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

```
from google.colab import files
uploaded = files.upload()
```

Wybierz pliki Customers.csv

- **Customers.csv**(application/vnd.ms-excel) - 87360 bytes, last modified: 10.01.2019 - 100% done

```
Saving Customers.csv to Customers.csv
```

```
cus = pd.read_csv('Customers.csv')
```

```
#Check the head of customers, and check out its info() and describe() methods.
cus.head()
```

	Email	Address	Avatar	Avg. Session Length	Ti
0	mstephenson@fernandez.com	835 Frank Tunnel\nWrightmouth, MI 82180-9605	Violet	34.497268	12.6
1	hduke@hotmail.com	4547 Archer Common\nDiazchester, CA 06566-8576	DarkGreen	31.926272	11.1
2	pallen@yahoo.com	24645 Valerie Unions Suite 582\nCobbborough, D...	Bisque	33.000915	11.3

```
cus.describe()
```

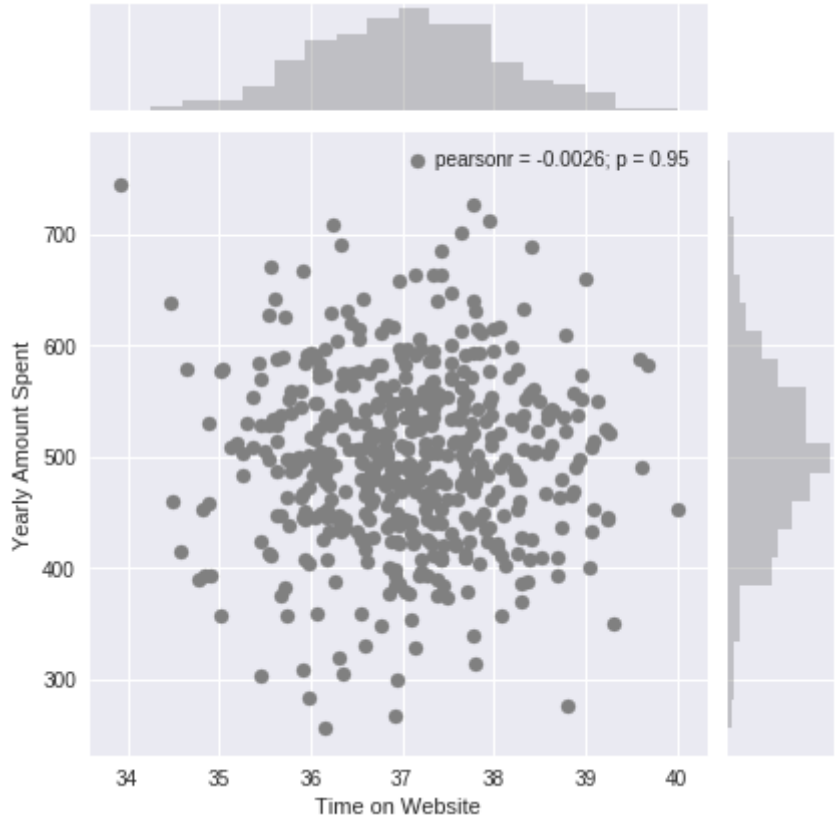
	Avg. Session Length	Time on App	Time on Website	Length of Membership	Yearly Amount Spent
count	500.000000	500.000000	500.000000	500.000000	500.000000
mean	33.053194	12.052488	37.060445	3.533462	499.314038
std	0.992563	0.994216	1.010489	0.999278	79.314782
min	29.532429	8.508152	33.913847	0.269901	256.670582
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

```
cus.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 8 columns):
Email                   500 non-null object
Address                 500 non-null object
Avatar                  500 non-null object
Avg. Session Length     500 non-null float64
Time on App             500 non-null float64
Time on Website         500 non-null float64
Length of Membership    500 non-null float64
Yearly Amount Spent     500 non-null float64
dtypes: float64(5), object(3)
memory usage: 31.3+ KB
```

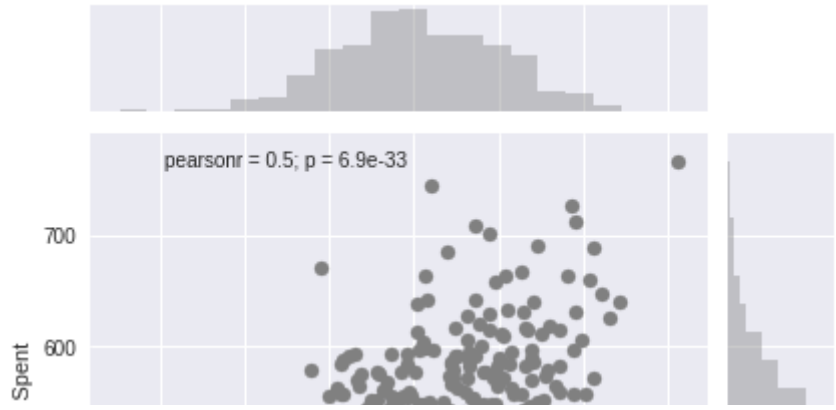
```
#Use seaborn to create a jointplot to compare the Time on Website and Yearly Amount Spent
sns.jointplot(x='Time on Website',y='Yearly Amount Spent', data=cus, color='grey')
#Imo there is no correlation
```

```
<seaborn.axisgrid.JointGrid at 0x7fe1330d4198>
```



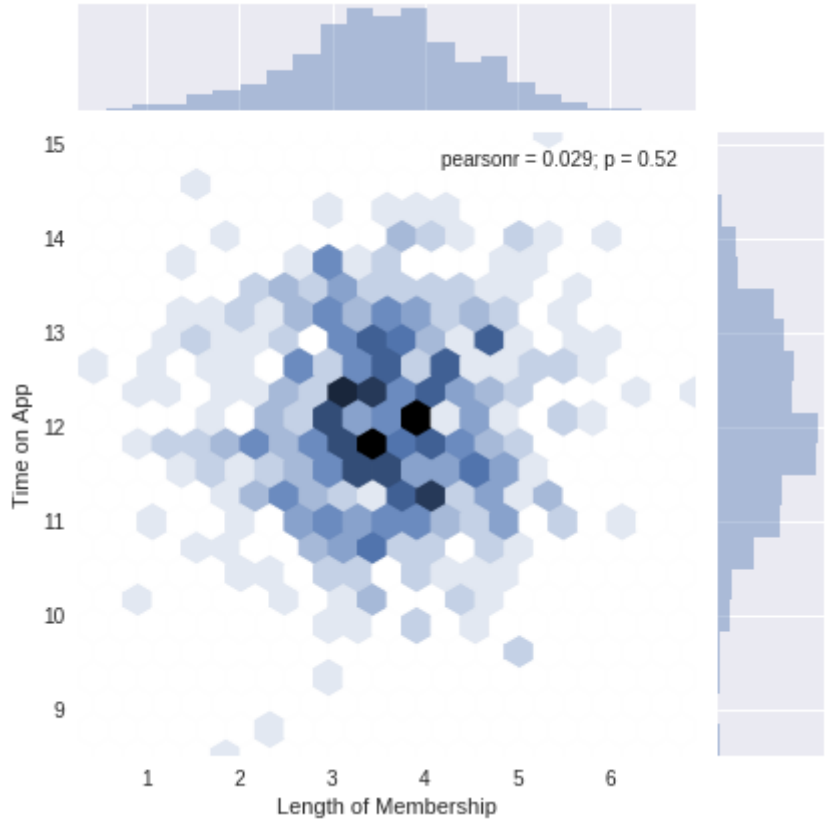

```
#** Do the same but with the Time on App column instead. **
sns.jointplot(x='Time on App', y='Yearly Amount Spent', data=cus, color='grey')
```

```
<seaborn.axisgrid.JointGrid at 0x7f1bc4fc54a8>
```



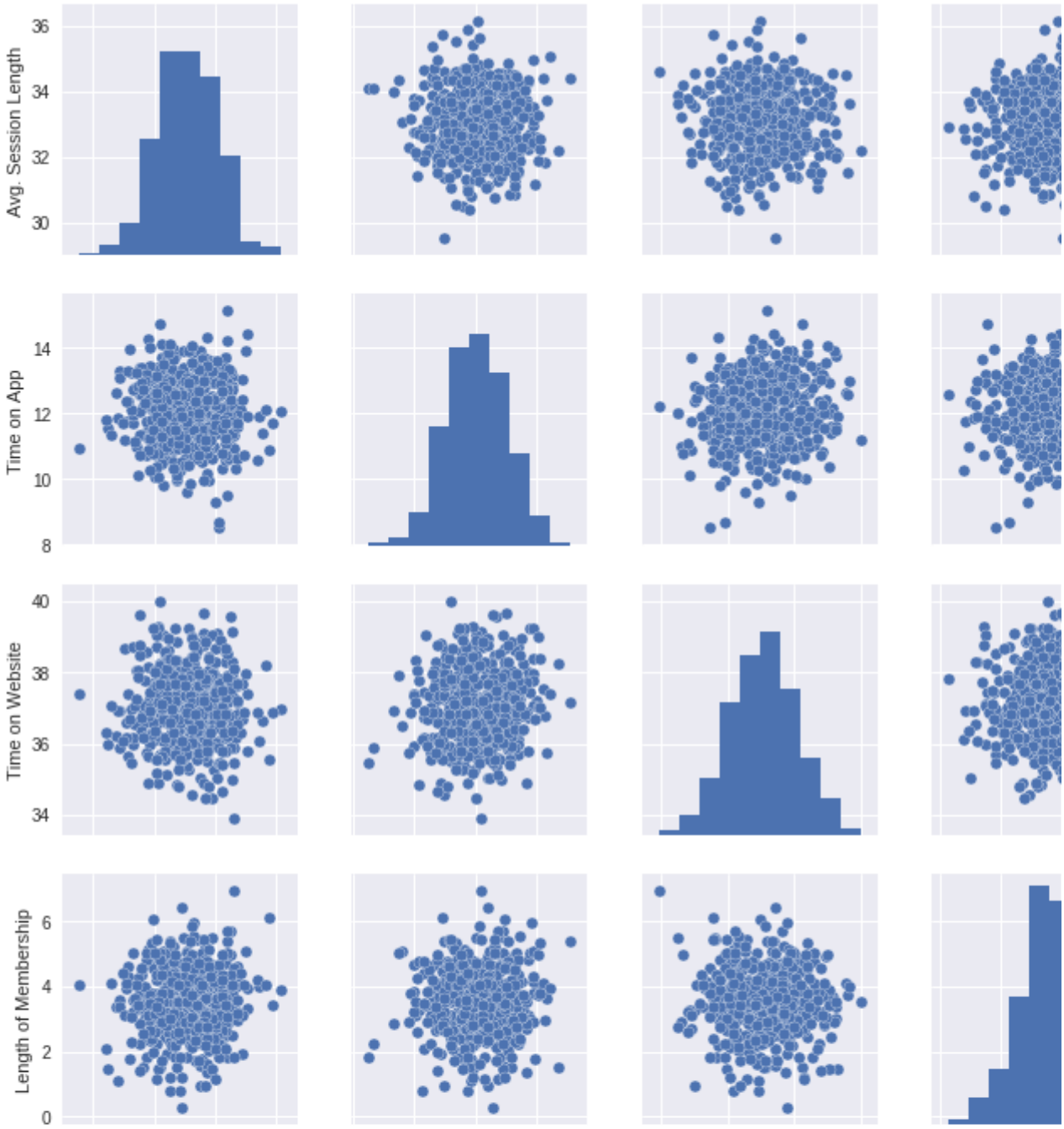

```
#Use jointplot to create a 2D hex bin plot comparing Time on App and Length of Membership.
sns.jointplot(x='Length of Membership', y='Time on App', data=cus, kind="hex")
```

```
<seaborn.axisgrid.JointGrid at 0x7f1bc5158320>
```



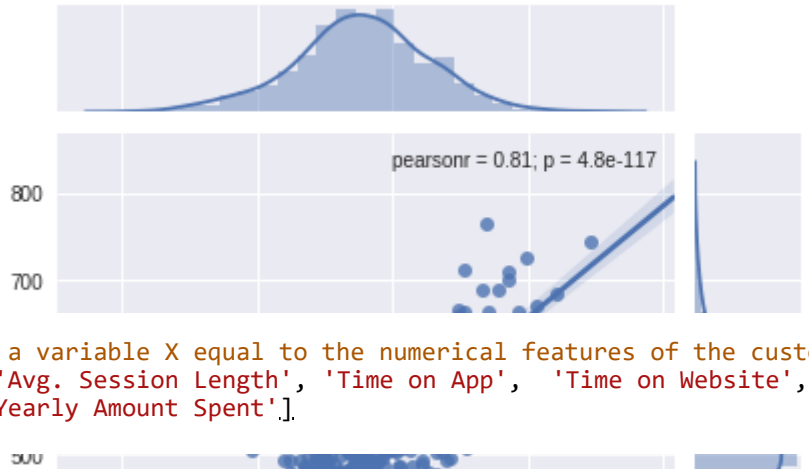

```
#Let’s explore these types of relationships across the entire data set. Use pairplot to re
sns.pairplot(cus)
#Based off this plot what looks to be the most correlated feature with Yearly Amount Spent
#answer: Length of Membership
```

```
<seaborn.axisgrid.PairGrid at 0x7f1bc41c44e0>
```




```
#Based off this plot what looks to be the most correlated feature with Yearly Amount Spent
sns.jointplot(x='Length of Membership', y='Yearly Amount Spent', data=cus, kind="reg")
```

```
<seaborn.axisgrid.JointGrid at 0x7f1bc1976470>
```




```
#** Set a variable X equal to the numerical features of the customers and a variable y equ
X=cus[['Avg. Session Length', 'Time on App',  'Time on Website',  'Length of Membership']]
y=cus['Yearly Amount Spent']
```

```
#** Use model_selection.train_test_split from sklearn to split the data into training and
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3, random_state=101)
```

```
#Create an instance of a LinearRegression() model named lm.
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
#** Train/fit lm on the training data.**
lm.fit(X_train, y_train)
```

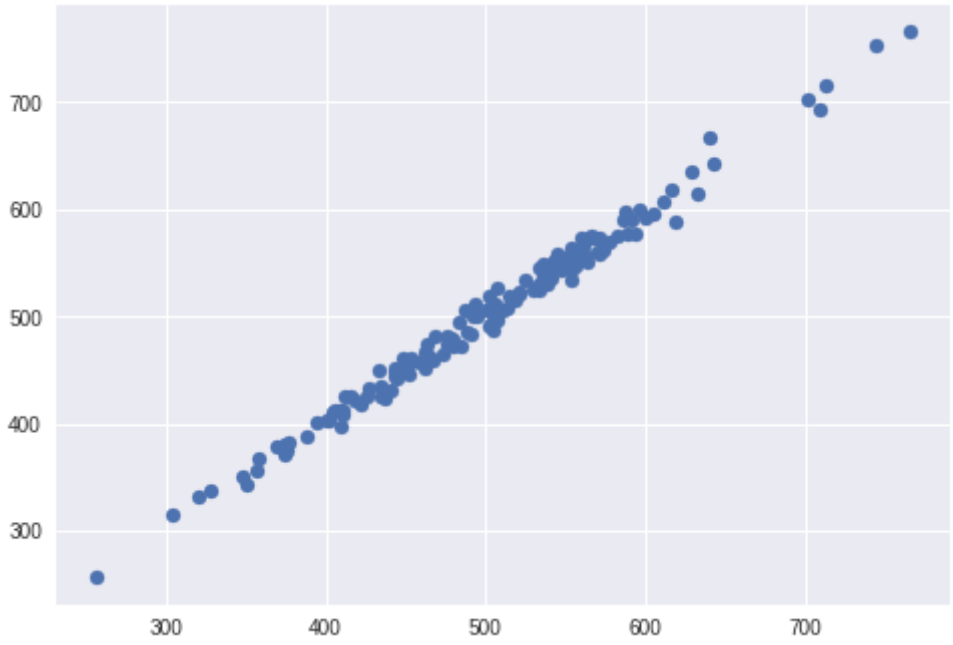
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

```
#Print out the coefficients of the model
lm.coef_
```

```
array([25.98154972, 38.59015875,  0.19040528, 61.27909654])
```

```
#** Use lm.predict() to predict off the X_test set of the data.**
predictions=lm.predict(X_test)
#** Create a scatterplot of the real test values versus the predicted values. **
plt.scatter(y_test, predictions)
```

```
<matplotlib.collections.PathCollection at 0x7fe12ccbe208>
```

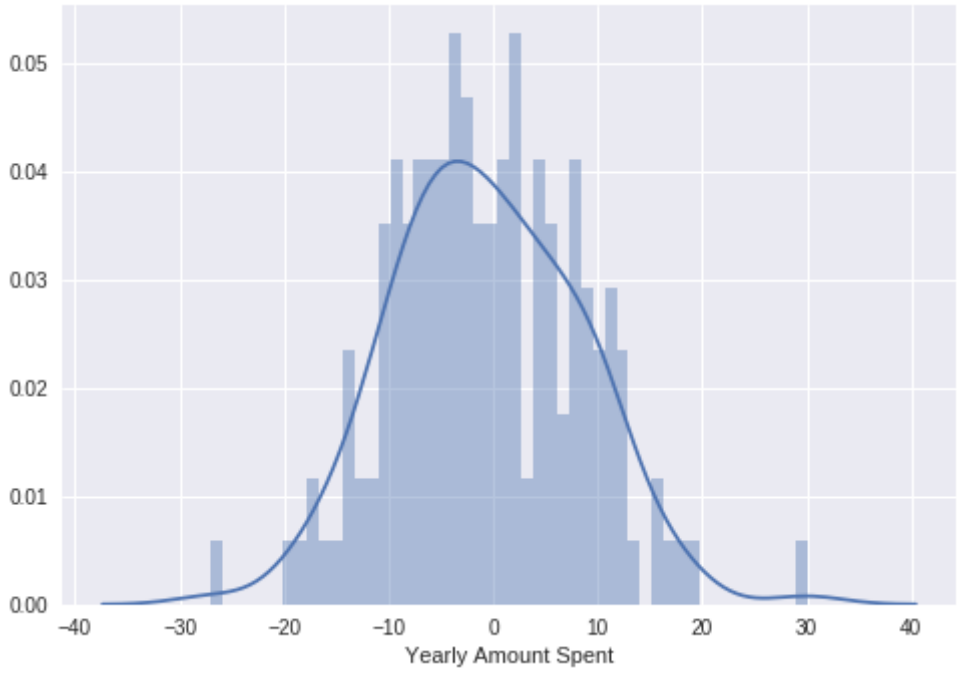


```
#** Calculate the Mean Absolute Error, Mean Squared Error, and the Root Mean Squared Error
from sklearn import metrics
print(f'MAE: {metrics.mean_absolute_error(y_test, predictions)}')
print(f'MSE: {metrics.mean_squared_error(y_test, predictions)}')
print(f'RMSE: {np.sqrt( metrics.mean_squared_error(y_test, predictions))}')
```

```
MAE: 7.2281486534308295
MSE: 79.8130516509743
RMSE: 8.933815066978626
```

```
sns.distplot((y_test - predictions), bins=50)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe12cc8a550>
```




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

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

The length of membership is crucial, but answer to the main question is that company should focus on app.