Experiment No - 9

9.1 Aim/Purpose of the Experiment: To familiarize the students with data visualization using more than two feature variables.

9.2 Learning Outcomes : Knowledge of the Data cleaning, Data preparation and data visualization using multivariate analysis in python.

9.3 Prerequisites : Basic knowledge of programming, python syntax, matplotlib, seaborn, different libraries.

9.4 Materials/Equipment/Apparatus / Devices/Software required : Jupyter Notebook

9.5 Introduction and Theory : Multivariate analysis is a statistical method used to understand the relationship between multiple variables simultaneously. In Python, you can perform multivariate analysis using libraries such as NumPy, Pandas, and Matplotlib/Seaborn for data manipulation, analysis, and visualization. Here's a brief outline of the process

- Data Preparation: Load your dataset into a Pandas DataFrame and clean/preprocess the data if necessary. Ensure that the variables of interest are properly formatted and ready for analysis.
- Descriptive Statistics: Compute descriptive statistics for all variables in the dataset. This includes measures such as mean, median, mode, standard deviation, variance, minimum, maximum, and quartiles for each variable.
- Visualization: Create visualizations to explore the relationships between multiple variables. Common plots for multivariate analysis include scatter plots, pair plots, heatmap correlation matrices, and parallel coordinate plots. Matplotlib and Seaborn are popular libraries for creating these visualizations.
- Correlation Analysis: Calculate correlation coefficients between pairs of variables to measure the strength and direction of the linear relationship. This helps identify potential associations between variables and can guide further analysis.

Multivariate Analysis

```
import warnings

warnings.filterwarnings("ignore")

import pandas as pd, numpy as np

import matplotlib.pyplot as plt, seaborn as sns

%matplotlib inline

inp0= pd.read_csv("bank_marketing_updated_v1.csv")

inp0.head()
```

Segment- 3, Fixing the Rows and Columns

Checklist for fixing rows:

• Delete summary rows: Total and Subtotal rows

• Delete incorrect rows: Header row and footer row

• Delete extra rows: Column number, indicators, Blank rows, Page No.

Checklist for fixing columns:

• Merge columns for creating unique identifiers, if needed, for example, merge the columns State and City into the column Full address.

• Split columns to get more data: Split the Address column to get State and City columns to analyse each separately.

• Add column names: Add column names if missing.

• Rename columns consistently: Abbreviations, encoded columns.

• Delete columns: Delete unnecessary columns.

• Align misaligned columns: The data set may have shifted columns, which you need to align correctly

```
inp0=pd.read_csv("bank_marketing_updated_v1.csv", skiprows= 2)
inp0.head()
inp0.drop("customerid", axis=1, inplace=True)
inp0.head()
inp0['job']=inp0.jobedu.apply(lambda x: x.split(",")[0])
inp0.head()
inp0['education']=inp0.jobedu.apply(lambda x: x.split(",")[1])
inp0.head()
inp0.drop('jobedu',axis= 1, inplace= True)
inp0.head()
inp0.isnull().sum()
inp0.age.isnull().sum()
inp0.shape
float(100.0*20/45211)
inp1=inp0[-inp0.age.isnull()].copy()
inp1.shape
inp1.month.isnull().sum()
```

```
float(100.0*50/45191)
month_mode=inp1.month.mode()[0]
month_mode
inp1.month.fillna(month_mode, inplace= True)
inp1.month.value_counts(normalize= True)
inp1.month.isnull().sum()
inp1.response.isnull().sum()
float(100.0*30/45191)
inp1= inp1[~inp1.response.isnull()]
inp1.isnull().sum()
inp1.pdays.describe()
inp1.loc[inp1.pdays<0,"pdays"]=np.NaN
inp1.pdays.describe()
```

Multivariate analysis

Education vs marital vs response

```
res=pd.pivot_table(data=inp1, index="education", columns="marital", values="campaign")
res
sns.heatmap(res, annot= True, cmap="RdYlGn")
plt.show()
sns.heatmap(res, annot= True, cmap="RdYlGn", center= 0.117)
plt.show()
res=pd.pivot_table(data=inp1, index="job", columns="marital", values="campaign")
res
sns.heatmap(res, annot= True, cmap="RdYlGn", center= 0.117)
plt.show()
res=pd.pivot_table(data=inp1, index="education", columns="poutcome", values="campaign")
sns.heatmap(res, annot= True, cmap="RdYlGn", center= 0.117)
plt.show()
```