03-Linear Regression Project - Solutions

March 27, 2022

0.1 Linear Regression - Project Exericse

Project - Company is trying to decide whether to focus their efforts on their mobile app experience or their website.

Task : Analysis between mobile app experience and their website

0.2 Imports

[1]:
```
# EDA

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

#Machine learning
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

0.3 Getting the Data

The Ecommerce Customers data is in csv file from the company. It has Customer info, such as 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.

[2]:
```
customers = pd.read_csv("Ecommerce Customers")
```

Checking the head of customers, info() and describe() methods.

[3]:
```
customers.head()
```

[3]:
```
                            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
```

[4]:

```
customers.describe()
```

[4]:

```
       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
```

[5]:

```
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
```

0.4 Exploratory Data Analysis

Using seaborn to understand :

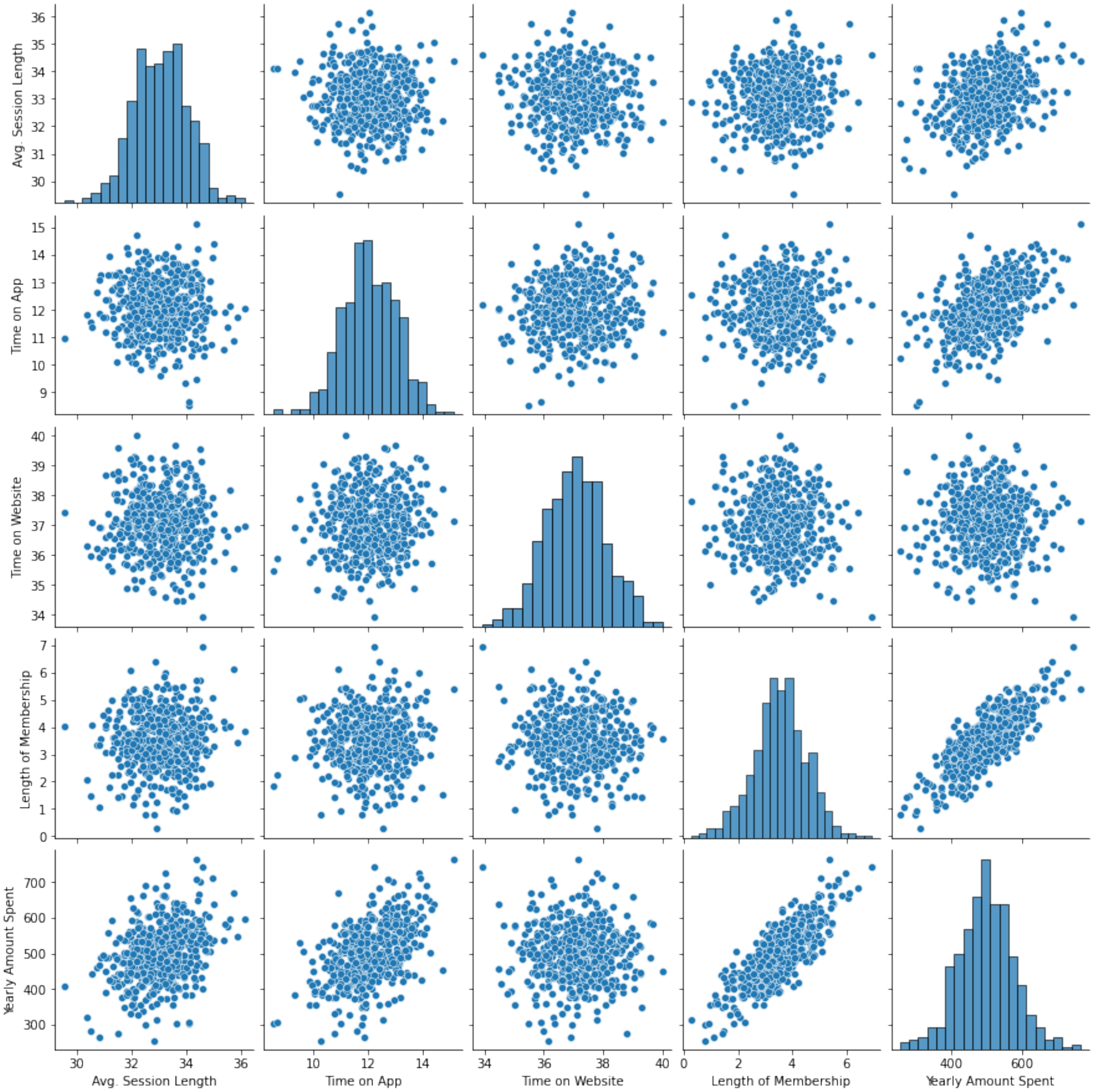
- Relationships across the entire data set
- Time on website and yearly amount spent columns.
- Time on mobile app and yearly amount spent columns.
- comparing Time on App and Length of Membership

[6]:

```
# Relationships across the entire data set
sns.pairplot(customers)
```

[6]:

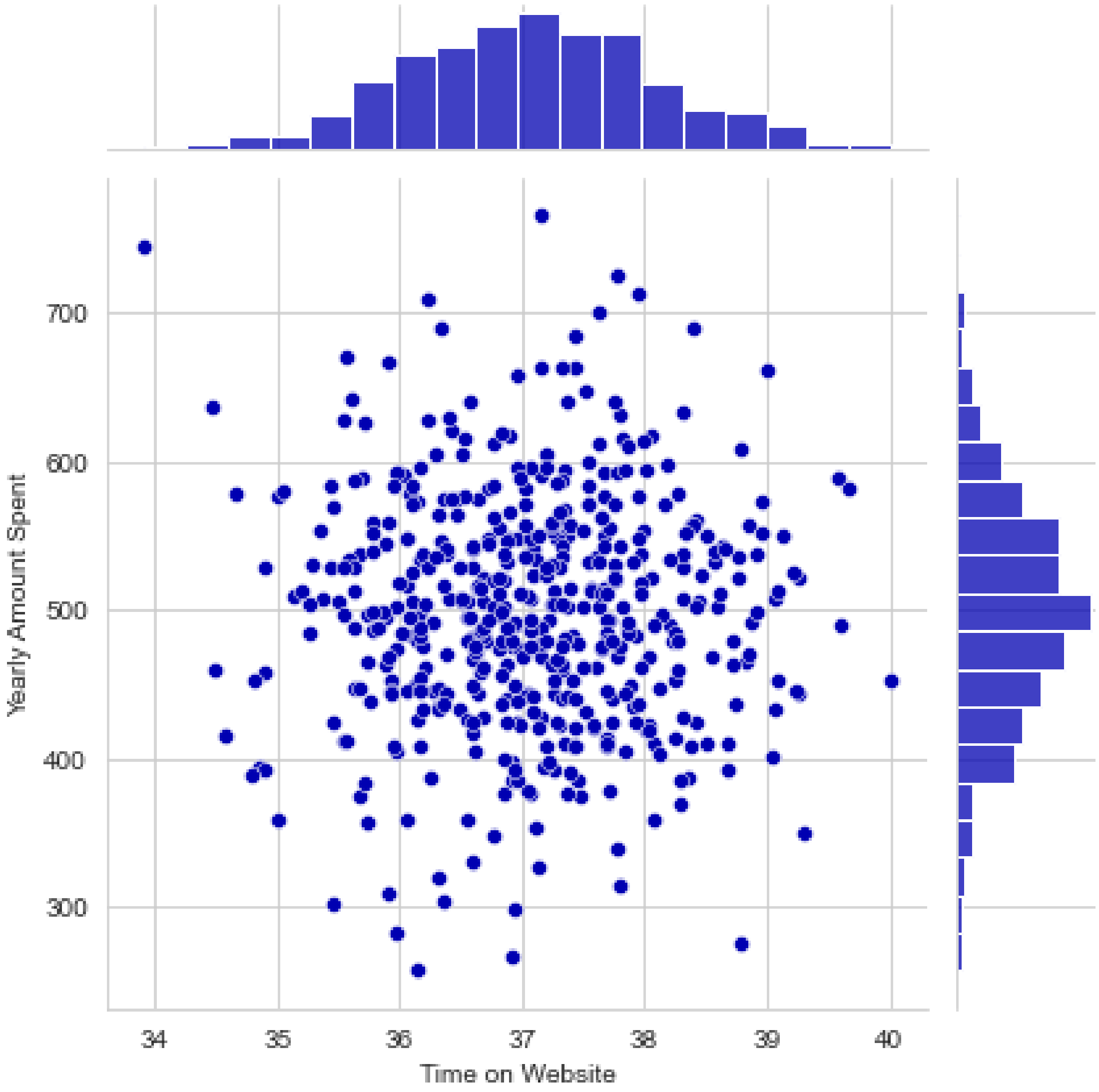
```
<seaborn.axisgrid.PairGrid at 0x16794d50b20>
```



[7]:
```
sns.set_palette("seismic") # color of the graph
sns.set_style('whitegrid') # color of the background
```

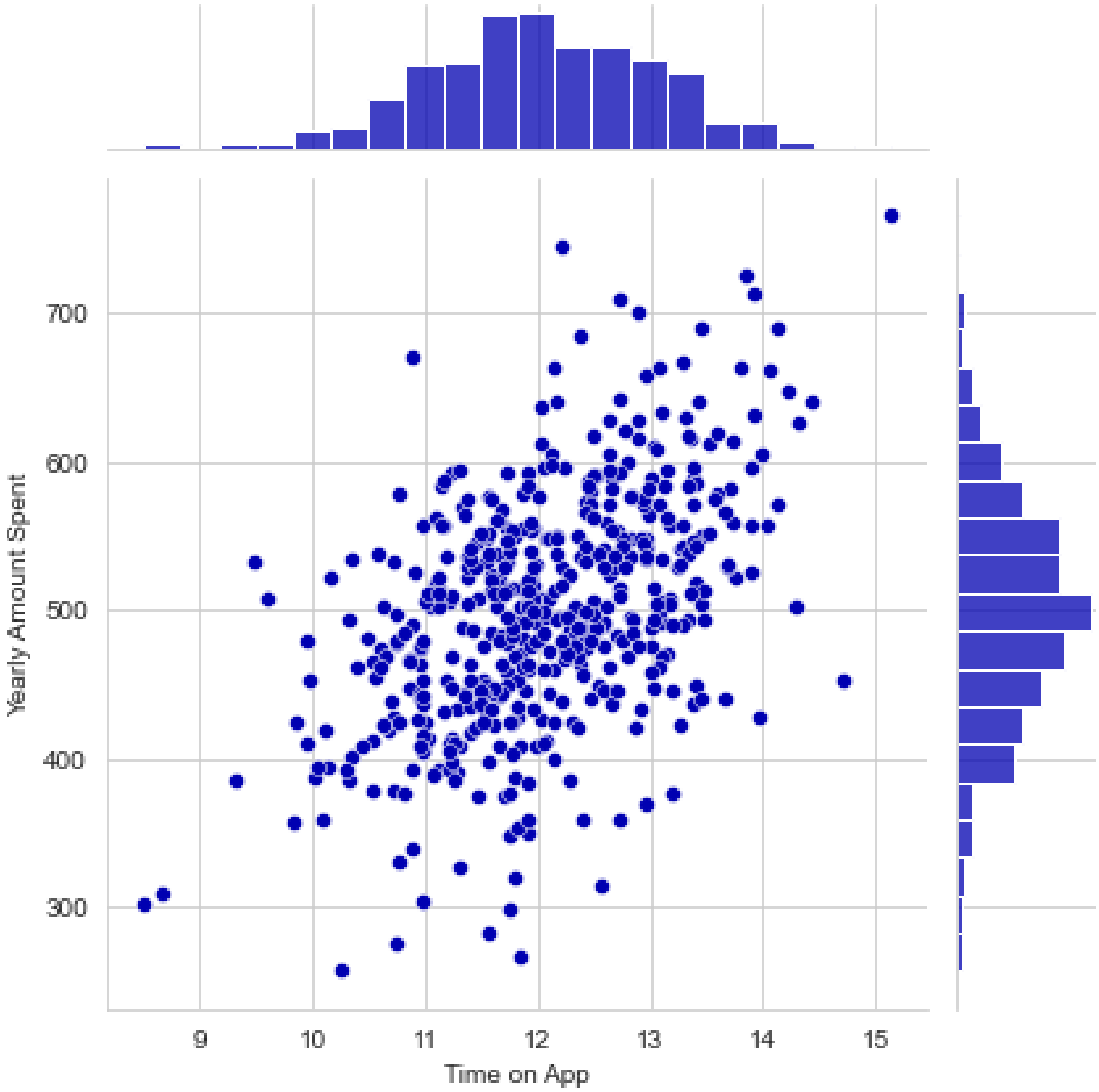
[8]:
```
#Time on website and yearly amount spent columns.

sns.jointplot(x='Time on Website',y='Yearly Amount Spent',data=customers)
```

[8]:
```
<seaborn.axisgrid.JointGrid at 0x167969a81c0>
```



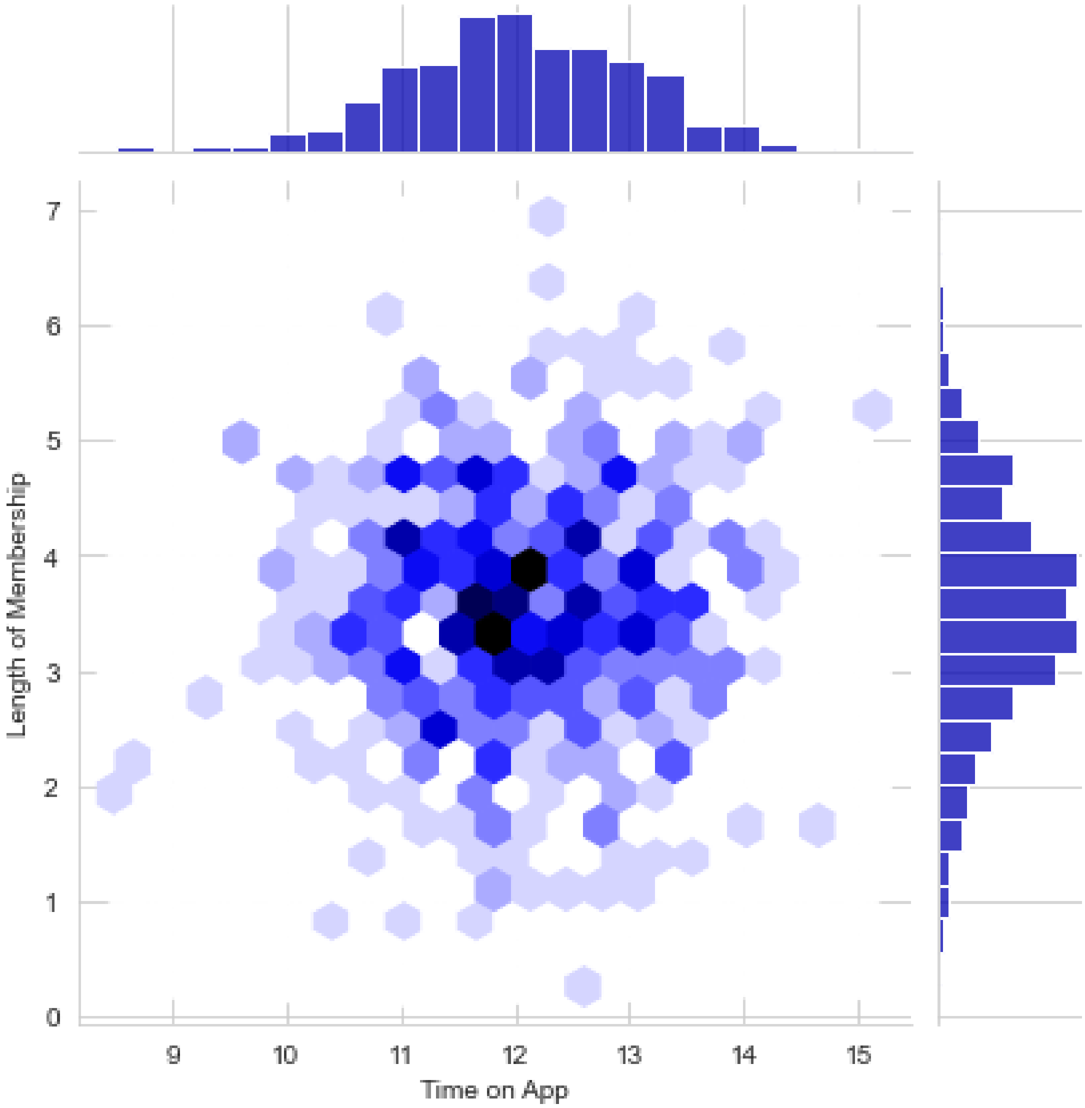
```
[9]: #Time on mobile app and yearly amount spent columns.

     sns.jointplot(x='Time on App',y='Yearly Amount Spent',data=customers)
```

```
[9]: <seaborn.axisgrid.JointGrid at 0x16796732940>
```



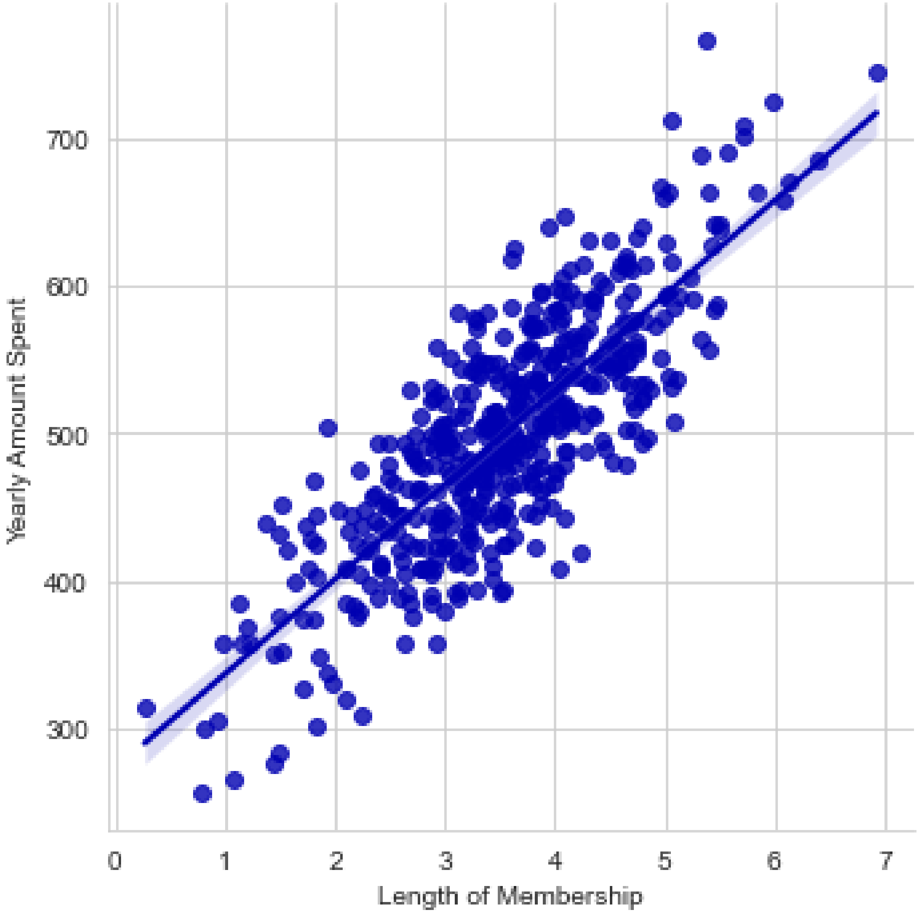
```
[10]: # comparing Time on App and Length of Membership
      sns.jointplot(x='Time on App',y='Length of␣
       ↪Membership',kind='hex',data=customers)
```

```
[10]: <seaborn.axisgrid.JointGrid at 0x16796889550>
```



```
[11]: #linear model plot (using seaborn's lmplot) of  Yearly Amount Spent vs. Length␣
       ↪of Membership.
      sns.lmplot(x='Length of Membership',y='Yearly Amount Spent',data=customers)
```

```
[11]: <seaborn.axisgrid.FacetGrid at 0x167979d9550>
```



0.5 Training and Testing Data

- 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.

[12]:
```
y = customers['Yearly Amount Spent']
```

[13]:
```
X = customers[['Avg. Session Length', 'Time on App','Time on Website', 'Length␣
↪of Membership']]
```

[14]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
↪random_state=101)
```

0.6 Training the Model

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

[15]:
```
lm = LinearRegression()
```

** Train/fit lm on the training data.**

[16]:
```
lm.fit(X_train,y_train)
```

[16]:
```
LinearRegression()
```

Coefficients of the model

[17]:
```
# The coefficients
print('Coefficients: \n', lm.coef_)
```

```
Coefficients:
 [25.98154972 38.59015875  0.19040528 61.27909654]
```

0.7 Predicting Test Data

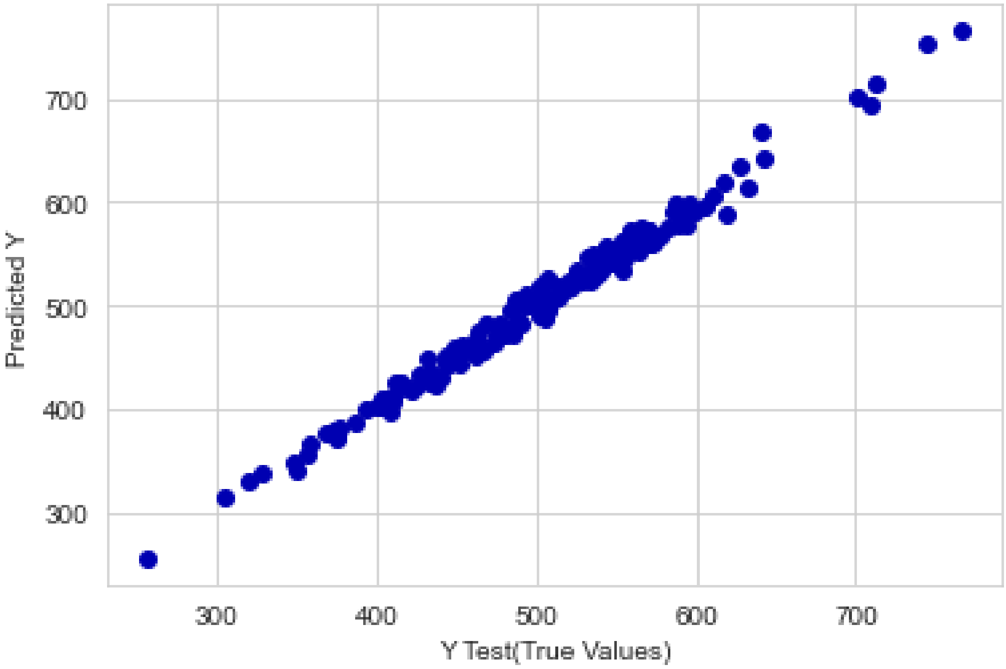
Evaluate its performance by predicting off the test values.

[18]:
```
predictions = lm.predict(X_test)
```

** Scatterplot of the real test values versus the predicted values. **

[19]:
```
plt.scatter(y_test,predictions)
plt.xlabel('Y Test(True Values)')
plt.ylabel('Predicted Y')
```

[19]:
```
Text(0, 0.5, 'Predicted Y')
```



0.8 Evaluating the Model

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

- Calculating the Mean Absolute Error
- Mean Squared Error
- The Root Mean Squared Error

[20]:

```
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: 79.81305165097456
RMSE: 8.93381506697864
```

[21]:

```
metrics.explained_variance_score(y_test, predictions)
```

[21]:

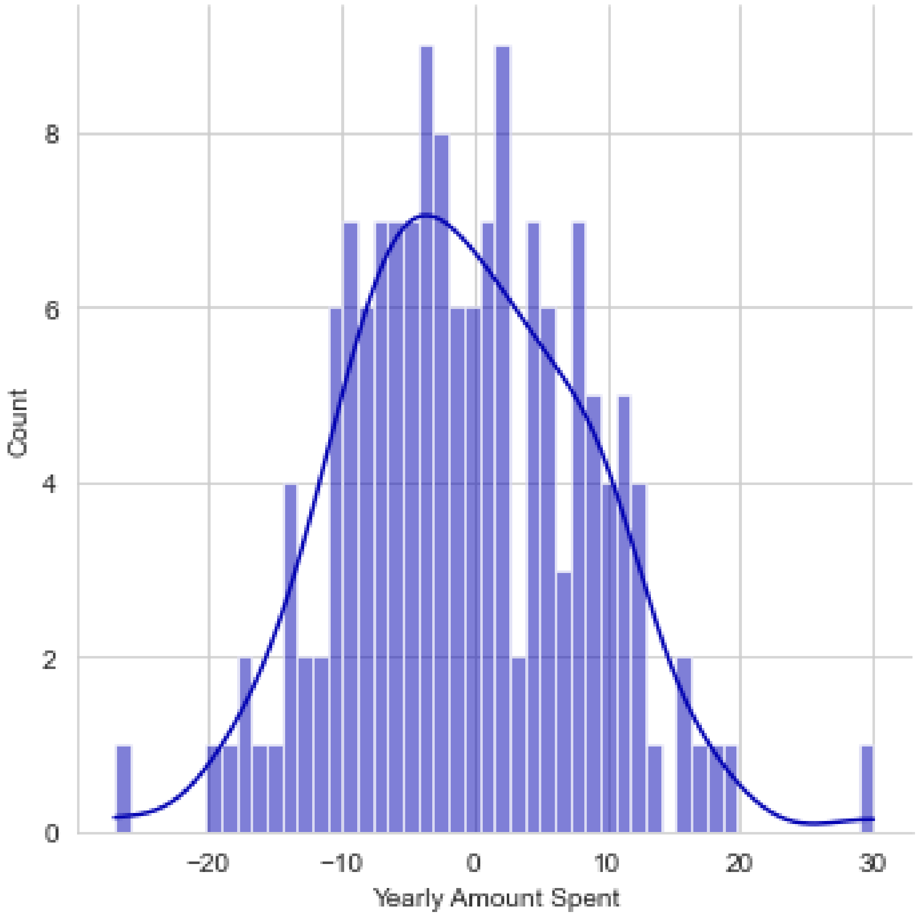
```
0.9890771231889607
```

0.9 Residuals

Explore the residuals to make sure everything was okay with our data.

[23]:

```
#y_test minus prediction
sns.displot((y_test-predictions),bins=50, kde =True);
```



0.10 Conclusion

Membership time is the most important feature.

```
[24]: coeffecients = pd.DataFrame(lm.coef_,X.columns, columns =['coeffecients'])␣
       ↪#columns = ['coeffecients'] adding and renaming it'
      #coeffecients.columns = ['Coeffecient']
      coeffecients
```

```
[24]:                         coeffecients
      Avg. Session Length        25.981550
      Time on App                38.590159
      Time on Website             0.190405
      Length of Membership       61.279097
```

0.10.1 Interpreting these coefficients :

- With all other features fixed, a 1 unit increase in **Avg. Session Length** is associated with an **increase of 25.98 total dollars spent**.

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

Answer to the inital question

There is two ways to go about this :

- Develop the better UI for Website to catch up to the performance of the mobile app.
- Develop the app more since it is working better.
- Finally, would probably want to explore the relationship between Length of Membership and the App or the Website before coming to a conclusion.