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

df = pd.read_csv("ecommerce.csv")

df.head()
```

```
                           Email  \
0      mstephenson@fernandez.com
1               hduke@hotmail.com
2                pallen@yahoo.com
3         riverarebecca@gmail.com
4  mstephens@davidson-herman.com

                                             Address            Avatar
\
0       835 Frank Tunnel\nWrightmouth, MI 82180-9605            Violet

1     4547 Archer Common\nDiazchester, CA 06566-8576         DarkGreen

2  24645 Valerie Unions Suite 582\nCobbborough, D...            Bisque

3   1414 David Throughway\nPort Jason, OH 22070-1220       SaddleBrown

4  14023 Rodriguez Passage\nPort Jacobville, PR 3...  MediumAquaMarine


   Avg. Session Length  Time on App  Time on Website  Length of
Membership  \
0            34.497268    12.655651        39.577668
4.082621
1            31.926272    11.109461        37.268959
2.664034
2            33.000915    11.330278        37.110597
4.104543
3            34.305557    13.717514        36.721283
3.120179
4            33.330673    12.795189        37.536653
4.446308

   Yearly Amount Spent
0           587.951054
1           392.204933
2           487.547505
3           581.852344
4           599.406092
```

```python
df.info()
```

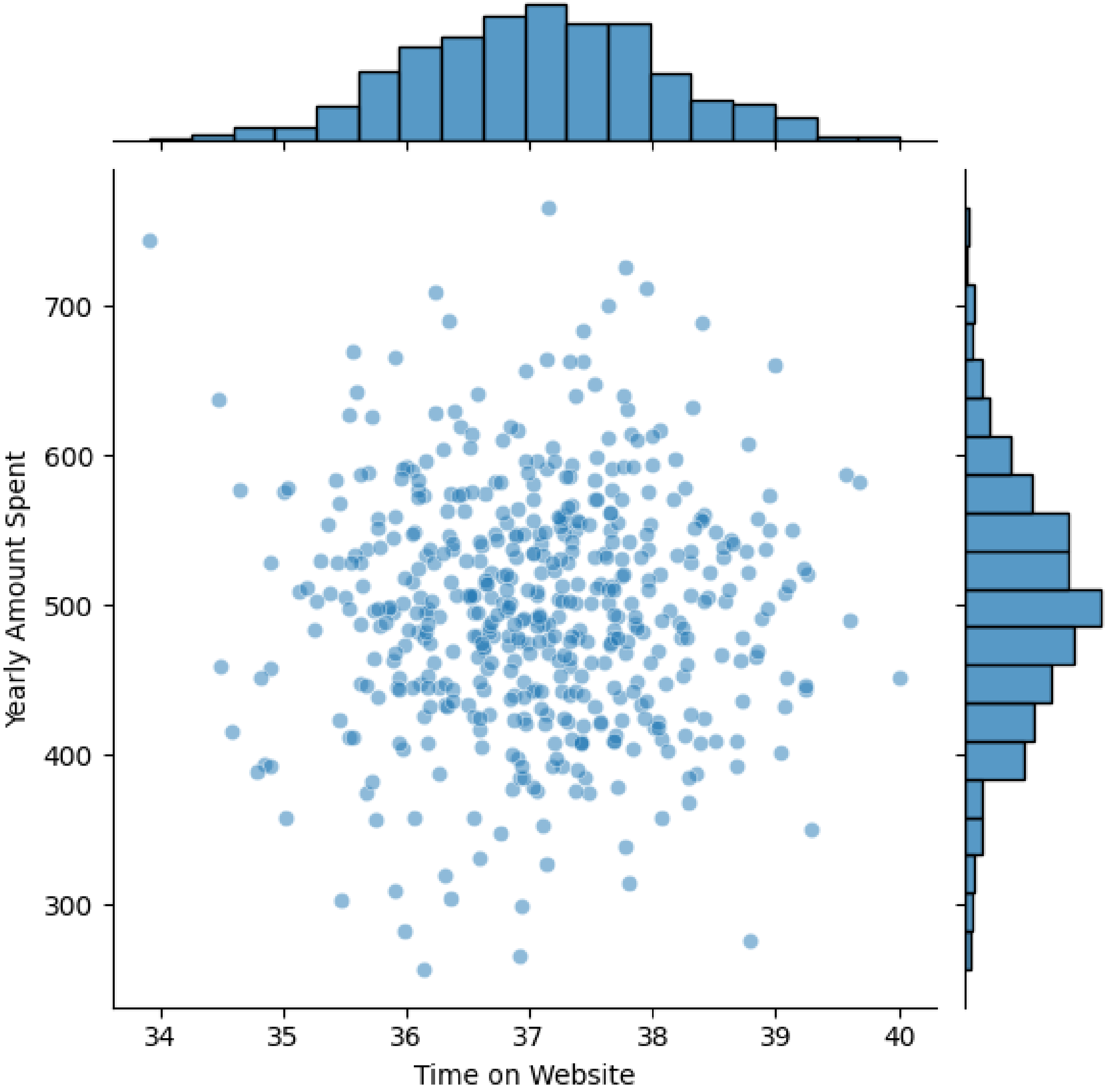
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Email                 500 non-null    object 
 1   Address               500 non-null    object 
 2   Avatar                500 non-null    object 
 3   Avg. Session Length   500 non-null    float64
 4   Time on App           500 non-null    float64
 5   Time on Website       500 non-null    float64
 6   Length of Membership  500 non-null    float64
 7   Yearly Amount Spent   500 non-null    float64
dtypes: float64(5), object(3)
memory usage: 31.4+ KB
```

```
df.describe()
```

```
       Avg. Session Length  Time on App  Time on Website  \
count           500.000000   500.000000       500.000000   
mean             33.053194    12.052488        37.060445   
std               0.992563     0.994216         1.010489   
min              29.532429     8.508152        33.913847   
25%              32.341822    11.388153        36.349257   
50%              33.082008    11.983231        37.069367   
75%              33.711985    12.753850        37.716432   
max              36.139662    15.126994        40.005182   

       Length of Membership  Yearly Amount Spent  
count            500.000000           500.000000  
mean               3.533462           499.314038  
std                0.999278            79.314782  
min                0.269901           256.670582  
25%                2.930450           445.038277  
50%                3.533975           498.887875  
75%                4.126502           549.313828  
max                6.922689           765.518462  
```

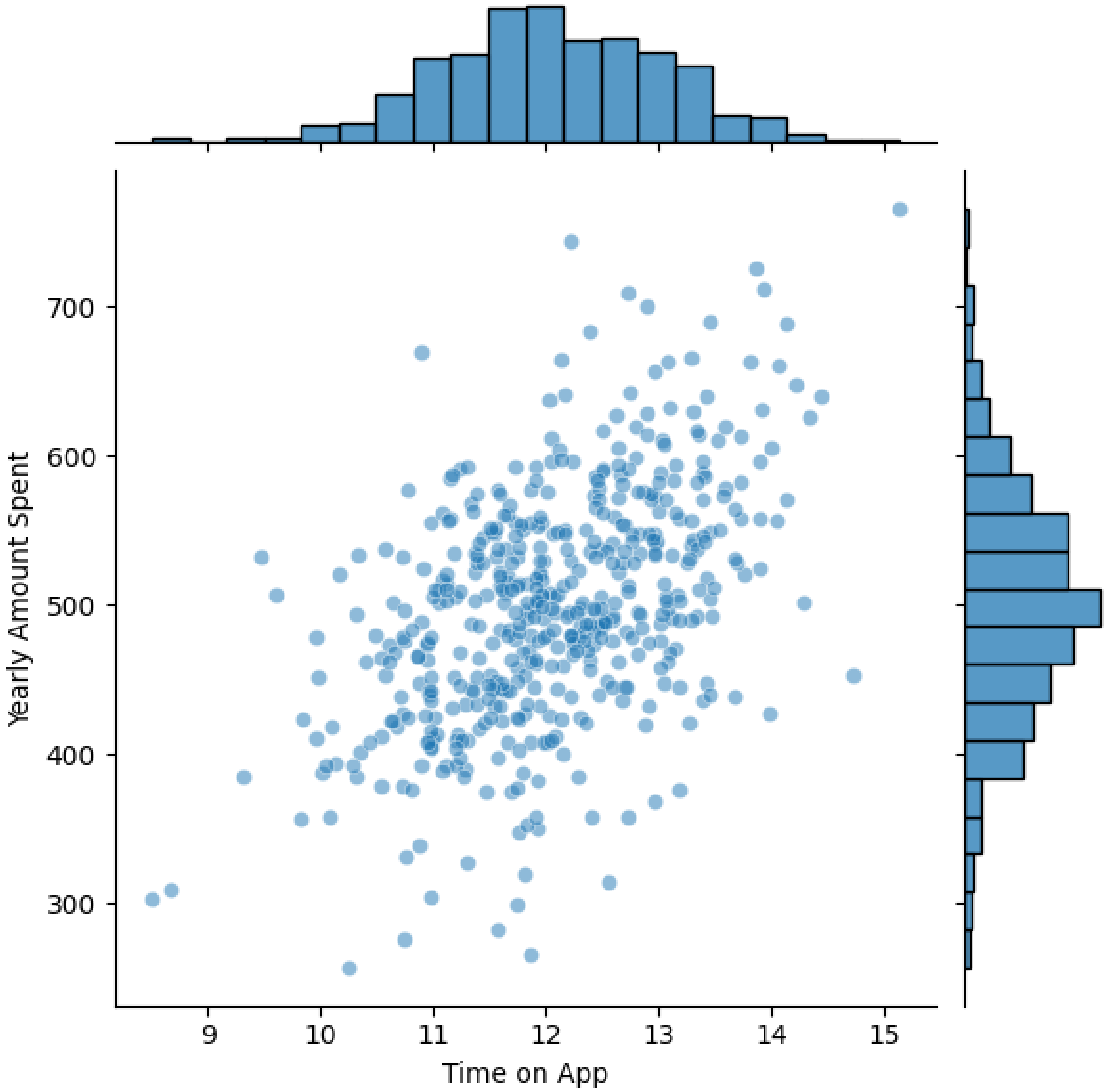
```
#EDA
sns.jointplot(x="Time on Website" , y="Yearly Amount Spent", data =df,
alpha = 0.5)
```

```
<seaborn.axisgrid.JointGrid at 0x24dfb572cf0>
```



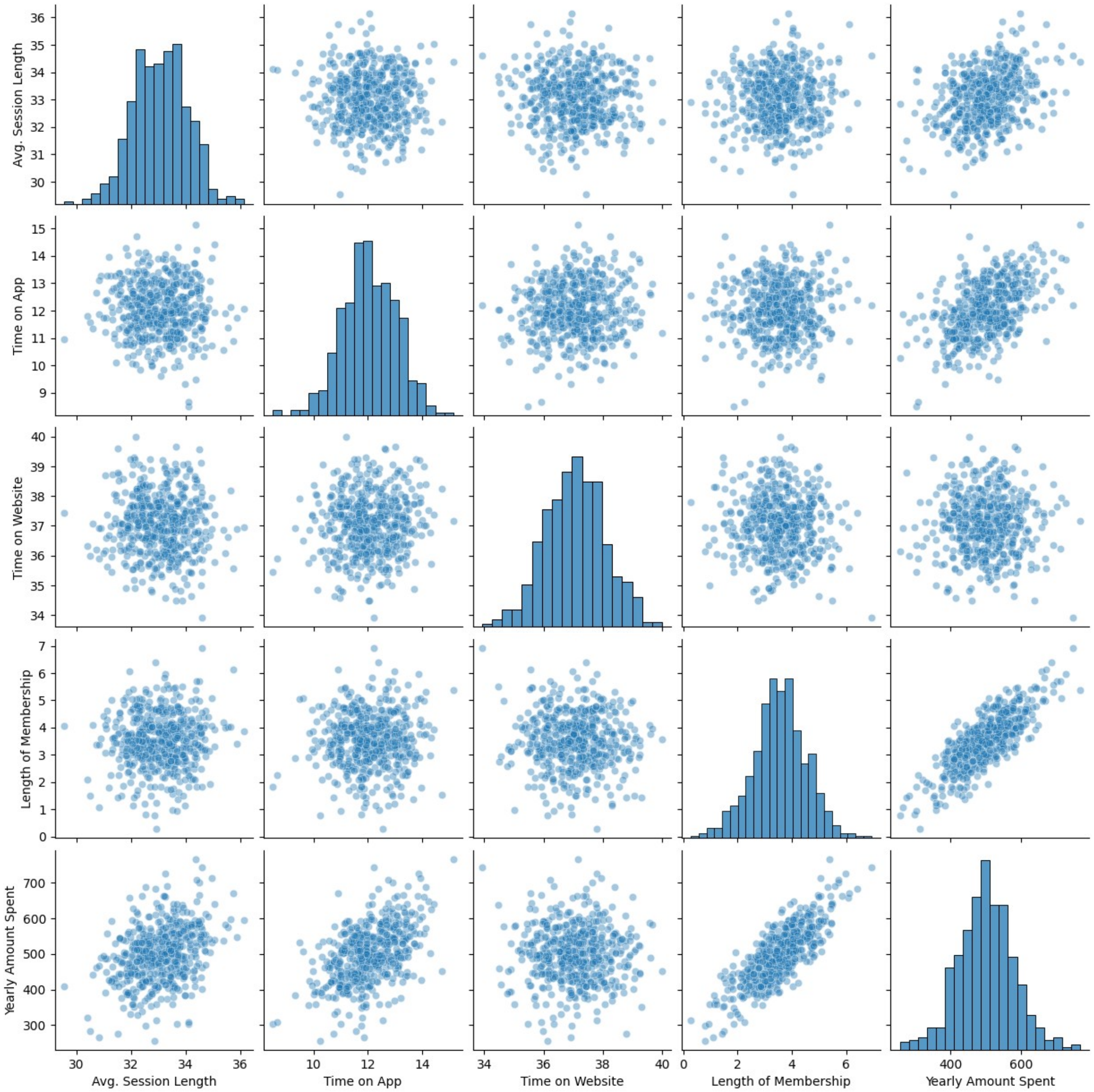
```
sns.jointplot(x= "Time on App" , y="Yearly Amount Spent", data = df,
alpha = 0.5)
```

```
<seaborn.axisgrid.JointGrid at 0x24dfb816350>
```



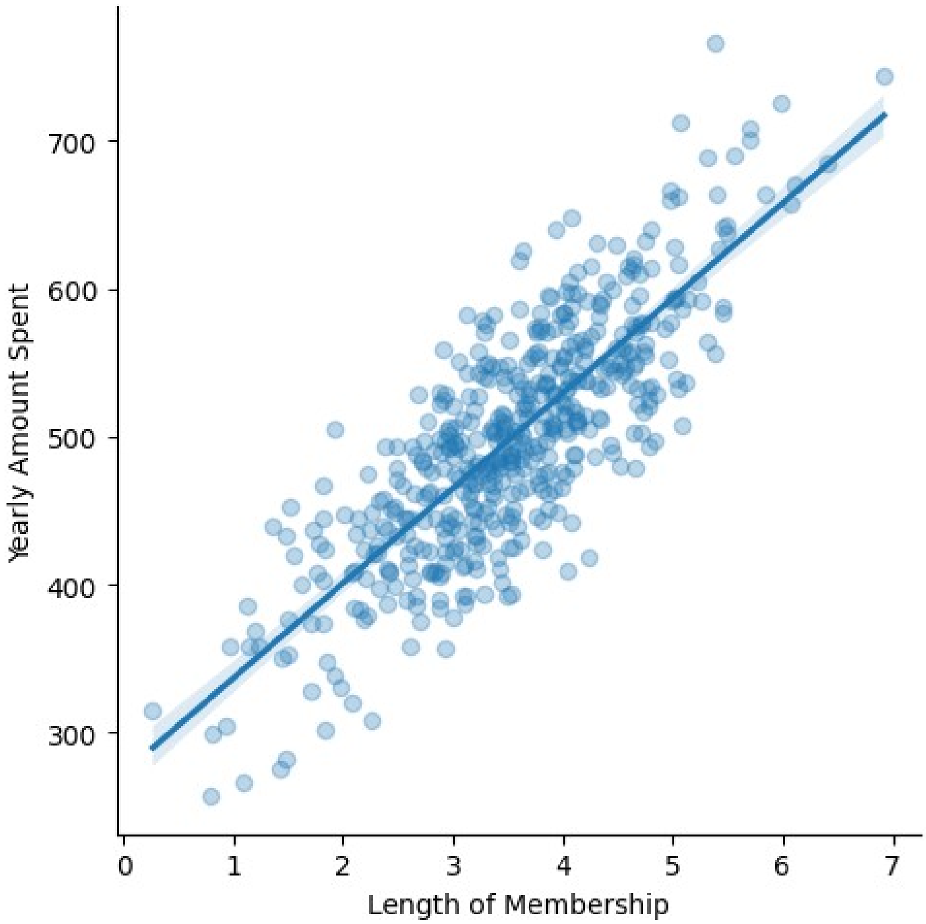
```
sns.pairplot(df,kind="scatter",plot_kws={"alpha" : 0.4})
```

```
<seaborn.axisgrid.PairGrid at 0x24dfb68f4d0>
```



```
sns.lmplot(x='Length of Membership',
           y='Yearly Amount Spent',
           data = df,
           scatter_kws = {'alpha':0.3})
```

```
<seaborn.axisgrid.FacetGrid at 0x24dfb68e510>
```



```
from sklearn.model_selection import train_test_split

X = df[['Time on App','Time on Website', 'Avg. Session Length','Length
of Membership']]
y= df['Yearly Amount Spent']
X

     Time on App  Time on Website  Avg. Session Length  Length of
Membership
0      12.655651        39.577668            34.497268
4.082621
1      11.109461        37.268959            31.926272
2.664034
2      11.330278        37.110597            33.000915
4.104543
3      13.717514        36.721283            34.305557
3.120179
4      12.795189        37.536653            33.330673
4.446308
..           ...              ...                  ...
...
```

```
495    13.566160        36.417985            33.237660
3.746573
496    11.695736        37.190268            34.702529
3.576526
497    11.499409        38.332576            32.646777
4.958264
498    12.391423        36.840086            33.322501
2.336485
499    12.418808        35.771016            33.715981
2.735160

[500 rows x 4 columns]
```

```
y
```

```
0      587.951054
1      392.204933
2      487.547505
3      581.852344
4      599.406092
          ...
495    573.847438
496    529.049004
497    551.620145
498    456.469510
499    497.778642
Name: Yearly Amount Spent, Length: 500, dtype: float64
```

```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size =
0.3, random_state = 42)
```

```
X_train
```

```
     Time on App  Time on Website  Avg. Session Length  Length of
Membership
5    12.026925        34.476878            33.871038
5.493507
116  12.011022        36.701052            33.925795
2.753424
45   12.170525        39.131097            34.555768
3.663105
16   11.733862        34.894093            32.125387
3.136133
462  11.233415        37.211153            33.503810
2.320550
..         ...              ...                  ...
...
106  12.190474        36.152462            32.291756
3.781823
270  12.956277        38.655095            34.006489
```

```
3.275734
348    10.886921        34.897828            31.812483
3.128639
435    14.132893        37.023479            32.259973
3.762070
102    11.448902        37.580190            32.425697
2.586968

[350 rows x 4 columns]

X_test

     Time on App  Time on Website  Avg. Session Length  Length of
Membership
361    10.347877        39.045156            32.077590
3.434560
73     12.817113        37.031539            32.808698
3.851579
374    10.101632        38.043453            31.447446
4.238296
155    13.457725        37.238806            32.449522
2.941411
104    10.994224        38.074452            31.389585
3.428860
..           ...              ...                  ...
...
266    11.777772        37.979827            34.555283
3.784273
23     11.657576        36.772604            32.903251
3.919302
222    11.109456        38.585855            34.334865
3.892891
261    13.041245        36.655208            32.550527
3.456234
426    13.271475        37.239847            31.425227
4.022103

[150 rows x 4 columns]

y_train

5      637.102448
116    479.231093
45     549.860590
16     457.847696
462    397.420584
          ...
106    494.551861
270    540.995739
348    392.810345
```

```
435    571.216005
102    420.737673
Name: Yearly Amount Spent, Length: 350, dtype: float64
```

```
#training the model
from sklearn.linear_model import LinearRegression

lm=LinearRegression()

lm.fit(X_train,y_train)
```

```
LinearRegression()
```

```
lm.coef_
```

```
array([38.59713548,  0.45914788, 25.72425621, 61.67473243])
```

```
cdf=pd.DataFrame(lm.coef_, X.columns, columns=['Coef'])

cdf
```

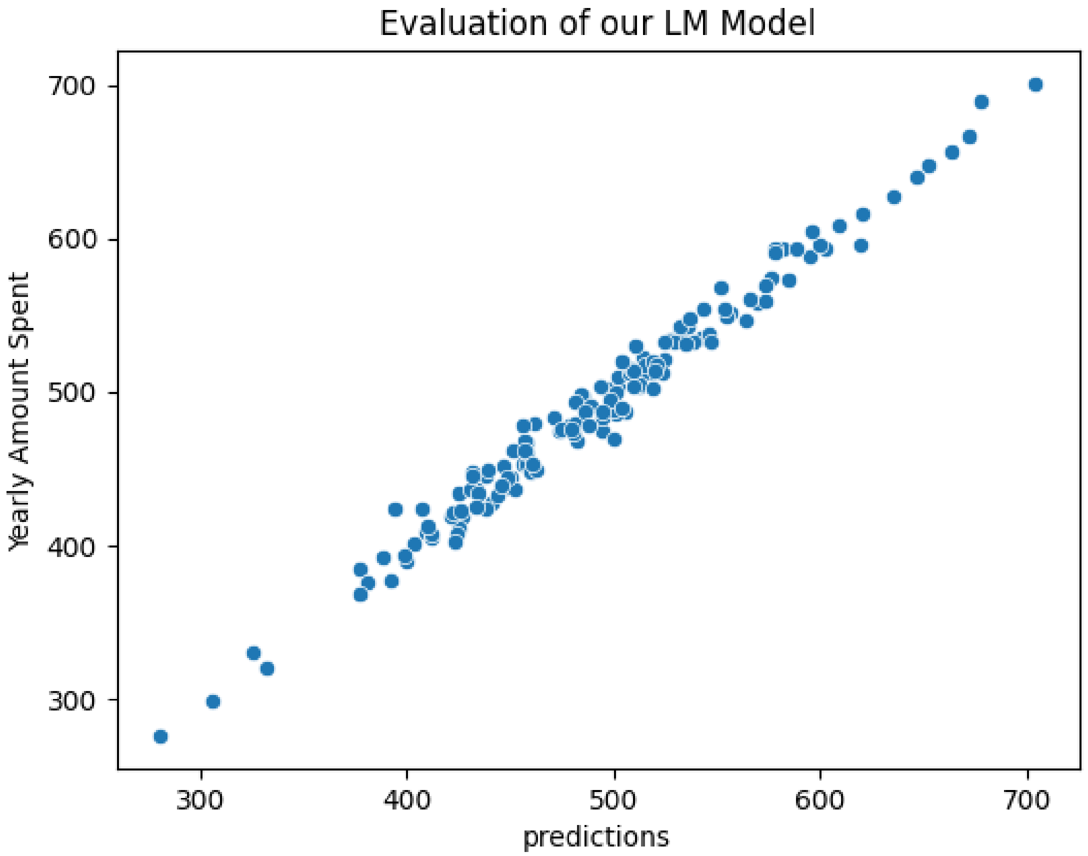
```
                           Coef
Time on App           38.597135
Time on Website        0.459148
Avg. Session Length   25.724256
Length of Membership  61.674732
```

```
#predictions
predictions=lm.predict(X_test)

predictions
```

```
array([403.66993069, 542.57756289, 427.06591658, 502.02460425,
       410.12143559, 569.93442508, 531.93431341, 506.29650969,
       408.71870658, 473.97737105, 441.46912726, 425.33703059,
       425.1297229 , 527.61676714, 431.45684016, 424.0769184 ,
       575.76543296, 484.89856554, 458.35936863, 481.96502182,
       502.32441491, 513.63783554, 507.58877002, 646.57464283,
       450.24372141, 496.27043415, 556.40457807, 554.95630839,
       399.64237199, 325.84623136, 532.89783259, 478.12238702,
       501.05701845, 305.97335848, 505.77244448, 483.79591969,
       518.8331528 , 438.18241857, 456.71094234, 471.04609461,
       494.44008972, 445.31155755, 508.78802753, 501.04594193,
       488.83499673, 535.38079541, 595.20129802, 514.04714872,
       280.76758312, 433.10112367, 421.70823427, 481.23640152,
       584.71372272, 608.7748096 , 563.98513427, 494.72804869,
       394.52133407, 456.4197529 , 573.08767515, 499.6984241 ,
       512.83277025, 392.12434043, 480.05057697, 481.54520299,
       475.1117359 , 546.2717533 , 430.85039085, 602.16082001,
       422.3695128 , 493.57280186, 528.74970313, 581.49002635,
       620.19139276, 512.56880298, 411.76623862, 498.47637494,
       461.51337557, 446.41371051, 448.07229961, 535.44710412,
       599.45225302, 619.33717662, 494.15919062, 671.99976398,
       532.46469814, 438.90606319, 515.04975242, 546.7821954 ,
       331.94282076, 510.51987447, 536.57891032, 500.19533618,
       376.92345776, 573.73961388, 479.68031607, 588.61435483,
       485.69922203, 456.40200844, 399.25197845, 451.5098931 ,
       519.40693826, 434.71194217, 596.13049586, 487.91791966,
       407.46691799, 524.16812757, 504.12982787, 452.11540623,
       524.21791295, 457.59311643, 444.19371592, 457.80432916,
       448.76590761, 438.31789012, 677.04967982, 566.09639245,
       651.93616661, 381.08127926, 577.5577254 , 578.35797052,
       518.61431291, 538.94532336, 377.4301223 , 663.30814872,
       523.83158824, 456.86065622, 446.07594402, 388.55038282,
       521.03242183, 431.94999241, 460.08016327, 426.31959507,
       433.30417088, 634.89577554, 462.41086078, 460.71673829,
       512.49535288, 703.83033889, 411.84238624, 551.54681408,
       553.33669558, 409.68202123, 423.34491341, 509.66438623,
       509.88865178, 543.67591782, 504.31300469, 519.18802223,
       520.03155195, 535.13855037])
```

```
sns.scatterplot(x=predictions,y=y_test)
plt.xlabel('predictions')
plt.title("Evaluation of our LM Model")
```

```
Text(0.5, 1.0, 'Evaluation of our LM Model')
```

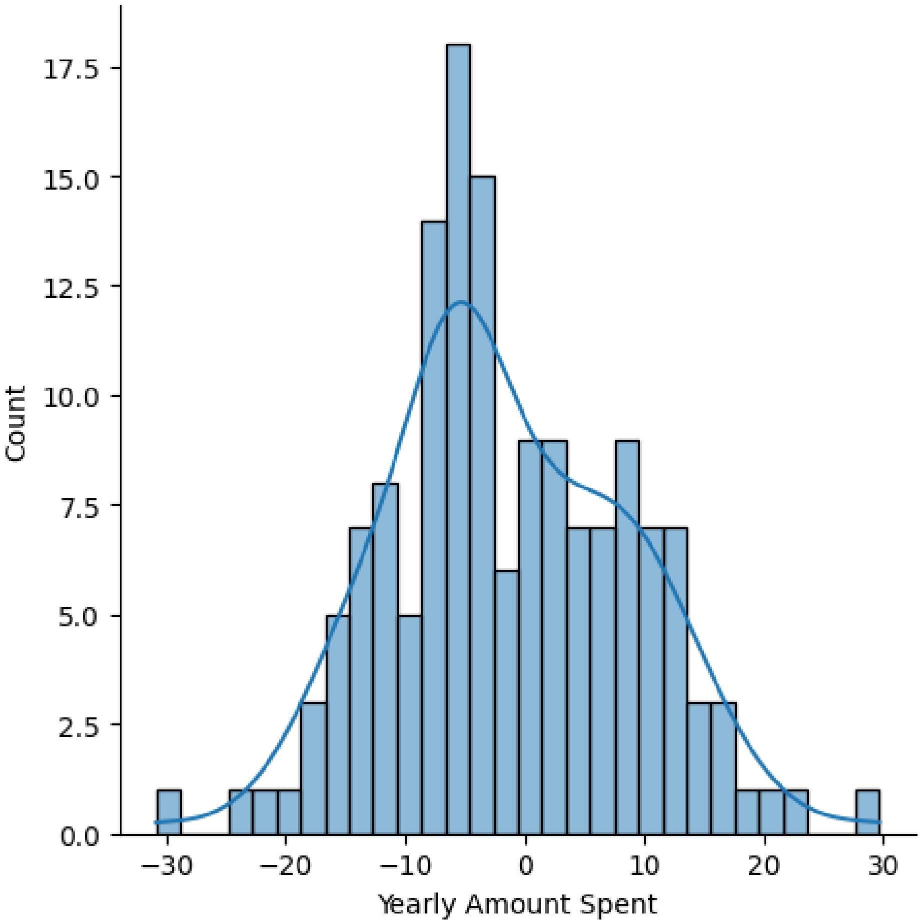


```
from sklearn.metrics import mean_squared_error, mean_absolute_error
import math

print("Mean Absolute Error: ",
mean_absolute_error(y_test,predictions))
print("Mean Squared Error: ", mean_squared_error(y_test,predictions))
print("RMSE: ", math.sqrt(mean_squared_error(y_test,predictions)))

Mean Absolute Error:  8.4260916414321
Mean Squared Error:  103.91554136503328
RMSE:  10.193897260863153

#residual analysis
residuals = y_test-predictions
residuals

361    -2.636795
73     -7.800375
374    -8.463174
155     1.953775
104    -0.051825
         ...
266    10.327176
23     15.027984
```

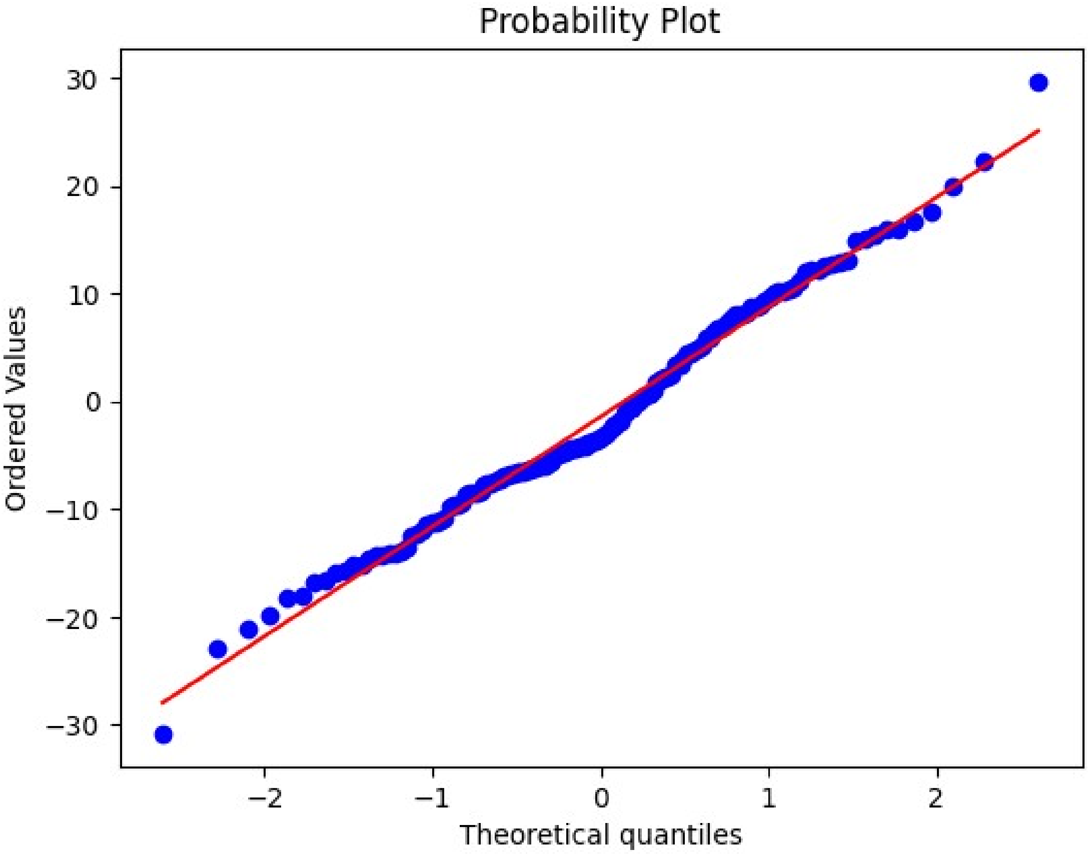
```
222   -16.778237
261    -6.021734
426    -4.371832
Name: Yearly Amount Spent, Length: 150, dtype: float64
```

```
#test normality of residuals
sns.displot(residuals, bins=30,kde=True)
```

```
<seaborn.axisgrid.FacetGrid at 0x24d85413750>
```



```
import pylab
import scipy.stats as stats
stats.probplot(residuals,dist='norm',plot=pylab)
pylab.show()
```


Probability Plot
Ordered Values
Theoretical quantiles
30
20
10
0
−10
−20
−30
−2
−1
0
1
2