CS 3300 Project 2

Team Members:
Elaine Lee (el532)
Pine Wu (zw299)
Rachel Kwak (sk2472)

A description of the data. Report where you got the data. Describe the variables. If you had to reformat the data or filter it in any way, provide enough details that someone could repeat your results. If you combined multiple datasets, specify how you integrated them. Mention any additional data that you used, such as shape files for maps. Editing is important! You are not required to use every part of the dataset. Selectively choosing a subset can improve usability. Describe any criteria you used for data selection. (10 pts)

HN refers to Hacker News: https://news.ycombinator.com/
Which is a popular website, similar to reddit but for technically minded people.
We are investigating how many GitHub stars one project would get if it get posted on Hacker News.

# Data Format

```
{
 "data": [
  {
   "started": "2016-04-22T12:25:00-04:00",
   "end": "2016-04-22T12:30:00-04:00",

   "ghName": "octref/RootIgnore",
   "ghDesc": "Set wildignore from git repo root",
   "ghUrl": "https://github.com/octref/RootIgnore",

   "startStar": 10,
   "endStar": 14,

   "ghStars": [
     {"user": "robot", "time": "2016-04-22T12:25:12-04:00"},
     {"user": "pine", "time": "2016-04-22T12:26:20-04:00"},
     {"user": "rachel", "time": "2016-04-22T12:26:25-04:00"},
     {"user": "elaine", "time": "2016-04-22T12:28:50-04:00"}
   ],

   "hnTitle": "Setting 'wildignore' from git repo root",
   "hnLink": "https://github.com/octref/RootIgnore",
   "hnRanks": [
    {
     "time": "2016-04-22T12:25:00-04:00",
        "rank": 5,
      },
      {
        "time": "2016-04-22T12:30:00-04:00",
        "rank": 2,
      }
    ]
  }
 ]
}
```

# Explanation

- started: When this item first hits HN top 100
- startStar: The amount of GitHub stars at the moment of `started`
- end: When this item drops off HN top 100
- endStar: The amount of GitHub stars at the moment of `end`
- ghStars: A list of stars generated while the HN post is on top 100
- hnRanks: A list of ranks for each HN post while it's on top 100

Other fields should be pretty self-explanatory.

# Source

The data is from two sources:

- https://developer.github.com/v3/
- https://github.com/HackerNews/API

We picked HN submissions which link directly to GitHub projects, and use GitHub API to get a list of all the star creations with timestamps.
As for HN data, we access this endpoint (https://hacker-news.firebaseio.com/v0/topstories.json?print=pretty) every 5 minute and record down the result, aggregate them, and find out which ones link to GitHub projects.

For example, this HN post (https://news.ycombinator.com/item?id=11576527) rose to rank 19 on Hacker News on Apr 26, 19:30, and it links directly to https://github.com/google/xi-editor
We then use this API endpoint (https://api.github.com/repos/google/xi-editor) to get the stars created and associate it with the HN post.

We did this for each HN post created since Apr 24, 9:20AM till tonight.

# Processing

We mainly used lodash, async, moment, request to process the data. All code for data processing live in /crawler/process

# Running the crawler

All scripts and data live in /crawler. To run it, make sure you have Node 4+ and NPM 3+. Get a GitHub token https://github.com/blog/1509-personal-api-tokens and change /crawler/util/token.js accordingly.

```
$ cd crawler
$ npm i
$ cd process
$ node getAllGH.js // Generate allGH.json
$ node getHNRanks // Generate hnRanks.json
$ node getStars // Generate ghStars.json
$ node merge.js // Merge everything and generate data.json
```

The current JSON files are generated Apr 28 afternoon. HN data is collected until Apr, 28th, 13:05. HN raw data is in /crawler/process/data.

B. A description of the mapping from data to visual elements. Describe the scales you used, such as position, color, or shape. Mention any transformations you performed, such as log scales. (10 pts)

The bulk of our data is made up by each project's rank on HackerNews over a period of time and by the timestamp of a newly added star on each project's Github page. We modeled project ranks by drawing a line graph displaying rank vs time for each project, and by plotting points on the line that correspond with each newly added star.

Each rank has a timestamp associated with it. We used a time scale to translate the rank's timestamp to an x-coordinate, and we used a linear scale to translate the rank to a y-coordinate. We drew an svg path using these rank/time pairs for each project.

Each star also has a timestamp associated with the time it was created. We created a point for each star and plotted it on its project's line graph. To do this, we used the same time scale to map the timestamp to an x-coordinate. For the y-coordinate, we created a findRank() function that found its corresponding y-coordinate.

Since our dataset was fairly large, we decided to incorporate features that would highlight selected projects and allow viewers to get details on demand. We achieved this by creating a checkbox list of projects on the sidebar. When a project is selected, the corresponding line graph is highlighted a bright color. We used d3's ordinal scale with a custom domain so that each line is mapped to a different color (although some colors repeat). Selected projects also have the star-points plotted on their line graphs. Otherwise, lines are light grey and with a low opacity, so they won't clutter the graph.

When you hover over a selected graph, there two possible results. If you click on the Main tab, then details about that project will pop up. This includes some information about the project, including title, description, and link to the project's Github page. If you hover over a point on the graph, more details about the project appear, namely the project's rank and total number of stars at that timestamp. We also included a "progress bar" that displays the growth in number of stars from the last timestamp. The progress bar also compares the current number of stars against the initial number of stars that the project had before we scraped our data (ideally, before the project was posted on HackerNews). We built this progress bar by linearly scaling the number of stars against the max width of the bar, and transitioning the bar width to that scaled number.

If you click on the Cluster tab, then clusters will appear on all selected projects. These clusters are the same color as their corresponding line, with 80% opacity so they don't obstruct the rest of the graph. When you hover over a cluster, then data about that cluster will pop up. We decided to include average rank and average timestamp of the points in that cluster, as well as the number of points in the cluster.

We also provided functionality for viewers to zoom in on the graph by adding a brush component. This was heavily inspired by Mike Bostock's Focus+Context visualization. The brush is superimposed over a smaller version of the graph that provides an overview of the graph. The domain of the graph's x-axis corresponds with the brush's selection. Viewers can drag to brush around to inspect a different time interval, and they can also increase/decrease the size of the brush to change the size of the time interval. As a bonus, we also highlighted selected lines in the graph overview, so viewers can see which lines are highlighted if the brush doesn't cover those lines.

Lastly, we added tooltip that displays the project's name when you hover over its line (whether it is selected or not). The tooltip slowly fades out if the cursor isn't hovered over an element in the graph, whether it's a line or a point. This allows viewers to know which line corresponds to which project.

C. The story. What does your visualization tell us? What was surprising about it? (5 pts)

Before we built the visualization, we hypothesized that rate of star increase for GitHub repositories posted on Hacker News with a higher rank would be greater than when it is at a lower rank. Our reasoning was that since Hacker News is a well-known social news website built by a reputable start up incubator, Y Combinator, people would take more interest in GitHub repositories that are more highly ranked.

We also hypothesized that there would be specific times where there would be the greatest increase in the number of stars. This is why we had the clustering function, because we thought that if we could see the different times at which there were clusters, we could say that the times with the greatest number of clusters are the ideal times for repositories to gain stars.

After collecting the data, we could clearly see that some projects conformed well with our hypothesis. For projects such as cisco/ChexScheme, we can see that the repository gained a great amount of stars when it had a high rank, with its stars rapidly growing from 23 to 1946 stars. As time went on, the rate of star increase decreased. It is reasonable to conclude here that there is a correlation between the rank on Hacker News and the number of stars it gained.

Our hypothesis, however, was not 100% accurate for all the GitHub repositories. For some repositories that accumulated a large number of stars when it was at a high rank, they continued to accumulate large number of stars even when its rank on Hacker News dropped to below 80. Some examples of such repositories are google/agera, Yelp/elastalert, graphitemaster/moreram, and CALEBMER/POSTGRAPHQL. This may be because the increase in stars while it was popular on Hacker News helped spread awareness of the repository even more and thus made it popular even after its rank on Hacker News dropped. An alternative reason is that the repository may be posted on another similar site such as Reddit, and this may have generated more stars.

On other repositories, there did not seem to be a correlation on the number of stars gained and its rank on Hacker News. The repository reverse-shell/routersploit seemed to be pretty popular on GitHub regardless of its increase and decrease in ranks on Hacker News. These projects most likely were previously well known and were featured on multiple different sites, allowing more people to know about the repository and give the repository stars.

We also found out that having a high rank on Hacker News does mean that the repository will gain more stars. Repositories with lower ranks rarely had increases in its star count, as seen with amznlabs/ion-java, wheresrhys/server-push-polyfill-demo, and trueadm/inferno.

Another thing that was interesting about each repository was that its rank on Hacker News dropped more rapidly than we thought. We were expecting more of a curve with ups and downs, but the graph rarely increased its rank and in general had a trend downwards.

The clusters did not seem to be present on any specific times. We speculate that this is because Hacker News is a website read by people on various parts of the US, and even around the world. Because of this, people are awake at different periods of time during the day and thus it makes it difficult to conclude a specific time as the best for gaining stars.

The greatest takeaways for this graph is that having a high rank in Hacker News is always good for getting more stars on Github. For lesser known projects that get exposed on Hacker News, it is clear that being on Hacker News greatly increased its number of stars. For already established projects, it seems that Hacker News did help, but was largely popular just on its own.

Sources:
http://www.d3noob.org/2013/01/adding-tooltips-to-d3js-graph.html
https://bl.ocks.org/mbostock/1667367