ST790: Quantopian Final Project

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.

Trading Strategy

The backtests we describe below are derived from the following trading algorithm:

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%.

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

	Metric	Our Result	Overall
1	rank	64	-
2	name	Gray Fox	-
3	score	0.502	0.36
4	max_beta_to_spy_126day	0.076	0.14
5	max_cumulative_common_returns	0.009	0.04
6	max_leverage	1.047	1.05
7	max_max_drawdown	0	-0.00
8	max_net_dollar_exposure	0.032	0.04
9	max_total_returns	0.025	0.14
10	min_total_returns	-0.007	-0.02
11	max_turnover	0.905	1.07
12	max_volatility_126day	0.044	0.06

Key to our 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 we do not have exact clarity on the natural language processing engine that calculates the bullish intensity and bearish intensity of a stock, we do know how the `bull_minus_bear` signal arises; traders attached either a bull emoji or a bear emoji to any message they release on the `StockTwits` platform as well as a ticker symbol that identifies which asset is under discussion. While traders attached a clear label to the asset of interest, `StockTwits` also has an in-house proprietary algorithm that processes some of the language in the message to ascribe an intensity level of the bull or bear indicator. Messages across the `StockTwits` platform is thus aggregated to arrive a sentiment score that is a function of subtracting the bearish intensity from the bullish intensity result.

Choice of the sentiment score

In addition to the ease of implementation, we chose to rely on the sentiment factor as it seems to have some good characteristics, namely:

1. **Predictive alpha** We calculated the mean information coefficient using the built-in function in `alphalens` and found our sentiment signal matches the direction of actual asset returns. As shown in the top-left of the figure below, we only get the full exposure of this factor 5 days after this signal becomes available. In fact, prior to 5 trading days, it appears the signal hurts us, in that its forecast is negative. Counterintuitively, after we cross 5 trading days, it appears our sentiment factor gains more predictive power. This is further corroborated in the top-right graph in that returns are highest when the signal is delayed by four days. Given this characteristic, we may want to build greater exposure to this factor since 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. Nevertheless, the mean information coefficient across the time frame displayed still lingers around 0, suggesting the forecasting benefits provided by our sentiment factor may be no better than the results we get from randomly selecting our asset weights.

2. **Low exposures** We quantified the exposures via the `perf_attrib` function in `pyfolio`. As shown in the bottom-left, our sentiment factor appears to have 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 also 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. Yet, this analysis was conducted only over the available sentiment signal in 2018, so there may be other extended periods prior to 2018 in which our factor over or underperforms. Given what we have here, we may want to identify asset classes to include in our portfolio that would offset the exposure risks identified above.

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 (bottom-right), 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.

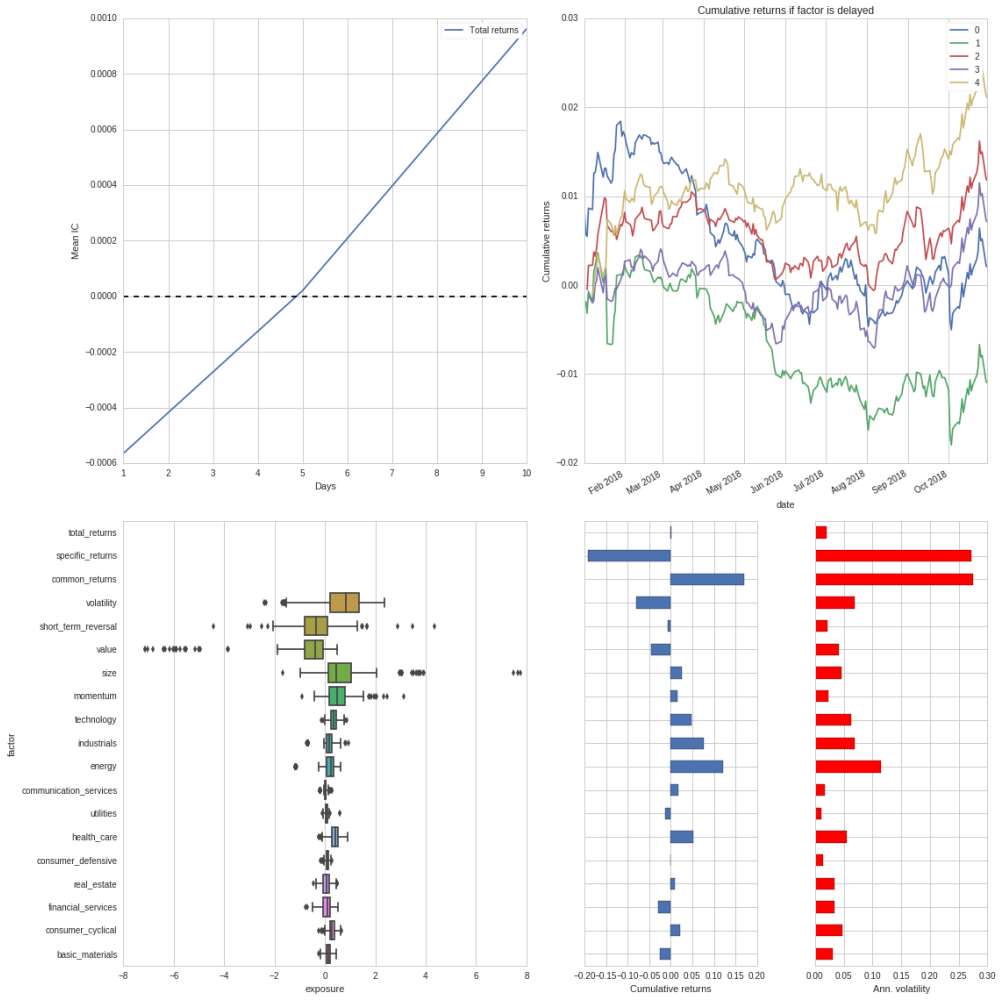

Figure 0.1: Examination of (top-right) information coefficient averaged over all possible asset returns, (top-left) cumulative returns when factor is delayed, (bottom-left) risk exposures, and (bottom-right) cumulative returns and volatility.

Backtesting

Despite the contradictory results from our initial explorations of our sentiment index appeared to indicate, given the extensive literature that exists on sentiment data predicting stock price, we forged ahead and backtested our trading strategy on Quantopian. The backtests for the period between January 2015 and August 2015 are shown below. Attempts were made to obtain backtest results for a longer, more representative, period that reflects the market today, but there were issues obtaining it through the backtesting platform on `Quantopian`.

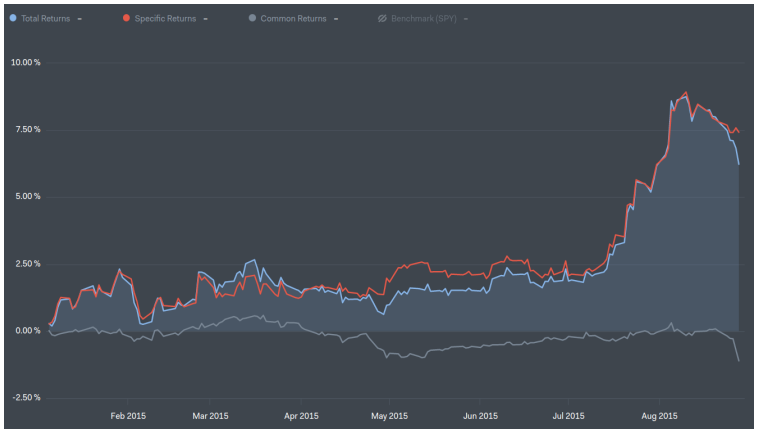

Figure 0.2: Total and specific returns of our trading strategy between January and August 2015.

While our exploratory analyses in Figure 1 suggests most of our returns are from common risk factors, when backtested in the period of 2015, we find that most of our cumulative returns are from specific returns–returns that are attributable to our trading strategy (Mackenzie 2018). Moreover, as desired, our algorithm can be characterized as a momentum strategy, which has time over time demonstrated profitability (Jegadeesh and Titman 2002). In fact, a number of authors, Hong and Stein (2002), has shown behavioral models that explain how momentum profits arise from how investors interpret data. They even go as far as demonstrating the advantages in longer holding periods, resulting from delayed overreaction to news that further push the prices of winners (or pull the prices of losers) from their long-term values as our initial analyses imply.

In Figure 4, our predominantly momentum-based strategy appears to have negative beta to the market. While we should seek to build an algorithm that has low beta in general, this negative beta might actually benefit us during our set trading window of November since the market has been trending down for the entire month of October. As this beta from our simple OLS estimator is particularly sensitive to different time periods, we should also have considered using more sophisticated estimation methods like a shrinkage estimator where we could include some prior belief on how we think the asset beta should behave.

Given the constraints we fed in our `optimize` API call, we are able to achieve the desired leverage, turnover and net dollar exposure in order for our trading strategy to remain on the `Quantopian` platform. As shown in Figure 5, the leverage remains between the 0.8x and 1.1x thresholds. In Figure 6, the absolute net dollar exposure is below the 10% requirement. And in Figure 7, the turnover is below 65% cutoff. Since we pass all the constraints, we submit our sentiment based strategy to the contest.

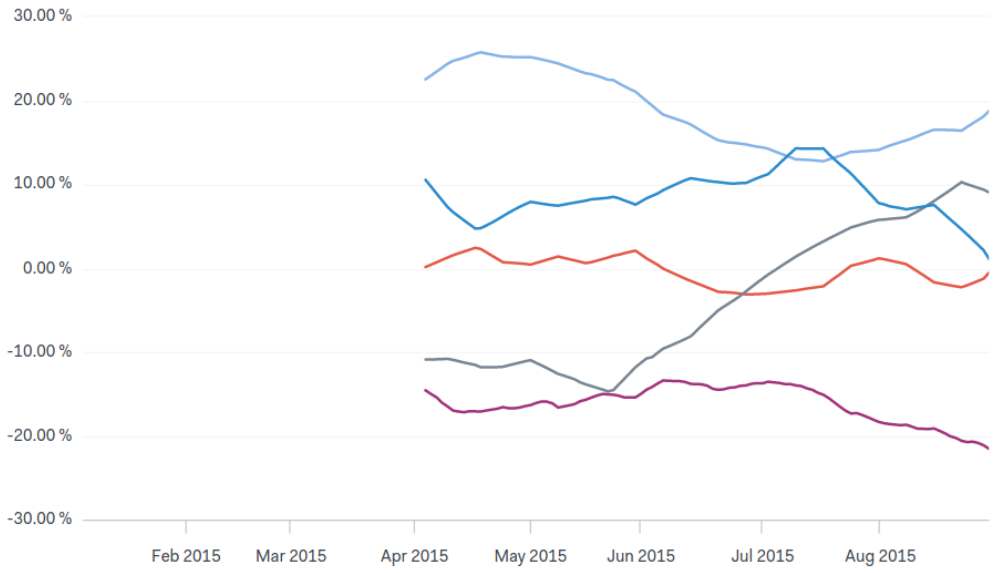

Figure 0.3: Exposure to investing styles assessed from a rolling 63-day mean. The 5 different investing styles presented are: momentum (light blue), volatility (dark blue), size (orange), value (gray) and short term reversal (purple).

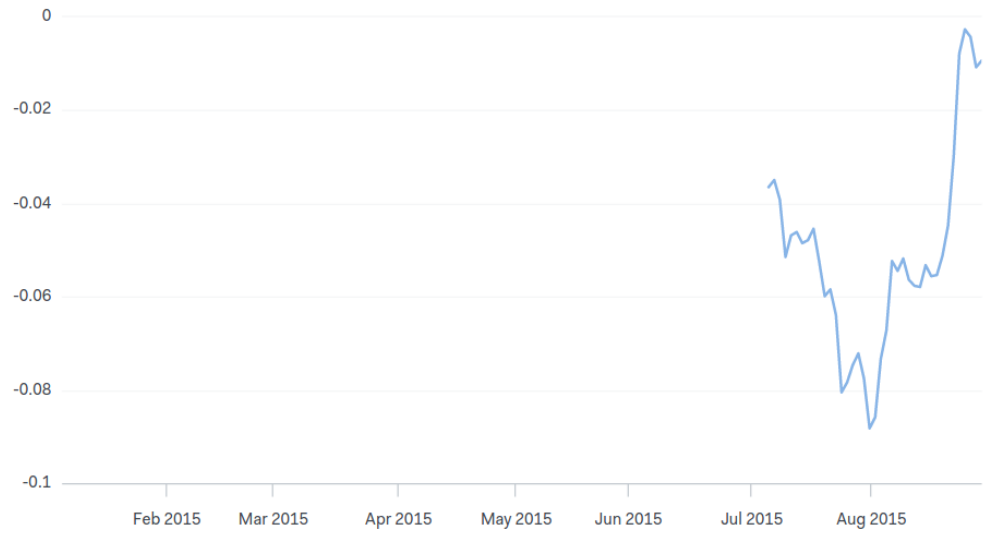

Figure 0.4: Rolling beta to SPY calculated on a 6 month rolling basis. A small window is shown since our backtest range is relatively short, and this metric can only be assessed 6-months after our initial backtest date.

Figure 0.5: Gross leverage calculated as a function of the current portfolio value to its capital base.

Figure 0.6: Net dollar exposure calculated by total value of our strategy's long and short positions.

Figure 0.7: Total turnover rate calculated over a rolling window of 63 days.

Evaluation

The following displays our portfolio performance between Nov 1 and Nov 30:

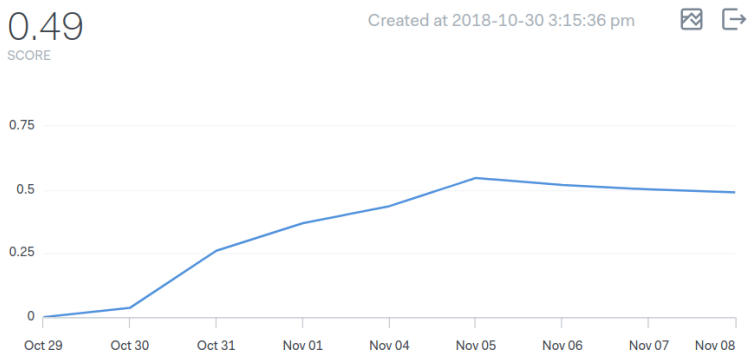

Figure 0.8: Quantopian score during November trading period.

As shown, we performed relatively well in that our strategy ranked 66 out of 266 competitors on the platform. We attribute the success of this simple strategy to its exposure to momentum, low turnover, and negative beta, which has benefitted us in the current market where there has been general downward trend and high volatility in asset prices.

Future Improvement

There are a number of avenues we might consider to improve upon our simplistic approach:

1. *Building a classifier from scratch*: While we relied on what was provided to us by `StockTwits`, we should attempt to build our own classifier from possibly richer sources of messages (e.g., Weibo, Twitter) that can provide greater clarity on how the bull and bear signals arise. With a number of public posts available on how `StockTwits` data is utilized (Capital 2018; StockTwits 2015), there may be more rigorous models available that capture the bull and bear signal. For example, one company, Social Market Analytics, arrives at a sentiment score by taking the difference between the number of bullish messages and bearish messages of an asset and normalizes it by the total number of messages available (Smith 2017). The problem with just looking at the relative frequency of bull and bear messages on social media is that the data comes with its fair share of noise. Users each have their own agenda in which they might make excessive claims about a particular asset, so more sophisticated metrics should be designed to parse through these online market conversations. A possible avenue includes borrowing concepts from topic modeling to develop finite mixture models that arrive at a more reflective distribution of positive and negative sentiment of an asset, other unsupervised or supervised methods can also be considered, as well as the fact that we can now use non-trading day sentiment data to inform how investors feel about a stock.

2. *Include fundamental factors*: Given the extensive literature on factor modeling (Fan and Yao 2015), we should consider including these time-tested metrics and ratios, like market cap and book-to-price, that have captured the financial characteristics of an asset. We can start with the typical (i) SMB, excess return of small market cap minus big market cap companies, (ii) HML, excess return of companies with high book-to-price against low book-to-price ratios, and (iii) MOM, excess return of companies that overperformed to those that underper-

formed, and study how the inclusion of these factors along with our sentiment index might affect returns. We might even find that some factors are more efficient than others, so swapping them in for our sentiment index might be necessary. Once we understand exactly how much we are exposed to specific factors, given the lack of predictability in the market in general, we should consider beta hedging to avoid any big dependencies on performance on a certain factor. This essentially entails taking the exposure β we have to a factor and shorting it by the proportional value of our total portfolio βM.

3. *More comprehensive exploratory data analysis*: There are basic steps we should have carried out prior to executing our strategy on `Quantopian`. Namely, we should have checked for normality, homoskedasticity, and autocorrelation in our model, as violations results in parameters that often biased, inconsistent and inefficient in the context of regression. If these assumptions hold, then we can actually normalize our factor values, allowing for them to be easily comparable across all assets and its different factors. If the assumptions hold weakly, we might consider transformations or techniques, like Winsorization, that will still allow us to build a robust statistical model for trading.

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,
        ]
    )
```

References

Capital, N. (2018), "Using google trends to predict stocks."

Fan, J., and Yao, Q. (2015), *The elements of financial econometrics*, Science Press.

Hong, L., Harrison, and Stein, J. (2002), "Bad news travels slowly: Size, analyst coverage and profitability of momentum strategies," The Journal of Finance, 15.

Jegadeesh, N., and Titman, S. (2002), "Profitability of momentum strategies: An evaluation of alternative explanations," The Journal of Finance.

Mackenzie, D. (2018), "Difference between specific and total returns."

Smith, C. (2017), "StockTwits sentiment analysis."

StockTwits (2015), "The remarkable relationship between sentiment and your favorite stock's earnings."