

# Patterns over time

EXPLORATORY DATA ANALYSIS IN PYTHON



Izzy Weber

Curriculum Manager, DataCamp

# Patterns over time

```
divorce = pd.read_csv("divorce.csv")
divorce.head()
```

```
marriage_date    marriage_duration
0   2000-06-26          5.0
1   2000-02-02          2.0
2   1991-10-09         10.0
3   1993-01-02         10.0
4   1998-12-11          7.0
```

# Importing DateTime data

- DateTime data needs to be explicitly declared to Pandas

```
divorce.dtypes
```

```
marriage_date      object
marriage_duration   float64
dtype: object
```

# Importing DateTime data

```
divorce = pd.read_csv("divorce.csv", parse_dates=["marriage_date"])
divorce.dtypes
```

```
marriage_date          datetime64[ns]
marriage_duration      float64
dtype: object
```

# Converting to DateTime data

- `pd.to_datetime()` converts arguments to DateTime data

```
divorce["marriage_date"] = pd.to_datetime(divorce["marriage_date"])
divorce.dtypes
```

```
marriage_date          datetime64[ns]
marriage_duration      float64
dtype: object
```

# Creating DateTime data

```
divorce.head(2)
```

```
month day year marriage_duration  
0      6   26  2000           5.0  
1      2    2  2000           2.0
```

```
divorce["marriage_date"] = pd.to_datetime(divorce[["month", "day", "year"]])  
divorce.head(2)
```

```
month day year marriage_duration marriage_date  
0      6   26  2000           5.0  2000-06-26  
1      2    2  2000           2.0  2000-02-02
```

# Creating DateTime data

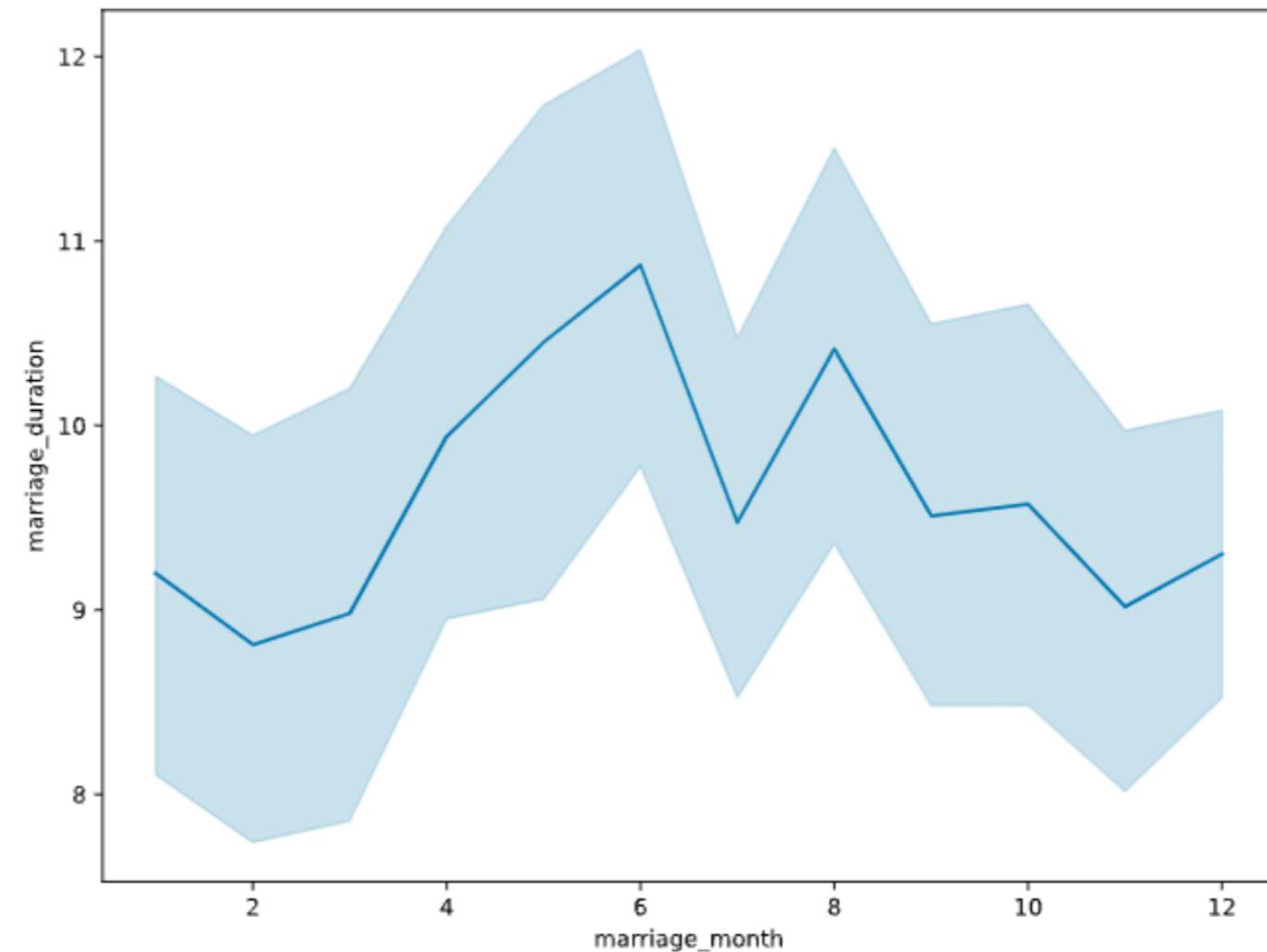
- Extract parts of a full date using `dt.month`, `dt.day`, and `dt.year` attributes

```
divorce["marriage_month"] = divorce["marriage_date"].dt.month  
divorce.head()
```

	marriage_date	marriage_duration	marriage_month
0	2000-06-26	5.0	6
1	2000-02-02	2.0	2
2	1991-10-09	10.0	10
3	1993-01-02	10.0	1
4	1998-12-11	7.0	12

# Visualizing patterns over time

```
sns.lineplot(data=divorce, x="marriage_month", y="marriage_duration")  
plt.show()
```



# **Let's practice!**

**EXPLORATORY DATA ANALYSIS IN PYTHON**

# Correlation

EXPLORATORY DATA ANALYSIS IN PYTHON



Izzy Weber

Curriculum Manager, DataCamp

# Correlation

- Describes direction and strength of relationship between two variables
- Can help us use variables to predict future outcomes

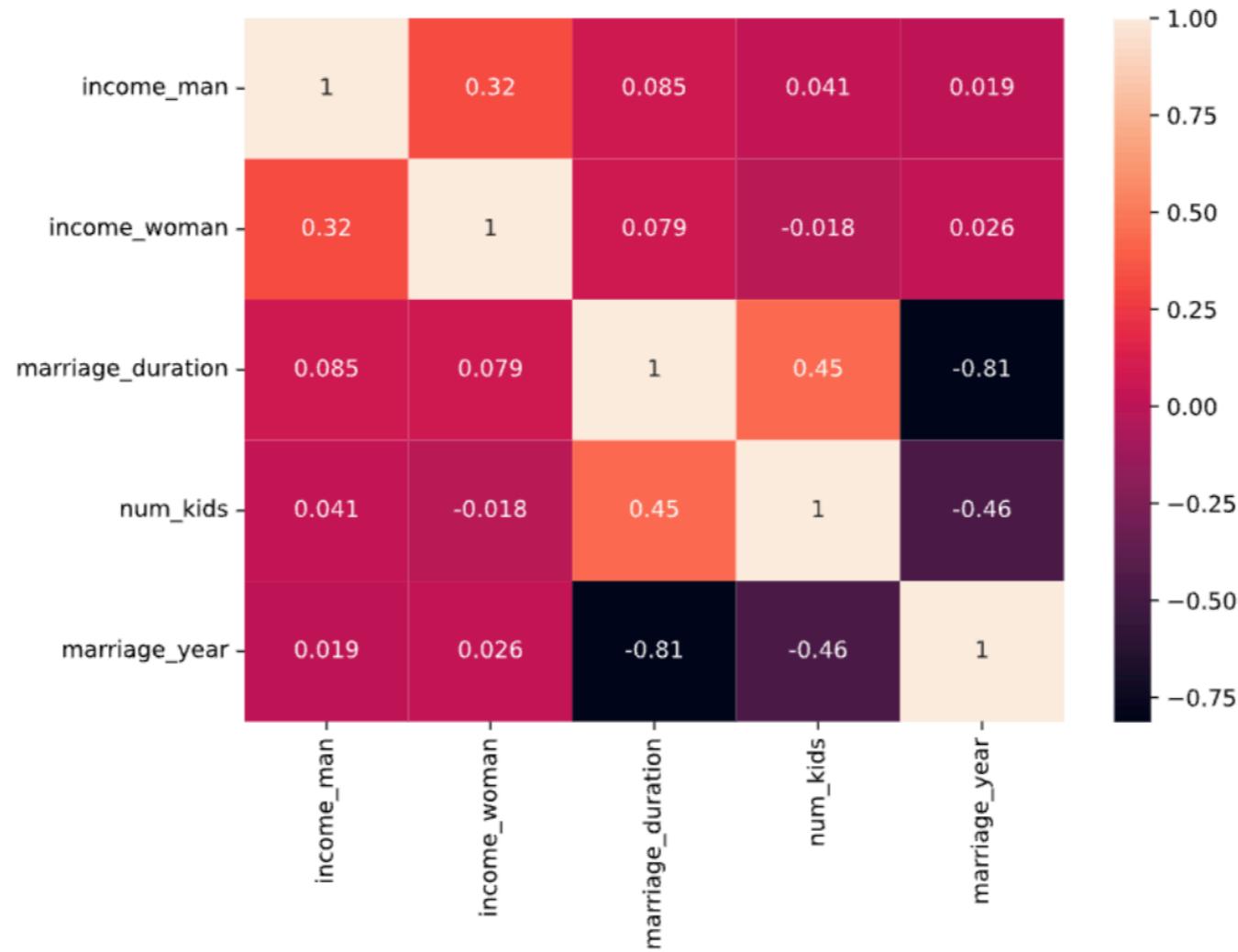
```
divorce.corr()
```

	income_man	income_woman	marriage_duration	num_kids	marriage_year
income_man	1.000	0.318	0.085	0.041	0.019
income_woman	0.318	1.000	0.079	-0.018	0.026
marriage_duration	0.085	0.079	1.000	0.447	-0.812
num_kids	0.041	-0.018	0.447	1.000	-0.461
marriage_year	0.019	0.026	-0.812	-0.461	1.000

- `.corr()` calculates Pearson correlation coefficient, measuring *linear* relationship

# Correlation heatmaps

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



# Correlation in context

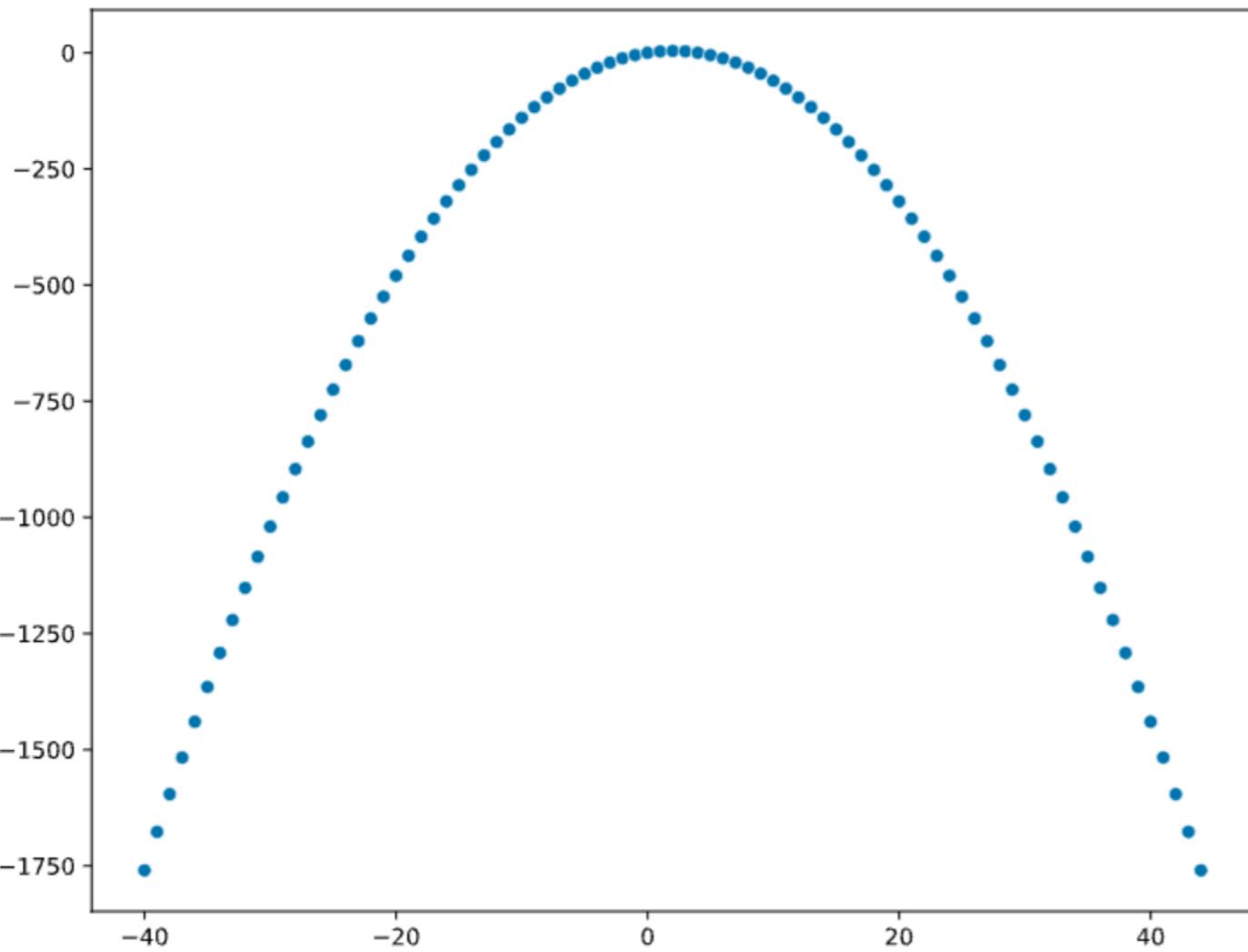
```
divorce["divorce_date"].min()
```

```
Timestamp('2000-01-08 00:00:00')
```

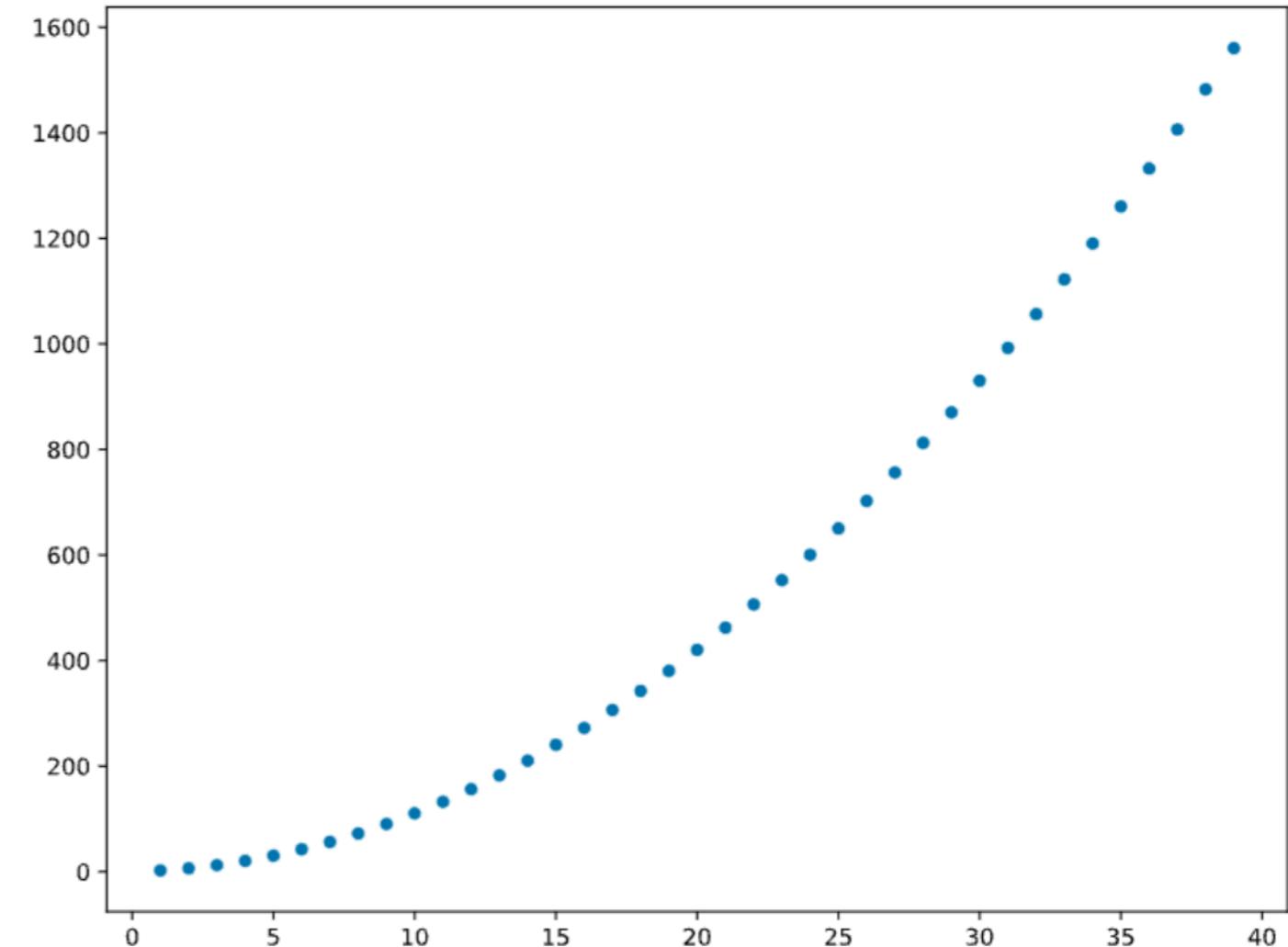
```
divorce["divorce_date"].max()
```

```
Timestamp('2015-11-03 00:00:00')
```

# Visualizing relationships



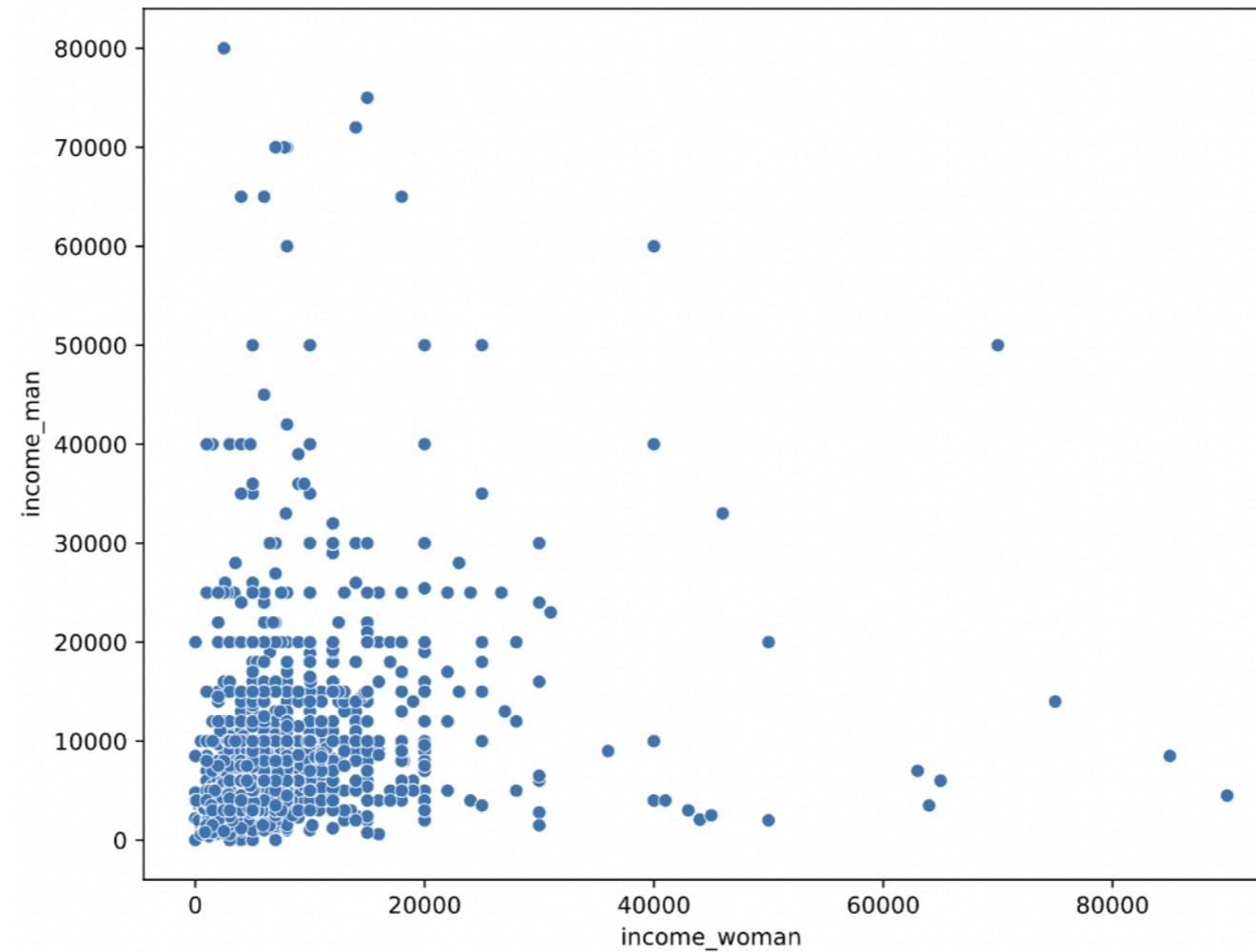
- Strong relationship—but not linear
- Pearson correlation coefficient: `-6.48e-18`



- Quadratic relationship; not linear
- Pearson correlation coefficient: `.971211`

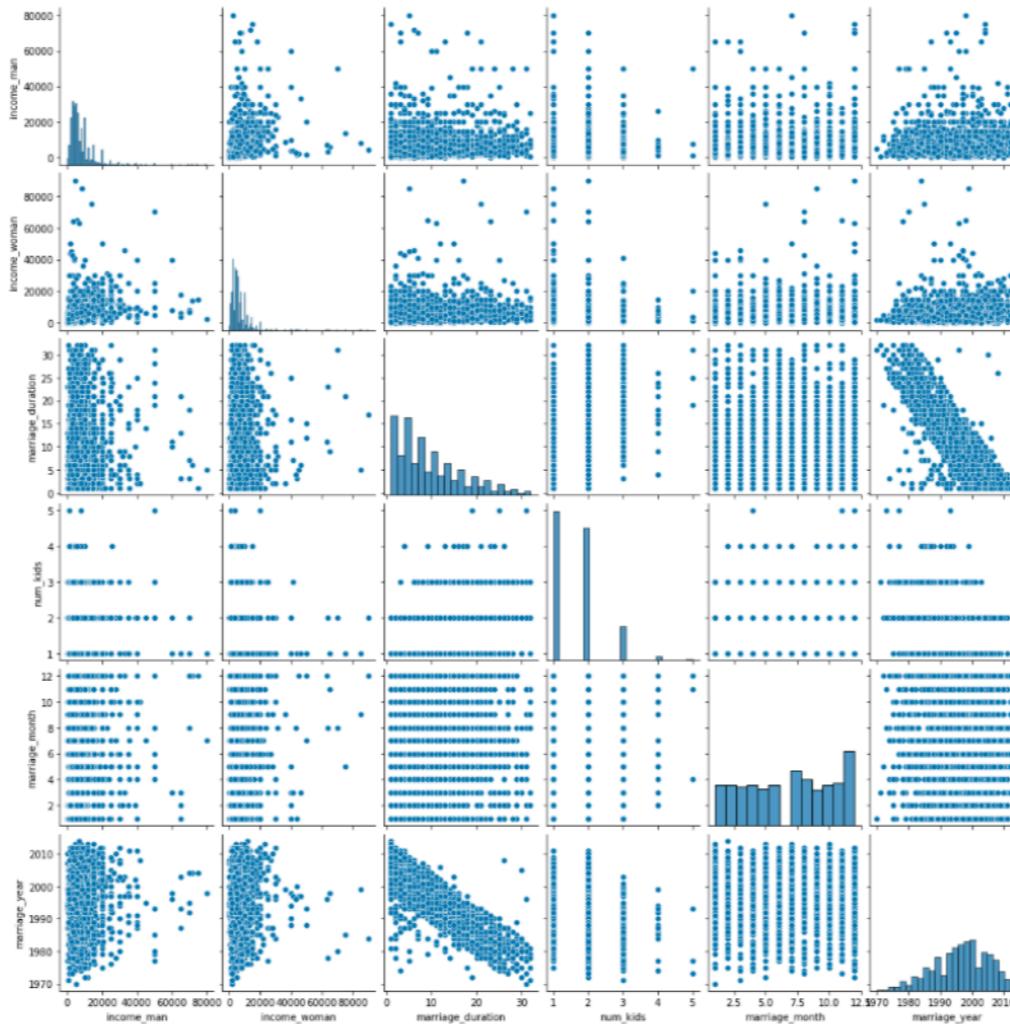
# Scatter plots

```
sns.scatterplot(data=divorce, x="income_man", y="income_woman")  
plt.show()
```



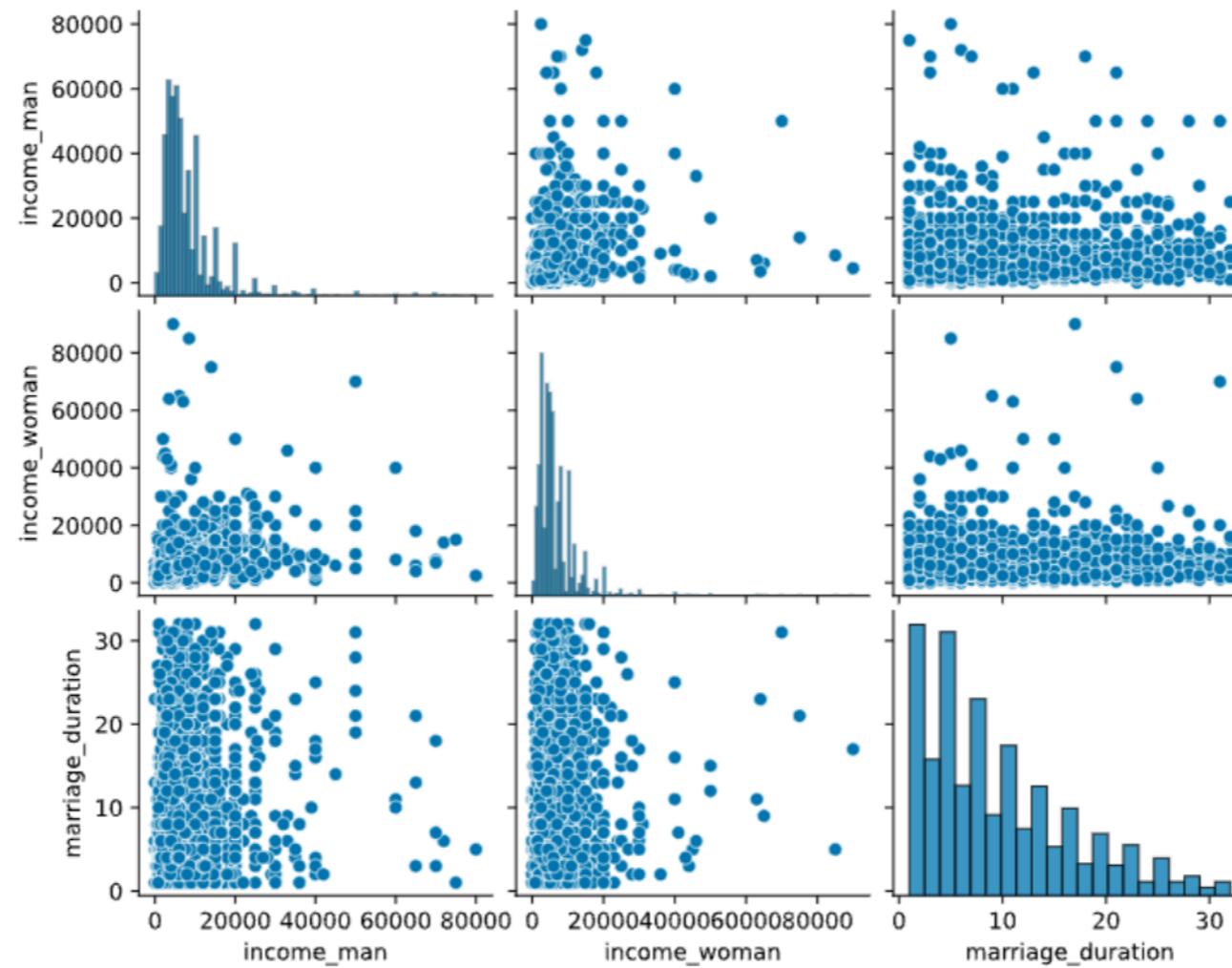
# Pairplots

```
sns.pairplot(data=divorce)  
plt.show()
```



# Pairplots

```
sns.pairplot(data=divorce, vars=["income_man", "income_woman", "marriage_duration"])
plt.show()
```



# **Let's practice!**

**EXPLORATORY DATA ANALYSIS IN PYTHON**

# Factor relationships and distributions

EXPLORATORY DATA ANALYSIS IN PYTHON



Izzy Weber

Curriculum Manager, DataCamp

# Level of education: male partner

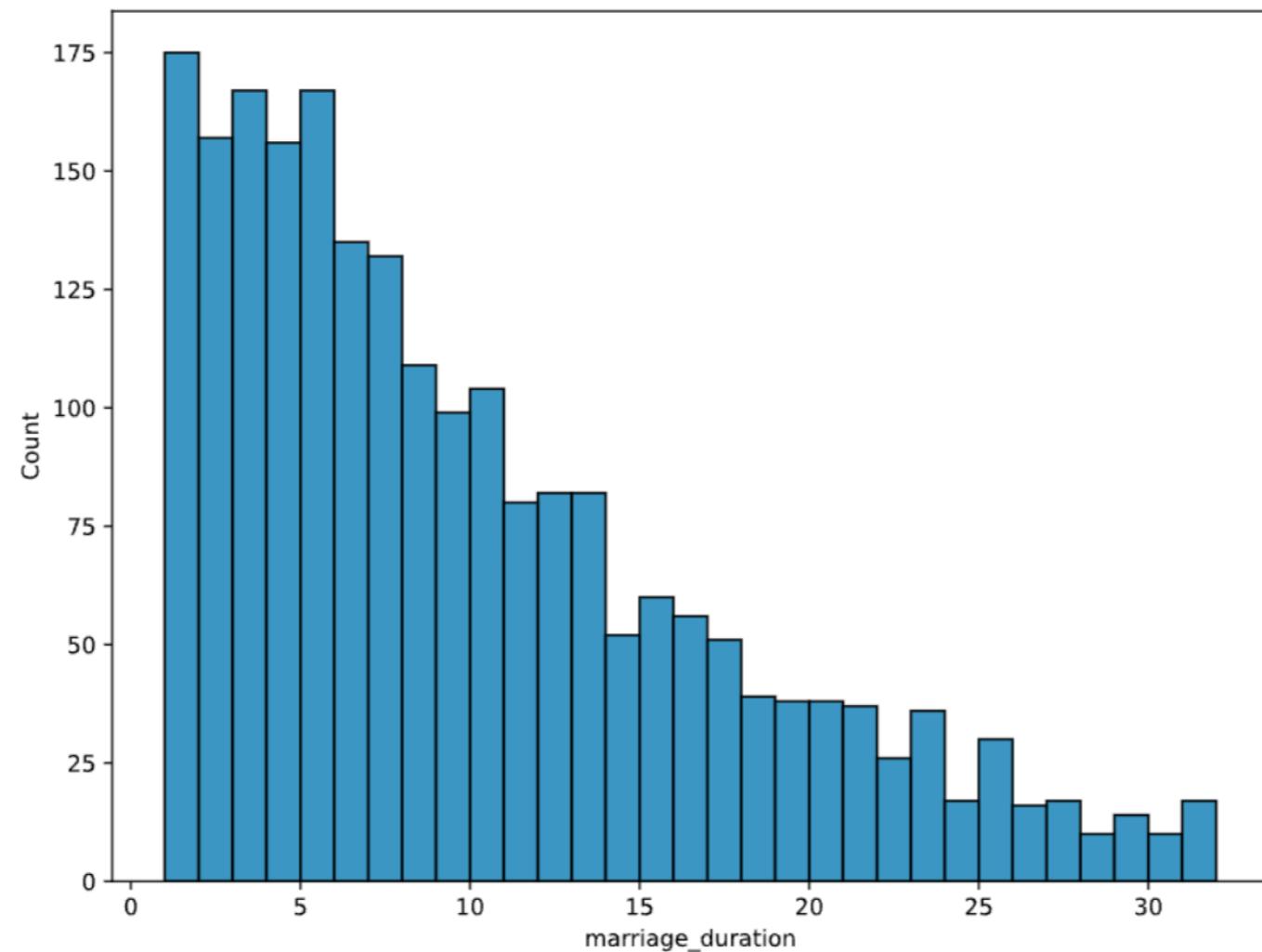
```
divorce["education_man"].value_counts()
```

```
Professional      1313  
Preparatory       501  
Secondary        288  
Primary           100  
None              4  
Other              3
```

```
Name: education_man, dtype: int64
```

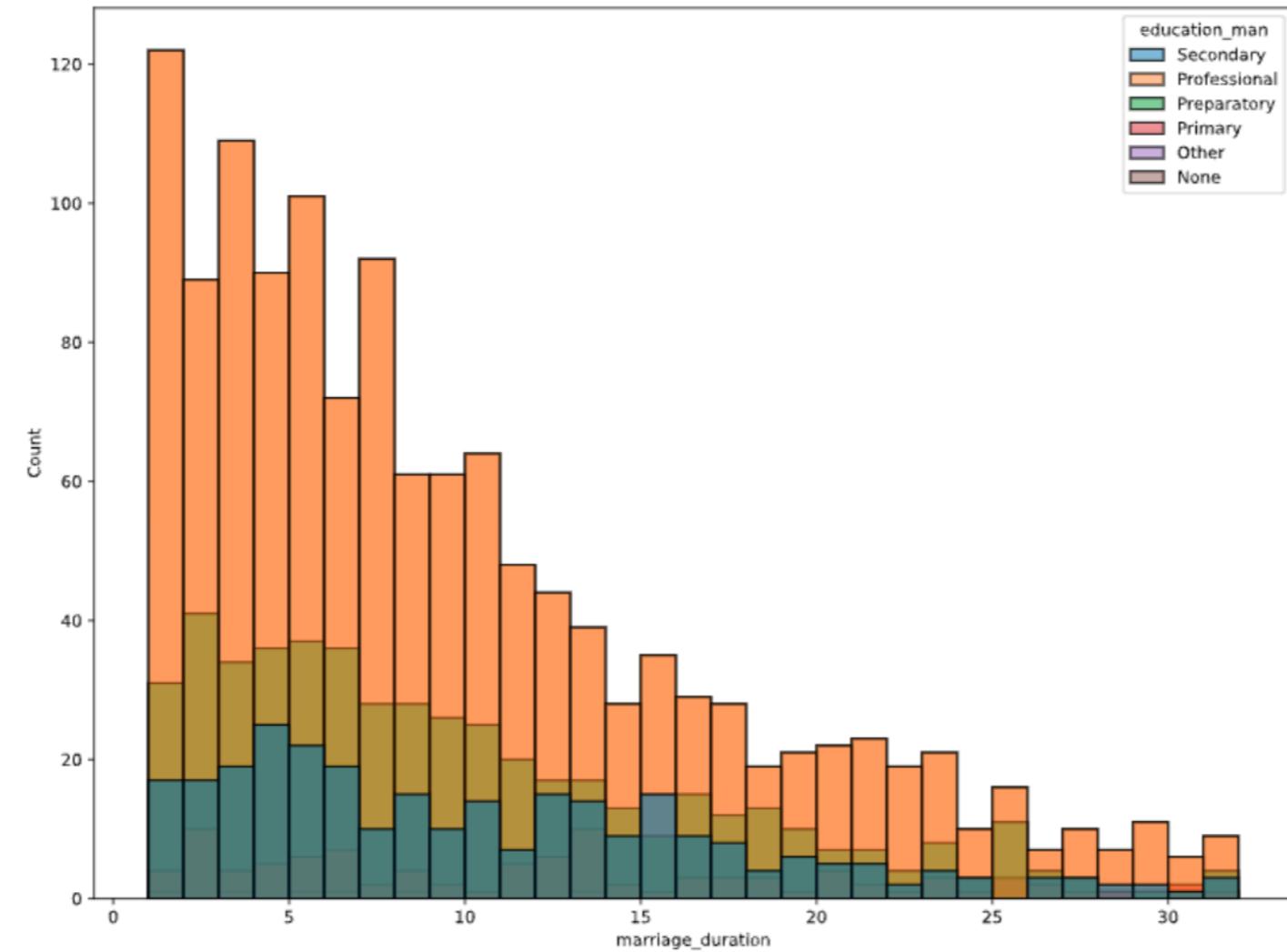
# Exploring categorical relationships

```
sns.histplot(data=divorce, x="marriage_duration", binwidth=1)  
plt.show()
```



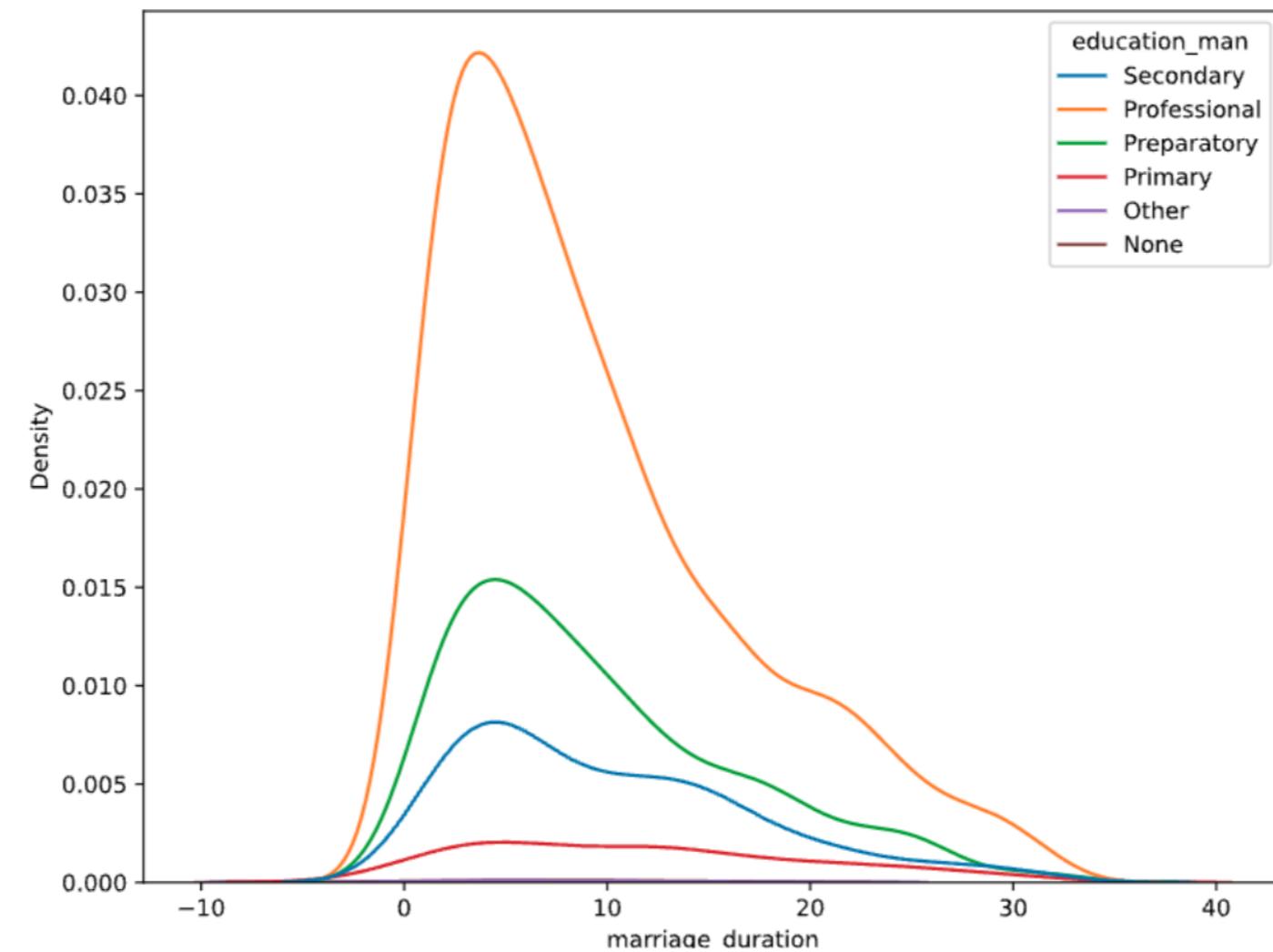
# Exploring categorical relationships

```
sns.histplot(data=divorce, x="marriage_duration", hue="education_man", binwidth=1)  
plt.show()
```

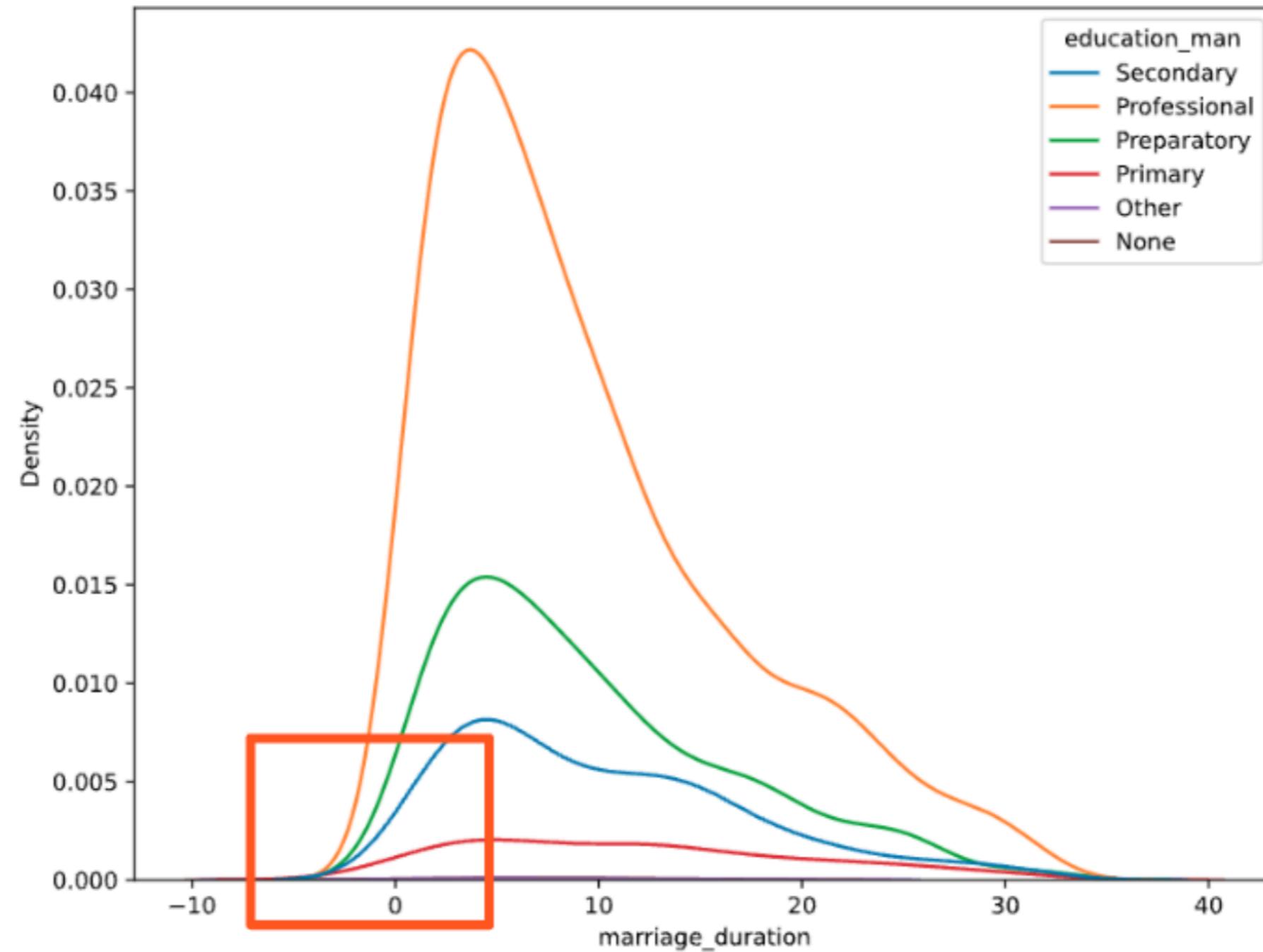


# Kernel Density Estimate (KDE) plots

```
sns.kdeplot(data=divorce, x="marriage_duration", hue="education_man")  
plt.show()
```

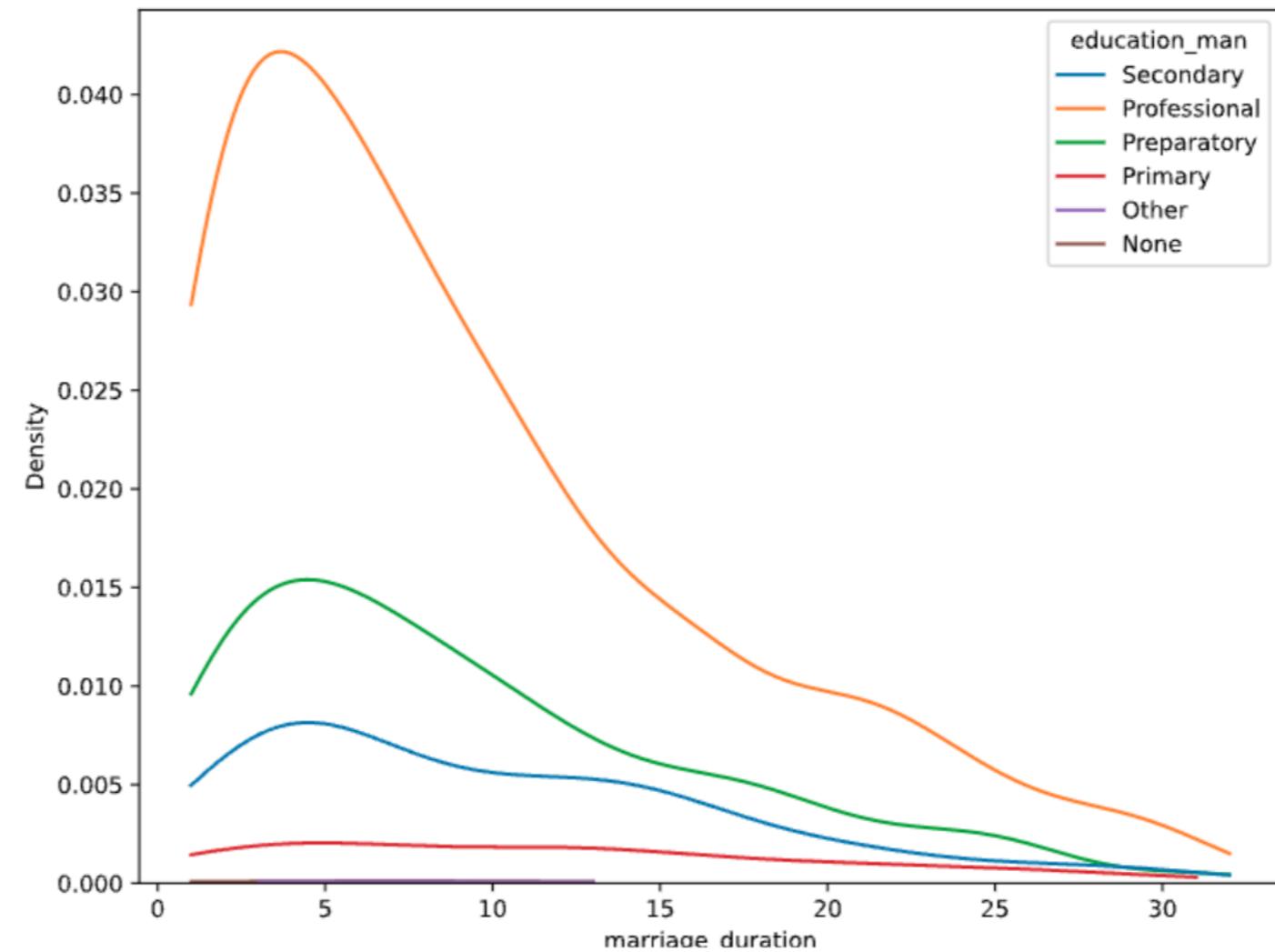


# Kernel Density Estimate (KDE) plots



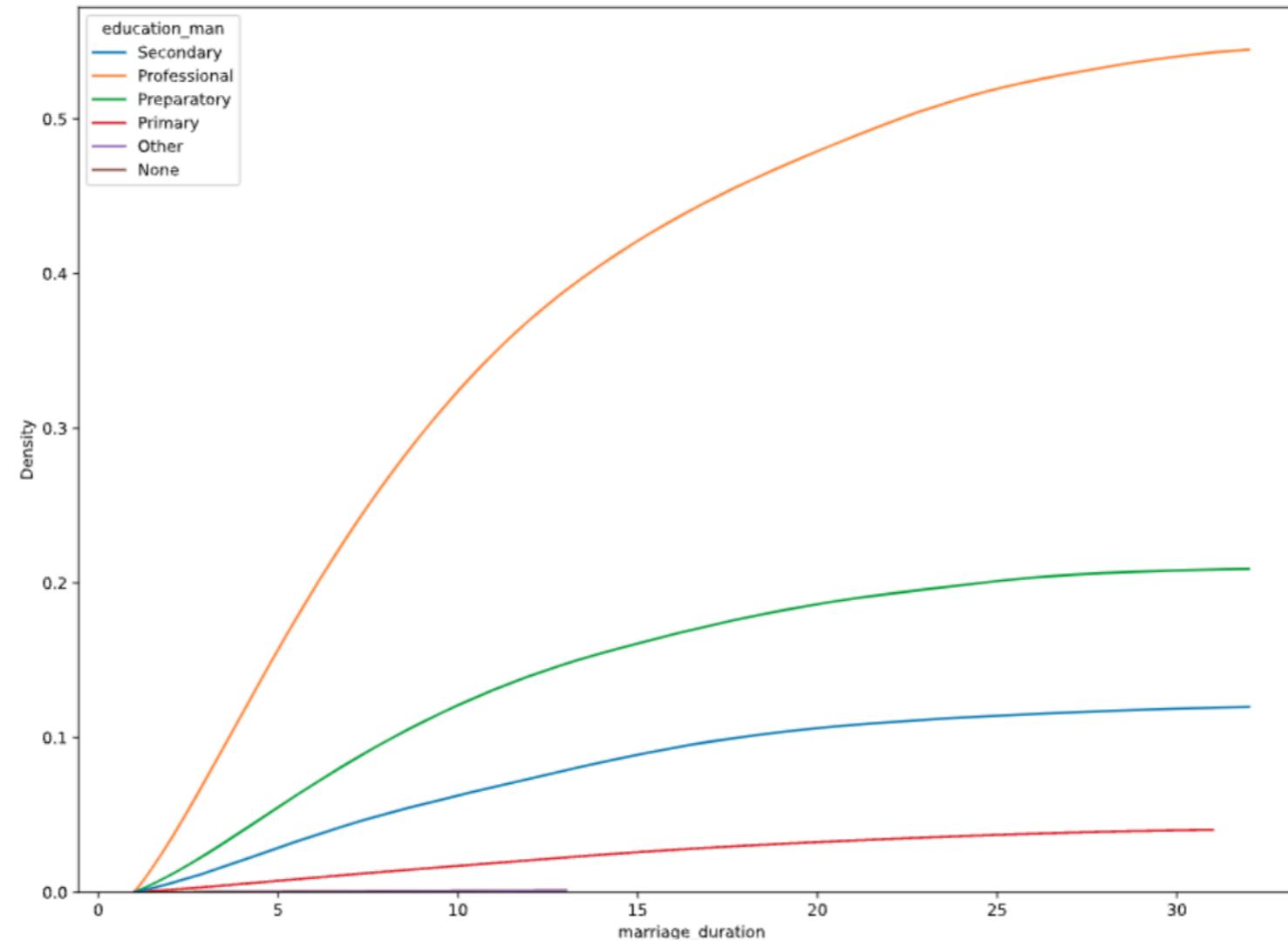
# Kernel Density Estimate (KDE) plots

```
sns.kdeplot(data=divorce, x="marriage_duration", hue="education_man", cut=0)  
plt.show()
```



# Cumulative KDE plots

```
sns.kdeplot(data=divorce, x="marriage_duration", hue="education_man", cut=0, cumulative=True)  
plt.show()
```



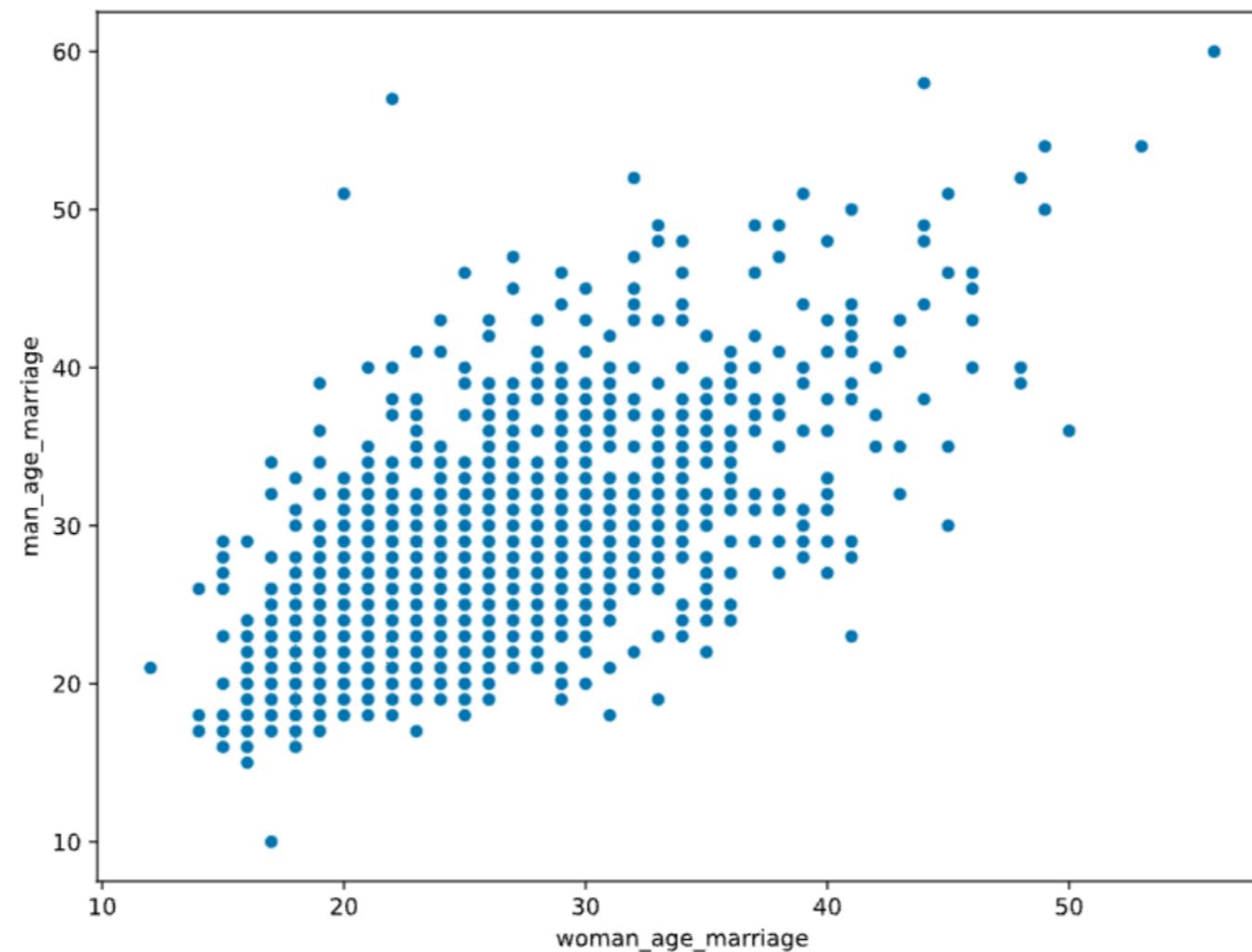
# Relationship between marriage age and education

- Is there a relationship between age at marriage and education level?

```
divorce["man_age_marriage"] = divorce["marriage_year"] - divorce["dob_man"].dt.year  
divorce["woman_age_marriage"] = divorce["marriage_year"] - divorce["dob_woman"].dt.year
```

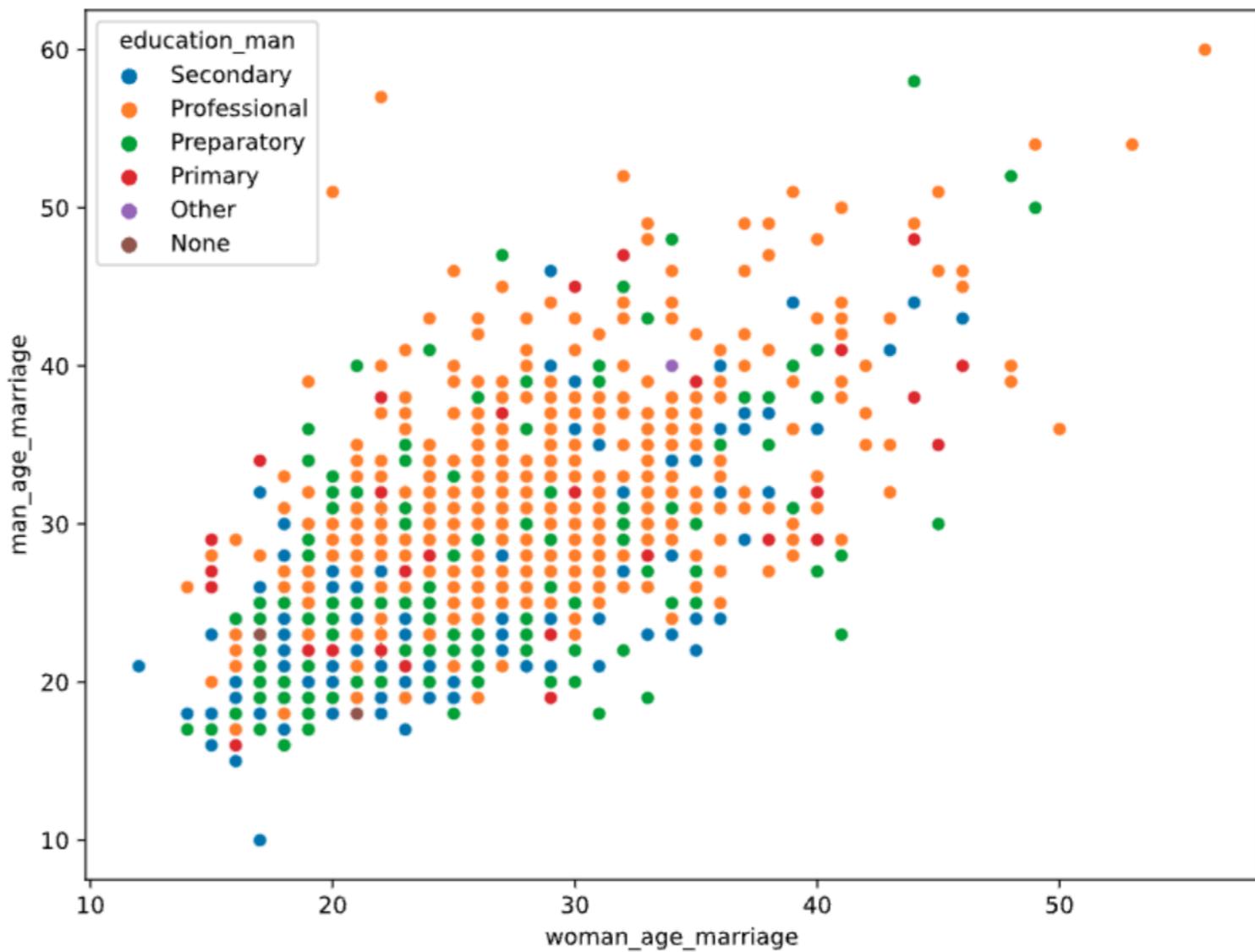
# Scatter plot with categorical variables

```
sns.scatterplot(data=divorce, x="woman_age_marriage", y="man_age_marriage")  
plt.show()
```



# Scatter plot with categorical variables

```
sns.scatterplot(data=divorce,  
                 x="woman_age_marriage",  
                 y="man_age_marriage",  
                 hue="education_man")  
  
plt.show()
```



# **Let's practice!**

**EXPLORATORY DATA ANALYSIS IN PYTHON**