Logistic Regression-Facebook Ads Project

Nawal Obaid

9/6/2022

PROBLEM STATEMENT

You have been hired as a consultant to a start-up that is running a targetted marketing ads on facebook. The company wants to anaylze customer behaviour by predicting which customer clicks on the advertisement. Customer data is as follows:

Inputs:

- Name
- e-mail
- Country
- Time on Facebook
- Estimated Salary (derived from other parameters)

Outputs:

- Click (1: customer clicked on Ad, 0: Customer did not click on the Ad)

STEP #0: LIBRARIES IMPORT

In [51]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

STEP #1: IMPORT DATASET

In [52]:
```
# because the file contains a lot of special characters, we need to specify a spe
fb_ad =  pd.read_csv('Facebook_Ads_2.csv', encoding='ISO-8859-1')
```

In [53]:
```
fb_ad.head()
```

Out[53]:

	Names	emails	Country	Time Spent on Site	Salary	C
0	Martina Avila	cubilia.Curae.Phasellus@quisaccumsanconvallis.edu	Bulgaria	25.649648	55330.06006	
1	Harlan Barnes	eu.dolor@diam.co.uk	Belize	32.456107	79049.07674	
2	Naomi Rodriquez	vulputate.mauris.sagittis@ametconsectetueradip...	Algeria	20.945978	41098.60826	
3	Jade Cunningham	malesuada@dignissim.com	Cook Islands	54.039325	37143.35536	
4	Cedric Leach	felis.ullamcorper.viverra@egetmollislectus.net	Brazil	34.249729	37355.11276	

In [54]:
```
fb_ad.tail()
```

Out[54]:

	Names	emails	Country	Time Spent on Site	Salary	Clicked
494	Rigel	egestas.blandit.Nam@semvitaealiquam.com	Sao Tome and Principe	19.222746	44969.13495	0
495	Walter	ligula@Cumsociis.ca	Nepal	22.665662	41686.20425	0
496	Vanna	Cum.sociis.natoque@Sedmolestie.edu	Zimbabwe	35.320239	23989.80864	0
497	Pearl	penatibus.et@massanonante.com	Philippines	26.539170	31708.57054	0
498	Nell	Quisque.varius@arcuVivamussit.net	Botswana	32.386148	74331.35442	1

In [55]:
```
fb_ad.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 499 entries, 0 to 498
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Names               499 non-null    object 
 1   emails              499 non-null    object 
 2   Country             499 non-null    object 
 3   Time Spent on Site  499 non-null    float64
 4   Salary              499 non-null    float64
 5   Clicked             499 non-null    int64  
dtypes: float64(2), int64(1), object(3)
memory usage: 23.5+ KB
```

In [56]:
```
fb_ad.describe()
```

Out[56]:

	Time Spent on Site	Salary	Clicked
count	499.000000	499.000000	499.000000
mean	32.920178	52896.992469	0.501002
std	9.103455	18989.183150	0.500501
min	5.000000	20.000000	0.000000
25%	26.425044	38888.117260	0.000000
50%	33.196067	52840.913110	1.000000
75%	39.114995	65837.288190	1.000000
max	60.000000	100000.000000	1.000000

STEP #2: EXPLORE/VISUALIZE DATASET

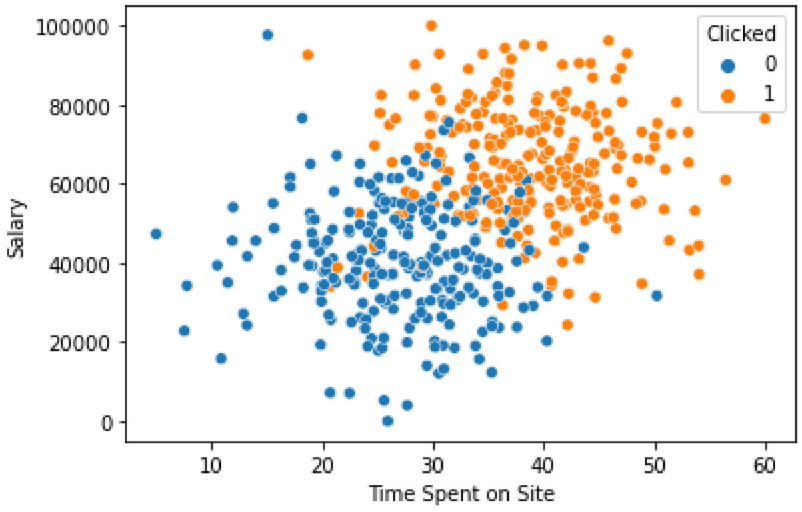
In [57]:
```
did_click= fb_ad[fb_ad['Clicked']==1]
no_click = fb_ad[fb_ad['Clicked']==0]
Total = fb_ad['Clicked']

print('Total number of customers=', len(Total))
print('Number of customers who clicked on Ad =', len(did_click))
print('Percentage of customers who clicked on Ad =', len(did_click)/len(Total)*10
print('Number of customers who did not click on Ad =', len(no_click))
print('Percentage of customers who did not click on Ad =', len(no_click)/len(Tota
```

```
Total number of customers= 499
Number of customers who clicked on Ad = 250
Percentage of customers who clicked on Ad = 50.1002004008016
Number of customers who did not click on Ad = 249
Percentage of customers who did not click on Ad = 49.899799599198396
```

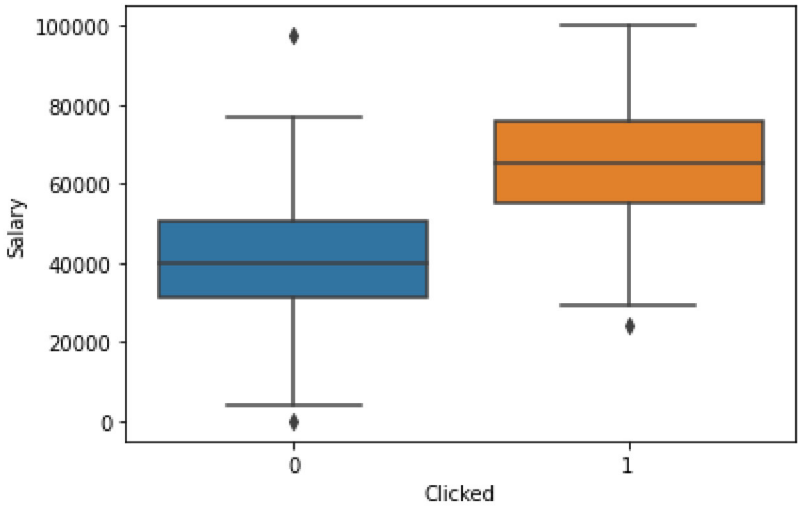
In [58]:
```
sns.scatterplot(x='Time Spent on Site', y= 'Salary', data= fb_ad, hue='Clicked')
```

Out[58]:
```
<AxesSubplot:xlabel='Time Spent on Site', ylabel='Salary'>
```



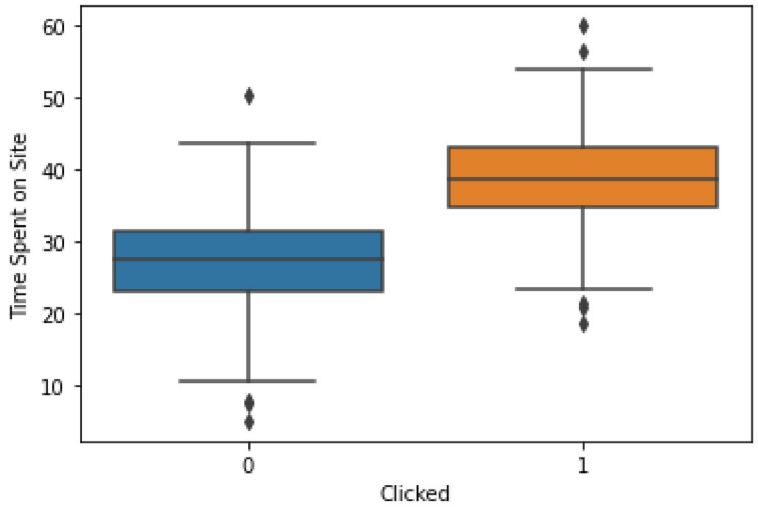
In [59]:
```
sns.boxplot( x='Clicked', y= 'Salary', data= fb_ad)
```

Out[59]:
```
<AxesSubplot:xlabel='Clicked', ylabel='Salary'>
```



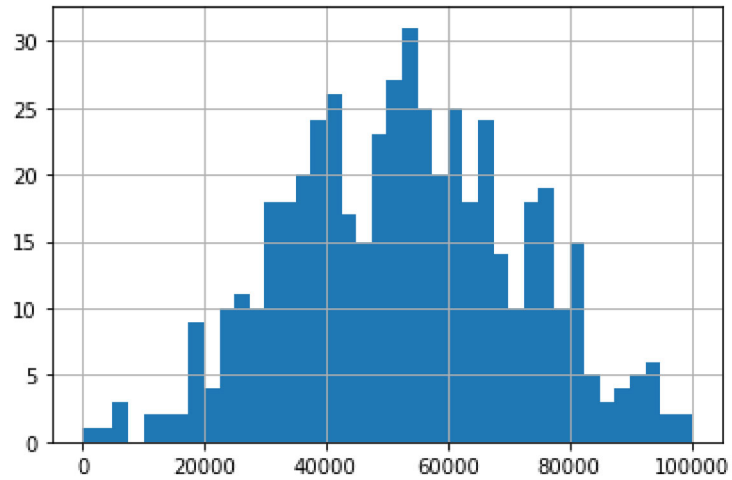
In [60]:
```
sns.boxplot(x= 'Clicked', y='Time Spent on Site', data= fb_ad)
```

Out[60]: <AxesSubplot:xlabel='Clicked', ylabel='Time Spent on Site'>



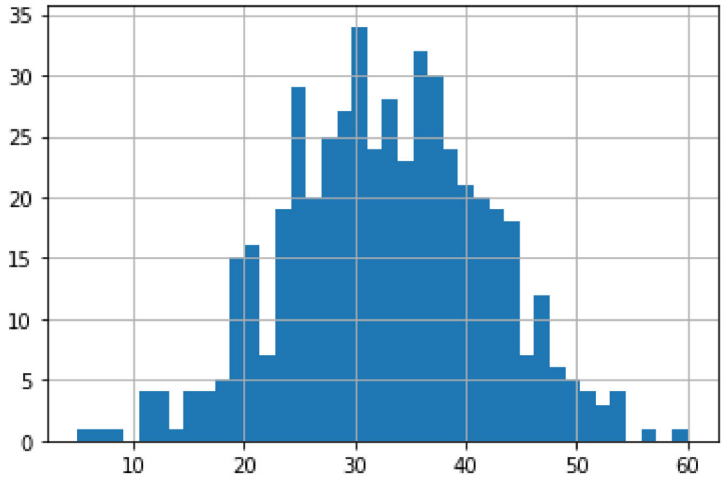
In [62]:
```
fb_ad['Salary'].hist(bins=40)
```

Out[62]: <AxesSubplot:>



In [63]:
```
fb_ad['Time Spent on Site'].hist(bins=40)
```

Out[63]:
```
<AxesSubplot:>
```



STEP #3: PREPARE THE DATA FOR TRAINING/ DATA CLEANING

In [64]:
```
fb_ad.head()
```

Out[64]:

	Names	emails	Country	Time Spent on Site	Salary	C
0	Martina Avila	cubilia.Curae.Phasellus@quisaccumsanconvallis.edu	Bulgaria	25.649648	55330.06006	
1	Harlan Barnes	eu.dolor@diam.co.uk	Belize	32.456107	79049.07674	
2	Naomi Rodriquez	vulputate.mauris.sagittis@ametconsectetueradip...	Algeria	20.945978	41098.60826	
3	Jade Cunningham	malesuada@dignissim.com	Cook Islands	54.039325	37143.35536	
4	Cedric Leach	felis.ullamcorper.viverra@egetmollislectus.net	Brazil	34.249729	37355.11276	

In [65]:
```
#Let's drop the emails, country and names (we can make use of the country later!)
fb_ad.drop(['Names', 'emails', 'Country'],axis=1,inplace=True)
```

In [66]:
```
fb_ad.head()
```

Out[66]:

	Time Spent on Site	Salary	Clicked
0	25.649648	55330.06006	0
1	32.456107	79049.07674	1
2	20.945978	41098.60826	0
3	54.039325	37143.35536	1
4	34.249729	37355.11276	0

STEP#4: MODEL TRAINING

In [93]:
```
X = fb_ad.drop('Clicked',axis=1)
y = fb_ad['Clicked']
```

In [94]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_s
```

In [95]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaled_X_train = scaler.fit_transform(X_train)
scaled_X_test = scaler.transform(X_test)
```

In [96]:
```
# Fitting Logistic Regression to the Training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
# now lwt's train/fit our model
classifier.fit(scaled_X_train,y_train)
```

Out[96]:
```
LogisticRegression
LogisticRegression(random_state=0)
```

STEP#5: MODEL TESTING----Prediction

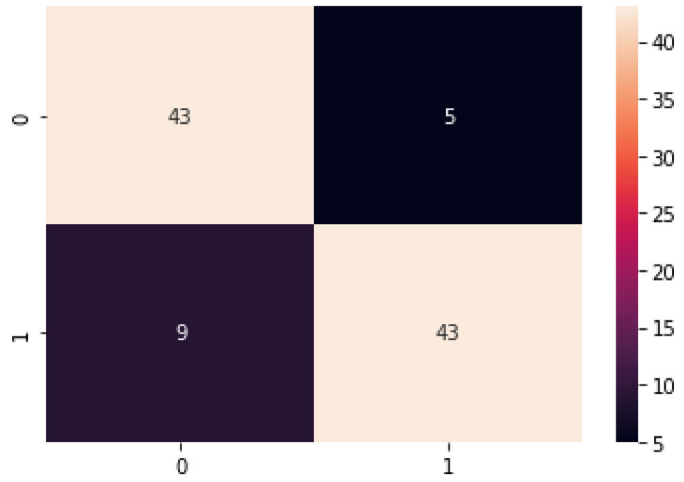
In [108]:
```
from sklearn.metrics import confusion_matrix,classification_report
```

In [109]:
```
y_prediction = classifier.predict(scaled_X_test)
```

In [110]:
```
cm= confusion_matrix(y_test,y_prediction)
```

In [111]:
```
sns.heatmap(cm, annot=True,fmt='d')
```

Out[111]: <AxesSubplot:>



In [112]:
```
print(classification_report(y_test,y_prediction))
```

```
              precision    recall  f1-score   support

           0       0.83      0.90      0.86        48
           1       0.90      0.83      0.86        52

    accuracy                           0.86       100
   macro avg       0.86      0.86      0.86       100
weighted avg       0.86      0.86      0.86       100
```