Trader Performance vs Market Sentiment (Primetrade.ai)

Goal: Analyze whether trader behavior/performance differs on Fear vs Greed days, segment traders, and propose actionable strategy ideas.

This notebook is the primary submission artifact (charts/tables + short write-up).

0) Setup

In [1]:
```
import os
import sys
from pathlib import Path

import numpy as np
import pandas as pd
import plotly.express as px

# Ensure repo root is on sys.path (works even if notebook runs from notebooks/)
REPO_ROOT = Path.cwd().resolve()
if not (REPO_ROOT / "src").exists():
    for parent in REPO_ROOT.parents:
        if (parent / "src").exists():
            REPO_ROOT = parent
            break

if str(REPO_ROOT) not in sys.path:
    sys.path.insert(0, str(REPO_ROOT))

# Ensure relative paths (data/, output/) resolve from repo root
os.chdir(REPO_ROOT)

from src.data_loader import DataLoader
from src.analysis import Analyzer

OUTPUT_DIR = REPO_ROOT / "output"
CHARTS_DIR = OUTPUT_DIR / "charts"
TABLES_DIR = OUTPUT_DIR / "tables"
REPORTS_DIR = OUTPUT_DIR / "reports"
for d in [CHARTS_DIR, TABLES_DIR, REPORTS_DIR]:
    d.mkdir(parents=True, exist_ok=True)

pd.set_option("display.max_columns", 200)
pd.set_option("display.width", 140)
pd.set_option("display.float_format", lambda x: f"{x:,.4f}")
```

1) Load Data + Basic QA (rows/cols, missing values, duplicates)

In [2]:
```
loader = DataLoader()
sentiment_df, trades_df = loader.load_data()
merged_df = loader.clean_and_merge()

print("Sentiment:", sentiment_df.shape)
print("Trades:", trades_df.shape)
print("Merged:", merged_df.shape)

display(merged_df.head())
```

```
Warning: Leverage column missing. Synthesizing data for analysis pipeline.
Trades Columns: ['Account', 'Coin', 'Execution Price', 'Size Tokens', 'Size USD', 'S
ide', 'Timestamp IST', 'Start Position', 'Direction', 'Closed PnL', 'Transaction Has
h', 'Order ID', 'Crossed', 'Fee', 'Trade ID', 'Timestamp', 'leverage']
Sentiment raw shape: (2644, 4)
Trades raw shape: (211224, 17)
Sentiment: (2644, 4)
Trades: (211224, 17)
Merged: (211218, 21)
```

	Account	Coin	Execution Price	Size Tokens	Size USD	Side
0	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9769	986.8700	7,872.1600	BUY
1	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9800	16.0000	127.6800	BUY
2	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9855	144.0900	1,150.6300	BUY
3	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9874	142.9800	1,142.0400	BUY
4	0xae5eacaf9c6b9111fd53034a602c192a04e082ed	@107	7.9894	8.7300	69.7500	BUY

In [3]:
```
def summarize_df(df: pd.DataFrame, name: str) -> pd.DataFrame:
    missing = df.isna().sum().sort_values(ascending=False)
    missing = missing[missing > 0]
    duplicates = int(df.duplicated().sum())
    summary = pd.DataFrame({
        "rows": [df.shape[0]],
        "cols": [df.shape[1]],
        "duplicate_rows": [duplicates],
    }, index=[name])
    if not missing.empty:
        display(pd.DataFrame({"missing_count": missing, "missing_pct": (missing / l
    return summary
```

```
qa = pd.concat([
    summarize_df(sentiment_df, "sentiment"),
    summarize_df(trades_df, "trades"),
    summarize_df(merged_df, "merged"),
])
qa
```

Out[3]:

	rows	cols	duplicate_rows
sentiment	2644	4	0
trades	211224	17	0
merged	211218	21	0

2) Daily Metrics (Fear vs Greed performance)

Required question: **Does performance differ between Fear vs Greed days?**

Added metrics (per assignment examples):

- Average trade size (USD)
- Long/short ratio

In [4]:
```
analyzer = Analyzer(merged_df)
daily = analyzer.calculate_daily_metrics()
daily.head()
```

Out[4]:

	date	Classification	avg_pnl	win_rate	trade_count	avg_leverage	avg_trade_size_usd
0	2023-05-01	Greed	0.0000	0.0000	3	24.0000	159.0000
1	2023-12-05	Extreme Greed	0.0000	0.0000	9	11.6667	5,556.2033
2	2023-12-14	Greed	-18.6759	0.3636	11	30.1818	10,291.2136
3	2023-12-15	Greed	-12.3160	0.0000	2	7.5000	5,304.9750
4	2023-12-16	Greed	0.0000	0.0000	3	19.0000	5,116.2567

In [5]:
```
# Bucket variants like "Extreme Fear" into Fear, and "Extreme Greed" into Greed
daily = daily.copy()
daily["sentiment_bucket"] = "Other"
daily.loc[daily["Classification"].astype(str).str.contains("fear", case=False, na=F
daily.loc[daily["Classification"].astype(str).str.contains("greed", case=False, na=
```

```
fear_greed = (
    daily[daily["sentiment_bucket"].isin(["Fear", "Greed"])]
    .groupby("sentiment_bucket", as_index=False)
    .agg(
        avg_pnl=("avg_pnl", "mean"),
        win_rate=("win_rate", "mean"),
        avg_trades_per_day=("trade_count", "mean"),
        avg_trade_size_usd=("avg_trade_size_usd", "mean"),
        long_count=("long_count", "sum"),
        short_count=("short_count", "sum"),
        long_share=("long_share", "mean"),
    )
)

fear_greed["long_short_ratio_total"] = np.divide(
    fear_greed["long_count"].to_numpy(dtype="float64"),
    fear_greed["short_count"].to_numpy(dtype="float64"),
    out=np.full(len(fear_greed), np.nan, dtype="float64"),
    where=fear_greed["short_count"].to_numpy(dtype="float64") != 0,
)
fear_greed["long_share_total"] = np.divide(
    fear_greed["long_count"].to_numpy(dtype="float64"),
    (fear_greed["long_count"] + fear_greed["short_count"]).to_numpy(dtype="float64"
    out=np.full(len(fear_greed), np.nan, dtype="float64"),
    where=(fear_greed["long_count"] + fear_greed["short_count"]).to_numpy(dtype="fl
)

fear_greed

fear_greed.to_csv(TABLES_DIR / "fear_vs_greed_summary.csv", index=False)
fear_greed.to_json(TABLES_DIR / "fear_vs_greed_summary.json", orient="records", ind
```

In [6]:
```
fig1 = px.bar(fear_greed, x="sentiment_bucket", y="win_rate", title="Win Rate: Fear
fig1.update_layout(yaxis_tickformat=".0%")
fig1

fig1.write_html(CHARTS_DIR / "win_rate_fear_vs_greed.html")
```

In [7]:
```
fig2 = px.bar(fear_greed, x="sentiment_bucket", y="avg_pnl", title="Avg PnL: Fear v
fig2

fig2.write_html(CHARTS_DIR / "avg_pnl_fear_vs_greed.html")
```

In [8]:
```
fig_size = px.bar(
    fear_greed,
    x="sentiment_bucket",
    y="avg_trade_size_usd",
    title="Average Trade Size (USD): Fear vs Greed",
)
fig_size

fig_size.write_html(CHARTS_DIR / "avg_trade_size_fear_vs_greed.html")
```

In [9]:
```
fig_ls = px.bar(
    fear_greed,
```

```
    x="sentiment_bucket",
    y="long_short_ratio_total",
    title="Long/Short Ratio (Total): Fear vs Greed",
)
fig_ls

fig_ls.write_html(CHARTS_DIR / "long_short_ratio_fear_vs_greed.html")
```

3) Behavior Changes by Sentiment

Required question: **Do traders change behavior based on sentiment?**

Suggested metrics:

- trade frequency (# trades per day)
- leverage distribution
- long/short ratio
- position sizes (USD)

In [10]:
```
# Leverage distribution (note: may be synthesized if missing in raw trades data)
if "leverage" in merged_df.columns:
    lev = pd.to_numeric(merged_df["leverage"], errors="coerce").dropna()
    fig3 = px.histogram(lev.to_frame(name="leverage"), x="leverage", title="Leverag
    fig3
else:
    print("No leverage column present.")
```

4) Trader Segmentation (2–3 segments)

Required: identify segments such as high vs low leverage, frequent vs infrequent, consistent winners, etc.

In [11]:
```
try:
    high_lev, low_lev = analyzer.segment_traders()
    print("High leverage traders:", high_lev["account"].nunique())
    print("Low leverage traders:", low_lev["account"].nunique())
    display(high_lev.head())
    display(low_lev.head())
except Exception as e:
    print("Segmentation failed:", e)
```

```
High leverage traders: 8
Low leverage traders: 8
```

	account	avg_leverage	total_pnl
0	0x271b280974205ca63b716753467d5a371de622ab	24.1113	-70,436.1913
1	0x420ab45e0bd8863569a5efbb9c05d91f40624641	25.2872	199,505.5927
2	0x6d6a4b953f202f8df5bed40692e7fd865318264a	24.4821	108,731.2168
3	0x72743ae2822edd658c0c50608fd7c5c501b2afbd	25.6101	429,355.5659
4	0x8381e6d82f1affd39a336e143e081ef7620a3b7f	24.7394	65,513.6579

	account	avg_leverage	total_pnl
0	0x2c229d22b100a7beb69122eed721cee9b24011dd	23.1562	168,658.0050
1	0x39cef799f8b69da1995852eea189df24eb5cae3c	23.3611	14,456.9193
2	0x3f9a0aadc7f04a7c9d75dc1b5a6ddd6e36486cf6	22.5120	53,496.2472
3	0x4f93fead39b70a1824f981a54d4e55b278e9f760	23.3354	308,975.8697
4	0x72c6a4624e1dffa724e6d00d64ceae698af892a0	23.2802	360,539.5100

In [12]:
```
# Profitability segment: Consistent Winners vs Net Losers
winners, losers = analyzer.segment_profitability()
print("Consistent Winners:", winners["account"].nunique())
print("Net Losers:", losers["account"].nunique())

display(winners.head())
display(losers.head())

profitability_counts = pd.DataFrame({
    "segment": ["Consistent Winners", "Net Losers"],
    "traders": [winners["account"].nunique(), losers["account"].nunique()],
})
profitability_counts.to_csv(TABLES_DIR / "profitability_segment_counts.csv", index=
profitability_counts.to_json(TABLES_DIR / "profitability_segment_counts.json", orie
```

```
Consistent Winners: 29
Net Losers: 3
```

	account	total_pnl
0	0x083384f897ee0f19899168e3b1bec365f52a9012	1,600,229.8200
1	0x23e7a7f8d14b550961925fbfdaa92f5d195ba5bd	47,885.3205
2	0x28736f43f1e871e6aa8b1148d38d4994275d72c4	132,464.8146
3	0x2c229d22b100a7beb69122eed721cee9b24011dd	168,658.0050
4	0x39cef799f8b69da1995852eea189df24eb5cae3c	14,456.9193

	account	total_pnl
0	0x271b280974205ca63b716753467d5a371de622ab	-70,436.1913
1	0x3998f134d6aaa2b6a5f723806d00fd2bbbbce891	-31,203.6000
2	0x8170715b3b381dffb7062c0298972d4727a0a63b	-167,621.1248

In [13]:
```
# Activity segment: High Frequency (>5 trades/day) vs Low Frequency
high_freq, low_freq = analyzer.segment_activity(trades_per_day_threshold=5.0)
print("High Frequency traders (>5/day):", high_freq["account"].nunique())
print("Low Frequency traders (<=5/day):", low_freq["account"].nunique())

display(high_freq.head())
display(low_freq.head())

activity_counts = pd.DataFrame({
    "segment": ["High Frequency (>5/day)", "Low Frequency (<=5/day)"],
    "traders": [high_freq["account"].nunique(), low_freq["account"].nunique()],
})
activity_counts.to_csv(TABLES_DIR / "activity_segment_counts.csv", index=False)
activity_counts.to_json(TABLES_DIR / "activity_segment_counts.json", orient="record
```

```
High Frequency traders (>5/day): 32
Low Frequency traders (<=5/day): 0
```

	account	avg_trades_per_day	total_trades	active_da
0	0x083384f897ee0f19899168e3b1bec365f52a9012	159.0833	3818	2
1	0x23e7a7f8d14b550961925fbfdaa92f5d195ba5bd	140.0000	7280	5
2	0x271b280974205ca63b716753467d5a371de622ab	317.4167	3809	1
3	0x28736f43f1e871e6aa8b1148d38d4994275d72c4	82.1667	13311	16
4	0x2c229d22b100a7beb69122eed721cee9b24011dd	46.9420	3239	6

account	avg_trades_per_day	total_trades	active_days

5) Insights + Strategy Recommendations (write-up)

Evidence Artifacts

- Summary table: `output/tables/fear_vs_greed_summary.json`
- Segment counts: `output/tables/segment_counts.json`
- Charts (interactive HTML): `output/charts/` (Win Rate, Avg PnL, Avg Trade Size, Long/Short Ratio, Segments)
- Bonus dashboard (interactive): `streamlit run app.py`

Key Insights (data-driven)

1. **Performance is stronger on Greed days.** Day-weighted averages show Greed days with higher win rate (**~38.5%**) and higher average PnL (**~45.85**) compared to Fear days (**~32.9%**, **~32.23**). Evidence: `output/charts/win_rate_fear_vs_greed.html`, `output/charts/avg_pnl_fear_vs_greed.html`.
2. **Fear triggers higher trading intensity (higher churn).** Average trades/day is materially higher on Fear days (**~792.7**) versus Greed days (**~294.1**), suggesting reactive over-trading under fear without a win-rate improvement. Evidence: `output/tables/fear_vs_greed_summary.json`.
3. **Trade sizing and directional bias shift with sentiment.** Average trade size (USD) is slightly higher on Fear days (**~6.20k**) than Greed (**~5.87k**), and the total long/short ratio is higher on Fear (**~0.98**) than Greed (**~0.89**), implying Greed regimes are relatively more short-leaning in this sample. Evidence: `output/charts/avg_trade_size_fear_vs_greed.html`, `output/charts/long_short_ratio_fear_vs_greed.html`.

Segments (required)

- **Profitability segment:** Consistent Winners (**29 accounts**) vs Net Losers (**3 accounts**). Evidence: `output/tables/segment_counts.json`.
- **Activity segment:** Using the assignment-style threshold (>5 trades/day), all accounts fall into **High Frequency** in this snapshot (Low Frequency = 0). This is still a useful data-quality signal; for richer segmentation, a quantile-based frequency split can be used as a sensitivity check.

Professional Strategy Recommendations

1. **Sentiment-Adjusted Activity Filter (Fear Throttle):** On Fear / Extreme Fear days, cap trade frequency and tighten entry criteria (reduce churn). The evidence shows far higher trades/day on Fear regimes without higher win rate.
2. **Sentiment-Aware Position Sizing + Bias Check:** On Fear days, apply a conservative size cap (or stricter risk budget) since average trade sizes skew higher; on Greed days, actively monitor long/short exposure (Greed shows a lower long/short ratio in this dataset) and avoid building one-sided risk.

Note on Leverage

If leverage is missing from the raw trades export, the pipeline synthesizes leverage tiers for pipeline continuity (and prints a warning). If you obtain a leverage-complete export, re-run the notebook/scripts to quantify true leverage shifts by sentiment.