LOGISTIC REGRESSION MODEL TO PREDICT ADVERTISEMENT CLICKS

Shivani Bimavarapu
UIN: 01025992
sbima001@odu.edu
sbimavar@odu.edu

Rajyalakshmi Mukkamala
UIN: 01030975
rmukk002@odu.edu
rmukkama@odu.edu

Bhavani Manthena
UIN: 01033879
bmant001@odu.edu
bmanthen@odu.edu

ABSTRACT

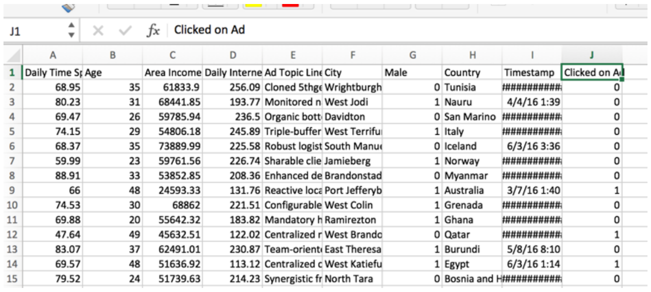
In this project a Logistic Regression model has been developed to predict whether or not a particular internet user clicked on an Advertisement on a company website based off the features of that user. The results give the number of Ad clicks the user did in the website and the Mobile App by exploring the time spent by the users on the website and the mobile App. Based upon which, company will decide whether to concentrate more on website development or the Mobile App development so as to maintain the existing users and also to grab more and more users.

1. PROBLEM DEFINITION

To predict the number of Ad clicks the user did on a Company website and also the number of clicks user did on the mobile App so as to help company to decide whether to develop the website more or the mobile App. The data used to analyze the above features is loaded into the advertising.csv. The problem here falls under Classification problem, specifically binary classification problem. Basically Logistic Regression model is used to solve such kind of problems, because logistic regression predicts discrete categories. The main goal is to interpret the results of the logistic regression through confusion matrix by predicting whether or not user clicks the Advertisements while they are using the sites.

1.1 DATASET

The dataset is acquired from github.



J1 fx Clicked on Ad

	A	B	C	D	E	F	G	H	I	J
1	Daily Time Sp	Age	Area Income	Daily Interne	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad
2	68.95	35	61833.9	256.09	Cloned 5thge	Wrightburgh	0	Tunisia	###########	0
3	80.23	31	68441.85	193.77	Monitored n	West Jodi	1	Nauru	4/4/16 1:39	0
4	69.47	26	59785.94	236.5	Organic bott	Davidton	0	San Marino	###########	0
5	74.15	29	54806.18	245.89	Triple-buffer	West Terrifu	1	Italy	###########	0
6	68.37	35	73889.99	225.58	Robust logist	South Manue	0	Iceland	6/3/16 3:36	0
7	59.99	23	59761.56	226.74	Sharable clie	Jamieberg	1	Norway	###########	0
8	88.91	33	53852.85	208.36	Enhanced de	Brandonstad	0	Myanmar	###########	0
9	66	48	24593.33	131.76	Reactive loca	Port Jefferyb	1	Australia	3/7/16 1:40	1
10	74.53	30	68862	221.51	Configurable	West Colin	1	Grenada	###########	0
11	69.88	20	55642.32	183.82	Mandatory h	Ramirezton	1	Ghana	###########	0
12	47.64	49	45632.51	122.02	Centralized r	West Brando	0	Qatar	###########	1
13	83.07	37	62491.01	230.87	Team-orient	East Theresa	1	Burundi	5/8/16 8:10	0
14	69.57	48	51636.92	113.12	Centralized c	West Katiefu	1	Egypt	6/3/16 1:14	1
15	79.52	24	51739.63	214.23	Synergistic fr	North Tara	0	Bosnia and H	###########	0

Table 1. Advertising.csv

This above data set shown in the table 1 contains the following features:

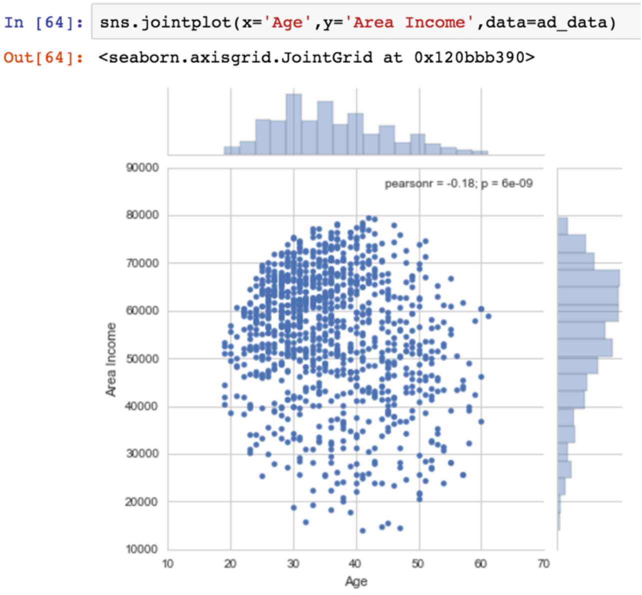
- 'Daily Time Spent on Site': consumer time on site in minutes
- 'Age': cutomer age in years
- 'Area Income': Avg. Income of geographical area of consumer
- 'Daily Internet Usage': Avg. minutes a day consumer is on the internet
- 'Ad Topic Line': Headline of the advertisement
- 'City': City of consumer

- '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

2. METHODS

2.1 EXPLORATORY DATA ANALYSIS

Seaborn, matplotlib and Pandas are used to analyze the data. Seaborn package is to explore data. Pandas is to carry out your entire data analysis workflow in Python. Matplotlib is for showing visualizations of the data in the Jupyter notebook, as Jupyter is the platform on which the project has been done.



```
In [64]: sns.jointplot(x='Age',y='Area Income',data=ad_data)
Out[64]: <seaborn.axisgrid.JointGrid at 0x120bbb390>
```

Figure 1. Joinplot showing Area income vs Age

Figure [1] have Age of the persons on x-axis and Area Income of the users on y-axis which showcase the fact that as the person gets older will not be able to get higher income. This can be better observed in the Figure [2] with the kde plot from the seaborn package. The darker portion gives the age group of persons who get the highest incomes.

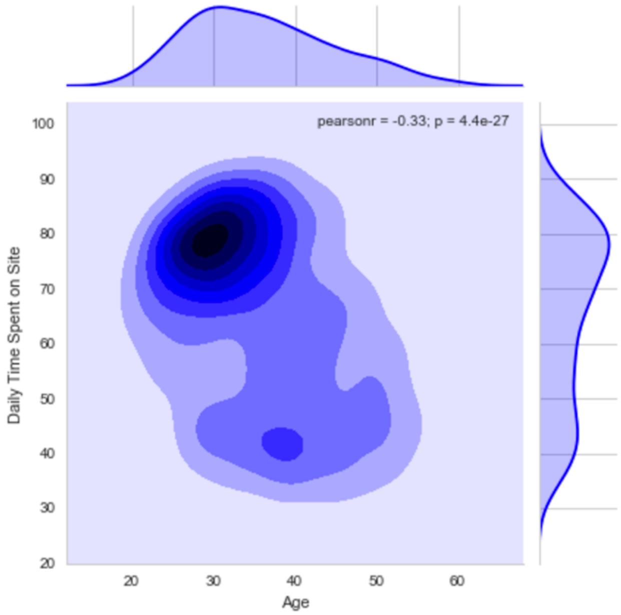


Figure 2. jointplot showing the kde distributions of Daily Time spent on site vs. Age.

The figure [3] below have 'Daily time spent on site' by the users on x-axis and 'Daily internet usage' by the users on y-axis. Based upon the daily time spent on the site, it is observed that the daily internet usage varies. Higher the internet usage, highest the daily time spent on the internet by the user. Lower the internet usage, lower the daily time spent on the internet by the user.

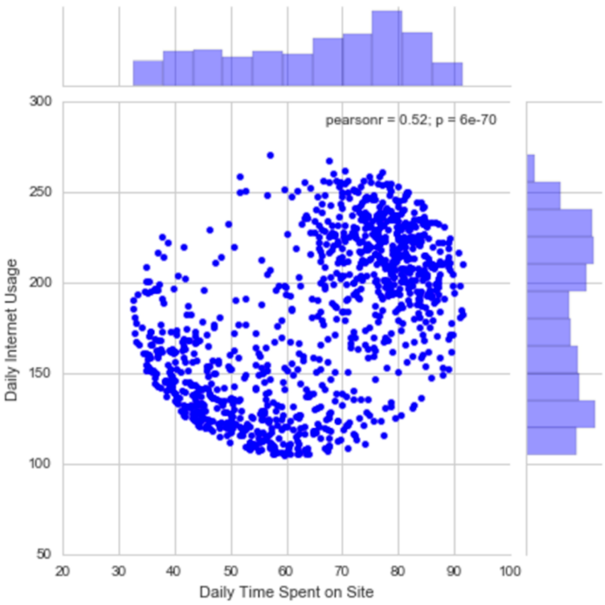


Figure 3. jointplot of 'Daily Time Spent on Site' vs. 'Daily Internet Usage'

2.2 TRAINING DATA & TEST DATA

Major portion of the data set is taken as training data and rest is taken as the testing data. This can be done by importing train_test_split method from scikit learn package. All the numerical data from the data set are taken into X which are the features. 'Clicked on Ad' alone is taken as Y which will be the label which the model is going to predict.

```
X = ad_data[['Daily Time Spent on Site',
'Age', 'Area Income','Daily Internet Usage',
'Male']]

y = ad_data['Clicked on Ad']

X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.33,
random_state=42)
```

70 % of the data is training data now and rest is the testing data. The training data is fitted to the logistic regression model. So, logistic regression is imported from scikit learn package.

2.3 LOGISTIC REGRESSION

Classification problem is the problem of identifying to which of the set of categories a new observation belongs to based off the training data.

Example: Disease diagnosis, loan default(yes/no), spam vs ham e-mails.

Binary classification – classifies two classes, and tries to predict on a continuous scale whereas logistic regression predicts on discrete categories. The convention for binary classification is to have two classes 0 and 1. We cannot use a normal linear regression model on binary groups, it won't lead to a good fit.

2.3.1 Sigmoid Function (Logistic function)

$$\phi(z) = \frac{1}{1 + e^{-z}}$$

It is the key to perform logistic regression. The linear model is not restricted to the values between 0 and 1, when we place the linear model into sigmoid function the result obtained is always between 0 and 1. Though we get results ranging from 0 to 1, we set a cut-off point at 0.5, anything below it results in class 0, anything above is class 1.

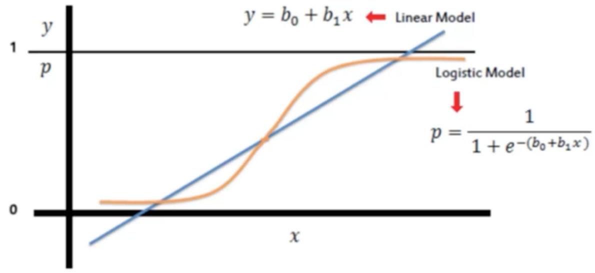
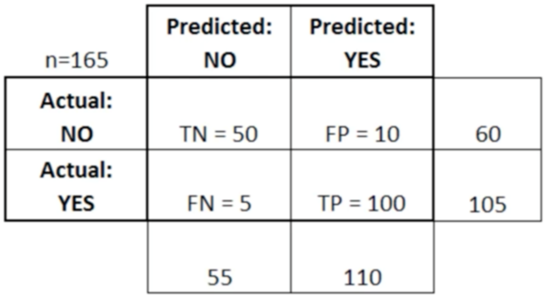


Figure 4. Comparing linear model line with the logistic model curve

The confusion matrix is used to evaluate the model behavior. Consider a sample confusion matrix:



n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Figure 5. Example confusion matrix

TP- True Positive TN- True Negative

FP- False Positive FN- False Negative

True positive is the proportions of positives that are correctly identified. Similarly, true negative is the proportions of negatives that are correctly identified. False positive is the condition that has been fulfilled when actually it is has not been fulfilled. Similarly, false negative is condition where the result is failed, when it was actually successful.

Precision= (TP+TN)/ Total

Precision is the ability of not to label a negative sample as positive.

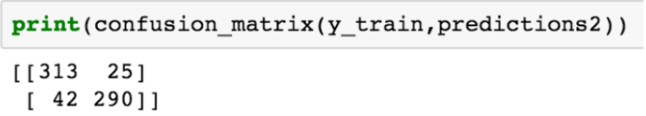
predictions = logmodel.predict(X_test)

The above code gives the predicted values.

3. RESULTS

Classification report and Confusion matrix are imported from the scikit learn package.

The Confusion Matrix for the training data is as follows:



```
print(confusion_matrix(y_train,predictions2))
```

```
[[313  25]
 [ 42 290]]
```

Figure 6. Confusion matrix for training data

The above is the pseudo code which prints the confusion matrix.

From Figure [6], the people were predicted to click on commercials and the actually clicked were 290, the people who were predicted not to click on the commercials and actually did not click on them were 313.

The people who were predicted to click on commercial and actually did not click on them are 25, and the people who were not predicted to click on the commercials and actually clicked on them are 42.

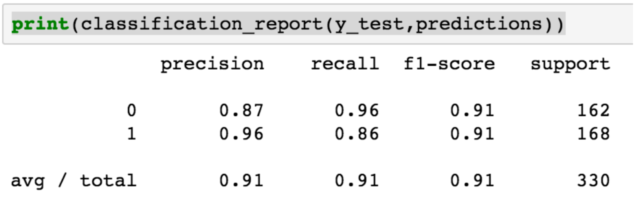
Based upon these values the precision, recall, fscore are calculated.

Precision is the fraction of retrieved values that are relevant to the data.

Recall is the fraction of successfully retrieved values that are relevant to the data.

Fscore is the harmonic mean of precision and recall.

The classification report of the testing data is as follows:



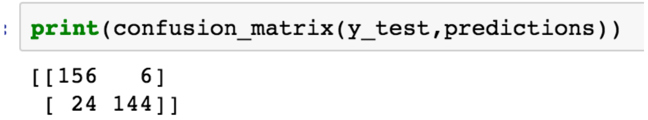
```
print(classification_report(y_test,predictions))
```

```
             precision    recall  f1-score   support

          0       0.87      0.96      0.91       162
          1       0.96      0.86      0.91       168

avg / total       0.91      0.91      0.91       330
```

Figure 7. Classification report of testing data



```
print(confusion_matrix(y_test,predictions))
```

```
[[156   6]
 [ 24 144]]
```

Figure 8. Confusion matrix for test data

From Figure [8], the people were predicted to click on commercials and the actually clicked were 144, the people who were predicted not to click on the commercials and actually did not click on them were 156. The people who were predicted to click on commercial and actually did not click on them are 6, and the people who were not predicted to click on the commercials and actually clicked on them are 24.

4. CONCLUSION

From the results obtained, the precision is 0.91 which depicts the predicted values are 91% accurate. Hence the probability that the user can click on the commercial is 0.91 which is a good precision value to get a good model.

5. REFERENCES

[1] Introduction to statistical learning by Gareth James.

[2] http://scikit-learn.org/

[3] http://pandas.pydata.org/

[4] http://www-bcf.usc.edu/~gareth/ISL/

[5] http://www.statisticssolutions.com/what-is-logistic-regression/

[6] http://matplotlib.org/

[7] http://seaborn.pydata.org/