PROJECT REPORT

-By Amit Yadav

ABSTRACT-

Social media platforms have become an undeniable force in our daily lives, fundamentally altering communication and information consumption patterns. This study delves into the average time individuals globally dedicate to social media, aiming to quantify this growing phenomenon. By analyzing recent data on social media penetration rates and user engagement, the research investigates:

- *The global average time spent on social media daily.*
- *Variations in average usage across different regions and demographics.*
- *Potential implications of these findings for social behavior, mental health, and information dissemination.*

Understanding these trends can provide valuable insights for researchers, policymakers, and social media platforms themselves, allowing them to address potential issues and optimize user experiences. The research concludes by highlighting areas for further exploration in this evolving landscape of social media usage.

INTRODUCTION-

Social media has become an omnipresent force, weaving itself into the fabric of our daily lives. Platforms like Facebook, Instagram, and Twitter have transformed the way we connect, share information, and consume content. Yet, a critical question remains: how much time, on average, do people dedicate to these platforms? This research aims to delve into this question by analyzing a dataset containing various user attributes and their social media usage patterns.

Understanding the Data:

The dataset provides a rich tapestry of information about social media users, including the following key columns:

- **age:** This captures the user's age, a demographic factor potentially influencing social media engagement.
- **gender:** Representing the user's gender identity, it can offer insights into potential usage differences between genders.
- **time_spent:** This crucial column quantifies the average daily time a user dedicates to social media platforms.
- **platform:** This identifies the specific platforms (e.g., Facebook, Twitter) that users frequent.
- **interests:** This column sheds light on users' areas of interest, potentially influencing their social media content choices.
- **location:** Geographical location can provide context regarding regional variations in social media usage patterns.
- **demographics:** This broader category might encompass additional demographic details beyond age and gender.
- **profession:** Understanding users' professions can reveal potential correlations between job type and social media behavior.
- **income:** This column sheds light on users' income levels, which may be linked to social media usage patterns.
- **indebt:** This indicates whether a user has any outstanding debt, potentially influencing their online behavior.
- **isHomeOwner:** This identifies whether a user owns their home, which might correlate with social media engagement.
- **Owns_Car:** This indicates car ownership, which, like other factors, could offer insights into social media usage trends.

By analyzing these interlinked data points, we can gain a comprehensive understanding of who uses social media, for how long, and what factors might influence their engagement patterns. Through this investigation, we aim to illuminate the evolving landscape of social media consumption and its implications for various aspects of our lives.

PROPOSED ALGORITHM-

Data Creation-

1. **Import pandas:** Imports the `pandas` library for data manipulation.
2. **File path:** Defines the path to your CSV file. Replace `"dummy_data.csv"` with the actual location of your file if it's different.

3. **Read CSV:** Uses `pd.read_csv()` to read the CSV file at the specified path and store the data in a pandas DataFrame named `df`.
4. **Print data (optional):** Uses `df.head()` to print the first few rows of the DataFrame, allowing you to verify that the data has been imported correctly.

Data Preprocessing-

1. Handling Missing Values:

- Check for missing values using `df.isnull().sum()`.
- If there are missing values, you can choose to:
    - Remove rows with missing values (if the number is small and doesn't significantly affect your analysis).
    - Impute missing values with statistical methods like mean/median for numerical data or mode for categorical data (be cautious with imputation, consider if it's appropriate for your analysis).

2. Outlier Detection:

- Identify outliers in numerical columns (like `time_spent` or `income`) using methods like boxplots or IQR (Interquartile Range).
- Decide how to handle outliers based on your research question. You might:
    - Remove them if they're genuine errors or extreme cases not representative of your target population.
    - Cap them to a specific value if they're valid data points but still skew the distribution significantly.

Data Cleaning-

1. Import libraries:

Python
```
import pandas as pd
```

2. Load data:

Python
```
# Replace 'path/to/your/file.csv' with the actual path
df = pd.read_csv('path/to/your/file.csv')
```

3. Check for missing values:

Python
```
print(df.isnull().sum())  # This shows how many missing values there are in
each column
```

- Analyze the output. Decide how to handle missing values (e.g., remove rows, impute values).

4. Identify and handle outliers:

- Use boxplots or IQR (Interquartile Range) to detect outliers in numerical columns.
- Decide how to handle outliers (e.g., remove, cap) based on your analysis goals.

5. Clean categorical data:

- Check for inconsistencies in spelling or formatting within categorical columns.
- Standardize categories (e.g., convert all lowercase to uppercase).

6. Address data type inconsistencies:

- Ensure numerical columns are numeric (e.g., convert strings representing numbers to numerical data types).

7. Explore specific data issues:

- Depending on your data, you might need to address specific issues like:
    - Inconsistent date formats.
    - Special characters or encoding problems.
    - Combining or splitting certain columns.

8. Review and iterate:

- After each cleaning step, it's helpful to review the data and potentially explore the impact of your actions on the analysis.
- You might need to revisit previous steps or perform additional cleaning tasks based on your findings.

BASIC EDA-

EDA stands for Exploratory Data Analysis. It's a crucial step in the data analysis process where you delve into your data to understand its characteristics, identify patterns, and uncover potential insights. It's all about exploring and getting a feel for your data before diving into more complex modeling or hypothesis testing.

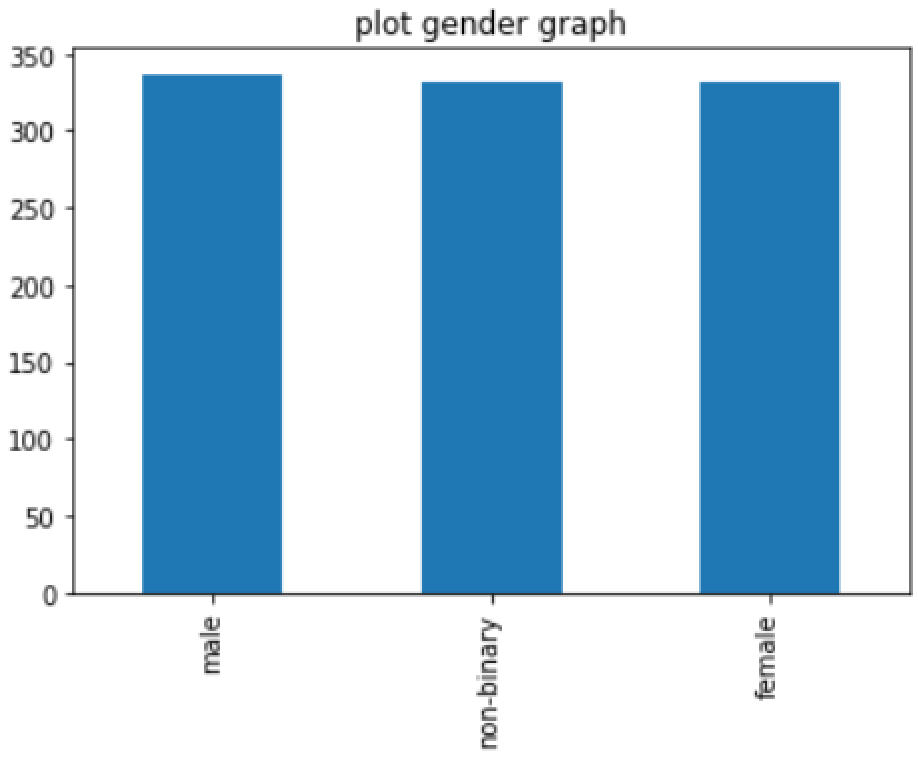
1. **Summarizing the Data:**
    - This involves calculating basic statistics like mean, median, standard deviation, and range for numerical variables.

    - For categorical variables, you might determine the frequency or percentage of each category.
2. **Visualization:**
    - Creating visualizations like histograms, boxplots, scatter plots, and bar charts helps you explore the distribution of data, identify potential outliers, and visualize relationships between variables.
3. **Identifying Patterns and Trends:**
    - By analyzing the data summaries and visualizations, you can start looking for patterns and trends within your data.
    - This might involve examining how different variables relate to each other, finding subgroups within your data, or identifying potential anomalies.
4. **Data Cleaning and Transformation:**
    - EDA often reveals issues with your data, such as missing values, outliers, or inconsistencies.
    - This step involves cleaning and transforming your data to address these issues and prepare it for further analysis.

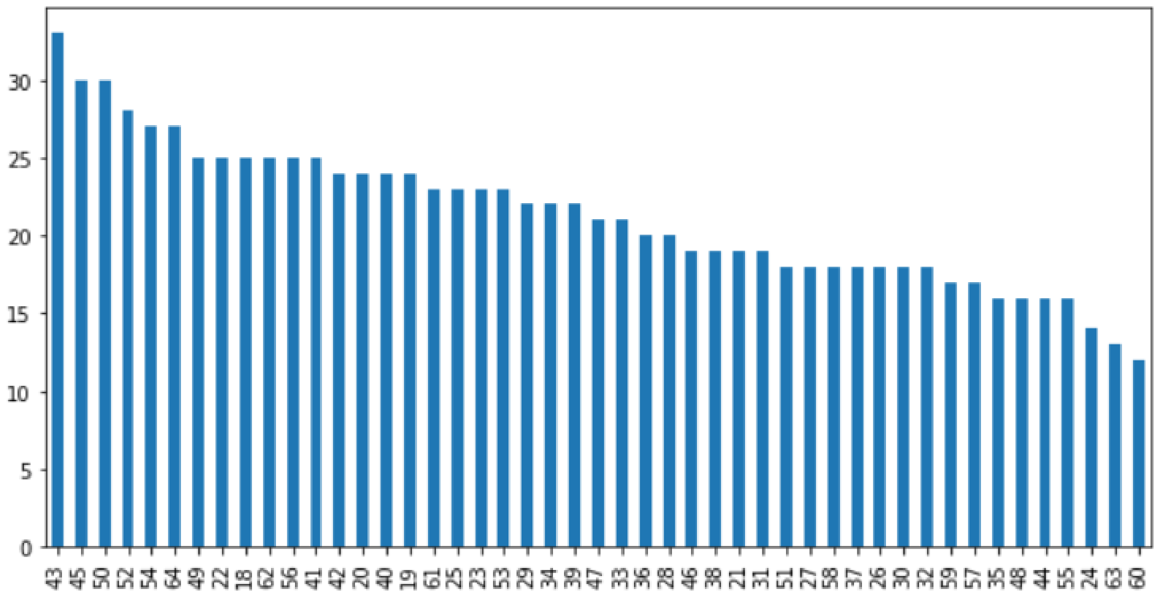
Univariate-

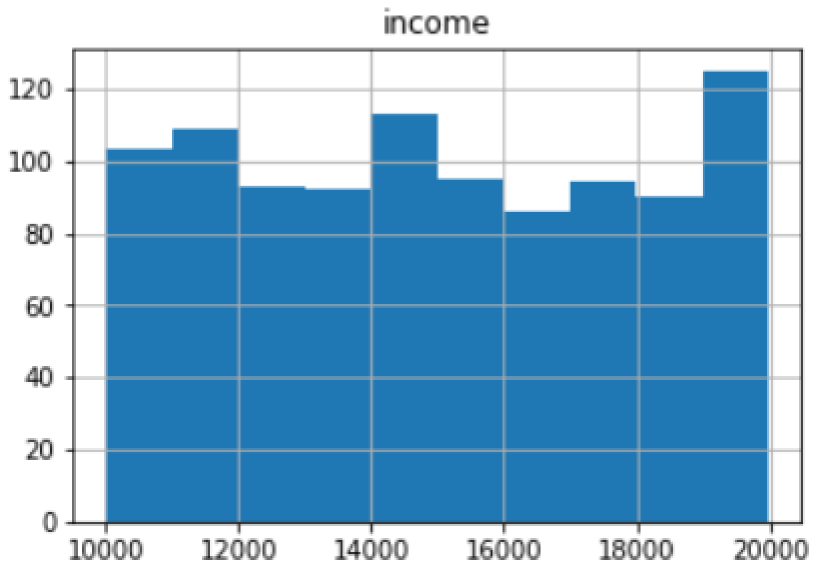
Univariate analysis is a statistical method to analyze data that involves one variable at a time. It provides a summary of the distribution of a single variable.

```
gg.plot(kind="bar")

plt.title("plot gender graph")
plt.show()
```

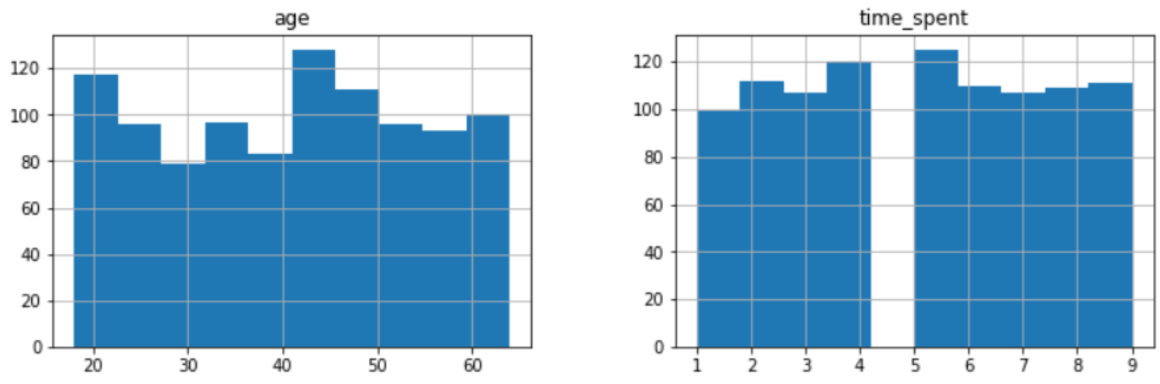


```
plt.figure(figsize=(10,5))
ageg.plot(kind="bar")
plt.show()
```



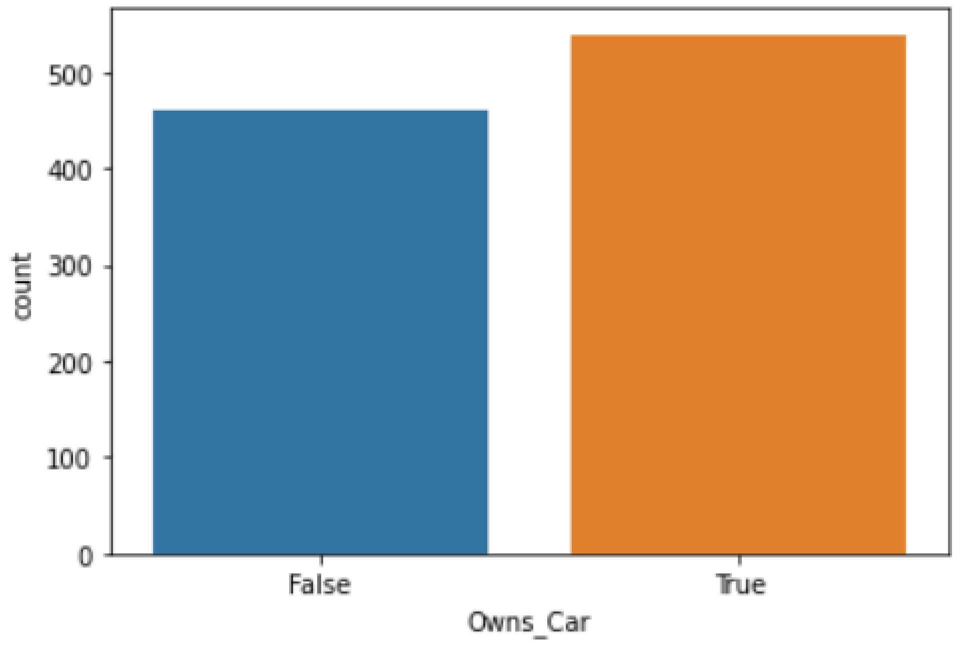


```
df.hist(figsize=(12,8))
plt.show()
```



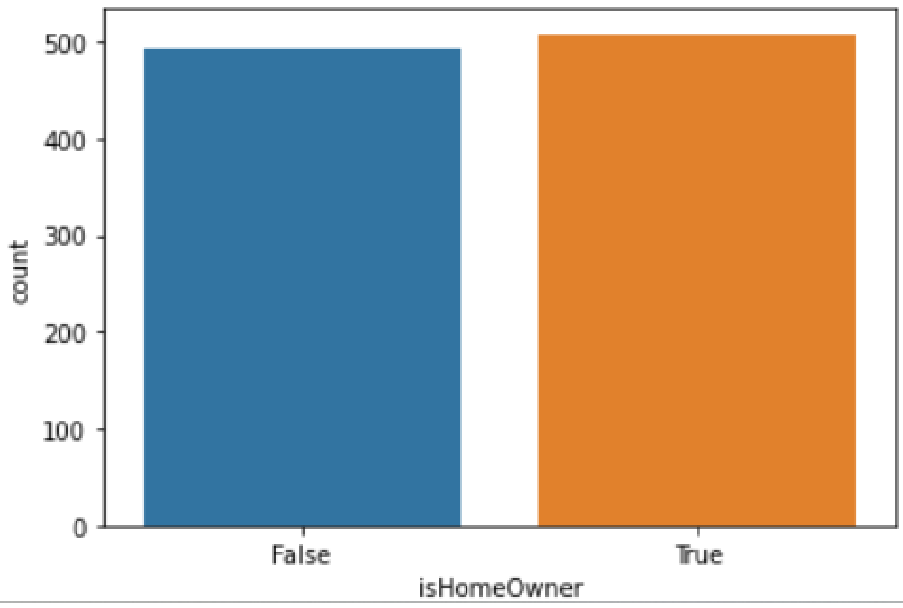
```
sns.countplot(x=df["Owns_Car"])
```

```
<AxesSubplot: xlabel='Owns_Car', ylabel='count'>
```



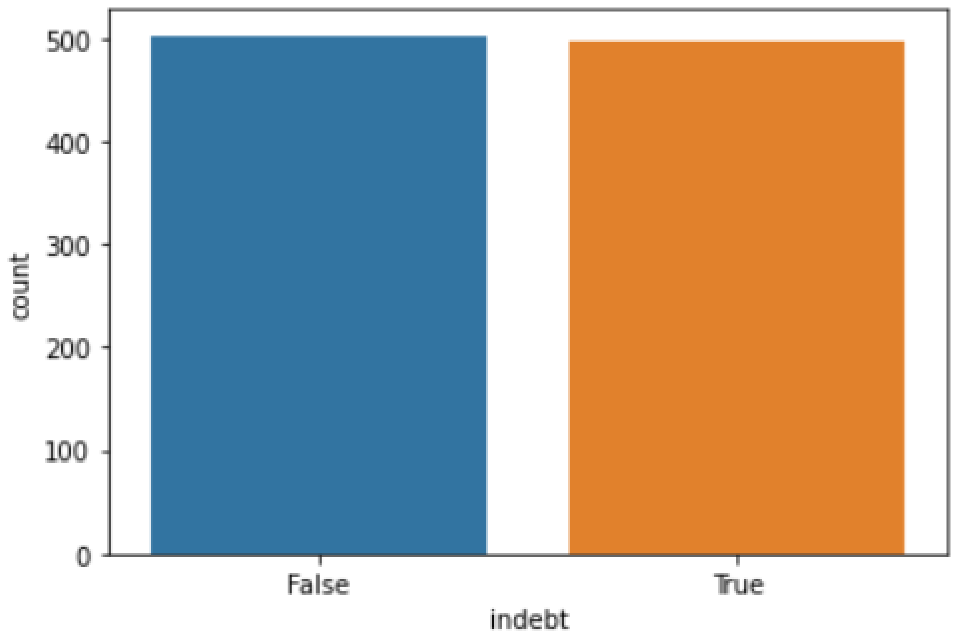
```
sns.countplot(x=df["isHomeOwner"])
```

```
<AxesSubplot: xlabel='isHomeOwner', ylabel='count'>
```



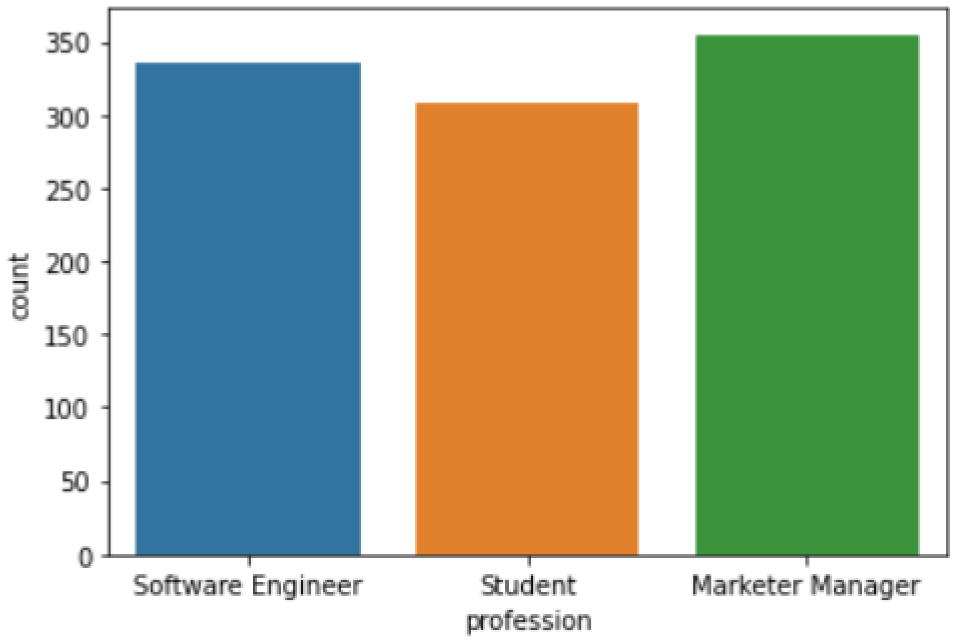
```
sns.countplot(x=df["indebt"])
```

```
<AxesSubplot: xlabel='indebt', ylabel='count'>
```



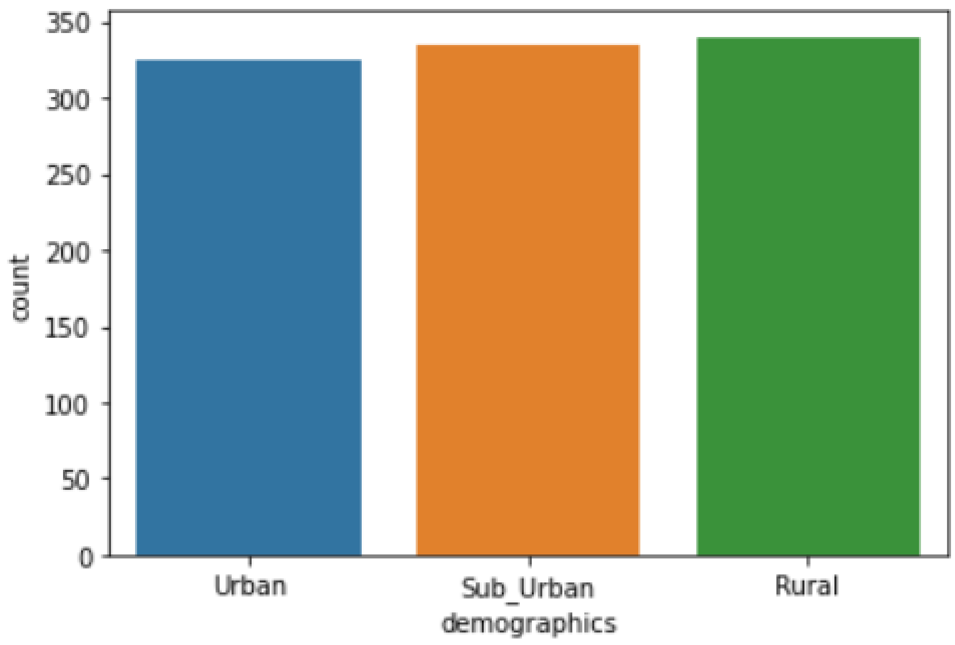
```
sns.countplot(x=df["profession"])
```

```
<AxesSubplot: xlabel='profession', ylabel='count'>
```



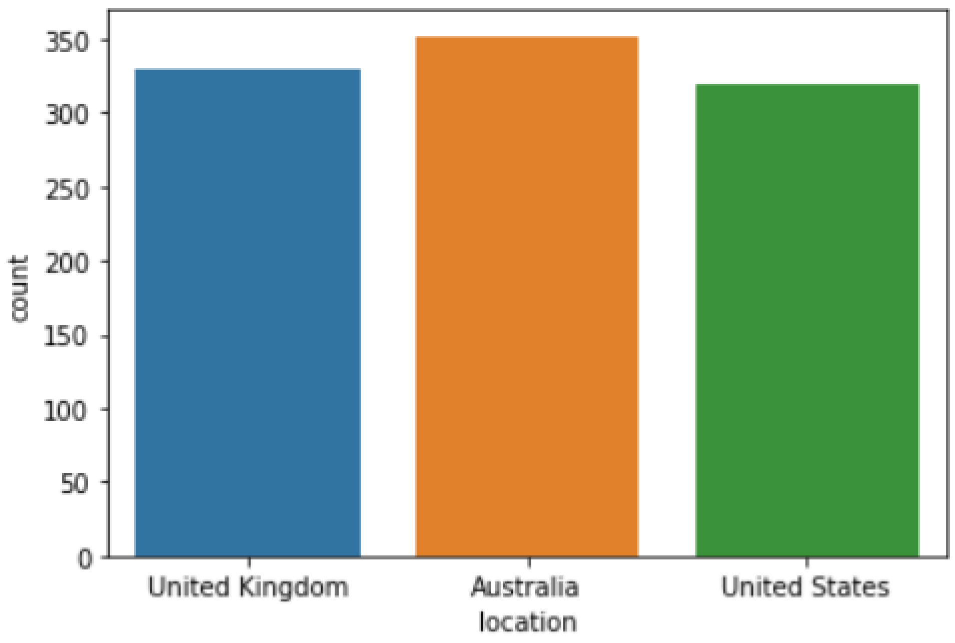
```
sns.countplot(x=df["demographics"])
```

```
<AxesSubplot: xlabel='demographics', ylabel='count'>
```



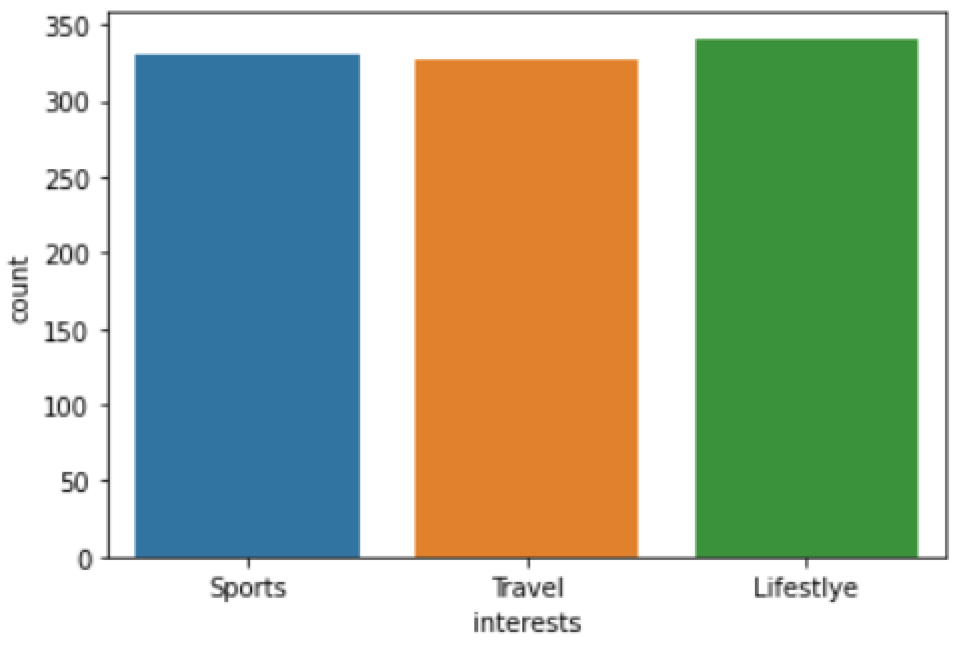
```
sns.countplot(x=df["location"])
```

```
<AxesSubplot: xlabel='location', ylabel='count'>
```



```
sns.countplot(x=df["interests"])
```

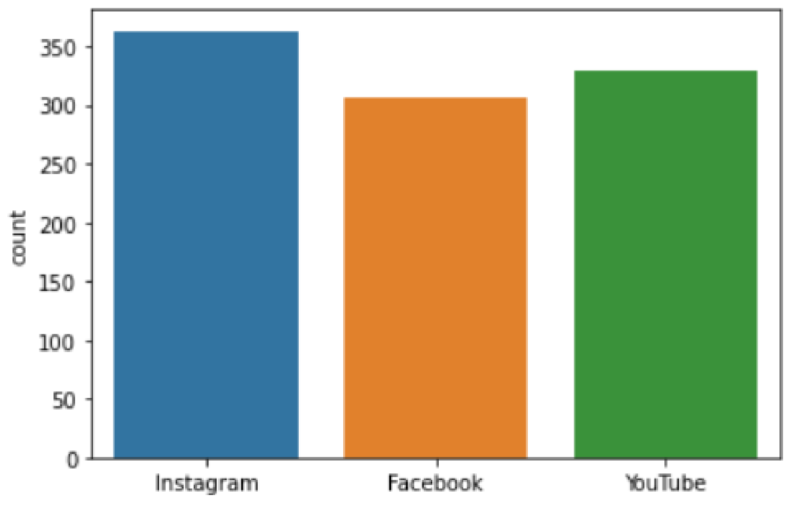
```
<AxesSubplot: xlabel='interests', ylabel='count'>
```



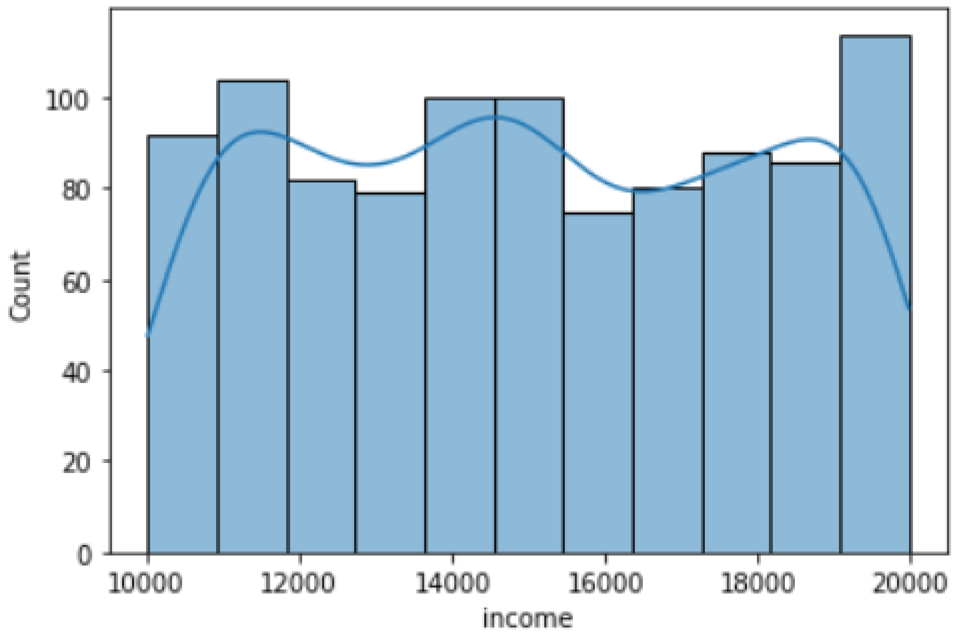
```
Index(['age', 'gender', 'time_spent', 'platform', 'interests', 'location',
       'demographics', 'profession', 'income', 'indebt', 'isHomeOwner',
       'Owns_Car'],
      dtype='object')
```

```
sns.countplot(x=df["platform"])
```

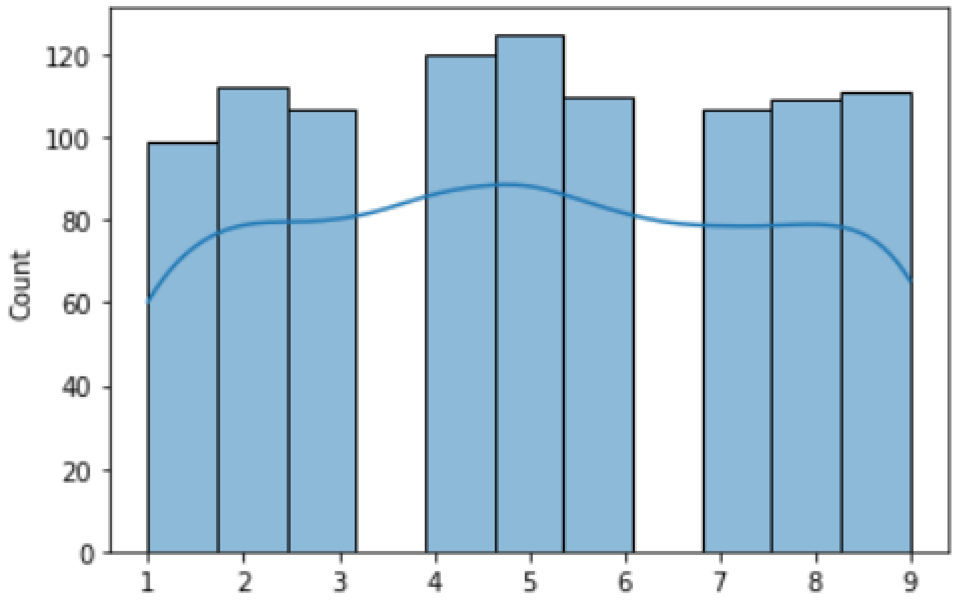
```
<AxesSubplot: xlabel='platform', ylabel='count'>
```



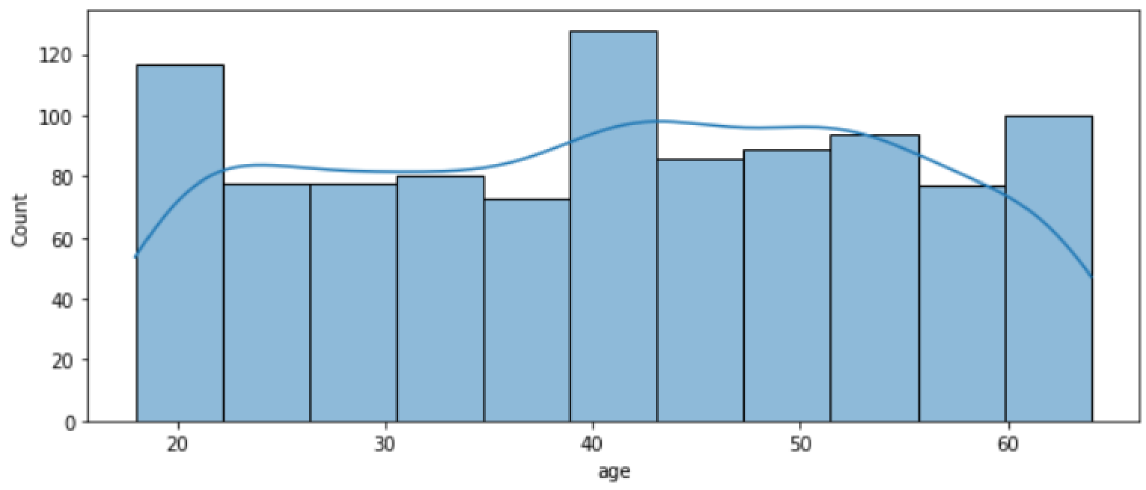
```python
sns.histplot(df["income"],kde=True)
plt.show()
```



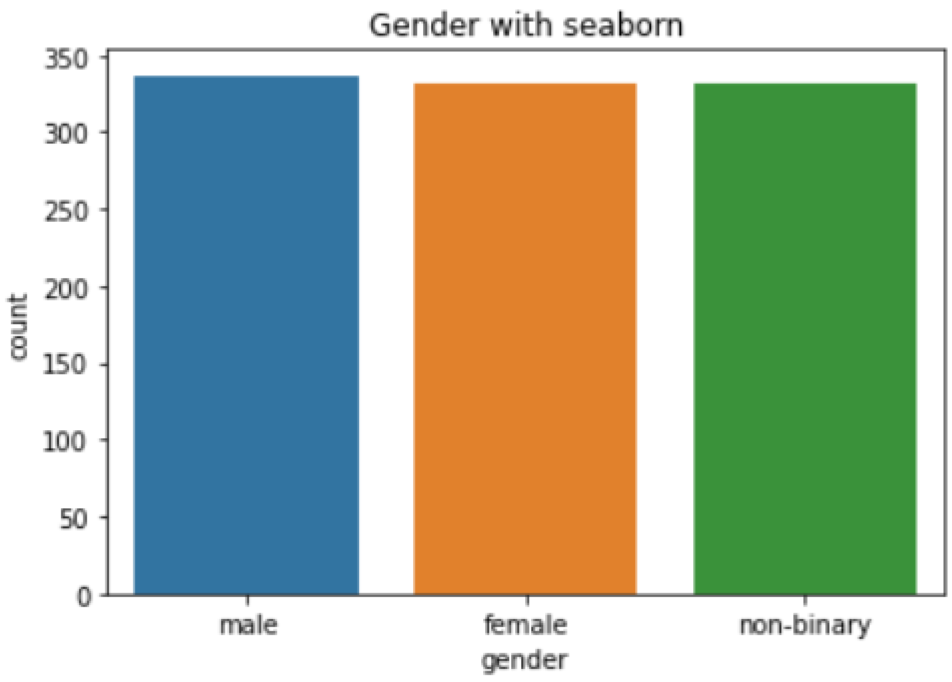
```python
sns.histplot(df["time_spent"],kde=True)
plt.show()

#time spent in hour
```



```
plt.figure(figsize=(10,4))
sns.histplot(df["age"],kde=True)
plt.show()
```



```
sns.countplot(x=df["gender"])
plt.title("Gender with seaborn")
plt.show()
```



Bivariate-

CORELATIONS

```
df
```

	age	gender	time_spent	platform	interests	location	demographics	profession	income	indebt	isHomeOwner	Owns_Car
0	56	male	3	Instagram	Sports	United Kingdom	Urban	Software Engineer	19774	True	False	False
1	46	female	2	Facebook	Travel	United Kingdom	Urban	Student	10564	True	True	True
2	32	male	8	Instagram	Sports	Australia	Sub_Urban	Marketer Manager	13258	False	False	False
3	60	non-binary	5	Instagram	Travel	United Kingdom	Urban	Student	12500	False	True	False
4	25	male	1	Instagram	Lifestlye	Australia	Urban	Software Engineer	14566	False	True	True
...	...	...	...	...	...	...	...	...	...	...	...	...
995	22	female	8	Instagram	Lifestlye	United Kingdom	Rural	Marketer Manager	18536	False	True	False
996	40	non-binary	6	YouTube	Travel	United Kingdom	Rural	Software Engineer	12711	True	False	False
997	27	non-binary	5	YouTube	Travel	United Kingdom	Rural	Student	17595	True	False	True
998	61	female	4	YouTube	Sports	Australia	Sub_Urban	Marketer Manager	16273	True	True	False
999	19	female	8	YouTube	Travel	Australia	Rural	Student	16284	False	True	False

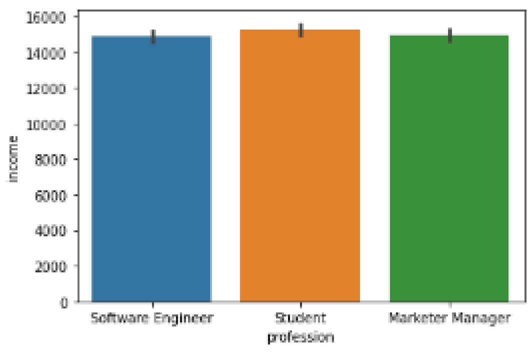
1000 rows × 12 columns

```
df2 = pd.DataFrame(df)

# Correlation matrix
correlation_matrix = df2.corr()

print(correlation_matrix)
```

```
                  age  time_spent    income    indebt  isHomeOwner  Owns_Car
age          1.000000   -0.033827 -0.087391 -0.017055    -0.005321  0.006921
time_spent  -0.033827    1.000000  0.004757  0.013079     0.029388 -0.020271
income      -0.087391    0.004757  1.000000  0.037860     0.006072  0.019789
indebt      -0.017055    0.013079  0.037860  1.000000     0.038102 -0.035641
isHomeOwner -0.005321    0.029388  0.006072  0.038102     1.000000 -0.051411
Owns_Car     0.006921   -0.020271  0.019789 -0.035641    -0.051411  1.000000
```

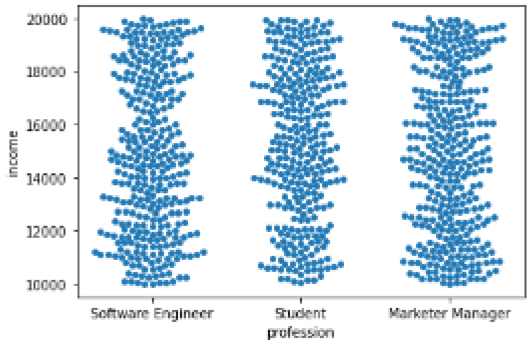
```
sns.barplot(y='income', x='profession', data=df)
```

```
<AxesSubplot: xlabel='profession', ylabel='income'>
```



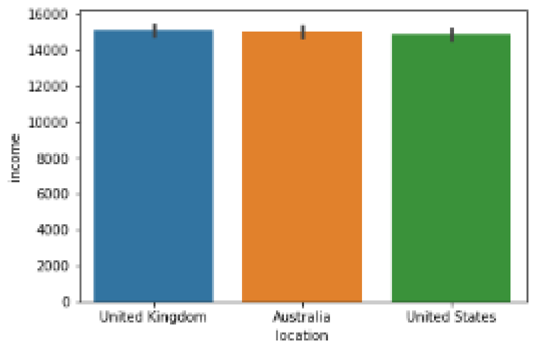
```
sns.swarmplot(y='income', x='profession', data=df)
```

```
<AxesSubplot: xlabel='profession', ylabel='income'>
```



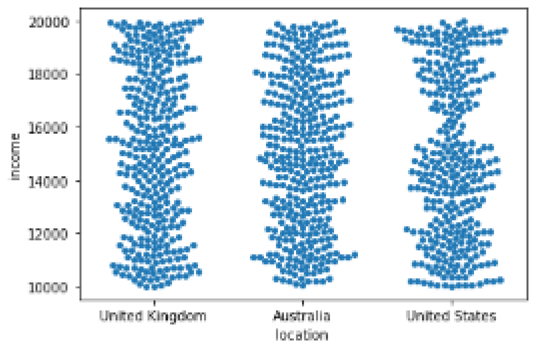
```
sns.barplot(y='income', x='location', data=df)
```

```
<AxesSubplot: xlabel='location', ylabel='income'>
```



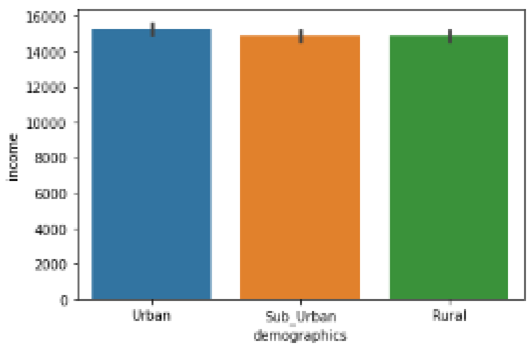
```
sns.swarmplot(y='income', x='location', data=df)
```

```
<AxesSubplot: xlabel='location', ylabel='income'>
```



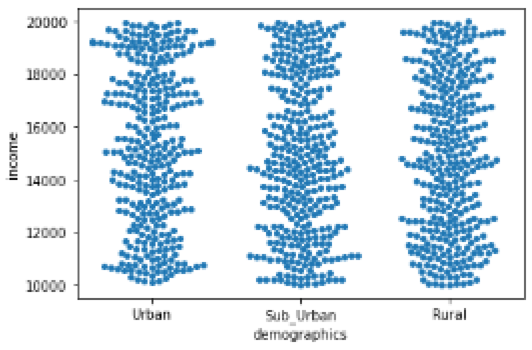
```
sns.barplot(y='income', x='demographics', data=df)
```

```
<AxesSubplot: xlabel='demographics', ylabel='income'>
```



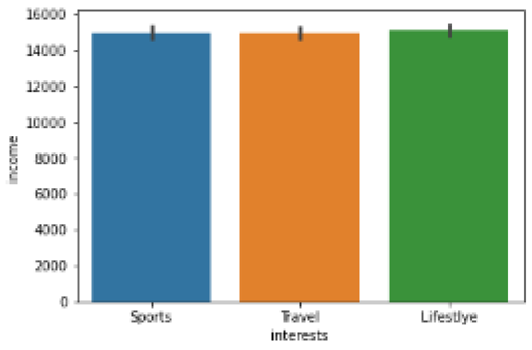
```
sns.swarmplot(y='income', x='demographics', data=df)
```

```
<AxesSubplot: xlabel='demographics', ylabel='income'>
```



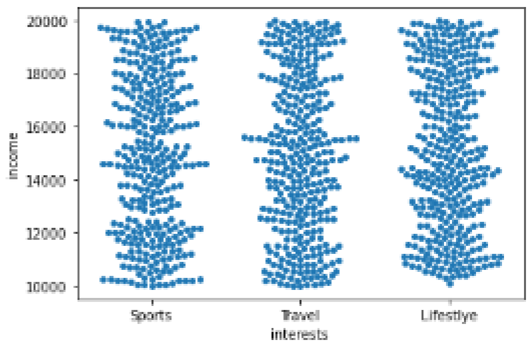
```
sns.barplot(y='income', x='interests', data=df)
```

```
<AxesSubplot: xlabel='interests', ylabel='income'>
```



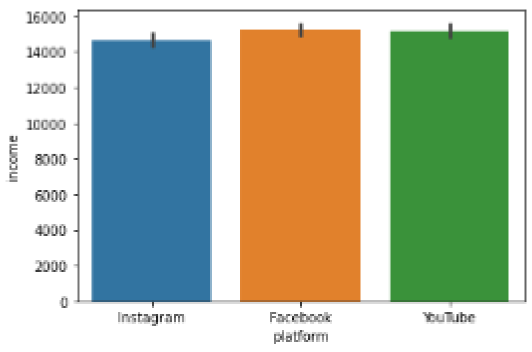
```
sns.swarmplot(y='income', x='interests', data=df)
```

```
<AxesSubplot: xlabel='interests', ylabel='income'>
```



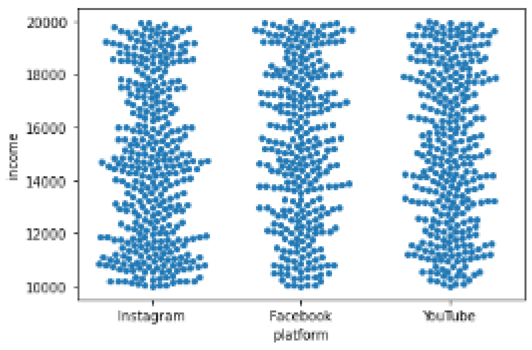
```
sns.barplot(y='income', x='platform', data=df)
```

```
<AxesSubplot: xlabel='platform', ylabel='income'>
```



```
sns.swarmplot(y='income', x='platform', data=df)
```

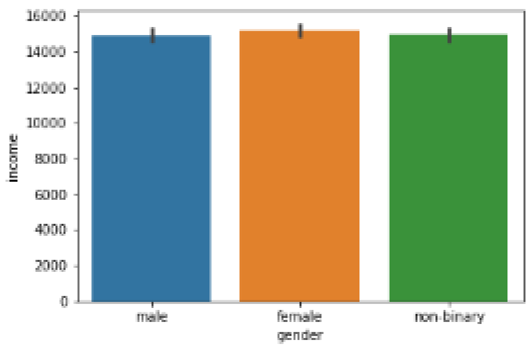
```
<AxesSubplot: xlabel='platform', ylabel='income'>
```



Income vs object columns other then boolean ones

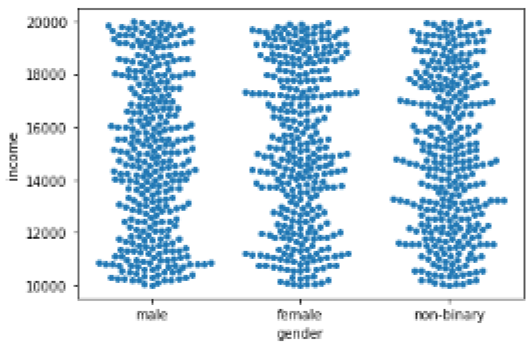
```
sns.barplot(y='income', x='gender', data=df)
```

```
<AxesSubplot: xlabel='gender', ylabel='income'>
```



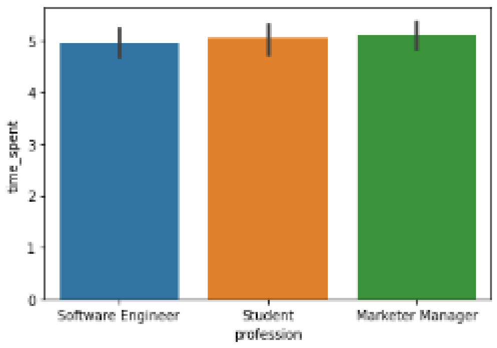
```
sns.swarmplot(y='income', x='gender', data=df)
```

```
<AxesSubplot: xlabel='gender', ylabel='income'>
```



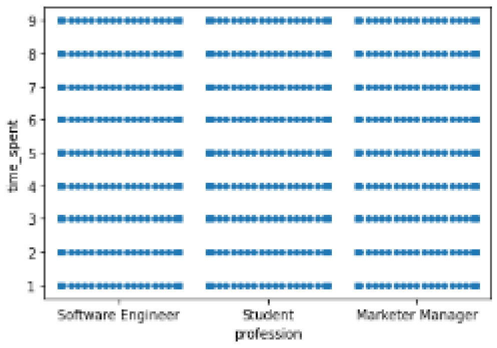
```
sns.barplot(y='time_spent', x='profession', data=df)
```

```
<AxesSubplot: xlabel='profession', ylabel='time_spent'>
```



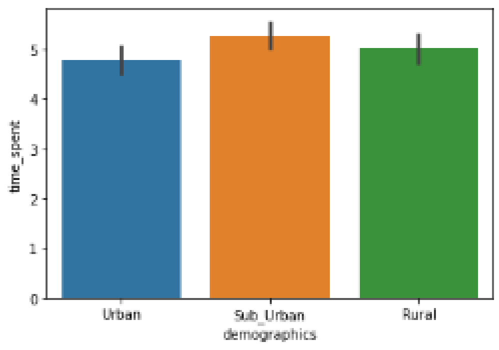
```
sns.swarmplot(y='time_spent', x='profession', data=df)
```

```
<AxesSubplot: xlabel='profession', ylabel='time_spent'>
```



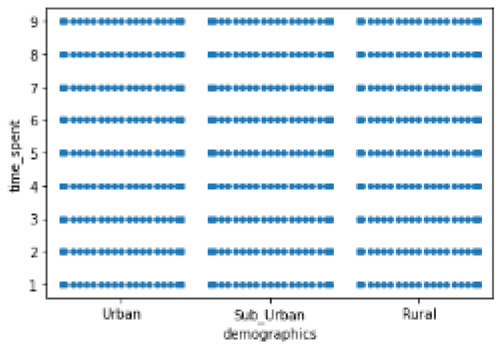
```
sns.barplot(y='time_spent', x='demographics', data=df)
```

```
<AxesSubplot: xlabel='demographics', ylabel='time_spent'>
```



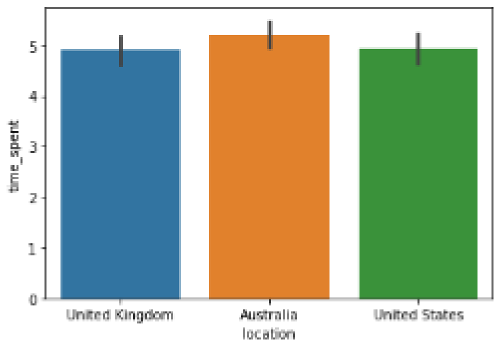
```
sns.swarmplot(y='time_spent', x='demographics', data=df)
```

```
<AxesSubplot: xlabel='demographics', ylabel='time_spent'>
```



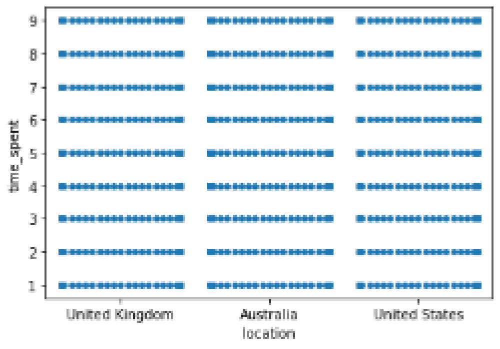
```
sns.barplot(y='time_spent', x='location', data=df)
```

```
<AxesSubplot: xlabel='location', ylabel='time_spent'>
```



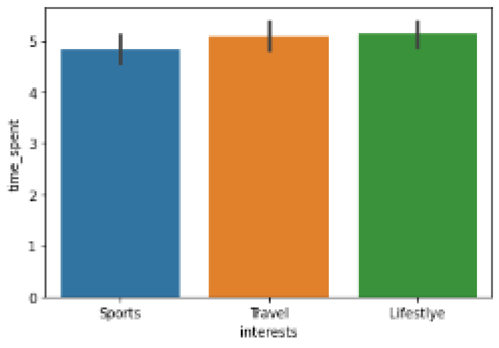
```
sns.swarmplot(y='time_spent', x='location', data=df)
```

```
<AxesSubplot: xlabel='location', ylabel='time_spent'>
```



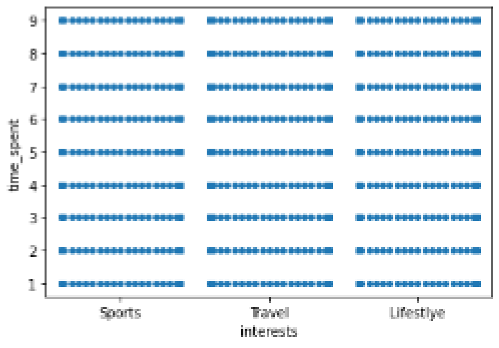
```
sns.barplot(y='time_spent', x='interests', data=df)
```

```
<AxesSubplot: xlabel='interests', ylabel='time_spent'>
```



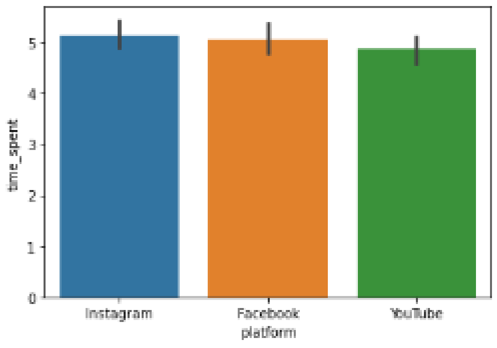
```
sns.swarmplot(y='time_spent', x='interests', data=df)
```

```
<AxesSubplot: xlabel='interests', ylabel='time_spent'>
```



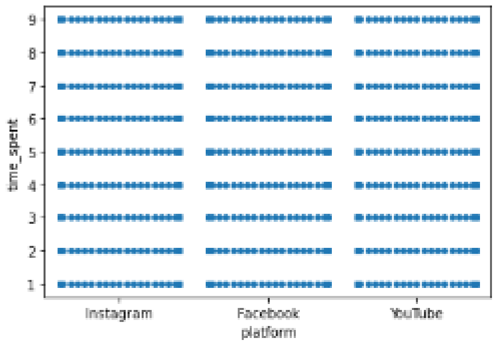
```
sns.barplot(y='time_spent', x='platform', data=df)
```

```
<AxesSubplot: xlabel='platform', ylabel='time_spent'>
```



```
sns.swarmplot(y='time_spent', x='platform', data=df)
```

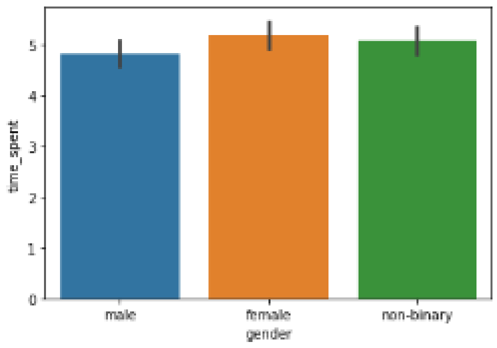
```
<AxesSubplot: xlabel='platform', ylabel='time_spent'>
```



timespent vs object columns other then boolean ones

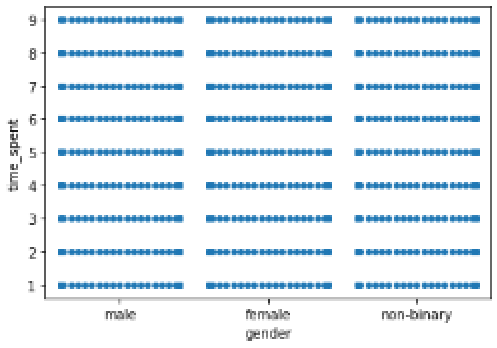
```
sns.barplot(y='time_spent', x='gender', data=df)
```

```
<AxesSubplot: xlabel='gender', ylabel='time_spent'>
```



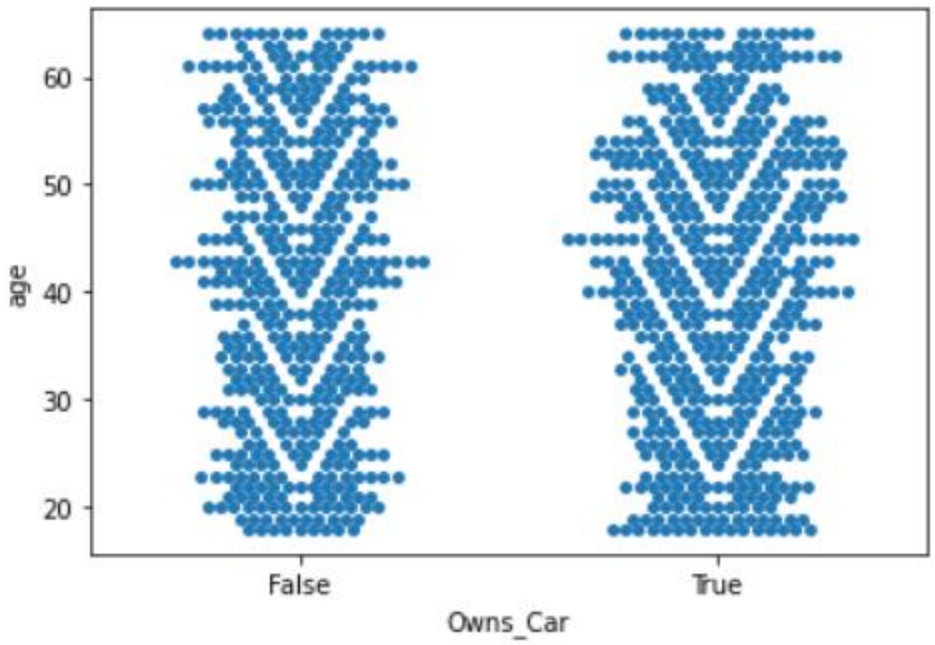
```
sns.swarmplot(y='time_spent', x='gender', data=df)
```

```
<AxesSubplot: xlabel='gender', ylabel='time_spent'>
```



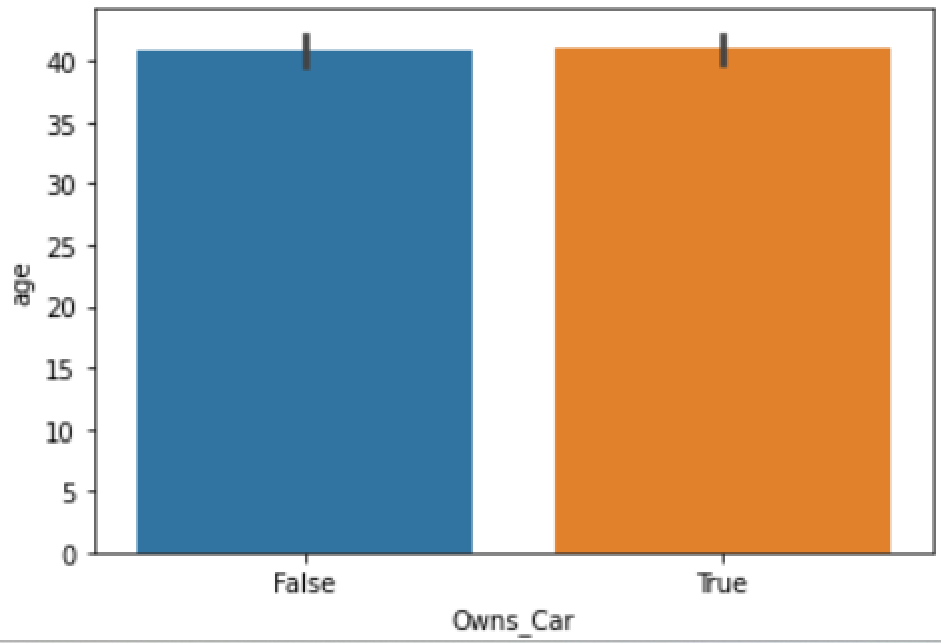
```
sns.swarmplot(y='age', x='Owns_Car', data=df)
```

```
<AxesSubplot: xlabel='Owns_Car', ylabel='age'>
```



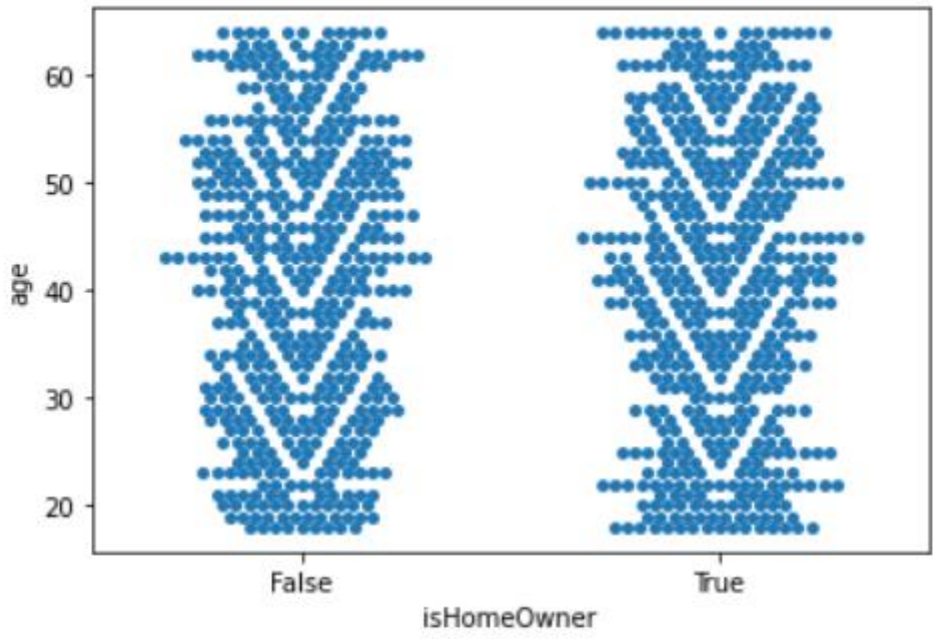
```
sns.barplot(y='age', x='Owns_Car', data=df)
```

```
<AxesSubplot: xlabel='Owns_Car', ylabel='age'>
```



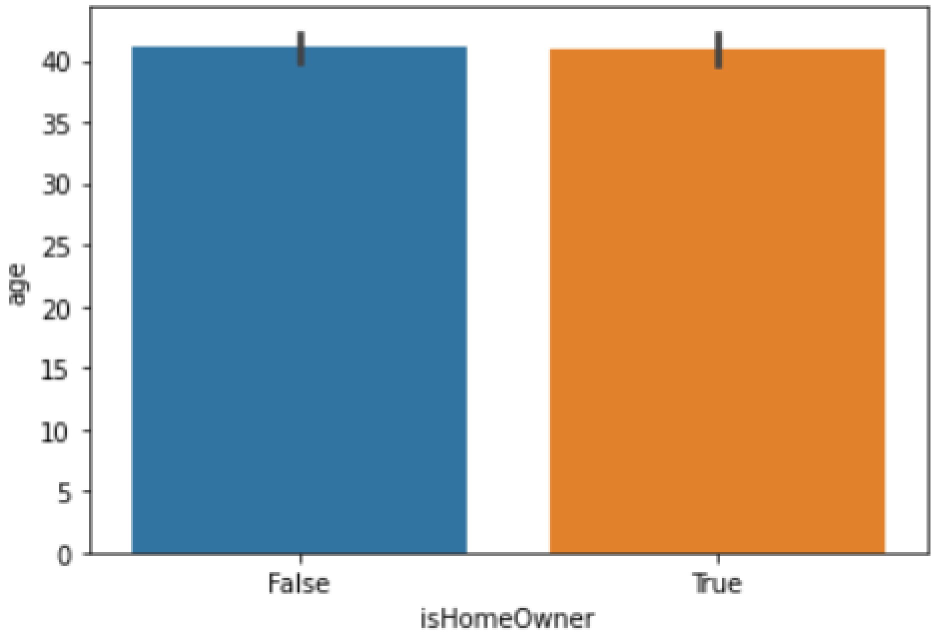
```
sns.swarmplot(y='age', x='isHomeOwner', data=df)
```

```
<AxesSubplot: xlabel='isHomeOwner', ylabel='age'>
```



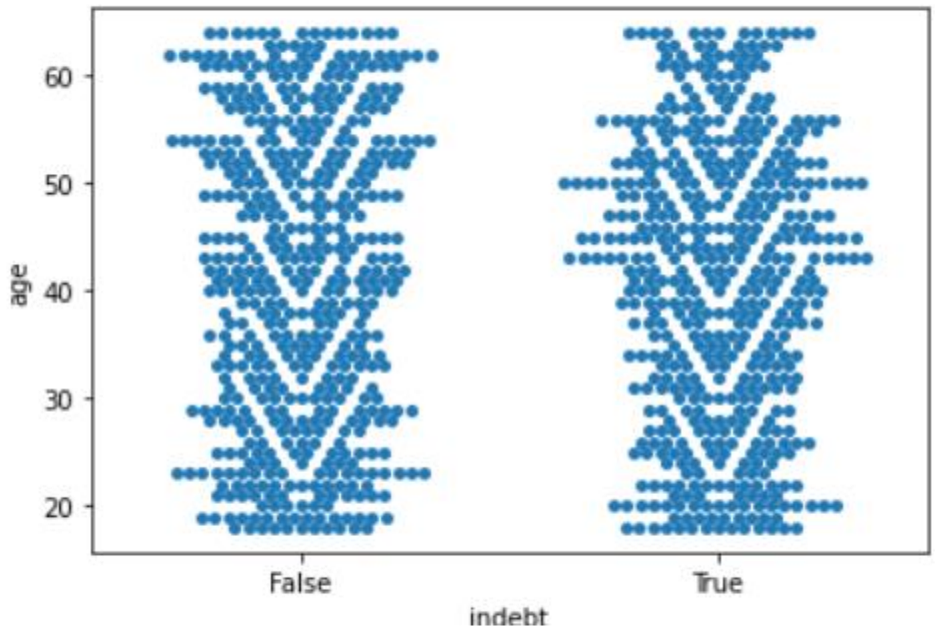
```
sns.barplot(y='age', x='isHomeOwner', data=df)
```

```
<AxesSubplot: xlabel='isHomeOwner', ylabel='age'>
```



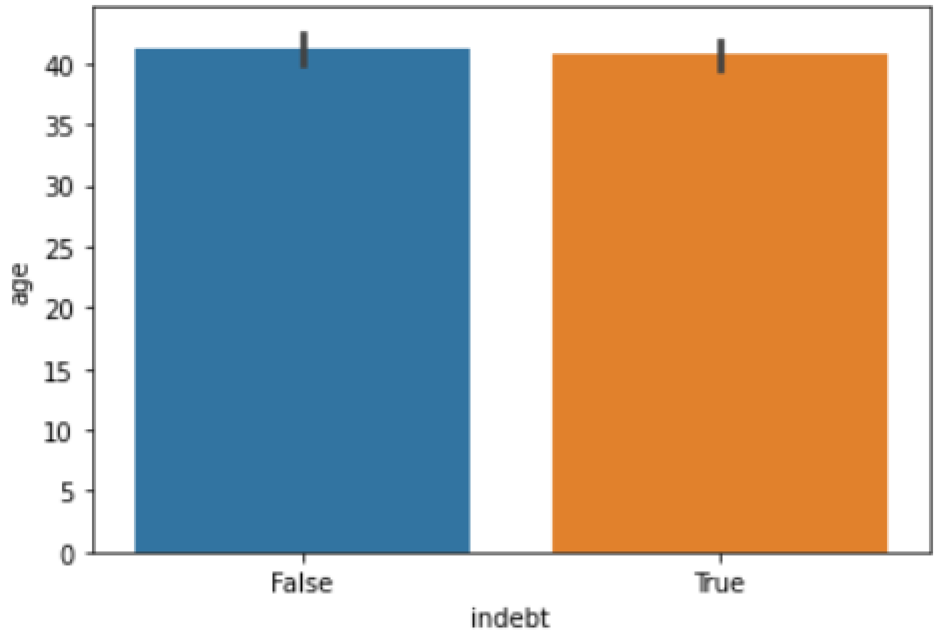
```
sns.swarmplot(y='age', x='indebt', data=df)
```

```
<AxesSubplot: xlabel='indebt', ylabel='age'>
```



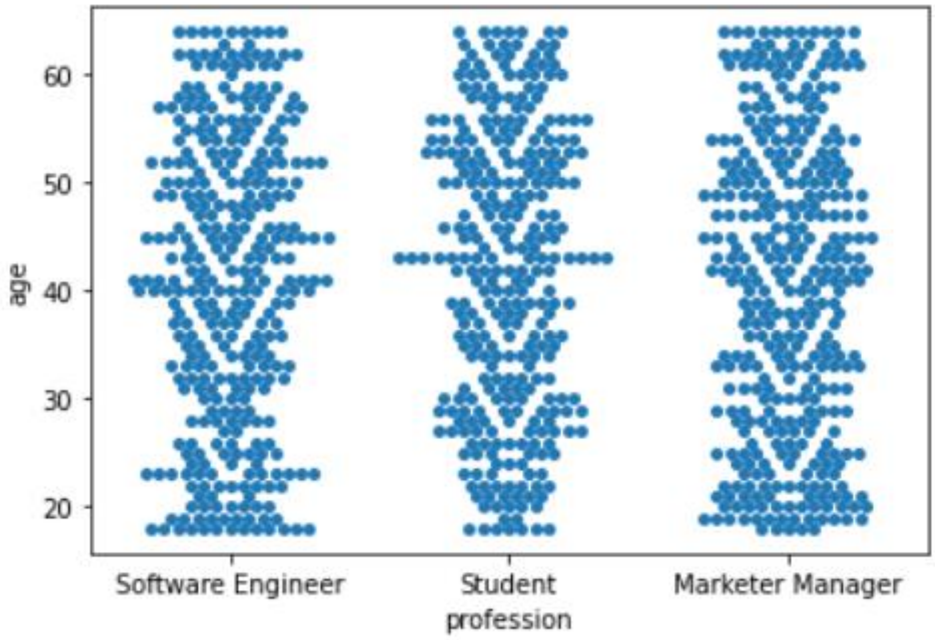
```
sns.barplot(y='age', x='indebt', data=df)
```

```
<AxesSubplot: xlabel='indebt', ylabel='age'>
```



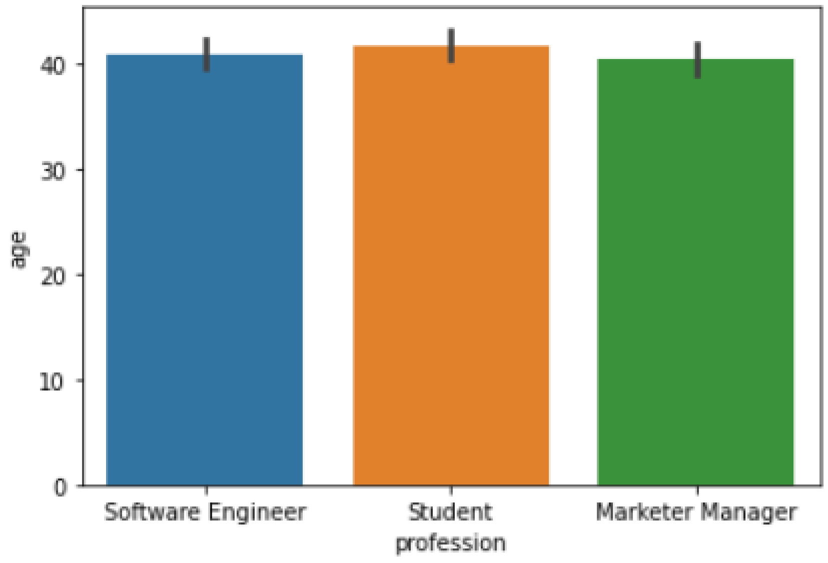
```
sns.swarmplot(y='age', x='profession', data=df)
```

```
<AxesSubplot: xlabel='profession', ylabel='age'>
```



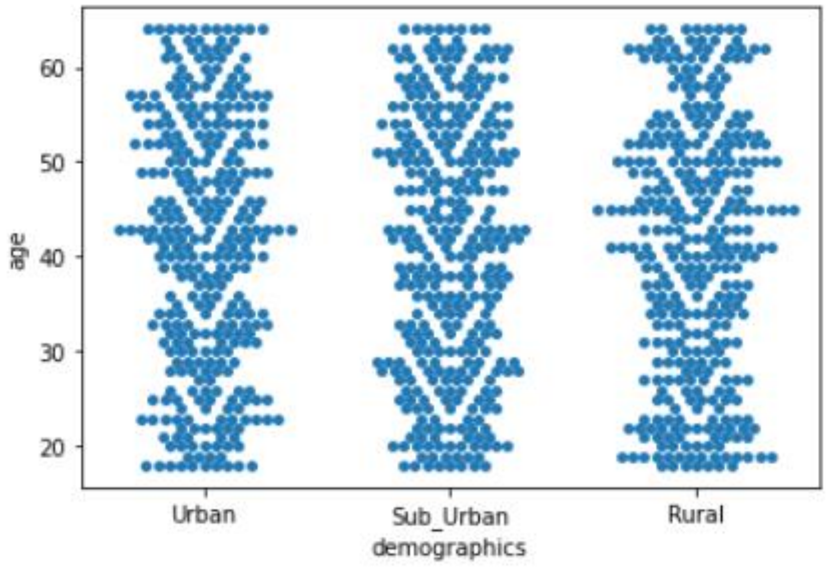
```
sns.barplot(y='age', x='profession', data=df)
```

```
<AxesSubplot: xlabel='profession', ylabel='age'>
```



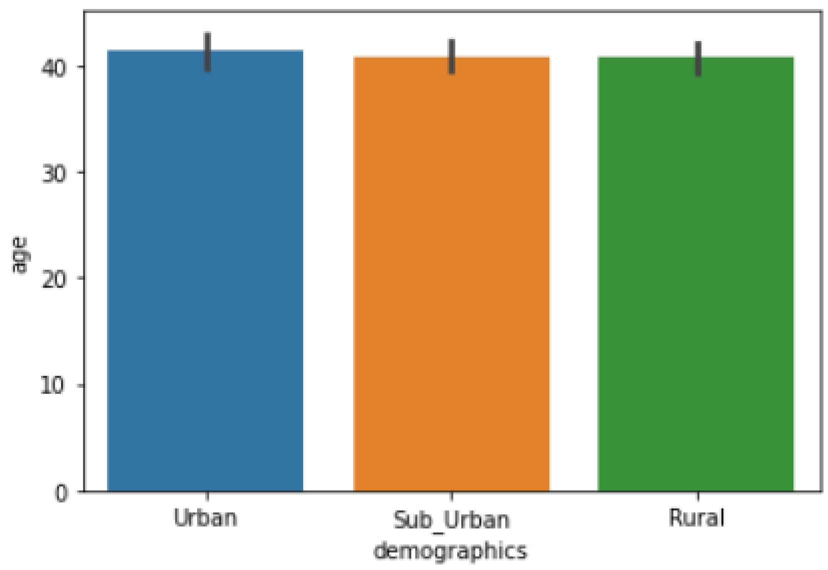
```
sns.swarmplot(y='age', x='demographics', data=df)
```

```
<AxesSubplot: xlabel='demographics', ylabel='age'>
```



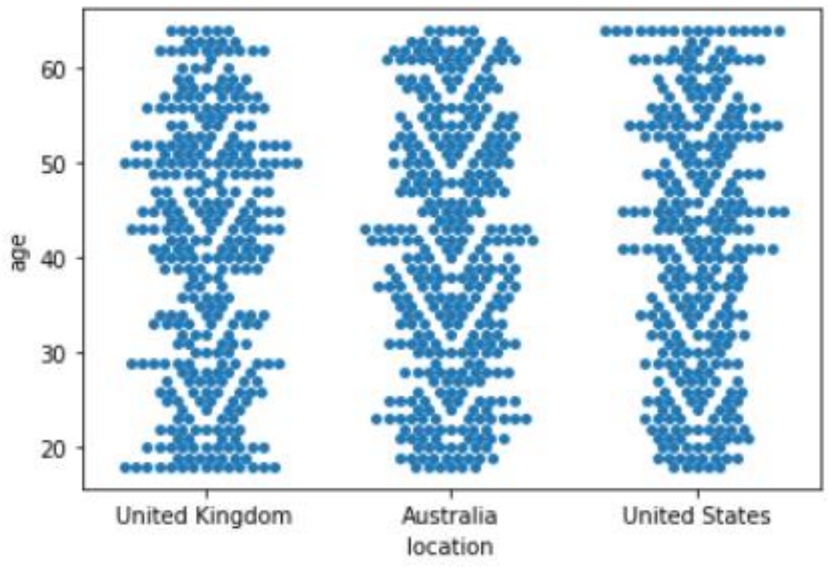
```
sns.barplot(y='age', x='demographics', data=df)
```

```
<AxesSubplot: xlabel='demographics', ylabel='age'>
```



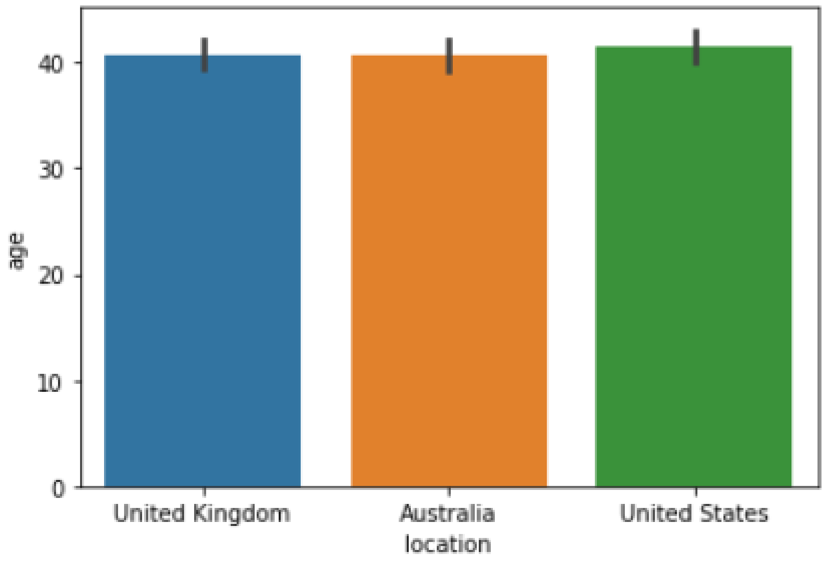
```
sns.swarmplot(y='age', x='location', data=df)
```

```
<AxesSubplot: xlabel='location', ylabel='age'>
```



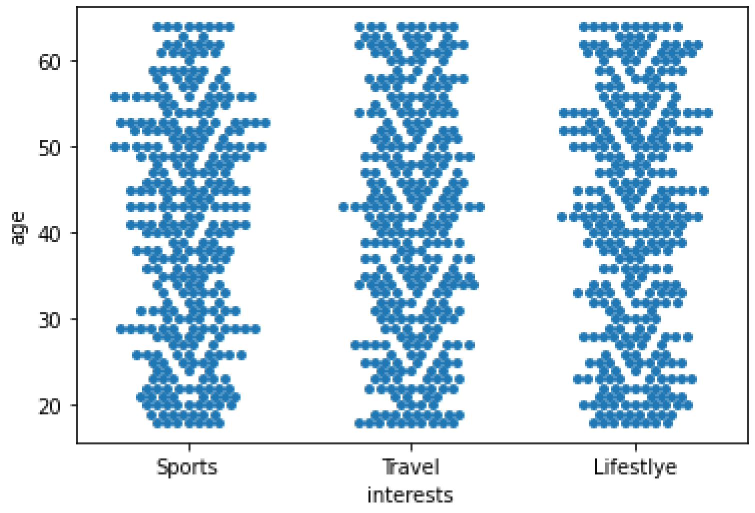
```
sns.barplot(y='age', x='location', data=df)
```

```
<AxesSubplot: xlabel='location', ylabel='age'>
```



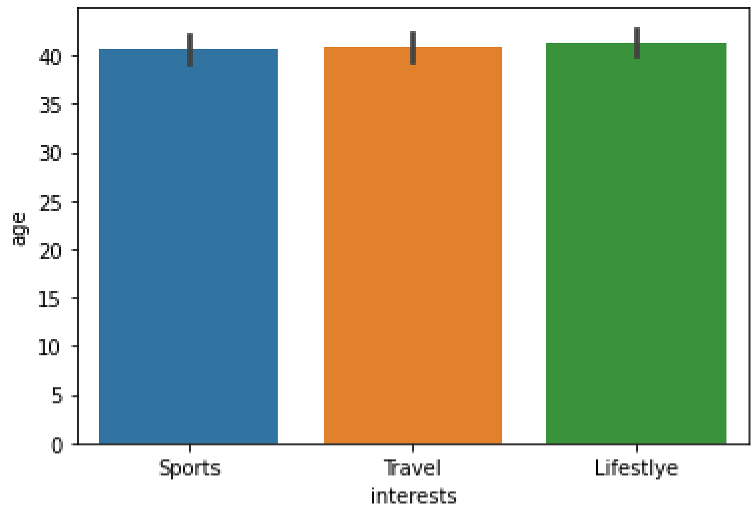
```
sns.swarmplot(y='age', x='interests', data=df)
```

```
<AxesSubplot: xlabel='interests', ylabel='age'>
```



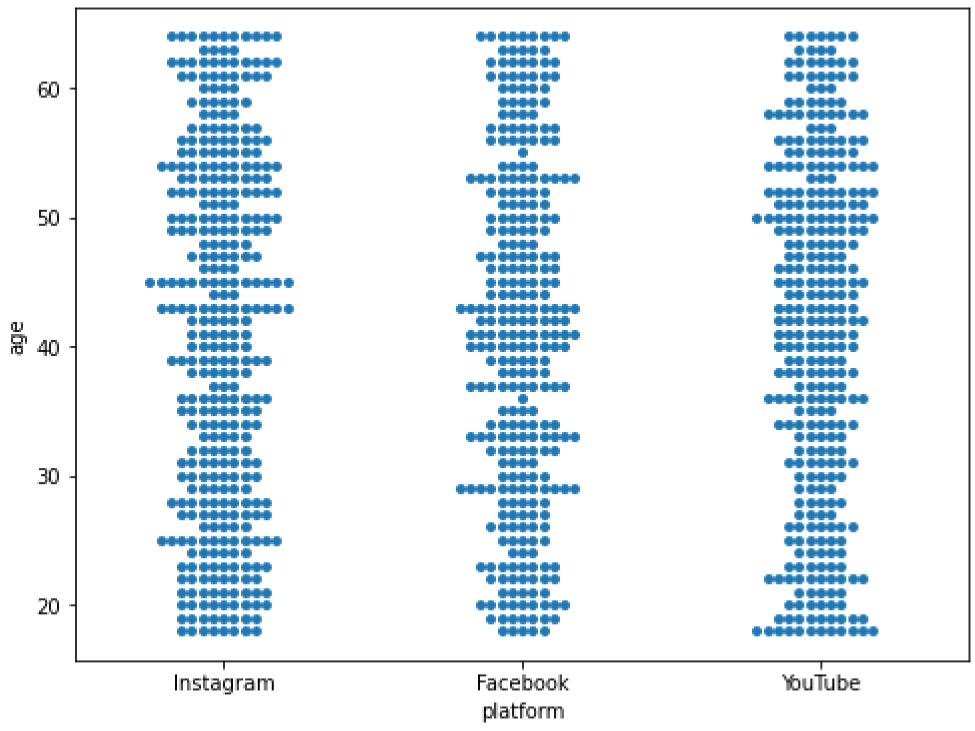
```
sns.barplot(y='age', x='interests', data=df)
```

```
<AxesSubplot: xlabel='interests', ylabel='age'>
```



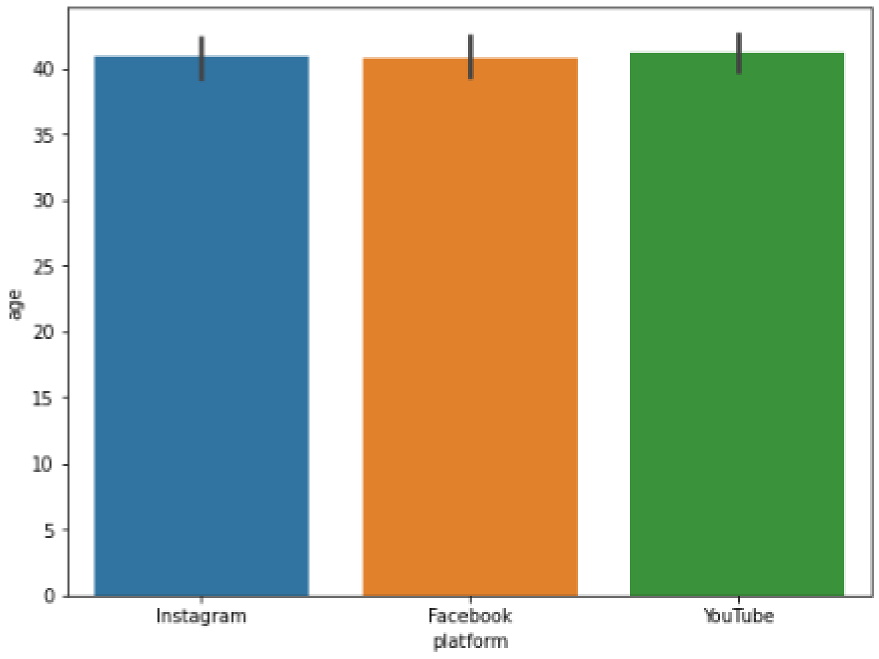
```
plt.figure(figsize=(8, 6))
sns.swarmplot(y='age', x='platform', data=df)
```

```
<AxesSubplot: xlabel='platform', ylabel='age'>
```



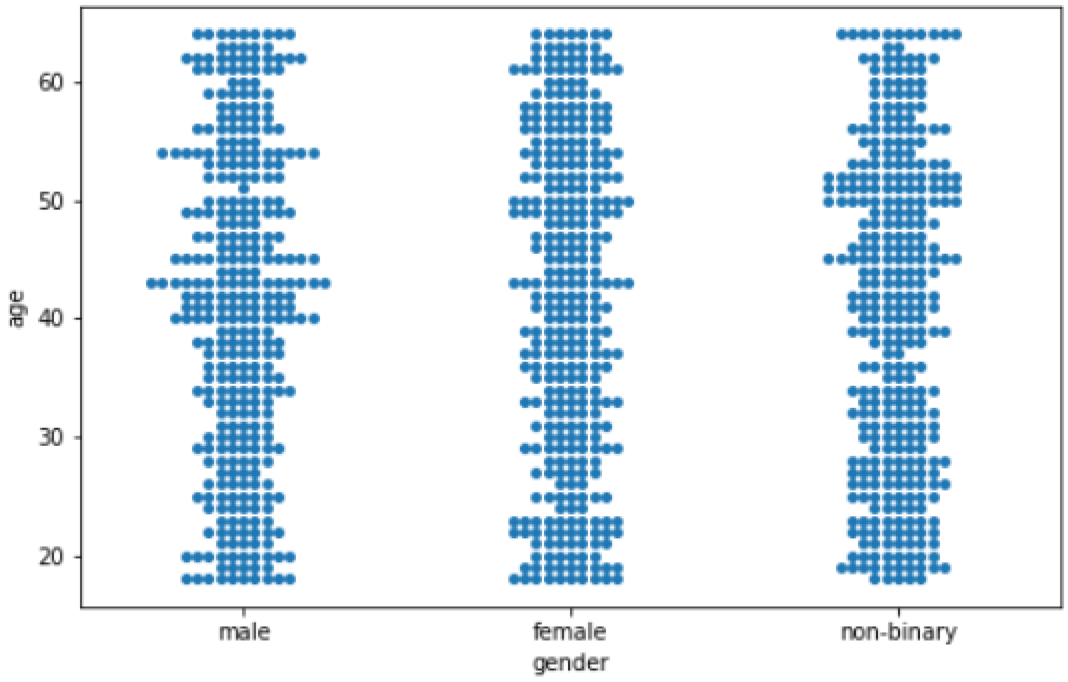
```
plt.figure(figsize=(8, 6))
sns.barplot(y='age', x='platform', data=df)
```

```
<AxesSubplot: xlabel='platform', ylabel='age'>
```



```
plt.figure(figsize=(8, 5))
sns.swarmplot(y='age', x='gender', data=df)
```

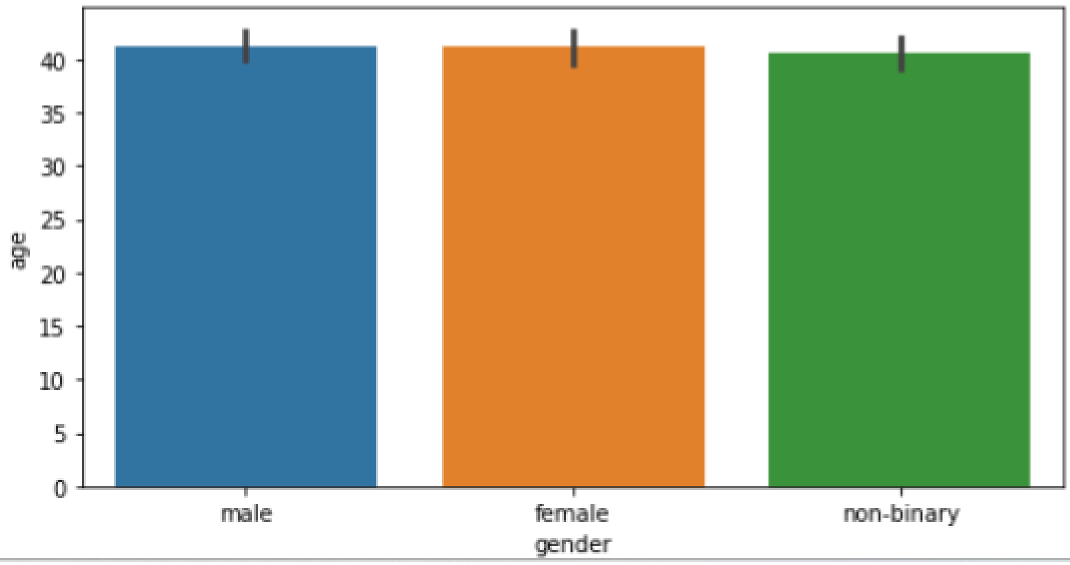
```
<AxesSubplot: xlabel='gender', ylabel='age'>
```



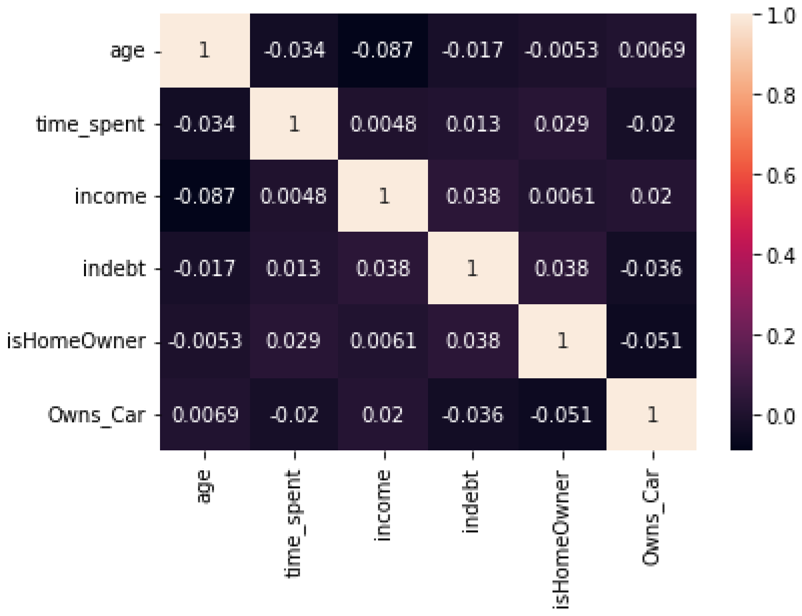
age vs other object columns

```
plt.figure(figsize=(8, 4))
sns.barplot(y='age', x='gender', data=df)
```

```
<AxesSubplot: xlabel='gender', ylabel='age'>
```



```
sns.heatmap(correlation_matrix ,annot=True)
plt.show()
```



	age	time_spent	income	indebt	isHomeOwner	Owns_Car
age	1	-0.034	-0.087	-0.017	-0.0053	0.0069
time_spent	-0.034	1	0.0048	0.013	0.029	-0.02
income	-0.087	0.0048	1	0.038	0.0061	0.02
indebt	-0.017	0.013	0.038	1	0.038	-0.036
isHomeOwner	-0.0053	0.029	0.0061	0.038	1	-0.051
Owns_Car	0.0069	-0.02	0.02	-0.036	-0.051	1

Multivariate-

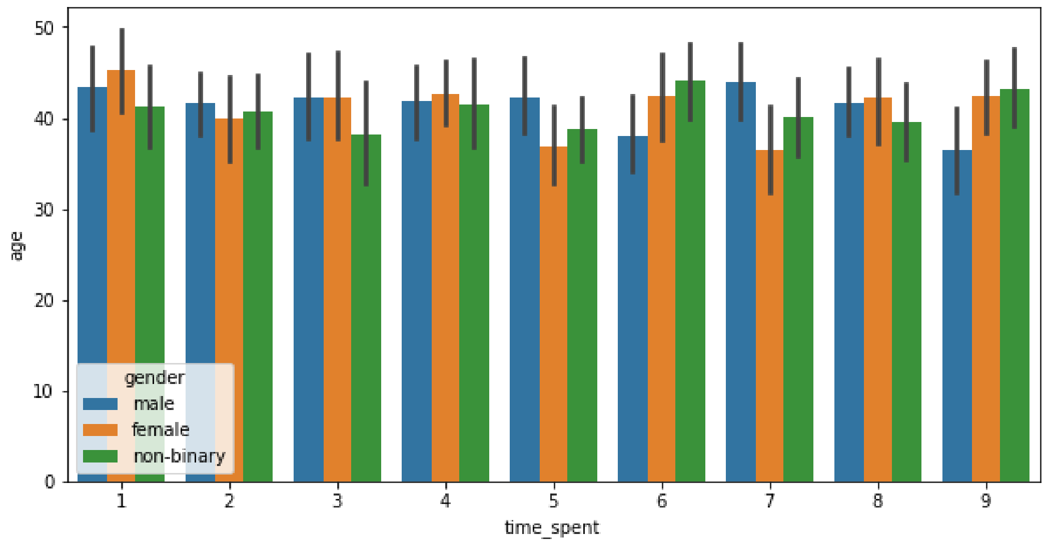
Multivariant ¶

```
df.columns
```

```
Index(['age', 'gender', 'time_spent', 'platform', 'interests', 'location',
       'demographics', 'profession', 'income', 'indebt', 'isHomeOwner',
       'Owns_Car'],
      dtype='object')
```

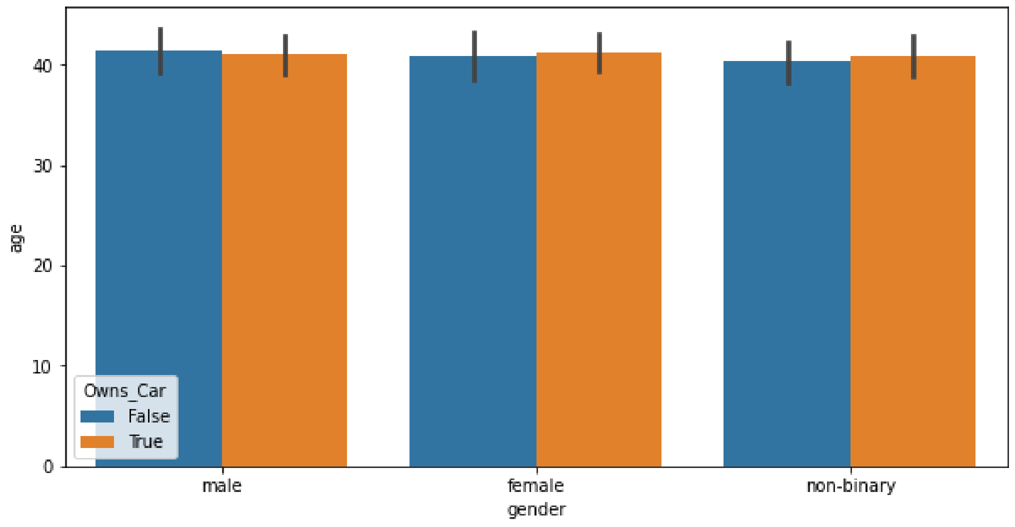
```
plt.figure(figsize=(10,5))
sns.barplot(x="time_spent", y="age", hue="gender", data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='age'>
```



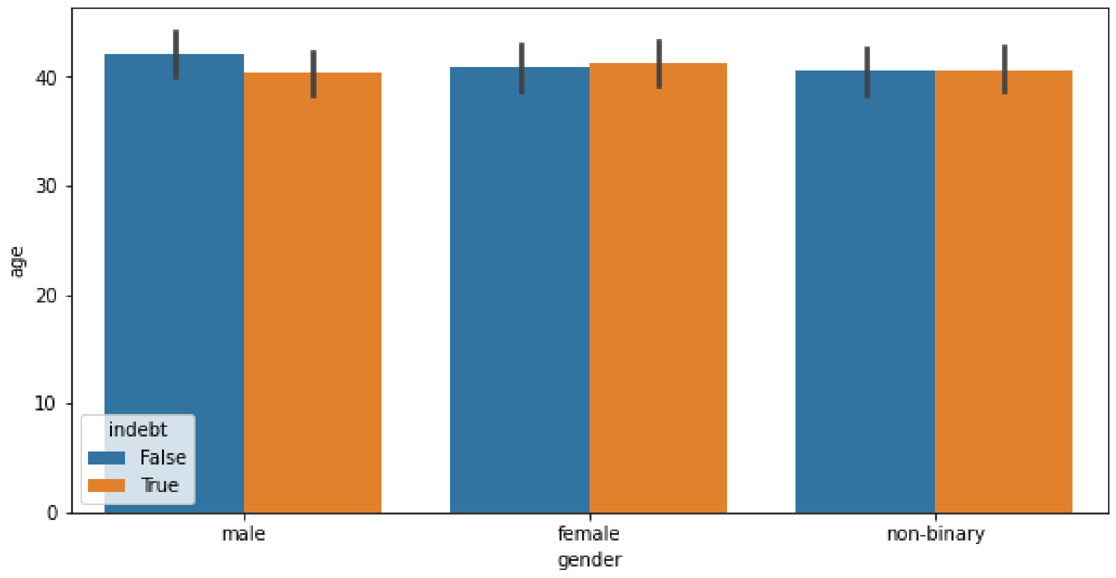
```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="gender", hue="Owns_Car", data=df)
```

```
<AxesSubplot: xlabel='gender', ylabel='age'>
```



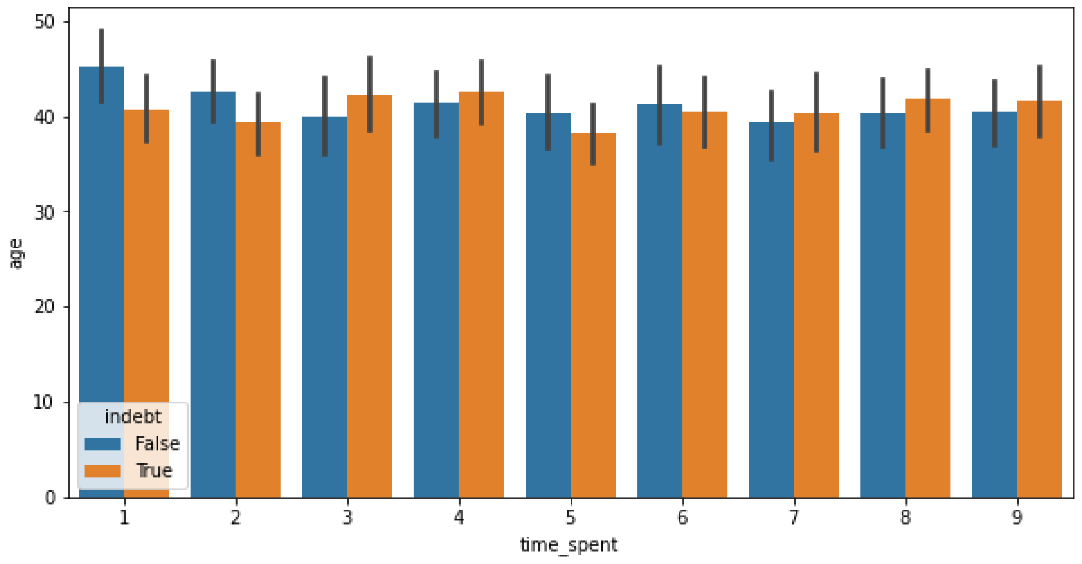
```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="gender", hue="indebt", data=df)
```

```
<AxesSubplot: xlabel='gender', ylabel='age'>
```



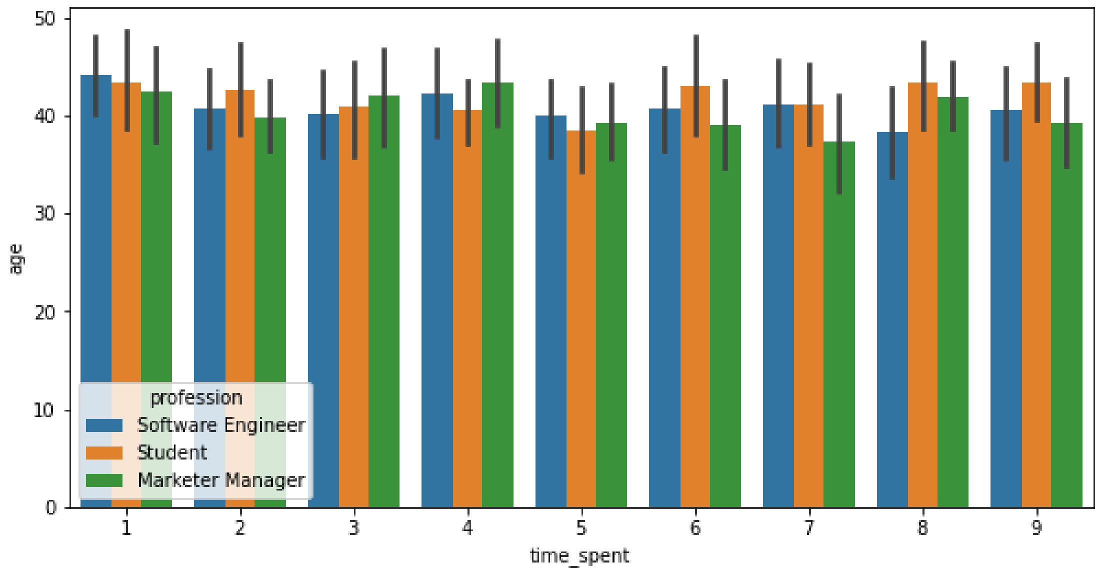
```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="time_spent", hue="indebt", data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='age'>
```



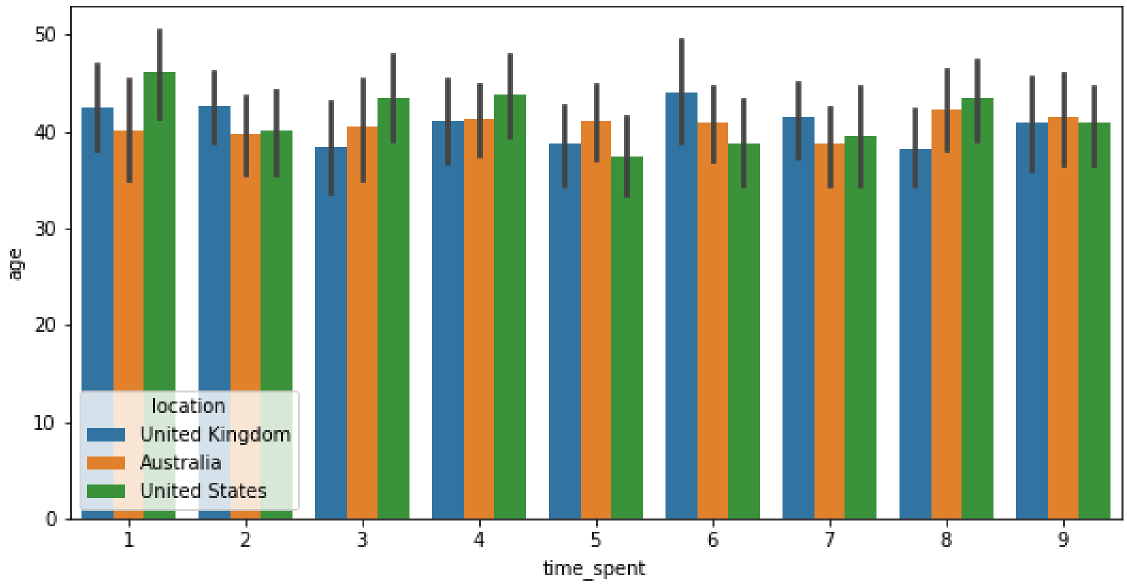
```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="time_spent", hue="profession", data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='age'>
```



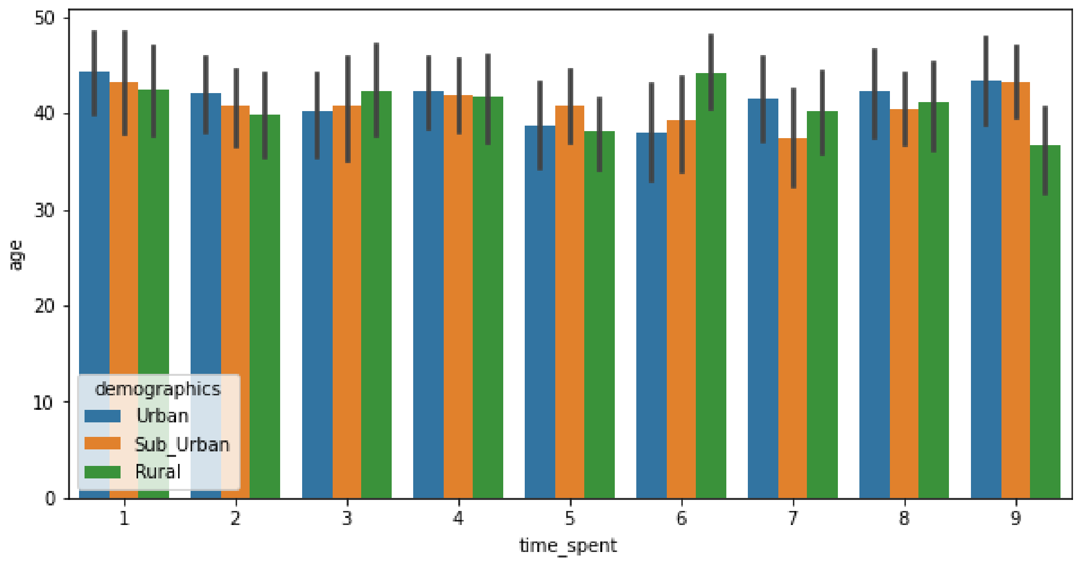
```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="time_spent", hue="location", data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='age'>
```



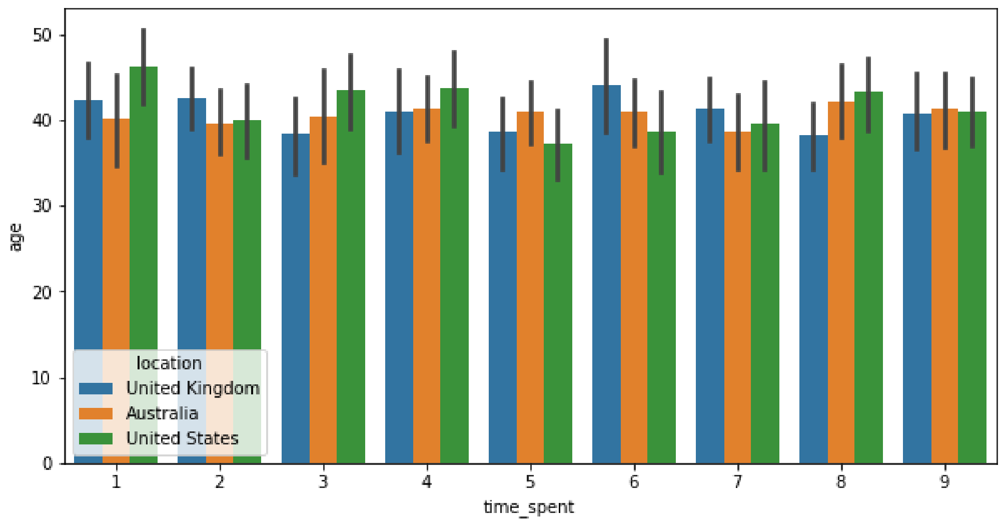
```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="time_spent", hue="demographics", data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='age'>
```



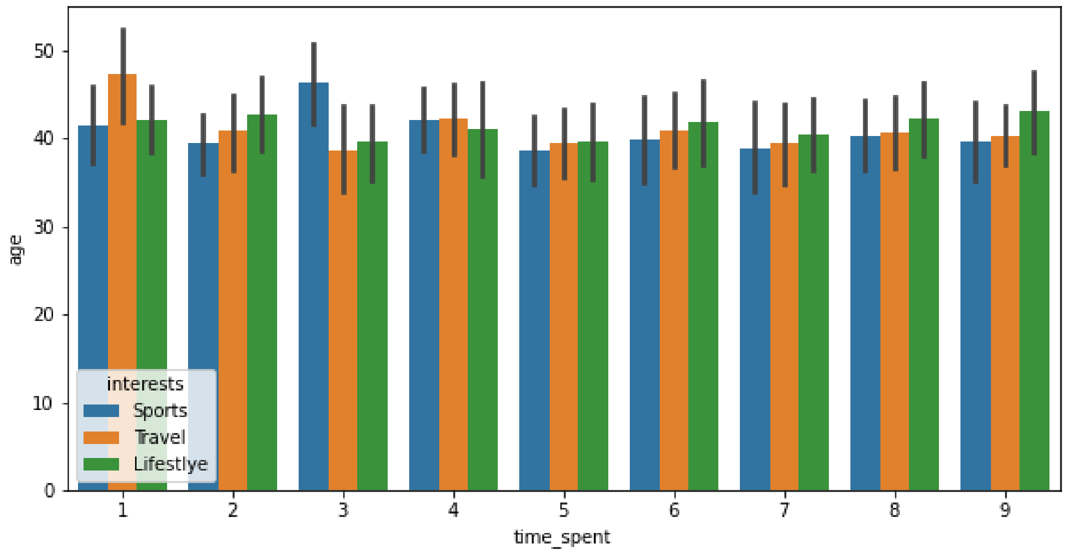
```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="time_spent", hue="location", data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='age'>
```



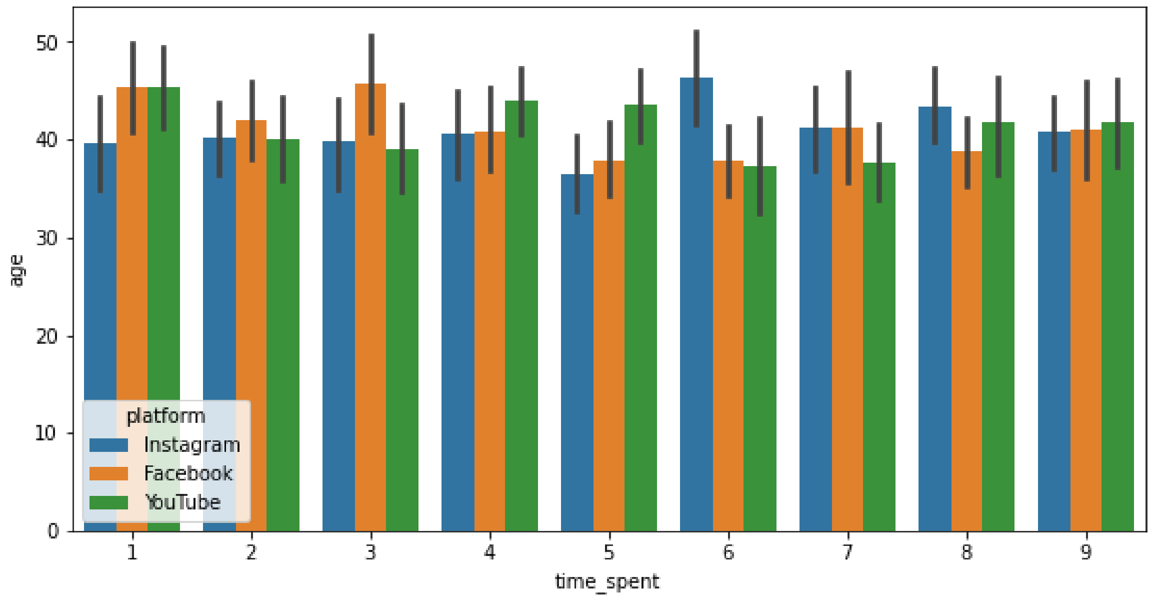
```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="time_spent", hue="interests", data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='age'>
```



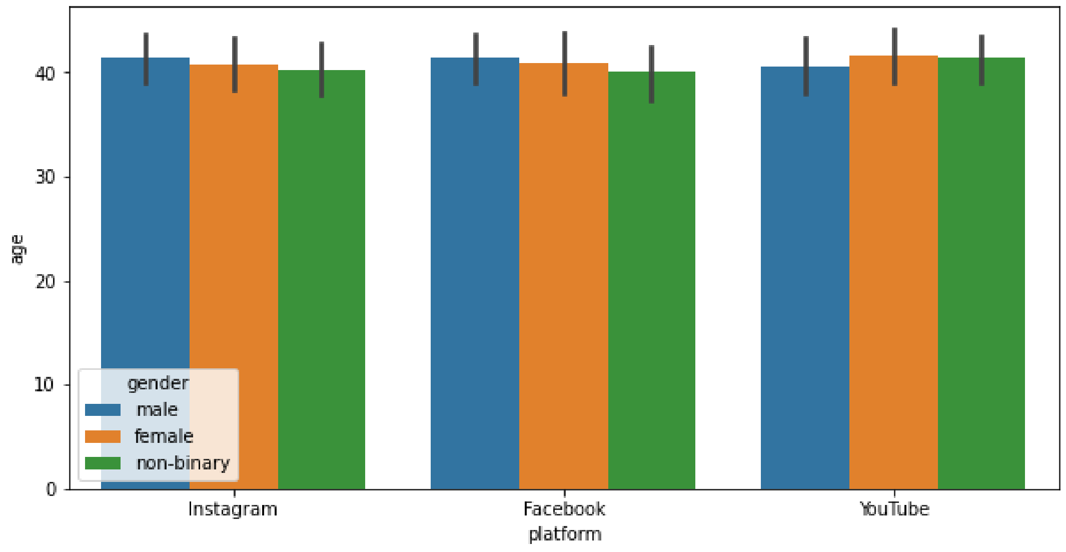
```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="time_spent", hue="platform", data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='age'>
```



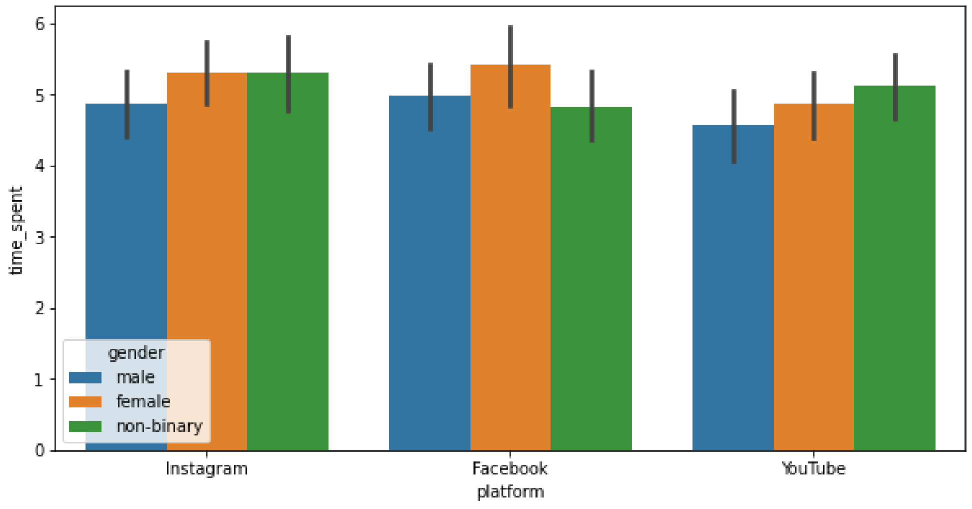
```
plt.figure(figsize=(10,5))
sns.barplot(x="platform", y="age", hue="gender", data=df)
```

```
<AxesSubplot: xlabel='platform', ylabel='age'>
```



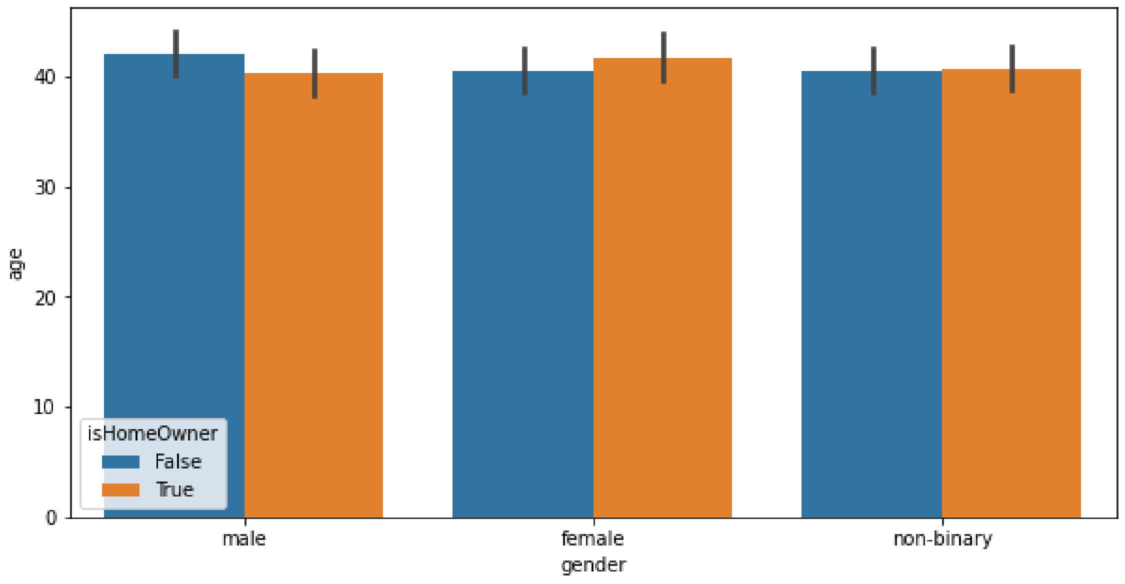
```
plt.figure(figsize=(10,5))
sns.barplot(x="platform", y="time_spent", hue="gender", data=df)
```

```
<AxesSubplot: xlabel='platform', ylabel='time_spent'>
```



```
plt.figure(figsize=(10,5))
sns.barplot(y="age", x="gender", hue="isHomeOwner", data=df)
```

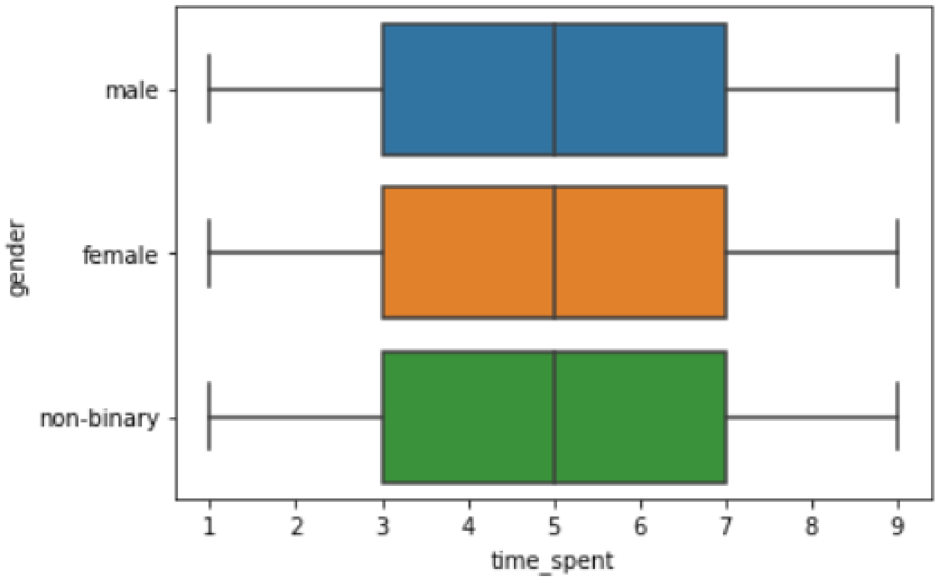
```
<AxesSubplot: xlabel='gender', ylabel='age'>
```



Outliers-

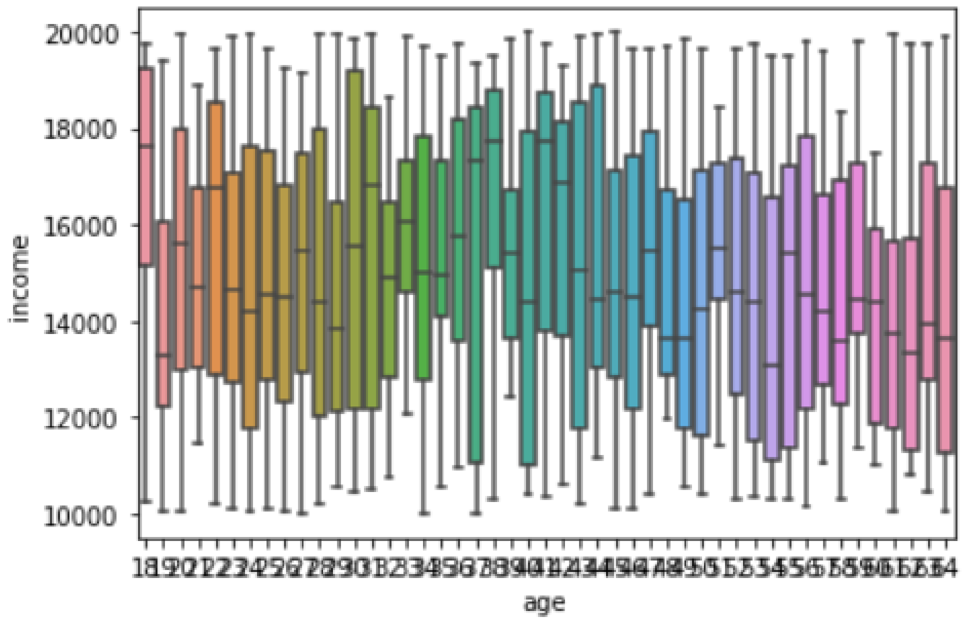
```
sns.boxplot(y='gender',x='time_spent', data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='gender'>
```



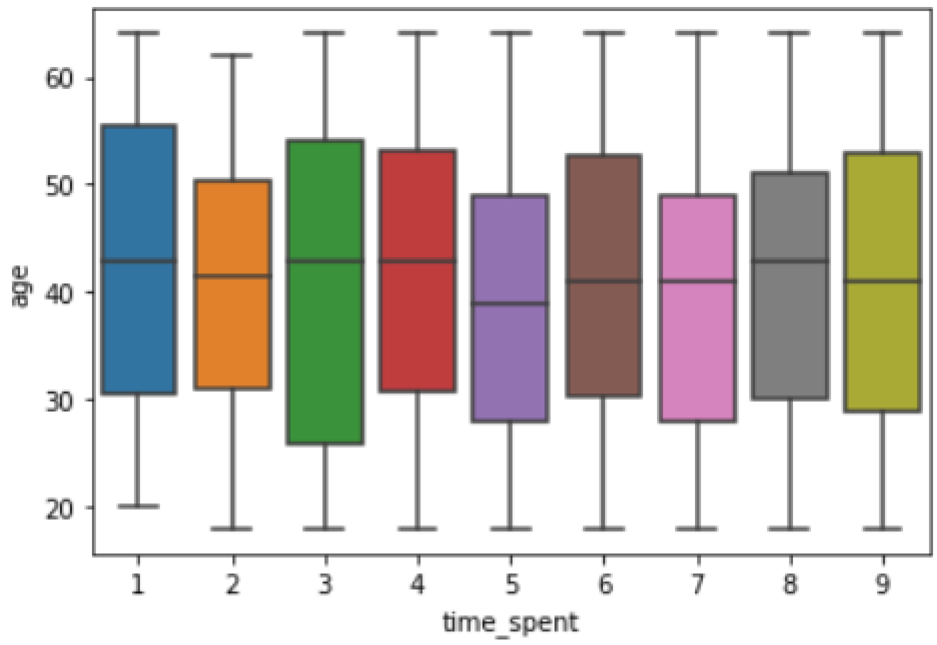
```
sns.boxplot(y='income',x='age', data=df)
```

```
<AxesSubplot: xlabel='age', ylabel='income'>
```



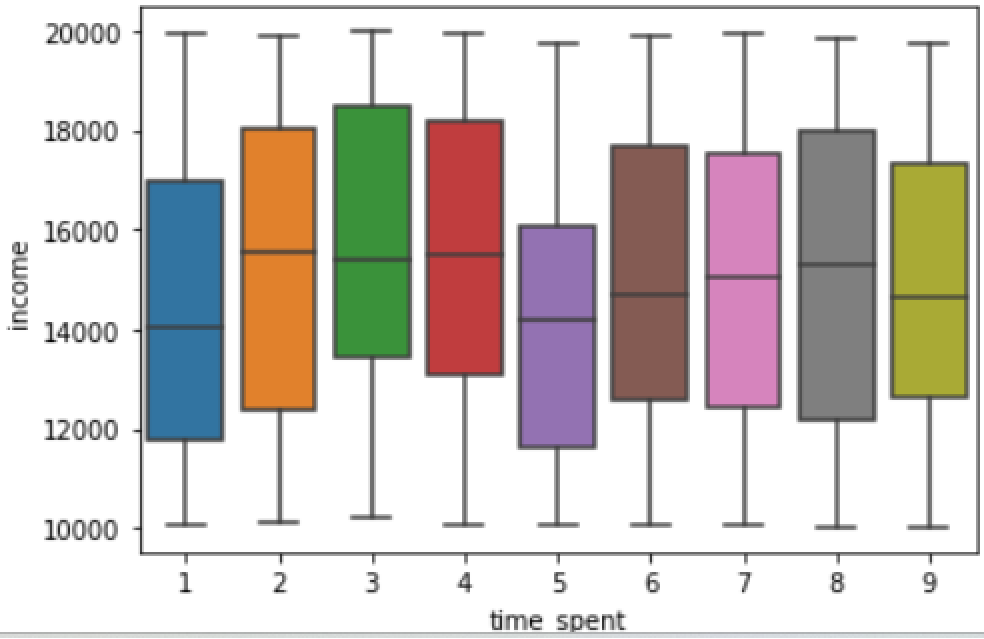
```
sns.boxplot(y='age',x='time_spent', data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='age'>
```



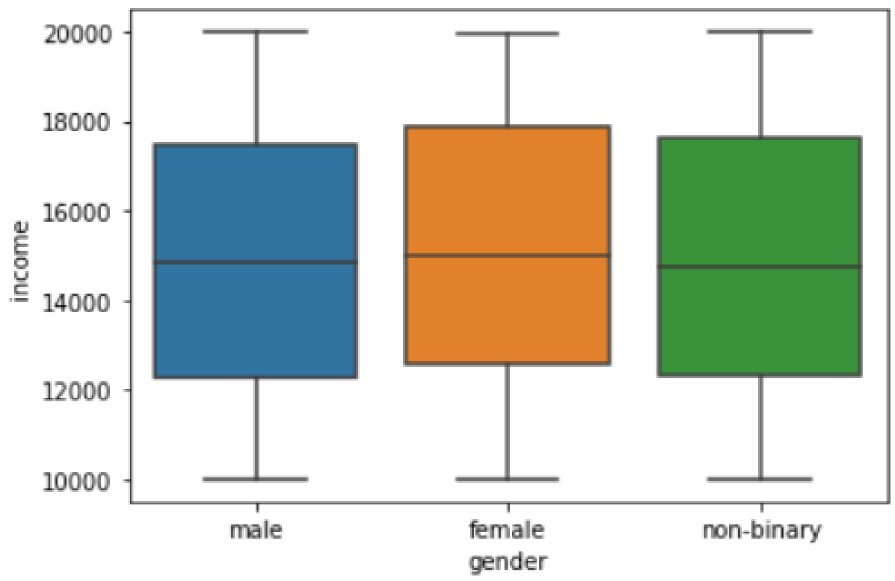
```
sns.boxplot(y='income',x='time_spent', data=df)
```

```
<AxesSubplot: xlabel='time_spent', ylabel='income'>
```



```
sns.boxplot(y='income',x='gender', data=df)
```

```
<AxesSubplot: xlabel='gender', ylabel='income'>
```


income
male
female
non-binary
gender