

Shino Thomas

TYBSc IT 32

**2305043**

Assignment: CIA 1

Data Analysis Using

**Shell Scripting and Linux Commands.**

# Introduction

This project focuses on analyzing unstructured Git log data extracted from the [Microsoft VS Code](https://github.com/microsoft/vscode) open-source repository. VS Code is one of the most popular code editors worldwide, and its Git history holds valuable insights about the development process, contributions, and collaboration patterns over time.

Reason for selecting vscode git log data:

The VS Code Git log data, sourced from one of the most active open-source projects on GitHub, serves as a rich real-time dataset for practical analysis. Originally unstructured, this commit history was converted into a structured .csv format using Linux commands and shell scripting. This project illustrates the end-to-end process of handling real-world data right from extraction and transformation to visualization and insightful analysis entirely through command-line tools.

## Methodology

The methodology of this project involves the complete end-to-end pipeline of transforming unstructured real-time data into a structured format, followed by analysis and visualization all using Linux shell scripting and command-line tools.

### 1. Data Extraction

The Git log data was extracted directly from the official Microsoft VS Code open-source GitHub repository - <https://github.com/microsoft/vscode>.

This was done using the command:

```
Git clone https://github.com/microsoft/vscode
```

```
git log > vscode_gitlog.txt
```

This resulted in an unstructured text file (vscode\_gitlog.txt) containing details such as:

- Commit Hashes
- Author Names

- Commit Dates
- Commit Messages

## 2. Data Preprocessing

Since the Git log was unstructured, command was employed to parse and clean the data. The relevant fields were extracted and formatted into a CSV structure.

```
shinothomas@pop-os:~/vscode$ git log >vscode_gitlog.txt
shinothomas@pop-os:~/vscode$ git log --pretty=format:'%h,%an,%ad,%s' --date=short > vscode_gitlog.csv
shinothomas@pop-os:~/vscode$ mv vscode_gitlog.txt vscode_gitlog.csv ~/
shinothomas@pop-os:~/vscode$ mkdir vscode-gitlog-analysis
```

This created vscode\_gitlog.csv

## 3. Data Analysis

The structured CSV file was analyzed using command-line utilities such as sort, uniq, grep, cut, and wc. Key analytical outputs included:

1. Top 5 Contributors by Number of Commits
2. Most Active Days of the Week
3. Monthly Commit Trends
4. Commits per Author (Sorted Alphabetically)
5. Commits per Year
6. Total Number of Fix-Related Commits
7. Git Commit Hashes
8. Commits Mentioning 'Hello'
9. Total Number of Commits
10. List of Unique Commit Messages

## 4. Data Visualization

For Visualization, I used gnuplot to visualize the processed data into charts and graphs.

- Bar charts
- Line graphs
- histograms

# Commands

## Some keywords that I have used in almost all the analysis:

- `sort` It sorts the data in the data set.
- `uniq -c` Count how many times each instance has occurred. It's basically 'group by'.
- `sort -nr` used to sort by count.-nr to arrange the data in reverse order.
- `head` Shows only the top lines by default, the first 10. If interested in only top 5 contributors, use `head -n 5`.

- `> file_name`

`(some_command) > file_name.dat`

Stores the output retrieved from the command to the file named 'file\_name.dat'

## General explanation of all the gnuplot scripts that I have used:

- `set terminal png size 800,600`

This command sets the output format to a PNG image with a resolution of 800×600 pixels.

- set output 'output\_file.png'

Defines the name of the image file to be generated (e.g., 'output\_file.png').  
Change this to any desired filename.

- set style data <plot\_type>

Specifies the chart type to be plotted. You can replace <plot\_type> with:

- ❖ histogram – for bar charts
- ❖ lines – for line charts
- ❖ linespoints – for line chart with points
- ❖ boxes – for box plots
- ❖ dots, points, etc. – for scatter-type visuals

- set style fill solid 1.00 border -1

Fills bars or boxes with a solid color and removes borders. (Mainly used with histogram or boxes.)

- set boxwidth 0.6

Sets the width of bars (when applicable). You can adjust this for visual spacing in bar/box charts.

- set title "Your Chart Title"

Adds a title to your plot. Replace with a suitable heading depending on your data.

- `set xlabel "x_name"`
- `set ylabel "y_name"`

Labels the X-axis and Y-axis. Replace "x\_name" and "y\_name" with the appropriate descriptions for your data.

- `set xtics rotate by -45`

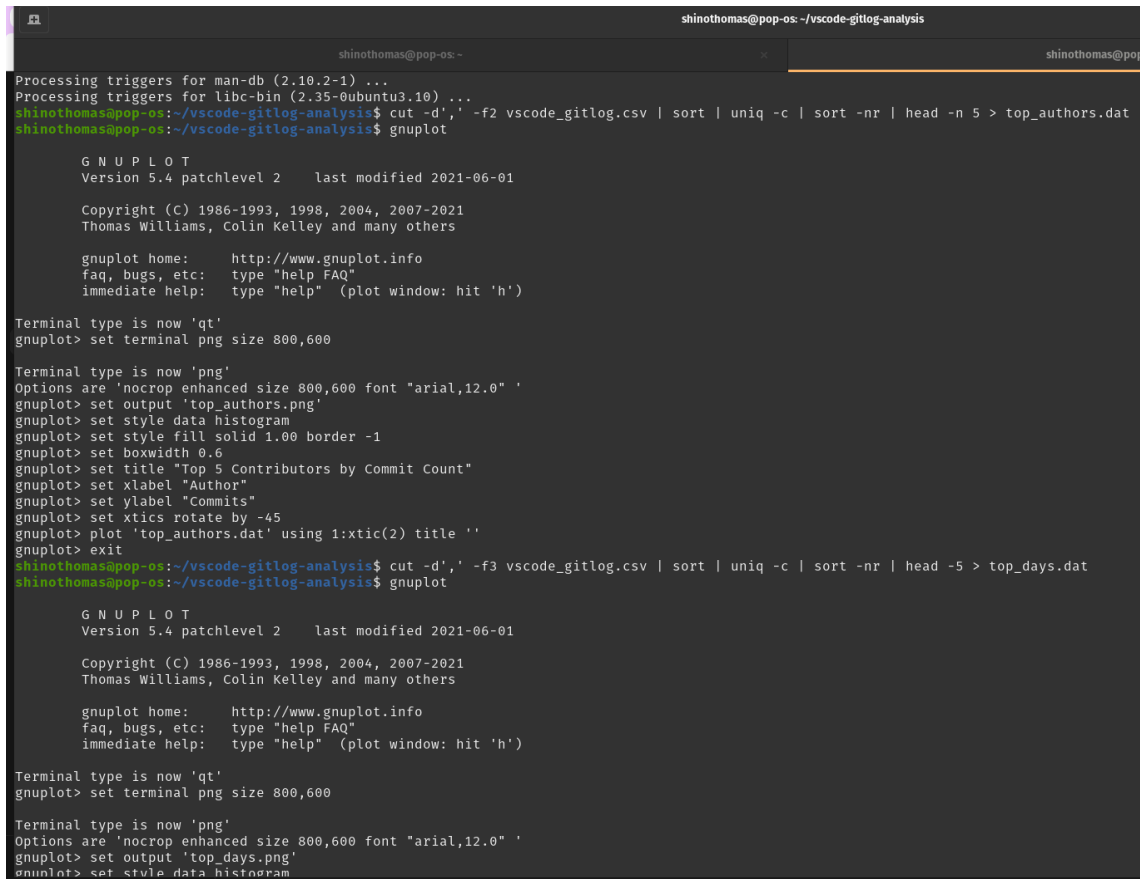
Rotates the X-axis labels (useful when labels are long or overlapping).

- `plot 'datafile.dat' using 1:xtic(2) title "`

Plots the data from your file:

- Column 1 is used for the Y-axis value (e.g., commit count)
- Column 2 provides the X-axis labels (e.g., names)

# Explanation



```
shinothomas@pop-os: ~/vscode-gitlog-analysis
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.10) ...
shinothomas@pop-os:~/vscode-gitlog-analysis$ cut -d',' -f2 vscode_gitlog.csv | sort | uniq -c | sort -nr | head -n 5 > top_authors.dat
shinothomas@pop-os:~/vscode-gitlog-analysis$ gnuplot

G N U P L O T
Version 5.4 patchlevel 2    last modified 2021-06-01

Copyright (C) 1986-1993, 1998, 2004, 2007-2021
Thomas Williams, Colin Kelley and many others

gnuplot home:      http://www.gnuplot.info
faq, bugs, etc:    type "help FAQ"
immediate help:    type "help" (plot window: hit 'h')

Terminal type is now 'qt'
gnuplot> set terminal png size 800,600

Terminal type is now 'png'
Options are 'nocrop enhanced size 800,600 font "arial,12.0" '
gnuplot> set output 'top_authors.png'
gnuplot> set style data histogram
gnuplot> set style fill solid 1.00 border -1
gnuplot> set boxwidth 0.6
gnuplot> set title "Top 5 Contributors by Commit Count"
gnuplot> set xlabel "Author"
gnuplot> set ylabel "Commits"
gnuplot> set xtics rotate by -45
gnuplot> plot 'top_authors.dat' using 1:xtic(2) title ''
gnuplot> exit
shinothomas@pop-os:~/vscode-gitlog-analysis$ cut -d',' -f3 vscode_gitlog.csv | sort | uniq -c | sort -nr | head -5 > top_days.dat
shinothomas@pop-os:~/vscode-gitlog-analysis$ gnuplot

G N U P L O T
Version 5.4 patchlevel 2    last modified 2021-06-01

Copyright (C) 1986-1993, 1998, 2004, 2007-2021
Thomas Williams, Colin Kelley and many others

gnuplot home:      http://www.gnuplot.info
faq, bugs, etc:    type "help FAQ"
immediate help:    type "help" (plot window: hit 'h')

Terminal type is now 'qt'
gnuplot> set terminal png size 800,600

Terminal type is now 'png'
Options are 'nocrop enhanced size 800,600 font "arial,12.0" '
gnuplot> set output 'top_days.png'
gnuplot> set style data histogram
```

This screenshot is of the terminal running the commands for extracting Top 5 Contributors by Number of Commits and Most Active Days of the Week.

Below each we have their respective gnuplot commands for plotting the graph.

## 1. Top 5 Contributors by Number of Commits

`cut -d',' -f2 vscode_gitlog.csv | sort | uniq -c | sort -nr | head -5 > top_authors.dat`

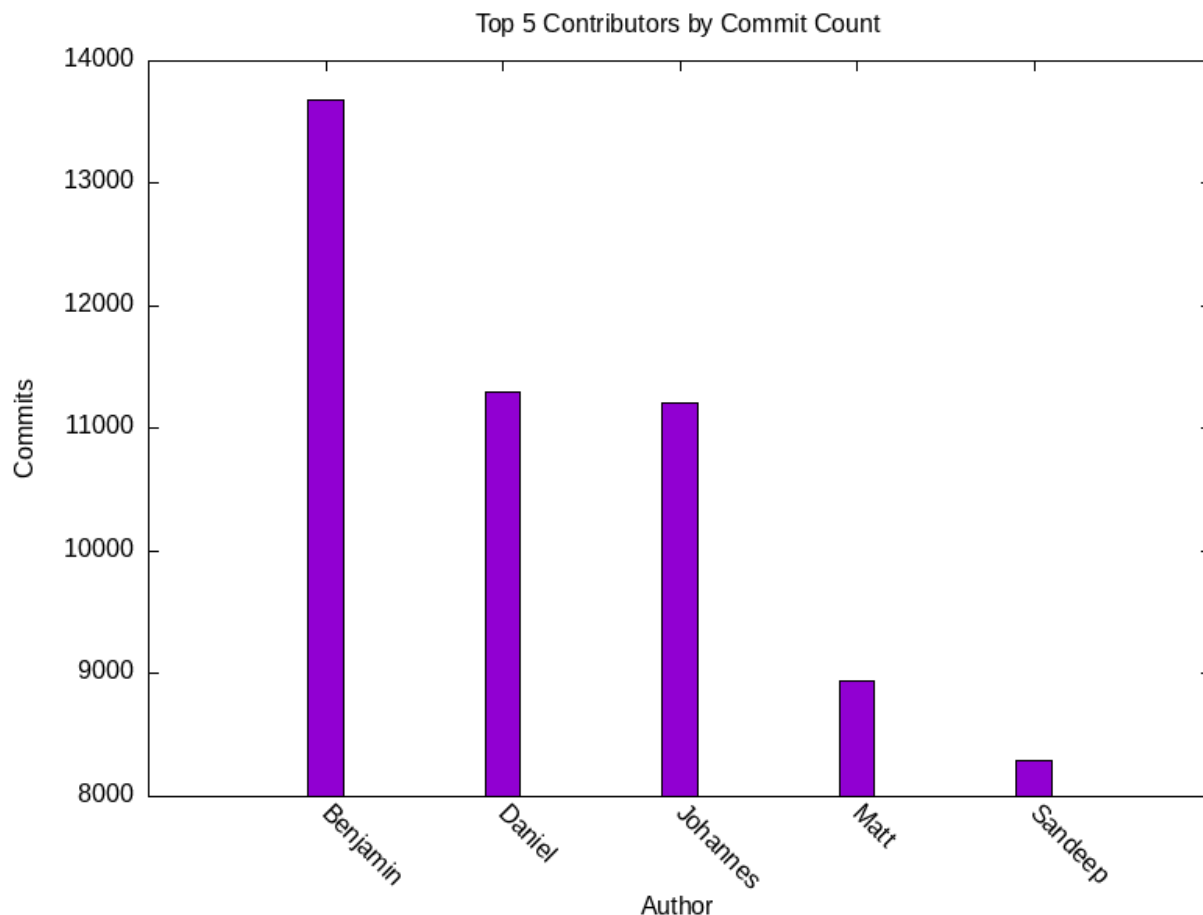
- `cut -d',' -f2`: Extracts the 2nd column (author names).
- `sort`: Sorts names so `uniq -c` can group them.
- `uniq -c`: Counts how many times each name appears.
- `sort -nr`: Sorts by count (descending).

- head -5: Shows top 5 authors.
- >: Redirects output to top\_authors.dat.

Gnuplot already explained in the general explanation. The ONLY difference is histogram will be given as the parameter and the graph is filled with solid color.

```
set style data histogram
```

```
set style fill solid 1.00 border -1
```



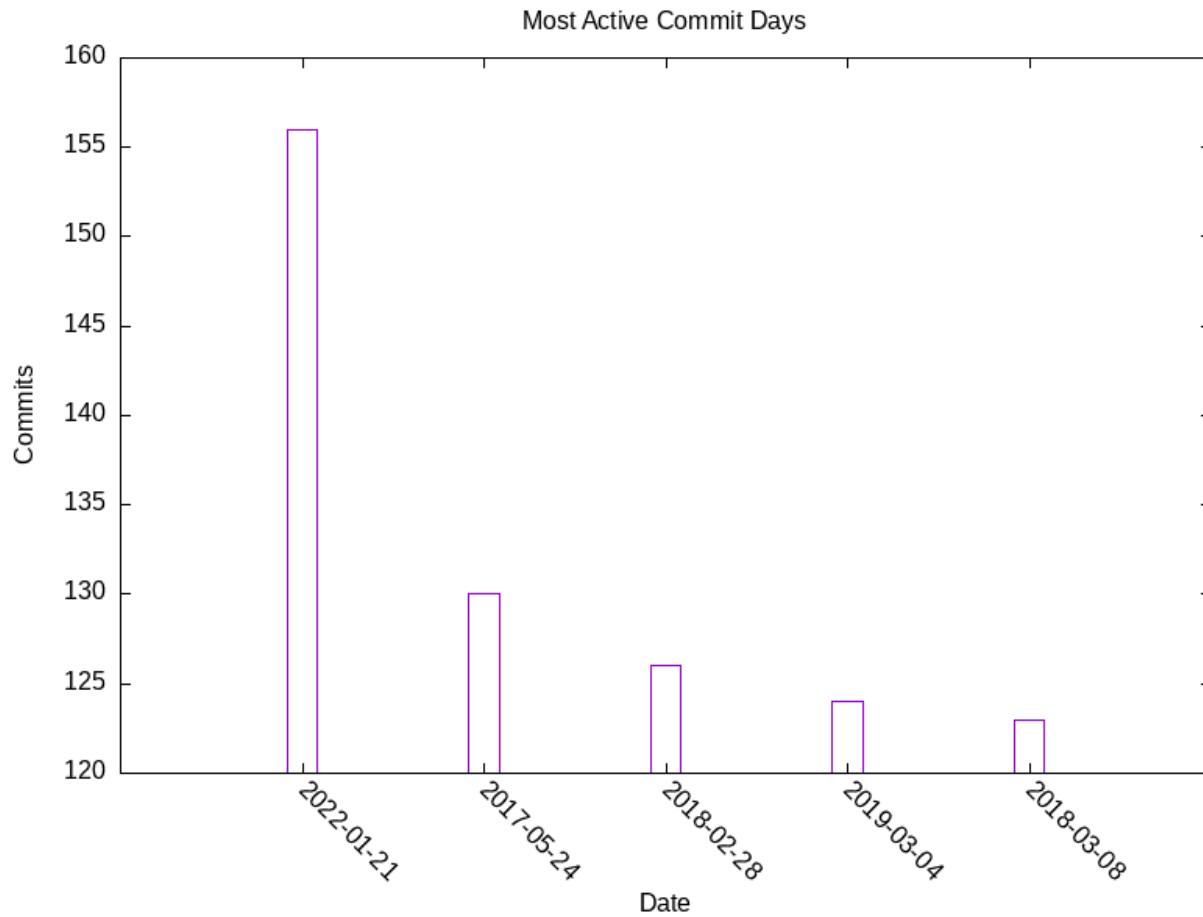
## 2. Most Active Days of the Week

```
cut -d',' -f3 vscode_gitlog.csv | sort | uniq -c | sort -nr | head -5 > top_days.dat
```

- Extracts 3rd column (dates).
- Count how often each date appears.
- Sorts descending and picks top 5 dates.



Gnuplot already explained in the general explanation. The ONLY fill parameter is not given, hence it shows a hollow histogram.



```
Workspaces Applications Aug 7 6:05 PM
shinothomas@pop-os: ~/vscode-gitlog-analysis
shinothomas@pop-os: ~
gnuplot> plot 'top_days.dat' using 1:xtic(2) title ''
gnuplot> exit
shinothomas@pop-os:~/vscode-gitlog-analysis$ cut -d',' -f3 vscode_gitlog.csv | cut -c1-7 | sort | uniq -c | sort -k2 > commits_per_month.dat
shinothomas@pop-os:~/vscode-gitlog-analysis$ gnuplot

      G N U P L O T
      Version 5.4 patchlevel 2      last modified 2021-06-01

      Copyright (C) 1986-1993, 1998, 2004, 2007-2021
      Thomas Williams, Colin Kelley and many others

      gnuplot home:      http://www.gnuplot.info
      faq, bugs, etc:    type "help FAQ"
      immediate help:    type "help" (plot window: hit 'h')

Terminal type is now 'qt'
gnuplot> set terminal png size 1000,400

Terminal type is now 'png'
Options are 'nocrop enhanced size 1000,400 font "arial,12.0" '
gnuplot> set output 'commits_per_month.png'
gnuplot> set xdata time
gnuplot> set timefmt "%Y-%m"
gnuplot> set format x "%Y-%m"
gnuplot> set title "Commits Per Month"
gnuplot> set xlabel "Month"
gnuplot> set ylabel "Commits"
gnuplot> set grid
gnuplot> plot 'commits_per_month.dat' using 2:1 with linespoints title 'Commits'
gnuplot> exit
shinothomas@pop-os:~/vscode-gitlog-analysis$ cut -d',' -f2 vscode_gitlog.csv | sort | uniq -c | sort -k2 > author_commits.dat
shinothomas@pop-os:~/vscode-gitlog-analysis$ set terminal png size 1000,600
set output 'author_commits.png'
set style data histograms
set style fill solid 1.0 border -1
set boxwidth 0.6
set title "Commits by Author"
set xlabel "Author"
set ylabel "Commits"
set xtics rotate by -45
plot 'author_commits.dat' using 1:xtic(2) notitle
bash: syntax error near unexpected token `('
shinothomas@pop-os:~/vscode-gitlog-analysis$ gnuplot

      G N U P L O T
      Version 5.4 patchlevel 2      last modified 2021-06-01

      Copyright (C) 1986-1993, 1998, 2004, 2007-2021
      Thomas Williams, Colin Kelley and many others
```

This screenshot is of the terminal running the commands for extracting Commits per month and Commits per Author (Sorted Alphabetically).

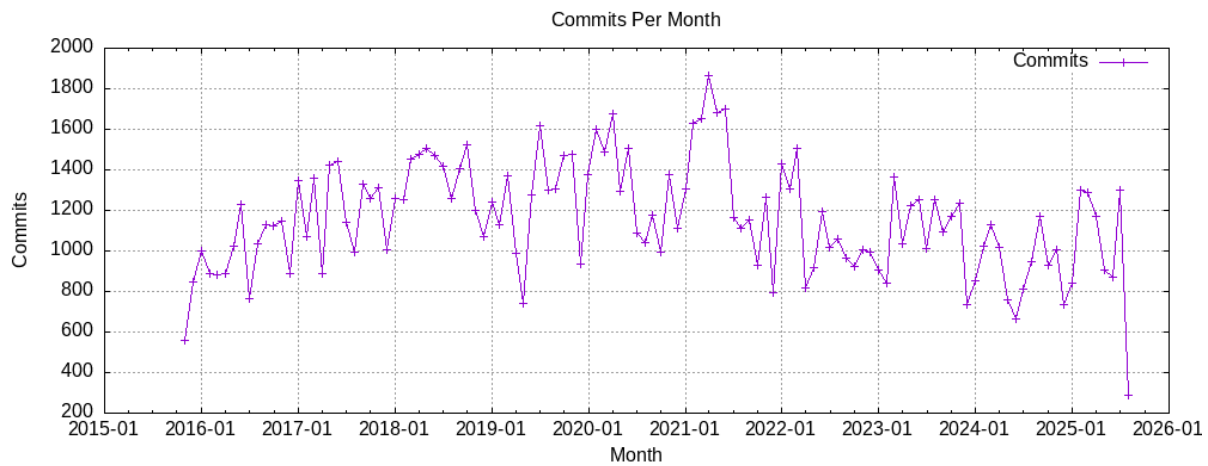
Below each we have their respective gnuplot commands for plotting the graph.

### 3.Commits per month

```
cut -d',' -f3 vscode_gitlog.csv | cut -c1-7 | sort | uniq -c | sort -k2 >
commits_per_month.dat
```

- Extracts date (3rd column).

- `cut -c1-7`: Extracts YYYY-MM.
- Counts how many commits per month.

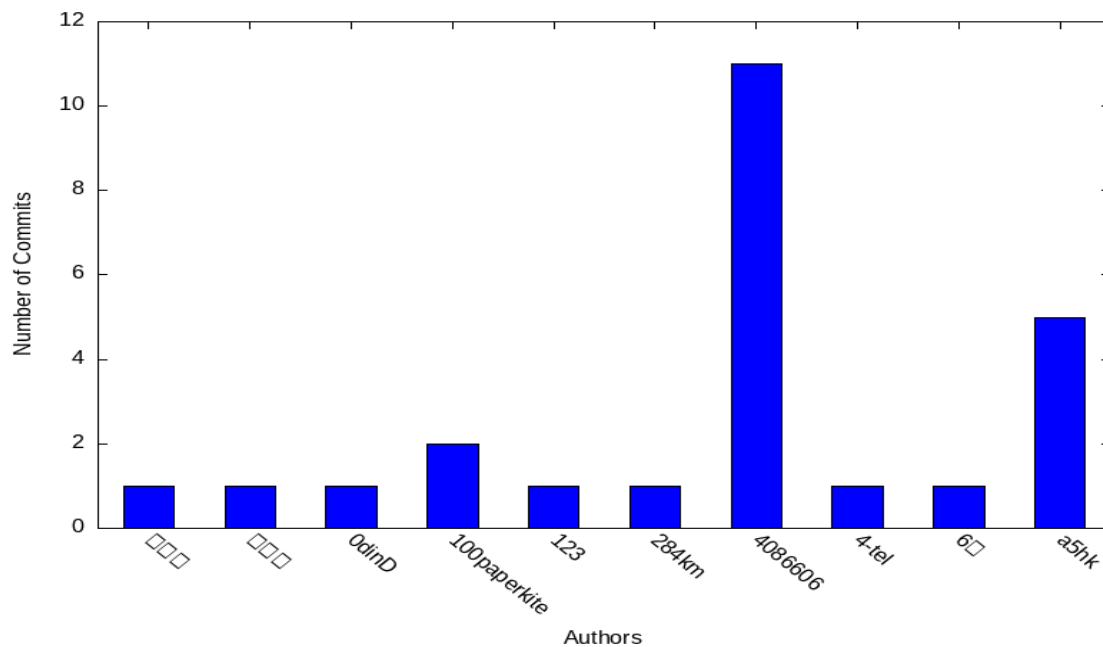


#### 4. Commits per Author (Sorted Alphabetically).

```
cut -d',' -f2 vscode_gitlog.csv | sort | uniq -c | sort -k2 > author_commits.dat
```

(Similar keywords that were used in the previous commands)

For plotting used only the first 10 rows of analysed data for better visualizations.



There were a few names with special characters hence they are denoted by squares in the X label.

```
shinothomas@pop-os:~/vscode-gitlog-analysis$ cut -d',' -f3 vscode_gitlog.csv | cut -c1-4 | sort | uniq -c
1403 2015
12003 2016
14564 2017
16283 2018
14851 2019
15735 2020
16257 2021
13131 2022
13129 2023
11059 2024
7966 2025
3 Abh
1 Max
1 Sei
shinothomas@pop-os:~/vscode-gitlog-analysis$ cut -d',' -f3 vscode_gitlog.csv | cut -c1-4 | sort | uniq -c | sort -nr
16283 2018
16257 2021
15735 2020
14851 2019
14564 2017
13131 2022
13129 2023
12003 2016
```

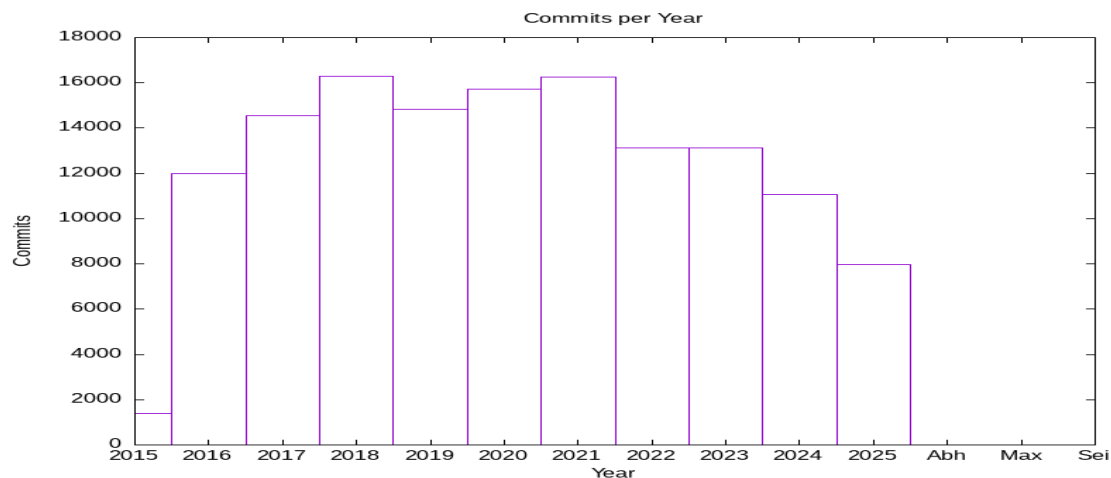
```
shinothomas@pop-os:~/vscode-gitlog-analysis$ cut -d',' -f3 vscode_gitlog.csv | cut -c1-4 | sort | uniq -c | sort -nr
shinothomas@pop-os:~/vscode-gitlog-analysis$ gnuplot
Command 'gnuplot' not found, did you mean:
  Command 'gnuplot' from deb gnuplot-nox (5.4.2+dfsg2-2)
  Command 'gnuplot' from deb gnuplot-qt (5.4.2+dfsg2-2)
  Command 'gnuplot' from deb gnuplot-x11 (5.4.2+dfsg2-2)
Try: sudo apt install <deb name>
shinothomas@pop-os:~/vscode-gitlog-analysis$ gnuplot
G N U P L O T
Version 5.4 patchlevel 2      last modified 2021-06-01
Copyright (C) 1986-1993, 1998, 2004, 2007-2021
Thomas Williams, Colin Kelley and many others
gnuplot home:      http://www.gnuplot.info
faq, bugs, etc:    type "help FAQ"
immediate help:    type "help" (plot window: hit 'h')
Terminal type is now 'qt'
gnuplot> set terminal png size 800,600
Terminal type is now 'png'
Options are 'nocrop enhanced size 800,600 font "arial,12.0" '
gnuplot> set output 'commits_per_year.png'
gnuplot> set style data histogram
gnuplot> set title "Commits per Year"
gnuplot> set xlabel "Year"
gnuplot> set ylabel "Commits"
gnuplot> plot 'commits_per_year.dat' using 1:xtic(2) with boxes notitle
gnuplot> exit
shinothomas@pop-os:~/vscode-gitlog-analysis$ nano vscode-analysis.sh
```

This screenshot is of the terminal running the commands for extracting Commits per Year.

## 5. Commits per year.

```
cut -d',' -f3 vscode_gitlog.csv | cut -c1-4 | sort | uniq -c > commits_per_year.dat
```

(Similar keywords that were used in the previous commands)



```

shinothomas@pop-os:~/vscode-gitlog-analysis$ grep -i 'fix' vscode_gitlog.csv | wc -l
34762
shinothomas@pop-os:~/vscode-gitlog-analysis$ cut -d',' -f1 vscode_gitlog.csv | head
b4d91402fe4
d5e090535a1
6f978585c1f
a941a5e28d3
ef775eb0b8f
27fe9fc76c2
3cef167a29b
0cc6dece52c
d96a1540fd1
4d9e43384fd
shinothomas@pop-os:~/vscode-gitlog-analysis$ grep -i 'Hello' vscode_gitlog.csv
998fb692680,Josh Spicer,2025-06-05,hello 1.102.0 (#250791)
45744133c30,hello-smile6,2022-03-30,Update package.json (#146278)
f8188579a36,Martin Aeschlimann,2020-02-20,Merge pull request #90960 from Helloimbob/patch-1
c65b3543567,Helloimbob,2020-02-20,Merge branch 'master' into patch-1
59391f4636d,Helloimbob,2020-02-20,Merge branch 'master' into patch-1
e7f92303406,Helloimbob,2020-02-17,Autodetect csh script as shellscript
67ad033d532,Johannes Rieken,2018-06-13,hello breadcrumbs widget
1454f1bb65a,Joao Moreno,2017-12-20,hello tree my old friend
e2c21a6923f,Johannes Rieken,2017-09-29,Hello 1.18
f163f351e17,Dirk Baeumer,2017-08-25,Fixes #30417: Default tasks.json "echo Hello" does not work
356ad324697,Johannes Rieken,2016-09-09,hello, monaco.d.ts, #11754
8f35cc47683,Erich Gamma,2015-11-13,Hello Code
shinothomas@pop-os:~/vscode-gitlog-analysis$ wc -l < vscode_gitlog.csv >
bash: syntax error near unexpected token `newline'
shinothomas@pop-os:~/vscode-gitlog-analysis$ wc -l < vscode_gitlog.csv
136385
shinothomas@pop-os:~/vscode-gitlog-analysis$ cut -d',' -f4 vscode_gitlog.csv | sort | uniq | wc -l
124513
shinothomas@pop-os:~/vscode-gitlog-analysis$

```

The above screenshot shows the terminal commands for extracting data of

## 6.Total Number of Fix-Related Commits

```
grep -i 'fix' vscode_gitlog.csv | wc -l > fix_count.txt
```

(grep -i is used to find specific words in a file or text)

## 7.Git Commit Hashes

```
cut -d',' -f1 vscode_gitlog.csv | cut -c1-7 > short_hashes.txt
```

(cut -c1-7 this helps in selecting letters from position 1 to 7)

## 8.Commits Mentioning 'Hello'

```
grep -i 'Hello' vscode_gitlog.csv > notebook_commits.txt
```

## 9.Total Number of Commits

```
wc -l < vscode_gitlog.csv > total_commits.txt
```

## 10.List of Unique Commit Messages

```
cut -d',' -f4 vscode_gitlog.csv | sort | uniq > unique_messages.txt
```

# Conclusion

The analysis of the VS Code Git log using shell scripting and Linux commands provided an insightful glimpse into real-time, unstructured data from an active open-source repository. By transforming raw commit logs into structured formats like .csv and visualizing them using tools like Gnuplot, this project demonstrates how command-line tools can be effectively used for practical data analysis. It offers hands-on experience in parsing, sorting, counting, and graphing meaningful insights from real-world datasets.

# Possible Future Improvements

1. Time-Series Analysis:  
Include commit trends over time (e.g., commits per month or year) to observe activity spikes or dips and correlate them with feature releases.
2. Contribution Heatmaps:  
Visualize contributions using heatmaps for better clarity of active contributors over time.
3. Author Categorization:  
Categorize authors into core maintainers vs. external contributors to analyze contribution diversity.
4. Integration with Python or R:  
Enhance analytical capability by integrating Python (pandas, matplotlib) or R for more complex data manipulation and visualizations.

## 5. Automated Reports:

Automate the entire process into a shell script that periodically pulls Git logs, updates datasets, and regenerates visual reports.

## 6. Branch-wise Analysis:

Include analysis for different branches (like main, dev, etc.) to observe development patterns across workflows.

# Appendix

The Git log data was extracted directly from the Visual Studio Code GitHub repository using the `git log` command with a custom **--pretty=format string**.

Only the essential fields commit hash, author name, commit date, and commit message were included.

The date was standardized to **YYYY-MM-DD (--date=short)**, and a **comma-separated format** was chosen to produce a CSV file that is immediately ready for analysis.

A header row was added to the file so that the columns are clearly labeled.

Since the extraction was already filtered for relevant fields, no further heavy cleanup was needed.

## Shell Script: Clean Git Log Export with Header(vscode\_data\_clean.sh)

```
#!/bin/bash

# File name for the cleaned dataset

OUTPUT_FILE="vscode_gitlog.csv"


# Add header row

echo "commit_hash,author_name,commit_date,commit_message" >
"$OUTPUT_FILE"
```

```
# Append Git log data in CSV format
```

```
git log \
```

```
--pretty=format:'%H,%an,%ad,%s' \
```

```
--date=short \
```

```
>> "$OUTPUT_FILE"
```

```
echo "Clean Git log data exported to $OUTPUT_FILE"
```

## Shell Script: Analysis (vscode\_analysis\_script.sh)

```
#!/bin/bash
```

```
echo "Starting VSCode Git Log Analysis"
```

```
# 1. Top 5 Contributors by Commit Count
```

```
cut -d',' -f2 vscode_gitlog.csv | sort | uniq -c | sort -nr | head -5 > top_authors.dat
```

```
# 2. Most Active Commit Days (Top 5)
```

```
cut -d',' -f3 vscode_gitlog.csv | sort | uniq -c | sort -nr | head -5 > op_days.dat
```

```
# 3. Commit Count Per Month
```

```
cut -d',' -f3 vscode_gitlog.csv | cut -c1-7 | sort | uniq -c | sort -k2 > commits_per_month.dat
```

```
# 4. Commits by Each Author (Alphabetical)
```

```
cut -d',' -f2 vscode_gitlog.csv | sort | uniq -c | sort -k2 > author_commits.dat
```

```
# 5. Commits Per Year
```

```
cut -d',' -f3 vscode_gitlog.csv | cut -c1-4 | sort | uniq -c > commits_per_year.dat
```

```
# 6. Count Commits with "fix" in Message
```

```
grep -i 'fix' vscode_gitlog.csv | wc -l
```

```
# 7. List of All Commit Hashes (Short)
```

```
cut -d',' -f1 vscode_gitlog.csv | cut -c1-7 | head
```



# 8. Search for Commits Containing Keyword

```
grep -i 'Hello' vscode_gitlog.csv
```

# 9. Total Number of Commits

```
wc -l < vscode_gitlog.csv
```

# 10. All Unique Commit Messages

```
cut -d',' -f4 vscode_gitlog.csv | sort | uniq
```

# -----

# Graphs using gnuplot

# -----

```
gnuplot <<EOF
```

```
set terminal png size 800,600
```

```
set output 'top_authors.png'
```

```
set style data histogram
```

```
set style fill solid 1.00 border -1
```

```
set boxwidth 0.6
```

```
set title "Top 5 Contributors by Commit Count"
```

```
set xlabel "Author"
```

```
set ylabel "Commits"
```

```
set xtics rotate by -45
```

```
plot 'top_authors.dat' using 1:xtic(2) title "
```

```
EOF
```

```
gnuplot <<EOF
```

```
set terminal png size 800,600
```

```
set output 'top_days.png'
```

```
set style data histogram
```

```
set boxwidth 0.5
```

```
set title "Most Active Commit Days"
```

```
set xlabel "Date"
```

```
set ylabel "Commits"
```

```
set xtics rotate by -45
```

```
plot 'top_days.dat' using 1:xtic(2) title "
```

```
EOF
```

```
gnuplot <<EOF
```

```
set terminal png size 1000,400
```

```
set output 'commits_per_month.png'
```

```
set xdata time
```

```
set timefmt "%Y-%m"
```

```
set format x "%Y-%m"
```

```
set title "Commits Per Month"
```

```
set xlabel "Month"
set ylabel "Commits"
set grid
plot 'commits_per_month.dat' using 2:1 with linespoints title 'Commits'
EOF
```

```
gnuplot <<EOF
set terminal png size 1000,600
set output 'author_commits.png'
set style data histograms
set style fill solid 1.0 border -1
set boxwidth 0.6
set title "Commits by Author"
set xlabel "Author"
set ylabel "Commits"
set xtics rotate by -45
plot 'author_commits.dat' using 1:xtic(2) notitle
EOF
```

```
gnuplot <<EOF
set terminal png size 800,600
set output 'commits_per_year.png'
set style data histogram
set title "Commits per Year"
set xlabel "Year"
set ylabel "Commits"
plot 'commits_per_year.dat' using 1:xtic(2) with boxes notitle
EOF
```

```
echo "Analysis Complete. Results saved in ./vscode-gitlog-anlaysis folder."
```