BA820 — Project Proposal

Project Title: Code Trends, Quantified: Mapping the Programming Language Ecosystem

Section and Team Number: B1 Team 14

Members: Drishti Chulani, Arshdeep Singh Oberoi, Ahrar Karim, Vishesh Goyal

1. Primary Dataset

1.1 Project Motivation and Preliminary Exploratory Data Analysis (EDA)

1.1.1 Project Motivation

As MS Business Analytics students with deep roots in technology, we are constantly navigating the global "fuss" and intense hype surrounding the world of coding. While we rely on Python as our primary tool for solving complex analytical problems, we realized we were only seeing a small fraction of a much vaster landscape. This project was born from a desire to look under the hood of that hype, using our data expertise to perform a comprehensive census of the programming paradigm and understand the thousands of languages that have shaped human logic.

Our goal is to quantify this ecosystem by exploring the sheer scale of languages in existence—from ancient numeral systems to the latest 2023 releases. By analyzing key metrics like job demand and user populations, we aim to identify which languages are the true titans of industry and which are emerging as the next big thing. Ultimately, we want to transform the noise surrounding "learning to code" into a clear, data-driven map that shows where the tech world has been and where it is heading.

Major stakeholders would be Chief Technology Officers (CTOs), Product Managers, School Educators, Mentors and Students.

1.1.2 Preliminary Exploratory Data Analysis (EDA)

One of the biggest surprises in the dataset is how broadly it defines “language,” which completely changes what “most employable” looks like. If you rank by job openings, the leader isn’t Python or Java—it’s HTTP (with SQL right behind), which suggests companies value people who understand how the web and data infrastructure actually work, not just syntax. The dataset also spans an astonishing 4,000+ year timeline, from Babylonian numerals to languages created as recently as 2023, and it even treats mathematical notations like binary as ancestors to modern programming—blurring the boundary between early logic systems and today’s code.

At the same time, the EDA highlights a few “quiet giants” and overlooked patterns. SQL stands out as the true lingua franca of modern tech with the highest user count (~7.1M), outperforming JavaScript and Java, reinforcing that almost every software or data role eventually depends on databases. Historically, it’s also striking that Niklaus Wirth is the most prolific creator in the dataset, credited with eight languages, showing that language design can be a sustained discipline rather than a one-off invention. Finally, the category breakdown reveals a classic “long tail”: while most entries are standard programming languages, there’s a large ecosystem of specialized formats and query/markup languages, suggesting that a major part of the industry is devoted not just to writing logic, but to structuring, describing, and retrieving data.

1.2 Domain / Business Questions (Core Section)

Question 1: Do programming languages naturally cluster into distinct groups based on shared technical features (syntax patterns, file types) and community signals (GitHub activity, Wikipedia view)?
Why this Question: Our EDA reveals a massive long tail of niche languages that may share the structural features of more popular repositories.

Question 2: Can we segment programming languages into distinct "Market Archetypes": specifically, "Hype Driven" (high GitHub stars/subscribers, low job counts) vs. "Silent" (low social hype, high job counts and user estimates)?

Why this Question: Scatter plots show that viral GitHub popularity does not necessarily correlate with real-world production use or job availability.

Question 3: Can we detect "Ghost Languages" technologies with significant cultural visibility (high Wikipedia views and rankings) but virtually no real employment ecosystem?
Why this Question: Despite high rankings, many languages show a median job count of zero, indicating a disconnect between discussion and actual employment.

Question 4: Among the 49 technical metadata features we have (syntax patterns, file types, naming conventions), which ones actually predict how long a language survives and remains relevant in the long run?
Why this Question: Reducing noise from 49 sparse columns helps identify the minimal technical characteristics required for a language's long-term staying power.

2. Backup Dataset

2.1 Project Motivation and Preliminary Exploratory Data Analysis (EDA)

2.1.1 Project Motivation

Bob Ross's "The Joy of Painting" is an unusual phenomenon in terms of the confluence of art, education, and visual consistency on a large scale. Over the course of numerous episodes, Bob Ross created paintings that, while visually consistent in terms of style at first glance, show interesting variation in terms of the use of colors, composition, and the scenes depicted. Determining the underlying patterns of variation in terms of latent styles, compositions, or temporal evolution is not only important from an artistic perspective but also from the perspective of content organization and decision-making.

The dataset used in this project, which represents paintings in terms of structured indicators for the use of colors, the presence of landscape, and the ordering of the episodes, creates an unusual opportunity for the analysis of artistic patterns on a large scale. Unlike traditional approaches to analyzing images, this dataset represents paintings in terms of abstract, understandable concepts, which makes it possible to analyze the patterns in terms of style, composition, and variation in an understandable manner.

From a business and organizational point of view, finding the latent structure within this data can be used to organize the visual content in terms of categorization, recommendation, and even repurposing. For instance, finding the visual styles or templates within the images can be used for content recommendation systems, art education platform curricula, and even thematic curation for streaming or media archives. Since the data set doesn’t offer any labels for the styles or categories, an unsupervised method can be used effectively to find the hidden patterns within the data.

2.1.2 Preliminary Exploratory Data Analysis (EDA)

The initial EDA suggests that color usage varies widely across paintings. The derived feature `num_colors` shows a skewed distribution: many paintings use a relatively small palette, while a smaller subset uses a much richer range of colors. In addition, colors are typically not used in isolation—most appear alongside other colors—implying that color combinations (not just single colors) may be a strong signal for differentiating visual styles.

For landscape elements and composition, the variables show uneven frequency: common natural elements (e.g., trees, mountains, water) appear often, while cabins or other man-made objects are comparatively rare. Elements also tend to co-occur in recurring bundles, hinting at repeated composition templates across paintings. Row-level aggregation reveals meaningful variation in overall composition density, where some paintings are sparse and others are complex, suggesting this may reflect stylistic choice rather than constraints of particular episodes. Finally, the temporal analysis using `chronological_order` indicates that while overall element and color presence stays fairly consistent, the mix and emphasis shift over time, with no simple linear trend in palette size or contrast—supporting the idea that style evolves in a non-monotonic way across seasons.

2.2 Domain / Business Questions (Core Section)

Question 1: Do Bob Ross paintings exhibit latent "visual styles" driven more by color palette structure than by the specific landscape elements depicted?
Why this Question: This dataset helps determine if painting style is driven more by color palette size than by the specific elements depicted.

Question 2: Do paintings at different points in the chronological order of the series differ more in composition structure than in subject matter?
Why this Question: Using the chronological_order variable, we can distinguish between shifts in subject matter versus changes in overall structural composition

Question 3: Are there distinct painting clusters defined by contrast intensity (light vs dark color mixes), independent of season order?
Why this Question: The lack of a monotonic trend in light and dark color combinations suggests that contrast intensity is a deliberate, recurring stylistic tool.

Question 4: Are there recurring "element bundles" that appear together far more often than expected given their individual frequencies?
Why this Question: Analyzing binary element data reveals stable compositional templates useful for scalable content categorization and recommendation engines.

3. Appendix

3.1 GitHub Repository

Link to Github Repo: Github Repo Link

3.2 Other Important Links

Link to Actual Proposal Notebook: Proposal Notebook

Link to Colab Notebook (EDA Primary Dataset): Primary Dataset EDA Notebook

Link to Colab Notebook (EDA Backup Dataset): Backup Dataset EDA Notebook

Primary Dataset: Programming Languages Dataset Link

Backup Dataset: Bob Ross Paintings Dataset Link

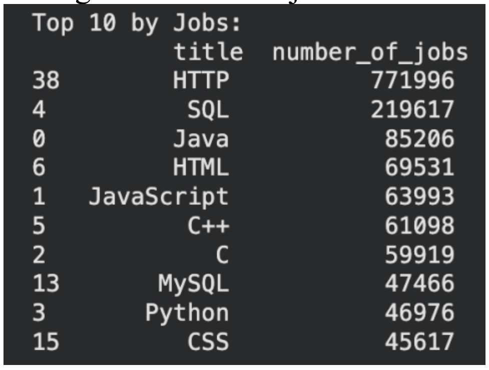
3.3 Process Overview

The exploratory data analysis (EDA) on the datasets aimed to lay a solid groundwork for understanding the programming languages and paintings ecosystem from a macro perspective. This process began with standardizing the datasets, quality checks, and validating them to ensure structural consistency. This was followed by a descriptive analysis to investigate specific categorical and textual attributes, identify major patterns, and unveil overall trends in language attributes and origins. Summary statistics and distributional tests were used to confirm assumptions, reveal overall relationships, and validate dataset representativeness. In aggregate, these EDA activities sought to unify understanding from the datasets, minimize uncertainty, and lay the groundwork for using the data for analysis.

3.4 Supplementary Material

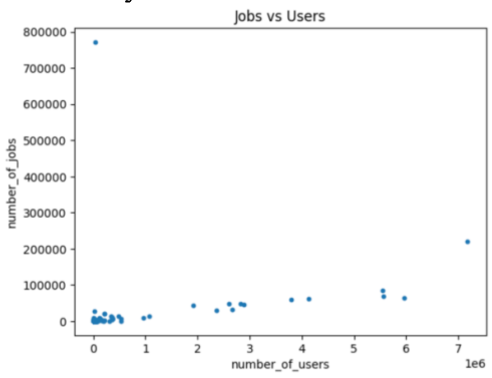
3.4.1 Primary Dataset

3.4.1.1 Top 10 Languages for Jobs: This table shows what are the top 10 languages which have the highest number of jobs

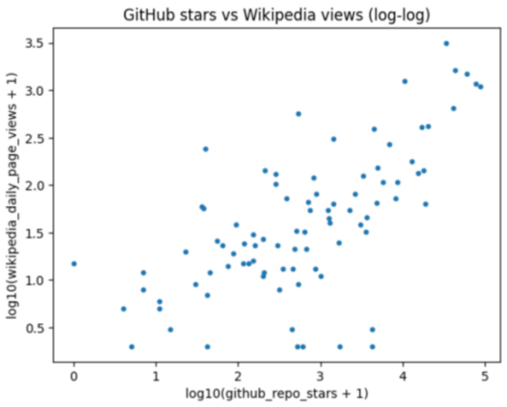


```
Top 10 by Jobs:
          title  number_of_jobs
38         HTTP          771996
4           SQL          219617
0          Java           85206
6          HTML           69531
1    JavaScript           63993
5           C++           61098
2             C           59919
13        MySQL           47466
3        Python           46976
15          CSS           45617
```

3.4.1.2 Job vs User: Most languages cluster in the lower-left, meaning they have low user adoption and low job demand.A few extreme outliers dominate, showing jobs and users are heavily concentrated in a small number of languages.

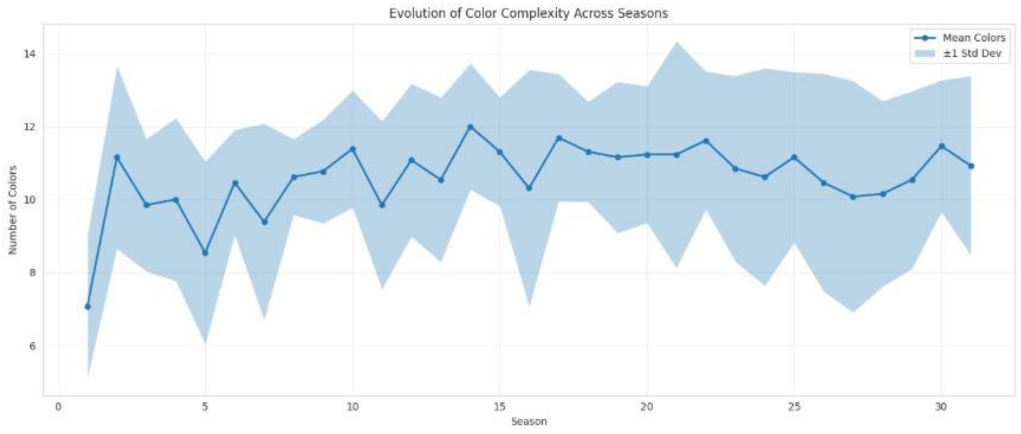


3.4.1.3 Github Stars vs Wikipedia Views: Languages with more GitHub stars generally also have higher Wikipedia daily page views, showing a positive relationship.But the spread is wide, meaning developer popularity and general-interest popularity don’t always match.

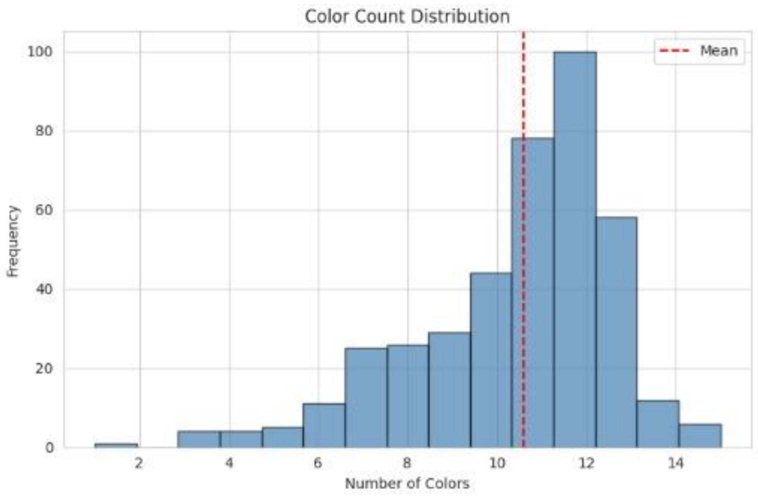


3.4.2 Backup Dataset

3.4.2.1 Evolution of Color Complexity Across Seasons: Average color complexity generally trends upward over seasons with noticeable fluctuations, suggesting gradual diversification.



3.4.2.2 Color Count Distribution: The number of colors is concentrated around 11, with a roughly normal distribution centered near the mean, indicating most designs use a similar level of color complexity.



3.5 Use of Generative AI Tools

In the development of this project, Generative AI tools—specifically OpenAI's ChatGPT and Google's Gemini—were utilized to augment the research and implementation process. The following sections outline the specific applications and the human oversight involved:

1. Proposal and Conceptualization: AI was used to structure the initial project proposal, helping to refine the scope and articulate the objectives clearly. It served as a brainstorming partner to ensure the project's goals were comprehensive and well-aligned with the dataset's potential.

2. Code Generation and Technical Implementation: To expedite the coding process, Gemini was integrated with Google Colab to generate Python code snippets for data cleaning, type conversion, and visualization. These tools were instrumental in handling complex data manipulations, such as managing missing values and implementing logarithmic scaling for skewed distributions.

3. Debugging and Error Rectification: Both ChatGPT and Gemini were utilized as technical assistants to identify and resolve coding errors. By providing specific error logs to the AI, we were able to understand the underlying causes of runtime warnings (such as infinite values in log-scales) and apply robust fixes efficiently.

4. Data Understanding and Domain Knowledge: AI models were used to gain a deeper understanding of the languages.csv dataset. This included interpreting the significance of various metadata fields (e.g., Semantic Scholar citations vs. GitHub stars) and implementing specific EDA techniques to derive meaningful insights from the text and numeric data.

Human Oversight and Responsibility: While AI was used to generate initial drafts of code and text, all outputs were manually reviewed, validated, and edited by the authors. The final interpretation of results, the selection of visualization parameters, and the overall conclusions drawn from the data remain the sole responsibility of the project authors.

Links to Chats for the Chatbots:

1. Generative AI Chat for Primary Dataset Content
2. Generative AI Chat for Primary Dataset EDA
3. Generative AI Chat for Backup Dataset EDA
4. Generative AI Chat for Backup Dataset Content

3.6 Special Notes

Given the page constraints, we leveraged Generative AI to streamline our findings and EDA description. We respectfully request that you review the attached BA820_Team_14_Project_Proposal_Notebook.ipynb notebook, which contains our complete primary report. Thank you for your time and consideration. We have uploaded the same notebook in the GitHub repo as well