Experiment-8

Aim/Purpose of the Experiment

To familiarize the students with data visualization using two feature variables.

Learning Outcomes

Knowledge of the Data cleaning, Data preparation and data visualization using bivariate analysis in python.

Prerequisites

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

Materials/Equipment/Apparatus / Devices/Software required

Jupyter Notebook.

Bivariate Analysis

```
# import the warnings.
import warnings
warnings.filterwarnings("ignore")

#import the useful libraries.
import pandas as pd, numpy as np
import matplotlib.pyplot as plt, seaborn as sns

# read the csv file
inp0= pd.read_csv("bank_marketing_updated_v1.csv")
#Print the head of the data frame.
inp0.head()
```

```
    age  salary  balance  marital                  jobedu targeted
default  \
0  58.0  100000     2143  married     management,tertiary      yes
no
1  44.0   60000       29   single    technician,secondary      yes
no
2  33.0  120000        2  married  entrepreneur,secondary      yes
no
3  47.0   20000     1506  married     blue-collar,unknown       no
no
4  33.0       0        1   single         unknown,unknown       no
no

  housing loan  contact  day      month duration  campaign  pdays
previous  \
0     yes   no  unknown    5  may, 2017  261 sec         1     -1
0
1     yes   no  unknown    5  may, 2017  151 sec         1     -1
0
2     yes  yes  unknown    5  may, 2017   76 sec         1     -1
0
3     yes   no  unknown    5  may, 2017   92 sec         1     -1
0
4      no   no  unknown    5  may, 2017  198 sec         1     -1
0

  poutcome response
0  unknown       no
1  unknown       no
2  unknown       no
3  unknown       no
4  unknown       no
```

```
#Extract job in newly created 'job' column from "jobedu" column.
inp0['job']=inp0.jobedu.apply(lambda x: x.split(",")[0])
inp0.head()
```

```
    age  salary  balance  marital                  jobedu targeted
default  \
0  58.0  100000     2143  married     management,tertiary      yes
no
1  44.0   60000       29   single    technician,secondary      yes
no
2  33.0  120000        2  married  entrepreneur,secondary      yes
no
3  47.0   20000     1506  married     blue-collar,unknown       no
no
4  33.0       0        1   single         unknown,unknown       no
no
```

```
  poutcome response           job  education
0  unknown       no    management   tertiary
1  unknown       no    technician  secondary
2  unknown       no  entrepreneur  secondary
3  unknown       no   blue-collar    unknown
4  unknown       no       unknown    unknown
```

```
#drop the "jobedu" column from the dataframe.
inp0.drop('jobedu',axis= 1, inplace= True)
inp0.head()
```

```
    age  salary  balance  marital targeted default housing loan
contact  day  \
0  58.0  100000     2143  married      yes      no     yes   no
unknown    5
1  44.0   60000       29   single      yes      no     yes   no
unknown    5
2  33.0  120000        2  married      yes      no     yes  yes
unknown    5
3  47.0   20000     1506  married       no      no     yes   no
unknown    5
4  33.0       0        1   single       no      no      no   no
unknown    5
```

```
       month duration  campaign  pdays  previous poutcome response  \
0  may, 2017  261 sec         1     -1         0  unknown       no
1  may, 2017  151 sec         1     -1         0  unknown       no
2  may, 2017   76 sec         1     -1         0  unknown       no
3  may, 2017   92 sec         1     -1         0  unknown       no
4  may, 2017  198 sec         1     -1         0  unknown       no
```

```
           job  education
0   management   tertiary
1   technician  secondary
2  entrepreneur  secondary
3  blue-collar    unknown
4      unknown    unknown
```

```
inp0[inp0.month.apply(lambda x: isinstance(x,float))== True]
```

```
inp0.isnull().sum()
```

```
age          20
salary        0
balance       0
marital       0
targeted      0
default       0
```

```
  poutcome response           job  education
0  unknown       no    management   tertiary
1  unknown       no    technician  secondary
2  unknown       no  entrepreneur  secondary
3  unknown       no   blue-collar    unknown
4  unknown       no       unknown    unknown
```

```
#drop the "jobedu" column from the dataframe.
inp0.drop('jobedu',axis= 1, inplace= True)
inp0.head()
```

```
    age  salary  balance  marital targeted default housing loan
contact  day  \
0  58.0  100000     2143  married      yes      no     yes   no
unknown    5
1  44.0   60000       29   single      yes      no     yes   no
unknown    5
2  33.0  120000        2  married      yes      no     yes  yes
unknown    5
3  47.0   20000     1506  married       no      no     yes   no
unknown    5
4  33.0       0        1   single       no      no      no   no
unknown    5
```

```
       month duration  campaign  pdays  previous poutcome response  \
0  may, 2017  261 sec         1     -1         0  unknown       no
1  may, 2017  151 sec         1     -1         0  unknown       no
2  may, 2017   76 sec         1     -1         0  unknown       no
3  may, 2017   92 sec         1     -1         0  unknown       no
4  may, 2017  198 sec         1     -1         0  unknown       no
```

```
           job  education
0   management   tertiary
1   technician  secondary
2  entrepreneur  secondary
3  blue-collar    unknown
4      unknown    unknown
```

```
inp0[inp0.month.apply(lambda x: isinstance(x,float))== True]
```

```
inp0.isnull().sum()
```

```
age          20
salary        0
balance       0
marital       0
targeted      0
default       0
```

```
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
response     0
job          0
education    0
dtype: int64
```

```
#describe the pdays column of inp1.
inp1.pdays.describe()
```

```
count    45161.000000
mean        40.182015
std        100.079372
min         -1.000000
25%         -1.000000
50%         -1.000000
75%         -1.000000
max        871.000000
Name: pdays, dtype: float64
```

```
#plot the scatter plot of balance and salary variable in inp1
plt.scatter(inp1.salary, inp1.balance)
plt.show()
```

```
#groupby the response to find the mean of the balance with respons
& yes seperatly.
inp1.groupby("response")["balance"].mean()
```

```
response
no     1304.292281
yes    1804.681362
Name: balance, dtype: float64
```

```
#groupby the response to find the median of the balance with respo
no & yes seperatly.
inp1.groupby("response")["balance"].median()
```

```
response
no     417.0
yes    733.0
Name: balance, dtype: float64
```

```
#function to find the 75th percentile.
def p75(x):
 return np.quantile(x, 0.75)

#calculate the mean, median and 75th percentile of balance with
response
inp1.groupby("response")["balance"].aggregate(["mean","median",p75
```

```
                mean  median     p75
response
no        1304.292281   417.0  1345.0
yes       1804.681362   733.0  2159.0
```

```
#plot the bar graph of balance's mean an median with response.
inp1.groupby("response")
["balance"].aggregate(["mean","median"]).plot.bar()
plt.show()
```

```
#groupby the education to find the mean of the salary education
category.
inp1.groupby("education")["salary"].mean()
```

```
education
primary      34232.343910
secondary    49731.449525
tertiary     82880.249887
unknown      46529.633621
Name: salary, dtype: float64
```

```
#groupby the education to find the median of the salary for each
education category.
inp1.groupby("education")["salary"].median()
```

```
education
primary       20000.0
secondary     55000.0
tertiary     100000.0
unknown       50000.0
Name: salary, dtype: float64
```

```
# Job vs salary
#groupby the job to find the mean of the salary for each job category.
inp1.groupby('job')['salary'].mean()
```

```
job
admin.            50000.0
blue-collar       20000.0
entrepreneur     120000.0
housemaid         16000.0
management       100000.0
retired           55000.0
self-employed     60000.0
services          70000.0
student            4000.0
technician        60000.0
unemployed         8000.0
unknown               0.0
Name: salary, dtype: float64
```

```
inp1.groupby('job')['salary'].median()
```

```
job
admin.            50000.0
blue-collar       20000.0
entrepreneur     120000.0
housemaid         16000.0
management       100000.0
retired           55000.0
self-employed     60000.0
services          70000.0
student            4000.0
technician        60000.0
unemployed         8000.0
unknown               0.0
Name: salary, dtype: float64
```

```
#create response_flag of numerical data type where response "yes"=
1,"no"= 0
inp1["response_flag"]=np.where(inp1.response=="yes", 1, 0)
inp1.response.value_counts()
```

```
no     39876
yes     5285
Name: response, dtype: int64
```

```
no     0.882974
yes    0.117026
Name: response, dtype: float64
```

```
inp1.response_flag.mean()
```

```
0.1170257523084077
```

```
#calculate the mean of response_flag with different education
categories.
inp1.groupby("education")["response_flag"].mean()
```

```
education
primary      0.086416
secondary    0.105608
tertiary     0.150083
unknown      0.135776
Name: response_flag, dtype: float64
```

```
# Marital vs response rate
#calculate the mean of response_flag with different marital status
categories.
inp1.groupby(["marital"])["response_flag"].mean()
```

```
marital
divorced    0.119469
married     0.101269
single      0.149554
Name: response_flag, dtype: float64
```

```
#plot the bar graph of marital status with average value of
response_flag
inp1.groupby(["marital"])["response_flag"].mean().plot.barh()
plt.show()
```

```
# Loans vs response rate
#plot the bar graph of personal loan status with average value of
response_flag
inp1.groupby(["loan"])["response_flag"].mean().plot.bar()
plt.show()
```

```
#plot the bar graph of housing loan status with average value of
response_flag
inp1.groupby(["housing"])["response_flag"].mean().plot.bar()
plt.show()
```

```
# Age vs response
#plot the boxplot of age with response_flag
sns.boxplot(data=inp1, x="response",y="age")
plt.show()
```

```
#plot the bar graph of job categories with response_flag mean value.
inp1.groupby(['job'])['response_flag'].mean().plot.barh()
plt.show()
```

Important Keywords:-

inp0.drop("customerid", axis=1, inplace=True):

Removes the customerid column from the inp0 DataFrame.

inp0['job'] = inp0.jobedu.apply(lambda x: x.split(",")[0]):

Extracts the job title from the jobedu column by splitting at the comma.

inp0.isnull().sum():

Displays the count of missing values in each column of inp0.

month_mode = inp1.month.mode()[0]:

Finds the most frequent month value (mode) from the month column in inp1.

inp1.loc[inp1.pdays < 0, "pdays"] = np.nan:

Replaces negative values in the pdays column with NaN (missing values).

inp1.plot.scatter(x="age", y="balence"):

Creates a scatter plot with age on the x-axis and balence on the y-axis.

sns.pairplot(data=inp1, vars=['salary', 'balance', 'age']):

Plots pairwise scatter plots and histograms for salary, balance, and age.

sns.heatmap(inp1[['salary','balance','age']].corr(), annot=True, cmap="Reds"):

Draws a heatmap showing correlation between salary, balance, and age with values and a red color palette.

sns.boxplot(data=inp1, x='response', y='salary'):

Shows salary distribution by response category using a boxplot.

inp1.groupby('response')['balance'].aggregate(['mean', 'median', p75]):

Computes mean, median, and 75th percentile of balance grouped by response.

inp1.groupby('job')['salary'].mean():

Calculates the average salary for each job category.