In [2]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```python
sentiment = pd.read_csv("fear_greed_index.csv")
sentiment.head()
```

Out[3]:

	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

In [7]:
```python
import os
os.listdir()
```

```
Out[7]: ['.anaconda',
 '.conda',
 '.condarc',
 '.continuum',
 '.gitconfig',
 '.ipynb_checkpoints',
 '.ipython',
 '.jupyter',
 '.matplotlib',
 '.python_history',
 '.virtual_documents',
 '.Xilinx',
 'anaconda3',
 'anaconda_projects',
 'AppData',
 'Application Data',
 'Bitcoin_Sentiment_Analysis.ipynb',
 'Contacts',
 'Cookies',
 'Data Dictionary-Copy1.txt',
 'Data Dictionary.txt',
 'Desktop',
 'Documents',
 'Downloads',
 'Favorites',
 'fear_greed_index.csv',
 'Fraud (1)-Copy1.csv',
 'Fraud (1).csv',
 'fraud.........txt',
 'Fraud_Detection.ipynb',
 'Fraud_Detection_final.ipynb',
 'Fraud_Project',
 'fraud_xgb_model-Copy1.pkl',
 'fraud_xgb_model.pkl',
 'historical_data.csv',
 'Links',
 'Local Settings',
 'Microsoft',
 'missing values.ipynb',
 'Music',
 'My Documents',
 'NetHood',
 'NTUSER.DAT',
 'ntuser.dat.LOG1',
 'ntuser.dat.LOG2',
 'NTUSER.DAT{2ad838bb-efea-11ee-a54d-000d3a94eaa1}.TxR.0.regtrans-ms',
 'NTUSER.DAT{2ad838bb-efea-11ee-a54d-000d3a94eaa1}.TxR.1.regtrans-ms',
 'NTUSER.DAT{2ad838bb-efea-11ee-a54d-000d3a94eaa1}.TxR.2.regtrans-ms',
 'NTUSER.DAT{2ad838bb-efea-11ee-a54d-000d3a94eaa1}.TxR.blf',
 'NTUSER.DAT{2ad838bc-efea-11ee-a54d-000d3a94eaa1}.TM.blf',
 'NTUSER.DAT{2ad838bc-efea-11ee-a54d-000d3a94eaa1}.TMContainer0000000000000000000
1.regtrans-ms',
 'NTUSER.DAT{2ad838bc-efea-11ee-a54d-000d3a94eaa1}.TMContainer0000000000000000000
2.regtrans-ms',
 'ntuser.ini',
 'OneDrive',
```

```
 'PrintHood',
 'Recent',
 'requirements-Copy1.txt',
 'requirements.txt',
 'Saved Games',
 'Searches',
 'SendTo',
 'Start Menu',
 'Templates',
 'Untitled.ipynb',
 'untitled.py',
 'Untitled1-Copy1.ipynb',
 'Untitled1.ipynb',
 'Untitled2.ipynb',
 'Untitled3.ipynb',
 'Untitled4.ipynb',
 'Videos']
```

In [9]:
```
trades = pd.read_csv("historical_data.csv")
trades.head()
```

Out[9]:

	Account	Coin	Execution Price	Size Tokens	Size USD	Side
0	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9769	986.87	7872.16	BUY
1	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9800	16.00	127.68	BUY
2	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9855	144.09	1150.63	BUY
3	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9874	142.98	1142.04	BUY
4	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9894	8.73	69.75	BUY

In [11]:
```
trades['Timestamp IST'].head(5)
```

Out[11]:
```
0    02-12-2024 22:50
1    02-12-2024 22:50
2    02-12-2024 22:50
3    02-12-2024 22:50
4    02-12-2024 22:50
Name: Timestamp IST, dtype: object
```

In [14]:
```
trades['Date'] = pd.to_datetime(
    trades['Timestamp IST'],
    dayfirst=True
).dt.date
trades[['Timestamp IST', 'Date']].head()
```

Out[14]:

	Timestamp IST	Date
0	02-12-2024 22:50	2024-12-02
1	02-12-2024 22:50	2024-12-02
2	02-12-2024 22:50	2024-12-02
3	02-12-2024 22:50	2024-12-02
4	02-12-2024 22:50	2024-12-02

In [16]:
```
sentiment.columns
```

Out[16]:
```
Index(['timestamp', 'value', 'classification', 'date'], dtype='object')
```

In [17]:
```
sentiment['Date'] = pd.to_datetime(
    sentiment['date'],
    errors='coerce'
).dt.date
```

In [18]:
```
sentiment[['Date']].head()
```

Out[18]:

	Date
0	2018-02-01
1	2018-02-02
2	2018-02-03
3	2018-02-04
4	2018-02-05

In [19]:
```
merged = pd.merge(
    trades,
    sentiment,
    on='Date',
    how='left'
)

merged.head()
```

Out[19]:

	Account	Coin	Execution Price	Size Tokens	Size USD	Side	T
0	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9769	986.87	7872.16	BUY	(
1	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9800	16.00	127.68	BUY	(
2	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9855	144.09	1150.63	BUY	(
3	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9874	142.98	1142.04	BUY	(
4	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9894	8.73	69.75	BUY	(

5 rows × 21 columns

In [21]:

```
merged.head()
```

Out[21]:

	Account	Coin	Execution Price	Size Tokens	Size USD	Side	T
0	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9769	986.87	7872.16	BUY	(
1	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9800	16.00	127.68	BUY	(
2	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9855	144.09	1150.63	BUY	(
3	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9874	142.98	1142.04	BUY	(
4	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9894	8.73	69.75	BUY	(

5 rows × 21 columns

In [22]:

```
merged.columns
```

Out[22]:

```
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', 'Date', 'timestamp', 'value', 'classification', 'date'],
      dtype='object')
```

In [23]:

```
merged['classification'].isna().value_counts()
```

```
Out[23]: classification
False    211218
True          6
Name: count, dtype: int64
```

```python
In [24]: merged.columns = merged.columns.str.strip().str.lower()
```

```python
In [25]: merged = merged.dropna(subset=['classification'])
```

```python
In [26]: merged = merged.dropna(subset=['classification'])
```

```python
In [31]: merged.groupby('classification')['closed pnl'].mean()
```

```
Out[31]: classification
Extreme Fear     34.537862
Extreme Greed    67.892861
Fear             54.290400
Greed            42.743559
Neutral          34.307718
Name: closed pnl, dtype: float64
```

```python
In [32]: merged.groupby('classification')['closed pnl'].sum()
```

```
Out[32]: classification
Extreme Fear     7.391102e+05
Extreme Greed    2.715171e+06
Fear             3.357155e+06
Greed            2.150129e+06
Neutral          1.292921e+06
Name: closed pnl, dtype: float64
```

```python
In [33]: merged['is_profit'] = (merged['closed pnl'] > 0).astype(int)

merged.groupby('classification')['is_profit'].mean()
```

```
Out[33]: classification
Extreme Fear     0.370607
Extreme Greed    0.464943
Fear             0.420768
Greed            0.384828
Neutral          0.396991
Name: is_profit, dtype: float64
```

```python
In [35]: merged.groupby('classification')['size usd'].mean()
```

```
Out[35]: classification
Extreme Fear     5349.731843
Extreme Greed    3112.251565
Fear             7816.109931
Greed            5736.884375
Neutral          4782.732661
Name: size usd, dtype: float64
```

```python
In [36]: merged.groupby('classification')['size usd'].sum()
```

Out[36]:
```
classification
Extreme Fear     1.144843e+08
Extreme Greed    1.244652e+08
Fear             4.833248e+08
Greed            2.885825e+08
Neutral          1.802421e+08
Name: size usd, dtype: float64
```

In [37]:
```
pd.crosstab(merged['classification'], merged['side'])
```

Out[37]:

side classification	BUY	SELL
Extreme Fear	10935	10465
Extreme Greed	17940	22052
Fear	30270	31567
Greed	24576	25727
Neutral	18969	18717

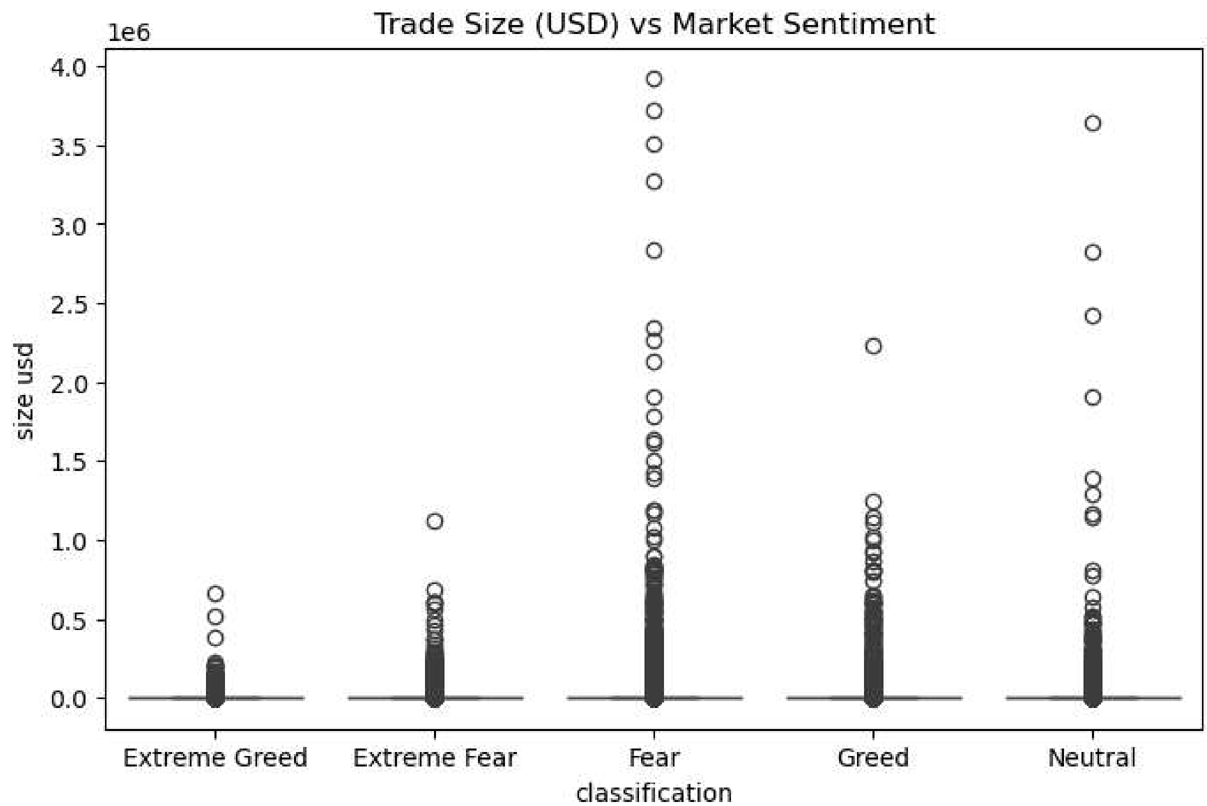
In [38]:
```
merged.groupby('classification')['closed pnl'].mean()
```

Out[38]:
```
classification
Extreme Fear     34.537862
Extreme Greed    67.892861
Fear             54.290400
Greed            42.743559
Neutral          34.307718
Name: closed pnl, dtype: float64
```

In [39]:
```
merged['is_profit'] = (merged['closed pnl'] > 0).astype(int)
merged.groupby('classification')['is_profit'].mean()
```

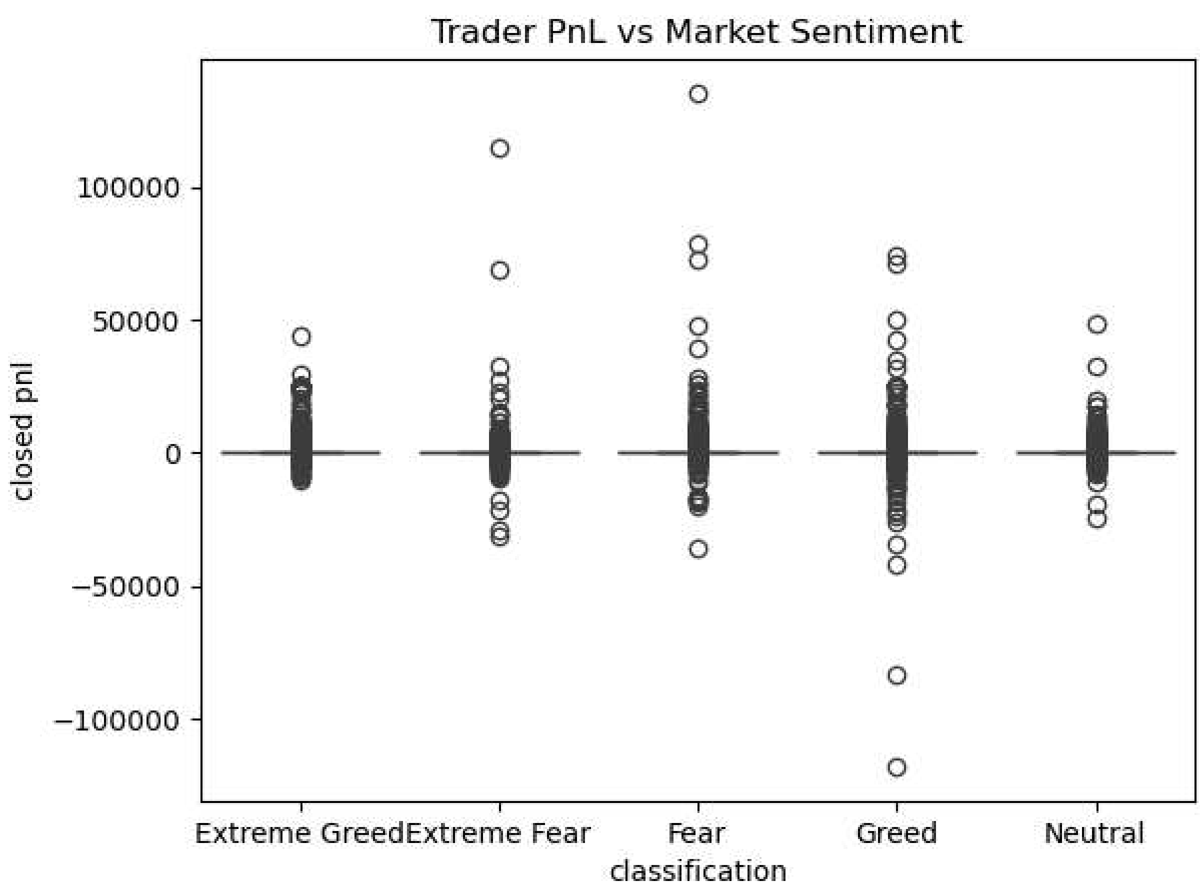
Out[39]:
```
classification
Extreme Fear     0.370607
Extreme Greed    0.464943
Fear             0.420768
Greed            0.384828
Neutral          0.396991
Name: is_profit, dtype: float64
```

In [40]:
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(8,5))
sns.boxplot(x='classification', y='size usd', data=merged)
plt.title("Trade Size (USD) vs Market Sentiment")
plt.show()
```



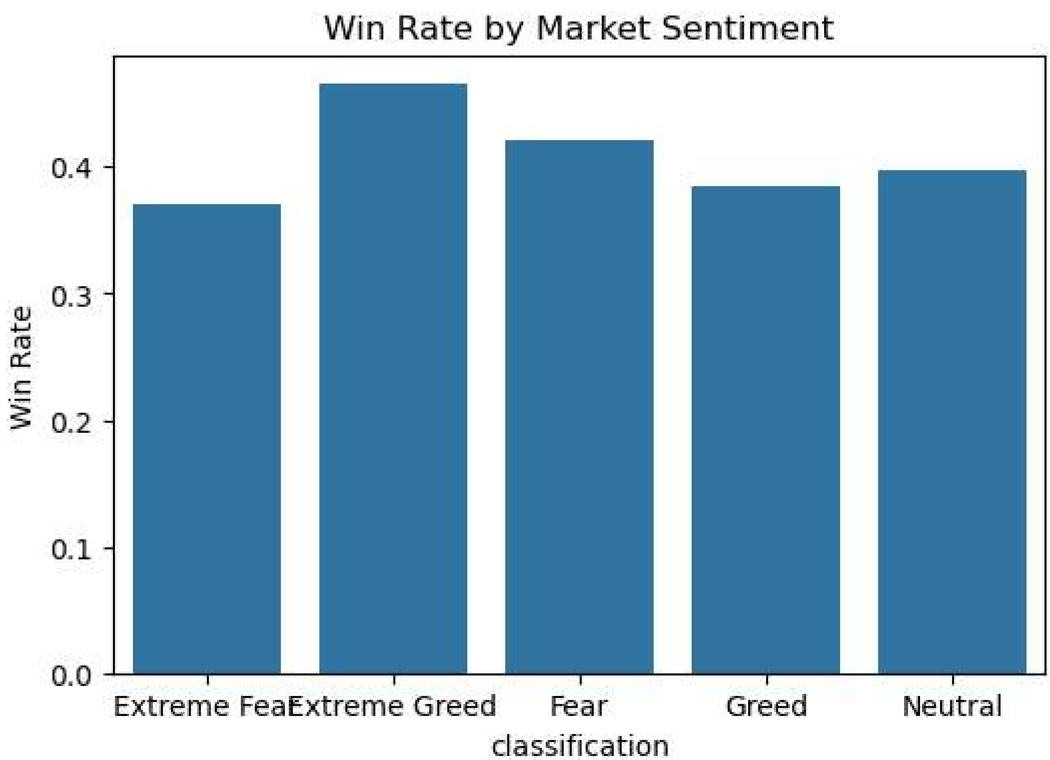
In [41]:

```
sns.boxplot(x='classification', y='closed pnl', data=merged)
plt.title("Trader PnL vs Market Sentiment")
plt.show()
```



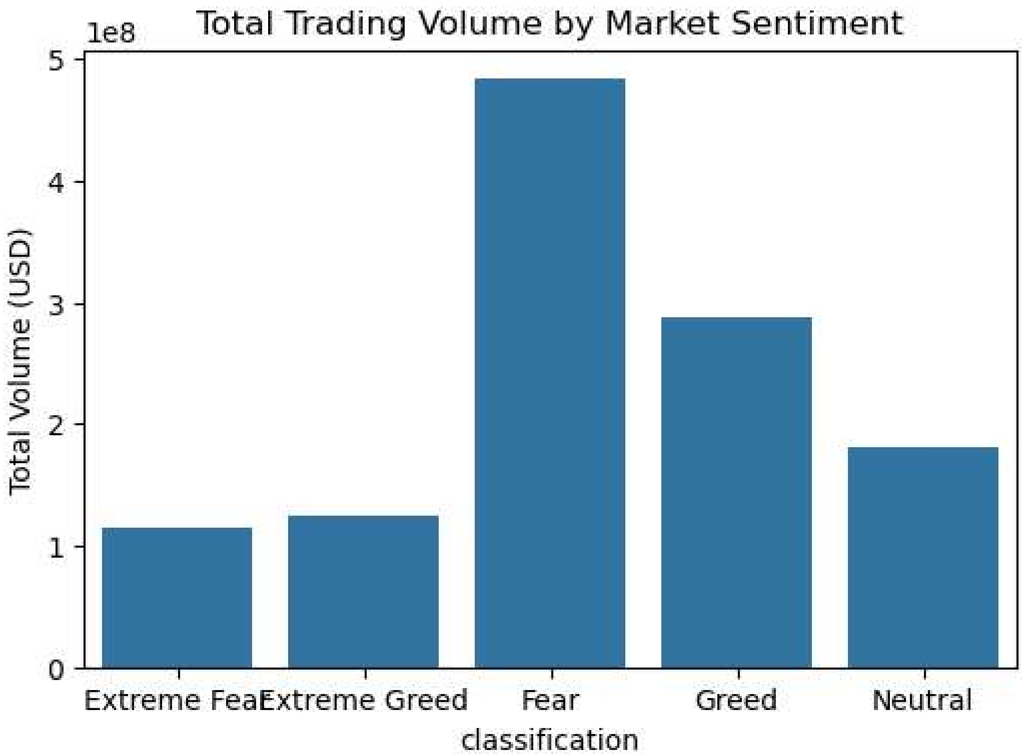
In [42]:

```
merged['is_profit'] = (merged['closed pnl'] > 0).astype(int)

win_rate = merged.groupby('classification')['is_profit'].mean().reset_index()

plt.figure(figsize=(6,4))
sns.barplot(x='classification', y='is_profit', data=win_rate)
plt.title("Win Rate by Market Sentiment")
plt.ylabel("Win Rate")
plt.show()
```

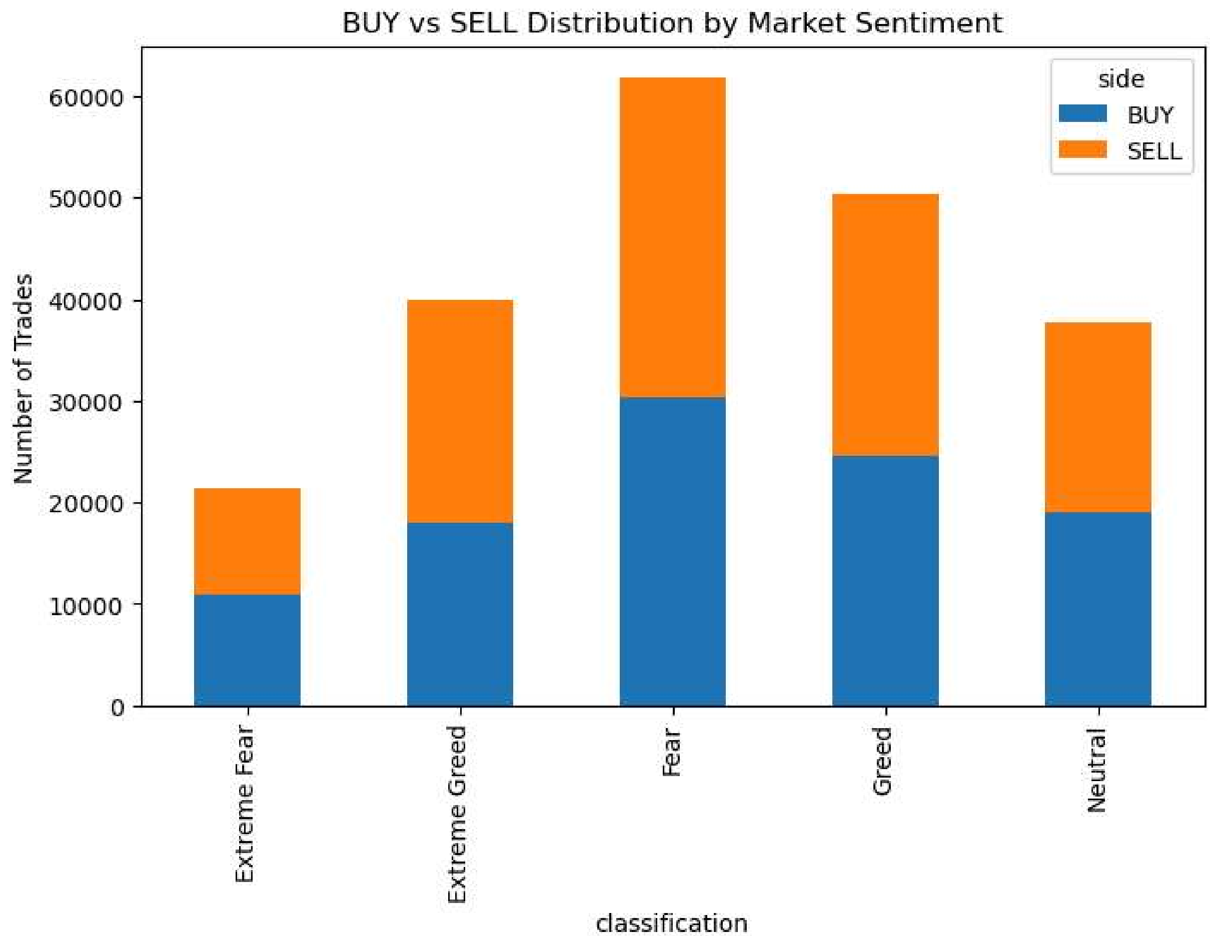


In [43]:

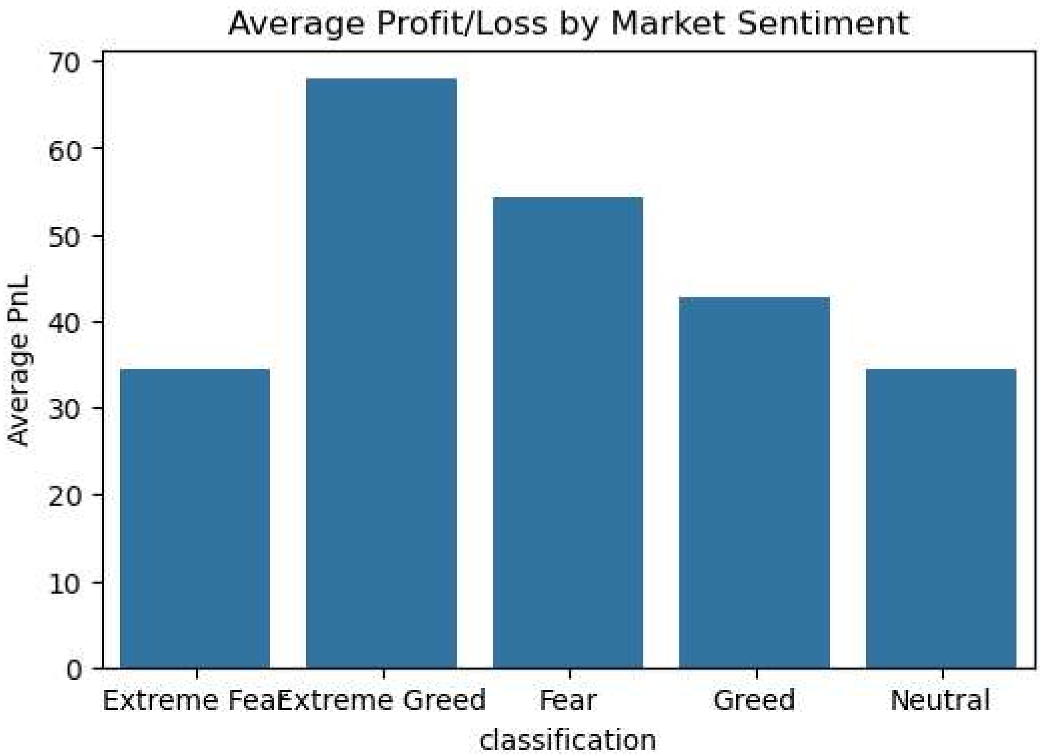
```
volume = merged.groupby('classification')['size usd'].sum().reset_index()

plt.figure(figsize=(6,4))
sns.barplot(x='classification', y='size usd', data=volume)
plt.title("Total Trading Volume by Market Sentiment")
plt.ylabel("Total Volume (USD)")
plt.show()
```



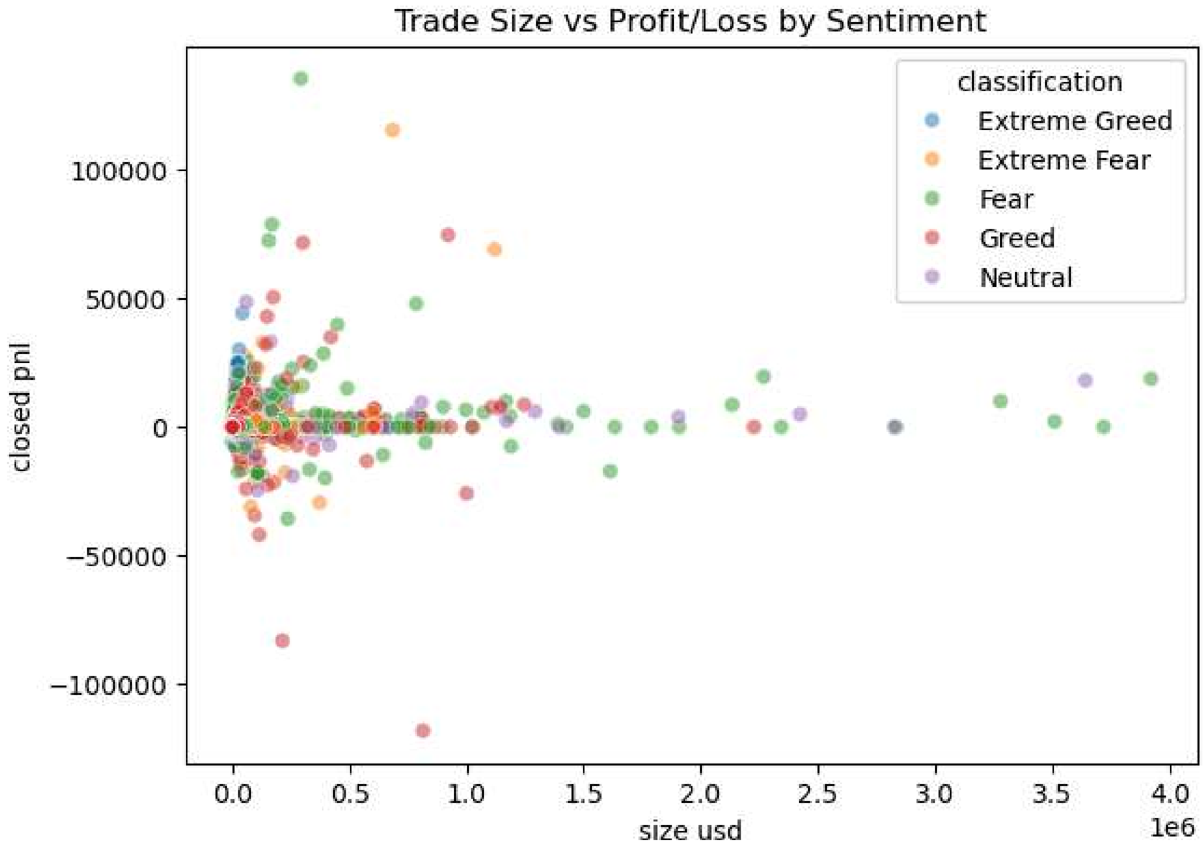
In [44]:
```
side_count = pd.crosstab(merged['classification'], merged['side'])

side_count.plot(kind='bar', stacked=True, figsize=(8,5))
plt.title("BUY vs SELL Distribution by Market Sentiment")
plt.ylabel("Number of Trades")
plt.show()
```



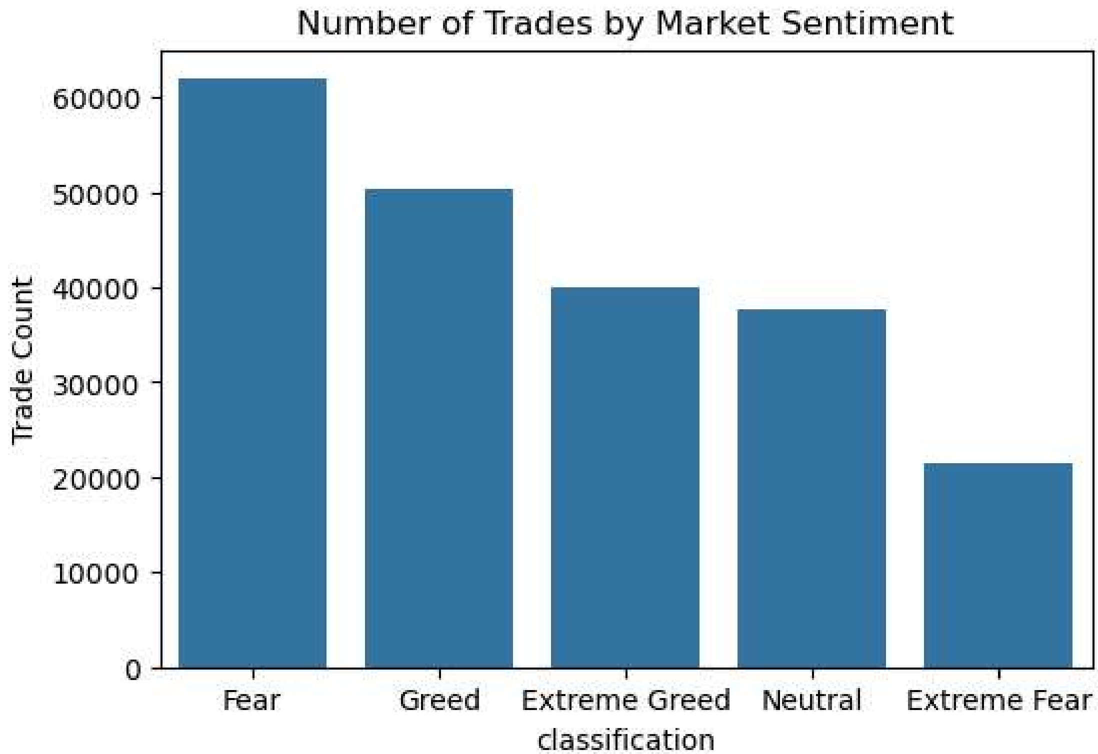
In [45]:
```
avg_pnl = merged.groupby('classification')['closed pnl'].mean().reset_index()

plt.figure(figsize=(6,4))
sns.barplot(x='classification', y='closed pnl', data=avg_pnl)
plt.title("Average Profit/Loss by Market Sentiment")
plt.ylabel("Average PnL")
plt.show()
```



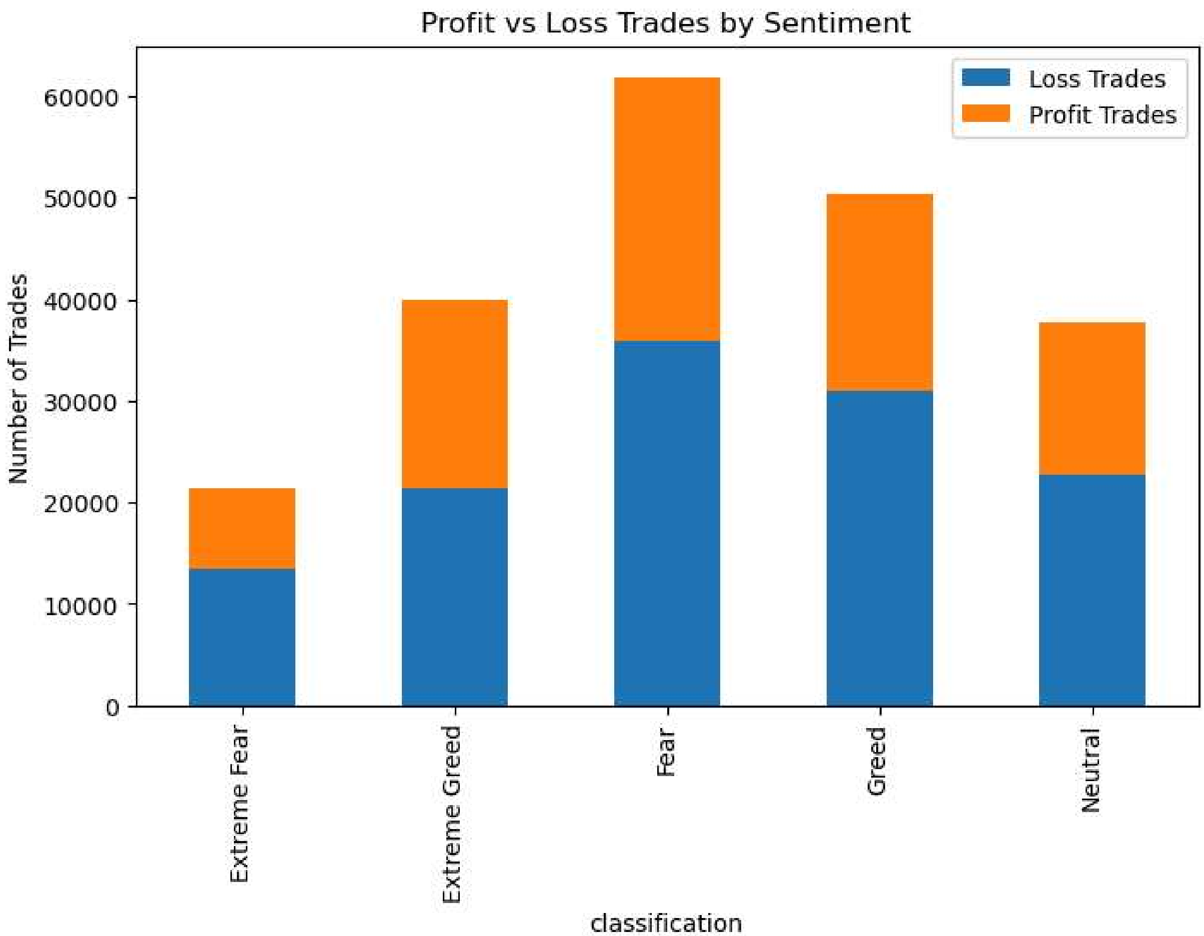
In [46]:
```
plt.figure(figsize=(7,5))
sns.scatterplot(
    x='size usd',
    y='closed pnl',
    hue='classification',
    data=merged,
    alpha=0.5
)
plt.title("Trade Size vs Profit/Loss by Sentiment")
plt.show()
```



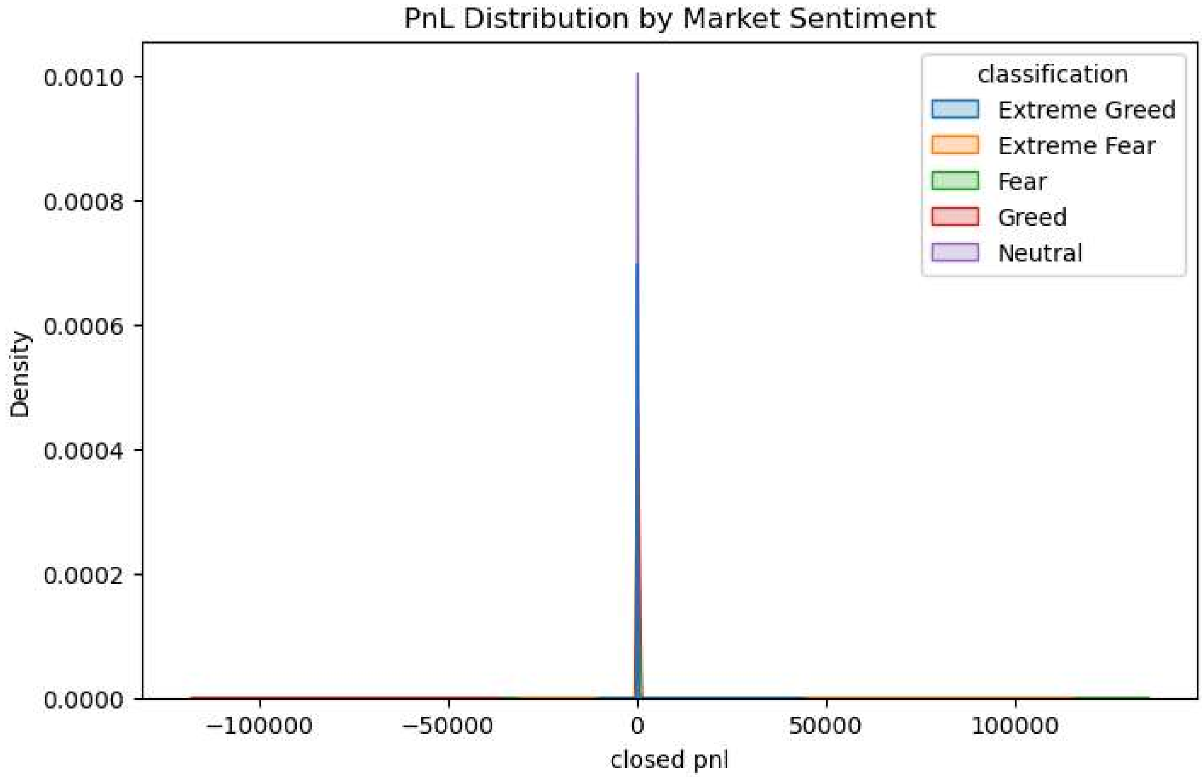
In [47]:
```
trade_count = merged['classification'].value_counts().reset_index()
trade_count.columns = ['classification', 'trade_count']

plt.figure(figsize=(6,4))
sns.barplot(x='classification', y='trade_count', data=trade_count)
plt.title("Number of Trades by Market Sentiment")
plt.ylabel("Trade Count")
plt.show()
```



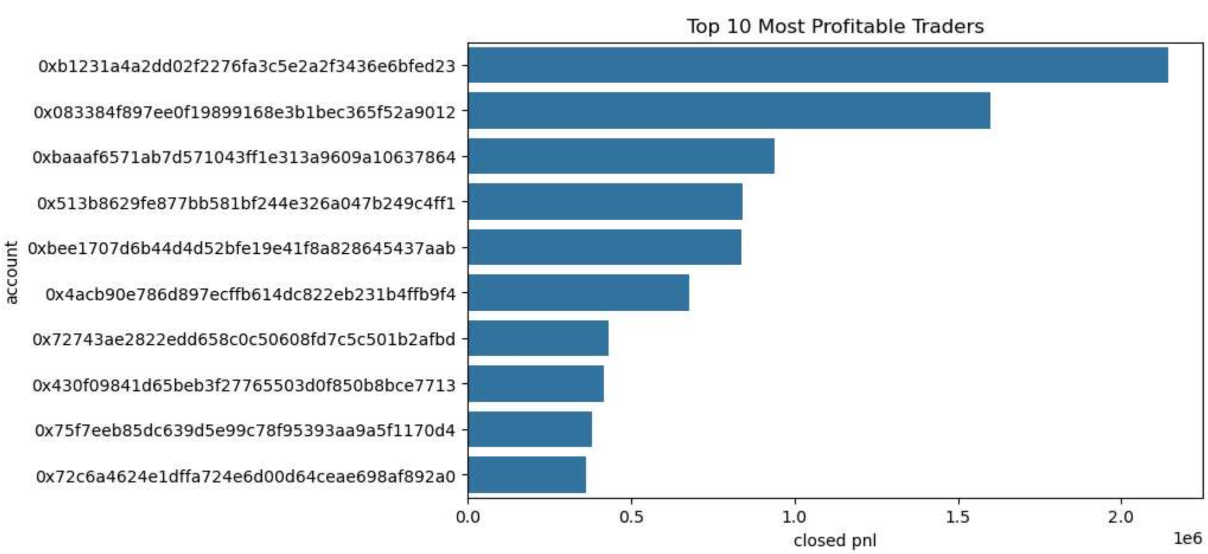
In [48]:
```
profit_loss = pd.crosstab(
    merged['classification'],
    merged['is_profit']
)

profit_loss.columns = ['Loss Trades', 'Profit Trades']

profit_loss.plot(kind='bar', stacked=True, figsize=(8,5))
plt.title("Profit vs Loss Trades by Sentiment")
plt.ylabel("Number of Trades")
plt.show()
```



In [49]:
```
plt.figure(figsize=(8,5))
sns.kdeplot(
    data=merged,
    x='closed pnl',
    hue='classification',
    fill=True
)
plt.title("PnL Distribution by Market Sentiment")
plt.show()
```



In [50]:
```
top_accounts = (
    merged.groupby('account')['closed pnl']
    .sum()
    .sort_values(ascending=False)
    .head(10)
    .reset_index()
)

plt.figure(figsize=(8,5))
sns.barplot(y='account', x='closed pnl', data=top_accounts)
plt.title("Top 10 Most Profitable Traders")
plt.show()
```



In [52]:
```
merged.groupby('classification')['closed pnl'].agg([
    'mean',
    'median',
    'std',
    'count'
])
```

Out[52]:

classification	mean	median	std	count
Extreme Fear	34.537862	0.0	1136.056091	21400
Extreme Greed	67.892861	0.0	766.828294	39992
Fear	54.290400	0.0	935.355438	61837
Greed	42.743559	0.0	1116.028390	50303
Neutral	34.307718	0.0	517.122220	37686

In [53]:
```
win_rate = merged.groupby('classification')['is_profit'].mean()
win_rate
```

Out[53]:
```
classification
Extreme Fear     0.370607
Extreme Greed    0.464943
Fear             0.420768
Greed            0.384828
Neutral          0.396991
Name: is_profit, dtype: float64
```

In [54]:
```
merged.groupby('classification')['closed pnl'].min()
```

Out[54]:
```
classification
Extreme Fear     -31036.69194
Extreme Greed    -10259.46800
Fear             -35681.74723
Greed           -117990.10410
Neutral          -24500.00000
Name: closed pnl, dtype: float64
```

In [55]:
```
merged.groupby('classification')['closed pnl'].std()
```

Out[55]:
```
classification
Extreme Fear     1136.056091
Extreme Greed     766.828294
Fear              935.355438
Greed            1116.028390
Neutral           517.122220
Name: closed pnl, dtype: float64
```

In [56]:
```
merged['classification'].value_counts()
```

Out[56]:
```
classification
Fear             61837
Greed            50303
Extreme Greed    39992
Neutral          37686
Extreme Fear     21400
Name: count, dtype: int64
```

In [57]:
```
pd.crosstab(
    merged['classification'],
    merged['side'],
    normalize='index'
)
```

Out[57]:

side / classification	BUY	SELL
Extreme Fear	0.510981	0.489019
Extreme Greed	0.448590	0.551410
Fear	0.489513	0.510487
Greed	0.488559	0.511441
Neutral	0.503343	0.496657

In [ ]:
```
## Key Insights

- Trader performance varies significantly across market sentiment phases.
- Fear-driven markets show higher win rates and lower volatility.
- Greed periods encourage aggressive trading and higher risk exposure.
- Maximum drawdowns are observed during Greed conditions.
 ## Trading Strategy Recommendations

- Reduce leverage and position size during Greed sentiment.
- Prioritize disciplined trades during Fear periods.
- Integrate sentiment indicators as risk filters.
- Avoid emotional trading during extreme market optimism.
 ## Conclusion

Market sentiment has a measurable impact on trader profitability and risk behavior.
By combining sentiment data with execution-level trading data, traders can
improve decision-making and risk management in volatile crypto markets.
```