E-Commerce Data Analysis and Modelling using Linear Regression in Machine Learning

Sneha Chowdary Mukkamala

sonasneha1994@gmail.com

AI Technology & Systems

https://ai-techsystems.com/

Abstract: *Machine Learning is a growing field today in AI. We discuss the use of Supervised Learning algorithm called as Linear Regression Learning in this paper for analyzing the e-commerce data set. Regression Learning is used as Prediction Model. The values of dependent variable are predicted by Regression Model based on values of Independent variables. By Regression Learning if after Experience E, program improves its performance P, then the program is said to be doing Regression Learning. In this project, we have chosen the e-commerce data. The main aim of this project is to check whether the designed model gives the desired accuracy on the chosen data.*

Keywords: *Regression Learning, ecommerce, Plotting Analysis, Modelling*

I. INTRODUCTION

This paper uses the e-commerce customer's data for analysis. In this data there are eight columns and five hundred rows. The columns include *Email* that contains customer's email id's, *Address possessing the address* of customer's, *Avatar* i.e., it refers to a character that represents an online user, *Average Session Length* which gives us the average time of the customer's spending in a session, *Time on App* includes the time spent by the customer in using app, *Time on Website* includes the time spent by the customer in using website, *Length of Membership* gives us the duration of customer's membership and the *Yearly amount Spent* is the total amount spent by the customer per annum.

The basic analysis of the data is done using Correlation, Scatter plot, Joint plot and Pair plots. After this overview of the data, the modelling will be done using the linear regression algorithm as mentioned already.

II. DATA ANALYSIS

The basic prerequisites for analyzing the data are Pandas, NumPy, Matplotlib and Seaborn.

- ***Pandas:*** Pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the python programming language. It provides highly optimized performance.
  We can analyze the data in pandas with:
  1. Series
  2. Data Frames

  1. *Series:*
     Series is a one dimensional array defined in pandas that can be used to store any data type.
     E.g.: x = pd.Series (Data, index = Index)
     Here, the data can be a Scalar value, a Python dictionary or an ndarray. Index by default is from 0, 1, 2… (n-1) where n is length of the data.
  2. *Data Frames:*
     Data Frames is two-dimensional data structure defined in pandas which consists of rows and columns.
     E.g.: x = pd.DataFrame (Data)
     Here, the data can be one or more dictionaries, one or more series or 2D-numpy ndarray.

- ***NumPy:*** NumPy is a general-purpose array-processing package. It provides a high-performance multi-dimensional array object and tools for working with these arrays. It is the fundamental package for scientific computing with python. It contains various features including these important ones:
  - A Powerful N-dimensional array object
  - Sophisticated (broadcasting) functions
  - Tools for integrating C/C++ and Fortran code
  - Useful linear algebra, Fourier transform and random number capabilities

  Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using NumPy which allows NumPy to seamlessly and speedily integrate with a wide variety of data bases.

- ***Matplotlib:*** Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely- histogram, bar charts, power spectra, error charts etc. It is used along with NumPy to provide an environment that is an effective open source alternative for Matplot. It can also be used with graphics toolkits like PyQt and wxPython.
- ***Seaborn:*** Seaborn is a library for making statistical graphics in python. It is built on top of matplotlib and closely integrated with pandas data structures. Here are some of the functionalities that seaborn offers:
  - A dataset-oriented API for examining relationships between multiple variables.
  - Specialized support for using categorical variables to show observations or aggregate statistics.
  - Options for visualizing univariate or bivariate distributions and for comparing them between subsets of data.
  - Automatic estimation and plotting of linear regression models for different kinds of dependent variables.
  - Convenient views onto the overall structure of complex datasets.
  - High-level abstractions for structuring multi-plot grids that let us easily build complex visualizations.
  - Concise control over matplotlib figure styling with several built-in themes.
  - Tools for choosing color palettes that faithfully reveal patterns in your data.

  Seaborn aims to make visualizations a central part of exploring and understanding data. Its dataset-oriented plotting functions operate on data frames and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.

  <ins>Basic commands used in analysis:</ins>

- The above packages are imported by using the commands as given below:

  ```
  import pandas as pd

  import numpy as np

  import matplotlib.pyplot as plt

  import seaborn as sns
  ```

- After importing these packages, the data is loaded by using the read command and is assigned to some variable named ecom as shown below:

  ```
  ecom=pd.read_csv('Ecommerce Customers.csv')
  ```

  In order to check the data, we can import only first five columns by using the head function as follows:

  ```
  ecom.head()
  ```

- To get the total number of rows and columns in the dataset, we use, ecom.shape command and to display the number of rows, data identification and data type of every column we use the command ecom.info ().

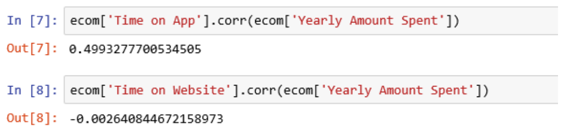
- ecom.describe() performs some pre-defined operations like count, mean, min, max, std etc. on every column in the dataset.
- ecom.columns is used to display the column names in the chosen data.

Correlation:

In general, correlation refers to the interdependency of the variables i.e., it gives the relation between the variables. Its values ranges between -1 to 1. There are two key components of a correlation value:

- *Magnitude:* The larger the magnitude (closer to 1 or -1) , the stronger the correlation
- *Sign:* If negative, there is an inverse correlation. If positive, there is a regular correlation

Now, correlation is performed on the selected columns in our data as shown in the below figure:



```
In [7]: ecom['Time on App'].corr(ecom['Yearly Amount Spent'])
Out[7]: 0.4993277700534505

In [8]: ecom['Time on Website'].corr(ecom['Yearly Amount Spent'])
Out[8]: -0.002640844672158973
```

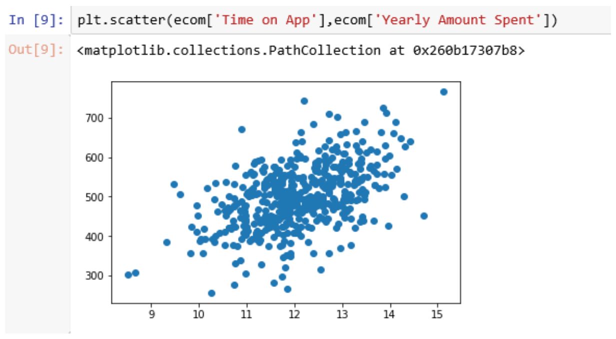
Figure 1: Performing the correlation and observing the results

Here, by observing the output values we can say that the Time spent on the App and the Yearly amount Spent are positively correlated whereas the Time spent on the Website and the Yearly amount Spent are inverse correlated. This shows that the customers who spend their time on Apps are likely to spend more amount annually compared to that of those who are using website.

Scatter plot:

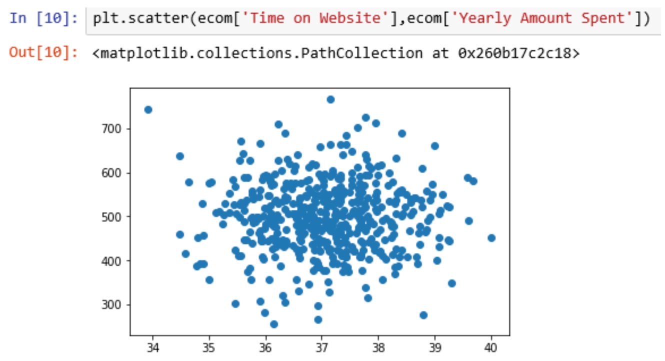
Now that we have obtained the correlation, let’s use the scatter plot to visualize how the data is scattered. For this let us frame an idea on what is scatter plot.

The scatter plot is a built-in function of the Matplot. It is a plot that shows the data as collection of points. The above correlated values are also plotted using this plot as shown below:



```
In [9]: plt.scatter(ecom['Time on App'],ecom['Yearly Amount Spent'])
Out[9]: <matplotlib.collections.PathCollection at 0x260b17307b8>
```

Figure 2: Correlation between Time on App and yearly amount spent using scatter plot



```
In [10]: plt.scatter(ecom['Time on Website'],ecom['Yearly Amount Spent'])
Out[10]: <matplotlib.collections.PathCollection at 0x260b17c2c18>
```

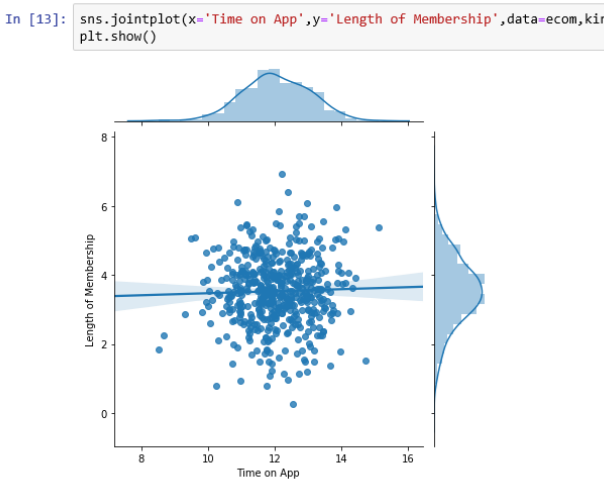
Figure 3: Correlation between Time on website and yearly amount spent using scatter plot

Joint Plot:

Similar to the scatter plot, now we perform the joint plotting on the same data as chosen above.

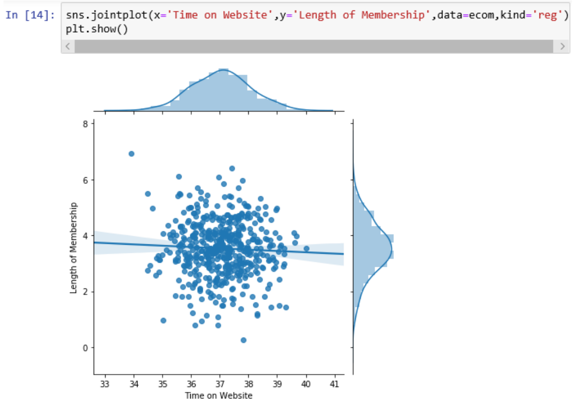
Joint plot is useful to visualize the bivariate distribution of two variables. For this we use a jointplot() function from seaborn, which creates a multi-panel figure that shows both the bivariate (or joint) relationship between two variables along with the univariate (or marginal) distribution of each on separate axes.

In the following figures, we have histogram fitted by a kernel density estimate (KDE) with the scatter plot.



```
In [13]: sns.jointplot(x='Time on App',y='Length of Membership',data=ecom,kir
         plt.show()
```

Figure 4: joint plot between the time on app and length of membership



```
In [14]: sns.jointplot(x='Time on Website',y='Length of Membership',data=ecom,kind='reg')
         plt.show()
```

Figure 5: joint plot between time on website and length of membership

Pair plot:

Pair plot plots pairwise relationships in a dataset. By default, this function will create a grid of axes such that each variable in the data will be shared in the y-axis across a single row and in the x-axis across a single column. The diagonal axes are treated differently, drawing a plot to show the univariate distribution of the data for the variable in that column. It is also possible to show a subset of variables or plot different variables on the rows and columns. The pair plot on the e-commerce data is as shown in the below figure:

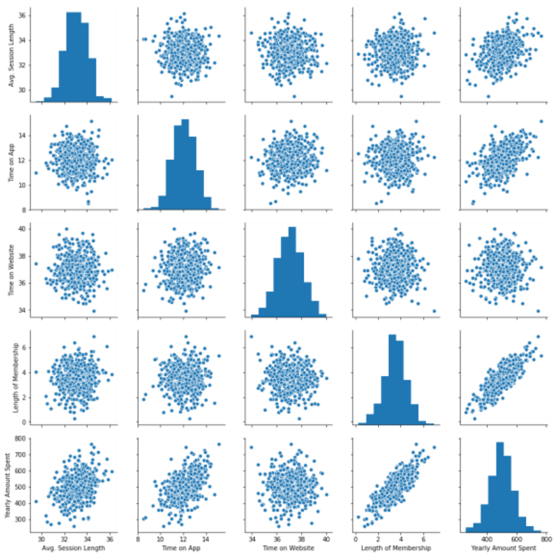


Figure 6: pair plotting of the e-commerce dataset

III. LINEAR REGRESSION

As we have completed our basic analysis process on the data, let us enter into the modelling of the data using Linear Regression Learning Algorithm. But before starting our algorithm modelling let us frame an idea about what is meant by linear regression.

Linear Regression is a Machine Learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on- the kind of relationship between dependent and independent variables they are considering and the number of independent variables being used.

Linear Regression performs the task to predict a dependent variable(y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y (output). Hence, the name is Linear Regression. This regression model can be with single and multiple variables.

Single variable Regression:

Here, the output only depends on single independent variable, x.

$y=\beta_0+\beta_1*x$

Where, y=Dependent variable/ output

x=Independent variable/ predictor variable

β_0= intercept

β_1 = coefficient

Multiple variable regression:

y= $\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \ldots\ldots + \beta_n * x_n$

Here, the outcome, y is based on the multiple predictor variables.

The difference between the y_{actual} and $y_{prediction}$ gives us the residual error. This can be represented as:

$Y_{actual} - y_{pred}$ =residual error

Again, this residual error is of three types. They are:

1. Mean error
2. Mean square error
3. Root mean square error

To build a regression model we need to select regression parameters for model. We need to perform a model by selecting the dependent and independent variables. This process is as follows:

1. We select output parameters that are suitable for our purpose by knowledge.
2. We will select input parameters by knowledge. We select the input parameters by knowing the relationship between output parameter and input parameters.

Formulae for β_0 and β_1 used by the algorithm are as follows:

β_1 = (correlation of x, y) * (standard deviation of y/ standard deviation of x)

where, standard deviation = square root of variance

variance = $\Sigma\,(x - \mu)^2 / n$

here, μ is mean of the data and n is the total number of data

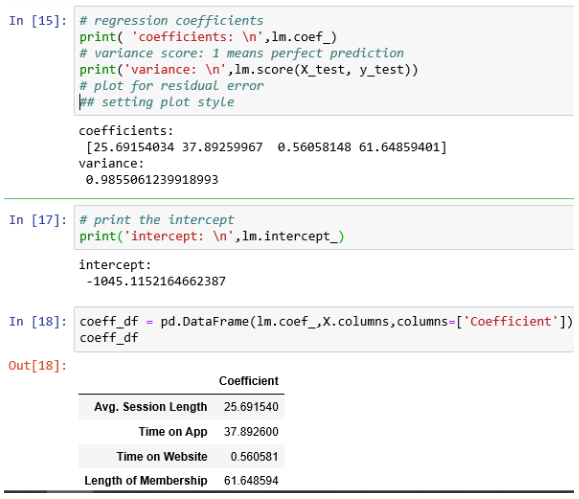
β_0 = mean(y) – β_1 * mean(x)

IV. MODELLING

Now, as we had an overview about the linear regression let's get into the modelling.

We divide the data into 2 parts (x and y). x would be the parameters that we would use to predict the y value and the y value in our case is Yearly Amount Spent. The attributes that come under x are 'Avg session length', 'time on app', 'time on website', 'length of membership' where as 'yearly amount spent' is the attribute for y.

After importing the package required for the train test operation, we will divide our dataset into test data and train data. 40% of the data is taken for test data and the rest is train data. After importing the linear regression from sklearn, assign it to some variable, say, lm. Now the x-train data and y-train data are fitted to this lm which means this data is given to the regression model. Once this training is done we need to calculate the regression coefficients, intercept and the variance of the data.



```
In [15]: # regression coefficients
         print( 'coefficients: \n',lm.coef_)
         # variance score: 1 means perfect prediction
         print('variance: \n',lm.score(X_test, y_test))
         # plot for residual error
         ## setting plot style

         coefficients:
          [25.69154034 37.89259967  0.56058148 61.64859401]
         variance:
          0.9855061239918993

In [17]: # print the intercept
         print('intercept: \n',lm.intercept_)

         intercept:
          -1045.1152164662387

In [18]: coeff_df = pd.DataFrame(lm.coef_,X.columns,columns=['Coefficient'])
         coeff_df

Out[18]:
```

	Coefficient
Avg. Session Length	25.691540
Time on App	37.892600
Time on Website	0.560581
Length of Membership	61.648594

Figure 7: Displaying the results for variance and regression coefficients

In figure 7, higher value is obtained for the length of membership. This implies that among the four predictor variables (chosen attributes) length of membership has more influence on the yearly amount spent.

And the other observation here is the variance. The variance value we have got is nearly equal to 1. This implies that we have designed our fit model. Now we need to predict the test values. So x-test data is assigned to the variable named predictions. Here a scatter plot is created for the actual test values and the predicted test values.

```
In [20]: plt.scatter(y_test,predictions)     #actual
Out[20]: <matplotlib.collections.PathCollection at 0x20362bd11d0>
```

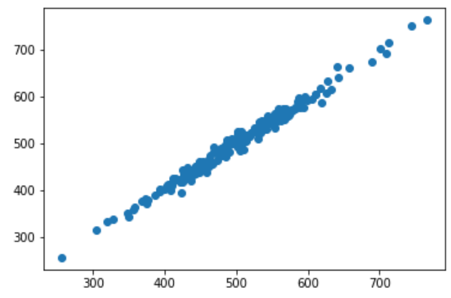


Figure 8: scatter plot for the actual test values and predicted test values

As the scatter plot is showing the good positive correlation, we can say that our model well trained. Now, the residual errors are calculated by importing the metrics package. The values we have got for this also shows that the model is good. The residuals here are plotted using the dist plot as shown below.

```
In [26]: sns.distplot((y_test-predictions),bins=50);
```

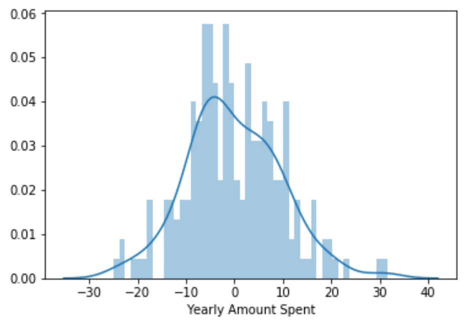


Figure 9: Dist plotting of the attributes

Dist plot here combines the matplotlib hist function with the seaborn kdeplot() function.

By giving one of the rows in the x-trained data and if the model is able to predict its corresponding value then also we can say that the accuracy of the designed model is good. In order to check this the third row values of the x-trained data, which is having 444 as its corresponding value, is given to the model. The output we have obtained here is 427.5 which is near to 444.

V. CONCLUSION

Based on the algorithm of linear regression learning, the required parameters like coefficients, variance etc., are found initially and then studied, computed and verified correctly.

The models of linear regression can be built for real life examples. The linear regression model for the e-commerce data is built and tested. The basic analysis by using the plots is also done. By considering the outputs mentioned above we find that the designed linear regression model gives the desired accuracy on the chosen data.

REFERENCES

[1] ML 1.2 “What is Supervised Learning ? ”,mathematicalmonk,

[2] Andrew Ng, ”Linear Regression with one Variable”, Stanford University, USA

[3] Hair,Black,Babin,Anderson, Tathan, “Multivariate Data Analysis”, Sixth Edition,Pearson.

[4] Tom Arnold, Jonathan M. Godbey, “Introducing Linear Regression: An Example Using Basketball Statistics”, University of Richmond-E, Claiborne Robins School of Business, Georgia State University- Department of Finance

[5] Andrew Ng, Machine Learning Course at www.Coursera.org Normal Equation Lecture, https://www.academia.edu/31150847/linear_regression

[6] https://matplotlib.org/