VibePolitics: Realistic Methodology

Signal Detection Based on Available Data

Version: 0.2 (Data-Grounded)
Date: February 5, 2026

---

1. Available Data Sources

1.1 Polymarket API □ CONFIRMED WORKING

Endpoint: `https://gamma-api.polymarket.com/markets`
Auth: None required
Rate limits: Reasonable (tested)

Available Fields (confirmed):

Field	Type	Use Case
`lastTradePrice`	float	Current probability
`bestBid` / `bestAsk`	float	Order book depth
`spread`	float	Bid-ask spread
`volume24hr`	float	Daily volume
`volume1wk`	float	Weekly volume
`volume1mo`	float	Monthly volume
`liquidityNum`	float	Market liquidity
`oneDayPriceChange`	float	24h price delta
`oneWeekPriceChange`	float	7d price delta

Field	Type	Use Case
`question`	string	Market description

Limitations:

- No historical tick data (only current snapshot + aggregates)
- No individual trade data
- No order book depth beyond top of book

1.2 Kalshi API ⚠ REQUIRES AUTH

Status: Requires API key/authentication
Availability: Need to apply for access

For now: Exclude from MVP, add later if we get access

1.3 Google Trends AVAILABLE VIA PYTRENDS

Method: pytrends library
Auth: None (but rate limited)

Available Data:

- Interest over time (daily/hourly)
- Interest by region (state-level US)
- Related queries
- Related topics

Limitations:

- Normalized 0-100 scale (relative, not absolute)
- Rate limited (~10-20 requests/minute)
- Can be unstable

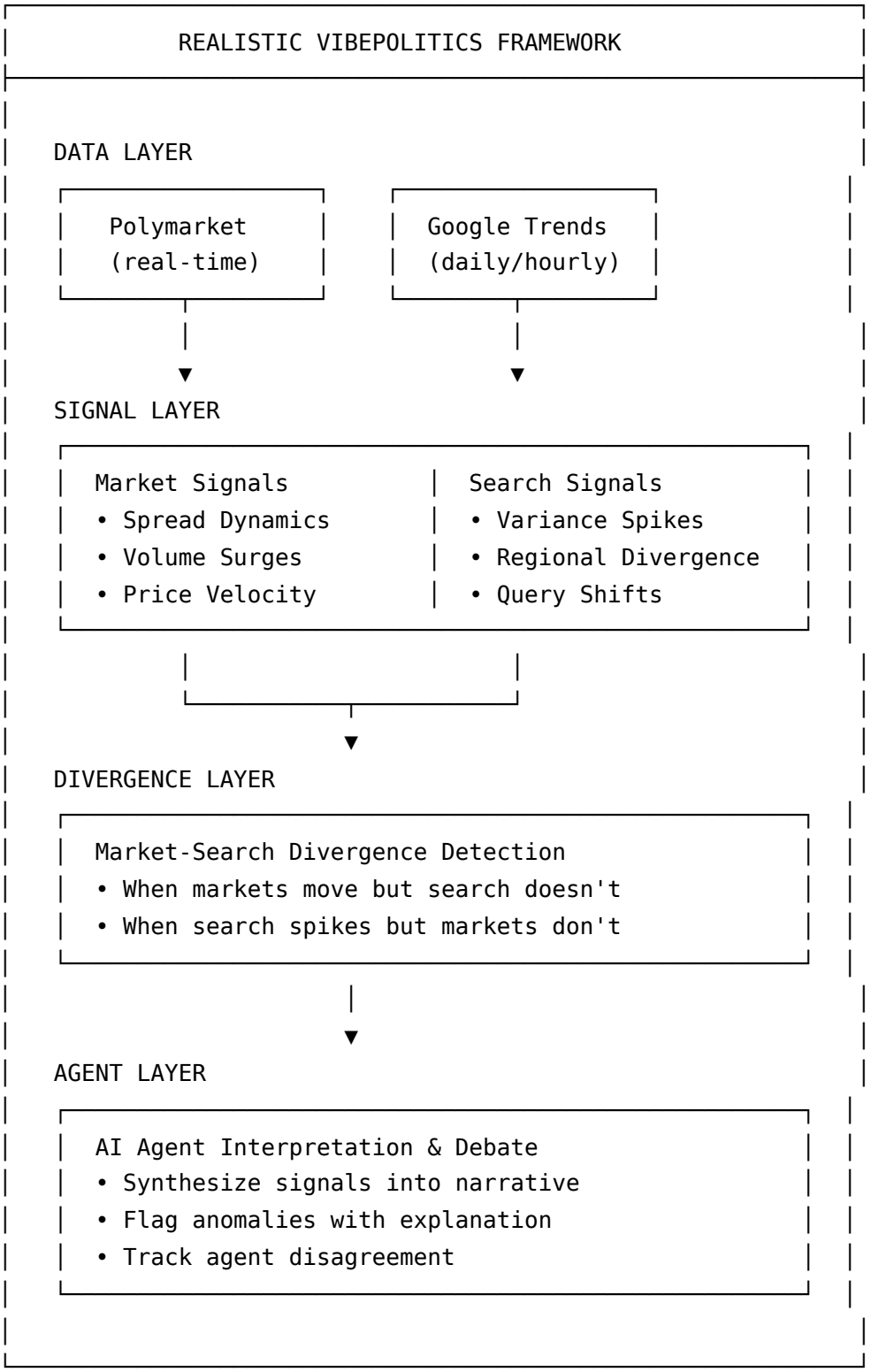
1.4 Supplementary Sources (Future)

- Twitter/X API (paid, expensive)
- Reddit API (available)
- News APIs (various)

---

2. Revised Signal Framework

Given data constraints, we focus on **what we can actually measure**:



---

3. Novel Algorithms (Realistic Implementation)

3.1 Spread Dynamics Index (SDI)

Data required: `spread`, `bestBid`, `bestAsk` from Polymarket
Computation: Simple, uses available snapshot data

Algorithm:

```
def spread_dynamics_index(current_spread, historical_spreads):
    """
    Detect unusual spread behavior.

    Widening spread = uncertainty/new information
    Narrowing spread = consensus forming
    """
    mean_spread = np.mean(historical_spreads[-30:])  # 30 snapshots
    std_spread = np.std(historical_spreads[-30:])

    if std_spread == 0:
        return 0

    sdi = (current_spread - mean_spread) / std_spread
    return sdi

# Interpretation:
# SDI > 2: Unusual spread widening (uncertainty surge)
# SDI < -2: Unusual spread narrowing (consensus forming)
```

What it detects:

- Information arriving that market makers can't price confidently
- Resolution of uncertainty as consensus forms

3.2 Volume Surge Ratio (VSR)

Data required: `volume24hr`, `volume1wk` from Polymarket
Computation: Ratio comparison

Algorithm:

```
def volume_surge_ratio(volume_24hr, volume_1wk):
    """
    Detect unusual volume relative to recent baseline.

    High VSR = sudden attention spike
    """
    daily_avg = volume_1wk / 7

    if daily_avg == 0:
        return 0

    vsr = volume_24hr / daily_avg
    return vsr

# Interpretation:
# VSR > 3: Volume 3x normal (attention spike)
# VSR > 5: Major event driving volume
# VSR < 0.5: Unusually quiet
```

What it detects:

- News events driving market attention
- Pre-announcement positioning
- "Calm before the storm" (unusually low)

3.3 Price Velocity Index (PVI)

Data required: `oneDayPriceChange` , `oneWeekPriceChange` from Polymarket
Computation: Acceleration detection

Algorithm:

```
def price_velocity_index(change_1d, change_1w):
    """
    Detect price momentum and acceleration.

    Measures if recent movement is accelerating or decelerating.
    """
    # Weekly rate (daily average)
    weekly_daily_rate = change_1w / 7

    if abs(weekly_daily_rate) < 0.001:
```

```
        weekly_daily_rate = 0.001  # Avoid division issues

    # How much faster is today vs. weekly average?
    pvi = change_1d / abs(weekly_daily_rate)

    return pvi

# Interpretation:
# PVI > 3: Today's move 3x the weekly daily rate (accelerating)
# PVI < -3: Sharp reversal
# |PVI| < 1: Normal volatility
```

What it detects:

- Momentum shifts
- Sudden sentiment reversals
- Trend acceleration/deceleration

3.4 Search Variance Spike (SVS)

Data required: Google Trends interest over time
Computation: Rolling variance analysis

Algorithm:

```
def search_variance_spike(search_data, keyword):
    """
    Detect variance spikes in search behavior.
    Based on Timoneda & Wibbels (2022).
    """
    recent = search_data[-7:]   # Last 7 days
    baseline = search_data[-30:-7]  # Previous 23 days

    var_recent = np.var(recent)
    var_baseline = np.var(baseline)

    if var_baseline == 0:
        return 0

    svs = var_recent / var_baseline

    return svs
```

```
# Interpretation:
# SVS > 3: Search behavior destabilizing (opinion in flux)
# SVS < 0.3: Abnormally stable search pattern
```

What it detects:

- Public attention becoming erratic (uncertainty)
- News events fragmenting attention
- Stabilization of narrative

3.5 Regional Divergence Score (RDS)

Data required: Google Trends by region (US states)
Computation: Cross-state correlation breakdown

Algorithm:

```
def regional_divergence_score(state_data):
    """
    Detect when states are diverging in search interest.

    High divergence = geographically uneven opinion shift
    """
    # Compute correlation matrix of state search patterns
    corr_matrix = np.corrcoef(state_data.T)

    # Average pairwise correlation
    n = corr_matrix.shape[0]
    avg_corr = (np.sum(corr_matrix) - n) / (n * (n - 1))

    # Divergence = inverse of correlation
    rds = 1 - avg_corr

    return rds

# Interpretation:
# RDS > 0.6: States searching very differently (regional divide)
# RDS < 0.3: Uniform national attention pattern
```

What it detects:

- Regional opinion polarization

- Local issues going national (or not)
- Swing state attention patterns

3.6 Market-Search Divergence (MSD)

Data required: Polymarket price + Google Trends volume
Computation: Normalized comparison

Algorithm:

```
def market_search_divergence(market_prices, search_volumes):
    """
    Detect when markets and search tell different stories.
    """
    # Normalize both to z-scores
    z_market = (market_prices - np.mean(market_prices)) / np.std(market_prices)
    z_search = (search_volumes - np.mean(search_volumes)) / np.std(search_volume

    # Recent divergence
    recent_msd = np.mean(np.abs(z_market[-7:] - z_search[-7:]))

    return recent_msd

# Interpretation:
# MSD > 1.5: Markets and search diverging (information asymmetry)
# MSD < 0.5: Markets and search aligned
```

What it detects:

- Markets have information public doesn't (insiders?)
- Public attention precedes market pricing
- Fundamental narrative disconnect

---

4. Agent System Design

4.1 Two-Agent MVP (Realistic Start)

Instead of four agents, start with two:

Agent Alpha (Bullish Interpreter)

- Looks for signals of momentum, energy, rising attention
- Interprets ambiguous signals optimistically
- Asks: "What would make this a big deal?"

Agent Beta (Skeptical Interpreter)

- Looks for noise, false positives, regression to mean
- Interprets ambiguous signals conservatively
- Asks: "Why might this be nothing?"

4.2 Agent Prompt Template

```
You are analyzing political market and search signals for the VibePolitics syste

Current signals for [MARKET_NAME]:
- Spread Dynamics Index (SDI): [VALUE] ([INTERPRETATION])
- Volume Surge Ratio (VSR): [VALUE] ([INTERPRETATION])
- Price Velocity Index (PVI): [VALUE] ([INTERPRETATION])
- Search Variance Spike (SVS): [VALUE] ([INTERPRETATION])
- Market-Search Divergence (MSD): [VALUE] ([INTERPRETATION])

Your role: [ALPHA/BETA - bullish/skeptical]

Questions to answer:
1. Is there a meaningful signal here, or is this noise?
2. If meaningful, what kind of public opinion shift might this indicate?
3. What additional information would help clarify?
4. Confidence level (0-100)?

Be specific. Cite the signal values in your reasoning.
```

4.3 Agent Disagreement as Signal

```
def agent_consensus_score(alpha_confidence, alpha_direction,
                          beta_confidence, beta_direction):
    """
    Measure agent agreement.

    High disagreement = genuinely ambiguous situation
    """
    # Direction agreement (-1 to 1 scale)
    direction_agreement = alpha_direction * beta_direction
```

```
    # Confidence-weighted disagreement
    weighted_disagreement = abs(alpha_confidence * alpha_direction -
                                beta_confidence * beta_direction)

    return {
        'agreement': direction_agreement > 0,
        'confidence_gap': abs(alpha_confidence - beta_confidence),
        'weighted_disagreement': weighted_disagreement
    }
```

---

5. Data Collection Schedule

5.1 Polymarket Polling

```
Every 15 minutes:
  - Fetch all active political markets
  - Store: timestamp, market_id, price, spread, volume24hr, liquidity

Every hour:
  - Compute rolling signals (SDI, VSR, PVI)
  - Flag anomalies above threshold
```

5.2 Google Trends Polling

```
Every 4 hours (to respect rate limits):
  - Fetch interest over time for key terms:
    - Candidate names
    - "election", "vote", "poll"
    - Current hot issues (immigration, economy, etc.)
  - Fetch regional breakdown for US

Daily:
  - Compute SVS, RDS signals
  - Compare with market signals (MSD)
```

5.3 Agent Analysis

```
When signal thresholds exceeded:
  - Trigger agent analysis
```

```
    - Log reasoning
    - Generate alert if warranted

  Daily summary:
    - Agent synthesis of all signals
    - Disagreement report
    - Notable patterns
```

---

6. Implementation Plan (Realistic)

Week 1-2: Data Pipeline

- [x] Polymarket API integration (confirmed working)
- [ ] Set up pytrends in venv for Google Trends
- [ ] PostgreSQL or SQLite for signal storage
- [ ] Cron jobs for data collection

Week 3-4: Signal Computation

- [ ] Implement SDI, VSR, PVI
- [ ] Implement SVS, RDS
- [ ] Implement MSD
- [ ] Backtest with historical data (where available)

Week 5-6: Agent System

- [ ] Design agent prompts
- [ ] Implement two-agent system
- [ ] Test agent disagreement detection
- [ ] Build alert system

Week 7-8: Dashboard & Validation

- [ ] Simple web dashboard
- [ ] Real-time signal display
- [ ] Agent reasoning logs
- [ ] Compare to known events

Week 9+: Academic Writing

- [ ] Document methodology formally
- [ ] Run during 2026 primary season
- [ ] Collect validation data
- [ ] Write paper

---

7. What We're NOT Doing (Scope Limits)

Given data constraints, we explicitly exclude:

1. **Order book depth analysis** - No data available
2. **Individual trade analysis** - No data available
3. **Cross-platform arbitrage** - Kalshi needs auth
4. **Social media sentiment** - Out of scope for MVP
5. **Real-time tick data** - Not available from Polymarket API

These can be added later if data becomes available.

---

8. Academic Contribution (Realistic)

What we can legitimately claim as novel:

1. **First systematic combination** of prediction market signals + Google Trends for opinion shift detection (not prediction)

2. **Novel signal definitions:**

    - Spread Dynamics Index (SDI)
    - Volume Surge Ratio (VSR) in political context
    - Price Velocity Index (PVI)
    - Market-Search Divergence (MSD)

3. **Agent disagreement as meta-signal** - Novel application of multi-agent AI for uncertainty quantification

4. **Shift detection vs. prediction framing** - Methodological reframing with different evaluation criteria

5. **Transparent reasoning** - All agent logic logged and auditable

---

Appendix: Sample Data Snapshot

From Polymarket (Feb 5, 2026):


```
{
  "question": "Will Trump deport 250,000-500,000 people?",
  "volume24hr": 96575.90,
  "volume1wk": 296162.52,
  "liquidity": 4542.85,
  "spread": 0.014,
  "bestBid": 0.883,
  "bestAsk": 0.897,
  "lastTradePrice": 0.897,
  "oneDayPriceChange": 0.0135
}
```


Signal computation:

- VSR = 96575.90 / (296162.52/7) = 2.28 (elevated but not extreme)
- SDI = need historical spread data
- PVI = 0.0135 / (weekly_rate) = need more context

This is real, actionable data we can build on.

Realistic methodology grounded in actual data availability.