# Data Extraction Project
# Wine Database & Map

Daniel Gagliardi, Barthélémy Charlier, Adrien Goldszal

December 19, 2025

# Motivation

- Fragmented and sparse wine data landscape
- No intuitive and high quality open source data with wines, geography, terroir, prices
- Useful downstream applications such as visualization (maps)

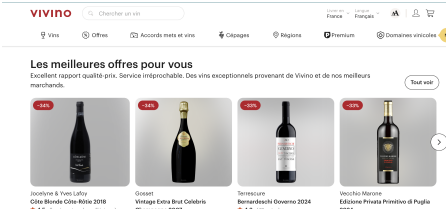



Figure: Vivino Wine Webpage

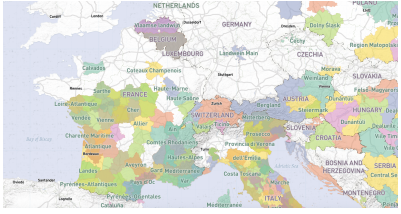



Figure: World Wine Regions Generic Map

# Crawling Wine Data : vivino.com

**Technical details**

- Selenium webdriver browser to click on wine cards and next pages
- robots.txt file and respectful scraping
- Parse html and find elements through CSS Selectors

**Results**

- 1503 unique wines after filtering duplicates (France, Italy, Spain)
- Winery, Place, Region, Rating, Price, Grapes, Alcohol %, Taste Characteristics, Pairings

# Getting Locations

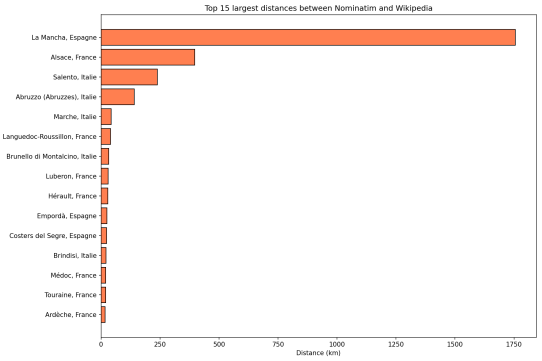
- Query **both** Nominatim API (OpenStreetMap) and Wikipedia API for each location.

| **Source Availability** | **Count** | **%** |
|---|---|---|
| Both Nominatim + Wikipedia | 90 | 27.9% |
| Nominatim only | 149 | 46.1% |
| Wikipedia only | 2 | 0.6% |
| Neither (failed) | 82 | 25.4% |
| Total locations | 323 | |
| Successfully geocoded | 241 | 74.6% |

# Getting Locations (Quality Verification)

- For locations resolved by both sources, compute the haversine distance:
  - If both agree (<50km apart): use Nominatim (more precise)
  - If they diverge (>50km): prefer Wikipedia (more reliable for wine regions)
  - If only one succeeds: use that source

| Threshold | Count | Cumulative % |
|---|---|---|
| $\leq$ 1 km | 59 | 65.6% |
| $\leq$ 5 km | 66 | 73.3% |
| $\leq$ 25 km | 81 | 90.0% |
| $\leq$ 50 km | 86 | 95.6% |
| $>$ 1000 km | 1 | 1.1% |



# Data Structure Choice: SQLite DBMS

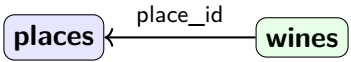
**Why SQLite?** Simple configuration, portable single file, built into Python, relational model with foreign keys, appropriate scale for 1500 wines.

**places** table

| | |
|---|---|
| `id` | INTEGER PK |
| `place` | TEXT |
| `region` | TEXT |
| `latitude` | REAL |
| `longitude` | REAL |
| `country` | TEXT |
| `source` | TEXT |

**wines** table

| | |
|---|---|
| `id` | INTEGER PK |
| `name` | TEXT |
| `vineyard` | TEXT |
| `rating` | REAL |
| `place_id` | FK $\rightarrow$ places |
| `grapes` | TEXT |
| `wine_style` | TEXT |
| `taste_*` | REAL ($\times 4$) |
| `food_pairings` | TEXT |



# Data Quality Assessment

### Dataset Overview

- 1,503 wines
- 323 unique regions
- 3 countries (France, Italy, Spain)
- Foreign key integrity: 100% (all wines linked to a place)

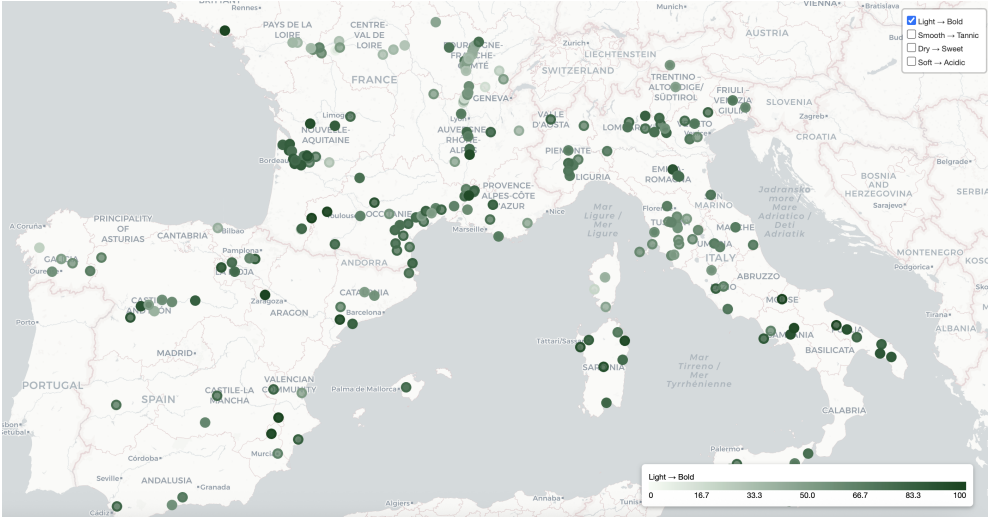
### Geocoding Coverage

- 241 / 323 regions (74.6%)
- Quality check : random sampling errors on 2 out of 20.

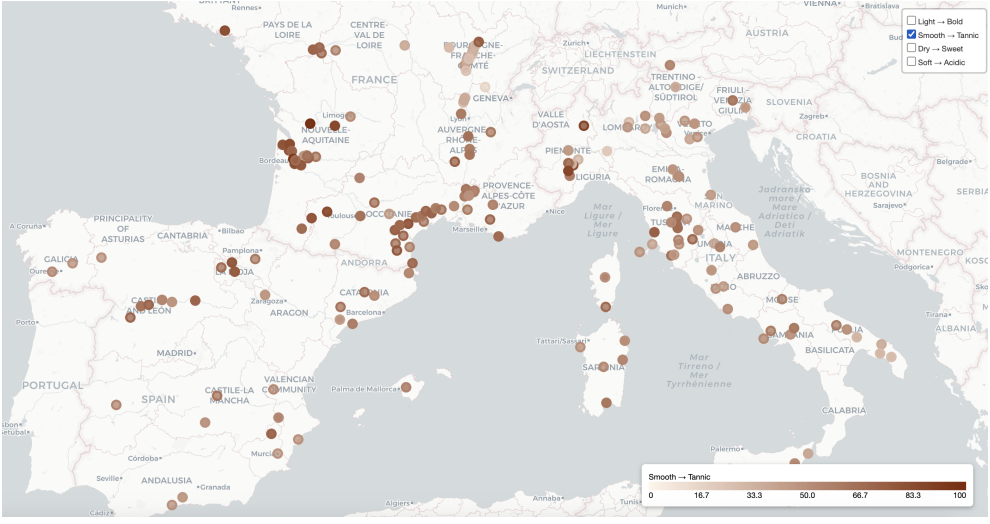
**Fill Rates (wines table)**

| Field | Rate |
|---|---|
| name, vineyard, rating | 100% |
| wine_style, url | 100% |
| food_pairings | 97.5% |
| grapes | 99.3% |
| taste characteristics | 78–100% |
| alcohol_content | 81.9% |
| description | 39.1% |

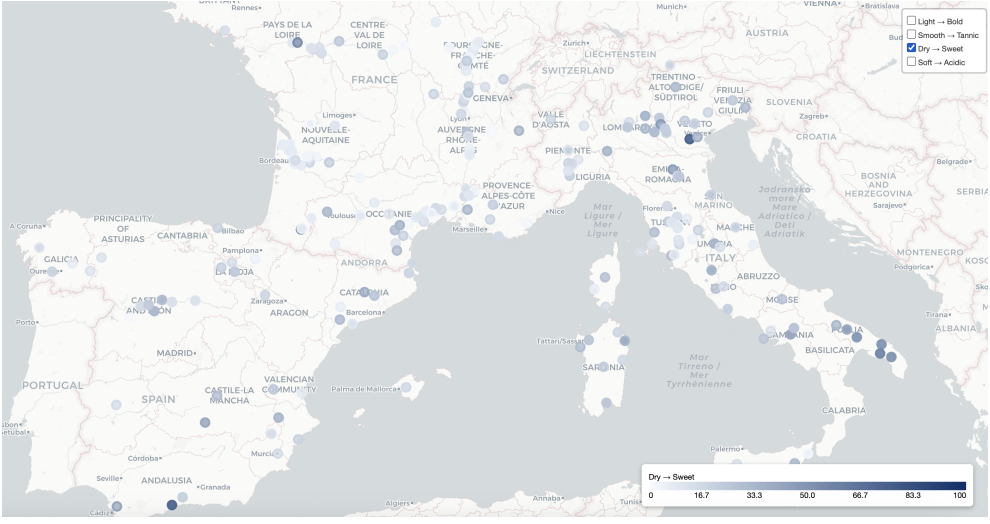
# Visualization: Wine Map (Light to Bold Wines)



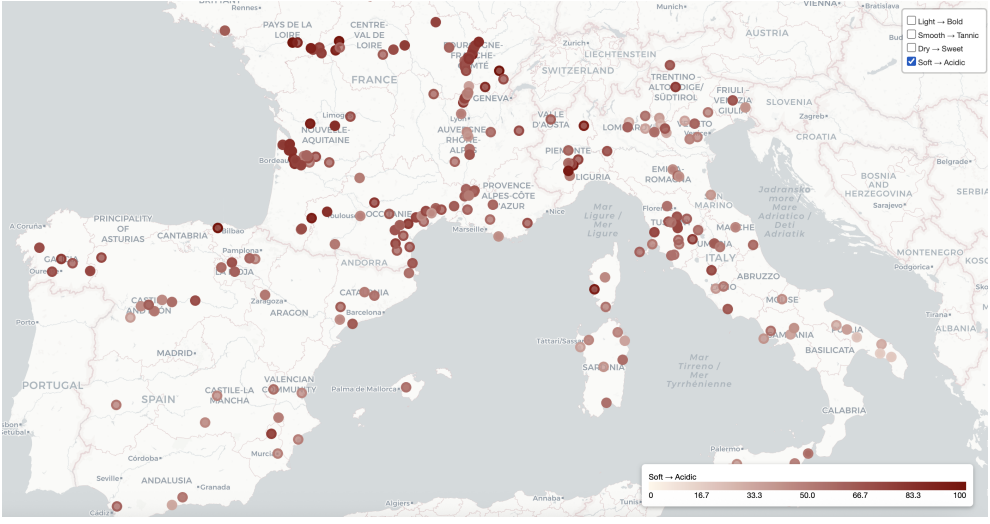
# Visualization: Wine Map (Smooth to Tannic Wines)



# Visualization: Wine Map (Dry to Sweet Wines)



# Visualization: Wine Map (Soft to Acidic Wines)



# Next Steps: There is a lot more that can be done!

- Use Google's API for accurate locations (requires setting up billing)
- How does price vary?
- Other metrics (grape type, alcohol percentage)
- Other data sources!
  - Weather, both long term and short term
  - Altitude
  - Other prices in other currencies
- More wines outside of 🇪🇸 🇫🇷 🇮🇹
- Have the database update continuously, while sticking to allowed directories