GitHub Repository Metadata Dataset (2015–2024)

A Short Research Paper

1. Title & Collection Method

Dataset Title:
GitHub Repository Metadata Dataset (2015–2024)

Description:
This dataset contains metadata of public GitHub repositories collected using the GitHub REST Search API. The goal of the dataset is to study repository characteristics and their relationship to popularity and engagement, and to prepare the data for machine learning experiments.

Collection Method:
The data was collected programmatically using the GitHub Search Repositories API. Due to GitHub's restriction of returning a maximum of 1000 results per query, a **date-window sampling strategy** was used. Small creation-date windows (1–3 days) were randomly selected between the years **2015 and 2024**. For each window, paginated API requests were made (up to 10 pages with 100 repositories per page).

To avoid duplicate repositories, each repository was uniquely identified using its `repo_id`. A set of previously seen IDs was maintained, ensuring that each repository appears only once in the dataset. Data collection continued until at least **2000 unique repositories** were obtained.

The dataset was saved in both **CSV** and **Excel (XLSX)** formats in its raw form, without any balancing, filtering, or feature transformations.

---

2. Description of Features & Labels

Input Variables (Features X)

Each row in the dataset represents a single GitHub repository. The main features include:

- **repo_id**: Unique numeric identifier of the repository
- **name**: Repository name
- **full_name**: Owner and repository name combined
- **owner_type**: Whether the owner is a User or an Organization
- **created_at**: Date and time when the repository was created
- **updated_at**: Date and time of the last update

- **language**: Primary programming language (may be missing)
- **size**: Repository size in kilobytes
- **stargazers_count**: Number of stars (main popularity indicator)
- **forks_count**: Number of forks
- **watchers_count**: Number of watchers
- **open_issues_count**: Number of open issues
- **has_issues**: Whether issues are enabled
- **has_projects**: Whether projects are enabled
- **description**: Short text description of the repository (optional)

These features describe repository activity, engagement, maintenance, and technical characteristics.

Output Variable (Label y)

The dataset supports multiple possible labels depending on the machine learning task:

1. **Regression (recommended):**
    - **Target:** `log(stars + 1)`
    - This transformation reduces the extreme skew in star counts and allows the model to learn smoother patterns.
2. **Classification (optional):**
    - **Target:** Popularity categories derived from stars (e.g., low, medium, high popularity).
    - These labels are **derived during preprocessing**, not collected directly from the API.

No derived labels were added during data collection to preserve the raw nature of the dataset.

---

3. Dataset Structure

- **Number of rows:** ≥ 1000 repositories
- **Number of columns:** 14–16 (depending on optional fields)

Sample of Dataset Structure (illustrative)

repo_id	full_name	owner_type	created_at	language	stars	forks	watchers
123456	userA/ml-tool	User	2016-05-12	Python	5200	140	520
234567	orgB/ui-lib	Organization	2022-11-01	TypeScript	320	40	320
345678	devC/graphx	User	2016-01-09	Rust	9800	400	980
456789	student/demo	User	2024-06-10	JavaScript	5	2	5

567890	ops/scripts	Organization	2019-09-30	Shell	60	5	60

(This table demonstrates structure; actual values come from the collected CSV.)

---

4. Data Quality Issues

Several real-world data quality challenges exist in the dataset:

- **Missing values:**
  Fields such as `language`, `description`, and topics may be null.
- **Imbalanced distribution:**
  Star counts follow a long-tail distribution, where most repositories have very few stars and only a small number are highly popular.
- **Skewed numerical values:**
  Popular repositories can have extremely high star counts compared to the majority.
- **Text noise:**
  Descriptions may contain emojis, URLs, or inconsistent formatting.
- **Temporal bias risk:**
  Without careful sampling, data could overrepresent certain years. Random date-window sampling helps mitigate this.
- **API-related issues:**
  Rate limits, pagination limits, and partial responses require careful handling.

These issues make the dataset realistic and suitable for demonstrating data preprocessing techniques.

---

5. Use Case in Machine Learning

This dataset can be used in multiple machine learning scenarios:

Regression

- **Goal:** Predict repository popularity (star count).
- **Target:** `log(stars + 1)`
- **Models:** Linear Regression, Random Forest, Gradient Boosting.
- **Metrics:** MAE, RMSE.

Classification

- **Goal:** Classify repositories into popularity levels.
- **Challenges:** Severe class imbalance.

- **Solutions:** Class weighting, resampling, appropriate evaluation metrics (F1-score, recall).

Clustering

- **Goal:** Group repositories based on language, activity, and engagement.
- **Use case:** Discover types of projects or developer communities.

NLP-based Analysis

- **Goal:** Use repository descriptions to predict popularity or categorize topics.
- **Techniques:** Text vectorization, embeddings.

---

6. Conclusion

The GitHub Repository Metadata Dataset provides a realistic, large-scale, and unbalanced dataset suitable for studying data foundations in machine learning. Its raw nature, natural skew, and temporal diversity make it ideal for demonstrating data collection challenges, preprocessing decisions, and model evaluation strategies. The dataset supports regression, classification, clustering, and exploratory analysis tasks, making it a strong foundation for practical machine learning projects.
