Statistical Arbitrage by Pair Trading using Clustering and Machine Learning

Do Tuan Minh Department of University of XXX City, Province, Zipcode [example@email.com]

Nguyen Khanh Trung Department of XXX University of XXX City, Province, Zipcode [example@email.com]

Nguyen Kim Tuyen Department of XXX University of XXX City, Province, Zipcode [example@email.com]

Contents

[Problem Description 3](#_Toc127840637)

[Customer 3](#_Toc127840638)

[Product Manager 3](#_Toc127840639)

[Sales Manager 4](#_Toc127840640)

[Admin 4](#_Toc127840641)

[Project Organization 5](#_Toc127840642)

[Team Members 5](#_Toc127840643)

[Work Allocation 5](#_Toc127840644)

[Database Design 5](#_Toc127840645)

[Identify the purpose of the database 5](#_Toc127840646)

[Develop a conceptual data model 5](#_Toc127840647)

[Develop a logical data model 5](#_Toc127840648)

[Normalize the data model 5](#_Toc127840649)

[Develop a physical data model 6](#_Toc127840650)

[Main Functions 6](#_Toc127840651)

[Admin 6](#_Toc127840652)

[sys.new\_account 6](#_Toc127840653)

[sys.new\_customer 6](#_Toc127840654)

[Customer 7](#_Toc127840655)

[product.view\_all\_product() 7](#_Toc127840656)

[Product Manager 8](#_Toc127840657)

[Additional Features 8](#_Toc127840658)

[Data Scaling 9](#_Toc127840659)

[Sales Manager 10](#_Toc127840660)

[Co-Integration Test 14](#_Toc127840661)

[Link to Code and Data 15](#_Toc127840662)

[Conclusion 15](#_Toc127840663)

[Bibliography 15](#_Toc127840664)

# Problem Description

We are building a database to store information related to a Smart Devices Store. The store sells different kinds of products: smart phones, smart watches, tablets, smart TVs, headsets… Products are supplied by many suppliers but through only a single warehouse.

The SDSMS is designed to help store owners manage inventory, sales, customer data, and other important aspects of their business. The system can track the stock levels of individual products, generate reports on sales and inventory, and provide real-time data on customer behavior and preferences. This data can be used to make informed decisions about product selection, pricing, and marketing strategies.

There will be 4 types of users expected for our system: Admin, Customer, Product Manager and Sale Manager.

## Customer

In a smart device store, the customers usually follow a specific course of actions when they engage with the store. First, they search for the specific product they need by using different criteria, such as product name, brand, or category. Once they find the desired product, they can choose from a selection of pre-determined configurations, which could include different colors, memory sizes, storage capacities, and other features. After selecting the desired configuration, the customer can add the product to their cart and continue browsing the store for additional items. They can repeat this process until they have selected all the items they need and are ready to complete the purchase.

## Product Manager

As a product manager, one of the primary responsibilities is to manage all product-related information for the smart device store. This includes various aspects such as:

* Products: The product manager is responsible for overseeing the product portfolio of the store, which includes managing the product information such as product names, descriptions, prices, and other relevant details.
* Configurations: The product manager is responsible for developing and managing the configurations available for each product. They will determine the options available to customers, such as color, memory, storage capacity, and other features.
* Stock: The product manager is responsible for monitoring and managing the stock levels for each product, ensuring that the store has sufficient inventory to meet customer demand.
* Brands: The product manager will manage the brands of the products sold in the store, ensuring that they align with the store's brand and target market.
* Product Categories: The product manager will oversee the product categories and subcategories in the store, ensuring that they are well-organized and easy for customers to navigate.

## Sales Manager

As a key member of the smart device store team, the sales manager is responsible for tracking and analyzing sales performance in order to ensure that the store is meeting its revenue targets and identify areas for improvement. To do this, the sales manager will evaluate sales performance through various sales analysis reports, which could include:

* Product Revenue: The sales manager will review sales data to analyze revenue generated by each product. By comparing revenue figures for different products, the sales manager can determine which products are the most popular and profitable, and use this information to make informed decisions about product selection, pricing, and promotions.
* Customer Revenue: The sales manager will track sales revenue on a per-customer basis to identify the most valuable customers. By analyzing customer revenue data, the sales manager can determine which customers generate the most revenue for the store and develop strategies to retain and upsell these customers.
* Staff Revenue: The sales manager will track and compare the sales revenue generated by different staff members. By identifying which staff members are the most successful in selling products, the sales manager can provide additional training and coaching to other staff members to improve their sales performance.
* Product Total Sold: The sales manager will track the number of products sold in order to identify which products are the most popular. By analyzing this data, the sales manager can determine which products are the most in demand and adjust inventory levels and product displays accordingly.

Using this sales analysis data, the sales manager can identify top-performing products, customers, and staff members, and use this information to make informed decisions to improve the store's overall sales performance.

## Admin

As the administrator of a smart device store's management system, one of the primary responsibilities is to manage user accounts and privileges. This involves creating user accounts for all users, including sales managers, product managers, and customers, and ensuring that each user has the appropriate level of access to the system.

The administrator must ensure that each user has a unique login and password that they can use to access the system. They must also assign the appropriate user roles and privileges to each user, depending on their job responsibilities and level of access required.

For example, the administrator may assign a sales manager with the ability to view and analyze sales reports and manage customer accounts, while a product manager may have the ability to update product information and manage inventory levels. Customers, on the other hand, would have more limited access to the system, allowing them to browse and purchase products.

# Project Organization

## Team Members

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Name** | **Student ID** | **Role** |
| 1 | Đỗ Tuấn Minh | 20200390 | Member |
| 2 | Nguyễn Thị Phương Thảo | 20205194 | Member |
| 3 | Nguyễn Khánh Trung | 20205133 | Leader |

## Work Allocation

# Database Design

## Identify the purpose of the database

The purpose of the database is to manage and store information related to a Smart Devices Store, including inventory, sales, customer data, and other important aspects of the business. The database is designed to track the stock levels of individual products, generate reports on sales and inventory, and provide real-time data on customer behavior and preferences. The database will be used by 4 types of users: Admin, Customer, Product Manager, and Sales Manager. The database will help the store owners make informed decisions about product selection, pricing, and marketing strategies.

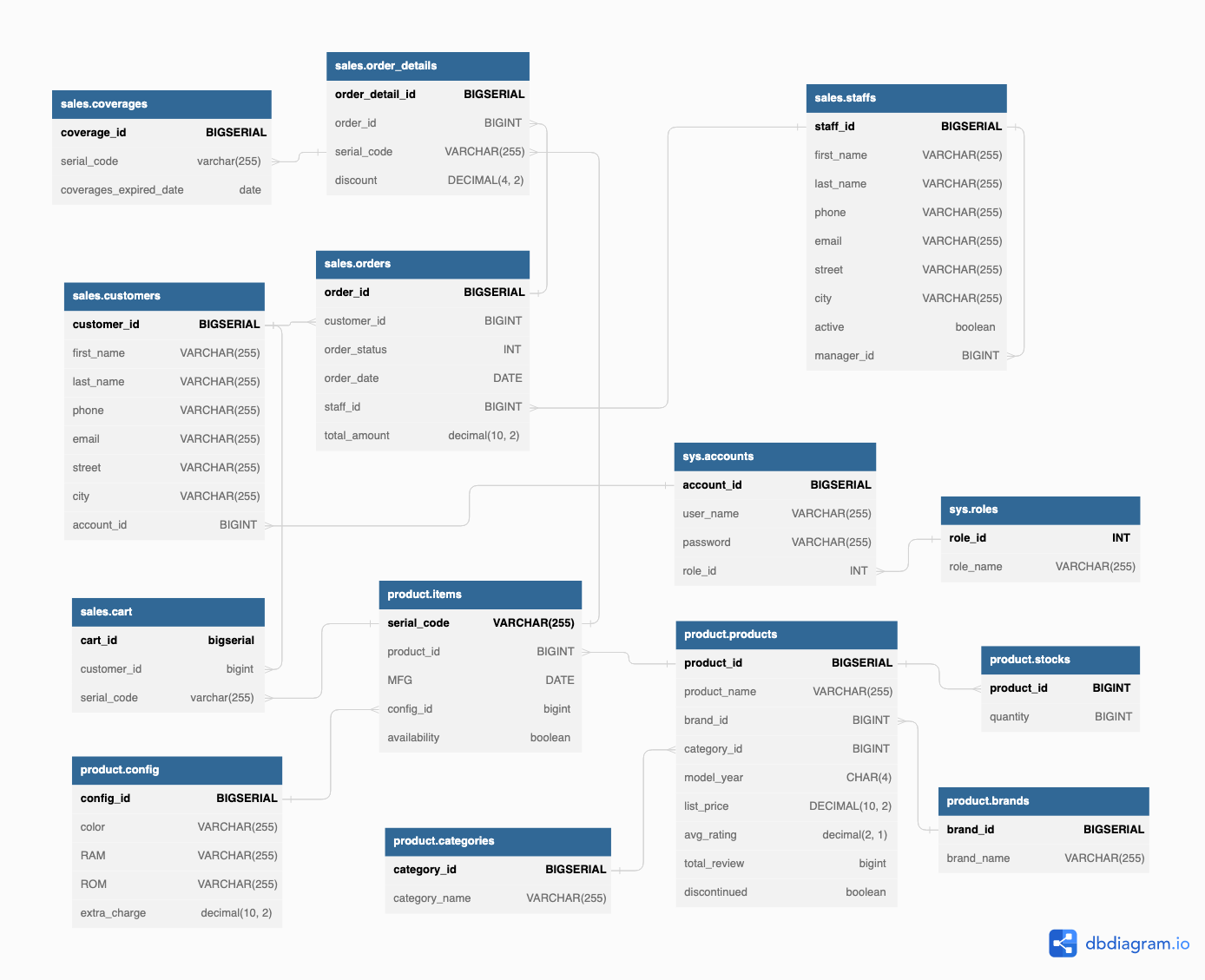
## Develop a conceptual data model

This step involves creating a conceptual data model that defines the high-level data structures and relationships that the database will need to support. This can be done using a tool such as an entity-relationship (ER) diagram that visually represents the entities and relationships involved in the database.

## Develop a logical data model

The next step is to create a logical data model, which involves defining the specific data elements that will be stored in the database and how they will be organized. This typically involves creating tables, defining columns and data types, and establishing relationships between tables.

## Normalize the data model

This step involves normalizing the data model to remove any redundancies and ensure that data is stored in the most efficient way possible. This can involve splitting large tables into smaller ones, establishing foreign keys to enforce data integrity, and other techniques to ensure data consistency. 

## Develop a physical data model

The final step is to create a physical data model, which involves implementing the database design in a specific database management system. This includes defining the database schema, creating tables and indexes, and establishing security and access controls.

# Main Functions

## Admin

### sys.new\_account

### sys.new\_customer

CREATE OR REPLACE PROCEDURE sys.new\_customer(first\_name VARCHAR(255),last\_name VARCHAR(255),phone VARCHAR(255),email VARCHAR(255),street VARCHAR(255),city VARCHAR(255),username VARCHAR(255), password VARCHAR(255))

LANGUAGE plpgsql

AS $$

DECLARE account\_id1 BIGINT ;

BEGIN

    --INSERT INTO TABLE accounts

    CALL sys.new\_account(username,password,0);

    SELECT account\_id INTO account\_id1 FROM sys.accounts WHERE user\_name=username;

    -- INSERT INTO TABLE customer

    INSERT INTO sales.customers(first\_name,last\_name,phone,email,street,city,account\_id) VALUES (first\_name,last\_name,phone,email,street,city,account\_id1);

END;

$$;

## Customer

### product.view\_all\_product

CREATE OR REPLACE FUNCTION product.view\_all\_product()

RETURNS TABLE(product\_id BIGINT, product\_name VARCHAR(255), brand\_name VARCHAR(255), category\_name VARCHAR(255), model\_year CHAR(4), list\_price DECIMAL(10,2), avg\_rating DECIMAL(2,1), total\_review BIGINT, discontinued boolean)

LANGUAGE plpgsql

AS $$

BEGIN

    RETURN QUERY

    SELECT p.product\_id product\_id,

        p.product\_name product\_name,

        b.brand\_name brand\_name,

        c.category\_name category\_name,

        p.model\_year,

        p.list\_price,

        p.avg\_rating,

        p.total\_review,

        p.discontinued

    FROM product.products p

        JOIN product.brands b USING (brand\_id)

        JOIN product.categories c USING (category\_id);

END;

$$;

### product.view\_active\_product

CREATE OR REPLACE FUNCTION product.view\_active\_product()

RETURNS TABLE(product\_id BIGINT, product\_name VARCHAR(255), brand\_name VARCHAR(255), category\_name VARCHAR(255), model\_year CHAR(4), list\_price DECIMAL(10,2), avg\_rating DECIMAL(2,1), total\_review BIGINT)

LANGUAGE plpgsql

AS $$

BEGIN

    RETURN QUERY

    SELECT v.product\_id ,

        v.product\_name ,

        v.brand\_name ,

        v.category\_name ,

        v.model\_year,

        v.list\_price,

        v.avg\_rating,

        v.total\_review

    FROM product.view\_all\_product() v

    where v.discontinued = false;

END;

$$;

## Product Manager

### Additional Features

*Mean Returns*

A return is the change in price of an asset, investment, or project over time, which may be represented in terms of price change or percentage change.

Returns are often annualized for comparison purposes, while a holding period return calculates the gain or loss during the entire period an investment was held.

*Volatility*

One way to measure an asset's variation is to quantify the daily returns (percent move on a daily basis) of the asset. Historical volatility is based on historical prices and represents the degree of variability in the returns of an asset. This number is without a unit and is expressed as a percentage.

While variance captures the dispersion of returns around the mean of an asset in general, volatility is a measure of that variance bounded by a specific period of time. Thus, we can report daily volatility, weekly, monthly, or annualized volatility. It is, therefore, useful to think of volatility as the annualized standard deviation.

where:

v = volatility over some interval of time

σ = standard deviation of returns

T = number of periods in the time horizon

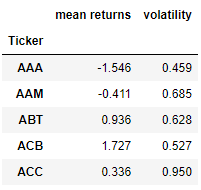


Figure : The mean returns and volatility dataframe example of the first interval

*Industry*

Business analysts often classify stocks into industry groups primarily based on similarity in revenue lines. Stocks of similar industries should be related in the future. Our industry data are crawled from <https://finance.vietstock.vn/>.

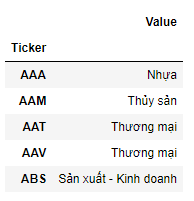


Figure : The industry dataframe

Most Machine Learning algorithms cannot work with categorical data and needs to be converted into numerical data. Our approach to this is using one-hot encoding, a technique used to represent categorical variables as numerical values in a machine learning model.

### Data Scaling

As we know, most of the machine learning models learn from the data by the time the learning model maps the data points from input to output. And the distribution of the data points can be different for every feature of the data. Larger differences between the data points of input variables increase the uncertainty in the results of the model.

The machine learning models provide weights to the input variables according to their data points and inferences for output. In that case, if the difference between the data points is so high, the model will need to provide the larger weight to the points and in final results, the model with a large weight value is often unstable. This means the model can produce poor results or can perform poorly during learning.

So if the data in any conditions has data points far from each other, scaling is a technique to make them closer to each other or in simpler words, we can say that the scaling is used for making data points generalized so that the distance between them will be lower.

Normalization and Standardization are the two main methods for the scaling of the data. Which are widely used in the algorithms where scaling is required. In our case, sklearn StandardScaler is prefered as it is more useful in classification problem and when the data have negative values.

## Sales Manager

Density-based spatial clustering of applications with noise (DBSCAN) is a data clustering algorithm proposed by Martin Ester, Hans-Peter Kriegel, Jorg Sander and Xiaowei Xu in 1996 (Ester, 1996). It is a density-based clustering non-parametric algorithm which can be described in the following steps [9]:

1. Find the points in the neighborhood of every point, and identify the core points with more than min neighbors, where min is a parameter to be tuned.
2. Find the connected components of core points on the neighbor graph, ignoring all non-core points.
3. Assign each non-core point to a nearby cluster if the cluster is an neighbor, otherwise assign it to noise.

Compared with K-Means, DBSCAN has advantages in our use case. Specifically, DBSCAN does not cluster all stocks,i.e. it leaves out stocks which do not neatly fit into a cluster, and the number of clusters does not need to be specified.

Once the data is clustered, we need a way to visualize the high dimensional data and its clusters into a two-dimensional graph. One approach to this visualization is using the nonlinear dimensionality technique known as t-Distributed Stochastic Neighbor Embedding (t-SNE) (Hakon Andersen, 2018).

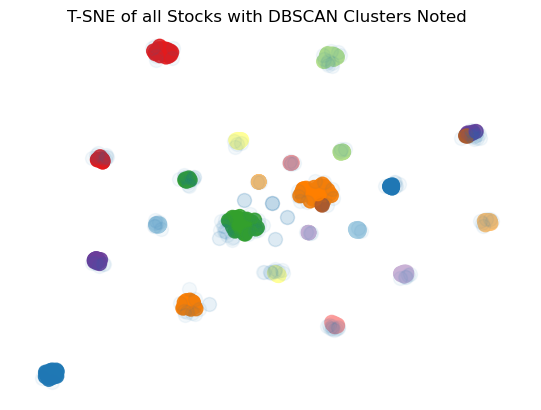


Figure : Visualization of DBSCAN clusters of the first interval

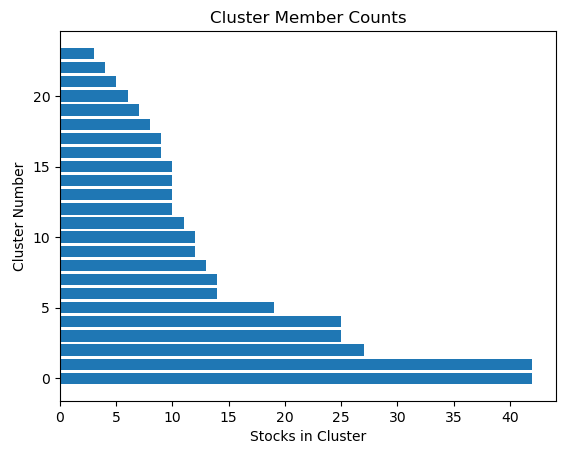


Figure : Stock number counted of DBSCAN clusters of the first interval

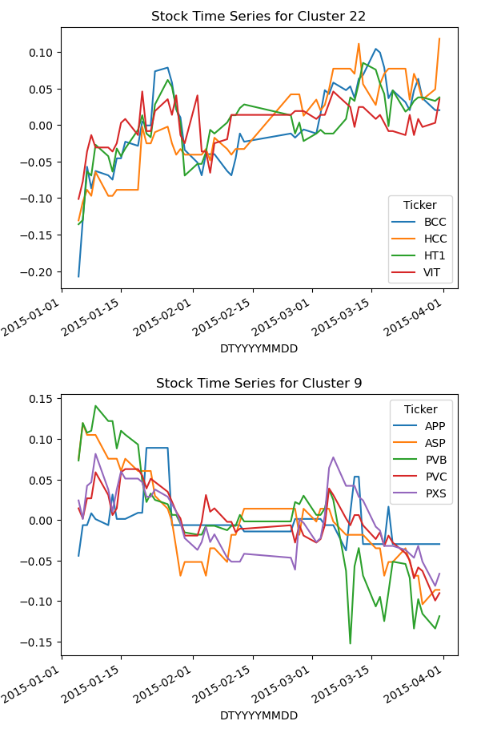


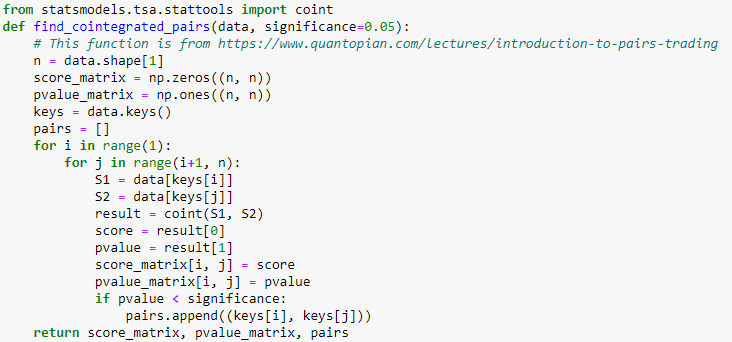
Figure : Time Series of some clusters noted

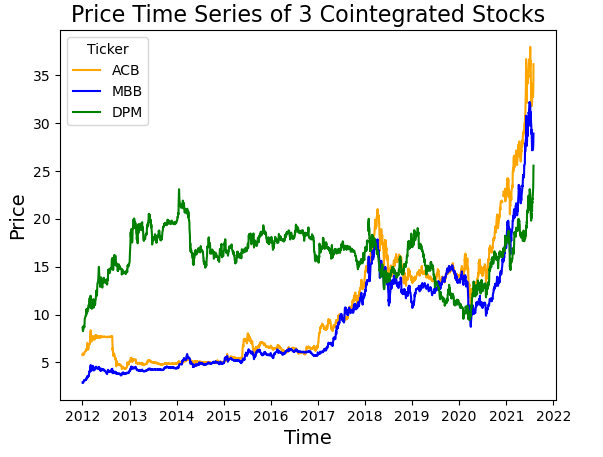
## Co-Integration Test

A time series is stationary when the parameters of the underlying data do not change over time (Wooldridge, 2009). While cointegration implies that 2 time series and share similar stochastic trends, and they never diverge too far from each other. The cointegrated variables exhibits a longterm equilibrium relationship defined by , where is the equlibrium error, which represents short-term deviations from the long-term relationship (Wooldridge, 2009).

For pairs trading, the intuition is that if we find two stocks and whose prices are cointegrated, then any short-term deviations from the spread mean, , can be an opportunity to place trades accordingly, as we bet on the relationship to be mean reverting. Pairs are deemed as cointegrated when they aren’t stationary and tend to move together.

The statsmodels.tsa.stattools module in Python comes with , a handy function to verify cointegration between 2 time series , which implements the Augmented-Dickey-Fuller (ADF) test [2].





## Link to Code and Data

Source to Github Repo: [Pair-Trading-Analysis-Repo](https://github.com/ktrung1709/Pair-Trading-Analysis)

# Conclusion

# Bibliography

Ester, M. (1996). *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.* AAAI Press. Retrieved from https://dblp.uni-trier.de/db/conf/kdd/kdd96.html#EsterKSX96

Hakon Andersen, H. T. (2018). *Statistical arbitrage trading with implementation of machine learning: an empirical analysis of pairs trading on the Norwegian stock market.* MA Thesis.