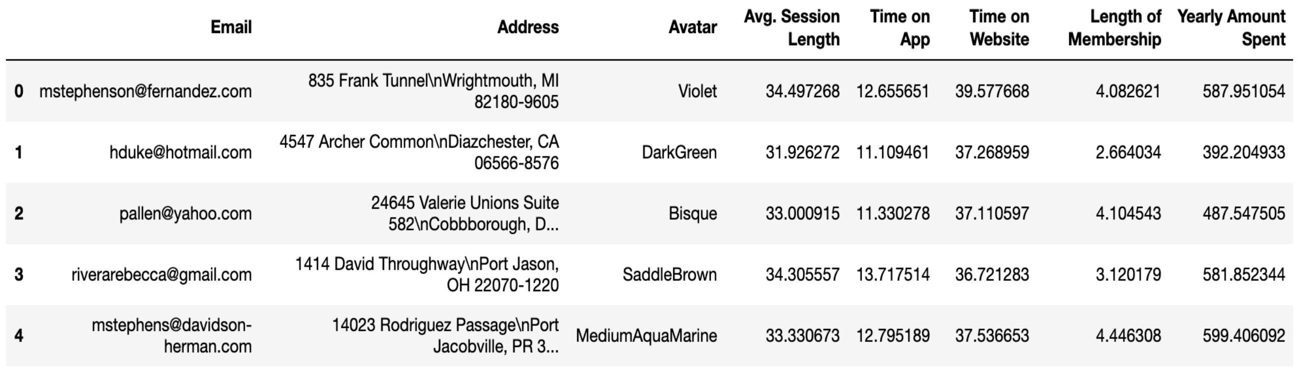
Machine Learning Analysis Report
Name: Rajat Dua

In this folder, I have incorporated some of my projects involving Machine Learning Algorithms such as Linear Regression, Logistic Regression, Neural Network, Natural Language Processing, and Quadratic discriminant analysis. These projects were highly intuitive in making me understand how complex process and analysis could be simply implemented using pre-defined Machine Learning Library. It helps reduce time, decrease human effort, increase productivity and efficiency. These Machine Learning algorithms are also simple to implement once you get theoretical knowledge of it and intuition behind these algorithms. Some of projects which I have included in this folder are as follows:

1. Linear_Regression_Machine Learning.ipynb: In this project, the main goal is to analyze an Ecommerce company based in New York City that sells clothing online, and they also have an in-store style and clothing advice sessions. Customers come into 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.
For this project, I have worked with the database called Ecommerce Customers csv file from the company. It has Customer info, such as Email, Address, and their color Avatar. Then it also has numerical value columns:

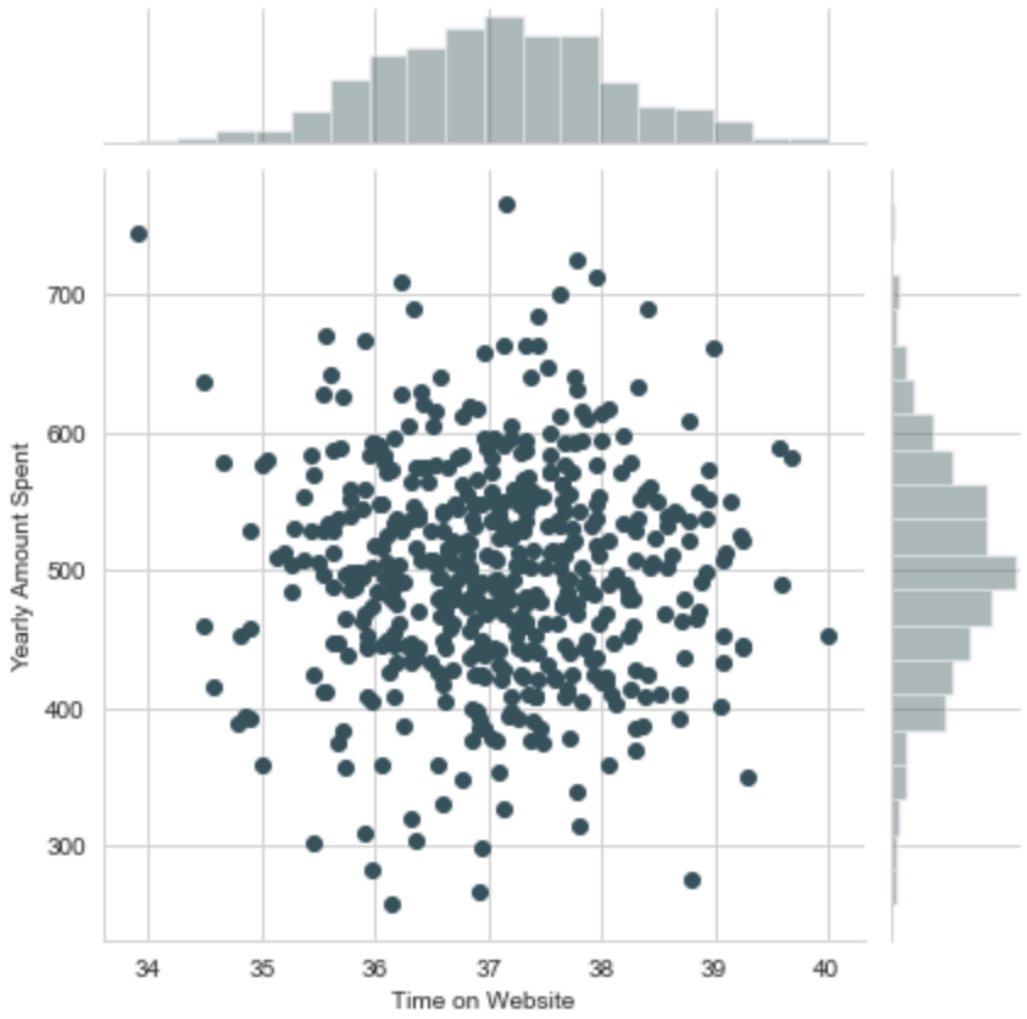
- Avg. Session Length: Average session of in-store style advice sessions.
- Time on App: Average time spent on App in minutes
- Time on Website: Average time spent on Website in minutes
- Length of Membership: How many years the customer has been a member.

The database top 5 rows using the head() function, are as follows:



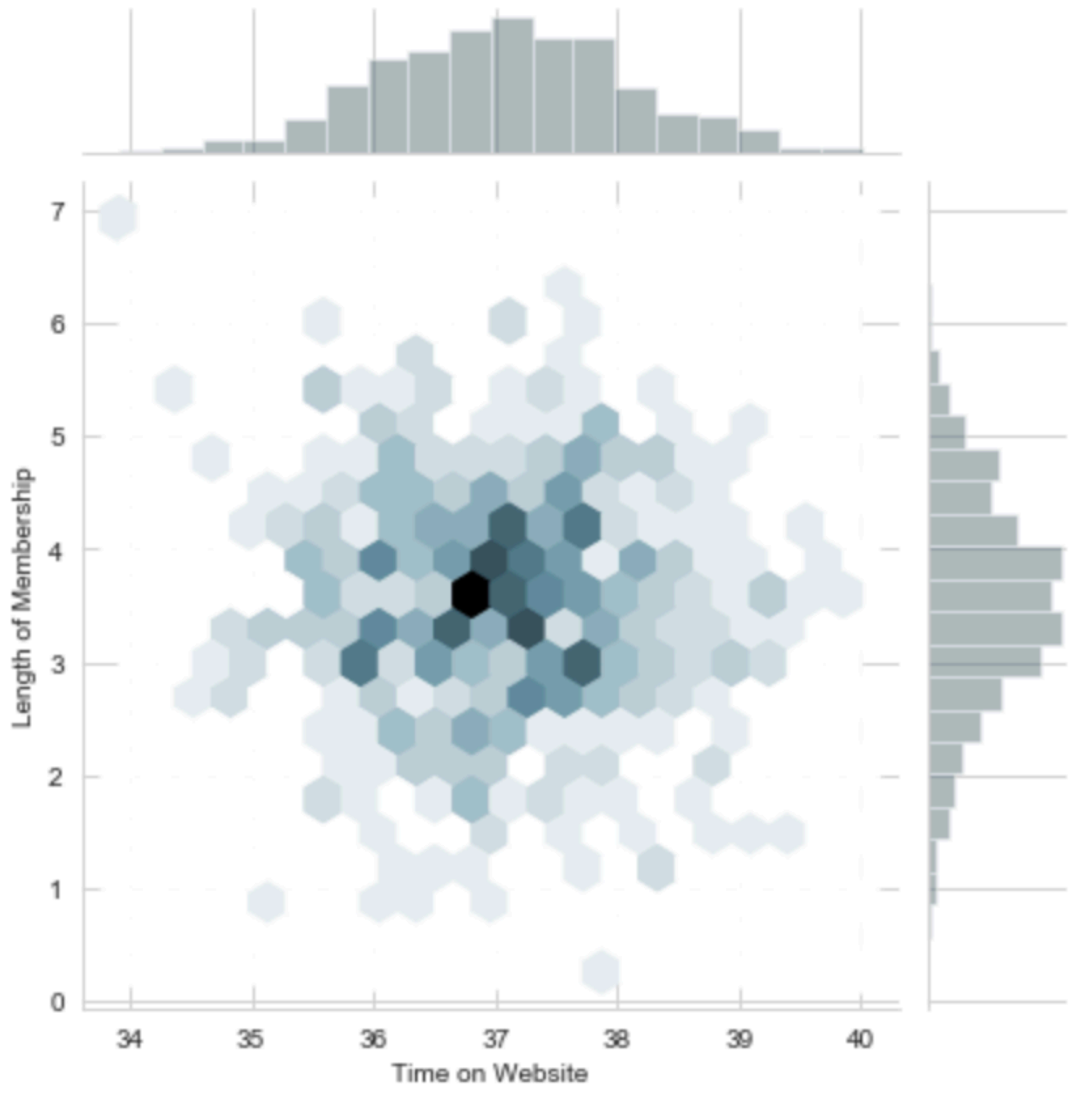
	Email	Address	Avatar	Avg. Session Length	Time on App	Time on Website	Length of Membership	Yearly Amount Spent
0	mstephenson@fernandez.com	835 Frank Tunnel\nWrightmouth, MI 82180-9605	Violet	34.497268	12.655651	39.577668	4.082621	587.951054
1	hduke@hotmail.com	4547 Archer Common\nDiazchester, CA 06566-8576	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

Now, I have explored the data using Exploratory Data Analysis (EDA). For EDA, I have used only the numerical data of the csv file. I have used seaborn to create a jointplot to compare the Time on Website and Yearly Amount Spent columns in order to understand the correlation between these two variables and also to check whether the correlation make sense.

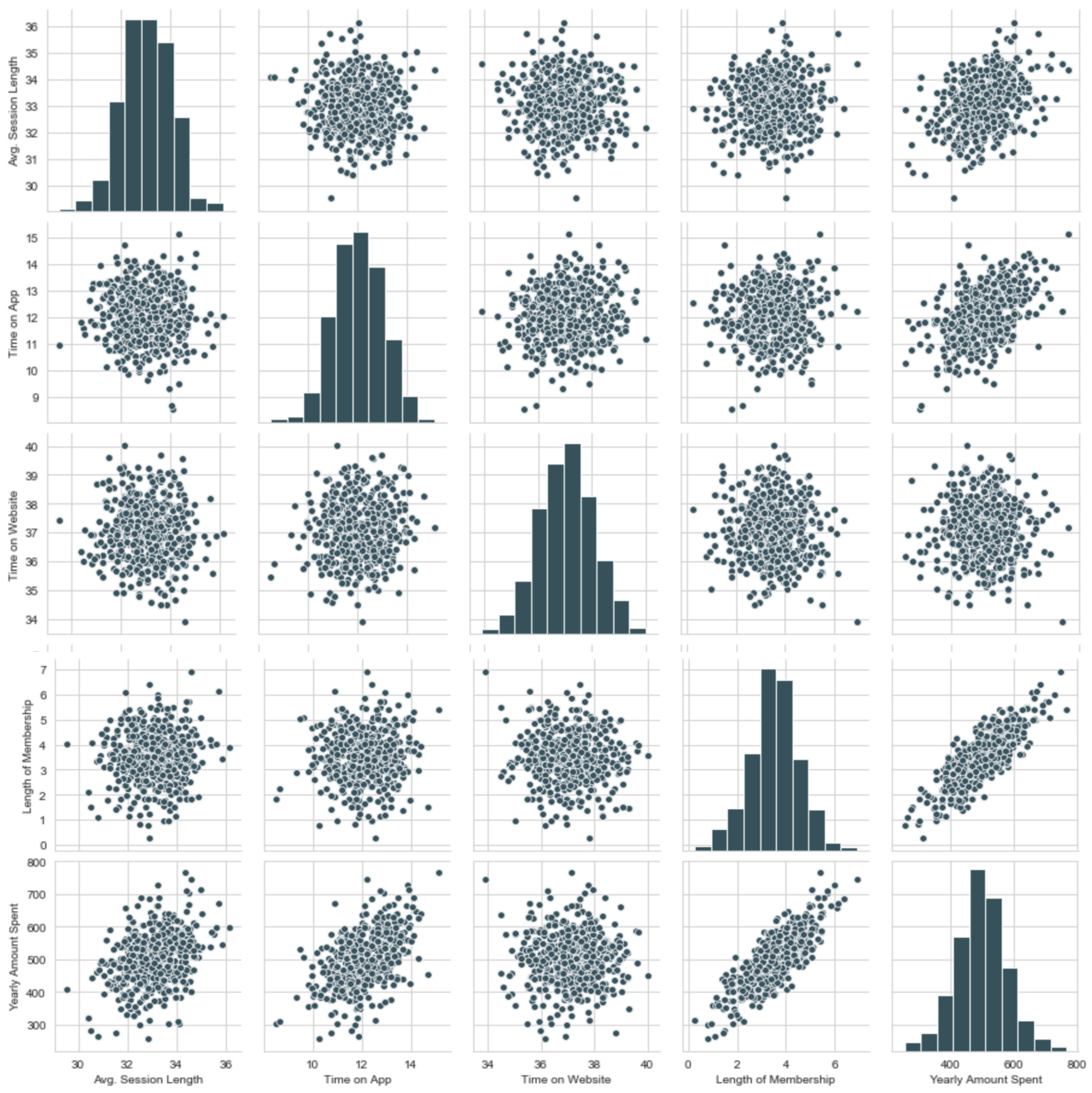


Next, I have used jointplot to create a 2D hex bin plot comparing Time on App and Length of Membership.

```
<seaborn.axisgrid.JointGrid at 0x1a2690b850>
```

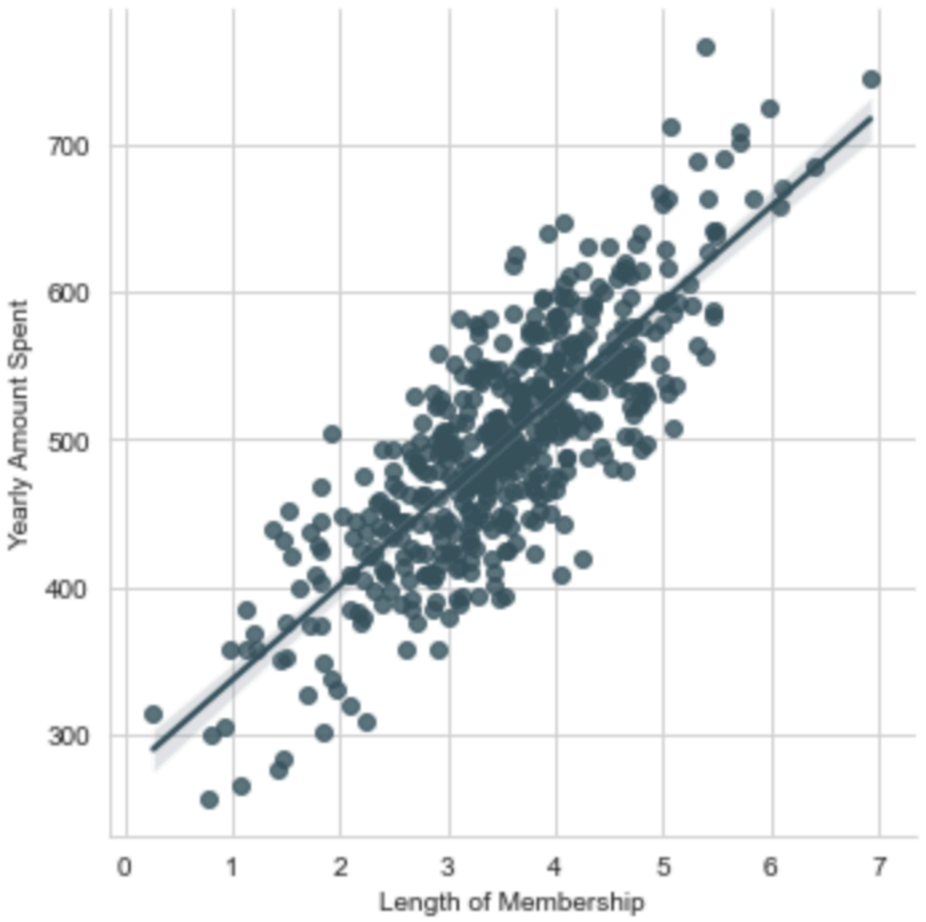


Next, I have explored these types of relationships across the entire data set. I have used pairplot to recreate the plot below.

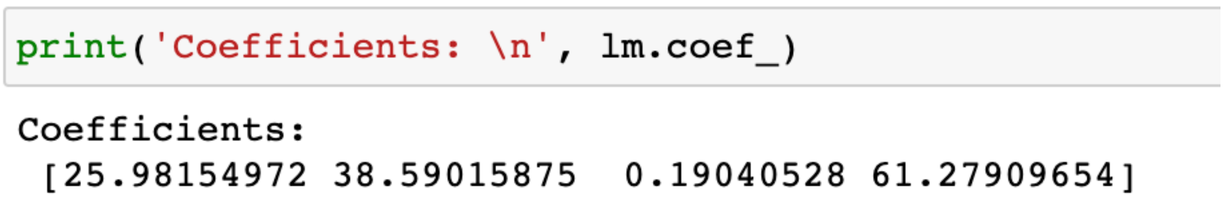


Based off this plot, Length of Membership looks to be the most correlated feature with Yearly Amount Spent. Now, I have created a linear model plot (using seaborn's lmplot) of Yearly Amount Spent vs. Length of Membership.

[illegible]



After I've explored the data a bit, I split the data into training and testing sets. For that, I set a variable X equal to the numerical features of the customers and a variable y equal to the "Yearly Amount Spent" column. I have used model_selection.train_test_split from sklearn library to split the data into training and testing sets and setting test_size=0.3 and random_state=101.
After splitting the data into test and train data, I have trained my Linear Regression Model based on training data and printed out the coefficients as follows:



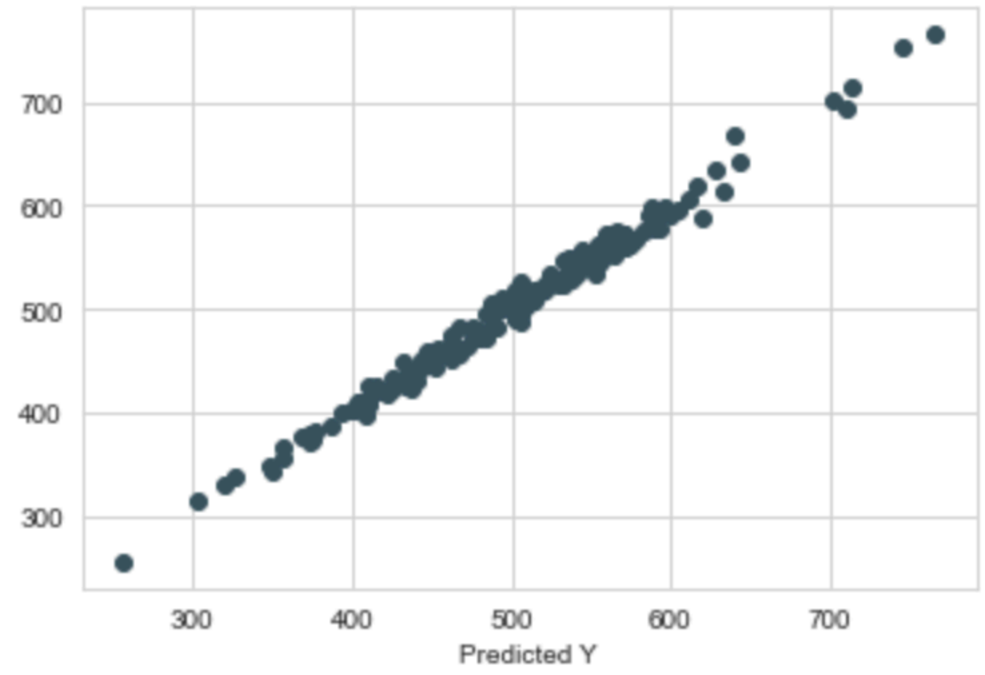
```
print('Coefficients: \n', lm.coef_)
```

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

Now that I have fitted my model using the training data, I evaluated the performance of linear regression model by predicting the test data using lm.predict() to predict off the X_test set of the data. Next, I have compared the test data and prediction by plotting a scatterplot of the real test values versus the predicted values, as follows:

```
plt.scatter(y_test, predictions)
plt.xlabel('Y Test')
plt.xlabel('Predicted Y')
```

```
Text(0.5, 0, 'Predicted Y')
```



Next, I have evaluated the linear regression model's performance by calculating the residual sum of squares and the explained variance score (R^2), as follows:

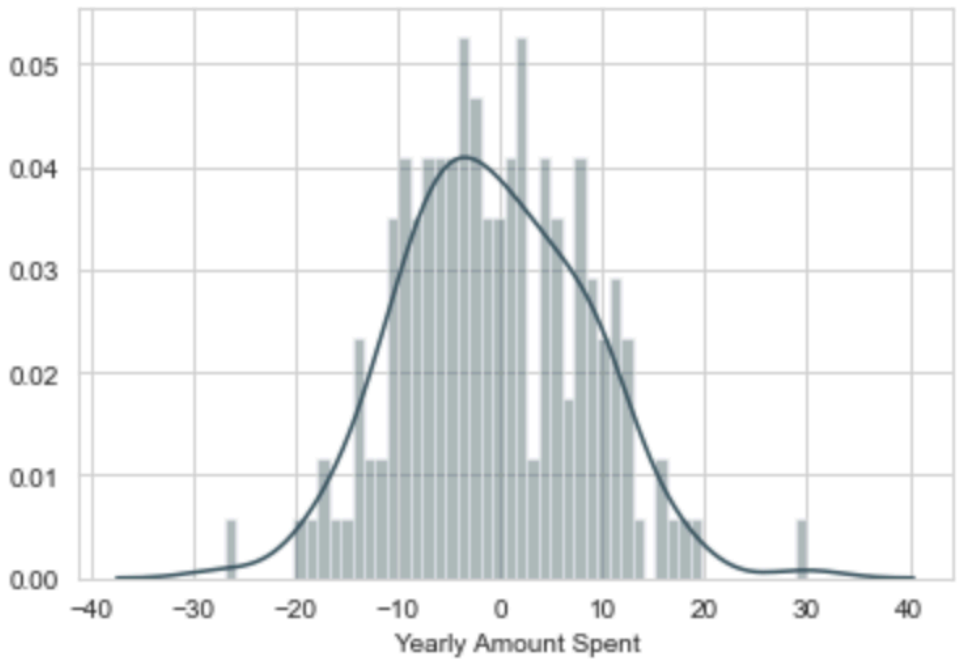
```
from sklearn import metrics
print('MAE:', metrics.mean_absolute_error(y_test, predictions))
print('MSE:', metrics.mean_squared_error(y_test, predictions))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions)))
```

```
MAE: 7.228148653430838
MSE: 79.81305165097461
RMSE: 8.933815066978642
```

Therefore, according to the values of MAE, MSE, and RMSE, the linear regression model does a good job in fitting the test data with the predicted data. So linear regression is a good model with a good fit. Next, I have plotted a histogram of the residuals in order to make sure that it looks normally distributed. I am using seaborn distplot to plot the histogram, as follows:

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

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a2865e790>
```



Finally, in order to determine whether the company should focus our effort on mobile app or website development, I have used the coefficients from linear regression models, as follows:

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

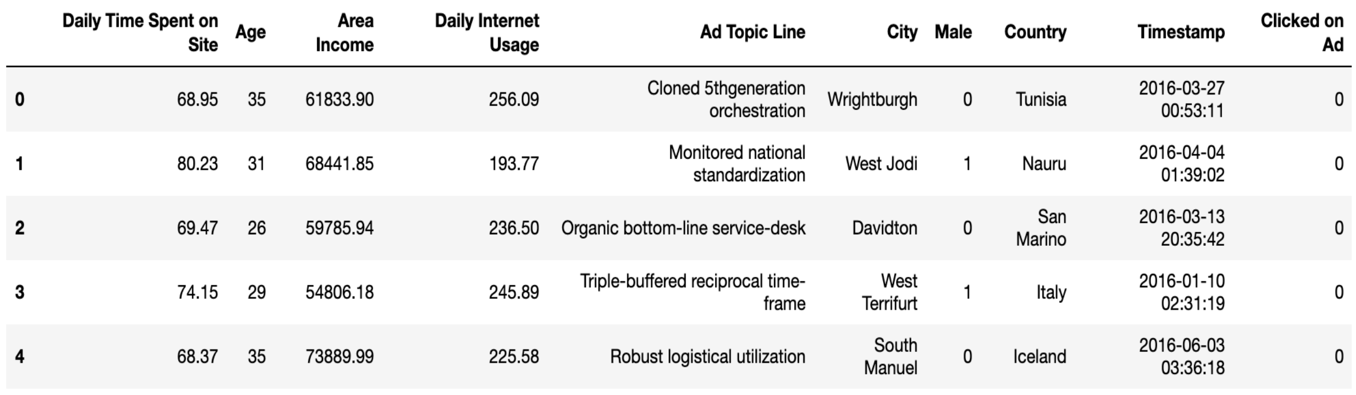
	Coefficient
Avg. Session Length	25.981550
Time on App	38.590159
Time on Website	0.190405
Length of Membership	61.279097

It is clearly evident that Membership Time is really important, and Time on App has a much higher coefficient than the Time on Website. Therefore, the company should focus far more on the mobile application.

2. Logistic_Regression_Machine Learning.ipynb: In this project, I have worked with an advertising data set, indicating whether or not a particular internet user clicked on an Advertisement. The goal is to create a model that will predict whether or not they will click on an ad based off the features of that user. This data set contains the following features:

- 'Daily Time Spent on Site': consumer time on site in minutes
- 'Age': customer age in years
- 'Area Income': Avg. Income of geographical area of consumer
- 'Daily Internet Usage': Avg. minutes a day consumer is on the internet
- 'Ad Topic Line': Headline of the advertisement
- 'City': City of consumer
- 'Male': Whether or not consumer was male
- 'Country': Country of consumer
- 'Timestamp': Time at which consumer clicked on Ad or closed window
- 'Clicked on Ad': 0 or 1 indicated clicking on Ad

The database top 5 rows using the head() function, are as follows:

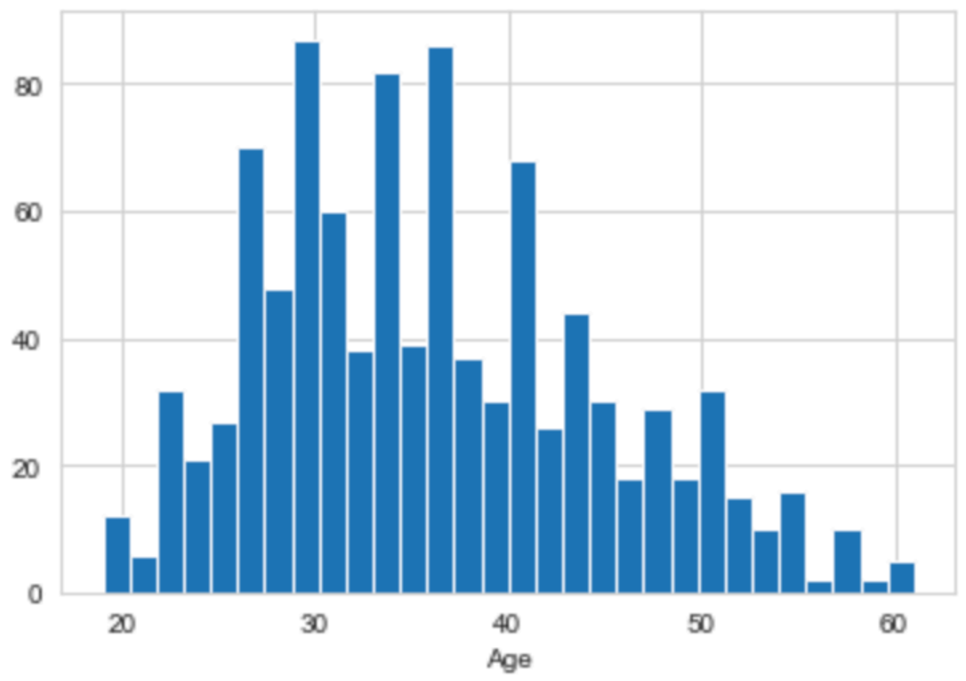


	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad
0	68.95	35	61833.90	256.09	Cloned 5thgeneration orchestration	Wrightburgh	0	Tunisia	2016-03-27 00:53:11	0
1	80.23	31	68441.85	193.77	Monitored national standardization	West Jodi	1	Nauru	2016-04-04 01:39:02	0
2	69.47	26	59785.94	236.50	Organic bottom-line service-desk	Davidton	0	San Marino	2016-03-13 20:35:42	0
3	74.15	29	54806.18	245.89	Triple-buffered reciprocal time-frame	West Terrifurt	1	Italy	2016-01-10 02:31:19	0
4	68.37	35	73889.99	225.58	Robust logistical utilization	South Manuel	0	Iceland	2016-06-03 03:36:18	0

I have explored the data using Exploratory Data Analysis (EDA). First, I have used seaborn library to plot histogram of the Age column, as follows:

```
sns.set_style('whitegrid')
ad_data['Age'].hist(bins = 30)
plt.xlabel('Age')
```

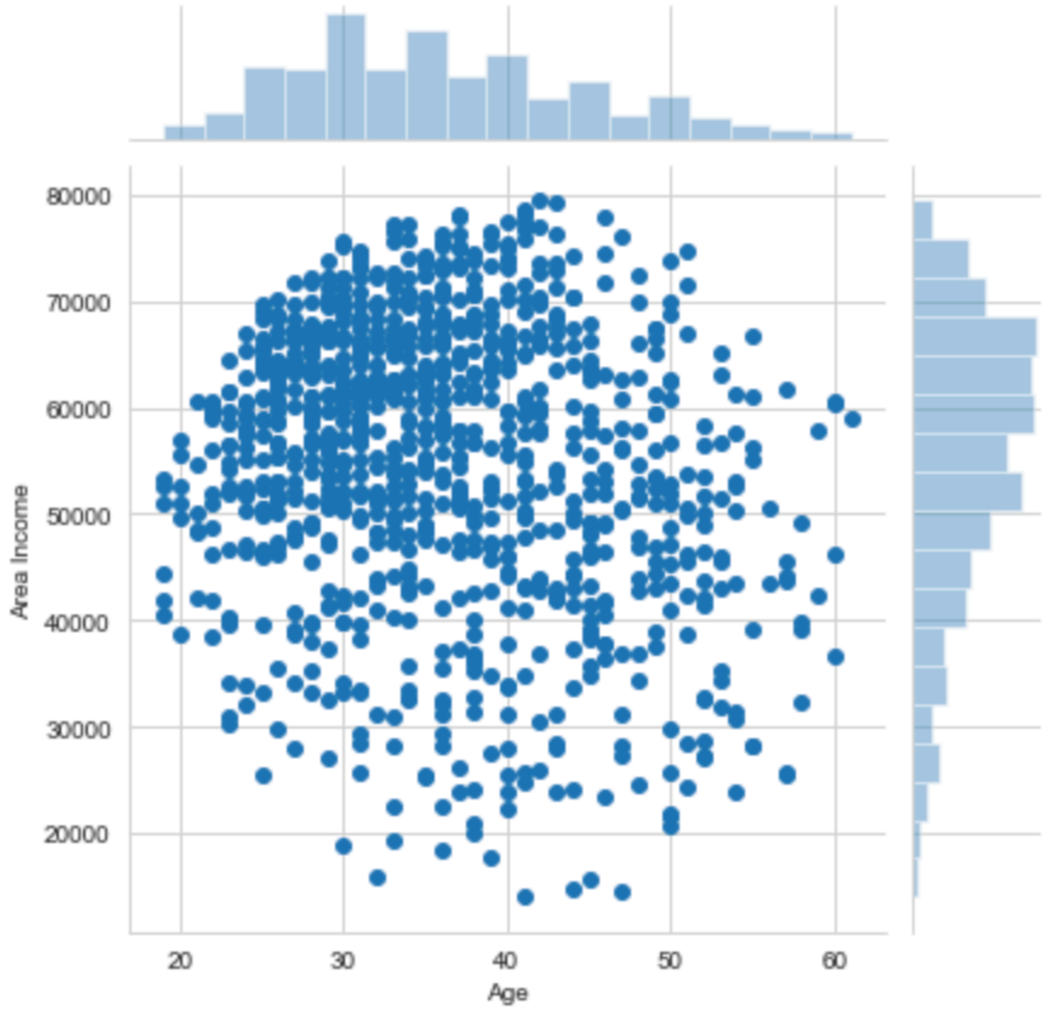
```
Text(0.5, 0, 'Age')
```



Next, I have created a jointplot showing Area Income versus Age, as follows:

```
: sns.jointplot(x = 'Age', y='Area Income', data= ad_data)
```

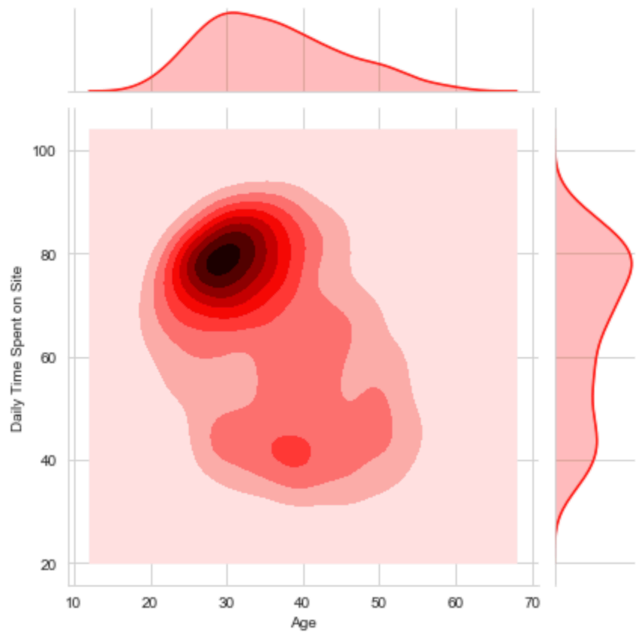
```
: <seaborn.axisgrid.JointGrid at 0x1a23c02a90>
```



After that, I have created a jointplot showing the kde distributions of Daily Time spent on site vs. Age, as follows:

```
: sns.jointplot(x = 'Age', y='Daily Time Spent on Site', data= ad_data, color='red', kind='kde')
```

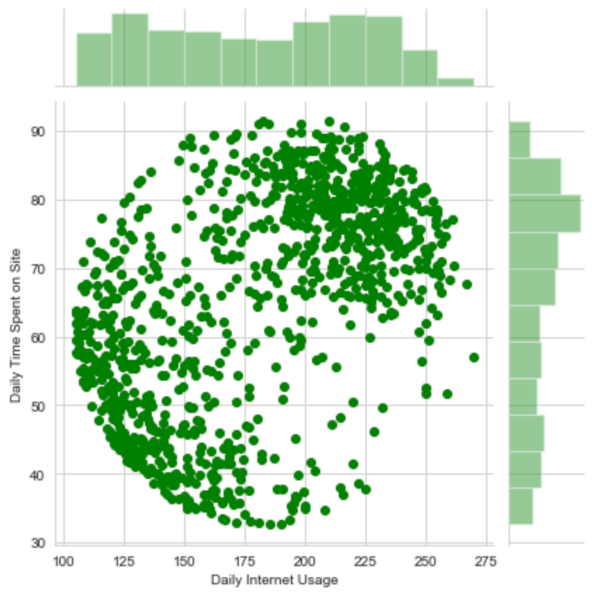
```
: <seaborn.axisgrid.JointGrid at 0x1a242a95f8>
```



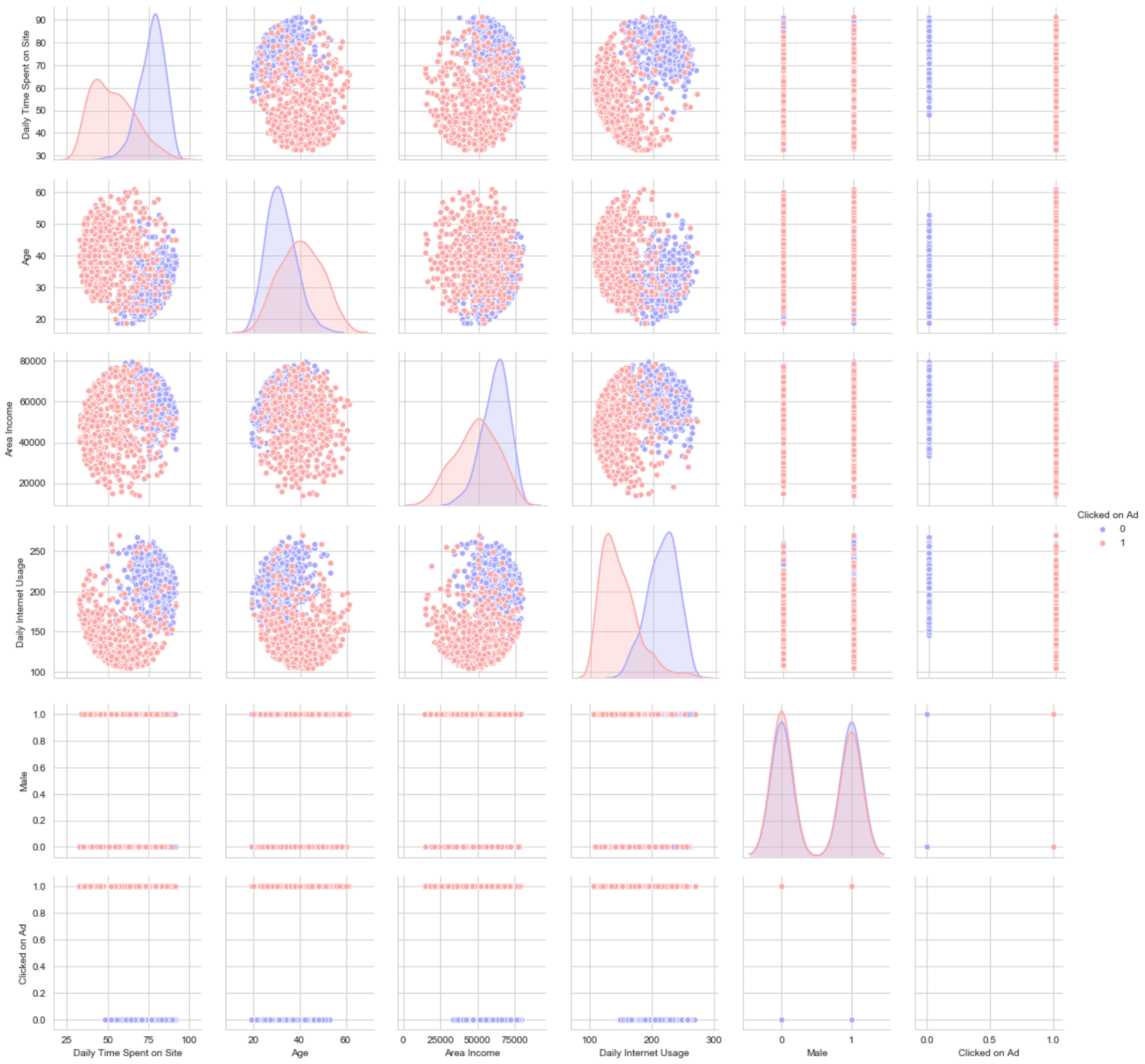
Next, I have created a jointplot of 'Daily Time Spent on Site' vs. 'Daily Internet Usage', as follows:

```
: sns.jointplot(x = 'Daily Internet Usage', y='Daily Time Spent on Site', data= ad_data, color='green')
```

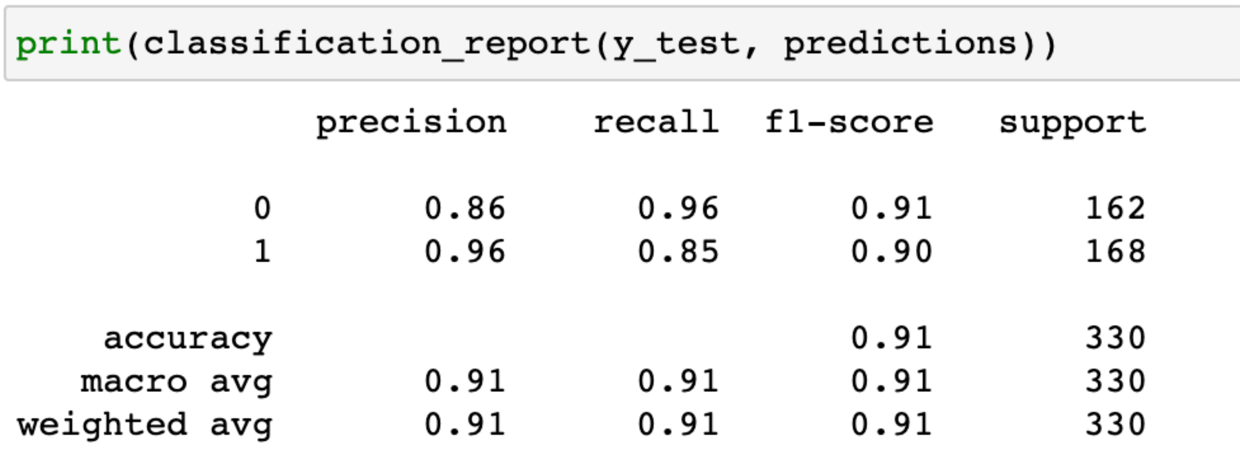
```
: <seaborn.axisgrid.JointGrid at 0x1a24467ac8>
```



Finally, I have created a pairplot with the hue defined by the 'Clicked on Ad' column feature, as follows:



After I have explored the data, I have used test train split function to split the data into training and testing dataset. In addition, I have taken the liberty to choose columns that I want to train the data on. After that, I have used logistic regression model to fit on the training data set. Next, I have used my model to predict values for the testing data. In order to determine the precision and accuracy of logistic regression model on the testing data, I have created a classification report as follows:

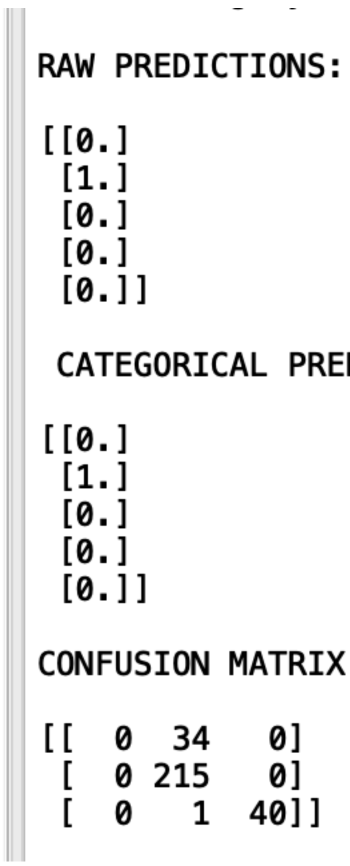

```
print(classification_report(y_test, predictions))
```

```
              precision    recall  f1-score   support

           0       0.86      0.96      0.91       162
           1       0.96      0.85      0.90       168

    accuracy                           0.91       330
   macro avg       0.91      0.91      0.91       330
weighted avg       0.91      0.91      0.91       330
```

Therefore, the logistic regression model is able to achieve a high accuracy score of 91%.

---

3. Neural Network_Machine Learning.py: In this problem, I am architecting a neural network that uses the S&P 500 data to predict the category of the following year's log return on the S&P 500 where the categories are down more than one standard deviation, between minus one and plus one standard deviation, and up more than one standard deviation. First of all, I have imported all the necessary machine learning libraries. Then I have created a database by extracting S&P daily stock price data from Yahoo. I have defined a dataset with all the price data such as Open, High, Low, Close, and Adj Close data from the dataframe. I have added other columns based on numeric manipulation of stock price columns such as High minus Low and Close minus Open. I have also added columns based on 3 days, 10 days, and 30 days moving averages of adjusted close stock price data on a rolling window basis. I have created a column based on standard deviation of 5 day rolling period of Adjusted Close price. I have created columns of Adjusted close stock price log return, its standard deviation and its mean. Finally, I have created a column 'category' where categories are down more than one standard deviation, between minus one and plus one standard deviation, and up more than one standard deviation. After standardizing and transforming the dataset, I split the data between testing and training datasets. I have made predictions based on training dataset using Sequential Neural Network. Then I have tested the model predictions on the training data. I have printed confusion matrix based on predictions, as follows:



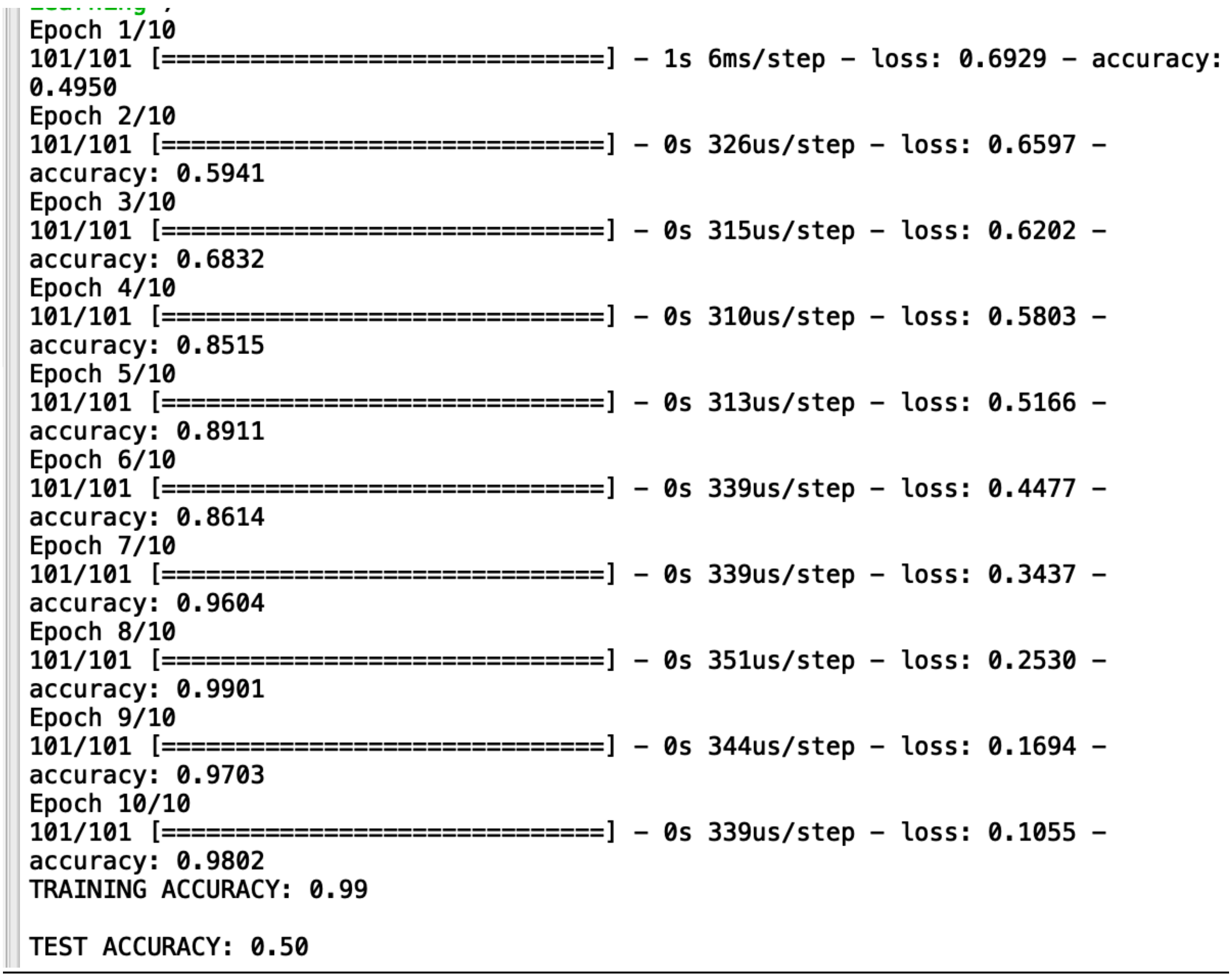
```
RAW PREDICTIONS:

[[0.]
 [1.]
 [0.]
 [0.]
 [0.]]

 CATEGORICAL PREDICTIONS:

[[0.]
 [1.]
 [0.]
 [0.]
 [0.]]

CONFUSION MATRIX:

[[  0  34   0]
 [  0 215   0]
 [  0   1  40]]
```

---

4. NLP_Machine Learning.py: In this problem, I have analyzed some tweet data for Tesla. The main goal is to make any sense of it. I went through my analyses using following steps:

- I have cleaned the data.
- I have deleted a lot of garbage and duplicate tweets.
- I have labeled the tweets with positive (1) or negative (0) sentiment by hand.

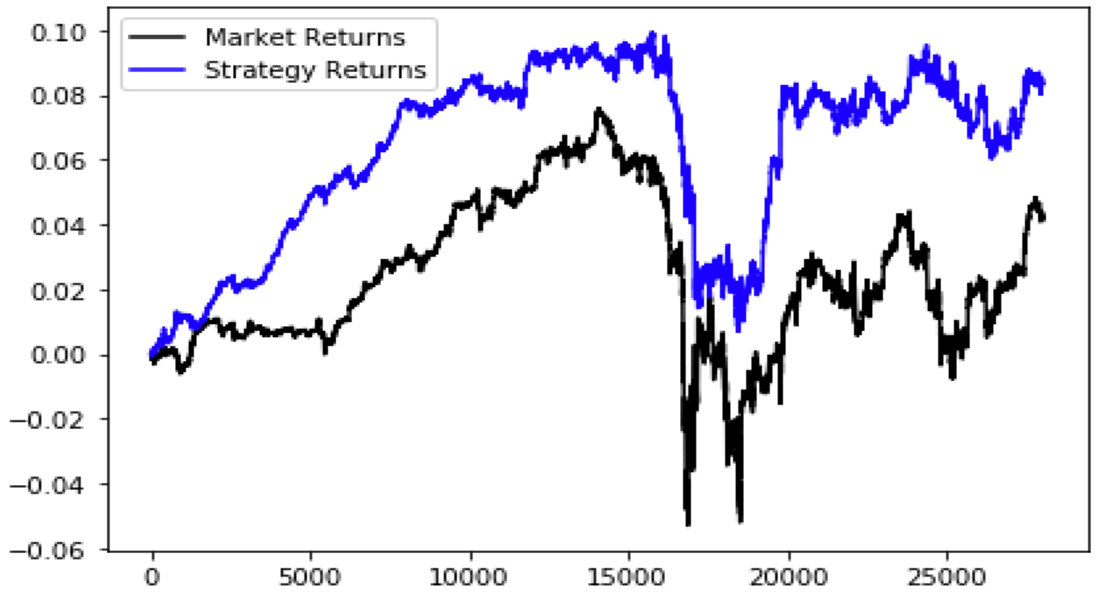
After that, I have created a convolutional neural network, with embedding and drop out. I have used the GloVe embedding weights to see if it improves my results. That is, I have separated the dataset into testing and training data and used some of the Tesla tweets data set as the test set. Finally, I have printed the testing and training accuracy of the convolutional neural network model, as follows:



```
Epoch 1/10
101/101 [==============================] - 1s 6ms/step - loss: 0.6929 - accuracy:
0.4950
Epoch 2/10
101/101 [==============================] - 0s 326us/step - loss: 0.6597 -
accuracy: 0.5941
Epoch 3/10
101/101 [==============================] - 0s 315us/step - loss: 0.6202 -
accuracy: 0.6832
Epoch 4/10
101/101 [==============================] - 0s 310us/step - loss: 0.5803 -
accuracy: 0.8515
Epoch 5/10
101/101 [==============================] - 0s 313us/step - loss: 0.5166 -
accuracy: 0.8911
Epoch 6/10
101/101 [==============================] - 0s 339us/step - loss: 0.4477 -
accuracy: 0.8614
Epoch 7/10
101/101 [==============================] - 0s 339us/step - loss: 0.3437 -
accuracy: 0.9604
Epoch 8/10
101/101 [==============================] - 0s 351us/step - loss: 0.2530 -
accuracy: 0.9901
Epoch 9/10
101/101 [==============================] - 0s 344us/step - loss: 0.1694 -
accuracy: 0.9703
Epoch 10/10
101/101 [==============================] - 0s 339us/step - loss: 0.1055 -
accuracy: 0.9802
TRAINING ACCURACY: 0.99

TEST ACCURACY: 0.50
```

Overall, the test accuracy using the model is pretty low at 50%.

<u>5. QDA_MachineLearning.py</u>: In this problem, I am creating a trading signal using QDA that predicts UP as moves > σ, DOWN as moves < -σ, and FLAT as moves in between the two. I am picking four features such as lagged log-returns, lagged volatilities, and some other technical indicators. These features I think are predictive of big price changes. Then I am back testing the trading strategy using signals generated using QDA machine learning algorithm. I am using QDA machine learning algorithm to generate signals because predicting the direction of returns, UP or DOWN, is difficult. One reason is because there is so much noise in the data. Observations close to zero are, by and large, randomly UP or DOWN. Maybe we can do better by trying to predict big moves only, rather than all the moves. I have further plotted Market Return and Strategy Return to show that the QDA strategy produces much better return and therefore, positive alpha, as compared to the Market returns, as follows:



```
Hit Ratio: 1.12
Win Percentage: 0.53
Loss Percentage: 0.47
Total Trades: 20123
```