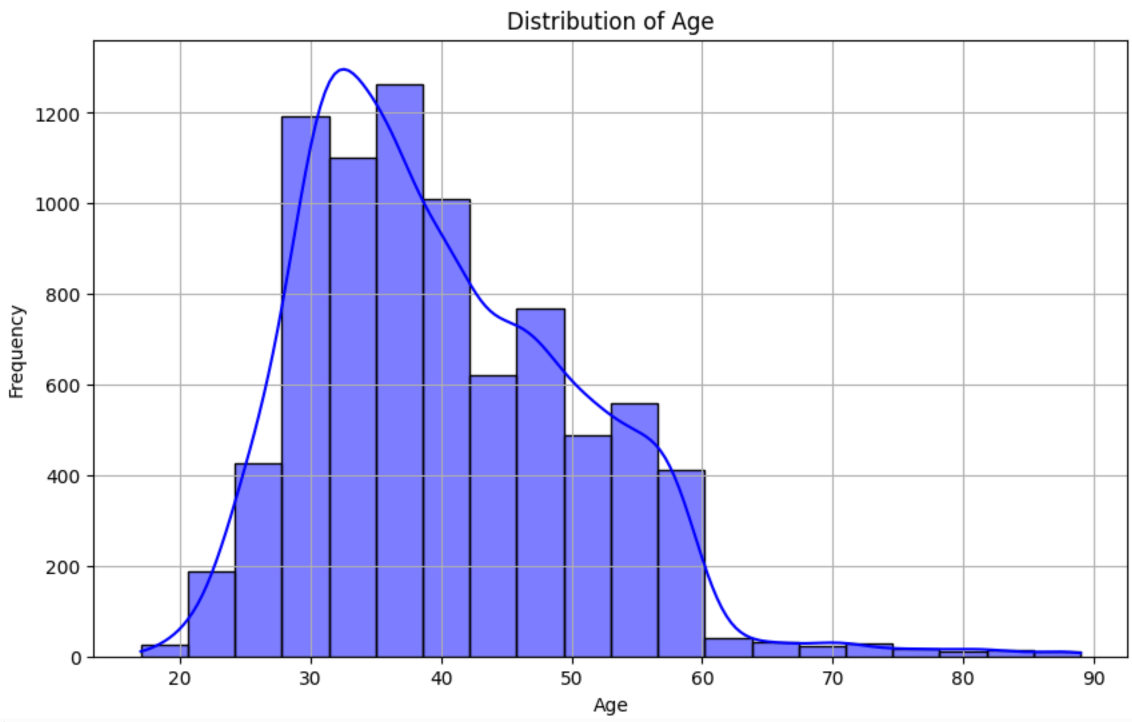
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Load the dataset
data = pd.read_csv('bank_marketing_test.csv')

# Displaying the statistical summary of the dataset
summary = data.describe()
print(summary)
```

```
               age     duration    campaign        pdays     previous  \
count  8237.000000  8237.000000  8237.00000  8237.000000  8237.000000
mean     40.116547   256.007648     2.60471   962.228724     0.174335
std      10.465328   259.728737     2.91562   187.533881     0.500565
min      17.000000     4.000000     1.00000     0.000000     0.000000
25%      32.000000   101.000000     1.00000   999.000000     0.000000
50%      38.000000   179.000000     2.00000   999.000000     0.000000
75%      47.000000   316.000000     3.00000   999.000000     0.000000
max      89.000000  4918.000000    43.00000   999.000000     6.000000

       emp.var.rate  cons.price.idx  cons.conf.idx    euribor3m  nr.employed
count   8237.000000     8237.000000    8237.000000  8237.000000  8237.000000
mean       0.070147       93.577806     -40.545320     3.608206  5166.589790
std        1.574685        0.582138       4.623626     1.735931    72.470977
min       -3.400000       92.201000     -50.800000     0.634000  4963.600000
25%       -1.800000       93.075000     -42.700000     1.344000  5099.100000
50%        1.100000       93.444000     -41.800000     4.857000  5191.000000
75%        1.400000       93.994000     -36.400000     4.961000  5228.100000
max        1.400000       94.767000     -26.900000     5.000000  5228.100000
```

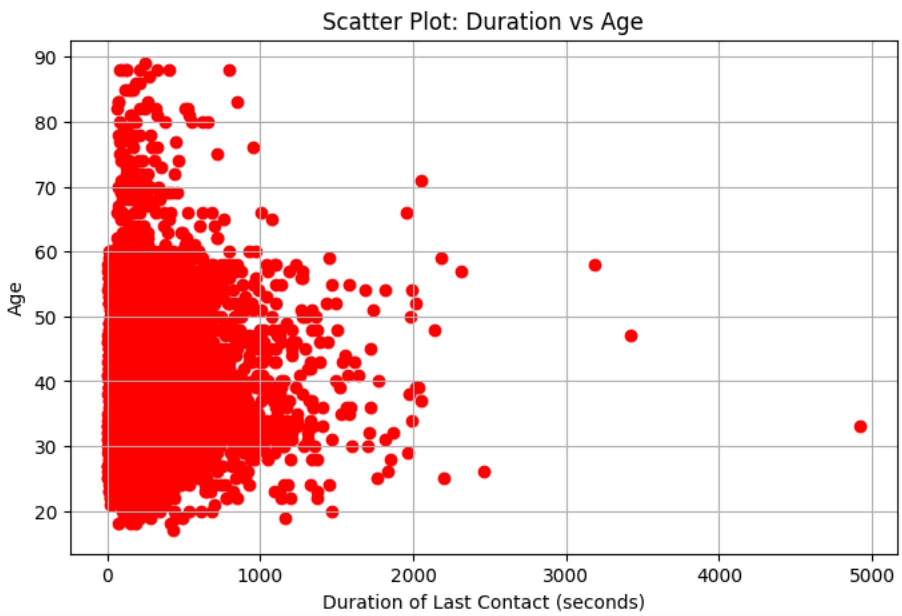
```python
# 2. Data Elaboration
# Select only numeric columns for analysis
numeric_data = data.select_dtypes(include=[np.number])

# Calculate additional statistical measures
data_summary = pd.DataFrame({
    'Mean': numeric_data.mean(),
    'Median': numeric_data.median(),
    'Variance': numeric_data.var(),
    'Standard Deviation': numeric_data.std(),
    'Skewness': numeric_data.skew(),
    'Kurtosis': numeric_data.kurt()
})
print(data_summary)
```

```
                       Mean    Median      Variance  Standard Deviation  \
age               40.116547    38.000    109.523083           10.465328
duration         256.007648   179.000  67459.016819          259.728737
campaign           2.604710     2.000      8.500842            2.915620
pdays            962.228724   999.000  35168.956664          187.533881
previous           0.174335     0.000      0.250565            0.500565
emp.var.rate       0.070147     1.100      2.479632            1.574685
cons.price.idx    93.577806    93.444      0.338884            0.582138
cons.conf.idx    -40.545320   -41.800     21.377920            4.623626
euribor3m          3.608206     4.857      3.013458            1.735931
nr.employed     5166.589790  5191.000   5252.042541           72.470977

                Skewness   Kurtosis
age             0.784895   0.821926
duration        3.556308  26.517561
campaign        5.017547  39.863569
pdays          -4.904583  22.060701
previous        3.835471  19.820252
emp.var.rate   -0.711475  -1.077152
cons.price.idx -0.198993  -0.884058
cons.conf.idx   0.343522  -0.287259
euribor3m      -0.693264  -1.428669
nr.employed    -1.033757  -0.028442
```

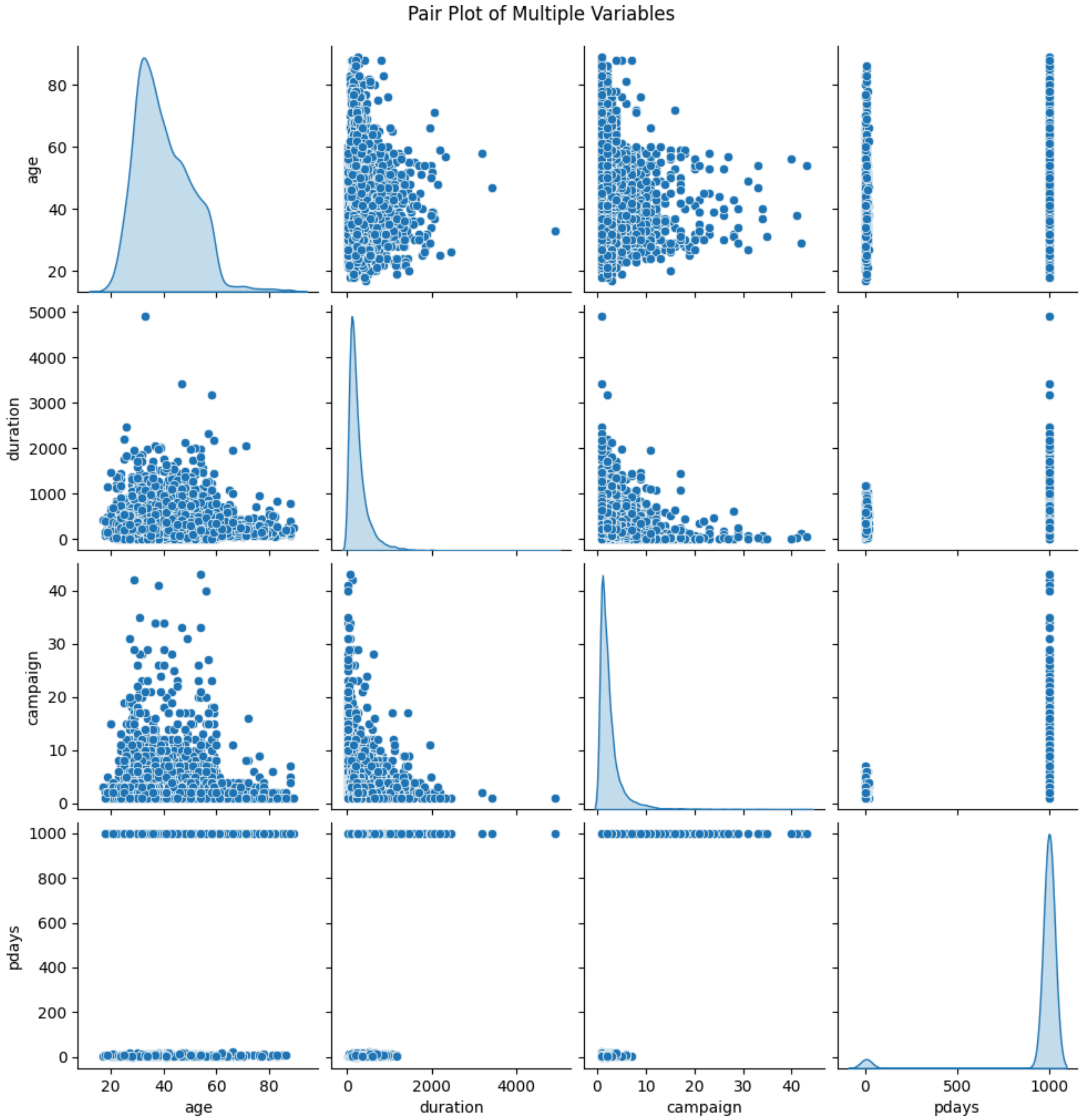
```python
# 3. 1-D Statistical Data Analysis
# Distribution plot for 'age'
plt.figure(figsize=(10, 6))
sns.histplot(data['age'], bins=20, kde=True, color='blue')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```




```
# 4. 2-D Statistical Data Analysis
# Scatter plot for 'duration' vs 'age'
plt.figure(figsize=(8, 5))
plt.scatter(data['duration'], data['age'], color='red')
plt.title('Scatter Plot: Duration vs Age')
plt.xlabel('Duration of Last Contact (seconds)')
plt.ylabel('Age')
plt.grid(True)
plt.show()
```




```
# 5. n-D Statistical Data Analysis
# Pair plot for multiple variables
sns.pairplot(data[['age', 'duration', 'campaign', 'pdays']], diag_kind='kde')
plt.suptitle('Pair Plot of Multiple Variables', y=1.02)
plt.show()
```

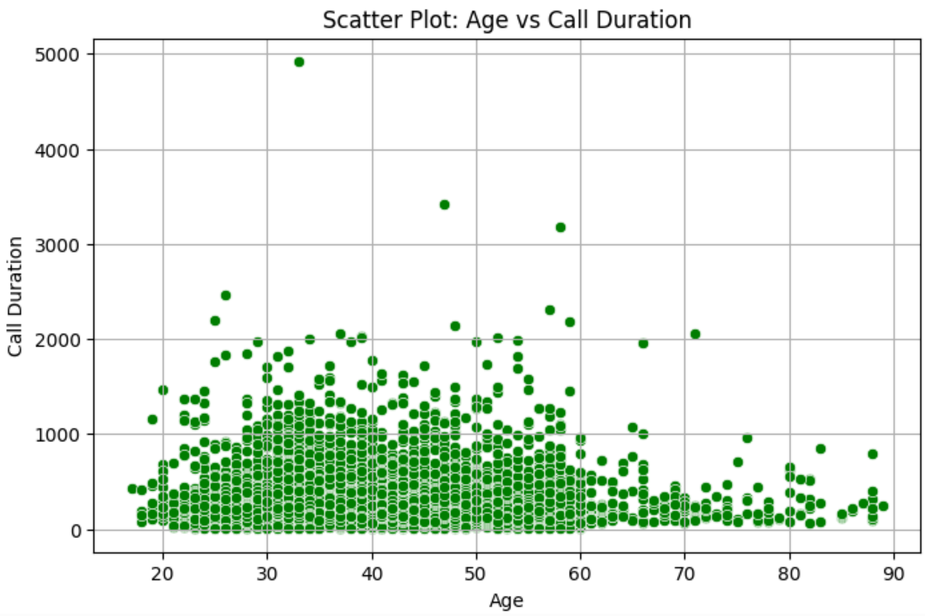



```
# 6. Contingency Tables
contingency_table = pd.crosstab(data['job'], data['marital'])
print(contingency_table)
```

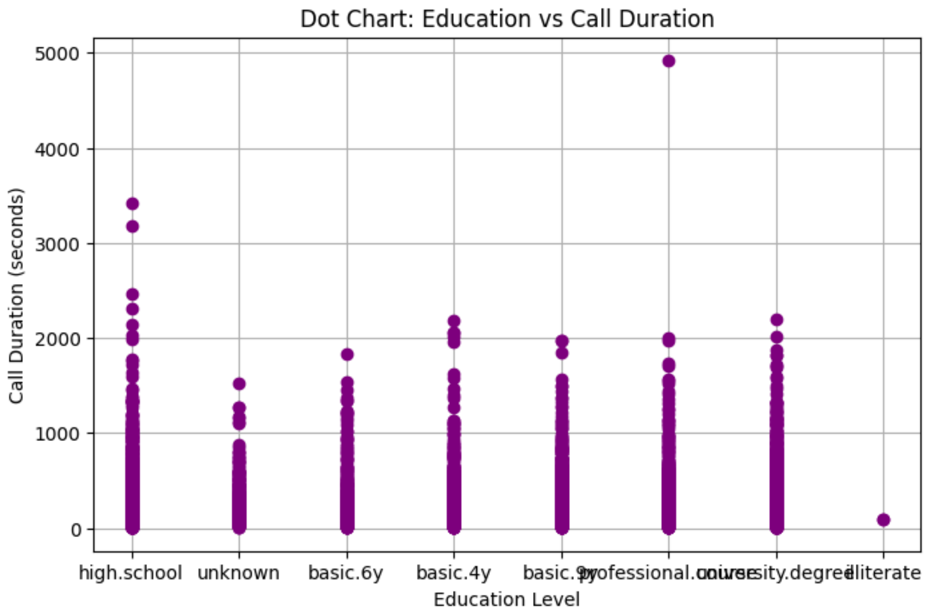
```
marital        divorced  married  single  unknown
job
admin.              273     1077     761        4
blue-collar         136     1362     363        5
entrepreneur         28      214      42        0
housemaid            39      159      24        0
management           72      472      88        1
retired              71      237      16        1
self-employed        21      158      76        0
services             95      436     232        1
student               4        9     154        1
technician          163      746     436        2
unemployed           24      129      45        0
unknown               5       42      12        1
```

```
# 7. Visualization: Scatter Plots, Dot Charts, and Bar Plots

# Scatter plot for 'age' vs 'duration'
plt.figure(figsize=(8, 5))
sns.scatterplot(x='age', y='duration', data=data, color='green')
plt.title('Scatter Plot: Age vs Call Duration')
plt.xlabel('Age')
```

```
plt.ylabel('Call Duration')
plt.grid(True)
plt.show()
```

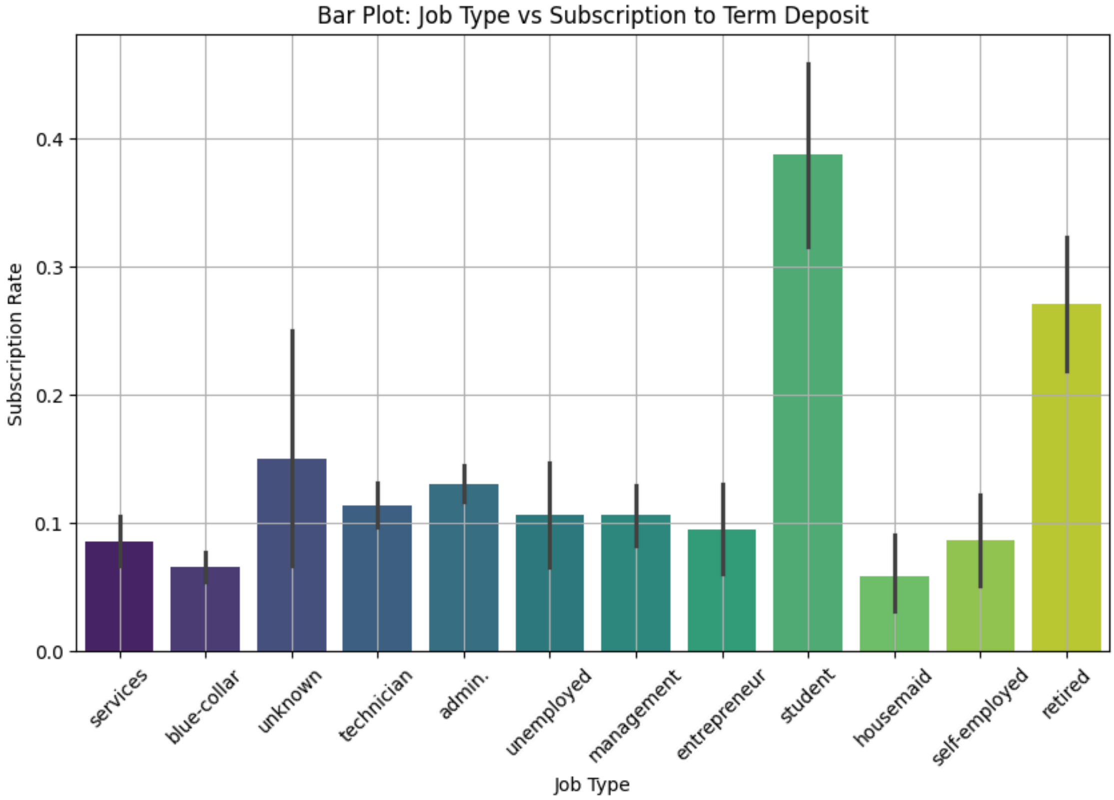



```
# Dot chart for 'education' vs 'duration'
plt.figure(figsize=(8, 5))
plt.plot(data['education'], data['duration'], 'o', color='purple')
plt.title('Dot Chart: Education vs Call Duration')
plt.xlabel('Education Level')
plt.ylabel('Call Duration (seconds)')
plt.grid(True)
plt.show()
```




```
# Bar plot for 'job' vs 'y'
plt.figure(figsize=(10, 6))
sns.barplot(x='job', y=data['y'].map({'yes': 1, 'no': 0}), data=data, palette='viridis')
plt.title('Bar Plot: Job Type vs Subscription to Term Deposit')
plt.xlabel('Job Type')
plt.ylabel('Subscription Rate')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```

```
<ipython-input-13-65630bc05b37>:3: FutureWarning:

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

  sns.barplot(x='job', y=data['y'].map({'yes': 1, 'no': 0}), data=data, palette='viridis')
```

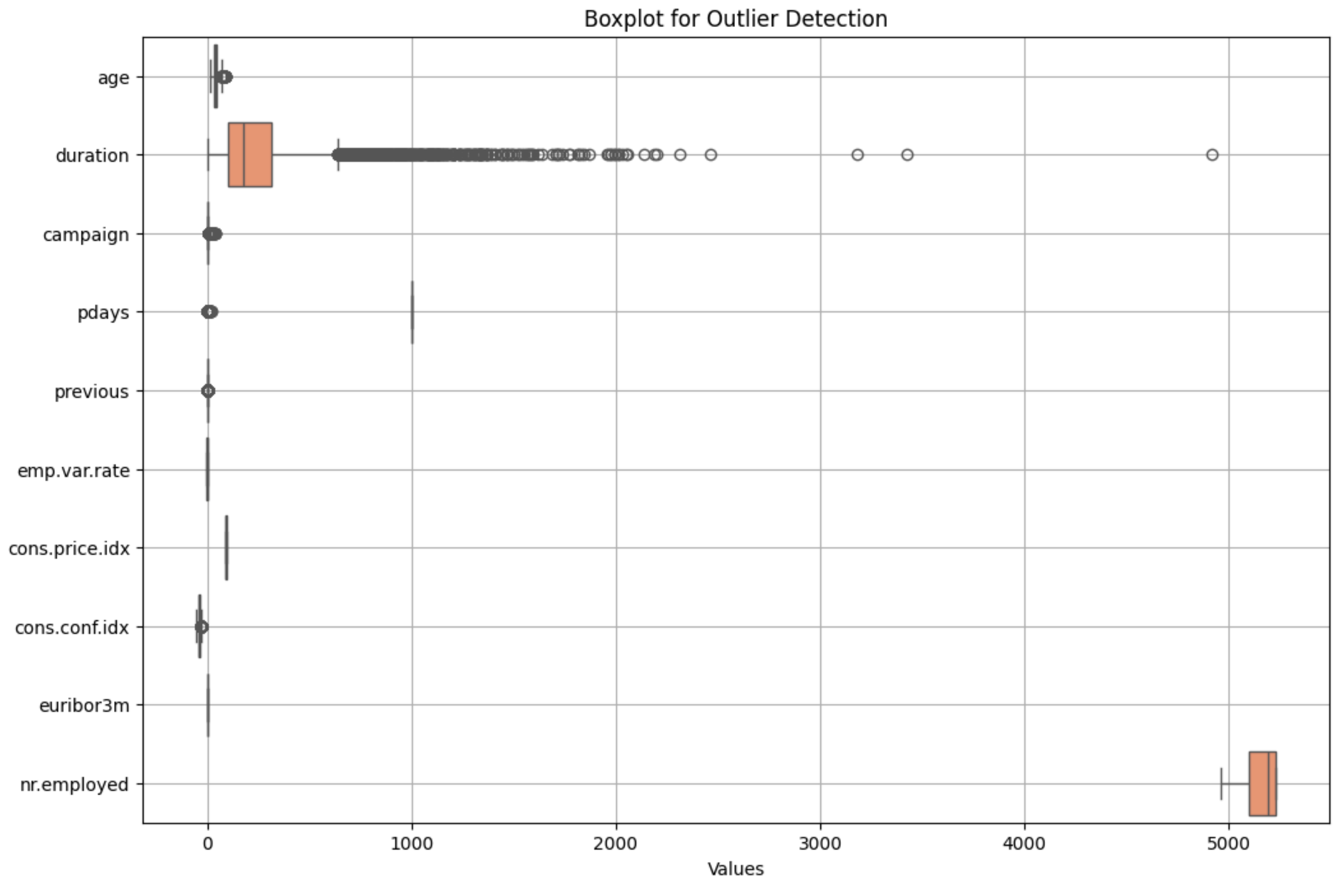



```
# 1. Expanded Statistical Summary Measures
numeric_data = data.select_dtypes(include=[np.number])

data_summary = pd.DataFrame({
    'Count': numeric_data.count(),
    'Mean': numeric_data.mean(),
    'Median': numeric_data.median(),
    'Variance': numeric_data.var(),
    'Standard Deviation': numeric_data.std(),
    'Minimum': numeric_data.min(),
    'Maximum': numeric_data.max(),
    'Skewness': numeric_data.skew(),
    'Kurtosis': numeric_data.kurt()
})
print(data_summary)
```

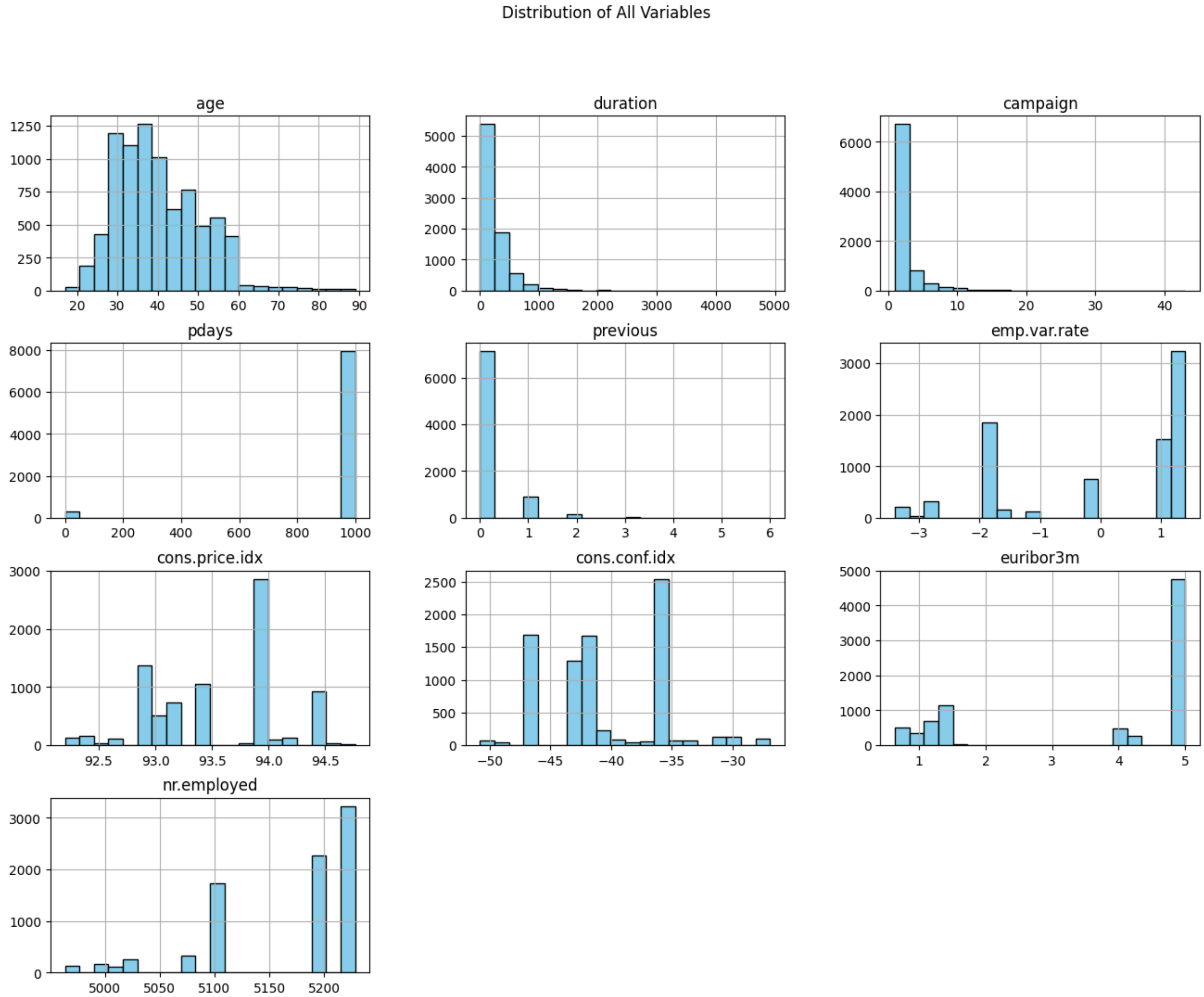
```
                Count         Mean    Median      Variance  \
age              8237    40.116547    38.000    109.523083
duration         8237   256.007648   179.000  67459.016819
campaign         8237     2.604710     2.000      8.500842
pdays            8237   962.228724   999.000  35168.956664
previous         8237     0.174335     0.000      0.250565
emp.var.rate     8237     0.070147     1.100      2.479632
cons.price.idx   8237    93.577806    93.444      0.338884
cons.conf.idx    8237   -40.545320   -41.800     21.377920
euribor3m        8237     3.608206     4.857      3.013458
nr.employed      8237  5166.589790  5191.000   5252.042541

                Standard Deviation   Minimum   Maximum  Skewness   Kurtosis
age                      10.465328    17.000    89.000  0.784895   0.821926
duration                259.728737     4.000  4918.000  3.556308  26.517561
campaign                  2.915620     1.000    43.000  5.017547  39.863569
pdays                   187.533881     0.000   999.000 -4.904583  22.060701
previous                  0.500565     0.000     6.000  3.835471  19.820252
emp.var.rate              1.574685    -3.400     1.400 -0.711475  -1.077152
cons.price.idx            0.582138    92.201    94.767 -0.198993  -0.884058
cons.conf.idx             4.623626   -50.800   -26.900  0.343522  -0.287259
euribor3m                 1.735931     0.634     5.000 -0.693264  -1.428669
nr.employed              72.470977  4963.600  5228.100 -1.033757  -0.028442
```

```
# 2. Data Elaboration: Distribution and Outliers
plt.figure(figsize=(12, 8))
sns.boxplot(data=numeric_data, orient='h', palette='Set2')
plt.title('Boxplot for Outlier Detection')
```

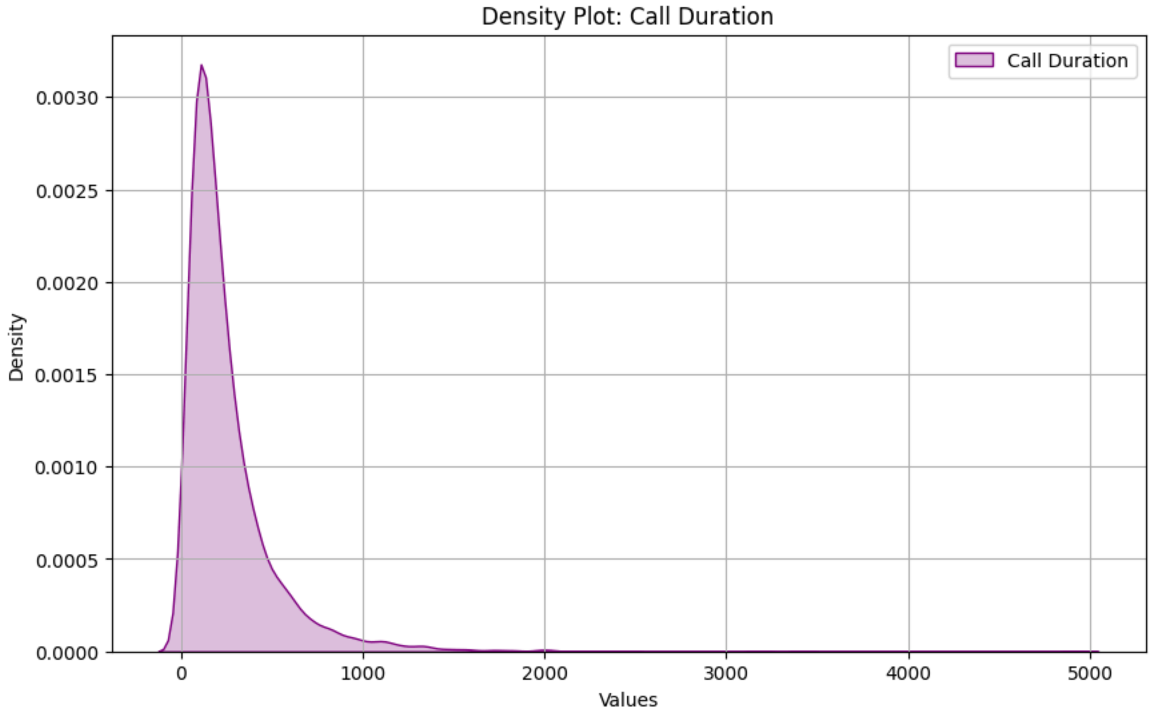
```
plt.xlabel('Values')
plt.grid(True)
plt.show()
```



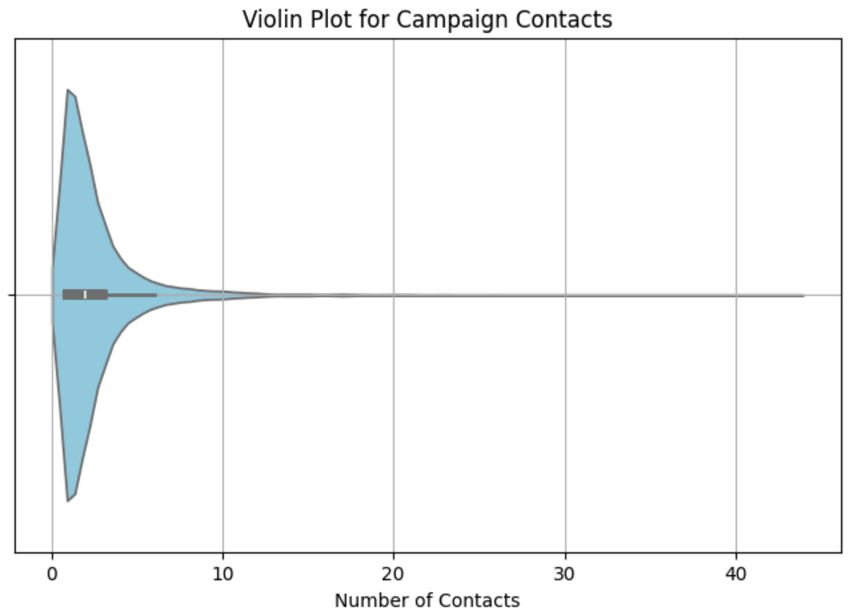

```
numeric_data.hist(bins=20, figsize=(16, 12), color='skyblue', edgecolor='black')
plt.suptitle('Distribution of All Variables')
plt.show()
```



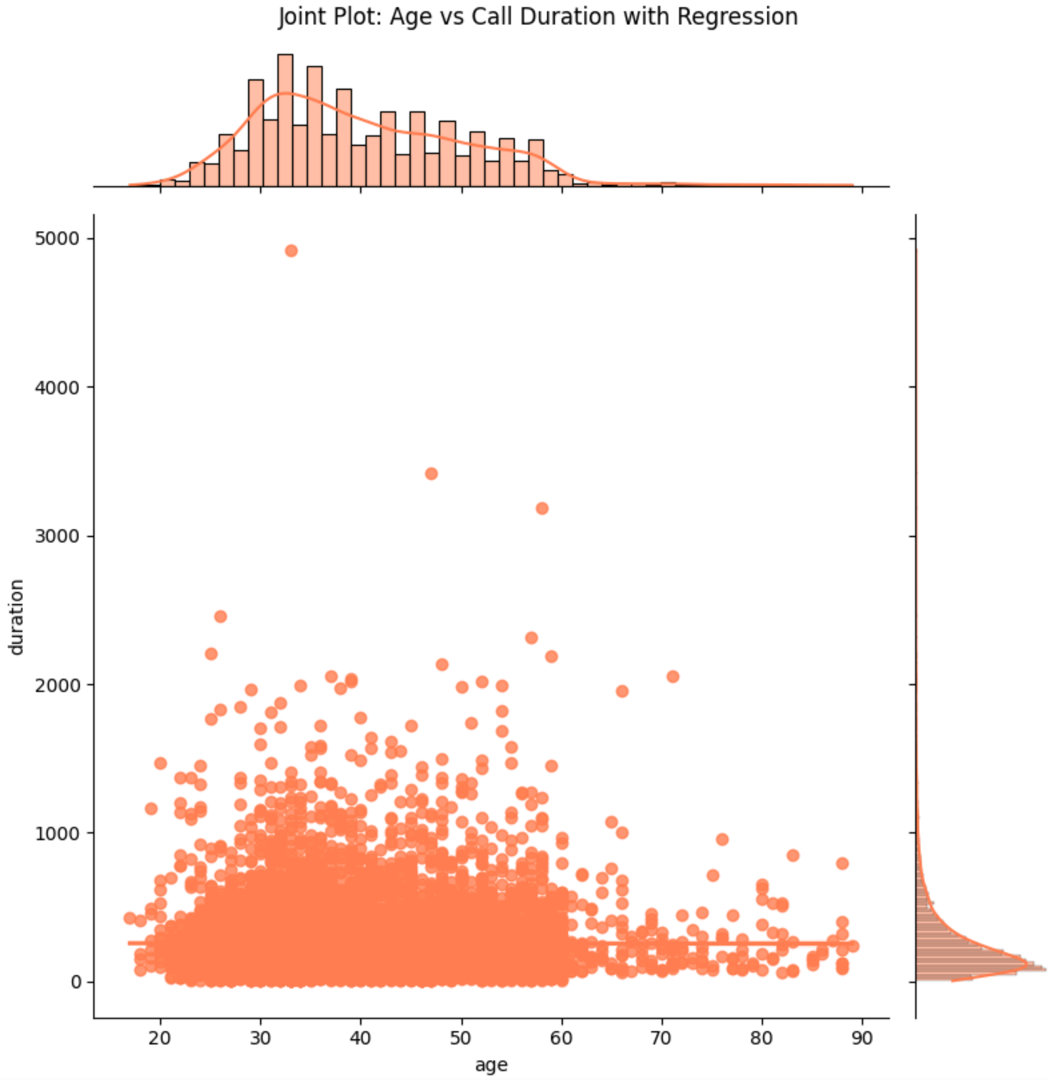

```
# 3. 1-D Statistical Data Analysis
plt.figure(figsize=(10, 6))
sns.kdeplot(data['duration'], fill=True, color='purple', label='Call Duration')
plt.title('Density Plot: Call Duration')
plt.xlabel('Values')
plt.ylabel('Density')
plt.legend()
plt.grid(True)
plt.show()
```



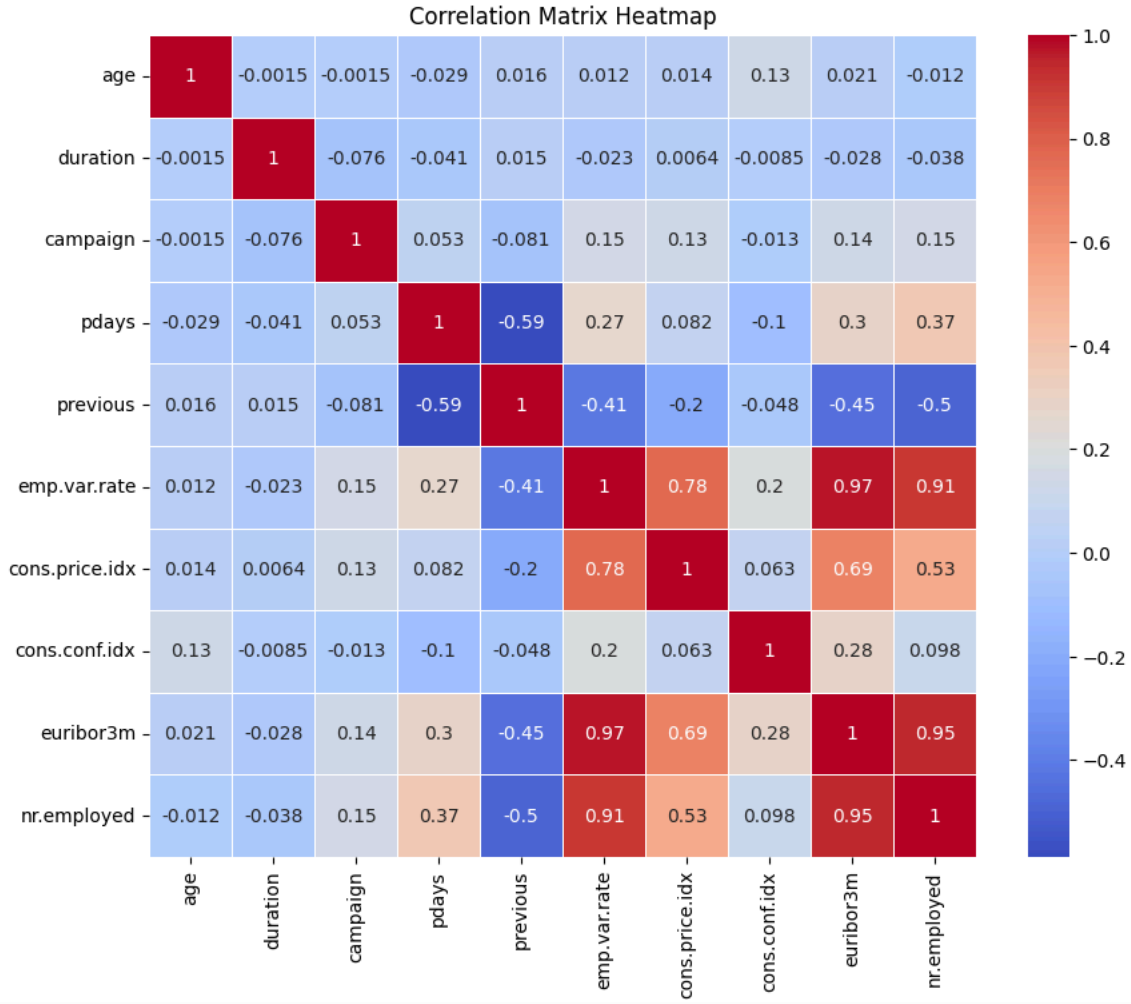

```
plt.figure(figsize=(8, 5))
sns.violinplot(x=data['campaign'], color='skyblue')
plt.title('Violin Plot for Campaign Contacts')
plt.xlabel('Number of Contacts')
plt.grid(True)
plt.show()
```




```
# 4. 2-D Statistical Data Analysis
sns.jointplot(x='age', y='duration', data=data, kind='reg', height=8, color='coral')
plt.suptitle('Joint Plot: Age vs Call Duration with Regression', y=1.02)
plt.show()
```

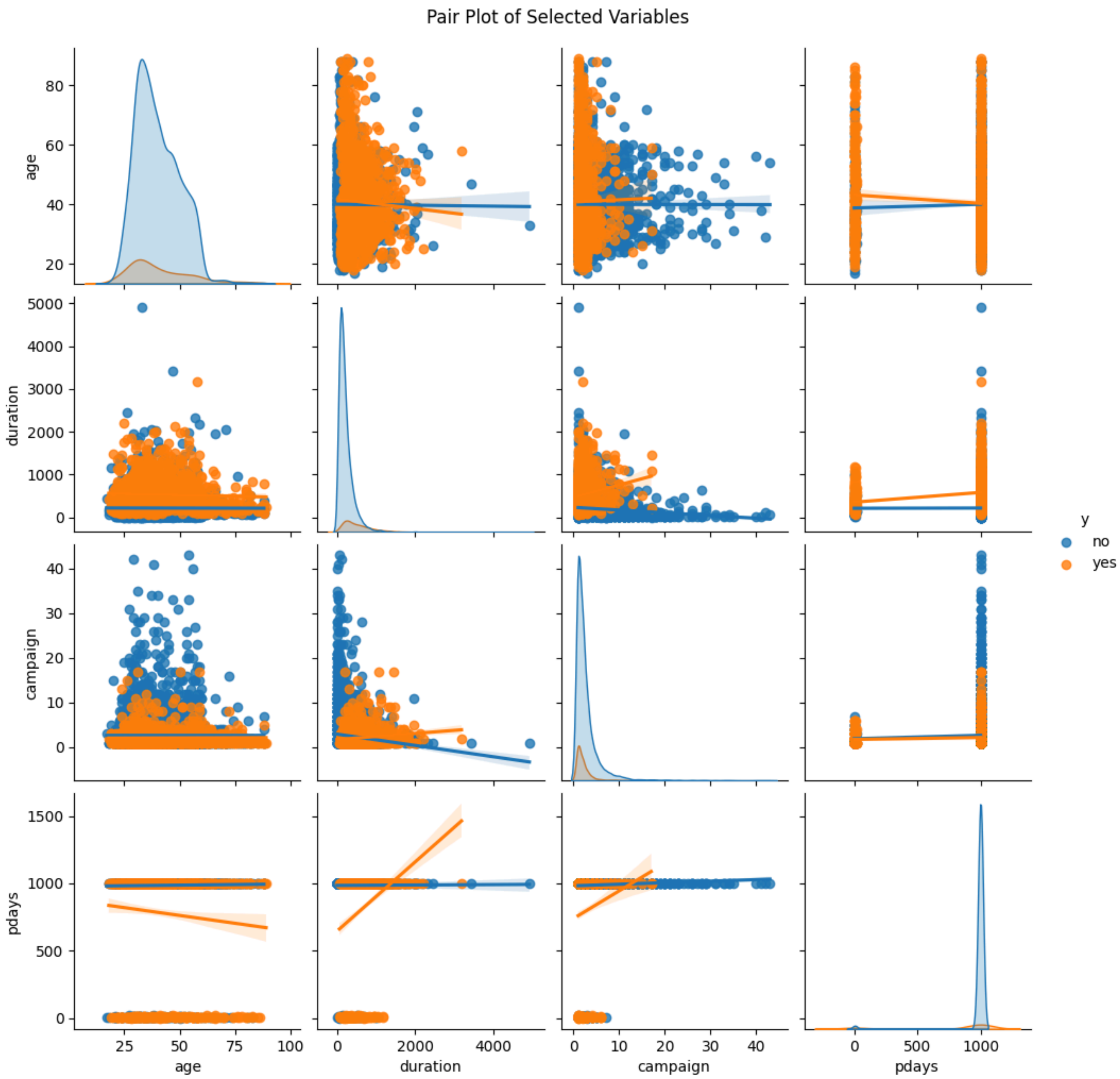



```
plt.figure(figsize=(10, 8))
sns.heatmap(numeric_data.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix Heatmap')
plt.show()
```



	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed
age	1	-0.0015	-0.0015	-0.029	0.016	0.012	0.014	0.13	0.021	-0.012
duration	-0.0015	1	-0.076	-0.041	0.015	-0.023	0.0064	-0.0085	-0.028	-0.038
campaign	-0.0015	-0.076	1	0.053	-0.081	0.15	0.13	-0.013	0.14	0.15
pdays	-0.029	-0.041	0.053	1	-0.59	0.27	0.082	-0.1	0.3	0.37
previous	0.016	0.015	-0.081	-0.59	1	-0.41	-0.2	-0.048	-0.45	-0.5
emp.var.rate	0.012	-0.023	0.15	0.27	-0.41	1	0.78	0.2	0.97	0.91
cons.price.idx	0.014	0.0064	0.13	0.082	-0.2	0.78	1	0.063	0.69	0.53
cons.conf.idx	0.13	-0.0085	-0.013	-0.1	-0.048	0.2	0.063	1	0.28	0.098
euribor3m	0.021	-0.028	0.14	0.3	-0.45	0.97	0.69	0.28	1	0.95
nr.employed	-0.012	-0.038	0.15	0.37	-0.5	0.91	0.53	0.098	0.95	1

```
# 5. n-D Statistical Data Analysis
sns.pairplot(data, vars=['age', 'duration', 'campaign', 'pdays'], hue='y', kind='reg')
plt.suptitle('Pair Plot of Selected Variables', y=1.02)
plt.show()
```



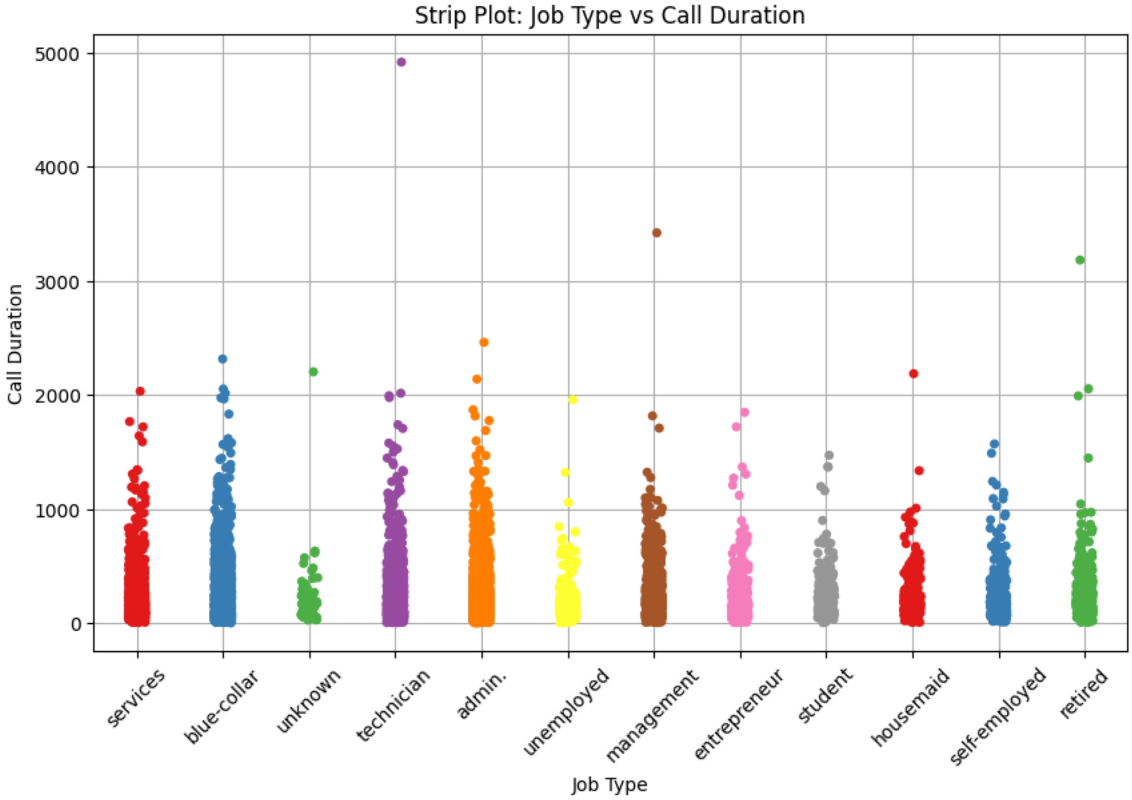

```
# 6. Advanced Contingency Tables
contingency_table_expanded = pd.crosstab(data['job'], data['marital'], margins=True, margins_name='Total')
print(contingency_table_expanded)
```

```
marital        divorced  married  single  unknown  Total
job
admin.              273     1077     761        4   2115
blue-collar         136     1362     363        5   1866
entrepreneur         28      214      42        0    284
housemaid            39      159      24        0    222
management           72      472      88        1    633
retired              71      237      16        1    325
self-employed        21      158      76        0    255
services             95      436     232        1    764
student               4        9     154        1    168
technician          163      746     436        2   1347
unemployed           24      129      45        0    198
unknown               5       42      12        1     60
Total               931     5041    2249       16   8237
```

```
# 7. Advanced Visualizations
plt.figure(figsize=(10, 6))
sns.stripplot(x='job', y='duration', data=data, jitter=True, palette='Set1')
plt.title('Strip Plot: Job Type vs Call Duration')
plt.xlabel('Job Type')
plt.ylabel('Call Duration')
plt.xticks(rotation=45)
```

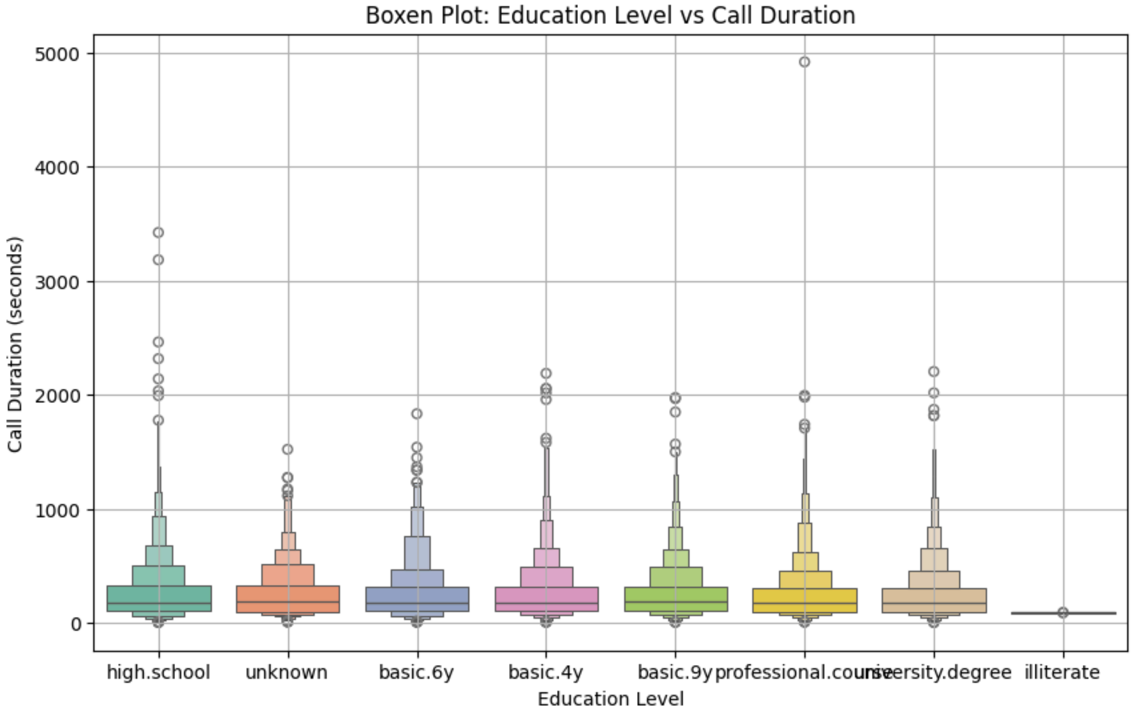
```
plt.grid(True)
plt.show()
```

```
<ipython-input-23-6ed60778a31f>:3: FutureWarning:

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

  sns.stripplot(x='job', y='duration', data=data, jitter=True, palette='Set1')
```

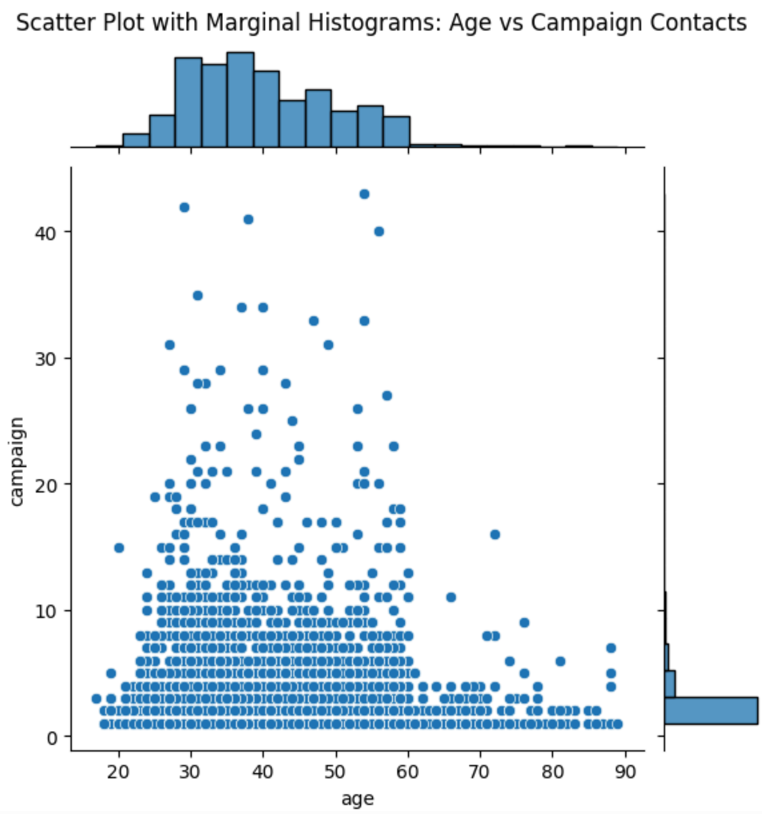



```
plt.figure(figsize=(10, 6))
sns.boxenplot(x='education', y='duration', data=data, palette='Set2')
plt.title('Boxen Plot: Education Level vs Call Duration')
plt.xlabel('Education Level')
plt.ylabel('Call Duration (seconds)')
plt.grid(True)
plt.show()
```

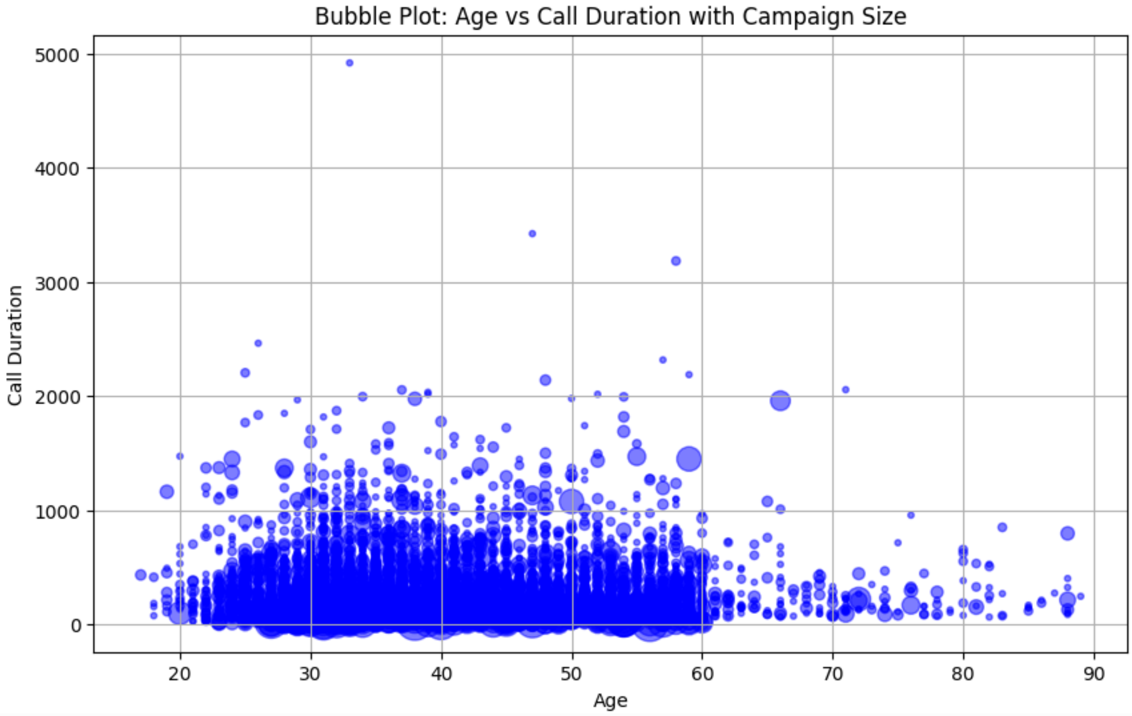
```
<ipython-input-24-be46688c9013>:2: FutureWarning:

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

  sns.boxenplot(x='education', y='duration', data=data, palette='Set2')
```




```
sns.jointplot(x='age', y='campaign', data=data, kind='scatter', marginal_kws=dict(bins=20, fill=True))
plt.suptitle('Scatter Plot with Marginal Histograms: Age vs Campaign Contacts', y=1.02)
plt.show()
```

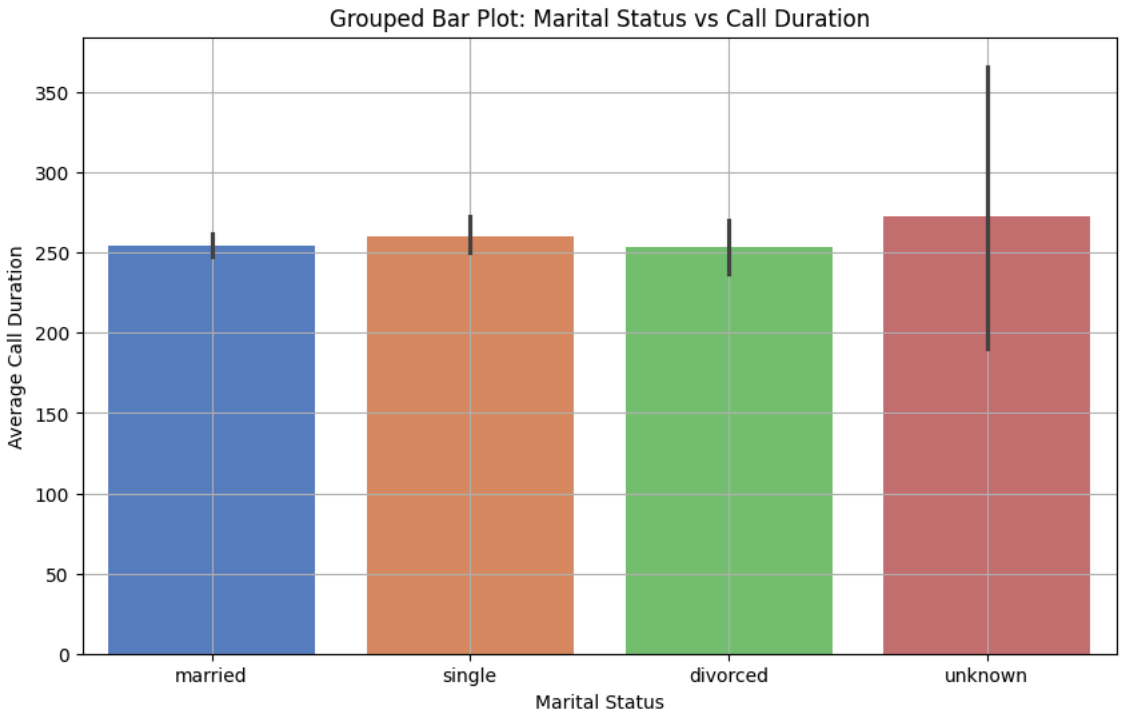



```
plt.figure(figsize=(10, 6))
plt.scatter(data['age'], data['duration'], s=data['campaign']*10, alpha=0.5, c='blue')
plt.title('Bubble Plot: Age vs Call Duration with Campaign Size')
plt.xlabel('Age')
plt.ylabel('Call Duration')
plt.grid(True)
plt.show()
```

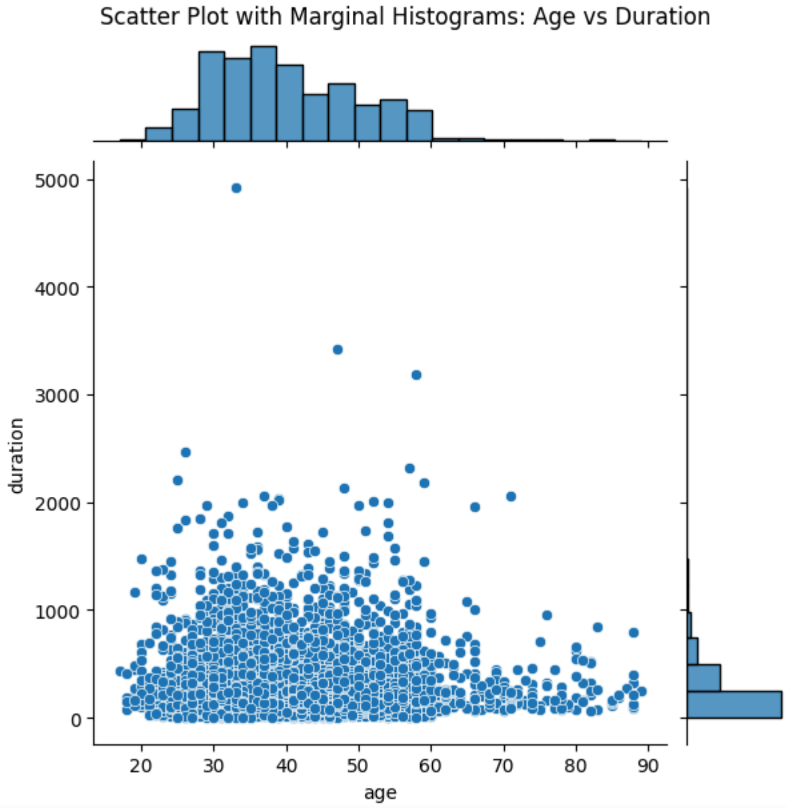



```
plt.figure(figsize=(10, 6))
sns.barplot(x='marital', y='duration', data=data, palette='muted')
plt.title('Grouped Bar Plot: Marital Status vs Call Duration')
plt.xlabel('Marital Status')
plt.ylabel('Average Call Duration')
plt.grid(True)
plt.show()
```

```
<ipython-input-27-416a4941681c>:2: FutureWarning:

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

  sns.barplot(x='marital', y='duration', data=data, palette='muted')
```

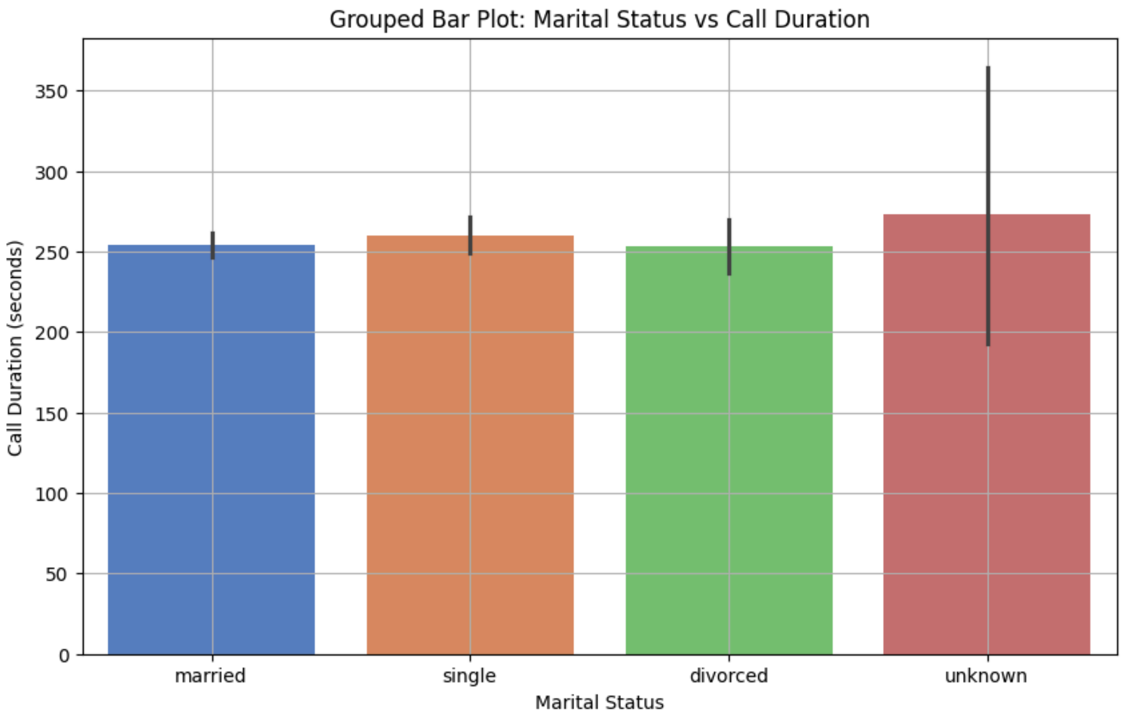



```
# 1. Advanced Scatter Plots
sns.jointplot(x='age', y='duration', data=data, kind='scatter', marginal_kws=dict(bins=20, fill=True))
plt.suptitle('Scatter Plot with Marginal Histograms: Age vs Duration', y=1.02)
plt.show()
```



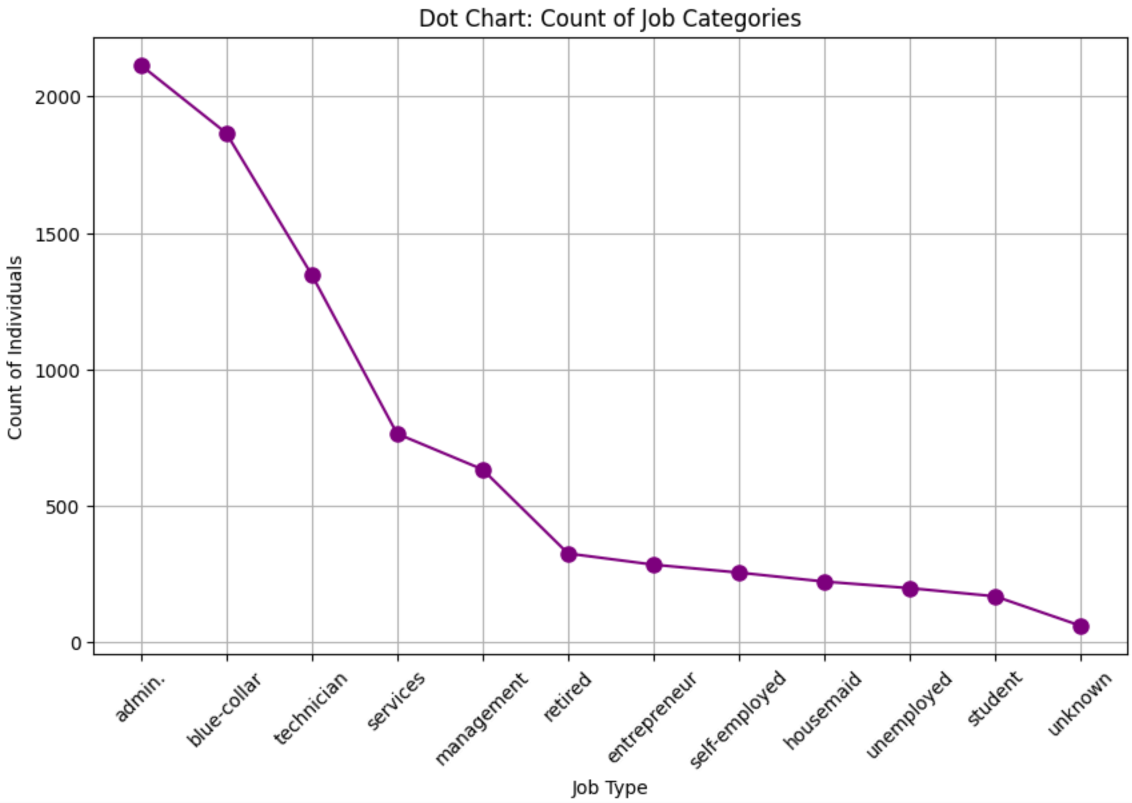

```
# 2. Advanced Bar Plots and Dot Charts
plt.figure(figsize=(10, 6))
sns.barplot(x='marital', y='duration', data=data, palette='muted')
plt.title('Grouped Bar Plot: Marital Status vs Call Duration')
plt.xlabel('Marital Status')
plt.ylabel('Call Duration (seconds)')
plt.grid(True)
plt.show()
```

```
<ipython-input-29-0e28cfcf3342>:3: FutureWarning:

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

  sns.barplot(x='marital', y='duration', data=data, palette='muted')
```

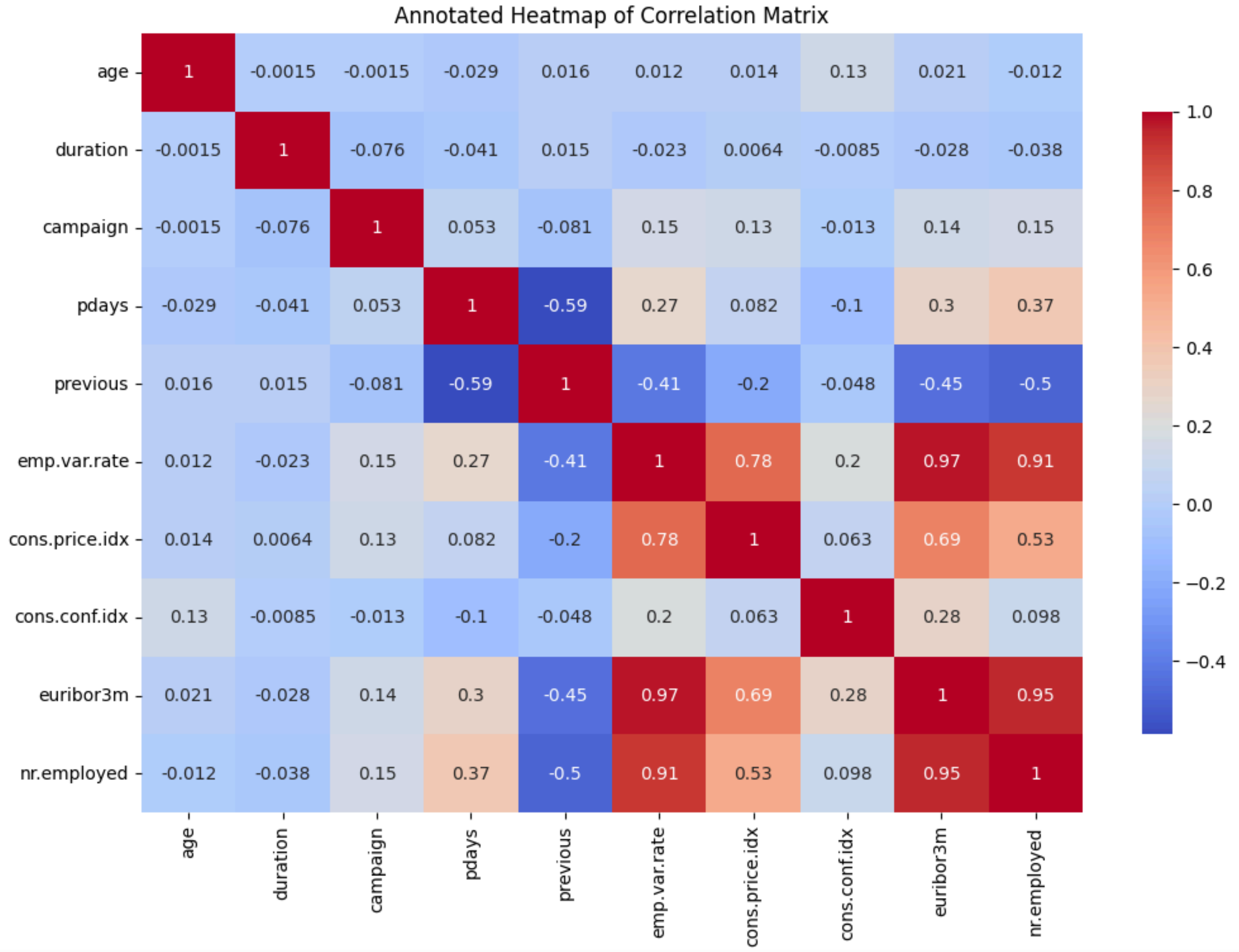



```
# Dot Chart: Count of Job Categories
plt.figure(figsize=(10, 6))
job_counts = data['job'].value_counts()
```

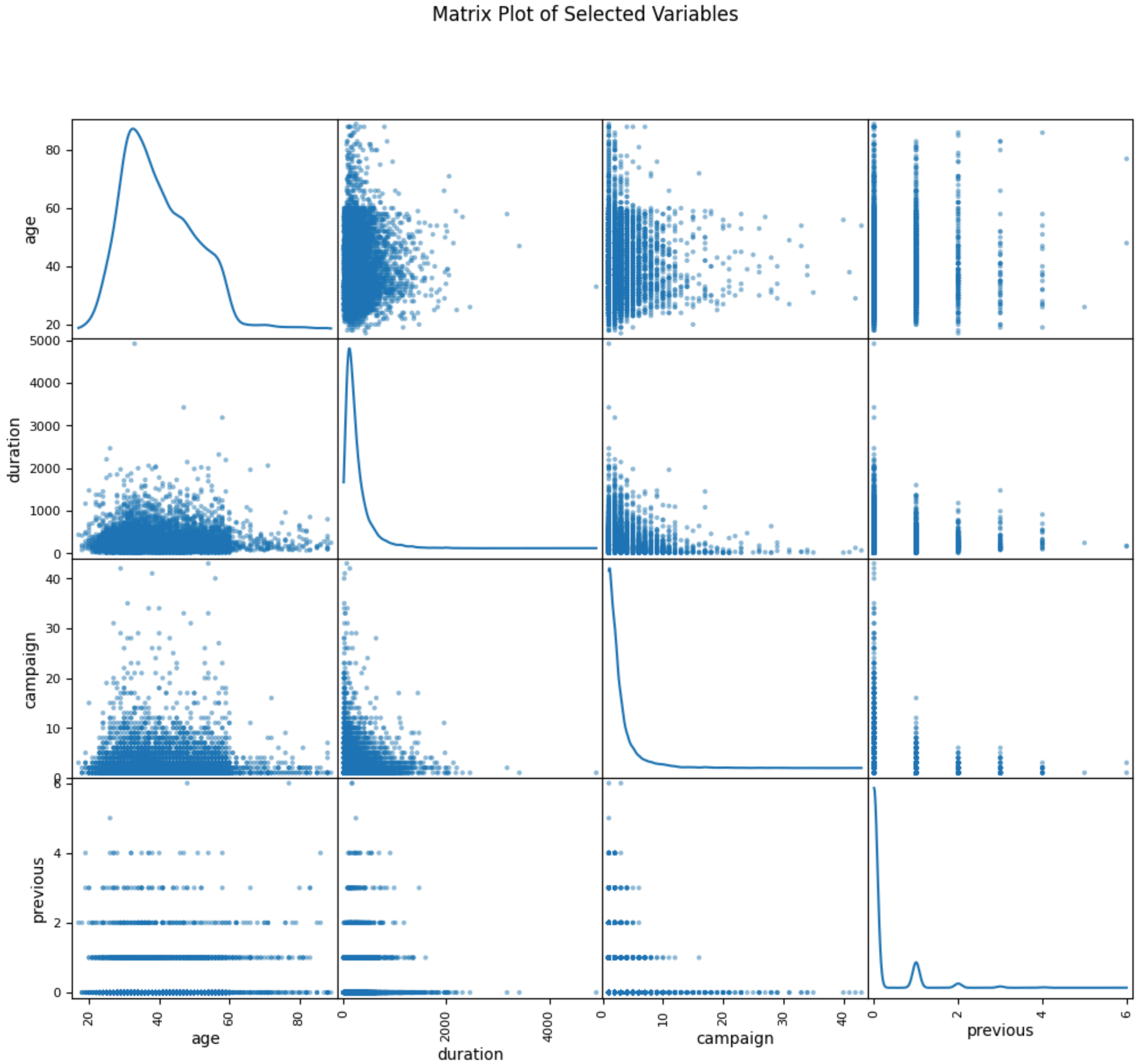
```
plt.plot(job_counts.index, job_counts.values, 'o-', color='purple', markersize=8)
plt.title('Dot Chart: Count of Job Categories')
plt.xlabel('Job Type')
plt.ylabel('Count of Individuals')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```



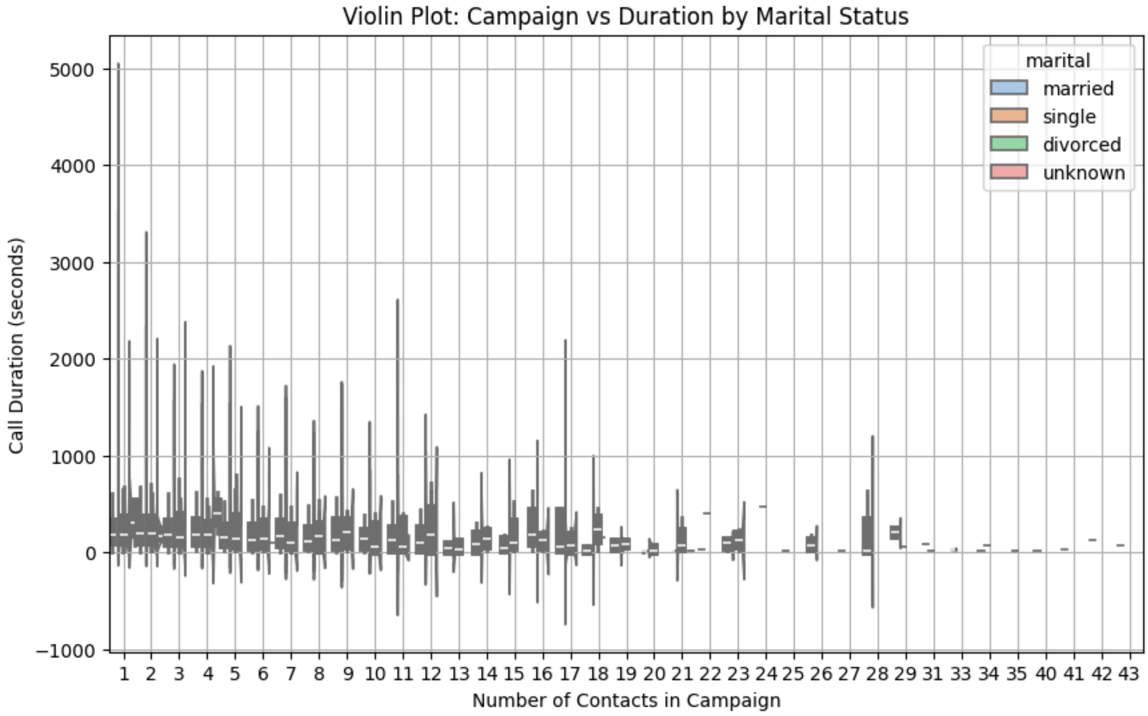

```
# 3. Advanced Heatmaps and Matrix Plots
numeric_data = data.select_dtypes(include=[np.number])
plt.figure(figsize=(12, 8))
sns.heatmap(numeric_data.corr(), annot=True, cmap='coolwarm', cbar_kws={"shrink": 0.8})
plt.title('Annotated Heatmap of Correlation Matrix')
plt.show()
```



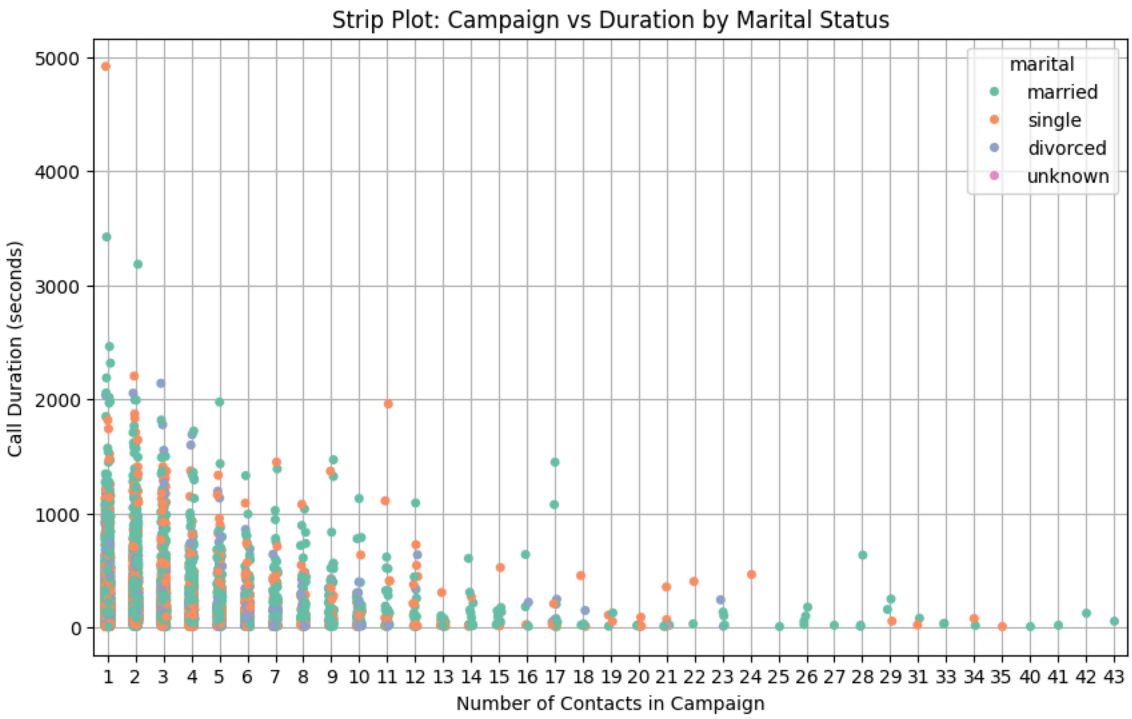

```
# Matrix plot for selected variables
pd.plotting.scatter_matrix(data[['age', 'duration', 'campaign', 'previous']], figsize=(12, 10), diagonal='kde')
plt.suptitle('Matrix Plot of Selected Variables')
plt.show()
```




```
# 4. Violin and Strip Plots
plt.figure(figsize=(10, 6))
sns.violinplot(x='campaign', y='duration', hue='marital', data=data, split=True, palette='pastel')
plt.title('Violin Plot: Campaign vs Duration by Marital Status')
plt.xlabel('Number of Contacts in Campaign')
plt.ylabel('Call Duration (seconds)')
plt.grid(True)
plt.show()
```




```
plt.figure(figsize=(10, 6))
sns.stripplot(x='campaign', y='duration', data=data, jitter=True, hue='marital', palette='Set2')
plt.title('Strip Plot: Campaign vs Duration by Marital Status')
plt.xlabel('Number of Contacts in Campaign')
plt.ylabel('Call Duration (seconds)')
plt.grid(True)
plt.show()
```




```
# Label encoding for categorical columns
encoded_data = data.copy()
encoded_data['marital'] = encoded_data['marital'].map({'married': 1, 'single': 0, 'divorced': 2})
encoded_data['housing'] = encoded_data['housing'].map({'yes': 1, 'no': 0})
print(encoded_data.head())
```

```
   age          job  marital    education  default  housing loan    contact  \
0   56     services      1.0  high.school       no      0.0  yes  telephone
1   41  blue-collar      1.0      unknown  unknown      0.0   no  telephone
2   25     services      0.0  high.school       no      1.0   no  telephone
3   35  blue-collar      1.0     basic.6y       no      1.0   no  telephone
4   46  blue-collar      1.0     basic.6y  unknown      1.0  yes  telephone

  month day_of_week  ...  campaign  pdays  previous     poutcome emp.var.rate  \
```

```
0   may         mon  ...        1    999         0  nonexistent          1.1
1   may         mon  ...        1    999         0  nonexistent          1.1
2   may         mon  ...        1    999         0  nonexistent          1.1
3   may         mon  ...        1    999         0  nonexistent          1.1
4   may         mon  ...        1    999         0  nonexistent          1.1

   cons.price.idx  cons.conf.idx  euribor3m  nr.employed   y
0          93.994          -36.4      4.857       5191.0  no
1          93.994          -36.4      4.857       5191.0  no
2          93.994          -36.4      4.857       5191.0  no
3          93.994          -36.4      4.857       5191.0  no
4          93.994          -36.4      4.857       5191.0  no

[5 rows x 21 columns]
```

```
# Selecting only numeric columns for statistical analysis
numeric_data = data.select_dtypes(include=[np.number])

# Displaying the first few rows of numeric data
print(numeric_data.head())
```

```
   age  duration  campaign  pdays  previous  emp.var.rate  cons.price.idx  \
0   56       307         1    999         0           1.1          93.994
1   41       217         1    999         0           1.1          93.994
2   25       222         1    999         0           1.1          93.994
3   35       312         1    999         0           1.1          93.994
4   46       440         1    999         0           1.1          93.994

   cons.conf.idx  euribor3m  nr.employed
0          -36.4      4.857       5191.0
1          -36.4      4.857       5191.0
2          -36.4      4.857       5191.0
3          -36.4      4.857       5191.0
4          -36.4      4.857       5191.0
```

```
# Calculating additional statistical measures for numeric columns
```