The information content of StockTwits

Joyce Cahoon

Abstract

blah

Contents

Introduction

As fund managers, we were tasked with constructing a cross-sectional, long-short US equity strategy on Quantopian. There were not many explicit constraints on the specific strategy we utilized, but it must pass the following criteria: (i) trade liquid stocks, (ii) have no more than 5% of capital invested in any one asset, (iii) have no more than 10% net dollar exposure, (iv) achieve mean daily turnover between 5% and 65% over a 63-trading-day rolling window, (v) attain gross leverage between 0.8x and 1.1x, (vi) have low correlation to the market, (vii) have less than 20% exposed to each of the 11 sectors as defined on Quantopian, and (viii) result in positive returns. While the last return criteria was not a constraint we included in our optimization, we did design our algorithm with the rest of the seven criteria in mind (Quantopian 2018).

The algorithm I eventually submitted was one based purely on the information content from StockTwits; more specifically, the 5-day moving average of the bull minus bear intensity associated with each tradable stock. I rely on this metric to assign each stock's weight in our portfolio, subject to certain constraints so that our algorithm passes `Quantopian`'s criteria to remain in competition. Given the large literature around news and social media sentiment in driving volatility and liquidity in the market, I wanted to examine the power of this unstructured data in driving returns.

My algorithm's performance over the November period resulted in a final rank of 105 out of 261 submissions and a total score of 0.338. Its performance is mediocre at best and can be attributed the fact that the tuning mechanism for my input parameters was based only on the backtest returns. Given the constraints and the length of moving window for our sentiment metric was overfit to a backtest window between 2016 and 2018, my strategy may have experienced bad out-of-sample performance.

Nevertheless, this strategy did perform within the top 40% of all submissions, which corroborates many findings in literature that social media has some effect on driving prices. The majority of literature suggests extreme sentiment or extreme volume of online media content tends to drive the positive and negative returns, and since the month of November can be characterized by extreme volatility this may explain the high performance of our sentiment strategy. As expected, for December, the peaks in the VIX, however, have subsided given greater clarity around the Trump administration's trade policy and Fed's decision to stop raising rates, and our strategy fell a few ranks to 113 with a score of 0.316 (Gurdus 2018).

Related Work

The first paper that explored the relationship of text—unstructured data—in driving stock prices was Cutler et al. (1988)'s "What moves stock prices?" At the time, the field of computational linguistics was yet sophisticated enough to process large collections of text, so Cutler and his team examined 49 major events discussed in the NY Times Business desk and assessed the market changes associated with each event. Unfortunately, given the number of significant market moves associated with days where nothing significant was released, they were unable to identify any link between market moves and news, possibly stymeing explorations into this realm (Cutler et al. 1988). Fortunately, another empirical study emerged in 1998, which leveraged information on online message boards instead, identifying strong correlations between stocks with high message volume and high market valuation. In fact, Wysocki found message volume forecasted not

only abnormal stock returns the next day, but also trading volume the next day (Wysocki 1998).

Yet, given these results and similar findings from papers like Bagnoli et al. (1999), Antweiler and Frank (2004), corroborating the link between prices and social chatter, a number of studies find the opposite. Tumarkin and Whitelaw (2001) found no relationship for equities in the tech sector and the message activity on RagingBull.com. Das and Chen (2007) developed a classifier for investor sentiment from message board text, yet were unable to to demonstrate its ability to forecast returns. Dewally (2003) found recommendations exchanged on sites like `misc.invest.stocks` and `alt.invest.penny-stocks` had no effect on the return characteristics of the stocks under discussion.

Despite these mixed claims, newer studies arose reporting the opposite. In fact, the seminar paper by Tetlock (2007) concluded high negative sentiment in the Wall Street Journal predicted temporary downward price pressure, echoing the thoughts expressed in Wysocki (1998) that extreme news affects market movement. Additional works that build on Tetlock's use of news as a fundamental factor is Mitra et al. (2008), Leinweber and Sisk (2011) and Fang and Peress (2009), each providing evidence for the predictive value in social chatter. In this class, we were also directed to the paper by Agrawal et al. (2019), which provides a great overview of literature that further promotes the use of Twitter and StockTwits to predict stock volatility, liquidity and return.

Our Reliance on StockTwits

Key to my strategy is the output from the `bull_minus_bear` API call as it is this signal that is fed into the optimization function and this result that determines the order size of each asset. While there is no disclosure on the natural language processing model that calculates the bull and bear intensity of each social media message, it is helpful to look at examples and understand the nature of posts that ultimately determine the power of our sentiment measure.

As shown in Figure 1, there is a lot of noise in StockTwit data as users each utilize the platform for different reasons and do not have to disclose any of their personal stakes in the equities they support. Logically, users that are more influential should have a higher impact in driving overall investor sentiment, and thus driving prices, but there currently is no easily available API call that provides such a weighting. Moreover, these influencers change over time, increasing the difficulty around identifying posts that might contain more information than another. As a result, relying on the aggregate bullish and bearish intensity averaged daily across all the tagged StockTwits messages is the best surrogate we have to understanding community sentiment over a specific stock.

From the free version of StockTwits API, we have access to the following daily metrics for 10,214 liquid US equities:

- *Bull and bear scored messages*: Total number of daily messages that were labeled either "Bullish" or "Bearish" on the StockTwits platform by the user
- *Bull or bear intensity*: Daily score between 0 and 4 aggregated across processed messages. 0 means no bull or bear sentiment was detected by StockTwits proprietary trading algorithm. 4 means very high bull or bear sentiment was detected.

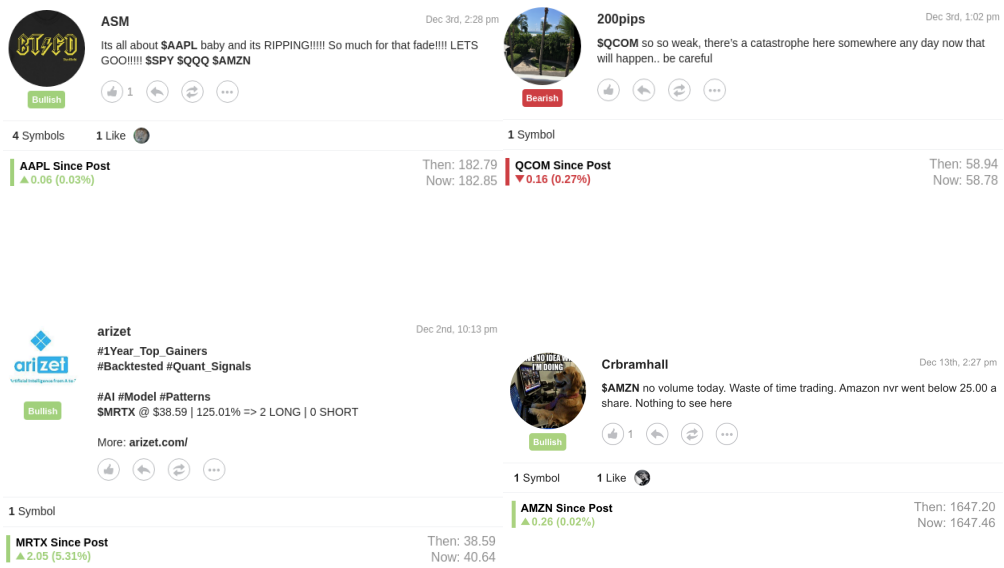

Figure 1: Four examples of messages posted on StockTwits. The bull and bear indicator is provided by the user, and all stock tickers that follow the cashtag is associated with this indicator. The bull and bear intensity is then computed by a proprietary StockTwits algorithm based on the content within each individual post.

- *Total scanned*: Total number of daily messages that appear on StockTwits platform that has at least one cashtag associated with it. The message is included in this count whether it is labeled or unlabeled and can be processed by the StockTwist language algorithm or not.

Since disagreement among message content has been shown to be associated with greater liquidity and volatility, we focus on the metric of `bull_minus_bear` as it is one proxy for disagreement among investor opinion (Antweiler and Frank 2004). Given the overall trend in this metric is not stable as we will see in Figure **??**, a moving average is taken. Eventually, I settle on 5-day moving average as the mean information coefficient (IC) is positive that this point. The IC is often chosen as a means to evaluate whether a factor is predictive of returns as it is more robust to outliers and normality assumptions, allowing one to better tease out whether two series move in concert or not. It is thus calculated from the rank of each data pair in each series as $IC = 1 - 6\sum_i d_i^2/n(n^2-1)$ where d_i represents the difference in ranks. As shown in the top left of Figure 2, prior to 5 trading days, our sentiment signal hurts our returns in that its forecast is negative, but after 5 trading days, we gain some predictive power. This is further corroborated in the top right plot in that returns are highest when the signal is delayed by four days.

Given the IC results, we may want to build greater exposure to this factor as it still benefits us after 10 trading days. We can also leverage the positive effects of this factor by increasing our turnover constraints as it appears it does not hurt our portfolio to keep in there for a longer period of time. Another asset is that our sentiment factor has relatively low exposures throughout, with most of the returns from volatility risk. However, there does appear to be persistent exposure to value and short term reversal, in that it is shifted to the left of zero; and persistent exposure to size and momentum, in that it is shifted to the right of zero. Ideally, we would choose a factor that doesn't display this skew, but the skew does not appear too large. We benefit from the fact that the width of each of our box and whisker plots are not that wide, so our exposures do not vary much over time.

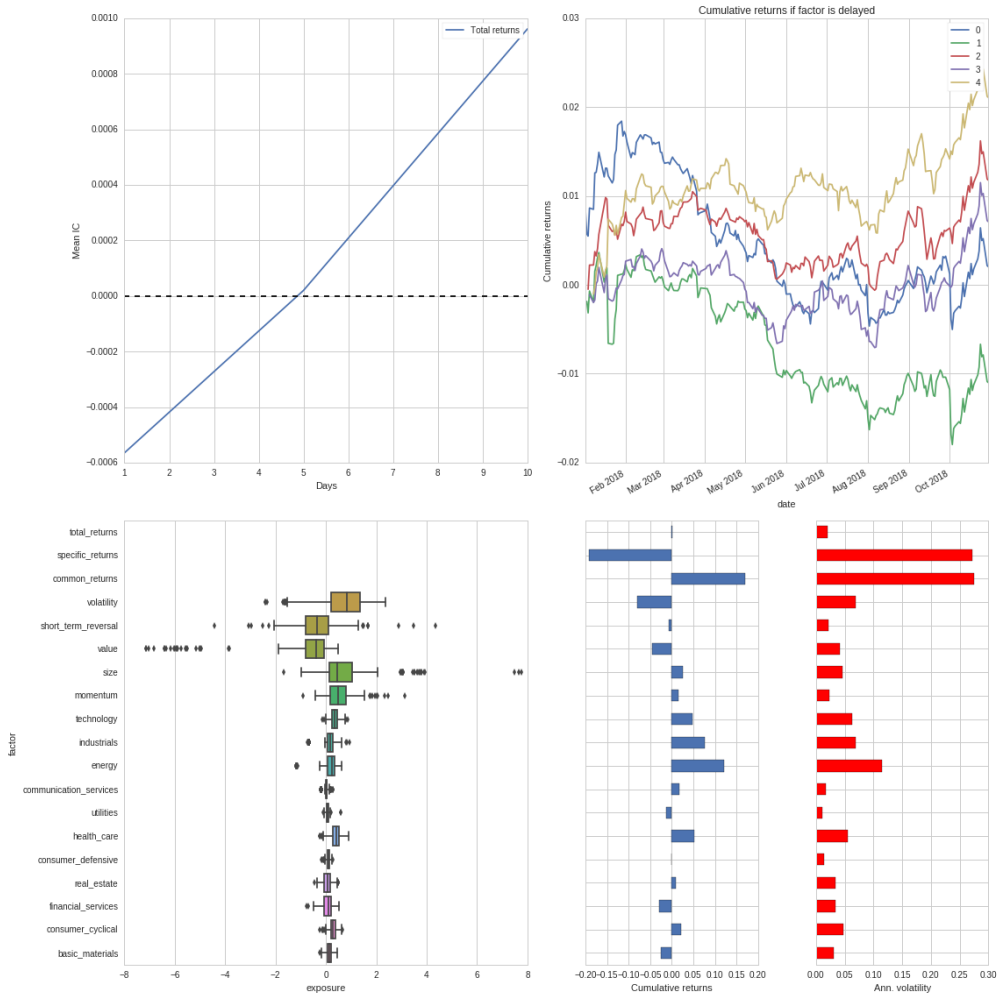

Figure 2: Examination of (top-right) information coefficient averaged over all possible asset returns, (top-left) cumulative returns when factor is delayed, (bottom-left) risk exposures, and (bottom-right) cumulative returns and volatility.

Unfortunately, when we examine our exposures through cumulative returns and volatility, we find that our sentiment factor may not be as strong as we desired. As shown by the left bar graph in Figure 2, most of our returns are obtained from exposure to common risk factors, and, as shown in the right bar graph, are in fact driven by these common risk factors. Ideally, we would like to strip away the contribution from these common risk factors as portfolio performance from this exposure can easily be replicated from ETFs or other easy, cost-effective methods. We can also work to identify asset classes to include in our portfolio that would offset the exposure risks identified above. Nevertheless, this factor analysis was conducted only over the sentiment signal in 2018, so there may be other extended periods prior to 2018 in which our factor over or underperforms.

Trading Strategy

Given the ease of implementation and some good alpha characteristics as delineated above, we chose to rely on the sentiment factor in our trading strategy. Continuous backtesting was run to derive the constraint parameters as outlined:

1. Once a week, we choose a universe of liquid assets from `QTradeableStocksUS` that pass the following filters:
   - It is not trading within 2 days of any earnings announcements as assets are generally more volatile within these dates.
   - It has not been announced as an acquisition target. To further reduce any possible volatility, we avoid acquisition targets as they often pose huge risk to quant strategies.
   - We are able to calculate a 5 day moving average of the bull-minus-bear signal from the `StockTwits` API.
2. Every day, we build an alpha vector for the universe of liquid assets filtered from our step above. The alpha model we use is quite simple: we rank the assets by its bull-to-bear intensity, averaged over the past 5 days as evaluated from StockTwits, and find a set of new portfolio weights that maximizes the sum of each asset's weight times this alpha value. Our objective defined in `MaximizeAlpha` is thus simply a function of this sentiment datastream as we believe this ranking is similar to expected returns of each asset. As a result, our routine effectively goes long on assets with high bullish signal and short on those with a high bearish signal.
3. Once a week, we calculate the portfolio that maximizes the alpha-weighted sum of our position sizes, subject to the following constraints:
   - Our portfolio maintains a gross leverage of, or less than, 1.0x.
   - Our portfolio has no more than 5% in any single asset.
   - Our portfolio does not pass mean daily turnover of 80%.

Our simple strategy can thus be summarized as follows:

$$\max_{\boldsymbol{w}\in\mathbb{R}^n} \boldsymbol{\alpha}^T\boldsymbol{w} \quad \text{subject to} \quad |w_i| \leq 0.05,\ \sum_i |w_i| \leq 1.00,\ \sum_i |w_i - w_{i-1}| \leq 0.80,\ \sum_i w_i = 1$$

where $\boldsymbol{w}$ represents the weights attached to each asset in our optimal portfolio.

Backtest Analysis

Performance

we achieve the following leaderboard results at the end of our one-month trading period from November 1^{st} to November 30^{th}:

Discussion

Section 7 will be the conclusion and discussion. You may revisit the advantage and disadvantage of your strategy and provide some insights for future exploration directions. In the appendix, all your programing codes should be attached.

Appendix

Quantopian Submission

```
# Import Algorithm API functions
from quantopian.algorithm import (
    attach_pipeline,
    pipeline_output,
    order_optimal_portfolio,
)
# Import Optimize API module
import quantopian.optimize as opt
# Pipeline imports
from quantopian.pipeline import Pipeline
from quantopian.pipeline.data.psychsignal import stocktwits
from quantopian.pipeline.factors import SimpleMovingAverage
# Import built-in universe and Risk API method
from quantopian.pipeline.filters import QTradableStocksUS
from quantopian.pipeline.experimental import risk_loading_pipeline
# Get event data
from quantopian.pipeline.factors.eventvestor import (
    BusinessDaysUntilNextEarnings,
    BusinessDaysSincePreviousEarnings,
)
from quantopian.pipeline.filters.eventvestor import IsAnnouncedAcqTarget
from quantopian.pipeline.factors import BusinessDaysSincePreviousEvent
def initialize(context):
    # Constraint parameters
    context.max_leverage = 1.0
    context.max_pos_size = 0.05
    context.max_turnover = 0.8
    # Attach data pipelines
    attach_pipeline(
        make_pipeline(),
        'data_pipe'
```

```
    )
    attach_pipeline(
        risk_loading_pipeline(),
        'risk_pipe'
    )
    # Schedule rebalance function
    schedule_function(
        rebalance,
        date_rules.week_start(),
        time_rules.market_open(),
    )
def before_trading_start(context, data):
    # Get pipeline outputs and
    # store them in context
    context.output = pipeline_output('data_pipe')
    context.risk_factor_betas = pipeline_output('risk_pipe')
# Pipeline definition
def make_pipeline():

    not_near_earnings = ~((BusinessDaysUntilNextEarnings() <= 2) |
      (BusinessDaysSincePreviousEarnings() <= 2))

    not_acq_tar = ~IsAnnouncedAcqTarget()

    universe = (
        QTradableStocksUS()
        & not_near_earnings
        & not_acq_tar
    )

    sentiment_score = SimpleMovingAverage(
        inputs=[stocktwits.bull_minus_bear],
        window_length=5,
        mask=universe
    )
    return Pipeline(
        columns={
            'sentiment_score': sentiment_score,
        },
        screen=sentiment_score.notnull()
    )
def rebalance(context, data):
    # Create MaximizeAlpha objective using
```

```
    # sentiment_score data from pipeline output
    objective = opt.MaximizeAlpha(
      context.output.sentiment_score
    )
    # Create position size constraint
    constrain_pos_size = opt.PositionConcentration.with_equal_bounds(
        -context.max_pos_size,
        context.max_pos_size
    )
    # Constrain target portfolio's leverage
    max_leverage = opt.MaxGrossExposure(context.max_leverage)
    # Constrain portfolio turnover
    max_turnover = opt.MaxTurnover(context.max_turnover)
    # Constrain target portfolio's risk exposure
    factor_risk_constraints = opt.experimental.RiskModelExposure(
        context.risk_factor_betas,
        version=opt.Newest
    )
    # Rebalance portfolio using objective
    # and list of constraints
    order_optimal_portfolio(
        objective=objective,
        constraints=[
            max_leverage,
            constrain_pos_size,
            max_turnover,
            factor_risk_constraints,
        ]
    )
```

Data Exploration

```
# import the free sample of the dataset
from quantopian.interactive.data.psychsignal import stocktwits_free  as dataset
# or if you want to import the full dataset, use:
# from quantopian.interactive.data.psychsignal import stocktwits
# import data operations
from odo import odo
# import other libraries we will use
import pandas as pd
import matplotlib.pyplot as plt
import datetime as dt
```

```
# Filtering for an equity (here SSD)
aapl = dataset[dataset.sid == 11386] # aapl is 24
aapl_df = odo(aapl.sort('asof_date'), pd.DataFrame)
aapl_df2 = aapl_df[aapl_df.asof_date >= dt.date(2016, 9, 30)]
plt.plot(aapl_df2.asof_date, aapl_df2.bull_scored_messages, marker='.', linestyle='None', color='r')
plt.plot(aapl_df2.asof_date, aapl_df2.bull_scored_messages.rolling(window = 5, center = False).mean())
plt.plot(aapl_df2.asof_date, aapl_df2.bear_scored_messages, marker='.', linestyle='None', color='y')
plt.plot(aapl_df2.asof_date, aapl_df2.bear_scored_messages.rolling(window = 5, center = False).mean(), c
plt.xlabel("As Of Date (asof_date)")
plt.ylabel("Count of Scored Messages")
plt.title("Count of Scored Messages for SSD")
plt.legend(["Bull Messages - Single Day", "5 Day Rolling Average for Bull Count", "Bear Messages - Singl
plt.plot(aapl_df2.asof_date, aapl_df2.bull_minus_bear, marker='.', linestyle='None', color='r')
plt.plot(aapl_df2.asof_date, aapl_df2.bull_minus_bear.rolling(window = 5, center = False).mean())
plt.xlabel("As Of Date (asof_date)")
plt.ylabel("Bull Minus Bear Intensity")
plt.title("Bull Minus Bear Intensity for SSD")
plt.legend(["Metric - Single Day", "5 Day Rolling Average"], loc=2)
```

References

Agrawal, S., Azar, P., Lo, A., and Singh, T. (2019), "Practical applications of momentum, mean-reversion, and social media: Evidence from stocktwits and twitter," *Practical Applications*, Institutional Investor Journals Umbrella, 6, 1–4.

Antweiler, W., and Frank, M. (2004), "Is all that talk just noise? The information content of internet stock message boards," *The Journal of finance*, Wiley Online Library, 59, 1259–1294.

Bagnoli, M., Beneish, M., and Watts, S. (1999), "Whisper forecasts of quarterly earnings per share," *Journal of Accounting and Economics*, Elsevier, 28, 27–50.

Cutler, D., Poterba, J., and Summers, L. (1988), "What moves stock prices?" National Bureau of Economic Research.

Das, S., and Chen, M. (2007), "Yahoo! For amazon: Sentiment extraction from small talk on the web," *Management science*, INFORMS, 53, 1375–1388.

Dewally, M. (2003), "Internet investment advice: Investing with a rock of salt," *Financial Analysts Journal*, CFA Institute, 59, 65–77.

Fang, L., and Peress, J. (2009), "Media coverage and the cross-section of stock returns," *The Journal of Finance*, Wiley Online Library, 64, 2023–2052.

Gurdus, E. (2018), "Cramer: Volatility charts suggest now is the time to buy into stocks," *CNBC*, https://www.cnbc.com/2018/11/27/cramer-volatility-charts-suggest-now-is-the-time-to-buy-into-stocks.html.

Leinweber, D., and Sisk, J. (2011), "Event driven trading and the'new news'."

Mitra, L., Mitra, G., and Bartolomeo, D. (2008), "Equity portfolio risk (volatility) estimation using market information and sentiment," *Quantitative Finance*.

Quantopian (2018), https://www.quantopian.com/contest.

Tetlock, P. (2007), "Giving content to investor sentiment: The role of media in the stock market," *The Journal of finance*, Wiley Online Library, 62, 1139–1168.

Tumarkin, R., and Whitelaw, R. (2001), "News or noise? Internet postings and stock prices," *Financial Analysts Journal*, CFA Institute, 57, 41–51.

Wysocki, P. (1998), "Cheap talk on the web: The determinants of postings on stock message boards."