R Programming Practical (Data Science)

Lobsang Lama

Question 1

Create a vector of size 10, having the values 5,7,9,11,13,13,11,9,7,5. Compute the sum, mean, highest and lowest of these values. Compute the length of this vector? Find the variance and standard deviation for the data of this vector, using the formula for variance and standard deviation. Compare these values by computing the variance and standard deviation using R function. Sort this array values in decreasing order.

```
value<-c(5,7,9,11,13,13,11,9,7,5)
print(paste("The sum of vectors: ",sum(value)))
```

```
## [1] "The sum of vectors:  90"
```

```
m<-mean(value)
print(paste("Mean of vectors: ",m))
```

```
## [1] "Mean of vectors:  9"
```

```
print(paste("Maximum in the vectors: ",max(value)))
```

```
## [1] "Maximum in the vectors:  13"
```

```
print(paste("Minimum in the vectors: ",min(value)))
```

```
## [1] "Minimum in the vectors:  5"
```

```
len=length(value)
print(paste("Length of the vector: ",len))
```

```
## [1] "Length of the vector:  10"
```

```
variance=0
for(i in 1:len){
  d=value[i]-m
  d<-d*d
  variance=variance+d
}
variance<-(variance/len)
print(paste("Variance: ",variance))
```

```
## [1] "Variance:  8"
```

```
sd=sqrt(variance)
print(paste("Standard Deviation: ",sd))
```

```
## [1] "Standard Deviation:  2.82842712474619"
```

```
print(paste("Actual Variance: ",var(value)))
```

```
## [1] "Actual Variance:  8.88888888888889"
```

```
print(paste("Actual Standard Deviation: ",sd(value)))
```

```
## [1] "Actual Standard Deviation:  2.98142396999972"
```

```
desc=sort(value, decreasing = TRUE)
print(desc)
```

```
##  [1] 13 13 11 11  9  9  7  7  5  5
```

Question 2

Create a vector of first 50 even numbers, starting from 2. Also create a vector having values 30 down to 1, as 30, 29, . . . ,1

```
vec1<-seq(from=2 , to=100 , by=2)
print(paste("50 Even numbers starting from 2"))
```

```
## [1] "50 Even numbers starting from 2"
```

```
print(vec1)
```

```
##  [1]   2   4   6   8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38
## [20]  40  42  44  46  48  50  52  54  56  58  60  62  64  66  68  70  72  74  76
## [39]  78  80  82  84  86  88  90  92  94  96  98 100
```

```
vec2<-seq(from=30 , to=1 , by=-1)
print(paste("vector having 30 to 1"))
```

```
## [1] "vector having 30 to 1"
```

```
print(vec2)
```

```
##  [1] 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10  9  8  7  6
## [26]  5  4  3  2  1
```

Question 3

Create a vector of size 10 with 5th and 7th values as missing (store these values as NA). Use the "is.na()" to find locations of missing data.

```
vector1 <- c(1,2,3,4,NA,6,NA,8,9,10)
is.na(vector1)
```

```
##  [1] FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE
```

```
which(is.na(vector1))
```

```
## [1] 5 7
```

Question 4

Create a vector of characters of size 5, consisting of values: "This" "is" "a" "character" "vector". Find the index of value "is" in the vector using which() or match().

```
vec<-c("this","is","a","character","vector")
i<-match("is",vec)
print(i)
```

```
## [1] 2
```

Question 5

It is always good to store numerical values rather than textual data. However, while input or output the textual values are easier to understand. An example, for this is as follows in R: Fivepointscale=c(1:5) names(Fivepointscale) = c("Not Satisfactory", "Satisfactory", "Fair", "Good", "Very Good") Feedback = Fivepointscale[c("Good", "Satisfactory")] Create a 7-point scale of information input and use this scale to input feedback of 5 students about a question like "Feedback of experience of using an application (Bad, Somewhat bad, not good, ok, good, very good, excellent)". Find the average of the feedback.

```
Sevenscale=c(1:7)
names(Sevenscale)=c("Bad","Somewhat bad","Not Good","Ok","Good","Very Good", "Excellent")

v<- c("Bad","Ok","Very Good","Bad","Not Good")

print("Feedback of 5 students")
```

```
## [1] "Feedback of 5 students"
```

```
print(v)
```

```
## [1] "Bad"       "Ok"        "Very Good" "Bad"       "Not Good"
```

```
Feedback = Sevenscale[v]
res<-mean(Feedback)
print(paste("Feedback of experience of using an application"))
```

```
## [1] "Feedback of experience of using an application"
```

```
print(names(Sevenscale[res]))
```

```
## [1] "Not Good"
```

Question 6:

Create two strings and concatenate them.

```
res <- "HELLO"
ult <- "WORLD"

result <- paste(res,ult)
print(res)
```

```
## [1] "HELLO"
```

```
print(ult)
```

```
## [1] "WORLD"
```

```
print(result)
```

```
## [1] "HELLO WORLD"
```

Question 7:

Create a long string of words separated by punctuation marks. Replace all the punctuation marks in the string using gsub("[[:punct:]]", " ", stringName) function. Find the number of words in the string without punctuation marks. Find the number of distinct words and its count, if possible.

```
str<-"This has a long string of words with a lot of words. The string has hello world? No,
it has hello hello world!"
res<-gsub("[[:punct:]]", "", str)
```

```
distinct_words<-strsplit(res," ")
print("String after split:")
```

```
## [1] "String after split:"
```

```
print(distinct_words)
```

```
## [[1]]
##  [1] "This"   "has"    "a"      "long"   "string" "of"     "words"  "with"
##  [9] "a"      "lot"    "of"     "words"  "The"    "string" "has"    "hello"
## [17] "world"  "No"     "\nit"   "has"    "hello"  "hello"  "world"
```

```
print("length of the string")
```

```
## [1] "length of the string"
```

```
print(sapply(distinct_words , length))
```

```
## [1] 23
```

```
print(table(distinct_words))
```

```
## distinct_words
##   \nit      a    has  hello   long    lot     No     of string    The   This
##      1      2      3      3      1      1      1      2      2      1      1
##   with  words  world
##      1      2      2
```

Question 8

Question 8: Store content in external files for the following types of data in R: (i) Vectors (ii) Lists (iii) Arrays (iv) Data frames (v) Factors Read those contents of csv file into R. Perform operations like sorting on vector data, finding the length of lists and adding data items in list, accessing different elements of array and comparing it to other values, accessing different components of data frames and factors.

```
setwd("D:\\college\\sixth semester\\Data Science\\R")
#print(getwd())

data<-read.csv("student_data.csv")

vectordata <- as.vector(data)#converting csv file into vector

sorted<-sort(vectordata$First_name)#sorting First_name field from the vector First_name
print(sorted)
```

```
##  [1] "ANANTA"     "Anshu"      "Anurag"     "Apurva"     "BRITIKA"
##  [6] "md "        "MILRED "    "Mohammad "  "Neeraj"     "Sharnojit"
## [11] "Siddhanth"  "SIDDHARTH " "Tashi "
```

```
#converting csv file into list
lst1=list()

for(i in 1:ncol(data)) {
  lst1[[i]] <- data[ , i]
}

names(lst1)=colnames(data)
```

```r
len<-length(lst1$First_name)#find the length of the list
print(len)
```

```
## [1] 13
```

```r
res<-append(lst1$First_name,"hello")#add items in the list
print(lst1$First_name)#printing...
```

```
##  [1] "BRITIKA"    "SIDDHARTH " "Anshu"      "md "        "Apurva"
##  [6] "Mohammad "  "Neeraj"     "ANANTA"     "Siddhanth"  "Sharnojit"
## [11] "Anurag"     "MILRED "    "Tashi "
```

```r
print(length(res))
```

```
## [1] 14
```

```r
#converting csv file data into array
arr<-array(unlist(lst1),
           dim = c(5, 5, 10))
#accessing array elements from 3 dimensional array
print(arr[5,1,5])
```

```
## [1] "BRITIKA"
```

```r
print(length(arr))
```

```
## [1] 250
```

```r
value <- c("Anshu",2580,NA)

if (any(is.na(arr)) && is.na(as.numeric(value))) {
  print(paste(value, "is present in the array."))
} else if (value %in% arr) {
  print(paste(value, "is present in the array."))
} else {
  print(paste(value, "is not present in the array."))
}
```

```
## Warning: NAs introduced by coercion
```

```
## Warning in any(is.na(arr)) && is.na(as.numeric(value)): 'length(x) = 3 > 1' in
## coercion to 'logical(1)'
```

```
## [1] "Anshu is present in the array." "2580 is present in the array."
## [3] "NA is present in the array."
```

```r
print(data$First_name)#accessing different components of data frames
```

```
##  [1] "BRITIKA"    "SIDDHARTH " "Anshu"      "md "        "Apurva"
##  [6] "Mohammad "  "Neeraj"     "ANANTA"     "Siddhanth"  "Sharnojit"
## [11] "Anurag"     "MILRED "    "Tashi "
```

```r
#converting csv file data into factor
factordata <- as.factor(data$First_name)

print(factordata[1])#accessing different components of factors
```

```
## [1] BRITIKA
## 13 Levels: ANANTA Anshu Anurag Apurva BRITIKA md  MILRED  Mohammad  ... Tashi
```

Question 9:

Create two matrices of 3*3 size using R, add, subtract and multiply these two matrices.

```
m1<-c(1,3,6,2,7,9,2,4,6)
m2<-c(3,6,9,1,4,5,7,2,8)

m1 = matrix(m1, nrow = 3, ncol=3)
print("Matrix 1:")
```

```
## [1] "Matrix 1:"
```

```
print(m1)
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    2
## [2,]    3    7    4
## [3,]    6    9    6
```

```
m2 = matrix(m2, nrow = 3, ncol=3)
print("Matrix 2:")
```

```
## [1] "Matrix 2:"
```

```
print(m2)
```

```
##      [,1] [,2] [,3]
## [1,]    3    1    7
## [2,]    6    4    2
## [3,]    9    5    8
```

```
result = m1 + m2
print("Result of addition")
```

```
## [1] "Result of addition"
```

```
print(result)
```

```
##      [,1] [,2] [,3]
## [1,]    4    3    9
## [2,]    9   11    6
## [3,]   15   14   14
```

```
result = m1 - m2
print("Result of subtraction")
```

```
## [1] "Result of subtraction"
```

```
print(result)
```

```
##      [,1] [,2] [,3]
## [1,]   -2    1   -5
## [2,]   -3    3    2
## [3,]   -3    4   -2
```

```
result = m1 * m2
print("Result of multiplication")
```

```
## [1] "Result of multiplication"
```

```
print(result)
```

```
##      [,1] [,2] [,3]
## [1,]    3    2   14
## [2,]   18   28    8
## [3,]   54   45   48
```

Question 10:

Perform transpose of a matrix.

```
data <- c(1, 4, 7, 2, 8, 4, 3, 0, 9)
A <- matrix(data, nrow = 3, ncol = 3)

A_T <- t(A)

print("Matrix A")
```

```
## [1] "Matrix A"
```

```
print(A)
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    8    0
## [3,]    7    4    9
```

```
print("Transpose of A")
```

```
## [1] "Transpose of A"
```

```
print(A_T)
```

```
##      [,1] [,2] [,3]
## [1,]    1    4    7
## [2,]    2    8    4
## [3,]    3    0    9
```

Question 11:

Find the inverse of a matrix.

```
data <- c(1, 4, 0, -1, 0, 1, 2, 6, -1)
A <- matrix(data, nrow = 3, ncol = 3)

A_I <- solve(A)

print("Matrix A")
```

```
## [1] "Matrix A"
```

```
print(A)
```

```
##      [,1] [,2] [,3]
## [1,]    1   -1    2
## [2,]    4    0    6
## [3,]    0    1   -1
```

```
print("Inverse Matrix of A")
```

```
## [1] "Inverse Matrix of A"
```

```
print(A_I)
```

```
##      [,1] [,2] [,3]
## [1,]    3 -0.5    3
## [2,]   -2  0.5   -1
## [3,]   -2  0.5   -2
```

Question 12:

Create a list of a factor. Find the occurrences of each factor in the list.

```
gender_list<-c("male","male","female","female","female","female")
print(table(gender_list))
```

```
## gender_list
## female   male
##      4      2
```

Question 13:

Write function to find the largest and smallest values in a 3-dimensional array of size 3*3*3. You should use parameter passing.

```
#function to find minimum and maximum
minmax <- function(data) {
  print("Maximum in the array")
  print(max(data))
  print("Minimum in the array")
  print(min(data))
}

data <- c(3,2,4,8,5,1)
result <- array(c(data), dim = c(3,3,3))

minmax(data) #calling minmax function
```

```
## [1] "Maximum in the array"
## [1] 8
## [1] "Minimum in the array"
## [1] 1
```

Question 14:

Create a table of showing the States of 20 students, assume these students stay in 5 different states. Now create a factor of these states and then compute the frequency of each factor (Hint: You may use factor() and tapply() functions)

```
vec1<-c("Aryan","Subigya","Anshu","Neeraj","Apurva","Dolly","Ritika","Shubham","Ashwini",
        "Kaavya","Rinzing","Siddhart","Aditya","Nandini","Komal","Riya","Shivangi",
        "Robert","Daniel","Rajeev")
vec2<-c("Manipur","Assam","Assam","Gujrat","Manipur","Kerela","Gujrat","Manipur","Kerela",
        "Gujrat","Jharkhand","Assam","Jharkhand","Gujrat","Manipur","Kerela","Assam",
        "Manipur","Kerela","Gujrat")

studentdata <-data.frame(name = vec1, state =vec2)
final=table(studentdata$name,studentdata$state)
print(final)
```

```
##
##            Assam Gujrat Jharkhand Kerela Manipur
##   Aditya       0      0         1      0       0
##   Anshu        1      0         0      0       0
##   Apurva       0      0         0      0       1
##   Aryan        0      0         0      0       1
##   Ashwini      0      0         0      1       0
##   Daniel       0      0         0      1       0
##   Dolly        0      0         0      1       0
##   Kaavya       0      1         0      0       0
##   Komal        0      0         0      0       1
##   Nandini      0      1         0      0       0
##   Neeraj       0      1         0      0       0
##   Rajeev       0      1         0      0       0
##   Rinzing      0      0         1      0       0
##   Ritika       0      1         0      0       0
##   Riya         0      0         0      1       0
##   Robert       0      0         0      0       1
##   Shivangi     1      0         0      0       0
##   Shubham      0      0         0      0       1
##   Siddhart     1      0         0      0       0
##   Subigya      1      0         0      0       0
```

```
stateslist<-as.factor(vec2)
print(stateslist)
```

```
##  [1] Manipur   Assam     Assam     Gujrat    Manipur   Kerela    Gujrat
##  [8] Manipur   Kerela    Gujrat    Jharkhand Assam     Jharkhand Gujrat
## [15] Manipur   Kerela    Assam     Manipur   Kerela    Gujrat
## Levels: Assam Gujrat Jharkhand Kerela Manipur
```

```
print(table(studentdata$state))
```

```
##
##     Assam    Gujrat Jharkhand    Kerela   Manipur
##         4         5         2         4         5
```

Question 15:

Consider a state wise list of income of few persons. Use factor function to create a frequency division of income into 5 factor classes e.g. 10000-50000; 50000-100000 etc.

```
income<-c(10000,45000,50000,12000,22000,48000,28000,12000,18000)

list_of_income=cut(income, 5, labels=c(10000,20000,30000,40000,50000))
print(table(list_of_income))
```

```
## list_of_income
## 10000 20000 30000 40000 50000
##     4     1     1     0     3
```

Question 16:

Explore different functions in R about strings, arrays, vectors, factors. You may also explore different methods of plotting the data.

```r
#Strings
str <- "Hello World!"
nchar(str)   #find the number of characters in a string
```

```
## [1] 12
```

```r
str <- "Hello World!"
grepl("H", str)
```

```
## [1] TRUE
```

```r
grepl("X", str)   #check if a character or a sequence of characters are present in a string
```

```
## [1] FALSE
```

```r
str1 <- "Hello"
str2 <- "World"
paste(str1, str2)   #merge/concatenate two strings
```

```
## [1] "Hello World"
```

```r
#Vectors
value<-c(5,7,9,11,13,13,11,9,7,5)
print(paste("The sum of vectors: ",sum(value)))
```

```
## [1] "The sum of vectors:  90"
```

```r
print(paste("Mean of vectors: ",mean(value)))
```

```
## [1] "Mean of vectors:  9"
```

```r
print(paste("Maximum in the vectors: ",max(value)))
```

```
## [1] "Maximum in the vectors:  13"
```

```r
print(paste("Minimum in the vectors: ",min(value)))
```

```
## [1] "Minimum in the vectors:  5"
```

```r
print(paste("Length of the vector: ",length(value)))
```

```
## [1] "Length of the vector:  10"
```

```r
print(paste("Variance: ",var(value)))
```

```
## [1] "Variance:  8.88888888888889"
```

```r
print(paste("Standard Deviation: ",sd(value)))
```

```
## [1] "Standard Deviation:  2.98142396999972"
```

```r
#Factors
music_genre <- factor(c("Jazz", "Rock", "Classic", "Classic", "Pop", "Jazz", "Rock", "Jazz"))
print(music_genre)
```

```
## [1] Jazz    Rock    Classic Classic Pop     Jazz    Rock    Jazz
## Levels: Classic Jazz Pop Rock
```

```r
#Visualisation
library(tidyverse)
```

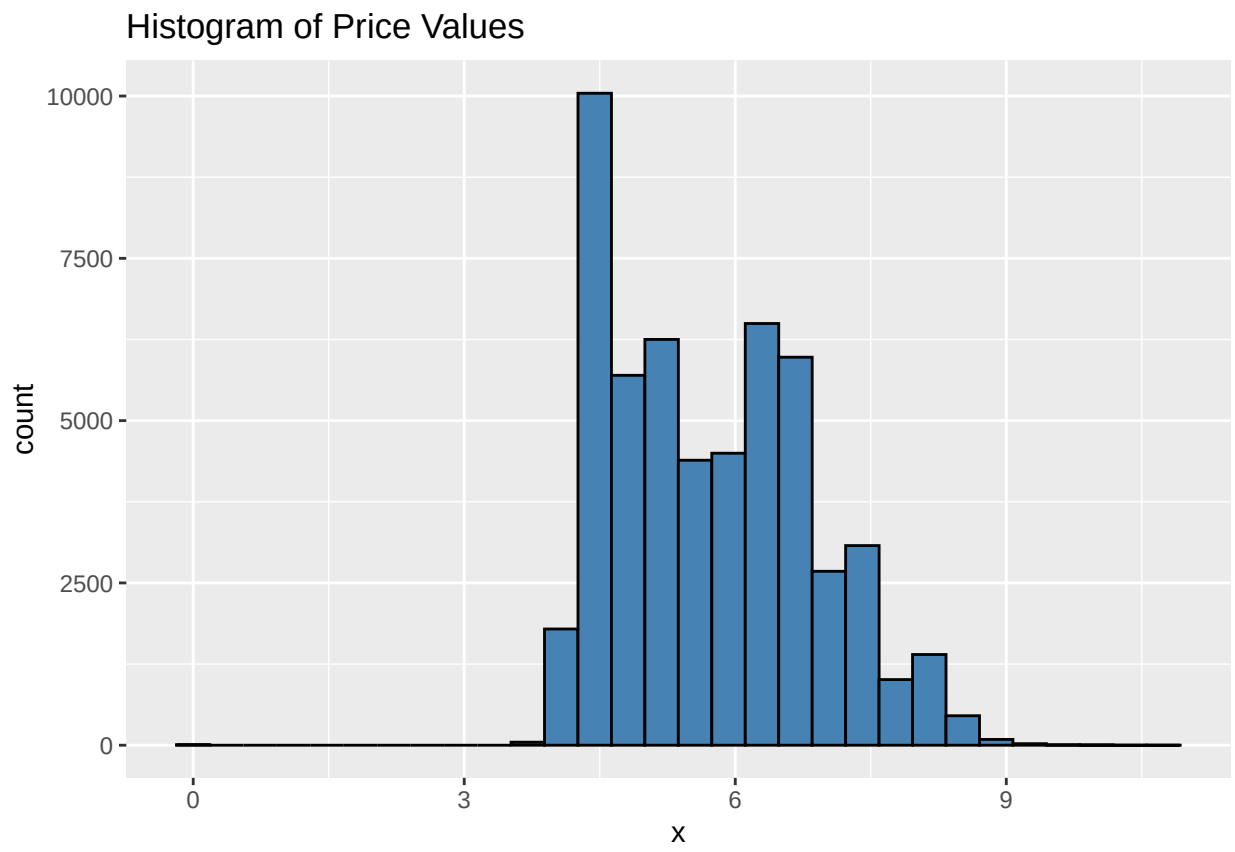
```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.1     v purrr   1.0.1
```

```
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.4     v forcats 1.0.0
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

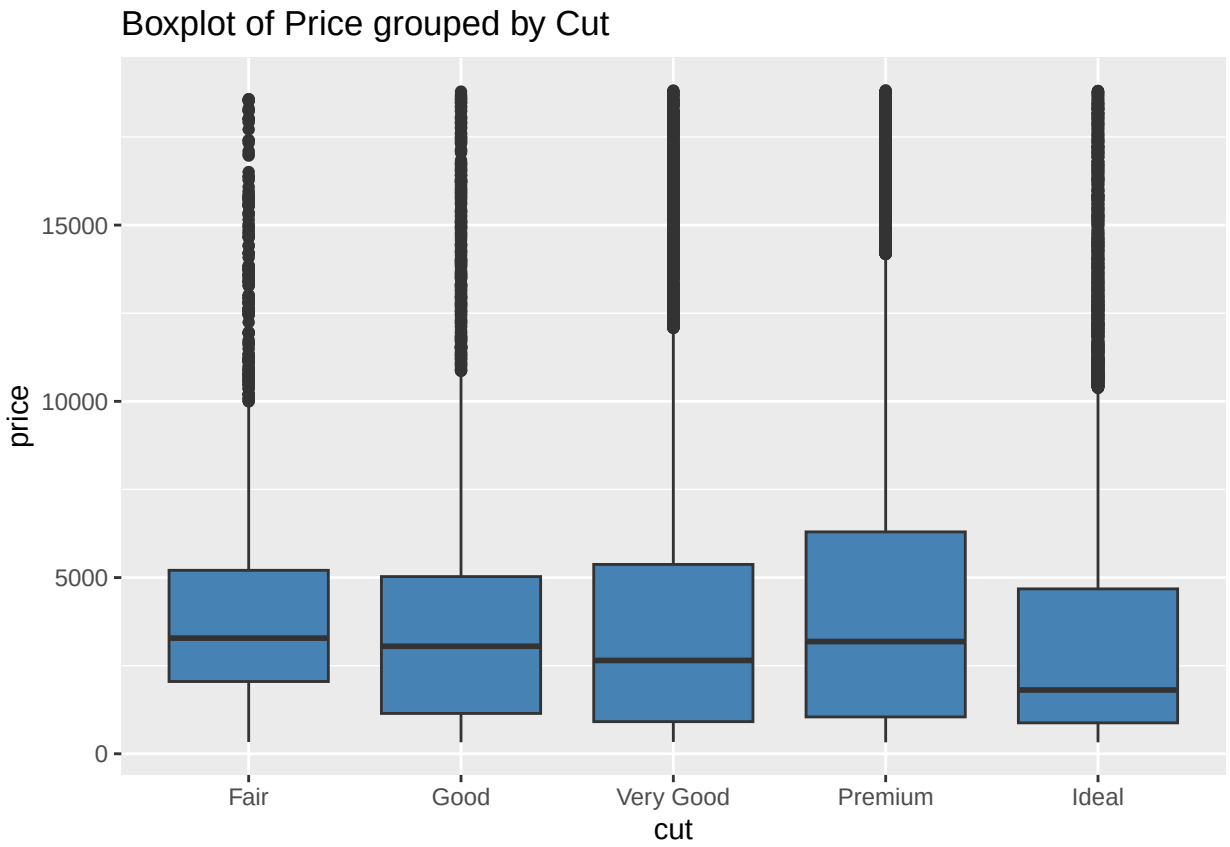
```
data(diamonds)

#create histogram
ggplot(data=diamonds, aes(x=x)) +
  geom_histogram(fill="steelblue", color="black") +
  ggtitle("Histogram of Price Values")
```

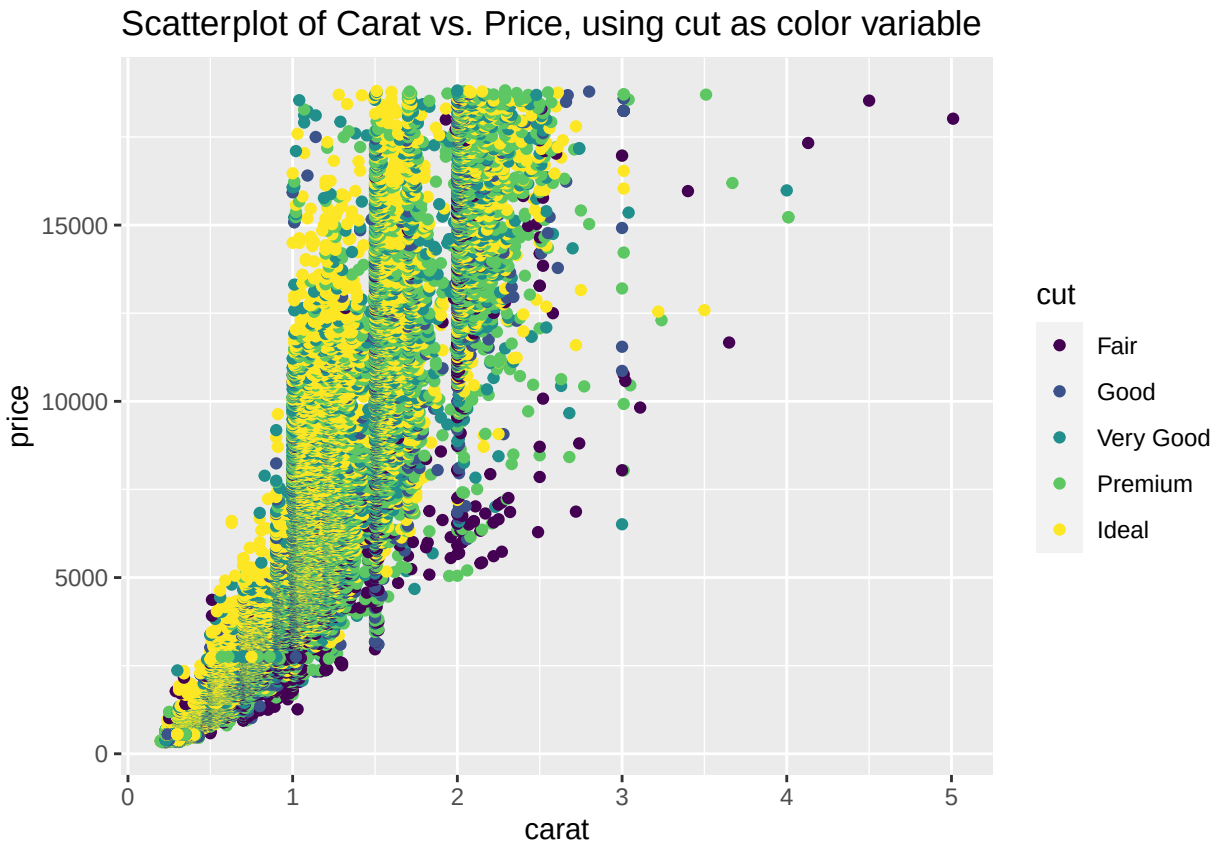
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



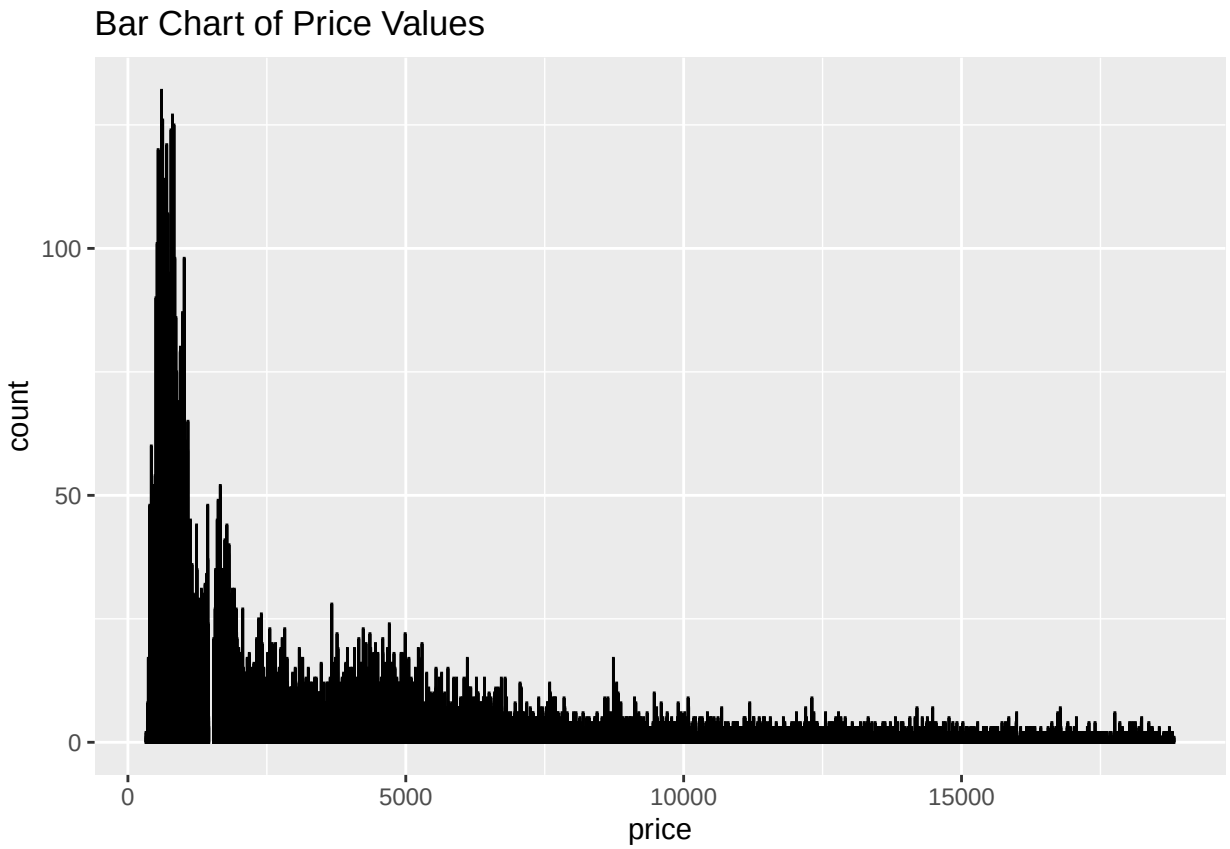

```
ggplot(data=diamonds, aes(x=cut, y=price)) +
  geom_boxplot(fill="steelblue") +
  ggtitle("Boxplot of Price grouped by Cut")
```



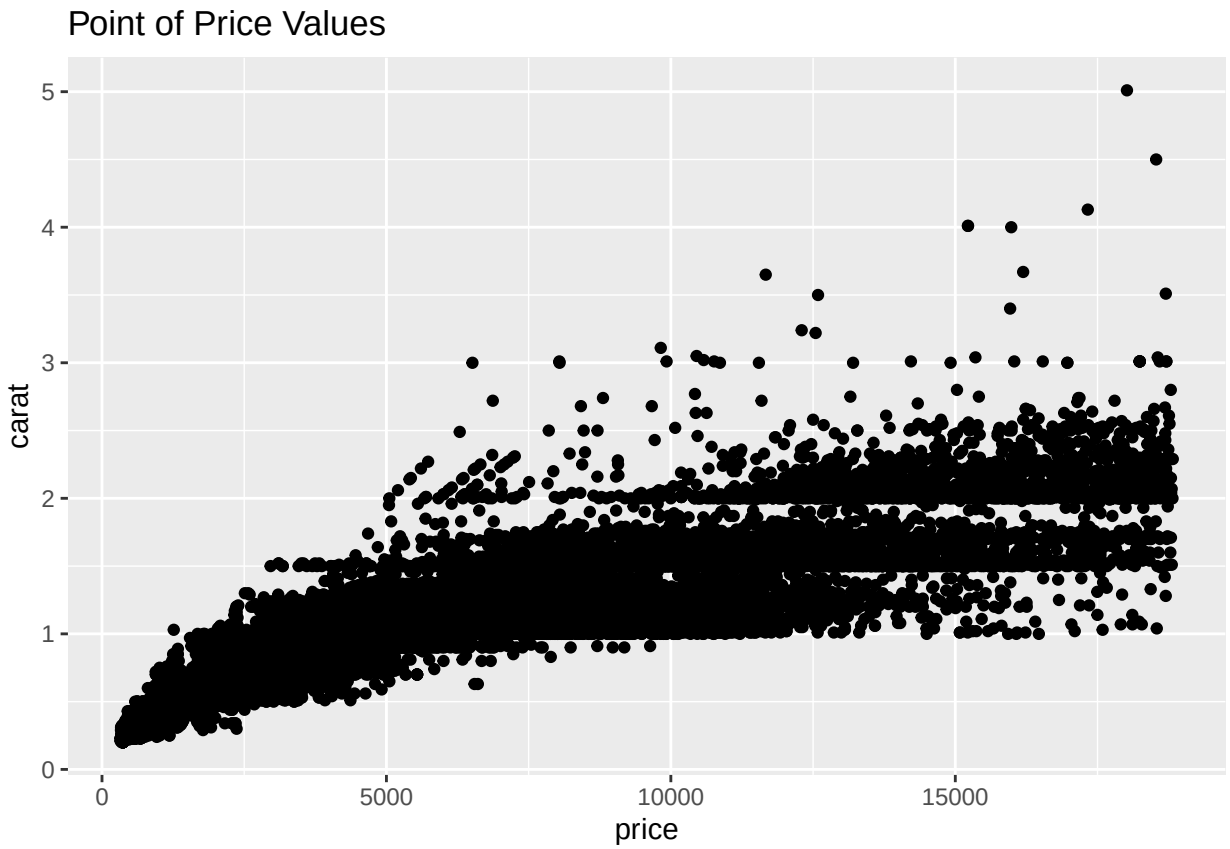

```
ggplot(data=diamonds, aes(x=carat, y=price, color=cut)) +
  geom_point() +
  ggtitle("Scatterplot of Carat vs. Price, using cut as color variable")
```




```
ggplot(data=diamonds, aes(x=price)) +
 geom_bar(fill="steelblue", color="black") +
 ggtitle("Bar Chart of Price Values")
```




```
ggplot(data=diamonds, aes(x=price,y=carat)) +
  geom_point() +
  ggtitle("Point of Price Values")
```


Point of Price Values
carat
price
0
1
2
3
4
5
0
5000
10000
15000