ASN3: Visualizing Feature-Review Traceability with Temporal Dimension

Team: Crimson

Course: CS5128

Team Members: Shruti Asolkar, Tharun Swaminathan Ravi Kumar

Introduction

Objective: Visualize traceability between app features and user reviews over time using a fully code-based pipeline.

Motivation: Understand how users respond to new features — especially in the first few weeks post-release.

Target Apps: Zoom/Webex/Firefox

Dataset Overview

Features1.csv: Combined feature list from all apps. Includes title, description, ID, and release date

AppReviews.csv: Aggregated reviews from all 3 apps. Includes review text, date, and MatchedFeatureID

Labeled_Reviews.csv: Manually labeled reviews for training. Used to classify reviews into clusters

Merged_Reviews_With_Features.csv: Merges reviews with feature metadata (release date, title). Adds calculated fields like review period and Within2Weeks

Preprocessed_Sankey_Data.csv: Aggregates number of reviews per Feature → Cluster per Time Filter. Enables temporal interactivity and filtering

Output of Preprocessing

Literature Inspiration

1. **Visual Trace Modeling Approaches**: Cleland-Huang et al. (2012) emphasize visual traceability as key to stakeholder understanding. Our Sankey-style diagram echoes this by intuitively mapping feature-to-review linkages. Inspired the use of nodes and flow strength to reflect volume and directionality of trace links.

2. **Temporal Analysis of Trace Links**: Maalej & Nabil (2015) explored temporal aspects of user feedback on software features. We built on this to show review spikes post-release, as in our detail view with review trend timelines.

3. **Feature Location and Feedback Mining**: Kamei et al. (2016) and Panichella et al. (2013) highlighted the value of mining user reviews to localize software features. This justifies our clustering of reviews and linking to respective features using Sankey paths.

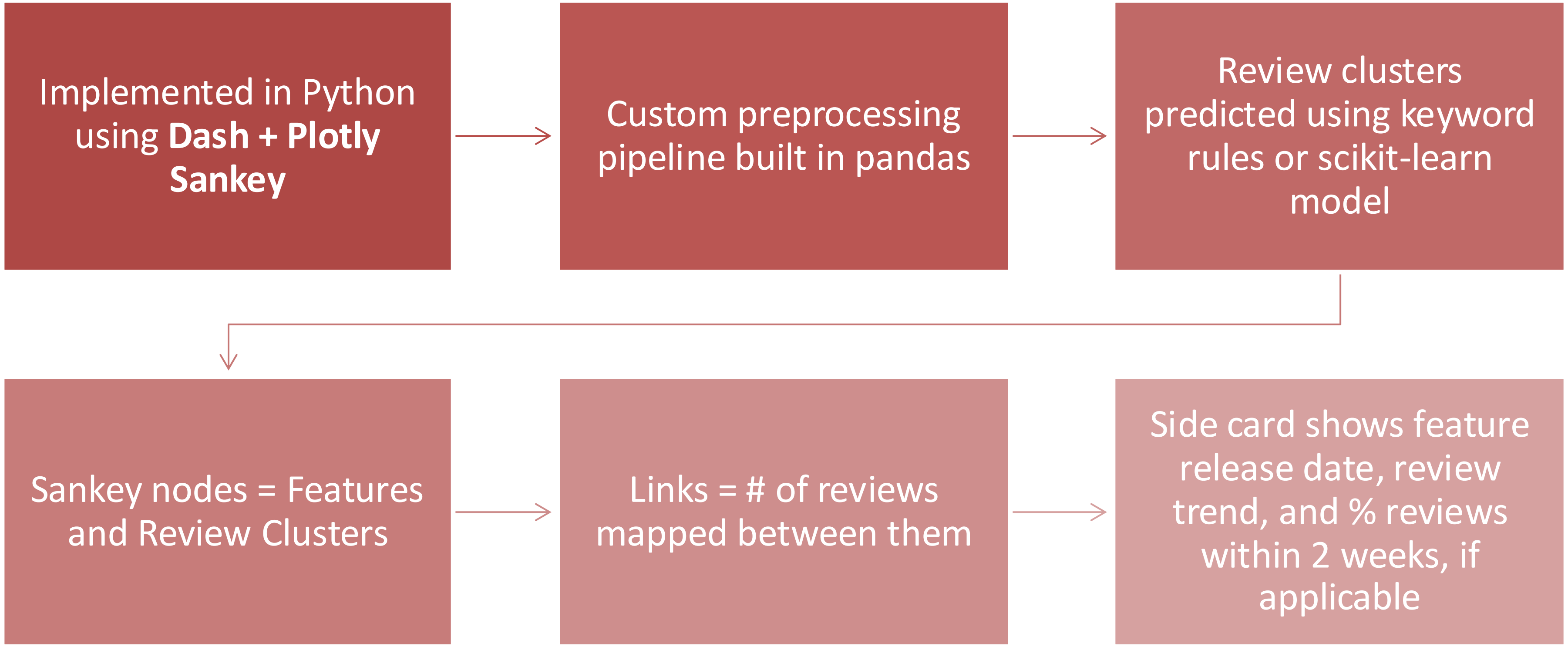
4. **Sankey Diagrams in Software Visualization**: Holten (2006) introduced advanced edge bundling and flow-based diagrams in software visualization. Our approach adapts this for traceability, representing flows from features to review sentiments.

5. **Text Mining + Visualization**: Howard et al. (2017) discussed how combining NLP and interactive visualizations aids decision-making. Our cluster nodes (e.g., “blurry,” “annoying”) stem from similar mined review themes.

Visualization Goals

- Link reviews to features with cluster classification
- Represent when features were released
- Show review volume and sentiment context over time
- Enable dynamic filtering by quarter and review timing
- Provide a side card with feature-level insights on click

Python-Based Sankey Design



Implemented in Python using **Dash + Plotly Sankey**

Custom preprocessing pipeline built in pandas

Review clusters predicted using keyword rules or scikit-learn model

Sankey nodes = Features and Review Clusters

Links = # of reviews mapped between them

Side card shows feature release date, review trend, and % reviews within 2 weeks, if applicable

Temporal Integration

Review Date → Used for filtering (multi-select by quarter)

Feature Release Date → Shown in Feature Card on click

Within2Weeks flag → Helps highlight early user feedback

Review Posted Graph → Shows the graph for reviews posted over time with dates

Novelty and Practical Value

- Fully code-based visualization
- Interactive and deployable — accessible via web
- Supports Scalable Automation
- Feature-level side card combines time, volume, and sentiment
- Helps product teams monitor feature reception
- Valuable for retrospective feature analysis and update prioritization

Implementation Access

- Deployed Visualization Link:

https://sankey-viz.onrender.com/

- GitHub Repo with code + data:
https://github.com/shrutia003/sankey-viz/tree/main

- Instructions to explore the dashboard are in the README.md file:
https://github.com/shrutia003/sankey-viz/blob/main/README.md