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

```
df=pd.read_csv("/content/Ecommerce Customers.csv")
```

```
df
```

	Email	Address	Avatar	Avg. Session Length	Time on App	Time on Website	Length of Membership	Yearly Amount Spent
0	mstephenson@fernandez.com	835 Frank Tunnel\nWrightmouth, MI 82180-9605	Violet	34.497268	12.655651	39.577668	4.082621	587.951054
1	hduke@hotmail.com	4547 Archer Common\nDiazchester, CA 06566-8576	DarkGreen	31.926272	11.109461	37.268959	2.664034	392.204933
2	pallen@yahoo.com	24645 Valerie Unions Suite 582\nCobbborough, D...	Bisque	33.000915	11.330278	37.110597	4.104543	487.547505
3	riverarebecca@gmail.com	1414 David Throughway\nPort Jason, OH 22070-1220	SaddleBrown	34.305557	13.717514	36.721283	3.120179	581.852344
4	mstephens@davidson-herman.com	14023 Rodriguez Passage\nPort Jacobville, PR 3...	MediumAquaMarine	33.330673	12.795189	37.536653	4.446308	599.406092
...	...	...	...	...	...	...	...	...
495	lewisjessica@craig-evans.com	4483 Jones Motorway Suite 872\nLake Jamiefurt,...	Tan	33.237660	13.566160	36.417985	3.746573	573.847438
496	katrina56@gmail.com	172 Owen Divide Suite 497\nWest Richard, CA 19320	PaleVioletRed	34.702529	11.695736	37.190268	3.576526	529.049004

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
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
```

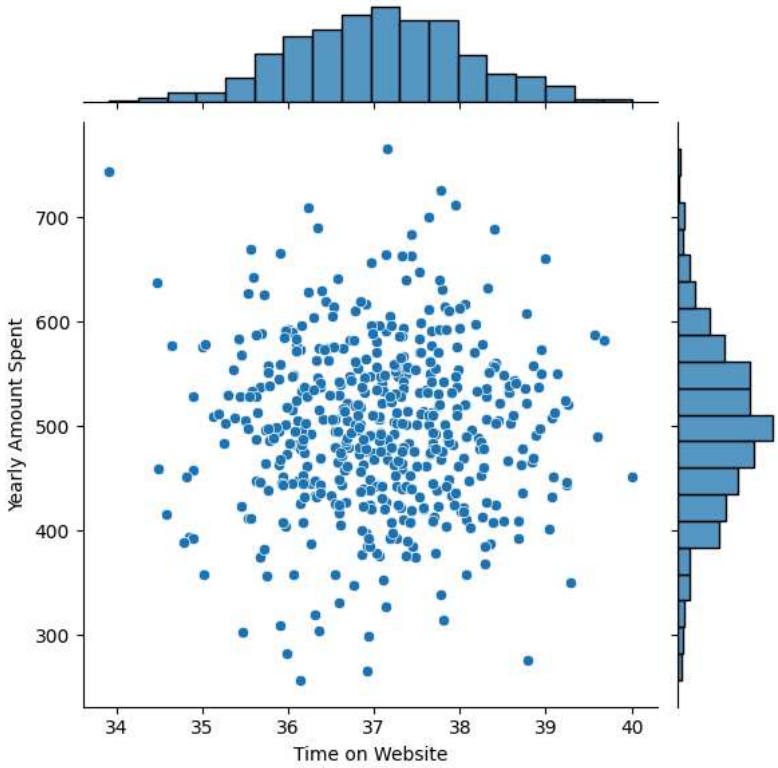
```
df.describe()
```

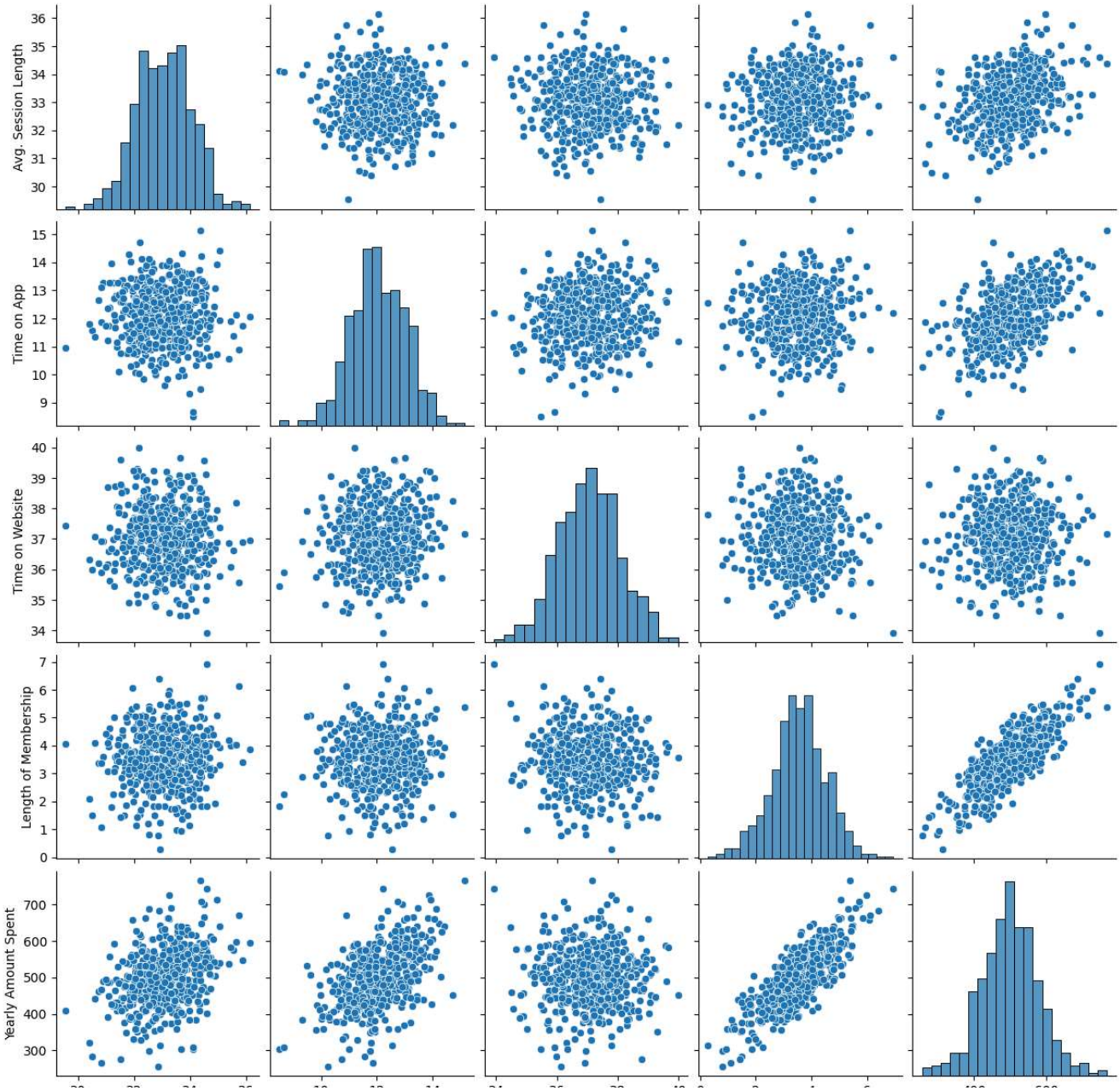
	Avg. Session Length	Time on App	Time on Website	Length of Membership	Yearly Amount Spent
count	500.000000	500.000000	500.000000	500.000000	500.000000
mean	33.053194	12.052488	37.060445	3.533462	499.314038
std	0.992563	0.994216	1.010489	0.999278	79.314782
min	29.532429	8.508152	33.913847	0.269901	256.670582
25%	32.341822	11.388153	36.349257	2.930450	445.038277
50%	33.082008	11.983231	37.069367	3.533975	498.887875
75%	33.711985	12.753850	37.716432	4.126502	549.313828
max	36.139662	15.126994	40.005182	6.922689	765.518462

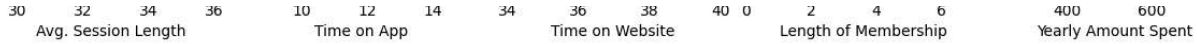
```
# EDA

sns.jointplot(x="Time on Website",y="Yearly Amount Spent",data=df)
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x78174c4f3340>
```

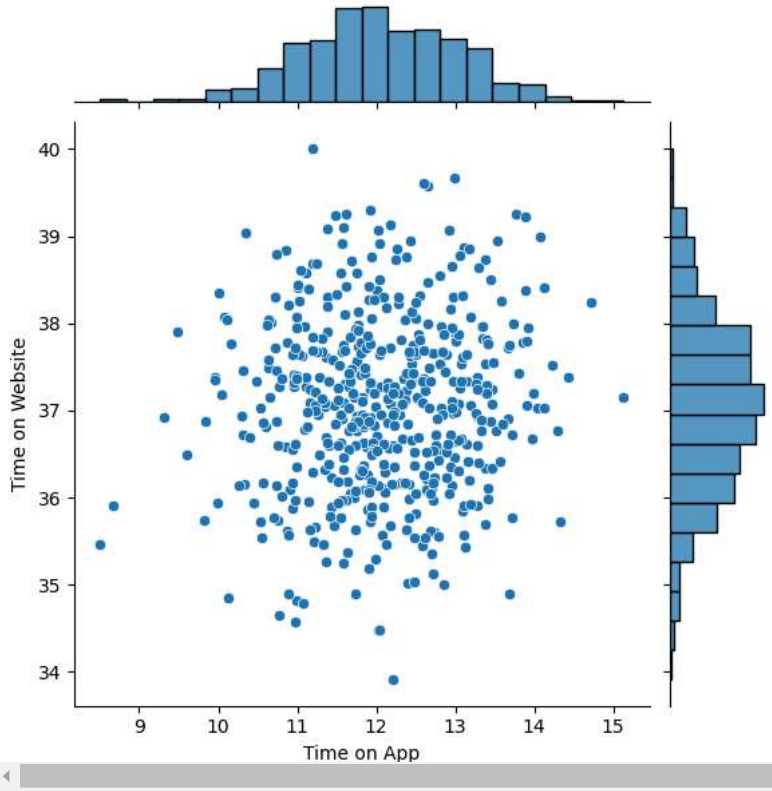






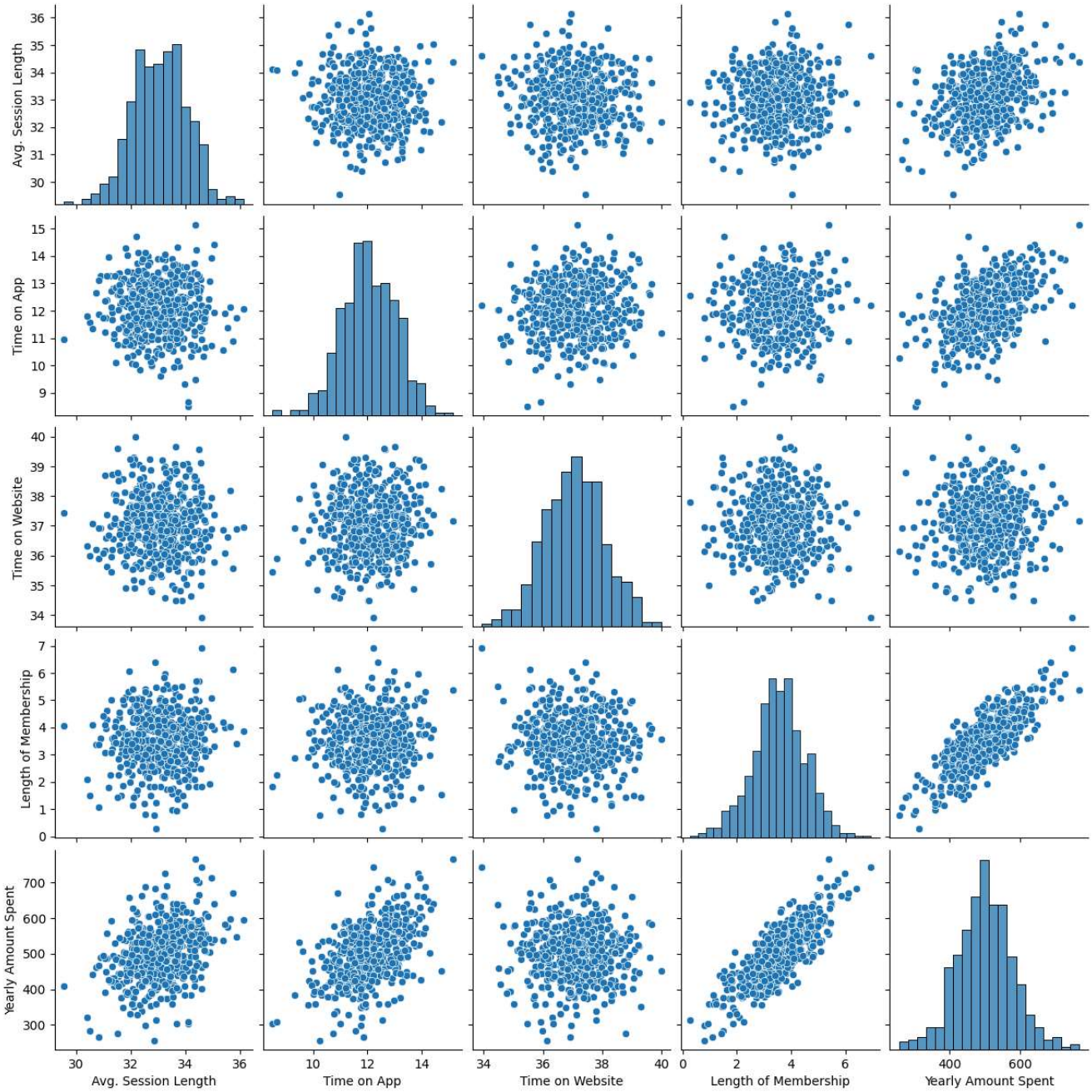

```
sns.jointplot(x="Time on App",y="Time on Website",data=df)
```

```
<seaborn.axisgrid.JointGrid at 0x781748eb1960>
```



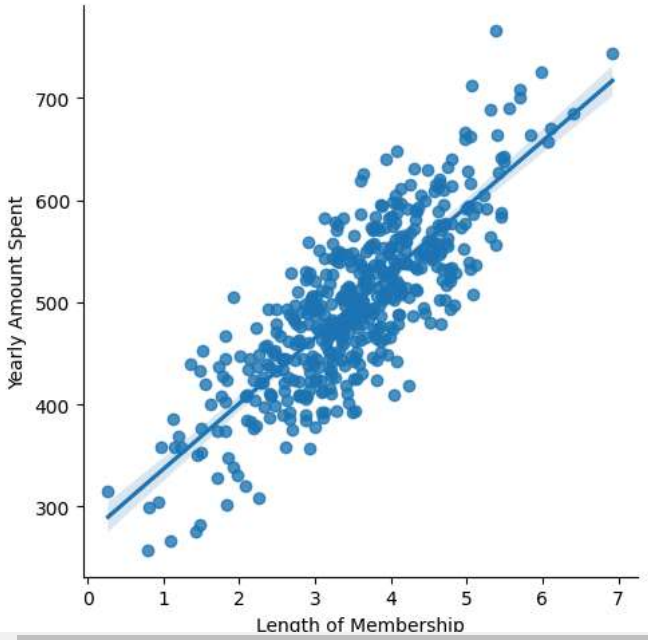

```
sns.pairplot(df,kind="scatter")
```

```
<seaborn.axisgrid.PairGrid at 0x78174c35ace0>
```




```
# sns.implot
sns.lmplot(x="Length of Membership",y="Yearly Amount Spent",data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x781749c613f0>
```




```
from sklearn.model_selection import train_test_split
```

```
x=df[["Time on App","Time on Website"]]
y=df["Yearly Amount Spent"]
```

```
x ,y
```

```
(     Time on App  Time on Website
0      12.655651        39.577668
1      11.109461        37.268959
2      11.330278        37.110597
3      13.717514        36.721283
4      12.795189        37.536653
..           ...              ...
495    13.566160        36.417985
496    11.695736        37.190268
497    11.499409        38.332576
498    12.391423        36.840086
499    12.418808        35.771016

[500 rows x 2 columns],
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=45)
```

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

lr=LinearRegression()
```

```
lr.fit(x_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
lr.coef_
```

```
array([42.35815577, -6.00537777])
```

```
#priction
```

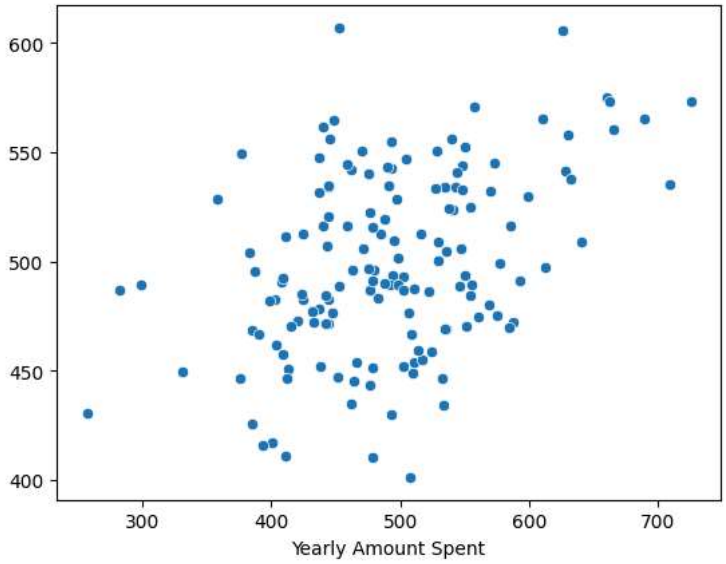
```
prridction=lr.predict(x_test)
```

```
prridction
```

```
array([480.4735308 , 528.73574536, 476.70265741, 410.54358784,
       472.47764743, 401.1880405 , 476.26817222, 534.24223016,
       429.74085179, 555.95876157, 474.92067802, 471.30685655,
       472.67123445, 504.93588855, 450.94452959, 541.59681219,
       478.61528854, 451.59100473, 573.51732675, 509.72480205,
       482.74437452, 512.34988313, 493.67776869, 468.76226114,
       512.85904642, 528.35223771, 469.26414071, 529.95892647,
       489.17442286, 516.26266417, 558.21845223, 496.2495323 ,
       523.50216871, 605.68537584, 511.26135243, 540.06912996,
       430.79079717, 491.38394178, 506.19835449, 542.8177852 ,
       523.98721087, 470.54718815, 446.57286316, 466.76895888,
       549.4138297 , 446.35065065, 503.77134733, 575.18357678,
       487.13191118, 607.0464422 , 543.36290099, 458.45273671,
       417.23619353, 449.03142534, 488.98878786, 482.74397319,
       546.66550032, 475.39082303, 541.8355542 , 445.16556898,
       507.29900691, 454.02330078, 446.63780233, 484.20161475,
       498.95320208, 493.3394981 , 543.96642246, 462.01980726,
       550.43921484, 541.03139247, 550.72523082, 434.16383252,
       489.57312807, 466.8200849 , 425.6193655 , 556.17732853,
       571.14135002, 497.32764428, 534.06349137, 490.67581002,
       524.86124939, 537.91827124, 472.09862736, 565.08849732,
       531.95287711, 486.18310388, 485.14414603, 492.37579255,
       410.75772061, 516.45607579, 533.07230026, 515.77525742,
       469.54368779, 484.29917014, 489.53291843, 496.9975533 ,
       457.49083155, 447.0374074 , 533.9210104 , 443.17860383,
       496.17573537, 489.62355158, 452.12444619, 483.06892355,
       454.95095857, 535.02005516, 547.47902608, 482.64767187,
       544.51794227, 505.75664953, 486.90849923, 573.55347922,
       560.40800358, 516.29685199, 468.97240043, 520.5610466 ,
       452.2535675 , 564.76320075, 534.60871036, 481.7494346 ,
       565.27540265, 512.79507932, 434.77431263, 508.76787197,
       490.99090597, 552.66276578, 488.79806443, 415.6972055 ,
       449.28033558, 545.26665954, 470.29227851, 495.18675651,
       459.6115419 , 554.85949121, 519.65613744, 522.46292403,
       561.36978949, 453.77427417, 500.32151213, 486.99400939,
       501.55860154, 508.98912367, 535.00555889, 533.47507929,
       471.47708742, 476.88816343, 490.26681123, 493.92411333,
       487.6209243 , 531.44371458])
```

```
sns.scatterplot(x=y_test,y=prridction)
```

```
<Axes: xlabel='Yearly Amount Spent'>
```




```
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
mse=mean_squared_error(y_test,prridction)
mse
```

```
5390.461863536032
```

```
mse=mean_absolute_error(y_test,prridction)
mse
```