Department of AI, Data and Decision Sciences
Bachelor's Degree in Management and Computer Science

Sentiment to Strategy: Leveraging Forums Discussions to Guide Automated Trading Decisions

An experimental study that analyzes whether Reddit crowd sentiment can power profitable and automated long–short stock trading strategies.

Supervisor:

Prof. Alessio Martino

Candidate:

Andrea Contino
Role No. 284471

Academic Year 2024/2025

Abstract

Can the daily chatter of retail investors on Reddit's r/WallStreetBets (WSB) be translated into a systematic equity-trading edge? This thesis investigates whether sentiment distilled from WSB posts alone can power a rules-based long–short strategy that rivals canonical benchmarks on risk-adjusted performance. A Python pipeline harvests a month of subreddit posts, filters genuine ticker mentions, scores each sentence with the FinBERT transformer, and aggregates those scores into daily net sentiment for every stock.
A rules-based portfolio then goes long the most bullish tickers and short the most bearish, rebalancing each day while staying market neutral. Its performance is benchmarked against three major equity indices and against an identical direction-randomized portfolio. The social media driven strategy fails to beat either yardstick, suggesting that WallStreetBets sentiment, when used in isolation, lacks predictive power. Future research should explore extended time horizons, alternative weighting schemes, or hybrid signals that blend social chatter with fundamental data.

Index

Chapter 1: Introduction

1.1 Reddit and WallStreetBets

Reddit is a social network platform launched in 2005 that organizes discussions into user-created forums known as subreddits. Each subreddit focuses on a specific topic, from knitting to astrophysics, and users vote posts up or down, pushing the most popular content to the top. Financial talk lives in several such communities, and the most notorious is r/WallStreetBets (WSB). Founded in 2012, WSB blends humor, memes and aggressive trading ideas, and its thread-like structure lets millions of retail investors react to market moves in near real time. Because every post and comment are publicly accessible through Reddit's API, WallStreetBets offers a uniquely transparent stream of crowd opinion that can be harvested, analyzed and, potentially, translated into trading signals.

1.2 Motivation

Since 2012 the r/WallStreetBets (WSB) subreddit has grown into a 20-million-member forum, where retail traders broadcast memes, trade ideas and market mood in real time. The GameStop short squeeze of January 2021 showed that WSB chatters can move prices before news wires or analyst notes react. At the same time, transformer-based language models such as FinBERT make it cheap to distil raw text into bullish-versus-bearish scores. Together these trends raise a provocative question: could a trading bot that simply "listens" to WallStreetBets each day, buys the stocks with the strongest bullish tone, and sizes positions by that tone, rival the performance of the most famous stock market indices?

1.3 Problem Statement

This thesis examines whether daily sentiment extracted from WallStreetBets alone can power a naive and rules-based equity strategy that competes with three canonical ETFs (S&P 500, Nasdaq-100 and MSCI World) by extracting reliable ticker-level sentiment from noisy WSB slang and sarcasm, translating those scores into position sizes for the *N* most bullish or bearish tickers, and benchmarking the resulting portfolio without look-ahead bias or unrealistic turnover.

1.4 Research Questions and Objectives

Two questions steer the study.

First, if a trading bot selects each trading day the *N* tickers that WallStreetBets rates most bullish and weights them proportionally to net sentiment, does that rule produce stable, interpretable allocations or erratic swings?

Second, when this sentiment-weighted portfolio is walked forward through time, does it outperform the S&P 500, Nasdaq-100 and MSCI World on compound annual growth rate, Sharpe ratio and maximum drawdown?

To address these questions the thesis scrapes daily WSB posts, scores each with FinBERT, rolls the top N sentiment portfolio across several years, and compares its risk-adjusted record with the three benchmarks.

1.5 Contributions and Thesis Roadmap

The work offers four concrete contributions. (1) An open-source pipeline that ingests WallStreetBets posts, filters genuine ticker mentions and assigns FinBERT sentiment scores in near-real-time. (2) A transparent trading rule: pick the *N* tickers with the highest bullish sentiment each day and weight positions by sentiment magnitude. (3) A back-testing framework that aligns

Reddit timestamps with market data and reports bias-free performance metrics. (4) The first direct evidence of whether this minimal “WSB sentiment” strategy can match or beat the flagship equity indices.

The rest of the thesis unfolds as follows. Chapter 2 outlines the end-to-end pipeline. Chapters 3 to 7 delve into data collection, sentiment scoring, data alignment, portfolio construction and testing. Chapter 8 presents results, discussion and conclusions, weighing the promise and limits of sentiment-driven trading from a single online crowd.

Chapter 2: End-to-End Pipeline Overview

2.1 High-Level Architecture and Data Flow

First of all, WallStreetBets submissions posted between 1 May and 31 May 2024 are retrieved retrospectively through the Reddit API (PRAW). The raw JSON payload is saved to disk so that the entire experiment can be reproduced without further calls to Reddit. A parser then sweeps through the corpus exactly once, isolating strings that resemble U.S. ticker symbols, validating each candidate against a reference list of listed equities, and discarding any false positives such as common English words or popular crypto or tech tickers. Every surviving mention is passed to the sentiment engine, where FinBERT assigns probabilities for positive, neutral and negative tone. For each trading day in May these probabilities collapse into a single net-sentiment score per ticker by subtracting the negative mass from the positive mass. In the second phase the strategy module ranks all tickers by that net score, selects the top-N names at day-end, and converts their scores into portfolio weights that sum to one. Then, through the use of Yahoo Finance, it collects price data for the corresponding calendar days that are then fetched in a single request and merged with the sentiment table so that daily returns, drawdowns and risk metrics can be computed in one pass and then compared to the performance of S&P500, Nasdaq-100 and MTSE World.

2.2 Key Technologies and Libraries

The pipeline is written entirely in Python. Data ingestion relies on PRAW (Python Reddit API Wrapper), whose cursor-based pagination makes it simple to exhaust a month's worth of subreddit content in chronological order. Sentiment scoring is performed with FinBERT loaded via the Hugging Face

transformers library. Pandas and NumPy provide tabular manipulation and vectorized arithmetic, while yfinance (Yahoo Finance library) supplies the historical open-high-low-close-volume data. Matplotlib is employed sparingly for sanity-check plots, but the core logic is headless and can run in continuous-integration pipelines.

2.3 Assumptions, Scope and Ethical Considerations

Several simplifying assumptions frame the study. All Reddit timestamps are treated as Eastern Time and are floored to the nearest day because the strategy rebalances only at a daily frequency. The back-test assumes that orders placed at the close of day t execute at the open of day $t + 1$ and that the market is sufficiently liquid to fill those orders at quoted prices; transaction costs and slippage are ignored, consistent with a proof-of-concept thesis rather than a production trading system. On the ethical side, only publicly available posts are analyzed, user handles are hashed before storage, and no raw text is redistributed, keeping the work within Reddit's terms of service and standard research-fair-use norms.

Chapter 3: Reddit Data Acquisition & Pre-processing

3.1 Subreddit Scrape Strategy and API Configuration

The empirical backbone of this thesis is a clean, fully replayable record of everything that was written in r/WallStreetBets during May 2024. All messages were retrieved retrospectively in a single big harvest performed on June 1st, 2025. The retrospective pull removes any possibility of look-ahead bias because the entire month's sentiment is observed only after the price paths are already fixed in history. The acquisition phase opens with a single authenticated PRAW session. From this session a lazy handle to r/WallStreetBets is instantiated: no data reach the kernel until an iterator is consumed, thereby keeping connection overhead to the minimum needed for each request. The notebook then consumes the subreddit.new() generator. This endpoint returns submissions in reverse-chronological order and, subject to Reddit's internal cap of roughly one thousand items, delivers every post that was visible at the moment the search ran. For every submission the code records a concise but complete snapshot: the eight-character post ID, the title and self-text in raw Markdown, a time zone aware datetime derived from the UTC timestamp, and a permalink that can be followed back to the live thread. Each snapshot is appended to an in-memory Python list and once the generator is exhausted the list is cast into a Pandas DataFrame (*raw_df*). Using a DataFrame at this early stage serves two purposes: it offers immediate visual inspection of the harvest inside the notebook interface, and it aligns the raw fields with the vectorized operations that will follow (most notably the regular expression scan for ticker symbols).

3.2 Ticker Detection and Company-Name Mapping

With the raw post table in place, now the task is to spot which securities the crowd is actually talking about. This section stitches together the Reddit text fields, aligns them with an authoritative symbol list, and embeds the resulting ticker tags back into the working DataFrame. The first step concatenates each post's *title* and *self-text* into a single free-text column named *text*. A title often carries the ticker while the body delivers the commentary, so merging the two guarantees that neither source is missed when the scan begins. At the same moment a pared-down copy of the harvest is made, *filtered_reddit*, that keeps only the columns that we are interest in for the following steps: the post ID, the UTC timestamp, the permalink and the newly minted text. Fields such as Reddit score and up-vote ratio are dropped because they play no role in identifying tickers and would only inflate memory usage when more rows are present. Next, an external reference list of equities is pulled in. The CSV file stocks.csv is the most recent S&P 500 constituent roster downloaded from a public data source. Two columns are read: Symbol, holding the exchange-traded ticker, and Name, the full corporate title. From these columns two Python lists emerge, *tickers* and *companies*, and a lowercase mapping is built that links every company name to its ticker. Lowercasing sidesteps the stylistic chaos of WallStreetBets prose, where capitalization is anything but consistent. Armed with this reference, the function *matching_tickers(text)* performs a two-layer search on each Reddit sentence. Layer one is direct: if any raw ticker string appears in the sentence exactly as it is written in the S&P 500 list, that symbol is captured. Layer two is indirect: the code loops through the corporate names, checks for a lowercase match inside the sentence, and, when found, swaps the name for its corresponding ticker code. The function returns all hits for the sentence, and a set conversion moments later removes duplicates that arise when, for example, both "Microsoft" and "MSFT" occur side by side. Calling this function across the entire text column yields a new field tickers that contains,

for every Reddit post, the unique set of S&P 500 symbols explicitly referenced by ticker or implicitly referenced by company name. The DataFrame now marries WallStreetBets chatter with a clean security identifier, positioning the corpus for the sentiment scoring stage in Chapter 4.

3.3 Research Questions and Objectives

After the step of mapping Reddit sentences (*text*) to candidate symbols, the notebook devotes a full block of code to purging the most common false positives. Three filters run back-to-back and progressively tighten the definition of what counts as a genuine equity reference.

The first filter, *remove_substring_tickers*, tackles the trivial, but surprisingly frequent, case in which a single-letter or short code is merely a fragment of a longer symbol that appears in the same sentence. The routine begins by deduplicating the list, then sorts it from the longest token to shortest. Walking through that list, it retains a ticker only if it is not found inside any of the longer strings that follow. In practice this wipes out artefacts such as the lone “A” that occurs whenever someone types “AMC”, or “M” inside “MSFT”, without harming legitimate pairs that never co-exist in one post.

Next comes an explicit blacklist pass. The script assembles a wide inventory of everyday words, slang, buzz-terms and single characters that are written in uppercase on Reddit but have nothing to do with listed securities—items like “AI”, “LOL”, “CPU”, the entire alphabet A–Z, and short verbs such as “GO” or “BUY”. This list lives in the variable *ambiguous_tickers*. The companion function *remove_ambiguous_tickers* converts the ticker set to lowercase, subtracts anything that appears in the blacklist, and hands back the survivors. Because the list is created directly in the notebook, its contents are visible and auditable, resulting in no hidden heuristics outcome steers.

Even after blacklisting, option strings and elongated memes can still sneak through. The final guard *keep_standalone_tickers*, revisits the original Reddit sentence and applies a precise regular expression to each remaining candidate. The pattern insists on a non-alphanumeric boundary on both sides of the symbol while allowing an optional leading dollar sign, e.g., \$?NVDA. Tokens embedded in longer alphanumeric ("AMD240621C105") or padded with extra letters ("GOOOOG") therefore fail the test and are dropped. By running row-wise the function preserves context—"SO" is legitimate in a sentence that mentions the Southern Company alone, but not if it is buried inside a word. Once the three filters have finished, the notebook deletes any row whose tickers field has become empty. What remains is a pared-down DataFrame in which every post cites at least one symbol that (i) is not a substring of another symbol in the same sentence, (ii) does not belong to a hard-coded list of everyday uppercase words or memes, and (iii) appears as an independent lexical unit rather than as part of an option code or elongated cheer. This refined table supplies the high-confidence links between WallStreetBets chatter and concrete equities that the sentiment model will score in the next chapter.

Chapter 4: Sentiment Analysis with FinBERT

4.1 Sentiment Analysis in Financial Text

Sentiment analysis (also called opinion mining) is the automatic inference of an author's attitude, positive, negative, or neutral, from written natural language. Traditional approaches relied on curated lexicons and logistic-regression models built on bag-of-words counts, but two features of market commentary, particularly on WSB, complicate those tools:

- Domain vocabulary: Words such as beat, crush or long carry meanings in finance that differ sharply from everyday usage.
- High-context sarcasm and slang: "Diamond hands" or "Paper hands" reads bullish or bearish only to those steeped in with slang financial culture.

Transformer language models, which are pre-trained on billions of tokens and then fine-tuned on task-specific labels, have displaced earlier methods because they learn contextual cues rather than memorizing word lists. Because transformers "pay attention" to the context around every word, the model can tell that "beat" in "beat earnings" is good news, while "beat up" means something is doing badly. In finance research, these sentiment scores serve as a quick stand-in for what retail traders expect: aggregating them over thousands of Reddit posts lets us spot changes in crowd mood before they show up in trading volume or official earnings forecasts.

In the present pipeline every Reddit sentence that mentions a ticker is mapped to a continuous sentiment score. Positive probability mass is treated as bullish pressure, negative mass as bearish, and the difference between positive and negative becomes the daily net sentiment that drives portfolio weights.

4.2 FinBERT: Domain-Specific Transformer

FinBERT is a three-class sentiment classifier derived from the original BERT base architecture and further pre-trained on a large corpus of financial filings, earnings-call transcripts and news headlines. The model keeps BERT's 12 transformer layers, 768-dimensional hidden size and 12-head self-attention, but swaps the generic Google Books and Wikipedia tokens for finance-heavy vocabulary learned during domain adaptation. A single linear classification head (three logits) sits on top of the [CLS] token representation, and then a soft-max converts those logits into the positive, negative and neutral class probabilities. For tokenization and emoji handling FinBERT inherits the WordPiece tokenizer. Unseen tokens, including WallStreetBets emojis, are split into sub-word fragments prefixed with ## that, although sub-optimal, still yield stable embeddings that the fine-tuned layers can interpret. Because the notebook feeds FinBERT one sentence at a time, the 128-token limit is rarely breached, and no truncation changes the meaning of the input.

4.3 From Raw Probabilities to a Single Sentence Score

Every call to FinBERT delivers three probabilities whose sum is exactly 1:

$$(P_{pos}, P_{neg}, P_{neu}).$$

These three values are then collapsed into a single sentiment score by subtracting the negative probability from the positive. This decision looks almost trivial in code, but it encodes several deliberate choices. First, the resulting scale is symmetric and bounded between −1 and +1, so the meaning of every value is immediately clear: +1 denotes a sentence FinBERT judges unequivocally bullish, −1 the mirror-image bearish case, and zero the knife-edge where opposing evidence cancels out. Second, using a difference rather

than an average ensures that purely factual or off-topic statements, which FinBERT classifies as neutral, drop out of the signal instead of diluting it toward the midpoint. In other words, a surge of neutral chatter about a ticker cannot mask a smaller but decisive cluster of bullish remarks. Third, the difference has a natural behavioral interpretation. If one treats the probabilities as the chances that an uninformed reader would decide to buy or sell after reading the sentence, the subtraction becomes the reader's expected trade direction on a ±1 scale. Summing up such expectations across many sentences is more faithful to crowd behavior than majority voting, which discards the intensity of conviction embedded in the probabilities themselves. Technically a helper function, *finbert_score,* performs three steps in a single line of Python: it invokes the Hugging Face pipeline, converts the list-of-dictionaries output into a flat dictionary keyed by the lower-case class labels, and returns the difference *scores["positive"] – scores["negative"]*. Neutral sentences simply score near zero. Posts that are sarcastic, for example if they include eyeroll emojis or similar sarcastic approaches, they are tagged as negative and push the score below zero, while plain factual headlines sit close to zero because they carry no clear bullish or bearish tone. The function is vectorized across the entire text column, so the refined Reddit table now carries a dense numeric field named sentiment beside every row.

4.4 Aggregating Scores Within a Trading Day

Reddit's conversational style format allows the same user, or thousands of users, to mention a ticker many times in a single day, often in wildly different tones. Collapsing all those granular sentences into one number per day therefore demands two linked decisions: how to combine multiple sentences that mention the same ticker, and how to balance direction against raw message volume. To reflect this in the calculation a running sum strategy is adopted:

$$S_{d,t} = \sum_{i \in I_{d,t}} s_i,$$

where *d* is a calendar day, *t* is a ticker symbol, $I_{d,t}$ is the set of all Reddit sentences posted on day *d* that mention ticker *t*, and s_i is the net-polarity score $p_{pos} - p_{neg}$ for sentence *i*. So, for each calendar date it simply adds up the sentence-level scores associated with a given ticker, without dividing by the number of mentions. The choice of a sum rather than an average is intentional. Averaging would treat ten mildly bullish posts as no more informative than one strongly bullish post, yet WallStreetBets attention is itself a scarce resource: the more users talk about a ticker, the more likely an information cascade or coordinated buying spree becomes. By letting volume reinforce direction, the summed score captures both the sign and the heat of the discussion in a single statistic. The procedure lives in *create_sentiment_dict*. As the routine walks through the DataFrame it converts each UTC timestamp to a date, opens (or retrieves) the nested dictionary for that date, and then increments the cumulative sentiment for every ticker mentioned in the row. The result is a plain Python structure whose outer keys are calendar days and whose inner keys are ticker symbols. Because the logic never touches global state, rerunning the cell with the same input always yields byte-identical output, a small but important guard against drift during later experimentation.

4.5 Casting to a Date Ticker Matrix

Time-series testing operates most efficiently on rectangular data structures, so the nested dictionary is immediately promoted to a wide Pandas DataFrame. Each row now represents one trading day, each column a ticker, and every cell holds the summed sentiment that the previous step produced. Combinations that never occur, for example a small-cap that is never once mentioned during May-2024, naturally become zeros when the DataFrame is created. Sorting the index imposes strict chronological order, which is essential for the vectorized

rolling windows that appear in the portfolio-construction chapter. With a daily direction and volume aware sentiment score in hand for every discussed ticker, now the pipeline owns the final ingredient it needs before it can build the portfolios. Chapter 5 takes that table as its starting point and shows how it becomes concrete position sizes in a mock trading account.

Chapter 5: Market Data Retrieval & Portfolio Construction

5.1 Design Principles for Daily Trading

Before any price series is pulled or any test is run, a handful of high-level rules have to be fixed to define what the strategy may and may not do each day. At the heart of the construction logic lies the signals matrix produced in Chapter 4. For every trading date it supplies a net-sentiment score for each ticker. The portfolio treats that matrix as an alpha table, having positive scores as potential longs and negative scores as potential shorts. To rank and select the tickers on every date, the strictly positive entries are isolated, ranked from highest to lowest, and retains a top number (*TOP_N*) of them. It does the mirror image for the strictly negative side, keeping the *TOP_N* most negative names. This approach codifies a simple belief: the strongest bullish chatter should earn an overweight long position, while the most bearish chatter justifies a symmetric short. Then to distribute the amount of capital allocation between positive and negative sentiment, exactly half the book is committed to longs and half to shorts whenever both sides exist. If, on a given day, there are fewer than the *TOP_N* qualified longs or shorts, the half allocated to that side is spread evenly across however many names survive the filter, and the other half remains intact on the opposite side. Each chosen ticker therefore receives an equal-weight slice of its respective 50% bucket, fixing an explicit diversification rule that prevents any single name from dominating the day's risk budget merely, because WallStreetBets is unusually loud about it. So, because the long and short buckets come to +0.5 and −0.5, the net market value is zero and the gross exposure equals 100% of capital. The book is thus approximately market-neutral, so its P&L should stem from stock-selection skill embedded in sentiment rather than from broad index moves.

5.2 Hyperparameters selection

Some of the rules described before are in part encoded in a small block of hyper-parameters and then carried faithfully through the code that follows. First of all, a *TOP_N* is set, meaning that the book will hold at most *TOP_N* long positions drawn from the most bullish scores and *TOP_N* shorts from the most bearish. For the comparison that will be done later, the *TOP_N* was set to 20, which is large enough to diversify away single-name idiosyncrasies yet small enough to focus the portfolio on the strongest sentiment signals, moving the cut-off materially changes the strategy's character. A very small *TOP_N* concentrates risk but may capture sharper sentiment edges, while a very large *TOP_N* dilutes signal strength as progressively weaker opinions enter the trade list. Later in the chapter the context for the choice of using 20 as *TOP_N* will be better demonstrated by comparing the results of WallStreetBets trading strategy with different *TOP_N.* Then also a starting capital for comparison is fixed at $100 000, a round figure that keeps the equity curve in intuitive, retail-sized dollars. Because portfolio weights are expressed as fractions of capital, the absolute starting value merely scales the y-axis of the curve and does not affect percentage returns. Finally, a 2% risk-free rate is embedded as the benchmark for excess-return calculations, matching the average yield on short-term U.S. Treasuries over the 2024–2025 window. This allows subsequent performance metrics such as the Sharpe ratio to reflect a realistic opportunity cost of idle cash.

5.3 Transforming Daily Signals into Tradeable Weights

After the guiding rules and hyper-parameters are set, the dense sentiment matrix as to be turned into an explicit weight for every ticker on every trading day. At the outset the code makes a full copy of the sentiment table, naming it *signals*, and then instantiates an equally shaped *weights* frame filled with zeros.

The empty frame acts like a ledger, only cells that correspond to an active long or short will later be overwritten. Using zeros from the start guarantees that a ticker not selected on a given day will carry no weight, eliminating the need for NaN handling in the return calculation. Then, the core loop iterates over the rows of *signals*, one calendar date at a time. Each row is split into a positive vector and a negative vector, where the positive slice is sorted from largest to smallest and the top *TOP_N* symbols are retained as *long candidates*, and the negative slice is sorted from the most negative upward and its first *TOP_N* entries serve as *short candidates*. If the sentiment distribution is skewed and fewer than *TOP_N* names qualify on one side, the code simply records how many legitimate longs and shorts survive. The capital is then allocated as follows: whenever at least one long exists, exactly half of the notional book, 0.5 in weight space, is divided equally across those names. The same rule applies symmetrically to the short list, except the sign is flipped. Because the divisor is the *count* of survivors, each position inherits an identical fraction of its side's 50% bucket. The choice of equal weighting keeps the implementation transparent and guards against the possibility that a single extreme sentiment score commanders an outsized share of the day's risk, to reflect the trading principals that where set at the beginning of the chapter. The rule also implies that if a side has fewer candidates than *TOP_N*, each surviving ticker automatically receives a larger slice, keeping total exposure constant even on days when chatter is lopsided. Once the loop completes, the *weights* DataFrame carries a full panel of long and short fractions whose rows sum to zero. The object is chronologically aligned with sentiment signals, but it no longer contains any information from prices. Every non-zero entry represents a forward commitment made solely on the basis of yesterday's Reddit mood as codified in Chapter 4. In the next section these weights will marry realized closing prices and translate them into daily profit, loss, and eventually an

equity curve, while continuing to respect the market-neutral posture baked into the construction rules outlined at the start of this chapter.

5.4 Prices, Returns & Equity Curve

With daily weights committed, those theoretical positions must be translated into realized gains or losses. This section orchestrates that translation while respecting two essential constraints: prices must be contemporary with the decision horizon, and the retrieval process must remain efficient enough to rerun many parameters sweeps without throttling Yahoo Finance.

The script begins by capturing the full calendar of trading dates from the weights index and the complete ticker universe from its columns. It then walks the calendar one day at a time. For each date the row of weights is inspected to learn which tickers carry non-zero positions and all others are ignored for that day's download. Limiting the Yahoo Finance query to the active subset dramatically shortens network time, especially on quiet days when sentiment produces only a handful of trades, while guaranteeing that every price fetched corresponds to a symbol that actually matters to the portfolio on that date.

A narrow two-day window, spanning the day in question and the next day, is passed to *yfinance.download* with *auto_adjust=True*. The flag instructs yfinance to pre-adjust for stock splits and dividends, so those corporate actions never contaminate the percentage-return calculation. If the API returns an empty frame, because of a market holiday, a delisting, or a transient network hiccup, the code records a Series of NaNs for that date, preserving the calendar without fabricating data. Each daily price vector is then appended to an in-memory list, which is later concatenated into a two-dimensional prices DataFrame aligned by date and ticker. Daily percentage gains are calculated, and whenever a stock lacks a closing price for a given date, that gap is treated

as a zero return so it doesn't add to or subtract from the portfolio, exactly what would happen when the strategy has no money in that name for that day.
Next, the script re-aligns the weights and signals frames to the surviving price universe, eliminating any mismatch that might have arisen from ticker columns discarded in the cleaning pass. The stage is now set for the core arithmetic: yesterday's committed weights, shifted one row forward, are multiplied by today's realized returns, and the products are summed across tickers to form the gross portfolio return for the day. The explicit *shift()* enforces a strict no-look-ahead discipline, because it forces the strategy to live with the positions it decided on before the new prices were known.
Finally, the daily net return sequence feeds a cumulative product that starts at the chosen $100 000 of initial capital. The resulting strategy equity curve is a day-by-day ledger of how the bot's bankroll would have evolved had it followed the WallStreetBets sentiment signal exactly as designed. That curve is the ultimate yardstick by which the strategy will be judged against the S&P 500, Nasdaq-100 and MSCI World benchmarks in Chapter 6.

5.5 Empirical Choice of *TOP_N*

Selecting how many names to hold from the sentiment ranking is not a cosmetic choice, because it shapes both the character of the portfolio and its risk profile. The discussion so far has treated *TOP_N* as a design knob that trades focus for diversification but, now that the strategy equity curve process is entirely laid out, it can also subject that knob to an out-of-sample experiment. Four versions of the strategy, holding the 5, 10, 15, and 20 strongest long and short names, are run side-by-side on identical capital and over the same May-2024 window.

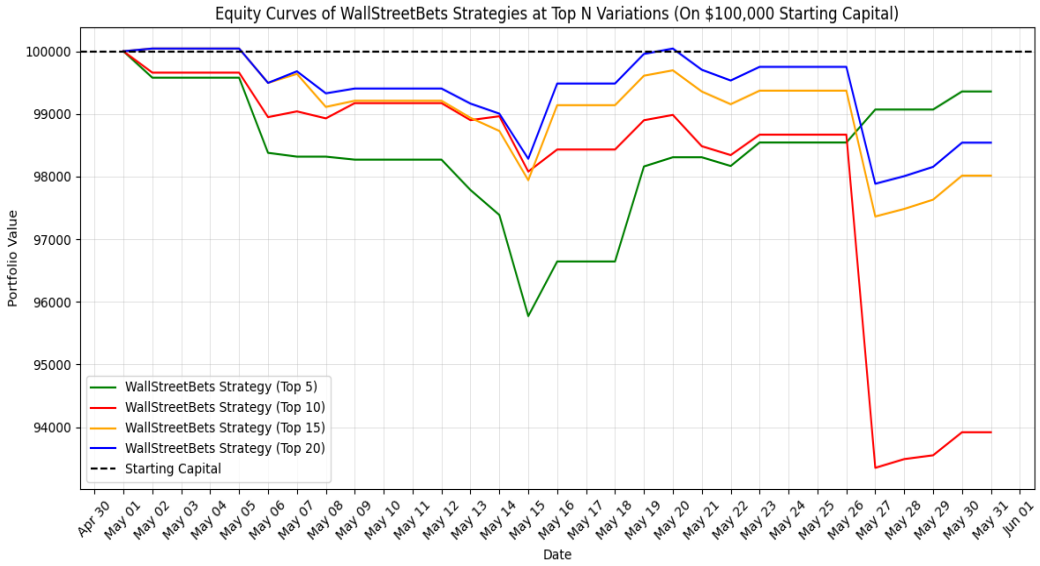

Figure 1: Equity Curves of the WallStreetBets Strategies at TOP_N Variations

The four test runs confirm how strongly concentration alters performance. When the book is limited to just five names per side, every trade carries real headline risk: single earnings miss, or meme-driven collapse can drag the entire portfolio several percentage points in a day. Widening the list to ten names does soften those shocks, yet the curve still shows deep troughs, evidence that idiosyncratic noise has not been diversified away. At fifteen names the drawdowns become shallower and recoveries quicker, indicating that most single-stock risk is now diluted. Pushing the cut-off to twenty offers one further, tangible benefit: the volatility of daily returns falls enough that the curve's path is visibly smoother, even though the strategy is still focused on the clear sentiment extremes. Beyond twenty names, however, the marginal benefit rapidly fades. Each additional position is drawn from nearer the middle of the sentiment ranking, where the signal-to-noise ratio decays: posts are more ambivalent, probability spreads shrink, and the scores oscillate around zero. Allocating capital to these middling opinions not only waters down the aggregate edge but also inflates turnover as tickers shuffle in and out of the tail of the list.

Balancing these considerations, 20 as *TOP_N* emerges as a sweet spot. It harnesses the crowd's strongest convictions, disperses idiosyncratic shocks across enough names to keep drawdowns tolerable, and avoids the churn and dilution that plague wider baskets. For those reasons the subsequent performance comparisons against the S&P 500, Nasdaq-100 and MSCI World will employ the 20-name configuration as the canonical version of the WallStreetBets sentiment strategy.

Chapter 6: Benchmarking the WallStreetBets Strategy

6.1 Comparison Selection

To judge whether WallStreetBets offers any real investing edge, the strategy is stacked against three proven yardsticks. The S&P 500 (GSPC) summarizes large-cap U.S. equities across every major sector and it's the canonical hurdle for active managers of American stocks. The Nasdaq-100 (IXIC) is far more concentrated in technology and growth names, mirroring the tilt one often sees in WallStreetBets discussions: out-performance there would suggest the strategy adds value even in the very corner of the market where retail enthusiasm is most intense. Finally, the MSCI World proxy (URTH) extends the comparison to a global opportunity set, answering the question of whether Reddit-driven stock picking can rival the risk-adjusted returns of a diversified international portfolio. To contextualize these institutional benchmarks, the chapter also introduces a random trader. This placebo strategy is built to mirror the WallStreetBets book in size, leverage, and rebalancing frequency, but it chooses long and short directions by coin-flip rather than by sentiment. If WallStreetBets hype does carry predictive power, its equity curve ought to sit comfortably above this blind benchmark, if it cannot, we gain empirical evidence that the "meme first, fundamentals later" philosophy delivers less than randomness.

6.2 Building the Random-Trader Baseline

The random comparator starts from the same *weights* matrix used by the WallStreetBets strategy, ensuring that both portfolios are exposed to exactly the same tickers on each day. All weights are reset to zero, after which the

algorithm scans the active symbols for a given date, counts them, and assigns half the book to longs and half to shorts, just as the main strategy does. The twist is that each ticker is flipped to the long or short bucket with 50% probability, so the resulting portfolio is market-neutral by construction yet carries no informational content. Daily returns are computed in the same way as before: yesterday's random weights are multiplied by today's realized percentage moves and summed across tickers. These gross returns are chained into an equity curve that begins at the same $100 000 of starting capital as every other test. Because the random trader obeys the identical rebalancing calendar and position-count rules, differences in performance isolate the incremental value of sentiment-driven stock selection rather than artefacts of leverage, timing, or universe choice. In the plots that follow, the WallStreetBets curve will be judged not only against the three passive index benchmarks but also against this statistically neutral baseline, completing a four-way comparison that spans global diversification, technology concentration, broad U.S. exposure, and pure randomness.

6.3 Metrics and Benchmark

A fair assessment demands that every candidate, the WallStreetBets book, blind coin-flip book, and the three passive indices, has to be observed over identical trading days and start with identical capital. The code therefore pulls adjusted close prices for the S&P 500, Nasdaq, and MSCI World across the precise span covered by the sentiment strategy's equity curve. Using Yahoo Finance's *auto-adjust* flag neutralizes splits and dividends at download time, so subsequent return calculations are not distorted by mechanical price jumps. Forward-filling the series tightens the alignment: if U.S. markets close for Memorial Day while London is open, the earlier U.S. close persists into the holiday row, ensuring that the comparison operates on a synchronous calendar

rather than on jagged, exchange-specific dates. Once raw prices are lined up, they are compounded into equity curves and then distilled into three headline metrics, CAGR, Sharpe, and maximum drawdown, chosen because they capture growth, efficiency, and pain respectively:

- Compound Annual Growth Rate (CAGR) normalizes performance to a per-year basis, answering: "If this behavior continued, what annual rate would reproduce the same ending value?". Using CAGR rather than a simple percentage gain prevents the May-2024 snapshot from looking artificially good (or bad) simply because it is only 23 trading days long, because a one-off total return can be misleading over a short window.

- The Sharpe Ratio (ex-risk-free) divides excess return by realized volatility and annualizes the result. A portfolio that climbs slowly but smoothly can out-Sharpe a racier book that lurches up and down, so this metric reveals how *efficiently* each strategy converts risk into reward. Including the risk-free subtraction matters: it ensures we credit only what the portfolio earns *beyond* what an investor could pocket by parking cash in ultrashort Treasuries over 2024–25.

- Maximum Drawdown measures the largest percentage drop from a portfolio's highest value to its subsequent low before a new peak is reached. It captures the deepest capital erosion experienced during the test period and highlights the scale of loss an investor must tolerate in real time. For a portfolio steered by rapidly shifting social-media sentiment, this statistic is vital: it reveals how severe the downswings can be, even if the curve eventually rebounds, and therefore signals the psychological and risk-management stress inherent in the strategy.

Calculating the same trio for the random-direction book seals the evaluation design. Because the random trader mirrors the WallStreetBets book in ticker set, position count, leverage, and rebalancing cadence, any edge shown by the sentiment model must stem from directional choice alone. If the meme-driven system cannot beat this null model, the conclusion is unambiguous: viral enthusiasm on Reddit is no better, and perhaps worse, than guessing. The resulting summary table therefore serves multiple roles.

	Final $	CAGR %	Sharpe	Max DD %
WallStreetBets	98540.849	-0.113	-1.833	-0.022
S&P 500	105511.047	0.547	3.774	-0.018
Nasdaq	105487.904	0.544	3.088	-0.027
MSCI World	107921.912	0.858	3.586	-0.025
Random Strategy	99646.352	-0.028	-4.217	-0.006

Figure 2: Comparison Score Table

Passive indices dominate every metric. All three benchmarks finish above their $100 000 starting stake and post positive CAGRs. MSCI World leads with an annualized 0.86%, followed by the S&P 500 and Nasdaq at roughly 0.55%. Their Sharpe ratios contained (3.1% to 3.8%) show that these gains came with very low day-to-day volatility. Drawdowns remain contained (-1.8% to -2.7%), underscoring why buy-and-hold index funds are hard to beat even over brief horizons. WallStreetBets sentiment fails to create alpha, meaning that does not deliver returns above what widely available and low-cost benchmarks already offer after accounting for risk. The strategy closes the period at $98 541, a loss of about 1.5%. Its CAGR rounds to -0.11 %, making it the only entry other than the random trader with a negative growth rate. The Sharpe ratio (-1.83) confirms that the portfolio not only lost money but did so inefficiently: on average each unit of realized volatility subtracted value rather than added it. Although the worst drawdown (-2.2 %) is in line with the indices, the strategy never recovers the capital it gives up in early May, so the draw remains

unrewarded. Random direction does no worse and, in some respects, better. The coin-flip book finishes at $99 646, trimming just 0.35% of capital. Its CAGR (-0.03%) and drawdown (-0.6%) are both less severe than WallStreetBets. The random Sharpe is a deeply negative -4.21, because variance is similar while returns hover near zero, yet the fact that a mechanical coin toss preserves more capital than “crowd wisdom” underlines the absence of predictive power in meme sentiment. Taken together, the metrics offer a clear verdict. Over this sample the strategy built on WallStreetBets hype neither matches passive exposure to broad equity markets nor surpasses a direction-agnostic baseline. Crowd excitement may be loud and fast, but as a standalone signal it fails to generate positive, risk-adjusted returns. The equity-curve chart reinforces these conclusions at a glance.

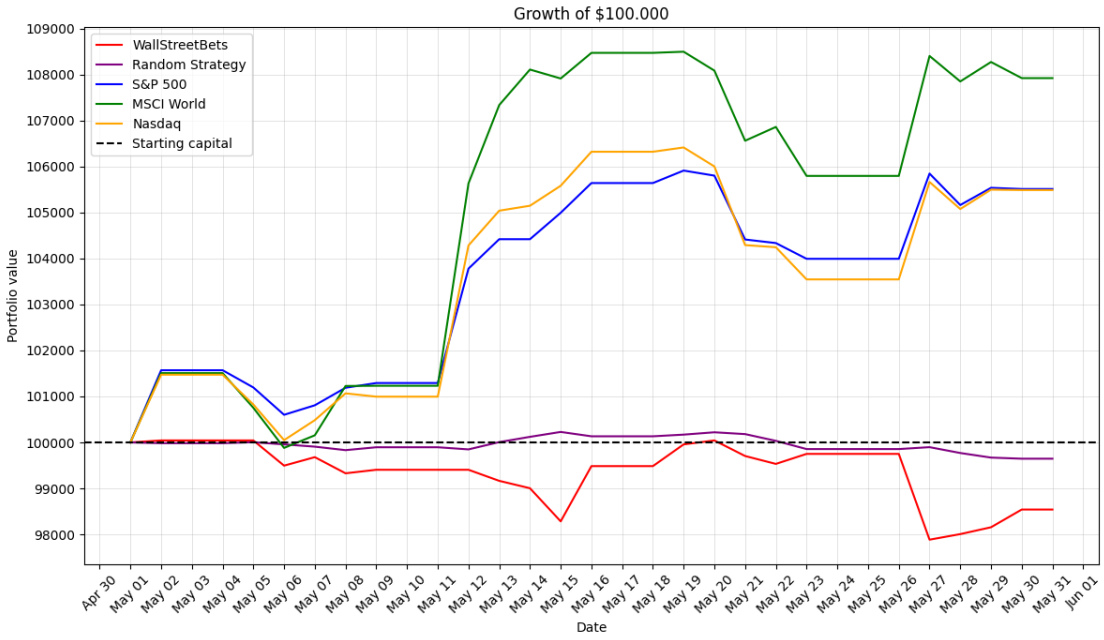

Figure 3: Growth of $100 000 in WallStreetBets Strategy vs. S&P 500, Nasdaq, MSCI World, and Random Trader

The green, blue, and orange lines representing MSCI World, S&P 500, and Nasdaq all trend upward, while the purple coin-flip line hugs the $100 000 baseline with only minor noise. By contrast, the red WallStreetBets curve slips

below break-even early in May and never recovers, ending the month as the clear laggard. The visual confirms what the score table quantifies: sentiment extracted from r/WallStreetBets not only fails to keep pace with mainstream indices but even underperforms a strategy guided entirely by chance.

6.4 Conclusions

The one-month back-test makes a clear statement: sentiment harvested from r/WallStreetBets, when used in isolation, does not translate into an investable edge. A portfolio that bought the tickers praised most loudly on the subreddit and shorted the most maligned finished the experiment with less money than it began, produced a negative annualized growth rate, and exhibited a Sharpe ratio that indicates each unit of risk destroyed rather than created value. Throughout the same period, passive exposure to the S&P 500, Nasdaq and MSCI World rose steadily, while a coin-flip portfolio holding the very same tickers but choosing direction at random, preserved capital more effectively than the WallStreetBets meme-based strategy. These outcomes imply that the crowd's enthusiasm is too volatile and too shallow to compete with simple diversification, let alone sophisticated index construction.

6.5 Next Step

Although the snapshot is conclusive for May 2024, it is not yet definitive for every market regime. Testing the approach over a longer horizon would reveal whether the month examined here was unusually harsh or whether similar under-performance persists. Introducing a short execution delay could show whether the market digests Reddit chatter almost immediately, leaving little room for profit by the next session's open. Blending Reddit scores with fundamental data like earnings surprises, valuation ratios, or options activity,

could filter out low-quality hype and preserve only signals confirmed by independent evidence. Finally, broadening the sentiment feed to include platforms such as Twitter, StockTwits or Discord would help determine whether the weakness lies in WallStreetBets specifically or in social-media opinion more generally. Any of these extensions would deepen the understanding of how, and whether, online crowd emotion can contribute to systematic equity selection.

Bibliography

1. Reddit Inc. (2024). Reddit API Documentation. https://www.reddit.com/dev/api

2. Reddit Inc. (2024). Reddit User and Community Statistics. https://www.reddit.com/press

3. Subreddit Stats. (2025). r/WallStreetBets - Subscriber Count History. https://subredditstats.com/r/wallstreetbets

4. Chague, F., De Losso, R., & Giovannetti, B. (2022). "Retail Trader Attention and the GameStop Short Squeeze." Journal of Financial Economics, 146(2), 486–506. https://doi.org/10.1016/j.jfineco.2022.07.004

5. Boing, W. (2024). *praw 7.7.1—Python Reddit API Wrapper: Documentation*. https://praw.readthedocs.io

6. Yahoo Inc. (2024). *Yahoo Finance Historical Data API Documentation*. https://finance.yahoo.com

7. S&P Dow Jones Indices. (2024). "S&P 500® Methodology." https://www.spglobal.com/spdji

8. Nasdaq Global Indexes. (2024). *Nasdaq-100 Index® Methodology*. https://indexes.nasdaq.com

9. MSCI Inc. (2024). *MSCI Global Investable Market Indexes Methodology*. https://www.msci.com

10. Araci, D. (2019). "FinBERT: Financial Sentiment Analysis with Pre-trained Language Models." arXiv preprint arXiv:1908.10063. arXiv

11. Hugging Face. (2025). *Transformers Library Documentation*. https://huggingface.co/docs/transformers

12. Lin, Y., Ren, X., & Zhang, H. (2022). "Stock Market Reactions to COVID-19: Evidence from Reddit." *Finance Research Letters*, 45, 102137.

13. United States Department of the Treasury. (2025). *Daily Treasury Yield Curve Rates*. https://home.treasury.gov

14. Ringkvist, I. (2024). *yfinance 0.2: Yahoo! Finance for Python—User Guide*. https://github.com/ranaroussi/yfinance

15. Loughran, T., & McDonald, B. (2011). "When Is a Liability Not a Liability? Textual Analysis, Dictionaries, and 10-Ks." *Journal of Finance*, 66(1), 35–65.

16. Trebeschi, G., & Di Febo, R. (2023). *Backtesting Strategies with Python: A Practitioner's Guide*. O'Reilly Media.

17. Pötsch, O., & Kizys, R. (2023). “When Social Sentiment Goes Viral: An Examination of Meme Stocks.” *Review of Behavioural Finance*, 15(4), 511-533.

18. Brock, W., Lakonishok, J., & LeBaron, B. (1992). “Simple Technical Trading Rules and the Stochastic Properties of Stock Returns.” *Journal of Finance*, 47(5), 1731-1764.

19. Kahneman, D. (2011). *Thinking, Fast and Slow*. Farrar, Straus and Giroux.