Assignment_3_codes_with_solution

16 July 2023
17:02

```
>
##*****************************************************************************
*****************##
>
> ## Task 1:Import the data to check its class and structure and display the
head and tail of the data
>
> #Import data
> input_df <- read_excel('Input_data.xlsx')
> print('-----------------------')
[1] "-----------------------"
> #Check the class of the data frame
> class(input_df)
[1] "tbl_df"     "tbl"        "data.frame"
> print('-----------------------')
[1] "-----------------------"
> #Check the structure of the data frame
> str(input_df)
tibble [1,000 × 8] (S3: tbl_df/tbl/data.frame)
 $ Employee_id: chr [1:1000] "S100" "S101" "S102" "S103" ...
 $ Pre        : num [1:1000] 4.26 3.96 3.89 4.29 3.58 ...
 $ Post       : num [1:1000] 4.64 5.2 5.66 5.85 4.49 ...
 $ Cold-Drink : chr [1:1000] "Coca-Cola" "Diet Coke" "Pepsi" "Diet Coke" ...
 $ Status     : chr [1:1000] "Member" "Member" "Member" "Observer" ...
 $ Rating     : chr [1:1000] "BB-" "AAA" "AAA" "BBB-" ...
 $ Outlook    : chr [1:1000] "Stable" "Stable" "Stable" "Positive" ...
 $ Salary     : num [1:1000] 1870 1866 1820 1728 1764 ...
> print('-----------------------')
[1] "-----------------------"
> #Print the head of dataframe
> head(input_df)
# A tibble: 6 × 8
  Employee_id   Pre  Post `Cold-Drink` Status   Rating Outlook  Salary
  <chr>       <dbl> <dbl> <chr>        <chr>    <chr>  <chr>     <dbl>
1 S100         4.26  4.64 Coca-Cola    Member   BB-    Stable     1870
2 S101         3.96  5.20 Diet Coke    Member   AAA    Stable     1866
3 S102         3.89  5.66 Pepsi        Member   AAA    Stable     1820
4 S103         4.29  5.85 Diet Coke    Observer BBB-   Positive   1728
5 S104         3.58  4.49 Coca-Cola    Member   BBB    Stable     1764
6 S105         3.76  4.42 Coca-Cola    Member   AA+    Negative   1744
> print('-----------------------')
[1] "-----------------------"
>
> #Print the tail of dataframe
> tail(input_df)
# A tibble: 6 × 8
  Employee_id   Pre  Post `Cold-Drink` Status   Rating Outlook  Salary
  <chr>       <dbl> <dbl> <chr>        <chr>    <chr>  <chr>     <dbl>
1 S1094        3.76  4.80 Pepsi        Observer B      Stable     1764
2 S1095        3.01  4.81 Pepsi        Member   BBB    Positive   1744
3 S1096        4.53  4.15 Pepsi        Member   A-     Stable     1656
```

```
4 S1097        5.00  5.99 Pepsi        Member   BBB    Stable     1734
5 S1098        3.53  4.31 Coca-Cola    Member   BBB    Stable     1788
6 S1099        4.32  5.54 Coca-Cola    Member   BB+    Stable     1610
> print('------------------------')
[1] "------------------------"
>
>
##***************************************************************************
*****************##
>
> ## Task 2: Calculate variaous parameters (listed below)
>
> #a.Difference in the means of the pre and post variables
> diff_mean = mean(input_df$Pre, na.rm = T) - mean(input_df$Post,na.rm = T)
> print(diff_mean)
[1] -0.9818101
> print('------------------------')
[1] "------------------------"
>
>
> #b.Values that divide the pre and post variable data into equal halves :
MEDIAN
> pre_median = median(input_df$Pre, na.rm = T)
> print(pre_median)
[1] 3.993653
> print('------------------------')
[1] "------------------------"
>
> post_median = median(input_df$Post, na.rm = T)
> print(post_median)
[1] 4.984026
> print('------------------------')
[1] "------------------------"
>
> #c.Mode for the pre variable
> pre_mode <- mfv(input_df$Pre)
> print(pre_mode)
   [1] 3.000516 3.001019 3.002317 3.002666 3.007824 3.008317 3.010276 3.014427
3.015152 3.022401 3.026144
  [12] 3.026553 3.027017 3.027560 3.027818 3.028733 3.031529 3.032695 3.033906
3.033993 3.034776 3.037438
  [23] 3.040522 3.040942 3.045648 3.047425 3.047839 3.051942 3.057589 3.059976
3.061811 3.062347 3.062719
  [34] 3.062754 3.067403 3.068372 3.071743 3.073048 3.076901 3.079030 3.091469
3.092340 3.092518 3.093089
  [45] 3.093699 3.094609 3.095153 3.095752 3.099267 3.099291 3.100438 3.101564
3.110600 3.111034 3.111712
  [56] 3.113950 3.115816 3.123805 3.126585 3.128255 3.130010 3.137011 3.137487
3.138503 3.140901 3.140998
  [67] 3.142797 3.146222 3.148313 3.148934 3.152483 3.152837 3.153226 3.158307
3.175198 3.176015 3.178292
  [78] 3.178676 3.178776 3.178844 3.184490 3.184861 3.188232 3.188798 3.190549
3.193347 3.193626 3.199078
  [89] 3.199571 3.200564 3.203557 3.204856 3.207414 3.207909 3.208076 3.208279
3.208595 3.210659 3.213310
 [100] 3.217215 3.221172 3.223232 3.227578 3.228907 3.229127 3.230350 3.230729
```

```
3.231172 3.231236 3.233091
 [111] 3.233855 3.235358 3.235722 3.237254 3.238710 3.247321 3.251143 3.251218
3.254102 3.254970 3.257643
 [122] 3.258515 3.258727 3.260640 3.270343 3.270623 3.271854 3.275067 3.276425
3.279362 3.279386 3.282758
 [133] 3.283849 3.284932 3.286404 3.292370 3.295642 3.295869 3.296953 3.297353
3.299311 3.302960 3.303465
 [144] 3.306583 3.310631 3.310989 3.311053 3.313105 3.314744 3.320973 3.321206
3.323525 3.325829 3.326490
 [155] 3.327869 3.328093 3.331028 3.334459 3.337845 3.338226 3.342918 3.344718
3.345248 3.345825 3.351224
 [166] 3.352452 3.352614 3.353160 3.360067 3.360190 3.361322 3.363471 3.368387
3.370526 3.373078 3.375688
 [177] 3.376475 3.378667 3.379846 3.380109 3.386187 3.388752 3.395252 3.397153
3.397290 3.400314 3.406976
 [188] 3.409287 3.410992 3.414440 3.415717 3.416269 3.417322 3.418406 3.419040
3.420121 3.430006 3.431410
 [199] 3.432900 3.436512 3.436584 3.437973 3.439102 3.439332 3.441715 3.444331
3.446867 3.450335 3.451404
 [210] 3.454736 3.456798 3.456866 3.457261 3.458586 3.460746 3.462342 3.464056
3.465862 3.466580 3.470517
 [221] 3.481011 3.482677 3.483743 3.484004 3.486857 3.488491 3.488746 3.489292
3.492865 3.493113 3.496022
 [232] 3.498578 3.500970 3.506899 3.506911 3.511083 3.514032 3.515518 3.518429
3.519311 3.519830 3.519912
 [243] 3.521644 3.523552 3.526324 3.527944 3.527964 3.530322 3.530932 3.533836
3.535127 3.538232 3.540352
 [254] 3.542181 3.542305 3.542634 3.544731 3.548291 3.548845 3.552983 3.554138
3.557382 3.559088 3.559488
 [265] 3.562974 3.563026 3.563804 3.563837 3.566009 3.573642 3.573761 3.574412
3.576459 3.576789 3.577588
 [276] 3.578021 3.580933 3.582527 3.583723 3.587403 3.587943 3.590670 3.591210
3.591399 3.592365 3.596601
 [287] 3.596604 3.596949 3.597723 3.597923 3.599272 3.603077 3.605832 3.607123
3.608894 3.609135 3.616723
 [298] 3.619801 3.620026 3.620335 3.621812 3.622009 3.624678 3.627352 3.627703
3.631602 3.633502 3.634833
 [309] 3.635029 3.636452 3.639358 3.639650 3.640407 3.641494 3.642973 3.643044
3.651269 3.652307 3.653028
 [320] 3.657147 3.659850 3.664189 3.665713 3.666010 3.669209 3.671369 3.673184
3.675000 3.675609 3.675813
 [331] 3.675978 3.677064 3.677861 3.678485 3.682272 3.682427 3.683713 3.684616
3.685086 3.687257 3.689259
 [342] 3.699287 3.699540 3.703076 3.704941 3.705198 3.713472 3.715888 3.717749
3.718255 3.720958 3.722697
 [353] 3.728106 3.730885 3.730897 3.732155 3.732297 3.732591 3.735338 3.735489
3.737500 3.740608 3.743946
 [364] 3.744808 3.744883 3.746829 3.747070 3.749494 3.751129 3.751170 3.753147
3.753436 3.755214 3.755983
 [375] 3.756223 3.758157 3.762264 3.765428 3.767455 3.773119 3.773718 3.773954
3.775509 3.778216 3.779461
 [386] 3.781661 3.781795 3.783504 3.787682 3.788312 3.793455 3.797220 3.797713
3.802175 3.802804 3.803744
 [397] 3.803907 3.806149 3.808687 3.809770 3.812320 3.812621 3.814644 3.814922
3.815639 3.816946 3.816975
 [408] 3.817339 3.819936 3.821386 3.821438 3.824401 3.825953 3.830255 3.830496
```

3.832766 3.834839 3.835071
 [419] 3.835704 3.835775 3.835902 3.837200 3.839154 3.841844 3.842478 3.845116
3.845766 3.846850 3.847279
 [430] 3.847538 3.848501 3.850822 3.860808 3.865708 3.866339 3.866949 3.868711
3.875630 3.879615 3.886153
 [441] 3.886197 3.887540 3.889079 3.891982 3.898064 3.900327 3.901365 3.902214
3.903667 3.910984 3.911284
 [452] 3.913975 3.915460 3.916484 3.918716 3.921333 3.921415 3.921859 3.923188
3.924014 3.927692 3.928293
 [463] 3.928863 3.929254 3.930008 3.933825 3.936587 3.938198 3.941346 3.943676
3.944099 3.944307 3.945925
 [474] 3.945934 3.946947 3.947232 3.947328 3.953572 3.956645 3.956750 3.958076
3.959236 3.961461 3.971984
 [485] 3.972727 3.975601 3.976293 3.976724 3.977990 3.979956 3.981814 3.982769
3.983074 3.985789 3.988020
 [496] 3.988169 3.988316 3.989913 3.990616 3.991068 3.996237 3.997734 4.000266
4.000594 4.003859 4.005047
 [507] 4.006119 4.006408 4.009320 4.010316 4.013011 4.020326 4.020484 4.021337
4.024206 4.025155 4.028707
 [518] 4.033537 4.033799 4.036167 4.038910 4.040450 4.040657 4.040700 4.044649
4.045162 4.045217 4.046144
 [529] 4.046347 4.047576 4.050508 4.051117 4.054111 4.057124 4.057514 4.057680
4.058848 4.061636 4.061960
 [540] 4.062028 4.062129 4.063946 4.067733 4.068576 4.069908 4.074194 4.075266
4.075771 4.077035 4.078480
 [551] 4.080740 4.086937 4.087249 4.087743 4.088638 4.088664 4.092944 4.097145
4.103368 4.103572 4.105976
 [562] 4.107293 4.108867 4.109368 4.111497 4.112706 4.115189 4.116429 4.120510
4.125802 4.129726 4.129764
 [573] 4.132103 4.135389 4.139683 4.141475 4.142245 4.144388 4.144701 4.145776
4.152238 4.153553 4.154160
 [584] 4.157020 4.158428 4.164290 4.164757 4.166979 4.167109 4.168870 4.169783
4.170193 4.171103 4.171680
 [595] 4.173704 4.176130 4.177429 4.182023 4.182147 4.185604 4.190217 4.191797
4.193531 4.196036 4.197651
 [606] 4.198525 4.202161 4.202760 4.203735 4.210056 4.211557 4.211628 4.212110
4.212152 4.212717 4.213213
 [617] 4.213705 4.216322 4.216943 4.218849 4.222363 4.223765 4.226893 4.227492
4.233615 4.233643 4.240194
 [628] 4.246752 4.248064 4.251078 4.251252 4.254717 4.255097 4.258112 4.258875
4.259190 4.260654 4.262640
 [639] 4.265604 4.267311 4.272569 4.276291 4.277206 4.277999 4.278862 4.279307
4.279424 4.279967 4.286536
 [650] 4.289869 4.290050 4.298111 4.298122 4.303573 4.304301 4.310573 4.315304
4.315515 4.316541 4.316586
 [661] 4.320891 4.321595 4.324525 4.329961 4.330579 4.334095 4.334378 4.335014
4.336693 4.338020 4.338268
 [672] 4.343573 4.343767 4.344003 4.345269 4.345671 4.347037 4.347465 4.349191
4.351503 4.352586 4.354788
 [683] 4.355007 4.356056 4.356947 4.360076 4.362970 4.364219 4.364387 4.364658
4.368566 4.369497 4.372885
 [694] 4.375150 4.381814 4.386423 4.386641 4.386754 4.389020 4.392143 4.394232
4.395240 4.399143 4.401146
 [705] 4.401342 4.408683 4.414223 4.415038 4.422540 4.423178 4.424812 4.425305
4.433147 4.433926 4.436358
 [716] 4.440952 4.441538 4.441722 4.448626 4.451805 4.455154 4.459901 4.462722

```
4.464068 4.468243 4.470161
 [727] 4.471265 4.471286 4.472763 4.473742 4.474165 4.475062 4.476553 4.479897
4.481357 4.482249 4.483549
 [738] 4.492690 4.494040 4.494162 4.494854 4.496478 4.497312 4.498534 4.499265
4.501645 4.501840 4.507069
 [749] 4.508499 4.508862 4.510433 4.511163 4.513769 4.517050 4.518629 4.520360
4.521058 4.522319 4.523706
 [760] 4.524721 4.528578 4.531798 4.534051 4.541094 4.541199 4.541459 4.543275
4.545926 4.546493 4.546590
 [771] 4.549206 4.549463 4.549874 4.550140 4.550325 4.550555 4.550654 4.550855
4.554462 4.556759 4.559367
 [782] 4.559578 4.562015 4.565536 4.566318 4.569638 4.572062 4.572331 4.573504
4.581554 4.582730 4.586025
 [793] 4.587448 4.596070 4.596502 4.598798 4.600013 4.600398 4.601927 4.602465
4.606063 4.608087 4.610225
 [804] 4.618148 4.622995 4.624881 4.625224 4.626724 4.628020 4.628341 4.628391
4.633722 4.633962 4.635949
 [815] 4.636384 4.639599 4.640875 4.642359 4.643068 4.643783 4.644058 4.644298
4.644519 4.648990 4.651164
 [826] 4.652888 4.654744 4.656597 4.657314 4.657605 4.661512 4.663451 4.671689
4.672372 4.673033 4.673831
 [837] 4.678946 4.681189 4.682967 4.685758 4.686665 4.688909 4.689286 4.692157
4.696353 4.696607 4.698213
 [848] 4.699539 4.699810 4.701431 4.704541 4.705085 4.705621 4.705890 4.706368
4.706920 4.709154 4.711690
 [859] 4.713367 4.715280 4.715460 4.720209 4.720841 4.722353 4.722370 4.726681
4.727988 4.732610 4.733383
 [870] 4.734220 4.734839 4.737399 4.737705 4.738506 4.742174 4.749183 4.751366
4.756152 4.756453 4.759851
 [881] 4.762114 4.767285 4.769646 4.769791 4.774971 4.775391 4.783761 4.783799
4.786940 4.786972 4.788103
 [892] 4.789540 4.790190 4.790666 4.790941 4.791025 4.791679 4.795092 4.796066
4.799581 4.799632 4.804928
 [903] 4.807898 4.809099 4.809914 4.809916 4.812066 4.812403 4.812721 4.812775
4.815741 4.820842 4.821192
 [914] 4.822469 4.823372 4.828455 4.830868 4.832173 4.834284 4.834923 4.834940
4.835354 4.837027 4.838635
 [925] 4.843400 4.845994 4.848013 4.854866 4.857606 4.858804 4.859774 4.860260
4.860832 4.863724 4.867416
 [936] 4.869447 4.873123 4.877073 4.877512 4.880728 4.881025 4.883736 4.896311
4.900583 4.901988 4.904643
 [947] 4.906406 4.906861 4.909047 4.910446 4.912172 4.912716 4.913012 4.914323
4.914369 4.918260 4.920246
 [958] 4.921101 4.924312 4.925182 4.925921 4.929569 4.931171 4.936913 4.939684
4.941775 4.944001 4.948553
 [969] 4.953126 4.955415 4.955757 4.956364 4.960254 4.964600 4.964881 4.965542
4.966527 4.968620 4.970488
 [980] 4.972764 4.973662 4.975106 4.975868 4.976463 4.982845 4.982908 4.983169
4.986713 4.987306 4.987634
 [991] 4.988739 4.989082 4.989848 4.990031 4.990623 4.991550 4.991551 4.995380
4.998340 4.999285
> print('-----------------------')
[1] "-----------------------"
>
> #d.First and third quantile for the pre and post variables
> pre_q1 <- quantile(input_df$Pre, 0.25)
```

```
> print(pre_q1)
     25%
3.534804
> print('------------------------')
[1] "------------------------"
> pre_q3 <- quantile(input_df$Pre, 0.75)
> print(pre_q3)
     75%
4.509255
> print('------------------------')
[1] "------------------------"
>
> post_q1 <- quantile(input_df$Post, 0.25)
> print(post_q1)
    25%
4.50269
> print('------------------------')
[1] "------------------------"
> post_q3 <- quantile(input_df$Post, 0.75)
> print(post_q3)
     75%
5.449862
> print('------------------------')
[1] "------------------------"
>
> #e.Range of the pre and post variables
> pre_range <- range(input_df$Pre)
> print(pre_range)
[1] 3.000516 4.999285
> print('------------------------')
[1] "------------------------"
> post_range <- range(input_df$Post)
> print(post_range)
[1] 4.001067 5.998279
> print('------------------------')
[1] "------------------------"
>
> #f.Variance and standard deviation for the pre and post variables
> pre_variance <- var(input_df$Pre)
> print(pre_variance)
[1] 0.3266061
> print('------------------------')
[1] "------------------------"
> post_variance <- var(input_df$Post)
> print(post_variance)
[1] 0.3250812
> print('------------------------')
[1] "------------------------"
>
> pre_sd <- sd(input_df$Pre)
> print(pre_sd)
[1] 0.5714946
> print('------------------------')
[1] "------------------------"
> post_sd <- sd(input_df$Post)
> print(post_sd)
```

```
[1] 0.5701589
> print('------------------------')
[1] "------------------------"
>
> #g.Coefficient of variation and mean absolute deviation for the pre and post
variables
> pre_cv <- sd(input_df$Pre) / mean(input_df$Pre) * 100
> print(pre_cv)
[1] 14.26208
> print('------------------------')
[1] "------------------------"
> post_cv <- sd(input_df$Post) / mean(input_df$Post) * 100
> print(post_cv)
[1] 11.42855
> print('------------------------')
[1] "------------------------"
>
> pre_mad <- mean(abs(input_df$Pre - mean(input_df$Pre)))
> print(pre_mad)
[1] 0.4916989
> print('------------------------')
[1] "------------------------"
>
> post_mad <- mean(abs(input_df$Post - mean(input_df$Post)))
> print(post_mad)
[1] 0.4898382
> print('------------------------')
[1] "------------------------"
>
> #h.Interquartile range of the pre and post variables
> pre_iqr <- pre_q3 - pre_q1
> print(pre_iqr)
      75%
0.9744509
> print('------------------------')
[1] "------------------------"
>
> post_iqr <- post_q3 - post_q1
> print(post_iqr)
      75%
0.9471727
> print('------------------------')
[1] "------------------------"
>
>
>
##****************************************************************************
*****************##
>
> ## Task 3:Measure the skewness for pre and post variables and apply the
Agostino test to check the skewness
>
> # Calculate the skewness for the Pre variable
> skewness_pre <- skewness(input_df$Pre)
> print(skewness_pre)
[1] 0.01174835
```

```
> print('------------------------')
[1] "------------------------"
>
> # Calculate the skewness for the Post variable
> skewness_post <- skewness(input_df$Post)
> print(skewness_post)
[1] 0.05197171
> print('------------------------')
[1] "------------------------"
>
> # Apply the Agostino test to the Pre variable
> agostino_test_pre <- agostino.test(input_df$Pre)
> print(agostino_test_pre)
D'Agostino skewness test
data:  input_df$Pre
skew = 0.011748, z = 0.152783, p-value = 0.8786
alternative hypothesis: data have a skewness
> print('------------------------')
[1] "------------------------"
>
> # Apply the Agostino test to the Post variable
> agostino_test_post <- agostino.test(input_df$Post)
> print(agostino_test_post)
D'Agostino skewness test
data:  input_df$Post
skew = 0.051972, z = 0.675450, p-value = 0.4994
alternative hypothesis: data have a skewness
> print('------------------------')
[1] "------------------------"
>
>
##****************************************************************************
*****************##
>
> ## Task 4:Identify the nature of distribution through kurtosis for both pre
and post variables and confirm the result through the Anscombe test
>
> # Calculate the kurtosis for the Pre variable
> kurtosis_pre <- kurtosis(input_df$Pre)
> print(kurtosis_pre)
[1] 1.840667
> print('------------------------')
[1] "------------------------"
>
> # Calculate the kurtosis for the Post variable
> kurtosis_post <- kurtosis(input_df$Post)
> print(kurtosis_post)
[1] 1.861554
> print('------------------------')
[1] "------------------------"
>
> # Apply the Anscombe test to the Pre variable
> anscombe_test_pre <- anscombe.test(input_df$Pre)
> print(anscombe_test_pre)
Anscombe-Glynn kurtosis test
```

```
data:  input_df$Pre
kurt = 1.8407, z = -23.3103, p-value < 2.2e-16
alternative hypothesis: kurtosis is not equal to 3
> print('-----------------------')
[1] "-----------------------"
>
> # Apply the Anscombe test to the Post variable
> anscombe_test_post <- anscombe.test(input_df$Post)
> print(anscombe_test_post)
Anscombe-Glynn kurtosis test
data:  input_df$Post
kurt = 1.8616, z = -21.7268, p-value < 2.2e-16
alternative hypothesis: kurtosis is not equal to 3
> print('-----------------------')
[1] "-----------------------"
>
>
>
##***************************************************************************
*****************##
>
> ## Task 5:Plot a graph to check the skewness and peakedness in the
distribution of pre and post variables
>
> # Plot the distribution of the pre variable
> plot(density(input_df$Pre), main="Distribution of Pre Variable",
+      xlab="Value", ylab="Density",
+      sub=paste("Skewness:", skewness_pre, "Kurtosis:", kurtosis_pre))
>
> print('----------------------')
[1] "----------------------"
>
> # Plot the distribution of the post variable
> plot(density(input_df$Post), main="Distribution of Post Variable",
+      xlab="Value", ylab="Density",
+      sub=paste("Skewness:", skewness_post, "Kurtosis:", kurtosis_post))
>
> print('----------------------')
[1] "----------------------"
>
>
##***************************************************************************
*****************##
>
> ## Task 6:Compute the frequency and relative frequency for each brand of
cold drink
>
> # Calculate the frequency and relative frequency for each brand
> df_grouped <- input_df %>%
+   group_by(input_df$`Cold-Drink`) %>%
+   summarise(
+     frequency = n(),
+     relative_frequency = n() / dim(input_df)[1]
+   )
>
> # Rename the column
```

```
> df_grouped <- df_grouped %>%
+   rename(cold_drink = `input_df$\`Cold-Drink\``)
>
> print(df_grouped)
# A tibble: 6 × 3
  cold_drink frequency relative_frequency
  <chr>          <int>              <dbl>
1 Coca-Cola        360              0.36
2 Cold-Drink        34              0.034
3 Diet Coke        178              0.178
4 Dr. Pepper        89              0.089
5 Pepsi            250              0.25
6 Sprite            89              0.089
>
> print('-----------------------')
[1] "-----------------------"
>
>
##*************************************************************************
****************##
>
> ## Task 7:Create a pie chart and bar chart to show the preferences of the
cold drinks available and provide the necessary labels
>
> # Create a pie chart
> pie_chart <- df_grouped %>%
+   plot_ly(labels = ~cold_drink,values = ~frequency, type = "pie") %>%
+   layout(title = "Preferences of Cold Drinks",
+          xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels =
FALSE),
+          yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels =
FALSE))
>
> pie_chart
>
> # Create a bar chart
> bar_chart = plot_ly(data = df_grouped,
+   x = ~cold_drink,
+   y = ~frequency,
+   type = "bar"
+ )
> # Print the bar chart
> bar_chart
>
>
>
##*************************************************************************
****************##
>
> ## Task 8:Plot a density graph on the cold-drink frequency and comment on
the skewness and kurtosis
>
> # Plot the distribution of the cold-drink frequency
>
> plot(density(df_grouped$frequency), main="Distribution of cold-drink
frequency",
```

```
+      xlab="Value", ylab="Density")
> print('-----------------------')
[1] "-----------------------"
>
> kurtosis_value = kurtosis(df_grouped$frequency)
>
> # Comment on Skewness and Kurtosis
> print("The plot looks bit drifted toward right, hence it indicates that the
distribution is slightly skewed to the right.")
[1] "The plot looks bit drifted toward right, hence it indicates that the
distribution is slightly skewed to the right."
> print(paste("The kurtosis is", kurtosis_value, ". This indicates that the
distribution is slightly leptokurtic."))
[1] "The kurtosis is 1.9965712222432 . This indicates that the distribution is
slightly leptokurtic."
>
>
>
##****************************************************************************
*****************##
>
> ## Task 9:Convert the ‘Status’, ‘Rating’, and ‘Outlook’ variables into
factor types and summarize them
>
> # Convert the variable to factor type
> input_df$Status <- as.factor(input_df$Status)
> input_df$Rating <- as.factor(input_df$Rating)
> input_df$Outlook <- as.factor(input_df$Outlook)
>
> # summarize above fvariables
> summary(input_df$Status)
  Member Observer
     901       99
> print('-----------------------')
[1] "-----------------------"
> summary(input_df$Rating)
   A   A+   A-   AA  AA+  AA-  AAA    B   B+   B-   BB  BB+  BB-  BBB BBB+
BBB-
  17  117   33   17   17   16  182   49   67   17   50   67   33  151   33
134
> print('-----------------------')
[1] "-----------------------"
> summary(input_df$Outlook)
Negative Positive   Stable
     184      167      649
> print('-----------------------')
[1] "-----------------------"
>
>
>
##****************************************************************************
*****************##
>
> ## Task 10: Calculate the difference in the average pre-training
satisfaction ratings of member and observer status and for the post-training
member and observer status
```

```
>
> # Group the data by status
> status_grouped_data <- input_df %>% group_by(Status)
>
> # Calculate the mean of Pre and Post satisfaction ratings
> mean_pre <- status_grouped_data %>% summarise(mean_pre = mean(Pre))
> mean_post <- status_grouped_data %>% summarise(mean_post = mean(Post))
>
> # Combine the mean of Pre and Post satisfaction ratings into a single data
frame
> status_results <- merge(mean_pre, mean_post, by = "Status")
>
> # Calculate the difference between observer and member values
> difference_mean_pre <- status_results$mean_pre[2] -
status_results$mean_pre[1]
> difference_mean_post <- status_results$mean_post[2] -
status_results$mean_post[1]
>
> # Print the difference
> print(difference_mean_pre)
[1] 0.03511248
> print('-----------------------')
[1] "-----------------------"
> print(difference_mean_post)
[1] 0.03398178
> print('-----------------------')
[1] "-----------------------"
>
>
##****************************************************************************
*****************##
>
> ## Task 11:Compute the average pre-satisfaction and post-satisfaction
ratings of employees with a ‘Stable’ Outlook
>
> # Filter the data for employees with a Stable Outlook
> data_stable <- input_df %>% filter(toupper(Outlook) == "STABLE")
>
> # Calculate the average pre-satisfaction rating
> mean_pre_stable <- mean(data_stable$Pre)
>
> # Calculate the average post-satisfaction rating
> mean_post_stable <- mean(data_stable$Post)
>
> # Print the results
> print(paste("The average pre-satisfaction rating for employees with a Stable
Outlook is", mean_pre_stable))
[1] "The average pre-satisfaction rating for employees with a Stable Outlook
is 4.00971778491598"
> print(paste("The average post-satisfaction rating for employees with a
Stable Outlook is", mean_post_stable))
[1] "The average post-satisfaction rating for employees with a Stable Outlook
is 4.99211404268603"
>
>
##****************************************************************************
```

```
****************##
>
> ## Task 12: Construct a confidence interval at a 2.5%, 5%, and 1% level of significance for the salary variable
>
> # Calculate the sample mean and standard deviation of the salary variable
> mean_salary <- mean(input_df$Salary)
> sd_salary <- sd(input_df$Salary)
>
> # Calculate the confidence intervals
> ci_2.5 <- mean_salary + c(-1.96, 1.96) * sd_salary
> ci_5 <- mean_salary + c(-1.645, 1.645) * sd_salary
> ci_1 <- mean_salary + c(-2.576, 2.576) * sd_salary
>
>
> # Print the confidence intervals
> print(paste("The mean salary is :",mean_salary))
[1] "The mean salary is : 1723.746"
> print('-----------------------')
[1] "-----------------------"
> print(paste("The 95% confidence interval for the salary variable is", ci_5[1],"to", ci_5[2] ))
[1] "The 95% confidence interval for the salary variable is 1578.86364332927 to 1868.62835667073"
> print('-----------------------')
[1] "-----------------------"
> print(paste("The 90% confidence interval for the salary variable is", ci_2.5[1],"to",ci_2.5[2]))
[1] "The 90% confidence interval for the salary variable is 1551.12021332849 to 1896.37178667151"
> print('-----------------------')
[1] "-----------------------"
> print(paste("The 99% confidence interval for the salary variable is", ci_1[1],"to",ci_1[2]))
[1] "The 99% confidence interval for the salary variable is 1496.8663946603 to 1950.6256053397"
> print('-----------------------')
[1] "-----------------------"
>
>
##******************************************************************************
****************##
>
> ## Task 13:Construct a 99%, 95%, and 90% confidence interval estimate for the pre and post variables
>
> # Calculate the mean and standard deviation of the pre variable
> mean_pre <- mean(input_df$Pre)
> std_dev_pre <- sd(input_df$Pre)
>
> # Calculate the confidence intervals
> ci_99_pre <- mean_pre + c(-2.576, 2.576) * std_dev_pre
> ci_95_pre <- mean_pre + c(-1.96, 1.96) * std_dev_pre
> ci_90_pre <- mean_pre + c(-1.645, 1.645) * std_dev_pre
>
> #Print the confidence Interval for Pre Variable
```

```
> print(paste("The mean salary of Pre variable is :",mean_pre))
[1] "The mean salary of Pre variable is : 4.00709069415415"
> print('-----------------------')
[1] "-----------------------"
> print(paste("The 99% confidence interval for the pre variable is",
ci_99_pre[1],"to", ci_99_pre[2]))
[1] "The 99% confidence interval for the pre variable is 2.5349205409879 to
5.47926084732039"
> print('-----------------------')
[1] "-----------------------"
> print(paste("The 95% confidence interval for the pre variable is",
ci_95_pre[1],"to", ci_95_pre[2]))
[1] "The 95% confidence interval for the pre variable is 2.88696122978853 to
5.12722015851977"
> print('-----------------------')
[1] "-----------------------"
> print(paste("The 90% confidence interval for the pre variable is",
ci_90_pre[1],"to", ci_90_pre[2]))
[1] "The 90% confidence interval for the pre variable is 3.06698203656157 to
4.94719935174672"
> print('-----------------------')
[1] "-----------------------"
>
> # Calculate the mean and standard deviation of the post variable
> mean_post <- mean(input_df$Post)
> std_dev_post <- sd(input_df$Post)
>
> # Calculate the confidence intervals
> ci_99_post <- mean_post + c(-2.576, 2.576) * std_dev_post
> ci_95_post <- mean_post + c(-1.96, 1.96) * std_dev_post
> ci_90_post <- mean_post + c(-1.645, 1.645) * std_dev_post
>
> #Print the confidence Interval for Post Variable
> print(paste("The mean salary of Post variable is :",mean_post))
[1] "The mean salary of Post variable is : 4.98890082785627"
> print('-----------------------')
[1] "-----------------------"
> print(paste("The 99% confidence interval for the post variable is",
ci_99_post[1],"to", ci_99_post[2]))
[1] "The 99% confidence interval for the post variable is 3.52017150573338 to
6.45763014997916"
> print('-----------------------')
[1] "-----------------------"
> print(paste("The 95% confidence interval for the post variable is",
ci_95_post[1],"to", ci_95_post[2]))
[1] "The 95% confidence interval for the post variable is 3.87138938711059 to
6.10641226860194"
> print('-----------------------')
[1] "-----------------------"
> print(paste("The 90% confidence interval for the post variable is",
ci_90_post[1],"to", ci_90_post[2]))
[1] "The 90% confidence interval for the post variable is 4.05098944008758 to
5.92681221562496"
>
>
>
```

```
##*****************************************************************************
****************##
>
> ## Task 14:Solve the below tasks:
>
> # Task 14a:Take a sample of 50 observations from the pre and post dataset
(without replacement)
>
> # Create a sample of 50 observations from the pre variable
> sample_pre <- sample(input_df$Pre, 50, replace = T)
>
> # Create a sample of 50 observations from the post variable
> sample_post <- sample(input_df$Post, 50, replace = T)
>
> # Combine the two samples into a single data frame
> sample <- cbind(sample_pre, sample_post)
> sample_df <- as.data.frame(sample)
> # Print the sample
> print(sample_df)
   sample_pre sample_post
1    3.639358    4.747680
2    3.028733    5.119293
3    4.759851    5.370200
4    3.027560    5.313465
5    4.222363    4.096971
6    3.221172    5.476602
7    4.087743    5.079035
8    3.577588    4.235959
9    3.237254    4.279799
10   3.320973    4.043972
11   4.673831    5.094153
12   3.788312    5.578626
13   4.265604    4.854841
14   3.552983    5.257430
15   4.158428    5.065089
16   3.898064    5.836932
17   3.148313    5.614285
18   4.227492    4.251116
19   4.944001    4.645114
20   3.981814    4.943504
21   3.325829    4.664976
22   3.704941    5.773890
23   4.462722    4.059480
24   3.488491    4.113547
25   4.791679    5.589856
26   3.270343    4.624577
27   3.715888    4.503595
28   3.489292    5.084983
29   4.601927    5.949319
30   3.027017    4.048878
31   4.910446    4.548212
32   3.153226    4.874213
33   3.093699    4.031512
34   3.458586    4.065813
35   4.643068    4.401598
36   3.217215    4.405841
```

```
37   3.641494    4.774473
38   3.094609    5.669282
39   3.254970    4.670603
40   4.190217    5.054730
41   4.483549    4.737258
42   4.795092    5.838173
43   3.178776    4.606550
44   4.251252    4.761736
45   4.848013    5.038395
46   4.737705    4.770746
47   4.324525    5.859131
48   4.696607    4.179288
49   4.989848    4.253544
50   3.208595    5.071552
>
> print('*********************************************')
[1] "*********************************************"
>
> # Task 14b:Construct a null hypothesis to examine whether the sample (50 observations) mean score of pre and post variables is significantly different from the population (1000 observations)
>
> # Calculate the mean of the pre variable in the population
> mean_pre_population <- mean(input_df$Pre)
>
> # Calculate the mean of the post variable in the population
> mean_post_population <- mean(input_df$Post)
>
> # Calculate the mean of the pre variable in the sample
> mean_pre_sample <- mean(sample_df$sample_pre)
>
> # Calculate the mean of the post variable in the sample
> mean_post_sample <- mean(sample_df$sample_post)
>
> # Construct the null hypothesis
> # H0: There is no significant difference between the mean score of pre and post variables in the population and the sample
>
> print("H0: There is no significant difference between the mean score of pre and post variables in the population and the sample")
[1] "H0: There is no significant difference between the mean score of pre and post variables in the population and the sample"
>
> print('*********************************************')
[1] "*********************************************"
>
> # Task 14c:Compute corresponding Z values for pre and post variables in the sample
>
> # Calculate the mean of the pre variable in the sample
> mean_pre_sample <- mean(sample_df$sample_pre)
>
> # Calculate the standard deviation of the pre variable in the sample
> std_dev_pre_sample <- sd(sample_df$sample_pre)
>
> # Calculate the z-score for the pre variable
```

```
> z_pre <- (sample_df$sample_pre - mean_pre_sample) / std_dev_pre_sample
>
> # Calculate the mean of the post variable in the sample
> mean_post_sample <- mean(sample_df$sample_post)
>
> # Calculate the standard deviation of the post variable in the sample
> std_dev_post_sample <- sd(sample_df$sample_post)
> # Calculate the z-score for the post variable
> z_post <- (sample_df$sample_post - mean_post_sample) / std_dev_post_sample
>
> # Print the z-scores
> print(z_pre)
 [1] -0.39680044 -1.34009155  1.33413167 -1.34190352  0.50382254 -1.04281301
0.29586271 -0.49222250
 [9] -1.01796871 -0.88864126  1.20124765 -0.16669741  0.57062035 -0.53023325
0.40505572  0.00284628
[17] -1.15536538  0.51174486  1.61860460  0.13222293 -0.88113911 -0.29548792
0.87512835 -0.62985932
[25]  1.38329824 -0.96685330 -0.27857842 -0.62862255  1.09017045 -1.34274221
1.56676956 -1.14777543
[33] -1.23973185 -0.67605642  1.15372535 -1.04892468 -0.39350087 -1.23832697 -
0.99060095  0.45416400
[41]  0.90730108  1.38857158 -1.10830524  0.54844990  1.47032389  1.29991995
0.66164109  1.23643192
[49]  1.68942937 -1.06224182
> print(z_post)
 [1] -0.194866931  0.458014488  0.898829111  0.799152980 -1.338089525
1.085765010  0.387285627
 [8] -1.093903627 -1.016881067 -1.431202575  0.413845502  1.265009353 -
0.006596923  0.700705848
[15]  0.362784780  1.718824112  1.327659313 -1.067273074 -0.375063883
0.149173507 -0.340168765
[22]  1.608067373 -1.403956088 -1.308967207  1.284739727 -0.411145569 -
0.623697398  0.397735033
[29]  1.916275353 -1.422582860 -0.545309590  0.027436818 -1.453093061 -
1.392830678 -0.802894984
[36] -0.795439438 -0.147794941  1.424282310 -0.330282525  0.344584665 -
0.213177399  1.721004469
[43] -0.442816616 -0.170171872  0.315886454 -0.154342586  1.757825318 -
1.193468060 -1.063008049
[50]  0.374138140
>
> print('*********************************************')
[1] "*********************************************"
>
>
##****************************************************************************
*****************##
>
> ## Task 15: Using the p-value method, determine whether the sample mean for
the pre and post variables differs significantly from the population mean at
the 10% significance level
>
> ###Starting with Z Test Hypothesis Testing####
>
> # Calculate the mean of the pre variable in the population
```

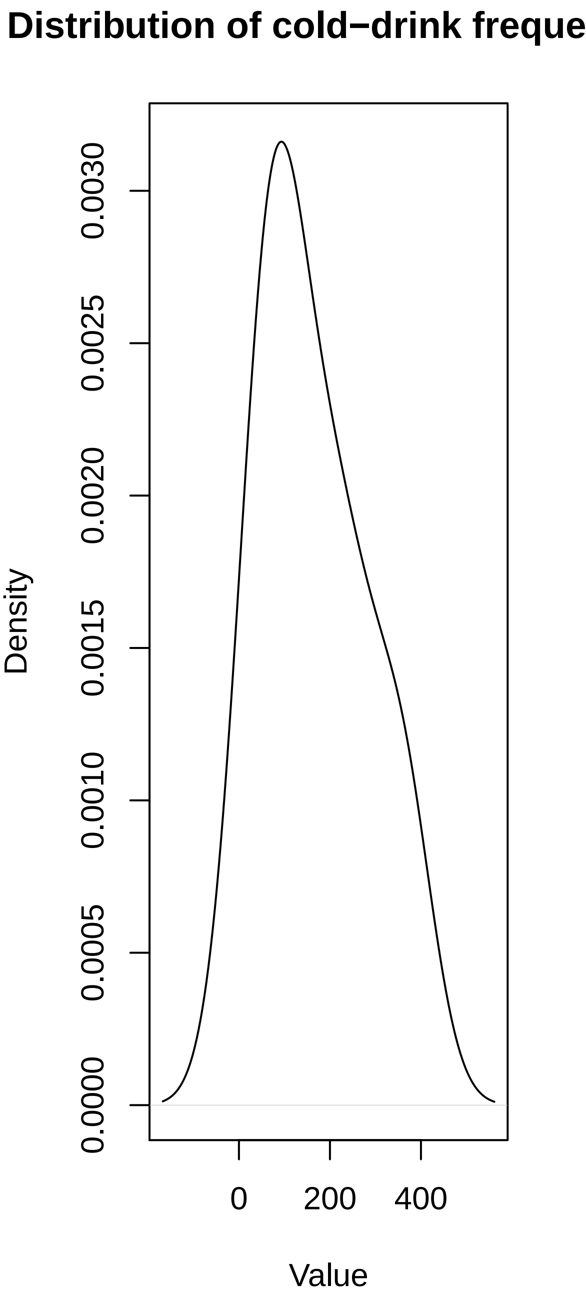
```
> mean_pre_population <- mean(input_df$Pre)
>
> # Calculate the sd of the pre variable in the population
> sd_pre_population <- sd(input_df$Pre)
>
> # Calculate the mean of the post variable in the population
> mean_post_population <- mean(input_df$Post)
>
> # Calculate the sd of the pre variable in the population
> sd_post_population <- sd(input_df$Post)
>
> z.test(sample_pre, sample_post, alternative = 'two.sided',
+        conf.level = 0.90, sigma.x = sd_pre_population, sigma.y =
sd_post_population)
Two-sample z-Test
data:  sample_pre and sample_post
z = -8.4297, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
90 percent confidence interval:
 -1.1501606 -0.7745898
sample estimates:
mean of x mean of y
 3.896221  4.858596
>
> p_value_pre_post <- z.test(sample_pre, sample_post, alternative =
'two.sided',
+        conf.level = 0.90, sigma.x = sd_pre_population, sigma.y =
sd_post_population)$p.value
>
> print(p_value_pre_post)
[1] 3.466853e-17
> print('-----------------------')
[1] "-----------------------"
> print(paste("Observation based on Results:", "As the p-value is < 0.05 it
clearly states that
+             there is a significant difference in mean of 2 categories. The
same can be validated
+             by looking at the mean of Pre and Post sample"))
[1] "Observation based on Results: As the p-value is < 0.05 it clearly states
that  \n            there is a significant difference in mean of 2 categories.
The same can be validated \n            by looking at the mean of Pre and Post
sample"
> print('-----------------------')
[1] "-----------------------"
>
>
>
##****************************************************************************
*****************##
>
> ## Task 16:       Calculate the critical Z value for the 10% level of
significance and the decision rule using the critical value approach
>
> ###Starting with Z Test Hypothesis Testing####
> z.test(sample_pre, sample_post, alternative = 'two.sided',
```

```
+        conf.level = 0.90, sigma.x = sd_pre_population, sigma.y =
sd_post_population)
Two-sample z-Test
data:  sample_pre and sample_post
z = -8.4297, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
90 percent confidence interval:
 -1.1501606 -0.7745898
sample estimates:
mean of x mean of y
 3.896221  4.858596
>
> critical_z_value = qnorm(p = 0.1/2, lower.tail = T) ## As Z-value is
negative
> print(critical_z_value)
[1] -1.644854
> print('-----------------------')
[1] "-----------------------"
>
>
>
##****************************************************************************
*****************##
>
> ## Task 17: Compute the T-statistics value for the pre and post variables
>
> ###Hypothesis Testing using T-Distribution###
>
> t.test(sample_pre, sample_post, alternative = 'two.sided', conf.level =
0.90)
Welch Two Sample t-test
data:  sample_pre and sample_post
t = -7.8946, df = 96.422, p-value = 4.636e-12
alternative hypothesis: true difference in means is not equal to 0
90 percent confidence interval:
 -1.1648332 -0.7599172
sample estimates:
mean of x mean of y
 3.896221  4.858596
>
> t.test(sample_pre, sample_post, alternative = 'greater', conf.level = 0.90)
Welch Two Sample t-test
data:  sample_pre and sample_post
t = -7.8946, df = 96.422, p-value = 1
alternative hypothesis: true difference in means is greater than 0
90 percent confidence interval:
 -1.119678       Inf
sample estimates:
mean of x mean of y
 3.896221  4.858596
>
> t.test(sample_pre, sample_post, alternative = 'less', conf.level = 0.90)
Welch Two Sample t-test
data:  sample_pre and sample_post
t = -7.8946, df = 96.422, p-value = 2.318e-12
alternative hypothesis: true difference in means is less than 0
```

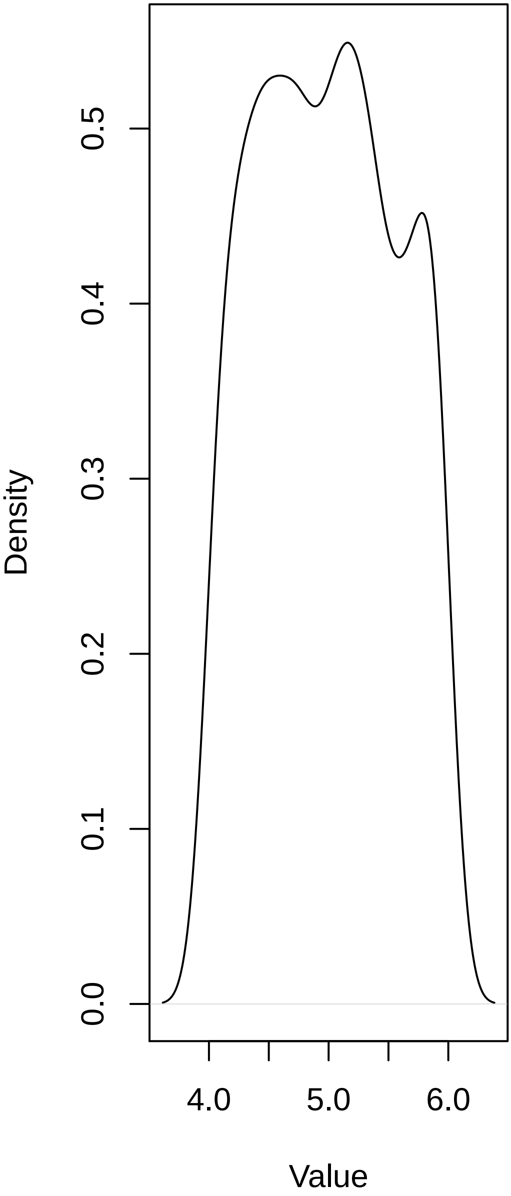
```
90 percent confidence interval:
       -Inf -0.8050722
sample estimates:
mean of x mean of y
 3.896221  4.858596
>
> print(paste(" Difference in Post and Pre sample mean: ",mean_post_sample -
mean_pre_sample))
[1] " Difference in Post and Pre sample mean:  0.962375211510807"
>
> print('-----------------------')
[1] "-----------------------"
>
> print("The above three test confirms that there is a significant difference
in Pre and Post Training
+       the p-value is quite significant in two-sample test, suggesting there
is a statistically significant
+       difference is score of Pre and post survey. And the One-tail test
confirms that the post sample mean is better than Pre-sample mean")
[1] "The above three test confirms that there is a significant difference in
Pre and Post Training \n      the p-value is quite significant in two-sample
test, suggesting there is a statistically significant \n      difference is
score of Pre and post survey. And the One-tail test confirms that the post
sample mean is better than Pre-sample mean"
>
>
>
>
##***************************************************************************
*****************##
>
> ## Task 18:        Calculate the p-value and the decision using the p-value
approach for pre and post variables at a 10% level of significance
>
> p_value_pre <- t.test(sample_pre, alternative = 'two.sided',
+                            conf.level = 0.90)$p.value
>
> print(p_value_pre)
[1] 2.317443e-40
> print('-----------------------')
[1] "-----------------------"
>
> p_value_post <- t.test(sample_post, alternative = 'two.sided',
+                       conf.level = 0.90)$p.value
>
> print(p_value_post)
[1] 1.171732e-47
>
> print('-----------------------')
[1] "-----------------------"
>
>
>
##***************************************************************************
*****************##
>
```

```
> ## Task 19:Calculate the critical T value for the level of significance of 10% and the decision rule using the critical value approach
>
> ###Hypothesis Testing using T-Distribution###
>
> t.test(sample_pre, sample_post, alternative = 'two.sided', conf.level = 0.90)
Welch Two Sample t-test
data:  sample_pre and sample_post
t = -7.8946, df = 96.422, p-value = 4.636e-12
alternative hypothesis: true difference in means is not equal to 0
90 percent confidence interval:
 -1.1648332 -0.7599172
sample estimates:
mean of x mean of y
 3.896221  4.858596
>
> DOF <- length(sample_pre) - 1
> print(DOF)
[1] 49
> print('-----------------------')
[1] "-----------------------"
>
> critical_t_value <- qt(p = 0.1/2, df = DOF, lower.tail = T)## As t-value is negative
> print(critical_t_value)
[1] -1.676551
> print('-----------------------')
[1] "-----------------------"
>
>
##*****************************************************************************************##
```

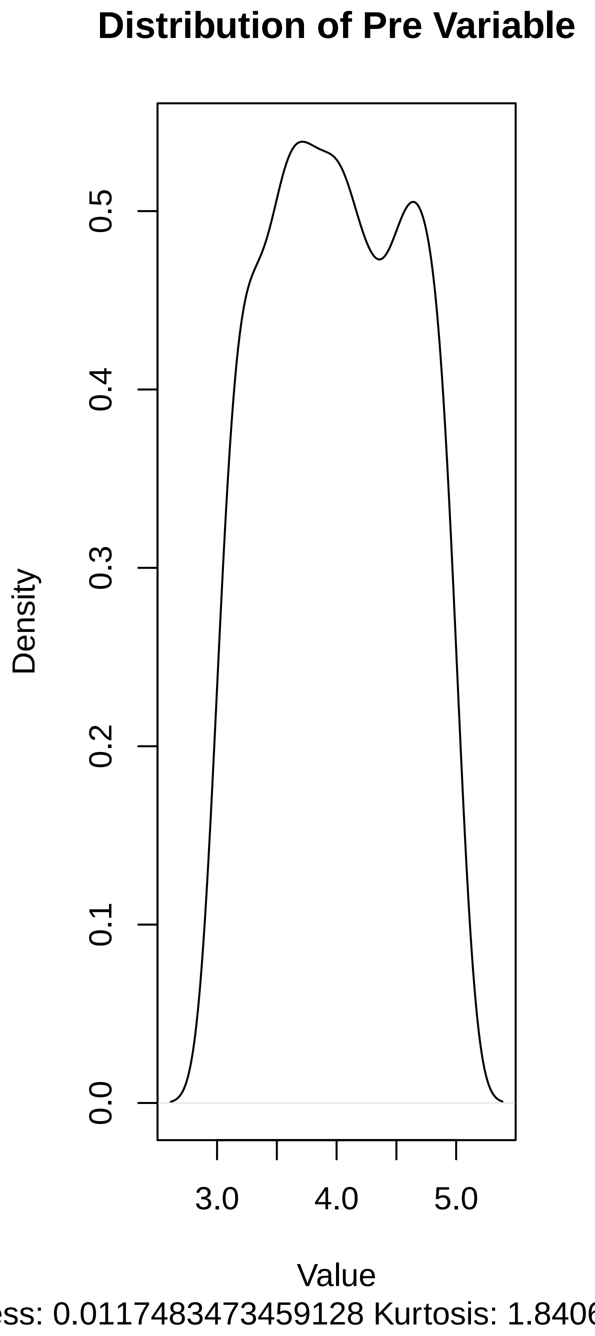
From <https://proxy-5.vocareum.com/>


Distribution of cold−drink freque
Density
0.0000
0.0005
0.0010
0.0015
0.0020
0.0025
0.0030
0
200
400
Value

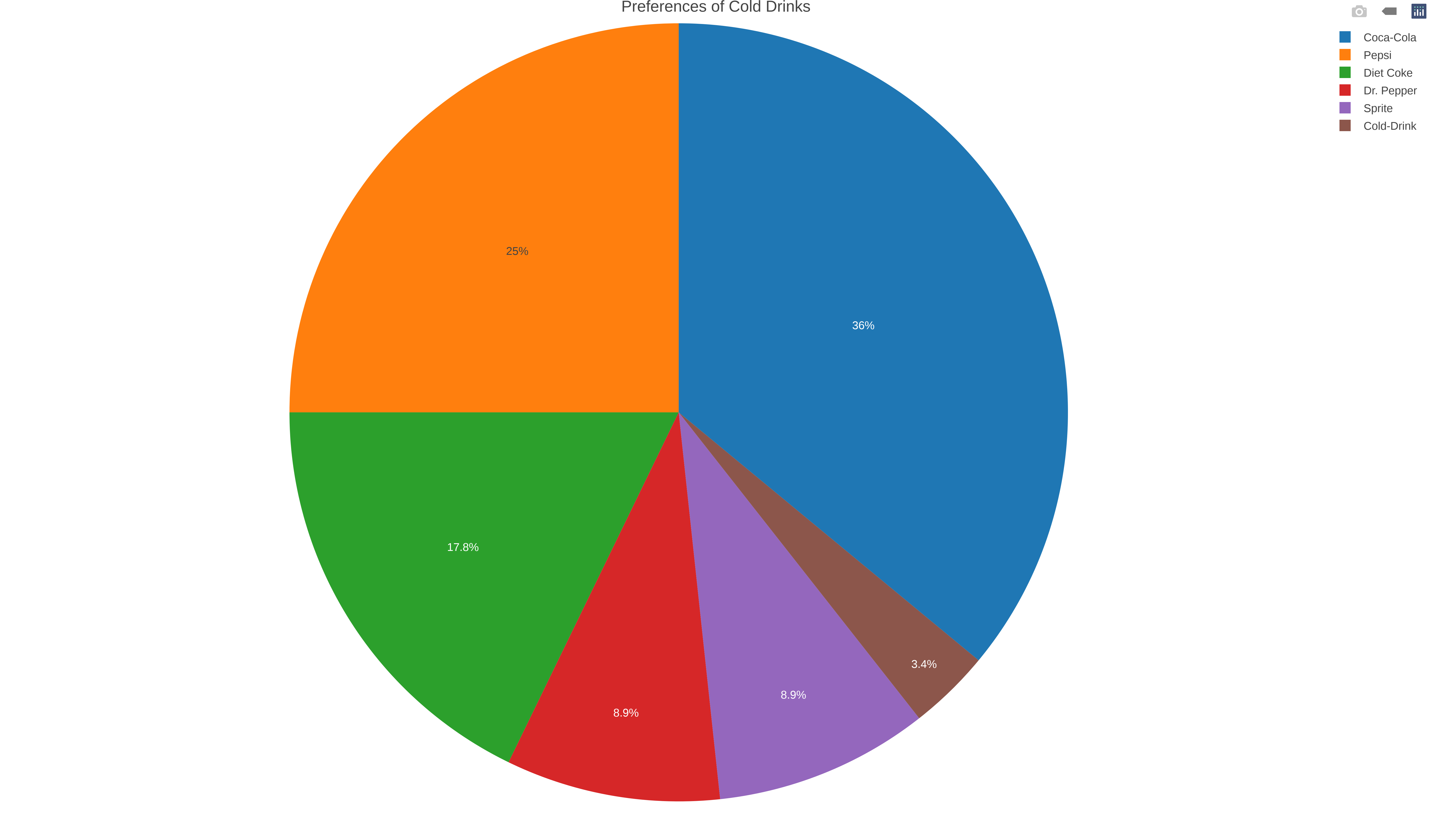
Distribution of Post Variable

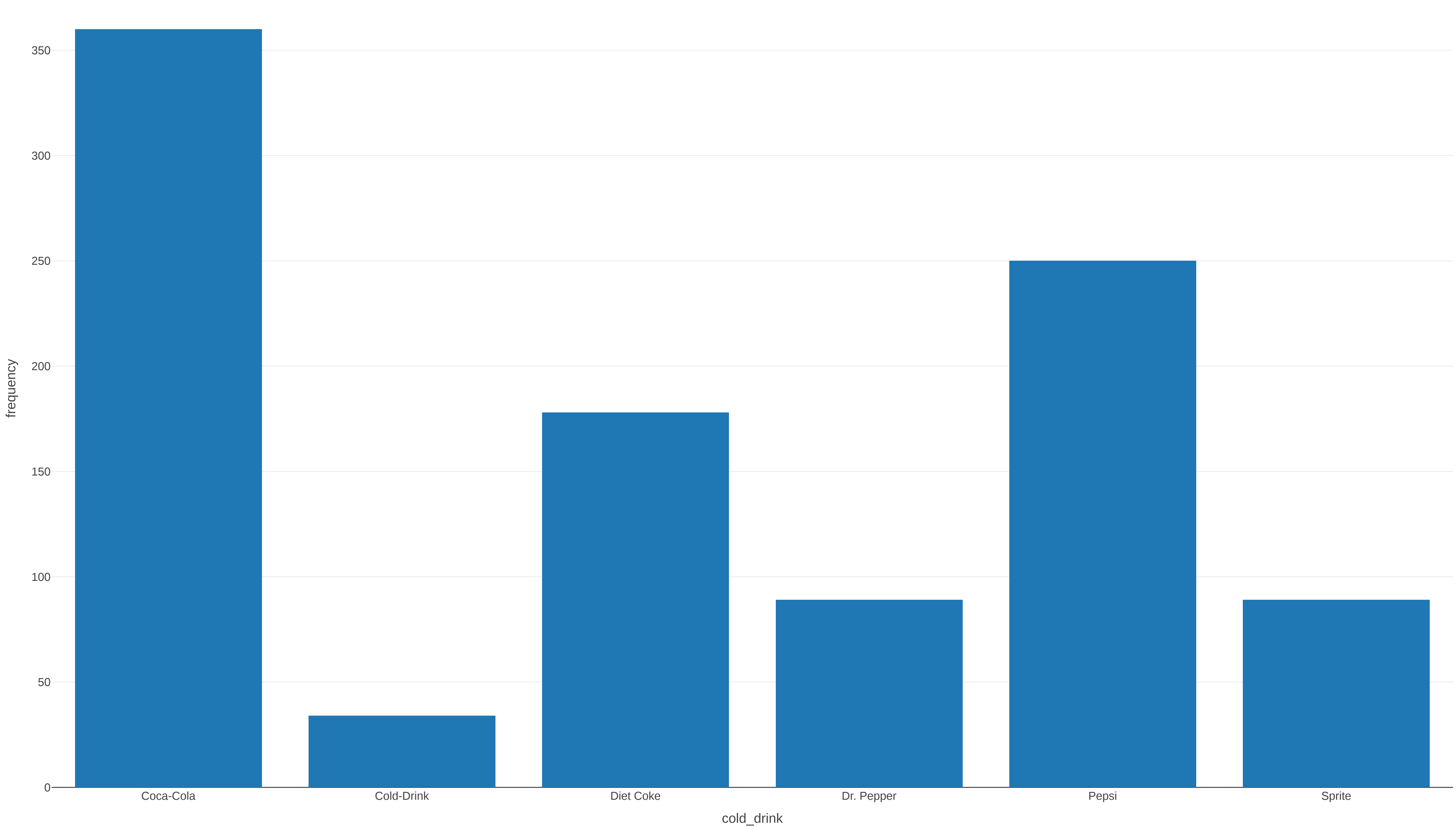


ess: 0.0519717059463907 Kurtosis: 1.8615



Distribution of Pre Variable
0.0
0.1
0.2
0.3
0.4
0.5
Density
3.0
4.0
5.0
Value
ess: 0.0117483473459128 Kurtosis: 1.8406


Preferences of Cold Drinks
Coca-Cola
Pepsi
Diet Coke
Dr. Pepper
Sprite
Cold-Drink
36%
25%
17.8%
8.9%
8.9%
3.4%


350
300
250
200
150
100
50
0
frequency
Coca-Cola
Cold-Drink
Diet Coke
Dr. Pepper
Pepsi
Sprite
cold_drink