Assignment # 1

Question 1:
Let's use the dataset TeleCustomers.xlsx, copyright © IBM Academic Initiative Program. The dataset contains the following fields describing the customers.

ID	Customer reference number
Sex	Gender
Status	Marital status
Children	Number of Children
Est_Income	Estimated income
Car_Owner	Car owner
Usage	Time spent on calls in total per month
Age	Age
RatePlan	Chosen rate plan (1, 2, …)
LongDistance	Time spent on long distance calls per month
International	Time spent on international calls per month
Local	Time spent on local calls per month
Dropped	Number of dropped calls
Payment	Payment method of the monthly telephone bill
LocalBillType	Tariff for locally based calls
LongDistanceBillType	Tariff for long distance calls
CHURNED	Current vs. Cancelled

Assume that you are a data scientist or a manger overseeing the marketing department in this telecommunications company.

1. (20 points) Try your best to answer the proposed questions below by filling in the form.

	Descriptive Analytics	Predictive Analytics	Prescriptive Analytics
	What **HAS** happened?	What **COULD** happen?	What **SHOULD** happen?
What do you need to **DO** (goal-driven tasks)	Decrease the customer churn rate and hence customer retention.	Predict if the customers are in risk of churning or not	Using customer segment targeted campaigns, we decrease the probability of customer churning. Thus increasing the revenue and profitability of the

			company. (Cluster based targeting)
What do you need to **KNOW** (insights that help you get the tasks done well)	Why the customers have churned, the number of churns based on description of other features (example churn by sex, age, usage, etc.)	How to anticipate why the customer's churn. How to determine features to decrease customer churn ration	How to decrease the churn ratio. Which cluster-based customer targeting will provide the highest customer retention ratio.

2. (20 points) What other data you will need to help you complete "What do you need to KNOW"? Please lists them based on types of analytics.

   Other data that might be useful is knowing how long the customer is a user of the services (tenure), if they have multiple lines and if they have other subscriptions as well.

	Descriptive Analytics	Predictive Analytics	Prescriptive Analytics
	What **HAS** happened?	What **COULD** happen?	What **SHOULD** happen?
What do you need to **KNOW** (insights that help you get the tasks done well)	Relation between the customer churn vs their tenure (subscription time)	Need to know how to anticipate why customers churn with different tenure bins. (For example, customers with lower tenure might me more probable to churn, hence given tenure value of a customer with other features you need to know if they could churn or not)	How to target customers with different tenure values for retention.

Question 2:

(20 points) Define your own functions in R to calculate mean, variance, and standard deviation. (**DO NOT** USE IN-BUILT R FUNCTIONS)

Load data

```
In [3]: library("readxl")
```

```
In [4]: data <- read_excel("TeleCustomers Spreadsheet.xlsx")
```

Manual Mean calculation function (Without in-built function)

In [7]:
```
mean_manual <- function(x)
{
  l  <-  length(x)
  sum <- 0
  for (i in 1:l)
  {
    sum <- sum + x[i]
  }
  return(sum/l)
}
```

In [8]:
```
variance_manual=function(x)
{
    sum <- 0
    l <- length(x)
    for (i in 1:l)
    {
    sum  <-   sum+((x[i]-mean_manual(x))^2)
    }
    return(sum/(l-1))
}
```

In [9]:
```
std_manual=function(x){
    return((variance_manual(x))^0.5)
}
```

Question 3:

(20 points) Use your own functions (in question 2) to calculate mean, variance, and standard deviation for “Estimated income” in the provided dataset.

```
mean_m <- formattable(mean_manual(data$Est_Income), digits = 2, format = "f")
var_m <- formattable(variance_manual(data$Est_Income), digits = 2, format = "f")
std_m <- formattable(std_manual(data$Est_Income), digits = 2, format = "f")
```

```
print(paste("mean without function",mean_m))

print(paste("variance without function",var_m))

print(paste("std without function",std_m))
```

```
[1] "mean without function 51464.26"
[1] "variance without function 948435534.13"
[1] "std without function 30796.68"
```

Function calculation

In [16]:
```
#install.packages("formattable")
```

In [17]:
```
library(formattable)
```

In [18]:
```
mean_m <- formattable(mean_manual(data$Est_Income), digits = 2, format = "f")
var_m <- formattable(variance_manual(data$Est_Income), digits = 2, format = "f")
std_m <- formattable(std_manual(data$Est_Income), digits = 2, format = "f")
mean <- formattable(mean(data$Est_Income), digits = 2, format = "f")
var <- formattable(var(data$Est_Income), digits = 2, format = "f")
std <- formattable(sd(data$Est_Income), digits = 2, format = "f")
```

Comparing with function and without function values

In [19]:
```
help(min)
```

In [20]:
```
?var
```

In [22]:
```
print(paste("mean without function",mean_m))
print(paste("mean with function",mean))
print(paste("variance without function",var_m))
print(paste("variance with function",var))
print(paste("std without function",std_m))
print(paste("std with function",std))
```

```
[1] "mean without function 51464.26"
[1] "mean with function 51464.26"
[1] "variance without function 948435534.13"
[1] "variance with function 948435534.13"
[1] "std without function 30796.68"
[1] "std with function 30796.68"
```

Question 4:

(20 points) Consider *iris* data set (It is available in your R workspace). Take a new sample vector {4.8, 2.9, 3.7, 1.7} as values of { "Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width" } respectively.
Now find out the Euclidean distances between the new sample and all the rows of *"iris"* (exclude the Species column).
Sort the distances in an ascending order.

Loading the Iris dataset

```
library(datasets)
data("iris")
iris
```

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa

Creating custom samples and giving names

```
custom_vector <- c(4.8, 2.9, 3.7, 1.7)
names(custom_vector) <- c("Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width" )
custom_vector
```

Sepal.Length	4.8
Sepal.Width	2.9
Petal.Length	3.7
Petal.Width	1.7

Removing the Species column

```
iris_nospecies <- iris[1:4]
iris_nospecies
```

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2

Creating function for Euclidean distance

In [27]:
```
eucl_dist_manual <- function(x1, x2)
{
  sqrt(sum((x1 - x2) ^ 2))
}
```

Calculating and sorting in ascending order the eucidian dist of created sample and each row in iris dataset

In [28]:
```
?apply()
```

In [29]:
```
distances <- apply(iris_nospecies,1,function(x) eucl_dist_manual(x,custom_vector))
distances <- sort(distances)
distances
```

```
0.574456264653803  0.9  0.9  0.948683298050514  0.953939201416946  0.98488578017961  1.02469507659596  1.02469507659596
1.04403065089105  1.04880884817015  1.04880884817015  1.05356537528527  1.06770782520313  1.09544511501033  1.10453610171873
1.10905365064094  1.14891252930761  1.14891252930761  1.15325625946708  1.15325625946708  1.17473401244707  1.19582607431014
1.19582607431014  1.22882057274445  1.27279220613579  1.3  1.39642400437689  1.4560219778561  1.47648230602334  1.52970585407784
1.57480157480236  1.58429795177549  1.58745078663875  1.61245154965971  1.63401346383682  1.6583123951777  1.66733320005331
1.71755640373177  1.74355957741627  1.74355957741627  1.75499287747842  1.76918060129541  1.77482393492988  1.78605710994918
1.78605710994918  1.80554700852678  1.82482875908947  1.84932420089069  1.85741756210067  1.86010752377383  1.9131126469709
1.93390796058137  1.93649167310371  1.95703857907809  1.9723082923316  2.03469899493758  2.05182845286832  2.05669638012031
2.06397674405503  2.16564078277077  2.24276614920058  2.28035085019828  2.28910462845192  2.30434372436058  2.30434372436058
2.33666428910959  2.34093998214392  2.36008474424119  2.38537208837531  2.39582971014219  2.41453929352993  2.41867732448957
2.42280828791714  2.43515913237718  2.45356882927706  2.45356882927706  2.47991935352745  2.49599679486974  2.51793566240283
2.52784493195291  2.53574446662119  2.56904651573303  2.58843582110896  2.59036676939772  2.6  2.60192236625154
2.61916017074176  2.62868788561898  2.65518360947035  2.67207784317748  2.67207784317748  2.67768556779918  2.6944387170615
2.6944387170615  2.71661554144123  2.72213151776324  2.7258026340878  2.72946881279124  2.73861278752583  2.7459060435492
2.7495454169735  2.7495454169735  2.7495454169735  2.75317997958724  2.75862284482674  2.76947648482525  2.77488738510232
2.77488738510232  2.77488738510232  2.78208554864871  2.78388218141501  2.80356915377524  2.81602556806574  2.81957443597434
2.82488937836511  2.82665880502051  2.84077454226836  2.84429253066558  2.84780617317963  2.849561369755  2.85832118559129
2.86006992921502  2.86181760425084  2.87402157263998  2.88790581563873  2.88963665535998  2.9  2.93768616431368
2.96816441593117  2.97657521322744  2.97657521322744  3  3.04795013082563  3.09515750810843  3.0967725134404  3.11769145362398
3.17332633052449  3.19217793990247  3.20936130717624  3.27261363439071  3.3391615714128  3.54541958024717  3.55668384875575
3.60832371053374  3.81313519298752  4.05215991787096  4.18449519058154  4.2190046219458  4.29767378938886  4.37035467668243
```