Output :-

```
> a ← 23
  b ← 340

addition : 363

Subtraction : -317

Multiplication : 7820

division : 14.78261
```

```
> class(a)
"numeric"
> class(c)
"character"
> class(x)
"integer"
> class(y)
"complex"
> sqrt(b)
18.43909
```

12/24

R - Arithmetic operators & Data Types

Examples of R - Arithmetic operators & Data Types

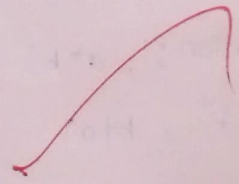
```
# operators
a ← 23

b ← 340

c ← "manu"

d ← "sitha has money"

x ← 1000L

y ← 3+4i

print("addition :", a+b)

print("subtraction :", a-b)

print("multiplication :", a*b)

print("division :", b/a)
```

```
# Data types

class(a)

class(b)

class(x)

class(y)

sqrt(b)
```

output :-

```
> if (a>b) {
+   print("a is greatu numbu")
+ } else
+   print("b is greatu numbu")
```

"b is greatu numbu"



Multiplication = 42

Control Statements in R

if-else :-

```
a ← 23
b ← 340

if (a>b) {
  print("a is greatu numbu")
} else
  print("b is greatu numbu")
```

Switch :-

```
val1 = 6
val2 = 7
val3 = "s"
result = switch (
   val3,
   "a" = cat("Addition=", val1 + val2),
   "d" = cat("Subtraction", val1 - val2),
   "d" = cat("Division=", val1 / val2),
   "m" = cat("multiplication", val1 * val2),
   "p" = cat("powu = ", val1 ^ val2).
)
```

O/P

[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
[1] 7
[1] 8

[1] "manu"
[2] "urishi"
[3] "anu"

While loop:-

```
a ← 9
i ← 1
while (i<a) {
  print(i)
  i ← i+1
}
```

for loop :-

```
names ← list ("manu", "urishi", "anu")
for (x in names) {
   print(x)
}
```

Result :-

The program has been executed successfully.

12/12/24

15/2/24 Task 2:-

Acquiring & plotting Data

Steps (Algorithm)

1) Create a new folder & download datasets

2) Open R studio & install any necessary packages

3) Open the folder in R studio

4) Acquire your data using functions like (read_csv2)

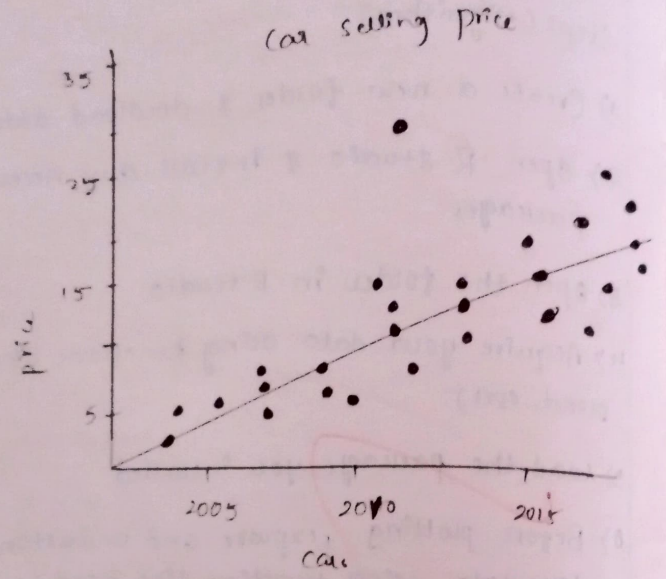
5) Load the package you installed

6) Before plotting, explore and understand your data using function like head(), summary() and str()

7) Create plots using ggplot2 or other plotting libraries based on your requirement.

8) Select all lines and run the program

9) End.



program :-

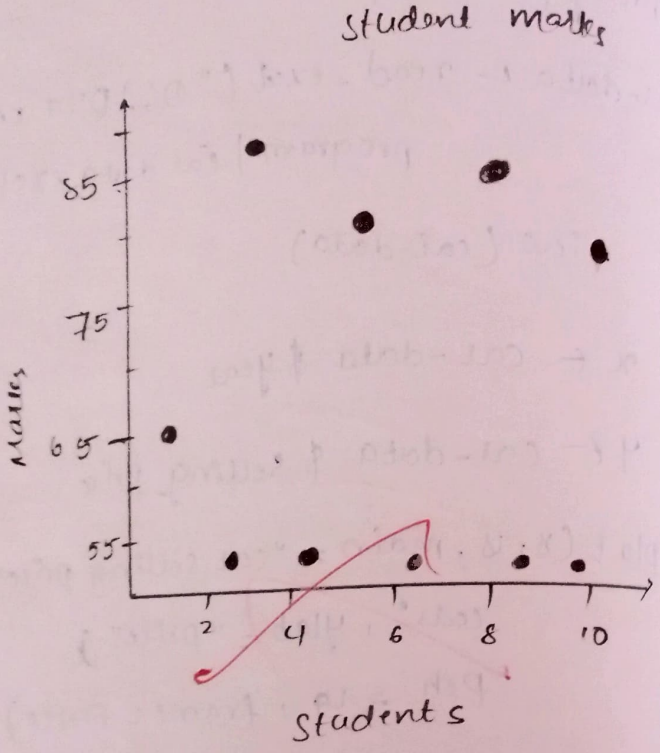
```
install.package("readxl")
library(readxl)
car_data <- read_excel("D:/IDUT LAB/car program/car data.xls/x")
view(car_data)
x <- car_data $ year
y <- car_data $ selling_price
plot(x, y, main = "car selling price", xlab = "cars", ylab = "price", pch = 19, frame = FALSE)
abline(lm(y~x, data = mtcars), col = "blue")
```

13/2/24

1b- student marks

program :-

```
install.packages("readxl")
library(readxl)
student_marks_data <- read_excel("D:/Dvtiq Tasks/student_data.xls/x")
view(student_marks_data)
```




```
x ← student_marks_data $ S.no
y ← student_marks_data $ spm
plot(x,y main = "student marks",
     x lab = "STUDENTS",
     y lab = "marks",
     pch = 19,
     frame = FALSE)
abline(lm(y~x) data = "student_marks_data",
       col = 'GREEN')
```

Result:-

The program has been executed successfully

778

15/2/24.

22/02/23

Task-03

7) Statistical Analysis

The R programming language provides a wide range of packages and functions for statistical analysis.

Multivariate Analysis :-

Principle Component Analysis (PCA) : PCA is used to identify the most important features or patterns in a dataset. The prcomp() function in base R or the procomp() function from the stats package can be used Ex :-

Syntax :-

```
pca_result <- prcomp(data)
summary(pca_result) # view summary of PCA
```

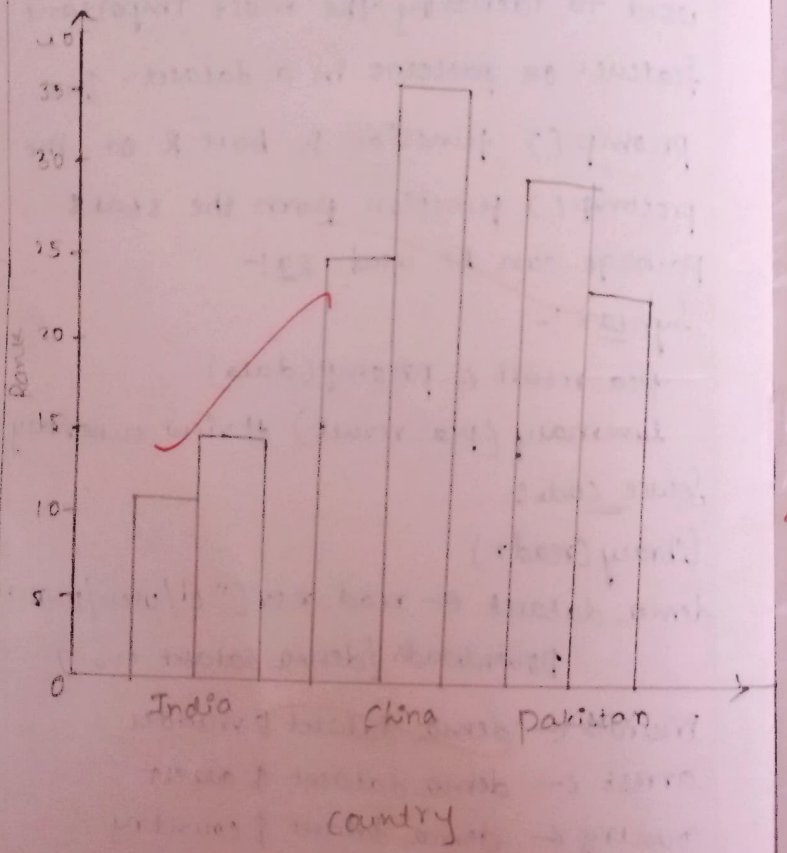
Source code :-

```
library(readr)
demo_dataset <- read_csv("C:/Users/LEGRLET/
        Downloads/demo_dataset.csv")
murder <- demo_dataset $ murder
arrest <- demo_dataset $ arrest
country <- demo_dataset $ country
# converting into Dataframes
data <- data_frame(category = country,
                   murder = murder,
                   arrest = arrest)
```

Output :-

Importance of components :

	PC1	PC2
Standard deviation	1.2442	0.6722
proportion of variance	0.7741	0.2259
Cummulative proportion	0.7741	1.0000



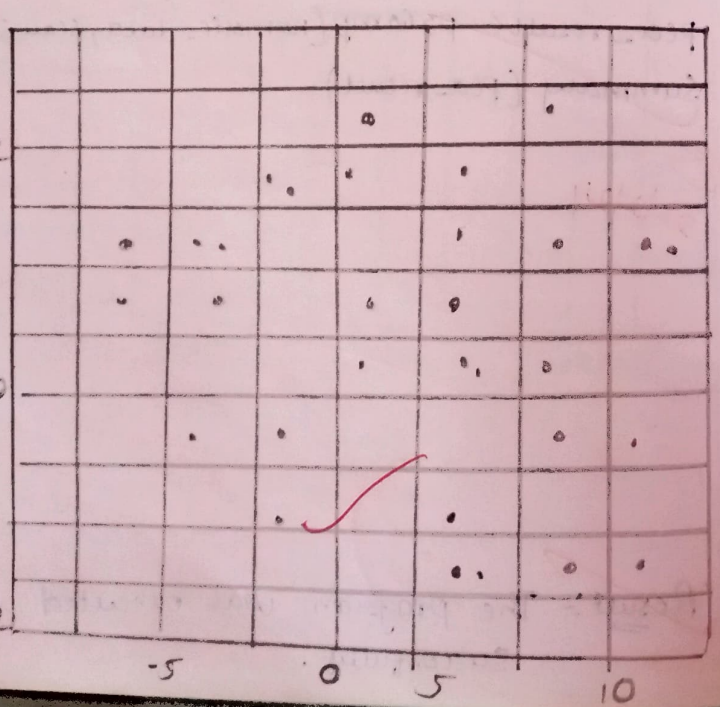

```
# plotting the Bar Graph
barplot(
    height = t(data, c("murder", "arrest")),
    beside = TRUE,
    col= c("blue", "green")
    names.arg = data$ category,
    main = paste("opening & closing Ranking"),
    xlab = "country",
    ylab = "Rank",
    legend.text = c("murder", "arrest"),
    args.legend = list(x = "topleft", bty="n")
)

# Applying the PCA
numeric_data ← data[, sapply(data, is.numeric)]
pca_result ← prcomp(numeric_data, scale=TRUE)
summary(pca_result)
```

22/2/24

Result :- The program was executed successfully.

Output:-

	LD1	LD2
4	7.150360	0.7177382
6	7.961538	-1.4839406
7	2.504038	-0.2731178
15	10.170378	-1.9859027
17	8.865168	-2.1026497
18	8.113448	-0.7563902.



24/02/24

B) Statistical Analysis such as multivariate Analysis, PCA, LDA, correlation regression and analysis of variance

```
Library (MASS)
Library (ggplot2)
str(iris)
iris[1:4] <- scale(iris[1:4], 2, mean)
apply(iris[1:4], 2, sd)
set.seed(1)
sample <- sample (c TRUE, FALSE), nrow(iris),
          replace = TRUE, prob(0.7, 0.3))
  train <- iris[sample]
  Model <- lda(species ~ , data = train)
Model
predicted <- predict(Model, test)
names(predicted)
head(predicted $class)
head(predicted $ posterior)
head(predicted $class == test $species)
Lda-plot <- cbind(train, predict(model)$x)
ggplot(da-plot, aes(LD1, LD2)) + geom_point
          (aes(color = species))
```

9/12/24

18/24

Task - 04

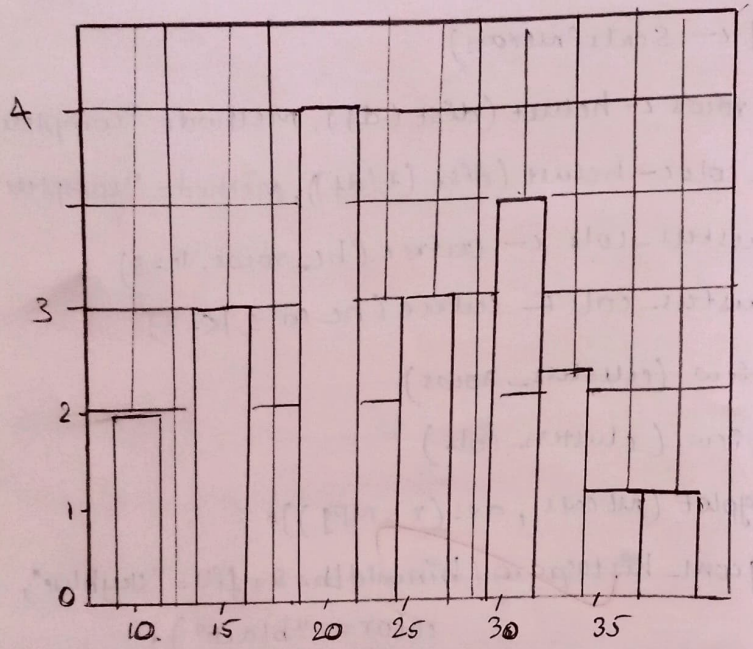
Financial Analysis using clustering, histogram and heatmap

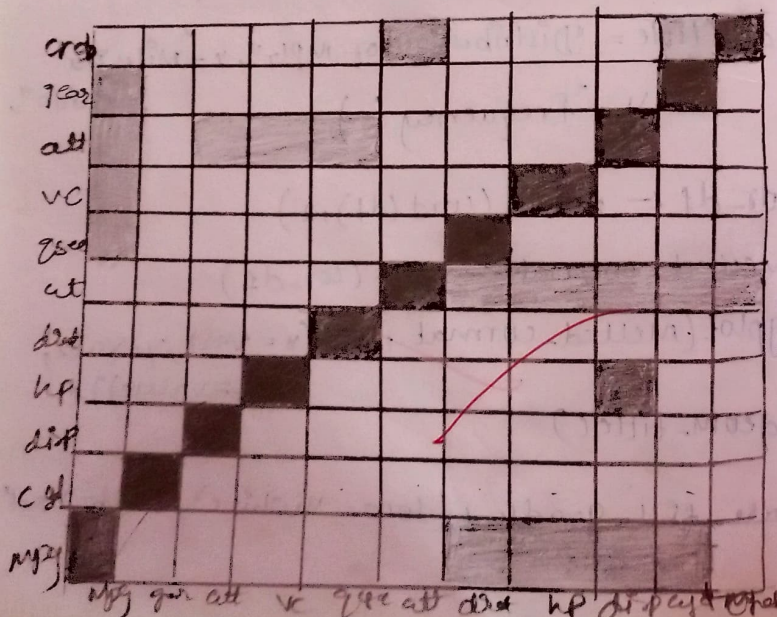
```
Library (ggplot2)
Library (reshape2)
df <- Scale(mtcars)
hc_rows <- hclust (dist (df), method= "complete")
hc_cols <- hclust (dist (t(df)), Method= "complete")
clusters_cols <- cutree (hc_rows, k=3)
clusters_cols <- cutree (hc_cols, k=3)
View (clusters_rows)
view (clusters_cols)
ggplot (mtcars , aes (x = MPG ))+
geom_histogram (binwidth=2, fill= "skyblue",
                color = "black")+
Lab (title = "Distribution of MPG", x = "Miles per Gallon",
     Y = "Frequency ")
cor_df <- round (cord (df), 2)
melted_cormat <- Melt (cor_df)
ggplot (melted_cormat , aes (x = Var1 , Y=Var2,
                             fill=value))+
geom_title()
Scale_fill_gradient (low= "white", high="red")+
```

Output:-

> clusters_cols

Mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
1	2	2	2	1	2	3	1	1	1	2






```
labs(title = "correlation heatMap", x="", y="")
clusters_rows <- cutree(hc_rocos, k=3)
clusters_cols <- cutree(hc_cols, k=3)
```

Result:-

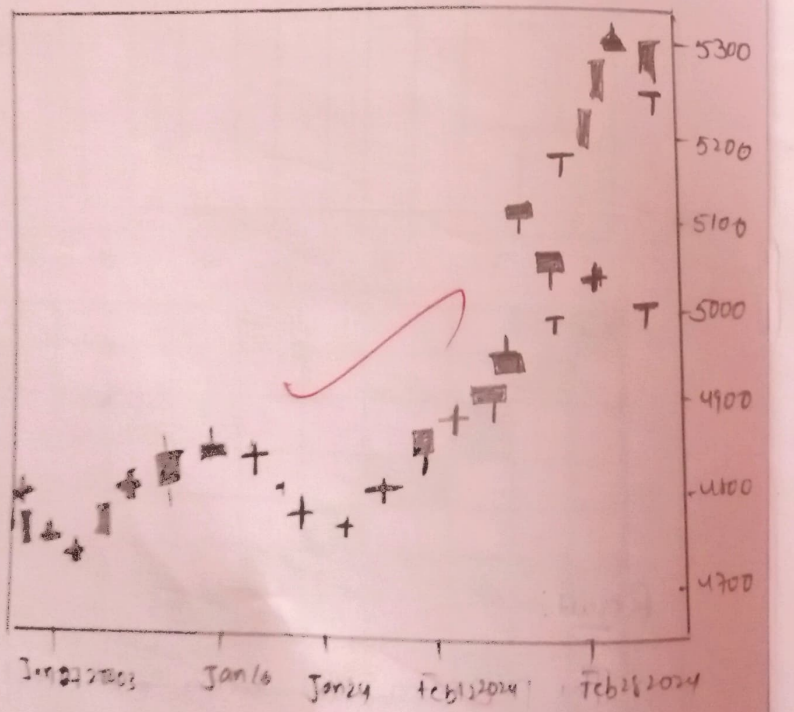
The program has been executed successfully.

21/3/24

Output:-

head(gspc)

gspc					
2024-01-02	4746.20	4754.33	4722.67	4742.83	3745500000
2024-01-03	4745.70	4729.29	4695.71	4704.81	3950760000
2024-01-04	4697.42	4726.98	4688.63	4688.68	3715490000
2024-01-05	4690.57	4721.40	4.69724	4697.24	3847000
2024-01-08	4903.70	4764.54	4691282	4763.54	3760320000
2024-01-09	4741.93	4766.47	4730.35	4754.50	3629960000



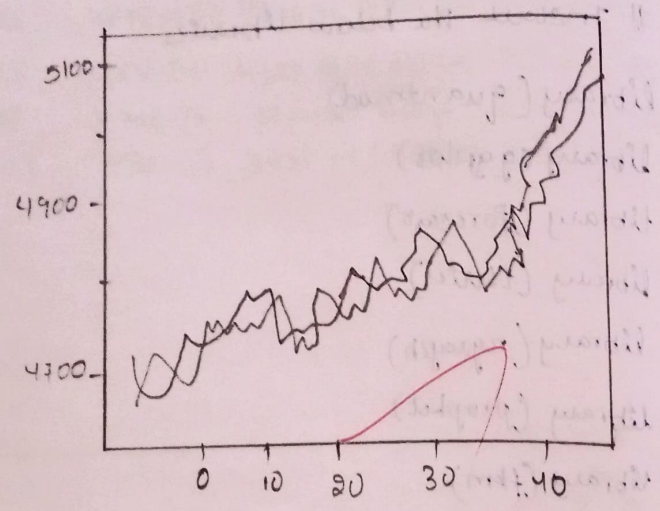
21/3/24

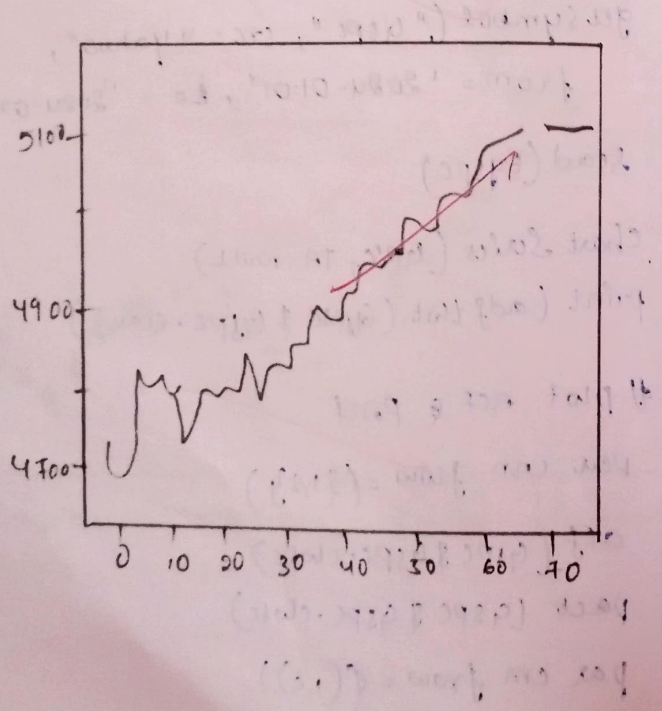
EXPERIMENT-5

Aim :- Time Series - Analysis - Stock Marketing

Code :-

```
# Install the below libraries
library(quantmod)
library(ggplot)
library(forecast)
library(tseries)
library(rugraph)
library(prophet)
library(tkm)
#import dataset from yahoo
getsymbol("gspc", src="yahoo",
    from='2024-01-01', to="2024-02-24")
head(gspc)
chartSeries(gspc, TA-NULL)
print(adf.test(Gspc$GSPC.Close))
# plot ACF & PACF
par(mfrow=c(1,2))
acf(GSPC$GSPC.Close)
pacf(GSPC$GSPC.Close)
par(mfrow=c(1,1))
```





Histogram of Residual of normally assumption

```
hist(resid(model_fit), freq=F, ylim=c(0,.5))
     main = "Histogram" of Residuals")
e = Resid(model_fit)
curve(dnorm(x, mean=mean(e), sd=sd(e)),
  add=true, col="darkred")
```

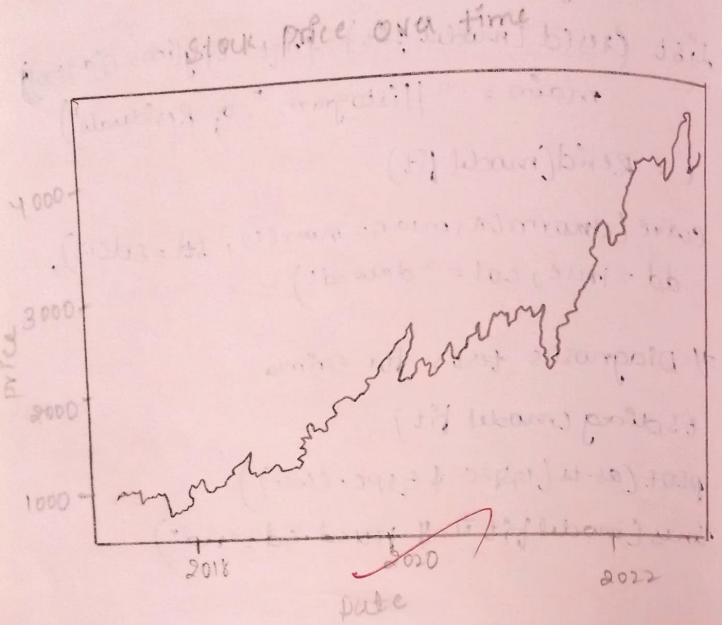
Diagnosis tests for arima

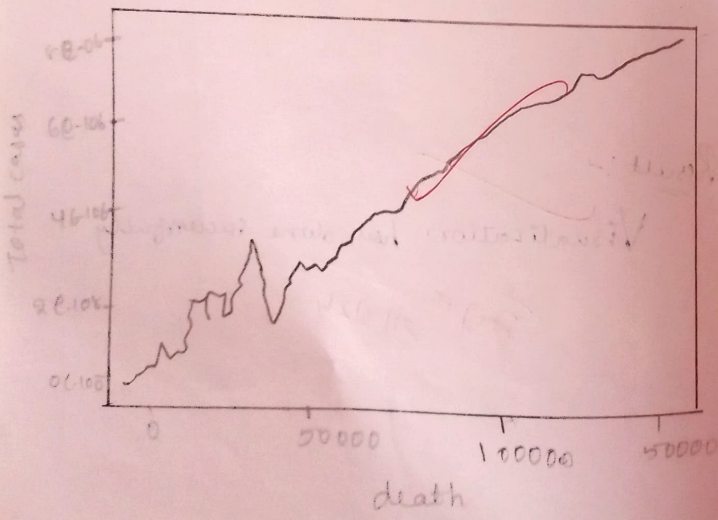
```
tsdiag(model_fit)
plot(as.ts(GSPC$GSPC.Close))
lines(model_fit$ fitted, col="red")
```

Result :-

Visualization has done successfully

21/3/24





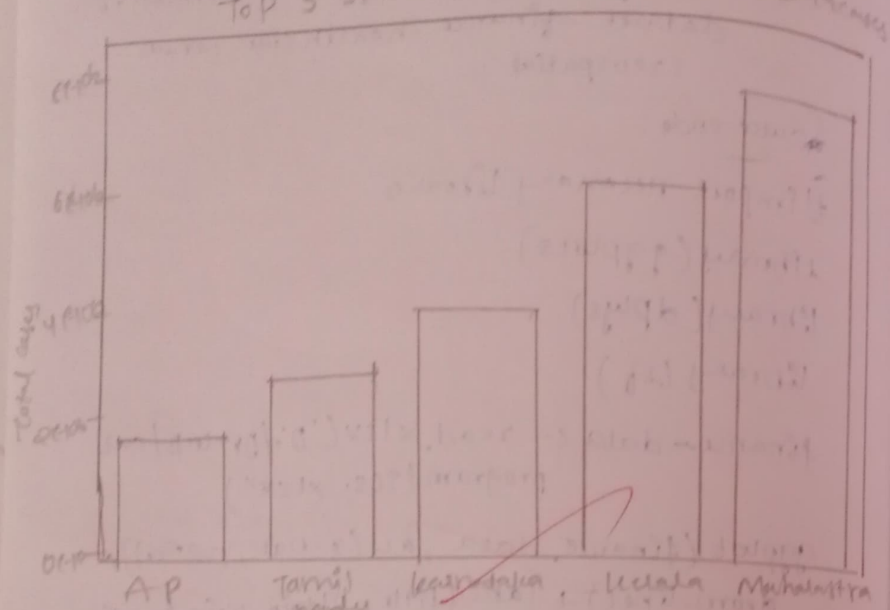
28/3/24

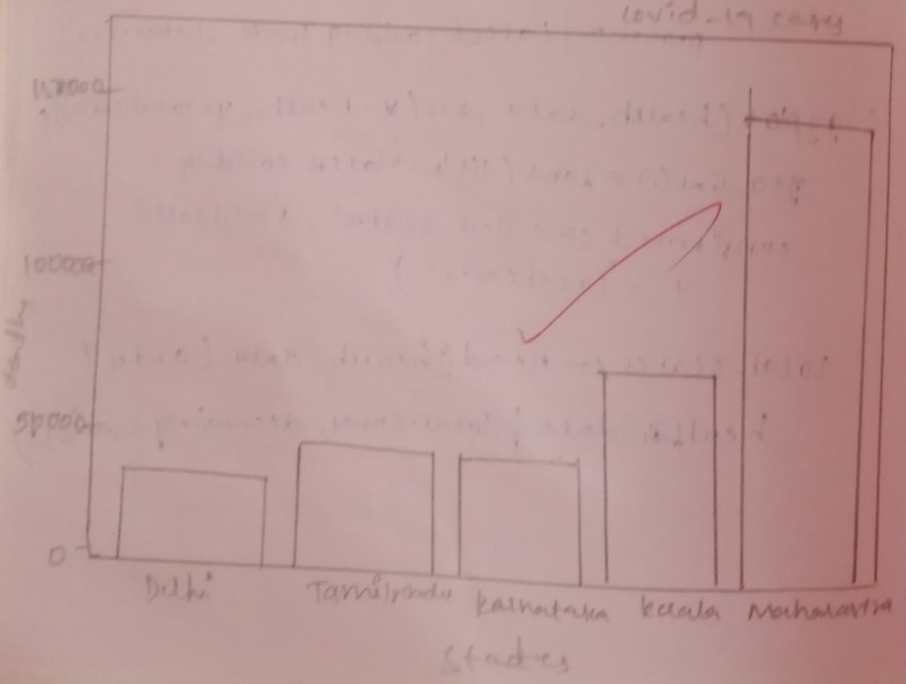
Task-06

Aim: The visualization of various massive dataset - finance - health care - census - Geospatial

Source code:

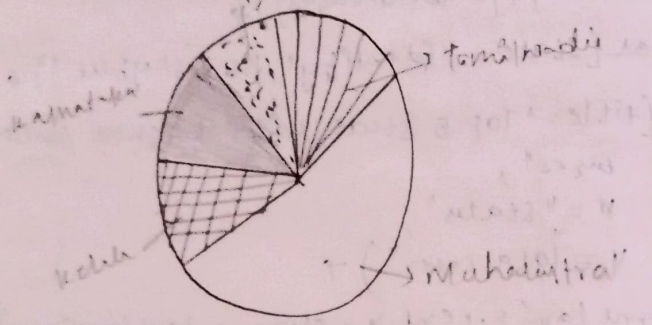
```
#import necessary libraries
library(ggplot2)
library(dplyr)
library(lsf)
finance_data <- read_xlsx("D:/DV1 LAB/cal program/TCS. xlsx")
ggplot(finance_data, aes(x=Date, y=price))+
  geom_line()+labs(title="stock price over time",
  x="Date", y="price")
#import health care dataset
health_data <- read.csv("D:/DV1 LAB/cal program/Latest covid-19 India status.csv")
ggplot(health_data, aes(x=Deaths, y=total.cases))+
  geo_line()+labs(title="latest covid-19
  confirmed case over states", x='death",
  y="total cases")
total_states <- head(health_data(order(
  health_data$total.cases, decreasing=True)),5)
```



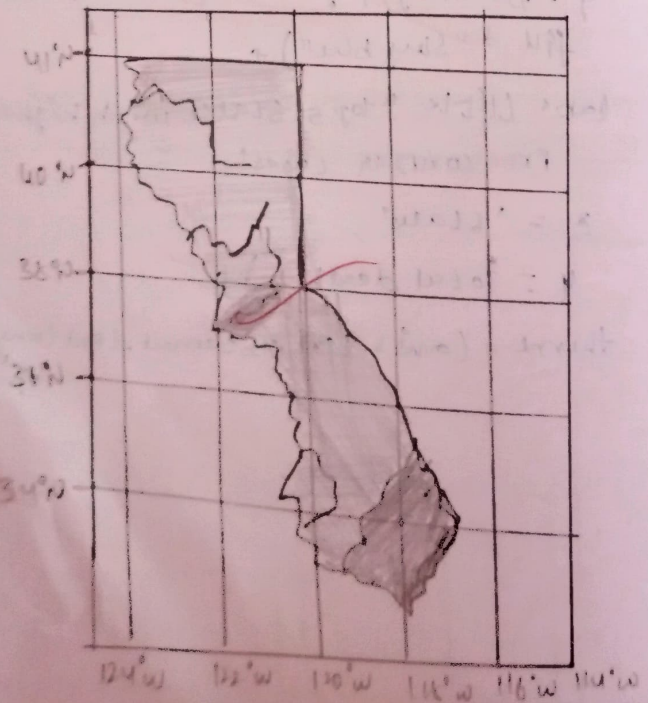



```
ggplot(top_states, aes(x = reorder(state.UTs,
  total.cases), y = total.cases)) +
geo_bar(state = "identity", fill = "skyblue") +
labs(title = "Top 5 states with highest covID19
      cases",
     x = "states",
     y = "total cases") +
theme(axis.text.x = element_text(angle=45))

top_states <- head(health_data(order(
  health_data$, Deaths, decreasing = TRUE),],5)

ggplot(top_status, aes(x = reorder(state.UTs, Deaths),
  y = Deaths)) + geom_bar(stat = "identity",
  fill = "sky blue") +
labs(title = "top 5 states with highest death
   in covID-19 cases",
  x = 'states',
  y = 'total deaths') +
theme = (axis.text.x = element_text(angle=45,
                                    hjust = 1))
```

Top 5 states with highest covid-19 Deaths



Population by country in California



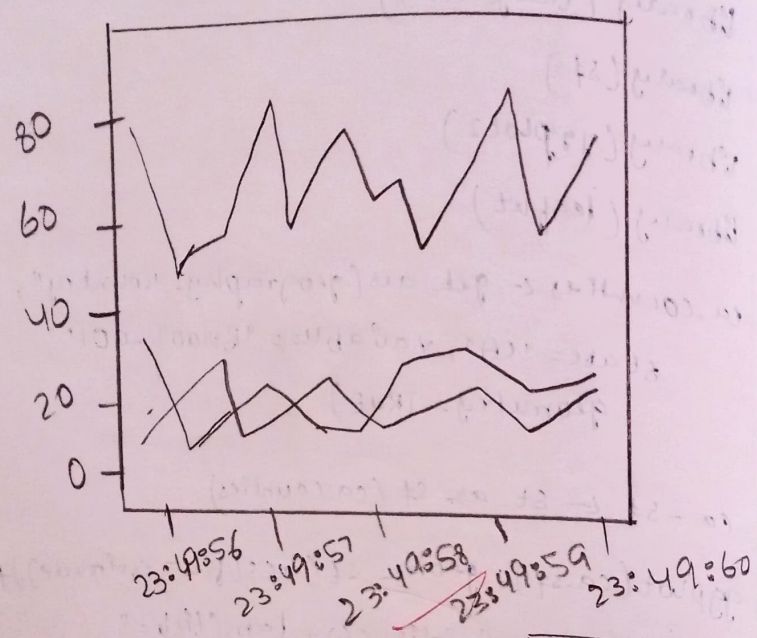

```
install.packages(c("tidycensus", "sf", "ggplot2",
                   "leaflet"))
library(tidycensus)
library(sf)
library(ggplot2)
library(leaflet)

ca_counties <- get_acs(geography = "country",
    state = "CA", variables = "B01003_001",
      geometry = TRUE)

ca_sf <- st_as_sf(ca_counties)

ggplot(ca_sf) + geom_sf(aes(fill = estimate)) +
  scale_fill_viridis_c() + labs(title =
  "Population by country in California")
leaflet(ca_sf) %>% addTiles() %>%
  color = "white", weight = 1, smoothFactor = 0.5))
```

Result:

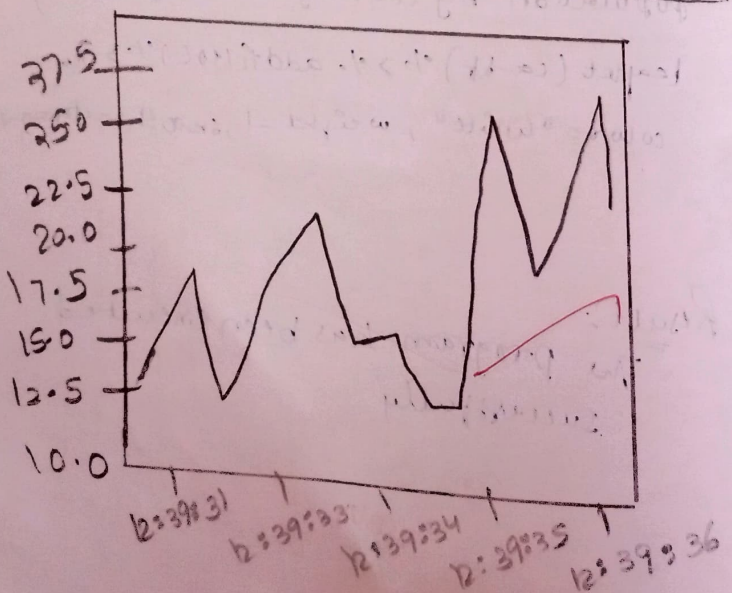
The program has been executed successfully

28/8/24

Weather data



Temp(C)



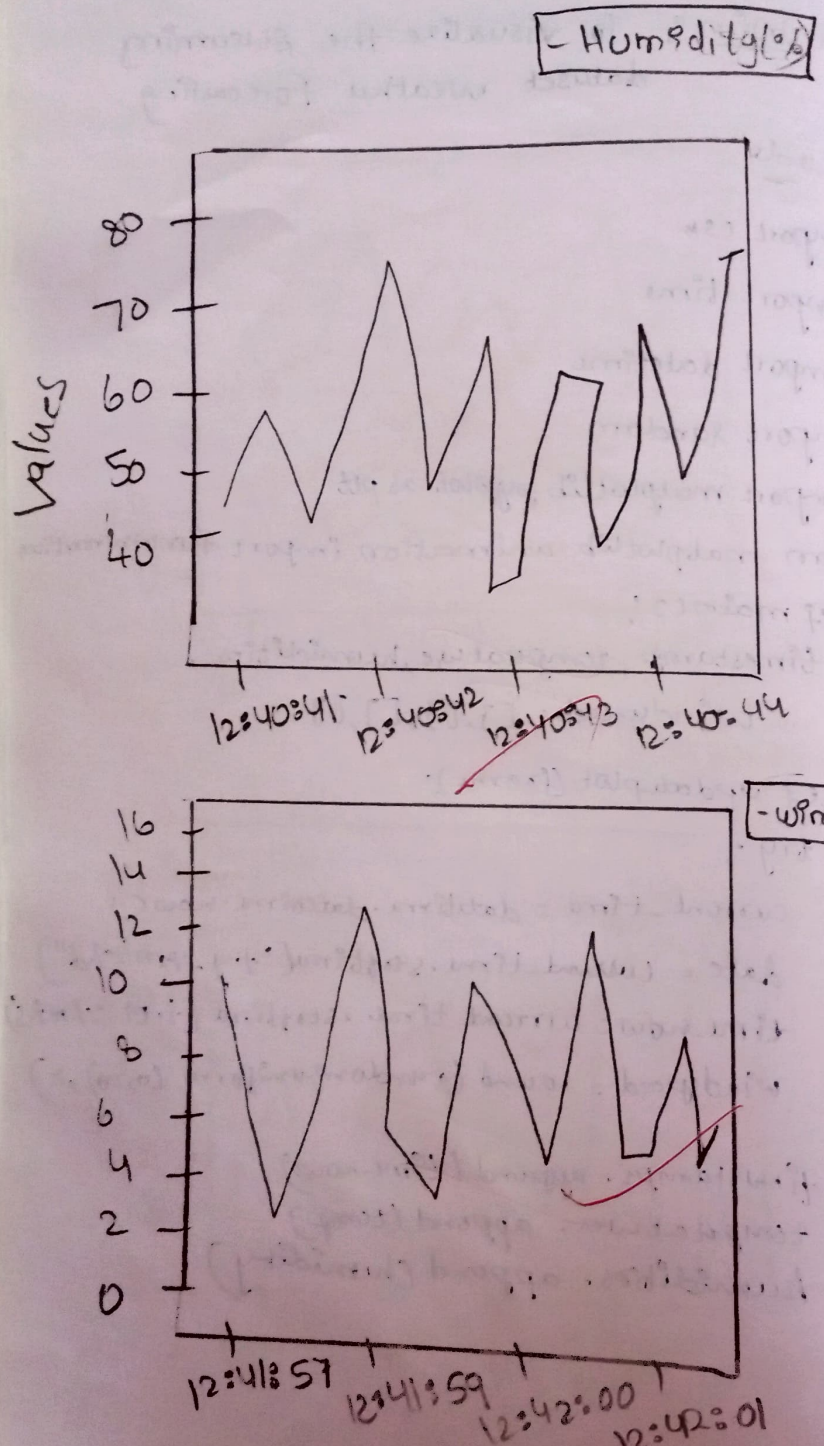
Oct 24

Experiment-7

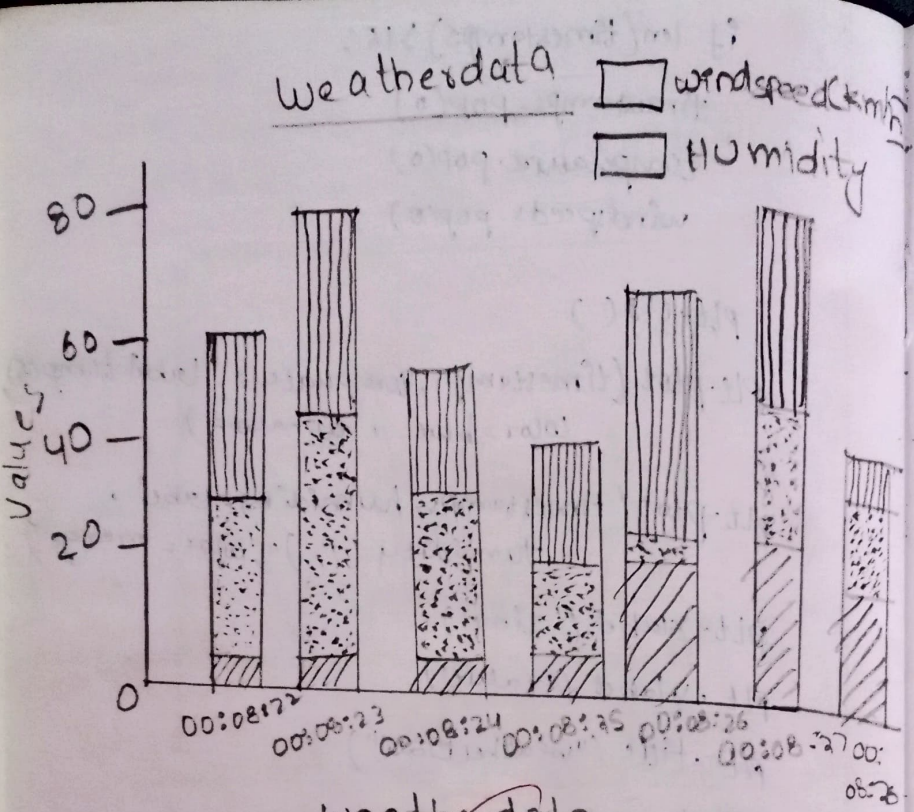
1(a) Aim :- To visualize the streaming dataset weather forecasting

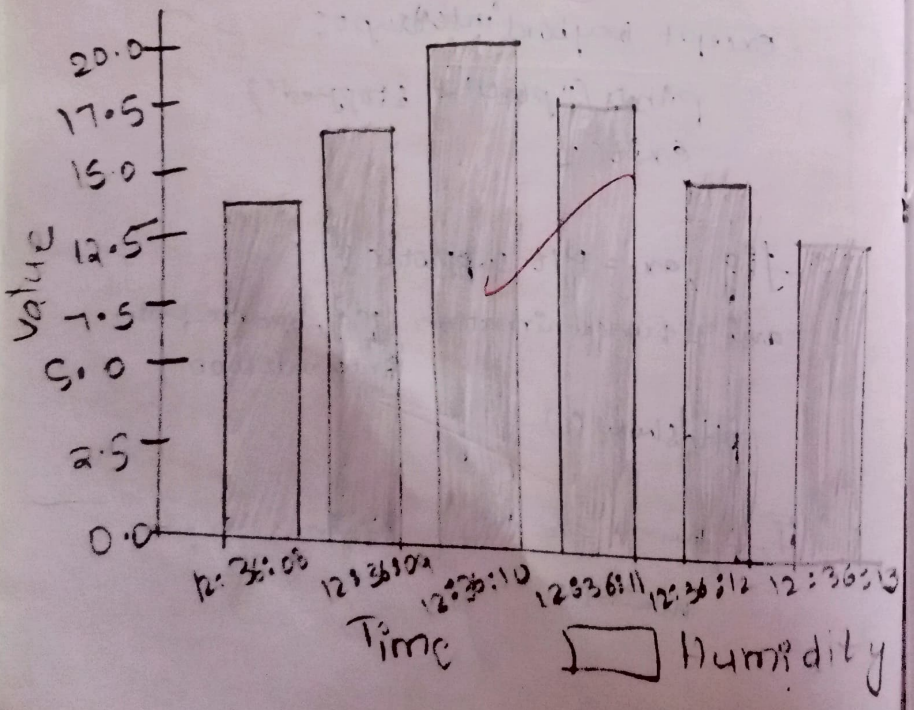
code :-

```
import csv
import time
import datetime
import random
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
def main():
    timestamp, temperature, humidities,
        windspeeds = [],[],[],[]
def updateplot(frame):
    try:
        current_time = datetime.datetime.now()
        date = current_time.strftime("%y-%m-%d")
        time_now = current_time.strftime("%H:%M:%S")
        windspeed = round(random.uniform(0,20),2)

        timestamps.append(time_now)
        temperatures.append(temp)
        humidities.append(humidity)
```

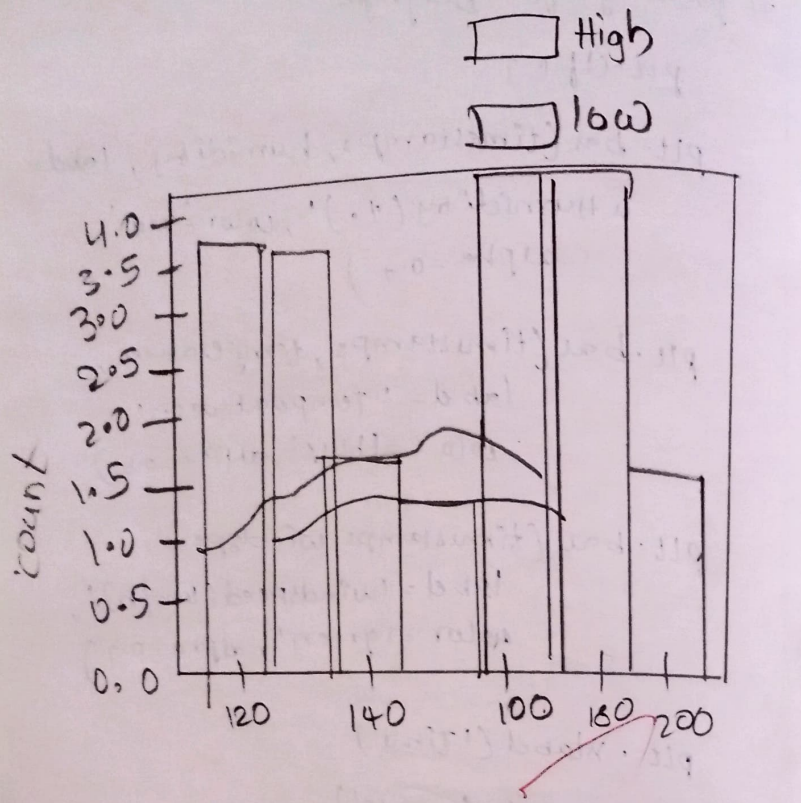


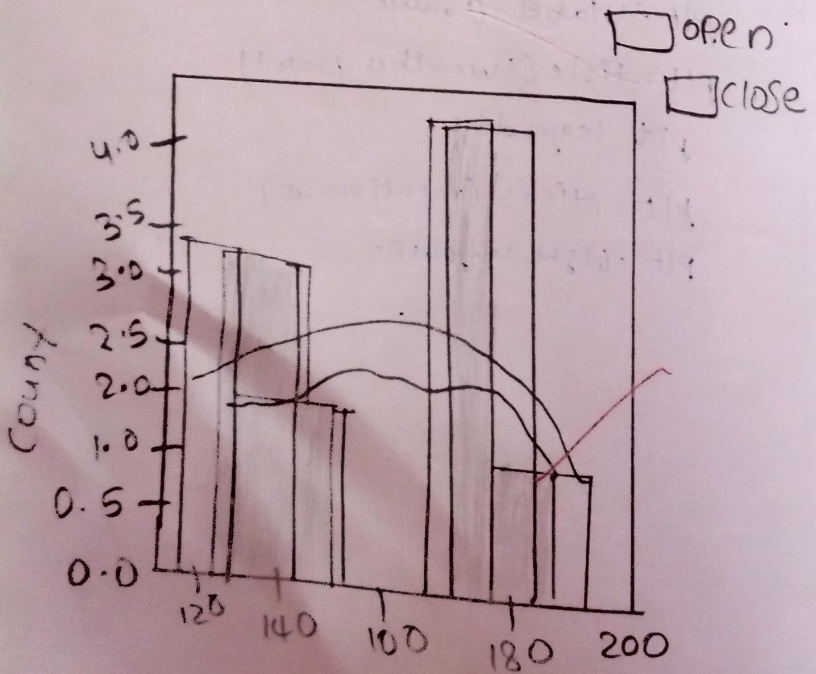

```
if len(timestamps) > 15:
    timestamps.pop(0)
    temperature.pop(0)
    windspeeds.pop(0)

plt.clf()
plt.plot(timestamps, temperature, label='Temp(C)',
         color='blue', alpha=0.7)
plt.plot(timestamps, humidity, label=
         'Humidity (%)', color='orange')
plt.xlabel('Time')
plt.ylabel('Value')
plt.title("Weather Data")
plt.legend()
except keyboardInterrupt:
    print("plotting stopped")
    exit()

fig, ax = plt.subplots()
ani = FuncAnimation(fig, update_plot,
                    interval=1000)
plt.show()

if __name__ == "__main__":
    main()
```






```
# plotting for bargraph
plt.clf()
plt.bar(timestamps, humidity, label=
    "Humidity (%)", color='red',
    alpha=0.7)
plt.bar(timestamps, temperatures,
    label="Temperature(C)",
    color='blue', alpha=0.5)
plt.bar(timestamps, windspeed,
    label='windspeed(km/h)',
    color='green', alpha=0.7)
plt.xlabel('Time')
plt.ylabel('value')
plt.title('weather Data')
plt.legend()
plt.xticks(rotation=45)
plt.tight_layout()
```

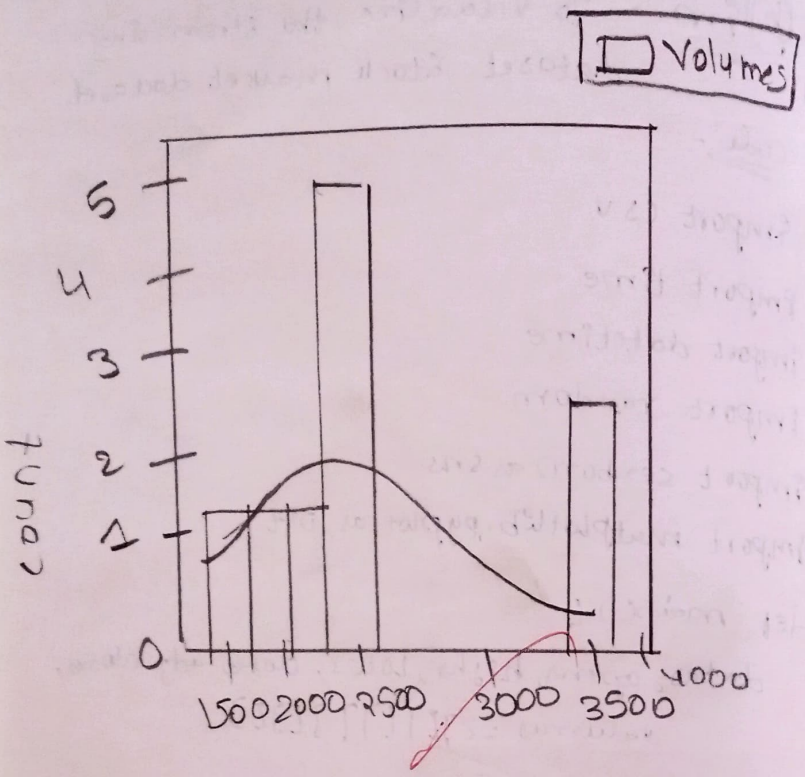




~~Title~~ Aim :- To visualize the streaming dataset stock market dataset

Code :-

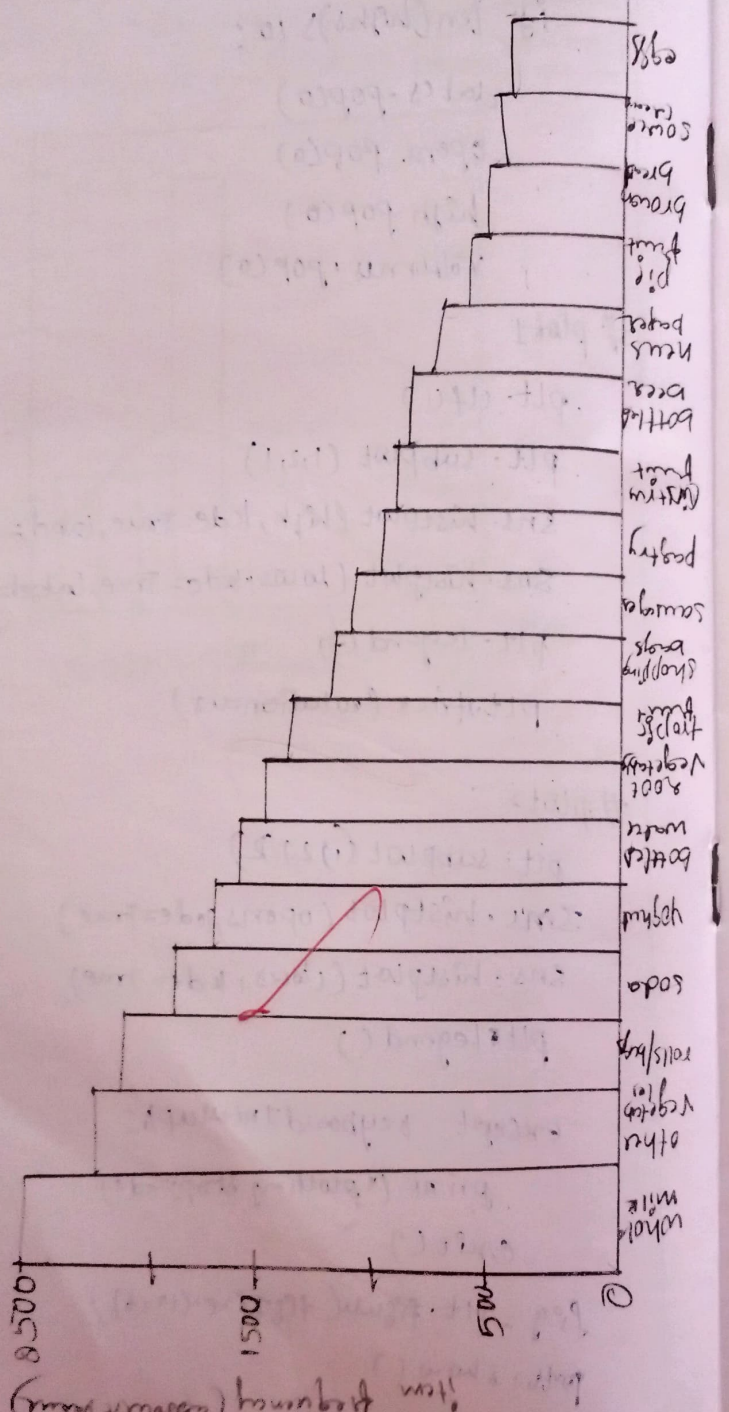
```
import csv
import time
import datetime
import random
import seaborn as sns
import matplotlib.pyplot as plt

def main():
    dates, opens, highs, lows, closes, adj-closes,
        volumes = [][][][][][][]

def update_plot (frame):
    try:
        current_time = datetime.datetime.now()
        date = current_time.strftime("%Y-%m-%d")
        time_now = current_time.strftime("%H-%M-%S")
        open_price = round(random.uniform(100,200),2)
        high_price = ~~random~~(open_price * random.uniform
                                (1,1,),),2)
        volume = ..random.randint(1000,50000)

    dates.append(date)
    opens.append(open_price)
    high.append(high_pri:a)
```




```
        volumes.append(volume)
        if len(highs) > 10:
            dates.pop(0)
            opens.pop(0)
            high.pop(0)
            volumes.pop(0)

    # plot 1
        plt.clf()
        plt.subplot(1,2,1)
        sns.histplot(high, kde=True, label='High')
        sns.histplot(lows, kde=True, label='Low')
        plt.legend()
        plt.xticks(rotation=45)

    # plot 2
        plt.subplot(1,2,2)
        sns.histplot(opens, kde=True)
        sns.histplot(closes, kde=True)
        plt.legend()

    except KeyboardInterrupt:
        print("plotting stopped")
        exit()

fig = plt.figure(figsize=(12,6))
plt.show()
if __name__ == '__main__':
    main()
```

4/4/24

Output



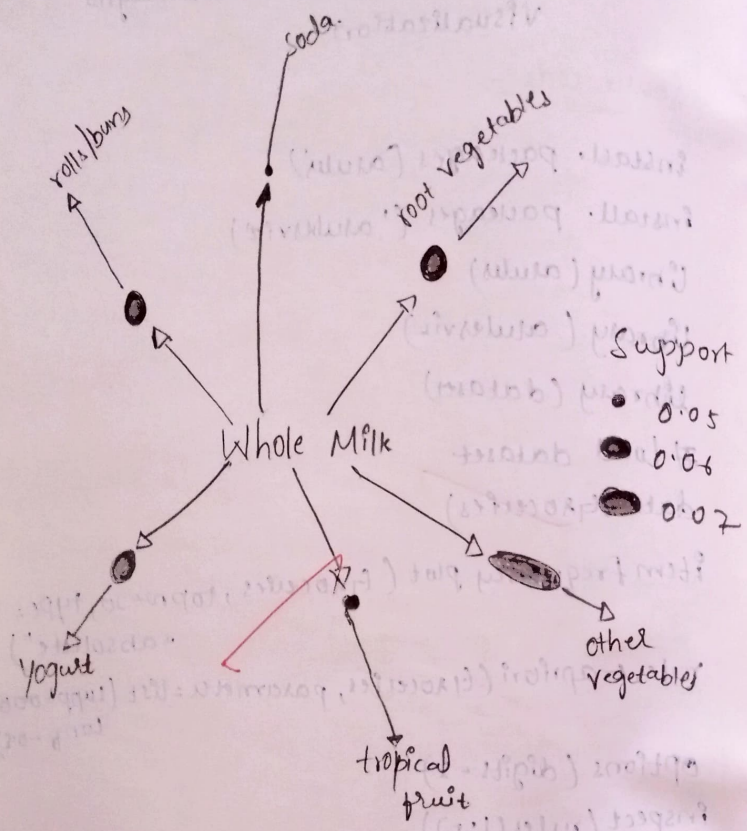
18/4/24

Experiment - 8

Aim :- Market Basket Data Analysis Visualization

Sowce code :-

```
install.packages ("arules")
install.packages ("arulesviz")
library (arules)
library (arulesviz)
library (dataset)
# load dataset
data (Groceries)
itemFrequencyPlot (Groceries, topN=20, type="absolute")
rules <- apriori (Groceries, parameter = list (supp=0.001, conf=0.8))
options (digits = 2)
inspect (rules[1:5])
rules <- sort (rules, by="confidence", decreasing=TRUE)
rules <- apriori (Groceries, parameter= list (supp=0.001, conf=0.1, maxlen=3))
subset.matrix <- is.subset (rules, rules)
subset.matrix [lower.tri (subset.matrix, na.rm=T)] >= 1
rules.pruned <- rules [! redundant]
rules <- rules.pruned
```




```
rules ← sort (rules, decreasing = TRUE, by =
                                    confidence)
inspect (rules[1:5])

rules ← apriori (data = Groceries, parameter =
        list(supp = 0.001, conf = 0.15, minlen = 2),
        appearance = list (default = 'rhs', lhs =
        'whole milk'), control = list (verbose = F))

rules ← sort (rules, decreasing = TRUE, by =
                                    "confidence")
inspect (rules [1:5])

plot (rules, method = "graph")
```

Result :-

The program was executed successfully

25/12/23

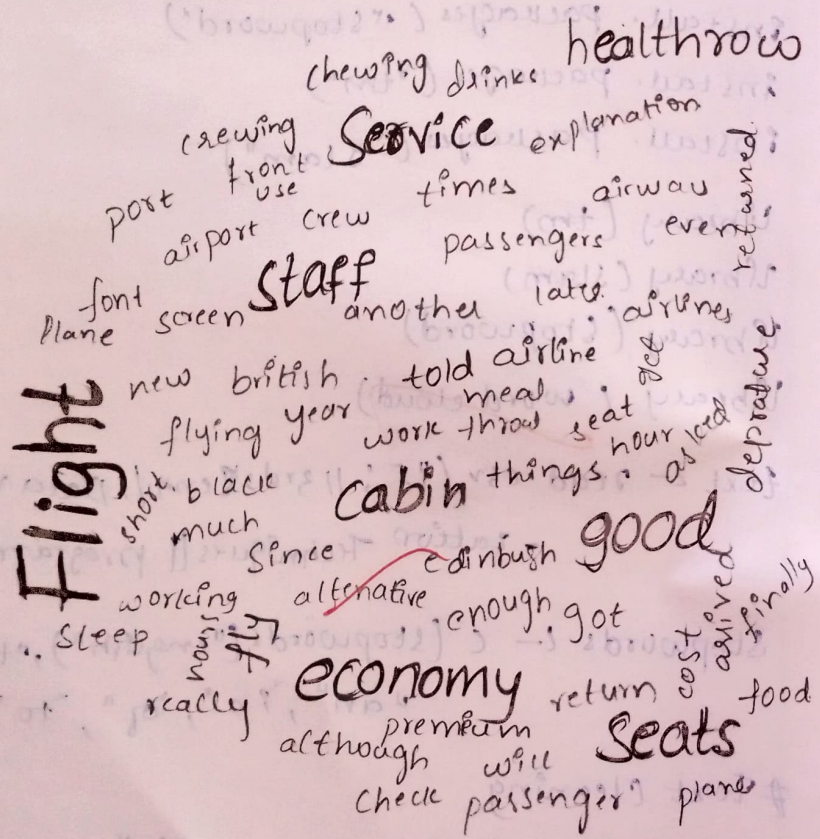
Experiment - 9

Aim :- Text visualization using web analytics.

Source code :-

```
install.packages("wordcloud")
install.packages("stopword")
install.packages("tm")
install.packages("slam")

library(tm)
library(slam)
library(stopword)
library(wordcloud)

text <- read.csv("F:\\3rd 2sem\\Data Visualization Techniques\\programs\data.csv")

stopwords <- c(stopwords("english"),"the","a","an","in","of","to","is")

#Text cleaning
text <- gsub("[^[:alnum:]]"," ",text)
text <- tolower(text)

wordcloud <- wordcloud(words = strsplit(text, split=" ")[[1]]
stopwords = stopwords
```

Output



min. frequency = 3

scale = c(3, 0.5)

colors = brewer.pal(8, "Dark2"))

Result :-

The program was executed successfully