Freelance Platform Project

1.Perform necessary EDA on the data.

```
import pandas as pd
df=pd.read_csv('Freelance Platform Projects.csv')
df.head()  #displays the first five rows
```

```
                                    Title         Category Name
Experience  \
0  Banner images for web desgin websites                Design
Entry ($)
1    Make my picture a solid silhouette   Video, Photo & Image
Entry ($)
2                      Bookkeeper needed              Business
Entry ($)
3                     Accountant needed               Business
Entry ($)
4          Guest Post on High DA Website     Digital Marketing  Expert
($$$)

           Sub Category Name Currency   Budget Location  \
0             Graphic Design      EUR     60.0   remote
1              Image Editing      GBP     20.0   remote
2       Finance & Accounting      GBP     12.0   remote
3  Tax Consulting & Advising      GBP     14.0   remote
4                        SEO      USD  10000.0   remote

  Freelancer Preferred From         Type          Date Posted  \
0                       ALL  fixed_price  2023-04-29 18:06:39
1                       ALL  fixed_price  2023-04-29 17:40:28
2                       ALL  fixed_price  2023-04-29 17:40:06
3                       ALL  fixed_price  2023-04-29 17:32:01
4                       ALL  fixed_price  2023-04-29 17:09:36

                                        Description Duration  \
0  We are looking to improve the banner images on...      NaN
1  Hello \n\nI need a quick designer to make 4 pi...      NaN
2  Hi - I need a bookkeeper to assist with bookke...      NaN
3  Hi - I need an accountant to assist me with un...      NaN
4  Hi, I am currently running a project where I w...      NaN

  Client Registration Date Client City  Client Country Client Currency
\
0               2010-11-03      Dublin         Ireland              EUR
```

```
1               2017-02-21      London  United Kingdom              GBP

2               2023-04-09      London  United Kingdom              GBP

3               2023-04-09      London  United Kingdom              GBP

4               2016-07-01      Mumbai           India              USD


    Client Job Title
0     PPC Management
1     Office manager
2          Paralegal
3          Paralegal
4  Guest posts buyer
```

```
df.tail() #displays the last five rows
```

```
                                                    Title  \
12217  Published Travel Writer required for content c...
12218  Shopify - Filtering Work (Product Selection/No...
12219                                  Simple SQL Query
12220  Create a Carbon, Water, Waste Calculating plat...
12221                                COMPANY REGISTERS

                   Category Name         Experience          Sub
Category Name  \
12217     Writing & Translation          Entry ($)            Content
Writing
12218                    Design  Intermediate ($$)                 Web
Design
12219  Technology & Programming          Entry ($)    Data Science &
Analysis
12220                    Design       Expert ($$$)                 Web
Design
12221                  Business       Expert ($$$)  Administration
Assistance

      Currency  Budget        Location Freelancer Preferred From
Type  \
12217      GBP    50.0          remote                       ALL
fixed_price
12218      GBP    65.0  remote_country                        GB
fixed_price
12219      GBP    50.0          remote                       ALL
fixed_price
12220      USD    39.0          remote                       ALL
hourly
12221      GBP    75.0          remote                       ALL
fixed_price
```

```
                Date Posted
Description  \
12217  2023-01-18 19:23:01  I am looking for a published travel writer
to ...
12218  2023-01-18 19:18:48  On our website www.juicebitz.co.uk we have
add...
12219  2023-01-18 19:18:48  I need someone to write a quick SQL query
on a...
12220  2023-01-18 19:18:47  I am seeking a full stack web developer
who sp...
12221  2023-01-18 19:18:47  Hi, the following administrative task
would be...

            Duration Client Registration Date Client City  Client
Country  \
12217            NaN               2011-06-06   Amsterdam
Netherlands
12218  1 day or less               2022-03-23       Filey  United
Kingdom
12219            NaN               2022-03-14      London  United
Kingdom
12220            NaN               2013-07-21       Noida
India
12221            NaN               2020-09-21       Grays  United
Kingdom

      Client Currency                                      Client Job
Title
12217             GBP                                      Wordpress
Expert
12218             GBP
Director
12219             GBP
NaN
12220             USD  Google Adwords, Pay Per Click, Google
Shopping...
12221             GBP
NaN
```

```
df.shape #no of rows and columns
```

```
(12222, 17)
```

```
df.info()
# Number of rows: 205
# Number of columns: 15
# For every column,
# => Column name
# => Number on Non null rows
```

```
# => Number of null values = Total rows - Non null rows
# => Data Type
# Number of colums for each data type
# Memory Usage

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12222 entries, 0 to 12221
Data columns (total 17 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Title                        12222 non-null  object
 1   Category Name                12222 non-null  object
 2   Experience                   12222 non-null  object
 3   Sub Category Name            12222 non-null  object
 4   Currency                     12222 non-null  object
 5   Budget                       12222 non-null  float64
 6   Location                     12222 non-null  object
 7   Freelancer Preferred From    12222 non-null  object
 8   Type                         12222 non-null  object
 9   Date Posted                  12222 non-null  object
 10  Description                  12222 non-null  object
 11  Duration                     1602 non-null   object
 12  Client Registration Date     12222 non-null  object
 13  Client City                  12222 non-null  object
 14  Client Country               12222 non-null  object
 15  Client Currency              12222 non-null  object
 16  Client Job Title             4588 non-null   object
dtypes: float64(1), object(16)
memory usage: 1.6+ MB
```

Handling Missing Values

```
df.isnull().sum()

Title                              0
Category Name                      0
Experience                         0
Sub Category Name                  0
Currency                           0
Budget                             0
Location                           0
Freelancer Preferred From          0
Type                               0
Date Posted                        0
Description                        0
Duration                       10620
Client Registration Date           0
Client City                        0
```

```
Client Country                    0
Client Currency                   0
Client Job Title               7634
dtype: int64
```

```
#Deletion
newdf = df.dropna(axis=0)
newdf.isnull().sum()
```

```
Title                          0
Category Name                  0
Experience                     0
Sub Category Name              0
Currency                       0
Budget                         0
Location                       0
Freelancer Preferred From      0
Type                           0
Date Posted                    0
Description                    0
Duration                       0
Client Registration Date       0
Client City                    0
Client Country                 0
Client Currency                0
Client Job Title               0
dtype: int64
```

```
df.isnull().sum()
```

```
Title                          0
Category Name                  0
Experience                     0
Sub Category Name              0
Currency                       0
Budget                         0
Location                       0
Freelancer Preferred From      0
Type                           0
Date Posted                    0
Description                    0
Duration                   10620
Client Registration Date       0
Client City                    0
Client Country                 0
Client Currency                0
Client Job Title            7634
dtype: int64
```

```
# Imputation

m = df['Budget'].mean()
m
```

```
229.22148584519718
```

```
missing_rows = df.index[df['Budget'].isna()==True]
missing_rows
```

```
Int64Index([], dtype='int64')
```

```
df['Budget'].fillna(m, inplace=True)
df.isnull().sum()
```

```
Title                              0
Category Name                      0
Experience                         0
Sub Category Name                  0
Currency                           0
Budget                             0
Location                           0
Freelancer Preferred From          0
Type                               0
Date Posted                        0
Description                        0
Duration                       10620
Client Registration Date           0
Client City                        0
Client Country                     0
Client Currency                    0
Client Job Title                7634
dtype: int64
```

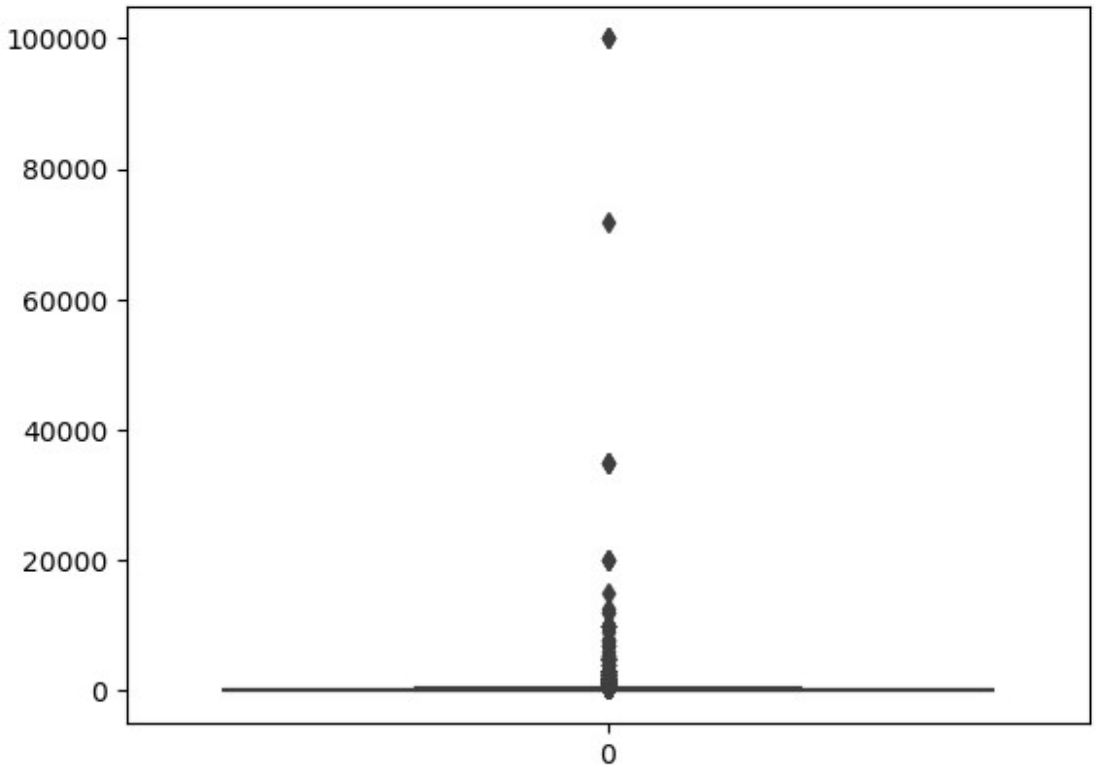
```
df.iloc[missing_rows, :]
```

```
Empty DataFrame
Columns: [Title, Category Name, Experience, Sub Category Name,
Currency, Budget, Location, Freelancer Preferred From, Type, Date
Posted, Description, Duration, Client Registration Date, Client City,
Client Country, Client Currency, Client Job Title]
Index: []
```

Handling Outliers

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

```
sns.boxplot(df['Budget'])
plt.show()
```

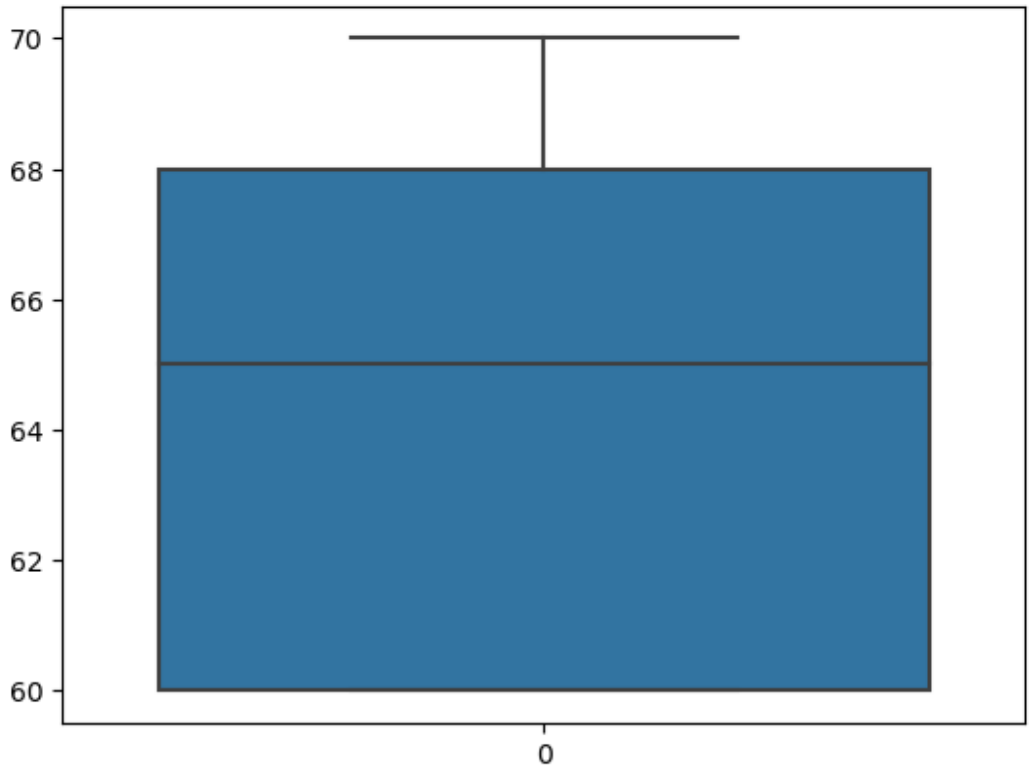



```
outliers = df[(df['Budget']>70)|(df['Budget']<60)].index
outliers
```

```
Int64Index([    1,     2,     3,     4,     5,     6,     7,     8,
9,
               10,
            ...
            12211, 12212, 12213, 12214, 12215, 12216, 12217, 12219,
12220,
            12221],
           dtype='int64', length=11549)
```

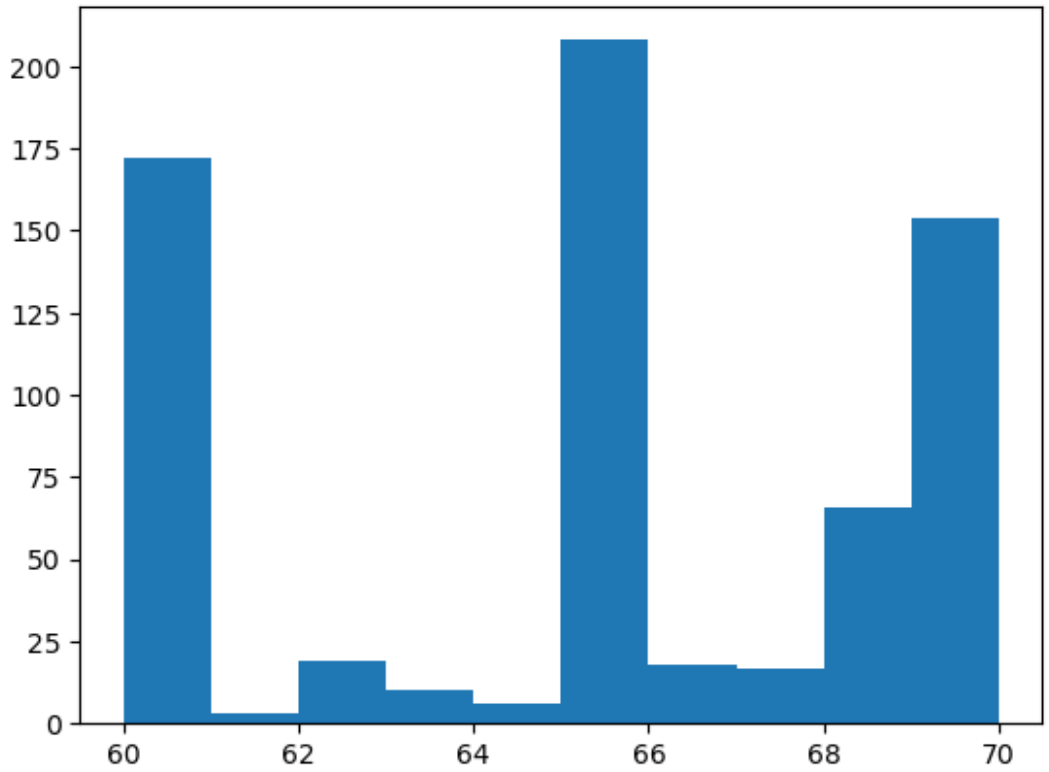
```
df.drop(outliers, axis=0, inplace=True)

sns.boxplot(df['Budget'])
plt.show()
```



Handling Skewness

```
plt.hist(df['Budget'])
plt.show()
```




```
df['Budget'].skew()
```

```
-0.12243982079446392
```

```
from scipy.stats import boxcox

result = boxcox(df['Budget'])[0]
result
```

```
array([3108.20540566, 3692.34570333, 3692.34570333, 3108.20540566,
       4330.60002333, 3692.34570333, 4068.76468272, 3692.34570333,
       3692.34570333, 3692.34570333, 3692.34570333, 3692.34570333,
       3692.34570333, 3108.20540566, 3108.20540566, 3692.34570333,
       3452.23141331, 3692.34570333, 4330.60002333, 4330.60002333,
       4330.60002333, 4330.60002333, 4330.60002333, 4330.60002333,
       3692.34570333, 3692.34570333, 3692.34570333, 3692.34570333,
       4330.60002333, 3108.20540566, 3108.20540566, 4068.76468272,
       4068.76468272, 4068.76468272, 4068.76468272, 3108.20540566,
       4068.76468272, 3692.34570333, 3335.40856243, 3692.34570333,
       3815.64735182, 3108.20540566, 3692.34570333, 4330.60002333,
       3692.34570333, 4068.76468272, 4068.76468272, 3692.34570333,
       4068.76468272, 4068.76468272, 4068.76468272, 4068.76468272,
       4068.76468272, 4330.60002333, 4330.60002333, 3108.20540566,
       4330.60002333, 4330.60002333, 4068.76468272, 4068.76468272,
       4330.60002333, 4068.76468272, 4068.76468272, 3108.20540566,
       3335.40856243, 3692.34570333, 3108.20540566, 3108.20540566,
```

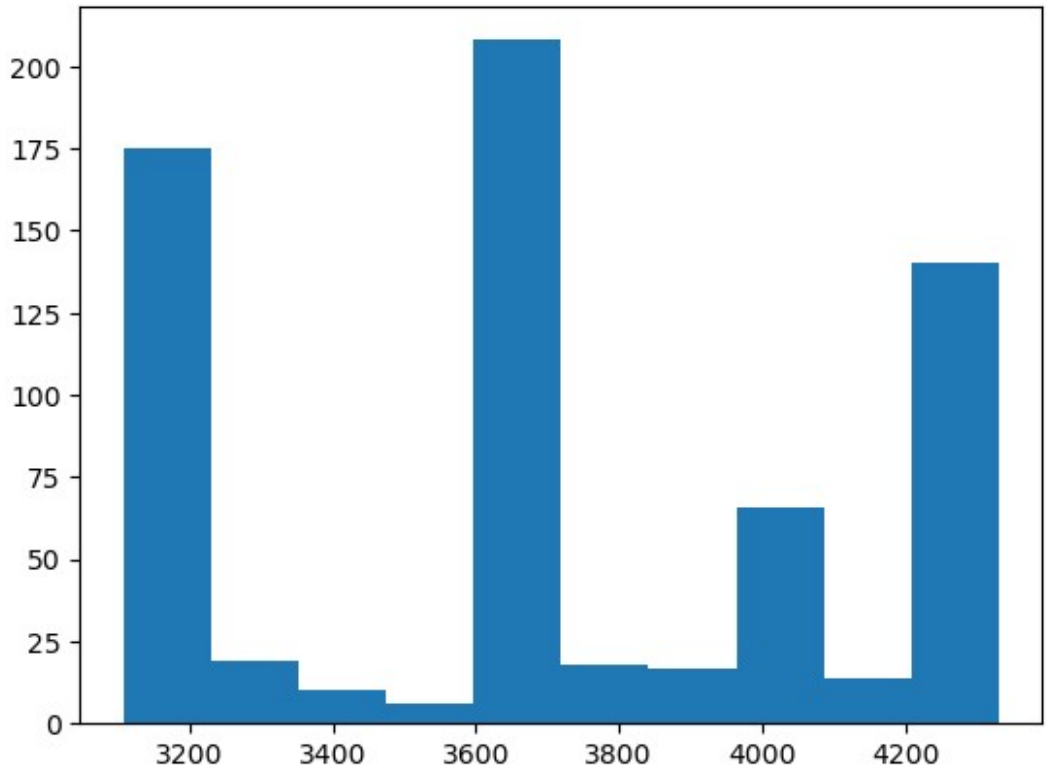
```
      4068.76468272, 4068.76468272, 4068.76468272, 4068.76468272,
      4068.76468272, 4068.76468272, 3108.20540566, 3108.20540566,
      3108.20540566, 4068.76468272, 4068.76468272, 3692.34570333,
      4068.76468272, 3815.64735182, 3692.34570333, 3108.20540566,
      4068.76468272, 3108.20540566, 3692.34570333, 3692.34570333,
      4068.76468272, 4068.76468272, 4068.76468272, 4068.76468272,
      4068.76468272, 4330.60002333, 4330.60002333, 4068.76468272,
      4068.76468272, 3692.34570333, 3571.20874744, 3692.34570333,
      4330.60002333, 4068.76468272, 4068.76468272, 4068.76468272,
      4068.76468272, 4068.76468272, 3108.20540566, 4330.60002333,
      3692.34570333, 4330.60002333, 3692.34570333, 4198.59018507,
      4198.59018507, 4198.59018507, 3692.34570333, 4198.59018507,
      3335.40856243, 3108.20540566, 4198.59018507, 3692.34570333,
      4330.60002333, 3692.34570333, 3692.34570333, 3692.34570333,
      4198.59018507, 4330.60002333, 3108.20540566, 3108.20540566,
      4198.59018507, 4330.60002333, 3692.34570333, 3108.20540566,
      3692.34570333, 3692.34570333, 4330.60002333, 3692.34570333,
      4068.76468272, 4330.60002333, 3108.20540566, 4330.60002333,
      3108.20540566, 3941.11869792, 4330.60002333, 3108.20540566,
      4330.60002333, 3108.20540566, 4330.60002333, 3692.34570333,
      4330.60002333, 3692.34570333, 4330.60002333, 4330.60002333,
      4198.59018507, 3108.20540566, 3108.20540566, 3692.34570333,
      3692.34570333, 4198.59018507, 4068.76468272, 3692.34570333,
      3692.34570333, 3692.34570333, 3108.20540566, 4068.76468272,
      3108.20540566, 4068.76468272, 3108.20540566, 3692.34570333,
      3108.20540566, 4068.76468272, 3108.20540566, 3108.20540566,
      3692.34570333, 4330.60002333, 3692.34570333, 4330.60002333,
      4330.60002333, 4068.76468272, 4068.76468272, 4068.76468272,
      4330.60002333, 3692.34570333, 3692.34570333, 3692.34570333,
      3452.23141331, 3571.20874744, 4068.76468272, 3108.20540566,
      3692.34570333, 3692.34570333, 4068.76468272, 4330.60002333,
      3220.73498655, 3692.34570333, 3692.34570333, 3108.20540566,
      3108.20540566, 3815.64735182, 3108.20540566, 4330.60002333,
      3108.20540566, 3108.20540566, 3108.20540566, 3571.20874744,
      4330.60002333, 3692.34570333, 3108.20540566, 4330.60002333,
      3692.34570333, 3692.34570333, 3692.34570333, 3692.34570333,
      3335.40856243, 3571.20874744, 3108.20540566, 4330.60002333,
      3692.34570333, 3692.34570333, 3692.34570333, 3108.20540566,
      3108.20540566, 3692.34570333, 3692.34570333, 3452.23141331,
      3692.34570333, 3452.23141331, 3335.40856243, 3335.40856243,
      3108.20540566, 3108.20540566, 3692.34570333, 4198.59018507,
      4330.60002333, 4330.60002333, 3692.34570333, 3692.34570333,
      4330.60002333, 4068.76468272, 4068.76468272, 4330.60002333,
      3692.34570333, 3692.34570333, 3108.20540566, 3108.20540566,
      4330.60002333, 4330.60002333, 4330.60002333, 4330.60002333,
      3692.34570333, 3692.34570333, 4198.59018507, 3692.34570333,
      4068.76468272, 4068.76468272, 4068.76468272, 4330.60002333,
      4330.60002333, 3108.20540566, 3692.34570333, 3692.34570333,
      3692.34570333, 4330.60002333, 4330.60002333, 3692.34570333,
```

```
      3692.34570333, 3108.20540566, 3108.20540566, 3692.34570333,
      4330.60002333, 3108.20540566, 3941.11869792, 3335.40856243,
      3108.20540566, 3692.34570333, 3692.34570333, 3108.20540566,
      3452.23141331, 3941.11869792, 3692.34570333, 3108.20540566,
      3108.20540566, 3692.34570333, 3692.34570333, 3692.34570333,
      3692.34570333, 4330.60002333, 3108.20540566, 3692.34570333,
      4330.60002333, 3108.20540566, 4330.60002333, 3692.34570333,
      3692.34570333, 3108.20540566, 3692.34570333, 3108.20540566,
      3941.11869792, 3692.34570333, 4330.60002333, 4330.60002333,
      3108.20540566, 3941.11869792, 3108.20540566, 3815.64735182,
      3452.23141331, 3108.20540566, 3692.34570333, 3692.34570333,
      3815.64735182, 3941.11869792, 3108.20540566, 4330.60002333,
      3692.34570333, 4330.60002333, 3108.20540566, 3108.20540566,
      3108.20540566, 3108.20540566, 3108.20540566, 4330.60002333,
      4330.60002333, 3692.34570333, 3108.20540566, 3692.34570333,
      4330.60002333, 4330.60002333, 3108.20540566, 3692.34570333,
      3692.34570333, 3692.34570333, 3108.20540566, 3108.20540566,
      3692.34570333, 3108.20540566, 3108.20540566, 3108.20540566,
      3108.20540566, 3108.20540566, 4330.60002333, 3941.11869792,
      4330.60002333, 3220.73498655, 3108.20540566, 4068.76468272,
      3692.34570333, 4330.60002333, 3692.34570333, 3692.34570333,
      3335.40856243, 3692.34570333, 3692.34570333, 3108.20540566,
      3692.34570333, 3452.23141331, 3108.20540566, 3108.20540566,
      4330.60002333, 3692.34570333, 3692.34570333, 3108.20540566,
      3692.34570333, 3692.34570333, 3815.64735182, 3108.20540566,
      4330.60002333, 3692.34570333, 4330.60002333, 4330.60002333,
      3692.34570333, 3692.34570333, 3692.34570333, 3108.20540566,
      3692.34570333, 3692.34570333, 4330.60002333, 3108.20540566,
      4330.60002333, 3692.34570333, 4330.60002333, 3692.34570333,
      3815.64735182, 3815.64735182, 3108.20540566, 3692.34570333,
      3941.11869792, 4198.59018507, 3108.20540566, 3692.34570333,
      3692.34570333, 3815.64735182, 3815.64735182, 3692.34570333,
      4330.60002333, 4330.60002333, 3108.20540566, 4330.60002333,
      3692.34570333, 3692.34570333, 4330.60002333, 3108.20540566,
      3108.20540566, 3108.20540566, 4068.76468272, 3108.20540566,
      3108.20540566, 4330.60002333, 4330.60002333, 3108.20540566,
      3692.34570333, 3108.20540566, 3108.20540566, 3108.20540566,
      3108.20540566, 3108.20540566, 4330.60002333, 3108.20540566,
      4330.60002333, 4330.60002333, 4330.60002333, 3108.20540566,
      3108.20540566, 3692.34570333, 3692.34570333, 4330.60002333,
      3692.34570333, 3692.34570333, 3108.20540566, 3692.34570333,
      3692.34570333, 3108.20540566, 3108.20540566, 3941.11869792,
      3941.11869792, 4330.60002333, 3692.34570333, 3108.20540566,
      3692.34570333, 4330.60002333, 3692.34570333, 4330.60002333,
      3692.34570333, 3941.11869792, 3108.20540566, 3692.34570333,
      3692.34570333, 4330.60002333, 3108.20540566, 3108.20540566,
      3692.34570333, 3692.34570333, 3692.34570333, 3335.40856243,
      3335.40856243, 3108.20540566, 3692.34570333, 4068.76468272,
      3692.34570333, 4330.60002333, 3108.20540566, 4330.60002333,
```

```
      3108.20540566, 3692.34570333, 3692.34570333, 4330.60002333,
      3335.40856243, 3108.20540566, 4330.60002333, 3108.20540566,
      3692.34570333, 3692.34570333, 3692.34570333, 3108.20540566,
      3815.64735182, 3692.34570333, 4330.60002333, 3108.20540566,
      3108.20540566, 4198.59018507, 4330.60002333, 4330.60002333,
      4330.60002333, 3692.34570333, 4330.60002333, 3692.34570333,
      3108.20540566, 4330.60002333, 4330.60002333, 3692.34570333,
      3108.20540566, 3108.20540566, 4330.60002333, 3692.34570333,
      3108.20540566, 3108.20540566, 3108.20540566, 4330.60002333,
      3692.34570333, 3692.34570333, 3108.20540566, 4330.60002333,
      3941.11869792, 3692.34570333, 4330.60002333, 4330.60002333,
      3692.34570333, 3692.34570333, 3941.11869792, 4330.60002333,
      3108.20540566, 3692.34570333, 4330.60002333, 3335.40856243,
      4068.76468272, 3335.40856243, 3335.40856243, 3941.11869792,
      3108.20540566, 3692.34570333, 4330.60002333, 3692.34570333,
      3692.34570333, 4330.60002333, 3692.34570333, 3815.64735182,
      3108.20540566, 3692.34570333, 4330.60002333, 3941.11869792,
      3108.20540566, 3108.20540566, 3571.20874744, 3692.34570333,
      3108.20540566, 3692.34570333, 3335.40856243, 3692.34570333,
      3108.20540566, 3692.34570333, 3692.34570333, 3815.64735182,
      3815.64735182, 3692.34570333, 4330.60002333, 4330.60002333,
      3692.34570333, 3108.20540566, 3815.64735182, 4330.60002333,
      3692.34570333, 3692.34570333, 3692.34570333, 3108.20540566,
      3815.64735182, 3108.20540566, 3692.34570333, 3108.20540566,
      3692.34570333, 3108.20540566, 3941.11869792, 3108.20540566,
      3692.34570333, 3108.20540566, 3941.11869792, 3692.34570333,
      3108.20540566, 4330.60002333, 4330.60002333, 3692.34570333,
      3108.20540566, 3108.20540566, 3692.34570333, 4330.60002333,
      3692.34570333, 3692.34570333, 3108.20540566, 3108.20540566,
      4330.60002333, 4330.60002333, 3692.34570333, 3335.40856243,
      3692.34570333, 3571.20874744, 4330.60002333, 4330.60002333,
      3108.20540566, 4330.60002333, 3815.64735182, 3108.20540566,
      3335.40856243, 3108.20540566, 4330.60002333, 3692.34570333,
      3108.20540566, 3108.20540566, 4330.60002333, 3692.34570333,
      3692.34570333, 3692.34570333, 4068.76468272, 3692.34570333,
      3692.34570333, 3692.34570333, 3108.20540566, 3108.20540566,
      3692.34570333, 4330.60002333, 3108.20540566, 3692.34570333,
      3692.34570333, 3108.20540566, 4330.60002333, 3692.34570333,
      3692.34570333, 3108.20540566, 3108.20540566, 4330.60002333,
      4330.60002333, 4068.76468272, 4068.76468272, 4068.76468272,
      4068.76468272, 3108.20540566, 4330.60002333, 3108.20540566,
      3692.34570333, 4330.60002333, 3108.20540566, 4330.60002333,
      4330.60002333, 3108.20540566, 3815.64735182, 3108.20540566,
      3335.40856243, 4330.60002333, 3692.34570333, 3108.20540566,
      3108.20540566, 3108.20540566, 4330.60002333, 3692.34570333,
      4330.60002333, 4330.60002333, 4330.60002333, 3335.40856243,
      3452.23141331, 3692.34570333, 3692.34570333, 3108.20540566,
      3452.23141331, 3692.34570333, 4330.60002333, 3108.20540566,
      3692.34570333, 4068.76468272, 4330.60002333, 3108.20540566,
```

```
       4330.60002333, 3108.20540566, 3692.34570333, 4330.60002333,
       3108.20540566, 3452.23141331, 3692.34570333, 3692.34570333,
       4068.76468272, 3108.20540566, 4198.59018507, 3108.20540566,
       3692.34570333, 4068.76468272, 3108.20540566, 3220.73498655,
       3692.34570333])

plt.hist(result)
plt.show()
```




```
pd.Series(result).skew()

-0.051698886498172676

df['Budget'] = result
df['Budget'].head()

0     3108.205406
29    3692.345703
46    3692.345703
49    3108.205406
88    4330.600023
Name: Budget, dtype: float64
```

Categorical Data Encoding

```
df.dtypes
```

```
Title                          object
Category Name                  object
Experience                     object
Sub Category Name              object
Currency                       object
Budget                        float64
Location                       object
Freelancer Preferred From      object
Type                           object
Date Posted                    object
Description                    object
Duration                       object
Client Registration Date       object
Client City                    object
Client Country                 object
Client Currency                object
Client Job Title               object
dtype: object
```

```
df['Budget'].unique()
```

```
array([3108.20540566, 3692.34570333, 4330.60002333, 4068.76468272,
       3452.23141331, 3335.40856243, 3815.64735182, 3571.20874744,
       4198.59018507, 3941.11869792, 3220.73498655])
```

```
one_hot = pd.get_dummies(df['Budget'], dtype='int')
one_hot.head()
```

```
    3108.205406  3220.734987  3335.408562  3452.231413  3571.208747  \
0             1            0            0            0            0
29            0            0            0            0            0
46            0            0            0            0            0
49            1            0            0            0            0
88            0            0            0            0            0

    3692.345703  3815.647352  3941.118698  4068.764683  4198.590185  \
0             0            0            0            0            0
29            1            0            0            0            0
46            1            0            0            0            0
49            0            0            0            0            0
88            0            0            0            0            0

    4330.600023
0             0
29            0
46            0
```

```
49            0
88            1
```

```
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()
encoding = encoder.fit_transform(df['Budget'])
encoding[:5]
```

```
array([ 0,  5,  5,  0, 10], dtype=int64)
```

```
df['Budget'].head()
```

```
0     3108.205406
29    3692.345703
46    3692.345703
49    3108.205406
88    4330.600023
Name: Budget, dtype: float64
```

```
df['Budget'] = encoding
df['Budget'].head()
```

```
0      0
29     5
46     5
49     0
88    10
Name: Budget, dtype: int64
```

Data Normalization or Scaling

```
df['Budget'].mean()
```

```
5.075780089153046
```

```
df['Budget'].std()
```

```
3.6525299482681506
```

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
result = scaler.fit_transform(df[['Budget']])
result[:5]
```

```
array([[-1.39069505],
       [-0.02076272],
       [-0.02076272],
```

```
       [-1.39069505],
       [ 1.34916961]])
```

```
result.mean()
```

```
3.1673524625264494e-17
```

```
result.std()
```

```
0.9999999999999998
```

```
df['Budget'].min()
```

```
0
```

```
df['Budget'].max()
```

```
10
```

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
result = scaler.fit_transform(df[['Budget']])
result[:5]
```

```
array([[0. ],
       [0.5],
       [0.5],
       [0. ],
       [1. ]])
```

```
result.min()
```

```
0.0
```

```
result.max()
```

```
1.0
```

2.Use machine learning to create clusters of similar projects.

```
#displays the first five rows
import pandas as pd
import numpy as np
df=pd.read_csv('Freelance Platform Projects.csv')
df.head()
```

```
                                    Title          Category Name
Experience  \
```

```
0  Banner images for web desgin websites                Design
Entry ($)
1    Make my picture a solid silhouette   Video, Photo & Image
Entry ($)
2                       Bookkeeper needed              Business
Entry ($)
3                      Accountant needed               Business
Entry ($)
4          Guest Post on High DA Website     Digital Marketing  Expert
($$$)

           Sub Category Name Currency   Budget Location  \
0             Graphic Design      EUR     60.0   remote
1              Image Editing      GBP     20.0   remote
2       Finance & Accounting      GBP     12.0   remote
3  Tax Consulting & Advising      GBP     14.0   remote
4                        SEO      USD  10000.0   remote

  Freelancer Preferred From         Type          Date Posted  \
0                      ALL  fixed_price  2023-04-29 18:06:39
1                      ALL  fixed_price  2023-04-29 17:40:28
2                      ALL  fixed_price  2023-04-29 17:40:06
3                      ALL  fixed_price  2023-04-29 17:32:01
4                      ALL  fixed_price  2023-04-29 17:09:36

                                          Description Duration  \
0  We are looking to improve the banner images on...      NaN
1  Hello \n\nI need a quick designer to make 4 pi...      NaN
2  Hi - I need a bookkeeper to assist with bookke...      NaN
3  Hi - I need an accountant to assist me with un...      NaN
4  Hi, I am currently running a project where I w...      NaN

  Client Registration Date Client City  Client Country Client Currency
\
0               2010-11-03      Dublin         Ireland              EUR

1               2017-02-21      London  United Kingdom              GBP

2               2023-04-09      London  United Kingdom              GBP

3               2023-04-09      London  United Kingdom              GBP

4               2016-07-01      Mumbai           India              USD


    Client Job Title
0     PPC Management
1     Office manager
2          Paralegal
```

```
3          Paralegal
4  Guest posts buyer
```

```
df.isna().sum()
```

```
Title                            0
Category Name                    0
Experience                       0
Sub Category Name                0
Currency                         0
Budget                           0
Location                         0
Freelancer Preferred From        0
Type                             0
Date Posted                      0
Description                      0
Duration                     10620
Client Registration Date         0
Client City                      0
Client Country                   0
Client Currency                  0
Client Job Title              7634
dtype: int64
```

```
newdf = df.dropna(axis=0)
newdf.isnull().sum()
```

```
Title                        0
Category Name                0
Experience                   0
Sub Category Name            0
Currency                     0
Budget                       0
Location                     0
Freelancer Preferred From    0
Type                         0
Date Posted                  0
Description                  0
Duration                     0
Client Registration Date     0
Client City                  0
Client Country               0
Client Currency              0
Client Job Title             0
dtype: int64
```

```
df.dtypes
```

```
Title                          object
Category Name                  object
Experience                     object
```

```
Sub Category Name              object
Currency                       object
Budget                        float64
Location                       object
Freelancer Preferred From      object
Type                           object
Date Posted                    object
Description                    object
Duration                       object
Client Registration Date       object
Client City                    object
Client Country                 object
Client Currency                object
Client Job Title               object
dtype: object
```

```
obj_cols = df.select_dtypes('object').columns
obj_cols
```

```
Index(['Title', 'Category Name', 'Experience', 'Sub Category Name',
'Currency',
       'Location', 'Freelancer Preferred From', 'Type', 'Date Posted',
       'Description', 'Duration', 'Client Registration Date', 'Client
City',
       'Client Country', 'Client Currency', 'Client Job Title'],
      dtype='object')
```

```
from sklearn.preprocessing import LabelEncoder

for col in obj_cols:
    encoder = LabelEncoder()
    df[col] = encoder.fit_transform(df[col])

df.dtypes
```

```
Title                          int32
Category Name                  int32
Experience                     int32
Sub Category Name              int32
Currency                       int32
Budget                        float64
Location                       int32
Freelancer Preferred From      int32
Type                           int32
Date Posted                    int32
Description                    int32
Duration                       int32
Client Registration Date       int32
Client City                    int32
Client Country                 int32
```

```
Client Currency                int32
Client Job Title               int32
dtype: object
```

```
from sklearn.model_selection import train_test_split

xtrain,xtest = train_test_split(df, train_size=0.8,
                                random_state=0)

print('xtrain shape=',xtrain.shape)
print('xtest shape=',xtest.shape)
```

```
xtrain shape= (9777, 17)
xtest shape= (2445, 17)
```

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
xtrain = scaler.fit_transform(xtrain)
xtest = scaler.transform(xtest)
```

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)
xtrain = pca.fit_transform(xtrain)
xtest = pca.transform(xtest)
xtrain.shape
```

```
(9777, 2)
```

```
xtest.shape
```

```
(2445, 2)
```

```
sum(pca.explained_variance_ratio_)
```

```
0.21918276106271065
```

```
from sklearn.cluster import KMeans

wcss_list = []

for i in range(1,11):
    model = KMeans(n_clusters=i)
    model.fit(xtrain)
    wcss = model.inertia_
    wcss_list.append(wcss)

wcss_list
```

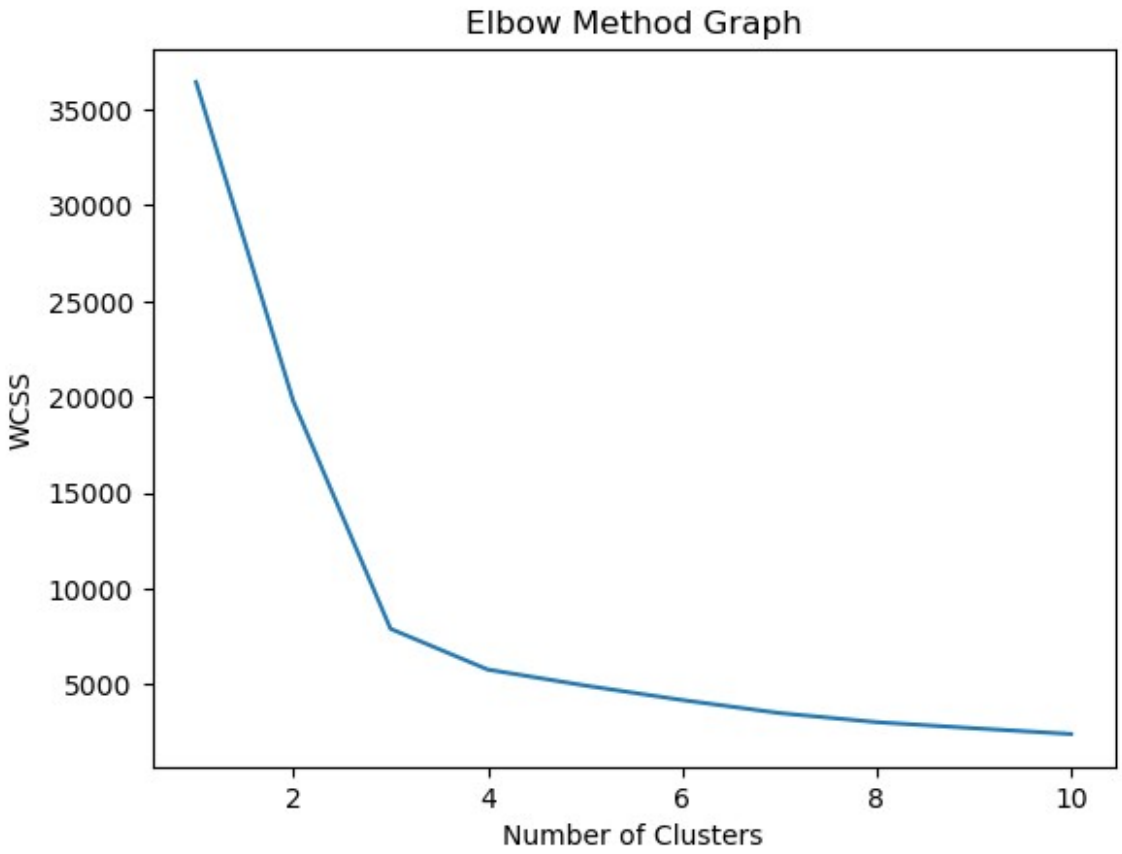
```
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
```

```
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
```

```
[36430.14753347206,
 19778.319970460754,
 7906.855029195654,
 5779.721048484167,
 4946.664933476803,
 4189.647039362779,
 3510.687396950805,
 3035.4287668136035,
 2719.9308476679726,
 2417.4875864058404]
```

```
import matplotlib.pyplot as plt

plt.plot(range(1,11), wcss_list)
plt.title('Elbow Method Graph')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```

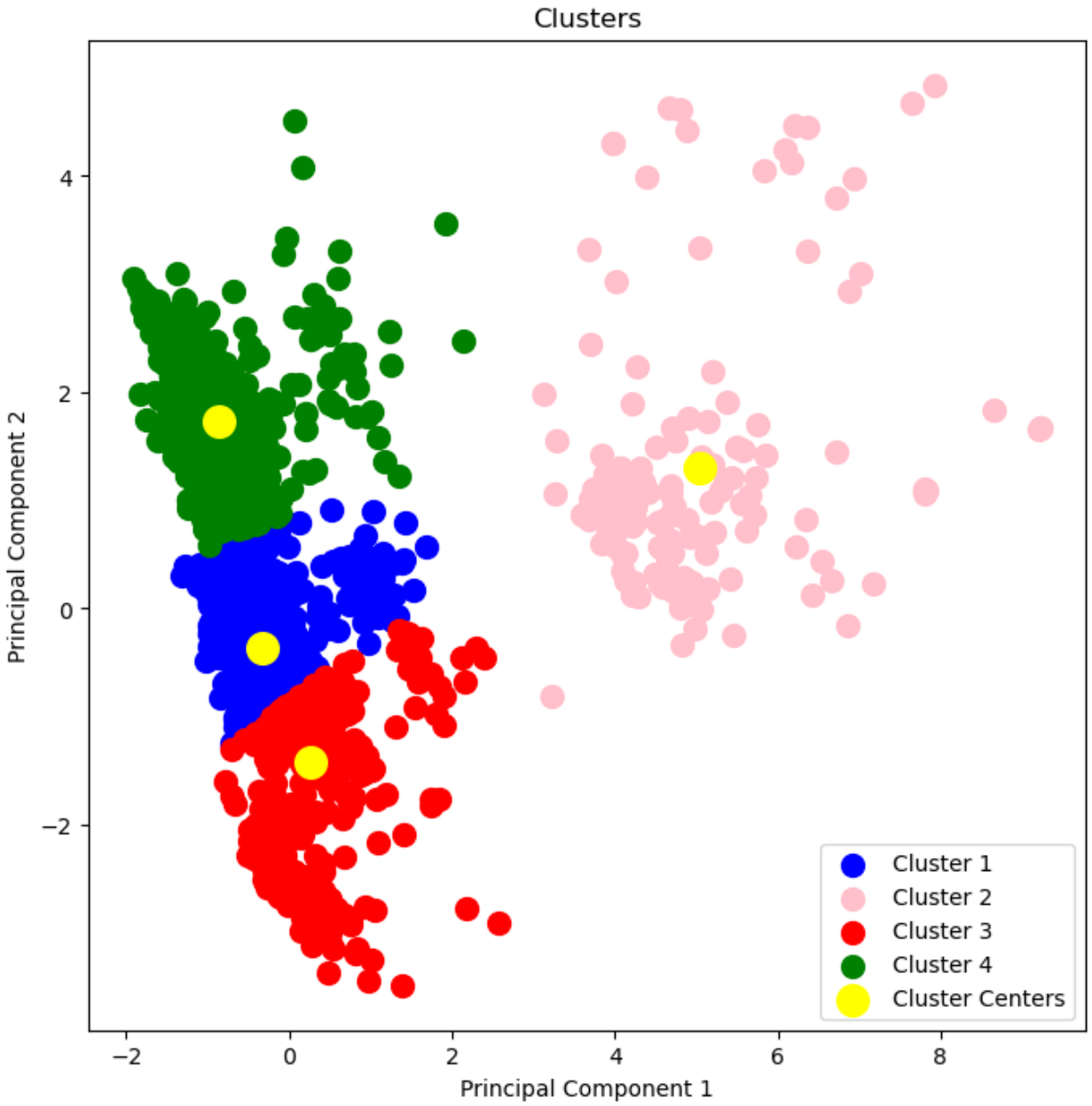



```
model = KMeans(n_clusters=4)
model.fit(xtrain)
pred = model.predict(xtest)
pred[:5]
```

```
C:\Users\user\anaconda3\Lib\site-packages\sklearn\cluster\
_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
  warnings.warn(
```

```
array([1, 3, 0, 3, 0])
```

```
plt.figure(figsize=(8,8))
plt.scatter(xtest[pred==0,0],xtest[pred==0,1],s=100, c='blue',
label='Cluster 1')
plt.scatter(xtest[pred==1,0],xtest[pred==1,1],s=100, c='pink',
label='Cluster 2')
plt.scatter(xtest[pred==2,0],xtest[pred==2,1],s=100, c='red',
label='Cluster 3')
plt.scatter(xtest[pred==3,0],xtest[pred==3,1],s=100, c='green',
label='Cluster 4')
plt.scatter (model.cluster_centers_[:,0], model.cluster_centers_[:,1],
             s=200, c='yellow', label='Cluster Centers')
plt.title('Clusters')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend()
plt.show()
```



3.Create a regression model to predict the budget.

```
import pandas as pd
import numpy as np
df=pd.read_csv('Freelance Platform Projects.csv')
df.head()
```

```
                                    Title         Category Name
Experience  \
0  Banner images for web desgin websites                Design
Entry ($)
1    Make my picture a solid silhouette   Video, Photo & Image
Entry ($)
2                      Bookkeeper needed              Business
Entry ($)
3                     Accountant needed               Business
Entry ($)
4          Guest Post on High DA Website     Digital Marketing  Expert
($$$)

           Sub Category Name Currency   Budget Location  \
0             Graphic Design      EUR     60.0   remote
1              Image Editing      GBP     20.0   remote
2       Finance & Accounting      GBP     12.0   remote
3  Tax Consulting & Advising      GBP     14.0   remote
4                        SEO      USD  10000.0   remote

  Freelancer Preferred From         Type          Date Posted  \
0                      ALL  fixed_price  2023-04-29 18:06:39
1                      ALL  fixed_price  2023-04-29 17:40:28
2                      ALL  fixed_price  2023-04-29 17:40:06
3                      ALL  fixed_price  2023-04-29 17:32:01
4                      ALL  fixed_price  2023-04-29 17:09:36

                                       Description Duration  \
0  We are looking to improve the banner images on...      NaN
1  Hello \n\nI need a quick designer to make 4 pi...      NaN
2  Hi - I need a bookkeeper to assist with bookke...      NaN
3  Hi - I need an accountant to assist me with un...      NaN
4  Hi, I am currently running a project where I w...      NaN

  Client Registration Date Client City  Client Country Client Currency
\
0               2010-11-03      Dublin         Ireland              EUR

1               2017-02-21      London  United Kingdom              GBP

2               2023-04-09      London  United Kingdom              GBP

3               2023-04-09      London  United Kingdom              GBP

4               2016-07-01      Mumbai           India              USD


    Client Job Title
0     PPC Management
1     Office manager
2          Paralegal
```

```
3          Paralegal
4  Guest posts buyer
```

```
df.isna().sum()
```

```
Title                            0
Category Name                    0
Experience                       0
Sub Category Name                0
Currency                         0
Budget                           0
Location                         0
Freelancer Preferred From        0
Type                             0
Date Posted                      0
Description                      0
Duration                     10620
Client Registration Date         0
Client City                      0
Client Country                   0
Client Currency                  0
Client Job Title              7634
dtype: int64
```

```
newdf = df.dropna(axis=0)
newdf.isnull().sum()
```

```
Title                        0
Category Name                0
Experience                   0
Sub Category Name            0
Currency                     0
Budget                       0
Location                     0
Freelancer Preferred From    0
Type                         0
Date Posted                  0
Description                  0
Duration                     0
Client Registration Date     0
Client City                  0
Client Country               0
Client Currency              0
Client Job Title             0
dtype: int64
```

```
df.shape
```

```
(12222, 17)
```

```
df.dropna(axis=0,inplace=True)
```

```
df.dtypes
```

```
Title                           object
Category Name                   object
Experience                      object
Sub Category Name               object
Currency                        object
Budget                         float64
Location                        object
Freelancer Preferred From       object
Type                            object
Date Posted                     object
Description                     object
Duration                        object
Client Registration Date        object
Client City                     object
Client Country                  object
Client Currency                 object
Client Job Title                object
dtype: object
```

```
obj_cols = df.select_dtypes('object').columns
obj_cols
```

```
Index(['Title', 'Category Name', 'Experience', 'Sub Category Name',
'Currency',
       'Location', 'Freelancer Preferred From', 'Type', 'Date Posted',
       'Description', 'Duration', 'Client Registration Date', 'Client
City',
       'Client Country', 'Client Currency', 'Client Job Title'],
      dtype='object')
```

```
from sklearn.preprocessing import LabelEncoder

for col in obj_cols:
    encoder = LabelEncoder()
    df[col] = encoder.fit_transform(df[col])

df.dtypes
```

```
Title                           int32
Category Name                   int32
Experience                      int32
Sub Category Name               int32
Currency                        int32
Budget                         float64
Location                        int32
Freelancer Preferred From       int32
Type                            int32
Date Posted                     int32
Description                     int32
```

```
Duration                        int32
Client Registration Date        int32
Client City                     int32
Client Country                  int32
Client Currency                 int32
Client Job Title                int32
dtype: object
```

```
x= df.drop(columns=['Budget'])
y= df['Budget']
```

```
from sklearn.model_selection import train_test_split
```

```
xtrain, xtest, ytrain, ytest = train_test_split(x,y,train_size=0.8,
random_state=4)
```

```
xtrain.head()
```

```
      Title  Category Name  Experience  Sub Category Name  Currency
Location  \
6320    416              0           0                  1         1
2
4512    403              8           0                 18         2
1
8728    367              6           0                 20         0
1
6333    730              6           0                 19         1
1
3506    436              0           0                  1         2
2

      Freelancer Preferred From  Type  Date Posted  Description
Duration  \
6320                         10     0          329          513
3
4512                          1     0          459          679
3
8728                          1     0          185          384
17
6333                          1     0          327          386
17
3506                         26     0          523          478
3

      Client Registration Date  Client City  Client Country  Client
Currency  \
6320                       169          254              12
2
4512                       413            4              23
2
```

```
8728                         450          217                26
0
6333                         277           82                63
1
3506                         332           53                63
0
```

```
      Client Job Title
6320               225
4512               318
8728                15
6333               357
3506                 2
```

```
ytrain.head()
```

```
6320    10.0
4512    20.0
8728    50.0
6333    50.0
3506    10.0
Name: Budget, dtype: float64
```

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()

model.fit(xtrain, ytrain)
```

```
LinearRegression()
```

```
print (model.coef_)
print(xtrain.columns)
```

```
[-6.76295586e-01  1.24159133e+01  4.60988567e+02  4.17627892e+00
 -2.41564001e+02 -1.59222837e+03  1.75174244e+02 -6.31477258e+02
  6.04170932e-01  1.29511561e-01  2.37530021e+01  2.56436550e+00
  7.70972366e-01  8.57579526e+00  3.98371047e+02  1.12652134e+00]
Index(['Title', 'Category Name', 'Experience', 'Sub Category Name',
'Currency',
       'Location', 'Freelancer Preferred From', 'Type', 'Date Posted',
       'Description', 'Duration', 'Client Registration Date', 'Client
City',
       'Client Country', 'Client Currency', 'Client Job Title'],
      dtype='object')
```

```
model.intercept_
```

```
-611.4765914652282
```

```
trainpred = model.predict(xtrain)
testpred = model.predict(xtest)
```

```
trainpred[:5]
```

```
array([-444.12322356,  241.19230932,    7.72892975,  155.77391831,
       1868.68635472])
```

```
ytrain.head()
```

```
6320    10.0
4512    20.0
8728    50.0
6333    50.0
3506    10.0
Name: Budget, dtype: float64
```

```
testpred[:5]
```

```
array([-342.44646096,  981.17184967, 1575.64046333, -304.4965187 ,
        -27.10330943])
```

```
ytest.head()
```

```
1435     25.0
2601     80.0
4314    150.0
5146    100.0
758      50.0
Name: Budget, dtype: float64
```

```
from sklearn.metrics import mean_squared_error

mse_train = mean_squared_error(ytrain, trainpred)
mse_train
```

```
11459532.142313251
```

```
mse_test = mean_squared_error(ytest, testpred)
mse_test
```

```
7302475.806764794
```

```
from sklearn.metrics import mean_absolute_error

mae_train = mean_absolute_error(ytrain, trainpred)
mae_train
```

```
912.2550712067672
```

```
mae_test = mean_absolute_error(ytest, testpred)
mae_test
```

```
917.112939991076
```

```
from sklearn.metrics import r2_score
r2= r2_score(ytest, testpred)
print(r2)
```

```
0.0704466635226545
```

```
import math

rmse_train = math.sqrt(mse_train)
rmse_train
```

```
3385.193073122012
```

```
rmse_test = math.sqrt(mse_test)
rmse_test
```

```
2702.3093469780238
```

4.Create a classification model to predict the value of the Type column.

```
import pandas as pd

df = pd.read_csv('Freelance Platform Projects.csv')
df.head()
```

```
                                    Title         Category Name
Experience  \
0  Banner images for web desgin websites                Design
Entry ($)
1    Make my picture a solid silhouette   Video, Photo & Image
Entry ($)
2                      Bookkeeper needed              Business
Entry ($)
3                    Accountant needed                Business
Entry ($)
4          Guest Post on High DA Website     Digital Marketing  Expert
($$$)

           Sub Category Name Currency   Budget Location  \
0             Graphic Design      EUR     60.0   remote
1              Image Editing      GBP     20.0   remote
2       Finance & Accounting      GBP     12.0   remote
3  Tax Consulting & Advising      GBP     14.0   remote
4                        SEO      USD  10000.0   remote

  Freelancer Preferred From         Type          Date Posted  \
0                       ALL  fixed_price  2023-04-29 18:06:39
```

```
1                       ALL  fixed_price  2023-04-29 17:40:28
2                       ALL  fixed_price  2023-04-29 17:40:06
3                       ALL  fixed_price  2023-04-29 17:32:01
4                       ALL  fixed_price  2023-04-29 17:09:36

                                          Description Duration  \
0  We are looking to improve the banner images on...      NaN
1  Hello \n\nI need a quick designer to make 4 pi...      NaN
2  Hi - I need a bookkeeper to assist with bookke...      NaN
3  Hi - I need an accountant to assist me with un...      NaN
4  Hi, I am currently running a project where I w...      NaN

  Client Registration Date Client City  Client Country Client Currency
\
0               2010-11-03      Dublin         Ireland              EUR

1               2017-02-21      London  United Kingdom              GBP

2               2023-04-09      London  United Kingdom              GBP

3               2023-04-09      London  United Kingdom              GBP

4               2016-07-01      Mumbai           India              USD


    Client Job Title
0     PPC Management
1     Office manager
2          Paralegal
3          Paralegal
4  Guest posts buyer

df.isna().sum()

Title                            0
Category Name                    0
Experience                       0
Sub Category Name                0
Currency                         0
Budget                           0
Location                         0
Freelancer Preferred From        0
Type                             0
Date Posted                      0
Description                      0
Duration                     10620
Client Registration Date         0
Client City                      0
Client Country                   0
Client Currency                  0
```

```
Client Job Title               7634
dtype: int64
```

```
df.dtypes
```

```
Title                          object
Category Name                  object
Experience                     object
Sub Category Name              object
Currency                       object
Budget                        float64
Location                       object
Freelancer Preferred From      object
Type                           object
Date Posted                    object
Description                    object
Duration                       object
Client Registration Date       object
Client City                    object
Client Country                 object
Client Currency                object
Client Job Title               object
dtype: object
```

```
obj_cols = df.select_dtypes('object').columns
obj_cols
```

```
Index(['Title', 'Category Name', 'Experience', 'Sub Category Name',
'Currency',
       'Location', 'Freelancer Preferred From', 'Type', 'Date Posted',
       'Description', 'Duration', 'Client Registration Date', 'Client
City',
       'Client Country', 'Client Currency', 'Client Job Title'],
      dtype='object')
```

```
from sklearn.preprocessing import LabelEncoder

for col in obj_cols:
    encoder = LabelEncoder()
    df[col] = encoder.fit_transform(df[col])
```

```
df.dtypes
```

```
Title                          int32
Category Name                  int32
Experience                     int32
Sub Category Name              int32
Currency                       int32
Budget                        float64
Location                       int32
Freelancer Preferred From      int32
```

```
Type                            int32
Date Posted                     int32
Description                     int32
Duration                        int32
Client Registration Date        int32
Client City                     int32
Client Country                  int32
Client Currency                 int32
Client Job Title                int32
dtype: object
```

```
x = df.drop(columns=['Type'])
y = df['Type']
```

```
from sklearn.model_selection import train_test_split

xtrain, xtest, ytrain, ytest = train_test_split(x,y,
                               train_size=0.8,
                               random_state=4,
                               stratify=y)
```

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(class_weight='balanced')
model.fit(xtrain, ytrain)
```

```
C:\Users\user\anaconda3\Lib\site-packages\sklearn\linear_model\
_logistic.py:458: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as
shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

```
LogisticRegression(class_weight='balanced')
```

```
trainpred = model.predict(xtrain)
trainpred[:5]
```

```
array([1, 1, 0, 0, 0])
```

```
testpred = model.predict(xtest)
testpred[:5]
```

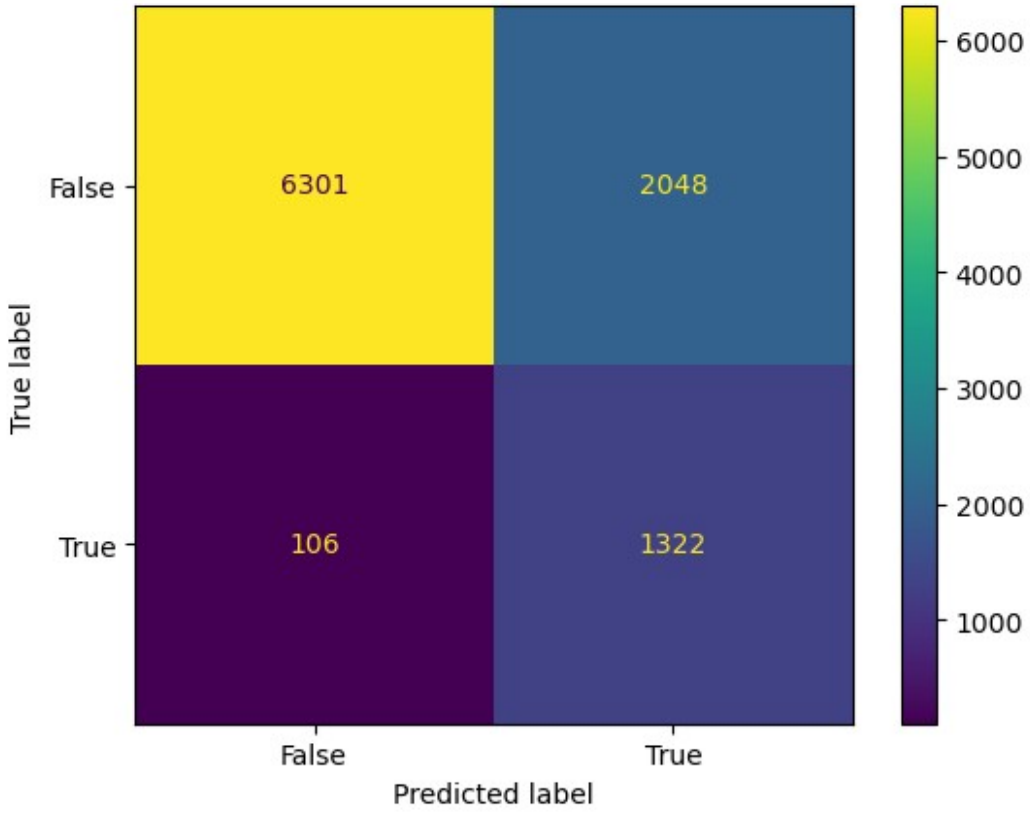
```
array([0, 0, 0, 1, 0])
```

```
from sklearn import metrics

cm = metrics.confusion_matrix(ytrain, trainpred)
cm
```

```
array([[6301, 2048],
       [ 106, 1322]], dtype=int64)
```

```
import matplotlib.pyplot as plt
cm_display = metrics.ConfusionMatrixDisplay(
                                  confusion_matrix=cm,
                                display_labels=[False, True])
cm_display.plot()
plt.show()
```




```
metrics.recall_score(ytrain, trainpred)
```

```
0.9257703081232493
```

```
metrics.precision_score(ytrain, trainpred)
```

```
0.39228486646884275
```

```
metrics.accuracy_score(ytrain, trainpred)
```

```
0.7796870205584535
```

```
print(metrics.classification_report(ytrain, trainpred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.75      0.85      8349
           1       0.39      0.93      0.55      1428

    accuracy                           0.78      9777
   macro avg       0.69      0.84      0.70      9777
weighted avg       0.90      0.78      0.81      9777
```

```
trainprob = model.predict_proba(xtrain)
trainprob
```

```
array([[1.35709508e-01, 8.64290492e-01],
       [3.96020180e-01, 6.03979820e-01],
       [8.99531809e-01, 1.00468191e-01],
       ...,
       [9.99999989e-01, 1.07468225e-08],
       [6.99030666e-01, 3.00969334e-01],
       [8.81502059e-01, 1.18497941e-01]])
```

```
trainpred = [0 if i<0.5 else 1 for i in trainprob[:,0]]
print(metrics.classification_report(ytrain, trainpred))
```

```
              precision    recall  f1-score   support

           0       0.61      0.25      0.35      8349
           1       0.02      0.07      0.03      1428

    accuracy                           0.22      9777
   macro avg       0.31      0.16      0.19      9777
weighted avg       0.52      0.22      0.30      9777
```