# Linking DAO Governance Participation to Token Performance: A Statistical Framework

## Introduction

Decentralised autonomous organisations (DAOs) are an organisational form in which decision making is encoded in smart contracts and executed by a distributed community. Because most governance actions are public on the blockchain, researchers have access to rich data on voter turnout, proposal creation and treasury management. A recurring question among practitioners and academics is whether the intensity of community participation is reflected in market valuations: do tokens issued by actively governed DAOs perform differently from those whose proposals attract little attention? Answering this question empirically requires a careful definition of governance participation and a robust statistical framework that is not biased by the idiosyncrasies of individual projects. The present study proposes such a framework using the DeepDAO API as the primary data source and a manual linkage to token price data from Token Terminal.

### Motivation and research question

The core idea of decentralised governance is to allocate decision making power to the community of token holders or contributors. *Web3 governance refers to the decision‑making process for a decentralised system*[[1]](https://www.hiro.so/blog/web3-governance-models-an-introduction-to-the-decision-making-process-in-web3-projects#:~:text=While%20Web3%20governance%20can%20help,can%20adopt%20for%20your%20project). It determines how consensus is formed and the degree to which the community can influence the direction of a project. Community governance can decentralise a project, improve resilience and create a tighter feedback loop with users, ultimately fostering engagement[[2]](https://www.hiro.so/blog/web3-governance-models-an-introduction-to-the-decision-making-process-in-web3-projects#:~:text=Decentralize%20Your%20Project). However, active participation also imposes costs on voters, such as keeping abreast of complex proposals, paying transaction fees (where voting is on‑chain) and exposing one’s position publicly. Voter apathy in many DAOs is well documented: a handful of whales often determine the outcome, and the majority of token holders rarely vote. By measuring and standardising participation metrics across DAOs we can test whether markets discount low engagement or reward active communities.

Our *research question* is therefore: **Does the intensity of governance participation, measured as unique voters per proposal, predict subsequent token performance?** While the price analysis is beyond the scope of this coding task (prices are imported manually), this report focuses on designing an unbiased sample, extracting governance metrics, standardising them and generating a master dataset for regression analysis.

### Contributions

Compared with earlier attempts to rank DAOs by governance activity, this project emphasises four improvements:

1. **Unbiased sample selection.** Previous analyses often relied on convenience samples (e.g., only DAOs with readily available price data) and implicitly excluded failed or inactive projects, leading to survivorship bias. Here we start from a universe of top AUM DAOs retrieved via the top\_aum\_organizations endpoint and systematically exclude organisations only when governance data is unavailable or unusable.
2. **Proper standardisation.** Simply averaging z‑scores across metrics is mathematically invalid because z‑scores are by construction centred at zero; taking their mean yields zero for any group. We therefore compute a single ratio – *unique voters per proposal* – and standardise that series via z‑scores. The z‑score transformation converts a raw observation x into the number of standard deviations it lies away from the mean[[3]](https://www.simplypsychology.org/z-score.html#:~:text=%3E%20A%20z,value%20lies%20below%20the%20mean). Formally, for raw ratio x\_t, the z‑score is defined as z\_t = (x\_t - m) / s, where m and s are the mean and standard deviation of all x values for that DAO[[4]](https://www.simplypsychology.org/z-score.html#:~:text=How%20To%20Calculate).
3. **Monthly aggregation and error logging.** Raw DeepDAO data is reported at daily frequency. We aggregate to monthly totals to smooth daily noise and align with typical investment horizons (e.g., monthly returns). Each month is handled independently; values do *not* accumulate across months. Every failed API call or missing metric is recorded so that the user can see which organisations were omitted.
4. **Reproducible pipeline.** All data extraction and processing steps are encapsulated in a reusable Python module (deepdao\_analysis.py), designed to run in a Jupyter environment. Configuration (date range, number of DAOs, use of votes vs voters) is exposed as parameters.

## Data sources and sample selection

### DeepDAO governance data

DeepDAO collects and normalises governance data from on‑chain and off‑chain voting platforms (Snapshot, Tally, etc.). For each organisation (identified by organizationId) it provides daily timeseries of:

* **Voters.** Number of distinct addresses that cast at least one vote on that day. Unique voters are a proxy for community size because they count each participant once, regardless of how many proposals they vote on.
* **Votes.** Number of votes cast. This counts each ballot and is often weighted by the voter’s token balance. Because large token holders can cast multiple weighted votes per proposal, vote counts can exaggerate participation. We therefore prefer the voters series.
* **Proposals.** Number of governance proposals created. Proposals may be created by core teams or by community members. A high proposal count can inflate vote counts; taking the ratio of voters to proposals adjusts for this.

The study period runs from **1 January 2023** to **31 July 2025**, a 31‑month window. The endpoints used are:

GET /v0.1/organizations/top\_aum\_organizations?limit={n}  
GET /v0.1/timeseries/daily\_dao\_voters/{org\_id}?startDate=…&endDate=…  
GET /v0.1/timeseries/daily\_dao\_proposals/{org\_id}?startDate=…&endDate=…

An API key is required to access these endpoints; the code reads it from an environment variable or command‑line argument. If a request fails (e.g., returns HTTP 403) or returns an empty series, the organisation is logged and removed from the sample. This ensures transparency: omitted DAOs are not silently dropped.

### Universe of organisations

We begin by calling top\_aum\_organizations?limit=2000 to obtain a ranked list of DAOs by assets under management. The limit of 2 000 far exceeds the planned sample so that projects near the boundary of the top 90 can be used as replacements. From this universe we select the first min\_valid\_daos + buffer organisations. In this study min\_valid\_daos = 80 ensures enough observations to form quintiles (top and bottom 10 %) after merging with price data; buffer = 10 provides slack to replace DAOs whose governance data is unavailable. Once 80 valid DAOs are successfully processed, remaining organisations are ignored.

### Handling missing data

Daily series occasionally contain zeros or missing observations. Because the ratio x\_t = voters\_t / proposals\_t is undefined when proposals\_t = 0, months with zero proposals are assigned x\_t = nan. When computing the mean m and standard deviation s of the ratio, these NaNs are excluded. A DAO is retained only if it has at least one month with a defined ratio. Empirically this rule filters out organisations that launched recently or have not held any votes.

### Manual price data

Token prices are not available through the DeepDAO API. Once the governance z‑scores are computed and verified, the user manually downloads monthly price series from Token Terminal and merges them with the output file using organizationId or token symbol. The pipeline therefore exports a clean CSV with organizationId, title, date (monthly period), x (voters per proposal) and z (z‑score). Prices are added outside the scope of this module.

## Methods

### Ratio of unique voters to proposals

Selecting an appropriate participation metric is crucial. Raw vote counts can be distorted by a small number of whales or by token weighting; a single voter who holds many governance tokens can cast thousands of weighted votes. Conversely, unique voter counts measure how many distinct individuals (or addresses) participate. Dividing unique voters by the number of proposals yields the *average community turnout per proposal*. This ratio is more comparable across DAOs with different levels of proposal activity. For example, a small DAO with three proposals and 300 unique voters has the same ratio (100 voters per proposal) as a large DAO with thirty proposals and 3 000 unique voters.

Formally, define for DAO d in month t:

and x\_{d,t} = \mathrm{nan} otherwise. The monthly series is then standardised using z‑scores:

where is the set of months with defined ratios and . This transformation maps each DAO’s participation series onto a standard normal scale. A positive z‑score indicates above‑average voter turnout relative to that DAO’s history, while a negative z‑score indicates below‑average turnout. The z‑score allows comparison across DAOs by removing scale differences: a month with z = 2 is two standard deviations above the DAO’s mean, irrespective of whether that mean is 10 or 1 000 voters per proposal. The definition of a z‑score, “the raw score minus the mean divided by the standard deviation”[[4]](https://www.simplypsychology.org/z-score.html#:~:text=How%20To%20Calculate), underpins this interpretation.

### Monthly aggregation

To align with price data and smooth daily volatility, daily series are aggregated to monthly totals. For voters, the daily counts are summed within each month; for proposals, daily counts are summed similarly. Months without any proposals or voters are automatically represented in the index but yield NaN ratios. The aggregation uses the first day of each month as the timestamp (e.g., 2023-01-01 for January 2023) to facilitate merging with monthly returns later.

### Sample size and weighting

The final dataset aims to include at least 80 DAOs. This number is a trade‑off: it is large enough to form quintile portfolios (top 20 %, bottom 20 %, etc.) yet small enough to permit manual inspection and price collection. Because each DAO contributes multiple observations (one per month), panel regressions can be run with fixed effects to control for time‑invariant DAO characteristics. The current study does not weight observations by treasury size or token supply; each DAO and month contributes equally. Future work could explore weighting by liquidity or market capitalisation.

### Error handling and logging

Data collection is subject to network failures, missing series and endpoint restrictions. The pipeline records every failure with the organisation identifier and error message. Common failure modes include HTTP 403 (insufficient permissions), HTTP 404 (unknown organisation) and empty response (no governance data). When a DAO fails, it is omitted and replaced by the next DAO in the AUM ranking until the minimum number of valid DAOs is reached. This ensures that the final sample is as unbiased as possible given data availability.

## Implementation

### Overview of the deepdao\_analysis module

The analysis pipeline is encapsulated in the Python module deepdao\_analysis.py. The design follows functional decomposition: each step of the workflow is implemented as a function that can be tested in isolation. The key functions are summarised below.

#### API key retrieval

The \_get\_api\_key() helper reads the API key from a function argument or from the DEEPDAO\_API\_KEY environment variable. If no key is available, it raises a ValueError. Centralising this logic ensures that all subsequent functions have consistent access to the key.

#### Getting the universe of DAOs

get\_top\_daos(api\_key, limit) calls the /organizations/top\_aum\_organizations endpoint and returns a pandas DataFrame with organizationId, title and aumUsd. The DataFrame is sorted in descending order of assets under management. If the HTTP status code is not 200, the function logs the error and raises an exception. Exposing the limit parameter allows the user to retrieve more organisations as a buffer for failures.

#### Fetching daily timeseries

fetch\_timeseries(api\_key, org\_id, metric, start\_date, end\_date) retrieves daily data for a single metric ("voters", "votes" or "proposals"). The function constructs the appropriate endpoint path (daily\_dao\_voters, daily\_dao\_votes or daily\_dao\_proposals), performs up to three retries on transient errors (HTTP 429 or 500) and parses the JSON response into a DataFrame with date and value columns. By handling retries internally, the caller does not need to manage back‑off logic. Non‑recoverable errors such as HTTP 403 are logged and raised immediately.

#### Aggregating and standardising

aggregate\_monthly(df) resamples a daily series to monthly frequency ("MS" stands for month‑start) and returns a Series of monthly sums. compute\_z\_scores(series) takes a monthly ratio series and returns a Series of z‑scores. NaN values are ignored when computing the mean and standard deviation. If the variance is zero (i.e., all valid months have identical ratios), z‑scores are set to zero instead of dividing by zero. This avoids infinite values and reflects the fact that all observations are exactly at the mean.

#### Processing a single DAO

process\_dao(api\_key, org, start\_date, end\_date, use\_voters) ties together the previous functions. It fetches daily voters (or votes) and proposals, aggregates them to monthly totals, aligns the index to include all months in the date range, computes the ratio and the z‑scores, and returns a DataFrame with organizationId, title, date, x and z. If any step fails or the DAO has no months with proposals, the function returns an empty DataFrame and an error message.

#### Processing multiple DAOs

process\_daos(api\_key, daos\_df, start\_date, end\_date, min\_valid\_daos, buffer, use\_voters) iterates through the list of candidate DAOs in order of decreasing AUM. For each organisation it calls process\_dao(). If the DAO is processed successfully, its results are appended; otherwise the organisation is logged as failed. The loop stops when the desired number of valid DAOs (min\_valid\_daos) is reached or when all candidates in daos\_df.head(min\_valid\_daos + buffer) have been tried. This mechanism automatically compensates for omissions by extending the sample to the next DAOs in the ranking.

#### Saving the master file

save\_master\_csv(df, filename) writes the combined results to a CSV file. It ensures that the directory exists and warns if the DataFrame is empty. The output file is encoded in UTF‑8 and can be opened in Excel or further processed in Python or R.

### Reproducibility and extensibility

The module is self‑contained and relies only on widely used dependencies (requests, pandas, numpy). Users can run it as a script by executing:

python deepdao\_analysis.py --api-key YOUR\_API\_KEY --start-date 2023-01-01 \  
 --end-date 2025-07-31 --min-valid-daos 80 --buffer 10 --output zscores.csv

Alternatively, they can import the functions into a Jupyter notebook to perform exploratory analysis, change parameters on the fly or experiment with alternative metrics (e.g., using votes instead of voters). The code is thoroughly documented so that future researchers can adapt it to other date ranges or governance metrics.

## Preliminary findings and result variants

At the time of writing this report, API keys are not available in the analysis environment, so actual governance data could not be retrieved. Nevertheless, the pipeline has been extensively tested with synthetic data to ensure that monthly aggregation and z‑score computations behave as expected. Once an API key is provided and the code is executed, several descriptive and inferential analyses can be performed:

1. **Distribution of voters per proposal (raw and z‑scored).** For each DAO, summarise the mean, median and volatility of the ratio x\_{d,t}. Visualising the distribution of z‑scores across DAOs can reveal whether voter participation exhibits heavy tails (i.e., a few exceptionally active communities).
2. **Cross‑sectional ranking.** Compute the average z‑score for each DAO across the sample period. Because z‑scores are centred at zero within each DAO, averaging them yields a measure of persistent outperformance or underperformance relative to the DAO’s own history. Rank DAOs by their average z‑score and report the top and bottom deciles. This ranking will guide manual price data collection.
3. **Trend analysis.** Examine whether z‑scores trend upward or downward over time for individual DAOs. A sustained decline might indicate governance fatigue, while a consistent increase might suggest improving engagement.
4. **Panel regression.** After merging with monthly token returns, fit panel regressions of the form:

where r\_{d,t+1} is the future return, z\_{d,t} is the governance z‑score, are DAO fixed effects and are month fixed effects. A positive and significant would support the hypothesis that higher participation predicts stronger subsequent price performance. Alternatively, one could regress contemporaneous returns on z\_{d,t} or examine quintile portfolios formed on z‑scores.

1. **Robustness checks.** Repeat the analysis using votes instead of voters in the numerator, or using the ratio of *voters to treasury size* to account for economic stakes. Other potential normalisations include dividing by the number of token holders or by average gas costs, though the latter would require additional data.

Each of these analyses can be implemented in a few lines of code using the exported master CSV and statistical packages such as statsmodels or linearmodels for panel regressions.

## Discussion and limitations

### Rationale for using unique voters

Unique voters count each participant once, regardless of how many proposals they vote on or the weight of their tokens. This aligns better with the democratic ethos of DAOs and avoids conflating participation with wealth. While vote counts are informative, they mix intensity (how many proposals are voted on) with concentration of voting power. A future improvement could combine both metrics via principal component analysis or factor analysis, but care must be taken to avoid averaging z‑scores across metrics, which would collapse the distribution to zero by definition.

### Survivorship bias and sample representativeness

Selecting DAOs solely on the basis of AUM may bias the sample toward large and financially successful projects. Smaller DAOs with active communities but modest treasuries might be excluded. However, the choice of AUM ranking is pragmatic: it is readily available via the API, and it provides a consistent ordering from which to draw a buffered sample. Inactive or failed DAOs that still have AUM on DeepDAO may remain in the universe; if they have no governance data they will be filtered out. To mitigate survivorship bias in future studies, researchers could construct a custom list of DAOs from archival data or include those with zero treasury but active voting on Snapshot.

### External validity and token performance

This report does not estimate the relation between participation and token performance because price data is imported manually. Even after merging, causality may be hard to establish: high participation could lead to better project outcomes (causing higher returns), or conversely rising prices could attract more voters. Instrumental variables (e.g., governance shocks such as security incidents) or Granger causality tests could help disentangle directionality.

### Limitations of the DeepDAO API

While DeepDAO aggregates governance data across multiple platforms, it may not capture off‑chain discussions, informal voting or delegate communications that occur outside of formal proposals. Moreover, top‑level metrics such as voters and proposals do not account for quorum thresholds, vote weights or controversialness of proposals. A proposal that passes unanimously with few voters may have different economic implications from one that narrowly passes with thousands of voters. Incorporating richer metadata (e.g., proposal category, passing margin) would yield a more nuanced governance score.

## Conclusion

This report outlines a rigorous methodology for quantifying governance participation across DAOs and preparing the data for subsequent econometric analysis. By extracting daily metrics from DeepDAO, aggregating them to monthly voters‑per‑proposal ratios, standardising via z‑scores and logging all failures, the pipeline yields a clean, reproducible dataset. The choice of unique voters per proposal as the primary variable reflects a desire to capture community engagement rather than raw voting power. The z‑score normalisation, grounded in standard statistical practice[[4]](https://www.simplypsychology.org/z-score.html#:~:text=How%20To%20Calculate), ensures comparability across DAOs. Once an API key is provided and the code executed, researchers can merge the output with token price data and test whether governance participation predicts financial performance. In doing so, they will not only contribute to the empirical literature on DAOs but also provide actionable insights for project founders and delegates seeking to improve community engagement.

[[1]](https://www.hiro.so/blog/web3-governance-models-an-introduction-to-the-decision-making-process-in-web3-projects#:~:text=While%20Web3%20governance%20can%20help,can%20adopt%20for%20your%20project) [[2]](https://www.hiro.so/blog/web3-governance-models-an-introduction-to-the-decision-making-process-in-web3-projects#:~:text=Decentralize%20Your%20Project) Web3 Governance Models: An Introduction to the Decision-Making Process in Web3 Projects

<https://www.hiro.so/blog/web3-governance-models-an-introduction-to-the-decision-making-process-in-web3-projects>

[[3]](https://www.simplypsychology.org/z-score.html#:~:text=%3E%20A%20z,value%20lies%20below%20the%20mean) [[4]](https://www.simplypsychology.org/z-score.html#:~:text=How%20To%20Calculate) Z-Score: Definition, Formula, Calculation & Interpretation

<https://www.simplypsychology.org/z-score.html>