King Saud University
Software Engineering Department
SWE485 Selected Topics in Software Engineering

Advertisement - Click on Ad

Project Phase #1

Section No. 60121 - Group No. 6

#	Name	ID
1	Noura AlKhorayef	441201063
2	Shoug AlShaybani	441200872
3	Fay AlShalwi	441200559
4	Shahad AlShaikh	441201213

Table of Contents

1. INTRODUCTION

Our dataset indicates whether or not a particular internet user clicked on an advertisement. We chose this dataset because we are interested in knowing what kinds of advertisements are more likely to get clicked on and by which ages and in which countries. As this also will help the advertising companies know how to distribute their ads to acquire the highest benefits possible. Each observation includes the following attributes:

'Daily Time Spent on Site': consumer time on site in minutes
'Age': customer age in years
'Area Income': Avg. Income of geographical area of consumer
'Daily Internet Usage': Avg. minutes a day consumer is on the internet
'Ad Topic Line': Headline of the advertisement
'City': City of consumer
'Male': Whether or not consumer was male
'Country': Country of consumer
'Timestamp': Time at which consumer clicked on Ad or closed window
'Clicked on Ad': 0 or 1 indicated clicking on Ad

2. MACHINE LEARNING TASKS

Our problem will be classification on A fake advertising data set indicating whether a user clicked on ad or not. So, data will be classified into 2 categories. 0=no, 1=yes

3. DATA SOURCE

The dataset is called the Advertisement - Click on Ad dataset which we got from kaggle.com website. Kaggle is an online community platform for data scientists and machine learning enthusiasts. Kaggle allows users to collaborate with other users, find and publish datasets, use GPU integrated notebooks, and compete with other data scientists to solve data science challenges. Something great about Kaggle is you can get any dataset you like by simply downloading it.

URL:https://www.kaggle.com/datasets/gabrielsantello/advertisement-click-on ad?resource=download

4. DATA EXPLORATION

It consists of exactly 1000 observations that each consist of 10 variables (Daily Time Spent on Site, age, Area income, Daily Internet Usage, Ad Topic Line, City, Male, Country, Timestamp, Clicked on Ad)

```
In [4]: #number of observations
        len(df)

Out[4]: 1000

In [5]: #number of variables
        df.shape[1]

Out[5]: 10
```

Figure 1: number of observations and variables

here is the table that shows each variable and its type:

Variable name	Variable type
Daily Time Spent on Site	float64
Age	int64
Area Income	float64
Daily Internet Usage	float64
Ad Topic Line	object
City	object
Male	int64
Country	object
Timestamp	object
Clicked on Ad	int64

Table 1

we got the variable names and types by computing this line of code:

```
In [7]: df.dtypes

Out[7]: Daily Time Spent on Site    float64
        Age                           int64
        Area Income                 float64
        Daily Internet Usage        float64
        Ad Topic Line                object
        City                         object
        Male                          int64
        Country                      object
        Timestamp                    object
        Clicked on Ad                 int64
        dtype: object
```

Figure 2: variables type

The first 10 records of our dataset:

```
In [4]: df.head(10)
```

Out[4]:

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad
0	68.95	35	61833.90	256.09	Cloned 5thgeneration orchestration	Wrightburgh	0	Tunisia	2016-03-27 00:53:11	0
1	80.23	31	68441.85	193.77	Monitored national standardization	West Jodi	1	Nauru	2016-04-04 01:39:02	0
2	69.47	26	59785.94	236.50	Organic bottom-line service-desk	Davidton	0	San Marino	2016-03-13 20:35:42	0
3	74.15	29	54806.18	245.89	Triple-buffered reciprocal time-frame	West Terrifurt	1	Italy	2016-01-10 02:31:19	0
4	68.37	35	73889.99	225.58	Robust logistical utilization	South Manuel	0	Iceland	2016-06-03 03:36:18	0
5	59.99	23	59761.56	226.74	Sharable client-driven software	Jamieberg	1	Norway	2016-05-19 14:30:17	0
6	88.91	33	53852.85	208.36	Enhanced dedicated support	Brandonstad	0	Myanmar	2016-01-28 20:59:32	0
7	66.00	48	24593.33	131.76	Reactive local challenge	Port Jefferybury	1	Australia	2016-03-07 01:40:15	1
8	74.53	30	68862.00	221.51	Configurable coherent function	West Colin	1	Grenada	2016-04-18 09:33:42	0
9	69.88	20	55642.32	183.82	Mandatory homogeneous architecture	Ramirezton	1	Ghana	2016-07-11 01:42:51	0

Figure 3: first 10 record in the dataset

Some statistical summaries about the dataset:

```
In [229]: #find the mean of the dataset
          df.describe()
```

Out[229]:

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Male	Clicked on Ad
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000
mean	65.000200	36.009000	55000.000080	180.000100	0.481000	0.50000
std	15.853615	8.785562	13414.634022	43.902339	0.499889	0.50025
min	32.600000	19.000000	13996.500000	104.780000	0.000000	0.00000
25%	51.360000	29.000000	47031.802500	138.830000	0.000000	0.00000
50%	68.215000	35.000000	57012.300000	183.130000	0.000000	0.50000
75%	78.547500	42.000000	65470.635000	218.792500	1.000000	1.00000
max	91.430000	61.000000	79484.800000	269.960000	1.000000	1.00000

Figure 4: statistical summaries

Our dataset doesn't contain any missing values as shown below:

```
In [6]: df.isnull().sum()

Out[6]: Daily Time Spent on Site    0
        Age                         0
        Area Income                 0
        Daily Internet Usage        0
        Ad Topic Line               0
        City                        0
        Male                        0
        Country                     0
        Timestamp                   0
        Clicked on Ad               0
        dtype: int64
```

Figure 5: number of missing values

The variance of columns that contains numeric values:

```
In [7]: df.var()

Out[7]: Daily Time Spent on Site    2.513371e+02
        Age                         7.718611e+01
        Area Income                 1.799524e+08
        Daily Internet Usage        1.927415e+03
        Male                        2.498889e-01
        Clicked on Ad               2.502503e-01
        dtype: float64
```

Figure 6: calculation of the variance

5. DATA VISUALIZING

```
In [8]: df.hist(column='Daily Time Spent on Site')

Out[8]: array([[<AxesSubplot:title={'center':'Daily Time Spent on Site'}>]],
              dtype=object)
```

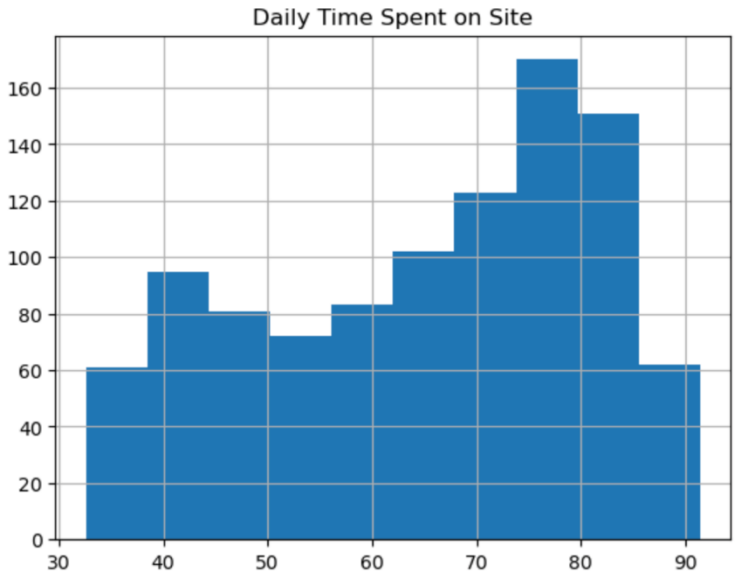

Figure 7: distribution of variable "Daily Time Spent on Site"

```
In [9]: df.hist(column='Age')

Out[9]: array([[<AxesSubplot:title={'center':'Age'}>]], dtype=object)
```

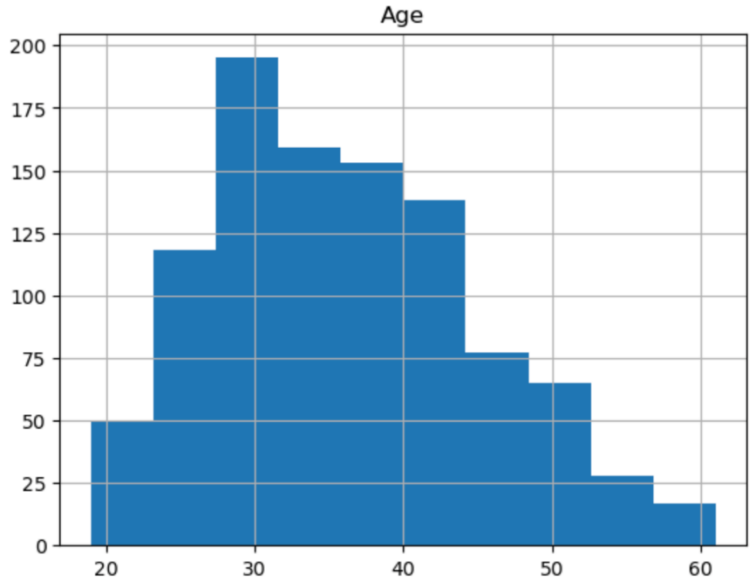

Figure 8: distribution of variable "Age"

```
In [10]: df.hist(column='Area Income')
```

```
Out[10]: array([[<AxesSubplot:title={'center':'Area Income'}>]], dtype=object)
```

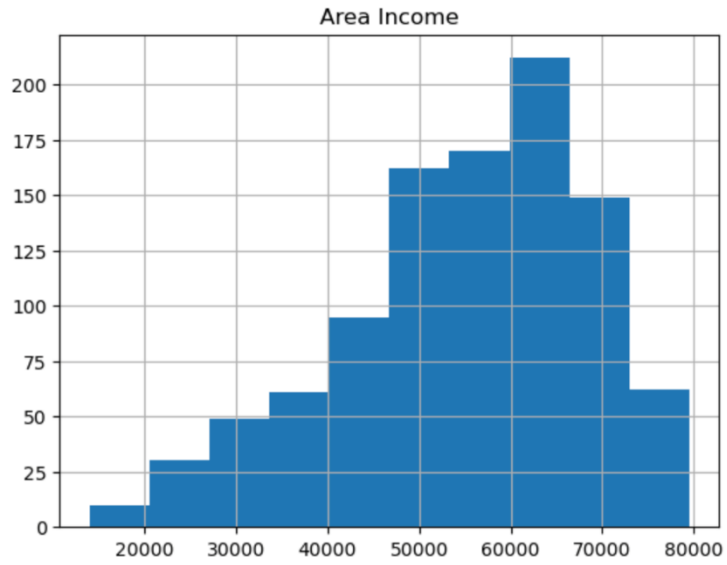

Figure 9: distribution of variable "Area Income"

```
In [11]: df.hist(column='Daily Internet Usage')
```

```
Out[11]: array([[<AxesSubplot:title={'center':'Daily Internet Usage'}>]],
       dtype=object)
```

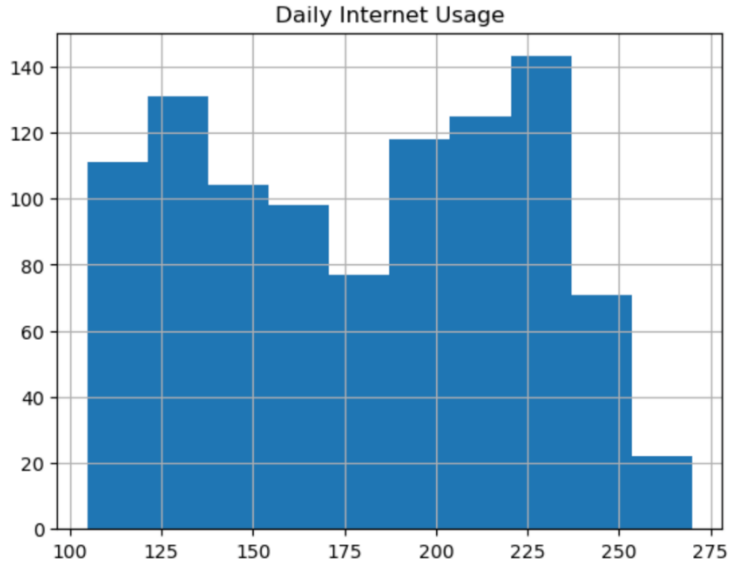

Figure 10: distribution of variable "Daily Internet Usage"

```
In [12]: df.hist(column='Male')
Out[12]: array([[<AxesSubplot:title={'center':'Male'}>]], dtype=object)
```

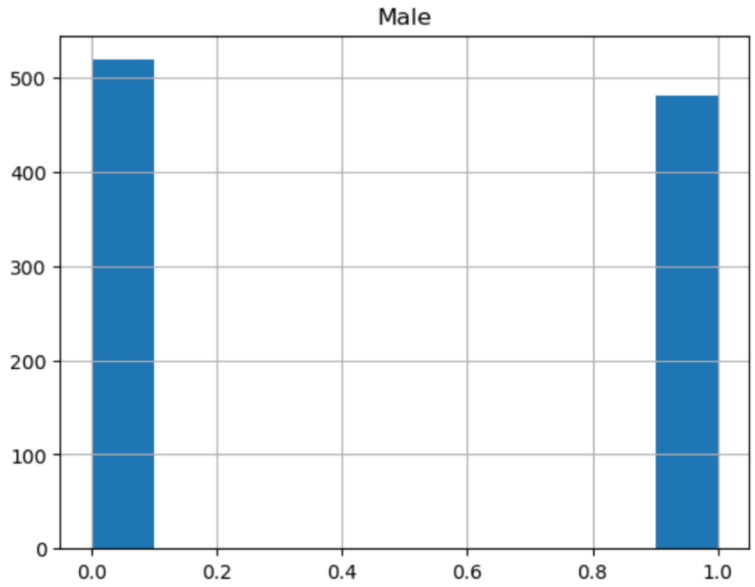

Figure 11: distribution of variable "Male"

```
In [13]: df.hist(column='Clicked on Ad')
Out[13]: array([[<AxesSubplot:title={'center':'Clicked on Ad'}>]], dtype=object)
```

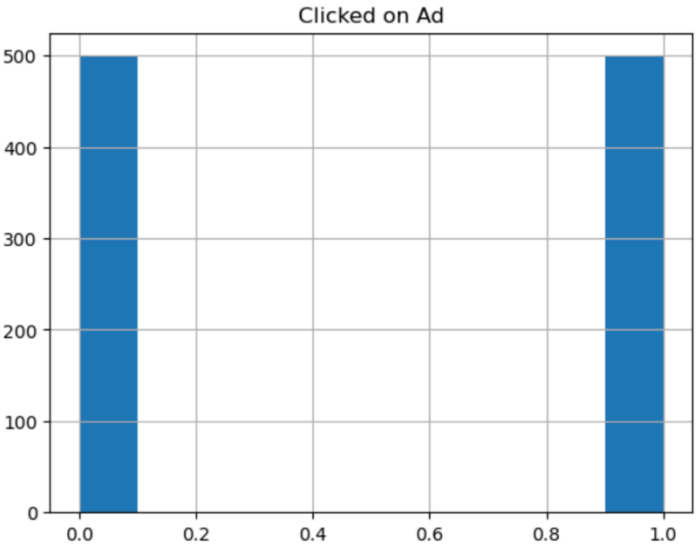

Figure 12: distribution of variable "Clicked on Ad"

```
In [17]: sns.jointplot(data=df, dropna= True)
         plt.show()
```

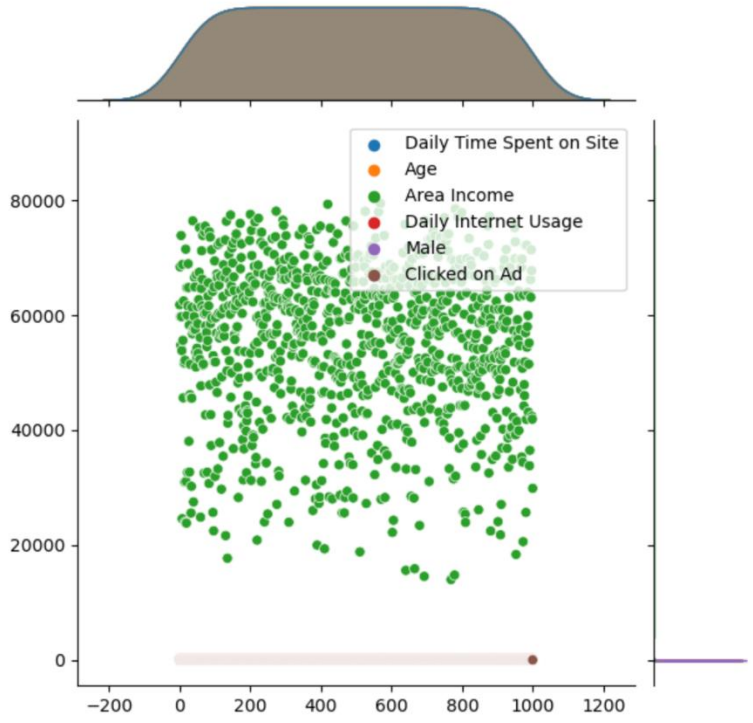

Figure 13: joint plot

6. DATA PREPROCESSING

Data preprocessing is important as it prepares the data and enhances the possible results.

Pre-Processing : DATA CLEANING Handling of Missing Data

```
# Any missing values?
df.isnull().values.any()
# Total missing values for each feature
df.isnull().sum()

# max occurence
frqMale = df['Male'].value_counts()

# Replace missing values with the value which has max occurence
df['Male'].fillna(frqMale, inplace=True)

df['Country'].fillna("Not given", inplace=True)
df['City'].fillna("Not given", inplace=True)

# second option is to drop these values --- lacking certain attributes of interest
df['Clicked on Ad'].dropna()

# Fill average values in place for nan, fill with mean
df['Daily Internet Usage'].fillna(df['Daily Internet Usage'].mean(), inplace=True)
df['Age'].fillna(df['Age'].mean(), inplace=True)
df['Area Income'].fillna(df['Area Income'].mean(), inplace=True)
```

[22] ✓ 0.1s

Figure 14: Handling of missing data

Pre-Processing : Data transformation: normalization and aggregation

```
# change the value to the correct one
df['Male'].replace(['no','yes'],[0,1],inplace=True)
```

[242] ✓ 0.0s

Figure 15: Data transformation for gender (Male) column

Data discretization is a part of data reduction, replacing numerical attributes with nominal ones. We choose internet usage column and put them into 3 buckets with labels (Below Average, Average, Above Average)

Data discretization: part of data reduction, replacing numerical attributes with nominal ones.

```
#GROUPING INTERNET USAGE RECORDS
df['Internet usage bucket']=pd.cut(df['Daily Internet Usage'],3,labels=['Below average','Average','Above average'])
df.head(10)
```

[343] ✓ 0.0s

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad	Internet usage bucket
0	68.95	35	61833.90	256.09	Cloned 5thgeneration orchestration	Wrightburgh	0	Tunisia	2016-03-27 00:53:11	0	Above average
1	80.23	31	68441.85	193.77	Monitored national standardization	West Jodi	1	Nauru	2016-04-04 01:39:02	0	Average
2	69.47	26	59785.94	236.50	Organic bottom-line service-desk	Davidton	0	San Marino	2016-03-13 20:35:42	0	Above average
3	74.15	29	54806.18	245.89	Triple-buffered reciprocal time-frame	West Terrifurt	1	Italy	2016-01-10 02:31:19	0	Above average
4	68.37	35	73889.99	225.58	Robust logistical utilization	South Manuel	0	Iceland	2016-06-03 03:36:18	0	Above average
5	59.99	23	59761.56	226.74	Sharable client-driven software	Jamieberg	1	Norway	2016-05-19 14:30:17	0	Above average
6	88.91	33	53852.85	208.36	Enhanced dedicated support	Brandonstad	0	Myanmar	2016-01-28 20:59:32	0	Average
7	66.00	48	24593.33	131.76	Reactive local challenge	Port Jefferybury	1	Australia	2016-03-07 01:40:15	1	Below average
8	74.53	30	68862.00	221.51	Configurable coherent function	West Colin	1	Grenada	2016-04-18 09:33:42	0	Above average
9	69.88	20	55642.32	183.82	Mandatory homogeneous architecture	Ramirezton	1	Ghana	2016-07-11 01:42:51	0	Average

Figure 16: Data Discretization for Daily internet usage column

Export Pandas DataFrame after pre-processing to a CSV File named “pre_advertisment”

```
#Export Pandas DataFrame after pre-processing to a CSV File
df.to_csv(r'C:\Ai-project\pr_advertisment.csv')
```

[244] ✓ 0.0s

Figure 17: Save the dataframe

7. REFERENCES

1. How to plot a histogram in Python. Available at: https://www.nbshare.io/notebook/204214467/How-to-Plot-a-Histogram-in-Python/
2. Joint plot in python - javatpoint (no date) www.javatpoint.com. Available at: https://www.javatpoint.com/joint-plot-in-python