initial labeled data

Use Case 3 Introduction to NLP for Parsing Free-Form Text into Data Tags

- ➢ NLP automates parsing of unstructured text for data insights.
- ➢ Focused Use Case: Extracting actionable data tags from portfolio manager notes to support trading decisions.
- ➢ Example Texts:
  - *"Target this order for the close"* ⟶ Tag: **Close**
  - *"Part of a prog, trade cash neutral"* ⟶ Tag: **Program trade**
  - *"Avoid impact, trade in line with volume"* ⟶ Tag: **Go along**
- ➢ **Purpose**: Enable systematic understanding and efficiency in trade instruction processing.

NLP Model Choice - BERT and Domain-Specific Variants

- ➢ Why BERT? Bi-directional context evaluation enhances understanding of complex instructions.
- ➢ Domain-Specific Adaptations:
  - FinBERT for financial documents, LEGAL-BERT for legal contexts.
- ➢ Implementation Tools:
  - Virtu Analytics team's use of JupyterHub for development.
  - AWS SageMaker and Comprehend offer quick setup options.

Use Case 3 Introduction to NLP for Parsing Free-Form Text into Data Tags

- Advantages:
  - Open-source model pre-trained on massive datasets.
  - Fine-tuning on small labeled datasets for high accuracy.

Benefits and Applications in Trading Strategy Optimization

- Post-Trade and Execution Planning:
  - Tags enhance data-driven insights on trading costs.
  - Enables better strategy alignment based on historical trade outcomes.
- Performance Metrics:
  - Graph showing accuracy improvements over time with BERT fine-tuning.
  - Tagging consistency bar chart comparing manual vs. automated extraction.
- Future Potential:
  - Expansion to other free-form fields in trading and beyond.

Use Case 4 - Transaction Cost and Market Impact Prediction

Objective:

- ❖ To improve market impact prediction for transaction cost analysis (TCA) by leveraging machine learning (ML) models, specifically focusing on using an algo wheel for enhanced accuracy in broker assessment and performance prediction.

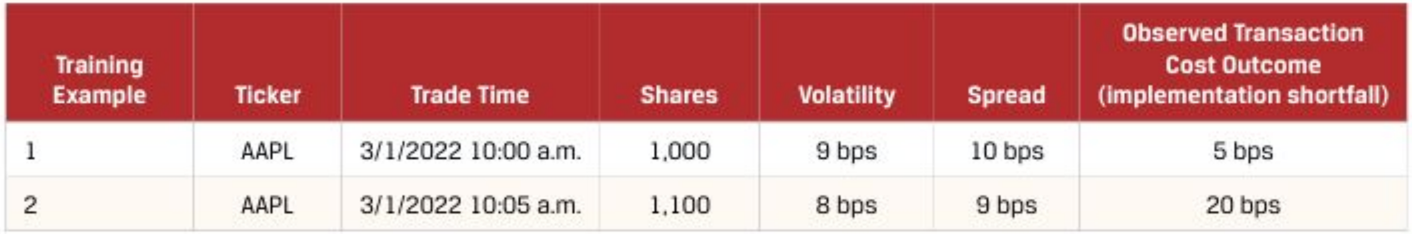
- ❖ Market impact models within TCA aim to evaluate the execution quality of trades by comparing the actual transaction cost to expected costs, which helps traders make more informed decisions and adjust future broker allocations.

Use Case 4 - Transaction Cost and Market Impact Prediction

Problem:

❖ **Data Noise and Inconsistencies**: The dataset used for TCA analysis often has "noise" or inconsistencies. For example, nearly identical trades (same ticker, time, shares, volatility, and spread) can result in different transaction costs. This inconsistency can confuse ML models and affect prediction accuracy.

❖ **Undetected Influencing Factors**: Not all factors that affect transaction costs are easily captured. For instance, market conditions or broker-specific nuances can impact trades in ways not covered by traditional TCA factors, making it challenging for the model to learn accurately.



Training Example	Ticker	Trade Time	Shares	Volatility	Spread	Observed Transaction Cost Outcome (implementation shortfall)
1	AAPL	3/1/2022 10:00 a.m.	1,000	9 bps	10 bps	5 bps
2	AAPL	3/1/2022 10:05 a.m.	1,100	8 bps	9 bps	20 bps

Use Case 4 - Transaction Cost and Market Impact Prediction

Solution:

- **Utilizing the Algo Wheel for Consistency**:
  An algo wheel is an automated system that randomly routes trades to brokers based on predefined allocations.
  Traders select their desired algo strategy and sending the order through the system. The algo wheel's engine then takes over, directing the order to a chosen set of normalized strategies that ensure consistency across trades.
  Over time, this process accumulates data from approximately 300 orders per broker, creating a robust dataset that allows for comprehensive performance assessment.

- **Insights from Market Impact Prediction**:
  The ML model analyzes cross-firm flows to compare broker performance and can be customized for firm-specific flows to adapt to individual trading strategies.
  By evaluating broker performance, the model reveals critical insights into transaction costs and execution quality, aiding in more informed trade routing decisions.

Use Case 4 - Transaction Cost and Market Impact Prediction

Solution:

- **Model Selection: Random Forest Approach**:
  The random forest model was selected for market impact prediction, outperforming other model types, including gradient-boosted trees, in this specific use case.
  Random forest models demonstrated high accuracy with the dataset, which includes both numeric and categorical data, while also being easy to explain and troubleshoot.

- **Third-Party Review**:
  Model builders engage non-model builders with expertise in the dataset, machine learning, and subject matter for a thorough review. This collaborative approach encourages critical assessment of the model’s data, features, and chosen techniques, ensuring robustness and reliability in the model development process.

Use Case 5- Trading Strategy Recommendation

Objective:

- ❖ Develop an ML-based trading strategy recommendation model to assist buy-side traders in selecting the optimal strategy that minimizes transaction costs and trade risks.

- ❖ Previous ML models provided insights but left the final strategy choice to traders, potentially increasing transaction costs if the strategy chosen was suboptimal.

Use Case 5- Trading Strategy Recommendation

PROBLEM

- ❖ **Challenge**: Choosing the best trading strategy can be complex and costly. Traders face high risks if they rely solely on manual decision-making or generic strategies.
- ❖ **Data Requirements:** Algorithmic trades need to be accurately tagged and standardized to ensure model predictions are reliable for firm-specific applications.

SOLUTION : ML Model for Strategy Recommendation:
Using data-driven insights, the model recommends the most cost-effective trading strategy while displaying cost and variability across available strategies.

- **Random Forest Model** can predict the likelihood of a trade being successful (or any other target outcome) based on the input features. For a given trade, it examines its `Ticker`, `Size`, `Market Cap`, `Stock Cluster`, `Volatility`, and `Spread` to predict a label (e.g., Buy, Sell, Hold) or probability of success.
- **Ensemble Decision**: Each tree within the forest makes a prediction, and the final prediction is made by majority voting (for classification) or averaging (for regression) across all trees

Use Case 5- Trading Strategy Recommendation

Data Preparation:

- ❖ **Source:** Virtu's Execution Management System data on all trades.
- ❖ **Features:** Order attributes (ticker, size, market cap), stock clustering (for filling data gaps), and real-time metrics (e.g., volatility, spread).

Modeling and Training:

- ❖ **Approach:** Train random forest models on historical data for various trading strategies (e.g., VWAP, implementation shortfall, liquidity seeking).
- ❖ **Goal:** Predict transaction costs for each strategy based on historical patterns and real-time data inputs.

Model Output:

- ❖ **Recommendation:** Displays the lowest-cost strategy with cost variability, helping traders assess potential outcomes.
- ❖ **Strategy Eligibility:** Flags high-risk trades as not eligible for low-touch strategies.

Use Case 5- Trading Strategy Recommendation

Benefits and Challenges

Pros	Cons
❑ Cost Efficiency: Reduces transaction costs with optimized strategy recommendations. ❑ Data-Driven: Provides traders with reliable, data-driven insights for informed decisions. ❑ Real-Time Adaptability: Incorporates real-time metrics, enhancing adaptability to market conditions.	❑ Bias Risks: Model accuracy can be skewed by data bias, e.g., if VWAP data primarily represents large-cap stocks. ❑ Complexity in Implementation: Requires comprehensive data tagging, cleaning, and robust modeling to be effective in diverse trading environments.

CONCLUSION

Unified ML Approach for Enhanced Trading: Across all five use cases, machine learning provides critical tools to optimize trade execution, reduce costs, and make informed, data-driven decisions.

Key Achievements:

- **Cost Efficiency:** Through predictive analysis, ML models guide traders to lower-cost and lower-risk strategies.
- **Improved Trade Precision:** Feature importance and transaction cost analysis enhance routing and strategy selection.
- **Data Structuring and Clustering:** NLP and clustering techniques improve data quality and provide meaningful order groupings for tailored strategies.

Impact: Together, these ML applications offer a cohesive framework that empowers traders with actionable insights, minimizes manual errors, and adapts to market complexities. This innovative approach makes trading smarter, faster, and more profitable in today's competitive financial environment.

THANK
YOU