DATA VISUALIZATION

Assignment

Name: ch. Koushikvarma

VTU NO: 22058

SLOT: S2L5

Assignment

Banking - Customer Insights:

1) Identify Data types

Aim: To classify banking datasets Columns into Categorical vs Continuous variables for further Analysis

Algorithm:

1) Load Dataset
2) for each column in dataset:
   If Column data type == object or Category:
      label as Categorical
   Else if numeric:
      label as Continuous
3) Display both lists

Code:

```
Import pandas as Pd
data = Pd.read_csv ("bank_Customer.csv")
Categorical = data.Select_dtypes (include = ['object', 'category']).
                                                        Columns
Continuous = data.Select_dtypes (include = ['int64', 'float64']).colum
```

Output:

Data Types:

Customer_ID	object
Customer_Name	object
Age	int64
Gender	object
State	object
Transaction_type	object
Amount	float64
Loan_Amount	float64
Loan_Default	int64
Loan_Officer	object
Customer_Review	object
Account_Balance	float64

dtype: object

```
print ("Categorical Variables:", list (categorical))
print ("Continuous Variables:", list (continuous))
```

2) Histogram (Transaction Frequency) & Density Plot (Loan Defaults)

Aim: To visualize how often Customers perform transactions and the likelihood of loan defaults

Algorithm:

1) Load dataset
2) Plot histogram for transaction frequency
3) Plot density Curve for loan amount by default Status

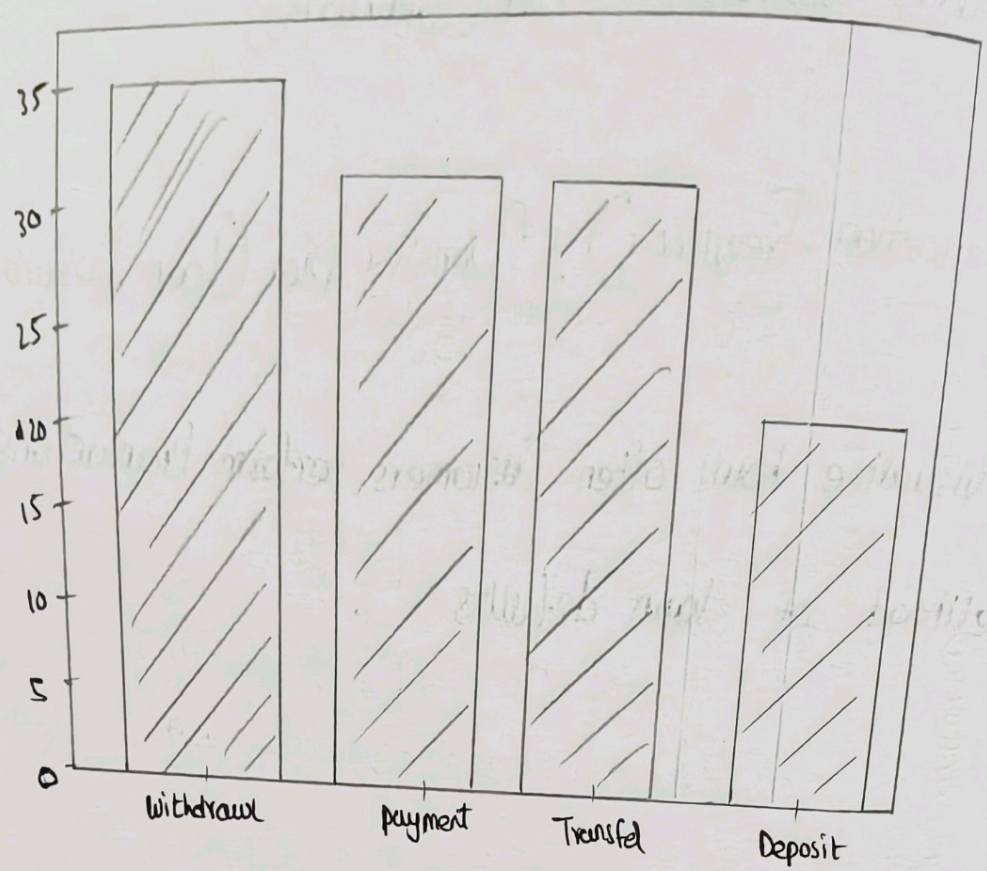
Program:

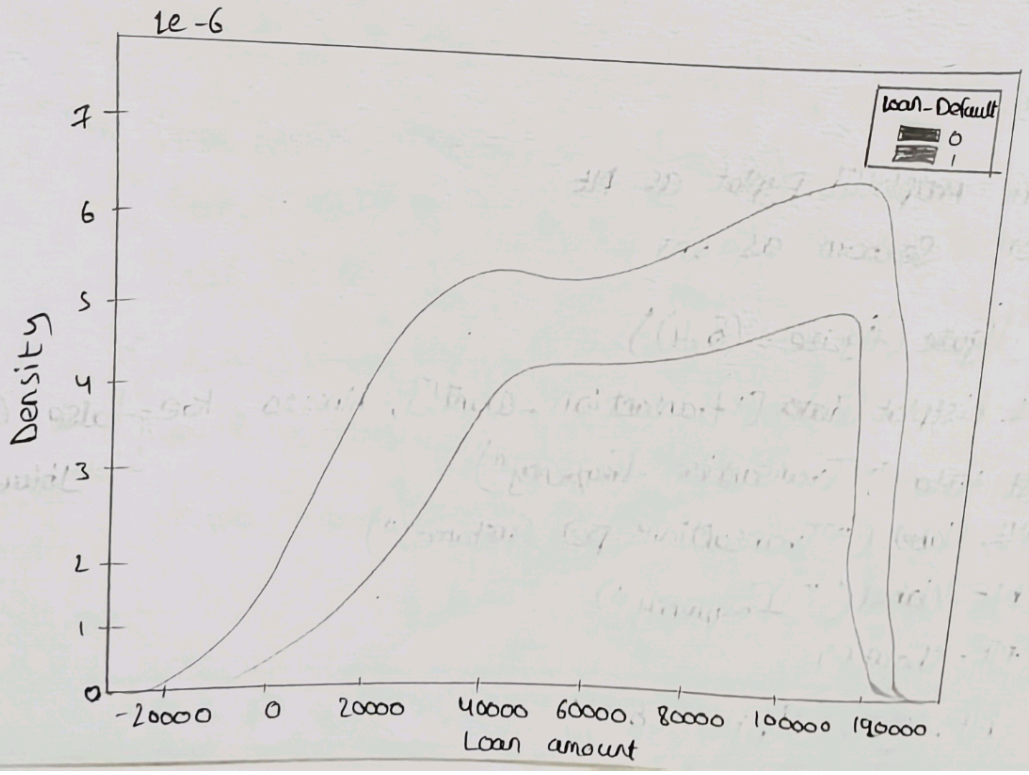
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure (figsize = (8,4))
sns.histplot (data ['transaction_count'], bins=20, kde=false, Color
plt.title ("Transaction Frequency")                    ='skyblue')
plt.xlabel ("Transactions per Customer")
plt.ylabel ("Frequency")
plt.show ()

plt.figure (figsize = (8,4))
```

Output:


35
30
25
20
15
10
5
0
Withdrawl
payment
Transfel
Deposit


1e-6
7
6
5
4
3
2
1
0
Density
loan_Default
0
1
-20000
0
20000
40000
60000
80000
100000
120000
Loan amount

```
sns.kdeplot(data=data, x="loan_amount", hue="default_status",
                                                fill=True)
plt.title("Loan Defaults Density Distribution")
plt.show()
```

3) Graph : Customers linked to loan Officers + Word Cloud of Reviews

Aim: To show relationships between Customers and loan officers as a network graph and analyze Customer feedback through a word cloud.

Algorithm:

1) Load dataset
2) Create edges : (loan_officer, Customer)
3) Draw network graph using NetworkX
4) Generate word cloud from Customer reviews

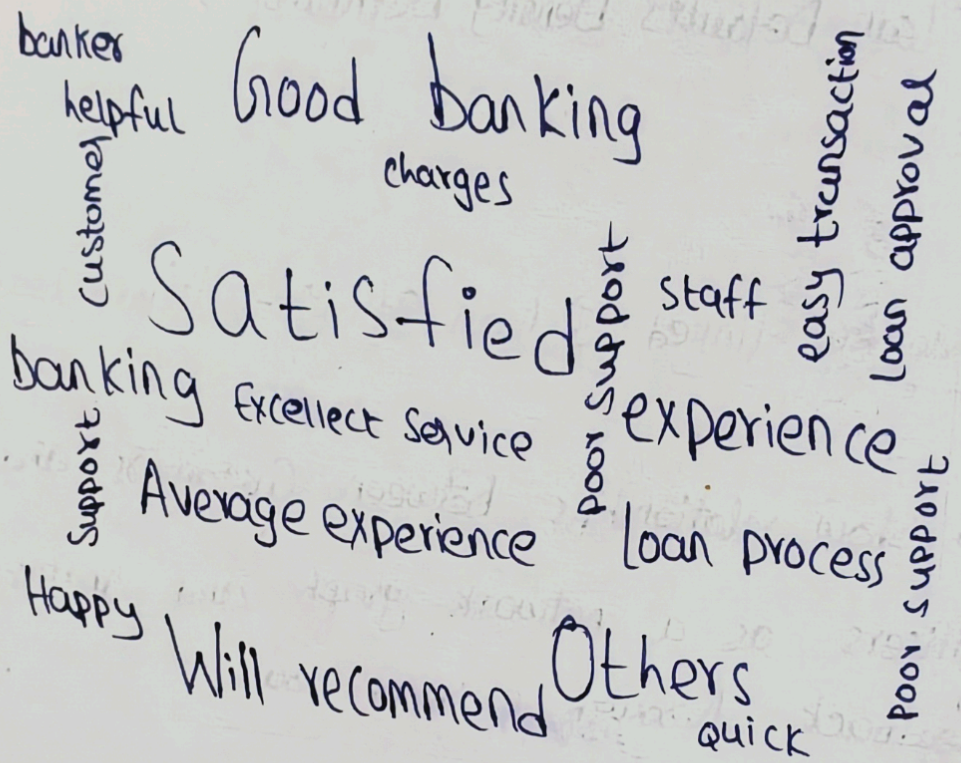
Program:

```
import networkx as nx
from wordcloud import wordcloud
import matplotlib.pyplot as plt

G = nx.from_pandas_edgelist(data, source='loan_officer', target
                                                = 'Customer_id'

plt.figure(figsize=(8,6))
```

output:



Word cloud

```
nx.draw(G, with_labels=False, node_size=30, node_color='orange')
plt.title("Customers linked to loan Officers")
plt.show()

text = " ".join(data['Customer_reviews'].dropna())
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(text)

plt.figure(figsize=(8,4))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("Customer Review Word cloud")
plt.show()
```

4) Choropleth Map - Customer Branches Across States

Aim: To visualize no of Customers per state on a map.

Algorithm:

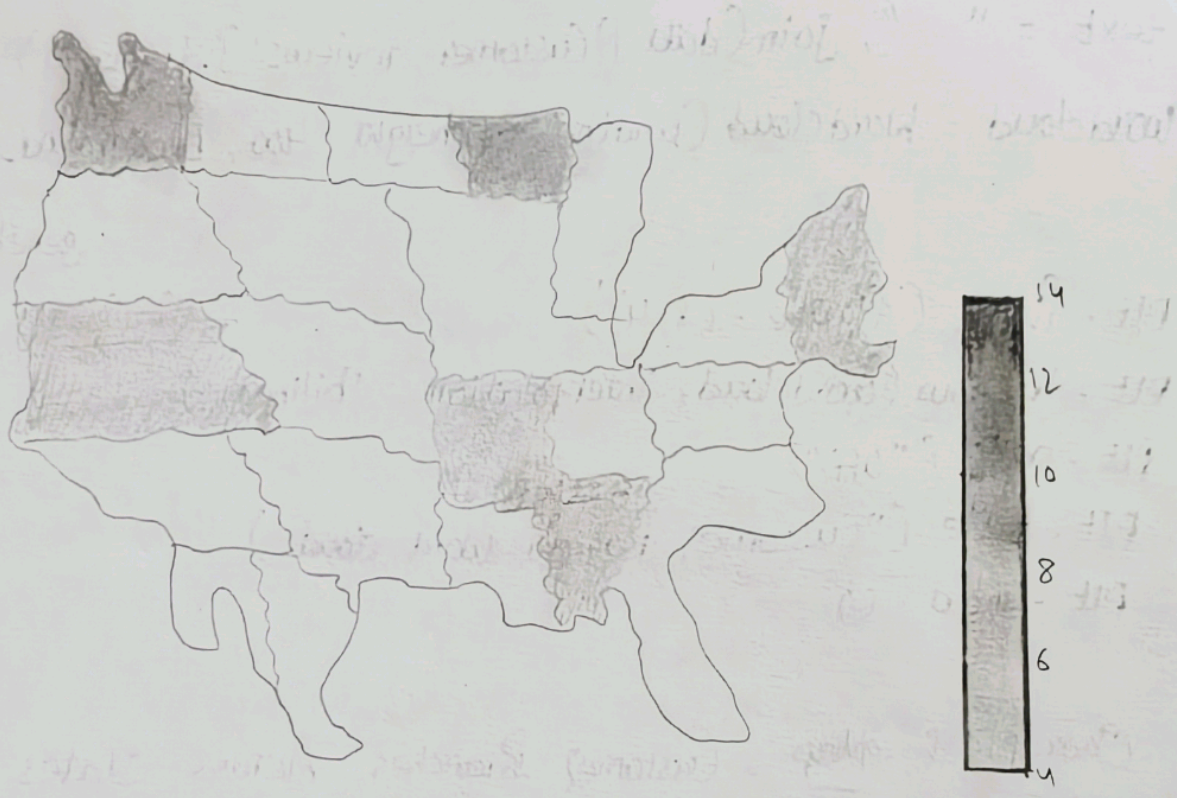
1) Load dataset
2) Group data by state
3) Merge grouped data with Shapefile or geojson
4) Plot choropleth map using plotly

Program:

```
import plotly.express as px
branch_data = data.groupby('state').size().reset_index(name=)
```

fig = px.ch
branch

output:



```
= px.choropleth(
    branch_data,
    locations = "state",
    locationmode = "USA-states",
    Color = "num_Customers",
    scope = "usa",
    Color_Continuous_scale = "Blues",
    title = "Customer Branches Across States"
)
fig.show()
```

5) Interactive Financial Dashboard

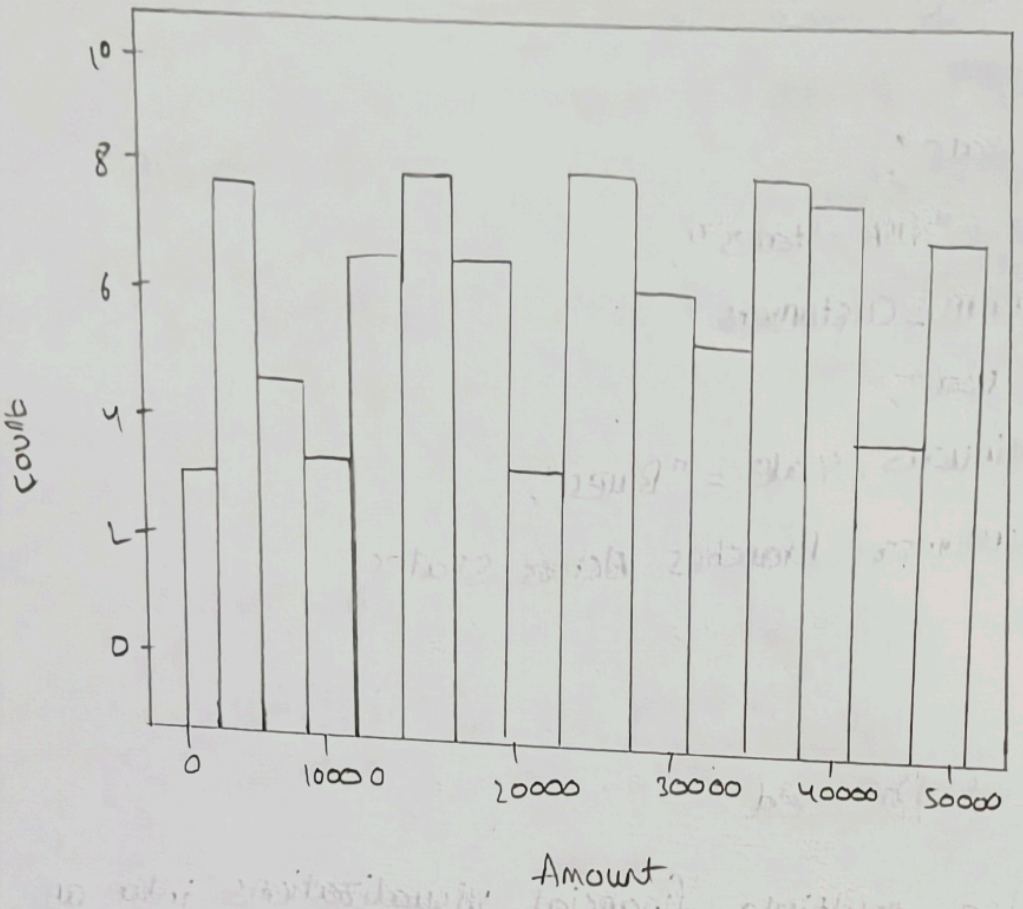
Aim: To Combine multiple financial visualizations into an interactive dashboard.

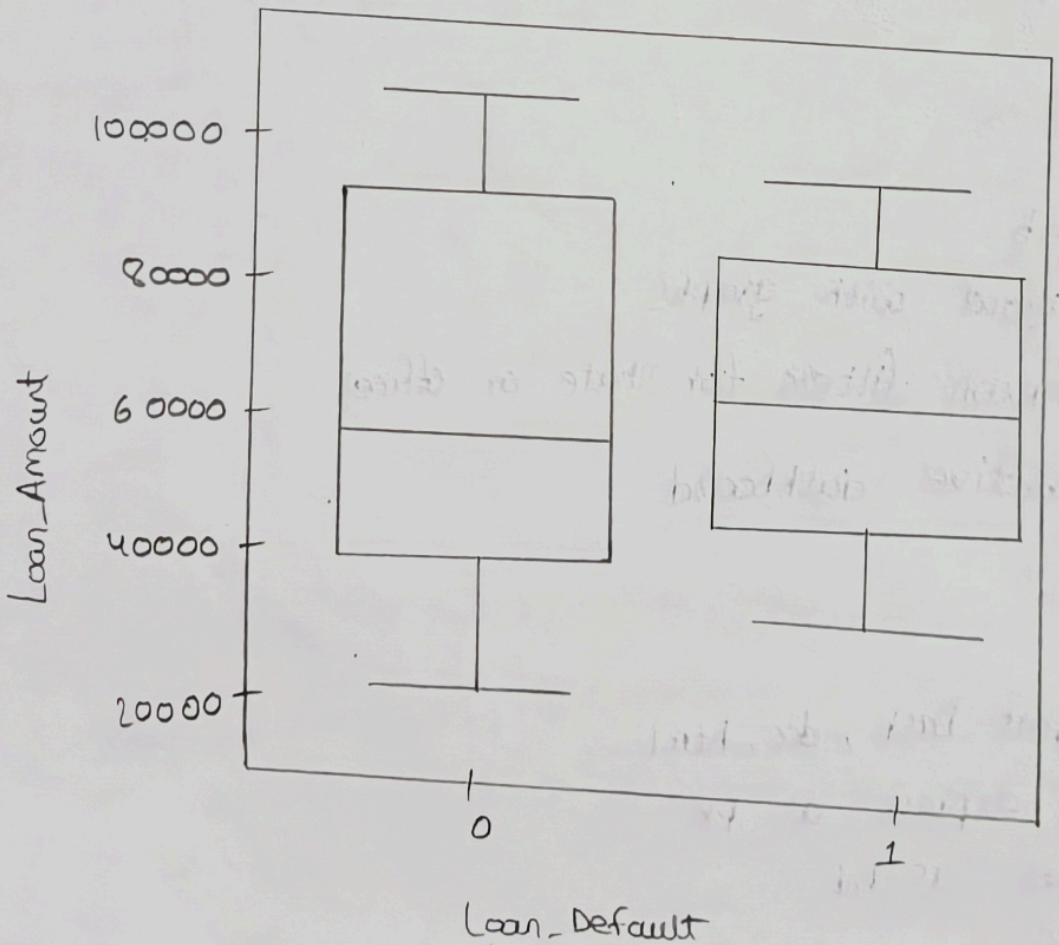
Algorithm:

1) Import Dash
2) Create layout with graphs
3) Add dropdown filters for state or offices
4) Run interactive dashboard.

Program:

```
from dash import Dash, dcc, html
import plotly.express as px
import pandas as pd
app = Dash(__name__)
```


count
10
8
6
4
2
0
0
10000
20000
30000
40000
50000
Amount


Loan_Amount
100000
80000
60000
40000
20000
0
1
Loan_Default

```
app.layout = html.Div([
    html.H1("Banking Customer Insights Dashboard"),
    dcc.Graph(figure=fig1),
    dcc.Graph(figure=fig2),
    dcc.Graph(figure=fig3)
])

if __name__ == "__main__":
    app.run_server(debug=False)
```