The report would explore the relationship between trader performance on Hyperliquid and the daily Bitcoin market sentiment (Fear/Greed Index). The analysis would focus on the following key areas:

- **Correlation Analysis**: A statistical breakdown of how changes in the Fear/Greed Index correlate with trader profitability (closed PnL). This section would determine if periods of "Greed" lead to higher profits and if "Fear" leads to losses.
- **Behavioral Patterns**: An examination of how trader behavior, such as trade size, side (buy/sell), and leverage, shifts during different sentiment phases. For example, do traders use higher leverage when the market is "Greedy"?
- **Performance by Sentiment Group**: A segmented analysis of trader performance metrics like average PnL, win rate, and risk-adjusted returns across the different sentiment classifications (e.g., Extreme Fear, Fear, Neutral, Greed, Extreme Greed).
- **Key Insights**: Actionable takeaways to help traders develop smarter strategies, such as when to be more cautious or aggressive based on market sentiment.

Important Lines of Python Code

The following code snippets illustrate the core steps of the analysis that would be performed using popular data science libraries like `pandas` and `matplotlib`.

1. Loading and Merging the Datasets
   The first step is to load the two datasets and combine them based on the Date column, which serves as the common link between trader activity and market sentiment.

Python

```python
import pandas as pd

# Load the datasets
trader_df = pd.read_csv('hyperliquid_trader_data.csv')
sentiment_df = pd.read_csv('bitcoin_sentiment.csv')

# Convert the 'time' column to a standardized date format for merging
trader_df['Date'] = pd.to_datetime(trader_df['time']).dt.date

# Merge the two datasets on the date
merged_df = pd.merge(trader_df, sentiment_df, on='Date')

# Display the first few rows of the merged data
print(merged_df.head())

```

16. Calculating Performance Metrics by Sentiment
    This code would group the data by the Classification (Fear/Greed) and calculate key performance metrics such as the mean closedPnL to see how profitability changes with sentiment.
    Python

```
    # Group the data by sentiment and calculate the average closed PnL

17.  sentiment_performance = merged_df.groupby('Classification')
     ['closedPnL'].mean().reset_index()
18.
19.  print(sentiment_performance)
```

20. Analyzing Behavioral Trends
    This snippet would analyze how trading behavior changes. For example, it would look at the proportion of "buy" vs. "sell" trades for each sentiment classification.
    Python

```
21. # Create a new column to identify buy (1) or sell (0)

22.  merged_df['is_buy'] = (merged_df['side'] == 'buy').astype(int)
23.
24.  # Calculate the average buy ratio for each sentiment classification
25.  buy_ratio_by_sentiment = merged_df.groupby('Classification')
     ['is_buy'].mean().reset_index()
26.
27.  print(buy_ratio_by_sentiment)
28.
```