



*Software Requirements Specification
Version 1.0*

StockPulse

Theme: Stock Price Anomaly Tracker

Project Name: StockPulse

Category: Data Science Dominion

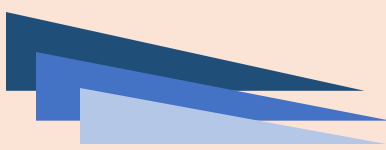


Table of Contents

1.1	Background and Necessity for the Application.....	3
1.2	Proposed Solution.....	4
1.3	Purpose of the Document	7
1.4	Scope of Project.....	7
1.5	Constraints.....	8
1.6	Functional Requirements.....	9
1.7	Non-Functional Requirements.....	9
1.8	Interface Requirements	10
1.8.1	Hardware	11
1.8.2	Software.....	11
1.9	Project Deliverables	12

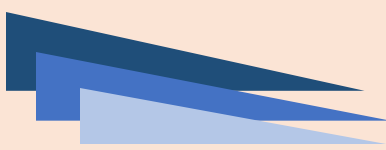


1.1 Background and Necessity for the Application

Financial time series data, such as stock prices and returns, are vital indicators of a company's performance and broader market dynamics. These datasets are often characterized by high volatility, noise, and non-linear trends making them complex to interpret using traditional methods. Amid these regular patterns lie anomalies such as sudden spikes, dips, or irregular fluctuations that deviate from expected market behavior. These anomalies could be early indicators of critical events such as insider trading, economic crises, cyber-attacks, or system malfunctions. For example, a sudden plunge in stock price not aligned with company news could suggest market manipulation or erroneous trading. Detecting such anomalies is crucial for financial analysts, regulatory bodies, and institutional investors to investigate root causes, mitigate risks, and maintain market integrity.

As financial markets become more digitized and data-intensive, the manual inspection of anomalies is neither practical nor scalable. Traditional statistical techniques often assume stationarity and may fail to capture dynamic changes or hidden patterns within large datasets. This necessitates the adoption of advanced anomaly detection and Data Science driven approaches that can learn complex temporal patterns and flag irregularities in near real-time. An effective anomaly detection system in stock price series can aid in early warning for investment risk, fraud detection, algorithmic trading adjustments, and regulatory compliance. Additionally, such systems can empower investors to make smarter and data-driven decisions. In a world where milliseconds can result in massive financial loss or gain, building an intelligent anomaly detection framework is not just beneficial, it is essential for modern financial infrastructure.



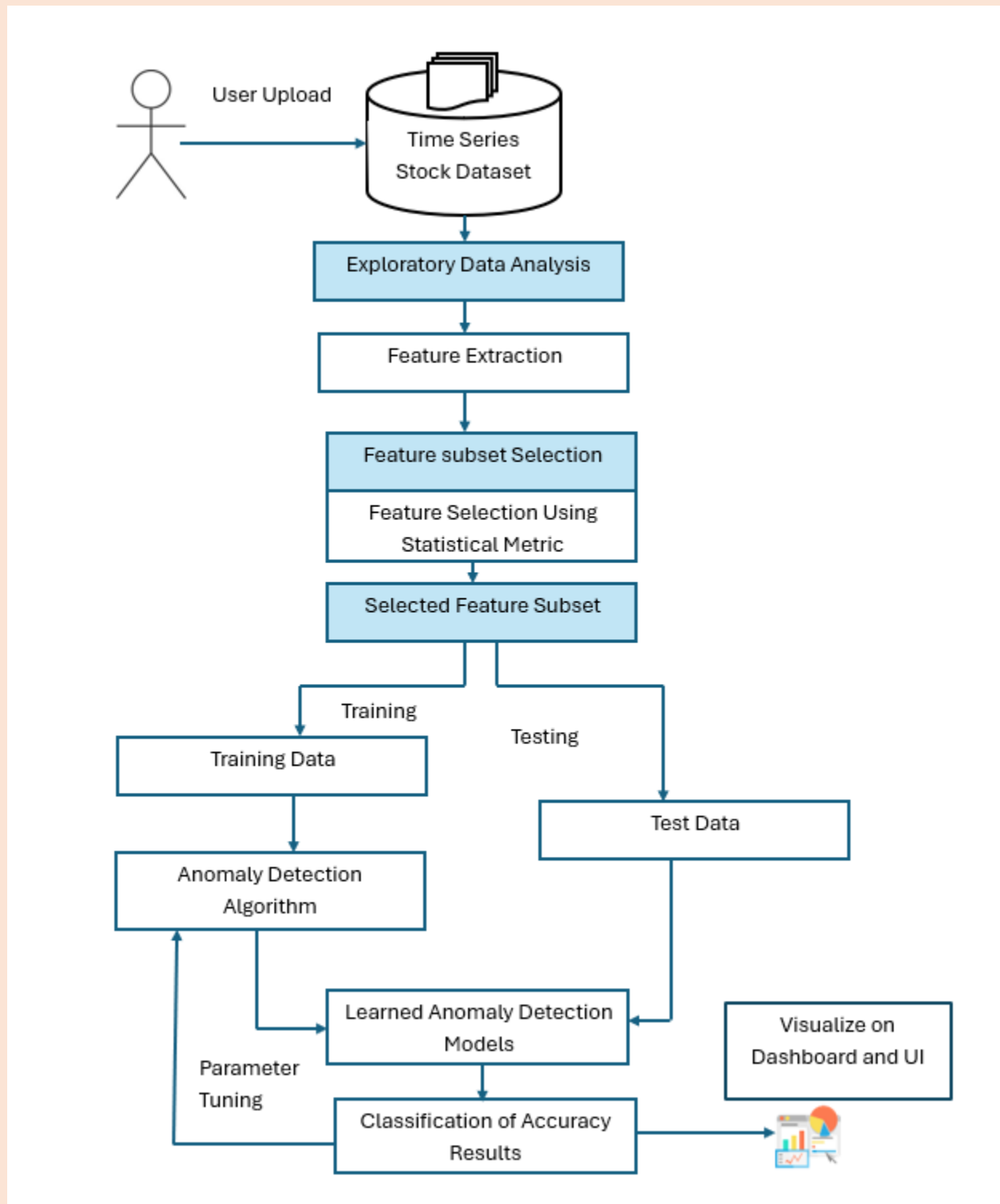


1.2 Proposed Solution

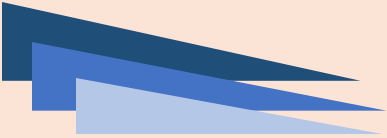
The significance of stock market anomalies resides in their ability to reveal potential opportunities or hazards. Positive news about the firm or its industry, for instance, may cause a rapid jump in the price of stock, indicating a possible investment opportunity. On the other hand, an unanticipated decline in price can indicate underlying problems or shifts in market sentiment, indicating a risk that investors would have to control. In the proposed solution '**StockPulse**', the aim is to detect anomalies in financial stock market data by identifying unusual patterns, spikes, or dips in stock prices that deviate from expected behavior.

- **Data Collection:** The process begins with collecting real-time stock market data. Choose any active company such as Apple, Google, or others and fetch stock data using reliable sources such as Yahoo Finance APIs, Alpha Vantage, or other real-time financial data providers. The dataset typically includes features such as Date, Open, High, Low, Close, Adjusted Close, and Volume which are essential for understanding market trends and detecting anomalies. Working with real-time data will provide more practical insights and make the analysis relevant to current market behavior.

The sample architecture for **StockPulse** application can be as follows:

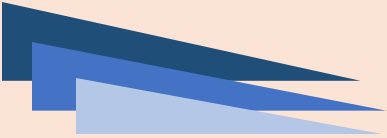


Sample Architecture of the Application



To achieve robust anomaly detection, a data science-driven approach is followed, with unsupervised anomaly detection algorithms for better interpretability and computational efficiency. Steps are outlined based on the proposed architecture as follows:

- **Exploratory Data Analysis (EDA)** - Begin with initial data exploration to understand trends, seasonality, and volatility in stock prices over time. Utilize tools such as line plots, moving averages, and boxplots to visualize variations and detect early signs of anomalies.
- **Data Pre-processing** - The dataset should be cleaned by handling missing values, parsing date formats, sorting records chronologically, and removing inconsistencies. Data should be normalized or standardized to ensure uniform scaling, which is especially helpful for distance-based anomaly detection methods.
- **Feature Extraction** - Derive new features such as returns, moving averages, and momentum indicators to enrich the dataset and provide deeper insights useful for identifying anomalies.
- **Feature Subset Selection** - Use statistical techniques such as variance thresholding, correlation analysis, or feature importance (for example, from tree-based models), select the most relevant and non-redundant features to reduce dimensionality and noise.
- **Data Splitting** - The processed dataset should be split into training and testing sets. The training set can be used to learn normal behavior patterns, while the testing set may contain potential anomalies for model evaluation.
- **Anomaly Detection Algorithm** - Anomaly detection techniques such as Isolation Forest, or Support Vector Machine (SVM) should be employed. These algorithms are well-suited for identifying outliers in time series without requiring labeled anomalies.
- **Parameter Tuning** - Parameters of the chosen should be fine-tuned using internal cross-validation and evaluation on the validation set to optimize the detection capability.
- **Model Evaluation** - The final model should be tested on unseen data to evaluate its performance in correctly classifying normal and anomalous patterns. Evaluation metrics such as precision, recall, F1-score, and visual inspection of flagged anomalies should be used to assess accuracy.



This proposed solution offers a complete, scalable, and interpretable workflow for financial anomaly detection, suitable for real-time fraud monitoring, risk assessment, or early warning systems in the stock market domain.



1.3 Purpose of the Document

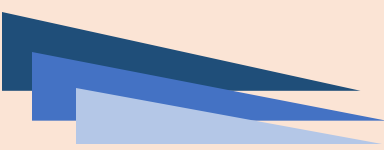
The purpose of this document is to present a trained and tested interactive model titled '**StockPulse**'.

This document explains the purpose and features of Data Science and the constraints under which it must operate. This document is intended for both stakeholders and developers of the system.

1.4 Scope of Project

The scope of this project is to develop a data science-driven solution for detecting anomalies in financial time series data by analyzing stock price movements over time. It includes key stages such as data exploration, preprocessing, feature engineering, and the application of anomaly detection algorithms to identify unusual patterns, spikes, or dips that deviate from expected trends.

The project also involves designing a simple user interface to display the results and visualizing the findings through interactive dashboards and charts, enabling users to easily interpret detected anomalies.



1.5 Constraints

Algorithmic assumptions like data stability may not be held in financial time series. The project also focuses on offline analysis, with a basic dashboard for visualization, and does not include real-time detection or interactive trading features.






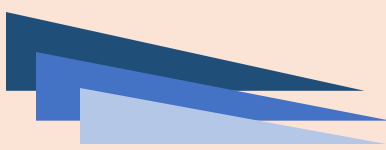


1.6 Functional Requirements

Functional requirements for a financial time series anomaly detection system are essential to ensure accurate identification of unusual stock price patterns and trends. These requirements define the capabilities and behaviors the system must exhibit to detect anomalies effectively and present findings in an intuitive, visual manner.

Key functional requirements of the application are as follows:

- 
 - i. Data Ingestion and Management:** The application shall ingest historical financial stock market data, including price and volume information, and organize it chronologically. It should allow importing data from structured CSV or other standard formats into the system for further analysis.
- ii. Data Pre-processing:** The application shall preprocess the data by cleaning missing values, parsing and formatting date fields, handling inconsistencies, and performing normalization or standardization to prepare the dataset for further analysis.
- iii. Feature Engineering:** The system shall compute derived metrics such as returns, moving averages, and volatility indicators to enhance the dataset. These features will provide rich input for anomaly detection algorithms.
- 
 - iv. Feature Selection:** The application shall include functionality to identify and retain relevant features using statistical techniques such as correlation analysis or variance thresholds to reduce noise and improve model efficiency.
- 
 - v. Data Partitioning:** The dataset shall be split into training and testing sets. The training set will represent typical behavior, while the testing set will be used to validate anomaly detection results.
- vi. Anomaly Detection Module:** The application shall implement data science-driven anomaly detection algorithms such as Isolation Forest or SVM to identify unusual stock behavior. It should allow configuration of algorithm parameters for optimal detection performance.
- vii. Model Evaluation:** The system shall evaluate the detection results using metrics such as precision, recall, and F1-score. It should also enable visual confirmation of detected anomalies to validate accuracy.



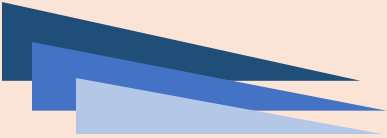
- viii. **User Interface (UI) and Interaction:** The application should provide a simple UI to allow users to upload datasets, initiate analysis, and view results. Users should be able to navigate through outputs and interact with visual elements easily.



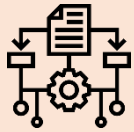
- ix. **Dashboard Visualization:** The system shall generate interactive dashboards using Tableau features or other visualization techniques. These dashboards will display key trends, time series plots, and highlight detected anomalies, providing users with clear, actionable insights.
- x. **Export and Reporting:** The application shall offer functionality to export anomaly reports and visual summaries for documentation or further decision-making support.



1.7 Non-Functional Requirements



There are several non-functional requirements that should be fulfilled by the application. The application should be:



1. **Usable:** The application should be designed with a clear and intuitive interface ensuring ease of use for all users.

2. **Scalable:** The system shall process large volumes of financial time series data efficiently, ensuring timely detection of anomalies. It should be scalable to handle additional data without significant degradation in performance.



3. **Accuracy:** The application shall provide consistent results with a high degree of accuracy in anomaly detection. It must minimize false positives and false negatives while maintaining dependable output across different data segments.



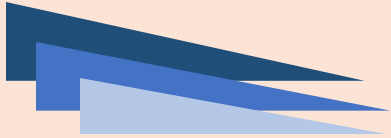
4. **Compatible:** The application should be compatible with common operating systems and support standard data formats. It should be integrated smoothly with external visualization tools such as Tableau for dashboard rendering.



These are the bare minimum expectations from the project. It is a must to implement the FUNCTIONAL and NON-FUNCTIONAL requirements given in this SRS.

Once they are complete, you can use your own creativity and imagination to add more features if required.

1.8 Interface Requirements



1.8.1 Hardware

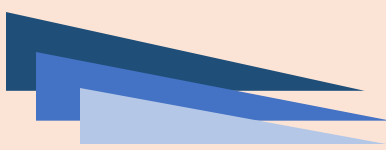
Intel Core i5/i7 Processor or higher
8 GB RAM or higher
Color SVGA monitor
500 GB Hard Disk space
Mouse and Keyboard

1.8.2 Software

Technologies to be used:

1. **Data Store:** CSV
2. **Backend:** Apache Spark or Apache Hive, Flask/Django
3. **Database:** MongoDB/MySQL
4. **Programming/IDE:** R programming/Python 3.11.4 or higher, Jupyter Notebook, Anaconda 23.1.0 or higher, or Google Colab
5. **Libraries:** OpenCV, TensorFlow, scikit-learn, Pandas, NumPy, PyTorch, Matplotlib, and Seaborn
6. **Visualization:** Tableau Desktop features/other visualization techniques





1.9 Project Deliverables

You will design and build the project and submit it along with a complete project report that includes:

- Problem Definition
- Design Specifications
- Diagrams such as User Flow Diagram/User Journey Map
- Test Data Used in the Project
- Project Installation Instructions
- Link of GitHub for accessing the uploaded project code (Link should have public access)
- Detailed Steps to Execute the Project
- Link of Published Blog

The source code, including .ipynb files for Jupyter Notebook and Google Colab, should be shared via GitHub. Appropriate access permissions should be granted to users to allow testing for Jupyter Notebook and Google Colab. The consolidated project must be submitted on GitHub with a ReadMe.doc file listing assumptions (if any) made at your end.

Provide the GitHub URL where the project has been uploaded for sharing. The repository on GitHub should have public access. Documentation is a very important part of the project; hence, all crucial aspects of the project must be documented properly. Ensure that documentation is complete and comprehensive.

You should publish a blog of minimum 2000 words on any free blogging Website such as Blogger, Tumblr, Ghost, or any other blogging Website. The link of the published blog should be submitted along with the project documentation.

Do NOT copy content or code from GPTs or other AI tools, although you are permitted to use images generated by AI tools for any visual representation purposes. It is mandatory to mention such tools used in case you add any AI generated images.

Submit a video (.mp4 file) demonstrating the working of the application, including all the functionalities of the project. This is MANDATORY.

~~~ End of Document ~~~