


```
from google.colab import files
uploaded = files.upload()
```

 Choose files 2 files


- **fear_greed_index.csv**(text/csv) - 90801 bytes, last modified: 31/07/2025 - 100% done
- **historical_data.csv**(text/csv) - 47516935 bytes, last modified: 31/07/2025 - 100% done

Saving fear_greed_index.csv to fear_greed_index.csv
Saving historical_data.csv to historical_data.csv

```
import pandas as pd
```

```
trades_df = pd.read_csv("historical_data.csv")
sentiment_df = pd.read_csv("fear_greed_index.csv")
```

```
print(trades_df.columns)
print(sentiment_df.columns)
```

 Index(['Account', 'Coin', 'Execution Price', 'Size Tokens', 'Size USD', 'Side', 'Timestamp IST', 'Start Position', 'Direction', 'Closed PnL', 'Transaction Hash', 'Order ID', 'Crossed', 'Fee', 'Trade ID', 'Timestamp'], dtype='object')
Index(['timestamp', 'value', 'classification', 'date'], dtype='object')

```
trades_df.head()
```



	Account	Coin	Execution Price	Size Tokens	Size USD	Side	Timestamp IST	Start Position	Direction	Closed PnL
0	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9769	986.87	7872.16	BUY	02-12-2024 22:50	0.000000	Buy	0.0
1	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9800	16.00	127.68	BUY	02-12-2024 22:50	986.524596	Buy	0.0
2	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9855	144.09	1150.63	BUY	02-12-2024 22:50	1002.518996	Buy	0.0
3	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9874	142.98	1142.04	BUY	02-12-2024 22:50	1146.558564	Buy	0.0
4	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9894	8.73	69.75	BUY	02-12-2024 22:50	1289.488521	Buy	0.0

```
sentiment_df.head()
```

 Show hidden output

Next steps: [Generate code with sentiment_df](#) [View recommended plots](#) [New interactive sheet](#)

✓ Converting Timestamps to DateFormat


```
#Convert timestamp in trade date
trades_df['Timestamp IST'] = pd.to_datetime(trades_df['Timestamp IST'],format='%d-%m-%Y %H:%M')
trades_df['Date'] = trades_df['Timestamp IST'].dt.date # Create date Column
```

```
#Convert 'date' column in sentiment data
sentiment_df['date'] = pd.to_datetime(sentiment_df['date']).dt.date
```



Merge on Data

```
#Merge the datasets on the date
merged_df = trades_df.merge(sentiment_df, left_on='Date', right_on='date', how='left')
```

```
#Preview Merged Data
merged_df[['Date', 'classification', 'Closed PnL', 'Size USD', 'Fee']].head()
```




	Date	classification	Closed PnL	Size USD	Fee
0	2024-12-02	Extreme Greed	0.0	7872.16	0.345404
1	2024-12-02	Extreme Greed	0.0	127.68	0.005600
2	2024-12-02	Extreme Greed	0.0	1150.63	0.050431
3	2024-12-02	Extreme Greed	0.0	1142.04	0.050043
4	2024-12-02	Extreme Greed	0.0	69.75	0.003055



```
#Making Columns Names Easier
merged_df.rename(columns={
    'classification': 'Sentiment',
    'Closed PnL' : 'PnL',
    'Size USD': 'TradeValue',
    'Fee': 'TradingFee'
}, inplace=True)
```

Analysis


```
#Averge PnL by Market Sentiment
merged_df.groupby('Sentiment')['PnL'].mean()
```



	PnL
Sentiment	
Extreme Fear	34.537862
Extreme Greed	67.892861
Fear	54.290400
Greed	42.743559
Neutral	34.307718

dtype: float64

```
#Average Trade Value(USD) by Sentiment
merged_df.groupby('Sentiment')['TradingValue'].mean()
```



	TradingValue
Sentiment	
Extreme Fear	5349.731843
Extreme Greed	3112.251565
Fear	7816.109931
Greed	5736.884375
Neutral	4782.732661

dtype: float64

```
#Total Trades During Each Sentiment
merged_df['Sentiment'].value_counts()
```



	count
Sentiment	
Fear	61837
Greed	50303
Extreme Greed	39992
Neutral	37686
Extreme Fear	21400

dtype: int64

Visualizing

```
import numpy as np

lower_percentile = merged_df['PnL'].quantile(0.05)
upper_percentile = merged_df['PnL'].quantile(0.95)

filtered_df = merged_df[
    (merged_df['PnL'] >= lower_percentile) &
    (merged_df['PnL'] <= upper_percentile)
]

import seaborn as sns
import matplotlib.pyplot as plt
import os
os.makedirs("outputs", exist_ok=True)

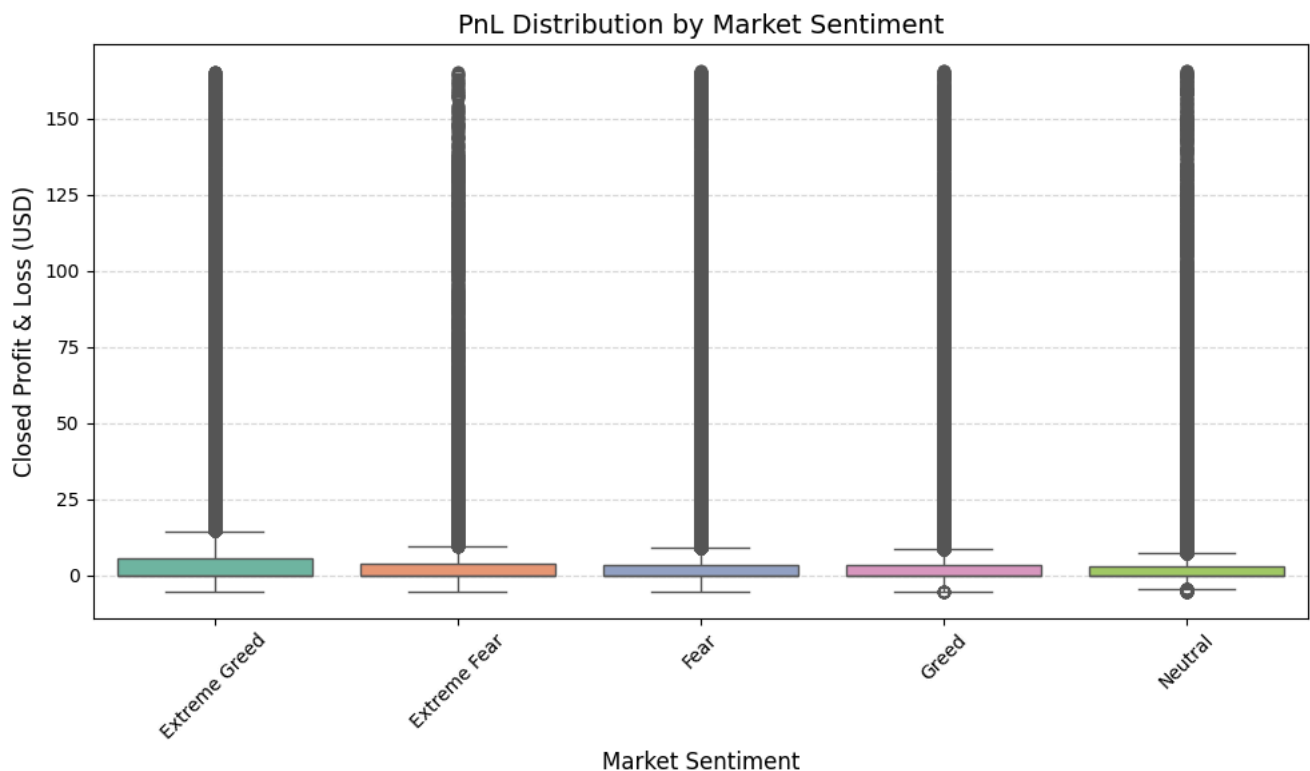
plt.figure(figsize=(10,6))
sns.boxplot(data=filtered_df, x='Sentiment', y='PnL', palette='Set2')
plt.title('PnL Distribution by Market Sentiment', fontsize=14)
plt.xlabel('Market Sentiment', fontsize=12)
plt.ylabel('Closed Profit & Loss (USD)', fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.xticks(rotation=45)
plt.tight_layout()

plt.savefig("outputs/pnl_by_sentiment.png")
plt.show()
```

↗ /tmp/ipython-input-3125914494.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue`

```
sns.boxplot(data=filtered_df, x='Sentiment', y='PnL', palette='Set2')
```



Average PnL Per Sentiment

```
avg_pnl = merged_df.groupby('Sentiment')['PnL'].mean().sort_values()

plt.figure(figsize=(8, 5))
sns.barplot(x=avg_pnl.index, y=avg_pnl.values, palette='coolwarm')

plt.title('Average PnL by Market Sentiment', fontsize=14)
plt.xlabel('Market Sentiment', fontsize=12)
plt.ylabel('Avg PnL (USD)', fontsize=12)
```

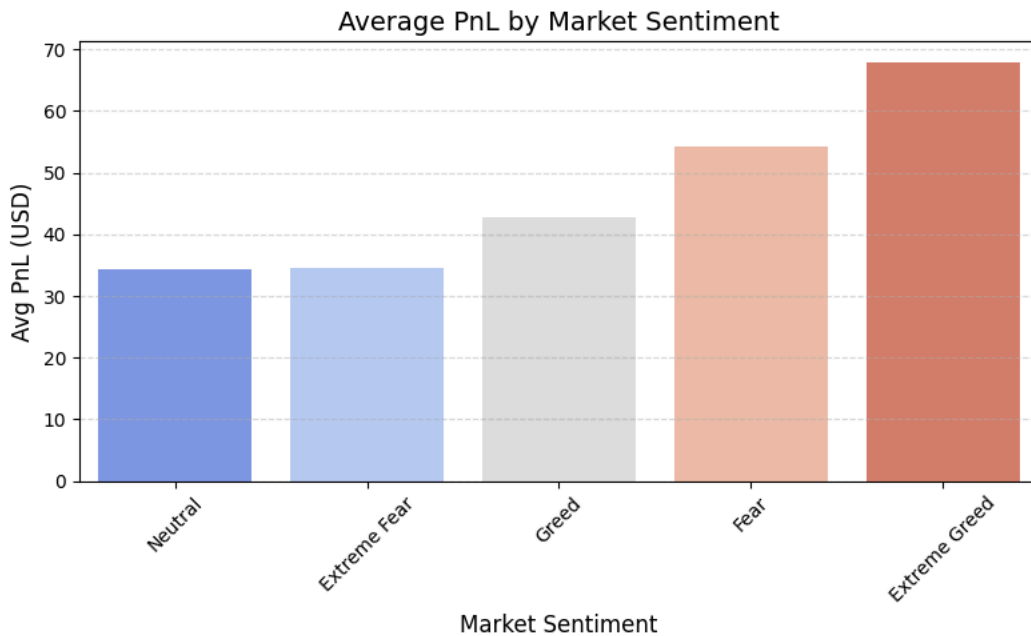
```
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()

plt.savefig("outputs/avg_pnl_by_sentiment.png")
plt.show()
```

 /tmp/ipython-input-1025592589.py:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue`


```
sns.barplot(x=avg_pnl.index, y=avg_pnl.values, palette='coolwarm')
```



✓ Volume vs Sentiment

```
#Group By Sentiment and Calculate Average Trading Value
volume_sentiment = merged_df.groupby('Sentiment')['TradingValue'].mean().reset_index()

plt.figure(figsize=(8,6))
sns.barplot(x='Sentiment', y='TradingValue', data=volume_sentiment, palette='coolwarm')
plt.title('Average Trading Volume per Sentiment')
plt.ylabel('Avg Trading Value')
plt.xlabel('Sentiment')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

 /tmp/ipython-input-3293003986.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue`

```
sns.barplot(x='Sentiment', y='TradingValue', data=volume_sentiment, palette='coolwarm')
```




✓ Risk(Variance of PnL) per Sentiment

#Group By Sentiment and Calculate STD Deviation of PnL

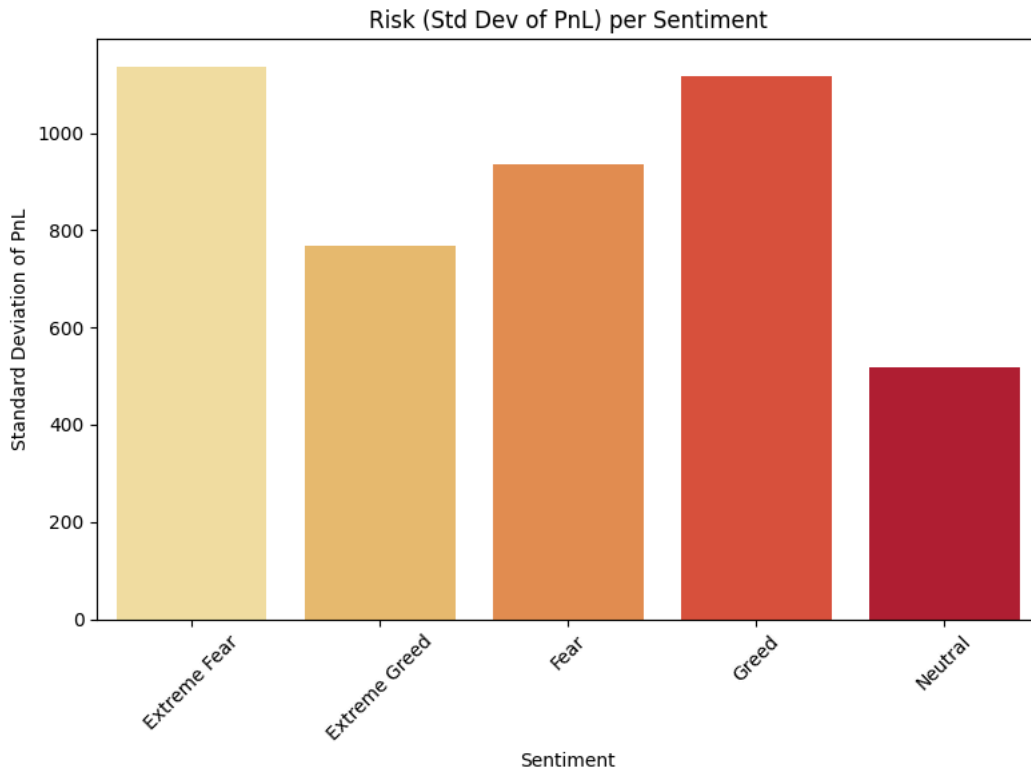
```
risk_sentiment = merged_df.groupby('Sentiment')['PnL'].std().reset_index()  
risk_sentiment.columns = ['Sentiment', 'PnL StdDev']
```

```
plt.figure(figsize=(8,6))  
sns.barplot(x='Sentiment', y='PnL StdDev', data=risk_sentiment, palette='YlOrRd')  
plt.title('Risk (Std Dev of PnL) per Sentiment')  
plt.ylabel('Standard Deviation of PnL')  
plt.xlabel('Sentiment')  
plt.xticks(rotation=45)  
plt.tight_layout()  
plt.show()
```

 /tmp/ipython-input-4101694977.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue`

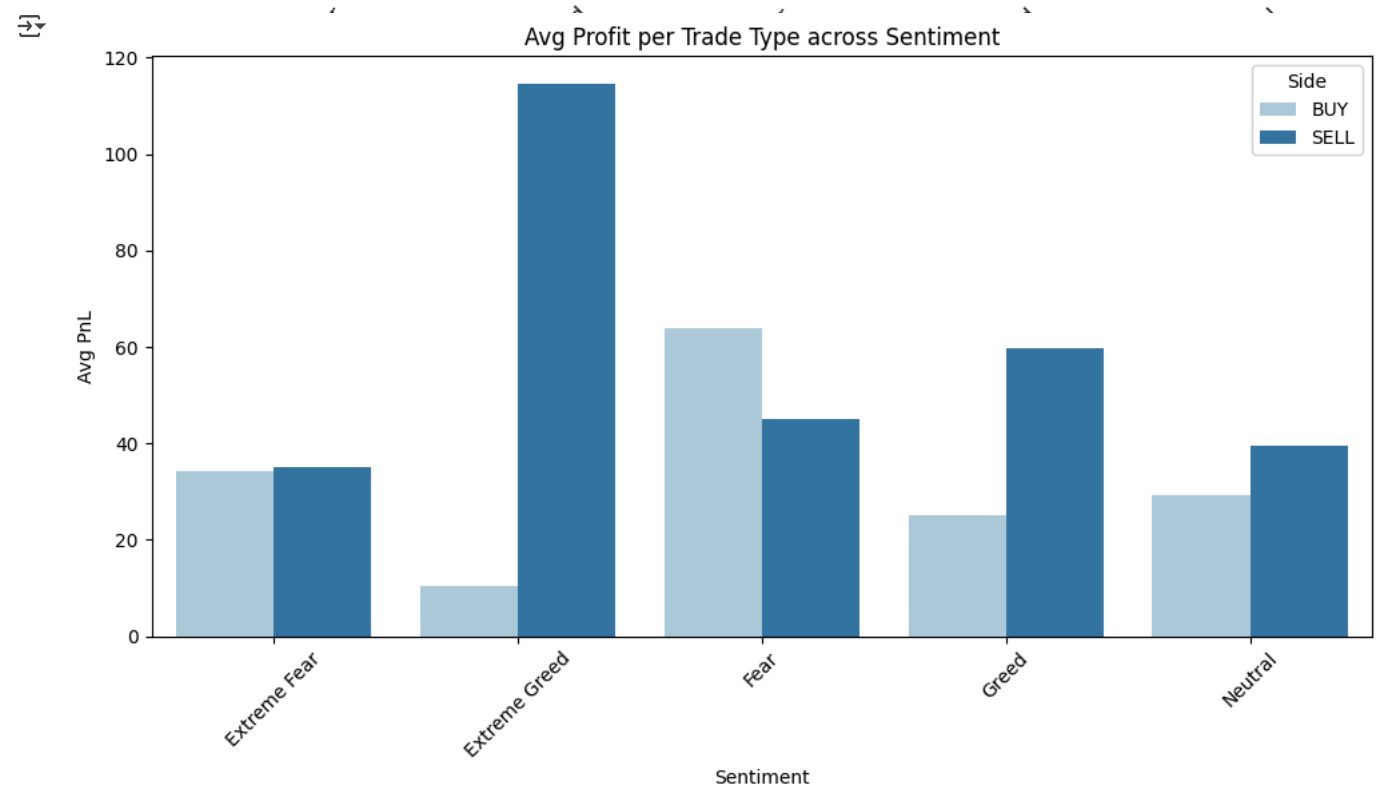
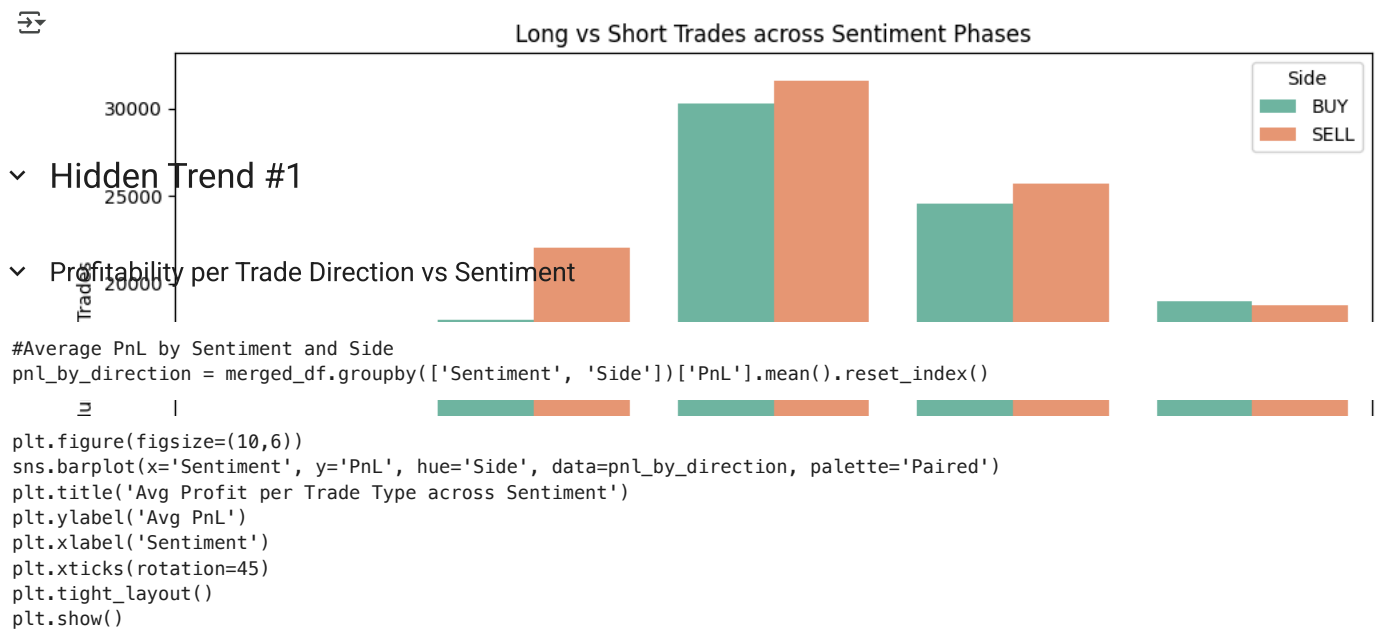
```
sns.barplot(x='Sentiment', y='PnL StdDev', data=risk_sentiment, palette='YlOrRd')
```



✓ Trade Direction (Long vs Short) by Sentiment

```
#Count of Long/Short Grouped by Sentiment
direction_df = merged_df.groupby(['Sentiment', 'Side']).size().reset_index(name='Count')
```

```
plt.figure(figsize=(10,6))
sns.barplot(x='Sentiment', y='Count', hue='Side', data=direction_df, palette='Set2')
plt.title('Long vs Short Trades across Sentiment Phases')
plt.ylabel('Number of Trades')
plt.xlabel('Sentiment')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



/tmp/ipython-input-3577183590.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue`