



## American International University-Bangladesh (AIUB)

### Department of Computer Science Faculty of Science & Technology (FST)

#### INTRODUCTION TO DATA SCIENCE

#### Report

Submitted By

Semester: Fall_24_25	
Student Name	Student ID
Md.Shariar Hosain Sanny	21-44677-1

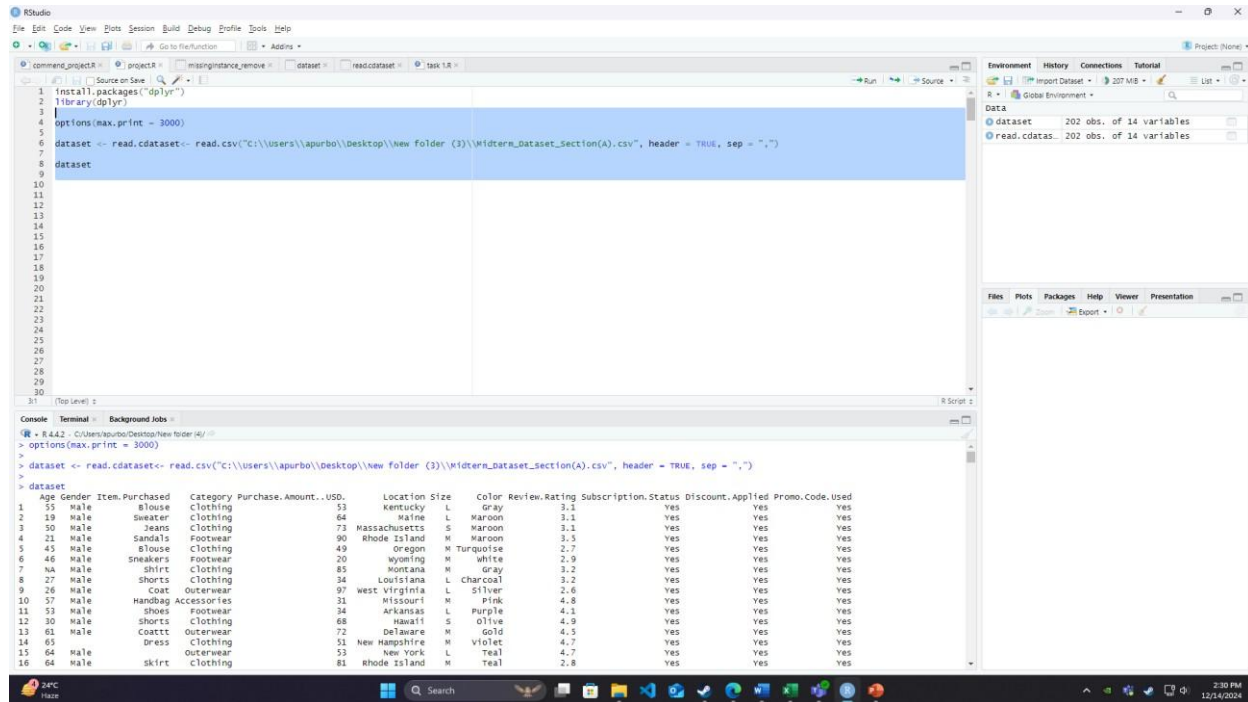
Serial Number	Task	Page Number
1	Short Description	3
1	Loading and Displaying Dataset	3
2	Find and handle the missing values:	3-8
3	Missing value in graph	9
4	Find and remove duplicate values.	9-10
5	Convert the imbalanced data set into the balanced data set	11
6	filtering methods	12-13
7	Convert attributes from numeric to categorical	13-14
8	Convert attributes from categorical to numeric	14-15
9	Outliers	15-16
10	Normalization method	18

## Short Description

The dataset contains detailed transactional information about individual purchases, including attributes such as Age, Gender, Item Purchased, Category, Purchase Amount (USD), Location, Size, Color, Review Rating, Subscription Status, Discount Applied, and Promo Code Used. It also includes behavioral metrics like Previous Purchases and Frequency of Purchases. The data features a mix of numerical values and boolean indicators, making it versatile for analysis. This dataset is labeled, as it includes target variables suitable for predictive analysis, and is designed for use in supervised learning tasks.

## Loading and Displaying Dataset :

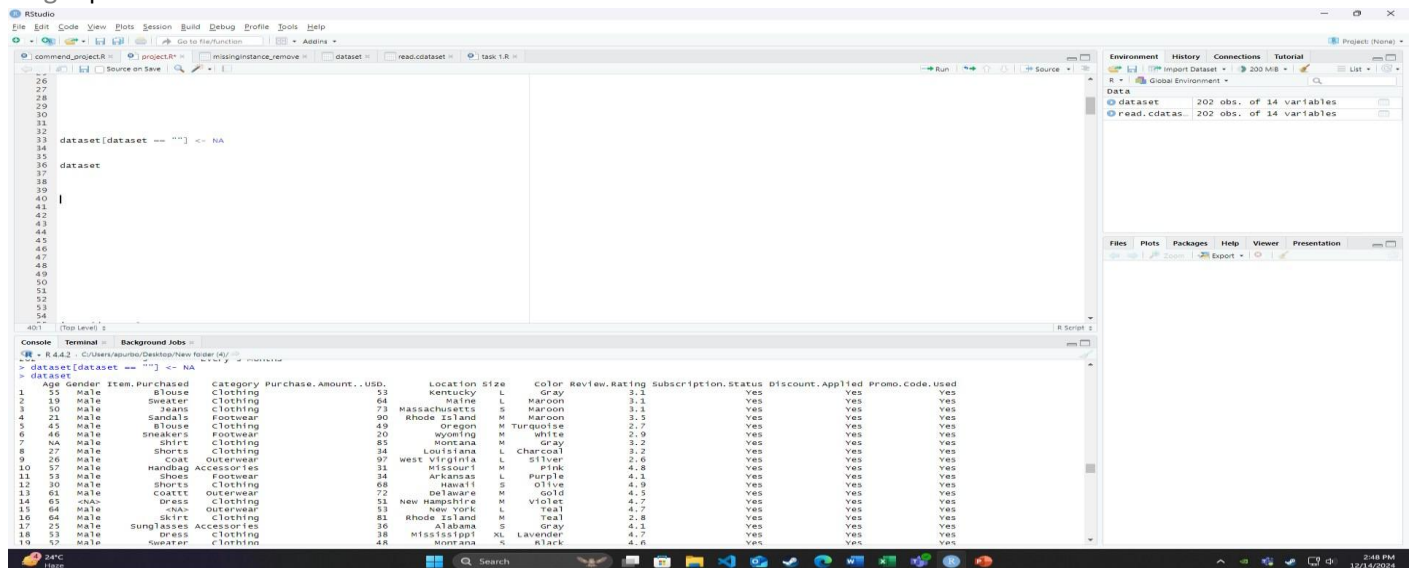
First we import the dataset using read.csv then we view the dataset by calling its variable.



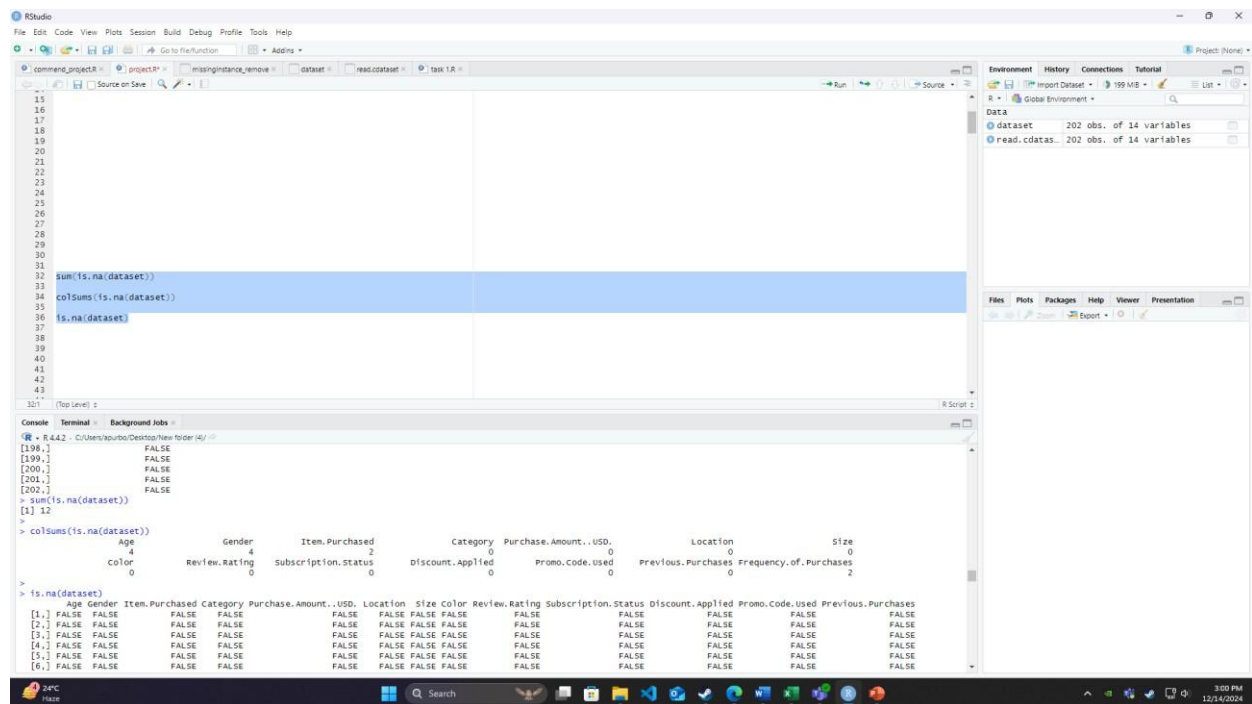
## Find and handle the missing values:

```
> dataset
  Age Gender Item.Purchased Category Purchase.Amount..USD. Location Size Color Review.Rating Subscription.Status Discount.Applied Promo.Code.Used
1  55 Male Blouse clothing 53 Kentucky L Gray 3.1 Yes Yes Yes Yes
2  19 Male Sweater clothing 64 Maine L Maroon 3.1 Yes Yes Yes Yes
3  50 Male Jeans clothing 73 Massachusetts S Maroon 3.1 Yes Yes Yes Yes
4  21 Male Sandals Footwear 90 Rhode Island M Maroon 3.5 Yes Yes Yes Yes
5  45 Male Blouse clothing 49 Oregon M Turquoise 2.7 Yes Yes Yes Yes
6  46 Male Sneakers Footwear 20 Wyoming M white 2.9 Yes Yes Yes Yes
7  NA Male shirt clothing 85 Montana M Gray 3.2 Yes Yes Yes Yes
8  27 Male shorts clothing 34 Louisiana L Charcoal 3.2 Yes Yes Yes Yes
9  26 Male Coat Outerwear 97 West Virginia L Silver 2.6 Yes Yes Yes Yes
10 57 Male Handbag accessories 31 Missouri M Pink 4.8 Yes Yes Yes Yes
11 53 Male shoes Footwear 34 Arkansas L Purple 4.1 Yes Yes Yes Yes
12 30 Male shorts clothing 68 Hawaii S Olive 4.9 Yes Yes Yes Yes
13 61 Male Coatt Outerwear 72 Delaware M Gold 4.5 Yes Yes Yes Yes
14 65 Male Dress clothing 51 New Hampshire M Violet 4.7 Yes Yes Yes Yes
15 64 Male skirt Outerwear 53 New York L Teal 4.7 Yes Yes Yes Yes
16 64 Male skirt clothing 81 Rhode Island M Teal 2.8 Yes Yes Yes Yes
```

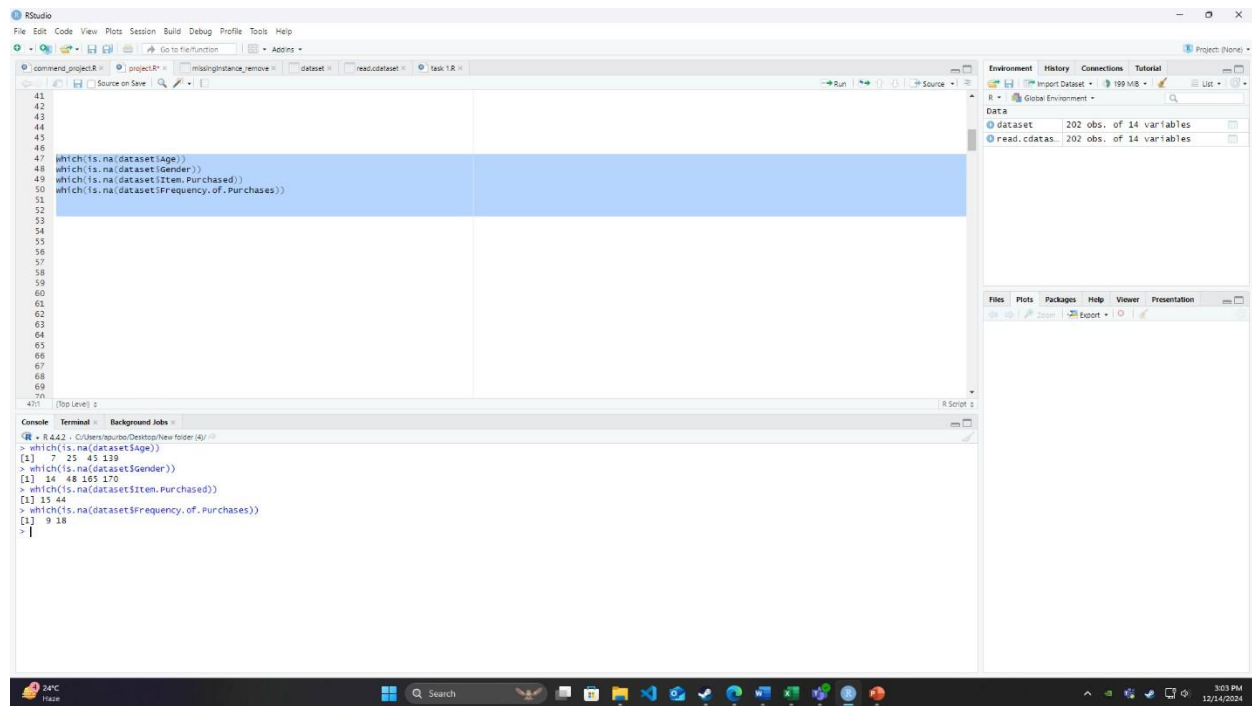
Some of the data are empty so we must first convert them to NA. We convert empty data using `dataset[dataset == ""] <- NA`



Then we use `is.na(dataset)` to find the missing values in the dataset `sum(is.na(dataset))` to get the total count of missing values and `colSums(is.na(dataset))` to count the missing values in each column.



We find the positions of missing values

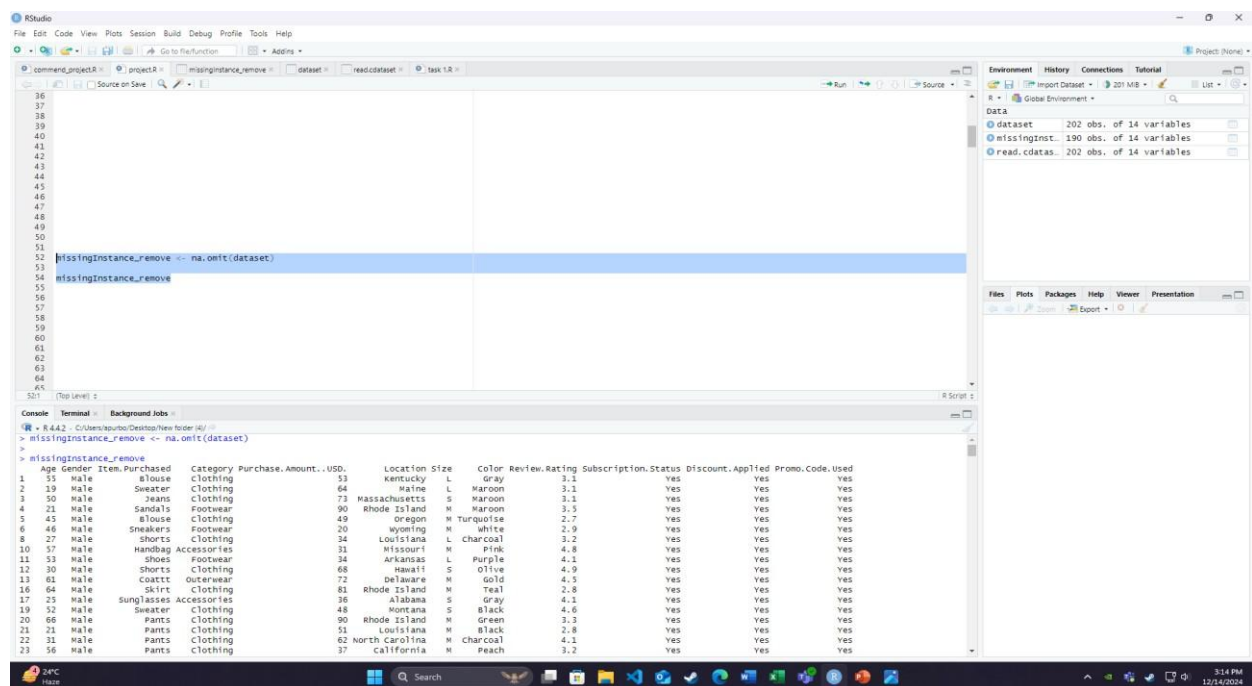


We use two methods handle the missing values

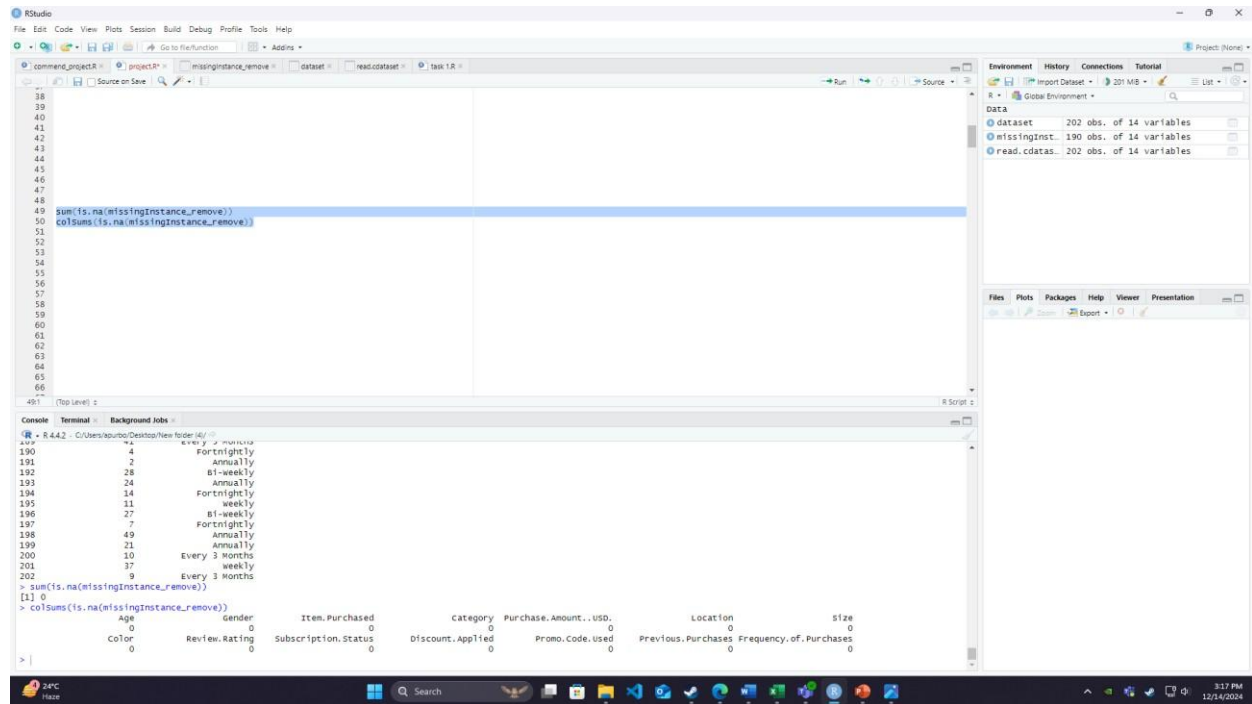
1. Discard Instances
2. Replace by Most Frequent/Average Value

### Discard Instances:

Here we handle missing value by remove it

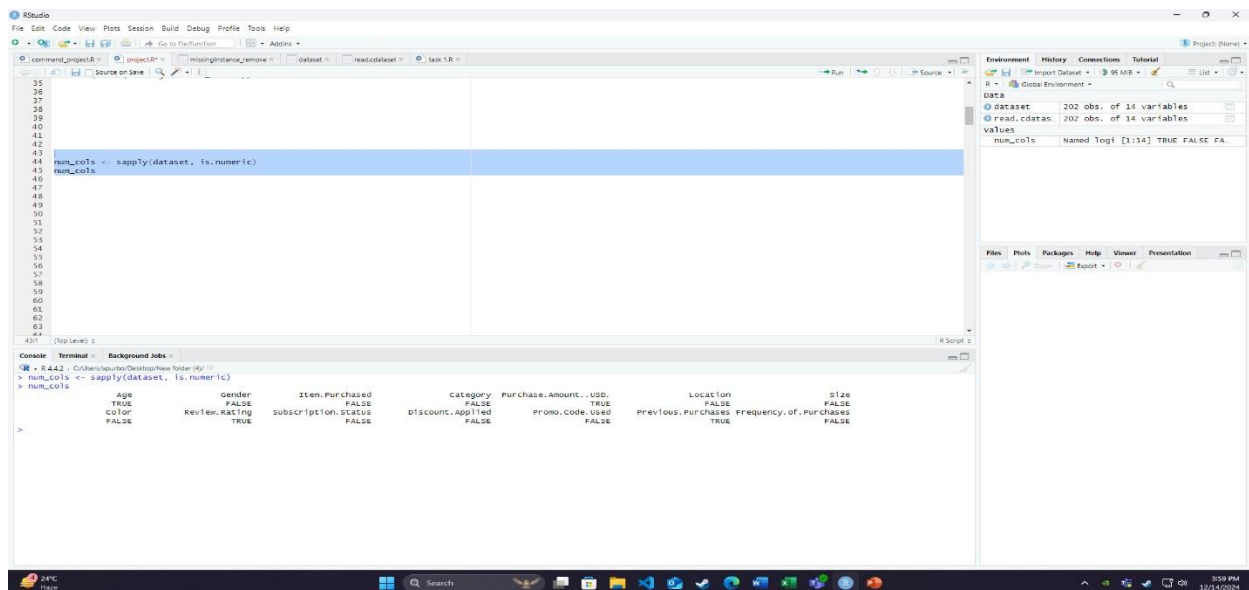


No missing values found.

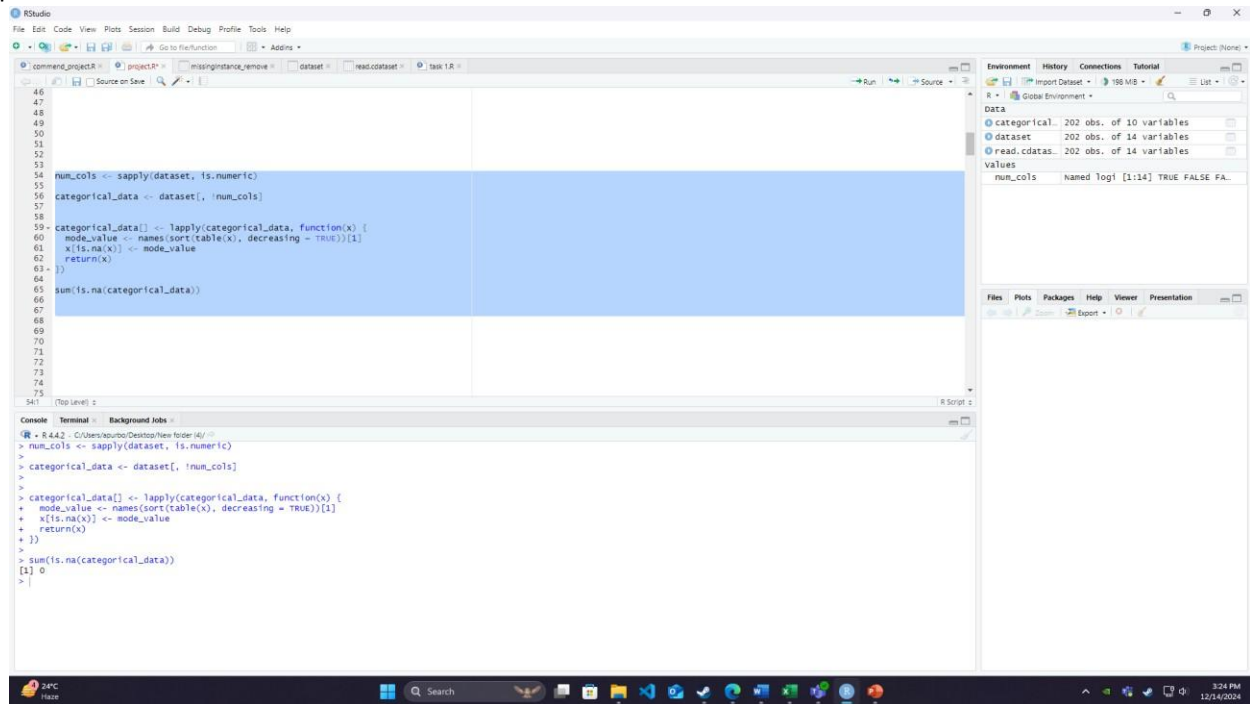


## Replace by Most Frequent/Average Value

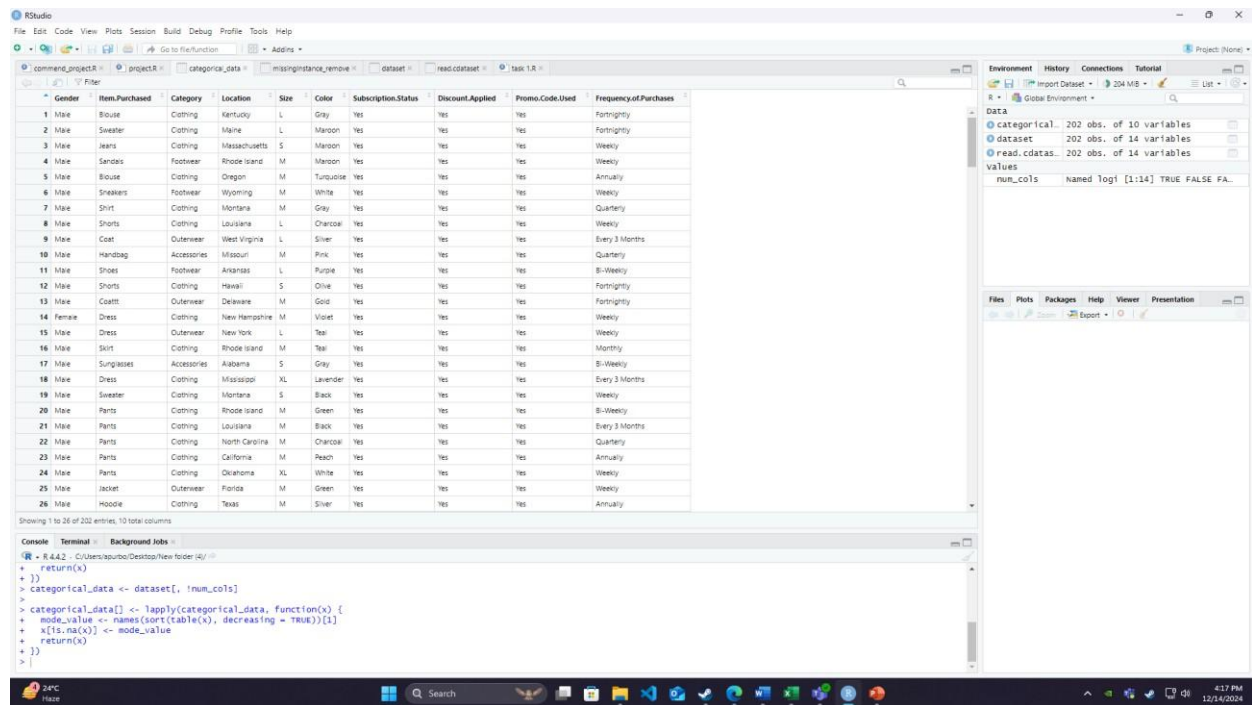
First, we detect categorical and numerical attributes. If the result is true the attribute is numerical, if False it is categorical



In the case of categorical attributes, we handle missing values by replacing them with the most frequent value (mode) in each instance.

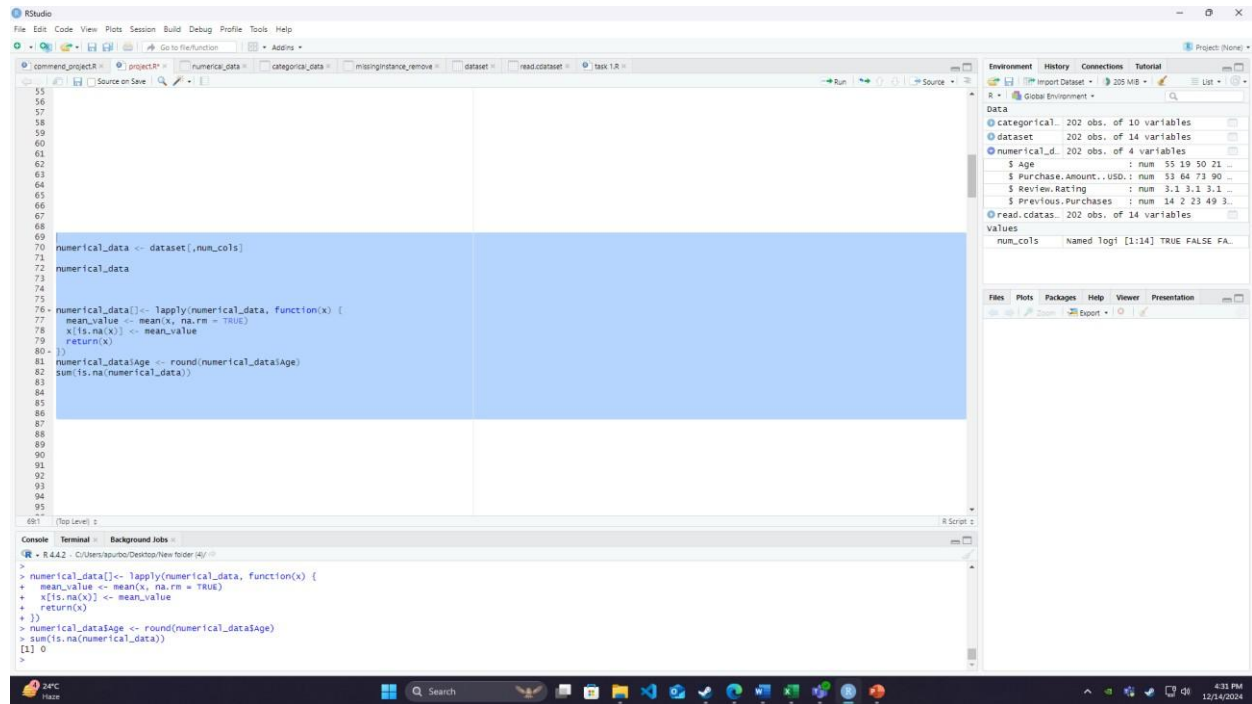


Replacing them with the most frequent value (mode) in each instance.

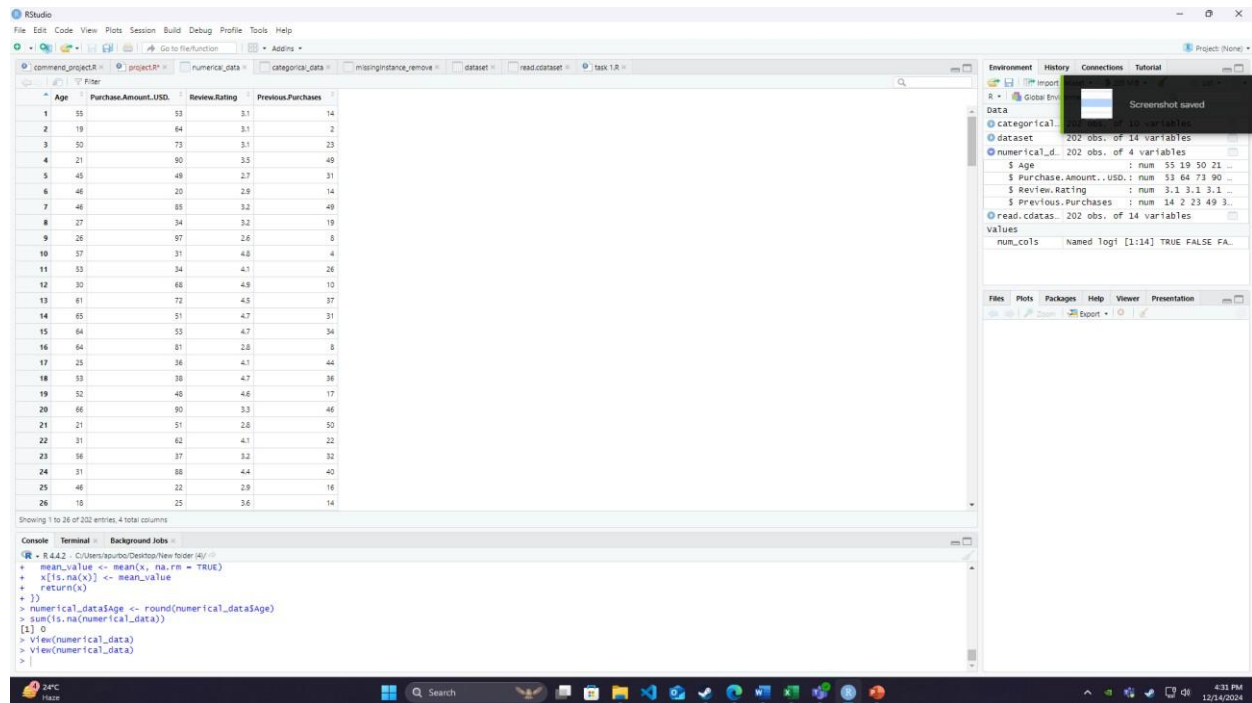


In the case of numerical attributes, we handle missing values by replacing them with the most average value (mean) in each instance.





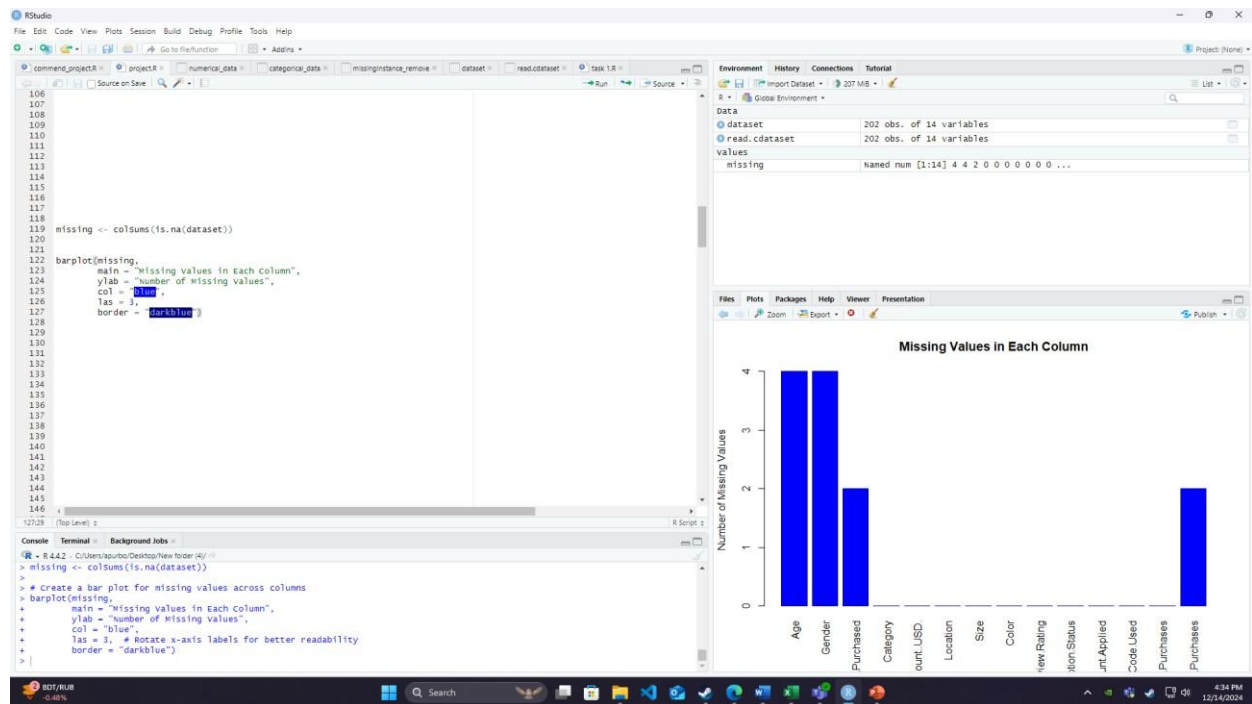
Replacing them with the most average value (mean) in each instance.





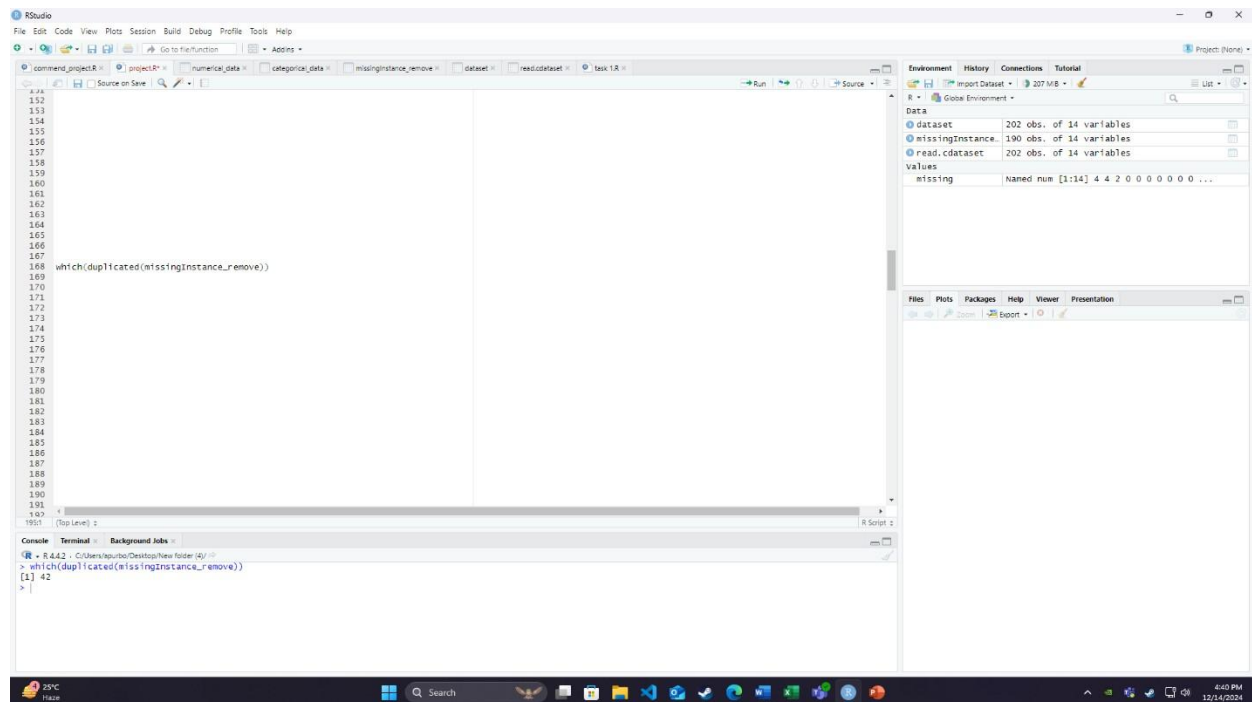
## Missing value in graph

We use barplot for the graph



## Find and remove duplicate values.

Find and identify the positions duplicate value.



## Remove duplicate and show the output

The screenshot shows the RStudio interface with a script editor on the left and the Environment pane on the right. The script editor contains the following code:

```
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189 duplicate_value <- distinct(missingInstance_remove)  
190  
191 which(duplicated(duplicate_value))  
192  
193 duplicate_value
```

The Environment pane on the right shows the following objects:

- dataset: 202 obs. of 14 variables
- duplicate\_value: 189 obs. of 14 variables
- duplicated\_rows: 1 obs. of 14 variables
- missingInstance: 190 obs. of 14 variables
- read.cdaset: 202 obs. of 14 variables

The Console shows the following output:

```
R - R4.2 - C:\Users\hurb\Desktop\New folder (4)\  
> duplicate_value <- distinct(missingInstance_remove)  
>  
> which(duplicated(duplicate_value))  
integer(0)  
>
```

The screenshot shows the RStudio interface with a data viewer on the left and the Environment pane on the right. The data viewer displays a table with 26 columns: Age, Gender, Item Purchased, Category, Purchase Amount (USD), Location, Size, Color, Review Rating, Subscription Status, Discount Applied, Promo Code Used, Previous Purchases, and Frequency. The table contains 26 rows of data.

The Environment pane on the right shows the following objects:

- dataset: 202 obs. of 14 variables
- duplicate\_value: 189 obs. of 14 variables
- duplicated\_rows: 1 obs. of 14 variables
- missingInstance: 190 obs. of 14 variables
- read.cdaset: 202 obs. of 14 variables

The Console shows the following output:

```
R - R4.2 - C:\Users\hurb\Desktop\New folder (4)\  
> duplicate_value <- distinct(missingInstance_remove)  
>  
> which(duplicated(duplicate_value))  
integer(0)  
> View(dataset)  
> View(duplicate_value)  
> View(duplicate_value)  
> View(duplicate_value)  
> View(duplicate_value)  
>
```

## Convert the imbalanced data set into the balanced data set

We have applied two method for Converting imbalanced to balanced dataset

1. Undersampling
2. Oversampling

### Undersampling

The screenshot shows the RStudio interface with the following code in the script editor:

```

class_counts <- duplicate_value %>%
  count(Frequency.of.Purchases)
minority_class_size <- min(class_counts$n)
balanced_data <- duplicate_value %>%
  group_by(Frequency.of.Purchases) %>%
  sample_n(size = min(n(), minority_class_size)) %>%
  ungroup()
table(balanced_data$Frequency.of.Purchases)

```

The console output shows the frequency of purchases for the balanced dataset:

```

> table(balanced_data$Frequency.of.Purchases)
Annually      21 81-weekly every 3 months 21 Fortnightly 21 Monthly 21 Quarterly 21 weekly 21

```

The Environment pane on the right shows the following objects:

- balanced\_data: 147 obs. of 14 variables
- class\_counts: 7 obs. of 2 variables
- dataset: 202 obs. of 14 variables
- duplicate\_value: 189 obs. of 14 variables
- duplicate\_rows: 1 obs. of 14 variables
- missingInstance: 190 obs. of 14 variables
- read.cdaset: 202 obs. of 14 variables

### Oversampling

The screenshot shows the RStudio interface with the following code in the script editor:

```

class_counts <- dataset %>%
  count(Frequency.of.Purchases)
majority_class_size <- max(class_counts$n)
balanced_data <- dataset %>%
  group_by(Frequency.of.Purchases) %>%
  sample_n(size = max(n(), majority_class_size), replace = TRUE) %>%
  ungroup()
table(balanced_data$Frequency.of.Purchases)

```

The console output shows the frequency of purchases for the balanced dataset:

```

> table(balanced_data$Frequency.of.Purchases)
Annually      36 81-weekly every 3 months 36 Fortnightly 36 Monthly 36 Quarterly 36 weekly 36

```

The Environment pane on the right shows the following objects:

- balanced\_data: 288 obs. of 14 variables
- class\_counts: 8 obs. of 2 variables
- dataset: 202 obs. of 14 variables
- missingInstance: 190 obs. of 14 variables
- read.cdaset: 202 obs. of 14 variables

### filtering methods

Filter age between 1 to 50

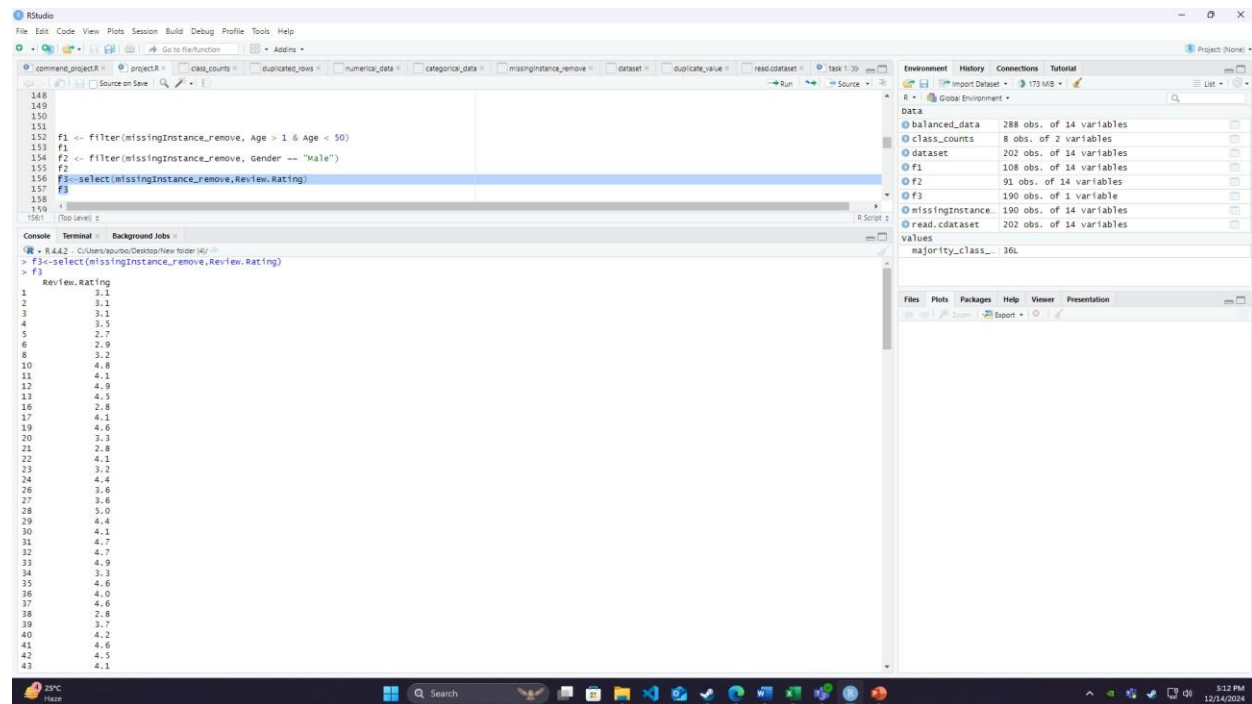
The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for data cleaning:
 

```
f1 <- filter(missingInstance_remove, Age > 1 & Age < 50)
f1
f2 <- filter(missingInstance_remove, gender == "Male")
f2
f3 <- select(missingInstance_remove, Review, Rating)
f3
f4 <- filter(missingInstance_remove, gender == "Male")
f4
```
- Environment Pane:** Shows the following objects:
  - `balanced_data`: 288 obs. of 14 variables
  - `class_counts`: 8 obs. of 2 variables
  - `dataset`: 202 obs. of 14 variables
  - `f1`: 108 obs. of 14 variables
  - `f2`: 91 obs. of 14 variables
  - `missingInstance`: 190 obs. of 14 variables
  - `read_dataset`: 202 obs. of 14 variables
  - `values`: majority\_class = 361
- Console:** Shows the execution of the R code:
 

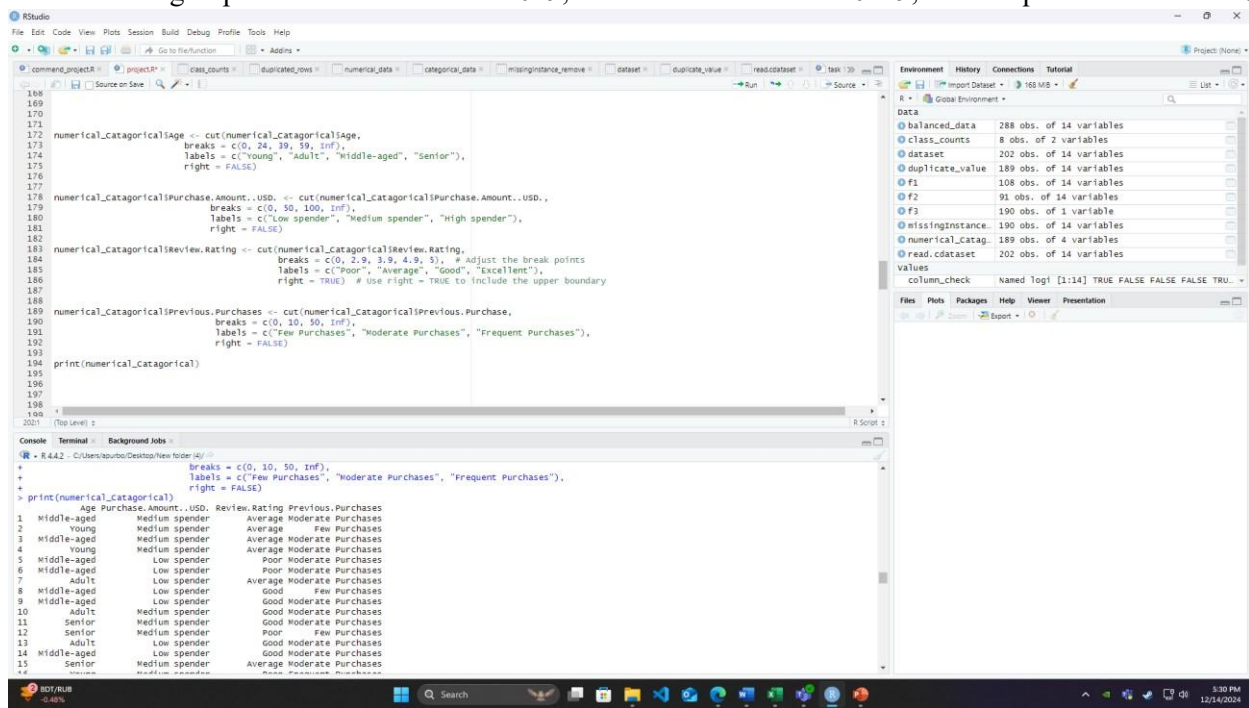
```
> f2 <- filter(missingInstance_remove, gender == "Male")
> f2
```
- Data Table:** A table with 14 columns: Age, Gender, Item, Purchased, Category, Purchase.Amount., USD, Location, Size, Color, Review, Rating, Subscription.status, Discount.Applied, Promo.Code, Used. It contains 202 rows of data.

we select only the review.rating column from the dataset



## Convert attributes from numeric to categorical

We categorizes numerical columns into groups. Age is categorized as Young =0–23, Adult =24–38, Middle-aged =39–58, and Senior =+59. Purchase.Amount.USD. is grouped into Low spender =0–49, Medium spender=50–99, and High spender=+100. Review.Rating is labeled as Poor= 0–2.9, Average=3–3.9, Good =4–4.9, and Excellent= 5. Previous.Purchases is grouped into Few Purchases=0–9, Moderate Purchases= 10–49, and Frequent Purchases =+50.





RStudio interface showing a data table with columns: Age, Purchase.Amount, USD, Review.Rating, Previous.Purchases. The table contains 19 rows of data. The Environment pane on the right shows variables: balanced\_data (288 obs. of 14 variables), class\_counts (8 obs. of 2 variables), dataset (202 obs. of 14 variables), duplicate\_value (189 obs. of 14 variables), f1 (108 obs. of 14 variables), f2 (91 obs. of 14 variables), f3 (190 obs. of 1 variable), missingInstance (190 obs. of 14 variables), numerical\_catag (189 obs. of 4 variables), read.cdaset (202 obs. of 14 variables). The Console shows the execution of R code to view numerical data.

## convert attributes from categorical to numeric

We convert categorical columns into numeric values. Gender, Item.Purchased, Category, Location, Size, Color, Subscription.Status, Discount.Applied, Promo.Code.Used, and Frequency.of.Purchases are all converted to numeric codes where each unique category is replaced with a unique numeric value.

RStudio interface showing R code to convert categorical variables to numeric. The code uses `as.numeric(factor(...))` for each categorical variable. The Environment pane shows the same variables as the first screenshot. The Console shows the execution of the code and the resulting data table.

```

187 categorical_numerical <- duplicate_value[,column_check]
188
189
190
191 categorical_numerical$gender <- as.numeric(factor(categorical_numerical$gender))
192
193
194
195 categorical_numerical$Item.Purchased <- as.numeric(factor(categorical_numerical$Item.Purchased))
196
197
198 categorical_numerical$Category <- as.numeric(factor(categorical_numerical$Category))
199
200
201 categorical_numerical$Location <- as.numeric(factor(categorical_numerical$Location))
202
203
204 categorical_numerical$Size <- as.numeric(factor(categorical_numerical$Size))
205
206
207 categorical_numerical$Color <- as.numeric(factor(categorical_numerical$Color))
208
209
210 categorical_numerical$Subscription.Status <- as.numeric(factor(categorical_numerical$Subscription.Status))
211
212
213 categorical_numerical$Discount.Applied <- as.numeric(factor(categorical_numerical$Discount.Applied))
214
215
216 categorical_numerical$Promo.Code.Used <- as.numeric(factor(categorical_numerical$Promo.Code.Used))
217
218
219 categorical_numerical$Frequency.of.Purchases <- as.numeric(factor(categorical_numerical$Frequency.of.Purchases))
220
221
222 categorical_numerical
223
224
225
226

```

The resulting data table in the console is:

	gender	Item.Purchased	Category	Location	size	color	Subscription.Status	discount.Applied	Promo.Code.Used	Frequency.of.Purchases
1	2	2	3	1	17	1	2	2	2	4
2	2	26	2	19	1	13	2	2	2	4
3	2	14	2	21	3	13	2	2	2	7
4	2	17	3	38	2	13	2	2	2	7
5	2	3	2	36	2	22	2	2	2	1
6	2	2	2	48	2	24	2	2	2	7
7	2	21	2	18	1	5	2	2	2	7
8	2	10	1	25	2	17	2	2	2	6
9	2	20	3	4	1	18	2	2	2	2

Gender	Item Purchased	Category	Location	Size	Color	Subscription Status	Discount Applied	Promo Code Used	Frequency of Purchases
1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9	9	9
10	10	10	10	10	10	10	10	10	10
11	11	11	11	11	11	11	11	11	11
12	12	12	12	12	12	12	12	12	12
13	13	13	13	13	13	13	13	13	13
14	14	14	14	14	14	14	14	14	14
15	15	15	15	15	15	15	15	15	15
16	16	16	16	16	16	16	16	16	16
17	17	17	17	17	17	17	17	17	17
18	18	18	18	18	18	18	18	18	18
19	19	19	19	19	19	19	19	19	19
20	20	20	20	20	20	20	20	20	20
21	21	21	21	21	21	21	21	21	21
22	22	22	22	22	22	22	22	22	22
23	23	23	23	23	23	23	23	23	23
24	24	24	24	24	24	24	24	24	24
25	25	25	25	25	25	25	25	25	25
26	26	26	26	26	26	26	26	26	26
27	27	27	27	27	27	27	27	27	27
28	28	28	28	28	28	28	28	28	28
29	29	29	29	29	29	29	29	29	29
30	30	30	30	30	30	30	30	30	30

## Outliers

We select specific attributes from the dataset, identified by `column_check`, and store them in `numerical_Categorical_normalization`. Then, we generate a summary of the dataset, which provides statistical details about the data.

```

class4.R
untitled2.R
missingInstance_remove
numerical_Categorical
project.R
numerical_data
categorical_data
balanced_data
class2.R
dataset
numerical_Categorical_normalization <- duplicate_value[,column_check]
summary(numerical_Categorical_normalization)
Q1 <- quantile(numerical_Categorical_normalization$Age, 0.25, na.rm = TRUE)
Q3 <- quantile(numerical_Categorical_normalization$Age, 0.75, na.rm = TRUE)
IQR <- Q3 - Q1
lower_bound <- Q1 - 1.5 * IQR
upper_bound <- Q3 + 1.5 * IQR
outliers <- numerical_Categorical_normalization$Age[numerical_Categorical_normalization$Age < lower_bound | numerical_Categorical_normalization$Age > upper_bound]

```

Console Output:

```

R - R4.4.1 ~ /
Citation() on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[workspace loaded from ~/.RData]

> numerical_Categorical_normalization <- duplicate_value[,column_check]
> summary(numerical_Categorical_normalization)
   Age      Purchase.Amount..USD.  Review.Rating Previous.Purchases
Min.   : 18.00   Min.   : 20.00   Min. :2.500   Min.   : 1.00
1st Qu.: 31.00   1st Qu.: 40.00   1st Qu.:3.100   1st Qu.:14.00
Median : 45.00   Median : 62.00   Median :3.800   Median :27.00
Mean   : 45.44   Mean   : 61.18   Mean  :3.772   Mean  :26.73
3rd Qu.: 56.00   3rd Qu.: 82.00   3rd Qu.:4.400   3rd Qu.:40.00
Max.   :256.00   Max.   :100.00   Max.   :5.000   Max.   :50.00

```

Environment:

- balanced\_data: 288 obs. of 14 variables
- categorical\_n: 189 obs. of 10 variables
- categorical\_d: 202 obs. of 10 variables
- class\_counts: 8 obs. of 2 variables
- dataset: 202 obs. of 14 variables
- duplicate\_val: 189 obs. of 14 variables
- filtered\_miss: 189 obs. of 0 variables
- missing\_data: 189 obs. of 4 variables
- missingData\_r: 190 obs. of 14 variables
- missingInstan: 190 obs. of 14 variables
- numerical\_Cat: 189 obs. of 4 variables
- numerical\_Cat: 189 obs. of 4 variables
- numerical\_data: 202 obs. of 4 variables
- outliers\_remo: 187 obs. of 4 variables
- read.cdatsset: 202 obs. of 14 variables
- stop\_words: 1149 obs. of 2 variables

We calculate the first (Q1) and third quartiles (Q3) of the age column in categorical normalization. Then we compute the Interquartile Range (IQR) by subtracting Q1 from Q3. The lower and upper bounds are determined using 1.5 times the IQR. At last we identify the outliers in the Age attribute that fall outside these bounds and store them in the outliers variable.



```

241
242
243
244 numerical_catagorical_normalization <- duplicate_value[,column_check]
245
246 summary(numerical_catagorical_normalization)
247
248
249 Q1 <- quantile(numerical_catagorical_normalization$Age, 0.25, na.rm = TRUE)
250 Q3 <- quantile(numerical_catagorical_normalization$Age, 0.75, na.rm = TRUE)
251 IQR <- Q3 - Q1
252
253
254 lower_bound <- Q1 - 1.5 * IQR
255 upper_bound <- Q3 + 1.5 * IQR
256
257
258 outliers <- numerical_catagorical_normalization$Age[numerical_catagorical_normalization$Age < lower_bound | numerical_catagorical_normalization$Age > upper_bound]
259 outliers
260
261
262 outliers_remove <- numerical_catagorical_normalization %>% filter(Age >= lower_bound & Age <= upper_bound)
263
264
265 summary(outliers_remove)
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
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

We filter the `Catagorical_normalization_normalization` dataset to remove the outliers from the Age column by keeping only the intence where Age is between the lower bound and upper bound. The resulting dataset outliers remove is then summarized to provide statistical details.

```

241
242
243
244 numerical_catagorical_normalization <- duplicate_value[,column_check]
245
246 summary(numerical_catagorical_normalization)
247
248
249 Q1 <- quantile(numerical_catagorical_normalization$Age, 0.25, na.rm = TRUE)
250 Q3 <- quantile(numerical_catagorical_normalization$Age, 0.75, na.rm = TRUE)
251 IQR <- Q3 - Q1
252
253
254 lower_bound <- Q1 - 1.5 * IQR
255 upper_bound <- Q3 + 1.5 * IQR
256
257
258 outliers <- numerical_catagorical_normalization$Age[numerical_catagorical_normalization$Age < lower_bound | numerical_catagorical_normalization$Age > upper_bound]
259 outliers
260
261
262 outliers_remove <- numerical_catagorical_normalization %>% filter(Age >= lower_bound & Age <= upper_bound)
263
264
265 summary(outliers_remove)
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
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

## normalization method

Normalizes the Review.Rating column to a 0–1 scale adds it as a new column (Rating\_Normalized) and then removes the original Review.Rating column from the dataset.

```

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

normalize <- function(column) {
  (column - min(column, na.rm = TRUE)) / (max(column, na.rm = TRUE) - min(column, na.rm = TRUE))
}

outliers_remove$Rating_Normalized <- normalize(outliers_remove$Review.Rating)
outliers_remove <- subset(outliers_remove, select = -Review.Rating)
outliers_remove

```

```

R Console:
R > 442 - C:\Users\jacob\Desktop\New folder\442
> outliers_remove$Rating_Normalized <- normalize(outliers_remove$Review.Rating)
>
> outliers_remove <- subset(outliers_remove, select = -Review.Rating)
> outliers_remove
  Age Purchase.Amount...USD Previous.Purchases Rating_Normalized
1  55             64         2             0.24
2  39             64         2             0.24
3  50             73         3             0.24
4  21             90         4             0.40
5  45             49         31            0.08
6  48             20         14            0.16
7  27             34         19            0.28
8  57             31         4             0.92
9  53             24         26            0.64
10 30             68         10            0.96
11 61             72         37            0.80
12 64             81         8             0.12
13 25             36         44            0.64

```

The Environment pane shows the following objects:

- numerical\_catag: 189 obs. of 4 variables
- outliers\_remove: 187 obs. of 4 variables
- read.cdaset: 202 obs. of 14 variables
- column\_check: Named logi [1:14] TRUE FALSE FALSE FALSE TRU...
- lower\_bound: Named num 25
- majority\_class: Named num -6.5
- outliers: int [1:2] 256 235
- Q1: Named num 31
- Q3: Named num 56
- upper\_bound: Named num 93.5
- normalize: function (column)