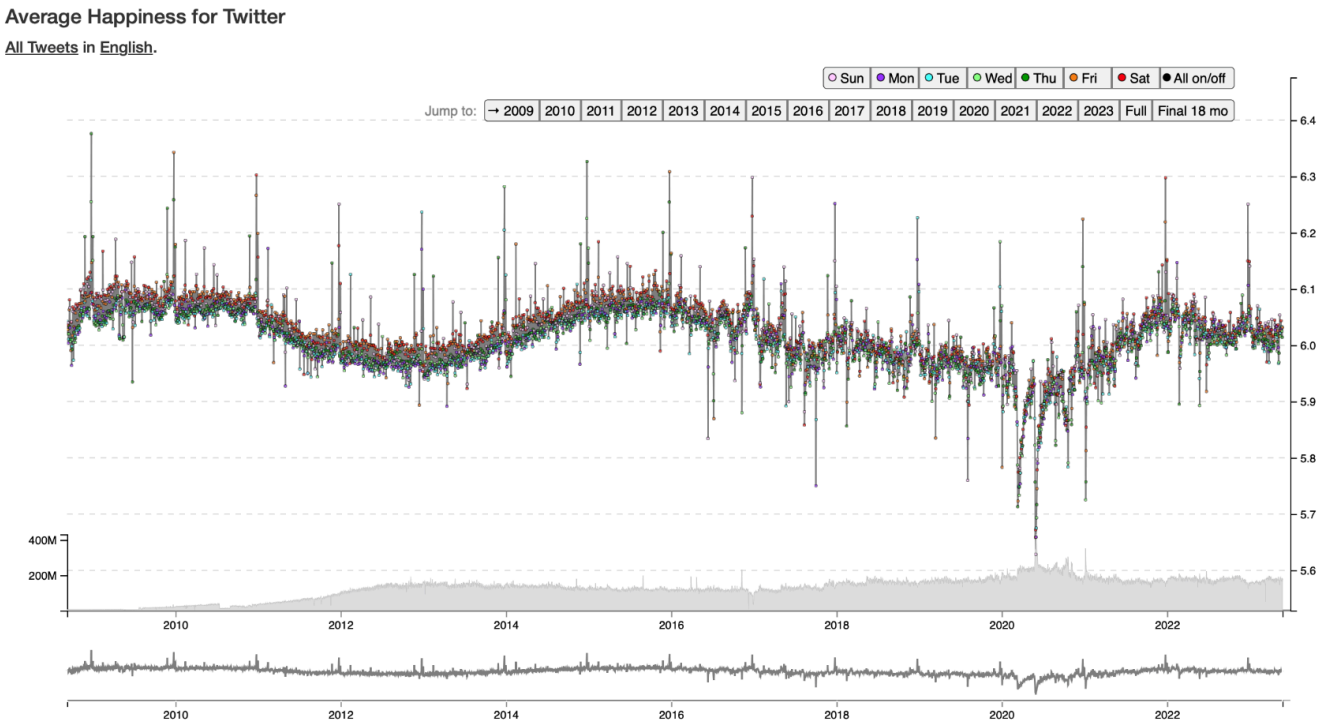
Happiness According to Mechanical Turks:

Quantitative + Qualitative Exploration of the Hedonometer (labMT 1.0) Dataset



Overview

In this group project, you will work with the **labMT 1.0** dataset ("language assessment by Mechanical Turk"), used in Dodds et al. (2011) to build a "hedonometer" for large-scale text (Twitter).

By the end of this project, you should be able to:

1. Load a real-world, tab-delimited dataset into Python and interpret its structure (columns, data types, missing values).
2. Create and explain a clear **data dictionary** for a dataset you did not build yourself.
3. Use basic descriptive statistics to summarize a distribution (mean, median, spread, outliers).
4. Make readable plots (with titles, labels, and captions) that support an interpretive point.

5. Combine **quantitative** exploration (counts, distributions, correlations) with **qualitative** exploration (close reading and interpretation of selected data points).
6. Critically analyze how a dataset was generated (collection, selection, annotation, preprocessing) and explain the consequences of those choices.
7. Communicate an argument through a reproducible **GitHub repository** (code + figures + written analysis in a README).

What you submit

Submit **one thing** per group:

- A link to your group's **GitHub repository**.

Your repository's front page (**README.md**) is your "final product." I will read your repository; nothing else counts as a submission.

Your repo must include:

- `README.md` (main deliverable; includes your narrative + plots)
- `src/` (your Python scripts)
- `data/` (your dataset, or a clear note on how to obtain it)
- `figures/` (saved images used in the README)
- `requirements.txt` (what someone needs to install to run your code)

Team size and roles

Teams of **4 to 6** students.

Pick roles so that the work is genuinely shared. One person can hold more than one role in a team of 4, but **everyone must contribute code and writing**.

Suggested roles (choose 4–6)

1. **Repo & workflow lead**
   - Creates the GitHub repo and folder structure.
   - Manages branches / merges (or coordinates who edits which files).
   - Ensures the README stays organized and readable.
2. **Data wrangler**
   - Loads the dataset, handles missing values, converts data types.
   - Produces the data dictionary and "what each column means" section.
3. **Quantitative analyst**
   - Leads descriptive statistics and at least 2 core plots.
   - Checks results for sanity and reproducibility.
4. **Qualitative / close-reading lead**

- Leads careful interpretation of selected words (examples, ambiguity, cultural meaning).
- Connects qualitative observations back to patterns in the plots.

5. **Provenance & critique lead**
   - Reconstructs how the dataset was generated (pipeline).
   - Writes the “critical reflection” sections: consequences, bias, limitations, and what the dataset makes easy/hard to see.
6. **Editor & figure curator**
   - Makes sure plots have labels and captions.
   - Ensures the README reads smoothly and makes a clear argument.

Project tasks

Your README should contain results from all tasks below. These tasks should be divided across team members. This might not be an exhaustive list of tasks, and you are allowed to deviate. In your final project, you will have to figure out tasks and division of labour yourselves – here we provide an example set. Each task includes:

- **What to do (code)**
- **What to write (interpretation / publication)**

1. Load, clean, and describe the dataset

1.1 Load the file

Code tasks

- Read the tab-delimited file into a pandas DataFrame.
- Skip or handle the comment lines at the top (the dataset begins after metadata lines).
- Convert numeric columns to numeric types (floats/ints).
- Replace `--` with missing values (`NaN`).
- Confirm the number of rows and columns.

Write-up tasks (README)

- Explain *how you loaded the file* (1–3 sentences).
- State the shape of the dataset (rows × columns).
- Give one sentence explaining what a missing rank (`--`) means in this dataset.

1.2 Create a data dictionary

Code tasks

- List each column name and its data type.
- Count missing values per column.

Write-up tasks (README)

- Make a short **data dictionary** section with one bullet per column:
  - What it represents
  - Type (text / integer / float)
  - Notes on missingness

1.3 Sanity checks

Code tasks

- Check for duplicated words (are any words repeated?).
- Inspect a random sample of 15 rows.

- Identify the 10 most positive and 10 most negative words by average happiness.

Write-up tasks (README)

- Choose 2–3 sanity checks and explain what they tell you about data quality.
- Briefly comment on whether the most positive/negative words "make sense" to you—and what "make sense" even means here.

2. Quantitative exploration: distributions and relationships

Your goal is to describe what the dataset "looks like" statistically, and to notice patterns that invite interpretation.

2.1 Distribution of happiness scores

Code tasks

- Plot a histogram of `happiness_average`.
- Compute summary statistics:
  - mean, median
  - standard deviation
  - 5th and 95th percentiles (or similar)

Write-up tasks (README)

- Interpret the histogram in words. Is the distribution centered? skewed? clustered?
- Identify 1 pattern you did not expect.

2.2 Disagreement: which words are "contested"?

The dataset includes `happiness_standard_deviation`. That means you can ask: which words did people disagree about?

Code tasks

- Plot `happiness_average` (x-axis) vs `happiness_standard_deviation` (y-axis) as a scatterplot.
- Identify the 15 words with the highest standard deviation.

Write-up tasks (README)

- Pick 5 of the "most disagreed-about" words and discuss *why* they might be contested:
  - ambiguity / multiple meanings
  - cultural references
  - slang and time period
  - irony, profanity, or taboo
- Connect your qualitative interpretation to the quantitative pattern.

2.3 Corpus comparison: what counts as "common language" depends on where you look

The dataset includes a rank column for each corpus. This lets you study overlap and difference.

Code tasks

- For each corpus (Twitter / Google Books / NYT / Lyrics):
  - count how many labMT words appear in its top 5000 (i.e., rank is not missing)
- Compute a simple overlap table:
  - e.g., how many words appear in both Twitter and NYT? in all four?
- Make at least **one plot** about corpus differences (your choice):
  - bar chart of "how many words are present"
  - heatmap-like table (even simple) of overlaps
  - scatterplot of Twitter rank vs NYT rank for words present in both (optional)

Write-up tasks (README)

- Interpret what your plot suggests about the four corpora.
- Give one concrete example of a word that is "common" in one corpus but missing in another, and interpret why that might be.

3. Qualitative exploration: close reading the lexicon as a cultural artifact

This is where you bring humanities skills directly into a data project.

3.1 Build a small "exhibit" of words

Code tasks

Create a small table (you can print it or save it) of **20 words** selected across categories:

- 5 very positive
- 5 very negative
- 5 highly contested (high standard deviation)
- 5 "weird / surprising / historically dated / culturally loaded" (your choice)

Write-up tasks (README)

Write an interpretative paragraph addressing things like:

- What meanings/contexts the words can have
- Why an happiness score might be high/low
- What kinds of voices or communities might use it differently

Your goal is not to be "right," but to show careful interpretive reasoning.

4. Critical reflection: how was this dataset generated, and why does it matter?

4.1 Reconstruct the pipeline (data provenance)

Write-up tasks (README)

In your own words, reconstruct the dataset's generation pipeline as a sequence of steps.

(You can present this as a numbered list, diagram, or short narrative.)

4.2 Consequences and limitations (your critical argument)

Write-up tasks (README)

Discuss at least **five** consequences of the dataset's design choices. For each consequence, include:

- **The choice** (what did they do?)
- **The consequence** (what does this make easier/harder to see?)
- **A concrete example** from your exploration (a word, a plot pattern, or a missingness pattern)

4.3 If you were to use this dataset as an instrument today…

Write-up tasks (README)

Write a short "instrument note" (200–400 words):

- What would you trust this dataset to measure well?
- What would you refuse to claim based on it?
- What improvements would you make if you rebuilt it?

README checklist (what your front page should contain)

Use this as your structure.

1. **Project title + 2–3 sentence overview**
2. **Dataset section**
   - where it came from
   - what each column means (data dictionary)
3. **Methods section (what you did in Python)**
4. **Results section**
   - plots + captions
   - interpretation in plain language
5. **Qualitative "exhibit" of words**
6. **Critical reflection**
7. **How to run your code**

   - setup steps
   - which script(s) to run
8. **Credits**
   - who did what (team roles)
   - citation for the paper / dataset

Technical constraints (to keep the work comparable)

- Use **Python + pandas + matplotlib** (no need for advanced libraries).
- Make sure your code runs from a clean environment using only `requirements.txt`.
- Your plots must have:
  - title
  - axis labels
  - readable tick marks
- Save plots into `figures/` and embed them in the README.

Academic integrity & AI note

Using an AI assistant to debug code get a first draft and clarify concepts is encouraged, but:

- You must understand and be able to explain the code in your repository.
- You must cite substantial AI assistance in the README ("Tools used" section).
- You are responsible for the interpretive claims you make.