

Aditya Shroff
101539669



PROGRAMMING FUNDAMENTALS FOR ANALYTICS

R_Assignment

Name: Aditya Shroff (101539669)
Professor: Mohsen Selseleh

Step 1: Load the dataset and necessary libraries

The screenshot shows the RStudio interface. In the top-left pane, there is an R script titled "Untitled1.R" containing the following code:

```
1 # Install necessary libraries (only needs to be done once)
2 if (!require(dplyr)) install.packages("dplyr", dependencies=TRUE)
3 if (!require(ggplot2)) install.packages("ggplot2", dependencies=TRUE)
4 if (!require(corrplot)) install.packages("corrplot", dependencies=TRUE)
5 if (!require(tidyr)) install.packages("tidyr", dependencies=TRUE)
6
7 # Load libraries into your R session
8 library(dplyr) # Data manipulation
9 library(ggplot2) # Data visualization
10 library(corrplot) # Correlation matrix visualization
11 library(tidyr) # Data reshaping
12
```

In the bottom-left pane, the "Console" tab is active, showing the same code being run in the R environment. The output includes messages about package installations and the loading of the "corrplot" package.

The right side of the interface includes the "Environment" and "Data" panes, which show the global environment and the "Customer..." dataset respectively. The "Files" pane on the far right lists various files and folders on the user's system.

2. Load the Dataset and Handle Missing Values

The screenshot shows the RStudio interface again. The R script "Untitled1.R" now includes additional code for handling missing values:

```
1 # Install necessary libraries (only needs to be done once)
2 if (!require(dplyr)) install.packages("dplyr", dependencies=TRUE)
3 if (!require(ggplot2)) install.packages("ggplot2", dependencies=TRUE)
4 if (!require(corrplot)) install.packages("corrplot", dependencies=TRUE)
5 if (!require(tidyr)) install.packages("tidyr", dependencies=TRUE)
6
7 # Load libraries into your R session
8 library(dplyr) # Data manipulation
9 library(ggplot2) # Data visualization
10 library(corrplot) # Correlation matrix visualization
11 library(tidyr) # Data reshaping
12
13
14
15 # Load the dataset
16 datainput <- read.csv("~/Users/adityarajoriya/Downloads/Customer Churn.csv")
17
18 # Handle Missing Values by Imputing with Mean
19 datainput <- datainput %>%
20   mutate(across(where(is.numeric), ~ ifelse(is.na(.), mean(., na.rm = TRUE), .)))
21
```

The "Console" tab shows the execution of this code. The output includes messages about the loading of the "corrplot" package and the execution of the data manipulation code.

The right side of the interface remains consistent with the previous screenshot, showing the "Environment", "Data", and "Files" panes.

3. Calculate IQR and Remove Outliers

The screenshot shows the RStudio interface with the following details:

- Code Editor:** Contains R code for calculating IQR bounds and removing outliers from the Customer.Churn dataset.
- Console:** Shows the execution of the R code in the R 4.4.1 environment.
- Environment:** Shows the global environment with variables like IQR, lower_bound, Q1, Q3, upper_bound, and datainput.
- Files:** Shows a file tree with various R scripts and notebooks.

```

library(corrplot) # Correlation matrix visualization
library(tidyverse)

# Load the dataset
datainput <- read.csv("/Users/adityarajoriya/Downloads/Customer Churn.csv")

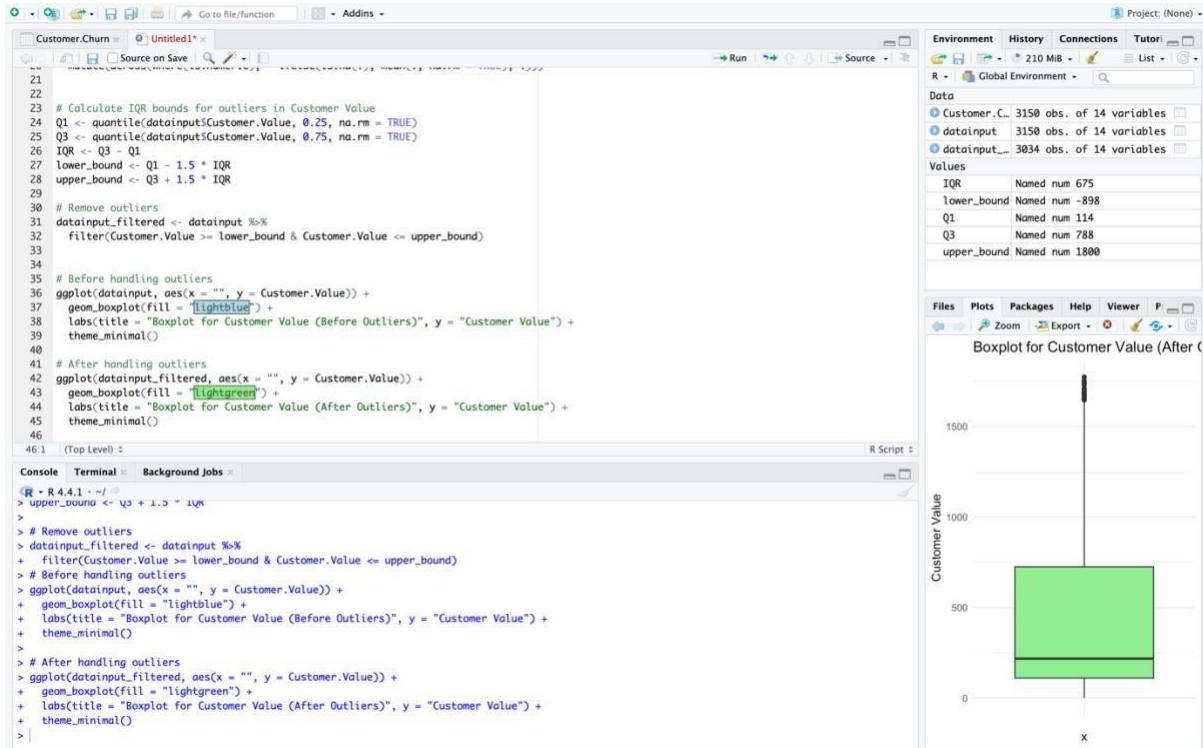
# Handle Missing Values by Imputing with Mean
datainput <- datainput %>%
  mutate(across(where(is.numeric), ~ ifelse(is.na(.), mean(., na.rm = TRUE), .)))

# Calculate IQR bounds for outliers in Customer Value
Q1 <- quantile(datainput$Customer.Value, 0.25, na.rm = TRUE)
Q3 <- quantile(datainput$Customer.Value, 0.75, na.rm = TRUE)
IQR <- Q3 - Q1
lower_bound <- Q1 - 1.5 * IQR
upper_bound <- Q3 + 1.5 * IQR

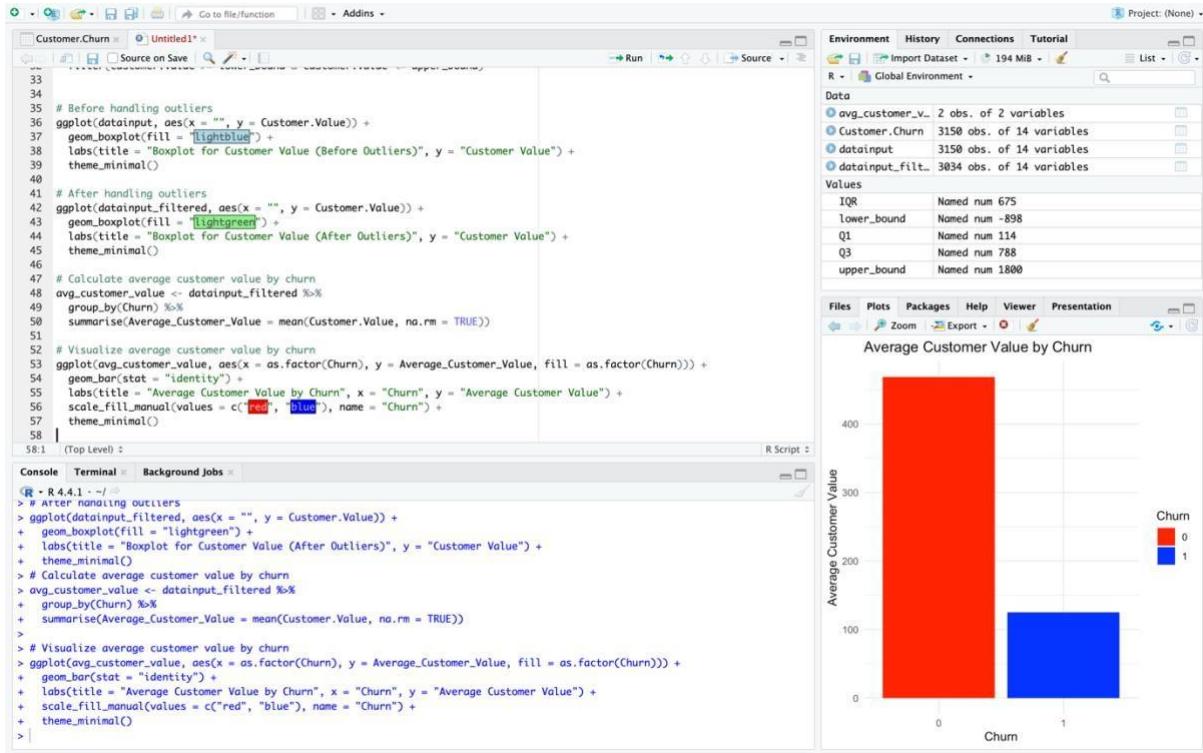
# Remove outliers
datainput_filtered <- datainput %>%
  filter(Customer.Value >= lower_bound & Customer.Value <= upper_bound)

```

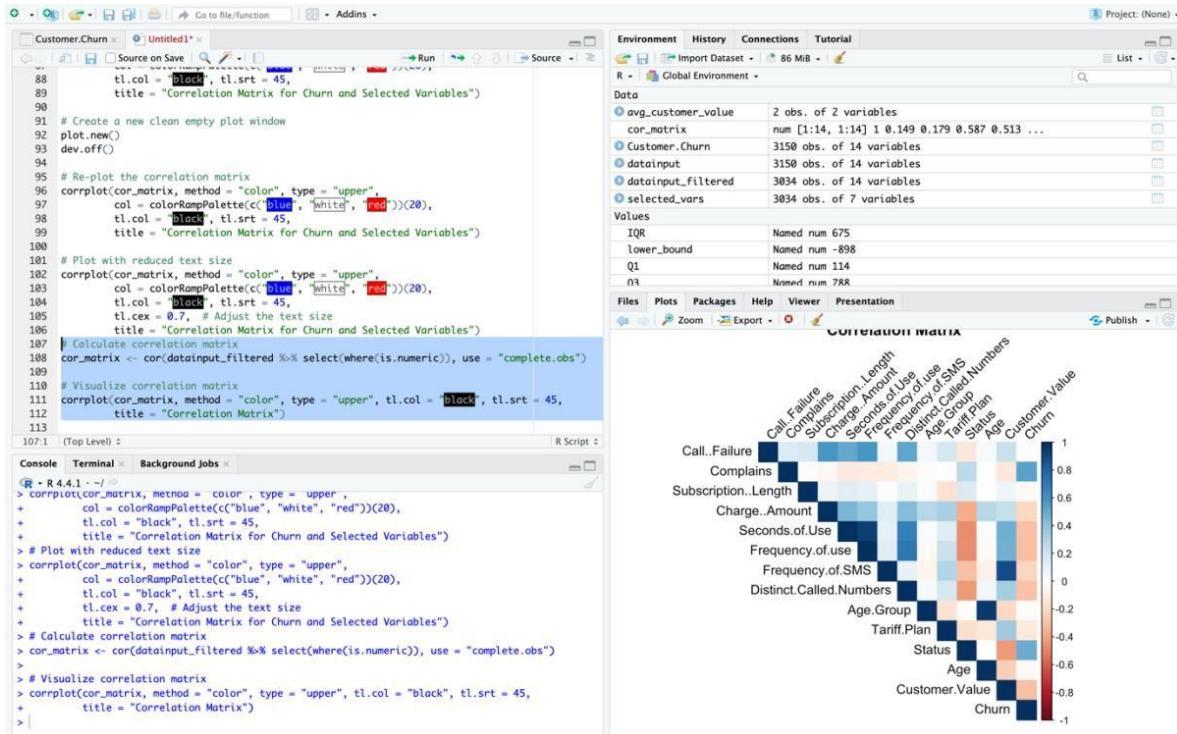
4. Visualize Before and After Outlier Removal



5. Group Data and Visualize Average Customer Value by Churn



6. Create and Visualize Correlation Matrix



7. Display Column Names

The screenshot shows the RStudio interface with the 'Customer.Churn' script open in the code editor. The script contains R code for calculating average customer value by churn, visualizing it, calculating a correlation matrix, and displaying column names. The 'Data' pane shows variables like avg_customer_value, cor_matrix, Customer.Churn, datainput, datainput_filtered, and selected_vars. The 'Plots' pane displays a correlation matrix heatmap titled 'Correlation Matrix' for the entire dataset.

```

45 theme_minimal()
46
47 # Calculate average customer value by churn
48 avg_customer_value <- datainput_filtered %>%
49   group_by(Churn) %>%
50   summarise(Average_Customer_Value = mean(Customer.Value, na.rm = TRUE))
51
52 # Visualize average customer value by churn
53 ggplot(avg_customer_value, aes(x = as.factor(Churn), y = Average_Customer_Value, fill = as.factor(Churn))) +
54   geom_bar(stat = "identity") +
55   labs(title = "Average Customer Value by Churn", x = "Churn", y = "Average Customer Value") +
56   scale_fill_manual(values = c("red", "blue"), name = "Churn") +
57   theme_minimal()
58
59
60 # Calculate correlation matrix
61 cor_matrix <- cor(datainput_filtered %>% select(where(is.numeric)), use = "complete.obs")
62
63 # Visualize correlation matrix
64 corrplot(cor_matrix, method = "color", type = "upper", tl.col = "black", tl.srt = 45,
65           title = "Correlation Matrix")
66
67
68 # Display column names
69 colnames(datainput_filtered)
70
71 (Top Level) :

```

```

Console Terminal Background Jobs
[R - R 4.4.1 - ~]
+ geom_bar$stat = "identity" j +
+ labs(title = "Average Customer Value by Churn", x = "Churn", y = "Average Customer Value") +
+ scale_fill_manual(values = c("red", "blue"), name = "Churn") +
+ theme_minimal()
> # Calculate correlation matrix
> cor_matrix <- cor(datainput_filtered %>% select(where(is.numeric)), use = "complete.obs")
>
> # Visualize correlation matrix
> corrplot(cor_matrix, method = "color", type = "upper", tl.col = "black", tl.srt = 45,
+           title = "Correlation Matrix")
> # Display column names
> colnames(datainput_filtered)
[1] "Call..Failure"          "Complaints"            "Subscription..Length" "Charge..Amount"
[5] "Seconds.of.Use"         "Frequency.of.use"      "Frequency.of.SMS"     "Distinct.Called.Numbers"
[9] "Age.Group"              "Tariff..Plan"          "Status"                "Age"
[13] "Customer.Value"        "Churn"
> 

```

8. Select Specific Variables and Create Another Correlation Matrix

The screenshot shows the RStudio interface with the 'Customer.Churn' script open. The script now includes code to select specific variables for correlation analysis, calculate a correlation matrix for these variables, and visualize it. The 'Data' pane shows the selected variables: avg_customer_value, cor_matrix, Customer.Churn, datainput, datainput_filtered, and selected_vars. The 'Plots' pane displays a correlation matrix heatmap titled 'Correlation Matrix for Churn and Selected Variables' for the subset of variables.

```

107 # Calculate correlation matrix
108 cor_matrix <- cor(datainput_filtered %>% select(where(is.numeric)), use = "complete.obs")
109
110 # Visualize correlation matrix
111 corrplot(cor_matrix, method = "color", type = "upper", tl.col = "black", tl.srt = 45,
112           title = "Correlation Matrix")
113
114
115 # Select specific variables for correlation analysis
116 selected_vars <- data.frame(
117   Churn = datainput_filtered$Churn,
118   Customer_Value = datainput_filtered$Customer.Value",
119   Subscription_Length = datainput_filtered$Subscription..Length",
120   Frequency_of_Use = datainput_filtered$Frequency.of.use",
121   Complaints = datainput_filtered$Complaints,
122   Age = datainput_filtered$Age,
123   Call_Failure = datainput_filtered$Call..Failure"
124 )
125
126 # Calculate and visualize correlation matrix
127 cor_matrix <- cor(selected_vars, use = "complete.obs")
128 corrplot(cor_matrix, method = "color", type = "upper",
129           col = colorRampPalette(c("blue", "white", "red"))(20),
130           tl.col = "black", tl.srt = 45,
131           title = "Correlation Matrix for Churn and Selected Variables")
132
133 (Top Level) :

```

```

Console Terminal Background Jobs
[R - R 4.4.1 - ~]
> selected_vars <- data.frame(
+   Churn = datainput_filtered$Churn,
+   Customer_Value = datainput_filtered$Customer.Value",
+   Subscription_Length = datainput_filtered$Subscription..Length",
+   Frequency_of_Use = datainput_filtered$Frequency.of.use",
+   Complaints = datainput_filtered$Complaints,
+   Age = datainput_filtered$Age,
+   Call_Failure = datainput_filtered$Call..Failure"
+ )
+
> # Calculate and visualize correlation matrix
> cor_matrix <- cor(selected_vars, use = "complete.obs")
> corrplot(cor_matrix, method = "color", type = "upper",
+           col = colorRampPalette(c("blue", "white", "red"))(20),
+           tl.col = "black", tl.srt = 45,
+           title = "Correlation Matrix for Churn and Selected Variables")

```

9. Aggregate Data and Calculate Summary Statistics

The screenshot shows an RStudio interface with the following components:

- Code Editor:** Displays R code for calculating a correlation matrix and summarizing data by Churn.
- Console:** Shows the execution of the R code, resulting in a 2x4 data frame with columns: Churn, Mean_Customer_Value, Total_Customer_Value, and Mean_Seconds_of_Use.
- Environment View:** Shows variables like agg_results, avg_cu, cor_ma, etc., and their values.
- Data View:** Shows a correlation matrix heatmap for variables: Churn, Customer_Value, Duration_Length, Frequency_of_Use, Complaints, Age, and Call_Failure.

```

121 Complaints = datainput_filtered$Complaints,
122 Age = datainput_filtered$Age,
123 Call_Failure = datainput_filtered$Call..Failure'
124 )
125
126 # Calculate and visualize correlation matrix
127 cor_matrix <- cor(selected_vars, use = "complete.obs")
128 corplot(cor_matrix, method = "color", type = "upper",
129 col = colorRampPalette(c("blue", "white", "red"))(20),
130 tl.col = "black", tl.srt = 45,
131 title = "Correlation Matrix for Churn and Selected Variables")
132
133
134
135 # Aggregate results by churn
136 agg_results <- datainput_filtered %>%
137 group_by(Churn) %>%
138 summarise(
139   Mean_Customer_Value = mean(Customer.Value, na.rm = TRUE),
140   Total_Customer_Value = sum(Customer.Value, na.rm = TRUE),
141   Mean_Seconds_of_Use = mean(Seconds.of.Use, na.rm = TRUE)
142 )
143
144 # Print aggregated results
145 print(agg_results)
146
135.1 (Top Level) : R Script

```

```

> # Aggregate results by churn
> agg_results <- datainput_filtered %>%
+   group_by(Churn) %>%
+   summarise(
+     Mean_Customer_Value = mean(Customer.Value, na.rm = TRUE),
+     Total_Customer_Value = sum(Customer.Value, na.rm = TRUE),
+     Mean_Seconds_of_Use = mean(Seconds.of.Use, na.rm = TRUE)
+   )
>
> # Print aggregated results
> print(agg_results)
# A tibble: 2 × 4
  Churn Mean_Customer_Value Total_Customer_Value Mean_Seconds_of_Use
  <dbl>        <dbl>            <dbl>           <dbl>
1     0          469.          1189672.         4997.
2     1          125.           61782.          1567.
> 

```

10. Perform T-Test

The screenshot shows an RStudio interface with the following components:

- Code Editor:** Displays R code for calculating a correlation matrix and performing a t-test on Customer.Value ~ Churn.
- Console:** Shows the execution of the R code, resulting in Welch Two Sample t-test output.
- Environment View:** Shows variables like agg_results, avg_cu, cor_ma, etc., and their values.
- Data View:** Shows a correlation matrix heatmap for variables: Churn, Customer_Value, Duration_Length, Frequency_of_Use, Complaints, Age, and Call_Failure.

```

128 corplot(cor_matrix, method = "color", type = "upper",
129 col = colorRampPalette(c("blue", "white", "red"))(20),
130 tl.col = "black", tl.srt = 45,
131 title = "Correlation Matrix for Churn and Selected Variables")
132
133
134
135 # Aggregate results by churn
136 agg_results <- datainput_filtered %>%
137 group_by(Churn) %>%
138 summarise(
139   Mean_Customer_Value = mean(Customer.Value, na.rm = TRUE),
140   Total_Customer_Value = sum(Customer.Value, na.rm = TRUE),
141   Mean_Seconds_of_Use = mean(Seconds.of.Use, na.rm = TRUE)
142 )
143
144 # Print aggregated results
145 print(agg_results)
146
147
148
149
150 # Perform t-test
151 t_test_result <- t.test(Customer.Value ~ Churn, data = datainput_filtered)
152 print(t_test_result)
153
150.1 (Top Level) : R Script

```

```

> # Perform t-test
> t_test_result <- t.test(Customer.Value ~ Churn, data = datainput_filtered)
> print(t_test_result)

Welch Two Sample t-test

data: Customer.Value by Churn
t = 32.523, df = 2650.9, p-value < 2.2e-16
alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
95 percent confidence interval:
 323.0231 364.4729
sample estimates:
mean in group 0 mean in group 1
 468.5594      124.8114
> 

```

11. Perform ANOVA



```

Customer.Churn <- Untitled1* >
Source on Save | Run | Source | Environment | History | Con...
133
134
135 # Aggregate results by churn
136 agg_results <- datainput_filtered %>%
137   group_by(Churn) %>%
138   summarise(
139     Mean_Customer_Value = mean(Customer.Value, na.rm = TRUE),
140     Total_Customer_Value = sum(Customer.Value, na.rm = TRUE),
141     Mean_Seconds_of_Use = mean(Seconds.of.Use, na.rm = TRUE)
142   )
143
144 # Print aggregated results
145 print(agg_results)
146
147
148 # Perform t-test
149 t_test_result <- t.test(Customer.Value ~ Churn, data = datainput_filtered)
150 print(t_test_result)
151
152
153
154 # Perform ANOVA
155 anova_result <- aov(Customer.Value ~ as.factor(Churn), data = datainput_filtered)
156 summary(anova_result)
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559.1 (Top Level) :

```

Console Terminal < Background Jobs >

R > R 4.4.1 - ~/

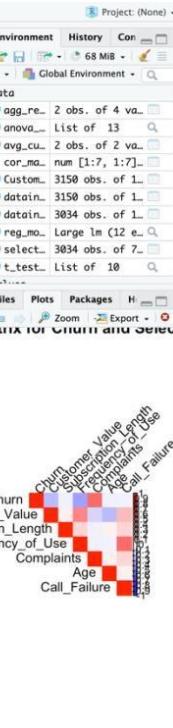
t = 2c.2c3, df = 2090.9, p-value < 2.2e-16
alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
95 percent confidence interval:
323.0231 364.4729
sample estimates:
mean in group 0 mean in group 1
468.5594 124.8114

> # Perform ANOVA
> anova_result <- aov(Customer.Value ~ as.factor(Churn), data = datainput_filtered)
> summary(anova_result)

	Sum Sq	Mean Sq	F value	Pr(>F)
as.factor(Churn)	1	48947745	48947745	291 <2e-16 ***
Residuals	3032	510052995	168223	

Signif. codes: 0 '****' 0.001 '***' 0.05 '.' 0.1 ' ' 1

12. Build and Summarize Regression Model



```

Customer.Churn <- Untitled1* >
Source on Save | Run | Source | Environment | History | Con...
160 # Perform t-test
161 t_test_result <- t.test(Customer.Value ~ Churn, data = datainput_filtered)
162 print(t_test_result)
163
164
165 # Perform ANOVA
166 anova_result <- aov(Customer.Value ~ as.factor(Churn), data = datainput_filtered)
167 summary(anova_result)
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259.1 (Top Level) :

```

Console Terminal < Background Jobs >

R > R 4.4.1 - ~/

> # Summarize the regression model
> summary(reg_model)

Call:

lm(formula = Customer.Value ~ Seconds.of.Use + Frequency.of.use +
Age, data = datainput_filtered)

Residuals:

Mn	Q1	Median	Q3	Max
-374.66	-228.05	-144.28	58.62	1349.44

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	539.652667	25.650006	21.039	<2e-16 ***
Seconds.of.Use	0.004061	0.004891	8.293	<2e-16 ***
Frequency.of.use	0.693357	0.360876	1.922	0.0547 .
Age	-11.418617	0.738961	-15.452	<2e-16 ***

Signif. codes: 0 '****' 0.001 '***' 0.01 '**' 0.05 '.' 0.1 ' ' 1

Residual standard error: 361 on 3030 degrees of freedom
Multiple R-squared: 0.2936, Adjusted R-squared: 0.2929
F-statistic: 419.8 on 3 and 3030 DF, p-value: < 2.2e-16