Exploring the Impact of AI Tools on Stack Overflow: Post-ChatGPT Trends

Sophie Xu

April 30, 2025

1 Introduction

1.1. Background

This report examines the impact of AI-assisted coding tools on coding discussion forums. Large language models (LLM) have rapidly gained popularity and become integrated into daily life. In particular, programmers now frequently use AI-assisted coding tools such as ChatGPT, Gemini Code Assist, and GitHub Copilot in their workflows. The capabilities of these AI tools drew widespread attention when ChatGPT launched in November 2022, reaching over 100 million users within two months (Wikipedia contributors, 2024). To capture the short-term trend of this development, we restricted the analysis timeframe to 2021–2025, allowing for a comparison before and after ChatGPT's release.

1.2 Dataset

The analysis focuses on Stack Overflow's public dataset, available through the Stack Exchange API. Stack Overflow is one of the largest question-and-answer platforms for programmers, with over 23 million users and nearly 60 million posts as of March 2024 (Wikipedia contributors, 2024). The API provides access to contents and interactions on Stack Overflow through entities such as Answers, Badges, and Comments (Stack Exchange, n.d.). For this study, we specifically examine the Posts, Tags, and Users data. Posts include both questions and answers, while Tags are the topics to categorize and organize content (Stack Overflow Teams, 2024). These data offer valuable insights into user engagement, content quality, and the evolving landscape of Stack Overflow over time.

1.3 Objective

To investigate the impact of AI-assisted coding tools on coding discussion forums, we focus on two key questions:

1. Have AI tools reduced reliance on Stack Overflow for coding assistance?
2. How have AI tools influenced the structure and content of forum posts?

By analyzing these aspects, we aim to infer the effects of ChatGPT's launch on Stack Overflow's users and content, providing a preliminary answer to the research question. (See project repository (https://github.com/Sophie-X31/JSC370-Final-Project) for data, code, and additional materials.)

2 Methods

2.1 Acquire Data

The Stack Overflow's public data was retrieved using the Stack Exchange API v2.3. Since anonymous API users have a daily quota limit, we registered for an API key (StackApps, n.d.). As instructed in the documentation, custom filters were created through the API to extract datasets with specific attributes. Once the desired filter was generated as an encoded string, we pass it along with the other parameters, `order`, `sort`, `site`, `fromdate`, `todate`, `page`, and `pagesize` to the API request. To ensure an even distribution of posts from 2021 to 2025, we generated a sequence of start and end dates for each month between 2021-01-01 and 2025-01-01, making separate requests for each month. We fixed pagesize at 100 and merged the retrieved monthly data to create a balanced dataset over the timeframe. For user data, we used a timeframe from 2021-01-01 to 2024-04-01, constrained by API quota limits. For tags, since the Stack Exchange database contains approximately 65,000 tags, we retrieved the entire dataset without filtering. In total, we collected three datasets—posts, users, and tags—amounting to approximately 97 MB. The data was stored as CSV files for easy reuse. The retrieval process was implemented using the `httr` package in R.

2.2 Data Cleaning and Wrangling

To ensure the dataset was in a usable format, we first inspected and summarized each dataset. We found that 93.77% of `location` data in the users dataset and 0.94% of `owner_id` data in the posts dataset were missing, while all other variables had complete data. Given the focus of this analysis, we omitted the `location` variable but retained `owner_id`. Next, we reviewed all variables, removing irrelevant ones and renaming the rest for clarity. Exploratory visualizations revealed highly correlated and duplicate variables, which we also removed. Additionally, we identified two article entries in the posts dataset that were neither questions nor answers and excluded them. Based on the distribution and usage of tags, we decided to retain only the 100 most popular tags for analysis.

We then consolidated variables into new variables tailored to our question. We introduced `lifespan`, calculated as the time gap between creation and last access/modification dates in both the users and posts datasets. Based on the distribution of lifespan, reputation, and total badge counts, we categorized users into three engagement levels: "Expert", "Experienced", and "Normal". Similarly, we classified posts into three engagement levels based on lifespan, vote counts, and score: "High", "Moderate", and "Low". We also introduced a `quality` variable for posts, categorizing them as: "Good", "Normal", "Bad", and "Controversial" based on the total score and the ratio between upvotes and downvotes. Lastly, we joined the users and posts datasets by matching the post owner's user ID, creating a fourth dataset that facilitates the analysis of relationships between users and their posts. The cleaning and wrangling procedures were carried out using R packages including `tidyverse`, `dplyr`, and `data.table`.

2.3 Natural Language Processing (NLP)

The text content of posts was also classified into new categorical variables. We identified error-related keywords (e.g., segmentation fault), actionable language (e.g., fix), and contextual phrases (e.g., ASAP), combining these into a score to categorize a post's `intention` as either debugging or discussion. Similarly, we developed a `complexity` score based on posts' lengths, markdown structure, number of technical terms, and vocabulary diversity, classifying posts as either basic or complex. In addition, we tokenized the post body and assigned one of the 100 most popular tags by matching tokens to tags and selecting the highest-frequency match. To identify whether a post was related to AI, we also applied Latent Dirichlet Allocation (LDA) topic modeling, with methodological details provided in section 2.5.1.

Given the limitations of manual classification, we further utilized several machine learning models—Decision Tree, Random Forest, Bagging, Boosting, and XGBoost—to enhance the classification process. To prepare the post content for these models, we generated word embeddings by first constructing a term co-occurrence matrix, which captured the contextual proximity of words within a fixed window. This matrix served as input for the Global Vectors for Word Representation (GloVe) algorithm, which produced 50-dimensional semantic vectors for each word. Post-level embeddings were then computed by averaging the vectors, creating numeric feature vectors suitable for classification. To accelerate this computationally intensive task, we implemented parallel processing using high-performance computing (HPC) packages. The processed datasets were saved as CSV and RDS files for efficient reuse. This NLP pipeline relied on `stringr`, `topicmodels`, `tokenizers`, and `text2vec`, along with `furrr`, `future`, and `parallel` for distributed processing.

2.4 Data Visualization

For exploratory data analysis and visually evident statistical modeling, we employed a range of visualizations: violin plots, box plots, and barcode plots to depict distributions; scatter plots and correlation heatmaps to demonstrate correlation; bar plots to present rankings; and circle timeline plots, area charts, and calendar heatmaps to visualize change over time. Several of these graphics were made interactive, allowing filtering, selecting, and animating data across time, enhancing interpretability and engagement. The visualizations were developed using R packages including `ggplot2`, `ggcorrplot`, `plotly`, and `gridExtra`. The choice and design of visuals were inspired by Financial Time's Visual Vocabulary (Financial Times Visual Vocabulary, n.d.) and Andy Kriebel's Tableau workbook (Kriebel, n.d.).

2.5 Statistical Modeling

2.5.1 Classification For NLP

As outlined in section 2.3, a range of classification models was used to analyze and categorize post content. A brief overview of each method is provided below:

- **Latent Dirichlet Allocation (LDA)** is an unsupervised topic modeling method that identifies abstract topics in a collection of documents by modeling each document as a mixture of topics and each topic as a distribution over words. It infers these topic structures from a document-term matrix using probabilistic inference.
- **Decision Tree** is a supervised learning method that splits the data into branches based on feature values to make predictions. It works by recursively dividing the dataset using rules that maximize the separation of target classes, forming a tree structure where each leaf represents a final decision or prediction.
- **Random Forest** is an ensemble method that builds multiple decision trees on random subsets of the data and features, and combines their outputs to make more accurate and robust predictions. It reduces overfitting by averaging the predictions of diverse trees.
- **Bagging** is an ensemble technique that improves model stability and accuracy by training multiple models on different random bootstrap samples of the data and aggregating their predictions. The technique is used on the decision trees to mitigate the high variance.
- **Boosting** is another ensemble technique that builds models sequentially, where each new model focuses on correcting the errors made by the previous ones. It combines the outputs of many weak learners to form a strong overall model, often improving performance on difficult prediction tasks. The technique is again used on the decision trees.
- **XGBoost** is a highly efficient and scalable implementation of gradient boosting that builds decision trees sequentially to minimize a regularized loss function. It includes optimizations like tree pruning, parallelization, and regularization to enhance accuracy and prevent overfitting.

LDA was applied to extract latent topics from post bodies to determine whether a post was AI-related. For classifying post characteristics such as engagement, quality, complexity, and intention, we evaluated the performance of Decision Tree, Random Forest, Bagging, Boosting, and XGBoost models. Among these, the Boosting model achieved the best performance, with the lowest Root Mean Squared Error (0.3603) and the highest R-squared value (0.3000). This final model was configured with a Gaussian distribution, 1,000 boosting rounds, a learning rate of 0.5, and five-fold cross-validation. Implementation of these models was supported by the `rpart`, `randomForest`, `gbm`, `xgboost`, and `caret` packages in R.

2.5.2 Model Selection

To address the report's core research questions, we transformed the creation timestamps of posts and users into daily intervals to facilitate time series analysis of posting and user activity trends. Alongside this continuous time predictor, we created a binary categorical variable, `turnpoint`, to indicate whether each observation occurred before or after the launch of ChatGPT (2022-11-01), enabling inferential comparisons. To model changes in post counts over time, we used models based on Negative Binomial distribution with a log link function to account for potential over-dispersion in count data. The following statistical models were considered:

- **Generalized Linear Models (GLMs)** extend traditional linear regression by allowing for non-normal error distributions and using link functions to relate the predictors to the response variable. **Generalized Linear Mixed Models (GLMMs)** build on GLMs by incorporating random effects to model grouped or nested data.

- **Generalized Additive Models (GAMs)** introduce flexibility by applying smooth functions to predictors, allowing the modeling of non-linear relationships. **Generalized Additive Mixed Models (GAMMs)** combine this flexibility with random effects.

To analyze the distribution of users across engagement levels, we treated user proportions as a compositional response and applied **Dirichlet Regression**. This model accommodates multivariate proportion data that sum to one and captures dependencies among components while modeling their relationships with covariates. For modeling the ratio of complex to basic posts, we considered both a Binomial GAM with a log-odds link function and a **Beta Regression** model. The latter offers flexibility in modeling proportional data constrained between 0 and 1. Across all GAM models, we used 30 basis functions (k = 30) to balance smoothness and model flexibility. Model fitting was performed using a suite of R packages, including `mgcv` , `glmmTMB` , `DirichletReg` , and `betareg` .

2.5.3 Test Statistics

Model evaluation was conducted using a range of statistical tests and performance metrics:

- The **Z-statistic**, used in GLMs and GLMMs, tests whether individual model coefficients differ significantly from zero, under the assumption of asymptotic normality.
- The **Chi-square test** is employed in GAMs to evaluate whether adding a smooth term significantly improves model fit by comparing deviance reductions.
- **Adjusted R-squared** quantifies the proportion of variance explained by a model, adjusted for the number of predictors. It penalizes model complexity to prevent overfitting and is calculated as $1 - (\sum_{i=1}^{n}(\hat{y}_i - y_i)^2)/(\sum_{i=1}^{n}(y_i - \bar{y}_i)^2)$.
- For models where traditional R-squared is not defined (e.g., GLMs, GLMMs, Dirichlet and Beta regressions), we used **McFadden's pseudo R-squared**, which compares the log-likelihoods of the fitted model and a null model to assess explanatory power.
- For classification model comparisons, we used **Root Mean Squared Error (RMSE)** to quantify prediction accuracy. RMSE is calculated as $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$.

While most statistics are provided automatically via R's `base::summary()` function, the pseudo R-squared was obtained using the `pscl` package.

3 Results

3.1 Exploratory Data Analysis

3.1.1 Post

Most of the variables measuring engagement and quality in the users and posts dataset exhibit strong skewness. Notably, 45.16% of posts are active for less than 24 hours, while 9.07% are active for more than a week. As illustrated in Figure 1, both distributions have a light, long right tail, emphasizing this skewness.

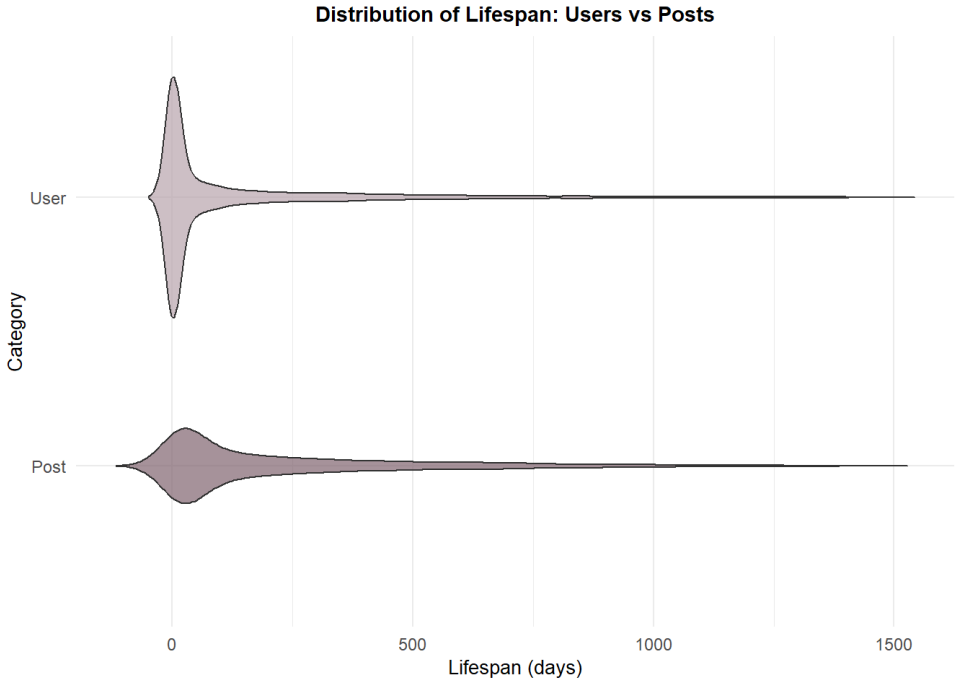

Figure. 1 Violin plots illustrating the lifespan distribution of posts that have been active for over a week (displayed at the bottom) along with the active lifespan of users (shown at the top).

Only 5.65% of posts are controversial, meaning they receive a significant number of both upvotes and downvotes. While one might expect a negative correlation between upvotes and downvotes, Figure 2 suggests otherwise. The line of best fit indicates that posts with more upvotes tend to also receive more downvotes, possibly due to the polarizing nature of controversial posts.

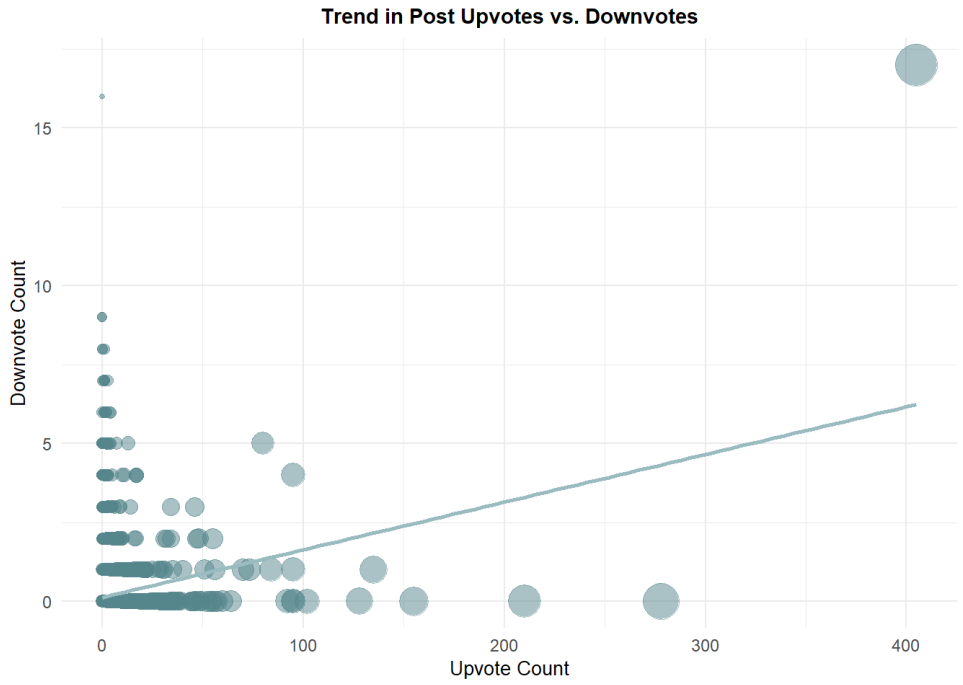

Figure. 2 Scatter plot comparing the number of upvotes and downvotes received by each post, with the size of the points indicating the score (computed as upvotes minus downvotes).

The skewness could be explained by the possibility that a small proportion of basic, broadly useful questions generate high engagement, while the majority of questions are niche-specific and engage only a small subset of users. Additionally, controversial posts may attract high engagement due to their inherently divisive nature.

3.1.2 User

A similar skewed distribution is observed in the users dataset. Users are far more likely to upvote posts as a form of agreement or encouragement, while only 0.04% have ever downvoted a post. Badge counts also reflect this imbalance, up to the third quantile, users hold zero silver or gold badges, as well as zero upvotes or downvotes. User reputation follows an extreme skew as shown in Figure 3, with the minimum, median, and third quantile all valued at 1, while the maximum reaches 4,341.

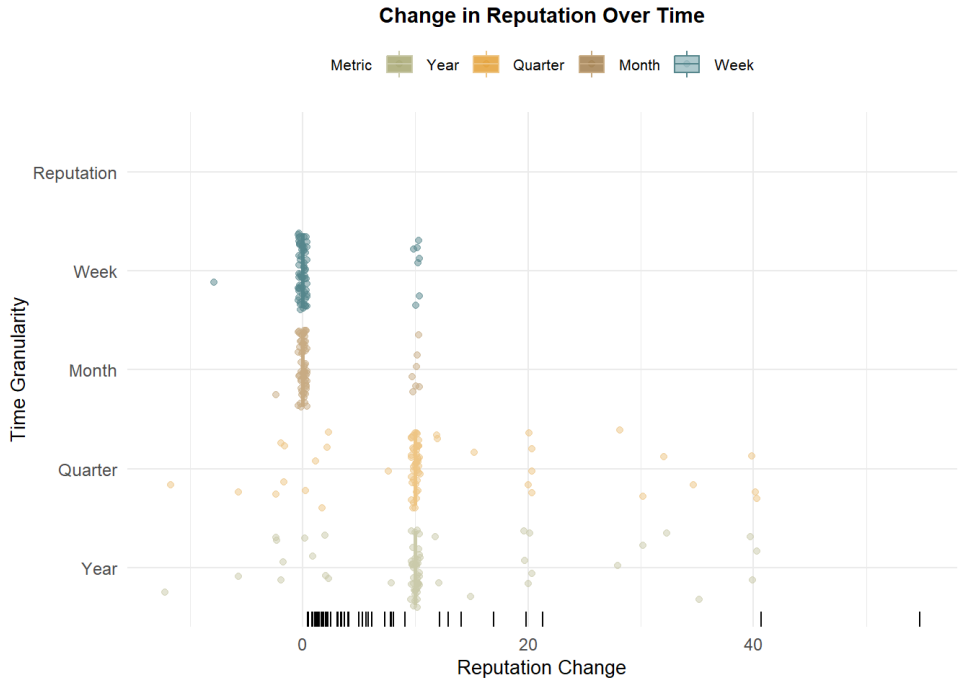

Figure. 3 Circle timelines showing the distribution of changes in reputation across various time intervals, with the total reputation represented as a barcode plot. Outliers (total reputation of 4341 and reputation change of 212) have been omitted from this representation.

Figure 3 shows that the density distribution of change in reputation over months nearly coincides with weekly changes, and similarly, quarterly changes align with yearly trends. This is supported by Figure 4, where the correlation matrix shows that reputation changes over different time

intervals are highly correlated (>0.95). Additionally, reputation is strongly correlated with answer count (0.85) and silver badge count (0.68).

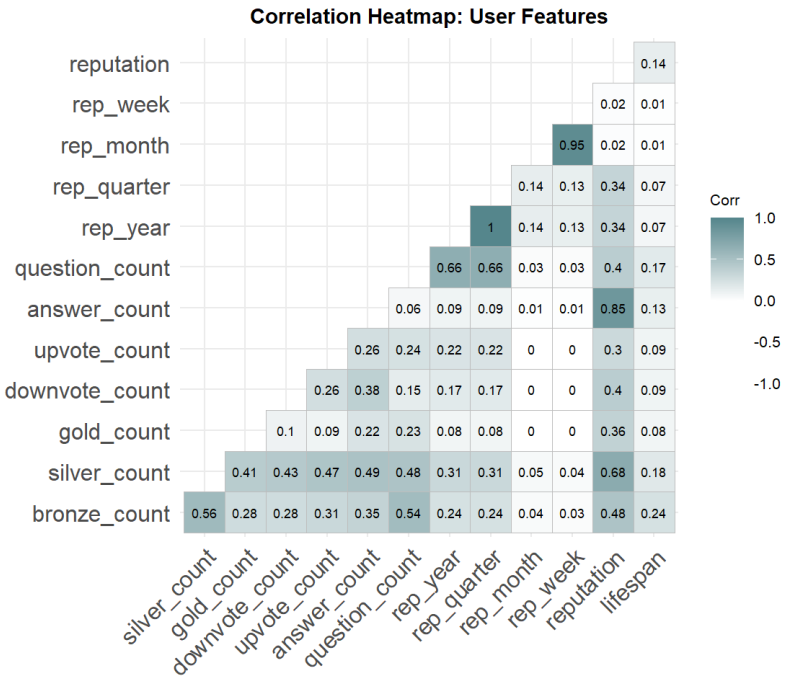

Figure. 4 Correlation heatmap depicting the correlation between different variables within the user data.

These findings suggest that a small percentage of users actively engage in discussions, while the majority passively browse and search for information.

3.1.3 Tag

The tags dataset is straightforward—the top 20 tags correspond to widely used programming languages and concepts. JavaScript ranks first, followed by Python and Java.

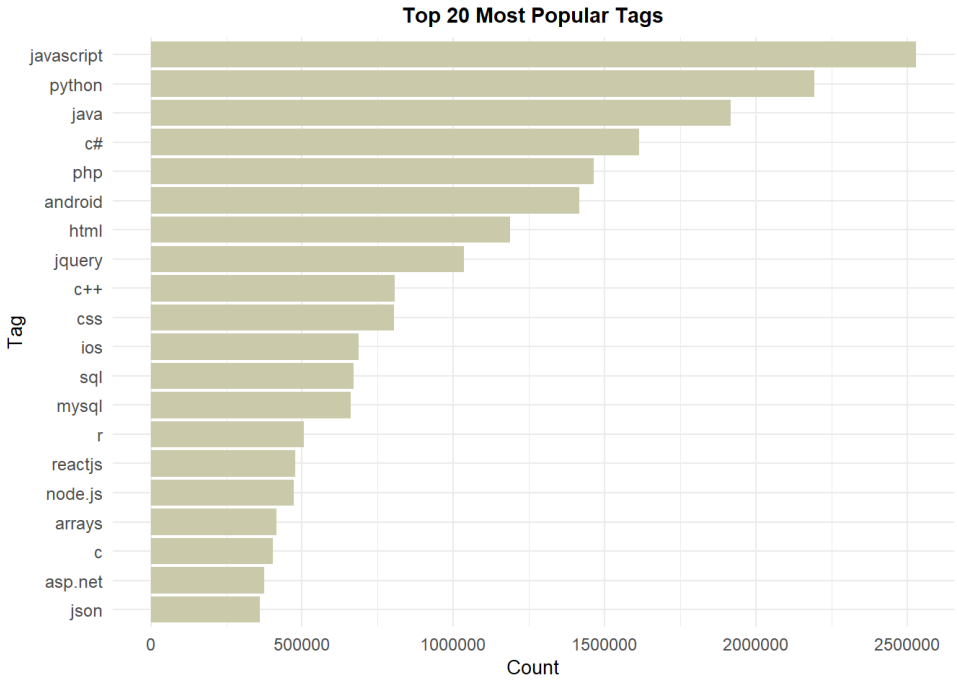

Figure. 5 Bar plot ranking the popularity of tags.

3.1.4 Classification

As outlined in the method section, relying solely on manually defined metrics to classify post content can be limiting. We ran classification with the best-performing boosting model to gain deeper insights. Figure 6 presents the top five most influential features for each classification. Notably, the total number of votes and the active lifespan of a post emerged as strong indicators for predicting engagement level, while the number of downvotes played a significant role in determining content quality. Additionally, word embeddings (`dim_#`) helped distinguish between debugging and discussion posts, and the inferred intention also contributed to identifying its complexity.

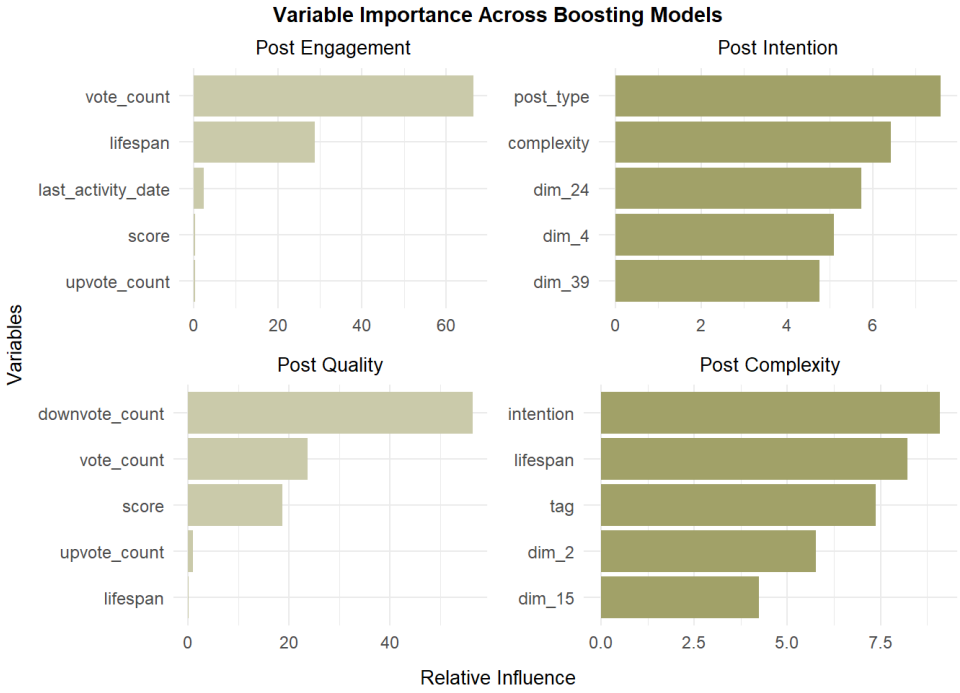

Figure. 6 Variable importance plots highlighting key variables used for classifying posts.

3.2 Statistical Modeling

3.2.1 Impact on Reliance

Recall we are interested in whether ChatGPT's launch reduced programmers' reliance on Stack Overflow. Figure 7 shows a clear decline in basic question posts since 2022. The GAM suggests that time and complexity significantly impacted post counts ($p < 2e-16$), explaining 91.4% of the deviance. Relying on the interaction between `turnpoint` and complexity improved the GLM, achieving a pseudo R-squared of 0.9611 ($p < 2e-16$). These results suggest a decrease in basic coding questions, possibly due to the use of AI tools instead of forums for basic coding help.

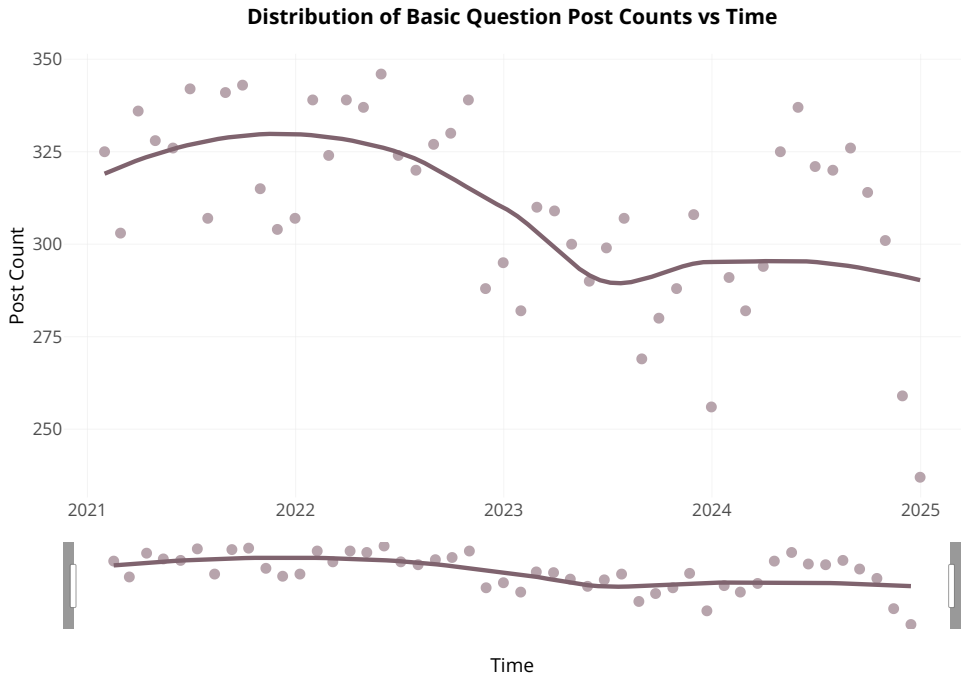

Figure. 7 Scatter plot demonstrating the trend of basic question posts over time. Please use the slider at the bottom to zoom in on specific periods.

A clear trend emerges when examining the complexity of posts. Figure 8 shows a consistent rise in the ratio of complex to basic posts. The Beta regression model indicates that both the time variable ($p \approx 0.0005$) and the `turnpoint` ($p < 2e-16$) significantly influenced this ratio, with the model explaining 71.67% of the deviance. These findings suggest that users are increasingly turning to forums for more complex coding inquiries and responses, possibly leaving simpler tasks to LLM coding assistance.

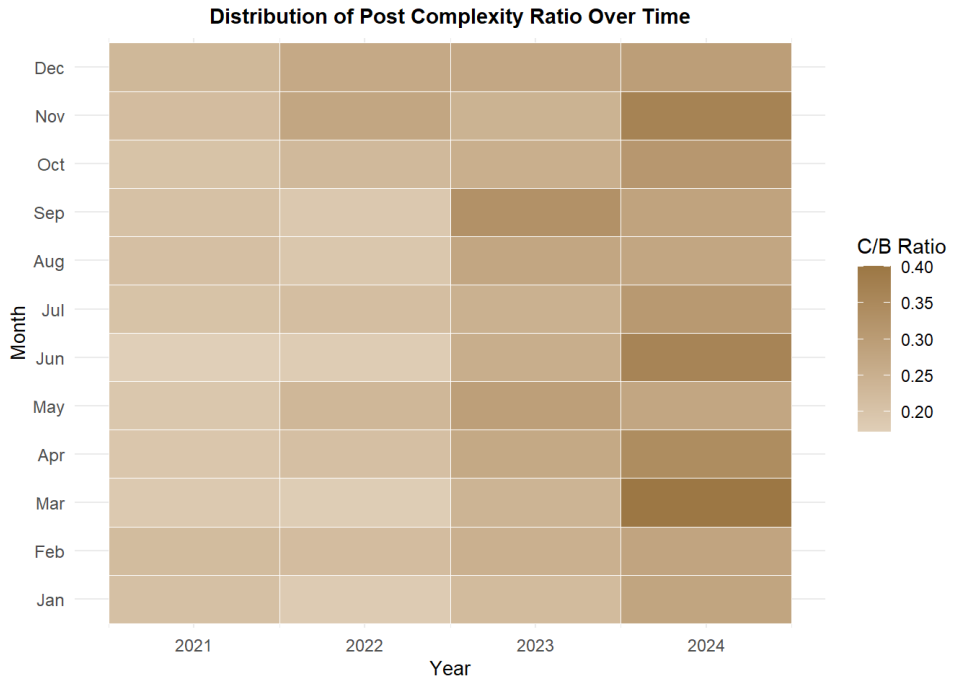

Figure. 8 Calendar heatmap presenting the trend in the ratio of complex to basic posts over time, with the ratio represented as C/B (complex/basic).

We also hypothesized that AI tools might reduce contributions from normal users. As shown in Figure 9, post activity among normal and experienced users rose from 2022 through mid-2023, followed by a noticeable drop, while expert users exhibited a steady decline in posting since 2021. This is contrary to our hypothesis. The GLM found that after ChatGPT's launch ($p < 2e-16$), normal user status ($p \approx 0.018$), and their interaction term ($p \approx 0.0407$) were all significant predictors of post counts. The GAM model explains 69.8% of the deviance, outperforming the GLM (pseudo R^2). Note these results are based on the merged dataset instead of full datasets, which may introduce sample bias due to its smaller size.

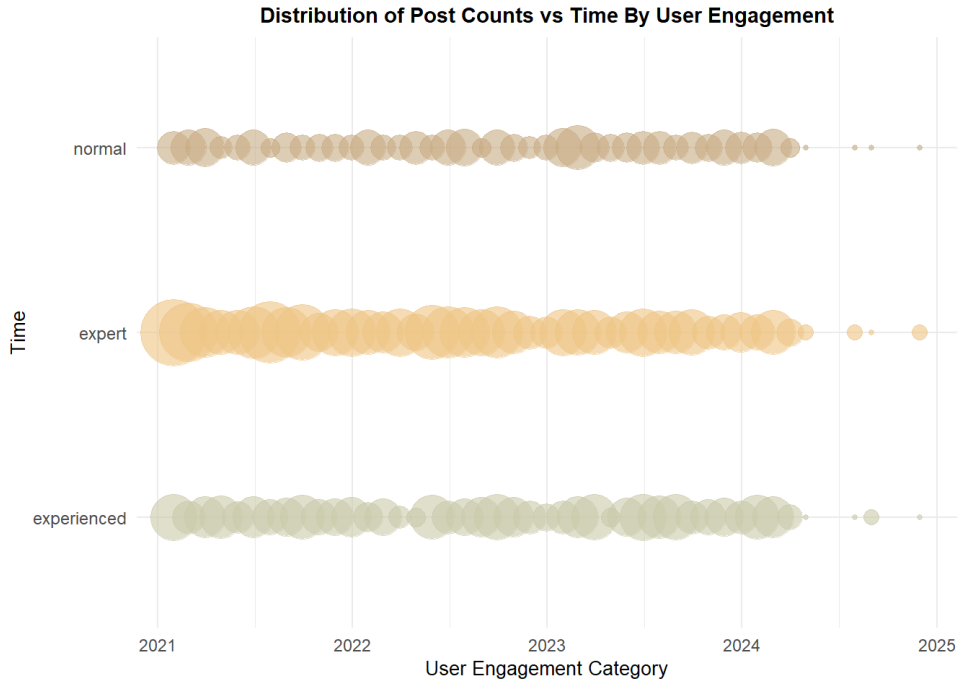

Figure. 9 Circle timelines illustrating the trend in the number of posts over time, influenced by the engagement level of the post owners, with circle sizes indicating the number of posts.

Additionally, we explored whether the composition of user types has shifted. Figure 10 shows the share of experienced and normal users has steadily increased, while the share of expert users has declined drastically. The Dirichlet regression model confirms the significance of `turnpoint` ($p \approx 5.85e-7; p \approx 5.8e-13$) and the time variable ($p \approx 0.0080; p \approx 0.0113$) in impacting experienced and normal user shares, respectively. Before the `turnpoint` also significantly impacted the share of expert users ($p \approx 0.0005$), underscoring the influence of AI tools in reshaping platform participation. The model achieved a strong McFadden's pseudo R-squared of 0.3675.

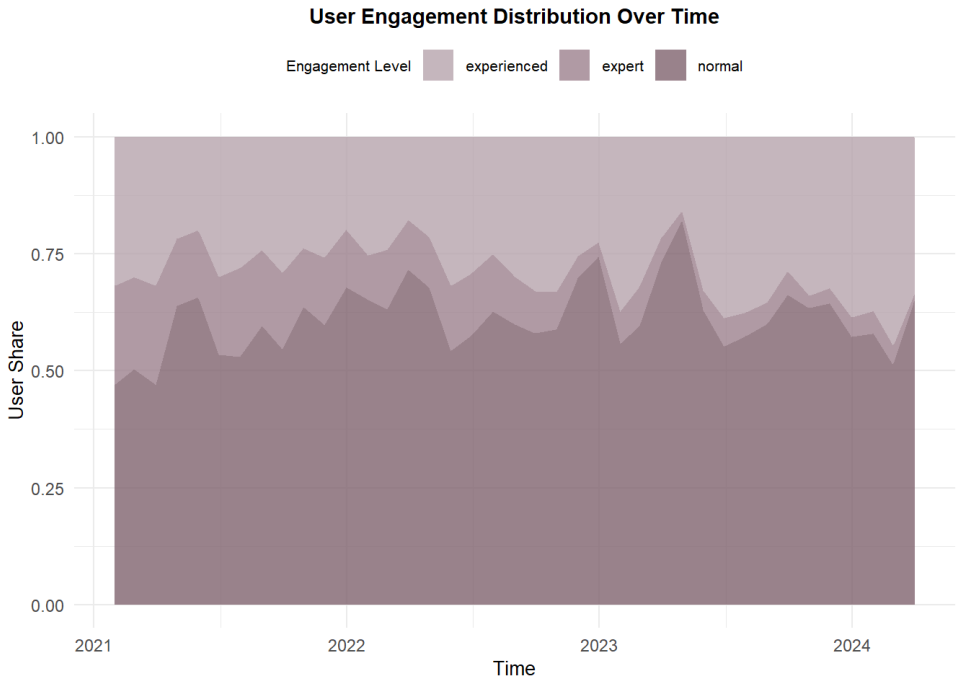

Figure. 10 Area chart showing the change in the proportion of users across different engagement levels over time.

In summary, AI-assisted coding tools appear to have reduced reliance on coding forums among beginner programmers asking basic questions. However, the engagement level of advanced users has declined.

3.2.2 Impact on Content Structure

Beyond engagement and reliance, we are also interested in examining whether LLM coding assistance have influenced the nature of content posted on coding forums. Figure 11 compares debugging and discussion-oriented posts, highlighting that the number of normal-quality debugging posts has slightly increased since 2023 ($p < 2e - 16$), while good-quality discussion posts have declined over the same period ($p < 1.84e - 6$). Other content categories remained relatively unchanged. The GAM including the interaction between post quality and intention explained 97.6% of the deviance in post volume.

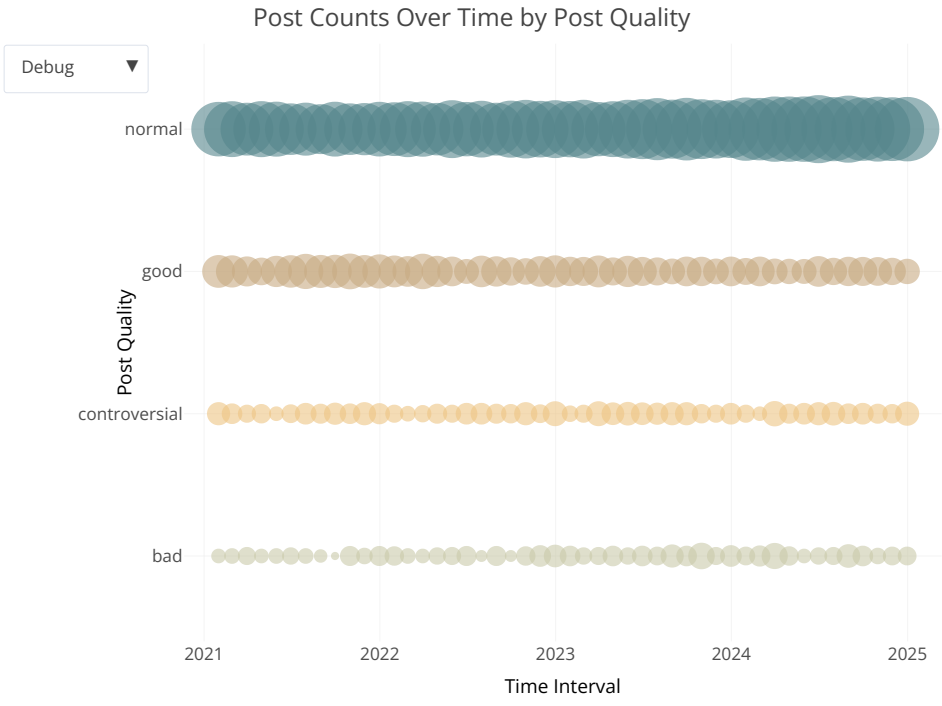

Figure. 11 Circle Timelines reflecting the variation in the number of posts categorized by quality over time. Please use the button to select distributions related to discussion or debug posts.

To further assess how post wording may have evolved, we projected word embeddings of post bodies into two dimensions. While this visualization does not provide direct semantic interpretation, it reveals visible shifts in clustering patterns over time, suggesting structural changes in how users compose their posts.

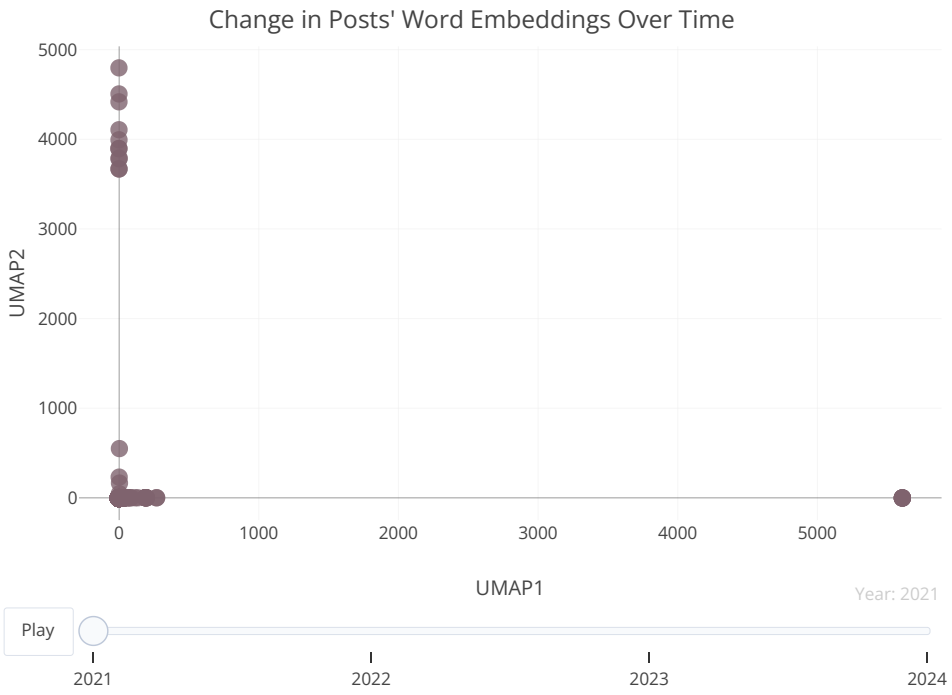

Figure. 12 Scatter plot illustrating the shift in post wording and structure over time based on word embedding projections. Please use the button to display the animated change through time.

Figure 13 demonstrates the examination of the impact of AI tools on posts in the top 10% of popular tags. It exhibits a significant decline in posts related to basic concepts such as lists and functions, while posts tagged with more complex topics like APIs have surged. The GLMM revealed that post timing relative to ChatGPT's launch significantly contributed to the difference in post counts ($p < 2e-16$), while the GAMM emphasized the importance of the random effects of tags ($p < 2e-16$), explaining 86.8% of the deviance. These results suggest that while AI tools are increasingly used for routine coding issues, users continue to seek forum support for more advanced or nuanced programming topics.

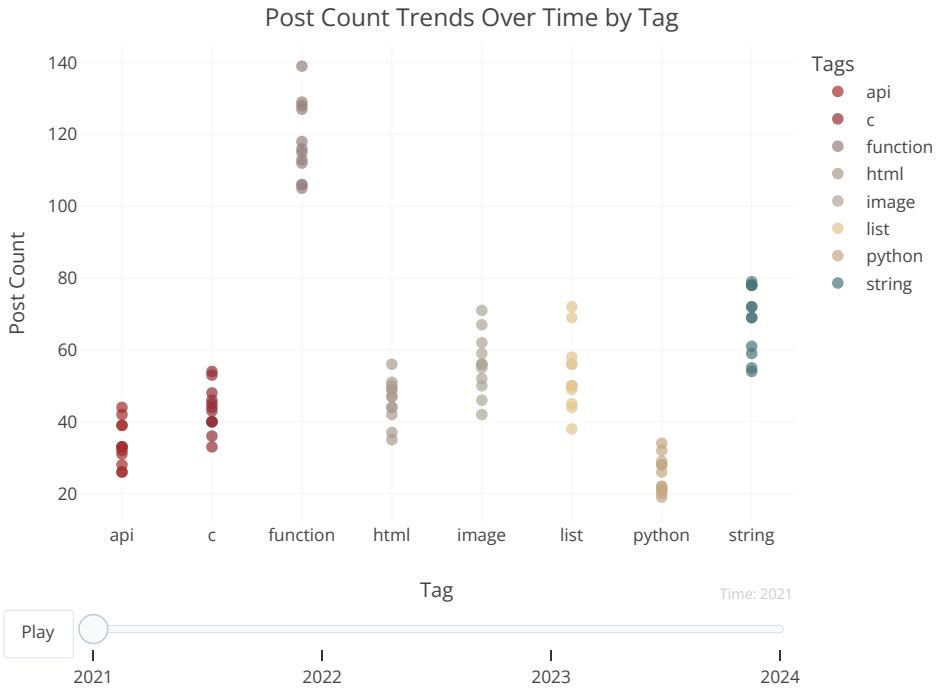

Figure. 13 Scatter plot illustrating the change in post counts associated with the most popular tags over time. Please use the button to display the animated changes through time.

In conclusion, the emergence of AI-assisted coding tools appears to encourage programmers to post more complex questions and discussions, rather than basic inquiries.

4 Summary

4.1 Limitations

While this study provides meaningful insights into the evolving landscape of Stack Overflow usage, several limitations should be considered. We also propose potential directions for improvement:

1. Manual Scoring for Classification

   The classification framework relied on manually defined scoring rules to assess post complexity and intention. Although machine learning models were introduced to mitigate bias, more advanced NLP methods such as zero-shot learning using transformer-based models or label-embedding approaches like Lbl2Vec could yield more accurate and less bias results.

2. Imbalance in AI-Relevant Content Detection

   Topic modeling via LDA was used to identify AI-relevant posts, but only two topics were categorized as such, resulting in a highly imbalanced dataset. Future work should consider collecting a larger volume of posts via the Stack Exchange API and applying more robust classification methods, allowing us to explore AI's influence on content creation as well.

3. Lack of Direct AI Tool Usage Data

   This study inferred the impact of AI tools from trends in posting behavior, but it did not directly measure AI tool usage. To conclude a causal relationship, future research should incorporate data on individual users' interactions with AI tools. Causal inference methods such as randomized controlled trials (RCTs) or quasi-experimental designs could then be employed for more rigorous evaluation.

By addressing these limitations, future studies can strengthen the validity and generalizability of findings, ultimately offering a clearer picture of how AI-assisted tools are transforming engagement and content on coding forums.

4.2 Discussion

Despite its limitations, this analysis offers valuable insights into how AI-assisted coding tools may be influencing engagement on platforms like Stack Overflow. The findings reveal a strong correlation between posting behavior and time, particularly around the release of ChatGPT in November 2022. Key shifts include a noticeable decline in basic question posts, a rising ratio of complex to basic posts, and evolving trends in tag-specific post counts. Interestingly, the decline in user proportion is most pronounced among expert users, while experienced and normal users have increased. These patterns suggest that beginner programmers may be turning to AI tools for quick solutions, reducing their reliance on forums for basic questions. Meanwhile, more advanced users might be using forums to engage in deeper, more complex discussions, potentially benefiting from AI assistance as a complement rather than a replacement. Overall, these trends point to the emergence of AI coding assistance tools, as a likely driving factor behind the observed changes in content and user behavior on coding forums.

References

1. Wikipedia contributors. (2024, March 10). *ChatGPT*. Wikipedia. https://en.wikipedia.org/wiki/ChatGPT (https://en.wikipedia.org/wiki/ChatGPT)
2. Wikipedia contributors. (2024, March 10). *Stack Overflow*. Wikipedia. https://en.wikipedia.org/wiki/Stack_Overflow (https://en.wikipedia.org/wiki/Stack_Overflow)
3. Stack Exchange. (n.d.). *Stack Exchange API documentation*. Retrieved April 21, 2025, from https://api.stackexchange.com/docs (https://api.stackexchange.com/docs)
4. Stack Overflow Teams. (2024, March 10). *Content tags on Stack Overflow*. Retrieved April 21, 2025, from https://stackoverflowteams.help/en/articles/8872158-content-tags (https://stackoverflowteams.help/en/articles/8872158-content-tags)
5. StackApps. (n.d.). *StackApps OAuth registration*. Retrieved April 21, 2025, from https://stackapps.com/apps/oauth/register (https://stackapps.com/apps/oauth/register)
6. Financial Times. (n.d.). *Visual vocabulary*. GitHub. Retrieved April 21, 2025, from https://github.com/Financial-Times/chart-doctor/tree/main/visual-vocabulary (https://github.com/Financial-Times/chart-doctor/tree/main/visual-vocabulary)
7. Berinato, S. (n.d.). *Visual vocabulary* [Tableau dashboard]. Tableau Public. Retrieved April 21, 2025, from https://public.tableau.com/views/VisualVocabulary/VisualVocabulary?%3Aembed=y&%3Adisplay_count=yes&publish=yes&%3AshowVizHome=no (https://public.tableau.com/views/VisualVocabulary/VisualVocabulary?%3Aembed=y&%3Adisplay_count=yes&publish=yes&%3AshowVizHome=no)