TasteRank Explorer

Technical Appendix

Mathematical Models, Algorithms, and Implementation

Jure Skarabot
April 2026

Table of Contents

A. Mathematical Framework

A.1 Feature Space Definition

Let $V = \{v_1, v_2, \ldots, v_{101}\}$ be the set of 101 grape varieties. Each variety v_i is represented by a profile vector:

$$p_i = (p_{i1}, p_{i2}, \ldots, p_{i13}) \in \mathbb{R}^{13}$$

where each dimension corresponds to one of the 13 SAT tasting dimensions, scored on integer scales as defined in the Data Appendix. The profile matrix $P \in \mathbb{R}^{101 \times 13}$ collects all variety profiles row-wise.

A.2 Cosine Similarity

The pairwise similarity between varieties i and j is computed as:

$$s_{ij} = \cos(\theta_{ij}) = (p_i \cdot p_j) / (\|p_i\|_2 \cdot \|p_j\|_2) = \sum_k p_{ik} p_{jk} / \sqrt{(\sum_k p_{ik}^2)} \cdot \sqrt{(\sum_k p_{jk}^2)}$$

This produces the full similarity matrix $S \in \mathbb{R}^{101 \times 101}$ with $s_{ii} = 1$ and $0 \leq s_{ij} \leq 1$ (since all profile entries are non-negative). The choice of cosine over Euclidean distance is motivated by three considerations: (1) cosine is scale-invariant, measuring profile shape rather than magnitude; (2) it is bounded in [0, 1] for non-negative vectors, providing a natural similarity interpretation; (3) it handles the tannin dimension (which is structurally 0 for most whites) more gracefully than Euclidean distance.

A.3 kNN Graph Construction

The graph $G = (V, E, w)$ is constructed as follows:

Step 1 (Neighbor selection): For each variety v_i, identify the $k = 5$ varieties with highest cosine similarity: $N_k(v_i) = \text{argmax}_k \{s_{ij} : j \neq i\}$.

Step 2 (Symmetrization): Create an undirected edge between v_i and v_j if $v_j \in N_k(v_i)$ OR $v_i \in N_k(v_j)$. The edge weight is $w_{ij} = s_{ij}$.

Step 3 (Maximum-weight union): When both varieties include each other in their top-k, the single edge carries the cosine similarity weight (which is identical by symmetry of cosine).

The resulting graph has $|E| = 341$ edges. The average degree is $2 \cdot 341 / 101 \approx 6.75$, reflecting the symmetrization which can push effective degree above k.

A.4 Weighted Adjacency Matrix

The weighted adjacency matrix $W \in \mathbb{R}^{101 \times 101}$ is defined as:

$$W_{ij} = s_{ij} \text{ if } (v_i, v_j) \in E, \text{ else } W_{ij} = 0$$

W is symmetric ($W = W^T$), non-negative, and sparse (density = 341 / 5050 ≈ 0.0675).

A.5 Eigenvector Centrality (TasteRank)

A.5.1 The Eigenvalue Problem and the Nature of the Definition

The weighted adjacency matrix W is a real, symmetric, non-negative 101 × 101 matrix. By the spectral theorem for real symmetric matrices, W has exactly 101 real eigenvalues (counted with multiplicity) and a corresponding set of orthogonal eigenvectors:

$$W \cdot x_k = \lambda_k \cdot x_k, \quad k = 1, 2, \ldots, 101$$

where the eigenvalues are ordered $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{101}$. Eigenvector centrality uses only the leading eigenpair (λ_1, x_1)—the eigenvector associated with the largest eigenvalue. The TasteRank score of variety i is defined as $x_{1,i}$, the i-th component of this leading eigenvector, normalized so that $\|x_1\|_2 = 1$.

A critical point must be emphasized: TasteRank has no closed-form analytical expression. There is no formula of the form "TR(i) = f(W_{i1}, W_{i2}, …)" that can be evaluated independently for a single variety by examining only its row of the adjacency matrix. The definition is implicit: TasteRank(i) = $x_{1,i}$ where x_1 is the vector that simultaneously satisfies the system of 101 coupled linear equations $W \cdot x_1 = \lambda_1 \cdot x_1$. Every variety's score depends on every other variety's score. The scores cannot be assigned independently—they must be solved for as a system.

This implicit, system-level nature is not a limitation but the defining feature of the measure. It is exactly analogous to Google's PageRank: there is no formula for any individual web page's rank; the rank is a fixed point of the entire hyperlink graph. Similarly, a grape variety's TasteRank is not a local property of its SAT profile or its immediate neighbors—it is a global property of the variety's position within the full network topology. Changing a single edge anywhere in the graph, even between two varieties far removed from variety i, can in principle alter variety i's TasteRank score. This is what makes eigenvector centrality fundamentally relational.

The operational definition of TasteRank—the way the implicit system is actually solved—is the limit of the power iteration process:

$$x_1 = \lim_{t \to \infty} W^t \cdot z^0 / \|W^t \cdot z^0\|$$

for any starting vector z^0 with positive projection onto x_1. This convergence is guaranteed by the Perron–Frobenius theorem (Section A.5.2) and is practically achieved in 35–45 iterations (Section A.5.3). The iterative process is not merely a computational convenience; it is the most intuitive way to understand what TasteRank measures: if you repeatedly ask "which varieties are similar to the currently important varieties?" and update importance accordingly, the answer stabilizes at the TasteRank ranking. The network's structure, encoded in W, determines the final ranking uniquely.

A.5.2 Why the Largest Eigenvalue Determines the Ranking

The fundamental insight is that the leading eigenvector captures the dominant mode of influence propagation through the network. To see why, consider any initial vector of scores z^0 (e.g., uniform scores $z^0 = (1, 1, \ldots, 1)$). Repeatedly applying the adjacency matrix produces:

$$z^t = W^t \cdot z^0$$

Since z^0 can be decomposed in the eigenbasis as $z^0 = c_1x_1 + c_2x_2 + \ldots + c_{101}x_{101}$, applying W repeatedly gives:

$$z^t = c_1\lambda_1{}^t x_1 + c_2\lambda_2{}^t x_2 + \ldots + c_{101}\lambda_{101}{}^t x_{101}$$

As t grows, the term $c_1\lambda_1{}^t x_1$ dominates because $\lambda_1 > \lambda_2$ (by Perron–Frobenius, the leading eigenvalue of a connected non-negative matrix is strictly dominant). All other terms decay

exponentially relative to the first: the ratio $|\lambda_2/\lambda_1|^t \to 0$ as $t \to \infty$. After normalization, $z^t/\|z^t\|$ converges to x_1 regardless of the starting vector.

This is the mathematical justification for why λ_1 alone determines the centrality ranking: after enough rounds of "passing influence through the network," the steady-state distribution of importance is entirely governed by the leading eigenvector. The higher eigenvalues (λ_2, λ_3, …) describe successively weaker modes of network structure that are relevant to community detection and spectral clustering (see A.7), but they wash out of the centrality computation.

A.5.3 The Spectral Gap and Convergence

The rate at which power iteration converges to x_1 depends on the spectral gap, defined as the ratio $|\lambda_2/\lambda_1|$. A smaller ratio means faster convergence. For the TasteRank graph, the spectral gap is approximately 0.72, meaning that each iteration of power iteration reduces the contribution of the second eigenvector by a factor of ~0.72. After t iterations, the second-order contamination is $(0.72)^t$—after 25 iterations this is below 10^{-3}, and after 50 iterations it is below 10^{-7}. This explains why the computation converges well within 50 iterations at tolerance $\varepsilon = 10^{-6}$.

The spectral gap also carries structural meaning. A large gap (small λ_2/λ_1) indicates that the network has a single, clearly dominant mode of centrality—a strong core-periphery structure. A small gap (ratio near 1) would indicate competing centers of influence. For TasteRank, the moderate gap of 0.72 reflects the fact that Community C0 (Mediterranean reds) forms a dense core, but Communities C2–C4 provide secondary structure that is not negligible.

A.5.4 The Recursive Interpretation

The eigenvector equation $W \cdot x = \lambda_1 \cdot x$ can be written component-wise as:

$$x_i = (1/\lambda_1) \cdot \Sigma_j W_{ij} \cdot x_j$$

This is not a formula that can be evaluated for a single variety in isolation. It is one equation in a system of 101 coupled equations: x_i depends on x_j for all neighbors j, each of which depends on its own neighbors' scores, and so on. The circularity is not a defect—it is the defining feature. The eigenvector is the unique self-consistent solution to this infinite chain of mutual dependencies. The TasteRank score of variety i is proportional to the similarity-weighted sum of the TasteRank scores of its neighbors, which are themselves defined by their neighbors' scores, ad infinitum.

This is the precise sense in which TasteRank is a recursive, relational measure: a variety earns high centrality not merely by having many connections (that would be degree centrality) or by being connected to many well-connected nodes (that would be a first-order approximation), but by being connected—with high cosine similarity—to varieties that are themselves connected to other high-centrality varieties, which are in turn connected to other high-centrality varieties, and so on to infinite depth. The eigenvector resolves this infinite recursion into a single, well-defined score for each variety.

Consider a concrete example. Sagrantino (rank 1, TasteRank = 0.3020) is connected to Nero d'Avola, Lagrein, Montepulciano, Petite Sirah, and Petit Verdot—all of which are themselves top-10 TasteRank varieties. Each of those neighbors is in turn connected to other high-centrality Mediterranean reds. The recursive amplification through this dense, mutually-reinforcing cluster pushes Sagrantino to the top. By contrast, Riesling (rank 85) is connected primarily to other peripheral varieties (Albariño, Furmint, Sauvignon Blanc) whose own neighbors are also low-centrality. The recursive logic works in reverse: peripheral connections propagate low scores.

This recursive property is precisely what distinguishes eigenvector centrality from simpler measures. Degree centrality counts edges. Closeness centrality measures path lengths. Betweenness centrality counts shortest paths. Only eigenvector centrality captures the full infinite-depth recursive structure of mutual reinforcement.

A.5.5 Role of Higher Eigenvalues

While λ_1 and x_1 determine the centrality ranking, the subsequent eigenvalues carry complementary structural information. The second eigenvector x_2 (associated with λ_2) captures the primary axis of variation orthogonal to centrality—in network science, this is closely related to the Fiedler vector used for spectral bisection. For the TasteRank graph, x_2 approximately separates red varieties (positive components) from white varieties (negative components), reflecting the strongest structural divide in the network.

The third and fourth eigenvectors further partition the network into sub-communities. Collectively, the first few eigenvectors define a low-dimensional embedding of the grape variety space (spectral embedding), which is closely related to the force-directed layout used in the TasteRank Explorer visualization. The Fruchterman–Reingold algorithm used by D3.js produces a 2D layout that empirically correlates with the positions defined by (x_2, x_3), which is why the visual clusters in the Explorer correspond well to the algorithmically detected communities.

A.5.6 Power Iteration Algorithm

The computation proceeds by power iteration, the standard algorithm for finding the leading eigenpair of a large matrix:

Step 1: Initialize $x^0 = (1/\sqrt{n}, 1/\sqrt{n}, \ldots, 1/\sqrt{n})$, a uniform vector.

Step 2: At each iteration t, compute $x^{t+1} = W \cdot x^t$.

Step 3: Normalize: $x^{t+1} \leftarrow x^{t+1} / \|x^{t+1}\|_2$.

Step 4: Repeat until $\|x^{t+1} - x^t\| < \varepsilon = 10^{-6}$ or $t > 2000$.

The eigenvalue λ_1 is recovered as the Rayleigh quotient: $\lambda_1 = x^{tT} W x^t$. For the TasteRank graph, convergence is achieved in approximately 35–45 iterations, well within the 2000-iteration budget. The final centrality vector x_1 is then sorted in descending order to produce the TasteRank ranking.

A.6 PageRank as Robustness Check

A.6.1 From Eigenvector Centrality to PageRank

PageRank, developed by Brin and Page (1998) for ranking web pages, is a modified form of eigenvector centrality that addresses a specific pathology: in networks with dense subgraphs, pure eigenvector centrality can concentrate almost entirely within the densest cluster, leaving nodes in sparser regions with near-zero scores. PageRank mitigates this by introducing a damping (or teleportation) mechanism.

The key modification is to replace the adjacency matrix W with a convex combination of the row-normalized transition matrix M and a uniform matrix:

$$G = \alpha \cdot M + (1 - \alpha) \cdot (1/n) \cdot J$$

where $M_{ij} = W_{ij} / \Sigma_j W_{ij}$ is the transition probability from i to j (proportional to edge weight), J is the n × n all-ones matrix, and $\alpha \in (0, 1)$ is the damping factor. The PageRank vector π is the stationary distribution of this modified matrix:

$$\pi = G^T \cdot \pi, \quad equivalently: \quad \pi = \alpha \cdot M^T \cdot \pi + (1 - \alpha) \cdot (1/n) \cdot 1$$

A.6.2 The Random Surfer Interpretation

The damping factor α admits an elegant probabilistic interpretation. Imagine a “random taster” navigating the grape variety network. At each step, the taster either (a) with probability α, follows an edge to a similar variety, choosing among neighbors with probability proportional to cosine similarity weight; or (b) with probability (1 − α), teleports to a completely random variety. The PageRank of each variety is the long-run fraction of time the random taster spends at that variety.

With α = 0.85, the random taster follows edges 85% of the time and teleports 15% of the time. This ensures that even isolated or peripheral varieties receive some baseline importance (the “teleportation floor” of (1 − α)/n ≈ 0.0015 per variety), preventing the extreme concentration that can occur with pure eigenvector centrality.

A.6.3 Structural Differences from Eigenvector Centrality

The two measures agree strongly overall (Spearman ρ ≈ 0.92) but diverge in predictable ways. PageRank rewards bridge varieties more than eigenvector centrality does. A variety connected to multiple communities serves as a bottleneck in the random walk—the random taster passes through it when moving between clusters. Eigenvector centrality, by contrast, rewards varieties embedded deep within the densest cluster, regardless of bridge position.

The most notable divergence is Sangiovese: TasteRank rank 40 vs. PageRank rank ~8. Sangiovese connects to 7 varieties spanning Communities C0 and C2, acting as a structural bridge between full-bodied Mediterranean reds and mid-weight structured reds. The random taster frequently passes through Sangiovese when transitioning between these clusters, inflating its PageRank. But Sangiovese’s neighbors have only moderate eigenvector centrality, so the recursive amplification of pure eigenvector centrality does not lift it as high.

Conversely, varieties deep inside Community C0 (e.g., Sagrantino, Nero d’Avola) have slightly lower PageRank relative to their TasteRank, because the teleportation mechanism redistributes some of their importance to peripheral nodes. The gap between the two measures is itself informative: large divergences identify bridge varieties (PageRank >> TasteRank) and cluster-core varieties (TasteRank >> PageRank).

A.7 Modularity and Community Detection

A.7.1 The Modularity Function

Community detection asks whether the graph can be partitioned into groups where within-group connectivity exceeds what would be expected by chance. The modularity function Q formalizes this:

$$Q = (1/2m) \cdot \Sigma_{ij} [W_{ij} - (s_i \cdot s_j / 2m)] \cdot \delta(c_i, c_j)$$

where m = $\Sigma_{ij} W_{ij} / 2$ is the total edge weight, $s_i = \Sigma_j W_{ij}$ is the weighted degree (strength) of node i, c_i is the community assignment of node i, and δ is the Kronecker delta (1 if $c_i = c_j$, 0 otherwise).

The term $s_i \cdot s_j / 2m$ represents the expected edge weight between nodes i and j under a null model (the configuration model) where edges are randomly rewired while preserving the degree sequence. Thus $W_{ij} - s_i \cdot s_j / 2m$ measures the excess connectivity between i and j relative to chance. Modularity Q sums this excess over all same-community pairs. A partition with Q > 0 has more within-community connectivity than expected; values of Q > 0.3 are generally considered indicative of significant community structure.

A.7.2 Connection to the Eigenvalue Spectrum

There is a deep connection between community detection and the eigenvalue spectrum of the modularity matrix B, where $B_{ij} = W_{ij} - s_i s_j / 2m$. Spectral community detection methods partition the network using the leading eigenvectors of B (rather than W). The number of positive eigenvalues of B provides an upper bound on the number of meaningful communities. For the TasteRank graph, the first six eigenvalues of B are positive and substantially separated from the bulk spectrum, consistent with the six-community partition detected by the greedy algorithm.

A.7.3 The Greedy Algorithm

The Clauset–Newman–Moore algorithm is an agglomerative approach: it starts with each node as its own community (101 singleton communities) and iteratively merges the pair of communities whose merger produces the largest increase in Q. The merge sequence defines a dendrogram, and the partition with the highest Q is selected as the final community structure.

For the TasteRank graph, the algorithm terminates with 6 communities and a modularity of Q ≈ 0.41, indicating strong community structure. The six-community partition is robust: alternative methods (Louvain, spectral bisection) produce partitions with similar membership and comparable modularity.

B. Graph Statistics

Statistic	Value
Nodes	101
Edges	341
k (neighbors)	5
Mean edge weight	0.9829
Median edge weight	0.9860
Min edge weight	0.9092
Max edge weight	1.0000
Communities detected	6
Graph density	0.0675

The edge weight distribution is tightly concentrated: the mean cosine similarity among connected pairs is 0.9829 with a range of [0.9092, 1.0000]. This reflects the high baseline similarity among wine grape varieties in SAT space—most varieties share the general shape of a wine sensory profile, with differentiation occurring in the relative magnitudes of specific dimensions.

C. Implementation

C.1 Technology Stack

The TasteRank Explorer is implemented as a single self-contained HTML file with no server-side dependencies. The technology stack consists of:

D3.js v7.9.0 for force-directed graph layout and interactive visualization, loaded from the Cloudflare CDN. All data (profiles, communities, TasteRank scores) is embedded directly in the JavaScript source. The visualization uses the Fruchterman–Reingold force-directed algorithm as implemented by D3's forceSimulation.

C.2 Computational Pipeline

The analytical computations were performed in Python using NetworkX for graph construction, eigenvector centrality, PageRank, and community detection. The pipeline proceeds as follows:

Stage 1 (Profile encoding): Grape varieties are encoded as 13-dimensional vectors based on WSET SAT calibration. Scores are stored in a JSON dictionary keyed by variety name.

Stage 2 (Similarity computation): The full 101 × 101 cosine similarity matrix is computed using the formula in Section A.2.

Stage 3 (Graph construction): For each variety, the k = 5 nearest neighbors are identified. The graph is symmetrized and weighted edges are stored.

Stage 4 (Centrality computation): NetworkX's eigenvector_centrality function computes TasteRank with max_iter=2000 and tol=1e-06. PageRank is computed with α = 0.85.

Stage 5 (Community detection): Clauset–Newman–Moore greedy modularity optimization identifies six communities.

Stage 6 (Visualization export): All computed data (nodes, edges, centrality scores, community assignments) is serialized into JavaScript objects and embedded in the HTML file.

C.3 Force-Directed Layout

The D3.js force simulation combines four forces:

Link force: Attraction between connected nodes with distance = 130 − 70w (where w is cosine similarity weight) and strength = 0.25w. Higher-similarity pairs are drawn closer.

Charge force: Repulsive many-body force with strength = −100, preventing node overlap.

Center force: Gravitational pull toward the viewport center, maintaining the graph within view.

Collision force: Radius-based collision detection with radius = r(TasteRank) + 4 pixels, where r is a square-root scale from 4px (lowest TasteRank) to 30px (highest).

The simulation runs for approximately 300 ticks (the D3 default α-decay) before stabilizing. After stabilization, the viewport auto-fits to the graph bounding box with 15% padding.

C.4 Visual Encoding

Node radius maps to TasteRank via a square-root scale: r = sqrt(TasteRank) scaled to [4, 30] pixels. Square-root scaling ensures area is proportional to centrality, avoiding visual exaggeration.

Node color maps to community assignment using a six-color palette designed for color-blind accessibility: C0 = #A8243B (crimson), C1 = #2E7D5B (forest), C2 = #C26A2F (amber), C3 = #B8860B (gold), C4 = #3A7CA5 (steel blue), C5 = #7B5EA7 (amethyst).

Edge opacity defaults to 0.18. On node selection, edges connected to the selected node increase to 0.75; all other edges fade to 0.015. Edge width ranges from 0.4 to 2.4 pixels based on weight.

Labels are shown for the top 30 varieties by default. On node or community selection, dynamic labels appear for all relevant varieties that were previously hidden, using a fade-in transition of 250ms.

C.5 Interactive Features

The interface supports six interaction modes: node click (inspect variety profile and connections), community click (isolate cluster), search (filter by variety name), drag (reposition nodes), zoom/pan (scroll wheel and drag), and keyboard reset (Escape key). The left panel displays contextual information: the top 15 varieties in default view, a full SAT profile bar chart on node selection, or a community member list on cluster selection.

C.6 Python Reference Implementation

The core analytical pipeline (cosine similarity, kNN graph, eigenvector centrality, PageRank, community detection) is implemented in approximately 80 lines of Python using NumPy and NetworkX. The key function calls are:

```
import numpy as np
import networkx as nx
from networkx.algorithms.community import greedy_modularity_communities

# Cosine similarity
def cosine_sim(a, b):
    return np.dot(a, b) / (np.linalg.norm(a) * np.linalg.norm(b))

# kNN graph construction (k=5)
G = nx.Graph()
for grape in grapes:
    sims = sorted([(g, cosine_sim(P[grape], P[g]))
                   for g in grapes if g != grape],
                  key=lambda x: -x[1])[:5]
    for neighbor, weight in sims:
        G.add_edge(grape, neighbor, weight=weight)

# TasteRank (eigenvector centrality)
tasterank = nx.eigenvector_centrality(G, max_iter=2000,
                                      tol=1e-06, weight='weight')
```

```
# PageRank robustness check
pagerank = nx.pagerank(G, alpha=0.85, weight='weight')

# Community detection
communities = greedy_modularity_communities(G, weight='weight')
```

C.7 JavaScript Implementation (D3.js)

The browser-side implementation recomputes cosine similarity and kNN edges at runtime from the embedded profile data, ensuring consistency. The core graph construction logic in JavaScript:

```
function cosineSim(a, b) {
  let d=0, na=0, nb=0;
  for (let i=0; i<a.length; i++) {
    d += a[i]*b[i]; na += a[i]*a[i]; nb += b[i]*b[i];
  }
  return d / (Math.sqrt(na) * Math.sqrt(nb) + 1e-9);
}

// Build kNN graph (k=5)
const K = 5;
for (const g of grapes) {
  const top = Object.entries(simMat[g])
    .sort((a,b) => b[1]-a[1]).slice(0, K);
  for (const [nb, w] of top) {
    const key = [g, nb].sort().join('|||');
    edgeMap[key] = { source:g, target:nb, weight:w };
  }
}
```

TasteRank scores and community assignments are precomputed in Python and embedded as JavaScript constants (TASTERANK and COMMUNITIES objects) to avoid reimplementing eigenvector centrality in the browser.