```
import pandas as pd

df_fear_greed = pd.read_csv("/content/fear_greed_index.csv")
df_historical = pd.read_csv("/content/historical_data.csv")

print("Fear and Greed Index Data:")
display(df_fear_greed.head())
print("\nHistorical Trading Data:")
display(df_historical.head())
```

```
Fear and Greed Index Data:
```

	timestamp	value	classification	date
0	1517463000	30	Fear	2018-02-01
1	1517549400	15	Extreme Fear	2018-02-02
2	1517635800	40	Fear	2018-02-03
3	1517722200	24	Extreme Fear	2018-02-04
4	1517808600	11	Extreme Fear	2018-02-05

```
Historical Trading Data:
```

	Account	Coin	Execution Price	Size Tokens	Size USD	Side	Timestamp IST	S Pos
0	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9769	986.87	7872.16	BUY	02-12-2024 22:50	0.0
1	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9800	16.00	127.68	BUY	02-12-2024 22:50	986.5
2	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9855	144.09	1150.63	BUY	02-12-2024 22:50	1002.5
3	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9874	142.98	1142.04	BUY	02-12-2024 22:50	1146.5
4	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9894	8.73	69.75	BUY	02-12-2024 22:50	1289.4

```
WARNING: Runtime no longer has a reference to this dataframe, please re-run this cell and try again.
```

```
print("Info for df_fear_greed:")
df_fear_greed.info()
print("\nInfo for df_historical:")
df_historical.info()

df_fear_greed['date'] = pd.to_datetime(df_fear_greed['date'])
df_historical['Timestamp'] = pd.to_datetime(df_historical['Timestamp IST'], format='mixed')
df_historical['date'] = df_historical['Timestamp'].dt.date
df_historical['date'] = pd.to_datetime(df_historical['date'])

merged_df = pd.merge(df_historical, df_fear_greed, on='date', how='inner')
```

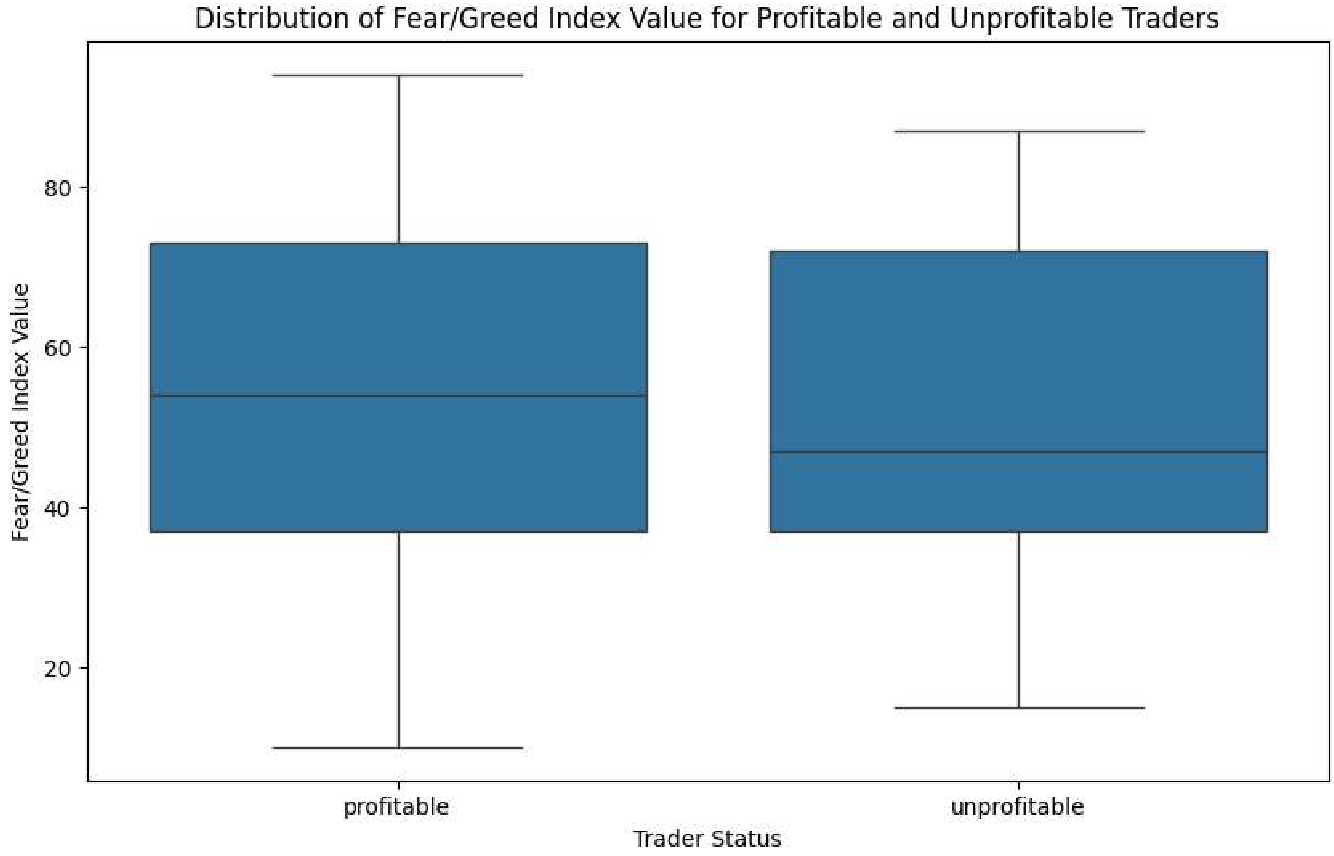
```
    print("\nMerged DataFrame Head:")
    display(merged_df.head())
    print("\nMerged DataFrame Info:")
    merged_df.info()
```

Analyze trader performance vs. market sentiment

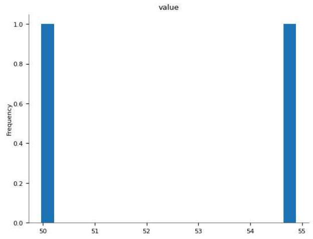
```
import matplotlib.pyplot as plt
import seaborn as sns

trader_pnl = merged_df.groupby('Account')['Closed PnL'].sum().reset_index()

trader_pnl['status'] = ['profitable' if pnl > 0 else 'unprofitable' for pnl in trader_pnl['Closed PnL']]

merged_df_with_pnl = pd.merge(merged_df, trader_pnl[['Account', 'status']], on='Account', how='left')

average_sentiment = merged_df_with_pnl.groupby('status')['value'].mean().reset_index()
print("Average Fear/Greed Value by Trader Status:")
display(average_sentiment)

plt.figure(figsize=(10, 6))
sns.boxplot(x='status', y='value', data=merged_df_with_pnl)
plt.title('Distribution of Fear/Greed Index Value for Profitable and Unprofitable Traders')
plt.xlabel('Trader Status')
plt.ylabel('Fear/Greed Index Value')
plt.show()
```

```
Average Fear/Greed Value by Trader Status:
```

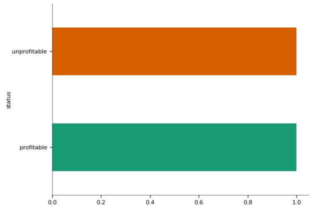
	status	value
0	profitable	54.888262
1	unprofitable	49.967084



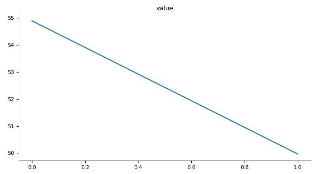
Distributions



Categorical distributions



Values



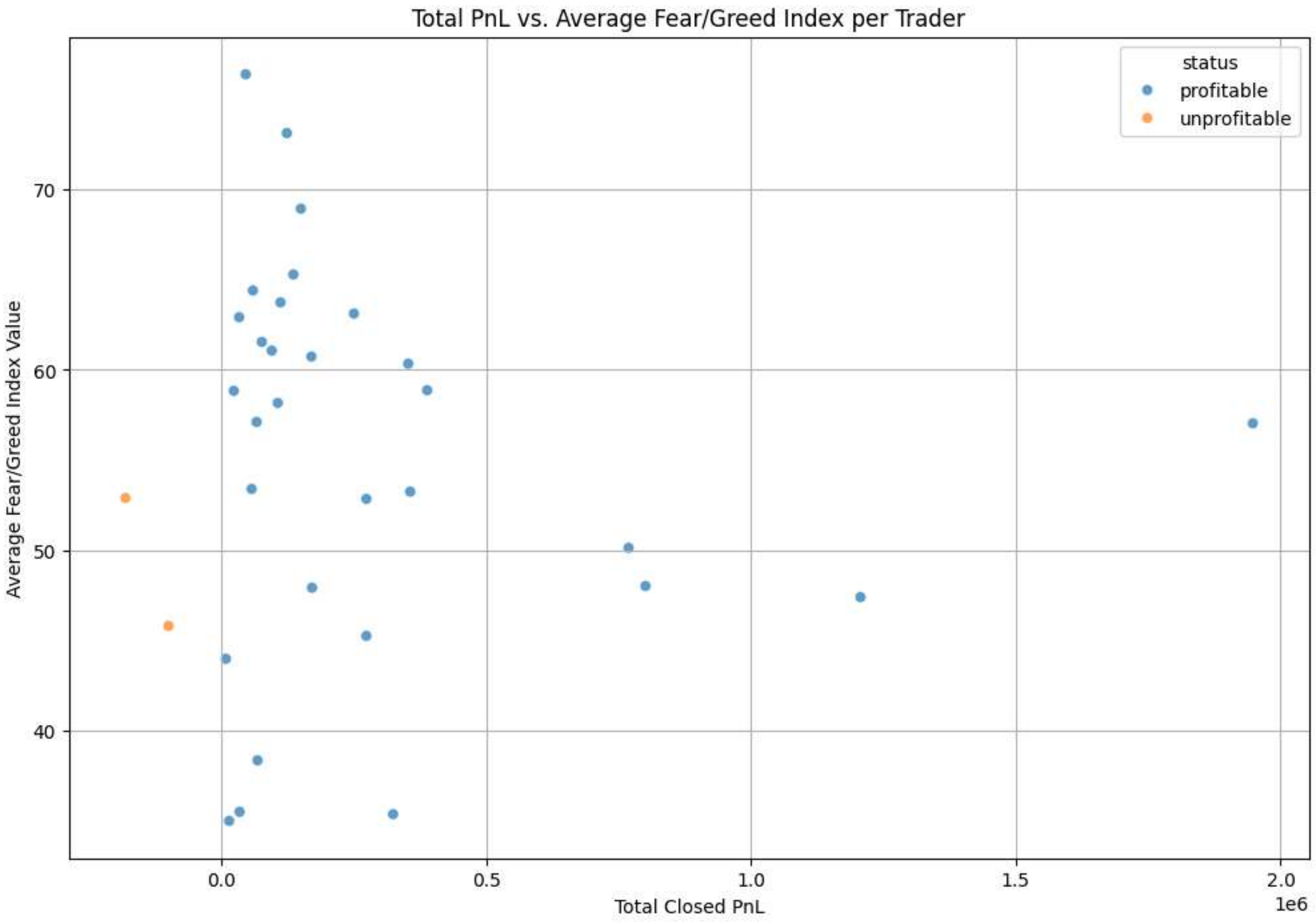
Faceted distributions

```
<string>:5: FutureWarning:
```

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

Next steps: Generate code with average_sentiment | View recommended plots | New interactive sheet

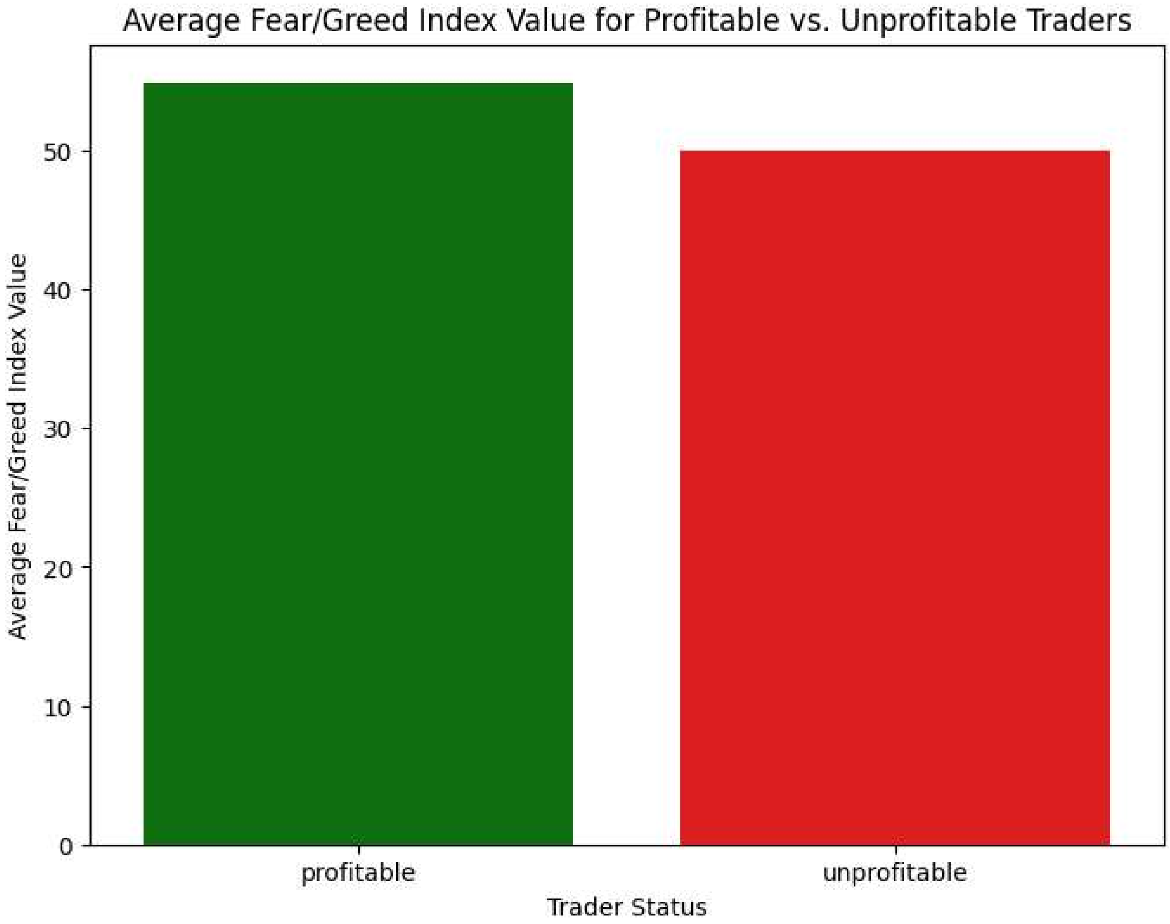
```
avg_sentiment_per_trader = merged_df_with_pnl.groupby('Account')['value'].mean().reset_index()
merged_pnl_sentiment = pd.merge(trader_pnl, avg_sentiment_per_trader, on='Account')

plt.figure(figsize=(12, 8))
sns.scatterplot(x='Closed PnL', y='value', data=merged_pnl_sentiment, hue='status', alpha=0.7)
plt.title('Total PnL vs. Average Fear/Greed Index per Trader')
plt.xlabel('Total Closed PnL')
plt.ylabel('Average Fear/Greed Index Value')
plt.grid(True)
plt.show()
```



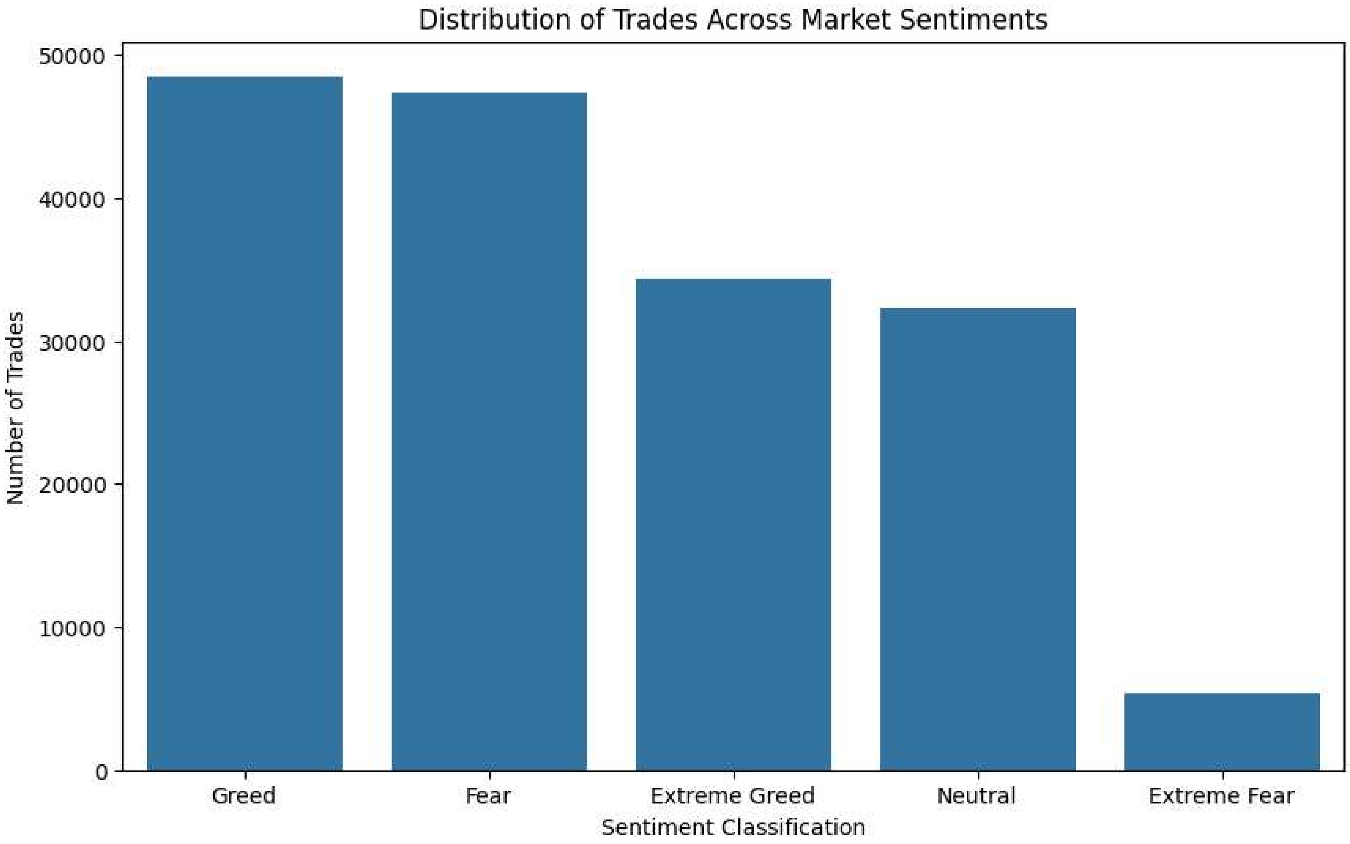
```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
plt.figure(figsize=(8, 6))
sns.barplot(x='status', y='value', data=average_sentiment, palette=['green', 'red'])
plt.title('Average Fear/Greed Index Value for Profitable vs. Unprofitable Traders')
plt.xlabel('Trader Status')
plt.ylabel('Average Fear/Greed Index Value')
plt.show()
```

```
/tmp/ipython-input-7-2725640262.py:5: FutureWarning:

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

  sns.barplot(x='status', y='value', data=average_sentiment, palette=['green', 'red'])
```



```
plt.figure(figsize=(10, 6))
sns.countplot(x='classification', data=merged_df_with_pnl, order=merged_df_with_pnl['classification'].value
plt.title('Distribution of Trades Across Market Sentiments')
plt.xlabel('Sentiment Classification')
plt.ylabel('Number of Trades')
plt.show()
```

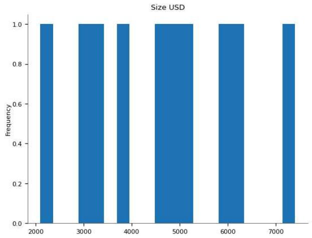


Uncover hidden patterns

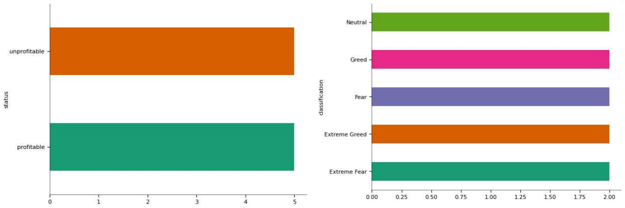
```
trade_behavior_by_sentiment = merged_df_with_pnl.groupby(['status', 'classification'])['Size USD'].mean().r
display(trade_behavior_by_sentiment)
```

	status	classification	Size USD
0	profitable	Extreme Fear	3791.328449
1	profitable	Extreme Greed	3269.159437
2	profitable	Fear	7405.582685
3	profitable	Greed	5830.589870
4	profitable	Neutral	4628.032156
5	unprofitable	Extreme Fear	5252.394409
6	unprofitable	Extreme Greed	2100.170303
7	unprofitable	Fear	2989.843446
8	unprofitable	Greed	6224.872062
9	unprofitable	Neutral	4942.737196

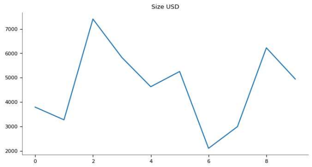
Distributions



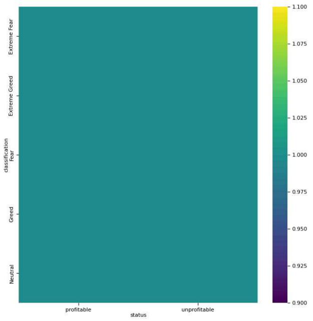
Categorical distributions



Values

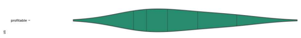


2-d categorical distributions



Faceted distributions

```
<string>:5: FutureWarning:

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



```
<string>:5: FutureWarning:
```