Exercise for ggplot2

Jamia Begum

2022-10-14

#Exercise:1 Use what you've learned to improve the visualisation of the departure times of cancelled vs. non-cancelled flights (NOTE: missing values in the dep_time variable indicate that the flight was cancelled).

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

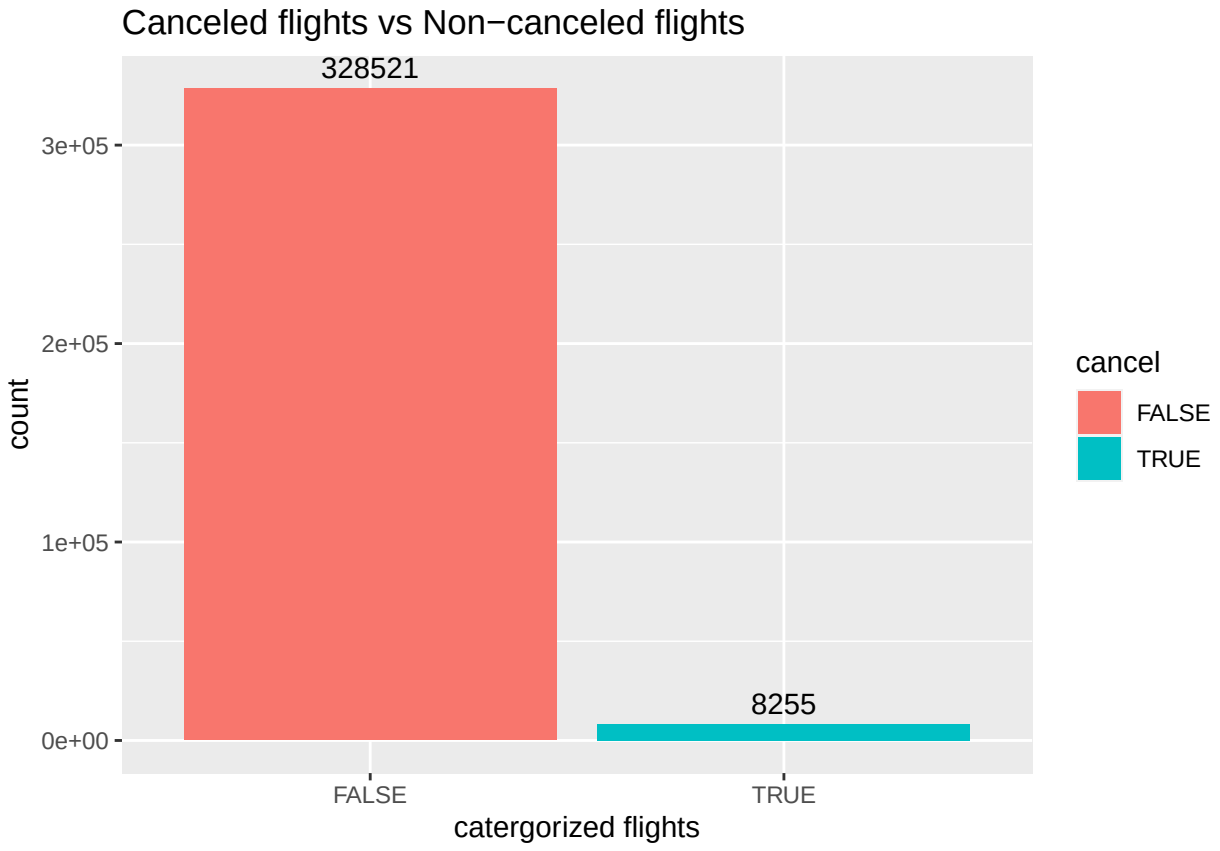
```
library(dplyr)
library(nycflights13) #to get flights tibble
flights
```

```
## # A tibble: 336,776 x 19
##     year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
##    <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
##  1  2013     1     1      517        515       2     830     819      11 UA
##  2  2013     1     1      533        529       4     850     830      20 UA
##  3  2013     1     1      542        540       2     923     850      33 AA
##  4  2013     1     1      544        545      -1    1004    1022     -18 B6
##  5  2013     1     1      554        600      -6     812     837     -25 DL
##  6  2013     1     1      554        558      -4     740     728      12 UA
##  7  2013     1     1      555        600      -5     913     854      19 B6
##  8  2013     1     1      557        600      -3     709     723     -14 EV
##  9  2013     1     1      557        600      -3     838     846      -8 B6
## 10  2013     1     1      558        600      -2     753     745       8 AA
## # ... with 336,766 more rows, 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

```
f<-mutate(flights,cancel=is.na(dep_time))
#true=canceled flights
#false=Non-canceled flights
pl<-select(f,dep_time,cancel)
pl
```

```
## # A tibble: 336,776 x 2
##    dep_time cancel
##       <int> <lgl>
##  1      517 FALSE
##  2      533 FALSE
##  3      542 FALSE
##  4      544 FALSE
##  5      554 FALSE
##  6      554 FALSE
##  7      555 FALSE
##  8      557 FALSE
##  9      557 FALSE
## 10      558 FALSE
## # ... with 336,766 more rows
```

```
library(ggplot2)

       ggplot(pl,aes(x=cancel,fill=cancel)) +ggtitle("Canceled flights vs Non-canceled flights") +
         xlab("catergorized flights") +
         geom_bar()+ geom_text(stat='count',aes(label=after_stat(count)), vjust=-0.5)
```



#Exercise:2 a)What variable in the diamonds dataset is most important for predicting the price of a diamond? Here,R package corrplot should be used to visualize the correlation matrix that shows the linear correlations between variables.

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

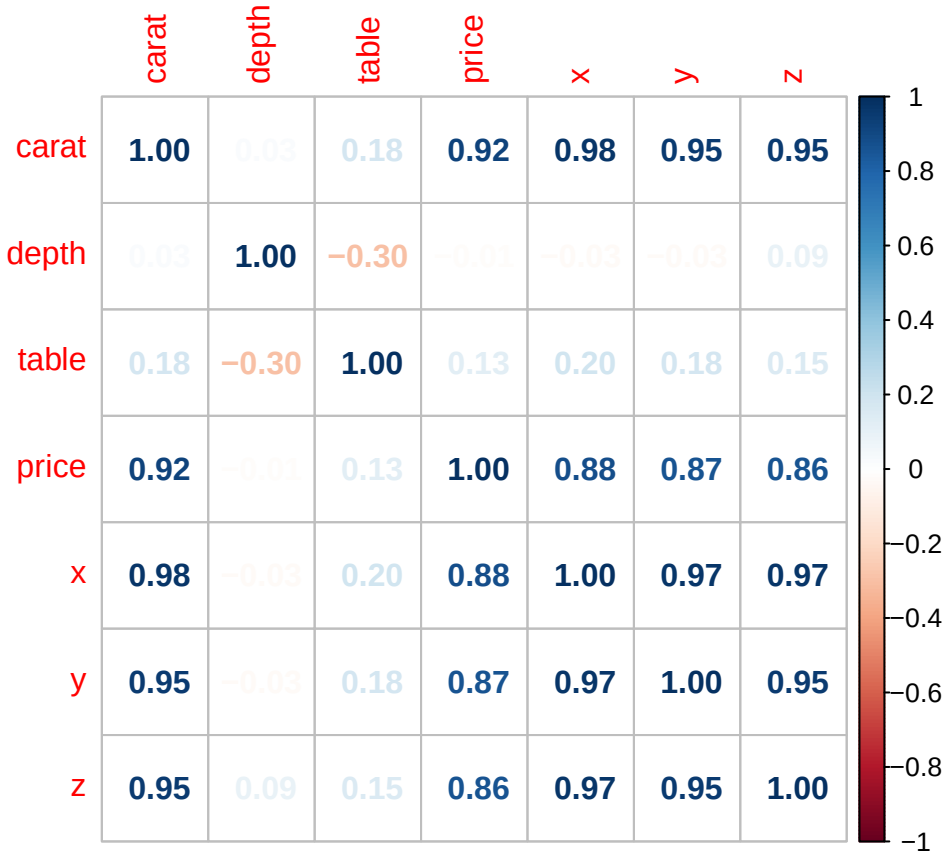
```
diamonds
```

```
## # A tibble: 53,940 x 10
##    carat cut       color clarity depth table price     x     y     z
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
##  2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
##  3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
##  4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
##  5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
##  6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
##  7  0.24 Very Good I     VVS1     62.3    57   336  3.95  3.98  2.47
##  8  0.26 Very Good H     SI1      61.9    55   337  4.07  4.11  2.53
##  9  0.22 Fair      E     VS2      65.1    61   337  3.87  3.78  2.49
## 10  0.23 Very Good H     VS1      59.4    61   338  4     4.05  2.39
## # ... with 53,930 more rows
```

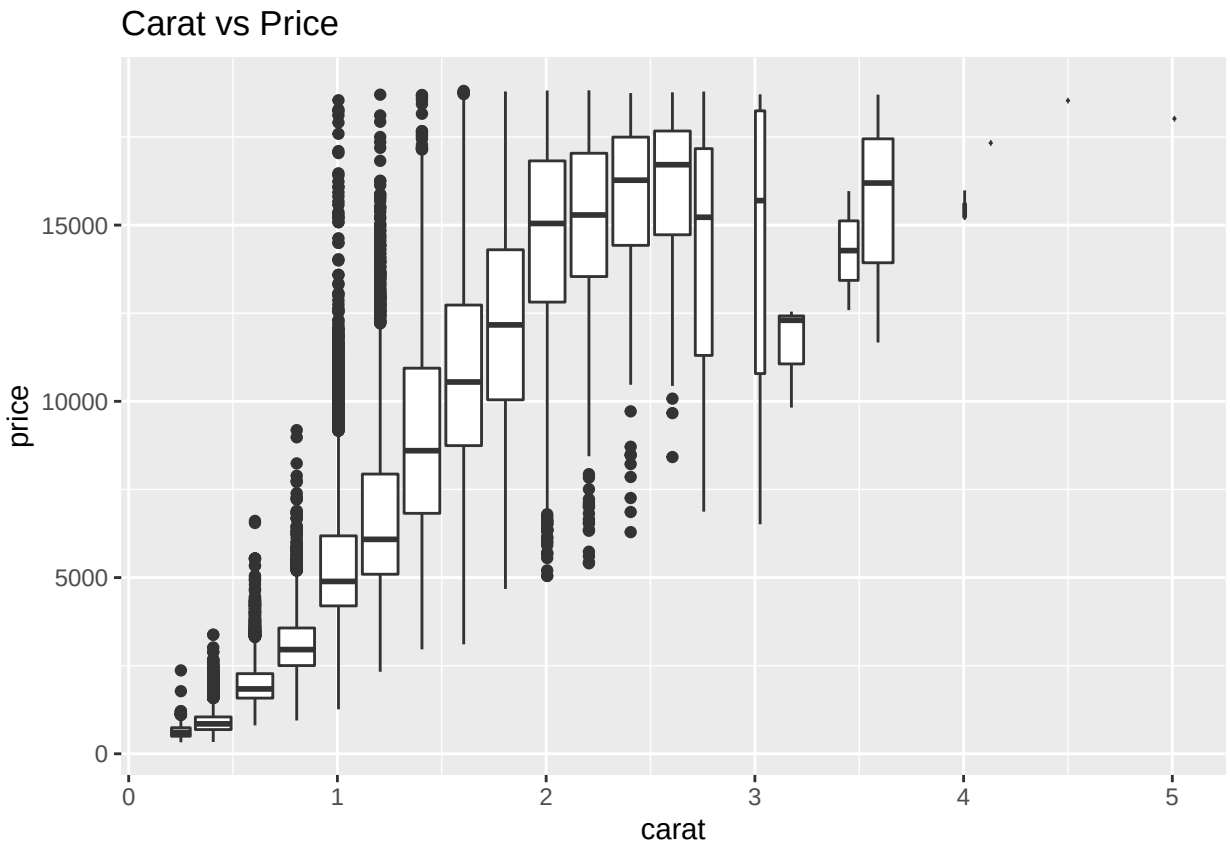
```
M <- cor(diamonds[,c(1,5,6,7,8,9,10)])
M
```

```
##            carat       depth      table      price           x           y
## carat 1.00000000  0.02822431  0.1816175  0.9215913  0.97509423  0.95172220
## depth 0.02822431  1.00000000 -0.2957785 -0.0106474 -0.02528925 -0.02934067
## table 0.18161755 -0.29577852  1.0000000  0.1271339  0.19534428  0.18376015
## price 0.92159130 -0.01064740  0.1271339  1.0000000  0.88443516  0.86542090
## x     0.97509423 -0.02528925  0.1953443  0.8844352  1.00000000  0.97470148
## y     0.95172220 -0.02934067  0.1837601  0.8654209  0.97470148  1.00000000
## z     0.95338738  0.09492388  0.1509287  0.8612494  0.97077180  0.95200572
##               z
## carat 0.95338738
## depth 0.09492388
## table 0.15092869
## price 0.86124944
## x     0.97077180
## y     0.95200572
## z     1.00000000
```

```
corrplot(M, method = 'number')
```




```
ggplot(data = diamonds, mapping = aes(x = carat, y = price)) +
  geom_boxplot(mapping = aes(group = cut_width(carat,0.2)))+
  ggtitle("Carat vs Price")
```

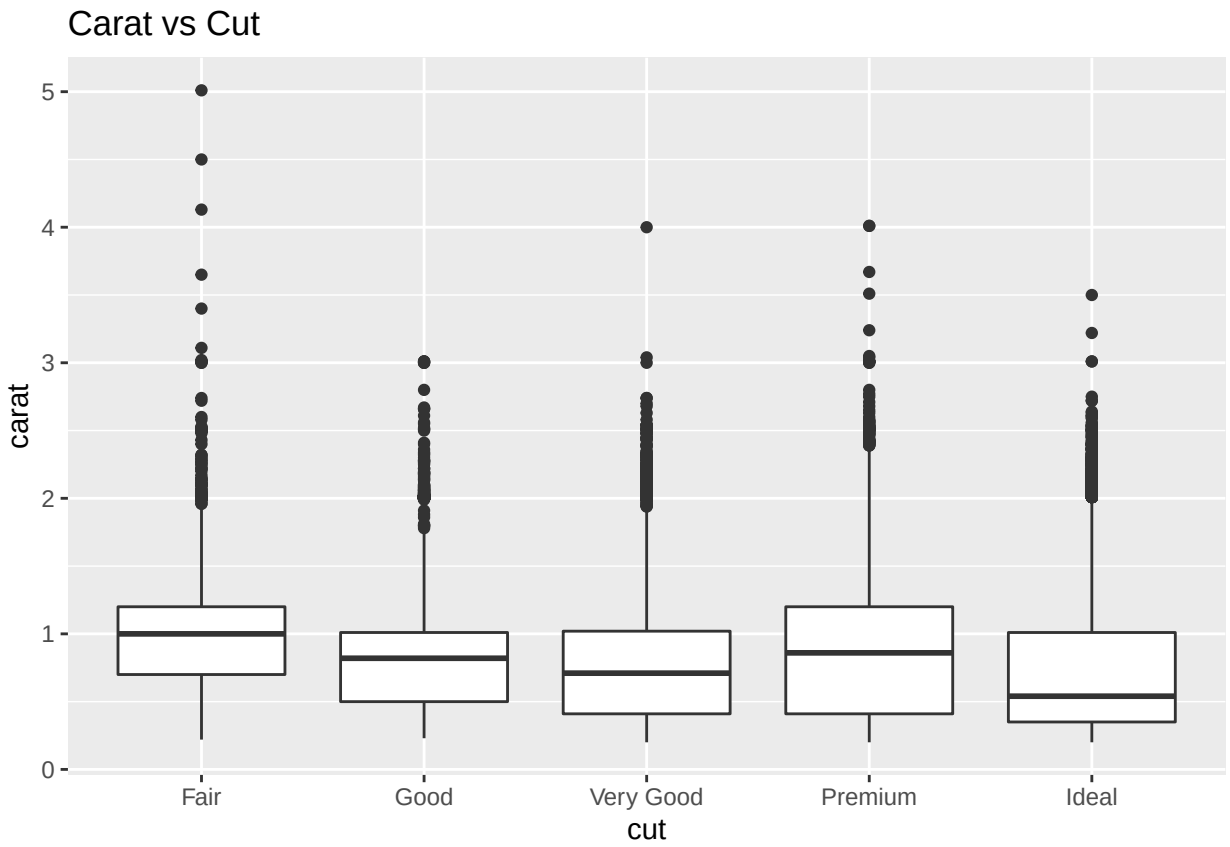


From the correlation Matrix, we see that “carat” has the highest correlation coefficient related to “price” .It means “carat” has a fairly strong positive relationship with “price”.Also, the boxplots between “carat” and “price” shows that in most of the cases the price is increasing as the “carat” sizes go up.So, “carat” is most important for predicting the price of a diamond.

b)How is that variable(“carat”) correlated with cut? How is that variable correlated with cut? Why does the combination of those two relationships lead to lower quality diamonds being more expensive?. NOTE: variable cut describes the quality (see ?diamonds).

Here, “carat” is a continuous variable but “cut” is a categorical variable. To visualize the correlation them ,it’s better to use boxplots

```
ggplot(diamonds, aes(x = cut, y = carat)) +
  geom_boxplot()+ggtitle("Carat vs Cut")
```

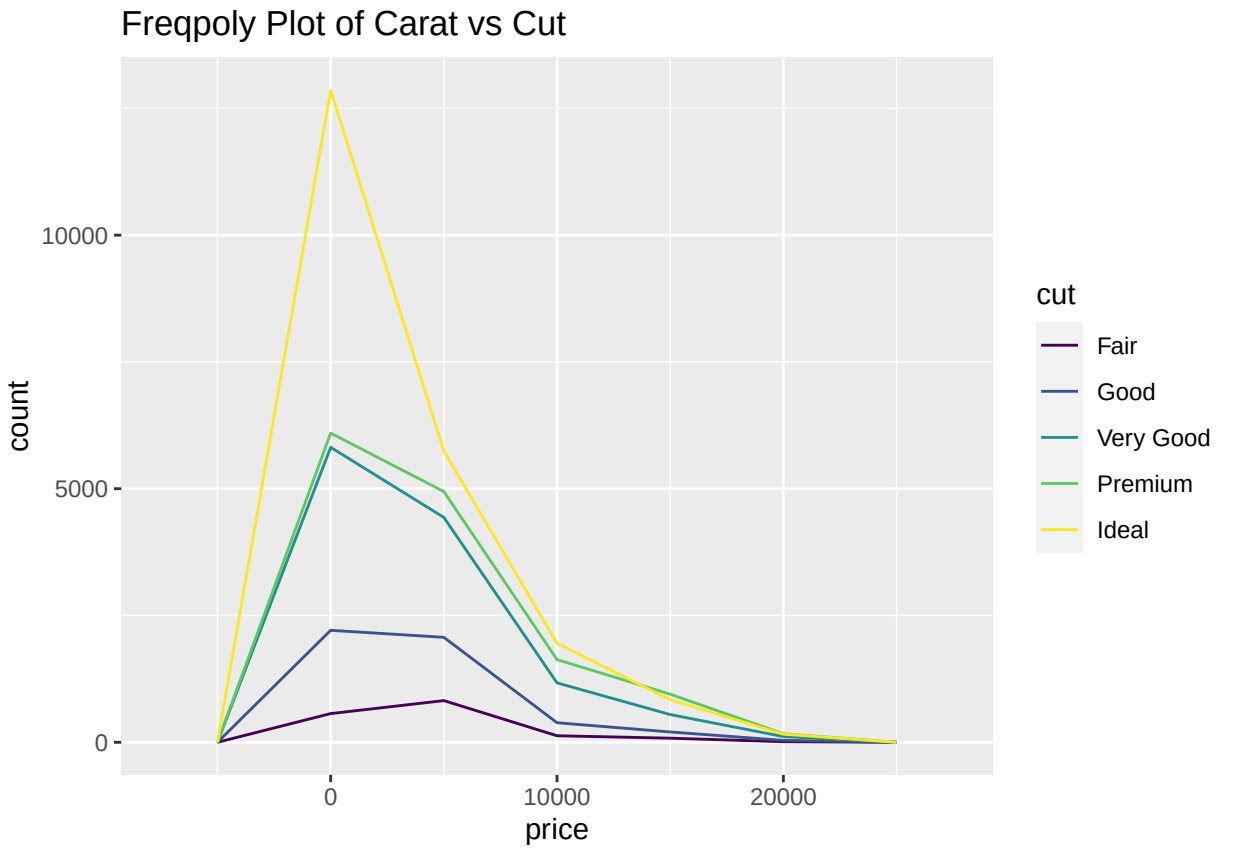


we see that the relation fluctuates so much between "cut" and "carat". Here, some low quality cuts (Fair) are having larger "carat" ,thus higher prices whereas good and ideal quality diamonds are having less "carat" and low prices.

#Exercise-4 Instead of summarising the conditional distribution with a boxplot, you could use a frequency polygon.

Here, "freqpoly" plot gives the conditional distribution by showing which cut has the highest density with the price.

```
ggplot(data = diamonds, mapping = aes(x = price)) +
  geom_freqpoly(mapping = aes(color = cut), binwidth = 5000)+
  ggtitle("Freqpoly Plot of Carat vs Cut")
```



What do you need to consider when using cut_width() vs cut_number()?

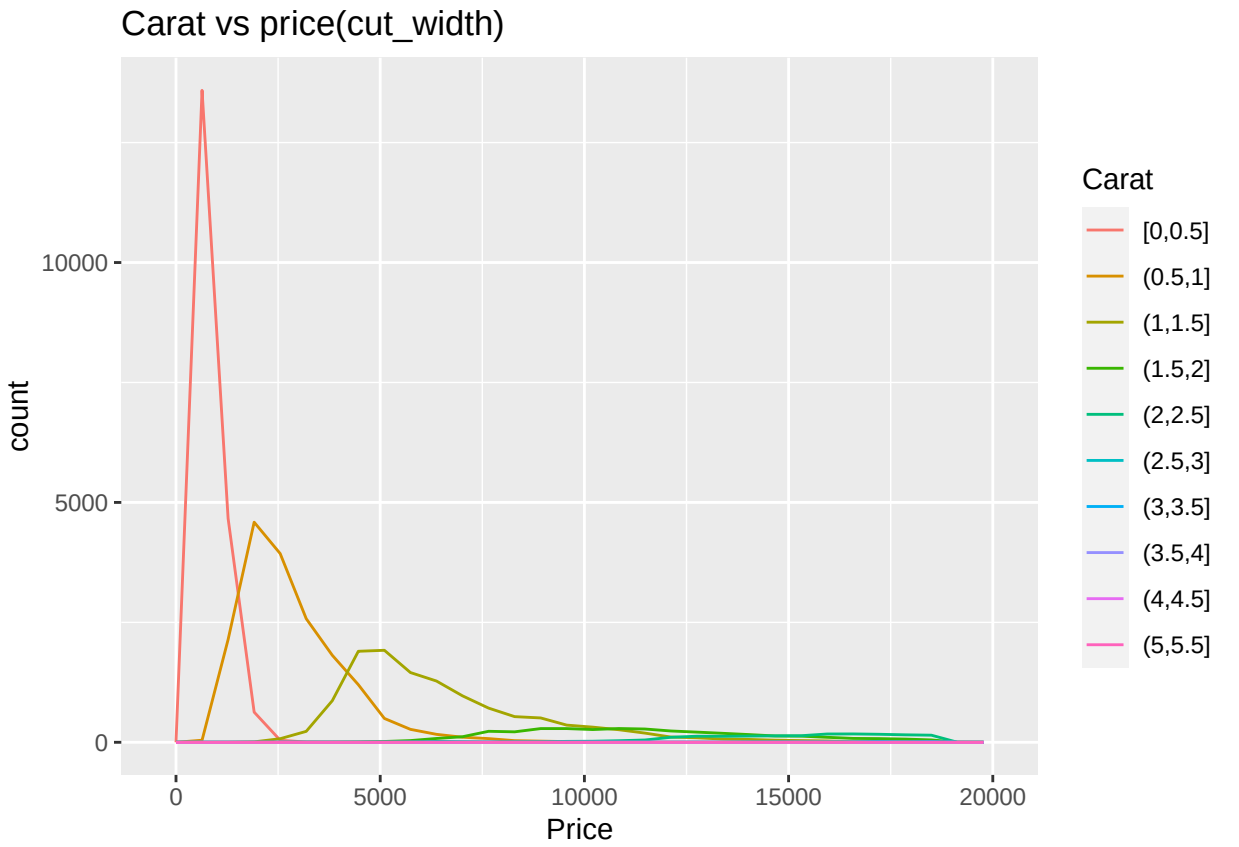
we can use the functions “cut_width()” and “cut_number()” in “geom_freqpoly” to split a variable into groups. #For “cut_width()”,we need to specify the widths and then number of groups will be calculated automatically. #For “cut_number()”,we need to specify the number of groups, the the width of each group will be adjusted automatically.

How does that impact a visualisation of the 2d distribution of carat and price (diamonds dataset)?

#using “cut_width()” to split “carat” variable into groups and then doing the freqpoly plot to see the density of the price for each group of carat

```
ggplot(data = diamonds,
  mapping = aes(color = cut_width(carat, .5, boundary = 0), x = price)) +
  geom_freqpoly() +
  labs(x = "Price", color = "Carat")+ggtitle("Carat vs price(cut_width)")
```

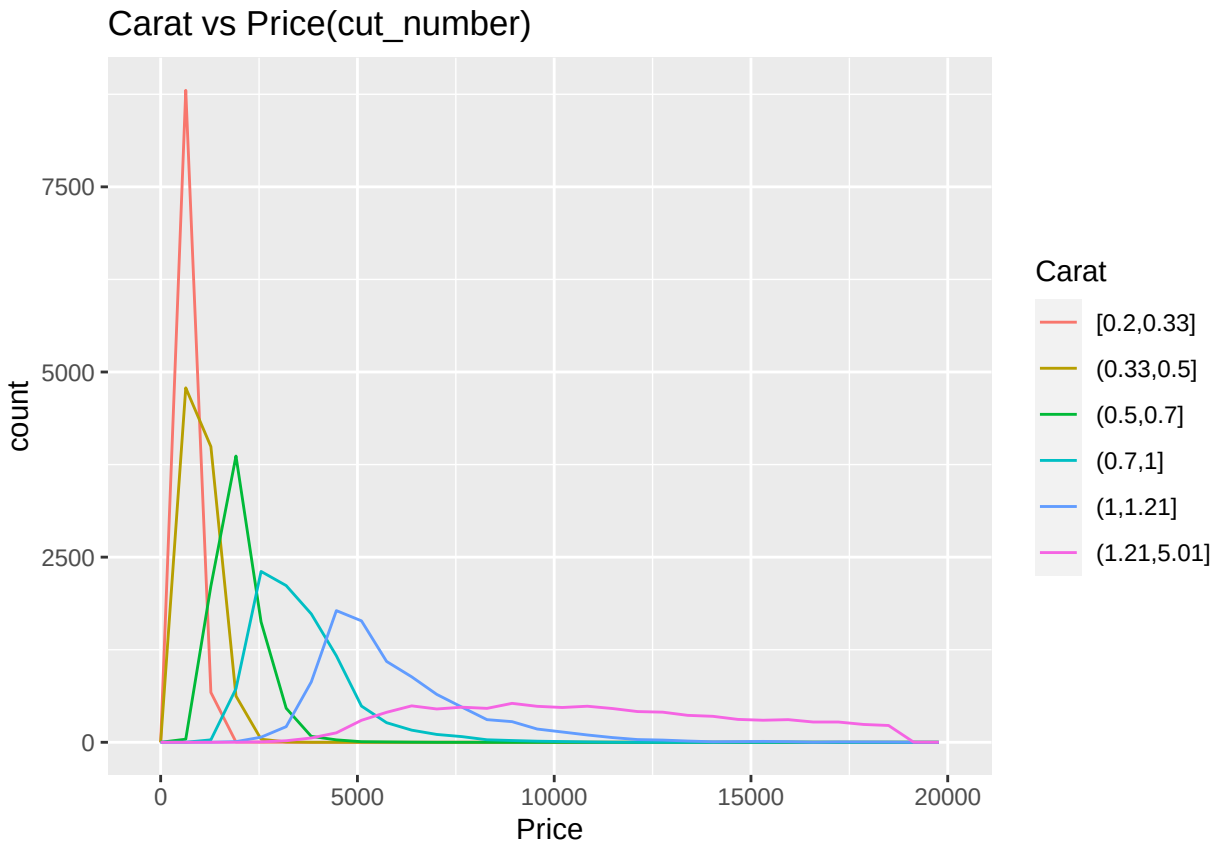
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



#using "cut_number()" to split "carat" variable into groups and then doing the freqpoly plot to see the density of the price for each group of carat

```
ggplot(data = diamonds,
  mapping = aes( x = price)) +
  geom_freqpoly( mapping = aes(color = cut_number(carat, 6))) +
  labs(x = "Price", color = "Carat")+ggtitle("Carat vs Price(cut_number)")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



#Exercise-6 Download genome data:(https://raw.githubusercontent.com/isglobalbrge/Master_Modelling/main/data/genome.txt) into your computer and load it into RStudio by using read_delim function (NOTE: data are tab-delimited).

```
setwd("F:/INTERMATH/intermath 2021-2023/spain/DV/R for data Science")
genome<-read_delim('genome.txt')
```

```
## Rows: 733202 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (2): Name, Chr
## dbl (3): Position, Log.R.Ratio, B.Allele.Freq
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
genome
```

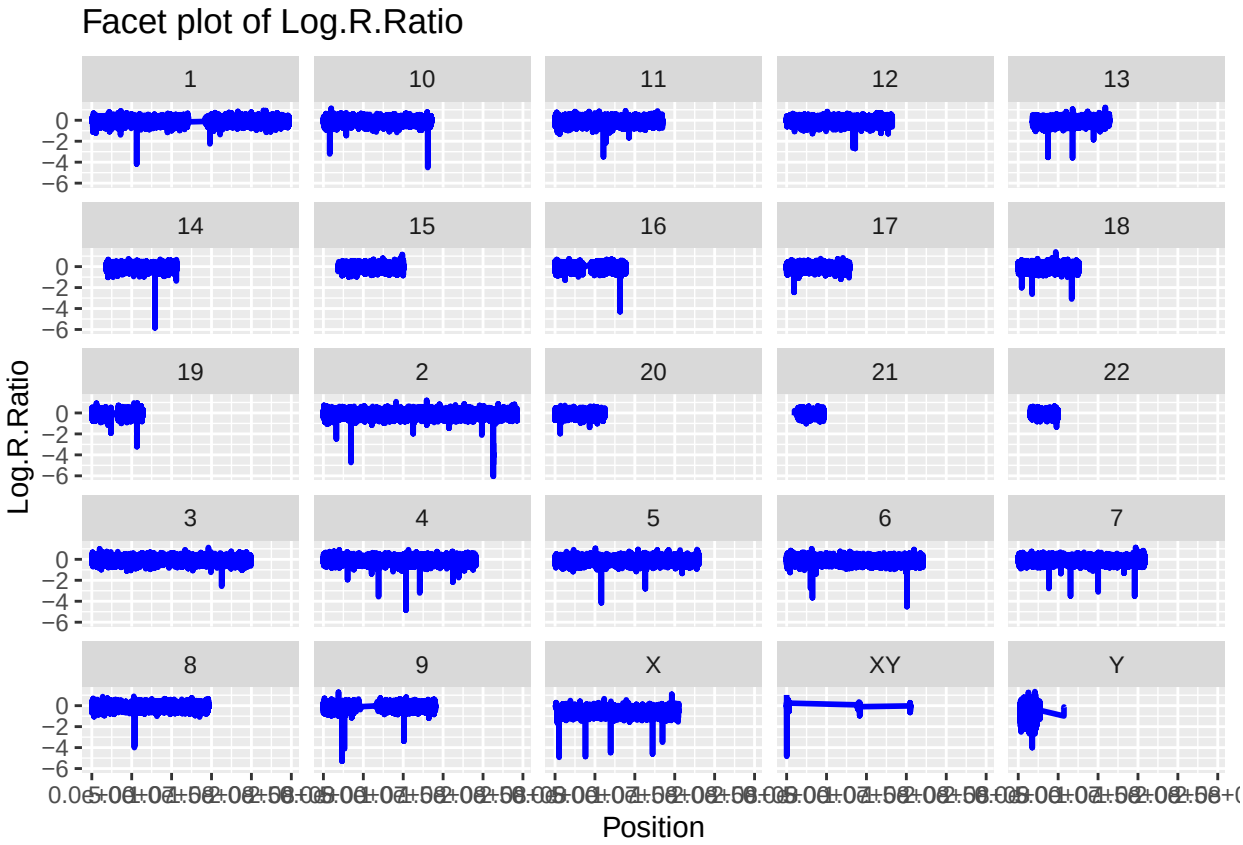
```
## # A tibble: 733,202 x 5
##    Name       Chr    Position Log.R.Ratio B.Allele.Freq
##    <chr>      <chr>     <dbl>       <dbl>         <dbl>
##  1 rs1000000  12    125456933    -0.00250      1
##  2 rs1000002  3     185118461    -0.0297       0.000336
##  3 rs10000023 4      95952928     0.00402      0.461
##  4 rs1000003  3      99825597    -0.143        0.541
```

```
##  5 rs10000030 4     103593179     0.365        1
##  6 rs10000037 4      38600725    -0.00518      0.505
##  7 rs10000041 4     165841405    -0.179        0
##  8 rs10000042 4       5288053     0.168        0.998
##  9 rs10000049 4     119167668    -0.00238      0
## 10 rs1000007  2     237416793    -0.00411      0
## # ... with 733,192 more rows
```

```
#6.1)Which is the expected value of Log.R.Ratio and B.Allel.Freq for each chromosome? (show the R code

genome %>%
  group_by(Chr) %>%
summarise(mean_log = mean(Log.R.Ratio, na.rm = TRUE)
          ,mean_B.Allele = mean( B.Allele.Freq, na.rm = TRUE))
```

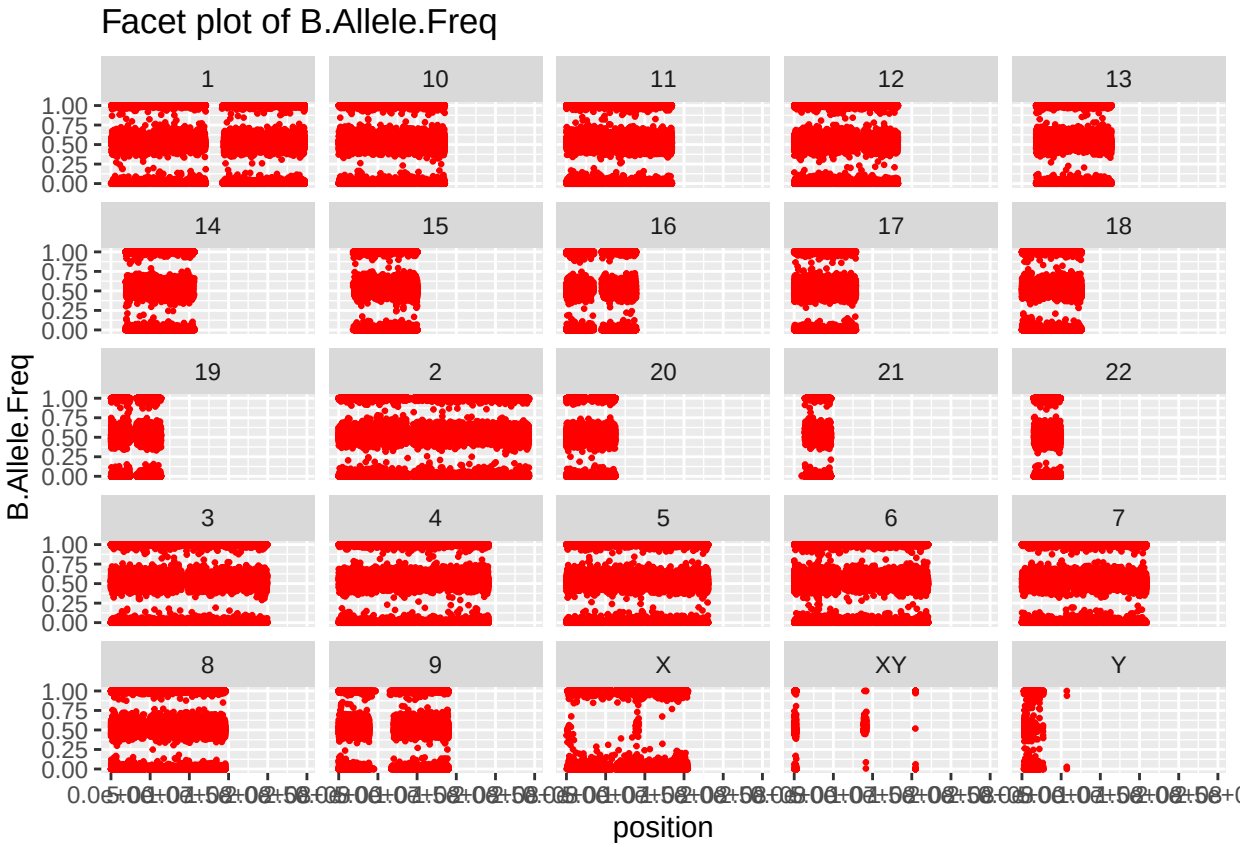
```
## # A tibble: 25 x 3
##    Chr   mean_log mean_B.Allele
##    <chr>    <dbl>         <dbl>
##  1 1     -0.0171          0.534
##  2 10    -0.0208          0.542
##  3 11    -0.0250          0.544
##  4 12    -0.0207          0.533
##  5 13    -0.0281          0.535
##  6 14    -0.0220          0.534
##  7 15    -0.0152          0.521
##  8 16    -0.0140          0.542
##  9 17    -0.00849         0.546
## 10 18    -0.0243          0.530
## # ... with 15 more rows
```

```
#6.2: Create a facet plot that represent the Log.R.Ratio for each chromosome

ggplot(data = genome, aes(Position,Log.R.Ratio)) +
  geom_line(color = "blue", size = 1) +
  labs(title = "Facet plot of Log.R.Ratio",
       y = "Log.R.Ratio", x = "Position") +
  facet_wrap(~ Chr)
```




```
#6.3: Create a facet plot of B.Allele.Freq for chromosomes 1, 2, 3, . . . , 6 drawing B.Allele.Freq inf

ggplot(data = genome, aes(Position,B.Allele.Freq)) +
  geom_point(color="red",size = .5) +
  labs(title = "Facet plot of B.Allele.Freq",
       x = "position", y = "B.Allele.Freq") +
  facet_wrap(~ Chr)
```

```
## Warning: Removed 414 rows containing missing values (geom_point).
```



Exercise: 3 Visualize the number of flights of each airline by month.

```
library(nycflights13)
print(count(flights,carrier,month) ,n=180)
```

```
## # A tibble: 185 x 3
##     carrier month     n
##     <chr>   <int> <int>
##   1 9E          1  1573
##   2 9E          2  1459
##   3 9E          3  1627
##   4 9E          4  1511
##   5 9E          5  1462
##   6 9E          6  1437
##   7 9E          7  1494
##   8 9E          8  1456
##   9 9E          9  1540
##  10 9E         10  1673
##  11 9E         11  1595
##  12 9E         12  1633
##  13 AA          1  2794
##  14 AA          2  2517
##  15 AA          3  2787
##  16 AA          4  2722
##  17 AA          5  2803
##  18 AA          6  2757
```

```
##  19 AA          7  2882
##  20 AA          8  2856
##  21 AA          9  2614
##  22 AA         10  2715
##  23 AA         11  2577
##  24 AA         12  2705
##  25 AS          1    62
##  26 AS          2    56
##  27 AS          3    62
##  28 AS          4    60
##  29 AS          5    62
##  30 AS          6    60
##  31 AS          7    62
##  32 AS          8    62
##  33 AS          9    60
##  34 AS         10    62
##  35 AS         11    52
##  36 AS         12    54
##  37 B6          1  4427
##  38 B6          2  4103
##  39 B6          3  4772
##  40 B6          4  4517
##  41 B6          5  4576
##  42 B6          6  4622
##  43 B6          7  4984
##  44 B6          8  4952
##  45 B6          9  4291
##  46 B6         10  4361
##  47 B6         11  4289
##  48 B6         12  4741
##  49 DL          1  3690
##  50 DL          2  3444
##  51 DL          3  4189
##  52 DL          4  4092
##  53 DL          5  4082
##  54 DL          6  4126
##  55 DL          7  4251
##  56 DL          8  4318
##  57 DL          9  3883
##  58 DL         10  4093
##  59 DL         11  3849
##  60 DL         12  4093
##  61 EV          1  4171
##  62 EV          2  3827
##  63 EV          3  4726
##  64 EV          4  4561
##  65 EV          5  4817
##  66 EV          6  4456
##  67 EV          7  4641
##  68 EV          8  4563
##  69 EV          9  4725
##  70 EV         10  4908
##  71 EV         11  4471
##  72 EV         12  4307
```

```
##  73 F9          1    59
##  74 F9          2    49
##  75 F9          3    57
##  76 F9          4    57
##  77 F9          5    58
##  78 F9          6    55
##  79 F9          7    58
##  80 F9          8    55
##  81 F9          9    58
##  82 F9         10    57
##  83 F9         11    61
##  84 F9         12    61
##  85 FL          1   328
##  86 FL          2   296
##  87 FL          3   316
##  88 FL          4   311
##  89 FL          5   325
##  90 FL          6   252
##  91 FL          7   263
##  92 FL          8   263
##  93 FL          9   255
##  94 FL         10   236
##  95 FL         11   202
##  96 FL         12   213
##  97 HA          1    31
##  98 HA          2    28
##  99 HA          3    31
## 100 HA          4    30
## 101 HA          5    31
## 102 HA          6    30
## 103 HA          7    31
## 104 HA          8    31
## 105 HA          9    25
## 106 HA         10    21
## 107 HA         11    25
## 108 HA         12    28
## 109 MQ          1  2271
## 110 MQ          2  2044
## 111 MQ          3  2256
## 112 MQ          4  2211
## 113 MQ          5  2284
## 114 MQ          6  2178
## 115 MQ          7  2261
## 116 MQ          8  2263
## 117 MQ          9  2206
## 118 MQ         10  2228
## 119 MQ         11  2056
## 120 MQ         12  2139
## 121 OO          1     1
## 122 OO          6     2
## 123 OO          8     4
## 124 OO          9    20
## 125 OO         11     5
## 126 UA          1  4637
```

```
## 127 UA          2  4346
## 128 UA          3  4971
## 129 UA          4  5047
## 130 UA          5  4960
## 131 UA          6  4975
## 132 UA          7  5066
## 133 UA          8  5124
## 134 UA          9  4694
## 135 UA         10  5060
## 136 UA         11  4854
## 137 UA         12  4931
## 138 US          1  1602
## 139 US          2  1552
## 140 US          3  1721
## 141 US          4  1727
## 142 US          5  1785
## 143 US          6  1736
## 144 US          7  1786
## 145 US          8  1779
## 146 US          9  1698
## 147 US         10  1846
## 148 US         11  1699
## 149 US         12  1605
## 150 VX          1   316
## 151 VX          2   271
## 152 VX          3   303
## 153 VX          4   466
## 154 VX          5   496
## 155 VX          6   480
## 156 VX          7   489
## 157 VX          8   489
## 158 VX          9   453
## 159 VX         10   472
## 160 VX         11   451
## 161 VX         12   476
## 162 WN          1   996
## 163 WN          2   911
## 164 WN          3   998
## 165 WN          4   980
## 166 WN          5  1006
## 167 WN          6  1028
## 168 WN          7  1076
## 169 WN          8  1047
## 170 WN          9  1010
## 171 WN         10  1091
## 172 WN         11  1033
## 173 WN         12  1099
## 174 YV          1    46
## 175 YV          2    48
## 176 YV          3    18
## 177 YV          4    38
## 178 YV          5    49
## 179 YV          6    49
## 180 YV          7    81
```

```
## # ... with 5 more rows
```

```
ggplot(data = flights, mapping = aes(x = factor(month),fill=factor(month))) +
  geom_bar() +
  facet_wrap(~ carrier, scales= "free")
```