MT2005 Probability & Statistics Semester Project Fall 2024

Stack Overflow Trends Analysis

1	Roll Number	Name	Section
	21F-9136	Ayesha Zahid	5A

1. Problem Statement

With the ever-evolving landscape of programming languages and technologies, developers rely heavily on platforms like Stack Overflow for learning, problem-solving, and community engagement. However, understanding long-term trends in programming language popularity and usage patterns can be challenging without data-driven insights. This project analyzes Stack Overflow trends over time, providing a comprehensive understanding of which languages are gaining or losing popularity. These insights will help educators, learners, and industry professionals make informed decisions regarding skill development and resource allocation.

2. Objective

The objective of this project is to analyze Stack Overflow trends to gain actionable insights into programming language popularity, usage patterns, and seasonal variations. By utilizing statistical techniques and visualization tools, the project aims to identify which programming languages dominate discussions, how their popularity shifts over time, and how questions are distributed monthly. These insights can assist students, professionals, and organizations in strategic decision-making, curriculum design, and adapting to industry demands.

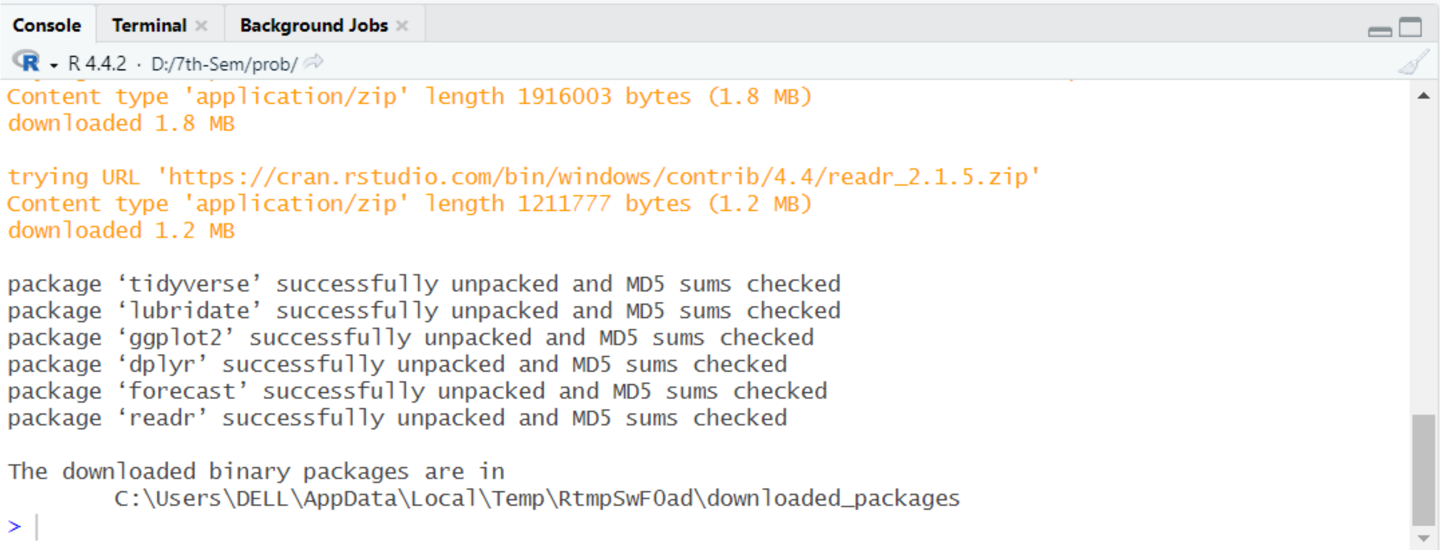
3. Data Description

The dataset contains the following columns:

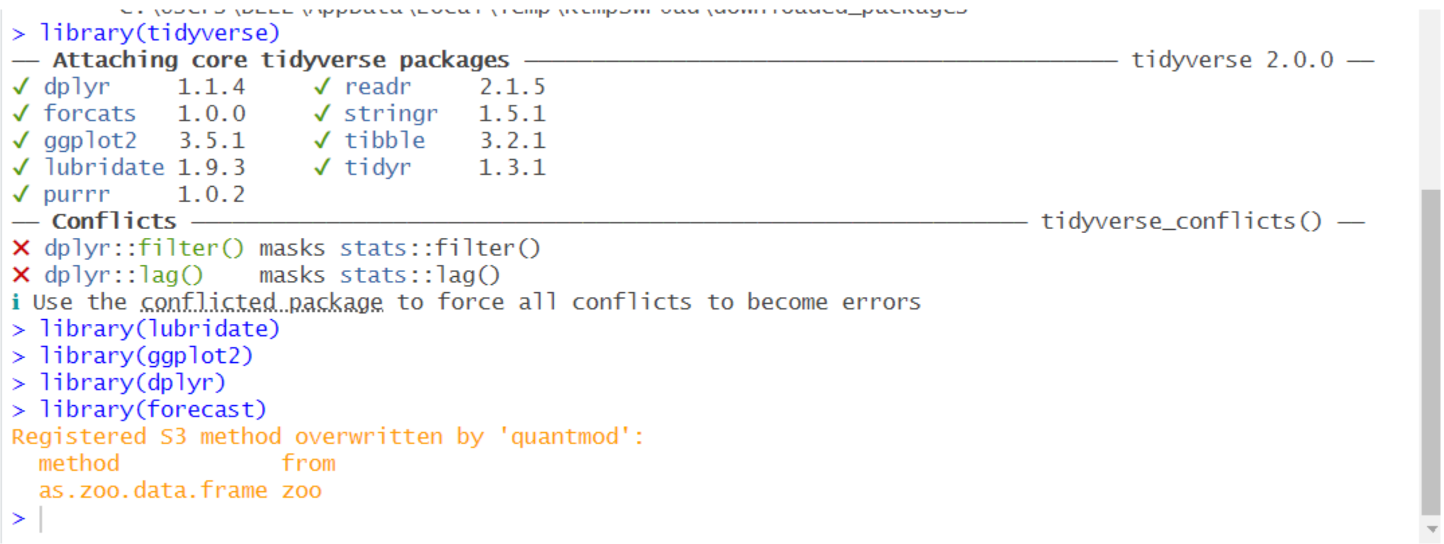
- Month: The month & year in which the data was recorded.
- C++, C#, TypeScript, PHP, Swift, Ruby, Go, SQL, Kotlin, Scala, Shell, C, HTML, Objective-C, Perl, Matlab, R, Python, Java, JavaScript: The total number of questions asked on Stack Overflow related to the specific programming language during the given month. The dataset used for this analysis can be found on Kaggle here:

  https://www.kaggle.com/datasets/computingvictor/monthly-trends-in-stack-overflow-questions.

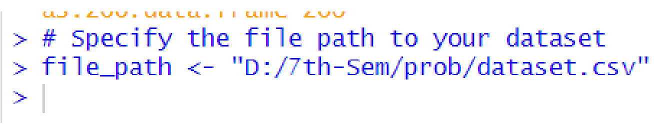
4. Results



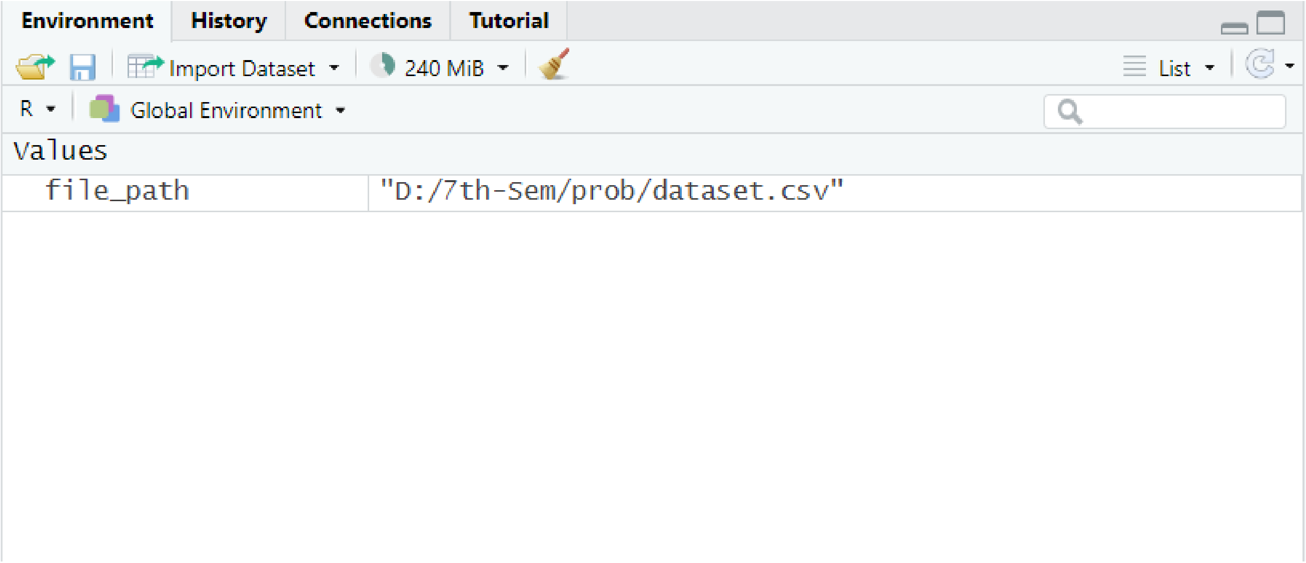
```
Content type 'application/zip' length 1916003 bytes (1.8 MB)
downloaded 1.8 MB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.4/readr_2.1.5.zip'
Content type 'application/zip' length 1211777 bytes (1.2 MB)
downloaded 1.2 MB

package ‘tidyverse’ successfully unpacked and MD5 sums checked
package ‘lubridate’ successfully unpacked and MD5 sums checked
package ‘ggplot2’ successfully unpacked and MD5 sums checked
package ‘dplyr’ successfully unpacked and MD5 sums checked
package ‘forecast’ successfully unpacked and MD5 sums checked
package ‘readr’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\DELL\AppData\Local\Temp\RtmpSwF0ad\downloaded_packages
> 
```



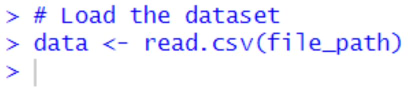
```
        C:\Users\DELL\AppData\Local\Temp\RtmpSwF0ad\downloaded_packages
> library(tidyverse)
── Attaching core tidyverse packages ──────────────────────────── tidyverse 2.0.0 ──
✓ dplyr     1.1.4     ✓ readr     2.1.5
✓ forcats   1.0.0     ✓ stringr   1.5.1
✓ ggplot2   3.5.1     ✓ tibble    3.2.1
✓ lubridate 1.9.3     ✓ tidyr     1.3.1
✓ purrr     1.0.2     
── Conflicts ────────────────────────────────────────────── tidyverse_conflicts() ──
✗ dplyr::filter() masks stats::filter()
✗ dplyr::lag()    masks stats::lag()
i Use the conflicted package to force all conflicts to become errors
> library(lubridate)
> library(ggplot2)
> library(dplyr)
> library(forecast)
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
> 
```



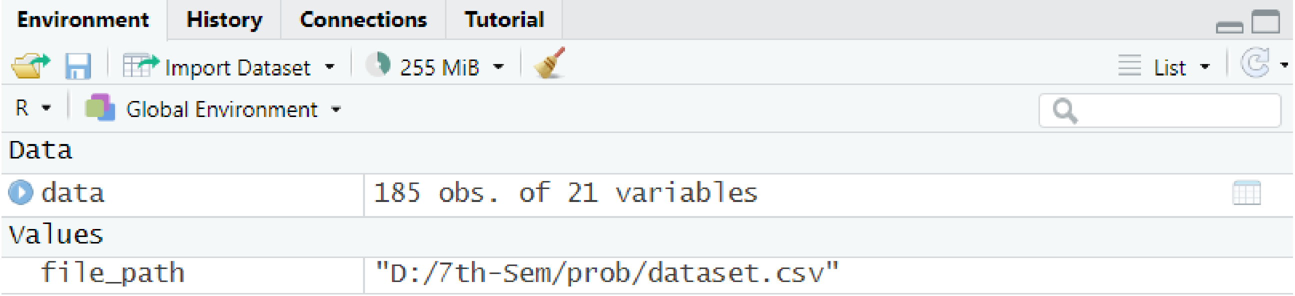
```
  as.zoo.data.frame zoo
> # Specify the file path to your dataset
> file_path <- "D:/7th-Sem/prob/dataset.csv"
> 
```



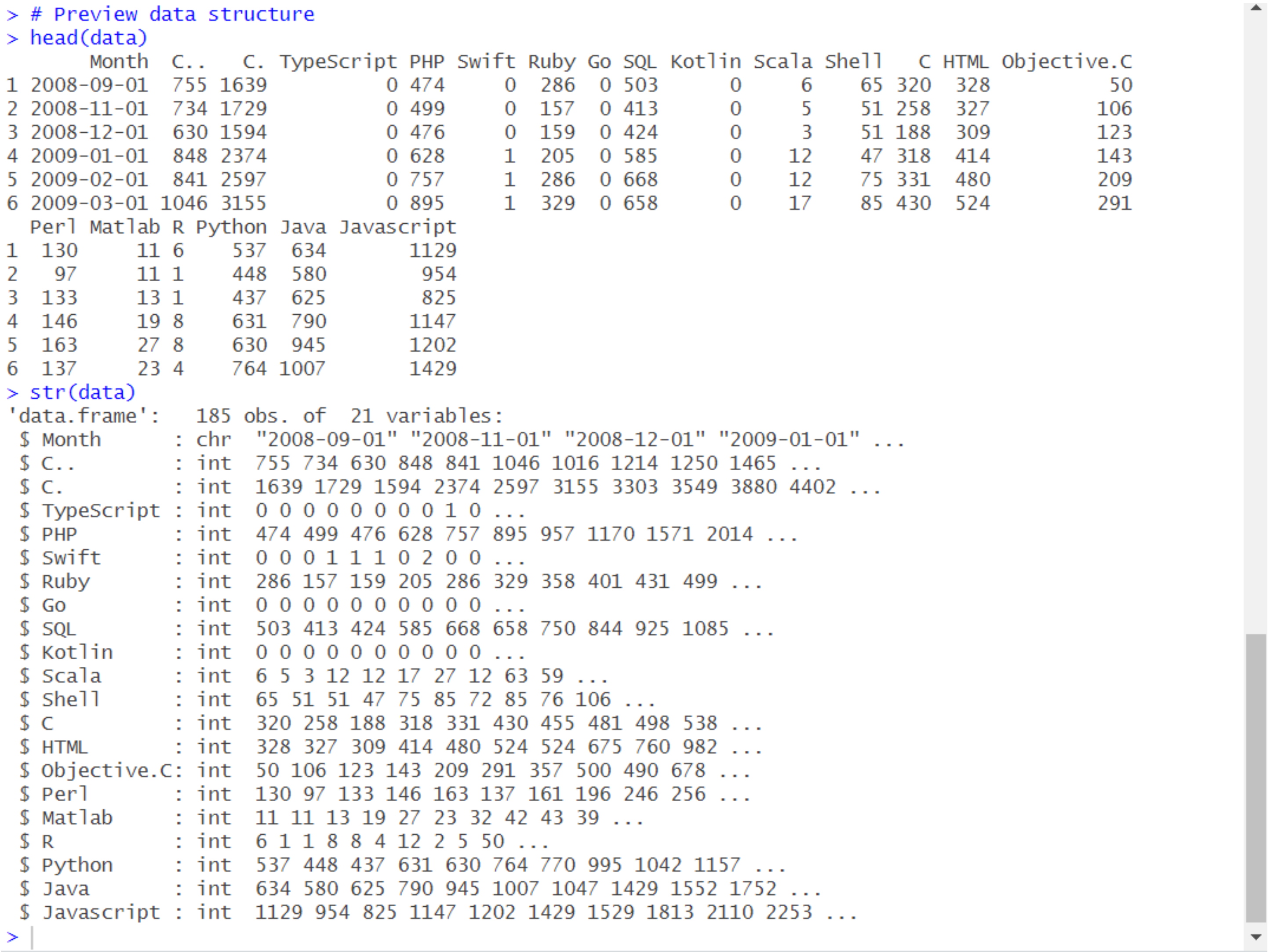
```
Values
  file_path             "D:/7th-Sem/prob/dataset.csv"
```



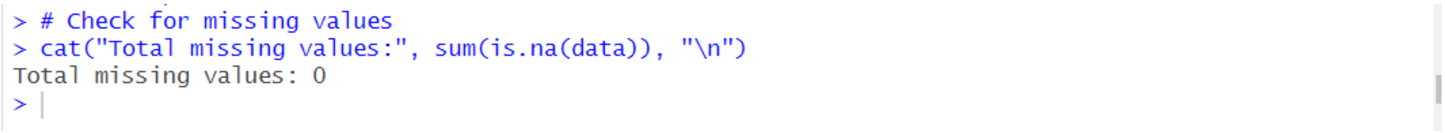
```
> # Load the dataset
> data <- read.csv(file_path)
> 
```



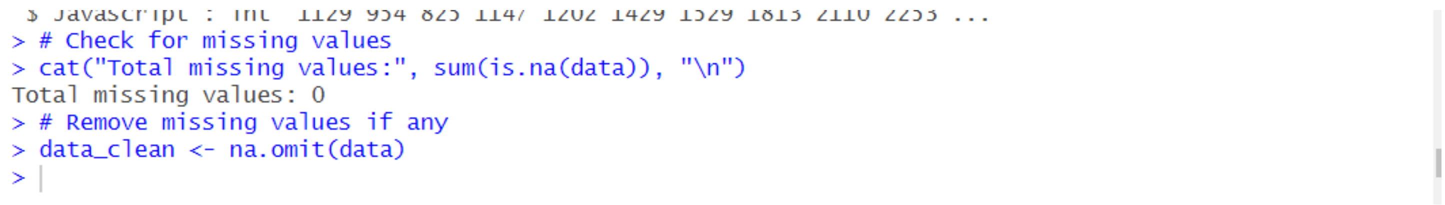
```
Data
 data                  185 obs. of 21 variables
Values
  file_path             "D:/7th-Sem/prob/dataset.csv"
```



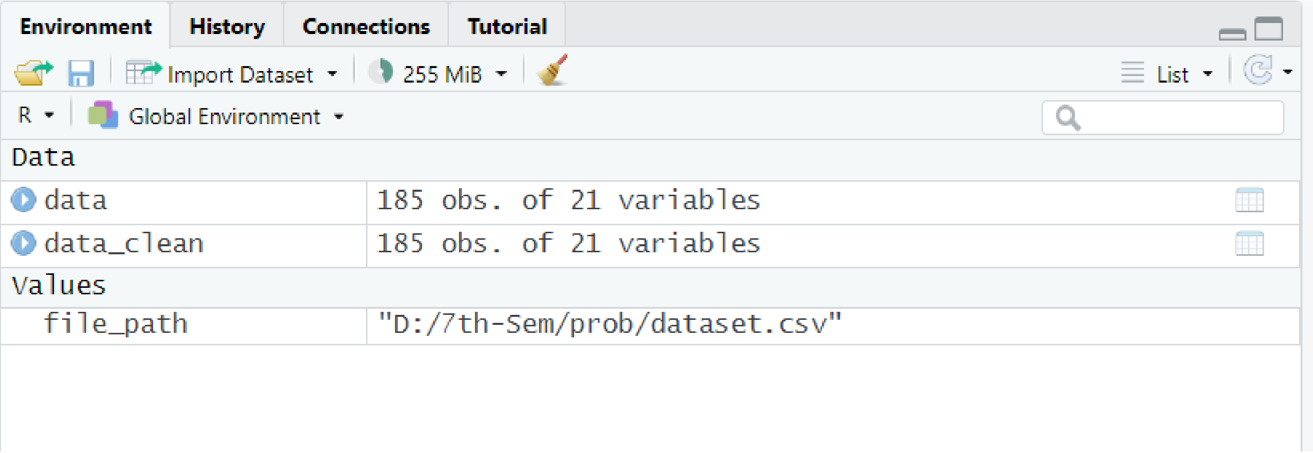
```
> # Preview data structure
> head(data)
       Month   C..    C. TypeScript PHP Swift Ruby Go SQL Kotlin Scala Shell   C HTML Objective.C
1 2008-09-01  755 1639          0 474     0  286  0 503      0     6    65 320  328          50
2 2008-11-01  734 1729          0 499     0  157  0 413      0     5    51 258  327         106
3 2008-12-01  630 1594          0 476     0  159  0 424      0     3    51 188  309         123
4 2009-01-01  848 2374          0 628     1  205  0 585      0    12    47 318  414         143
5 2009-02-01  841 2597          0 757     1  286  0 668      0    12    75 331  480         209
6 2009-03-01 1046 3155          0 895     1  329  0 658      0    17    85 430  524         291
  Perl Matlab R Python Java Javascript
1  130     11 6    537  634       1129
2   97     11 1    448  580        954
3  133     13 1    437  625        825
4  146     19 8    631  790       1147
5  163     27 8    630  945       1202
6  137     23 4    764 1007       1429
> str(data)
'data.frame':	185 obs. of  21 variables:
 $ Month      : chr  "2008-09-01" "2008-11-01" "2008-12-01" "2009-01-01" ...
 $ C..        : int  755 734 630 848 841 1046 1016 1214 1250 1465 ...
 $ C.         : int  1639 1729 1594 2374 2597 3155 3303 3549 3880 4402 ...
 $ TypeScript : int  0 0 0 0 0 0 0 0 1 0 ...
 $ PHP        : int  474 499 476 628 757 895 957 1170 1571 2014 ...
 $ Swift      : int  0 0 0 1 1 1 0 2 0 0 ...
 $ Ruby       : int  286 157 159 205 286 329 358 401 431 499 ...
 $ Go         : int  0 0 0 0 0 0 0 0 0 0 ...
 $ SQL        : int  503 413 424 585 668 658 750 844 925 1085 ...
 $ Kotlin     : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Scala      : int  6 5 3 12 12 17 27 12 63 59 ...
 $ Shell      : int  65 51 51 47 75 85 72 85 76 106 ...
 $ C          : int  320 258 188 318 331 430 455 481 498 538 ...
 $ HTML       : int  328 327 309 414 480 524 524 675 760 982 ...
 $ Objective.C: int  50 106 123 143 209 291 357 500 490 678 ...
 $ Perl       : int  130 97 133 146 163 137 161 196 246 256 ...
 $ Matlab     : int  11 11 13 19 27 23 32 42 43 39 ...
 $ R          : int  6 1 1 8 8 4 12 2 5 50 ...
 $ Python     : int  537 448 437 631 630 764 770 995 1042 1157 ...
 $ Java       : int  634 580 625 790 945 1007 1047 1429 1552 1752 ...
 $ Javascript : int  1129 954 825 1147 1202 1429 1529 1813 2110 2253 ...
> 
```



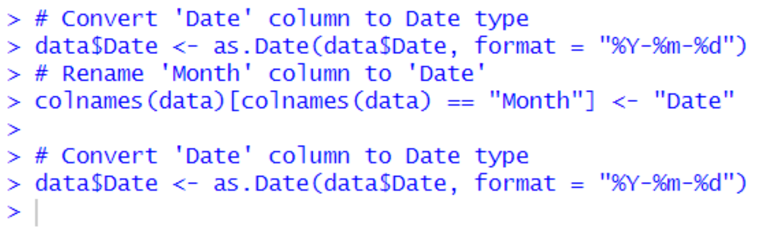
```
> # Check for missing values
> cat("Total missing values:", sum(is.na(data)), "\n")
Total missing values: 0
> 
```



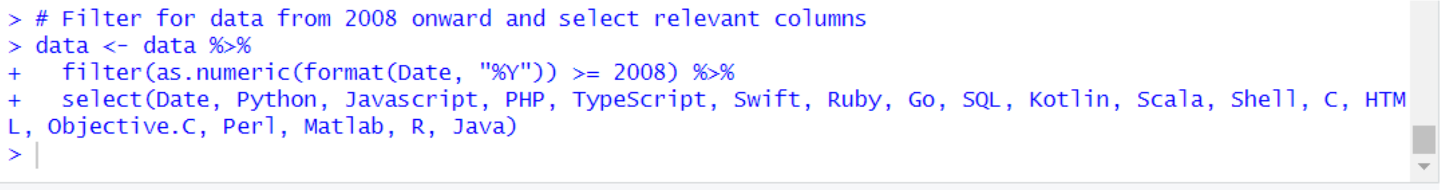
```
 $ Javascript : int  1129 954 825 1147 1202 1429 1529 1813 2110 2253 ...
> # Check for missing values
> cat("Total missing values:", sum(is.na(data)), "\n")
Total missing values: 0
> # Remove missing values if any
> data_clean <- na.omit(data)
> 
```



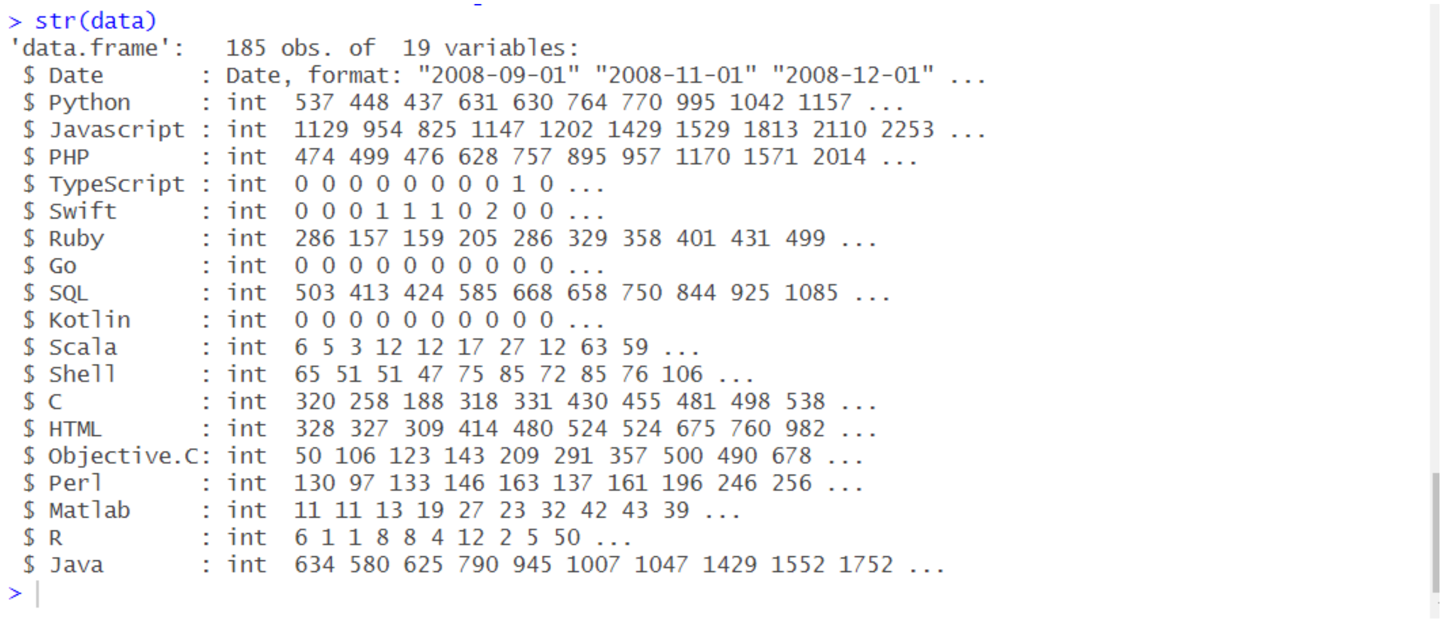
```
Data
data          185 obs. of 21 variables
data_clean    185 obs. of 21 variables
Values
  file_path   "D:/7th-Sem/prob/dataset.csv"
```



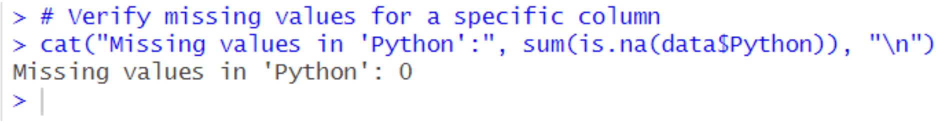
```
> # Convert 'Date' column to Date type
> data$Date <- as.Date(data$Date, format = "%Y-%m-%d")
> # Rename 'Month' column to 'Date'
> colnames(data)[colnames(data) == "Month"] <- "Date"
>
> # Convert 'Date' column to Date type
> data$Date <- as.Date(data$Date, format = "%Y-%m-%d")
> 
```



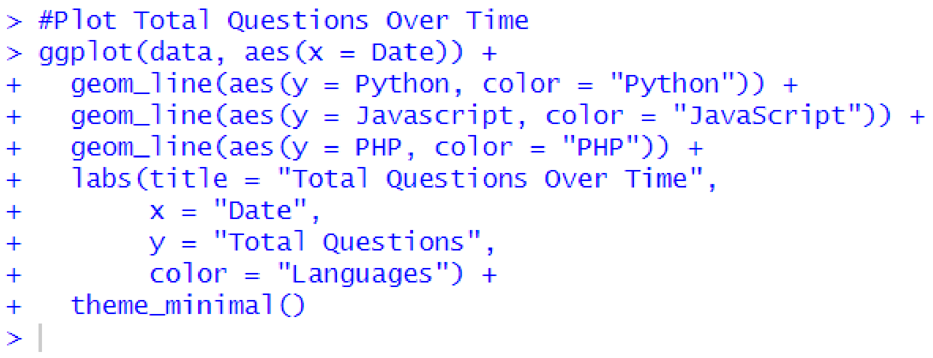
```
> # Filter for data from 2008 onward and select relevant columns
> data <- data %>%
+   filter(as.numeric(format(Date, "%Y")) >= 2008) %>%
+   select(Date, Python, Javascript, PHP, TypeScript, Swift, Ruby, Go, SQL, Kotlin, Scala, Shell, C, HTM
L, Objective.C, Perl, Matlab, R, Java)
> 
```



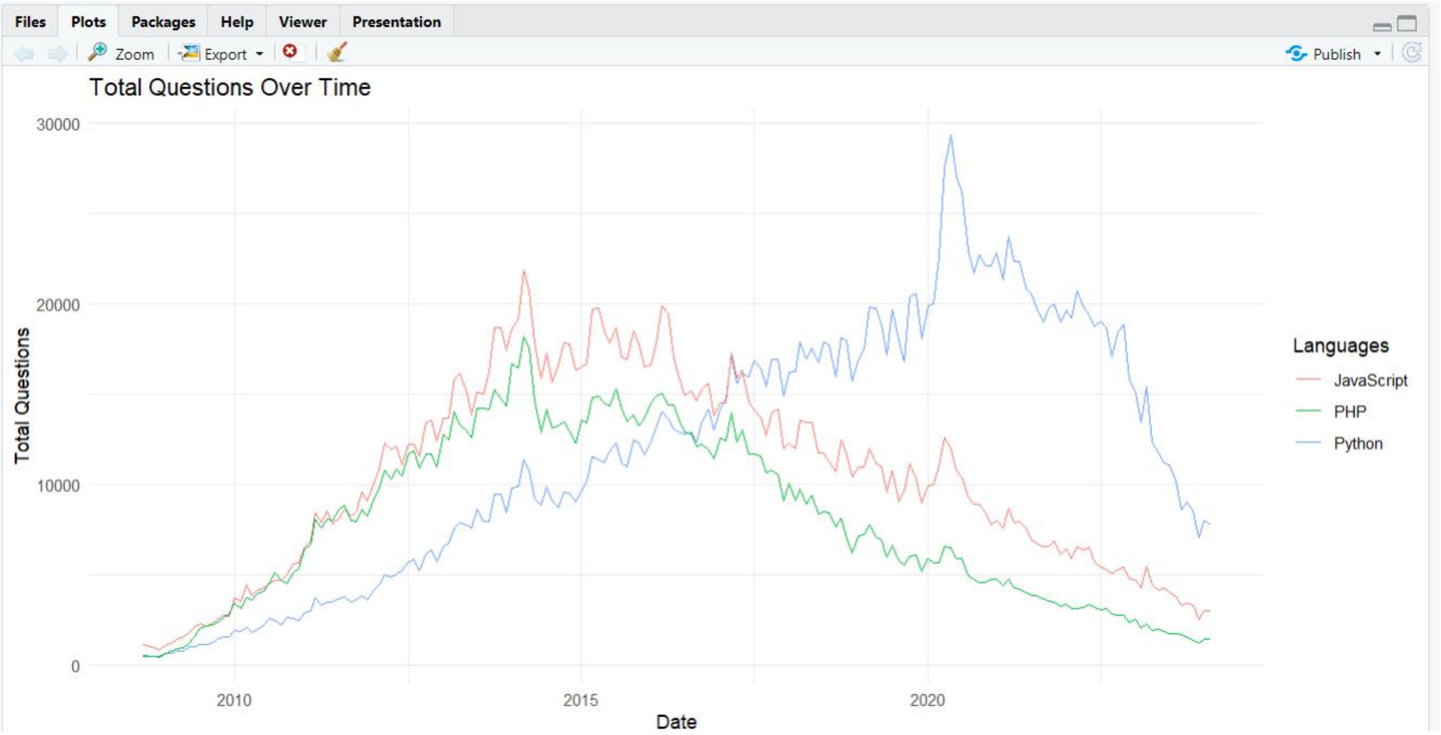
```
> str(data)
'data.frame':	185 obs. of  19 variables:
 $ Date       : Date, format: "2008-09-01" "2008-11-01" "2008-12-01" ...
 $ Python     : int  537 448 437 631 630 764 770 995 1042 1157 ...
 $ Javascript : int  1129 954 825 1147 1202 1429 1529 1813 2110 2253 ...
 $ PHP        : int  474 499 476 628 757 895 957 1170 1571 2014 ...
 $ TypeScript : int  0 0 0 0 0 0 0 0 1 0 ...
 $ Swift      : int  0 0 0 1 1 1 0 2 0 0 ...
 $ Ruby       : int  286 157 159 205 286 329 358 401 431 499 ...
 $ Go         : int  0 0 0 0 0 0 0 0 0 0 ...
 $ SQL        : int  503 413 424 585 668 658 750 844 925 1085 ...
 $ Kotlin     : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Scala      : int  6 5 3 12 12 17 27 12 63 59 ...
 $ Shell      : int  65 51 51 47 75 85 72 85 76 106 ...
 $ C          : int  320 258 188 318 331 430 455 481 498 538 ...
 $ HTML       : int  328 327 309 414 480 524 524 675 760 982 ...
 $ Objective.C: int  50 106 123 143 209 291 357 500 490 678 ...
 $ Perl       : int  130 97 133 146 163 137 161 196 246 256 ...
 $ Matlab     : int  11 11 13 19 27 23 32 42 43 39 ...
 $ R          : int  6 1 1 8 8 4 12 2 5 50 ...
 $ Java       : int  634 580 625 790 945 1007 1047 1429 1552 1752 ...
> 
```

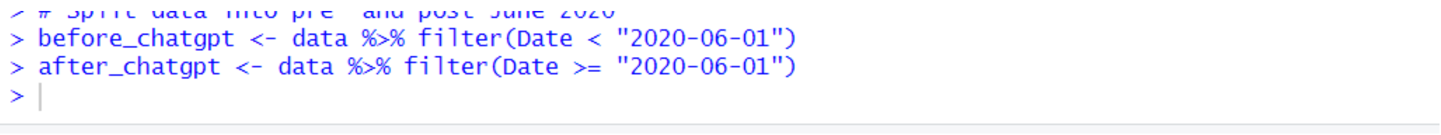


```
> # Verify missing values for a specific column
> cat("Missing values in 'Python':", sum(is.na(data$Python)), "\n")
Missing values in 'Python': 0
> 
```

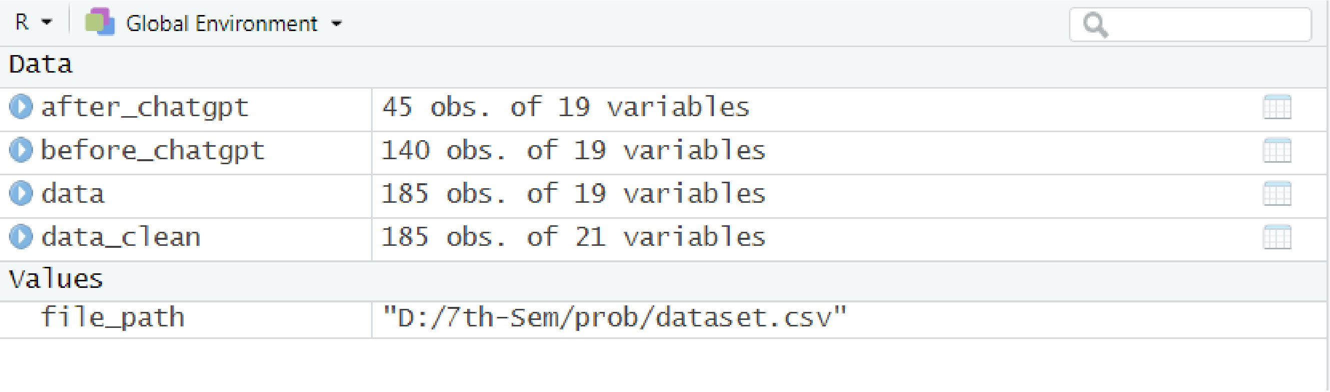


```
> #Plot Total Questions Over Time
> ggplot(data, aes(x = Date)) +
+   geom_line(aes(y = Python, color = "Python")) +
+   geom_line(aes(y = Javascript, color = "JavaScript")) +
+   geom_line(aes(y = PHP, color = "PHP")) +
+   labs(title = "Total Questions Over Time",
+        x = "Date",
+        y = "Total Questions",
+        color = "Languages") +
+   theme_minimal()
> 
```

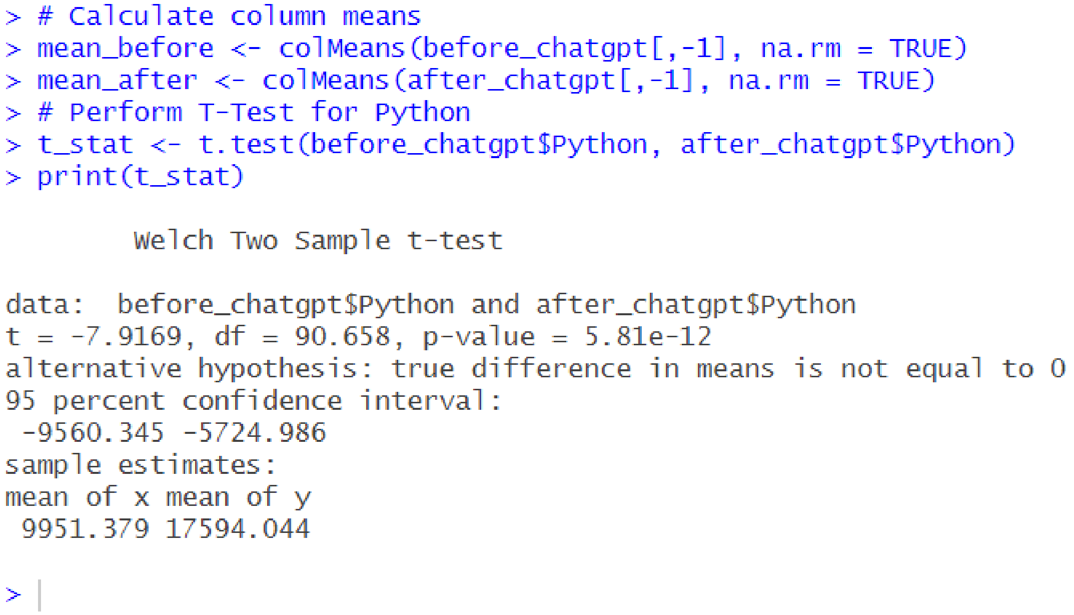




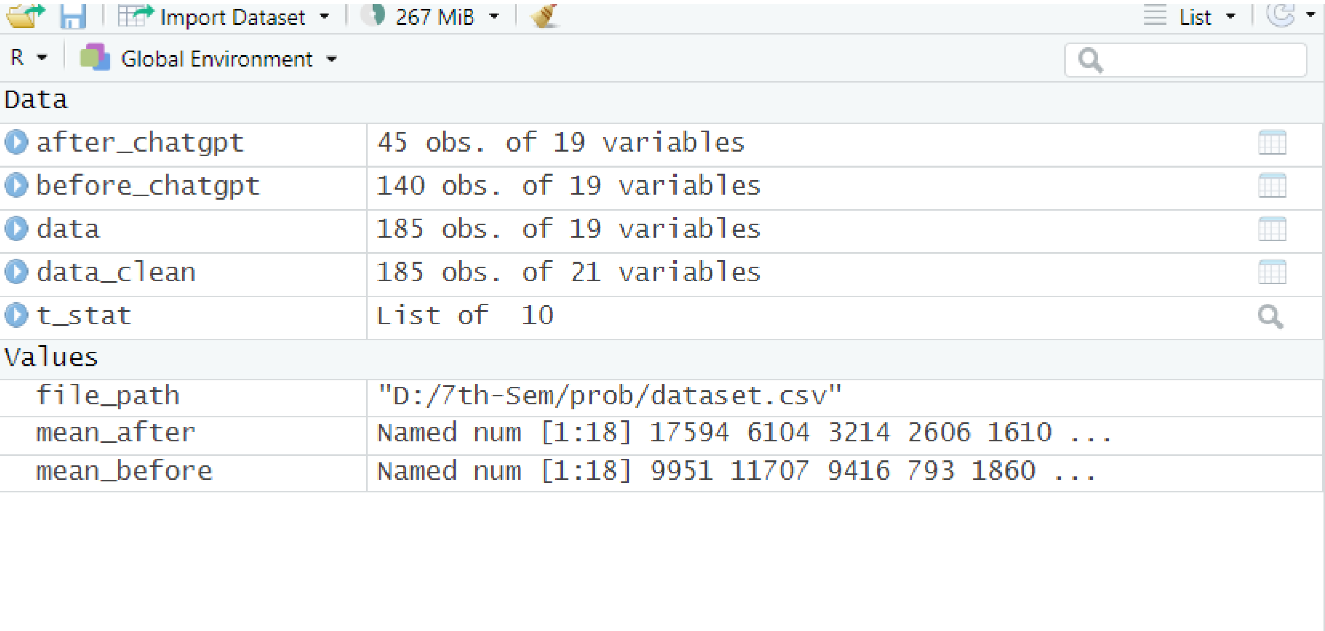
```
> # Split data into pre- and post-June 2020
> before_chatgpt <- data %>% filter(Date < "2020-06-01")
> after_chatgpt <- data %>% filter(Date >= "2020-06-01")
> 
```



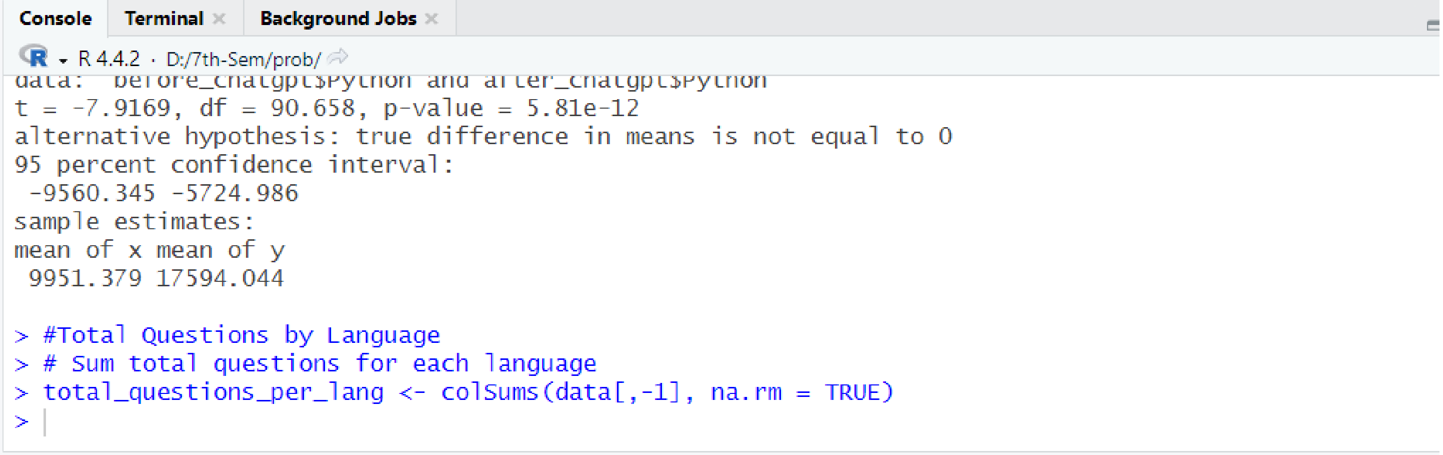
Data	
after_chatgpt	45 obs. of 19 variables
before_chatgpt	140 obs. of 19 variables
data	185 obs. of 19 variables
data_clean	185 obs. of 21 variables
Values	
file_path	"D:/7th-Sem/prob/dataset.csv"



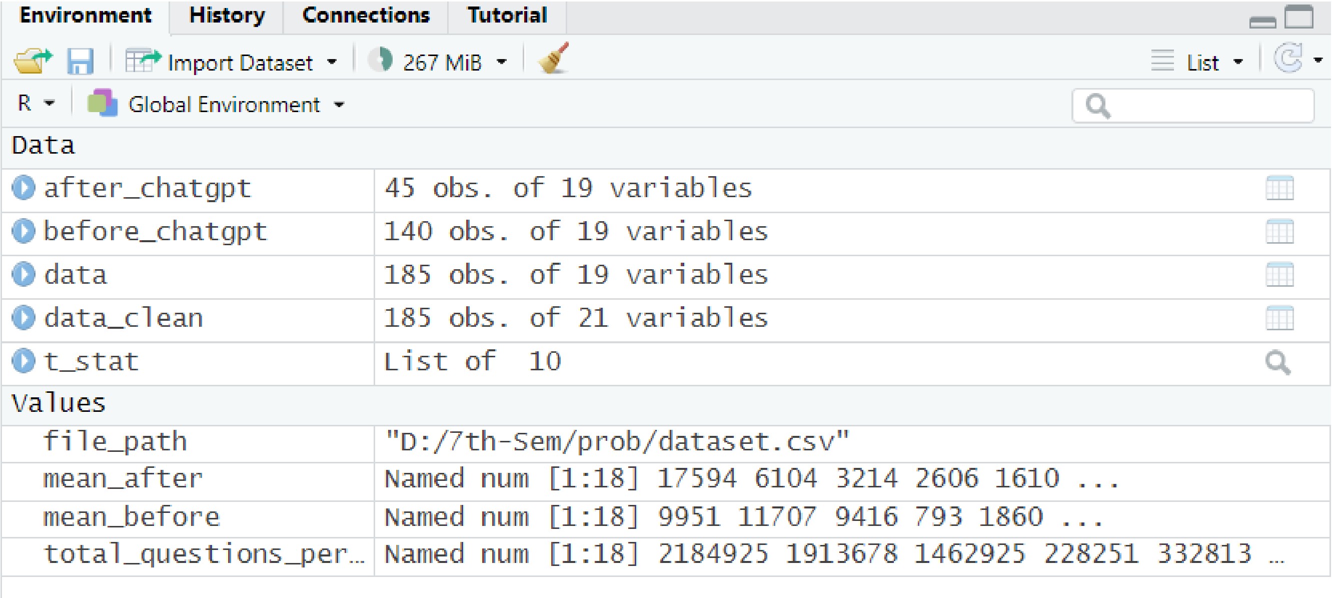
```
> # Calculate column means
> mean_before <- colMeans(before_chatgpt[,-1], na.rm = TRUE)
> mean_after <- colMeans(after_chatgpt[,-1], na.rm = TRUE)
> # Perform T-Test for Python
> t_stat <- t.test(before_chatgpt$Python, after_chatgpt$Python)
> print(t_stat)

        Welch Two Sample t-test

data:  before_chatgpt$Python and after_chatgpt$Python
t = -7.9169, df = 90.658, p-value = 5.81e-12
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -9560.345 -5724.986
sample estimates:
mean of x mean of y 
 9951.379 17594.044 

> 
```



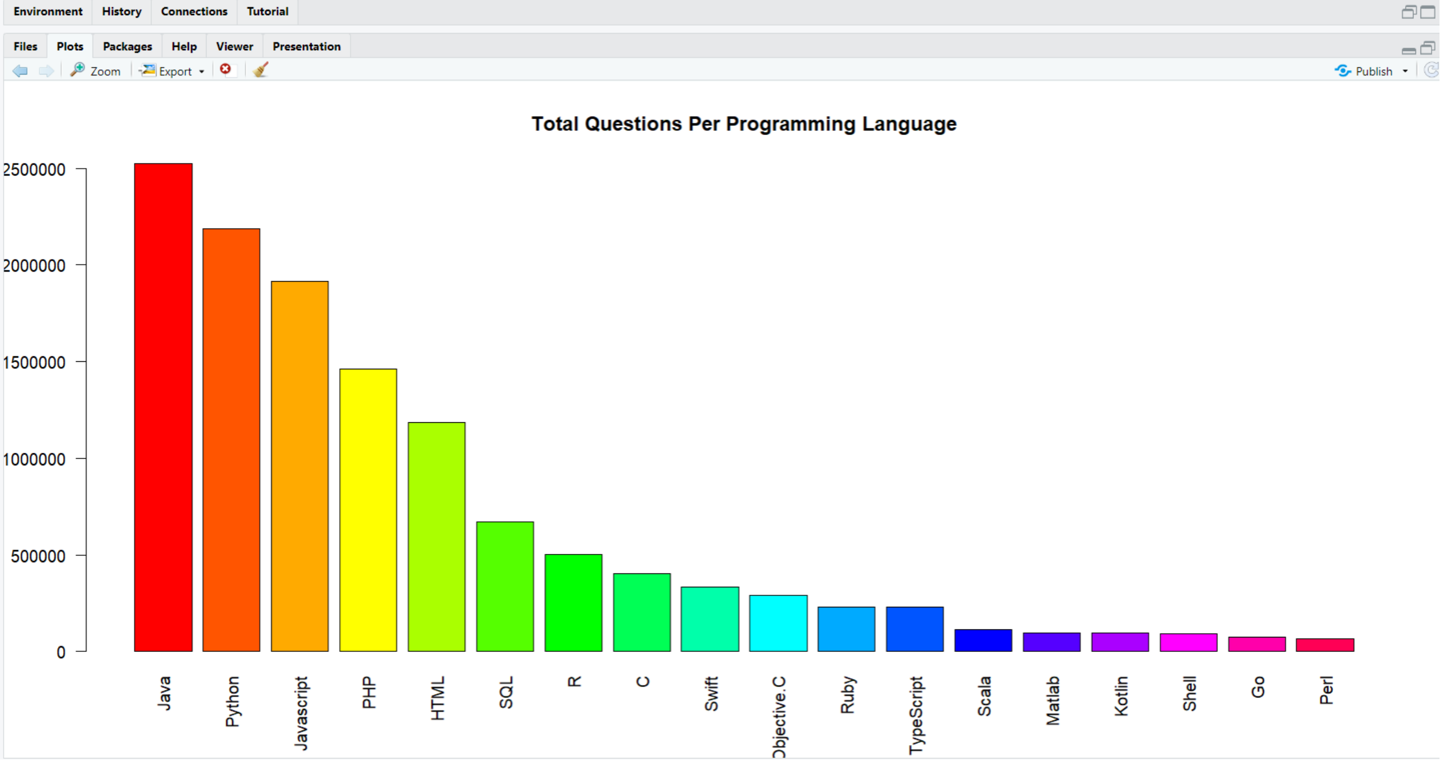
Data	
after_chatgpt	45 obs. of 19 variables
before_chatgpt	140 obs. of 19 variables
data	185 obs. of 19 variables
data_clean	185 obs. of 21 variables
t_stat	List of 10
Values	
file_path	"D:/7th-Sem/prob/dataset.csv"
mean_after	Named num [1:18] 17594 6104 3214 2606 1610 ...
mean_before	Named num [1:18] 9951 11707 9416 793 1860 ...

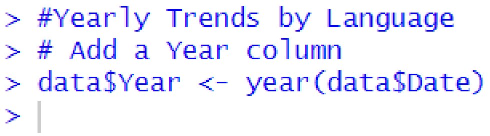


```
data:  before_chatgpt$Python and after_chatgpt$Python
t = -7.9169, df = 90.658, p-value = 5.81e-12
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -9560.345 -5724.986
sample estimates:
mean of x mean of y 
 9951.379 17594.044 

> #Total Questions by Language
> # Sum total questions for each language
> total_questions_per_lang <- colSums(data[,-1], na.rm = TRUE)
> 
```

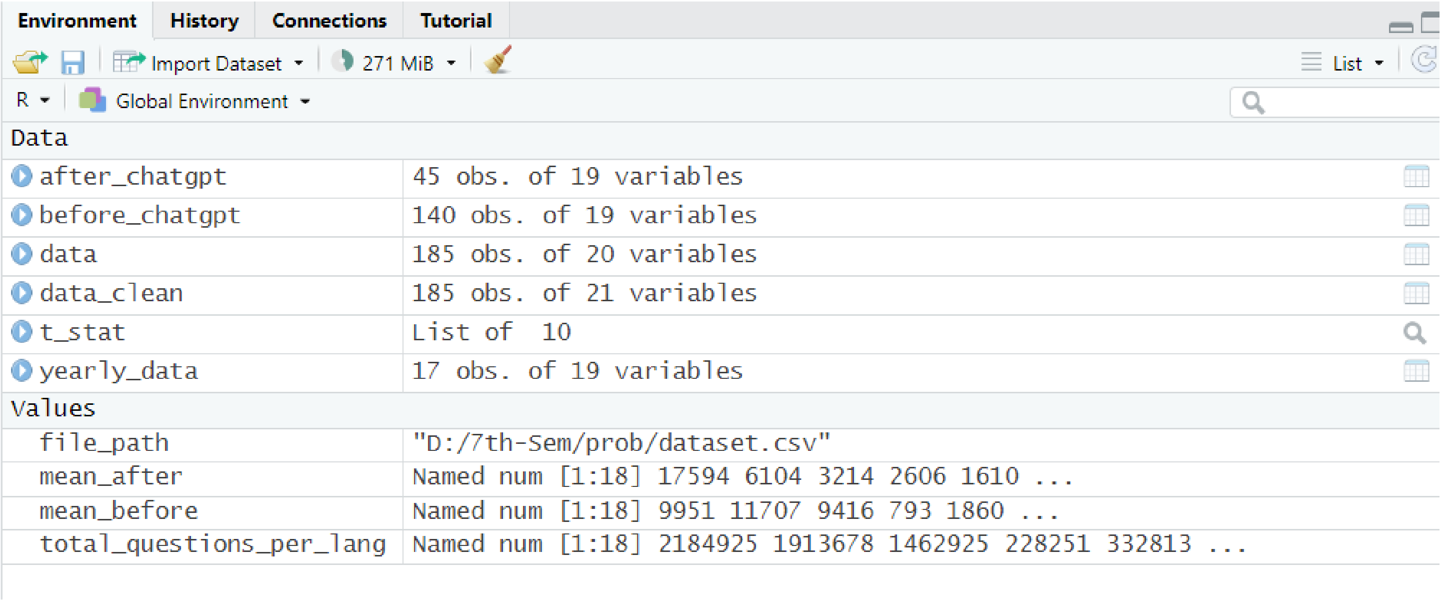


Data	
after_chatgpt	45 obs. of 19 variables
before_chatgpt	140 obs. of 19 variables
data	185 obs. of 19 variables
data_clean	185 obs. of 21 variables
t_stat	List of 10
Values	
file_path	"D:/7th-Sem/prob/dataset.csv"
mean_after	Named num [1:18] 17594 6104 3214 2606 1610 ...
mean_before	Named num [1:18] 9951 11707 9416 793 1860 ...
total_questions_per…	Named num [1:18] 2184925 1913678 1462925 228251 332813 …

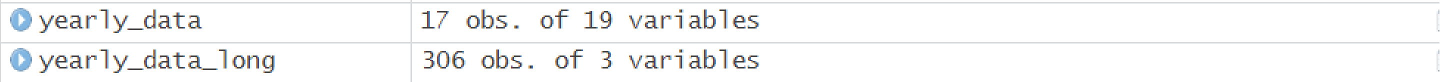




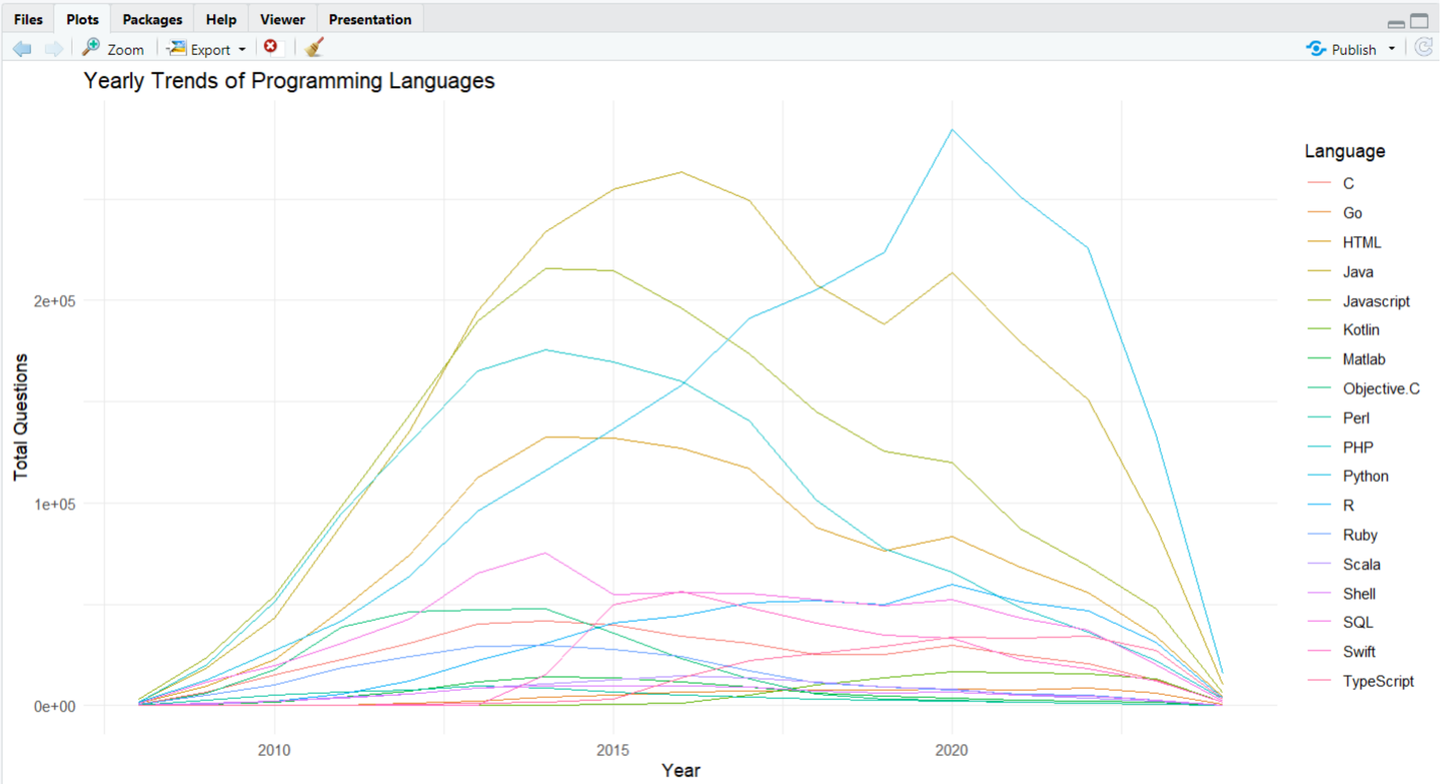
```
> #Yearly Trends by Language
> # Add a Year column
> data$Year <- year(data$Date)
> 
```

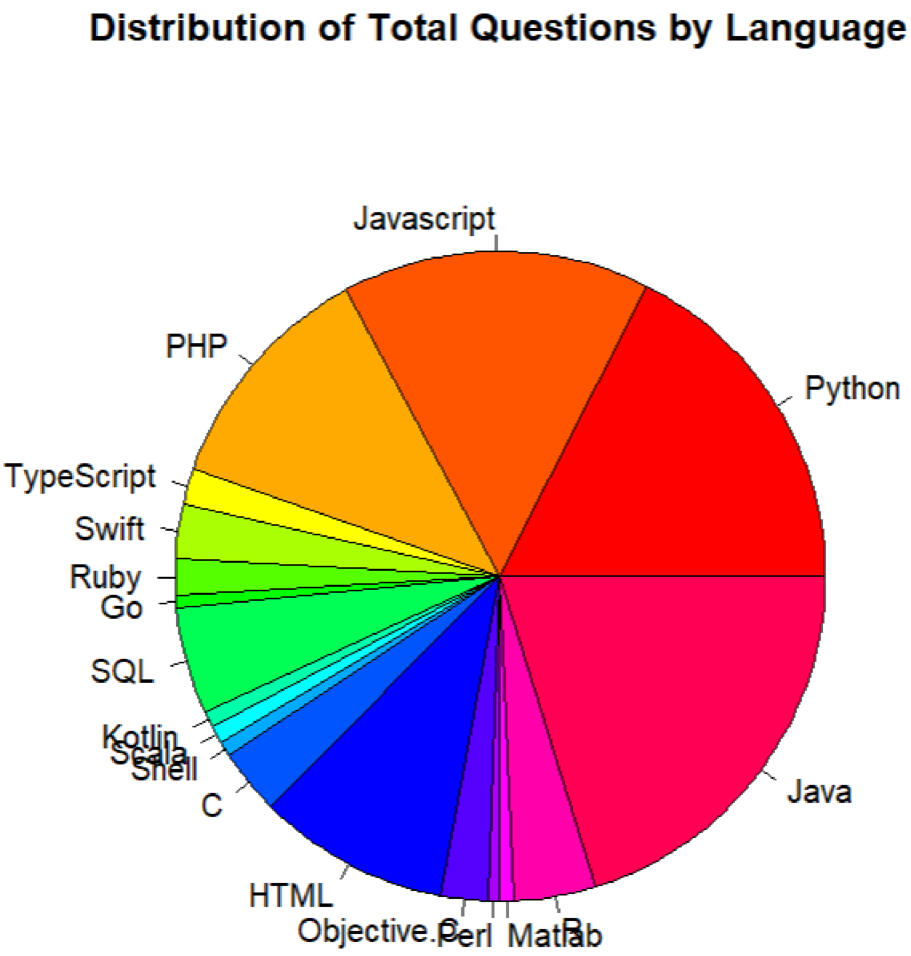


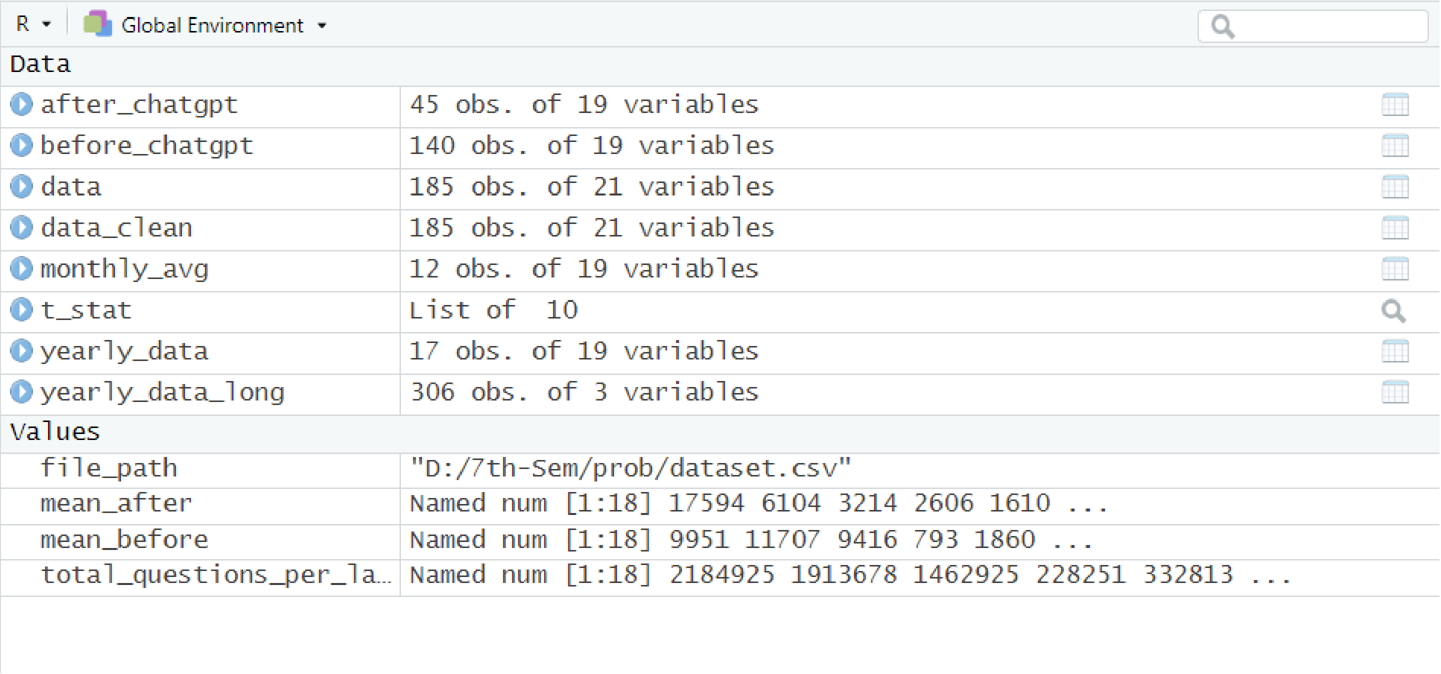
```
Data
after_chatgpt          45 obs. of 19 variables
before_chatgpt         140 obs. of 19 variables
data                   185 obs. of 20 variables
data_clean             185 obs. of 21 variables
t_stat                 List of  10
yearly_data            17 obs. of 19 variables
Values
file_path              "D:/7th-Sem/prob/dataset.csv"
mean_after             Named num [1:18] 17594 6104 3214 2606 1610 ...
mean_before            Named num [1:18] 9951 11707 9416 793 1860 ...
total_questions_per_lang Named num [1:18] 2184925 1913678 1462925 228251 332813 ...
```



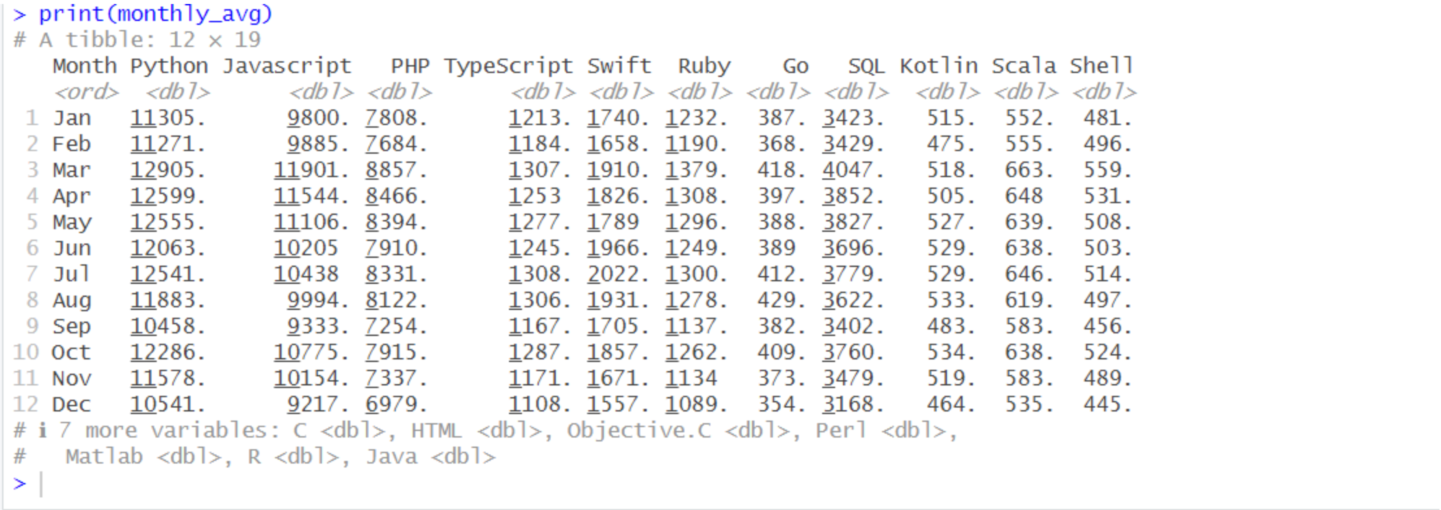
```
yearly_data            17 obs. of 19 variables
yearly_data_long       306 obs. of 3 variables
```



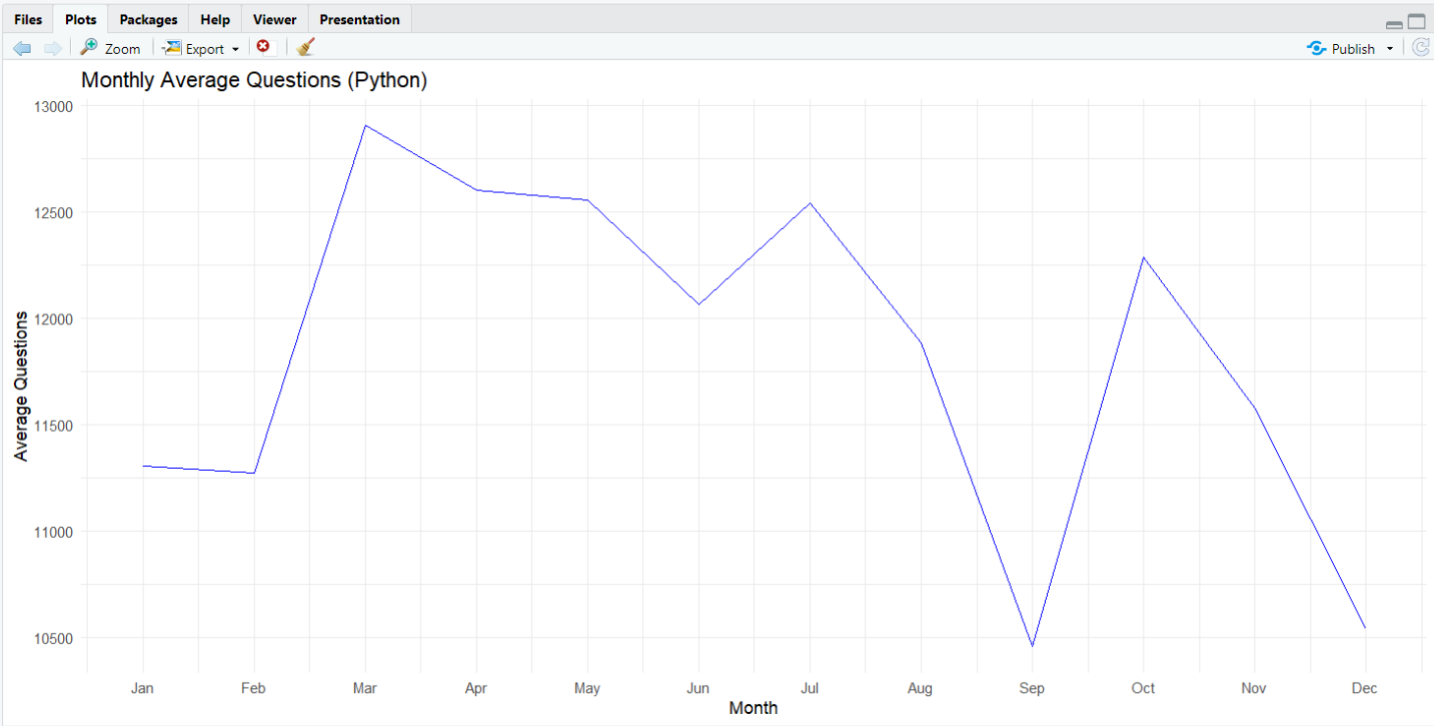


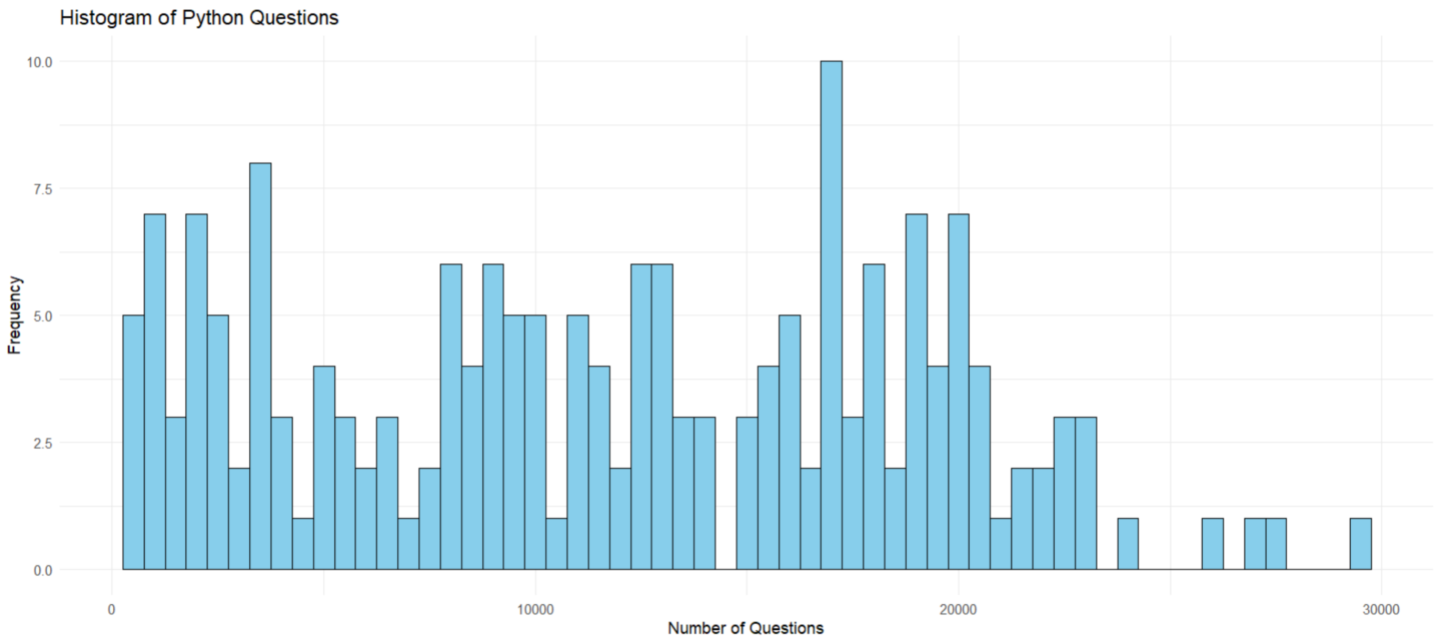


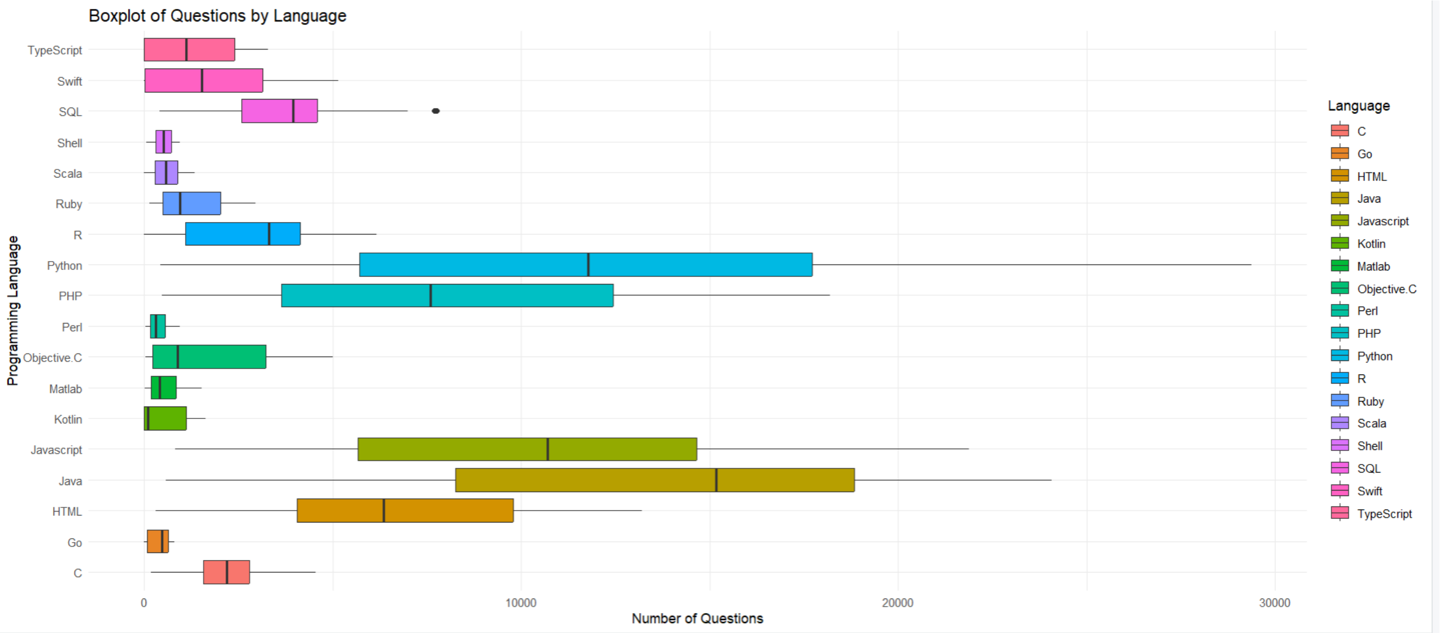
```
Data
after_chatgpt          45 obs. of 19 variables
before_chatgpt         140 obs. of 19 variables
data                   185 obs. of 21 variables
data_clean             185 obs. of 21 variables
monthly_avg            12 obs. of 19 variables
t_stat                 List of  10
yearly_data            17 obs. of 19 variables
yearly_data_long       306 obs. of 3 variables
Values
file_path              "D:/7th-Sem/prob/dataset.csv"
mean_after             Named num [1:18] 17594 6104 3214 2606 1610 ...
mean_before            Named num [1:18] 9951 11707 9416 793 1860 ...
total_questions_per_la… Named num [1:18] 2184925 1913678 1462925 228251 332813 ...
```

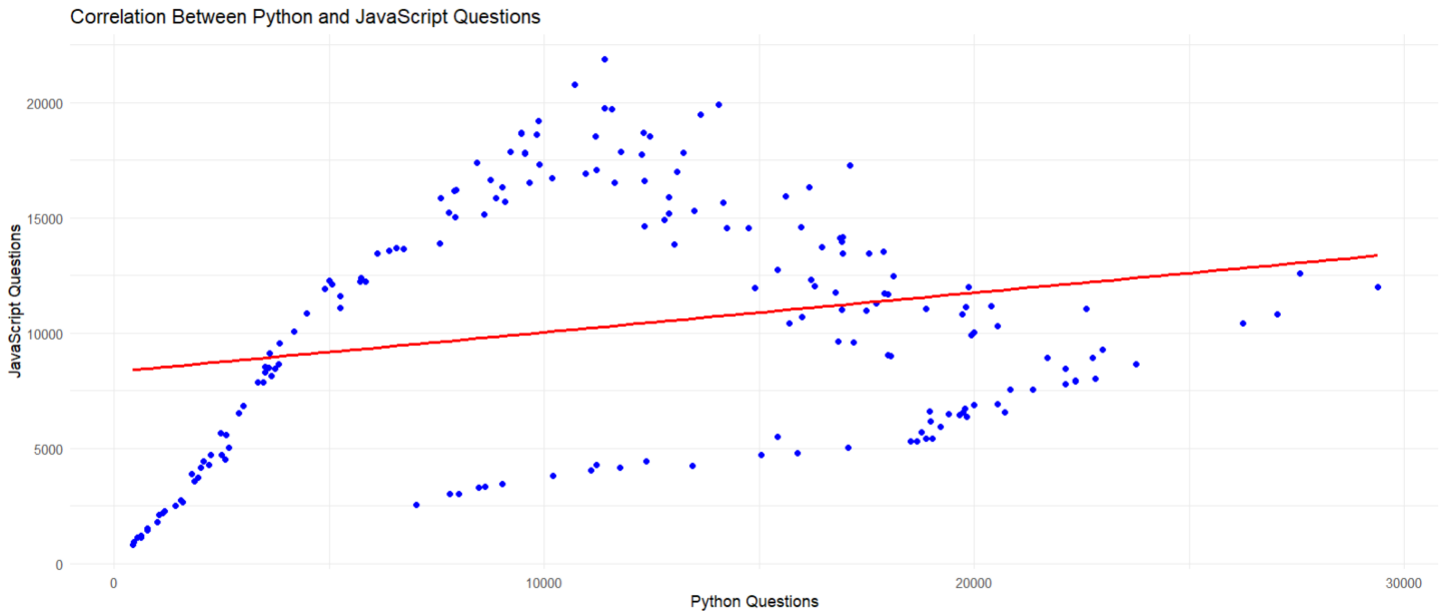


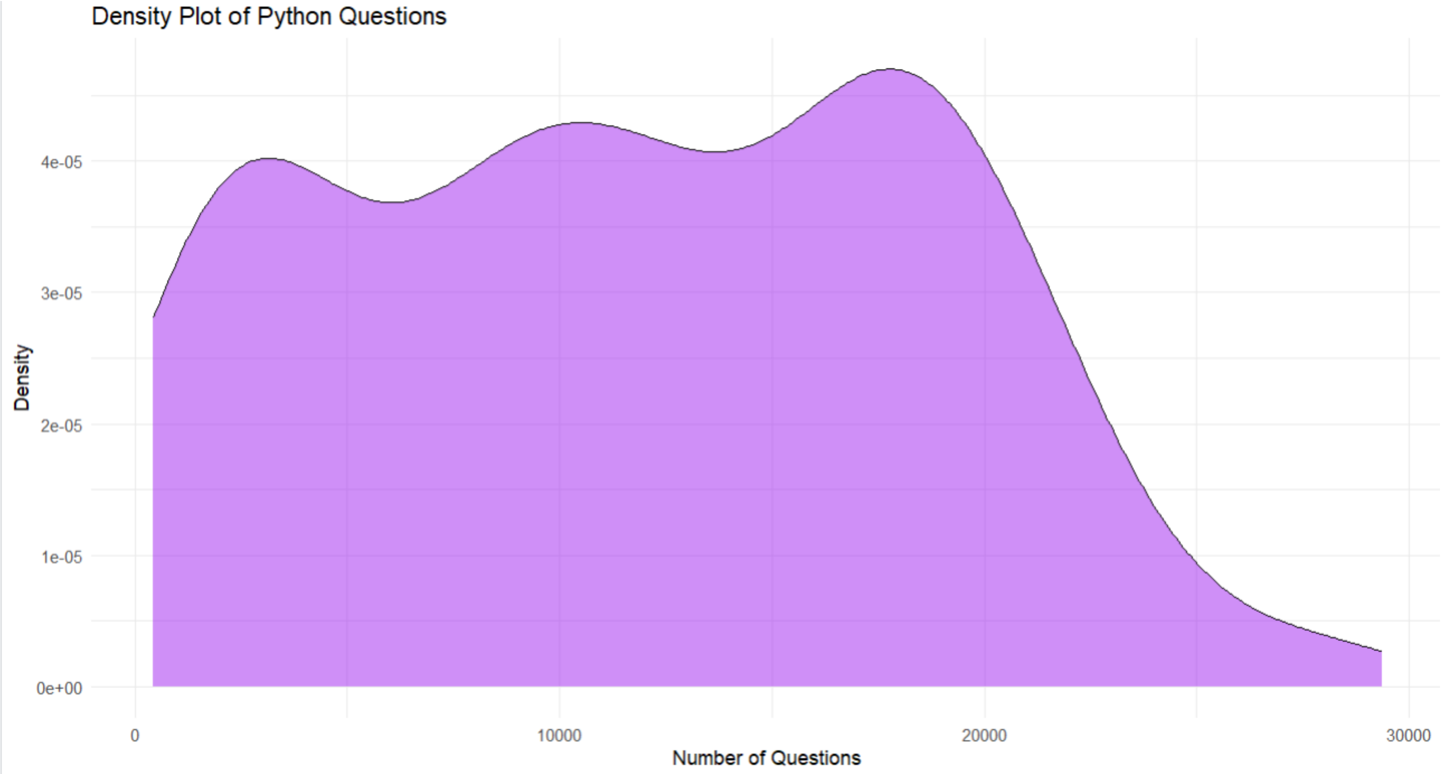
```
> print(monthly_avg)
# A tibble: 12 × 19
   Month Python Javascript   PHP TypeScript Swift  Ruby    Go   SQL Kotlin Scala Shell
   <ord>  <dbl>      <dbl> <dbl>      <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl>
 1 Jan   11305.      9800. 7808.      1213. 1740. 1232.  387. 3423.   515.  552.  481.
 2 Feb   11271.      9885. 7684.      1184. 1658. 1190.  368. 3429.   475.  555.  496.
 3 Mar   12905.     11901. 8857.      1307. 1910. 1379.  418. 4047.   518.  663.  559.
 4 Apr   12599.     11544. 8466.      1253  1826. 1308.  397. 3852.   505.  648   531.
 5 May   12555.     11106. 8394.      1277. 1789  1296.  388. 3827.   527.  639.  508.
 6 Jun   12063.     10205  7910.      1245. 1966. 1249.  389  3696.   529.  638.  503.
 7 Jul   12541.     10438  8331.      1308. 2022. 1300.  412. 3779.   529.  646.  514.
 8 Aug   11883.      9994. 8122.      1306. 1931. 1278.  429. 3622.   533.  619.  497.
 9 Sep   10458.      9333. 7254.      1167. 1705. 1137.  382. 3402.   483.  583.  456.
10 Oct   12286.     10775. 7915.      1287. 1857. 1262.  409. 3760.   534.  638.  524.
11 Nov   11578.     10154. 7337.      1171. 1671. 1134   373. 3479.   519.  583.  489.
12 Dec   10541.      9217. 6979.      1108. 1557. 1089.  354. 3168.   464.  535.  445.
# ℹ 7 more variables: C <dbl>, HTML <dbl>, Objective.C <dbl>, Perl <dbl>,
#   Matlab <dbl>, R <dbl>, Java <dbl>
> 
```

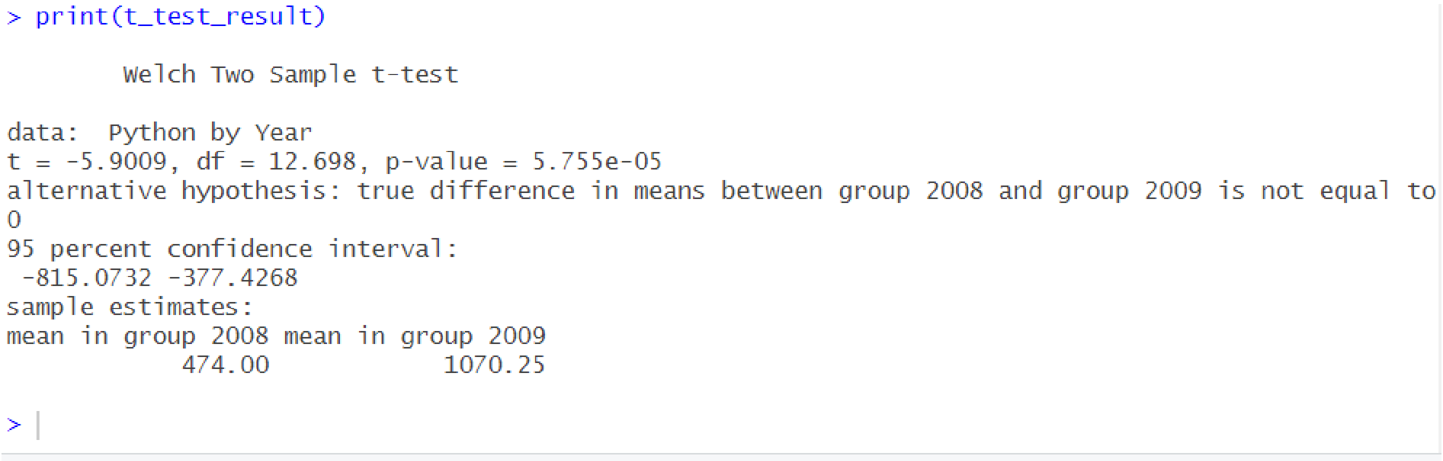



Histogram of Python Questions
Frequency
10.0
7.5
5.0
2.5
0.0
0
10000
20000
30000
Number of Questions

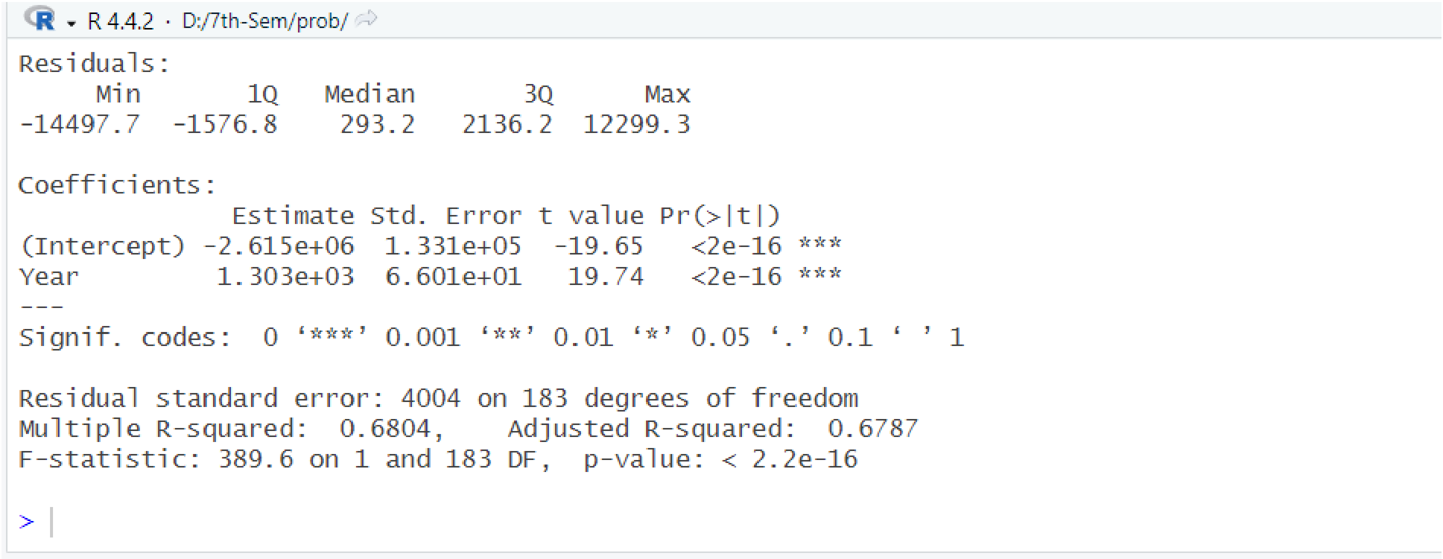

Boxplot of Questions by Language
Programming Language
TypeScript
Swift
SQL
Shell
Scala
Ruby
R
Python
PHP
Perl
Objective.C
Matlab
Kotlin
Javascript
Java
HTML
Go
C
0
10000
20000
30000
Number of Questions
Language
C
Go
HTML
Java
Javascript
Kotlin
Matlab
Objective.C
Perl
PHP
Python
R
Ruby
Scala
Shell
SQL
Swift
TypeScript


Correlation Between Python and JavaScript Questions
JavaScript Questions
Python Questions
0
5000
10000
15000
20000
0
10000
20000
30000

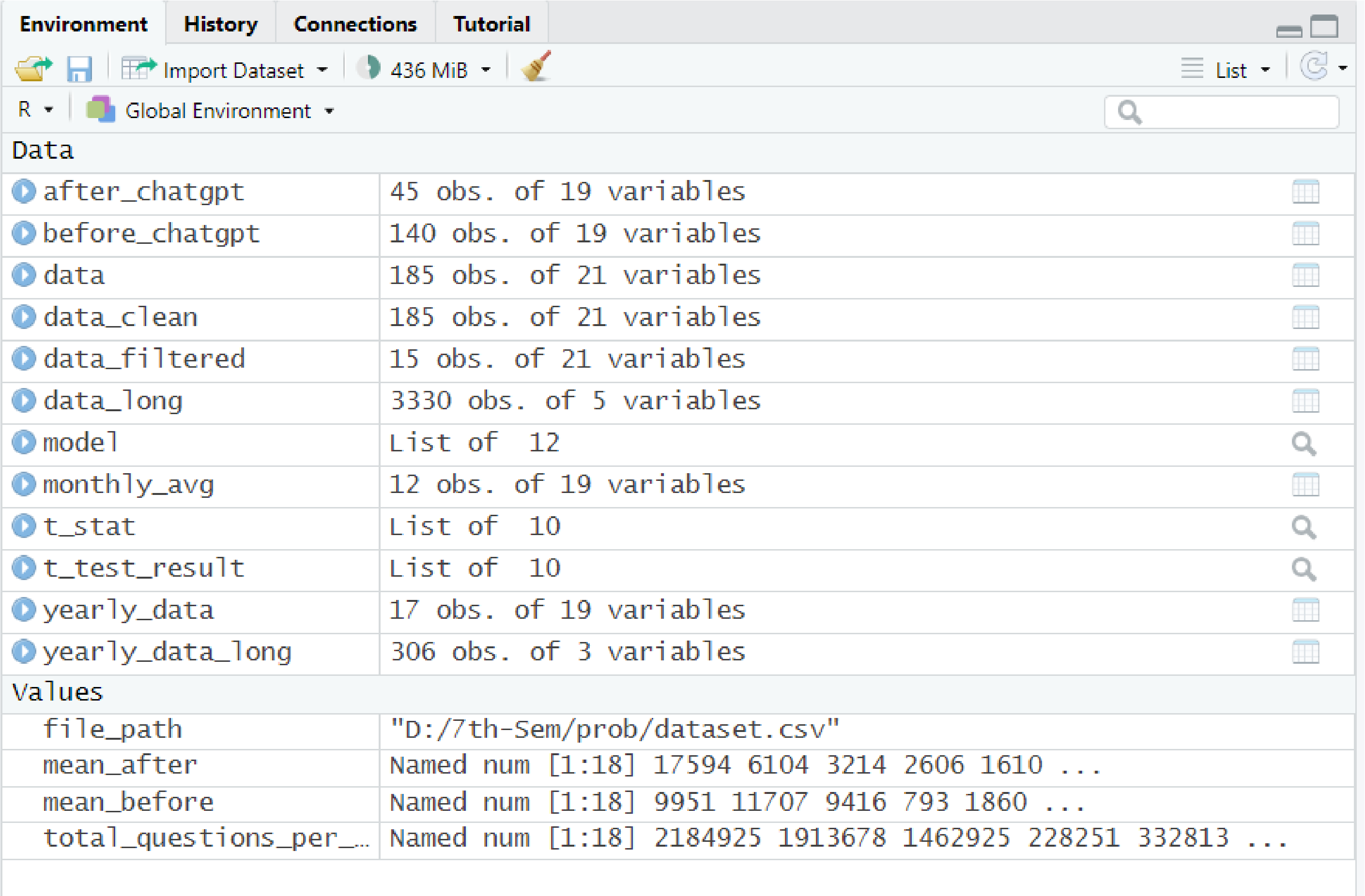

Density Plot of Python Questions
Density
Number of Questions
0e+00
1e-05
2e-05
3e-05
4e-05
0
10000
20000
30000



```
> print(t_test_result)

        Welch Two Sample t-test

data:  Python by Year
t = -5.9009, df = 12.698, p-value = 5.755e-05
alternative hypothesis: true difference in means between group 2008 and group 2009 is not equal to
0
95 percent confidence interval:
 -815.0732 -377.4268
sample estimates:
mean in group 2008 mean in group 2009 
            474.00            1070.25 

> 
```



```
Residuals:
     Min       1Q   Median       3Q      Max 
-14497.7  -1576.8    293.2   2136.2  12299.3 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -2.615e+06  1.331e+05  -19.65   <2e-16 ***
Year         1.303e+03  6.601e+01   19.74   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4004 on 183 degrees of freedom
Multiple R-squared:  0.6804,	Adjusted R-squared:  0.6787 
F-statistic: 389.6 on 1 and 183 DF,  p-value: < 2.2e-16

> 
```



```
Data
after_chatgpt          45 obs. of 19 variables
before_chatgpt         140 obs. of 19 variables
data                   185 obs. of 21 variables
data_clean             185 obs. of 21 variables
data_filtered          15 obs. of 21 variables
data_long              3330 obs. of 5 variables
model                  List of  12
monthly_avg            12 obs. of 19 variables
t_stat                 List of  10
t_test_result          List of  10
yearly_data            17 obs. of 19 variables
yearly_data_long       306 obs. of 3 variables
Values
file_path              "D:/7th-Sem/prob/dataset.csv"
mean_after             Named num [1:18] 17594 6104 3214 2606 1610 ...
mean_before            Named num [1:18] 9951 11707 9416 793 1860 ...
total_questions_per_…  Named num [1:18] 2184925 1913678 1462925 228251 332813 ...
```

5. Codes

```
# Install necessary packages

install.packages(c("tidyverse", "lubridate", "ggplot2", "dplyr", "forecast", "readr"))

# Load required libraries

library(tidyverse)

library(lubridate)

library(ggplot2)

library(dplyr)

library(forecast)

# Specify the file path to your dataset

file_path <- "D:/7th-Sem/prob/dataset.csv"

# Load the dataset
```

```
data <- read.csv(file_path)
# Preview data structure
head(data)
str(data)
# Check for missing values
cat("Total missing values:", sum(is.na(data)), "\n")
# Remove missing values if any
data_clean <- na.omit(data)
# Rename 'Month' column to 'Date'
colnames(data)[colnames(data) == "Month"] <- "Date"
# Convert 'Date' column to Date type
data$Date <- as.Date(data$Date, format = "%Y-%m-%d")
#Filter Data and Select Relevant Columns
# Filter for data from 2008 onward and select relevant columns
data <- data %>%
  filter(as.numeric(format(Date, "%Y")) >= 2008) %>%
  select(Date, Python, Javascript, PHP, TypeScript, Swift, Ruby, Go, SQL, Kotlin, Scala, Shell, C, HTML, Objective.C, Perl, Matlab, R, Java)
# Check structure after filtering
str(data)
# Verify missing values for a specific column
cat("Missing values in 'Python':", sum(is.na(data$Python)), "\n")
#Plot Total Questions Over Time
ggplot(data, aes(x = Date)) +
  geom_line(aes(y = Python, color = "Python")) +
  geom_line(aes(y = Javascript, color = "JavaScript")) +
  geom_line(aes(y = PHP, color = "PHP")) +
  labs(title = "Total Questions Over Time",
       x = "Date",
       y = "Total Questions",
       color = "Languages") +
  theme_minimal()
#Statistical Analysis (T-Test)
# Split data into pre- and post-June 2020
before_chatgpt <- data %>% filter(Date < "2020-06-01")
after_chatgpt <- data %>% filter(Date >= "2020-06-01")
# Perform T-Test for Python
t_stat <- t.test(before_chatgpt$Python, after_chatgpt$Python)
print(t_stat)
#Total Questions by Language
# Sum total questions for each language
total_questions_per_lang <- colSums(data[,-1], na.rm = TRUE)
# Plot bar chart for total questions
barplot(sort(total_questions_per_lang, decreasing = TRUE),
        col = rainbow(length(total_questions_per_lang)),
        main = "Total Questions Per Programming Language",
        las = 2)
#Yearly Trends by Language
# Add a Year column
data$Year <- year(data$Date)
```

```r
# Aggregate by year
yearly_data <- data %>%
  group_by(Year) %>%
  summarise(across(-Date, sum, na.rm = TRUE))
# Convert to long format for plotting
yearly_data_long <- yearly_data %>%
  pivot_longer(-Year, names_to = "Language", values_to = "Total_Questions")
# Plot yearly trends
ggplot(yearly_data_long, aes(x = Year, y = Total_Questions, color = Language)) +
  geom_line() +
  labs(title = "Yearly Trends of Programming Languages",
       x = "Year",
       y = "Total Questions") +
  theme_minimal()
#Distribution of Questions
# Pie chart for language distribution
pie(total_questions_per_lang,
    labels = names(total_questions_per_lang),
    col = rainbow(length(total_questions_per_lang)),
    main = "Distribution of Total Questions by Language")
#Monthly Average Questions
# Extract month from date
data$Month <- month(data$Date, label = TRUE)

# Calculate monthly averages
```

```
monthly_avg <- data %>%
  group_by(Month) %>%
  summarise(across(-c(Date, Year), mean, na.rm = TRUE))
# Plot monthly averages for Python
ggplot(monthly_avg, aes(x = as.numeric(Month), y = Python)) +
  geom_line(color = "blue") +
  scale_x_continuous(breaks = 1:12, labels = month.abb) +
  labs(title = "Monthly Average Questions (Python)",
       x = "Month",
       y = "Average Questions") +
  theme_minimal()

print(monthly_avg)
#Histogram
ggplot(data, aes(x = Python)) +
  geom_histogram(binwidth = 500, fill = "skyblue", color = "black") +
  labs(title = "Histogram of Python Questions",
       x = "Number of Questions",
       y = "Frequency") +
  theme_minimal()
#Boxplot
data_long <- data %>%
  pivot_longer(cols = -c(Date, Year, Month), names_to = "Language", values_to = "Questions")

ggplot(data_long, aes(x = Language, y = Questions, fill = Language)) +
  geom_boxplot() +
  coord_flip() +
  labs(title = "Boxplot of Questions by Language",
       x = "Programming Language",
       y = "Number of Questions") +
  theme_minimal()
#Scatter Plot (Correlation)
ggplot(data, aes(x = Python, y = Javascript)) +
  geom_point(color = "blue") +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(title = "Correlation Between Python and JavaScript Questions",
       x = "Python Questions",
       y = "JavaScript Questions") +
  theme_minimal()
#Density Plot
ggplot(data, aes(x = Python)) +
  geom_density(fill = "purple", alpha = 0.5) +
  labs(title = "Density Plot of Python Questions",
       x = "Number of Questions",
       y = "Density") +
  theme_minimal()
#Hypothesis Testing (T-Test)
# Filter data for specific years
data_filtered <- data %>% filter(Year %in% c(2008, 2009))
# Perform T-Test
```

```
t_test_result <- t.test(Python ~ Year, data = data_filtered)
print(t_test_result)
#Linear Regression
model <- lm(Python ~ Year, data = data)
summary(model)
```

6. Conclusion

The analysis of Stack Overflow trends in R provided insights into the evolution of programming languages' popularity over time, highlighting significant shifts in usage patterns. The data revealed Python's consistent growth in popularity, reflecting its broad applicability in data science and machine learning. In contrast, older languages like PHP and Ruby showed a decline, indicating shifts in development practices and technology preferences. Statistical tests, such as T-tests, identified notable differences in question volume before and after significant events (e.g., the introduction of ChatGPT). Visualizations, including line charts and bar plots, effectively illustrated these trends, showcasing Python's dominance and emerging interest in languages like TypeScript and Go. Overall, the study underlined how Stack Overflow serves as a valuable resource for tracking industry trends and forecasting future developments in programming.