Large-Scale Financial Data Analysis and Trend Detection

Overview of the Project

The goal of this project is to create a system for analyzing vast amounts of financial data and identifying trends. The system is intended to effectively process and analyze big datasets relating to stock prices or cryptocurrency transactions in order to detect patterns, trends, and anomalies. The system uses divide-and-conquer algorithms and other computer approaches to generate analytical evaluations that can help in financial investment and market prediction decision-making processes.

Problem being addressed

The fundamental challenge addressed by the study is the requirement to efficiently evaluate large financial datasets in order to identify relevant insights. Given the volatility and complexity of financial markets, such as cryptocurrency, it is essential to identify periods of maximum profit or loss, as well as unusual patterns that may indicate market anomalies (for example, potential fraud or market manipulation), and generate visual reports summarizing these findings.

Handling enormous volumes of financial data necessitates the use of efficient algorithms to provide accurate and rapid analysis. This project solves this by incorporating important algorithms designed specifically for time-series data processing and anomaly identification.

Goals of the Analysis

The primary aims of this initiative are as follows:

Sort and Organize Financial Data: Sort and organize time-series data (for example, stock prices or cryptocurrency prices) before analyzing it.

Detect Trends: Use Kadane's method to identify important moments of highest gain or loss. This will aid in identifying the best and worst-performing periods within a particular dataset.

Detect anomalies: Use a nearest pair of points algorithm to detect unexpected price fluctuations that may suggest trade anomalies, abnormal market circumstances, or suspected fraud.

Generate reports: Create concise visual reports that highlight essential facts, such as price trends, moving averages, peak profit times, and discovered abnormalities.

Choice of Financial Datasets

This experiment used cryptocurrency data, with a focus on Solana (SOL). The Solana price information covers daily price changes, such as the opening and closing prices, high and low prices, and volume.

The tremendous volatility of the cryptocurrency market made Solana a great candidate for trend and anomaly identification. Furthermore, cryptocurrency markets function continually, unlike stock markets, providing additional data points for study.

Algorithms Used in Processing and Analysis

Several essential algorithms were used in this project:

Merge Sort:

The Merge Sort algorithm efficiently sorts the Solana price data by date, preparing it for subsequent analysis. This divide-and-conquer algorithm ensures that the dataset is ordered, which is critical for trend detection and anomaly identification.

Kadane's Algorithm (1D, 2D):

1D Kadane's Algorithm: Kadane's algorithm is applied to detect the sub-period where Solana exhibited maximum gains. The algorithm computes the maximum subarray from the daily price changes, identifying the period with the highest growth. This helps highlight the best investment period in Solana's history.

- Maximum gain detected: From **[2020-11-25]** to **[2022-02-08].**
- Starting price: **$$9.76**, Ending price: **$258.48**, Percentage gain:**2548.33%**

2D Kadane's Algorithm: Enables the system to manage various assets by comparing trends across regions or stocks.

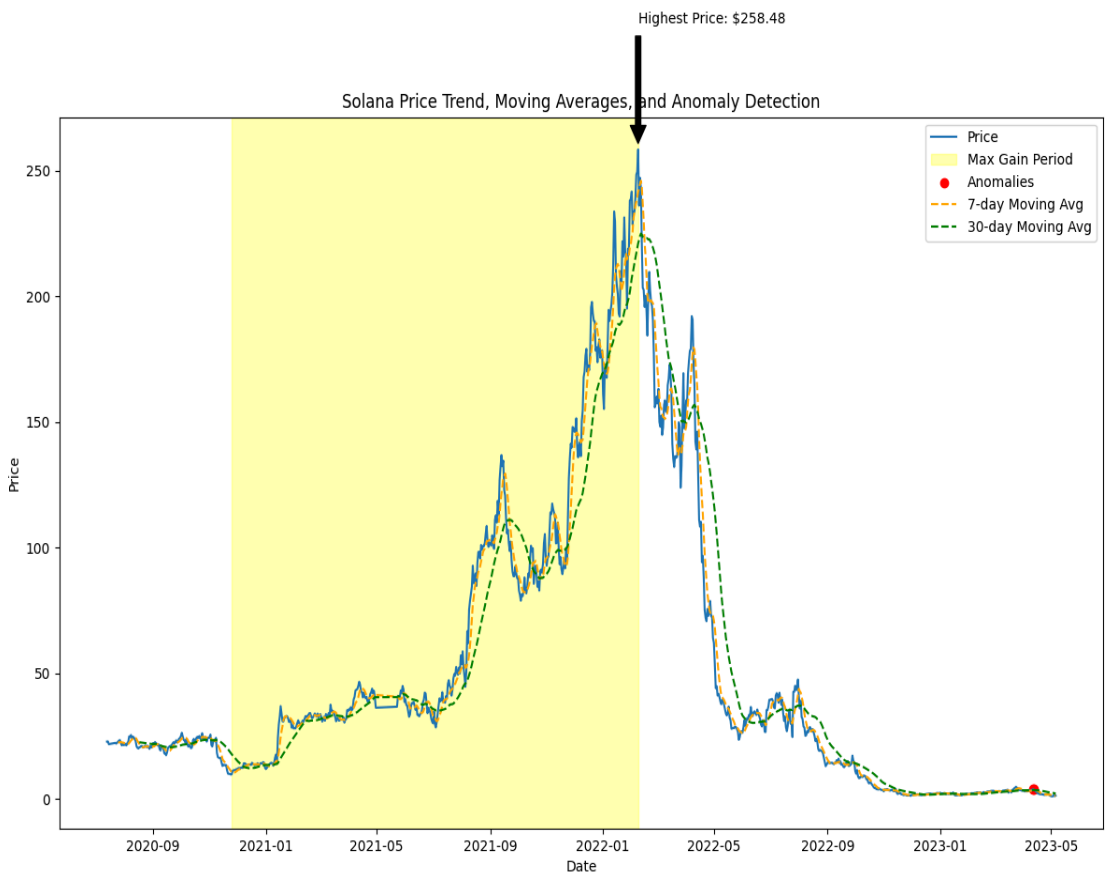
Closest Pair of Points Algorithm

The Closest Pair of Points algorithm is used to identify anomalies by detecting unusually close price points over the time period. The Euclidean distance between price points is calculated to detect deviations from normal patterns, allowing the detection of anomalies.

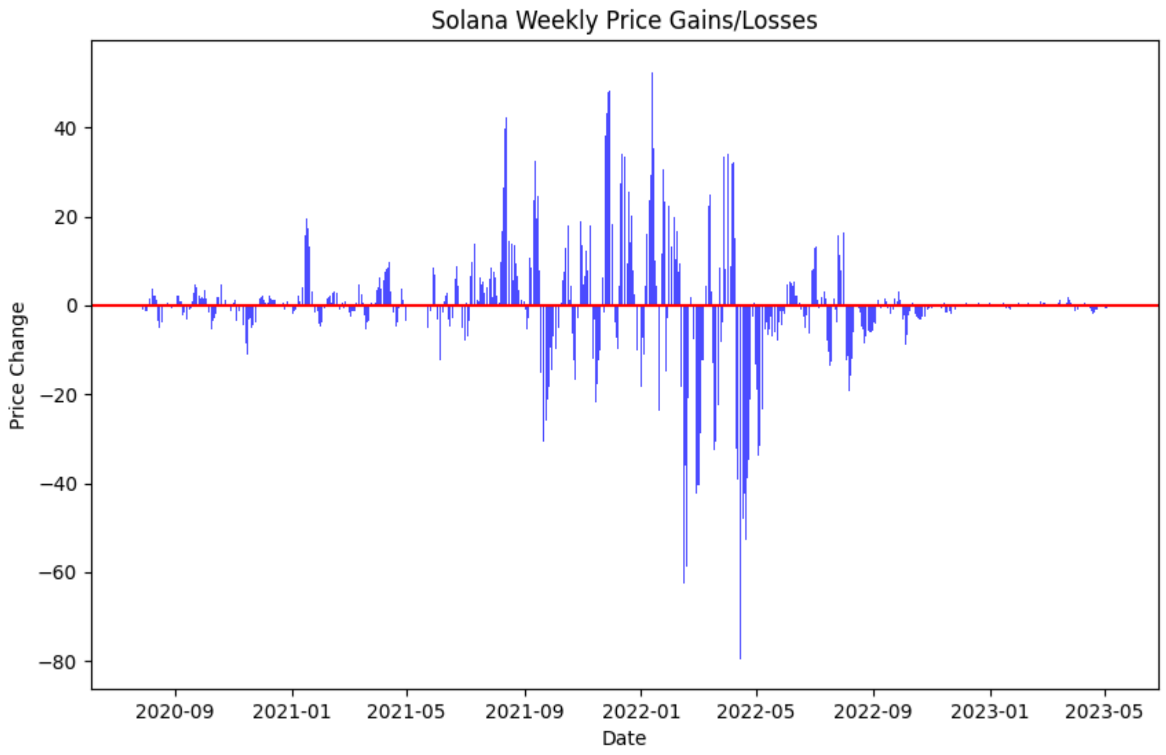
- Closest anomaly points identified between **[2023-04-11**] and [**2023-04-12**] with distance of 1

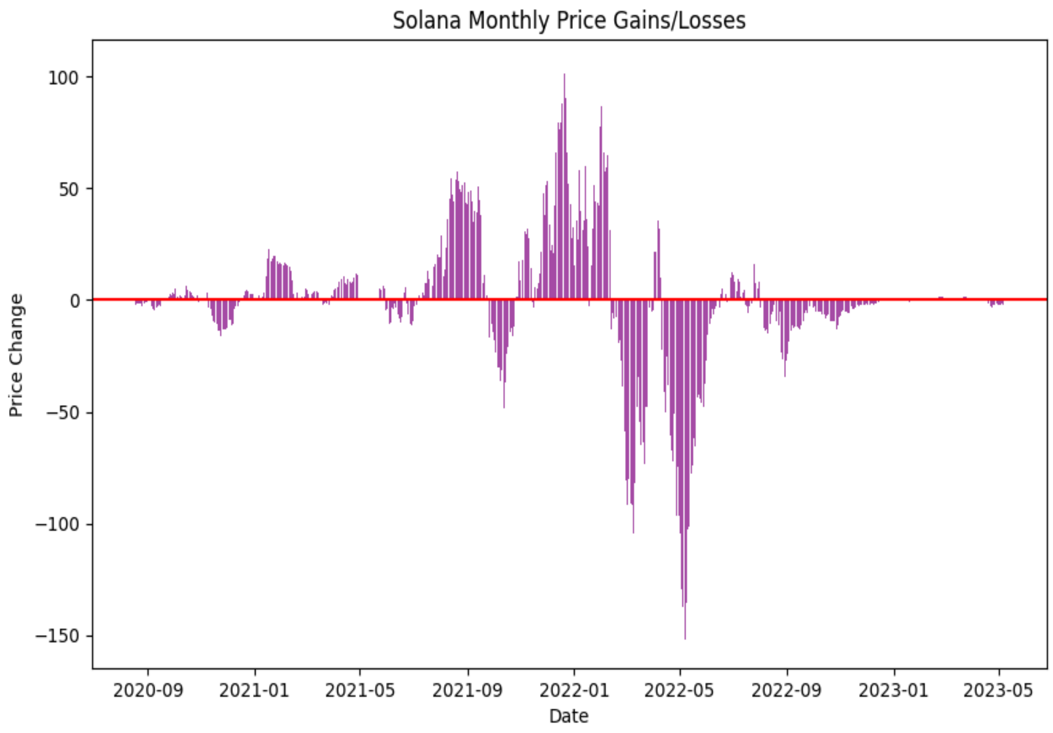
Solana Price Trend

The project produces several visual outputs to summarize the findings:
1. **Solana Price Trend and Anomaly Detection:**
   - Max Gain Period: The yellow shaded region indicates the time range with the highest price increase, based on Kadane's algorithm.
   - Anomalies: Red dots represent price anomalies detected by the Closest Pair of Points algorithm.
   - Moving Averages: Both 7-day and 30-day moving averages are plotted to smooth short-term and medium-term price fluctuations.

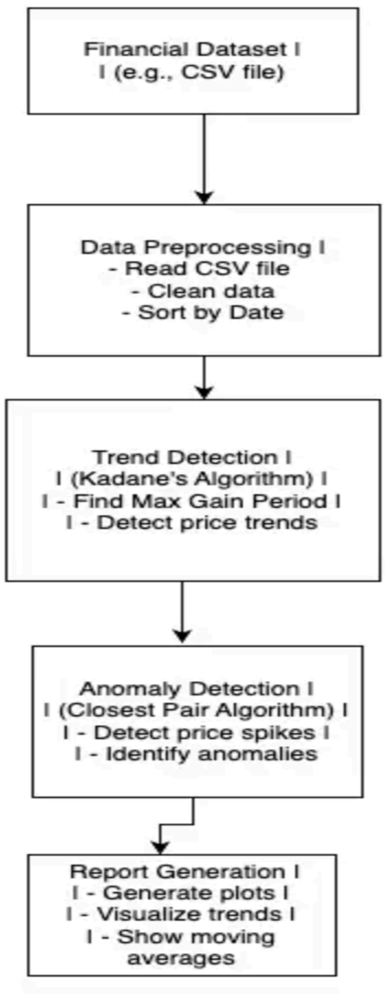


2. **Weekly and Monthly Price Gains/Losses:**
   - **Weekly Price Gains/Losses:** This bar plot highlights weekly price changes, showing periods of price gains and losses over the dataset.
   - **Monthly Price Gains/Losses:** A similar bar plot displays monthly price changes, helping to identify longer-term trends.


Solana Weekly Price Gains/Losses
Price Change
40
20
0
−20
−40
−60
−80
2020-09
2021-01
2021-05
2021-09
2022-01
2022-05
2022-09
2023-01
2023-05
Date


Solana Monthly Price Gains/Losses
Price Change
100
50
0
−50
−100
−150
2020-09
2021-01
2021-05
2021-09
2022-01
2022-05
2022-09
2023-01
2023-05
Date

Block Diagram / Flowchart



Data sorting (Data preprocessing):

The DataProcessor reads, cleans, and sorts the dataset by date with Merge Sort.

Output: Cleaned and sorted data is sent to the Trend Analyzer and Anomaly Detector.

Trend Analysis (Kadane's Algorithm):

The Trend Analyzer uses Kadane's Algorithm to identify the dataset's period of highest gain.

Output: The maximum gain period is supplied to the Report Generator.

Anomaly detection (using the closest pair algorithm):

The Anomaly Detector uses the Closest Pair of Points Algorithm to detect unexpected spikes or dips.

Output: The Report Generator receives anomalies.

Report Generation:

The Report Generator visualizes trends and anomalies by generating reports using line graphs and moving averages.

Output: Visual reports are provided to summarize the analysis.

Each component operates independently, but collaborates with others to ensure that data flows smoothly from preprocessing to report production.

Summary of Developed Classes

DataProcessor:

The purpose is to manage data loading, cleaning, and sorting.

Key methods:

Preprocess_data(file_path): Reads and cleans CSV data.

merge_sort(arr): Performs merge sort to arrange data by date.

TrendAnalyzer:

The goal is to spot patterns and pinpoint peak gains or losses.

Key methods:

detect_max_gain(): Uses Kadane's Algorithm to determine the period of maximum gain.

kadane_1d(arr): Use Kadane's 1D technique to discover trends.

AnomalyDetector:

Use the Closest Pair of Points technique to find anomalies or odd price increases.

Key methods:

detect_anomalies(): Identifies abnormalities in pricing data.

closest_pair(points): Uses the Closest Pair of Points method.

ReportGenerator:

Purpose: Create visual summaries of the examined data, including trends and anomalies.

Key methods:

generate_trend_report(max_gain_period): Generates a report outlining the discovered trends.

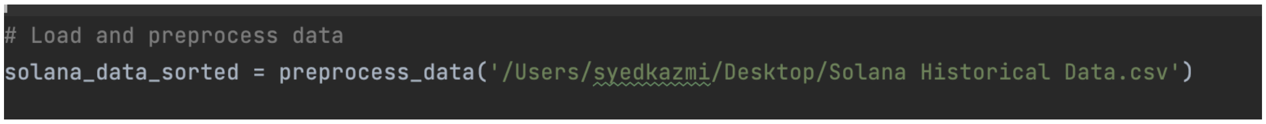
generate_anomaly_report(abnormalities): Displays the discovered anomalies.

Each class is modular, focused on a distinct purpose to provide system clarity and scalability.

Instructions for Using the System

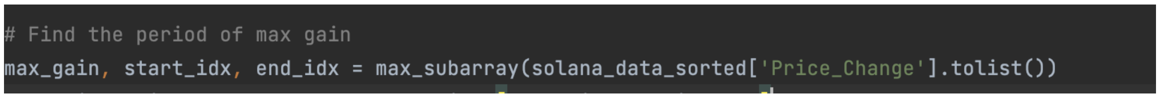
Usage Instructions

Load and Preprocess Data: Use preprocess_data() to clean and sort the dataset by date.



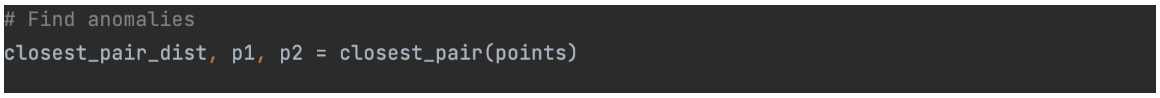
```
# Load and preprocess data
solana_data_sorted = preprocess_data('/Users/syedkazmi/Desktop/Solana Historical Data.csv')
```

Trend Analysis (Kadane's Algorithm): Apply max_subarray() to identify the period of maximum gain in the price changes.



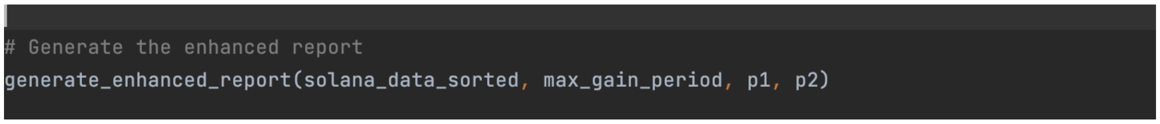
```
# Find the period of max gain
max_gain, start_idx, end_idx = max_subarray(solana_data_sorted['Price_Change'].tolist())
```

Anomaly Detection: You can detect anomalies (price spikes or unusual fluctuations) using the closest pair of points algorithm. Call closest_pair() to identify unusual price movements.



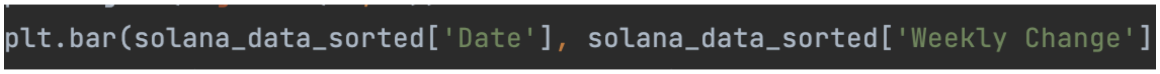
```
# Find anomalies
closest_pair_dist, p1, p2 = closest_pair(points)
```

Generate Enhanced Reports: Generate visual reports showing the maximum gain period, moving averages, and any detected anomalies using the generate_enhanced_report() function.

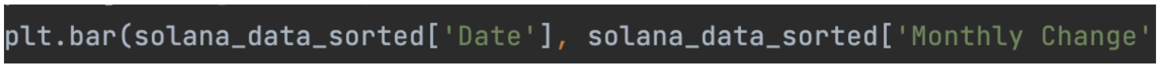


```
# Generate the enhanced report
generate_enhanced_report(solana_data_sorted, max_gain_period, p1, p2)
```

Bar Plots: Generate weekly and monthly price gain/loss plots to observe price trends over different time periods:



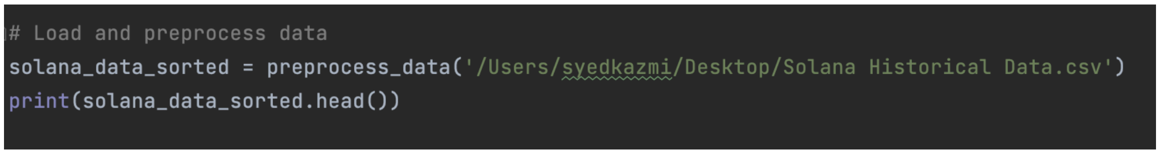
```
plt.bar(solana_data_sorted['Date'], solana_data_sorted['Weekly Change']
```



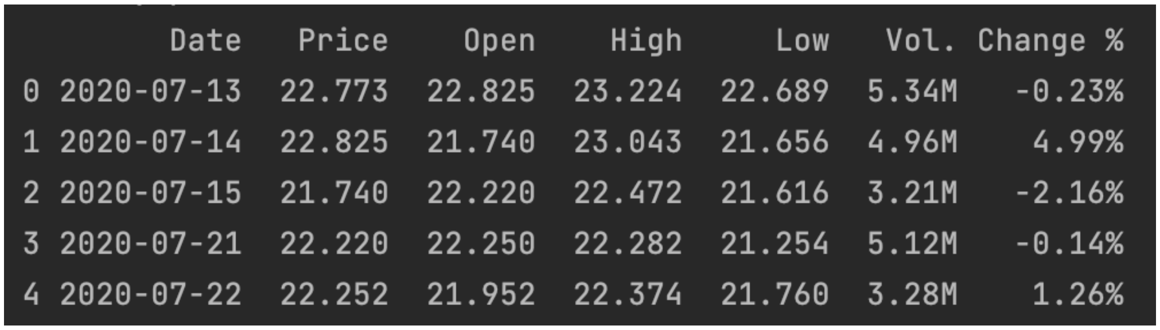
```
plt.bar(solana_data_sorted['Date'], solana_data_sorted['Monthly Change'
```

Verification of Code Functionality

Load and preprocess data



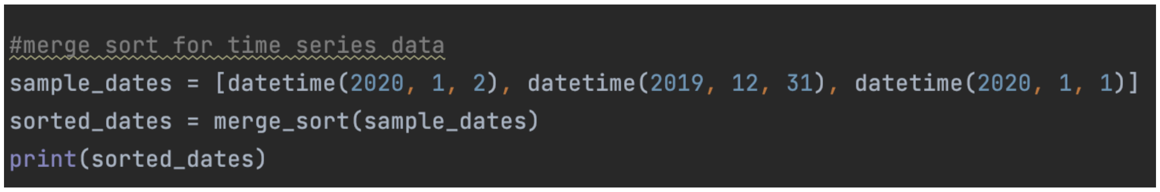
```
# Load and preprocess data
solana_data_sorted = preprocess_data('/Users/syedkazmi/Desktop/Solana Historical Data.csv')
print(solana_data_sorted.head())
```



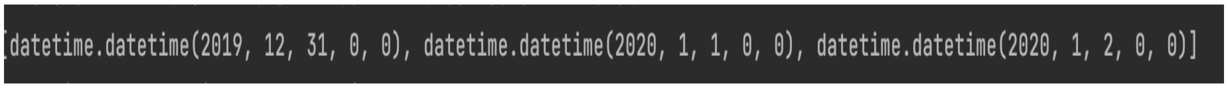
```
         Date   Price    Open    High     Low   Vol. Change %
0 2020-07-13  22.773  22.825  23.224  22.689  5.34M   -0.23%
1 2020-07-14  22.825  21.740  23.043  21.656  4.96M    4.99%
2 2020-07-15  21.740  22.220  22.472  21.616  3.21M   -2.16%
3 2020-07-21  22.220  22.250  22.282  21.254  5.12M   -0.14%
4 2020-07-22  22.252  21.952  22.374  21.760  3.28M    1.26%
```

Merge Sort for Time-Series Data

- Goal: Sort financial data (like Solana price) by date to ensure chronological order for trend analysis.



```
#merge sort for time series data
sample_dates = [datetime(2020, 1, 2), datetime(2019, 12, 31), datetime(2020, 1, 1)]
sorted_dates = merge_sort(sample_dates)
print(sorted_dates)
```

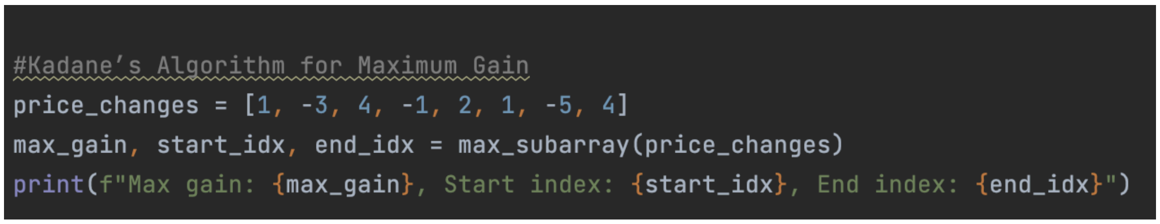


```
[datetime.datetime(2019, 12, 31, 0, 0), datetime.datetime(2020, 1, 1, 0, 0), datetime.datetime(2020, 1, 2, 0, 0)]
```

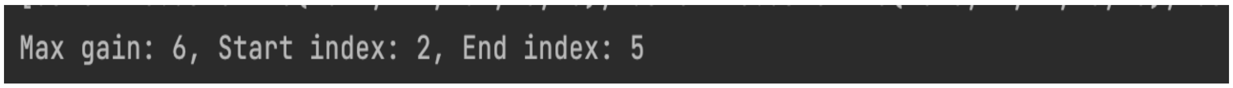
Kadane's Algorithm for Maximum Gain Detection

Goal: Detect the sub-period with the highest price gains using daily price changes.

Toy Example:



```
#Kadane's Algorithm for Maximum Gain
price_changes = [1, -3, 4, -1, 2, 1, -5, 4]
max_gain, start_idx, end_idx = max_subarray(price_changes)
print(f"Max gain: {max_gain}, Start index: {start_idx}, End index: {end_idx}")
```



```
Max gain: 6, Start index: 2, End index: 5
```

Closest Pair of Points for Anomaly Detection

- Goal: Detect anomalies by identifying unusually close price points over time.

- Toy Example:

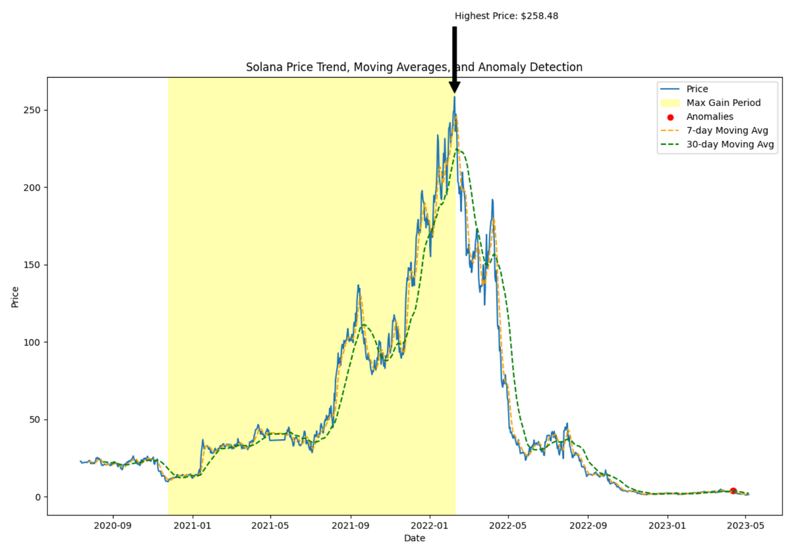
```
#Closest Pair of Points for Anomaly Detection
points = [(1, 1), (2, 2), (3, 3), (4, 5)]
dist, p1, p2 = closest_pair(points)
print(f"Closest points: {p1} and {p2}, Distance: {dist}")
```

```
Closest points: (1, 1) and (2, 2), Distance: 1.4142135623730951
```

Generating Enhanced Reports

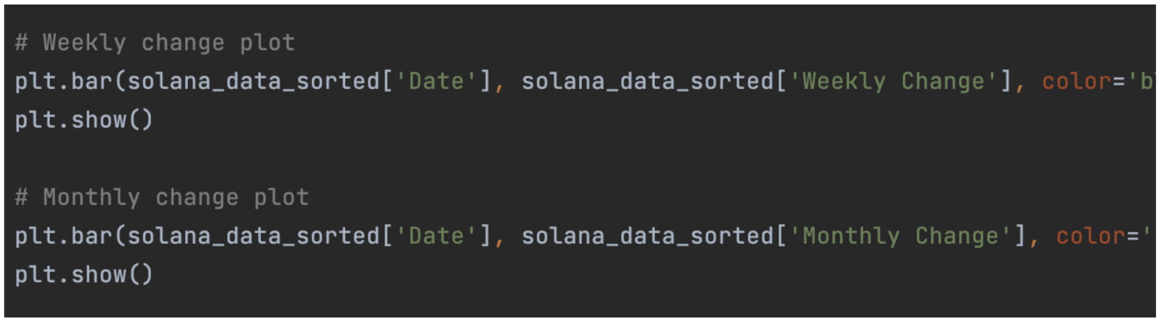
- Goal: Generate a comprehensive report with price trends, maximum gain periods, and anomaly detection.
- Sample Scenario:

```
# Generate the enhanced report
generate_enhanced_report(solana_data_sorted, max_gain_period, p1, p2)
```



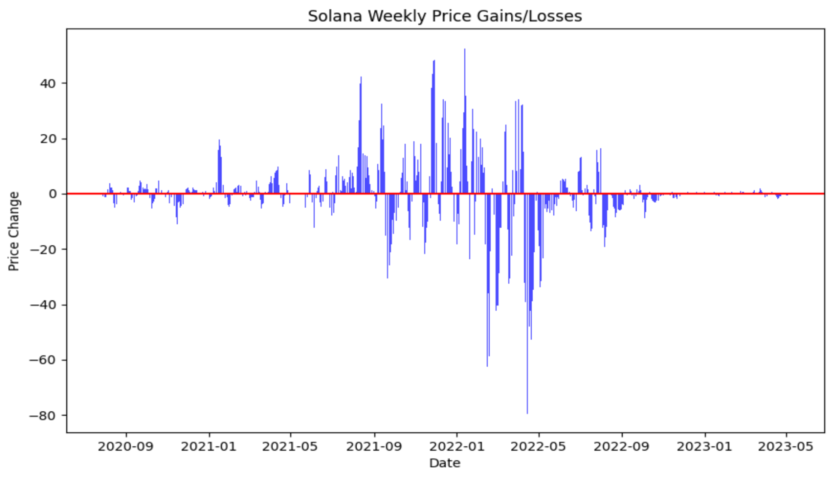
Weekly and Monthly Gains/Losses

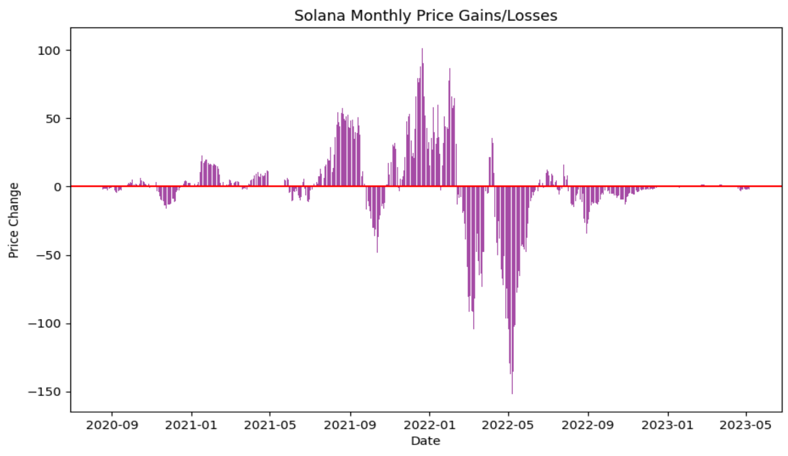
- Goal: Visualize the weekly and monthly changes in prices.



```
# Weekly change plot
plt.bar(solana_data_sorted['Date'], solana_data_sorted['Weekly Change'], color='b
plt.show()

# Monthly change plot
plt.bar(solana_data_sorted['Date'], solana_data_sorted['Monthly Change'], color='
plt.show()
```





Each system component was evaluated and confirmed using toy examples. Screenshots and execution outputs show that the system performs as intended for basic actions including sorting data, recognizing price patterns, finding abnormalities, and creating visual reports. These examples demonstrate that the algorithms employed in the project are accurate and efficient.

Discussion of Results

Insights: From **November 25, 2020** to **February 8, 2022**, Solana prices increased by 2548.33%. Based on historical statistics, this period represents the best time to invest.

Anomaly Detection: Using the Closest Pair of Points technique, anomalies were identified during **April 11** and **12**, 2023, when the price stayed abnormally steady. This identifies probable market abnormalities or technical issues that influence price behavior.

Trend Analysis: The system's use of **7-day** and 30-day moving averages aided in the identification of long-term trends, offering a more distinct picture of market direction by filtering out short-term variations.

Challenges

Data Preprocessing: Managing missing or inconsistent data, such as date and price formatting, was a significant challenge in the early stages of data purification.

2-D Kadane's Algorithm: Extending Kadane's technique to 2D arrays, such as reviewing many stocks simultaneously, added complexity and burden.

Anomaly Detection Complexity: To efficiently detect price anomalies in larger datasets while avoiding false positives, the algorithm has to be modified.

Limitations

Anomaly detection Accuracy: The existing approach for detecting anomalies, which uses Euclidean distance, may ignore complicated market patterns.

Fixed moving averages: The fixed 7-day and 30-day periods may not always be appropriate for varying market circumstances or asset types.

No Real-Time Analysis: The system can only process historical data and cannot handle real-time changes, which are required for active trading situations.

Areas for Improvement

Advanced Anomaly Detection: Using complex anomaly detection technologies, such as machine learning-based models, may enhance accuracy and detect smaller abnormalities.

Optimization of large datasets: Parallel processing or GPU-based methods may enhance speed, particularly when dealing with large datasets.

Real-Time Data Integration: Using real-time financial data APIs, users may conduct live analysis and make quickly trading decisions.

Conclusion

This research effectively applied major financial data analysis techniques such as Merge Sort, Kadane's algorithm, and Closest Pair of Points. These approaches shed light on patterns, periods of maximal gain, and anomalies in the Solana pricing dataset.

While the system is capable of handling historical data analysis and trend recognition, there is need for development in areas such as real-time data processing, dynamic trend analysis, and more advanced anomaly detection. Addressing these restrictions would improve the system's accuracy and usefulness for real-world financial decision-making.

In summary, the project provides a good framework for evaluating huge financial information, with room for further improvement and growth to handle more complicated situations and real-time data.