📊 Stock Trading Day Clustering Project

🧠 Overview

This project uses unsupervised machine learning to explore how stock trading days behave. By looking at key features like returns, volatility, and trading volume, we group similar days together using clustering algorithms like K-Means, DBSCAN, and GMM. These clusters help identify patterns, such as “volatile high-volume gain days” or “quiet loss days,” which can help investors and analysts make more informed decisions.

🔍 Background & Concepts

🤖 What is Clustering?

Clustering is like sorting socks after laundry—but you don't know in advance how many types of socks you have. Instead, you look at characteristics (like color, size, material) and let a program group them based on similarity. In this project, each trading day is a "sock", and we group them based on things like return, volatility, and volume.

📌 Core Concepts

1. K-Means Clustering

Imagine you have a bunch of dots (data points) on a piece of paper. K-Means picks K centers and pulls each dot toward the closest center. It keeps adjusting the centers until everything is nicely grouped.

2. DBSCAN (Density-Based Clustering)

Think of people at a party. DBSCAN groups people standing close together into groups. If someone is standing alone or far from others, they’re called "noise" or an outlier.

3. Gaussian Mixture Models (GMM)

Instead of assigning each data point strictly to a group, GMM gives each day a probability of belonging to each group.

⚙️ Data Processing (Before Clustering)

We create new numbers (features) to describe each trading day. Think of them as different “dimensions” that describe market behavior.

🔬 Feature Engineering

Feature	Formula	What it tells us
daily_return	(Close - Open) / Open	Was the price up or down? Measures daily % change.
price_range	(High - Low) / Open	How much price moved during the day. Reflects range.
volatility	Standard deviation of [Open, High, Low, Close]	A measure of price "bounciness" during the day.

volume_change	Volume[t] / Volume[t-1] - 1	How much volume changed vs the previous day.
volume_vs_avg	Volume / 5-day average volume	Whether today's volume is high or low compared to recent days.
rsi (using TA)	RSI (14-day)	Indicates momentum. >70 = overbought, <30 = oversold.
macd (using TA)	MACD (12-26 EMA)	Shows bullish/bearish momentum.
macd_signal (using TA)	MACD signal line	Used for buy/sell triggers.

⚖️ Scaling the Data

We use StandardScaler to scale features so no single one dominates. Like resizing all pictures before printing so they fit equally.

📉 PCA (Principal Component Analysis)

PCA simplifies complex data into 2D visuals while keeping patterns. It's like showing a 3D object from the best angle in 2D.

📚 What is TA?

TA stands for Technical Analysis—a Python library that calculates financial indicators. These indicators add more trading intuition to our clustering model.

Indicator	What it Means	Why it's Useful
RSI	Measures momentum—how strongly price is moving up/down	>70 overbought, <30 oversold
MACD	Difference between short and long-term moving averages	Detect trend reversals and momentum shifts
MACD Signal	Smoothed MACD used as a signal line	Triggers buy/sell decisions

🔬 Step-by-Step Analysis Process

1. 📥 Data Collection

We used Yahoo Finance to download daily stock data for NMR (Nomura Holdings). This included Open, Close, High, Low, and Volume for each trading day from 2023-01-01 to 2024-12-30.

2. 🧮 Feature Engineering

We calculated eight key features to describe trading behavior. These were already detailed here.

3. ⚖️ Scaling the Data

We standardized all features using StandardScaler so they have the same scale. This prevents large-value features like Volume from overpowering smaller ones like Return.

4. 📉 PCA for Dimensionality Reduction

PCA reduces complex, multi-dimensional data down to 2D for easy visualization. This helps in plotting clusters on a simple chart.

5. 🔗 Clustering the Data

Based on the selected method (KMeans, DBSCAN, or GMM), trading days are grouped into clusters:

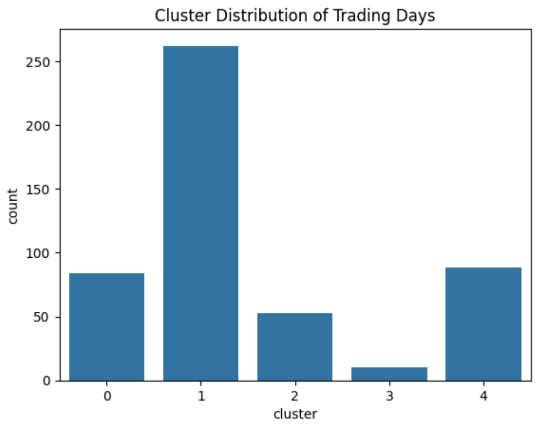
- KMeans: Divides days into a set number of clusters (e.g., 3 or 5).
- DBSCAN: Finds dense groups and labels isolated points as noise.
- GMM: Assigns probabilities to each day for each cluster.

Each cluster gets a human-readable label based on average volume, return, volatility, etc., e.g., 'Volume Trend Up, High Volume, High Volatility, Moderate Gain'.

📊 Visual Outputs

1. 📈 Cluster Distribution

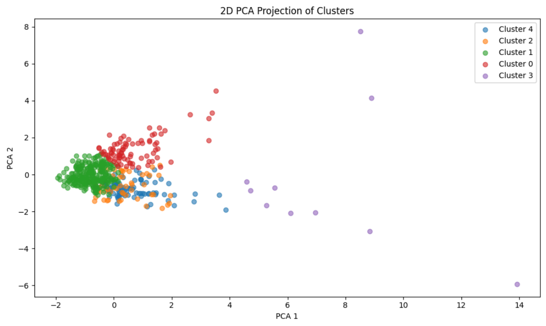
Shows how many trading days belong to each cluster.



This bar chart shows how many trading days fall into each cluster. We observe that **Cluster 1** dominates, suggesting a “normal” or typical market behavior that occurs most often. Smaller clusters like **Cluster 3** may represent outliers or unusual market conditions worth deeper investigation.

2. 🌐 PCA Projection

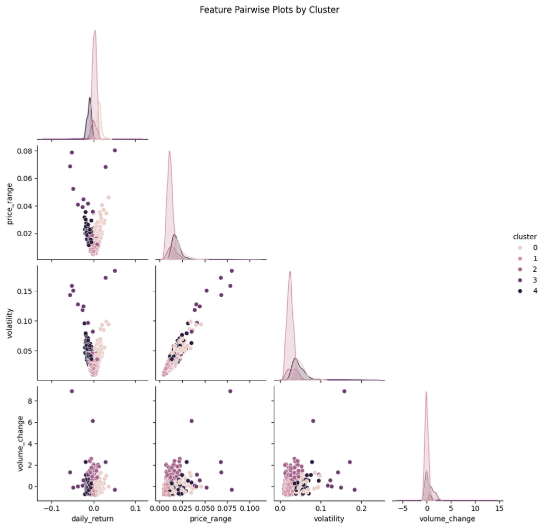
Plots trading days in a 2D space to visualize cluster separation.



The PCA plot reduces our high-dimensional features into two components, allowing us to visually inspect how trading days are grouped. Distinct cluster separation (e.g., Cluster 3 being far apart) implies that those days have significantly different behaviors —potentially tied to earnings releases, economic events, or extreme volume surges.

3. 🔄 Feature Pairplot 🔗

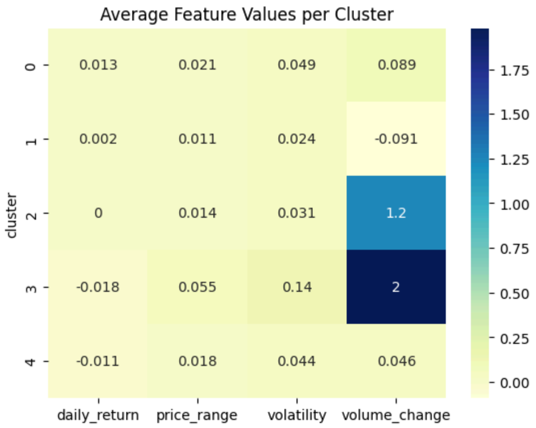
Shows how features relate to each other within clusters.



These pairplots show how features interact within and across clusters. For instance, higher **volatility** often aligns with wider **price ranges** and sometimes larger **volume changes**. Clusters with tighter, less varied distributions (e.g., Cluster 1) reflect consistent, less volatile market days, while scattered patterns (e.g., Cluster 4) reveal erratic behaviors.

4. 🔥 Heatmap - Raw Values 🔗

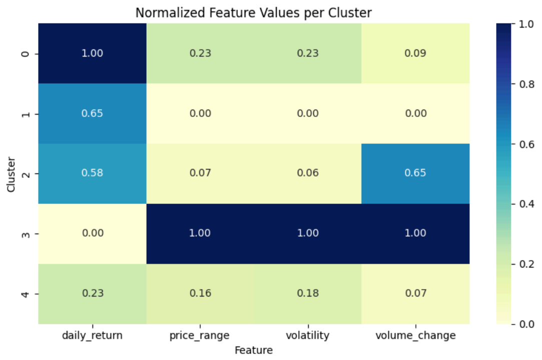
Displays the average feature values for each cluster.



This raw-value heatmap highlights the **absolute average** of each feature across clusters. For example, **Cluster 3** stands out with high values in both **price range** and **volatility**, indicating extreme movement days. In contrast, **Cluster 1** shows low volatility and minimal returns, matching steady, uneventful trading days.

5. 🔥 Heatmap - Normalized 🔗

Normalized to 0–1 scale to highlight differences across features and clusters.



The normalized heatmap brings all features to a common scale (0–1) for easy comparison. Here, **Cluster 0** shows the highest normalized **daily return**, while **Cluster 3** dominates in volatility and volume change. This view makes it easier to identify each cluster's defining characteristics regardless of raw magnitude.

🖥️ Using the Streamlit App

🚀 Features

- Select stock ticker, date range, and clustering method.
- Set parameters for KMeans, DBSCAN, or GMM.
- View and download charts.
- Toggle raw data visibility.
- Export clustered data as CSV.

🛠️ How to Run Locally

1. Install dependencies:

```
pip install -r requirements.txt
```

2. Launch the app:

```
streamlit run app.py
```

The app will open at http://localhost:8501

📁 Project Folder Structure

```
.
├── app/app.py                          # Main Streamlit application
├── requirements.txt                    # Dependencies
├── notebooks/clustening_model.ipynb    # Jupyter notebook to rerun analysis
└── clustening_model_results.pdf        # Jupyter Notebook snapshots
```

💡 Insights & Learnings

- Unsupervised learning can uncover hidden market patterns.
- Volume and return are especially helpful in clustering trading behaviors.
- Visual tools like PCA and heatmaps aid in explaining results to non-technical stakeholders.
- Streamlit enables an interactive, low-code way to share results.