<>

Machine Learning Proposal for Pairs Trading

Introduction/Background

Pairs trading is a market-neutral trading strategy that involves matching a long position with a short position in two highly correlated stocks or securities [1]. The idea is to profit from the relative price movements of two stocks that typically move together called spread. The reason why market-neutral strategy are important is the underlying principal behind them. These investment strategies are designed to minimize market risk by offsetting both long and short positions. When markets are bearish, market neutral strategies are not affected by the downturn in price movement [8]. If the price relationship between the two stocks diverges, the trader would short the over-performing stock and go long on the under-performing stock, expecting the prices to eventually converge again. Pairs trading works on the assumption that the prices of the stocks will revert to their historical correlation [1].

Literature Review

Pairs trading is a market-neutral trading strategy that involves matching a long position with a short position in two highly correlated stocks or securities. The idea is to profit from the relative price movements of two stocks that typically move together. If the price relationship between the two stocks diverges, the trader would short the over-performing stock and go long on the under-performing stock, expecting the prices to eventually converge again (Sarmento & Horta, 2020b). Pairs trading works on the assumption that the prices of the stocks will revert to their historical correlation.

The correlation between two stocks refers to how similarly or differently their prices move in relation to each other. A positive correlation means that both stocks tend to move in the same direction, while a negative correlation implies that they move in opposite directions.

- **A correlation of +1** means the two stocks move perfectly in tandem.
- **A correlation of -1** means the two stocks move in exactly opposite directions.

- **A correlation close to 0** means that the two stocks have little to no relationship in their price movements.

This relationship can be represented visually through scatter plots or line charts, and the correlation coefficient can be calculated using historical price data.

Machine learning can be applied to pairs trading in several ways, including:

- **Stock Selection and Pair Identification:** ML algorithms can analyze historical data to identify stock pairs that show high correlation or co-integration. Techniques such as clustering, PCA (Principal Component Analysis), or regression models can be used.
- **Predicting Divergence and Convergence:** Using supervised learning algorithms (such as Random Forest, Support Vector Machines) or unsupervised learning techniques, machine learning can predict when a pair's price will diverge and when it is likely to converge back [3]. Time-series forecasting methods like LSTM (Long Short-Term Memory) networks can also be useful in predicting future price movements.
- **Risk Management and Execution:** ML models can optimize trade execution by predicting price slippage, reducing transaction costs, and managing risks dynamically.

A well known approach detailed in one of many researches employs the Kalman Filter to develop a quantitative pairs trading strategy using linear state space models. They modeled the spread between paired assets as a mean-reverting process utilizing a Gaussian linear state space model derived from Ornstein-Uhlenbeck process [6]. The Kalman filter plays a pivotal role in their strategy as estimating the spread is important to capture price movement. Using the probability and statistics, pricing the model based on a set standard deviation away from regular price can indicate a stock being overbought or oversold. A part of the model is the dynamic hedging ratio. The hedging ratio represents the proportion of a short position in one security needed to hedge the exposure from a long position in another security. This ratio is dynamic because it evolves over time in response to changes in the relationship between the securities.It continuously updates the hedging ratio as new data becomes available, using a state-space approach [7]. This project utilizes this model to generate meaningful returns. Linear Regression is a statistical method used to model the relationship between two or more variables by fitting a linear equation to observed data [9]. In the context of pairs trading, linear regression is employed to analyze the relationship between the price of two stocks as well. Just like with Kalman Filtering, linear regression also uses a hedging ratio. The foundations of linear regression and Kalman filtering are completely different related to its purpose and assumptions. Linear regression is assumed to be static and serves to explain the impact of an independent variable on a dependent variable as a function. Linear regression can be used for interpreting past events in which changes between variables were stable. For instance, such methods can help evaluate the average relationship between the two stocks within a certain time frame. Kalman filtering, on the other hand, is a time dependent, recursive and self improving algorithm which helps in estimating the present condition of the system over a span of time, in the presence of uncertainty, faults and noise. In comparison to linear regression, Kalman filtering is unique in a sense that it does not remain stable in a sense that when new data arrives, it can adjust its responses according to the new information. This is useful in times where real time processing is required, such as financial markets, as the relationships are always changing between the variables during times. Linear regression provides a mere ‘still image’ with regards to time, in this case progression. Kalman filtering allows one to assess the system at any period and track changes that can happen.

Dataset Description

The Kaggle dataset contains the following: Data, Open, High, Low, Close, Adj Close, Volume. For the stocks, we can preprocess the data to find the rolling correlation between two assets over a moving time window. The difference in the price of the two assets can be monitored. Volatility is another feature that can be calculated using the prices. The Alpaca market API can be used to get concurrent information for pairs to find the price correlation.

Links:

- Alpaca Market API
- Kaggle Stock Market Dataset

Feature Engineering

To improve the predictive capability of the models, the following features were engineered:

- **Daily Return:** Percentage change in the closing price from the previous day.
- **Moving Averages (MA):** Calculated over different windows:
    - MA-5, MA-10, MA-20: Simple moving averages over 5, 10, and 20 days, respectively.
- **Exponential Moving Average (EMA):** A 20-day exponential moving average of the closing price.

- **Bollinger Bands (BB):**
  - BB_upper, BB_lower: Calculated as the 20-day moving average plus/minus two standard deviations.
- **Volatility-10:** Standard deviation of daily returns over a 10-day window.
- **Relative Strength Index (RSI):** A momentum oscillator that measures the speed and change of price movements, calculated over a 14-day window.
- **Volume Moving Average (Volume MA-20):** A 20-day moving average of the volume.
- **High-Low Percentage:** Difference between high and low prices as a percentage of the low price.
- **Close-Open Percentage:** Percentage difference between closing and opening prices.
- **Rate of Change (ROC-10):** Percentage change in closing price over a 10-day period.
- **Log Returns:** Natural log of the ratio of the current closing price to the previous day's closing price.

Problem Definition

The primary research questions for this project include:

- **Can machine learning predict profitable entry and exit points for pairs trading based on historical price data?**

  This question investigates whether ML models, trained on historical price patterns, can accurately forecast moments when a stock pair's price divergence is likely to revert. By timing trades precisely, the goal is to enhance profitability.

- **What machine learning techniques are most effective for improving pairs trading strategies?**

  This question explores the most effective ML methods for identifying pairs, forecasting price movements, and optimizing trades. Techniques such as clustering, time-series analysis, and reinforcement learning will be analyzed.

- **How does ML affect the risk and return profile of pairs trading compared to traditional methods?**

This question seeks to evaluate the advantages of ML-based strategies over traditional statistical models in terms of risk and return. It examines if ML can better manage risk while achieving profitability.

Motivation

Pairs trading is often viewed as a low-risk strategy, but it faces challenges in selecting optimal pairs and timing trades. Machine learning offers promising solutions to address these challenges, making the strategy more efficient and potentially more profitable. By utilizing ML, this project aims to optimize pair selection and trade timing, which are critical to maximizing returns and minimizing risk.

Methods

Data Preprocessing

- **Normalization/Scaling:** Due to the variance in stock prices across companies, it is essential to normalize and scale data. Min-Max Scaling (0-1 range) and Z-score normalization are used to standardize data, enabling algorithms to focus on relationships without price magnitude interference.
- **Missing Data Interpolation:** Gaps in stock data from market holidays or incomplete records can disrupt calculations and training. Forward filling missing values with the previous data point is an effective solution.
- **Feature Engineering:** New features, such as price ratios, moving averages, volatility, and log returns, are engineered to uncover stock pair relationships and aid in identifying key trading moments.
- **ADF Test:** The Augmented Dickey-Fuller (ADF) test is used to check for stationarity in time series data, indicating whether a stock price series is mean-reverting and suitable for pairs trading.

ML Algorithms/Models

- **Support Vector Machines (SVM):** SVMs are used to classify "buy" or "sell" states based on price spreads or other features, finding the optimal hyperplane for accurate separation.
- **Random Forest:** An ensemble of decision trees that improves prediction accuracy, useful for predicting spreads or classifying trading signals.
- **Gradient Boosting Machines (GBM):** Models like XGBoost and LightGBM use an ensemble of trees to detect complex patterns in security relationships, enhancing predictive accuracy.
- **K-Nearest Neighbors (KNN):** KNN can classify states by comparing them to historical patterns, potentially identifying similar past trading opportunities.
- **LSTM:** Long Short-Term Memory networks, a type of recurrent neural network, capture complex relationships over time between asset prices, revealing advanced patterns.
- **Linear Regression:** This model identifies linear relationships between asset prices, establishing the historical price correlation as a baseline for future forecasts.

Results and Discussion

Quantitative metrics to evaluate the project include:

- **Sharpe Ratio:** Measures the risk-adjusted return, with SOFR as the risk-free rate.
- **Cumulative Return:** Tracks the total portfolio return during backtesting.
- **Mean Squared Error (MSE):** Measures the accuracy of spread forecasts.
- **Alpha and Beta:** Alpha indicates performance relative to a benchmark, while beta measures volatility compared to the market.

Project Goals

The objective is to predict optimal entry and exit points for trading pairs by analyzing the price spread using ML. Our goal is to enhance pair selection while minimizing risk, with backtesting to evaluate quantitative metrics. A secondary goal is to explore reinforcement learning for adapting strategies to market conditions.

Expected Results

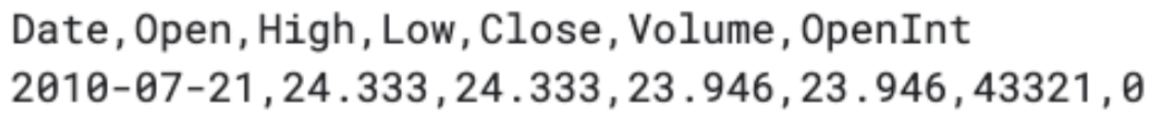
The expected outcome is to identify highly correlated pairs reflected by cumulative returns in backtesting. The strategy should remain market-neutral with a low beta and strong quantitative metrics, including low MSE for spread forecasts. Ultimately, the goal is a profitable, risk-adjusted return rate.

<>

Final Report

Data

We decided to switch our dataset to the “Huge Stock Market Dataset” on Kaggle, which contains more recent information, aiding in our backtesting and model training. The dataset contains both ETF and single equity data, but we only used the single equity data. The dataset originally contained 7000 text files, which we converted to CSV files.



```
Date,Open,High,Low,Close,Volume,OpenInt
2010-07-21,24.333,24.333,23.946,23.946,43321,0
```

Feature Engineering

The original dataset contained only a limited set of features, primarily basic price and volume data. While informative, these features alone were insufficient for effectively training our clustering models, as they did not capture essential quantitative metrics related to price volatility, trend momentum, and trading behavior. Therefore, as outlined in our proposal, we performed extensive feature engineering to enhance the dataset's depth for clustering.

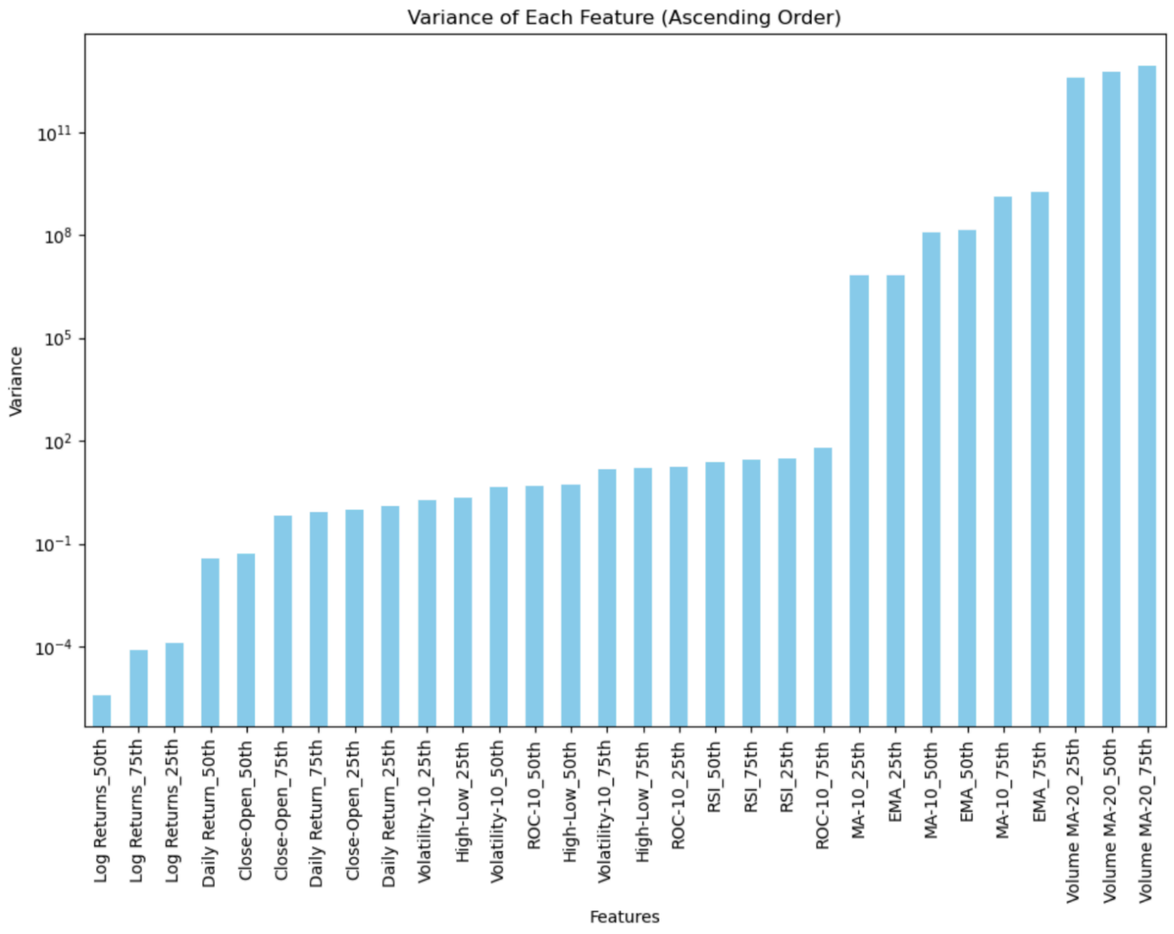
To build more robust features, we introduced several new quantitative features. Key additions included moving averages over multiple time windows (e.g., 5, 10, and 20 days) to smooth out price fluctuations and reveal long-term trends, as well as exponential moving averages (EMAs). By incorporating Bollinger Bands and volatility metrics, we enhanced our ability to assess price variability and detect breakout points, which are critical for understanding stock behavior across market conditions (Elliott, Van Der Hoek, & Malcolm, 2005).

Furthermore, we added momentum indicators like the Relative Strength Index (RSI) and Rate of Change (ROC), which measure price momentum and help distinguish between bullish and bearish trends. To capture intraday price dynamics, we calculated percentage changes between daily high and low prices (High-Low %) and closing and opening prices (Close-Open %). We also transformed price data into log returns for a statistically sound measurement of continuous compounding returns, making the dataset suitable for models that assume normally distributed returns.

Additionally, volume-based features such as the Volume Moving Average allowed us to incorporate trading activity trends, highlighting periods of increased buying or selling pressure. Each of these engineered features provided insights into the underlying market structure and behavior, enhancing the dataset’s utility for clustering.

Feature Selection

In clustering, variances of features play a significant role in determining how well the data can be grouped into clusters (Chang, Man, Xu, & Hsu, 2020). Variance measures how features differ from each other, with high variance features providing a more distinct separation across data points. To find the optimal features, we used a variance threshold to filter out features that do not vary much, as low variance features don’t effectively separate the data points for clustering (Chang, Man, Xu, & Hsu, 2020).

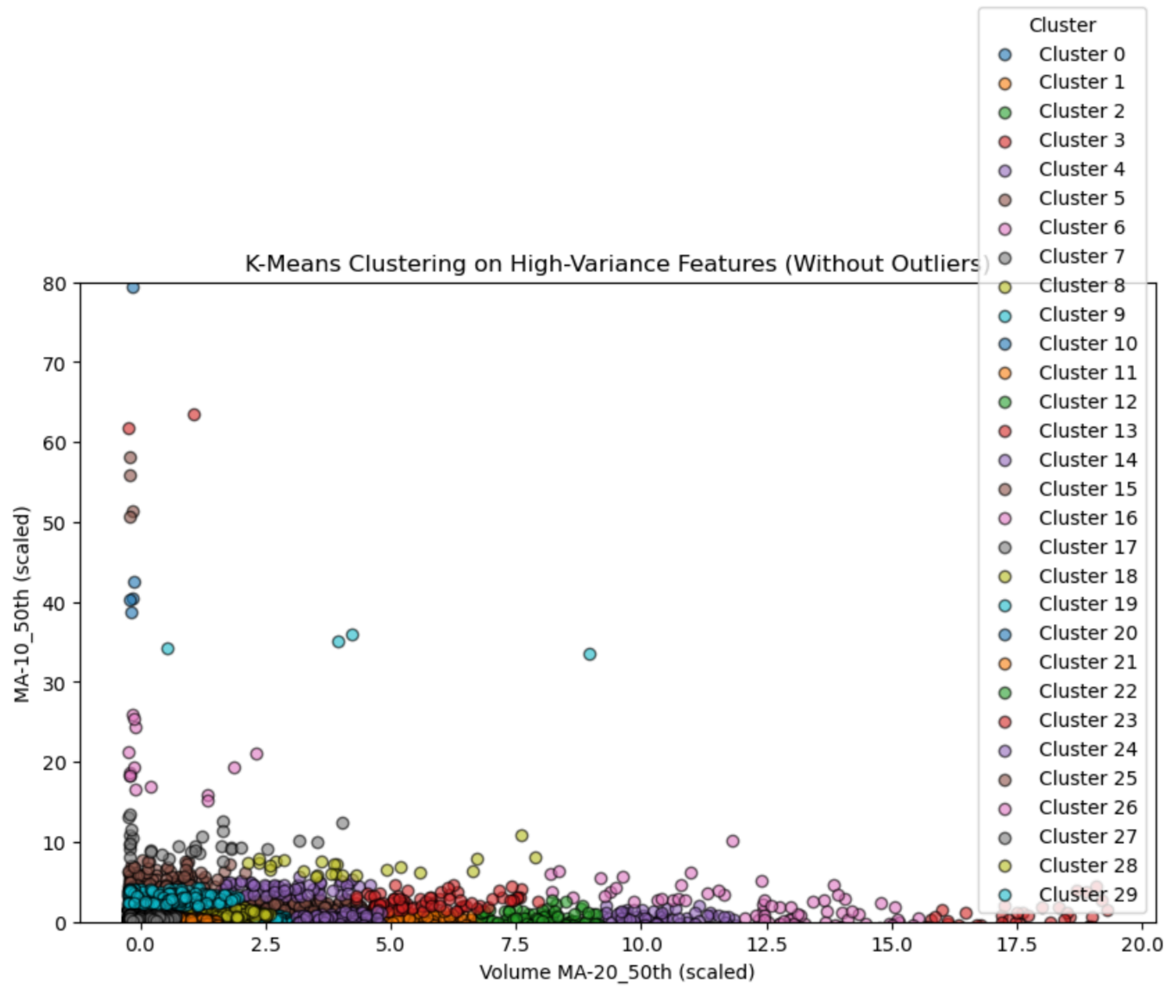


The graph above shows the variance for each feature in our dataset. We aimed to condense the time series data into summary statistics without using the full time series to simplify the data. For each time series metric, like daily return, moving average, exponential moving average, volatility, etc., we calculated the 25th percentile, 50th percentile, and 75th percentile values over a two-year time span. These percentiles provided the distribution of each metric. Taking these percentile values, we created features that represent the range and central tendencies. While it condenses the data effectively, it sacrifices temporal details. By reducing each time series to a few key percentiles, we lose the ability to capture day-to-day patterns, which would offer more insights for clustering.

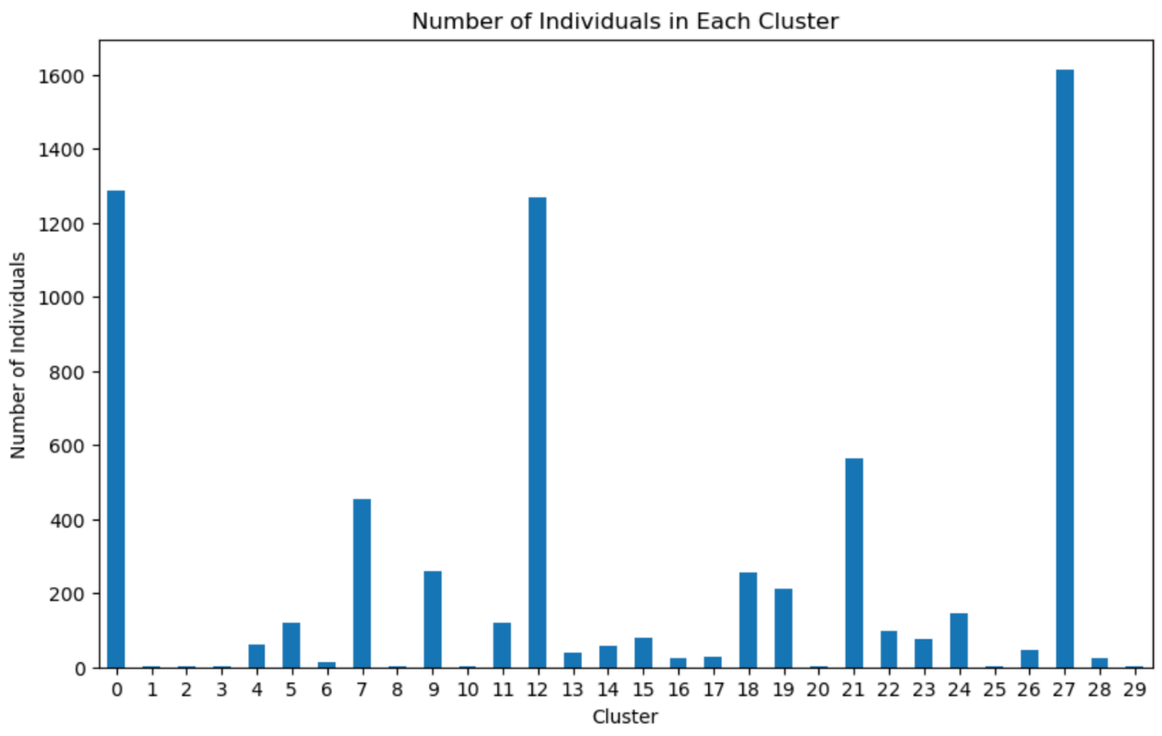
Clustering Techniques

K-Means

K-Means clustering is an unsupervised machine learning algorithm used to group similar points of data based on their features. It organizes points to minimize differences within clusters while maximizing the distance between each cluster. The number K is the number of clusters or centroids. Each point is assigned to the nearest cluster, and the centroids are recalculated. This algorithm stops when the clusters stabilize.

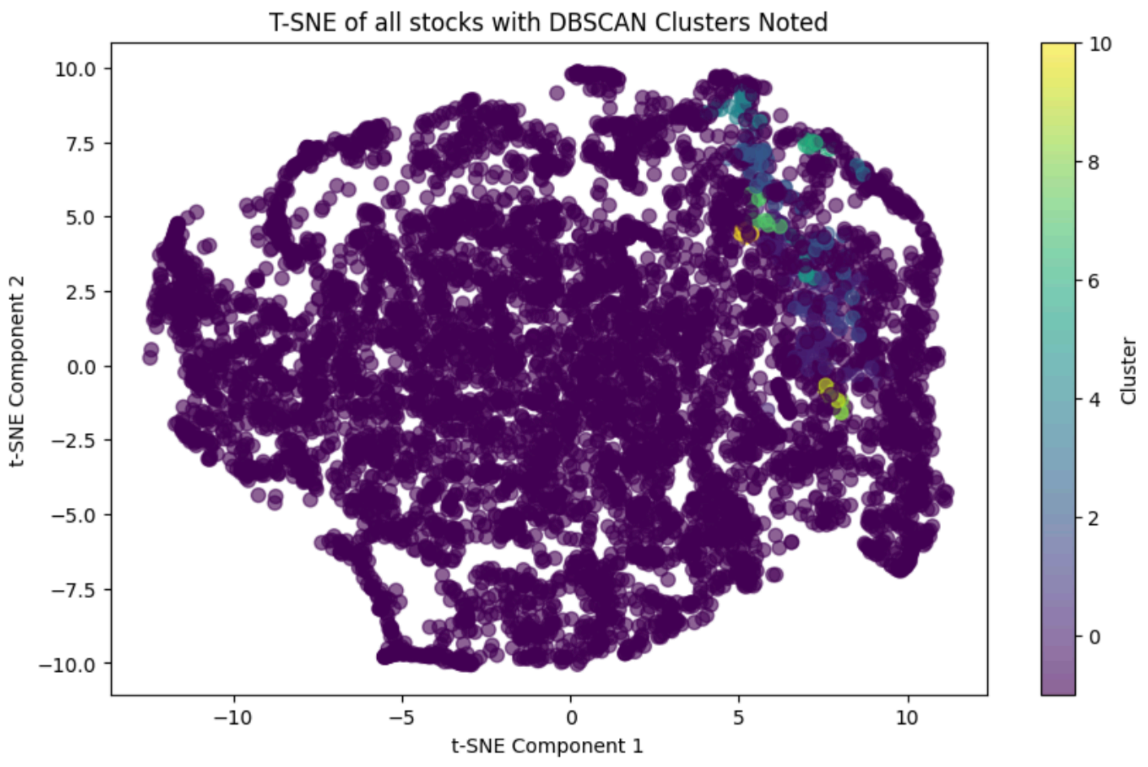


First, we loaded the dataset and removed any missing values. Only numeric features were used for clustering. Using the 20-day volume moving average (50th percentile) and 10-day moving average (50th percentile), we chose 30 clusters. The features were standardized to have a mean of 0 and a standard deviation of 1, which is important as it treats the features equally when calculating distances between points. We used a Silhouette score to determine the quality of clusters, where a higher score indicates a better-defined cluster.



DBSCAN

DBSCAN is an unsupervised machine learning algorithm particularly effective for identifying clusters in data with varying densities and for dealing with noisy data. Unlike K-Means, DBSCAN does not require the number of clusters to be specified in advance. Instead, it groups points based on density, identifying regions of high point concentration and classifying sparse points as noise.

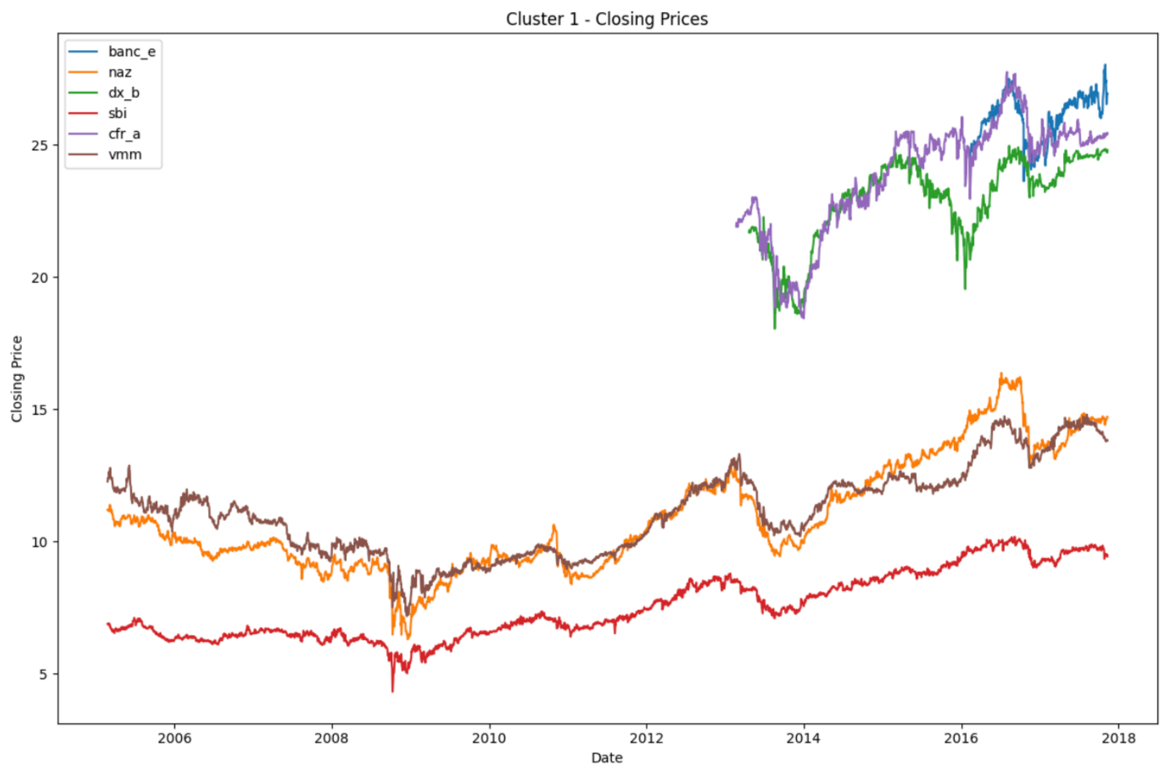


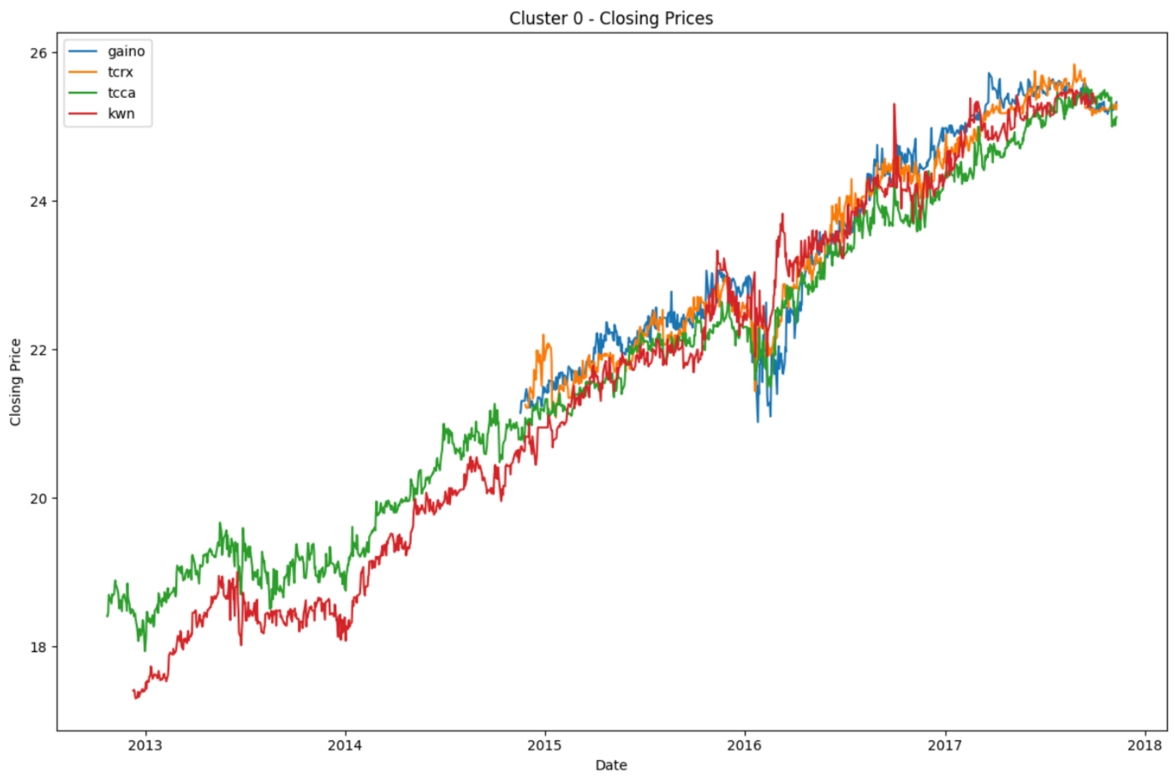
For this analysis, we first transformed our high-dimensional feature space into a 2D space using t-SNE (t-distributed Stochastic Neighbor Embedding), with parameters set to TSNE(n_components=2, perplexity=30, n_iter=300). t-SNE is a powerful tool for visualizing high-dimensional data, preserving relative distances and ensuring that similar data points are closer in the new 2D representation.

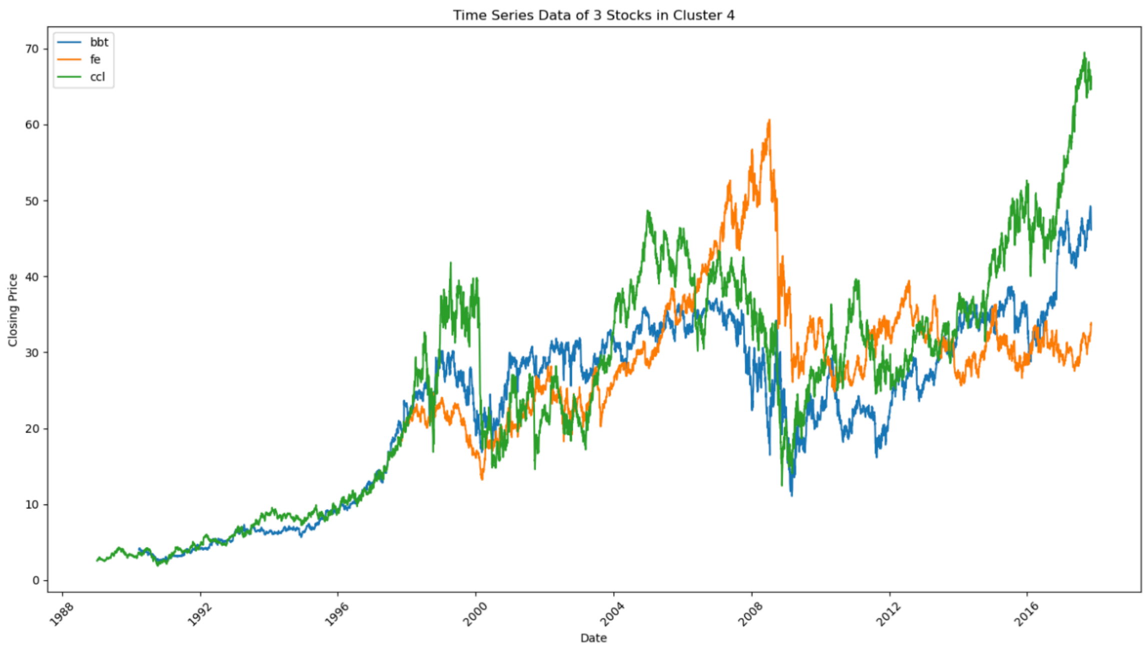
We then applied DBSCAN to the t-SNE-transformed data with parameters eps=0.3 and min_samples=8. These values were chosen to balance the identification of meaningful clusters and minimize noise. DBSCAN effectively identified stocks with similar characteristics while isolating outliers, enabling a nuanced view of data that did not fit neatly into any specific cluster.

Quantitative Findings/Graphs

K-Means and DBSCAN clustering graphs represent all clusters formed. The graph shows the number of individuals in each cluster. Stocks within clusters exhibit similar patterns and are deemed as pairs.


Cluster 1 - Closing Prices
banc_e
naz
dx_b
sbi
cfr_a
vmm
Closing Price
Date
5
10
15
20
25
2006
2008
2010
2012
2014
2016
2018


Cluster 0 - Closing Prices
gaino
tcrx
tcca
kwn
Closing Price
Date
18
20
22
24
26
2013
2014
2015
2016
2017
2018



Linear Regression

Overview

For our supervised learning algorithm, we chose to use linear regression. Linear regression is better suited for modeling the direct relationship between two stocks as it can quantify how closely two data points move together. This will help quantify their dependency and calculate their residual spread.

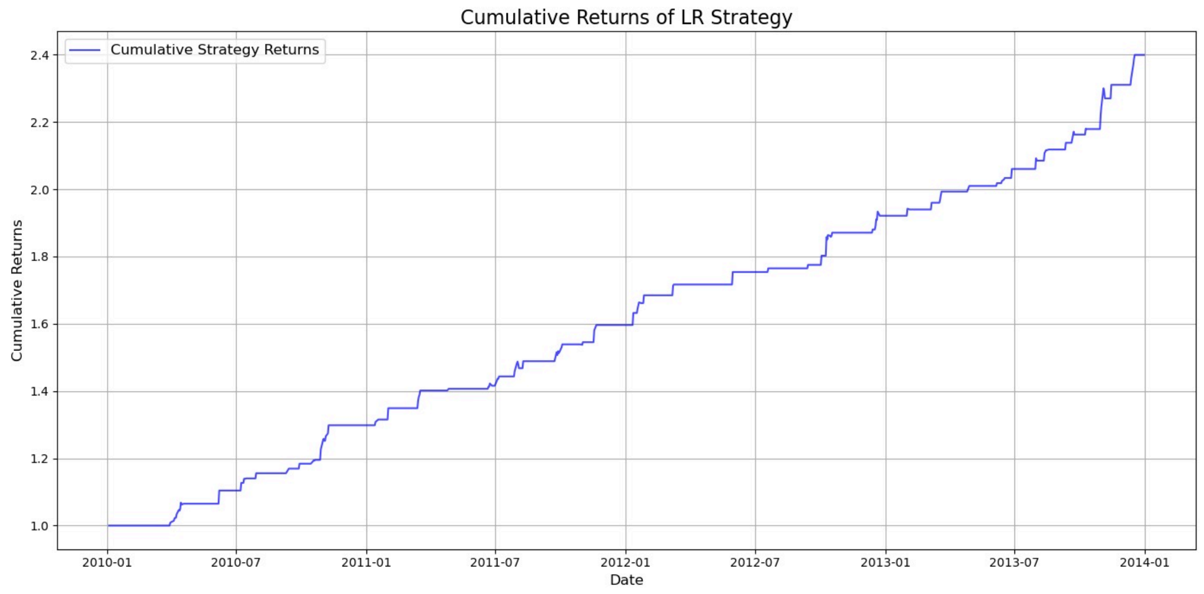
Data for Linear Regression Model

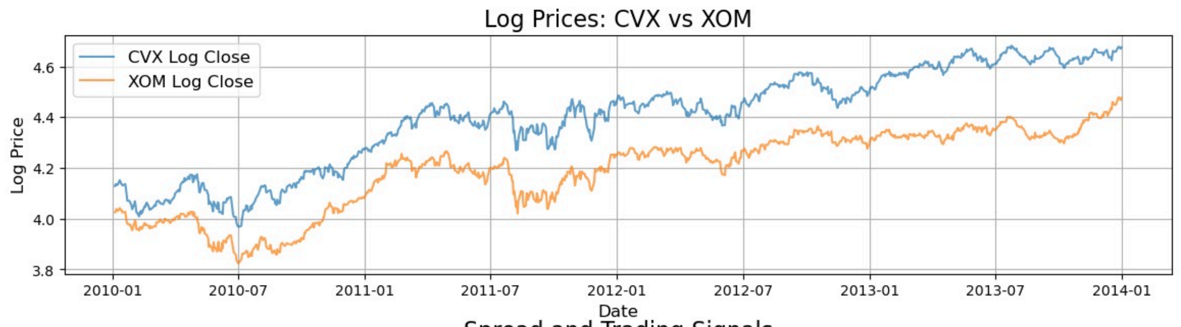
We chose to use two stocks (XOM and CVX) that were already known to be used in pairs trading. Log transformation was applied to the stocks' closing prices to stabilize variance and make the data more suitable for linear analysis. The log-transformed prices of the two stocks were then merged into their common date to align the data with their time-series. Then, the log-transformed closing prices of XOM were used as the independent variable, while the log-transformed closing prices of CVX were used as the dependent variable.

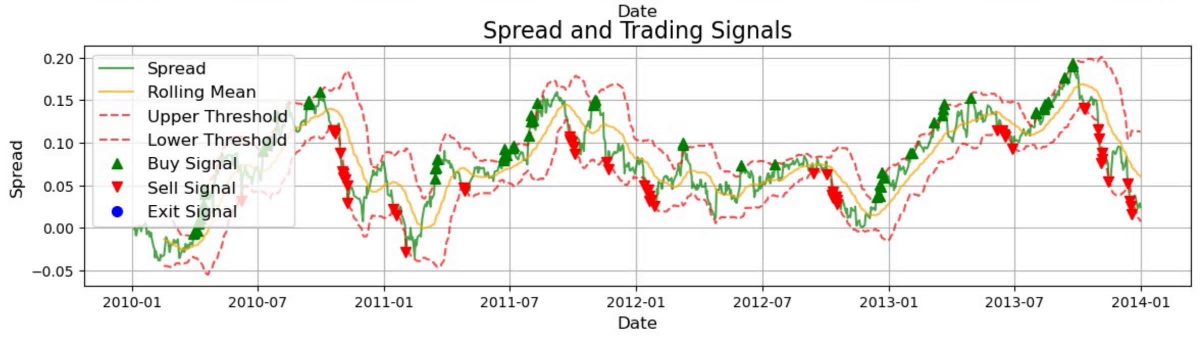
Model Training and Evaluation

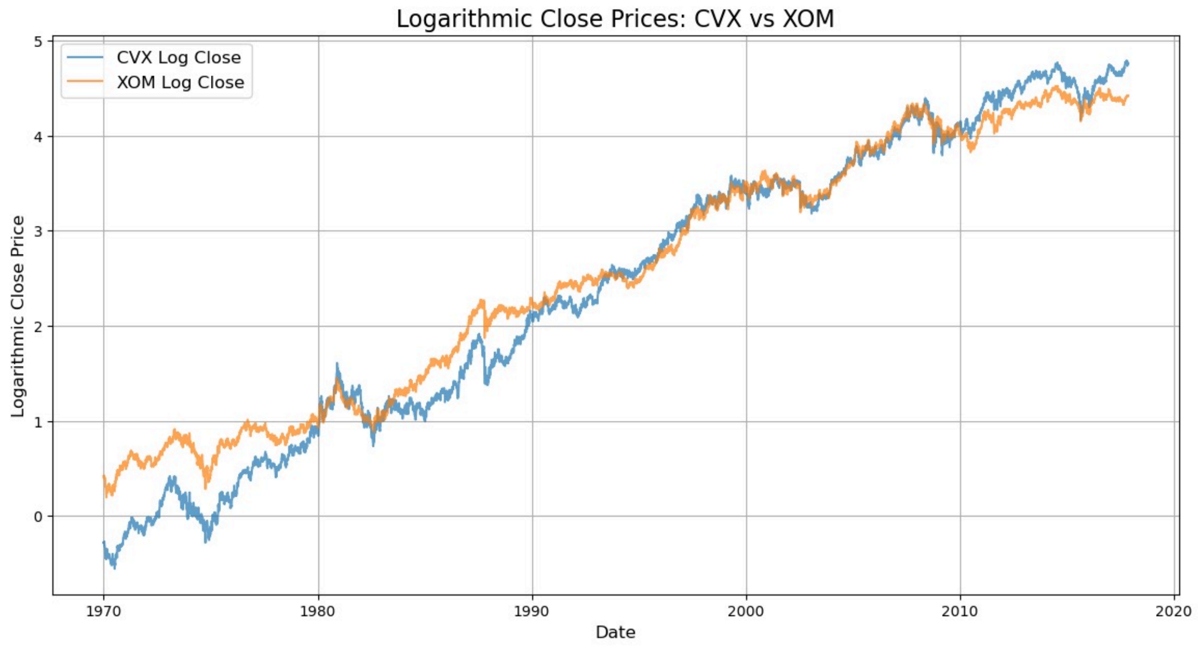
A linear regression model was trained on the data to model the relationship between the log-transformed prices. This model would provide us with a hedge-ratio and an intercept which represented the statistical relationship between two stocks. Then using the fitted regression model, the residuals (difference form predicted and actual) were calculated to capture deviations from the expected equillibrium.

We then normalized the residual spread using z-score transformation which standardizes values to identify deviations from the mean. We then were able to create trading signals based off the z-score thresholds. A z-score of -1.0 or lower indicated a long signal and indicated that the pair was likely to revert upward. A short signal of z-score higher than 1.0 indicated that the pair is overvalued and likely to revert downward. A hold signal would be when the z-score remained within the threshold.









Linear Regression Analysis

Overall, the linear regression model performed strongly, generating a 294.48% return over the four year time period. Below is a strengths/weaknesses analysis of the algorithm.

Strengths

Cumulative Return: A 294.48% return over the observed period of four yeras is a good result for a pairs trading strategy.

Sharpe Ratio: The linear regression model generated a Sharpe Ratio of 3.82, which indicates excellent risk-adjusted performance. A ratio above 3 is typically regarded as outstanding in financial contexts

Log Price and Spread Behavior: The signals generated in the graph above align well with the spread reverting to the mean, validating the linear regression model's ability to predict profitable trades

Weaknesses

Dataset Time: The results are based on historical data (2010 through 2014). For a thorough evaluation, it's essential to validate the strategy's robustness on out-of-sample data

Comparison of dbscan and k-means

K-Means is a powerful clustering method for well-structured data, DBSCAN's ability to identify irregularly shaped clusters, handle noise, and adapt without requiring a predefined number of clusters makes it a better choice for pairs trading under volatile conditions. DBSCAN's flexibility ensures more meaningful clustering in the context of financial relationships, which can ultimately lead to more accurate trading signals and better performance in dynamic markets.

Comparison of dbscan and linear regression

DBSCAN and Linear Regression can work together effectively in pairs trading, each addressing different aspects of the strategy. DBSCAN's ability to identify irregularly shaped clusters, handle noise, and adapt to dynamic relationships makes it ideal for the initial step of selecting potential pairs. Linear Regression then takes over to provide a quantitative framework for trade execution, determining optimal entry and exit points. By using these two methods in tandem, traders can build a robust and adaptive pairs trading strategy that is well-suited to navigating the complexities of financial markets.

Next Steps

To further enhance the pairs trading model, one could explore additional clustering methods like hierarchical clustering or Gaussian Mixture Models, apply feature engineering to derive new indicators, utilize advanced models such as LSTMs for forecasting, and conduct extensive backtesting with optimization techniques to ensure the robustness of the model.

References

- [1] S. M. Sarmento and N. Horta, *A Machine Learning Based Pairs Trading Investment Strategy*. SpringerBriefs in Applied Sciences and Technology, 2020. [Online]. Available: https://doi.org/10.1007/978-3-030-47251-1.
- [2] S. M. Sarmento and N. Horta, "Enhancing a Pairs Trading strategy with the application of Machine Learning," *Expert Systems with Applications*, vol. 158, p. 113490, 2020. [Online]. Available: https://doi.org/10.1016/j.eswa.2020.113490.
- [3] V. Chang, X. Man, Q. Xu, and C. Hsu, "Pairs trading on different portfolios based on machine learning," *Expert Systems*, vol. 38, no. 3, 2020. [Online]. Available: https://doi.org/10.1111/exsy.12649.
- [4] R. J. Elliott, J. Van Der Hoek, and W. P. Malcolm, "Pairs trading," *Quantitative Finance*, vol. 5, no. 3, pp. 271–276, 2005. [Online]. Available: https://doi.org/10.1080/14697680500149370.

- [5] R. T. Smith and X. Xu, "A good pair: Alternative pairs-trading strategies," *Finance Markets Portfolio Management*, vol. 31, no. 1, pp. 1–26, 2017. [Online]. Available: https://doi.org/10.1007/s11408-016-0280-x.
- [6] C. E. de Moura, A. Pizzinga, and J. Zubelli, "A pairs trading strategy based on linear state space models and the Kalman filter," *Quantitative Finance*, vol. 16, no. 10, pp. 1559–1573, 2016. [Online]. Available: https://doi.org/10.1080/14697688.2016.1164886.
- [7] M. J. Nourahmadi and M. Nourahmadi, "Application of Kalman Filter to Estimate Dynamic Hedge Ratio in Pairs Trading Strategy: A Case Study of the Automobile Industry," *Financial Research Journal*, vol. 21, no. 3, 2021. [Online]. Available: https://doi.org/10.22059/FRJ.2021.325988.1007206.
- [8] A. J. Patton, "Are 'Market Neutral' Hedge Funds Really Market Neutral?," *The Review of Financial Studies*, vol. 22, no. 7, pp. 2495–2530, 2009. [Online]. Available: https://doi.org/10.1093/rfs/hhn113.
- [9] B. Do, R. Faff, and K. Hamza, "A New Approach to Modeling and Estimation for Pairs Trading," 2006. [Online]. Available: https://doi.org/10.2139/ssrn.901289.

Contributions

Name	Final Contributions
Wesley Lu	Presentation Slide, K-mean Clustering, Kalman Filtering, Linear Regression
Matthew She	Presentation Slide, Data Processing, Feature Engineering
Matthew Lu	Presentation Slide, DBScan Clustering
Justin Lee	Presentation Slide, Data Processing, Feature Engineering
Victor Wu	Presentation Slide, Data Processing, Feature Engineering, Linear Regression

Gaant chart

GANTT CHART

PROJECT TITLE	Machine Learning for Trading Pairs

TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	M	T	W
Project Proposal							
Introduction & Background	Wesley Lu	9/27/2024	10/4/2024	7			
Problem Definition	Justin Lee	9/27/2024	10/4/2024	7			
Methods	Victor Wu	9/27/2024	10/4/2024	7			
Potential Results & Discussion	Mathew Lu	9/27/2024	10/4/2024	7			
Video Recording	Mathew She	9/27/2024	10/4/2024	7			
GitHub Page	Wesley Lu	9/27/2024	10/4/2024	7			
Model 1							
Data Sourcing and Cleaning	Wesley Lu	10/7/2024	10/15/2024	8			
Model Selection	Justin Lee	10/15/2024	10/18/2024	3			
Data Pre-Processing	Victor Wu	10/18/2024	10/25/2024	7			
Model Coding	Mathew Lu	10/25/2024	11/8/2024	13			
Results Evaluation and Analysis	Mathew She	11/1/2024	11/8/2024	7			
Midterm Report	Victor Wu	11/1/2024	11/8/2024	7			
Model 2							
Data Sourcing and Cleaning	Wesley Lu	10/18/2024	10/22/2024	4			
Model Selection	Justin Lee	10/22/2024	10/25/2024	3			
Data Pre-Processing	Victor Wu	10/25/2024	10/29/2024	4			
Model Coding	Mathew Lu	10/25/2024	11/19/2024	24			