Ecommerce_Project_Linear_Regression

February 3, 2022

Ecommerce Project

by : Joseph Ramon

A clothing company based in New York City sells clothing online and also offers in-store style and clothing advice sessions. Customers come in to the store, have sessions/meetings with a personal stylist, then they can go home and order either on a mobile app or website for the clothes they want.

The company is trying to decide whether to focus their efforts on their mobile app experience or their website.

Will build and use a Linear Regression model.

Imports

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

0.1 Get the Data

Will work with the Ecommerce Customers csv file from the company. It has Customer info, suchas Email, Address, and their color Avatar. Then it also has numerical value columns:

- Avg. Session Length: Average session of in-store style advice sessions.
- Time on App: Average time spent on App in minutes
- Time on Website: Average time spent on Website in minutes
- Length of Membership: How many years the customer has been a member.

[3]:
```
customers = pd.read_csv("Ecommerce Customers")
```

[4]:
```
customers.head()
```

[4]:
```
                           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
```

[5]:
```python
customers.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
```

[6]:
```python
customers.describe()
```

[6]:
```
       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
```

0.2 Exploratory Data Analysis

Will only be using the numerical data of the csv file.

Will use seaborn to create a jointplot to compare the Time on Website and Yearly Amount Spent columns. Does the correlation make sense?

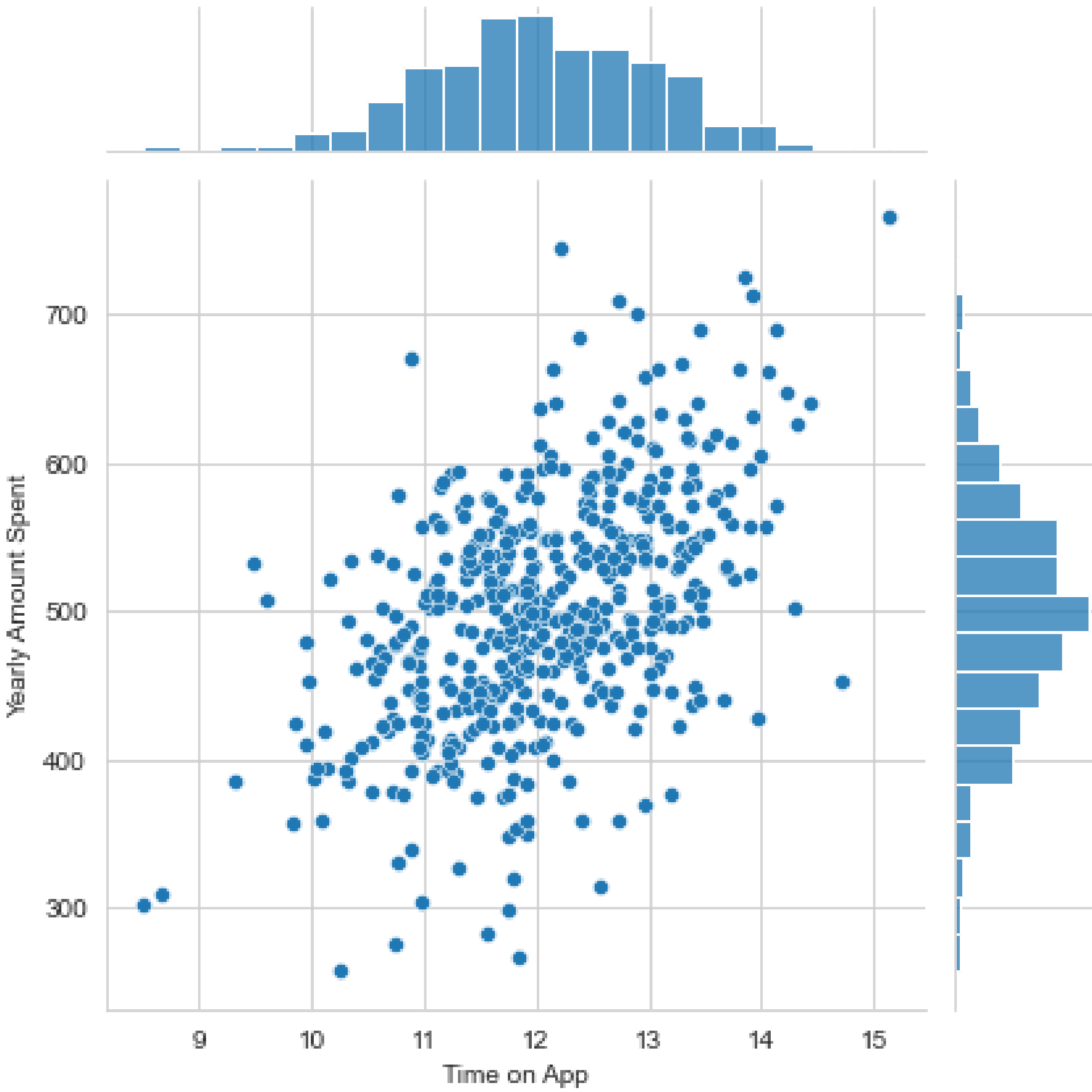
[7]:
```
customers.columns
```

[7]:
```
Index(['Email', 'Address', 'Avatar', 'Avg. Session Length', 'Time on App',
       'Time on Website', 'Length of Membership', 'Yearly Amount Spent'],
      dtype='object')
```

[8]:
```
sns.set_palette('tab10')
sns.set_style('whitegrid')
```

[9]:
```
sns.jointplot(x='Time on App',y='Yearly Amount Spent',\
              data=customers,kind='scatter')
```

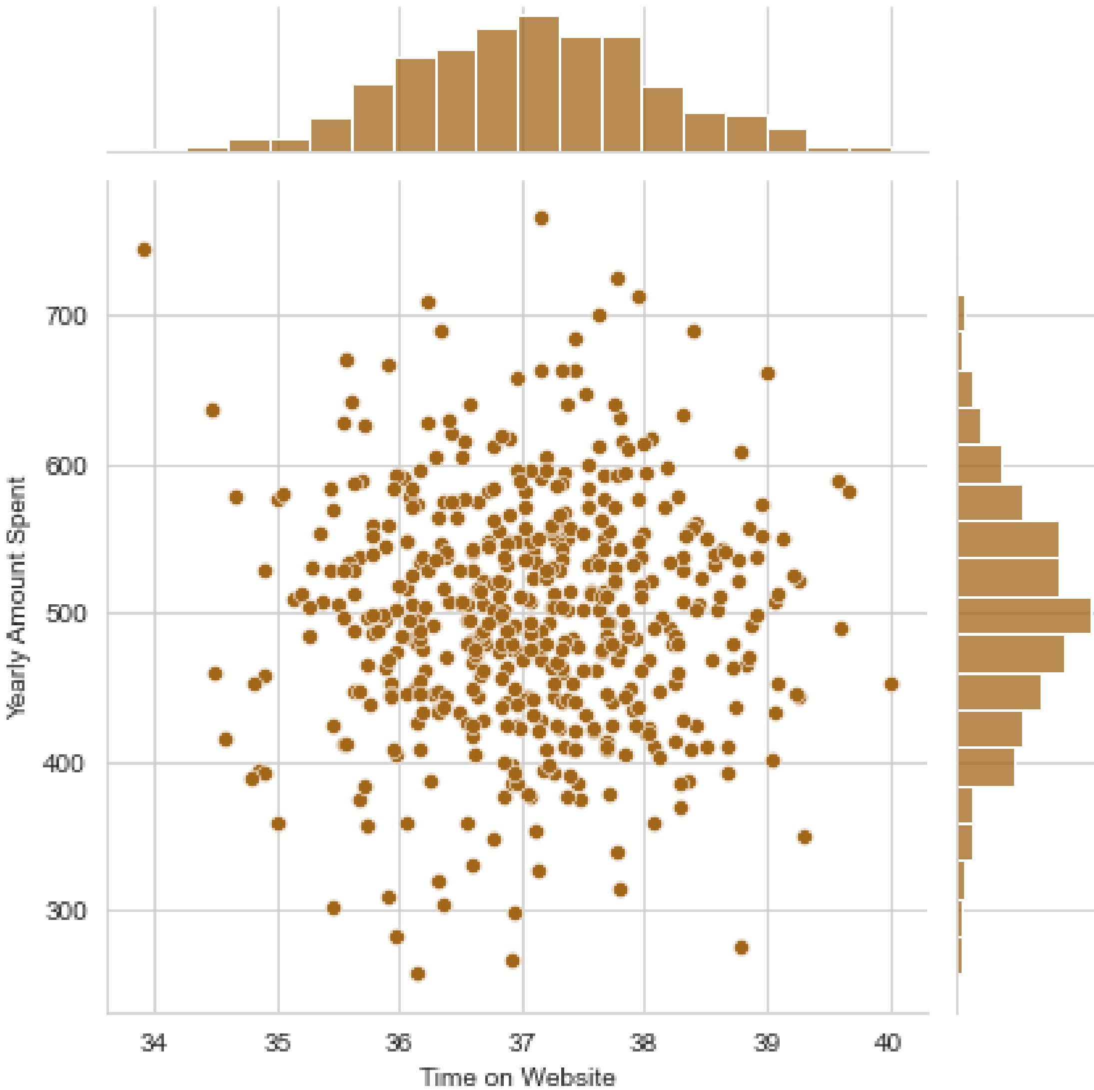
[9]:
```
<seaborn.axisgrid.JointGrid at 0x1a426a38070>
```




```
[10]: sns.set_palette('BrBG')
```

```
[11]: sns.jointplot(x='Time on Website',y='Yearly Amount Spent',\
                 data=customers,kind='scatter')
```

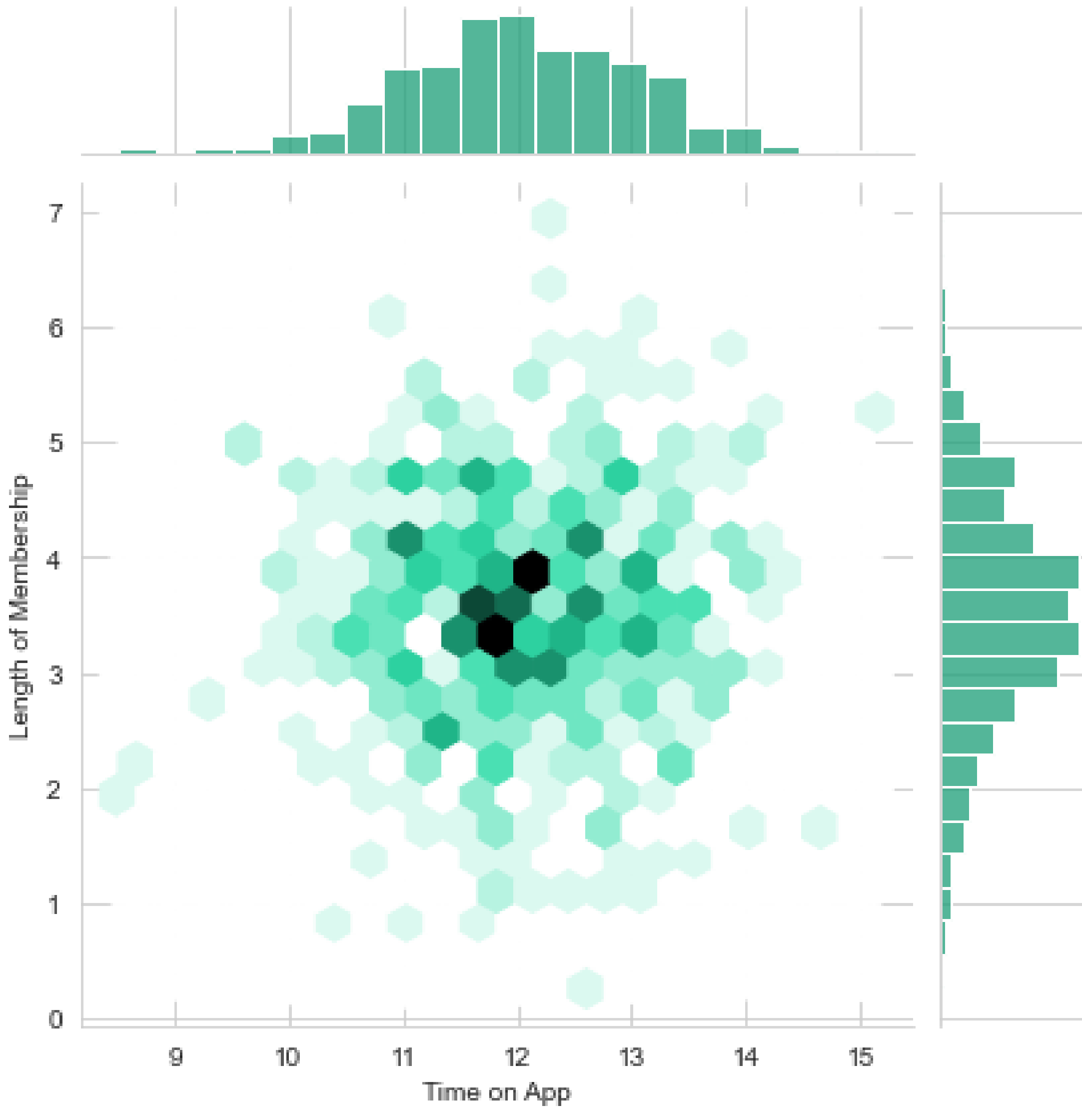
[11]: <seaborn.axisgrid.JointGrid at 0x1a427216640>



[12]:
```
sns.set_palette('Dark2')
```

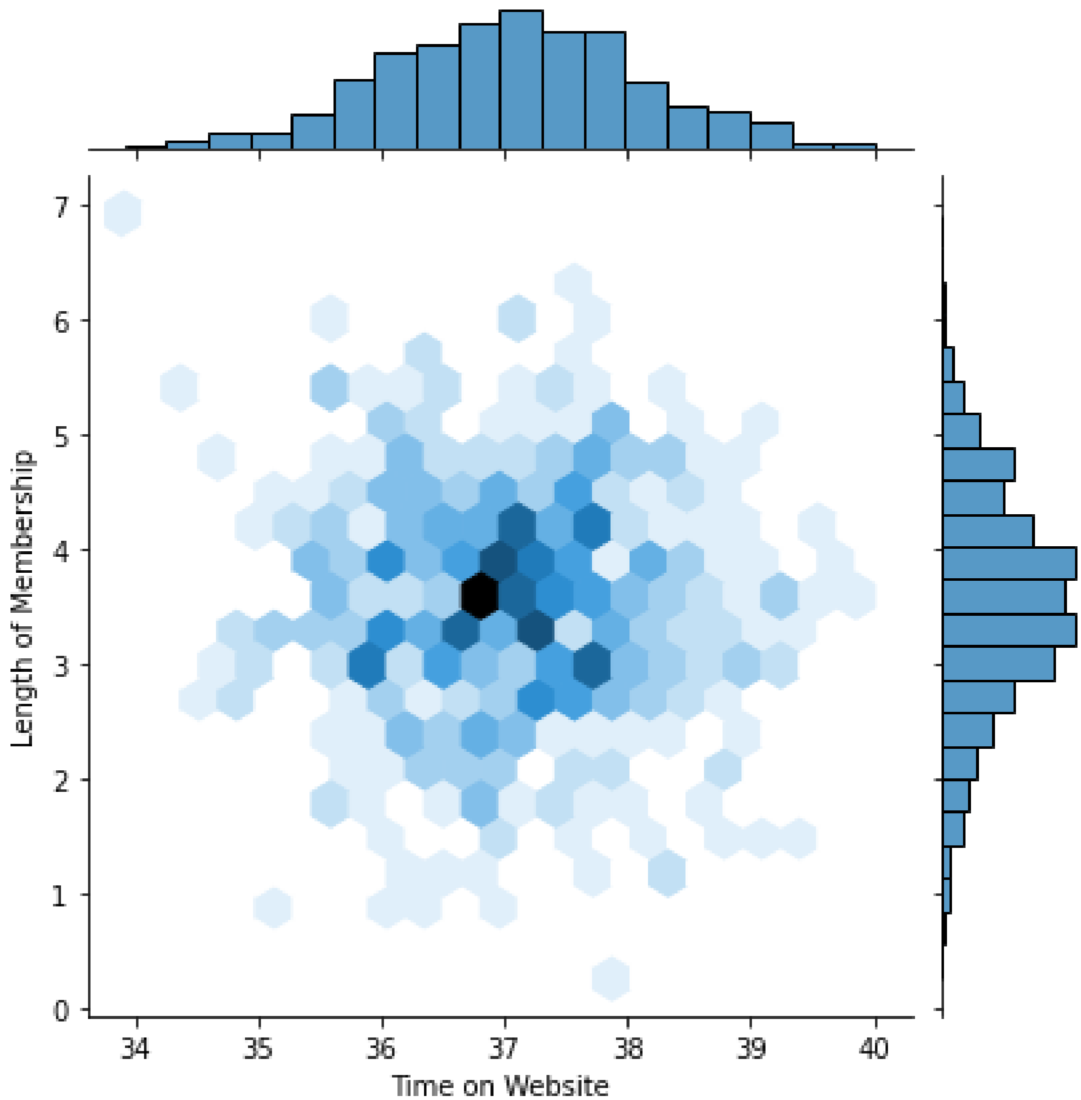
[13]:
```
sns.jointplot(x='Time on App',y='Length of Membership',\
              data=customers,kind='hex')
```

[13]: <seaborn.axisgrid.JointGrid at 0x1a426a38550>



[13]:
```
sns.jointplot(x='Time on Website',y='Length of Membership',\
                data=customers,kind='hex')
```

[13]: <seaborn.axisgrid.JointGrid at 0x15909eb6490>

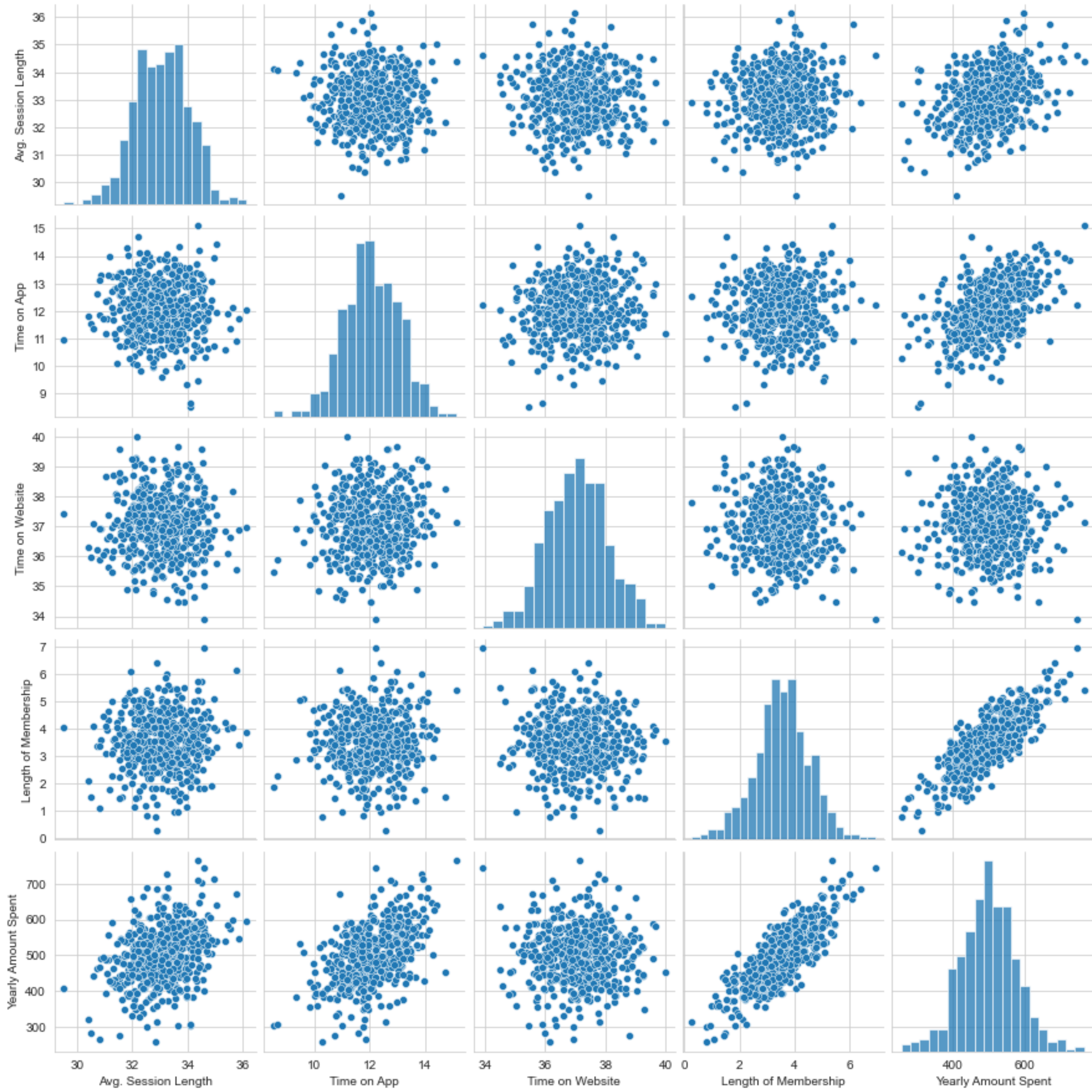


**Let's explore these types of relationships across the entire data set. Will use seaborn's pairplot.

[14]:
```
sns.set_palette('tab10')
```

[15]:
```
sns.pairplot(customers,palette='coolwarm')
```

[15]: <seaborn.axisgrid.PairGrid at 0x1a427413c70>



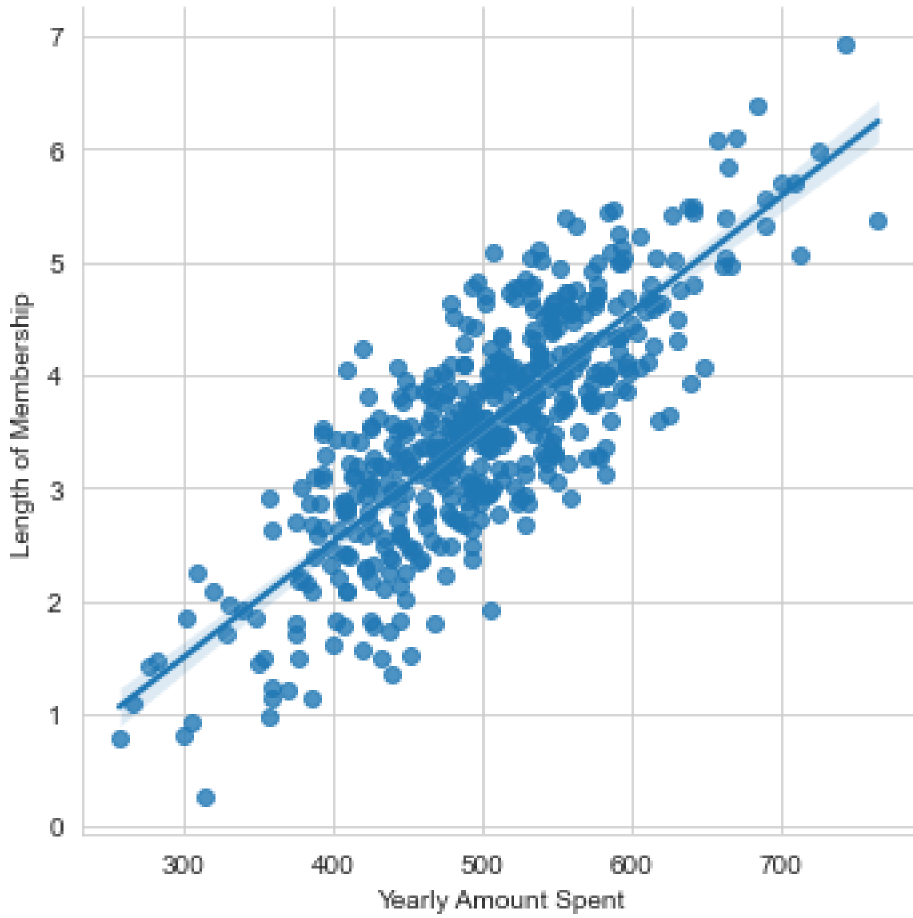
Based off this plot what looks to be the most correlated feature with Yearly Amount Spent is Length Of Membership which is closer to a straight line

Create a linear model plot using seaborn's lmplot of Yearly Amount Spent vs. Length of Membership.

[16]:
```python
print(customers.columns)
sns.lmplot(x='Yearly Amount Spent',y='Length of Membership',data=customers)
```

```
Index(['Email', 'Address', 'Avatar', 'Avg. Session Length', 'Time on App',
       'Time on Website', 'Length of Membership', 'Yearly Amount Spent'],
      dtype='object')
```

[16]:
```
<seaborn.axisgrid.FacetGrid at 0x1a4293d59a0>
```

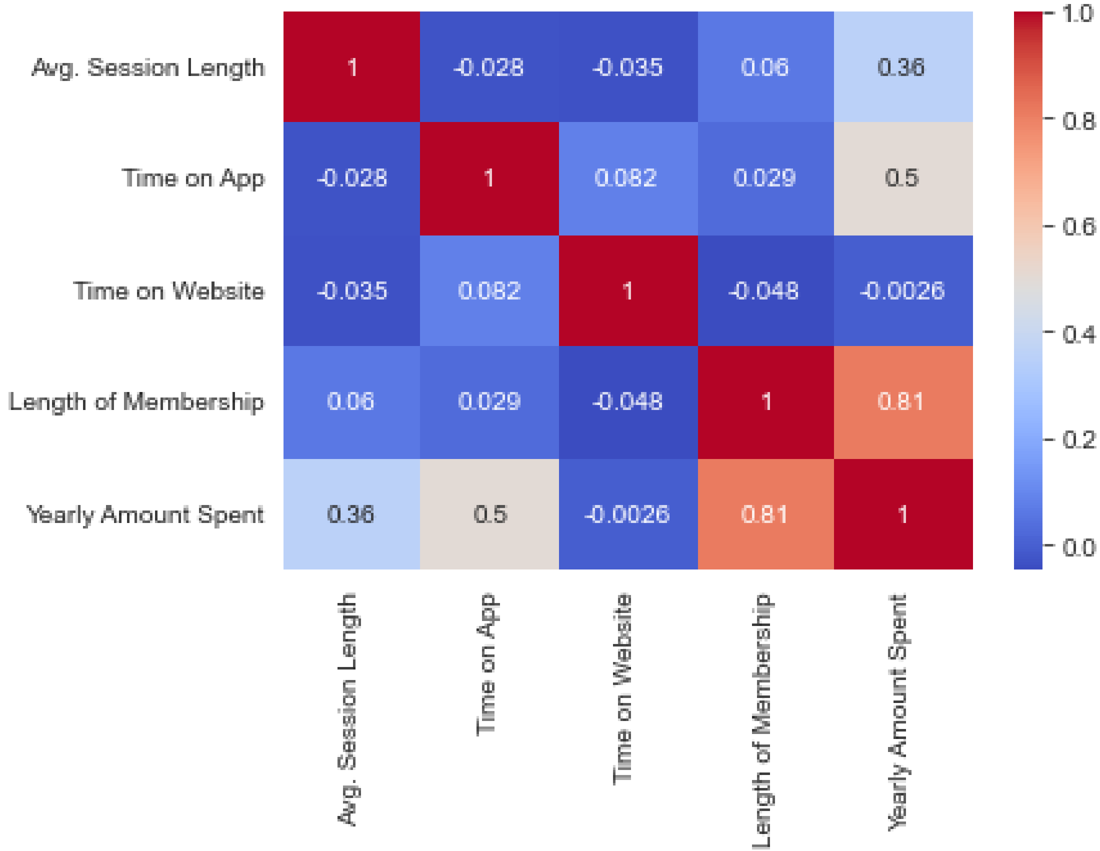


The above plot shows a very good linear fit, as shown by the low error.

Create a correlation matrix.

[17]:
```
sns.heatmap(customers.corr(),cmap='coolwarm',annot=True)
```

[17]: <AxesSubplot:>



We can see that Length Of Membership has a very strong correlation to Yearly Amount Spent, followed by Time On App with a moderate association to Yearly Amount Spent

0.3 Training and Testing Data

Now that we've explored the data a bit, let's go ahead and split the data into training and testing sets. ** Set a variable X equal to the numerical features of the customers and a variable y equal to the "Yearly Amount Spent" column. **

[18]:
```
customers.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
```

[19]:
```
# create features and target datasets
# y would be the target, x are the features (all numeric)
X = customers[['Avg. Session Length','Time on App','Time on Website',\
               'Length of Membership']]
y = customers['Yearly Amount Spent']
```

0.4 Train Test Split

Now let's split the data into a training set and a testing set. We will train out model on the training set and then use the test set to evaluate the model.

2 sets - x_train and y_train, x_test and y_test

[20]:
```
from sklearn.model_selection import train_test_split
```

** Use model_selection.train_test_split from sklearn to split the data into training and testing sets. Set test_size=0.3 and random_state=101**

[21]:
```
# tuple unpacking to populate vars - create train and test data for x and y
# 0.3 is the percentage of test data
X_train, X_test, y_train, y_test = train_test_split\
    (X, y, test_size=0.3, random_state=101)
```

0.5 Training the Model

Now its time to train our model on our training data!

** Import LinearRegression from sklearn.linear_model **

[22]:
```
from sklearn.linear_model import LinearRegression
```

[23]:
```
lm = LinearRegression()
```

[24]:
```
# fit both training data
lm.fit(X_train,y_train)
```

[24]:
```
LinearRegression()
```

0.6 Model Evaluation

Let's evaluate the model by checking out it's coefficients and how we can interpret them

[25]:
```python
# print the intercept
print(lm.intercept_)
```

```
-1047.932782250239
```

[26]:
```python
coeff_df = pd.DataFrame(lm.coef_,X.columns,columns=['Coefficient'])
coeff_df
```

[26]:

	Coefficient
Avg. Session Length	25.981550
Time on App	38.590159
Time on Website	0.190405
Length of Membership	61.279097

Interpreting the coefficients:

Holding all other features fixed,

- a 1 unit increase in Avg. Session Length is associated with an increase of $ 25.98 total spent.
- a 1 unit increase in Time on App is associated with an increase of $ 38.59 total spent.
- a 1 unit increase in Time on Website is associated with an increase of $ 0.19 total spent.
- a 1 unit increase in Length of Membership is associated with an increase of $ 61.27 total spent.

0.7 Predicting Test Data

Now that we have fit our model, let's evaluate its performance by predicting off the test values!

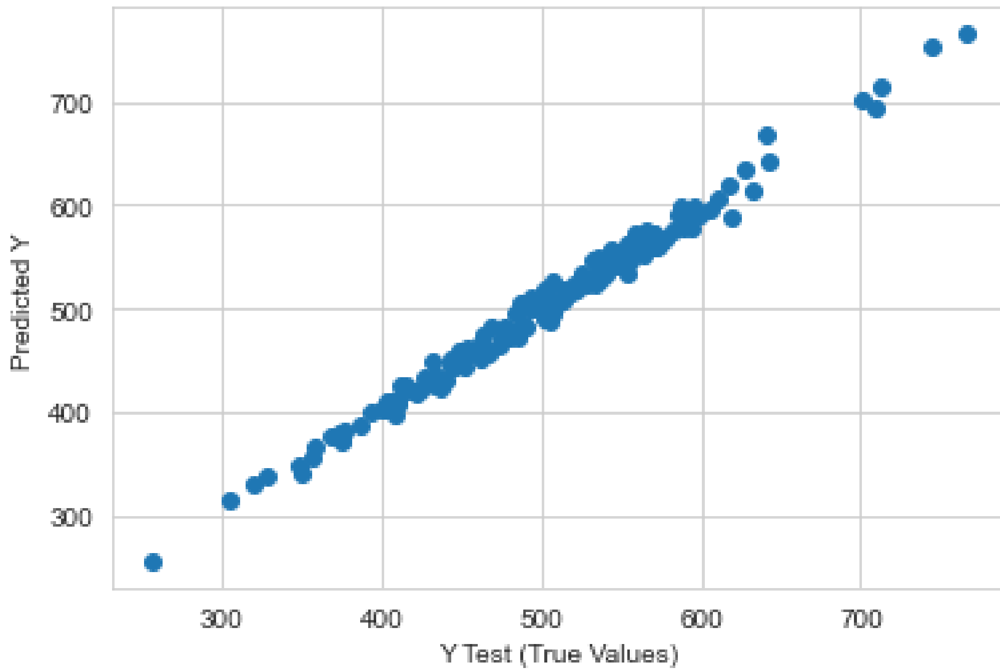
** Use lm.predict() to predict off the X_test set of the data.**

[27]:
```python
predictions = lm.predict(X_test)
```

** Create a scatterplot of the real test values versus the predicted values. **

[37]:
```python
plt.scatter(y_test,predictions)
plt.xlabel('Y Test (True Values)')
plt.ylabel('Predicted Y')
```

[37]:
```
Text(0, 0.5, 'Predicted Y')
```



The plot above shows a very good model, with very little noise and error, considering that we are only using 4 numerical columns.
A perfect model would be a straight line.

0.8 Evaluating the Model

Let's evaluate our model performance by calculating the residual sum of squares and the explained variance score (R^2).

** Calculate the Mean Absolute Error, Mean Squared Error, and the Root Mean Squared Error.**

[29]:
```
from sklearn import metrics
```

[30]:
```
print('MAE:', metrics.mean_absolute_error(y_test, predictions))
print('MSE:', metrics.mean_squared_error(y_test, predictions))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions)))
```

```
MAE: 7.228148653430832
MSE: 79.81305165097444
RMSE: 8.933815066978633
```

[31]:
```
# This shows the varisance of our model, and 98.9% is quite good
metrics.explained_variance_score(y_test,predictions)
```

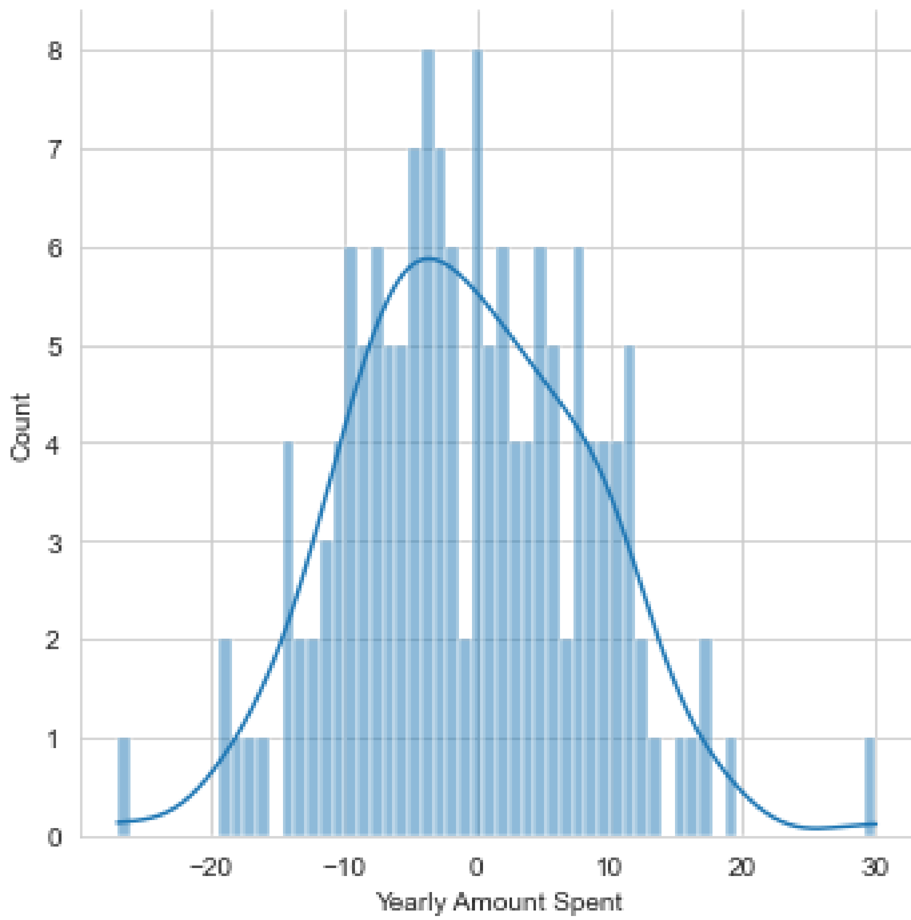
[31]:
```
0.9890771231889607
```

0.9 Residuals

A very good model with a good fit. Let's quickly explore the residuals to make sure everything was okay with our data.

Plot a histogram of the residuals and make sure it looks normally distributed. Use either seaborn distplot, or just plt.hist().

[32]:
```
sns.displot((y_test-predictions),bins=60, kde=True);
# residuals = y_test - predictions
```



0.10 Preconclusion

We still want to figure out the answer to the original question, do we focus our efforts on mobile app or website development? Or maybe that doesn't even really matter, and Membership Time is what is really important. Let's see if we can interpret the coefficients at all to get an idea.

[34]:
```
coeff_df = pd.DataFrame(lm.coef_,X.columns,columns=['Coefficient'])
coeff_df
```

[34]:

```
                      Coefficient
Avg. Session Length     25.981550
Time on App             38.590159
Time on Website          0.190405
Length of Membership    61.279097
```

** How can you interpret these coefficients? **

Interpreting the coefficients:

- Holding all other features fixed, a 1 unit increase in Avg. Session Length is associated with an increase of 25.98 total dollars spent.
- Holding all other features fixed, a 1 unit increase in Time on App is associated with an increase of 38.59 total dollars spent.
- Holding all other features fixed, a 1 unit increase in Time on Website is associated with an increase of 0.19 total dollars spent.
- Holding all other features fixed, a 1 unit increase in Length of Membership is associated with an increase of 61.27 total dollars spent.

0.11 CONCLUSION

[35]:

```
customers.columns
```

[35]:

```
Index(['Email', 'Address', 'Avatar', 'Avg. Session Length', 'Time on App',
       'Time on Website', 'Length of Membership', 'Yearly Amount Spent'],
      dtype='object')
```

Should the company focus more on their mobile app or on their website?

Association between Time on App and Yearly Amount Spent is obviously strong compared to a poor association between Time on Website and Yearly Amount Spent.

On the other hand, the strongest association is between Length of Membership and Yearly Amount Spent.

My recommendation is to improve the website to attract more memberships as well as attract spending, while focusing on more enhancements for the app, but these enhancements should also be ported to the website as much as possible.

The next analysis could be analyzing customer retention as members. There will be valuable insight there as well.

And of course, the costs. This analysis is without taking into considerations costs of development, and management can decide, given the recommendations.