```
#Assignment 1:Download a Kaggle Dataset,Perform data Loading,Cleaning and Basic Transformations Using Pandas
```

```
#Data Loading

import pandas as pd

# Load training data
df = pd.read_csv("clv_data.csv")

# Preview the dataset
print(df.head())
```

```
   Unnamed: 0  id   age  gender  income  days_on_platform           city  \
0           0   0   NaN    Male  126895              14.0  San Francisco
1           1   1   NaN    Male  161474              14.0          Tokyo
2           2   2  24.0    Male  104723              34.0         London
3           3   3  29.0    Male   43791              28.0         London
4           4   4  18.0  Female  132181              26.0         London

   purchases
0          0
1          0
2          1
3          2
4          2
```

```
 #Check for Missing Values
print("\nMissing Values Count:")
print(df.isnull().sum())
```

```
Missing Values Count:
Unnamed: 0             0
id                     0
age                 2446
gender                 0
income                 0
days_on_platform     141
city                   0
purchases              0
dtype: int64
```

```
#Handle Missing Values
# Fill numeric columns with mean
numeric_cols = df.select_dtypes(include=['int64', 'float64']).columns
for col in numeric_cols:
    df[col] = df[col].fillna(df[col].mean())
```

```
 #Check for Missing Values
print("\nMissing Values Count:")
print(df.isnull().sum())
```

```
Missing Values Count:
Unnamed: 0          0
id                  0
age                 0
gender              0
income              0
days_on_platform    0
city                0
purchases           0
dtype: int64
```

```
print(df.head())
```

```
   Unnamed: 0  id        age  gender  income  days_on_platform           city  \
0           0   0  30.202036    Male  126895              14.0  San Francisco
1           1   1  30.202036    Male  161474              14.0          Tokyo
2           2   2  24.000000    Male  104723              34.0         London
3           3   3  29.000000    Male   43791              28.0         London
4           4   4  18.000000  Female  132181              26.0         London

   purchases
0          0
1          0
```

```
2          1
3          2
4          2
```

```
#This rounds the age to the nearest whole number, then converts to int.
df['age'] = df['age'].round().astype(int)
#Different Methods
#Floor the age (just take the integer part)
#(e.g., 30.9 → 30).
#Ceil the age (round up)
#(e.g 30.202036->31)
```

```
print(df.head())
```

```
   Unnamed: 0  id  age  gender  income  days_on_platform           city  \
0           0   0   30    Male  126895              14.0  San Francisco
1           1   1   30    Male  161474              14.0          Tokyo
2           2   2   24    Male  104723              34.0         London
3           3   3   29    Male   43791              28.0         London
4           4   4   18  Female  132181              26.0         London

   purchases
0          0
1          0
2          1
3          2
4          2
```

```
#To Check if any Duplicates Exsists
print(df.duplicated().any())
```

```
False
```

```
#Basic Transformation
```

```
#Filtering Rows
#Select rows based on condition.
df_young = df[df['age'] < 30]
print(df_young)
```

```
      Unnamed: 0    id   age  gender  income  days_on_platform           city  \
2              2     2  24.0    Male  104723              34.0         London
3              3     3  29.0    Male   43791              28.0         London
4              4     4  18.0  Female  132181              26.0         London
5              5     5  23.0    Male   12315              14.0  New York City
12            12    12  12.0    Male  130521              12.0         London
...          ...   ...   ...     ...     ...               ...            ...
4967        4967  4967  25.0    Male   73732              56.0         London
4976        4976  4976  29.0  Female    6881              25.0  New York City
4984        4984  4984  24.0  Female  225155               8.0  San Francisco
4986        4986  4986  23.0    Male   75425               6.0         London
4992        4992  4992  26.0    Male   88858              14.0          Miami

      purchases
2             1
3             2
4             2
5             0
12            1
...         ...
4967          0
4976          1
4984          2
4986          1
4992          3

[1247 rows x 8 columns]
```

```
#Creating New Columns
#Derive new features from existing ones.
df['income_per_day'] = df['income'] / df['days_on_platform']
print(df.head())
```

```
   Unnamed: 0  id   age  gender  income  days_on_platform           city  \
0           0   0   NaN    Male  126895              14.0  San Francisco
1           1   1   NaN    Male  161474              14.0          Tokyo
2           2   2  24.0    Male  104723              34.0         London
3           3   3  29.0    Male   43791              28.0         London
4           4   4  18.0  Female  132181              26.0         London
```

```
   purchases  income_per_day
0          0     9063.928571
1          0    11533.857143
2          1     3080.088235
3          2     1563.964286
4          2     5083.884615
```

```
# Renaming Columns
#For easier reference.
df.rename(columns={'Unnamed: 0': 'index'}, inplace=True)
print(df.head())
```

```
   index  id   age  gender  income  days_on_platform           city  \
0      0   0   NaN    Male  126895              14.0  San Francisco
1      1   1   NaN    Male  161474              14.0          Tokyo
2      2   2  24.0    Male  104723              34.0         London
3      3   3  29.0    Male   43791              28.0         London
4      4   4  18.0  Female  132181              26.0         London

   purchases  income_per_day
0          0     9063.928571
1          0    11533.857143
2          1     3080.088235
3          2     1563.964286
4          2     5083.884615
```

```
#Aggregations / GroupBy
#Summarize data.
avg_income_by_city = df.groupby('city')['income'].mean()
print(avg_income_by_city)
```

```
city
London           78967.785861
Miami            78820.004921
New York City    81137.229312
San Francisco    79456.316109
Tokyo            79576.593320
Name: income, dtype: float64
```

```
#Sorting Data
#Sort by a column.

df = df.sort_values(by='age', ascending=True)
print(df.head(20))
```

```
       index    id   age  gender  income  days_on_platform           city  \
4336   4336  4336  10.0    Male   57548              24.0  San Francisco
2758   2758  2758  10.0  Female   42407              26.0          Miami
4305   4305  4305  10.0    Male  132324              11.0          Tokyo
3773   3773  3773  10.0  Female  137218              44.0         London
18       18    18  10.0  Female     260              32.0  San Francisco
2315   2315  2315  10.0  Female   14812              13.0          Miami
2002   2002  2002  10.0    Male  152330              42.0         London
3399   3399  3399  10.0    Male   69764               3.0          Tokyo
966     966   966  10.0  Female   98873               1.0          Tokyo
4181   4181  4181  10.0    Male   38039              19.0         London
3337   3337  3337  10.0    Male  112398              20.0  New York City
470     470   470  10.0  Female    5837              39.0          Miami
1865   1865  1865  10.0  Female   66869              16.0  San Francisco
327     327   327  10.0  Female  133043               1.0          Tokyo
2888   2888  2888  10.0  Female   38859              18.0          Tokyo
3807   3807  3807  10.0  Female   81814               5.0          Miami
2366   2366  2366  10.0  Female   73755               8.0         London
829     829   829  10.0  Female   74966               8.0         London
763     763   763  10.0    Male    6712              18.0  New York City
2944   2944  2944  10.0  Female   82691              32.0          Tokyo

       purchases  income_per_day
4336           2     2397.833333
2758           0     1631.038462
4305           1    12029.454545
3773           2     3118.590909
18             0        8.125000
2315           1     1139.384615
2002           0     3626.904762
3399           1    23254.666667
966            2    98873.000000
4181           1     2002.052632
3337           0     5619.900000
470            2      149.666667
1865           1     4179.312500
```

```
327          2   133043.000000
2888         0     2158.833333
3807         1    16362.800000
2366         3     9219.375000
829          1     9370.750000
763          0      372.888889
2944         2     2584.093750
```

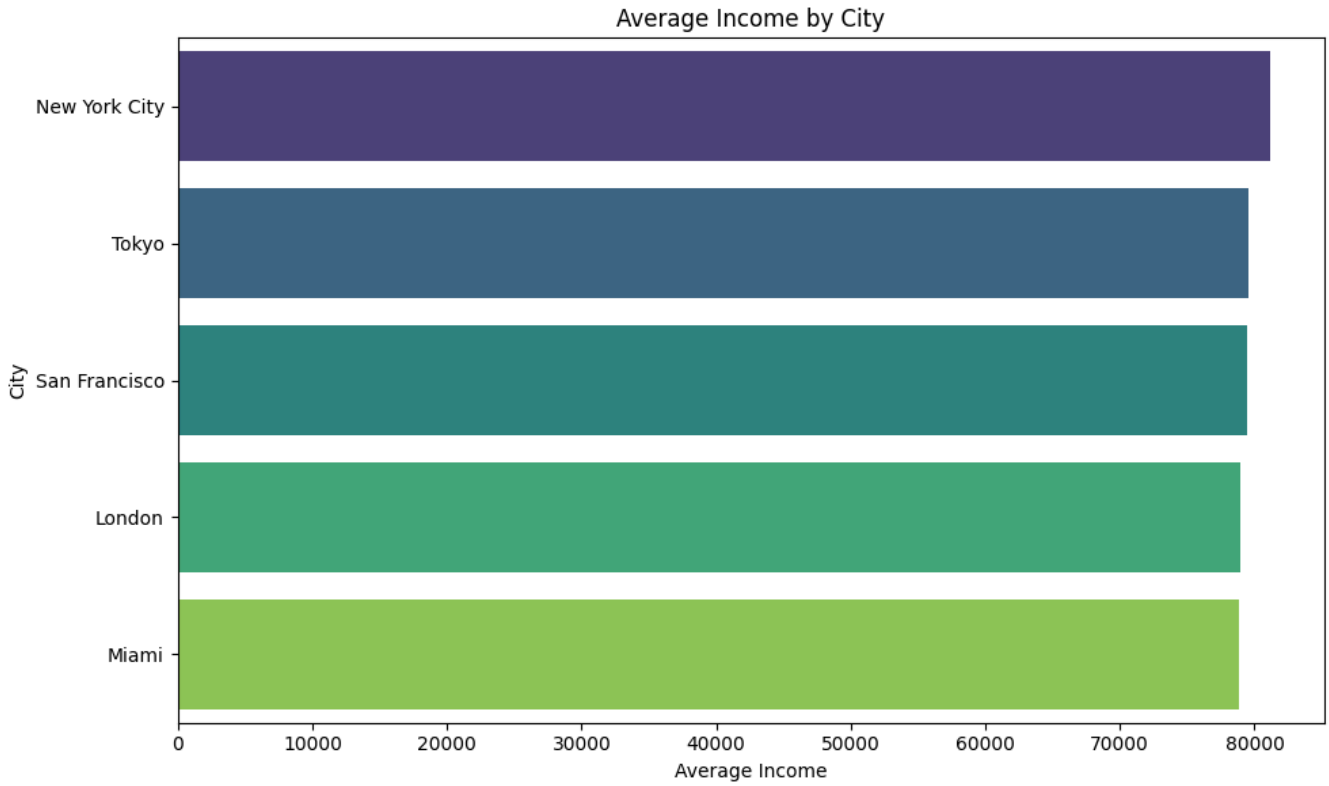
```
#Create at least three Insightful Visualizations from thr Processed data
```

```
#Average Income by City (Bar Chart)
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10, 6))
avg_income = df.groupby('city')['income'].mean().sort_values(ascending=False)
sns.barplot(x=avg_income.values, y=avg_income.index, palette='viridis')
plt.title('Average Income by City')
plt.xlabel('Average Income')
plt.ylabel('City')
plt.tight_layout()
plt.show()
```

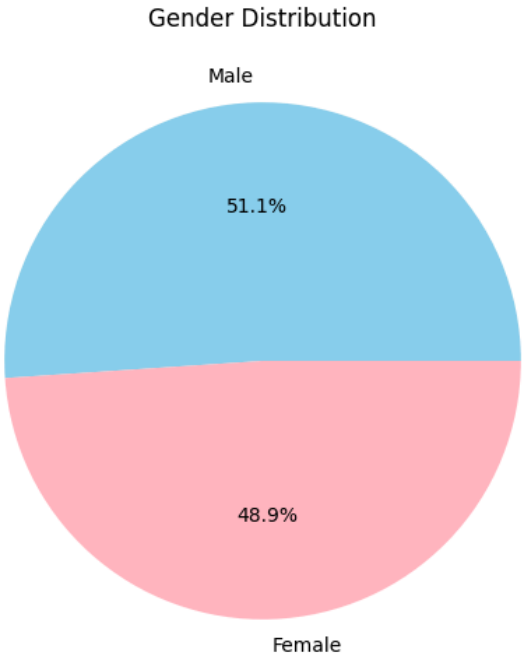
```
<ipython-input-41-e821132818a6>:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and

  sns.barplot(x=avg_income.values, y=avg_income.index, palette='viridis')
```




```
#Gender Distribution (Pie Chart)
gender_counts = df['gender'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(gender_counts, labels=gender_counts.index, autopct='%1.1f%%', colors=['skyblue', 'lightpink'])
plt.title('Gender Distribution')
plt.show()
```




```
#Relationship Between Age and Income (Scatter Plot)
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='age', y='income', hue='gender', palette='coolwarm')
plt.title('Age vs Income by Gender')
plt.xlabel('Age')
plt.ylabel('Income')
plt.legend(title='Gender')
plt.show()
```

Start coding or generate with AI.

+ Code + Text