Analyzing financial datasets with AI tools

"Good morning, everyone! Now, I’m excited to share my analysis. We’ll dive into the process of calculating key metrics like daily and annualized volatility, explore some interesting trends in the data, and discuss what these results mean in real-world investing. Let’s get started!"

1. Introduction

**This study aims to analyze financial datasets using AI tools, presenting the findings interactively through dashboards, visualizations, and live demonstrations.** The research will also explore the functionality, applications, and implications of the AI tools used. The analysis will focus on five publicly available stocks: Nvidia, Meta, Adobe, PayPal, and Netflix. Stock data will be fetched from Yahoo Finance using the Yahoo Finance API.AI tools will be utilized specifically to identify patterns, predict price movements, and uncover actionable insights for investment strategies. **A Random Forest machine learning model** will be employed to predict price trends, i.e., whether prices will move up or down. The primary visualizations used to present the data will include cumulative stock price graphs, portfolio cumulative returns graphs, moving average plots, correlation plots, correlation heat maps, and interactive dashboards.

1. Data import and cleaning

The data for the five stocks is imported from Yahoo Finance. The main objective in this part is to prepare the dataset for further analysis and machine learning applications.

**Firstly**, to effectively conduct this study, it is essential to install several Python packages that provide robust tools for data analysis, visualization, and machine learning. Each package serves a specific purpose in the workflow:

\* pandas: Used for **data manipulation and analysis**, especially for handling stock data efficiently in tabular formats.

\* numpy: Provides powerful numerical operations and support for **handling large datasets**, enabling fast computations.

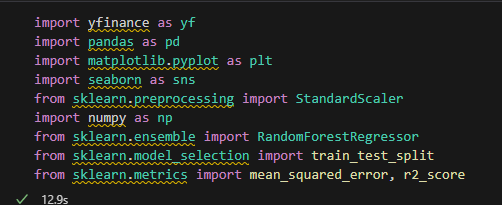
\* matplotlib & seaborn: These are essential for creating **static and interactive visualizations**, helping to explore trends and correlations in the data.

\* yfinance: A dedicated library for **fetching real-time and historical stock market data** from Yahoo Finance.

\* scikit-learn: Provides **machine learning algorithms**, such as the Random Forest model, for building predictive models and uncovering patterns in stock price movements.

\* dash: A web-based framework to create **interactive dashboards** for presenting findings in a user-friendly manner.

Installing these packages ensures a seamless pipeline for gathering, analyzing, visualizing, and presenting financial data using AI tools. Here is the grammar for modules and functions importing.



**Secondly**, the following code processes stock market data for a selected list of stocks and prepares it for further analysis.

A loop iterates over these tickers, **downloading** historical stock data from Yahoo Finance for the period between 2020 and 2024 using the yf.download function. The data is stored in a dictionary, stock\_data, with the ticker symbol as the key.

**Then, data cleaning and formatting**: for each stock’s data, missing values are forward-filled using .ffill(inplace=True), the index is reset, and the Date column is reformatted to a standard date time format using pd.to\_datetime().

**Then, feature scaling**: the Close prices and Volume are standardized (scaled to have mean 0 and standard deviation 1) using StandardScaler from sklearn. This step ensures consistent scaling across different stocks for analytical purposes.

**Then, data combination:** a new DataFrame, combined\_data, is created to consolidate the Date and scaled Close prices for all stocks, for each ticker, the Close column is renamed to <Ticker>\_Close, and the data is merged into combined\_data on the Date column using an outer join and merge.

**Then, final cleaning**: missing values in the combined DataFrame are forward-filled and bakwardfilled to ensure no gaps in the data. **Finally**, **export to CSV**: the cleaned and combined dataset is saved to a CSV file named cleaned\_stock\_data.csv. This file contains scaled close prices for all selected stocks, aligned by date.

This script automates the process of fetching, cleaning, scaling, and consolidating stock market data for easy visualization and analysis. The cleaned dataset can be used for tasks such as trend analysis, predictive modeling, and portfolio evaluation follow on.



1. Data analysis

**Thirdly**,this code performs descriptive and statistical analysis on stock data, focusing on individual stock characteristics and overall portfolio performance. Here's a breakdown of the main tasks:

**Summary statistics** ( for each stock ticker in tickers, the code computes and prints summary statistics for the scaled Close prices (e.g., mean, standard deviation, min, max, etc.) using .describe(). This provides a quick overview of the distribution of closing prices for each stock.);

**Correlation analysis** ( a correlation matrix is calculated using .corr() to analyze the relationships between the stocks' closing prices, highlighting how movements in one stock's price relate to another's ); It is visualized in the following section.

**Portfolio analysis**: the daily percentage changes for all stocks in the combined\_data are computed using .pct\_change(), portfolio returns are calculated as a weighted sum of the individual stocks' returns using .dot(weights), where each stock is equally weighted (1/len(tickers)), the standard deviation of portfolio returns is calculated using .std(). This value represents the overall risk or variability of the portfolio's daily returns.

In conclusion, the outputs, such as correlation matrices, SMAs, and portfolio volatility, are critical for investment decision-making and strategy development.

The other major data analysis performed was the use of technical indicators such as the daily returns indicators, simple moving average indicators (50 day and 200 day, and volatility indicators. Daily return indicators are used to determine short-term price movements by calculating the percentage change in closing prices. **The 50 and 200 simple moving averages are used to identify the market trends and help in making trading and investment decisions.** The volatility is determined using the 30-day rolling standard deviation of daily returns.

The calculated portfolio volatility from our portfolio is 78.65%. This is the combined risk level of the selected stocks. **This indicates that our portfolio has a significantly high risk due to high volatility.**

1. Data visualization

The major visualizations used in this section include: stock prices time series plots, moving averages, and correlation heatmaps. The stocks time series plots is used to show how stock prices have evolved over the specified period. The closing price of each stock is used. The closing price of each stock is shown in the figure below for time period from 2020 to 2024. The correlation matrix of stock closing prices is visualized using a heatmap. Annotated values show the degree of correlation, and the color gradient represents the strength of relationships between stocks. Strong positive correlations indicate that stocks move together while low or negative correlations suggest diversification opportunities within the portfolio. The normalized closing prices are plotted along with their 50-day and 200-day Simple Moving Averages (SMA). This visualization highlights trends and potential crossover points, often used in trading strategies.

1. Application of AI

~~The main objective of this study was to analyze the use of AI tools in analyzing financial datasets. This section covers the use of AI specifically machine learning in analyzing financial datasets.~~ This part will explore how machine learning models can be used in financial data analysis. This part will apply the use of Random Forest Regressor to predict the closing prices of the selected stock prices using engineered features. The process includes data preparation, feature engineering, model training, evaluation, and interpretation. Feature engineering is applied to create indicators such as: Daily Return, SMA\_10, and Volatility\_10. ~~The features selected for the modeling include: SMA\_10, Volatility\_10, Daily\_Return.~~ The target variable is the closing price of NVIDIA stock.

The next step is the model training. A Random Forest Regressor with 100 trees is trained on the training dataset. Data is split into training (80%) and testing (20%) sets. The model is then evaluated using Root Mean Square Error (RMSE) and R-squared (R²) metrics.

RMSE measures prediction error, while R² indicates the proportion of variance in the target variable. The output of the model is the visualized using actual against predicted stock prices graph. The model evaluation gives a RMSE of 0.0674 which indicates a small average prediction error and R² of 0.9925 suggesting the model explains 99.25% of the variance in NVIDIA's closing prices. In terms of feature importance, SMA\_10 dominates with 99.25% importance, showing it is the most influential feature for predicting prices while daily return have minimal impact, contributing 0.11%. This shows that SMA-10 has the highest influence on the stock’s closing prices.

1. Dashboards

This part covers advanced visualization of our stocks using interactive dashboards. The dashboard includes multiple tabs for visualizing stock prices, portfolio cumulative returns, correlations, rolling volatility, and candlestick charts. The interactive dashboard helps in: identifying trends and comparing performance across selected stocks, tracking overall portfolio growth and performance, monitoring risk levels through rolling volatility, and analyzing daily price movements clearly and interactively.

So far, that’s all my presentation. Some financial data visualization and ML model application in finance, in accordance with data analyzing techniques, and a framework with tabs showing different aspects of your finance data.