Vel Tech
Rangarajan Dr. Sagunthala
R&D Institute of Science and Technology

School of Computing
Department of Computer Science & Engineering
(Artificial Intelligence and Machine Learning)

ACADEMIC YEAR 2025-26 (SUMMER SEMESTER)

LAB RECORD NOTEBOOK

10212CA214 - DATA VISUALIZATION

NAME: S. Rubiya

VTU.NO: 24905

REG.NO: 23UECL0056

BRANCH: CSE(AIML)

YEAR/SEM: III rd / summer 25-26

SLOT: $S_{12}L_1$

Vel Tech
Rangarajan Dr. Sagunthala
R&D Institute of Science and Technology

NAAC A++

CATEGORY-1 UGC

School of Computing
Department of Computer Science & Engineering
(Artificial Intelligence and Machine Learning)

ACADEMIC YEAR 2025-26 (SUMMER SEMESTER)

BONAFIDE CERTIFICATE

NAME : S. Rubiya **BRANCH** : CSE(AIML)

VTU NO. : 24905 **REG.NO.** : 23UECL0056

YEAR/SEM : III rd / Summer 25-26 **SLOT NO.** : S12L1

Certified that this is a bonafide record of work done by above student in the **"10212CA214 - DATA VISUALIZATION LABORATORY"** during the year 2025-2026 (Summer Semester).

R.T. Thayani 29/10/25

SIGNATURE OF LAB HANDLING FACULTY **SIGNATURE OF HOD**

Submitted for the Semester Practical Examination held on **03.11.25** at Vel Tech Rangarajan Dr.Sagunthala R&D Institute of Science and Technology.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

INDEX

S. No	Date	Title	Page No.	Marks	Faculty Signature
1.	21/07/25	Exploration of Data Visualization Tools like Tableau, Python libraries, D3.js	1-4	17	4/7
2.	28/07/25	To visualize and perform Univariate analysis using continuous and categorical data	5-10	16	28/7
3.	04/08/25	To visualize and perform Bivariate analysis using continuous and categorical data	11-14	18	4/8
4.	11/08/25	To visualize and perform Multivariate analysis using Multiple variables involving Multiple measures	15-17	16	11/8
5.	18/08/25	To design and perform visualization for Trees	18-21	19	18/8
6.	25/08/25	To design and perform visualization for Graphs and Networks	22-24	19	25/8
7.	08/09/25	To generate insight using Text Network Analysis and Visualization	25-30	16	8/9
8.	15/09/25	To analyze and visualize Spatial and Geospatial data	31-33	19	15/9
9.	29/09/25	To analyze and visualize Time Oriented Data	34-37	19	29/9
10.	13/10/25	Use Case: performance of sales company departments over year	38-42	17	13/10
11.	27/10/25	Use Case: Earthquake and Geospatial Data Analysis	43-47	18	27/10

Completed

Total Marks: 194/220

R.T. Tripathi

Signature of Faculty

21/7/25

TASK-1A

Exploration of data visualization To do table python libraries D3e.js

- Connecting dataset
- Preparation of data

Aim :

To visualize the distribution of different graph leaf conditions and explore their metadata using python or other visualization tools.

Algorithms :

1. Impact Dataset
   - load the dataset containing folders of grapevine leaf images classified by conditions
2. Extract metadata
   - list all images files from each class folder
   - create a structured format with two attributes . Images Name class label.
3. Processes and clean data :
   - Ensure only valid images files are included (jpg . png)
   - Remove duplicate if any
4. create summary Tables
   - count summary tables , number of images in each class

S.No	plant ID	variety	species/ origin	Seeded or Seedless	Typical use (Table / Juice
1	Apple	Honeycrisp	Malus domestica	Seeded	Eating fresh
2	Apple	Grannysmith	Australia	Seeded	cooking
3	Apple	Fuji	Japan	seeded	Eating fresh
4	Apple	Golden delicious	USA	Seeded	Baking
5	Apple	Red delicious	USA	seeded	Eating fresh
6	Apple	Gala	Newzealand	seeded	Eating fresh
7	Apple	Braeburn	Newzealand	Seeded	cooking
8	Apple	cortland	USA	Seeded	salads
9	Apple Mcintosh	Mcintosh canada	canada	Seeded	cider
10	Apple	pinklady	Australia	Seeded	Eatingfresh

o/p

5. visualization coptionals:
· Generate a bar chart to show the distribution of images per class using python (matplotlib / seaborn)

Result: Thus to visualize the distribution of different grapple conditions and explane their metadata using python or other visualization tools was executed successfully.

output:

Program Explanation

1. Pandas is a powerfull library for data analysis and manipulation
2. Pd. read_csv is used to read a csv
3. df head() displays the first5 rows of dataset.
4. df. isnull () creates a bodean Data frame we the true for mpssing values.
5. dropna () removes any rows that contain null values.
6. df_cleane. duplicated() returns a boolean serve with true for duplicates.
7. drop - duplicates () removes any repeated duplicate rows.

Task-1B

Aim:

To Perform data preprocessing on a dataset by identifying and handling null values and duplicate entries.

Algorithm:

1. Import necessary python libraries
2. Load the dataset using pandas
3. Display the first 5 rows of the dataset.
4. check for nullvalues on each column
5. Remove null values of any.
6. Identify duplicate rows.
7. Remove duplicate rows.

output:

First 5 rows of the dataset:

	Speal-length	Speal-length	petal-length	petal-width	
0	5.1	3.5	1.4	0.2	[illegible]
1	4.9	3.0	1.4	0.2	Set
2	4.7	3.2	1.3	0.2	Seto
3	4.6	3.1	1.5	0.2	Set
4	5.0	3.6	1.4	0.2	Se

count of null values in each column

speal-length 0
Speal-width 0
petal-length 0
petal-width 0
Specpes
dtype : int64

Dataframe after removing null values (if any)

	Speal-length	Speal-width	petal-length	petal-width	Spec
0	5.1	3.5	1.4	0.2	Seto
1	4.9	3.0	1.4	0.2	Seto
2	4.7	3.2	1.3	0.2	Seto
3	4.6	3.1	1.5	0.2	seto
4	5.0	3.6	1.4	0.2	set

o/p

Duplicate rows in the dataset

	Speal-len	Speal-width	petal-len	petal-width	Specie
34	4.9	3.1	1.5	0.1	Setase
37	4.9	3.1	1.5	0.1	Setase

Dataframe after removing duplicates

	Speal-length	Speal-width	petal-length	petal-width	Speci
0	5.1	3.5	1.4	0.2	Setase
1	4.9	3.0	1.4	0.2	Setase
2	4.7	3.2	1.3	0.2	setase
3	4.6	3.1	1.5	0.2	Setac
4	5.0	3.6	1.4	0.2	setose

Program:

```
import pandas as pd
df = pd.read_csv("https://raw.github
        use.content.com/upuc_cse/data
            fo14 lgh-page/data/exps.csv")
print ("First 5 rows as the dataset")
print (df.head())
print ("\n count of null values in each
                          column:")
print (df.isnull().sum())
df_cleaned = df.dropna()
print ("\n DataFrame after removing
                null values:")
print (df_cleaned.head())
duplicates = df_cleaned [df_cleaned.duplicated()]
print ("\n Duplicate rows in dataset:")
df_no_duplicates = df_cleaned.duplicates()
print ("\n DataFrame after removing duplicates")
print (df_no_duplicates.head())
```

VEL TECH	
EX No.	1
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	2
RECORD (5)	3+2
TOTAL (20)	5+5+3+2=17
SIGN WITH DATE	

21/7/25

Result: Thus to perform data preprocessing on dataset by identifying and handling null values and duplicates are done and executed successfully.

28/7/25

Task-2

To visualize and perform univeriate Analysis using continuous and categorical data

Categorical Data : Bar chart, pie chart

Contignous Data : scatter plot, line-plot, strip plot, swarm plot, Rug plot

Tools : Table - ID, language Python

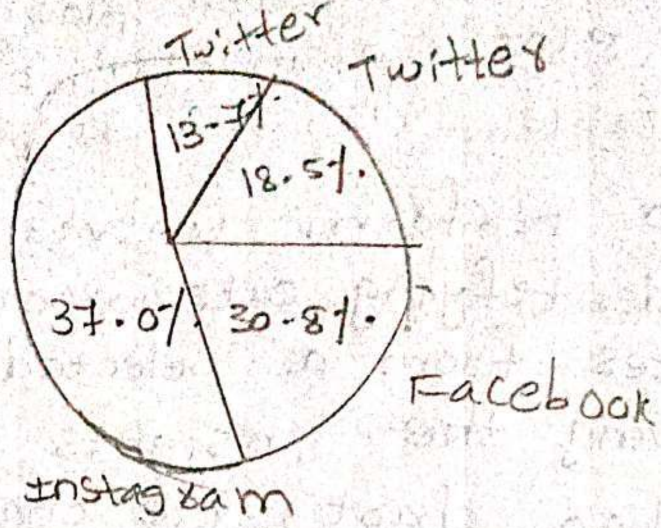
Aim: To perform univariate analysis by identifying categorical and contignous attributes from a selected dataset and upsualozing the insights using plots such bar chart, pie chart, scatterplot, line plot, strip plot and swarm plot.

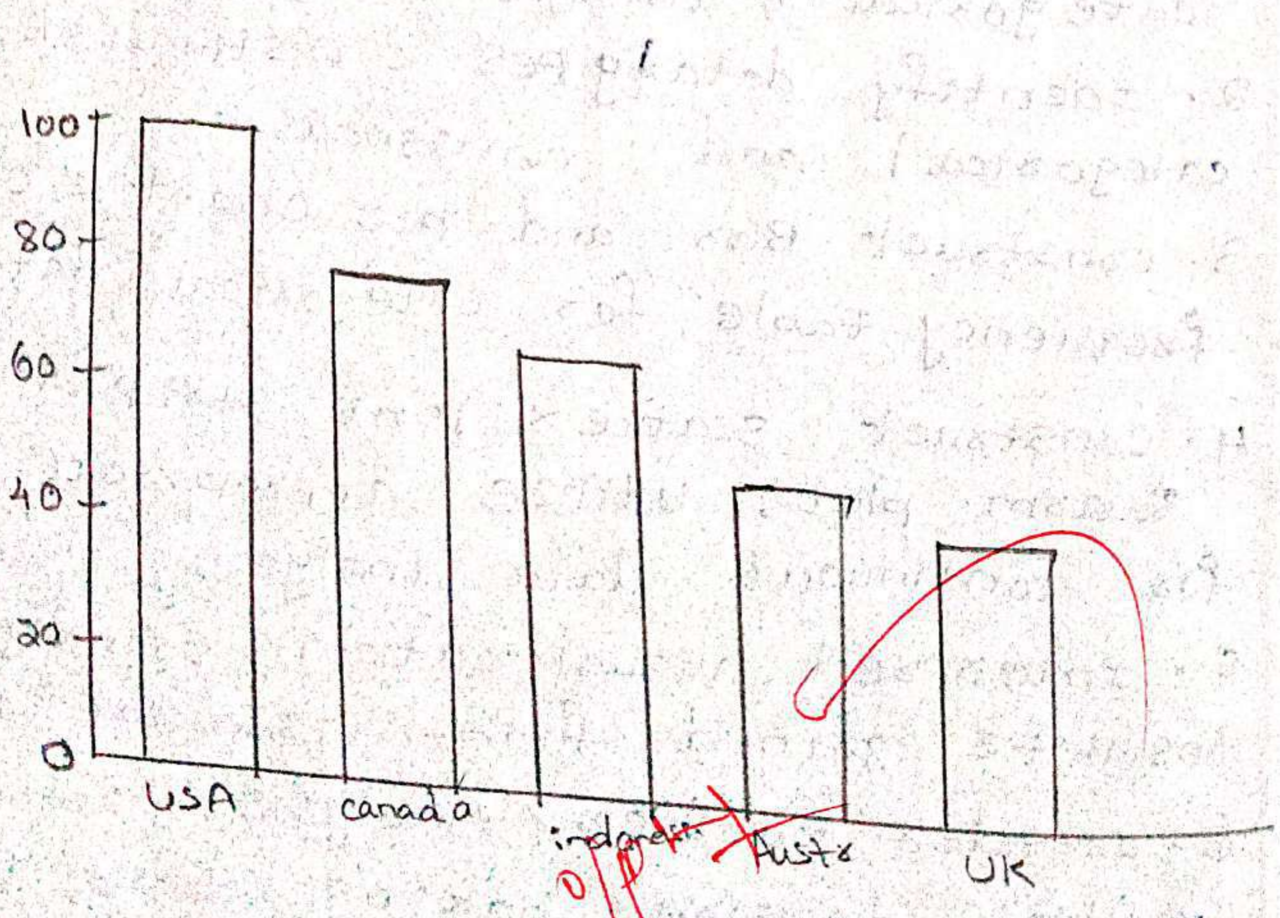
Algorithm:

1. choose Dataset : select a dataset with categorical & contignous attributes
2. Identify datatypes : Distinguish b/w categorical and contisnous
3. construct Bar and pie chart : create frequency table for categorical data.
4. construct scatter, line, strip and swarm plot : utilize appropriate plots for continuous data analysis
5. Interpret visualizations : Analyze insights gained from visualizations.
6. consider Further Analysis : summarize reflect decision-making base the analysis.
7. Document Findings : Summarize findings and observations for reporting or presentation.

output:

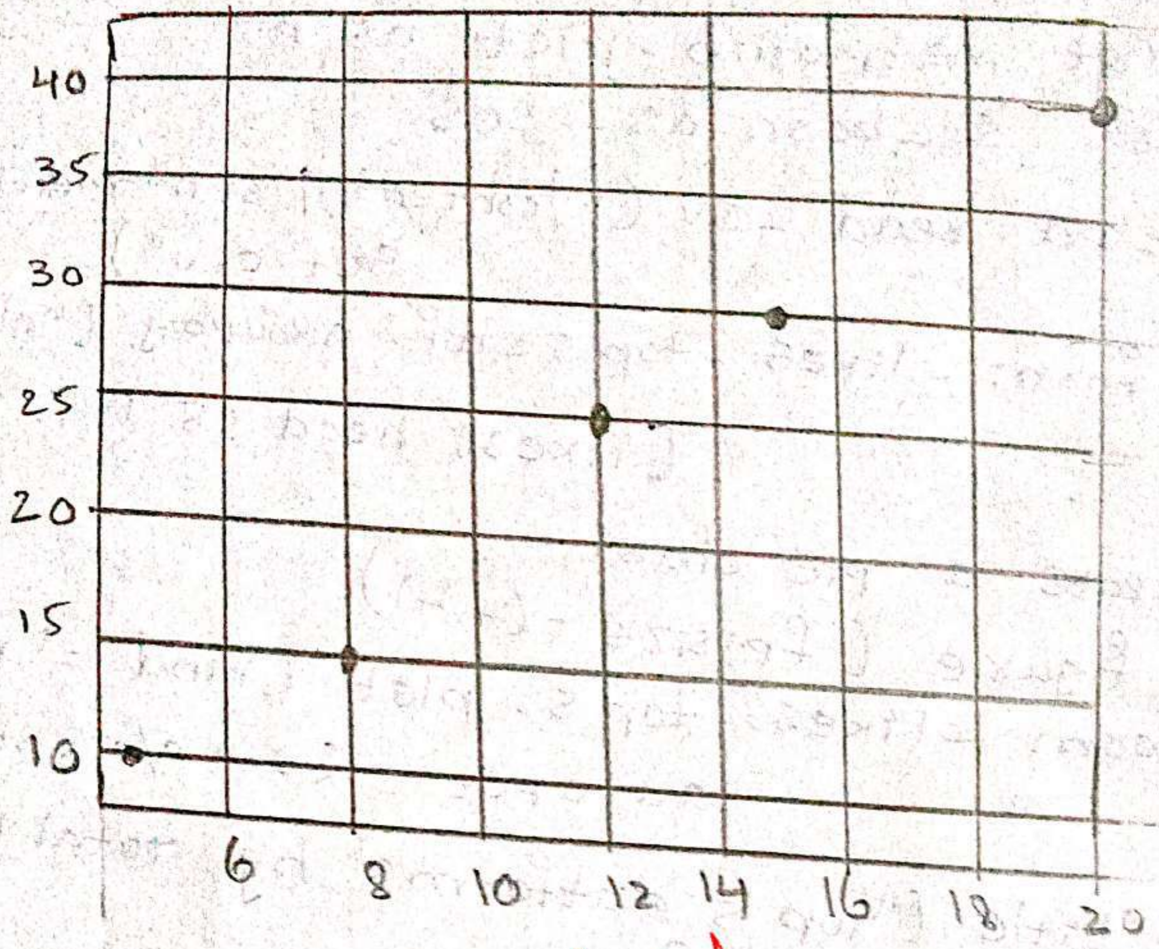
Top 5 platforms by Total likes





Program:

```
import numpyas.np
import pandas as pd
import matplotlib.plot as plt
import Seaborn as sns
df=pd.read_csv("/content/sentiment data
                    set.csv")
plot form_likes_top 5=df.groupby ("plotform')
                    [likes]' head (5)
# create a pie chart
plt.figure (figsize =(4,4))
platform_likes.top 5.plot (kind ='ppe',
                    autopct "{0.1 o/o _startangle
plt.title ("Top 5 platforms by Total likes)
plt.y label ("')
plt.show ()
top 5_countries = df.n largest (5,'likes)
#create a bar chart
plt.figure (figsize = (10,6))
plt.bar [top 5-countries ['country'],
plt.xlabel ('country')
plt.y label ("Total likes')
plt.title ("Top 5 countries Total likes)
plt.show ()
data = {
'Text' : ['Enjoying a beautiful day at
the park' Traffic was terrible this
morning. 'Just finished an amazing
workout', 'excited about upcoming
weekend gateway',
```



40
35
30
25
20
15
10
6
8
10
12
14
16
18
20

```
'Retweets': [15, 5, 20, 8, 12],
'likes' : [30, 10, 40, 15, 25]'
}
df = pd. Dataframe (data)
# set the Seaborn style
plt. figure (figsize = (10,6))
sns. scatterplot (x='retweets', y='Likes',
        data=df. color='blue', 'alpha'=0.7)
plt. title ('scatter plot of Retweets vs Likes')
plt. show()
```

Result: thus to perform univariate calculus analysis by identifying categorical and contignous attributes and visualizing the insights are done by executed Successfully.

28/7/25

Task-2b

Aim:

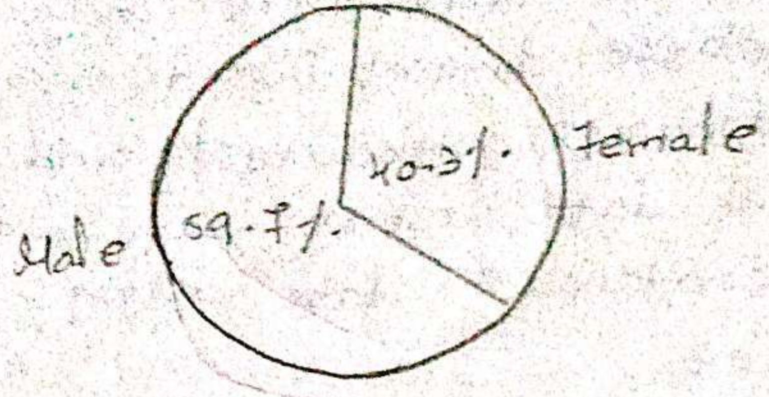
To perform univariate analysis on a given employee dataset by identifing categorica and contignuous attributes and visualizin categorical data using bar and pie chart continous data using instasum, rus plot.

Algorithm:

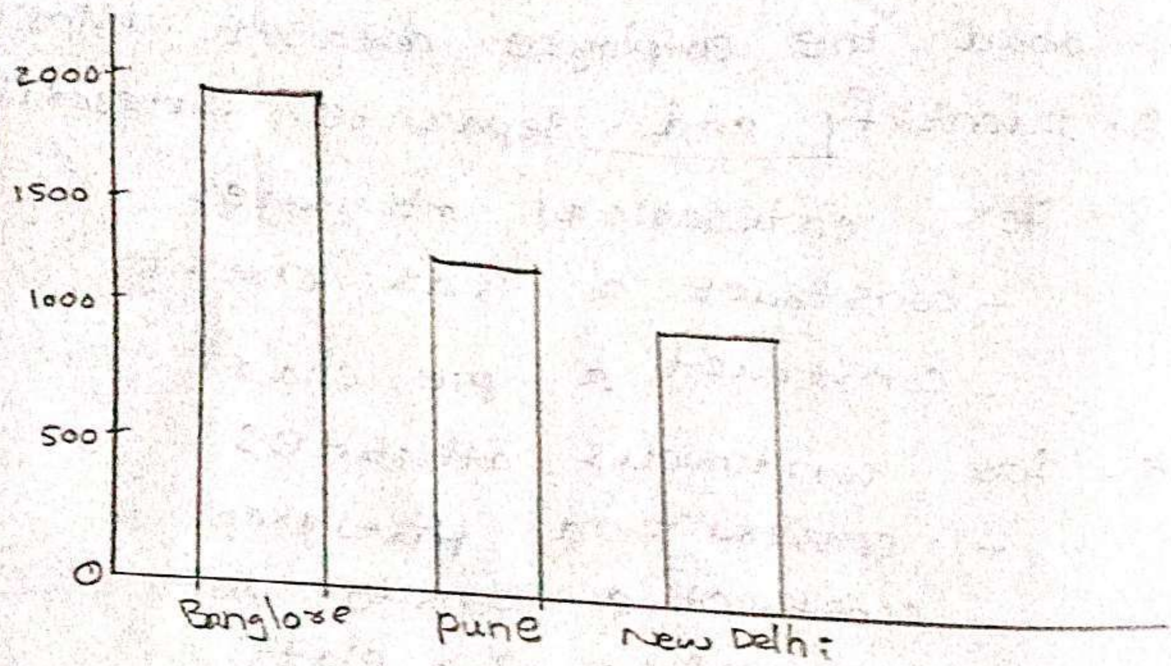
1. Load the employee dataset using pandas
2. Identify and separate categorical.
3. For categorical attributes
   - construct a Bar chart
   - construct a pie chart
4. for continuous attributes
   - construct a histogram
   - construct a Density plot
   - construct a Rug Plot.
5. Display and interpret each visualization.

output:

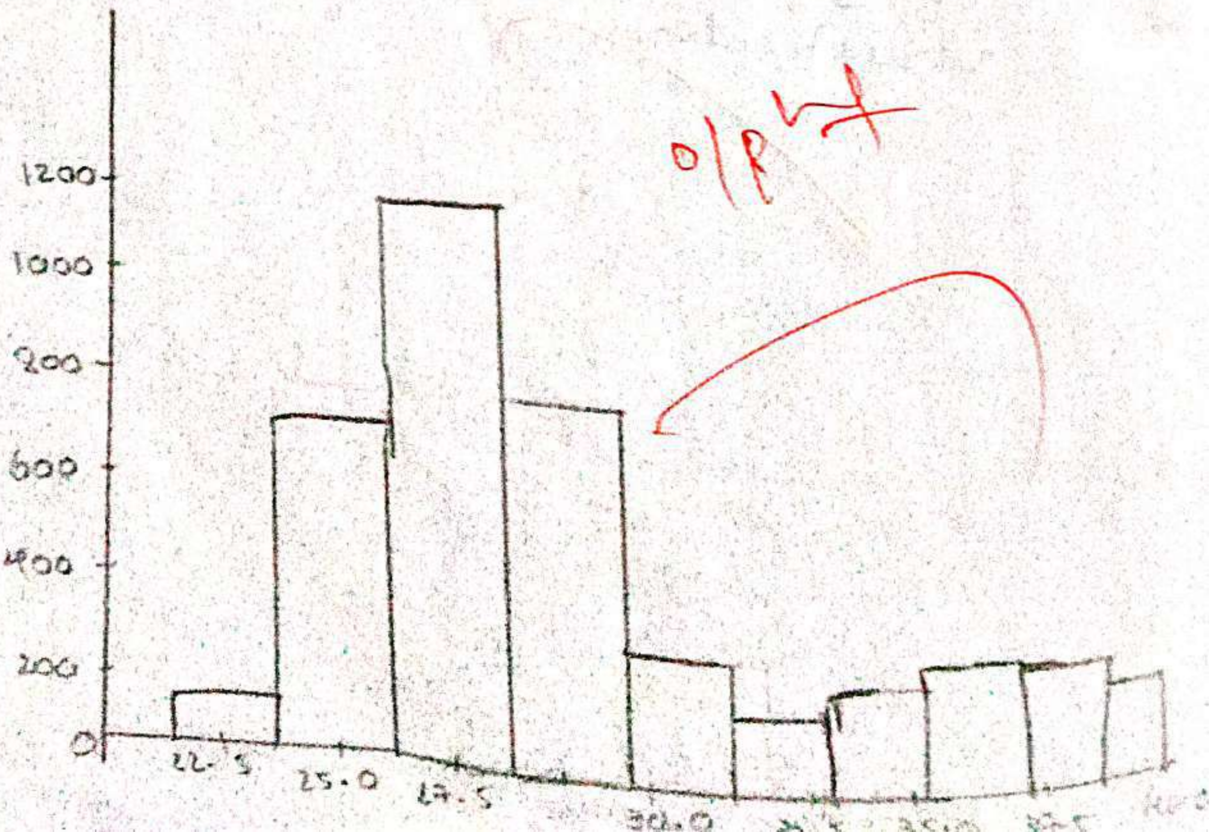
Piechart of Gender



Bar chart of city



Age Distribution

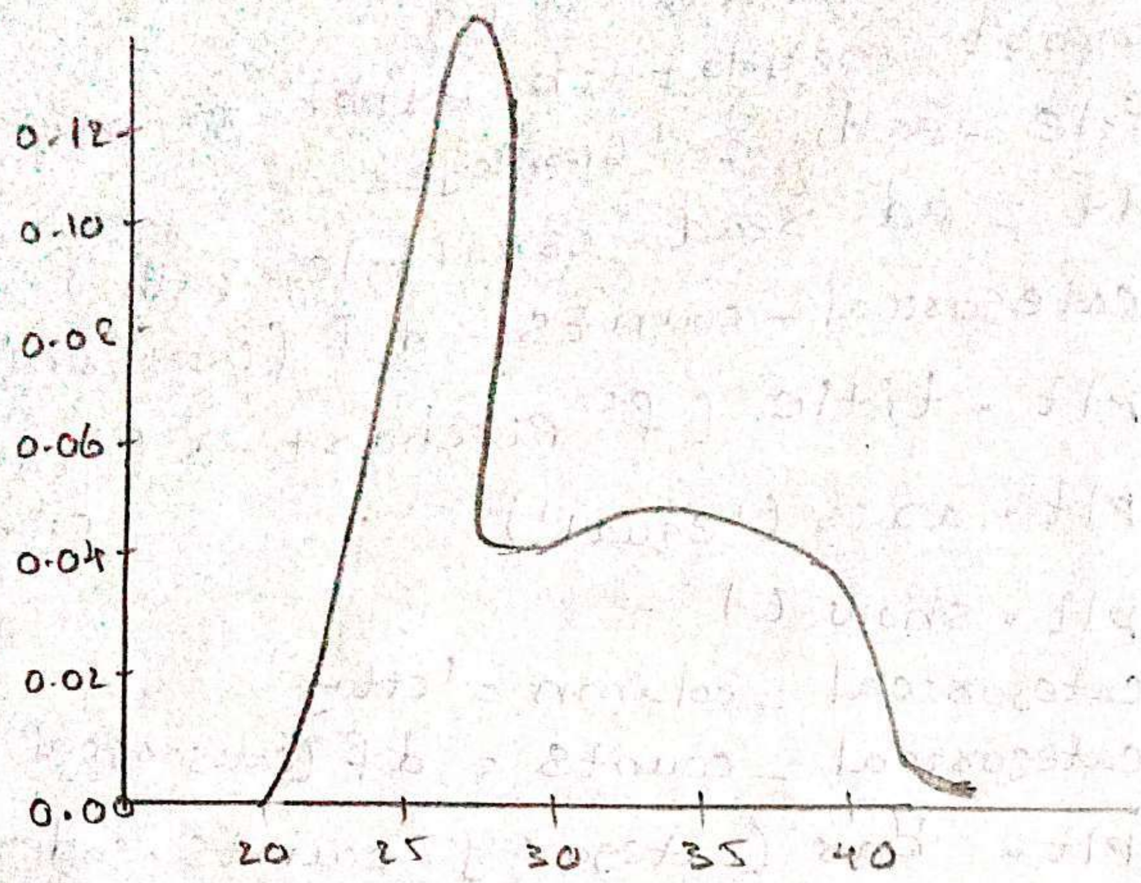


o/p

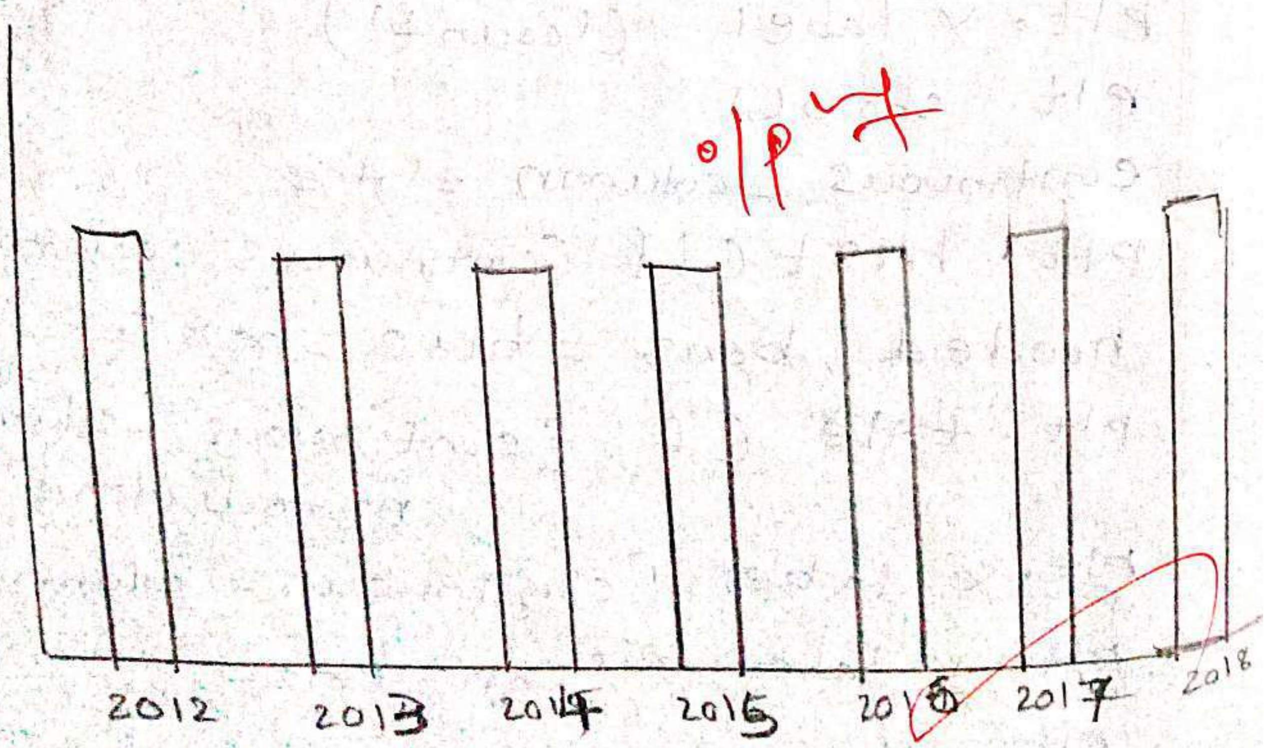
Program:

```
import Seaborn as sns
import pandas as pd
import matplotlib pyplot as plt
file_path = 'Employee.csv'
df = pd.read_csv(file_path)
categorical_counts = df(categorical_column)
plt.title(f'piechart of categorical_column}')
plt.axis('equal')
plt.show()
categorical_column = 'city'
categorical_counts = df[categorical_column]
plt.bar(category_counts.index, category_counts =['skyblue' 'light coral,'orange']
plt.title(f'Bar chart of {categorical_column})
plt.xlabel(categorical_column)
plt.ylabel('count')
plt.show()
continuous_column = 'Age'
plt.hist(df[continuous_column]
needed, bens = blue_bars
plt.title(f'{continuous_column} Distribution')
plt.xlabel(continuous_column)
plt.ylabel('Frequency')
plt.show()
```

Density plot of Age



Rug plot of Joining Year



```
continuous - column ='Age'
sns. kdeplot (df [continuous _column]
plt. title (f' Density plot of {continuous
                                column}'
plt. xlabel (continuous - column )
plt. y label ('Density')
plt. show ()
continuous - column ='Joining Year'
sns. rugplot (df [continuous - column]
plt. title ('f Rug plot as {continuous-
                              column}')
plt. xlabel (continuous - column)
plt. y ticks ([])
plt. show ()
```

VELTECH	
EX No	2
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	3
RECORD (5)	3
TOTAL (20)	16
SIGN WITH DATE	

Result: Thus to perform univerate analysis on a given employee dataset by identifing categorical and continuous attributes and visualizing are done and executed successfully.

04/8/25

Task-3

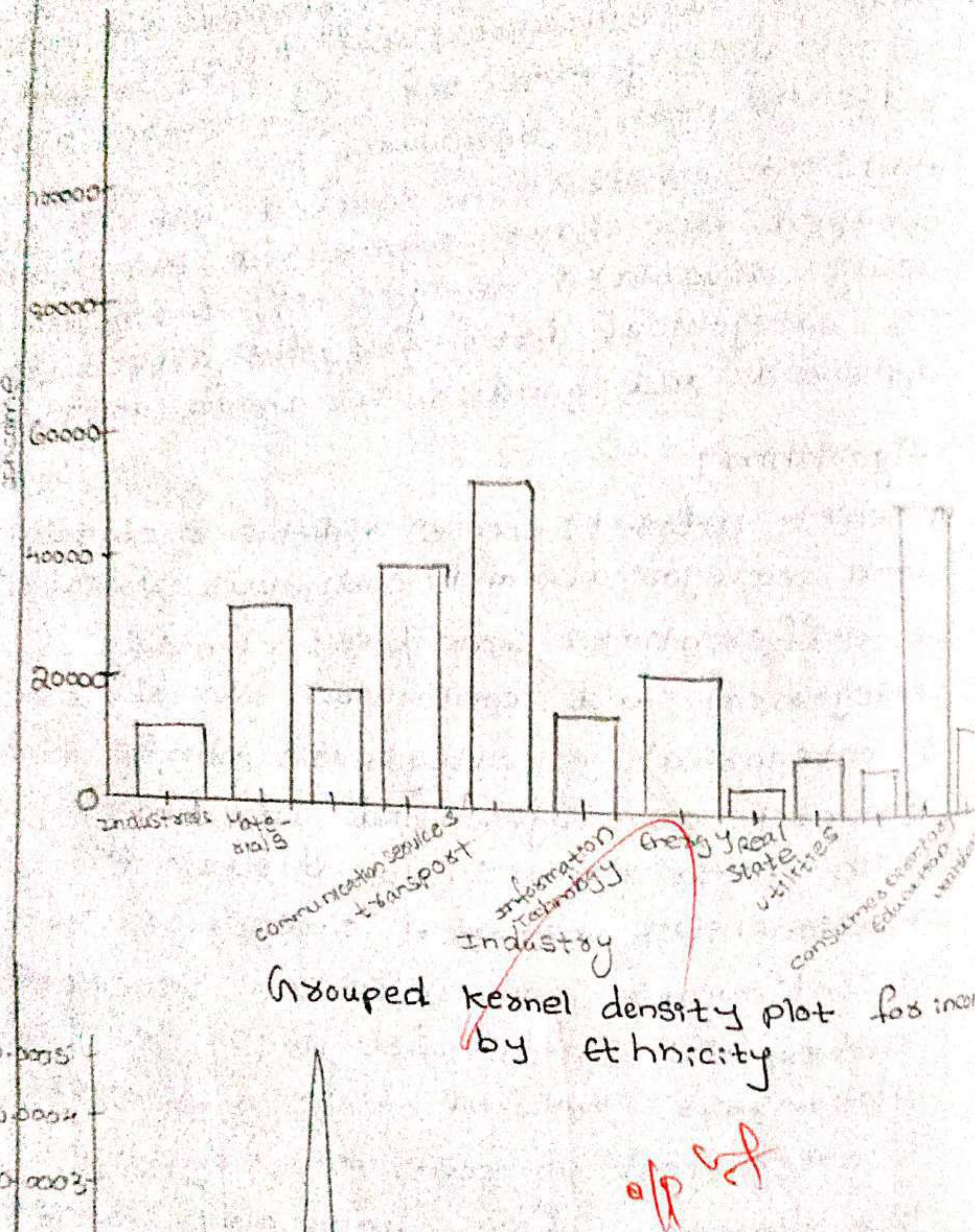
To visualize and perform Bivariate analysis using continuous and categorical data.
categorical vs. categorical : Stacked Bar chart, Grouped Bar chart, segmented Bar chart, Mosaic plots.
continuous vs. continuous : Scatterplot Fit lines
categorical vs. continuous : Bar chart, Grouped kernel density plots, Box plots, violin plots, Ridgeline plots, Beeswarm plots.

Aim: To construct a Stacked Bar chart, Grouped Bar chart, segmented Bar chart using Bivariate analysis of categorical vs categorical data for the attributes of approved and gender in above data set.

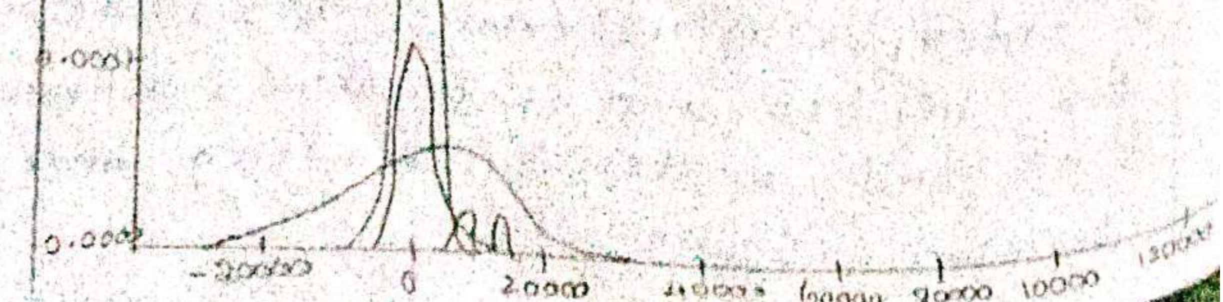
Algorithm:

1. Select Dataset: choose a dataset containing both categorical and continuous variables.
2. Differentiate variables: Identify categorical and continuous variables.
3. categorical vs. categorical: create stacked, grouped or segmented bar charts, along with mosaic plots, to visualize relationships between categorical variables.
4. Continuous vs. continuous : Generate Scatterplots with fit lines to explore relationships between two variables.
5. categorical vs. continuous : construct bar charts for summary statistics grouped kernel density plots.
6. Interpretation: Analyze visualizations for insights into relationships, distributions and patterns, drawing and

Income



Grouped kernel density plot for inco by ethnicity



o/p

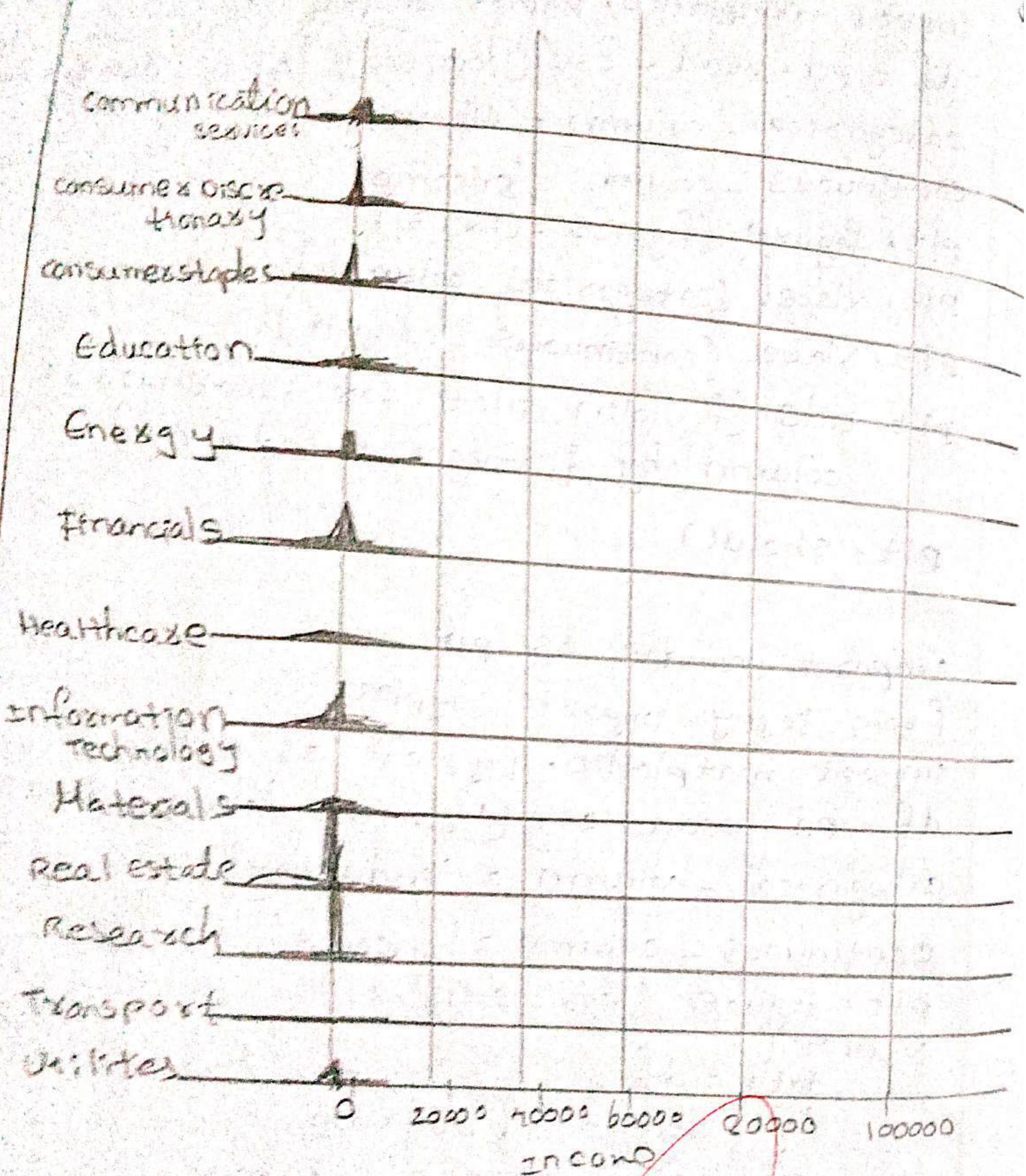
Program:

```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv('/content/clean_dataset.csv')
industry = df('industry')
income = df('income')
plt.figure(figsize = (10, 6))
plt.bar(industry, income, color='skyblue')
plt.xlabel('industry')
plt.ylabel('income')
plt.title('income by industry')
plt.xticks(rotation = 45)
plt.show()
import pandas as pd
import seaborn as sns
import matplotlib.pylot as plt
df = pd.read_csv('/content/clean_dataset.csv')
categorical_column = 'Ethnicity'
continuous_column = 'Income'
plt.figure(figsize = (12, 8))
sns.kdeplot(data = df, x = continuous_column,
  hue = categorical_column, fill = True, common_
  norm = False)
plt.xlabel(continuous_column)
plt.title(f'Grouped kernel Density plot for
plt.legend(title = categorical_column)
plt.show()
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
plt.xlabel(categorical_column)
plt.ylabel(continuous_column)
plt.title(f'Grouped Box plot)
plt.show()
```

```
import pandas as pd
import Seaborn as sns
import matplotlib. pyplot as plt
df = pd. read - csv ('/content /clean - dataset.csv')
categorical - column = 'Ethnicity'
continuous - column = 'Income'
plt. figure (figsize = (12, 8))
plt. xlabel (categorical - column)
plt. ylabel (continuous - column)
plt. title (f'violin plot for {continuous -
      column} by {categorical - column}')
plt. show()
```

```
import pandas as pd
from joypy import joyplot
import matplotlib. pyplot as plt
df = pd. read - csv ('/content /clean - dataset.csv')
categorical - column = 'industry'
continuous - column = 'Income'
plt. figure (figsize (12, 8))
joyplot (
      data = df,
      by = categorical - column,
      column = continuous - column,
      kind = 'kde',
      fill = True,
      linecolor = "black",
       grid = True,
       linewidth = 1,
       legend = True,
       )
    plt. xlabel (continuous - column)
    plt. title (f'Ridseline plot for
```

Ridgeline plot for income by Industry



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df=pd.read_csv('/content/ovtask_4.csv')
sns.lmplot(d
plt.show()
```

VEL TECH	
EX No.	3
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	4
RECORD (5)	4
TOTAL (20)	14+4=18
... WITH DATE	

4/8/25

Result: Hence, to visualize and perform Bivariate analysis using cantinous and categorical data is Successfully completed

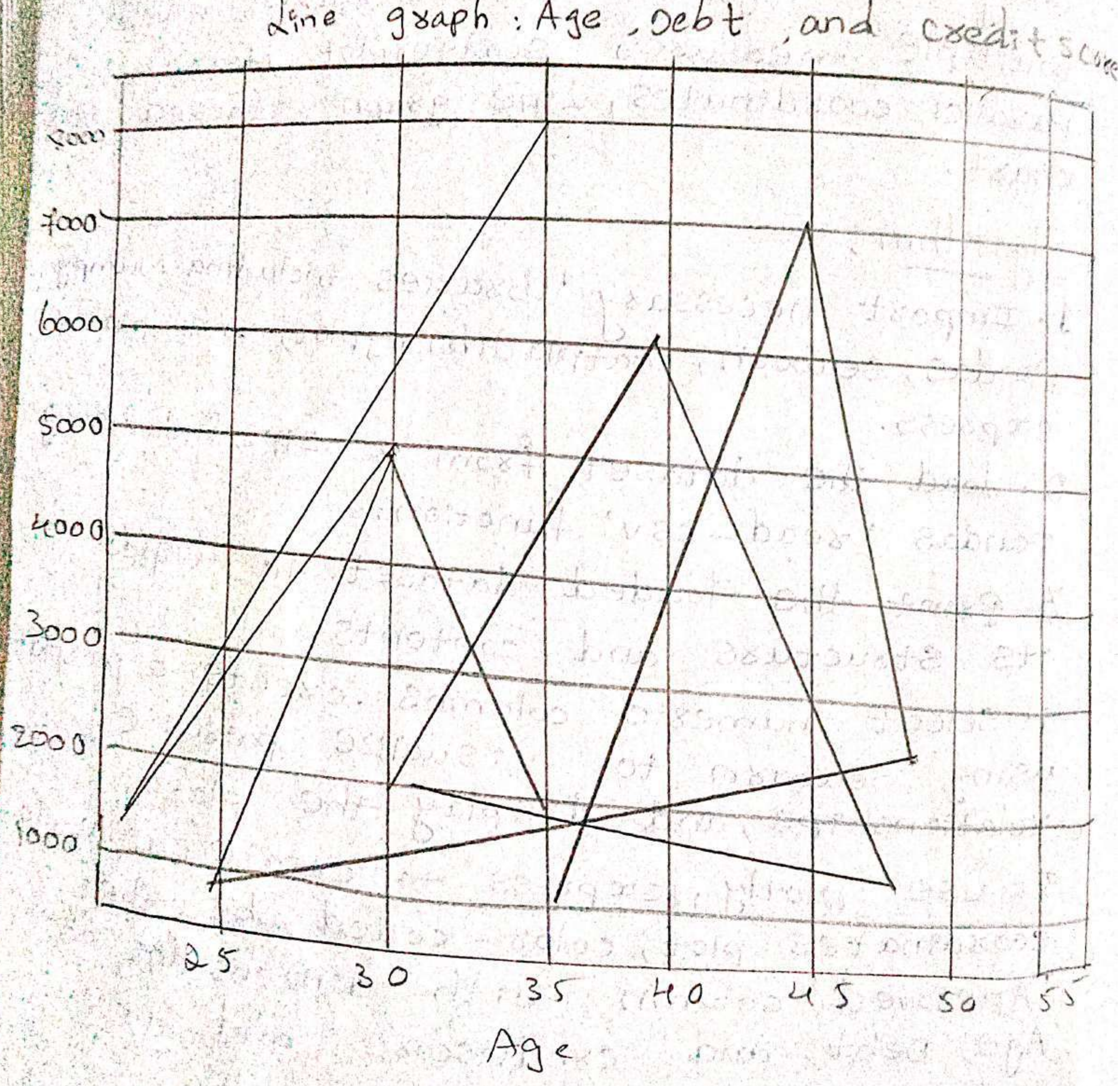
Task-4

To visualize and perform multivariant analysis using multiple variables involving multiple measures.

Aim: To visualize and perform multivariant analysis using multiple variables involving multiple measures Scatterplot Matrix, parallel coordinates, Line graph, stacked Bar chart.

Algorithm:

1. Import necessary libraries including Numpy, Pandas, seaborn, matplotlib, pyplot, and plotly, express.
2. Load the dataset from a zip file using pandas 'read_csv' function.
3. Print the loaded dataset to inspect its structure and contents.
4. Select numeric columns, create a pairplot using seaborn to visualize pairwise relationships, and display the plot.
5. use plotly, express to create a parallel coordinates plot, color-coded by the 'Approved' column, with dimensions as Age, Debt, and creditscore, and show the plot.
6. Extract the first 20 entries of dataset and plot the Age against Debt.
7. Group the first 20 entries by Age, summing up Debt and creditscore for each Age group, create a stacked bar chart.


Line graph : Age , Debt , and credit score
8000
7000
6000
5000
4000
3000
2000
1000
25
30
35
40
45
50
55
Age

Program:

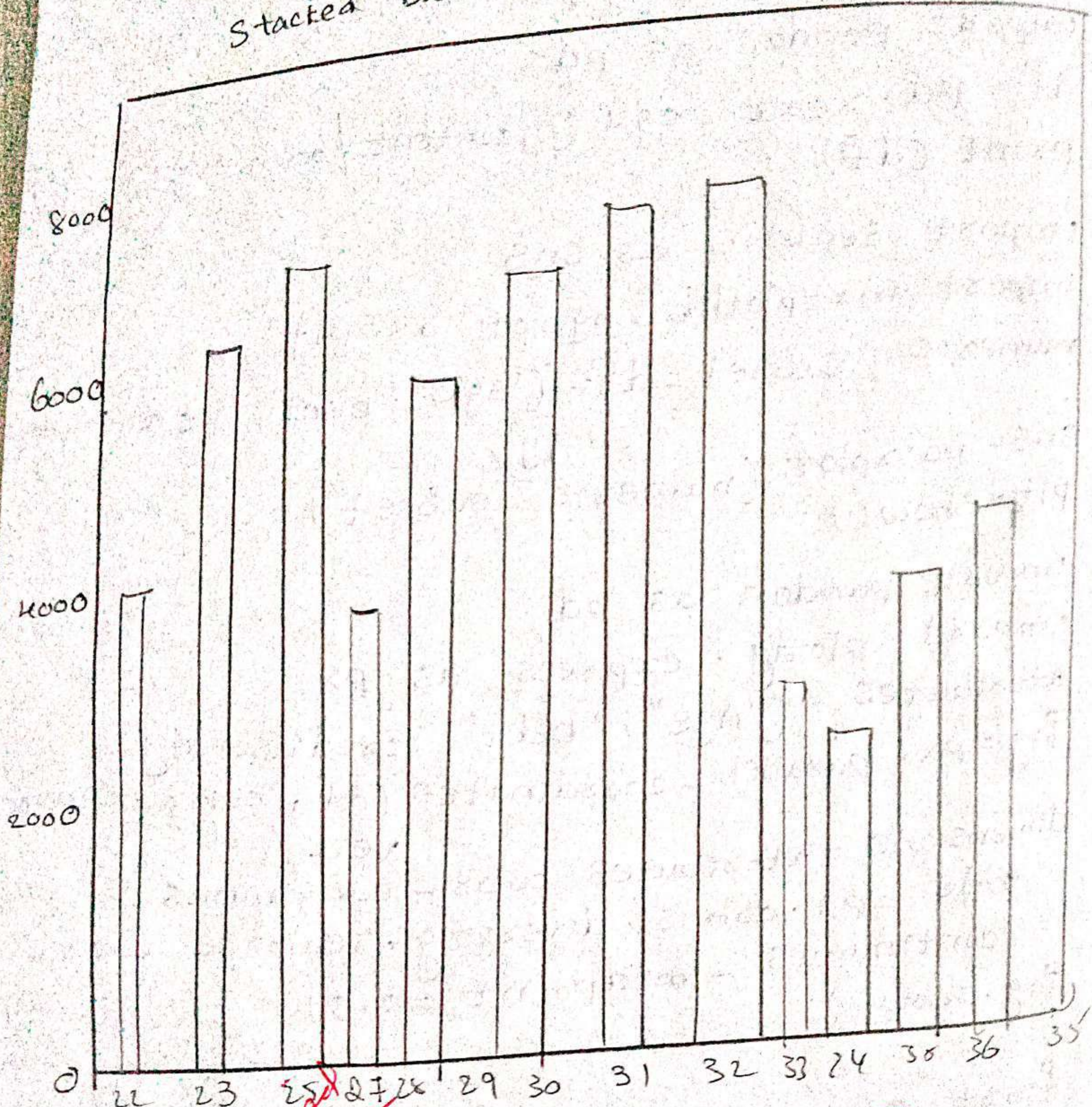
```
import numpy as np
import pandas as pd
df = pd.read_csv ("/content/archieve(4).zip")
Print (df)

import seaborn as sns
import matplotlib.pyplot as plt
numeric_subset = df ([['Age', 'Debt', 'yearEmployed']]
sns.pairplot (numeric_subset)
plt.show()

import pandas as pd
import plotly.express as px
attributes = ["Age", "Debt", "creditscore"]
fig = px.parallel_coordinates (df, color = "Approved",
dimensions = attributes, color_continuous_
    scale = px.colors.diverging.Tealrose, color_
    continuous_midpoint = 0.5)
fig.show()

first_20_entries = df.head (20)
age = first_20_entries ('Age')
debt = first_20_entries ['Debt']
credit_score = first_20_entries ['credit
                                  score']
plt.figure (figsize = (10, 6))
plt.plot (age, debt, label = 'Debt', color = 'blue')
plt.plot (age, credit_score, label = 'credit
                score', color = 'green')
plt.xlabel ('Age')
plt.ylabel ('value')
plt.title ('Line graph : Age, Debt, and
                        credit score')
plt.legend ()
plt.grid (True)
```

Stacked Bar chart debt and credits



o/p verified

```
grouped_data = first_20_entries.groupby
      ('Age').sum()[['Debt', 'credit score']]
grouped_data.plot(kind='bar', stacked=
            True, figsize=(12,8))
plt.xlabel('Age')
plt.ylabel('value')
plt.title('Stacked Bar chart - Debt and credit
        score by Age')
plt.legend(title='Attribute')
plt.show()
```

VEL TECH	
EX No.	4
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	3
VIVA VOCE (5)	3
RECORD (5)	13+3=16
TOTAL (20)	16
SIGN WITH DATE	

11/8

Result: Thus, visualize and perform multi-variant analysis using multiple variables involving multiple measures is successfully executed.

18/9/25

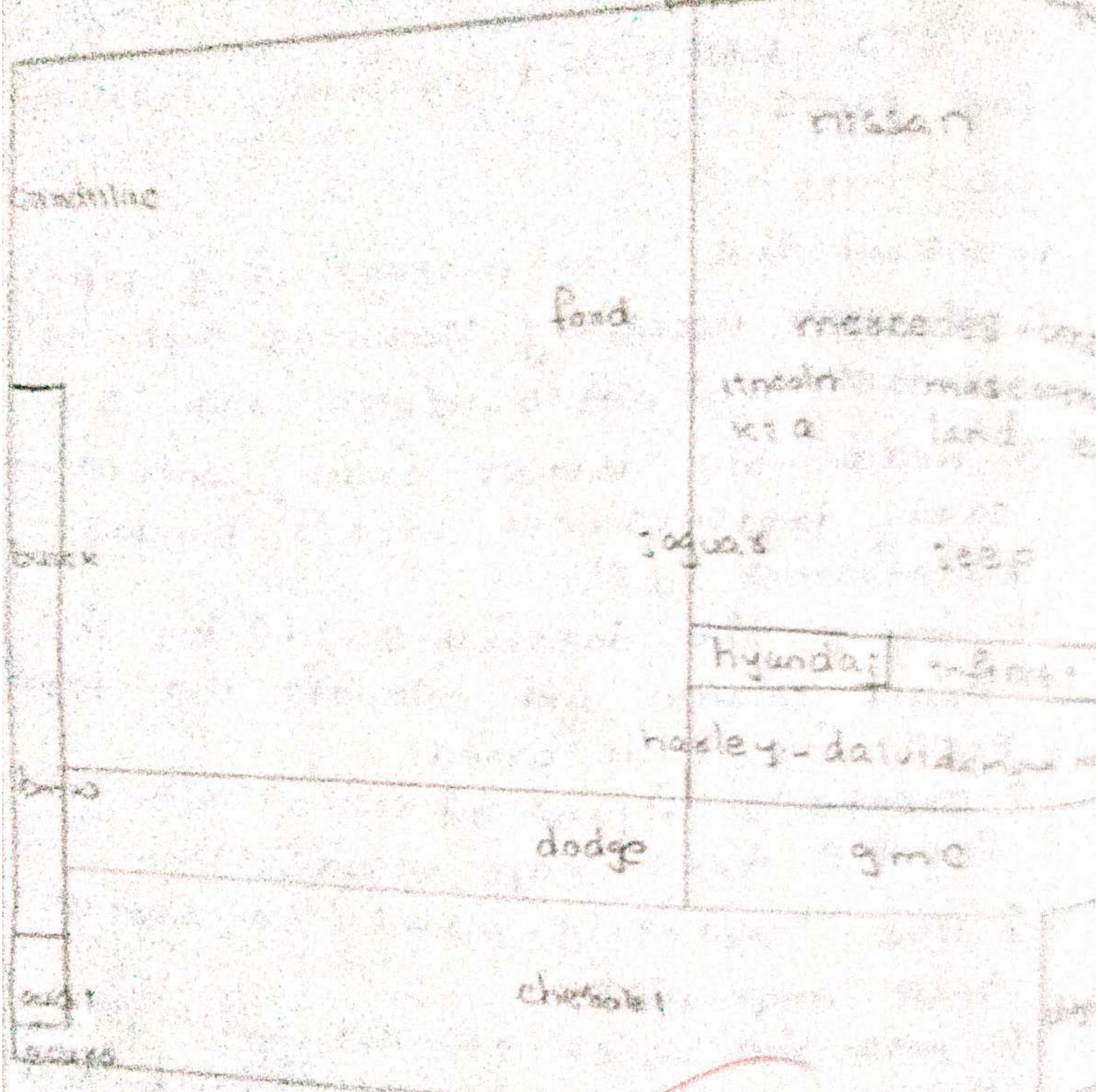
To design and perform visu... Trees. Tree Map, sun Burst Display.

Aim: To design and perform visualization for Trees.

Algorithm:

1. Install the 'squa' package using pip.rify
2. Import necessary libraries such as 'pandas', 'matplotlib'.pyplot', and 'squarify'.
3. Read the dataset from '/content/USA-cars_datasets.csv' into a pandas Dataframe 'df'.
4. Group the dataframe 'df' by the 'brand' column and calculate the total price for each brand.
5. Create a figure of size 18x12 inches for the tree map visualization.
6. Use 'squarify.plot()' to create a tree map plot.
7. pass the sizes of the rectangles as the total price of each brand and label as the brand names.
8. Set transparency with alpha = 0.4 for better visualization.
9. Turn off the axis
10. Set the title of plot as 'Tree Map of Total price by Brand".
11. Display the plot using plt.show().

Output: Tree Map of total price by Brand



Program:

```
pip install squarify
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import squarify
df = pd.read_csv('/content/USA_cars_datasets')
print(df)
brand_total_price = df.groupby('brand')
    ['price'].sum().reset_index()
plt.figure(figsize=(18, 12))
squarify.plot(sizes=brand_total_price
              ['price'], label=brand_total_
              price['brand'], alpha=0.4)
plt.axis('off')
plt.title('Tree Map of Total price by Brand')
plt.show()
```

Result: Thus, design and perform visualization for trees is executed successfully.

Build a sunburst display using above program dataset.

Aim: To build a sunburst display using above program dataset.

Algorithm:

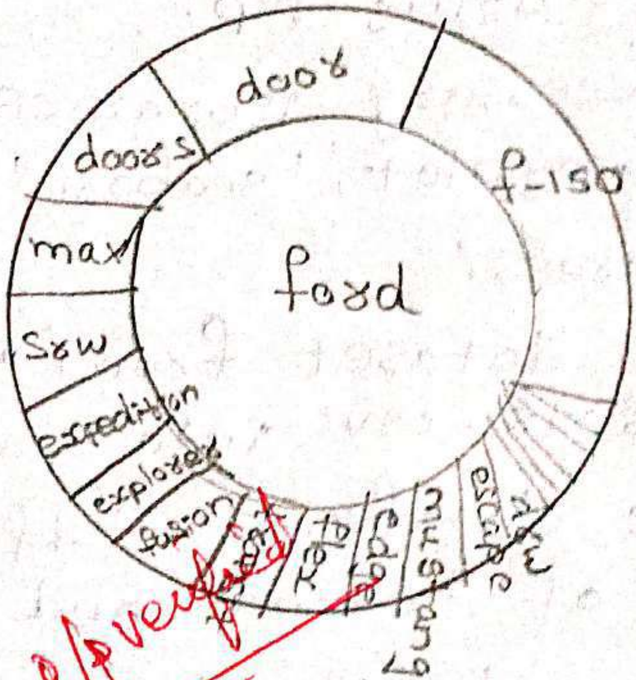
1. Install the required packages 'squarify' and 'plotly' using pip.
2. Import necessary libraries: 'pandas', 'numpy', 'matplotlib.pyplot', 'seaborn', 'squarify', and 'plotly.express.'
3. Read the dataset from '/content/USA-cars-datasets.csv' into pandas.
4. Group the dataframe 'df' by the combination of 'brand' and 'model' columns
5. create a sunburst plot using plotly Express ('px.sunburst()'):
   - specify the dataframe 'sunburst-data' as the data source.
   - set the path for the sunburst plot to follow the hierarchy of 'brand' and 'model'.
   - Define 'price' as the values to be represented
   - set the title of the plot as 'Sunburst Display of Total price'.
6. Display the plot using 'fig.show()'.

Program:

```
pip install squarify
pip install plotly
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Output:

Sunburst Display of Total price by Brand and Model



O/p verified

```
        datasets.csv')
print(df)
sunburst_data = df.groupby(['brand', 'model']).
    sum().reset_index()
fig=px.sunburst(sunburst_data, path=['brand',
  'model'], values='price', title='sunburst
  Display of Total price by Brand and
        Model')
fig.show()
```

VELTECH	
EX No.	5
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	4
RECORD (5)	5
TOTAL (20)	14+5=19
SIGN WITH DATE	

18/8

<u>Result</u>: Thus, to build a sunburst display using above program dataset is executed successfully.

To design and perform visualization for Graphs and Networks.
. force based layout.

Aim: To design and perform visualization for Graphs and Networks.

Algorithm:

Import necessary libraries:

Import NetworkX as nx for creating and manipulating network graphs.
Import Matplotlib.pyplot as plt for data visualization.
Import pandas as pd for data handling
Create an empty graph object G and set an attribute 'day' to "stackoverflow".

Read node and edge data from csv files into
DataFrames df-nodes and df-links, respectively.

Add nodes to the Graph G.

Iterate through the rows in df_links.
For each row, add a weighted edge from the 'source' to the 'target' with the 'value' as the weight.
Set visualization parameters:
Create a figure with a specified size,
Define visualization options like edge color, width, node labels, and font weight,
Determine colors for nodes based on their 'group' attribute.
Adjust node sizes based on the 'nodesize' attribute.
Draw the network graph:
Use nx.draw to visualize the graph
customize the node colors and sizes based on the calculated values.

Define the layout of the nodes using spring layout with specified parameters.
set the edge color to specific color
show the graph using plt.show()
End of the algorithm.

Code:

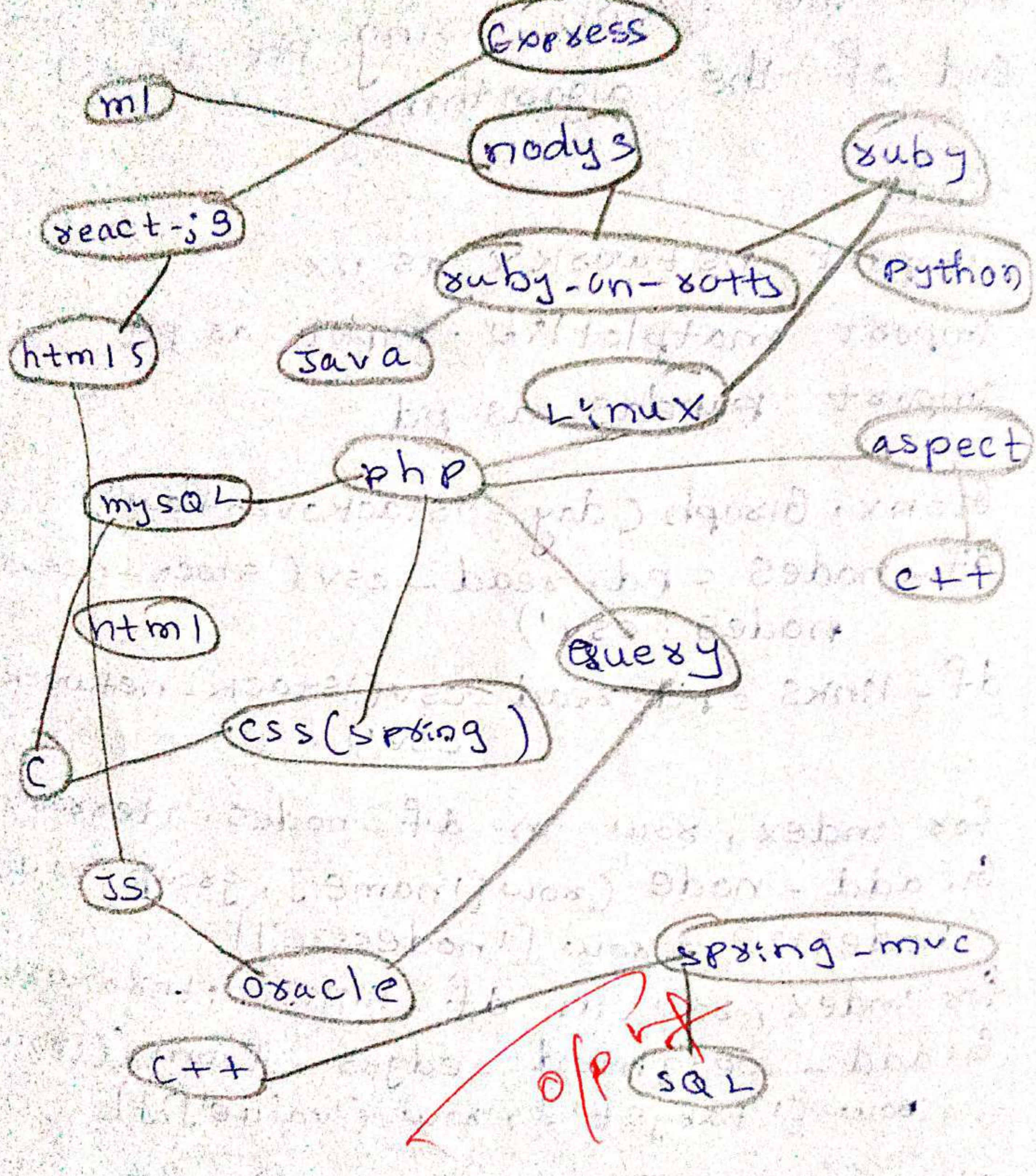
```
import networkx as nx
import matplotlib.pyplot as plt
import pandas as pd

G=nx.Graph(day="stackoverflow")
df_nodes =pd.read_csv('stack_network_nodes.csv')
df_links =pd.read_csv('stack_network_links.csv')

for index, row in df_nodes.iterrows():
G.add_node(row['name'], group=row('group')
  nodesize =row['nodesize'])
for index, row in df_links.iterrows():
G.add_weighted_edges_from([(row['source'
  row['target'], row('value'])])

plt.figure (figsize =(15,15))
options = {
    'edge_color': #FFDEA2',
    'width': 1.5,
    'with_labels': True,
    'font_weight': 'regular':
  }
colors = [color_map(G.nodes(node)['group'])
          for node in G)
sizes = (G.nodes[node]['nodesize']*25 for
                      node in G]
nx.draw(G, node_color =colors, node_size =
          sizes, pos =nx.spring_layout
      (G, k=1.5, iterations =15, **options)
```

output:



Express
ml
nodys
ruby
react-js
ruby-on-rails
python
html5
Java
Linux
aspect
php
mySQL
c++
html
Query
css(spring)
C
JS
spring-mvc
oracle
c++
SQL

o/p

```
ax=plt.gca()
ax.collections[0].set_edgecolor("#555555")
plt.show()
```

VEL TECH	
EX No.	6
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	4
RECORD (5)	5
TOTAL (20)	14+5=19
SIGN WITH DATE	

25/8

Result: Thus, to design and perform visualization for graphs and networks is executed successfully.

08

To generate insight using Text Network Analysis and visualization

7(a) Tag crowd tool using build a word cloud it contains your details.

Aim:

To generate insights from text, data by creating a word cloud visualization using the Tagcrowd approach word clouds visually represent word frequency, where more frequent words appear larger. This helps in quickly identifying key themes and insights from textual data.

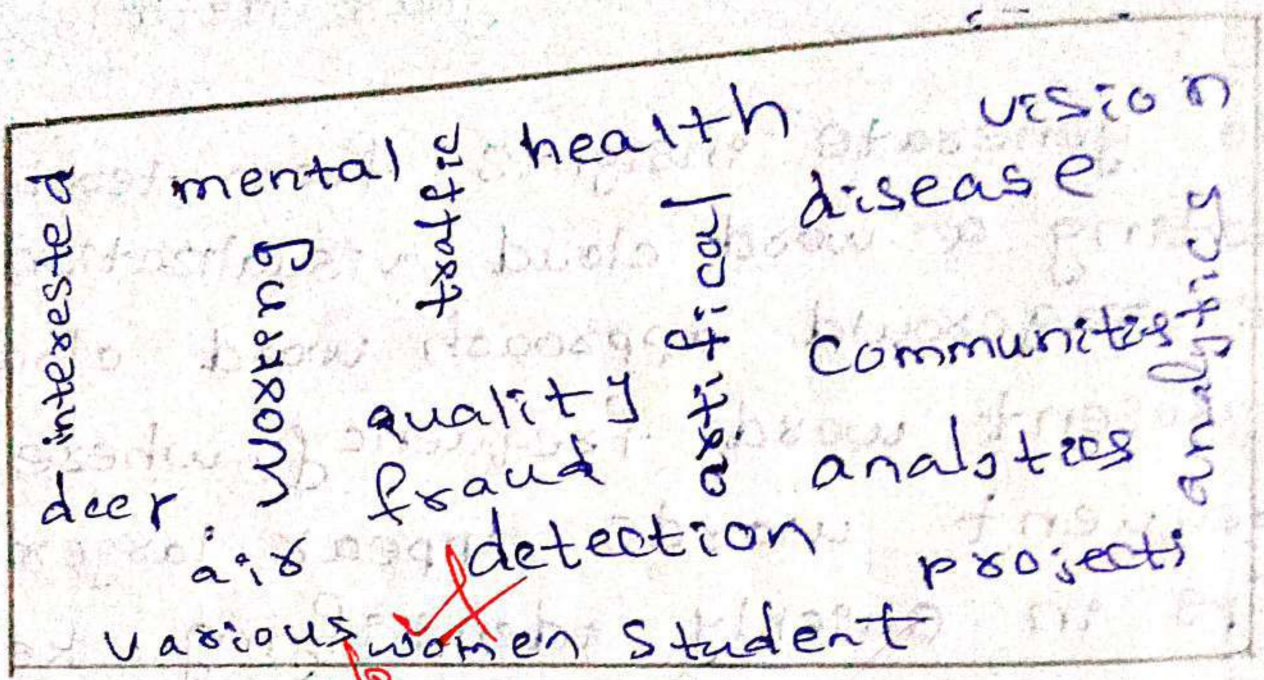
Algorithm:

1. Input data collection : provides a text document.
2. preprocessing : convert text to lowercase Remove punctuation, stop words, and numbers, Tokenize words.
3. word frequency calculation : count occurences of each word.
4. use word cloud library to create the word cloud.
5. Display using Matplotlib.
6. Larger words in the cloud = higher frequency in dataset.

Program:

```
!pip install wordcloud matplotlib nltk
from wordcloud, import wordcloud
import matplotlib.pyplot as plt
import re
text = """ " I am a B.Tech student
```

Output:


interested
mental
traffic
health
vision
disease
working
artifical
communities
analytics
quality
fraud
analstics
deer
air
detection
projects
various
women
Student

plant disease Detection, AI-based mental health Monitoring
Fraud detection in Rural communities, and Air Quality classifier
I am interested in Artificial intelligence, Machine learning, Deep learning, and con vision"""

```
Stop-words = {"i", "am", "a", "the", "is",
"on", "and", "of", "in", "my", "me", "to", "an",
"for", "with", "like"}
text = text.lower()
text = re.sub(r'[^a-z\s]', '', text)
filtered_words = [word for word in text.
  split() if word not in stop_words]
processed_text = " ".join(filtered_words)
wordcloud = WordCloud(width = 800,
  height=400, background_color = "white").
          generate(processed_text)
plt.figure(figsize = (10, 5))
plt.imshow(wordcloud, interpolation
plt.axis("off")
plt.show()
```

Result: thus to generate crowd tool using build a word cloud it contains your details was verified.

08/09/25 64

1b) Try crowd utilize wordtree package to generate the cloud of text and to plot graph using matplotlib library and visualize it.

Aim: To generate the cloud of text and to plot graph using matplotlib library and visualize it

Algorithm:

Import necessary libraries:

Import wordcloud from the wordcloud library for cloud creation.

Import matplotlib.pyplot as plt for data visualization.

Define the text from which you want to create a word cloud.

Set the minimum font size for displayed words

Create a matplotlib figure for displaying the word cloud.

Define the figure size and specify the facecolor.

Generate the word cloud image using the generate method of object, passing in text.

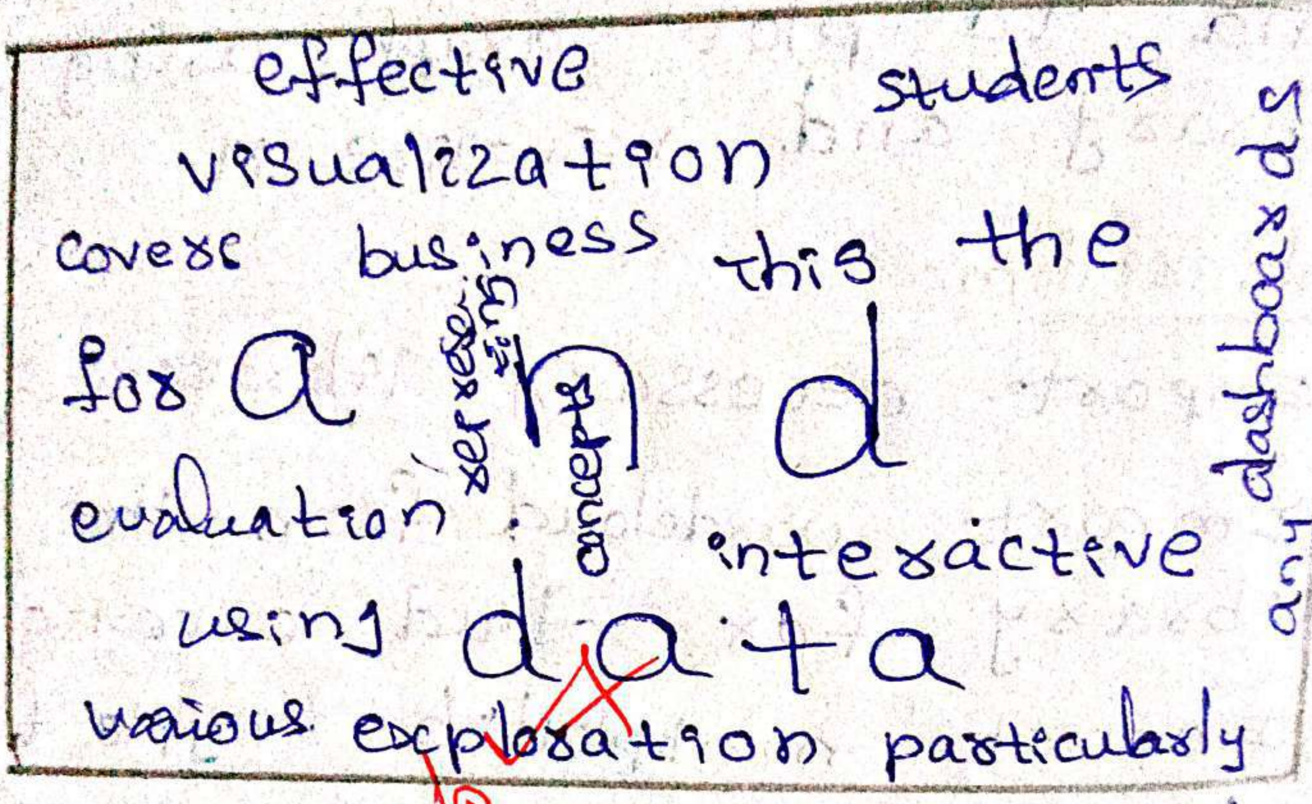
Display the word cloud image

use plt.imshow to display word cloud.

Turn off axis to remove axis and label using plt.show().

end of the algorithm.

output:


effective
students
visualization
covers
business
this
the
for
a
h
d
concepts
dashboards
evaluation
interactive
any
using
data
various
exploration
particularly

o/p

Code:

```
from wordcloud import wordcloud
import matplotlib.pyplot as plt
details = "Data visualization is the visual
and interactive exploration and graphic
evaluate, design and develop effective
visualizations and dashboards, using
various development tools"
wordcloud = wordcloud (width = 800,
   height = 800,
 background_color = 'white',
 stopwords = [],
min_font_size = 10).generate (details)
plt.figure (figsize = (5,5), facecolor = None)
plt.imshow (wordcloud)
plt.axis ("off")
plt.tight_layout (pad = 0)
plt.show()
```

Result:- Thus, the generate the cloud of text and to plot graph using matplotlib library is successfully executed.

Real-time application related to Bivariate

Aim: To analyze their exam scores to evaluate the effectiveness of study habits

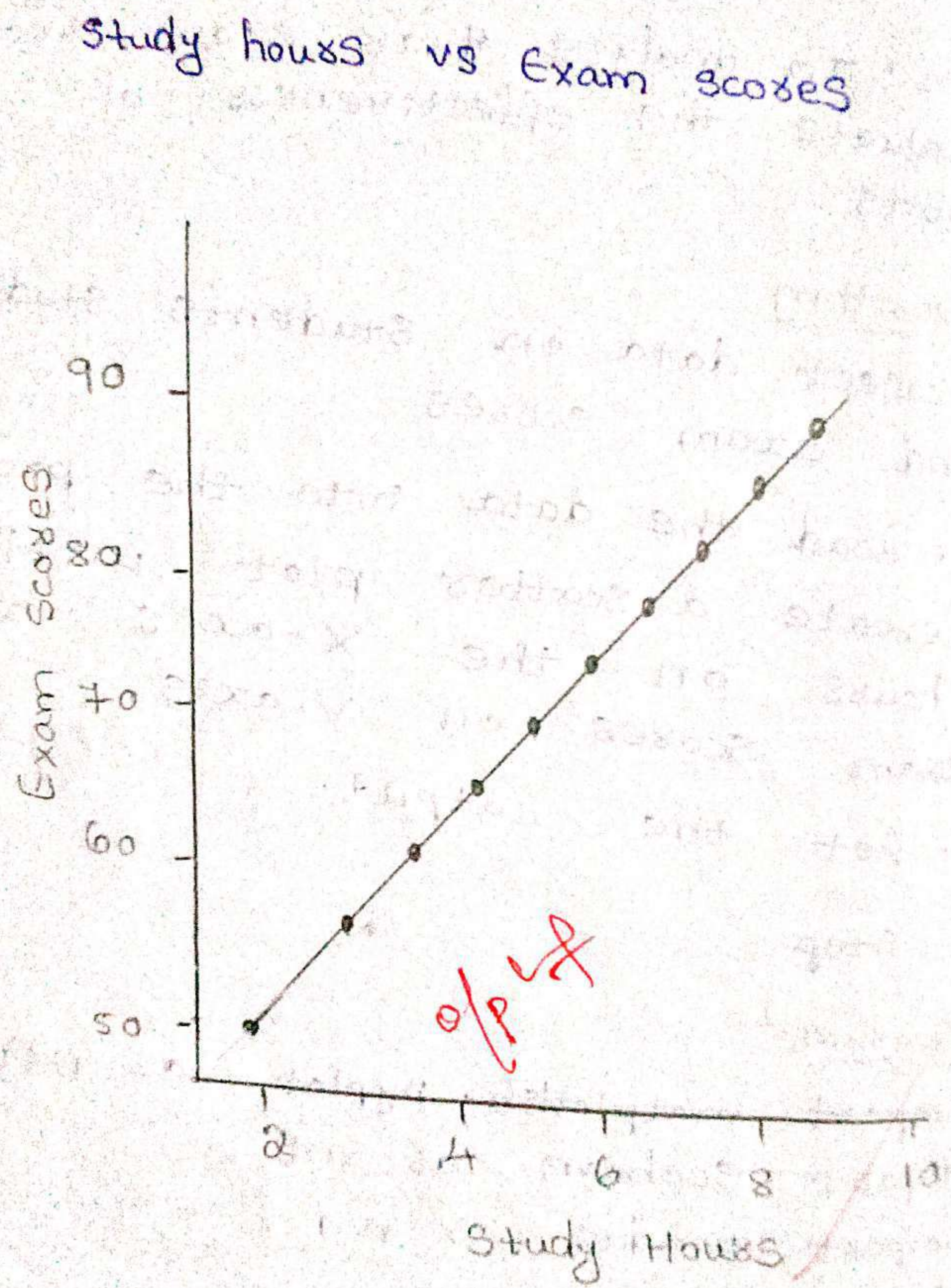
Algorithm:

1. collect data on students study hours and exam scores.
2. Load the data into the program.
3. create a scatter plot with study hours on the x-axis and exam scores on y-axis
4. Get the output.
5. Stop

Program:

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
data = {
    'study_Hours' : [2,3,4,5,6,7,8,9,
                      10,11],
    'Exam_scores': [50,55,60,65,70,75,80,85,90,95]
}
df = pd.DataFrame(data)
sns.lmplot(x='study_Hours', y='Exam_
           scores', data=df, ci=None)
plt.title('study Hours vs Exam scores')
```

output:


Study hours vs Exam scores
90
80
70
60
50
Exam Scores
2
4
6
8
10
Study Hours
o/p ✓

```
plt.xlabel('study hours')
plt.ylabel('Exam scores')
plt.show()
```

VELTECH	
EX No.	7
PERFORMANCE (5)	4
RESULT AND ANALYSIS (5)	4
VIVA VOCE (5)	3
RECORD (5)	3
TOTAL (20)	14
SIGN WITH DATE	

Result: Thus, analyzing the exam scores to evaluate the effectiveness of study habits is verified successfully

To analyze and visualize spatial and geospatial data

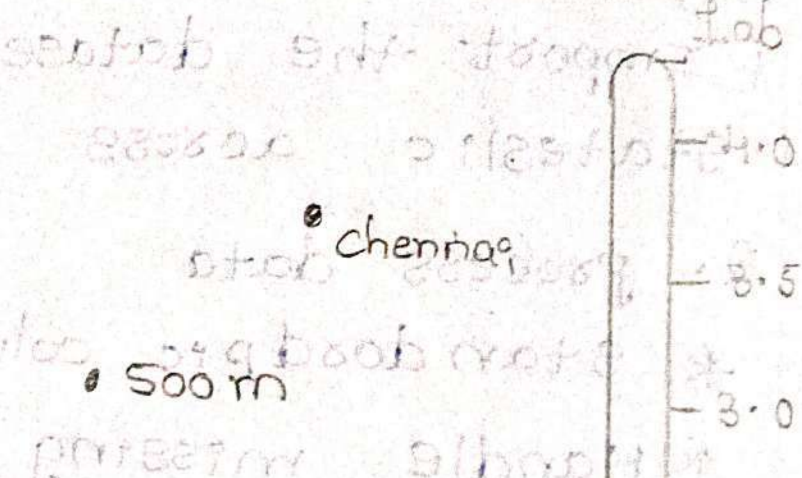
Aim: To analyze the population statistics at a region using spatial and geospatial techniques and to visualize them on maps.

Algorithm:

1. Import the dataset containing population statistic across census years.
2. Process data
   * Standardize column names
   * Handle missing values
   * Extract census year population years
3. Calculate population growth rates b/w census years.
4. Integrate geospatial data
   * Load regional boundary shapefile
   * Merge population data with spatial boundaries.
5. Visualization
   * Create point-based visualizations for location specific values.
6. Analyses
   * Study spatial variations.
   * Observe clustering and distribution patterns.

Output:

Tamilnadu major cities preparation in 991-0



Program:

```
import pandas as pd
import matplotlib as gplt
import folpum
df = pd.read.csv ("population.data.csv")
df.column = ["Region", "status", "2000", 1991",
        "200", "2011"]
df = dropna()
df ["Growth-1991 - 2011"] = ((df("2011"].df
        ("1991")//df ["1991"]) * 100
map.data = gpd.read.file ("region-boundari-
        es shp")
regional.stds = df.groupby ("zone").sum().
        reset.index()
merged = map.data.merge (regional stats,
        o h = "zone")
fig.ax = plt.subplots (figsize = (1018))
merged.plot (column = "2011", cmap = "orRd",
    linewidth = 0.8, ax = ax, edgecolor =
        "08", legend = True)
plt.show()
m = folium.Map (location = [11.0, 78.0],
        Zoom_start = 6)
for - 1 row in df.pterrows ():
    folium.circle marker (
        location = [row ["latitude"],
    row ["longitude"]],
        radius = row ["2011"] |100 000,
        popup = f"" population: {row ['2011']}",
        dor = "blue",
        fill = True,
        fill : True,
        fill. oapacity = 0.6
        ).add = to(m)
```

m.save("population_spatial_map.html")

VEL TECH	
[illegible]	
[illegible]	
[illegible]	
[illegible]	
[illegible]	
[illegible]	
[illegible]	

Result: thus to analyze the population statistics of a program using spatial and geospatial techniques and to visualize the maps is done and executed successfully.

To analyze and visualize time oriented Data

Aim: To analyze and visualize time oriented data using line graph, trend lines, area chart.

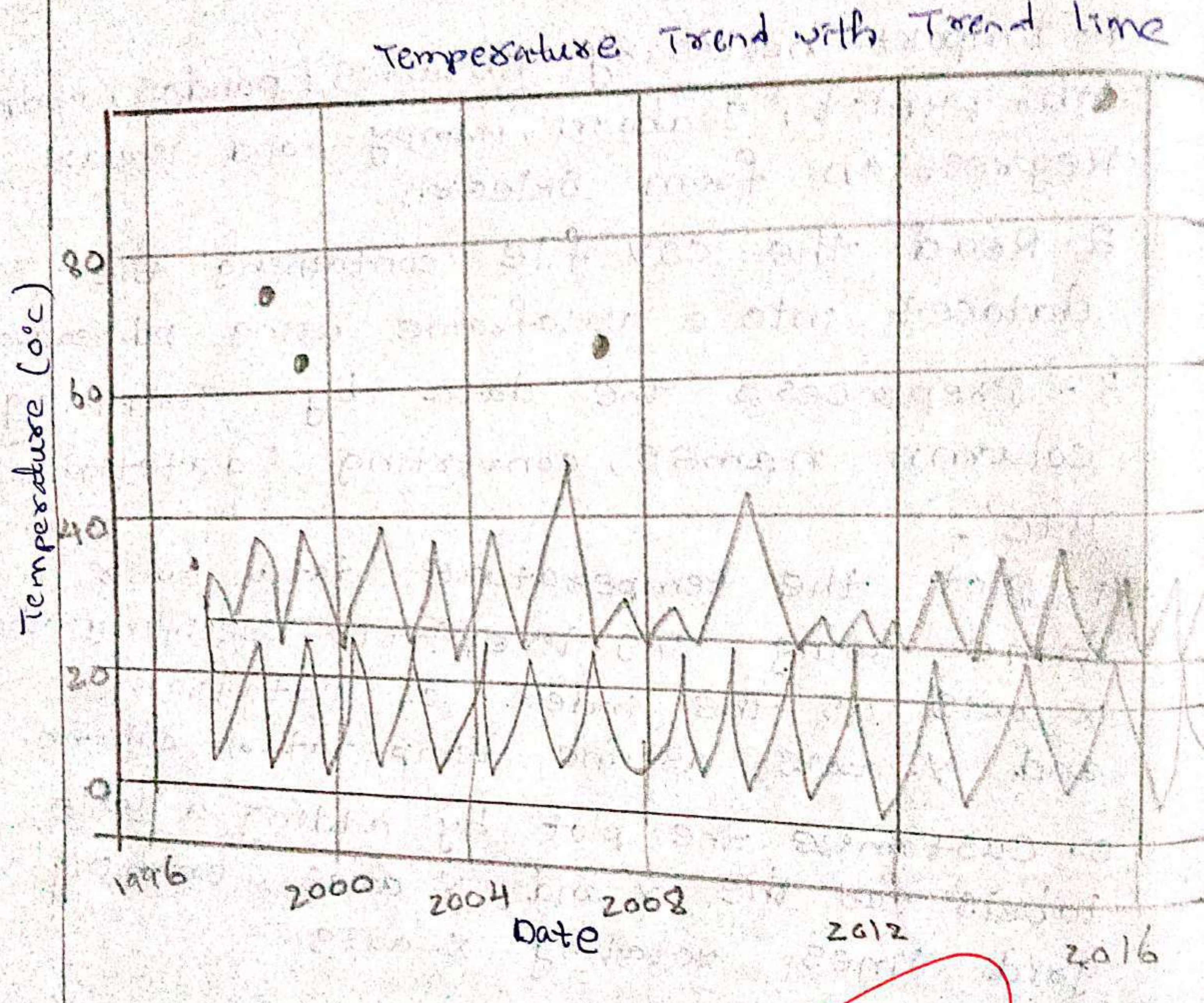
Algorithm:

1. Import necessary libraries: pandas, matplotlib, pyplot, seaborn, numpy, and linear Regression from sklearn.
2. Read the csv file containing the dataset into a Dataframe using pd.read_csv()
3. Preprocess the data by stripping column names, converting 'datetime_utc'.
4. plot the temperature data over time using sns.lineplot(), specifing the x-axis as the index of Dataframe and y-axis as the temperature column.
5. Customize the plot by adding a title, labels for the x and y axes, enabling grid lines, rotating x-axis.

Program:

line graph

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.linear_model import LinearRegression
data = pd.read_csv('/content/testset.csv')
data.columns = data.columns.str.strip()
data['datetime_utc'] = pd.to_date
        time(data['datetime_utc'])
data.set_index('datetime_utc', inplace=True)
```



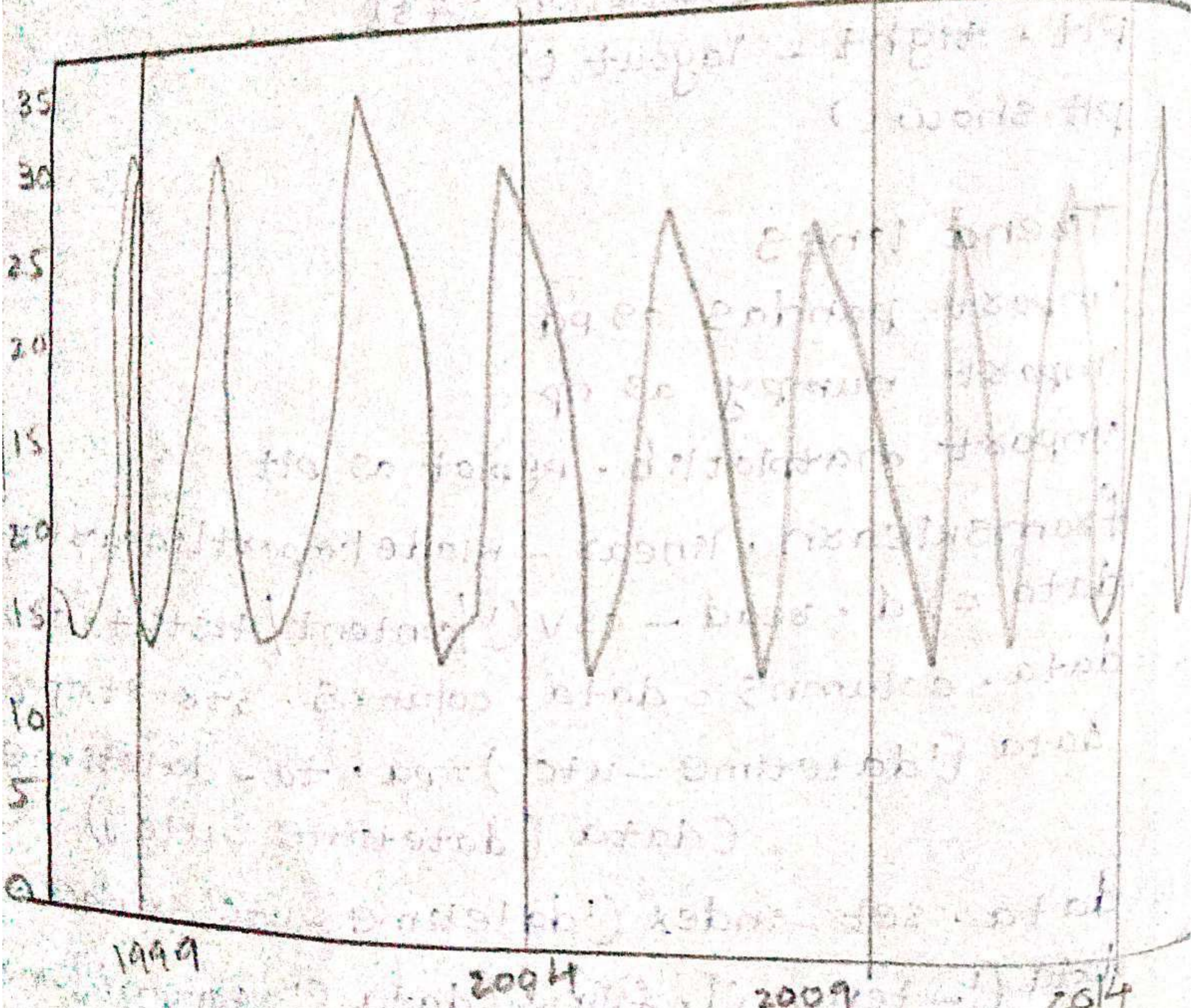
Temperature Trend with Trend line
80
60
40
20
0
Temperature (0°c)
1996
2000
2004
2008
2012
2016
Date

```
plt.figure (figsize =(10,6))
sns.lineplot (data=data ,x=data.index, y='_
 tempm')
plt.title ('Line graph of Temperature overtime')
plt. xlabel ('Date')
plt. ylabel ('Temperature (oc)')
plt. grid (True)
plt. xticks (rotation =45)
plt. tight_layout ()
plt.show()
```

Trend lines

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
fromsklearn.linear_modelimportLinearRegression
data =pd.read_csv('/content/testset.csv')
data.columns =data.columns.str.strip()
data ['datetime_utc')=pd.to_datetime
            (data ['datetime _utc'])
data.set_index ('datetime_utc',inplace =True)
data ['_tempm'].fillna (data ['_tempm'].mean(),
        inplace =True )
x=data.index.astype (int).values.reshape(-1,1)
Y=data ['_tempm'].values
model =LinearRegression()
model .fit (x, Y)
slope =model.coef_[0]
intercept =model.intercept_
plt.figure (figsize =(10,6))
plt. scatter (data.index, data ['tempm'],
        color='blue', label='Temperature Data')
plt.plot(data.index, ...
```

Monthly Mean Temperature



```
        color ='red', linestyle ='-', linewidth =2,
        label ='Trend line')
plt.title ('Temperature Trend with Trend line')
plt.xlabel ('Date')
plt.ylabel ('Temperature (°C)')
plt.grid (True)
plt.legend ()
plt.xticks (rotation =45)
plt.tight_layout ()
plt.show ()
```

Area chart:

```
import pandas as pd
import matplotlib.pyplot as plt
data =pd.read_csv ('/content/testset.csv')
data.columns = data.columns.str.strip ()
data ('datetime_utc) =pd.to_datetime
        (data ['datetime_utc'])
data.set_index ('datetime_utc', inplace=True)
monthly_mean_temp = data ['_tempm'].
        resample ('1M').mean ()
plt.figure (figsize = (10,6))
monthly_mean_temp.plot (kind ='area',
        color ='skyblue', alpha =0.7)
plt.title ('Monthly Mean Temperature')
plt.xlabel ('Date')
plt.ylabel ('Temperature (oc)')
plt.grid (True)
plt.xticks (rotation =45)
plt.tight_layout ()
plt.show ()
```

VEL TECH	
EX No.	9
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	4
RECORD (5)	5
TOTAL (20)	19
[illegible]TH DATE	

29/9

Result: Thus, we successfully implemented the Line graph, Trend lines, Area chart by time oriented data and output is verified.

USE CASE – 1

Earthquake and Geospatial Data Analysis

Batch Mates:

1. VTU24801 JINKALA SHASHIDHAR
2. VTU24802 SHAIK JAKEER HUSSAIN
3. VTU24858 SHAIK MOHAMMED SHAHID
4. VTU24867 KUSUMARAJU BUKARI BABA
5. VTU24905 SHAIK RUBIYA
6. VTU24923 MALLU RUDRA TEJESHWAR REDDY
7. VTU25011 RAAVI NAVADEEP CHOWDARY
8. VTU25104 DONGALA JAYANDAR
9. VTU25106 MADAKKA GARI KRISHNA KOWSHIK
10. VTU25112 SEELAM NAVEEN

Aim:

To analyze and visualize earthquake occurrences using geospatial data.
The goal is to identify regions with the highest seismic activity, visualize earthquake magnitudes on maps,
and observe temporal patterns to support disaster risk assessment and preparedness.

Algorithm:

1. Load the dataset **(earthquake_data.csv)** containing Date, Latitude, Longitude, Magnitude, Depth, Region.
2. Convert **Date** into datetime format and handle missing or invalid entries.
3. Filter earthquakes based on magnitude thresholds (e.g., ≥4.0 for moderate and above).
4. Compute:
   - Average magnitude per region
   - Total earthquakes per region
   - Maximum depth and strongest quake recorded
5. Visualize data using:
   - Geographic Scatter Map (Latitude vs Longitude) color-coded by magnitude
   - Bar chart showing top earthquake-prone regions
   - Line plot showing monthly earthquake counts

6. Generate summary metrics (average magnitude, deepest quake, most active region).
7. Export visualizations and summary results as CSV/PNG files if needed.

Program:

```
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px

df = pd.read_csv("earthquake_data.csv")

df['Date'] = pd.to_datetime(df['Date'], errors='coerce')
df['Magnitude'] = pd.to_numeric(df['Magnitude'], errors='coerce')
df = df.dropna(subset=['Latitude', 'Longitude', 'Magnitude'])

avg_mag = df['Magnitude'].mean().round(2)
max_mag = df['Magnitude'].max()
most_active_region = df['Region'].value_counts().idxmax()

print("\n=== Earthquake Data Summary ===")
print(f"Average Magnitude     : {avg_mag}")
print(f"Strongest Earthquake  : {max_mag}")
print(f"Most Active Region    : {most_active_region}")

fig = px.scatter_geo(df, lat='Latitude', lon='Longitude', color='Magnitude',
                     title='Global Earthquake Distribution by Magnitude',
                     color_continuous_scale='Reds', size='Magnitude', projection='natural earth')
fig.show()
```

```
region_counts = df['Region'].value_counts().head(10)
plt.figure(figsize=(10,6))
plt.bar(region_counts.index, region_counts.values, color='orange')


plt.title("Top 10 Most Earthquake-Prone Regions")
plt.xlabel("Region")
plt.ylabel("Number of Earthquakes")
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.4)
plt.tight_layout()
plt.show()

monthly = df.groupby(df['Date'].dt.to_period('M')).size()
monthly.plot(kind='line', figsize=(10,5), color='green')
plt.title("Monthly Earthquake Occurrence Trend")
plt.xlabel("Month")
plt.ylabel("Number of Earthquakes")
plt.grid(True)
plt.tight_layout()
plt.show()
```

Output:

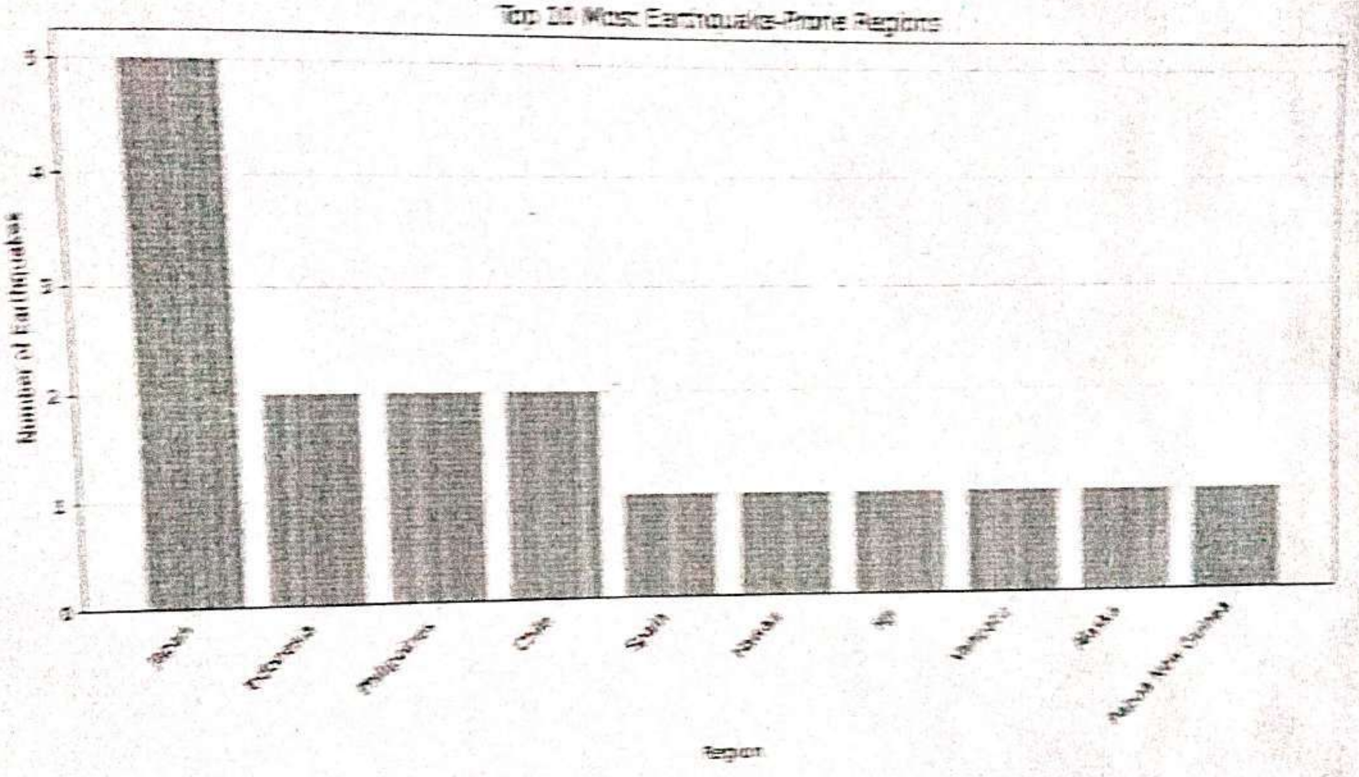
=== Earthquake Data Summary ===

Average Magnitude : 5.98

Strongest Earthquake : 8.3

Most Active Region : Japan

4.555.566.577.58MagnitudeGlobal Earthquake Distribution by Magnitude



VELTECH	
EX No.	[illegible]
PERFORMANCE (5)	[illegible]
RESULT AND ANALYSIS (5)	[illegible]
VIVA VOCE (5)	[illegible]
RECORD (5)	[illegible]
TOTAL (20)	[illegible]
SIGN WITH DATE	

13/10

Result:

Thus, earthquake occurrences were successfully analyzed and visualized using geospatial data. The analysis revealed regions with the highest seismic activity, identified the strongest earthquakes, and showcased temporal patterns in frequency, enabling better understanding of global earthquake behavior and aiding disaster management planning.

27/10/25

USE CASE – 2

Performance of Different Company Departments Over Year

Batch Mates :

1. VTU24801 JINKALA SHASHIDHAR
2. VTU24802 SHAIK JAKEER HUSSAIN
3. VTU24858 SHAIK MOHAMMED SHAHID
4. VTU24867 KUSUMARAJU BUKARI BABA
5. VTU24905 SHAIK RUBIYA
6. VTU24923 MALLU RUDRA TEJESHWAR REDDY
7. VTU25011 RAAVI NAVADEEP CHOWDARY
8. VTU25104 DONGALA JAYANDAR
9. VTU25106 MADAKKA GARI KRISHNA KOWSHIK
10. VTU25112 SEELAM NAVEEN

Aim:

To analyze and visualize yearly department performance using time-series plots and summary metrics to identify trends, best/worst months, and overall strengths or weaknesses.

Algorithm: 1. Define departments and the time dimension (months for the year).

2. Acquire or generate monthly performance metrics for each department (normalized scores, KPIs, etc.).
3. Build a tidy DataFrame: rows = months, columns = departments.
4. Compute derived metrics: yearly average per department, month-over-month changes, best/worst months.
5. Visualize:

   □ Line chart: performance over months (one line per department). □ Bar chart: average yearly performance per department.
6. Export charts and data for reporting (PNG/CSV/PDF).

Program:

```
import numpy as np import
pandas as pd import
matplotlib.pyplot as plt
```

```
df=pd.read_csv("/content/drive/MyDrive/ColabNotebooks/department_performance_2024.csv")
df.set_index('Month', inplace=True) yearly_avg =
df.mean().round(2).sort_values(ascending=False)

best_month = df.idxmax() worst_month
= df.idxmin() monthly_changes =
df.diff().round(2)

print("\n=== Yearly Average Performance (Descending) ===")
print(yearly_avg) print("\n=== Best and Worst Month per Department ===")
for dept in df.columns:    print(f"{dept}: Best = {best_month[dept]}, Worst =
{worst_month[dept]}")

plt.figure(figsize=(10,6)) for
dept in df.columns:
    plt.plot(df.index, df[dept], marker='o', linewidth=2, label=dept)
plt.title("Monthly Performance by Department")
plt.xlabel("Month") plt.ylabel("Performance Score (0-100)")
plt.xticks(rotation=45) plt.grid(axis='y', linestyle='--', alpha=0.4)
plt.legend() plt.tight_layout() plt.show()

plt.figure(figsize=(8,5))
plt.bar(yearly_avg.index, yearly_avg.values)
plt.title("Average Yearly Performance by Department")
plt.xlabel("Department") plt.ylabel("Average
Performance Score (0-100)") plt.xticks(rotation=30)
plt.tight_layout() plt.show()
```

OUTPUT:

=== Yearly Average Performance (Descending) ===

Sales	70.30
Finance	57.68
Marketing	54.05
R&D	49.08
HR	46.83

=== Best and Worst Month per Department ===

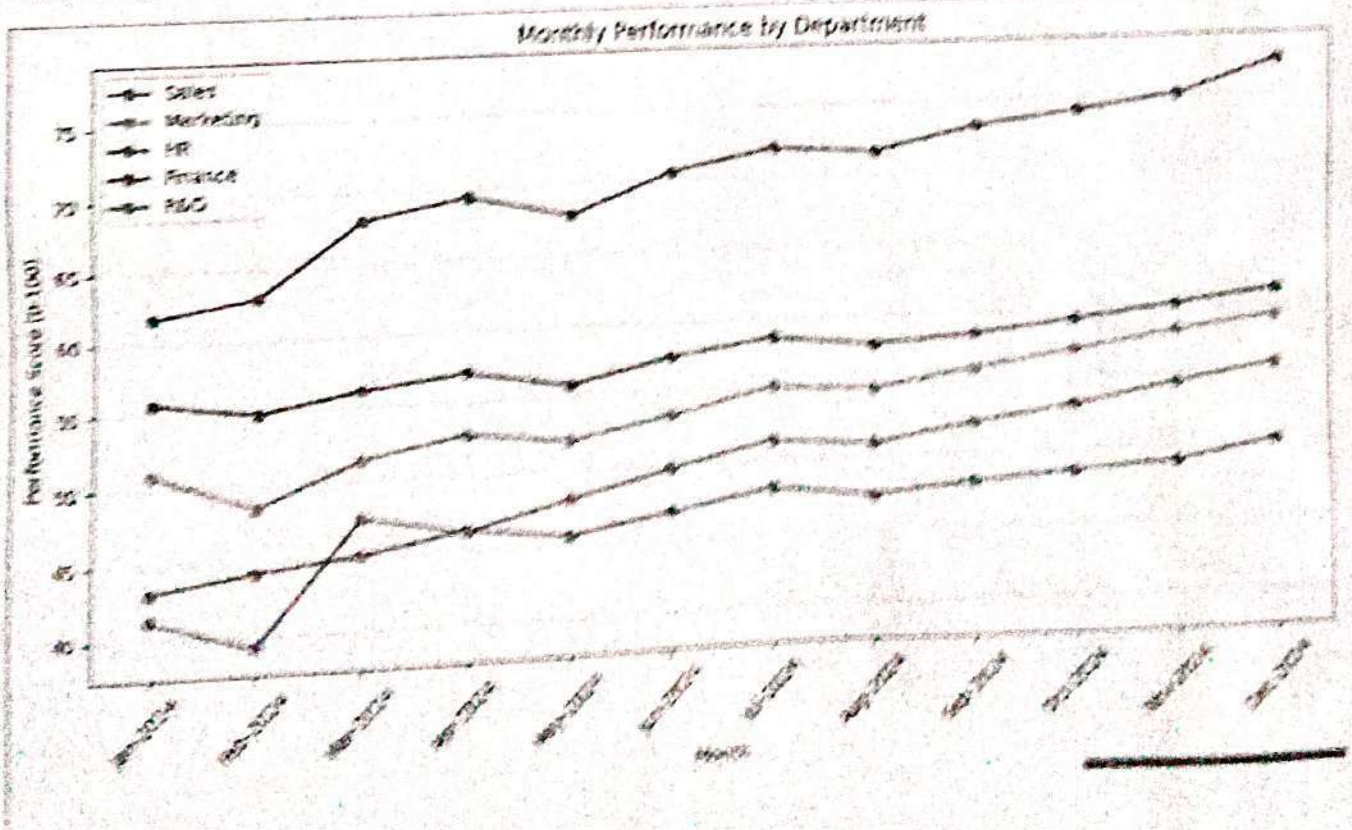
Sales: Best = Dec-2024, Worst = Jan-2024

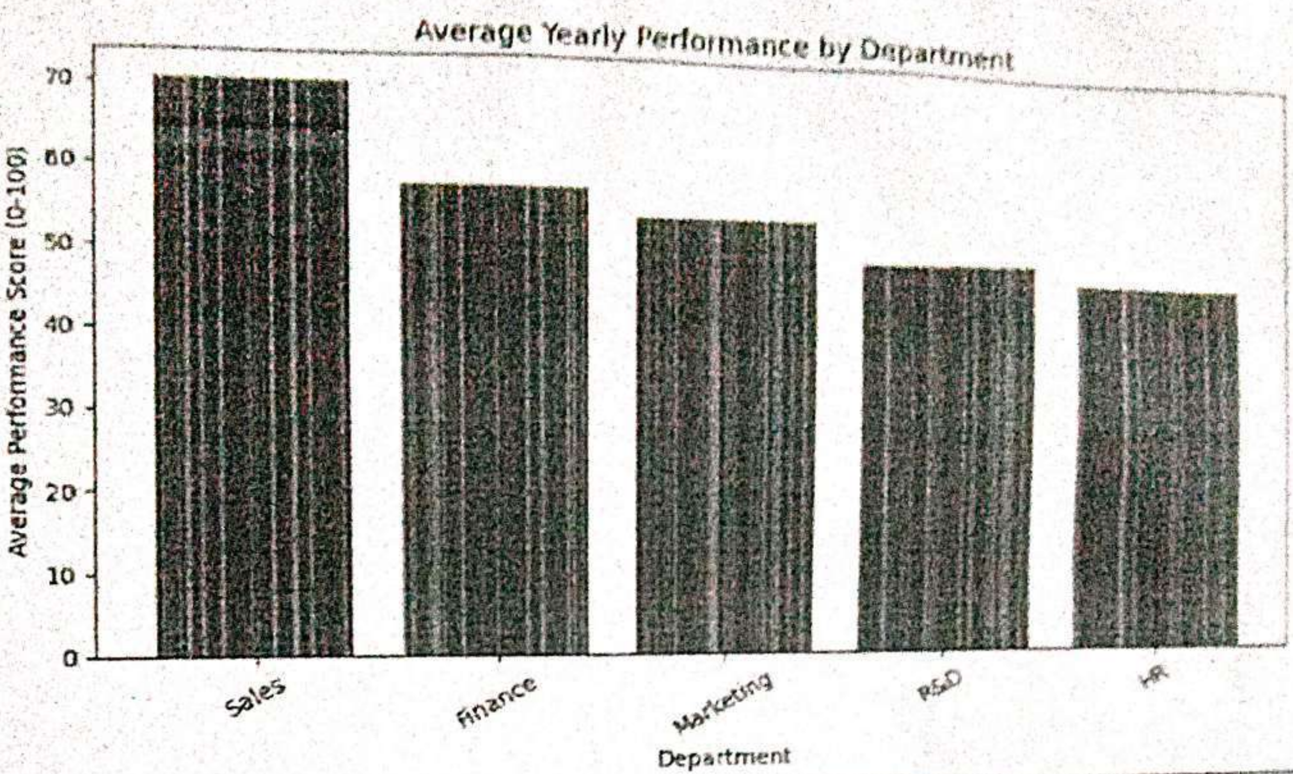
Marketing: Best = Dec-2024, Worst = Feb-2024

HR: Best = Dec-2024, Worst = Jan-2024

Finance: Best = Dec-2024, Worst = Feb-2024

R&D: Best = Dec-2024, Worst = Feb-2024




Average Yearly Performance by Department
Average Performance Score (0-100)
0
10
20
30
40
50
60
70
Sales
Finance
Marketing
R&D
HR
Department

VEL TECH	
EX. No.	002
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	2
RECORD (5)	5
[illegible] (20)	18
[illegible]	[illegible]

27/10

Result:

Thus, the yearly department performance was successfully analyzed and visualized, revealing key trends, best and worst months, and insights for data-driven decisions.