code

November 17, 2024

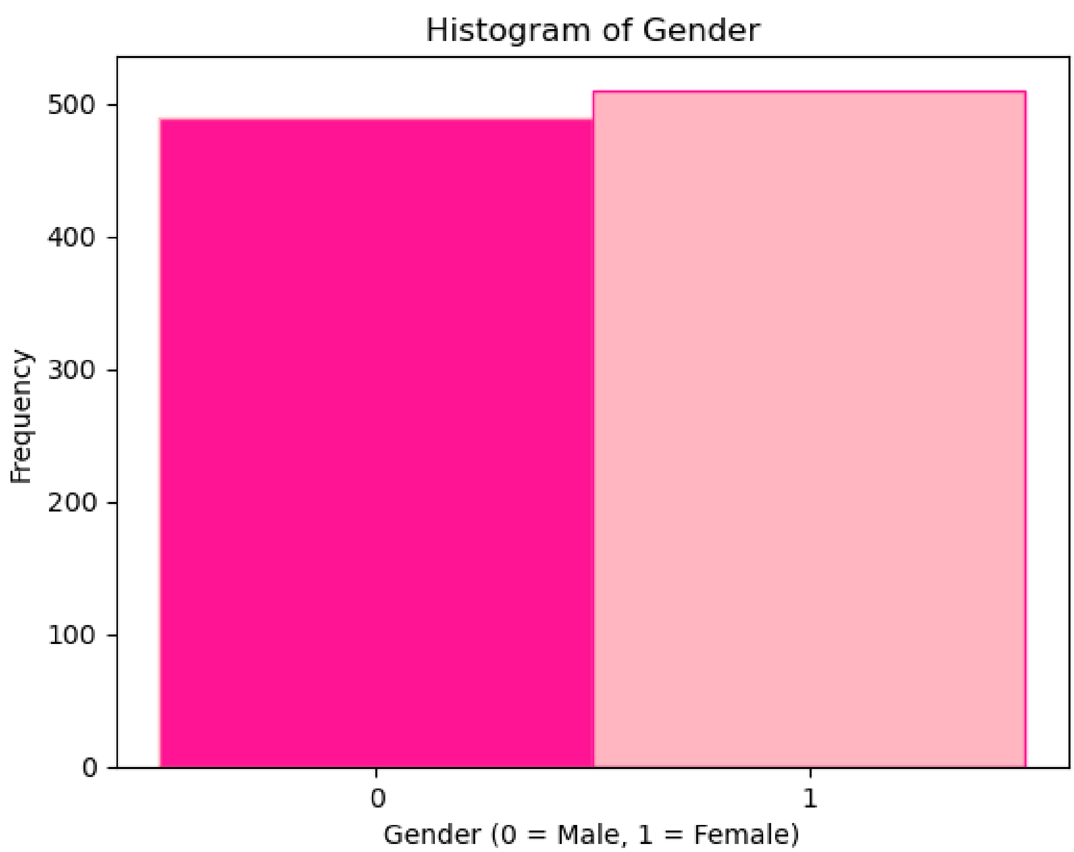
[112]:
```
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import skew, kurtosis
import thinkstats2
import thinkplot
import seaborn as sns
import numpy as np
from scipy.stats import norm
from scipy.stats import kstest
from scipy.stats import pearsonr
from thinkstats2 import HypothesisTest
import statsmodels.api as sm

file_path = 'Online_Dating_Behavior_Dataset.csv'

dating_data = pd.read_csv(file_path)

print(dating_data.head())
```

```
   Gender  PurchasedVIP  Income  Children  Age  Attractiveness  Matches
0       0             1   51777         3   47               5       70
1       1             0   36646         0   42               7      130
2       0             0   53801         1   25               5        0
3       0             0   56105         0   35               8        0
4       0             0   55597         1   36               6        0
```

[114]:
```
gender = 'Gender'
vip = 'PurchasedVIP'
income = 'Income'
children = 'Children'
age = 'Age'
attract = 'Attractiveness'
matches = 'Matches'
```

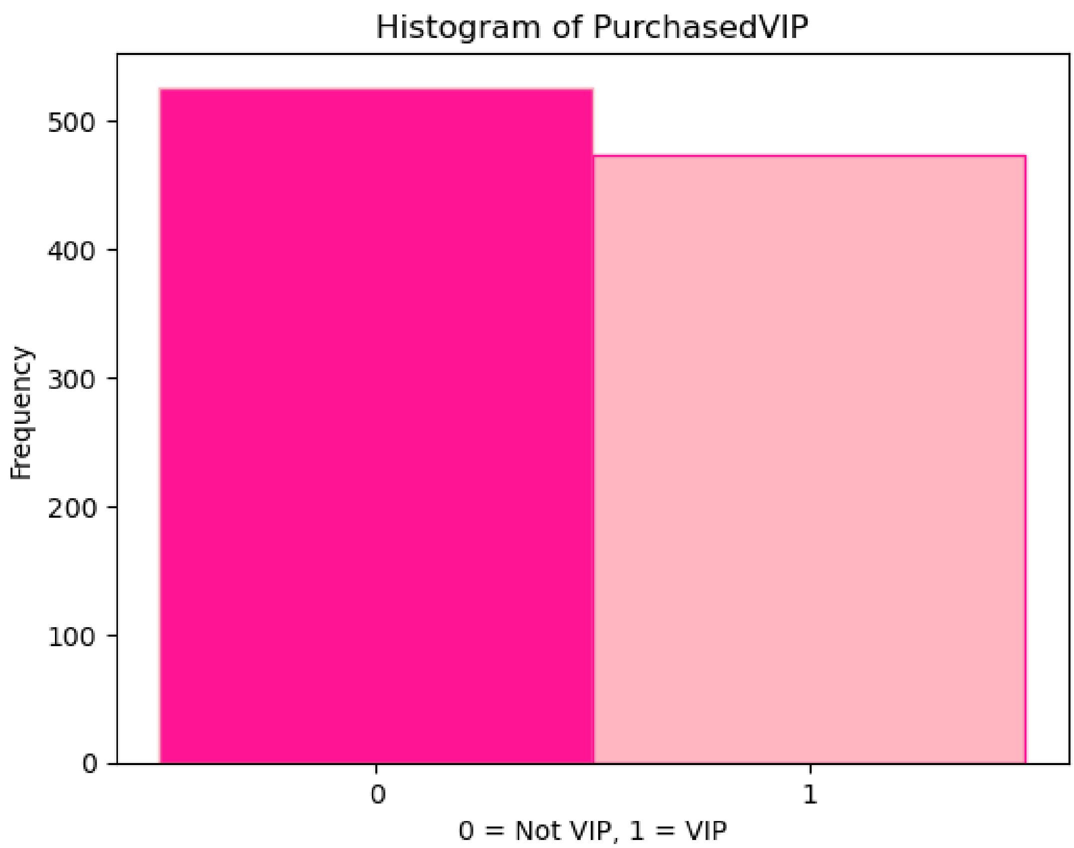
[132]:
```
# Separate the data by gender
males = dating_data[dating_data[gender] == 0]
females = dating_data[dating_data[gender] == 1]
```

```
# Plot the histograms for males and females with different colors
plt.hist(males[gender], bins=[-0.5, 0.5, 1.5], color='deeppink',␣
 ↪edgecolor='lightpink')
plt.hist(females[gender], bins=[-0.5, 0.5, 1.5], color='lightpink',␣
 ↪edgecolor='deeppink')

# Add titles and labels
plt.title('Histogram of Gender')
plt.xlabel('Gender (0 = Male, 1 = Female)')
plt.ylabel('Frequency')
plt.xticks([0, 1])
plt.show()
```

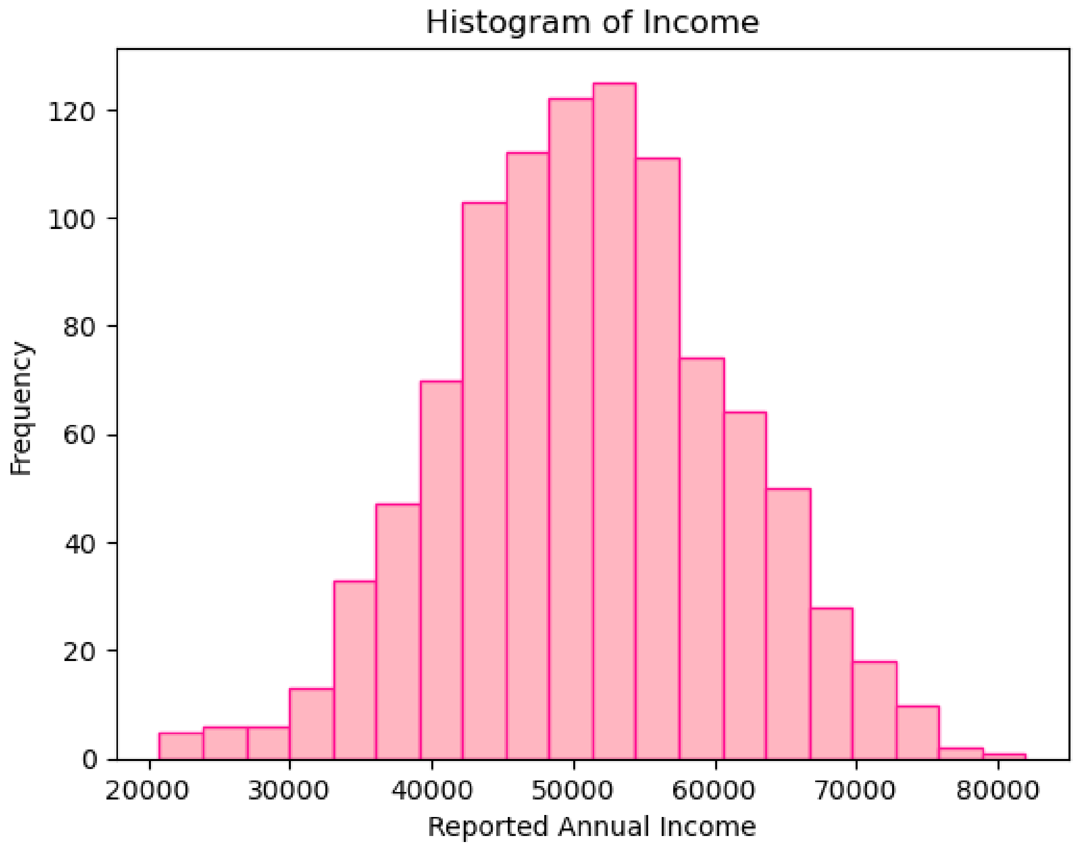


[134]:
```
not_vip = dating_data[dating_data[vip] == 0]
vip_users = dating_data[dating_data[vip] == 1]

plt.hist(not_vip[vip], bins=[-0.5, 0.5, 1.5], color='deeppink',␣
 ↪edgecolor='lightpink')
```

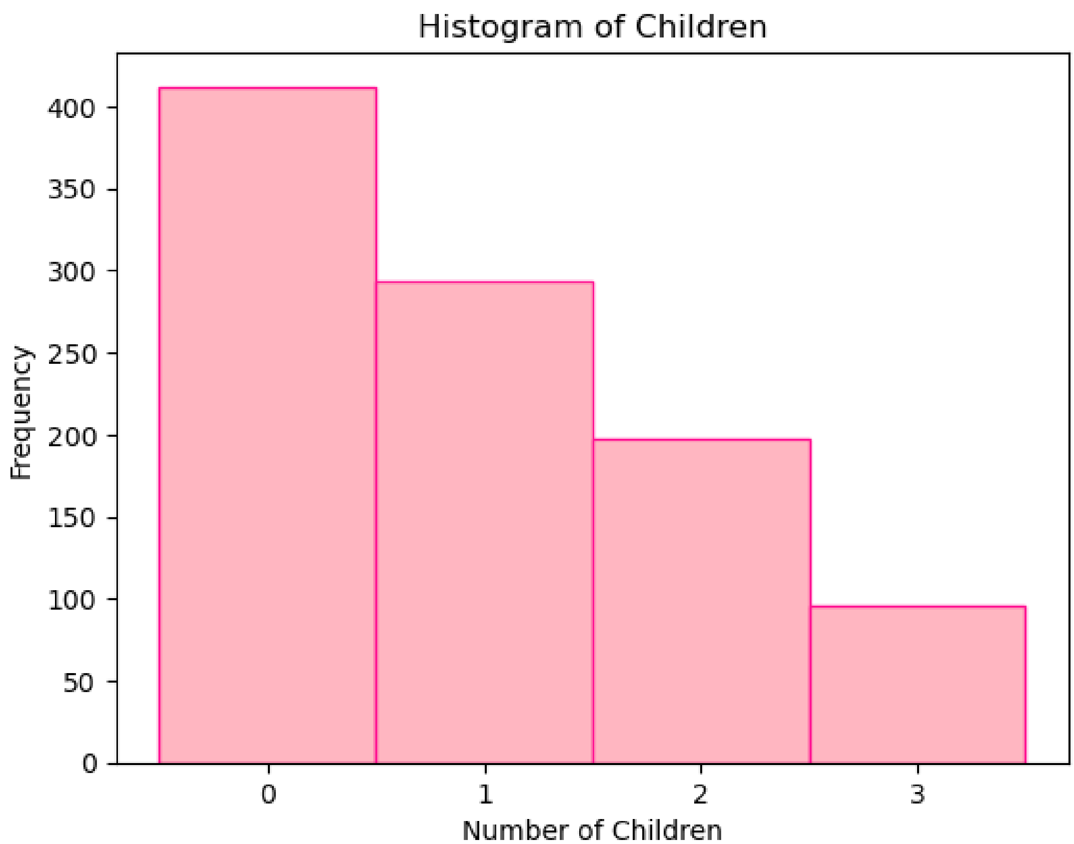
```
plt.hist(vip_users[vip], bins=[-0.5, 0.5, 1.5], color='lightpink',␣
↪edgecolor='deeppink')

plt.title('Histogram of ' + vip)
plt.xlabel('0 = Not VIP, 1 = VIP')
plt.ylabel('Frequency')
plt.xticks([0, 1])
plt.show()
```



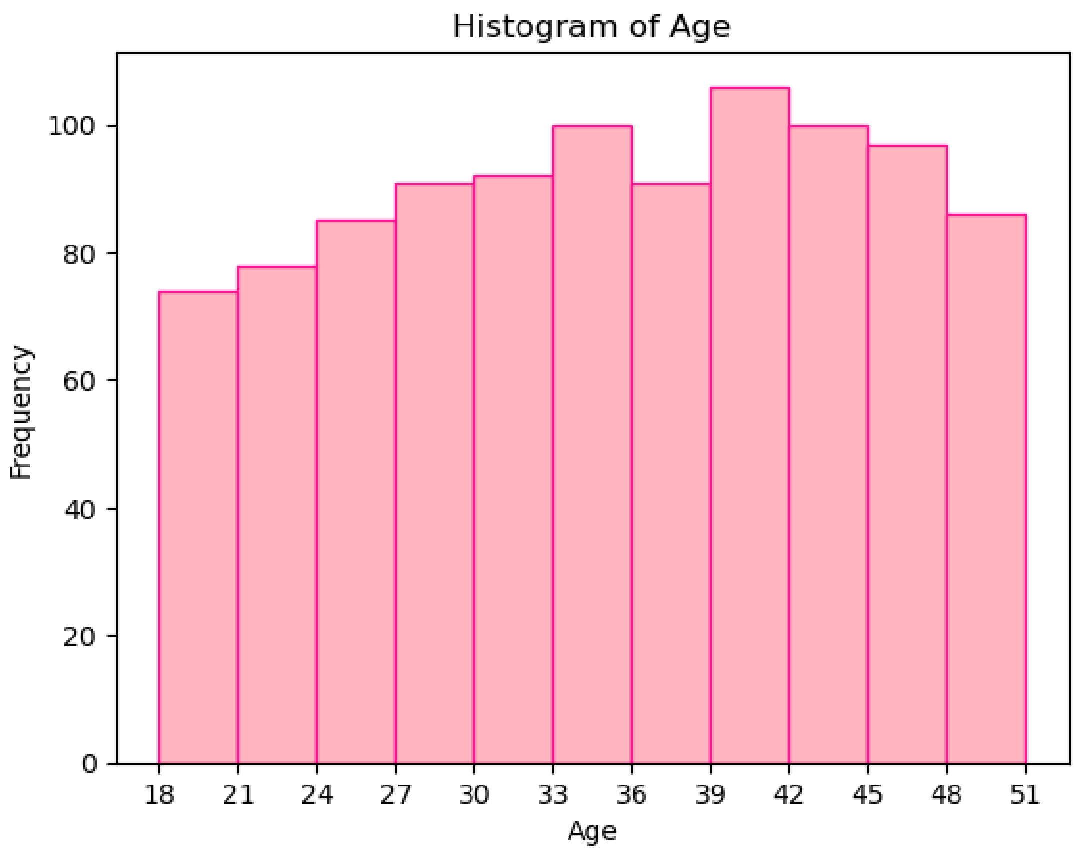
[140]:
```
plt.hist(dating_data[income], bins=20, color='lightpink', edgecolor='deeppink')
plt.title('Histogram of ' + income)
plt.xlabel('Reported Annual Income')
plt.ylabel('Frequency')
plt.show()
```



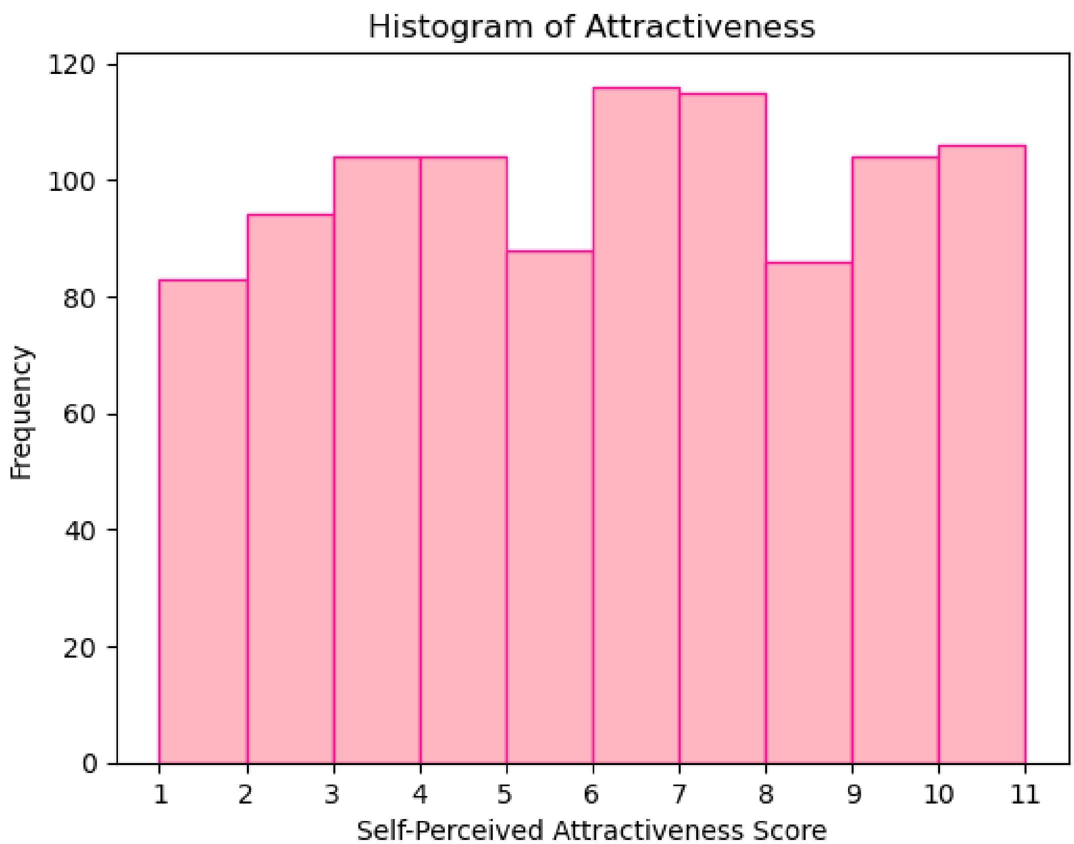
[144]:
```
plt.hist(dating_data[children], bins=[-0.5, 0.5, 1.5, 2.5, 3.5],␣
↪color='lightpink', edgecolor='deeppink')
plt.title('Histogram of ' + children)
plt.xlabel('Number of Children')
plt.ylabel('Frequency')
plt.xticks([0, 1, 2, 3])
plt.show()
```



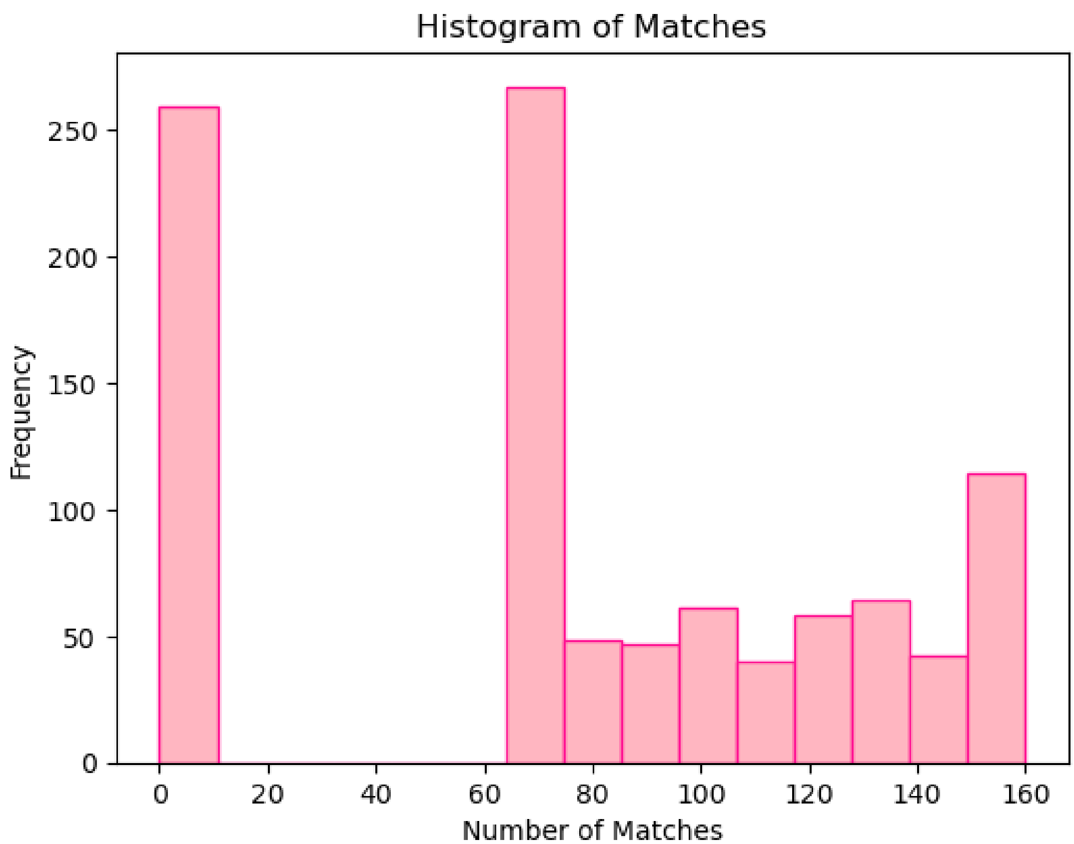
[146]:
```
plt.hist(dating_data[age], bins=[18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48,␣
 ↪51], color='lightpink', edgecolor='deeppink')
plt.title('Histogram of ' + age)
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.xticks([18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51])
plt.show()
```




```
[150]: plt.hist(dating_data[attract], bins=range(1, 12), color='lightpink',
        edgecolor='deeppink')
       plt.title('Histogram of ' + attract)
       plt.xlabel('Self-Perceived Attractiveness Score')
       plt.ylabel('Frequency')
       plt.xticks(range(1, 12))
       plt.show()
```



[154]:
```
plt.hist(dating_data[matches], bins=15, color='lightpink', edgecolor='deeppink')
plt.title('Histogram of ' + matches)
plt.xlabel('Number of Matches')
plt.ylabel('Frequency')
plt.show()
```



[123]:

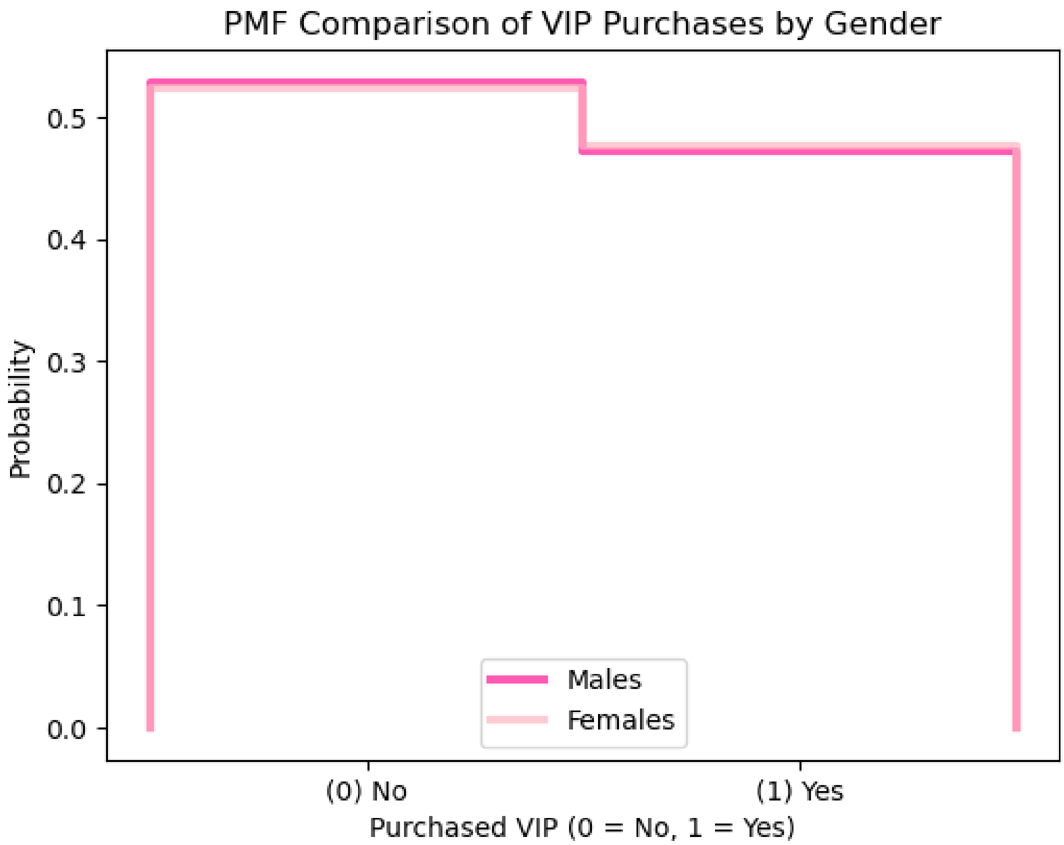
```
variables = ['Gender', 'PurchasedVIP', 'Income', 'Children', 'Age',␣
↪'Attractiveness', 'Matches']

for var in variables:
    if pd.api.types.is_numeric_dtype(dating_data[var]):
        mean = dating_data[var].mean()
        mode = dating_data[var].mode()[0]
        std_dev = dating_data[var].std()
        skewness = skew(dating_data[var].dropna())
        kurt = kurtosis(dating_data[var].dropna())

        print(f"Descriptive Characteristics for {var}:")
        print(f"  Mean: {mean}")
        print(f"  Mode: {mode}")
        print(f"  Spread (Std Dev): {std_dev}")
        print(f"  Skewness (Tails): {skewness}")
        print(f"  Kurtosis (Tails): {kurt}")
        print("-" * 40)
```

```
Descriptive Characteristics for Gender:
```

```
  Mean: 0.51
  Mode: 1
  Spread (Std Dev): 0.5001501276118512
  Skewness (Tails): -0.04000800240080025
  Kurtosis (Tails): -1.9983993597438978
----------------------------------------
Descriptive Characteristics for PurchasedVIP:
  Mean: 0.474
  Mode: 0
  Spread (Std Dev): 0.49957339157882746
  Skewness (Tails): 0.10414089379709283
  Kurtosis (Tails): -1.9891546742391433
----------------------------------------
Descriptive Characteristics for Income:
  Mean: 50988.447
  Mode: 37903
  Spread (Std Dev): 9889.336141142308
  Skewness (Tails): -0.03666032538780486
  Kurtosis (Tails): -0.027115305890066388
----------------------------------------
Descriptive Characteristics for Children:
  Mean: 0.978
  Mode: 0
  Spread (Std Dev): 0.9972514780688579
  Skewness (Tails): 0.6256579787886272
  Kurtosis (Tails): -0.7735835619694704
----------------------------------------
Descriptive Characteristics for Age:
  Mean: 34.616
  Mode: 25
  Spread (Std Dev): 9.147798982609224
  Skewness (Tails): -0.10252215291775699
  Kurtosis (Tails): -1.1651087682368406
----------------------------------------
Descriptive Characteristics for Attractiveness:
  Mean: 5.624
  Mode: 6
  Spread (Std Dev): 2.824287627101849
  Skewness (Tails): -0.030329051970477503
  Kurtosis (Tails): -1.1832697215455519
----------------------------------------
Descriptive Characteristics for Matches:
  Mean: 76.05
  Mode: 70
  Spread (Std Dev): 52.71314993600937
  Skewness (Tails): -0.20143833386713741
  Kurtosis (Tails): -1.0979256415927363
----------------------------------------
```

[156]:

```
scenario1 = dating_data[dating_data['Gender'] == 0]['PurchasedVIP']
scenario2 = dating_data[dating_data['Gender'] == 1]['PurchasedVIP']

pmf1 = thinkstats2.Pmf(scenario1, label='Males')
pmf2 = thinkstats2.Pmf(scenario2, label='Females')

thinkplot.PrePlot(2)
thinkplot.Pmf(pmf1, color='deeppink')
thinkplot.Pmf(pmf2, color='lightpink')
thinkplot.Config(title='PMF Comparison of VIP Purchases by Gender',
                 xlabel='Purchased VIP (0 = No, 1 = Yes)',
                 ylabel='Probability')
plt.xticks([0, 1], ['(0) No', '(1) Yes'])
thinkplot.Show()
```



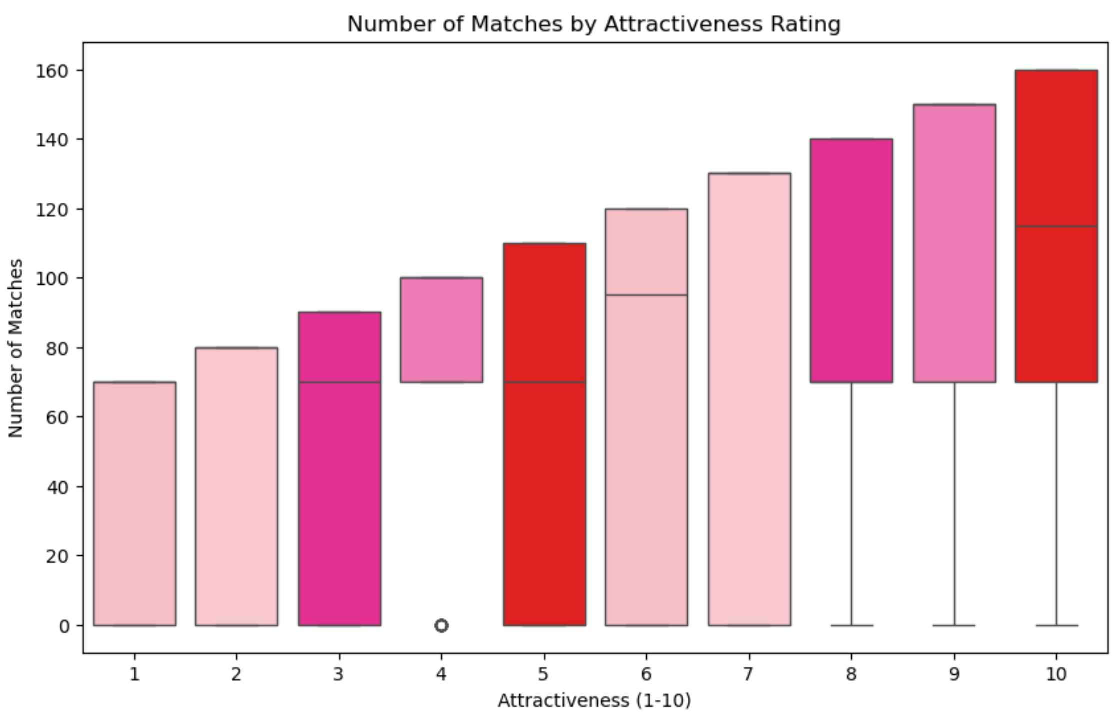

```
<Figure size 800x600 with 0 Axes>
```

[158]:

```
plt.figure(figsize=(10, 6))

palette = ['lightpink', 'pink', 'deeppink', 'hotpink', 'red']
```

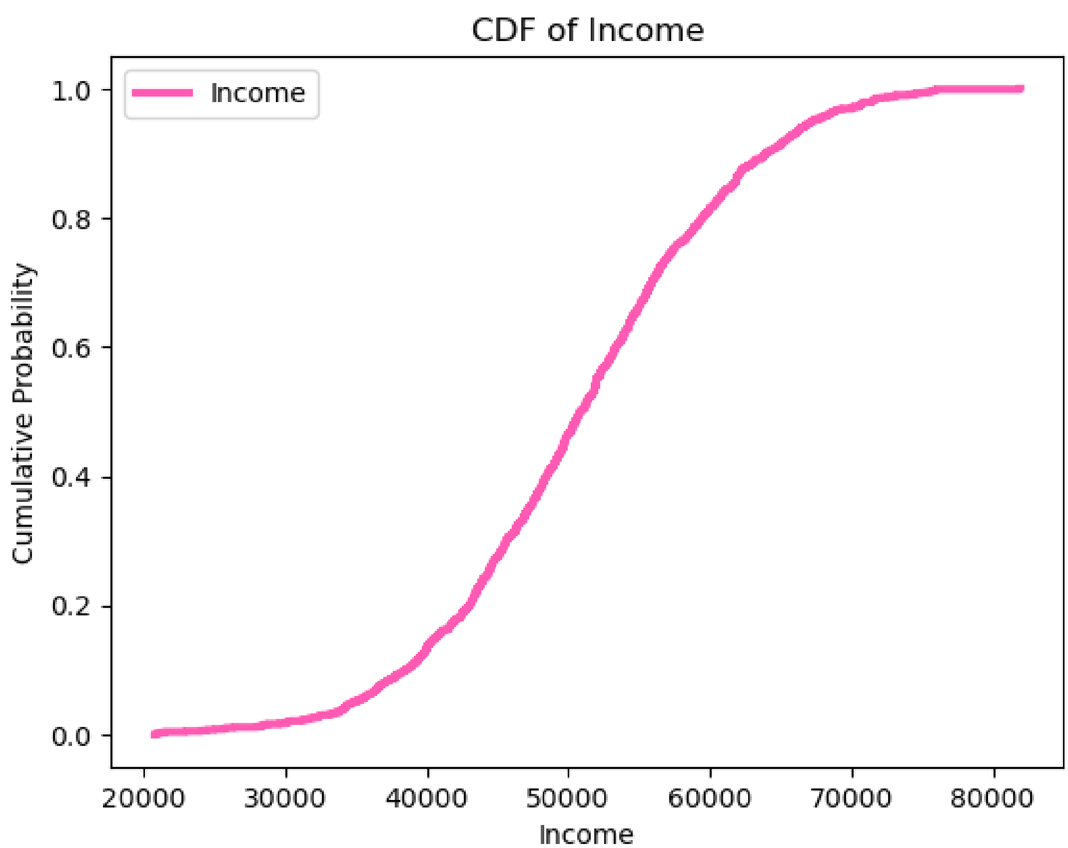
```
sns.boxplot(x=dating_data['Attractiveness'], y=dating_data['Matches'],␣
↪palette=palette)

plt.title('Number of Matches by Attractiveness Rating')
plt.xlabel('Attractiveness (1-10)')
plt.ylabel('Number of Matches')

plt.show()
```

```
/var/folders/sn/lbd9cnz96x19vj8w9wklrrdh0000gn/T/ipykernel_30275/2873105853.py:5
: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.boxplot(x=dating_data['Attractiveness'], y=dating_data['Matches'],
palette=palette)
/var/folders/sn/lbd9cnz96x19vj8w9wklrrdh0000gn/T/ipykernel_30275/2873105853.py:5
: UserWarning:
The palette list has fewer values (5) than needed (10) and will cycle, which may
produce an uninterpretable plot.
  sns.boxplot(x=dating_data['Attractiveness'], y=dating_data['Matches'],
palette=palette)
```



[160]:

```
income = dating_data['Income']
cdf = thinkstats2.Cdf(income, label='Income')
thinkplot.Cdf(cdf, color='deeppink')
thinkplot.Config(title='CDF of Income',
                 xlabel='Income',
                 ylabel='Cumulative Probability')

# Show the plot
thinkplot.Show()
```

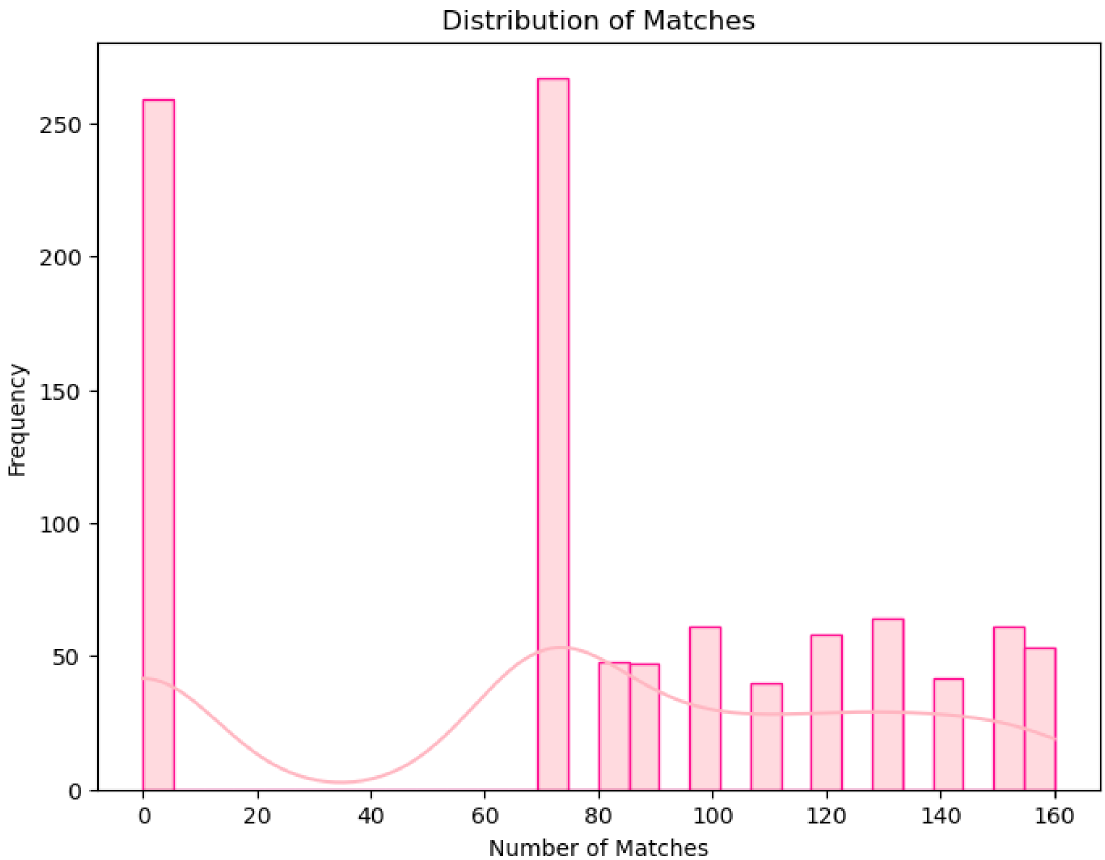



```
<Figure size 800x600 with 0 Axes>
```

[162]:

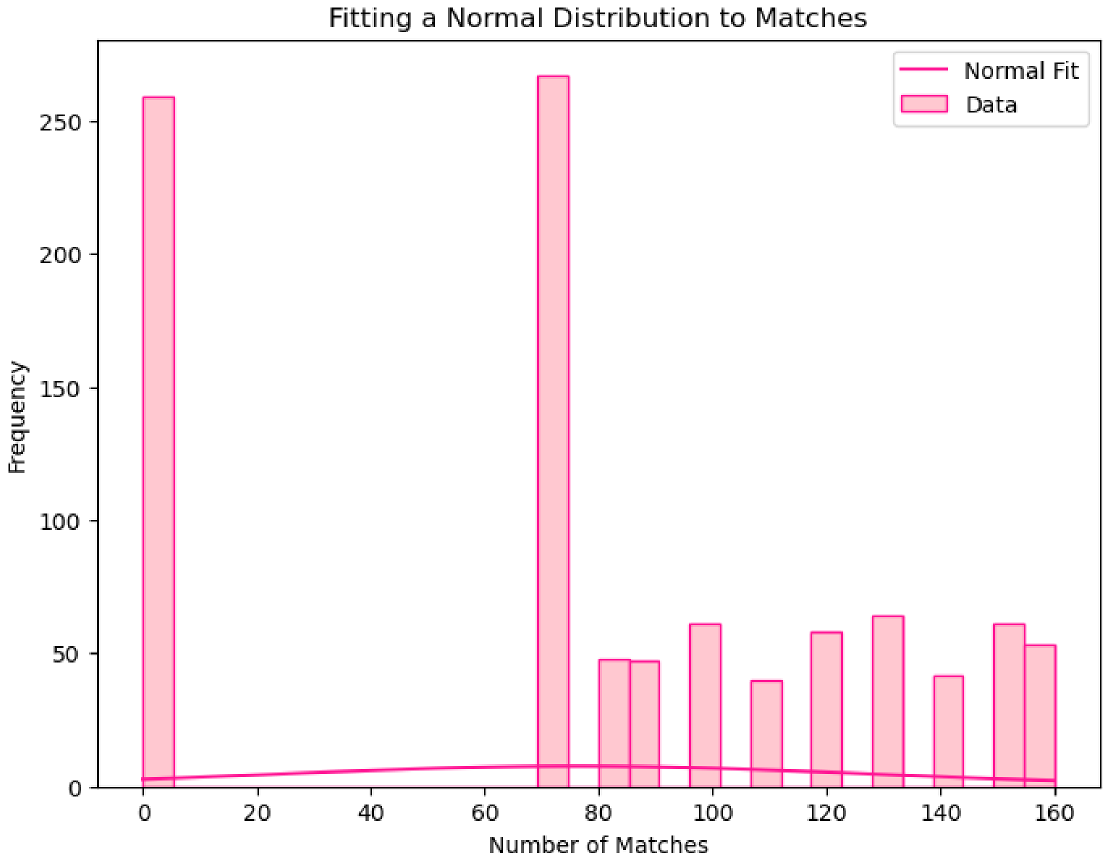
```
plt.figure(figsize=(8, 6))
sns.histplot(dating_data['Matches'], kde=True, bins=30, color='lightpink',␣
 ↪edgecolor='deeppink')

plt.title('Distribution of Matches')
plt.xlabel('Number of Matches')
```

```
plt.ylabel('Frequency')

plt.show()
```



[172]:
```
mu, std = norm.fit(dating_data['Matches'])

x = np.linspace(dating_data['Matches'].min(), dating_data['Matches'].max(), 100)
pdf = norm.pdf(x, mu, std)

plt.figure(figsize=(8, 6))
sns.histplot(dating_data['Matches'], kde=False, bins=30, color='lightpink',␣
↪edgecolor='deeppink', label='Data')
plt.plot(x, pdf * len(dating_data['Matches']), color='deeppink', label='Normal␣
↪Fit')

plt.title('Fitting a Normal Distribution to Matches')
plt.xlabel('Number of Matches')
plt.ylabel('Frequency')
plt.legend()
```

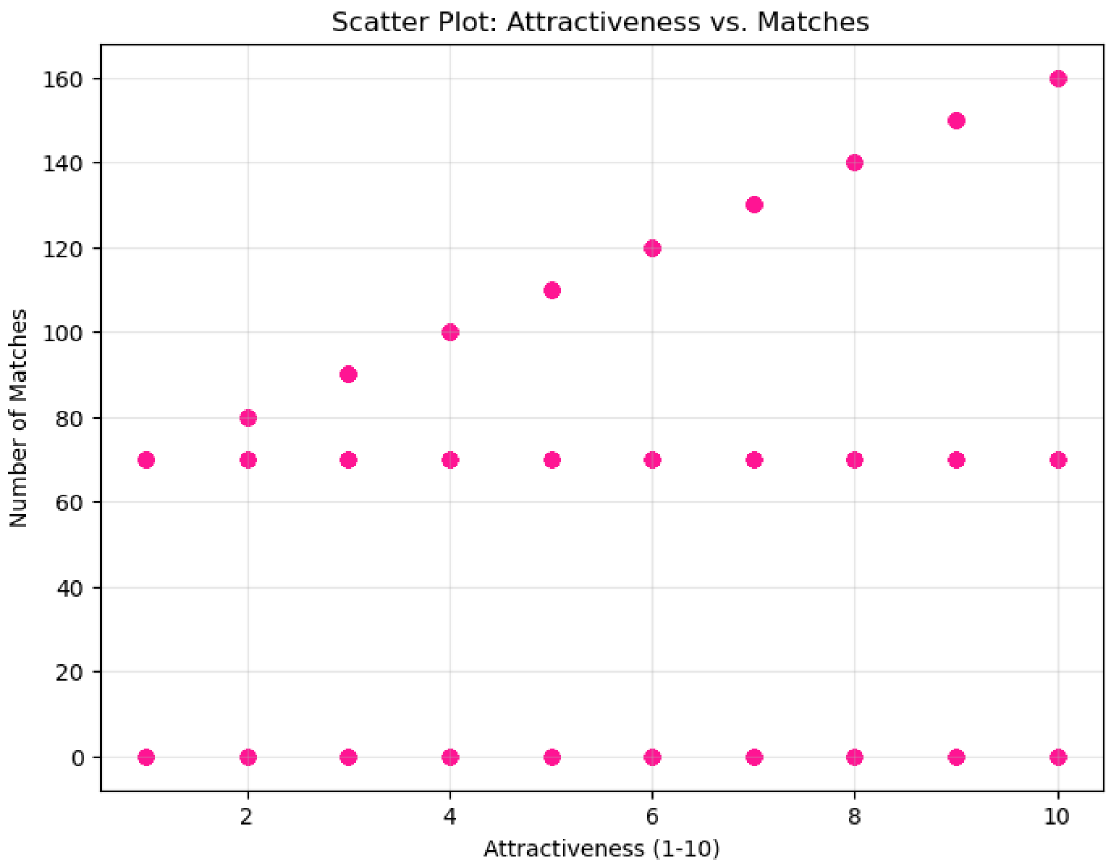
```
plt.show()
```



[72]:
```
ks_stat, p_value = kstest(dating_data['Matches'], 'norm', args=(mu, std))
print(f"K-S Statistic: {ks_stat:.4f}, P-value: {p_value:.4f}")
```

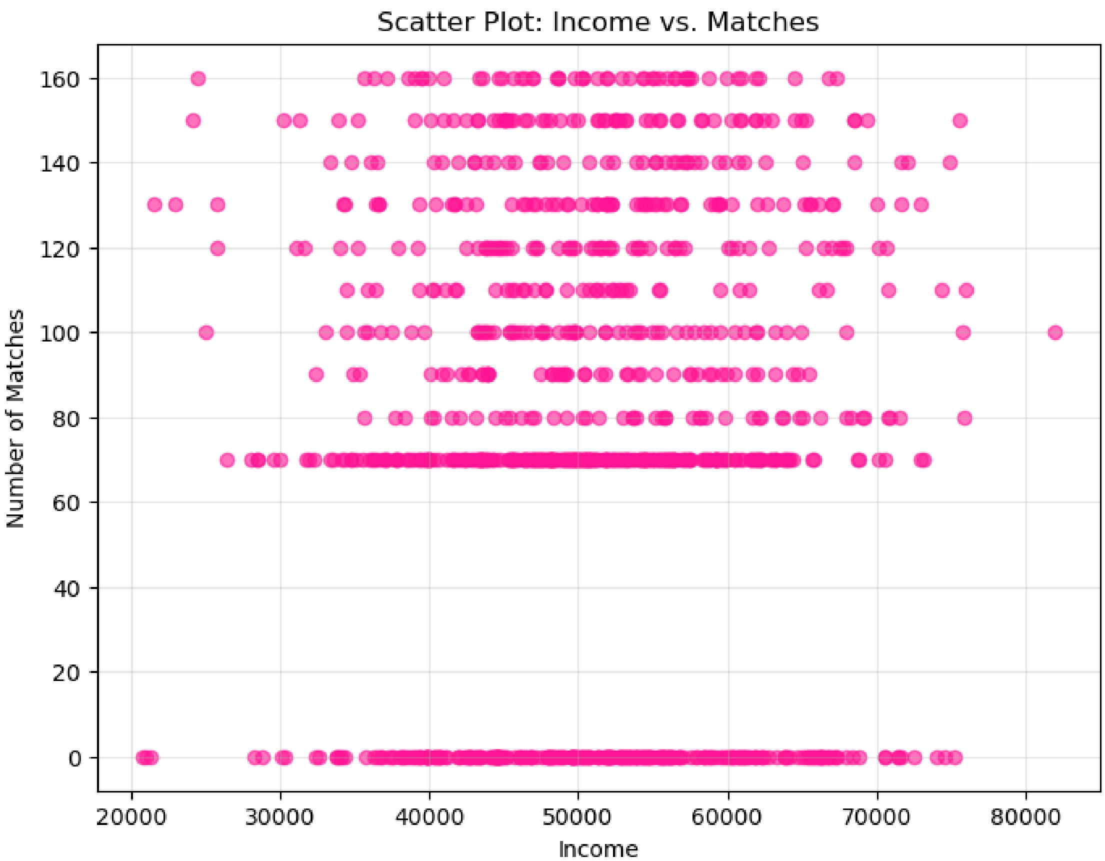
```
K-S Statistic: 0.1953, P-value: 0.0000
```

[164]:
```
plt.figure(figsize=(8, 6))
plt.scatter(dating_data['Attractiveness'], dating_data['Matches'], alpha=0.6,␣
 ↪color='deeppink')
plt.title('Scatter Plot: Attractiveness vs. Matches')
plt.xlabel('Attractiveness (1-10)')
plt.ylabel('Number of Matches')
plt.grid(alpha=0.3)
plt.show()
```



[168]:

```
plt.figure(figsize=(8, 6))
plt.scatter(dating_data['Income'], dating_data['Matches'], alpha=0.6,␣
 ↪color='deeppink')
plt.title('Scatter Plot: Income vs. Matches')
plt.xlabel('Income')
plt.ylabel('Number of Matches')
plt.grid(alpha=0.3)
plt.show()
```



[78]:
```
cov_attract_matches = np.cov(dating_data['Attractiveness'],␣
↪dating_data['Matches'])[0, 1]
cov_income_matches = np.cov(dating_data['Income'], dating_data['Matches'])[0, 1]
print(f"Covariance (Attractiveness vs. Matches): {cov_attract_matches:.2f}")
print(f"Covariance (Income vs. Matches): {cov_income_matches:.2f}")
```

```
Covariance (Attractiveness vs. Matches): 46.68
Covariance (Income vs. Matches): 3721.89
```

[82]:
```
corr_attract_matches, _ = pearsonr(dating_data['Attractiveness'],␣
↪dating_data['Matches'])
corr_income_matches, _ = pearsonr(dating_data['Income'], dating_data['Matches'])
print(f"Pearson's Correlation (Attractiveness vs. Matches):␣
↪{corr_attract_matches:.2f}")
print(f"Pearson's Correlation (Income vs. Matches): {corr_income_matches:.2f}")
```

```
Pearson's Correlation (Attractiveness vs. Matches): 0.31
Pearson's Correlation (Income vs. Matches): 0.01
```

[96]:
```
class AttractivenessMatchesTest(HypothesisTest):
    def TestStatistic(self, data):
        attract, matches = data
        return thinkstats2.Corr(attract, matches)

    def MakeModel(self):
        attract, matches = self.data
        self.data = attract, np.random.permutation(matches)

    def RunModel(self):
        attract, matches = self.data
        shuffled_matches = np.random.permutation(matches)
        return attract, shuffled_matches

attractiveness = dating_data['Attractiveness']
matches = dating_data['Matches']

test = AttractivenessMatchesTest((attractiveness, matches))
p_value = test.PValue(iters=1000)  # Number of iterations
print(f"P-Value: {p_value:.4f}")
```

```
P-Value: 0.0000
```

[100]:
```
X_single = dating_data['Attractiveness']
y = dating_data['Matches']

X_multiple = dating_data[['Attractiveness', 'Income', 'Age']]
```

[102]:
```
X_single = sm.add_constant(X_single)
X_multiple = sm.add_constant(X_multiple)
```

[104]:
```
model_single = sm.OLS(y, X_single).fit()

print(model_single.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                Matches   R-squared:                       0.098
Model:                            OLS   Adj. R-squared:                  0.097
Method:                 Least Squares   F-statistic:                     108.8
Date:                Sun, 17 Nov 2024   Prob (F-statistic):           2.98e-24
Time:                        17:42:46   Log-Likelihood:                -5331.6
No. Observations:                1000   AIC:                         1.067e+04
Df Residuals:                     998   BIC:                         1.068e+04
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
==
```

```
                      coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--
const              43.1366      3.530     12.219      0.000      36.209
50.064
Attractiveness      5.8523      0.561     10.432      0.000       4.751
6.953
==============================================================================
Omnibus:                      259.986   Durbin-Watson:                   2.004
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              112.928
Skew:                          -0.659   Prob(JB):                     3.01e-25
Kurtosis:                       2.014   Cond. No.                         14.3
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

[106]:

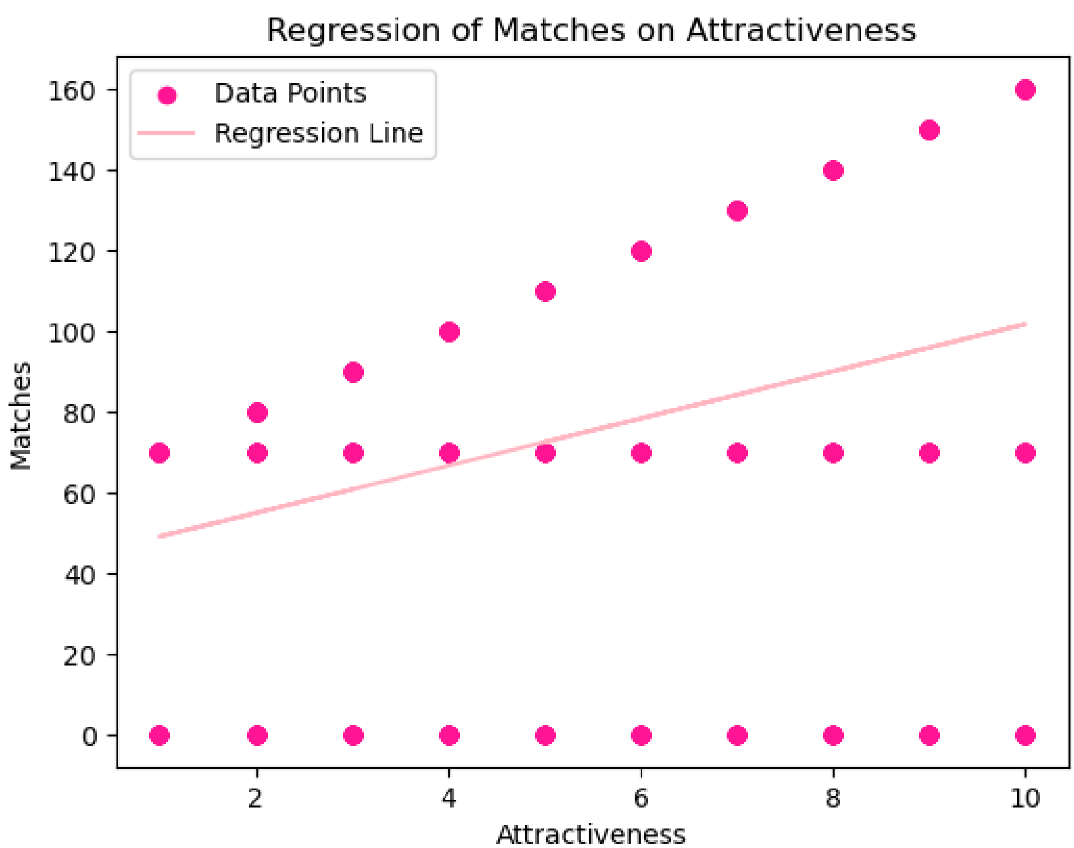
```python
model_multiple = sm.OLS(y, X_multiple).fit()

print(model_multiple.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                Matches   R-squared:                       0.099
Model:                            OLS   Adj. R-squared:                  0.096
Method:                 Least Squares   F-statistic:                     36.53
Date:                Sun, 17 Nov 2024   Prob (F-statistic):           2.13e-22
Time:                        17:43:12   Log-Likelihood:                -5331.1
No. Observations:                1000   AIC:                         1.067e+04
Df Residuals:                     996   BIC:                         1.069e+04
Df Model:                           3
Covariance Type:            nonrobust
================================================================================
==
                      coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--
const              47.1622     10.703      4.406      0.000      26.158
68.166
Attractiveness      5.8253      0.562     10.364      0.000       4.722
6.928
Income           3.256e-05      0.000      0.203      0.839      -0.000
0.000
Age                -0.1599      0.174     -0.921      0.357      -0.501
0.181
```

```
==============================================================================
Omnibus:                      260.058   Durbin-Watson:                   2.007
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              112.832
Skew:                          -0.659   Prob(JB):                     3.15e-25
Kurtosis:                       2.014   Cond. No.                     3.51e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 3.51e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[170]:

```
plt.scatter(dating_data['Attractiveness'], y, color='deeppink', label='Data␣
↪Points')
plt.plot(dating_data['Attractiveness'], model_single.predict(X_single),␣
↪color='lightpink', label='Regression Line')
plt.xlabel('Attractiveness')
plt.ylabel('Matches')
plt.title('Regression of Matches on Attractiveness')
plt.legend()
plt.show()
```



[ ]:

```

```