Project on Online Shopping Intention using Machine Learning Models

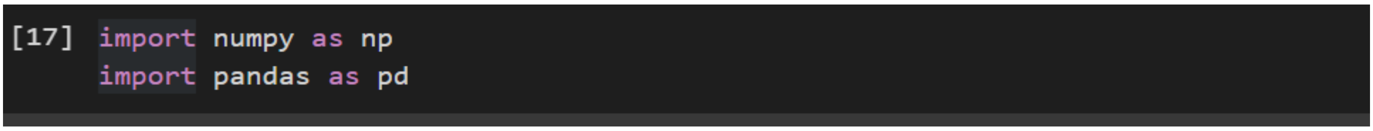
- AIM: To create a Data Science project, where we'll be predicting whether the Revenue Generated by the visiting customer for online shopping & to make seller understand what are patterns and intention of the customer. So, here I have used machine learning models with the help of csv-dataset(s) provided which contains 10 numerical & 8 categorical values of different-different areas which gives us idea what kind of customers are arriving through which flatform & how much time they are spending searching for the product needed.

- Steps to be taken in the project is sub-divided into the following sections. These are:

    - Loading necessary libraries such as 'numpy', 'pandas', 'sklearn. model' etc.
    - Loading Dataset(s) as a CSV file. Here we are using two different files for training & testing the models.
    - Data cleaning was performed by changing string values to integer values.
    - Visualisation of Trend of Revenue from online shopping using Tableau.
    - Splitting the data set into independent & dependent sets (only train data set was taken in use).
    - Importing the train_test_split model from sklearn.model for splitting data into train & test sets.
    - Importing different kinds of classification models & then training those models with the help of fit().
    - Predicting the trained models & then checking their accuracy of the model using confusion matrix & accuracy score.
    - Then recalled test_dataset & splitted the data into testing & training sets using X1_train & X1_test.
    - Then, trained the test_dataset with tain_dataset with the help of better accuracy's model.
    - Finally, predicted whether the revenue was generated or not for test_dataset.

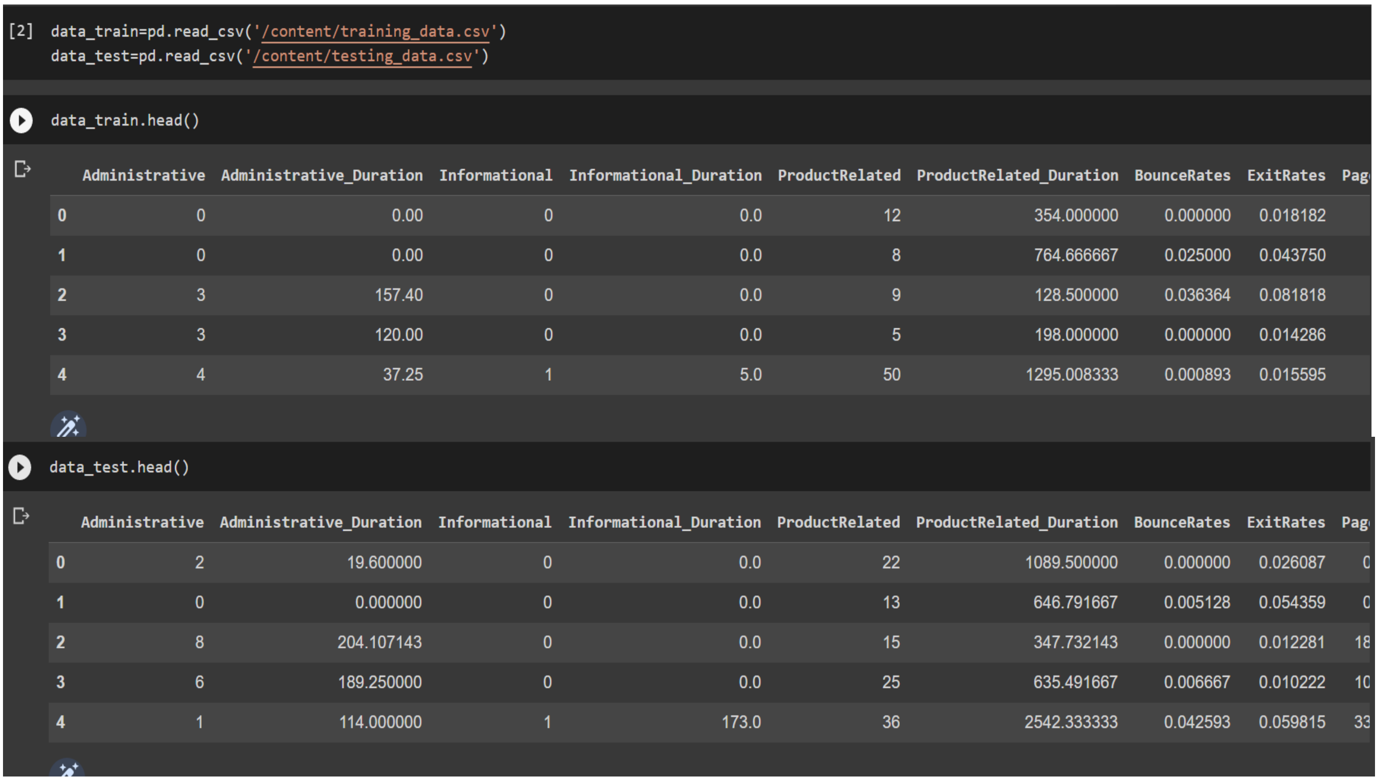
- Steps of creating ML model:

- Step-1: Importing numpy as np & pandas as pd for loading and reading the data-set.



```
[17] import numpy as np
     import pandas as pd
```

- Step-2: Loading the csv-dataset(s) in the variable name(s) 'data_train' & 'data_test'. Then viewing the data(s) with data_train.head() & data_test.head().



```
[2] data_train=pd.read_csv('/content/training_data.csv')
    data_test=pd.read_csv('/content/testing_data.csv')
```

```
data_train.head()
```

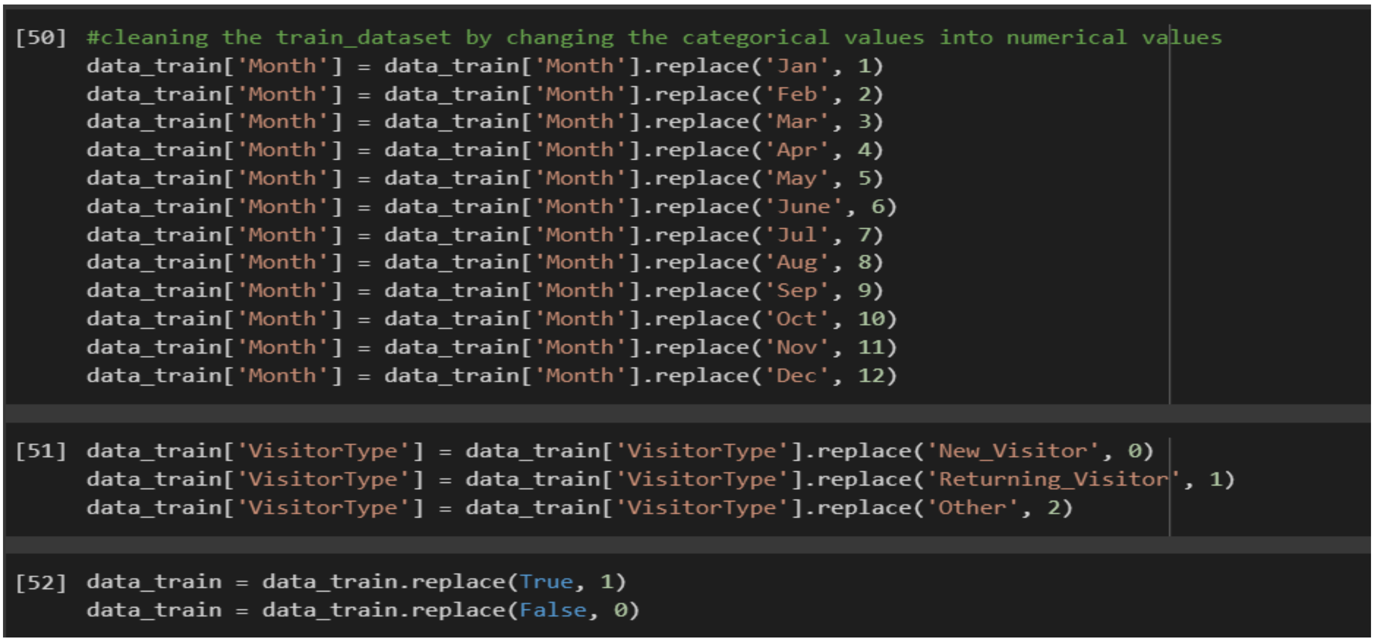
	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	Pag
0	0	0.00	0	0.0	12	354.000000	0.000000	0.018182	
1	0	0.00	0	0.0	8	764.666667	0.025000	0.043750	
2	3	157.40	0	0.0	9	128.500000	0.036364	0.081818	
3	3	120.00	0	0.0	5	198.000000	0.000000	0.014286	
4	4	37.25	1	5.0	50	1295.008333	0.000893	0.015595	

```
data_test.head()
```

	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	Pag
0	2	19.600000	0	0.0	22	1089.500000	0.000000	0.026087	0
1	0	0.000000	0	0.0	13	646.791667	0.005128	0.054359	0
2	8	204.107143	0	0.0	15	347.732143	0.000000	0.012281	18
3	6	189.250000	0	0.0	25	635.491667	0.006667	0.010222	10
4	1	114.000000	1	173.0	36	2542.333333	0.042593	0.059815	33

-Viewing train & test dataset(s)

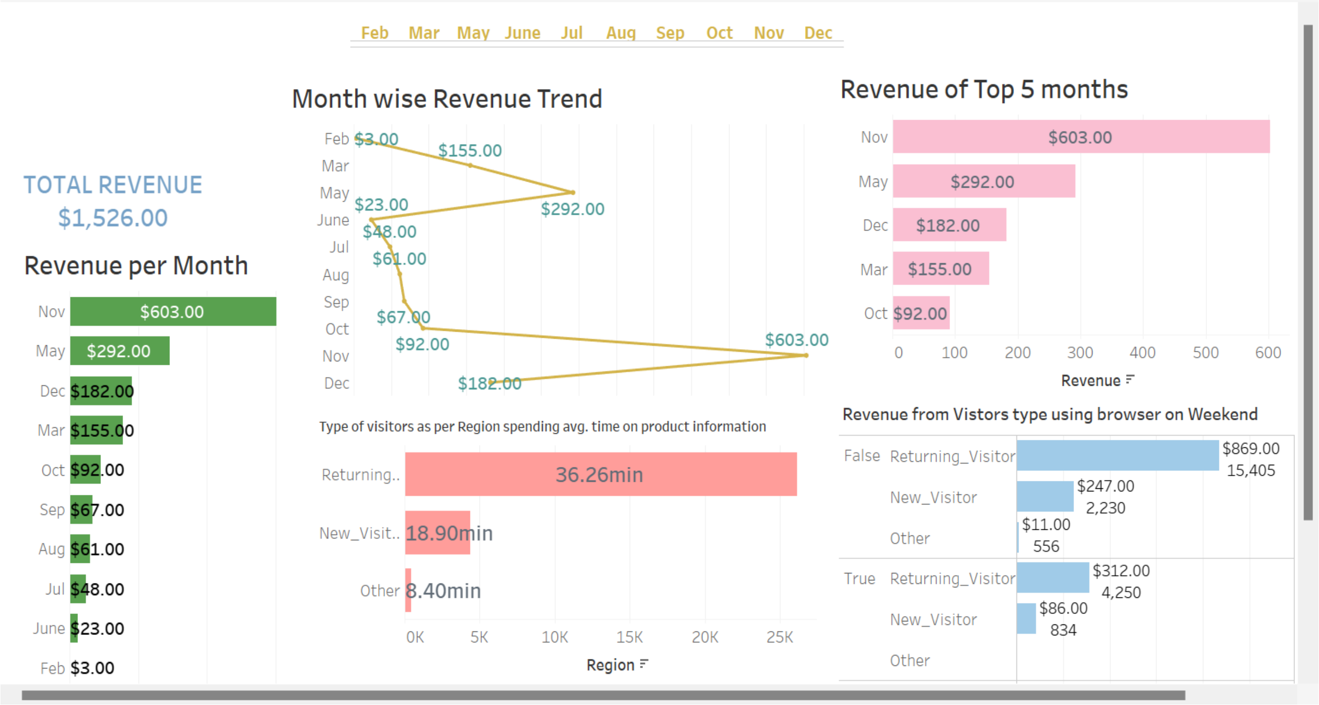
- Step-3: Cleaning the datasets by changing any categorical values to numerical value.



```
[50] #cleaning the train_dataset by changing the categorical values into numerical values
     data_train['Month'] = data_train['Month'].replace('Jan', 1)
     data_train['Month'] = data_train['Month'].replace('Feb', 2)
     data_train['Month'] = data_train['Month'].replace('Mar', 3)
     data_train['Month'] = data_train['Month'].replace('Apr', 4)
     data_train['Month'] = data_train['Month'].replace('May', 5)
     data_train['Month'] = data_train['Month'].replace('June', 6)
     data_train['Month'] = data_train['Month'].replace('Jul', 7)
     data_train['Month'] = data_train['Month'].replace('Aug', 8)
     data_train['Month'] = data_train['Month'].replace('Sep', 9)
     data_train['Month'] = data_train['Month'].replace('Oct', 10)
     data_train['Month'] = data_train['Month'].replace('Nov', 11)
     data_train['Month'] = data_train['Month'].replace('Dec', 12)

[51] data_train['VisitorType'] = data_train['VisitorType'].replace('New_Visitor', 0)
     data_train['VisitorType'] = data_train['VisitorType'].replace('Returning_Visitor', 1)
     data_train['VisitorType'] = data_train['VisitorType'].replace('Other', 2)

[52] data_train = data_train.replace(True, 1)
     data_train = data_train.replace(False, 0)
```

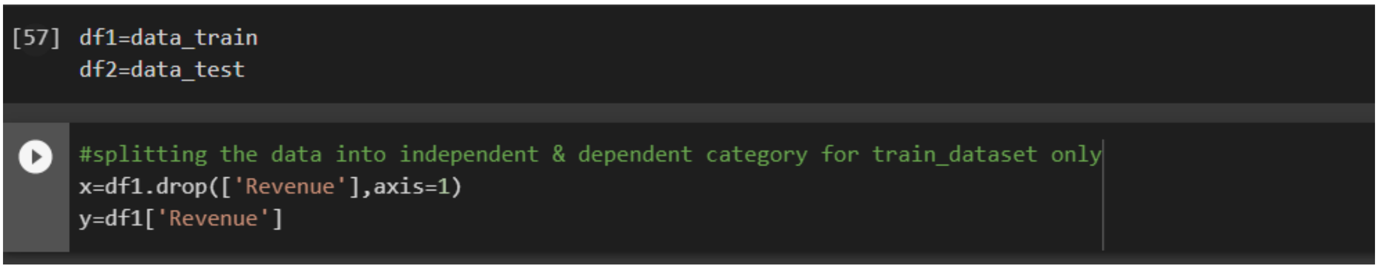
-Same has been applied for (test_dataset).

- Step-4: Visualising the Revenue generated per month & from the type of visitor depending how much time they are spending on a particular type of product.



-Insights of revenue generated in total and also as per months, top 5 months & using browsers on weekend. I have also added a feature with which we can select individual month for visualizing the particular months revenue. You can also visit and check how that works by clicking here

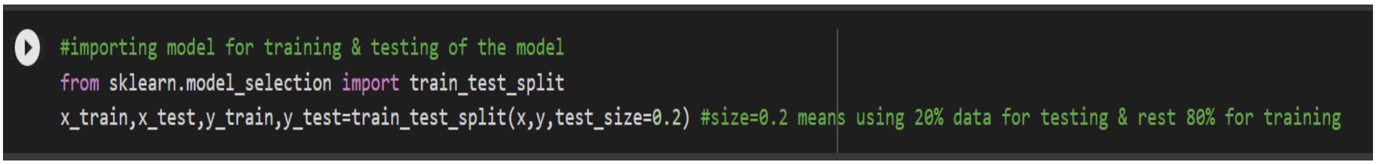
- Step-5: Splitting the dataset into dependent & independent sets (taken only train dataset).



```
[57] df1=data_train
     df2=data_test
```

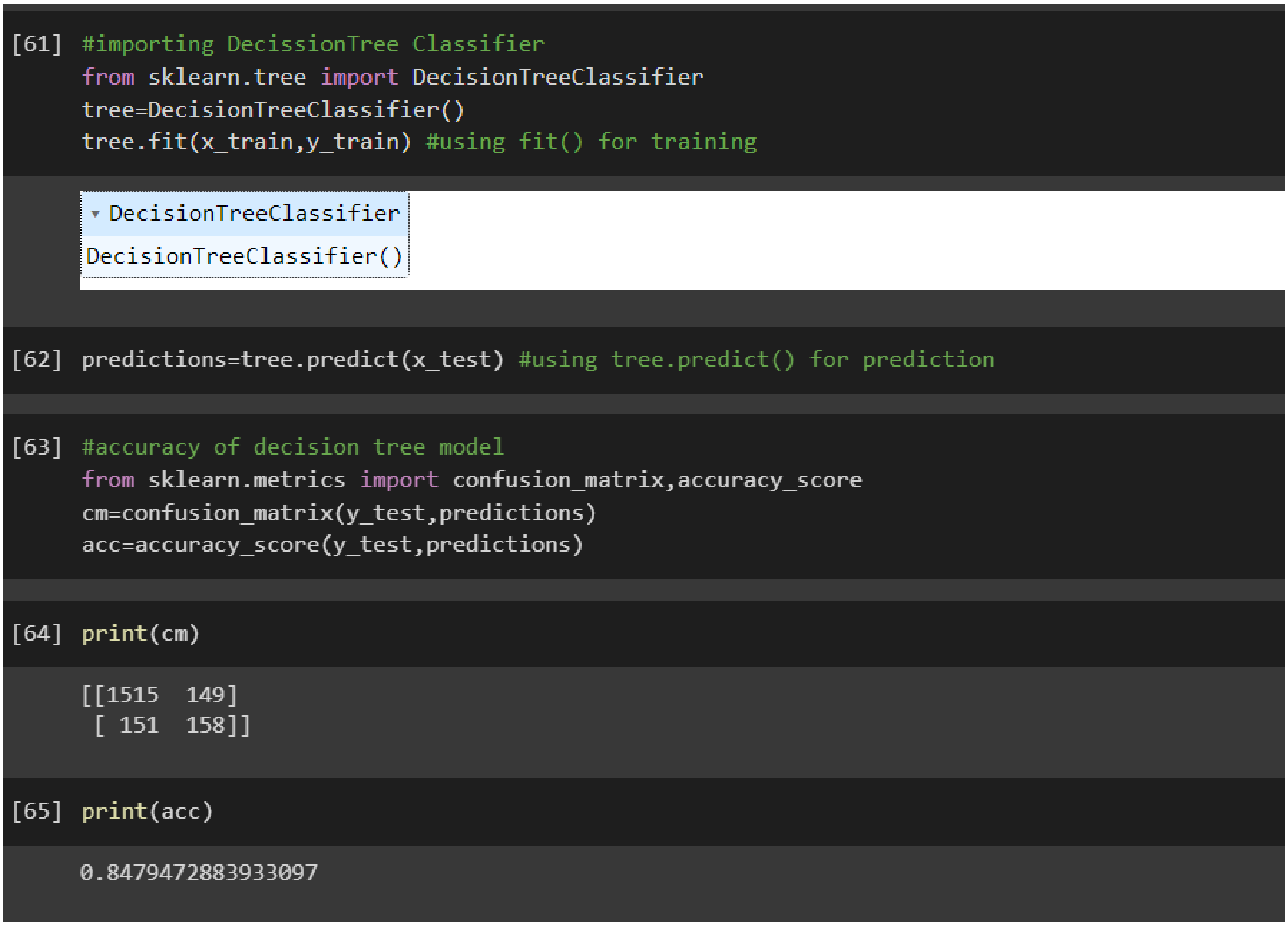
```
#splitting the data into independent & dependent category for train_dataset only
x=df1.drop(['Revenue'],axis=1)
y=df1['Revenue']
```

- Step-6: Importing train_test_split from sklearn.model library for splitting the data into train and test sets.



```
#importing model for training & testing of the model
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) #size=0.2 means using 20% data for testing & rest 80% for training
```

- Step-7: Importing DecisionTreeClassifier from sklearn.tree & then activating it by storing into the variable name tree. Then used tree.fit() to train the model by providing train & test sets as x & y. And then predicted the trained model & the checked accuracy of the model using confusion_matrix & accuracy_score.



```
[61] #importing DecissionTree Classifier
     from sklearn.tree import DecisionTreeClassifier
     tree=DecisionTreeClassifier()
     tree.fit(x_train,y_train) #using fit() for training
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
[62] predictions=tree.predict(x_test) #using tree.predict() for prediction
```

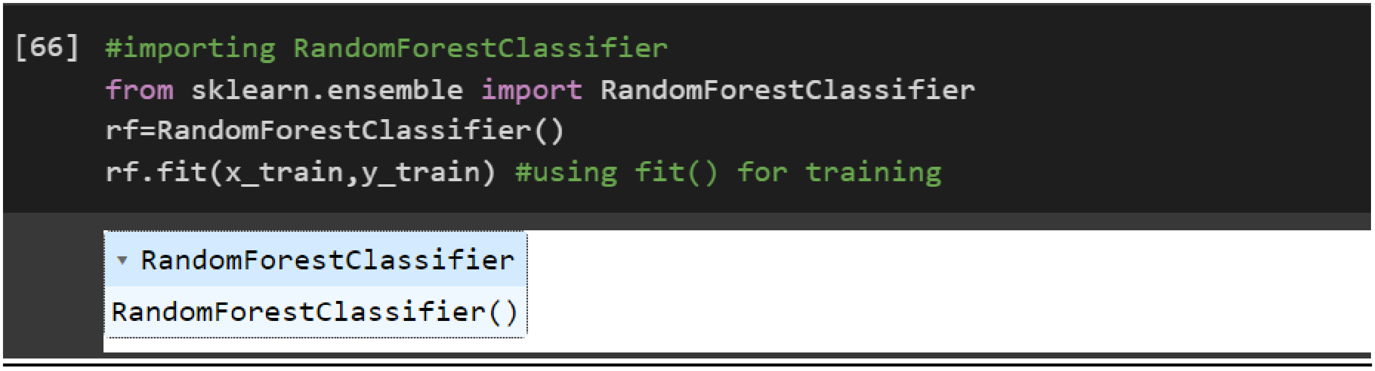
```
[63] #accuracy of decision tree model
     from sklearn.metrics import confusion_matrix,accuracy_score
     cm=confusion_matrix(y_test,predictions)
     acc=accuracy_score(y_test,predictions)
```

```
[64] print(cm)

     [[1515  149]
      [ 151  158]]
```

```
[65] print(acc)

     0.8479472883933097
```

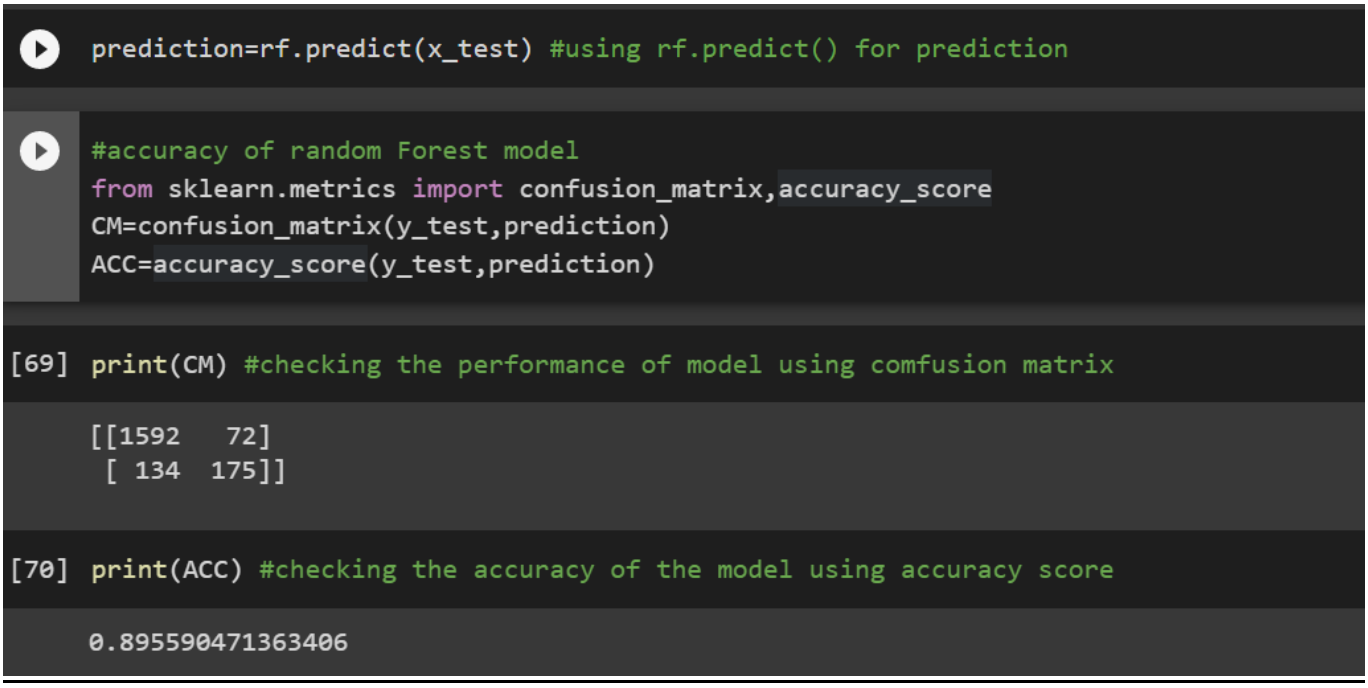
-In the above model we can see that the accuracy obtained is 84% which is quite good but we can also try using different models to see if we can get better accuracy than this or not.

- So I have also used RandomForestClassifier & SVM for obtaining better accuracy of the model.
- Model using RandomForestClassifier.



```
[66] #importing RandomForestClassifier
     from sklearn.ensemble import RandomForestClassifier
     rf=RandomForestClassifier()
     rf.fit(x_train,y_train) #using fit() for training
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```



```
prediction=rf.predict(x_test) #using rf.predict() for prediction
```

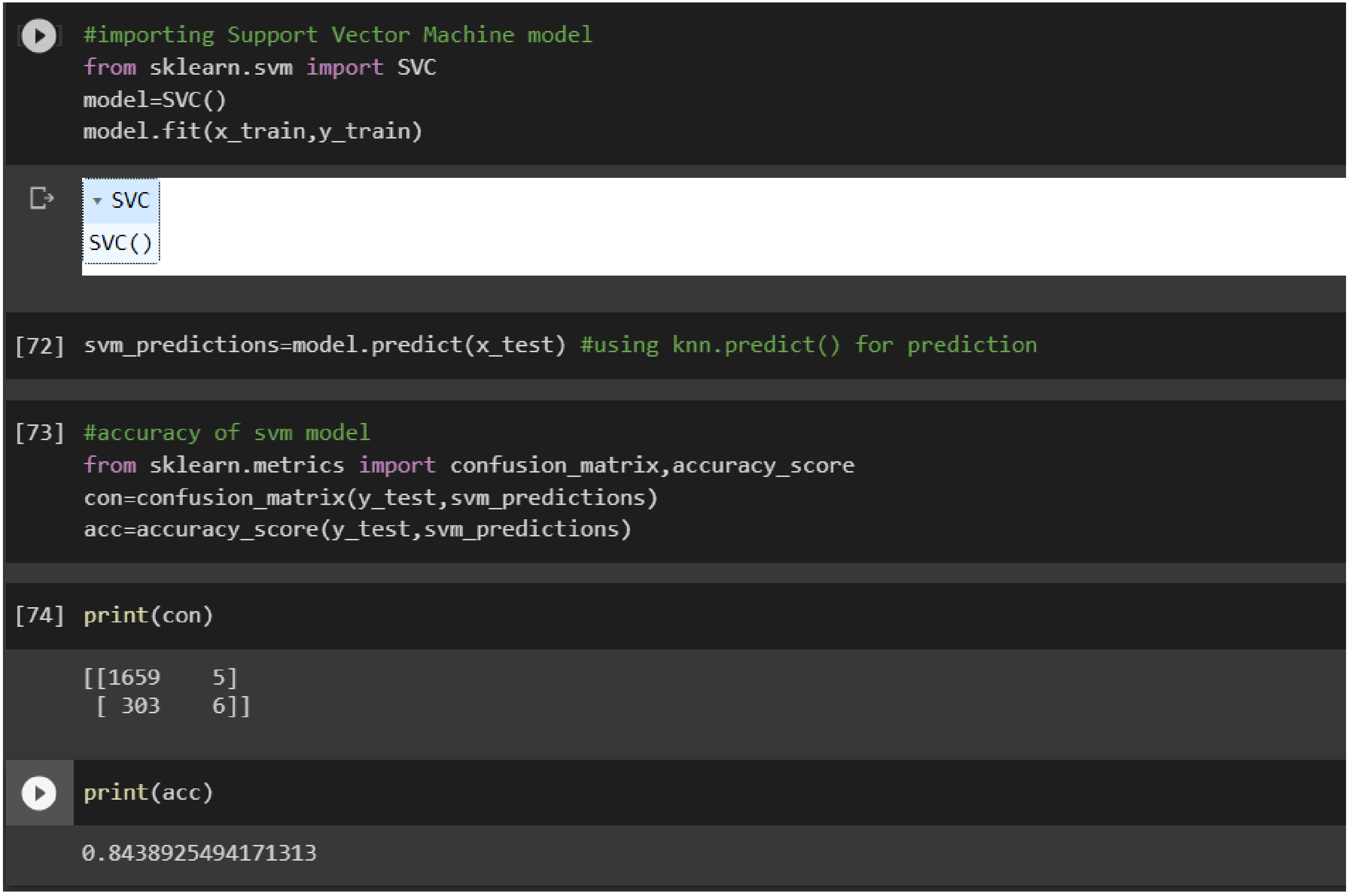
```
#accuracy of random Forest model
from sklearn.metrics import confusion_matrix,accuracy_score
CM=confusion_matrix(y_test,prediction)
ACC=accuracy_score(y_test,prediction)
```

```
[69] print(CM) #checking the performance of model using comfusion matrix

     [[1592   72]
      [ 134  175]]
```

```
[70] print(ACC) #checking the accuracy of the model using accuracy score

     0.895590471363406
```

-In the above model we have obtained accuracy of 89% using RandomForestClassifier which is more accurate than DecisionTreeClassifier.

- Now model using SVM.



```
#importing Support Vector Machine model
from sklearn.svm import SVC
model=SVC()
model.fit(x_train,y_train)
```

```
▾ SVC
SVC()
```

```
[72] svm_predictions=model.predict(x_test) #using knn.predict() for prediction
```

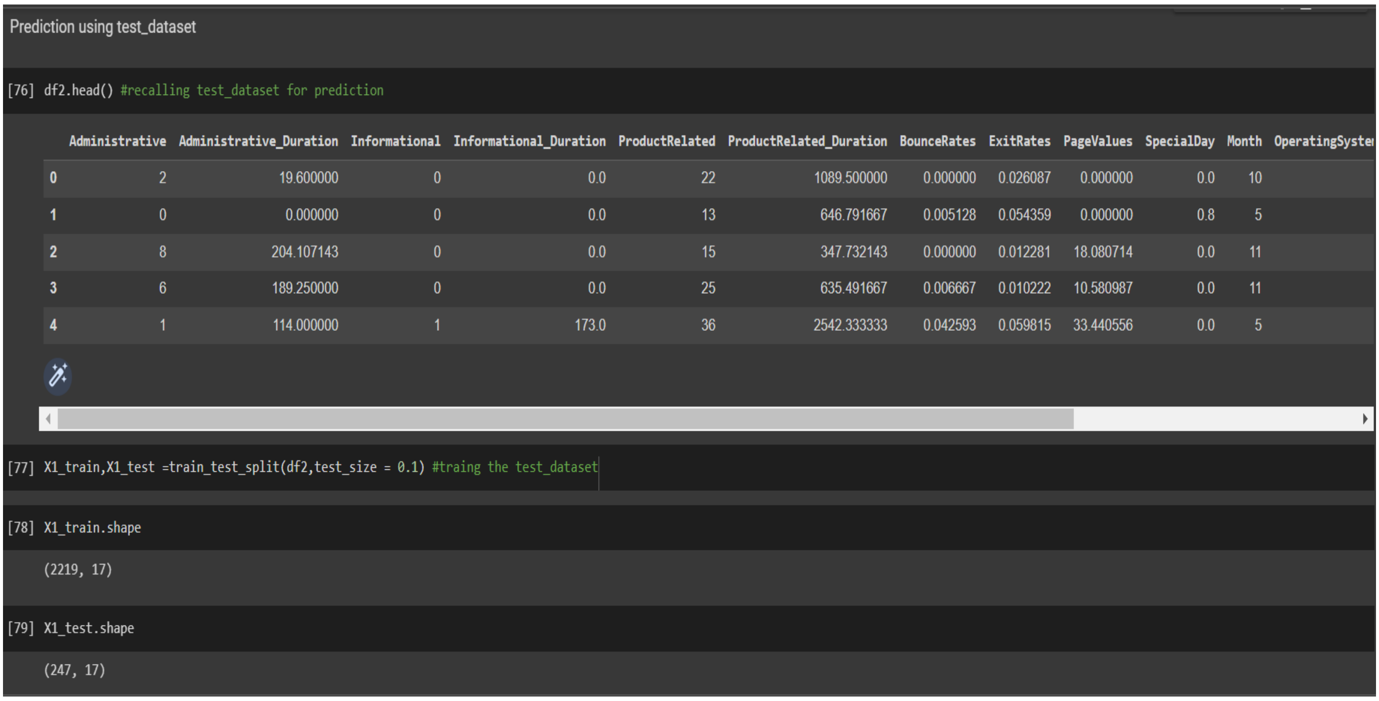
```
[73] #accuracy of svm model
     from sklearn.metrics import confusion_matrix,accuracy_score
     con=confusion_matrix(y_test,svm_predictions)
     acc=accuracy_score(y_test,svm_predictions)
```

```
[74] print(con)

     [[1659    5]
      [ 303    6]]
```

```
print(acc)

0.8438925494171313
```

-Here in this model we have obtained accuracy of 84% which is same as DecisionTreeClassifier model.

- Step-8: Recalling test_dataset as df2 & then splitting into test & train sets as X1_test & X1_train.



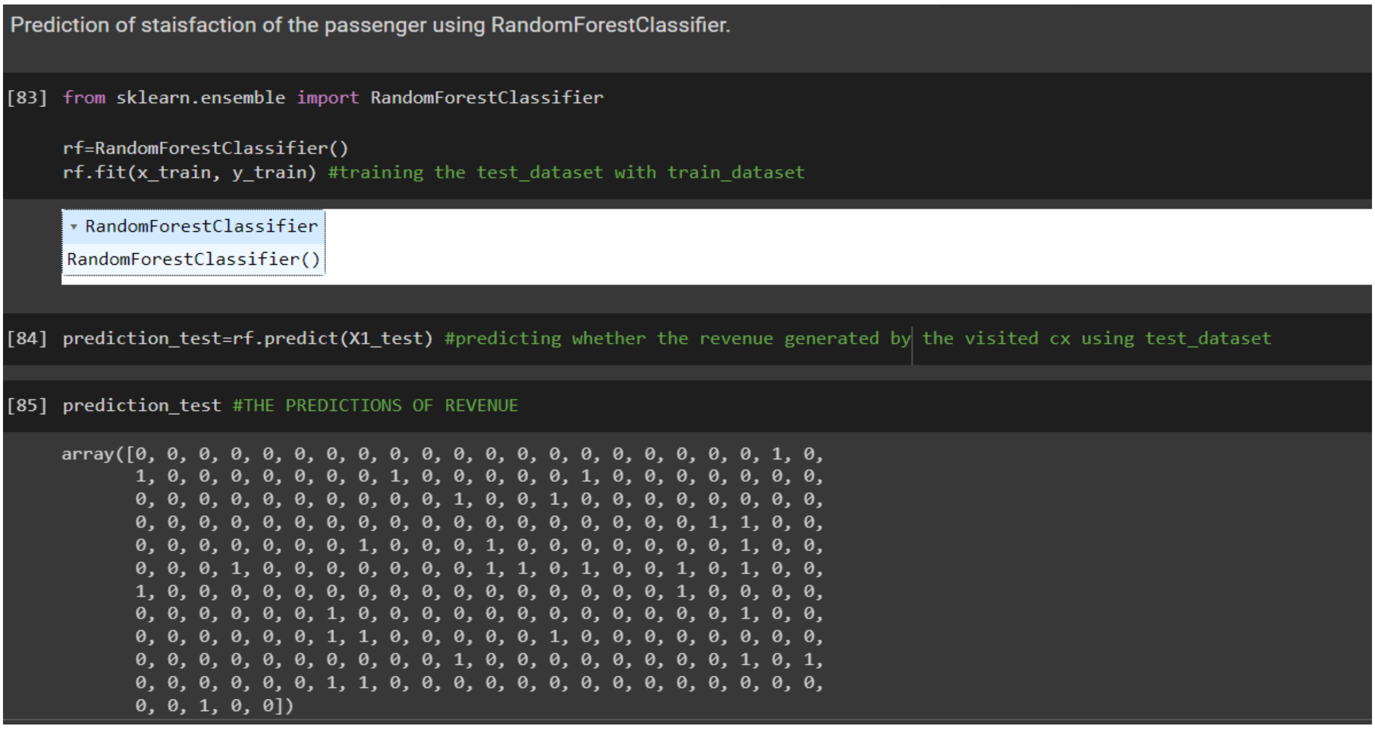
Prediction using test_dataset

```
[76] df2.head() #recalling test_dataset for prediction
```

	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	PageValues	SpecialDay	Month	OperatingSyster
0	2	19.600000	0	0.0	22	1089.500000	0.000000	0.026087	0.000000	0.0	10	
1	0	0.000000	0	0.0	13	646.791667	0.005128	0.054359	0.000000	0.8	5	
2	8	204.107143	0	0.0	15	347.732143	0.000000	0.012281	18.080714	0.0	11	
3	6	189.250000	0	0.0	25	635.491667	0.006667	0.010222	10.580987	0.0	11	
4	1	114.000000	1	173.0	36	2542.333333	0.042593	0.059815	33.440556	0.0	5	

```
[77] X1_train,X1_test =train_test_split(df2,test_size = 0.1) #traing the test_dataset

[78] X1_train.shape

     (2219, 17)

[79] X1_test.shape

     (247, 17)
```

- Step-9: Predicting whether the revenue has been generated from the customers using RandomForestClassifier model for test_dataset because it has the more accuracy percentage as compare to DecisionTree & SVM.



Prediction of staisfaction of the passenger using RandomForestClassifier.

```
[83] from sklearn.ensemble import RandomForestClassifier

     rf=RandomForestClassifier()
     rf.fit(x_train, y_train) #training the test_dataset with train_dataset

     ▾ RandomForestClassifier
     RandomForestClassifier()

[84] prediction_test=rf.predict(X1_test) #predicting whether the revenue generated by the visited cx using test_dataset

[85] prediction_test #THE PREDICTIONS OF REVENUE

     array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
            1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0,
            1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
            0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1,
            0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 1, 0, 0])
```

-Prediction(s) whether the revenue generated or not for test_dataset.

- Conclusion: From this project we have analysed and visualized what are trends of revenue from the customer(s) spending their time for online shopping what are there needs. So that the seller can avail the stocks of demanding product and also the backend team can suggest customer(s) more relevant products as per customers need.

THANK YOU