COLD STORAGE

-Report

By,

Vompolu Sai Tanuj

G1 PGP-BABI

Table of Contents

1. Project Objective:

The major objective of the project is to get a detailed analysis on the operations of Cold Storage Company after they have outsourced their storage facility to **TMC Company** in 2018 for a year and give insights. It involves an analysis of the two datasets **Storage Data for the year 2018 and the Storage data for March 2018**. Along with the analysis, we need to give answers for the questions asked as part of the project and infer from the grievances of the customers to check whether **TMC** company had a smooth operation or not.

2. Exploration of the dataset:

Two datasets were used for the project. **Storage Data for the year 2018** in the CSV file, “Cold_Storage_Temp_Data.csv” and **Storage data for March 2018** in the CSV file, “Cold_Storage_Mar2018.csv”. The first dataset can be termed as **Population** set in the terms of Statistics while the second set can be termed as **Sample.** The second dataset is a random sample collected from the large dataset of the first population dataset.

a. Setting up the environment and importing the dataset:

i. Calling out required libraries that will be used in the R code:

For exploration and analysis of data, the following libraries must be installed (if not present in the directory) and invoked at the beginning of the code. The packages are as follows:

1. **Readr –** This is the most important library that must be invoked. It is required to import the dataset which is in CSV file into R environment as dataset

```
install.packages('readr')
library(readr)
```

2. **GGplot2 –** This library is useful to create Plots and Graphs for the variables. It offers a lot customizations compared to the built-in library.

```
install.packages('ggplot2')
library(ggplot2)
```

3. **rpivotTable –** This library is useful for creating Pivot tables and Pivot graphs in the Univariate and Bi-Variate Analysis with ease.

```
install.packages('rpivotTable')
library(rpivotTable)
```

4. **Lattice –** This library is used to easily display multivariate relationships by printing many graphs at once.

```
install.packages('lattice')
library(lattice)
```

ii. Setting up the working directory:

The **working directory** is a default location in a computer or server from which the R console accesses the files called from the console. Setting up of working directory is important for importing and saving files with ease right from the commands in the R Console. The setting up of working directory can be done with **setwd(_)** and to check current directory, we can use the function, **getwd(_).** The code is as below:

```
setwd("C:/R programs great lakes/smdm")
getwd()
```

iii. Importing the dataset and reading the dataset in R console:

There are two datasets to import, **Temp Storage data** and **Temp Storage March data**. They both are situated in two files, **"Cold Storage_Temp_Data.csv"** and **"Cold_Storage_Mar2018_.csv"** respectively. The first dataset is imported into R console and stored under the variable "**Coldstorage"**. The second dataset is imported under the variable name "**StorageMarchData"**. The function used is **read.csv() with header = TRUE.** The code is as follows:

```
### Importing the dataset in consideration
StorageMarchData = read.csv("K2_Cold_Storage_Mar2018.csv",header = TRUE)
Coldstorage = read.csv("K2_Cold_Storage_Temp_Data.csv",header = TRUE)
```

b. Variable Identification:

A basic exploration of the dataset **Coldstorage** and getting an idea about the variables present in them can be done using the following functions.

i. **dim() –** This function gives out the Dimensions of the Data Frame with number of Rows and Columns.

```
> dim(Coldstorage)
[1] 365   4
```

ii. **summary()** – This function prints out a small summary regarding each of the variables. The output contains basic information like, Mean, Median, Quartiles, etc.

```
> summary(Coldstorage)
    Season        Month          Date        Temperature
 Rainy :122   Aug    : 31   Min.   : 1.00   Min.   :1.700
 Summer:120   Dec    : 31   1st Qu.: 8.00   1st Qu.:2.500
 Winter:123   Jan    : 31   Median :16.00   Median :2.900
              Jul    : 31   Mean   :15.72   Mean   :2.963
              Mar    : 31   3rd Qu.:23.00   3rd Qu.:3.300
              May    : 31   Max.   :31.00   Max.   :5.000
              (Other):179
>
```

iii. **str() –** This function prints out the type of variable, number of elements or levels in that particular variable and print out some of its variables.

```
> str(Coldstorage)
'data.frame':   365 obs. of  4 variables:
 $ Season     : Factor w/ 3 levels "Rainy","Summer",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ Month      : Factor w/ 12 levels "Apr","Aug","Dec",..: 5 5 5 5 5 5 5 5 5 5 ...
 $ Date       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Temperature: num  2.4 2.3 2.4 2.8 2.5 2.4 2.8 2.3 2.4 2.8 ...
```

iv. **head() –** This function gives out the top values of the Data Frame when arranged in the default order. An additional argument gives out number of top values to be printed. The default is **6.**

```
> head(Coldstorage)
  Season Month Date Temperature
1 Winter   Jan    1         2.4
2 Winter   Jan    2         2.3
3 Winter   Jan    3         2.4
4 Winter   Jan    4         2.8
5 Winter   Jan    5         2.5
6 Winter   Jan    6         2.4
```

v. **tail() –** This function gives out the bottom values of the Data Frame when arranged in the default order. An additional argument gives out number of bottom values to be printed. The default is **6.**

```
    Season Month Date Temperature
360 Winter   Dec   26         2.7
361 Winter   Dec   27         2.7
362 Winter   Dec   28         2.3
363 Winter   Dec   29         2.6
364 Winter   Dec   30         2.3
365 Winter   Dec   31         2.9
```

vi. **range() –** This function gives out the range of **Maximum** and **Minimum** value for numeric and integer variables.

```
> range(Coldstorage$Date)
[1]  1 31
> range(Coldstorage$Temperature)
[1] 1.7 5.0
```

Inferences:

1) The Data Frame **Coldstorage** has **360 rows** and **4 columns.**
2) The first **(Season)** and second (**Month**) columns of the Data Frame contains **Character** variables with **Factors.**
3) The last two columns, **Date** and **Temperature** are numeric in value.

4) The default order of the dataset is in ascending order starting from the earliest date to the latest date, i.e. from January 1st to December 31st

5) The variable **season** contains 3 factors, namely **Summer, Winter** and **Rainy.**

6) According to the dataset, the year consists of **120 days** of Summer, **123 days** of winter and **122 days** of Rainy.

7) Over the year, the temperature ranged from **1.7 to 5.0**

8) The number of days has minimum value of **1** and maximum value of **31**.

9) The year taken into consideration is not a **leap year** since it has only **365** days.

c. Univariate Analysis:

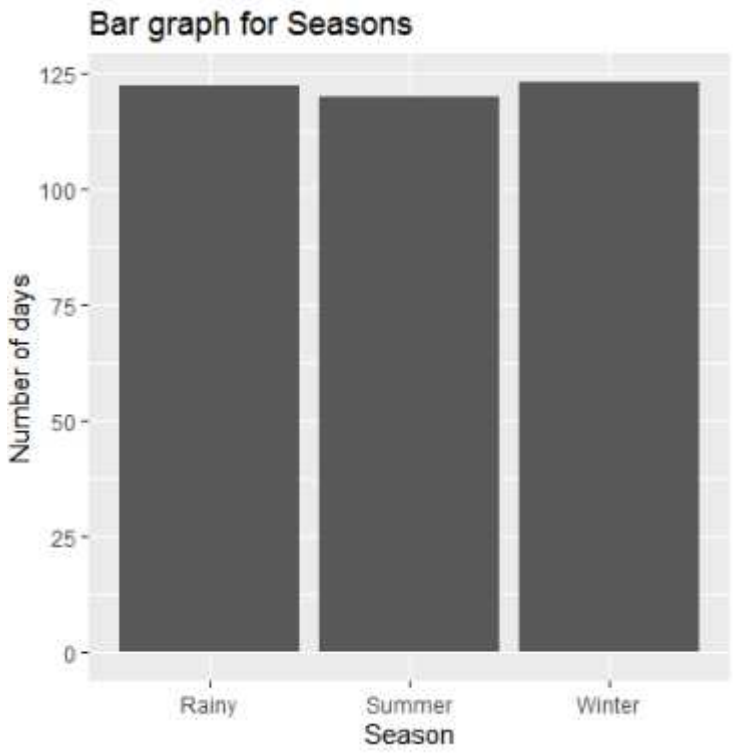
We have to do analysis for three variables, i.e. **Season, Month** and **Temperature.**

Season and Month are Categorical variables with Factors so they can be analysed by using **table ()** and **Bar Graph**.

```
Season.table = table(Coldstorage$Season)
View(Season.table)
```

	Var1	Freq
1	Rainy	122
2	Summer	120
3	Winter	123

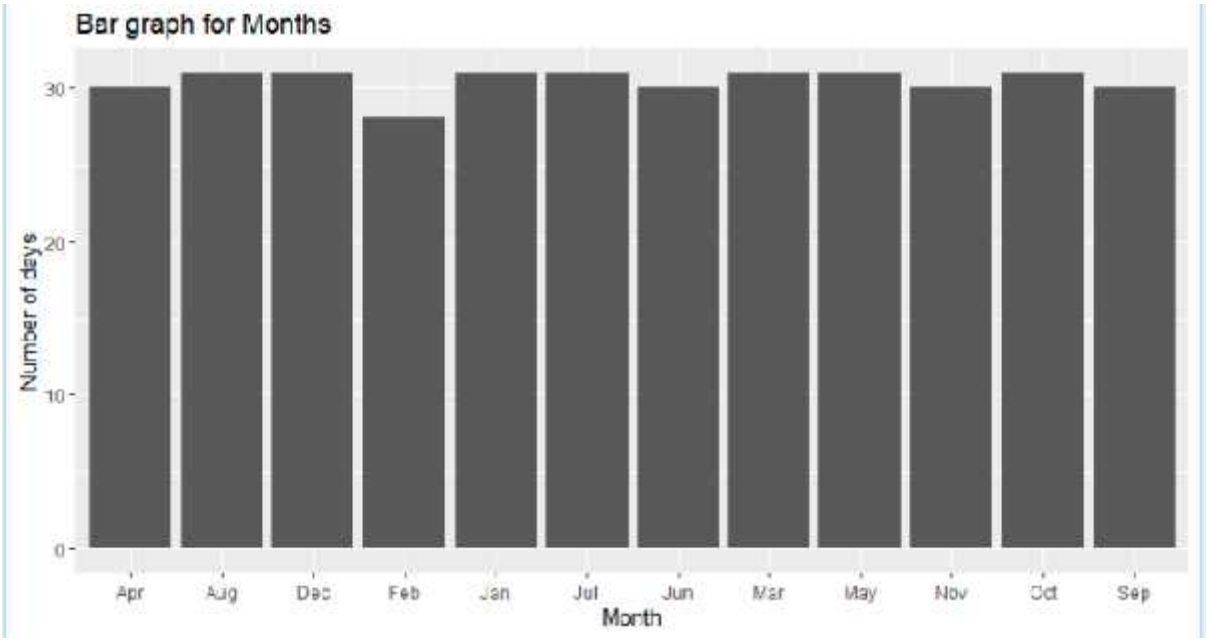
```
qplot(Season,data = Coldstorage,xlab = "Season",ylab = "Number of days",main = 'Bar graph for Seasons")
```




```
Month.table = table(Coldstorage$Month)
View(Month.table)
```

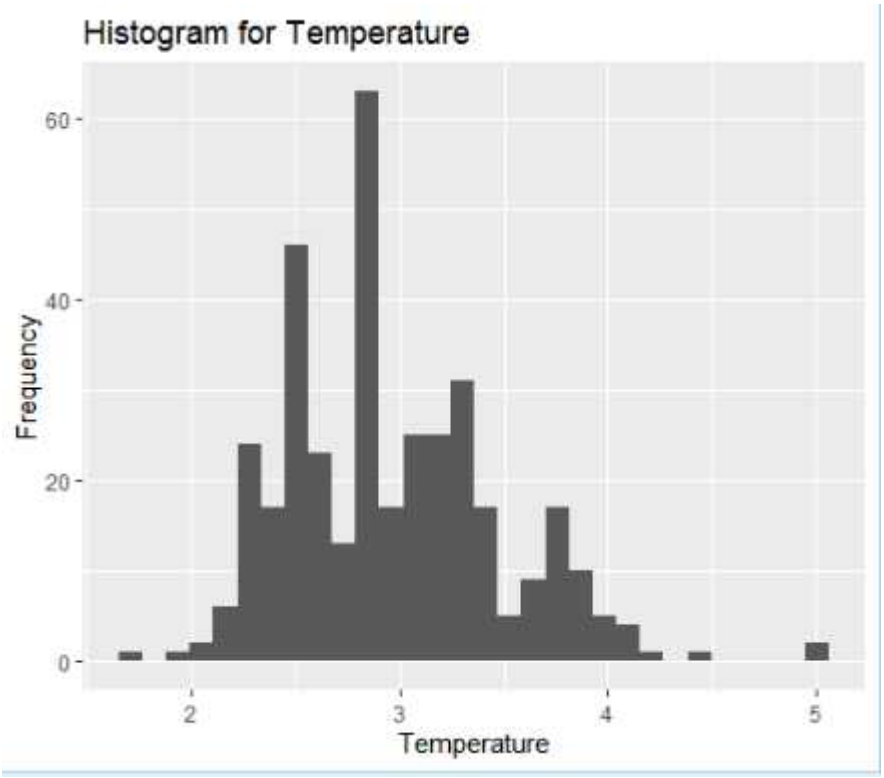
	Var1	Freq
1	Apr	30
2	Aug	31
3	Dec	31
4	Feb	28
5	Jan	31
6	Jul	31
7	Jun	30
8	Mar	31
9	May	31
10	Nov	30
11	Oct	31
12	Sep	30

```
qplot(Month,data = Coldstorage,xlab = "Month",ylab = "Number of days",main = "Ber graph for Months")
```



The Numerical Variable **Temperature** be analysed using the **Histogram** from **ggplot2()** and **summary()** function**.**

```
qplot(Temperature,data = Coldstorage,xlab = "Temperature",ylab = "Frequency",main = "Histogram for Temperature")
```



```
summary(Coldstorage$Temperature)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.700   2.500   2.900   2.963   3.300   5.000 
```

Inferences:

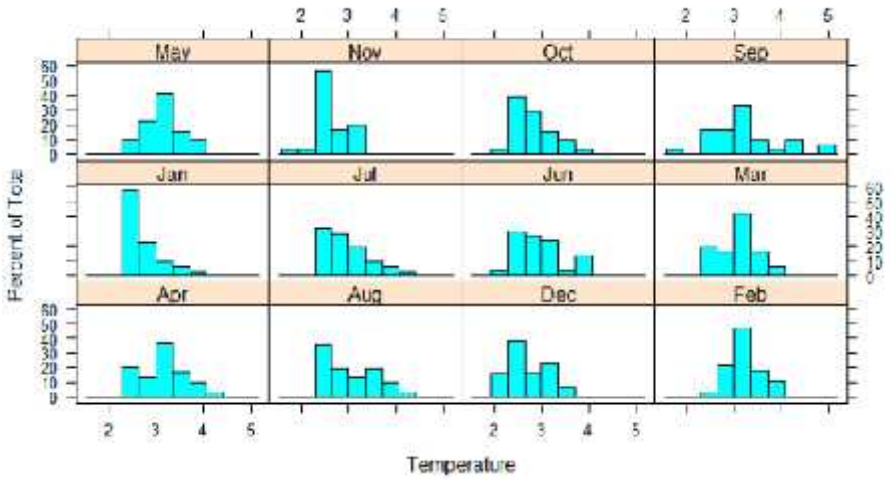
1) The February month has **28** days which means the year taken into consideration in the dataset is **not a leap year.**
2) The three seasons are Summer, Winter and Rainy.
3) According to the dataset, highest number of days are in the **winter** and the least number of days are in the **summer.**
4) The Mean temperature for the whole year is **2.963**.
5) The Median temperature for the whole year is **2.900**
6) The Maximum temperature attained is **5.0**
7) The Minimum temperature attained is **1.7**
8) When a Histogram is plotted for the variable Temperature, the graph is a **Left-Skewed** graph with mean as **2.963**.
9) The temperatures **5.0** and **1.7** are two extreme values which appear away from the cluster of temperatures close to the **Mean.**
10) We can see that the distribution of the values on the graph is a **bell-shaped** graph. It means the values follow a **Normal Distribution.**

d. Bi-Variate Analysis:

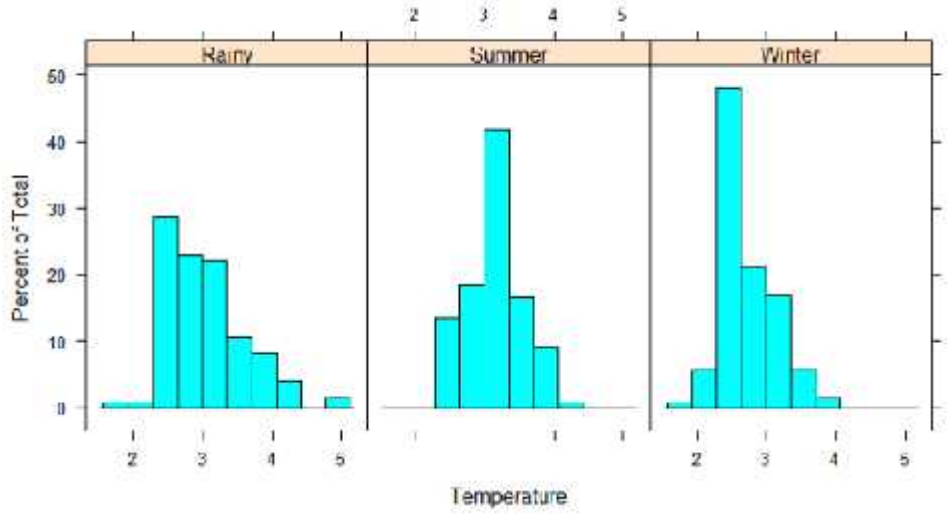
The analysis must be done on the **Temperature** using the two Categorical variables as factors. This can be achieved using

the **Histogram()** function from the **Lattice** library and **qplot(_)** function from ggplot2. The code is as follows.

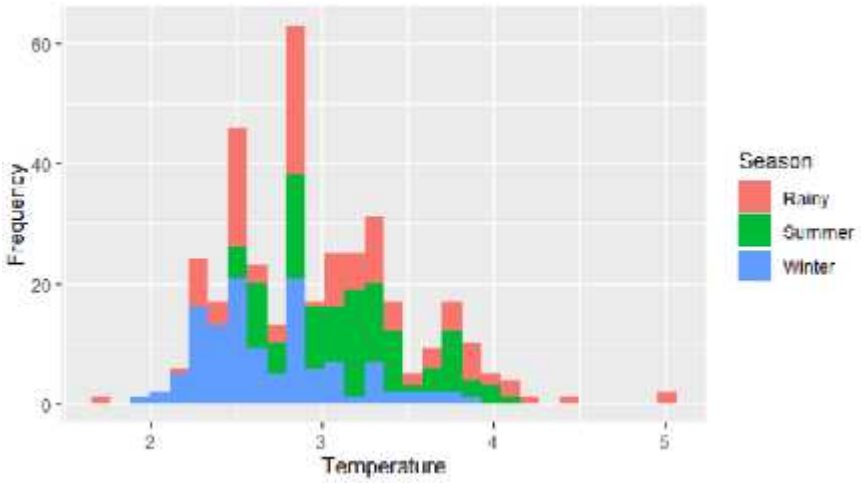
```
histogram(~Temperature|factor(Month),data = Coldstorage)
```

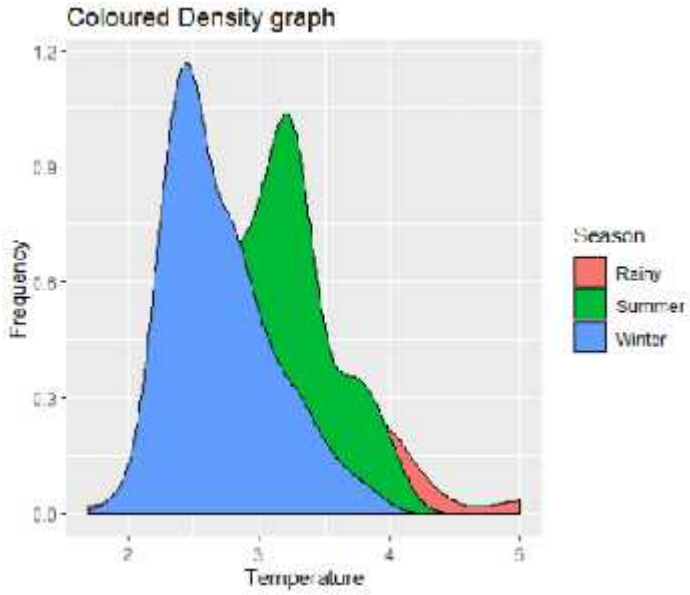


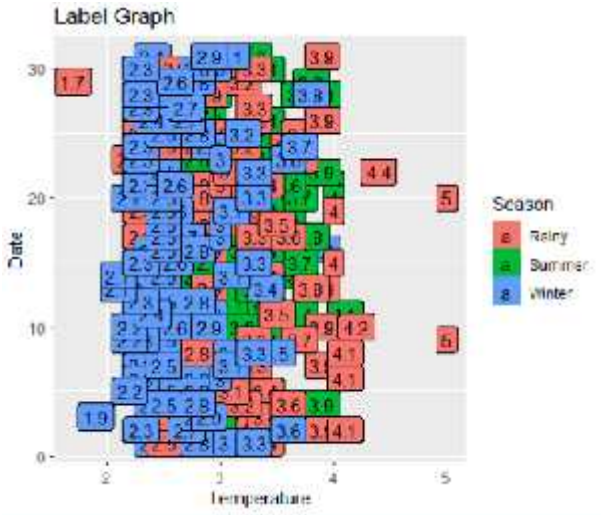

```
histogram(~Temperature|factor(Season),data = Coldstorage)
```

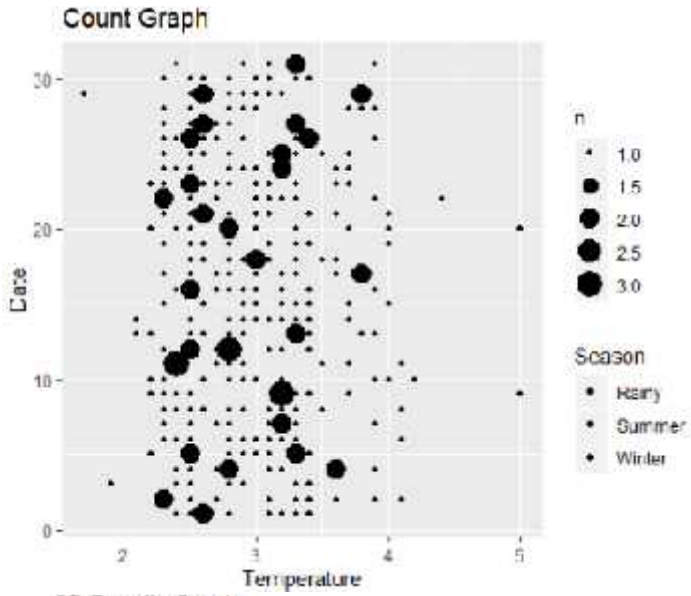


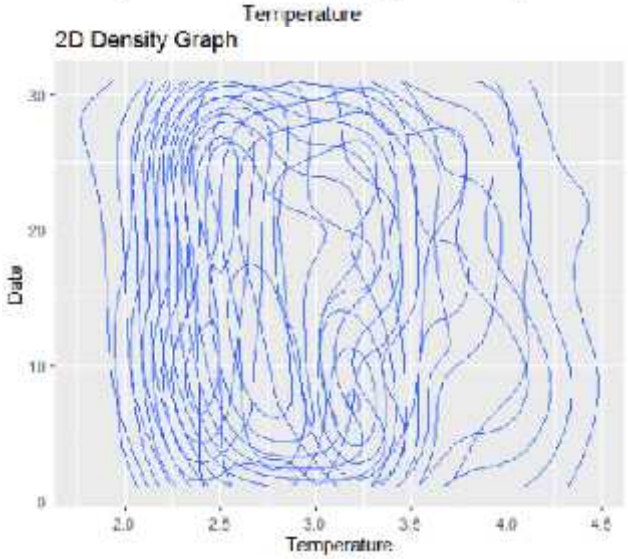

```
> qplot(Temperature,data = Coldstorage, fill = Season, ylab = "Frequency",
title = "Coloured Histogram")
> qplot(Temperature,data = Coldstorage, fill = Season,geom = "density",ylab
= "Frequency",main = "Coloured Density graph"
> qplot(Temperature,Date,data = Coldstorage, fill = Season,geom = "label",l
abel = Temperature,main = "Label Graph")
> qplot(Temperature,Date,data = Coldstorage, fill = Season,geom = "count",m
ain = "Count Graph")
> qplot(Temperature,Date,data = Coldstorage, fill = Season,geom = "density_
2d",main = "2D Density Graph")
> qplot(Temperature,Date,data = Coldstorage, fill = Season,geom = "violin")
```

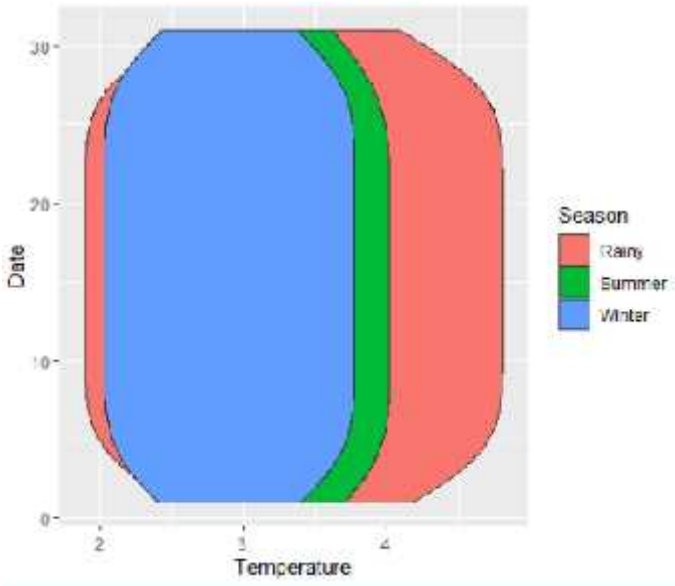

60
40
20
0
Frequency
2
3
4
5
Temperature
Season
Rainy
Summer
Winter


Coloured Density graph
1.2
0.9
0.6
0.3
0.0
Frequency
2
3
4
5
Temperature
Season
Rainy
Summer
Winter

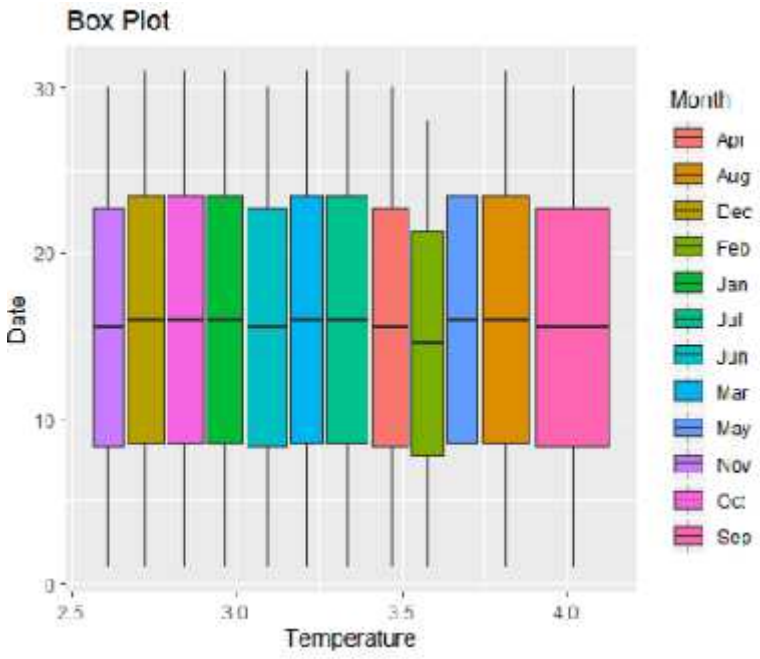

Label Graph
30
20
10
0
Date
2
3
4
5
Temperature
Season
Rainy
Summer
Winter

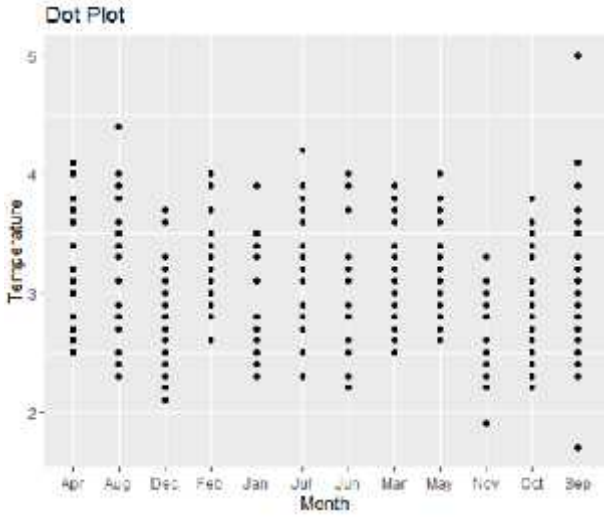

Count Graph
Date
Temperature
n
1.0
1.5
2.0
2.5
3.0
Season
Rainy
Summer
Winter

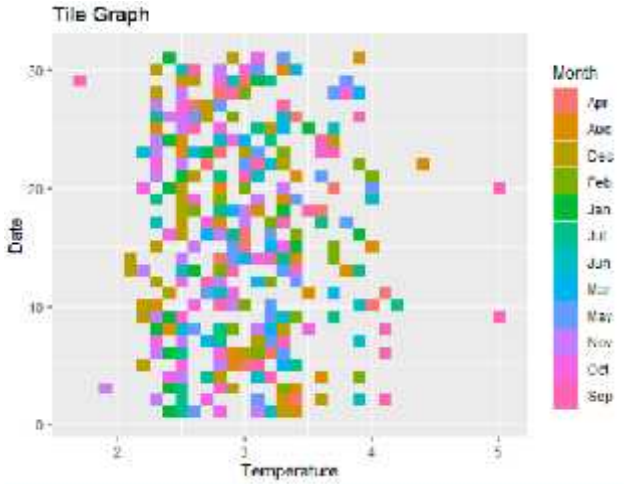

2D Density Graph
Date
Temperature

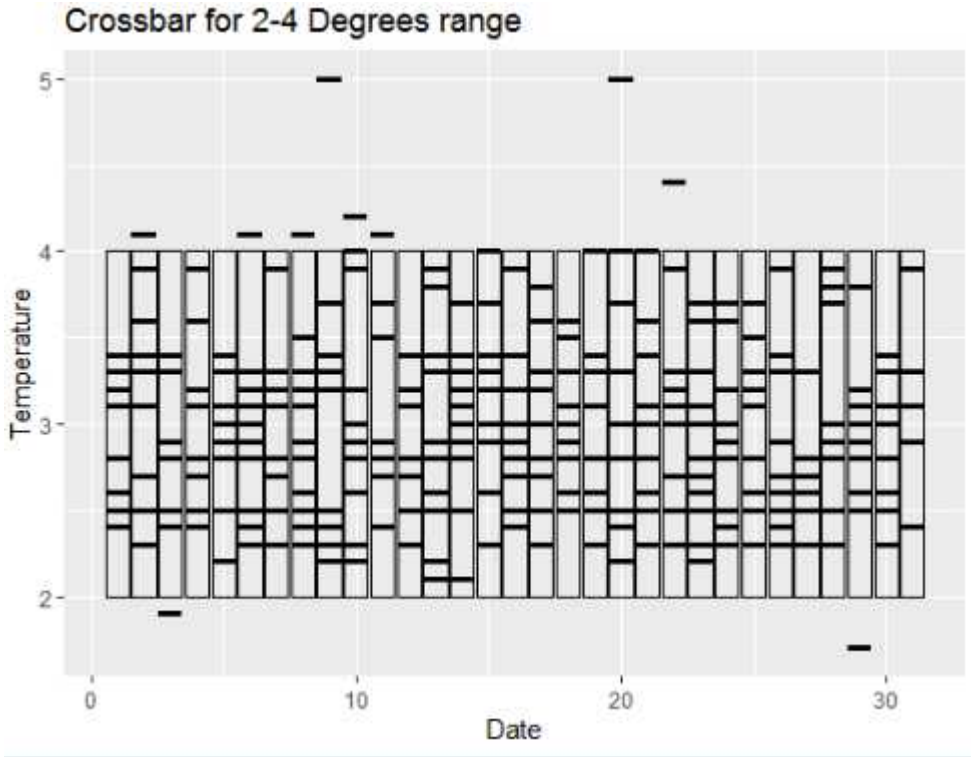

Date
Temperature
Season
Rainy
Summer
Winter

```
> qplot(Temperature,Date,data = Coldstorage,fill = Month, geom = "boxplot",main = "
Box Plot")
> qplot(Month,Temperature,data = Coldstorage,main = "Dot Plot")
> qplot(Temperature,Date,data = Coldstorage,fill = Month, geom = "tile",main = "Til
e Graph")
> qplot(Temperature,Date,data = Coldstorage,fill = Month, geom = "polygon",main = "
Polygon Graph")
> qplot(Date,Temperature,data = Coldstorage, geom = "crossbar",ymin = 2, ymax = 4,m
ain = "Crossbar for 2-4 Degrees range")
```




Dot Plot
Temperature
Month
Apr
Aug
Dec
Feb
Jan
Jul
Jun
Mar
May
Nov
Oct
Sep


Tile Graph
Date
Temperature
Month
Apr
Aug
Dec
Feb
Jan
Jul
Jun
Mar
May
Nov
Oct
Sep


Crossbar for 2-4 Degrees range
Temperature
Date

Inferences:

1) The distribution of the temperatures in the months **May, September, March, February and April,** the distribution is **skewed to the centre**, and i.e. it **is a bell curve.**

2) The distribution of the temperatures in the months other than mentioned above, the distribution is **left skewed.**

3) The distribution for the Rainy Season and Winter season is **left skewed.**

4) The distribution for the Summer season is a bell curve **without any skew** to the either sides.

5) The Winter season has the **highest skew** in its distribution.

6) The Rainy season has the **least skew** in its distribution.

7) The temperatures in the Rainy season are more inconsistent than in any other season.

8) The temperatures in the months of September are more inconsistent than in any other months.

9) The month of **September** contains the extreme most values of the temperature.

e. Outlier Identification:

The outliers can be found out by creating a custom function and applying it to the **Temperature** variable since there is no library for Outlier function.

f. Variable transformation:

Since all the variables and the values are clear, we don't need to transform any variable or apply changes to the dataset. The dataset can be used directly to perform analysis and solve the questions asked in the project.

g. Solutions to the questions asked:

Problem 1

1. ### Find mean cold storage temperature for Summer, Winter and Rainy Season?

Solution:

To find the mean for the Summer, Winter and Rainy season, we need to **slice** the dataset into based on 3 Seasons and Calculate the **Mean** using the **mean(_)** function for each of those sliced datasets.

The code is as below.

```
> ## 1.Finding the mean Temperature for Summer, Winter and Rainy ####
> ### Finding the mean for the summer season
> Summer = Coldstorage[Coldstorage$Season == "Summer",]
> mean(Summer$Temperature)
[1] 3.153333
> ### Finding the mean for the winter season
> Winter = Coldstorage[Coldstorage$Season == "Winter",]
> mean(Winter$Temperature)
[1] 2.700813
> ### Finding the mean for the Rainy season
> Rainy = Coldstorage[Coldstorage$Season == "Rainy",]
> mean(Rainy$Temperature)
[1] 3.039344
```

Inferences:

From the means of the three Seasons, we can see that the Mean temperature for all the seasons were maintained in the

range of 2 to 4 Degrees. The Mean temperature in the **summer** is the **highest** while the Mean temperature in the winter is the **lowest.**

The Mean temperature in the Winter is 2.700813, Summer is 3.153333 and Rainy is 3.039344

2. Find overall mean for the full year.

Solution:

The mean for the year can be calculated by applying the **mean()** function to the Temperature variable. The code is as below:

```
> ## 2.Finding the mean Temperature for the whole year #######
> Mean = mean(Coldstorage$Temperature)
> Mean
[1] 2.96274
```

Inferences:

From the above output, we can infer that the Mean temperature maintained for the whole year is really close to the Mean temperature of the Rainy Season (3.039344).

The mean temperature for the whole year is 2.96274.

3. Find Standard Deviation for the full year.

Solution:

The **Standard Deviation** can be found out for the temperature of the whole year using the function **sd(_)**. This code is as follows:

```
> ## 3.Finding the Standard Deviation for the whole year ######
> SD = sd(Coldstorage$Temperature)
> SD
[1] 0.508589
```

Inferences:

From the Standard deviation, we can infer that even though the temperature values deviate by 0.508589 from the **mean,** they still stay in the range of **2 to 4 degrees range.**

The standard deviation for the temperature for the whole year is 0.508589.

4. Assume Normal distribution, what is the probability of temperature having fallen below 2 degrees C?

Solution:

To find the probability for this normal distribution, we need to find the probability distribution for the value according to the condition given. For finding the probability distribution for below 2 Degrees, we find the value of the cumulative probability for that value by using the **pnorm(_)** function with the arguments, **Value, Mean of the whole year temperature and Standard Deviation of the whole year.**

The code is as below:

```
> ##4. Finding the probability of the temperature being below 2 Degrees ######
> Q4 = pnorm(2,Mean,SD)
> Q4
[1] 0.02918146
```

Inferences:

As seen in the previous outputs, we can see that only few values have been below 2 Degrees. This has been made even clear with the probability distribution.

The probability distribution for the temperature being less than 2 Degrees 0.02918146

5. Assume Normal distribution, what is the probability of temperature having gone above 4 degrees C?

Solution:

To find the probability for this normal distribution, we need to find the probability distribution for the value according to the condition given. For finding the probability distribution for above 4 degrees, we find the **cumulative probability distribution** for the value and find the **inverse of that probability** since we need **right side** probability distribution. For that we find the value using the **pnorm()** function and find the inverse by subtracting the answer by **1.** The code is as follows:

```
> ##5. Finding the probability of the temperature being above 4 Degrees ######
> Q5 = 1-pnorm(4,Mean,SD)
> Q5
[1] 0.02070077
```

Inferences:

As seen in the previous outputs, we can see that only few values have been above 4 Degrees. This has been made even clear with the probability distribution as around 2% is very less.

The probability distribution for the temperature being more than 4 Degrees 0.02070077

6. What will be the penalty for the AMC Company?

Solution:

The penalty rates will be the following percentages of the Annual Maintenance Cost:

i. No Penalty – If the probability of temperatures crossing the 2-4 degrees range is less than 2.5%.

ii. 10% Penalty – If the probability of temperatures crossing the 2-4 degrees range is between 2% and 5%

iii. 25% Penalty – If the probability of temperatures crossing the 2-4 degrees range is more than 5%

So the to calculate the probability of temperatures crossing the range of 2-4 degrees will be the sum of the **probability of temperatures being below 2 Degrees and the probability of temperatures being above 4 Degrees.**

Since we have calculated both the probabilities, we just have to add them, and judge the penalty accordingly. This judgement of the penalty was done by using **if and if else loops.** The code is as follows:

```
> ###Total probability of temperature crossing the given thresholds(2-4)
> Q4
[1] 0.02918146
> Q5
[1] 0.02070077
> prob = Q4 + Q5
> prob = prob * 100
> prob
[1] 4.988223
> if(prob > 2 && prob < 5) {
+     print("Penalty is 10% of AMC")
+ } else if (prob > 5) {
+     print("Penalty is 25% of AMC")
+ } else {
+     print("No need to pay penalty")
+ }
[1] "Penalty is 10% of AMC"
```

Inferences:

Even though most of the Temperature values lied between 2-4 Degrees, due to the some extreme values such as **1.7, 5.0**, etc., the probability of crossing the temperature range increased.

The penalty incurred will 10% of AMC since the probability of the temperature values crossing the range of 2-4 degrees is 4.988223%.

Problem 2

Assumptions:

In the above problem, we must determine whether the cold storage plant was able to maintain the temperature below **3.9 Degrees** so that the dairy products don't go sour. But since there have been complaints from the Clients about the Dairy products getting sour, we got the need to begin our analysis in the first place. So in the according to the above scenario, we must assume our **Null**

Hypothesis (H_0) will be that *mean of the temperature for the whole year will be equal to or below 3.9 Degrees.* Accordingly our **Alternative Hypothesis (H_1)** will be the opposite of the Null Hypothesis which is ***the mean temperature for the whole year is more than 3.9.***

$H_0 \rightarrow \mu = 3.9$

$H_1 \rightarrow \mu > 3.9$

We must import another dataset, **sample** data given in the CSV file,

"Cold_Storage_Mar2018_.csv". The **population** data is given in the file, **"Cold_Storage_Temp_Data.csv".**

We must perform **Hypothesis Testing** to do a proper analysis.

1. State the Hypothesis, do the calculation using z test.

Solution:

For performing Z-Test, we first need to calculate the **Z-Stat** score. It is given by the formula,

$$Z.S = \frac{x - \mu}{\frac{S}{\sqrt{n}}}$$

Where, x = Sample Mean

μ = Population Mean

S = Standard Deviation of the Population

n = Sample Size.

After Finding Z-Stat, we find the value of the **Z-Critical** from the given **Alpha Value** (0.10). This can be done using the **qnorm(_)** function.

*Note: The argument in qnorm(_) must be 1- α since the this is a right-tailed test and we need the Z-Critical value for the probability distribution for 0.10 from the **right tail.***

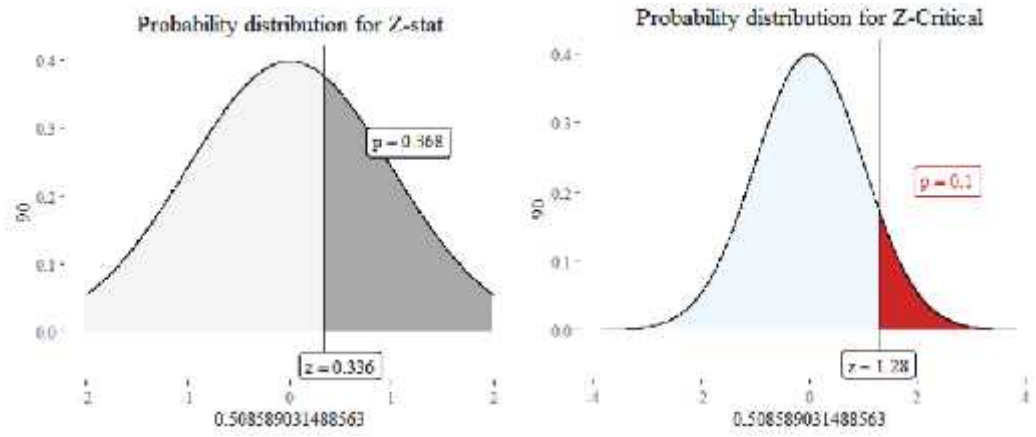
```
> ### Finding the requiered parameters for the Z test ####
> Mean.sample = mean(StorageMarchData$Temperature)
> Mean.sample
[1] 3.974286
> Mean.population = mean(Coldstorage$Temperature) ##Taken from the total da
taset where the mean of all the temperatures for the whole year was calcula
ted
> Mean.population
[1] 2.96274
> SD.population = sd(Coldstorage$Temperature)
> SD.population
[1] 0.508589
> Sample.size = nrow(StorageMarchData)
> Sample.size
[1] 35
> Z.value.1 = (Mean.sample - Mean.population)
> Z.value.2 = (SD.population)/(Sample.size)^0.5
> Z.stat = (Mean.sample - Mean.population)/(SD.population)/(Sample.size)^0.
5
> Z.stat
[1] 0.3361899
> ##Finding the critical value
> Z.critical = qnorm(0.90)
> Z.critical
[1] 1.281552
> ##Decision of rejection or accpetance of the null Hypothesis
> if(Z.stat > Z.critical){
+     "The Null hypothesis is rejected"
+ }else {
+     "The null hypothesis is accepted"
+ }
[1] "The null hypothesis is accepted"
```

Relevant Plots:

To get the Z.test graphs, we need to install and import an additional library called "**nhstplot**"

```
install.packages('nhstplot')
library(nhstplot)
```

```
zplot  = plotztest(Z.stat,Mean.population,SD.population,90,tails = "one",th
eme = "blackandwhite")
zcplot = plotztest(Z.critical,Mean.population,SD.population,90,tails = "one
")
zplot
zcplot
```



Inferences:

According to the Z-Test conducted, we saw that Z.Stat (0.3361899) is greater than Z-Critical (1.281552). Since **this is a right-tailed test** if our Z-Stat is less than the Z-Critical value, we **fail to reject (accept)** the **Null Hypothesis.** And we have to reject the **Alternative Hypothesis**.

From this, it is safe to say that the mean temperature was maintained at 3.9 degrees or below 3.9 degrees.

From the Z-test performed, we concluded that ***Mean temperature for the whole year was maintained below or equal to 3.9 degrees.***

2. State the Hypothesis, do the calculation using t-test.

Solution:

For performing T-Test, we first need to calculate the **T-Stat** score. It is given by the formula,

$$T.S = \frac{x - \mu}{\frac{s}{\sqrt{n}}}$$

Where, x = Sample Mean

μ = Population Mean

s = Standard Deviation of the Sample

n = Sample Size.

From the Calculated T-Stat value, we need to find the corresponding **P-Value** using the function **pt (_)** with T.Stat and **Degrees of Freedom** as arguments. Degrees of Freedom for a single sample is DF = Sample Size – 1.

From that **P-Value,** we compare it to the Alpha value given to us and accordingly reject or accept the null Hypothesis. The code is as below,

```
> ### T. test ######
> Mean.sample = mean(StorageMarchData$Temperature)
> Mean.sample
[1] 3.974286
> Mean.population = mean(Coldstorage$Temperature) ##Taken from the total da
taset where the mean of all the temperatures for the whole year was calcula
ted
> Mean.population
[1] 2.96274
> SD.sample = sd(StorageMarchData$Temperature)
> SD.sample
```

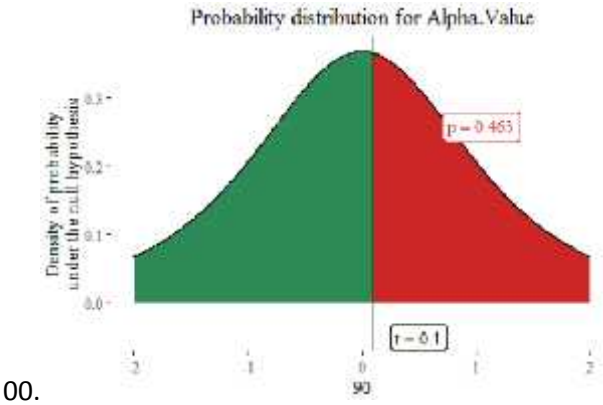
```
[1] 0.159674
> Sample.size = nrow(StorageMarchData)
> Sample.size
[1] 35
> Degrees.of.freedom = Sample.size - 1
> Degrees.of.freedom
[1] 34
> T.stat = (Mean.sample - Mean.population)/(SD.sample)/(Sample.size)^0.5
> T.stat
[1] 1.070822
> P.value = 1-(pt(T.stat,Degrees.of.freedom))
> P.value
[1] 0.1458962

> ##Decision of rejection or accpetance of the null Hypothesis ######

> Alpha.value = 0.10
> ##Decision of rejection or accpetance of the null Hypothesis
> if(P.value < Alpha.value ){
+   "The Null hypothesis is rejected"
+ }else {
+   "The null hypothesis is accepted"
+ }
[1] "The null hypothesis is accepted"
```

Relevant Plots:

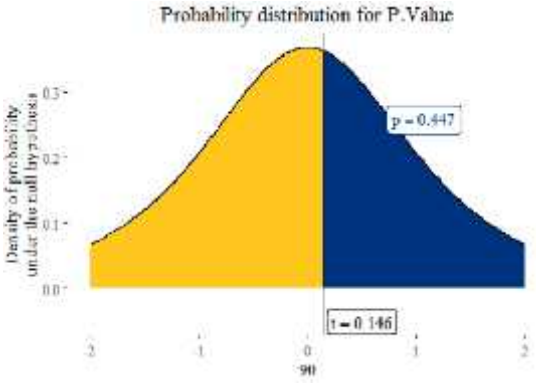
To get the T.test graphs, we need to install and import an additional library called "**nhstplot**"

```
install.packages('nhstplot')
library(nhstplot)
```

```
> tplot  = plotttest(P.value,Mean.population,SD.population,90,tails = "one"
,theme = "goldandblue",title = "Probability distribution for P.Value")
> taplot = plotttest(Alpha.value,Mean.population,SD.population,90,tails = "
one",theme = "greenandred",title = "Probability distribution for Alpha.Valu
e")
> tplot
> taplot
```



00.



Inferences:

According to the T-Test conducted, we saw that P-Value (0.1458962) is greater than Alpha Value (0.10). When the P-Value is **less than** Alpha Value, we must **reject** the **Null Hypothesis.** But here, P-Value is more than Alpha Value. Therefore we **fail to reject (accept) the Null Hypothesis.**

From the T-test performed, we concluded ***again*** *that* ***Mean temperature for the whole year was maintained below or equal to 3.9 degrees.***

3. Give your inference after doing both the tests.

Inference:

In both the **Z and T tests,** we **had to accept** the Null Hypothesis and **reject** the **Alternative Hypothesis** with a significance level of 10%. Other than the tests, this inference could roughly be made from the **Bi-Variate Analysis** where it was inferred that temperature was mostly maintained between 2-4 Degrees range. This tells us that the supervisor managed to keep the Temperature in the Cold Storage **equal to or less than 3.9 Degrees.** We say equal to or less than because we have only **failed to reject** the Null Hypothesis but not actually **accept** the Null Hypothesis. So in the current scenario, the cause of the Clients complaints for the food items getting Sour is **not the Cold Storage**. The cause for the complaints is from the **procurement side** from where the food items are placed into the Cold Storage**.**

h. Conclusion:

The dataset Cold Storage was compiled with the idea to get a detailed on the analysis of the operations that are going on in the Cold Storage. The cold storage's operations included storage of Dairy products like Whole or Skimmed Milk, Sweet Cream, Flavoured Milk Drinks, Whole or Skimmed Milk, Sweet Cream, Flavoured Milk Drinks, etc. Their major aim is to keep the Products fresh when they reach the

clients and this is achieved by smooth operation of Cold Storage to ensure the right temperatures. . From the analysis, the following inferences were made.

1) The Maximum temperature (5.0) and the minimum temperature (1.7) were recorded in the month of **September** and in the season of **Rainy**. And also there was wide spread of temperatures during this period which tells that the operations were not consistent and had disruptions. This may be due to the improper maintenance of the temperature in the cold storage. So the operations could be improved better in the **Rainy season** especially at the time of **September.**

2) To better improve the operations, there must be **Monthly Maintenance Penalty** so that performance of each month is improved. If Annual is put such as now, the operations for few months could differ from the standards but the result wouldn't affect the whole year's operations.

3) We proved from **Hypothesis Testing** that we have been getting complaints about the food getting sour. This was due to the problem at the procurement side of the operations from the where the food got into Cold Storage. If there was a way to get the Dairy Products checked for quality before ensuring their entry into **Cold Storage,** it is easy to make sure that no food ever gets sour in the **Cold Storage.**

4) The operations in the **Cold Storage** should also be flexible enough to maintain its temperatures to support all kinds of products that can are procured. This ensures that

products maintain their quality no matter how the procurement is.

5) The **Cold Storage** equipment should be checked from time to time because there are inconsistencies in the performance from season to season which suggests that the climate outside might be affecting the equipment in the Cold Storage.

j. Appendix - A(Source Code):

Source code for problem 1

```
###  LIBRARIES ###
install.packages('ggplot2')
library(ggplot2)
install.packages('lattice')
library(lattice)
install.packages('readr')
library(readr)
install.packages('rpivotTable')
library(rpivotTable)

## Setting up Working Directory ####
 > setwd("C:/R programs great lakes/smdm")
 > getwd()
 [1] "C:/R programs great lakes/smdm"

> ## IMPORTING THE REQUIERED DATASET #####
>
> Coldstorage = read.csv("K2_Cold_Storage_Temp_Data.csv",header = TRUE)
> ## PERFORMING PRELIMINARY ANALYSIS ON THE DATASET ##
> summary(Coldstorage)
    Season        Month          Date        Temperature
 Rainy :122   Aug    : 31   Min.   : 1.00   Min.   :1.700
 Summer:120   Dec    : 31   1st Qu.: 8.00   1st Qu.:2.500
 Winter:123   Jan    : 31   Median :16.00   Median :2.900
              Jul    : 31   Mean   :15.72   Mean   :2.963
              Mar    : 31   3rd Qu.:23.00   3rd Qu.:3.300
              May    : 31   Max.   :31.00   Max.   :5.000
```

```
             (Other):179
> str(Coldstorage)
'data.frame':       365 obs. of  4 variables:
 $ Season     : Factor w/ 3 levels "Rainy","Summer",..: 3 3 3 3 3 3 3 3
3 3 ...
 $ Month      : Factor w/ 12 levels "Apr","Aug","Dec",..: 5 5 5 5 5 5 5
5 5 5 ...
 $ Date       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Temperature: num  2.4 2.3 2.4 2.8 2.5 2.4 2.8 2.3 2.4 2.8 ...
> class(Coldstorage$Temperature)
[1] "numeric"
> nrow(Coldstorage)
[1] 365
> ### Univariate Analysis ###
> Season.table = table(Coldstorage$Season)
> Season.table

 Rainy Summer Winter
   122    120    123
qplot(Season,data = Coldstorage,xlab = "Season",ylab = "Number of days",
main = "Bar graph for Seasons")

> Month.table = table(Coldstorage$Month)
Month.table

>qplot(Month,data = Coldstorage,xlab = "Month",ylab = "Number of days",m
ain = "Bar graph for Months")
> qplot(Temperature,data = Coldstorage,ylab = "Frequency",main = "Histog
ram for Temperature")
> qplot(Temperature,data = Coldstorage,geom = "density",ylab = "Frequenc
y",main = "Histogram for Temperature")

> ### Bivariate Analysis ####
> histogram(~Temperature|factor(Month),data = Coldstorage)
> histogram(~Temperature|factor(Season),data = Coldstorage)

> qplot(Temperature,data = Coldstorage, fill = Season, ylab = "Frequency
", title = "Coloured Histogram")
> qplot(Temperature,data = Coldstorage, fill = Season,geom = "density",y
lab = "Frequency",main = "Coloured Density graph"
> qplot(Temperature,Date,data = Coldstorage, fill = Season,geom = "label
",label = Temperature,main = "Label Graph")
> qplot(Temperature,Date,data = Coldstorage, fill = Season,geom = "count
",main = "Count Graph")
> qplot(Temperature,Date,data = Coldstorage, fill = Season,geom = "densi
ty_2d",main = "2D Density Graph")
> qplot(Temperature,Date,data = Coldstorage, fill = Season,geom = "violi
n")


> qplot(Temperature,Date,data = Coldstorage,fill = Month, geom = "boxplo
t",main = "Box Plot")
> qplot(Month,Temperature,data = Coldstorage,main = "Dot Plot")
> qplot(Temperature,Date,data = Coldstorage,fill = Month, geom = "tile",
main = "Tile Graph")
> qplot(Temperature,Date,data = Coldstorage,fill = Month, geom = "polygo
n",main = "Polygon Graph")
> qplot(Date,Temperature,data = Coldstorage, geom = "crossbar",ymin = 2,
ymax = 4,main = "Crossbar for 2-4 Degrees range")
```

```
> ## 1.Finding the mean Temperature for Summer, Winter and Rainy ####
> ### Finding the mean for the summer season
> Summer = Coldstorage[Coldstorage$Season == "Summer",]
> mean(Summer$Temperature)
[1] 3.153333
> ### Finding the mean for the winter season
> Winter = Coldstorage[Coldstorage$Season == "Winter",]
> mean(Winter$Temperature)
[1] 2.700813
> ### Finding the mean for the Rainy season
> Rainy = Coldstorage[Coldstorage$Season == "Rainy",]
> mean(Rainy$Temperature)
[1] 3.039344
>
>
>
> ## 2.Finding the mean Temperature for the whole year ########
> Mean = mean(Coldstorage$Temperature)
> Mean
[1] 2.96274
> median(Coldstorage$Temperature)
[1] 2.9
>
>
> ## 3.Finding the Standard Deviation for the whole year ######
> SD = sd(Coldstorage$Temperature)
> SD
[1] 0.508589
>
>
> ##4. Finding the probability of the temperature being below 2 Degrees
######
> Q4 = pnorm(2,Mean,SD)
> Q4
[1] 0.02918146
>
> ##5. Finding the probability of the temperature being above 4 Degrees
######
> Q5 = 1-pnorm(4,Mean,SD)
> Q5
[1] 0.02070077
>
>
>
> ###Total probability of temperature crossing the given thresholds(2-4)
> Q4
[1] 0.02918146
> Q5
[1] 0.02070077
> prob = Q4 + Q5
> prob = prob * 100
> prob
[1] 4.988223
> if(prob > 2 && prob < 5) {
+     print("Penalty is 10% of AMC")
+ } else if (prob > 5) {
+     print("Penalty is 25% of AMC")
+ } else {
+     print("No need to pay penalty")
+ }
```

```
[1] "Penalty is 10% of AMC"
```

Source code for problem 2

```
### LIBRARIES
install.packages('readr')
library(readr)
install.packages('nhstplot')
library(nhstplot)

### setting up Working Directory
> setwd("C:/R programs great lakes/smdm")
> getwd()
[1] "C:/R programs great lakes/smdm"


> ### Importing the dataset in consideration and the population dataset
> StorageMarchData = read.csv("K2_Cold_Storage_Mar2018.csv",header = TRUE)
> Coldstorage = read.csv("K2_Cold_Storage_Temp_Data.csv",header = TRUE)
> ### Performing prelimnary tests on dataset
> summary(StorageMarchData)
    Season   Month         Date       Temperature
 Summer:35   Feb:18   Min.   : 1.0   Min.   :3.800
             Mar:17   1st Qu.: 9.5   1st Qu.:3.900
                      Median :14.0   Median :3.900
                      Mean   :14.4   Mean   :3.974
                      3rd Qu.:19.5   3rd Qu.:4.100
                      Max.   :28.0   Max.   :4.600
> View(StorageMarchData)
> dim(StorageMarchData)
[1] 35  4
> str(StorageMarchData)
'data.frame':  35 obs. of  4 variables:
 $ Season     : Factor w/ 1 level "Summer": 1 1 1 1 1 1 1 1 1 1 ...
 $ Month      : Factor w/ 2 levels "Feb","Mar": 1 1 1 1 1 1 1 1 1 1 ...
 $ Date       : int  11 12 13 14 15 16 17 18 19 20 ...
 $ Temperature: num  4 3.9 3.9 4 3.8 4 4.1 4 3.8 3.9 ...
> attach(StorageMarchData)
The following objects are masked from StorageMarchData (pos = 5):

    Date, Month, Season, Temperature

>
>
> ### Finding the requiered parameters for the Z test ####
> Mean.sample = mean(StorageMarchData$Temperature)
> Mean.sample
[1] 3.974286
> Mean.population = mean(Coldstorage$Temperature) ##Taken from the total da
taset where the mean of all the temperatures for the whole year was calcula
ted
```

```
> Mean.population
[1] 2.96274
> SD.population = sd(Coldstorage$Temperature)
> SD.population
[1] 0.508589
> Sample.size = nrow(StorageMarchData)
> Sample.size
[1] 35
> Z.value.1 = (Mean.sample - Mean.population)
> Z.value.2 = (SD.population)/(Sample.size)^0.5
> Z.stat = (Mean.sample - Mean.population)/(SD.population)/(Sample.size)^0.
5
> Z.stat
[1] 0.3361899
> ##Finding the critical value
> Z.critical = qnorm(0.90)
> Z.critical
[1] 1.281552
> ##Decision of rejection or accpetance of the null Hypothesis
> if(Z.stat > Z.critical){
+     "The Null hypothesis is rejected"
+ }else {
+     "The null hypothesis is accepted"
+ }
[1] "The null hypothesis is accepted"
> ### Plotting the graphs for Z-Test ###

> zplot  = plotztest(Z.stat,Mean.population,SD.population,90,tails = "one",
theme = "blackandwhite",title = "Probability distribution for Z-stat")
> zcplot = plotztest(Z.critical,Mean.population,SD.population,90,tails = "o
ne",title = "Probability distribution for Z-Critical")
> zplot
> zcplot
>
>
> ### T. test ######
> Mean.sample = mean(StorageMarchData$Temperature)
> Mean.sample
[1] 3.974286
> Mean.population = mean(Coldstorage$Temperature) ##Taken from the total da
taset where the mean of all the temperatures for the whole year was calcula
ted
> Mean.population
[1] 2.96274
> SD.sample = sd(StorageMarchData$Temperature)
> SD.sample
[1] 0.159674
> Sample.size = nrow(StorageMarchData)
> Sample.size
[1] 35
> Degrees.of.freedom = Sample.size - 1
> Degrees.of.freedom
[1] 34
> T.stat = (Mean.sample - Mean.population)/(SD.sample)/(Sample.size)^0.5
> T.stat
[1] 1.070822
> P.value = 1-(pt(T.stat,Degrees.of.freedom))
> P.value
[1] 0.1458962
> Alpha.value = 0.10
> ##Decision of rejection or accpetance of the null Hypothesis
```

```
> if(P.value < Alpha.value ){
+     "The Null hypothesis is rejected"
+ }else {
+     "The null hypothesis is accepted"
+ }
[1] "The null hypothesis is accepted"
>

> ### Plotting the graphs for T-Tests ###
> tpplot  = plotttest(P.value,Mean.population,SD.population,90,tails = "one
",theme = "goldandblue",title = "Probability distribution for P.Value")
> taplot = plotttest(Alpha.value,Mean.population,SD.population,90,tails = "
one",theme = "greenandred",title = "Probability distribution for Alpha.Valu
e")
> tpplot
> taplot
```