Project on Online shopping intention prediction using Machine Learning

- **Aim:-**To create a Data science Project, where we will be predicting the Online shopping intention. our objective here is to build a Machine Learning model that can help in predicting whether a customer will purchase or not, Prediction Online shopping intention with help of :-
  Special Day, Bounce Rate, Administrative.
  Steps to be taken in the project is sub-divided into the following sections. These are:
  - Importing the libraries such as 'numpy', 'pandas','sklearn. model' etc.
  - Loading Dataset as a CSV file for training & testing the models.
  - Splitting the data set into independent & dependent sets.
  - Checking if still any null values or any other data types other than float and integers are present into the dataset or not.
  - Importing the train_test_split model from sklearn.model for splitting data into train & test sets.
  - Applying the different kinds of ML Algorithms .which gives Best accuracy of model.
  - Also checking with new data set for predicting the values.
- Steps of creating ML model:-
- Importing numpy as np & pandas as pd for loading and reading the data-set & using matplotlib.pyplot and Seaborn for visualization of data.

```
[1]
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as snr
```

- Loading the csv-dataset in the variable name 'data_train' Then viewing the data with data_train.head()

```
[5] data_train=pd.read_csv('/content/training_data.csv')
    data_train.head()
```

d	ProductRelated_Duration	BounceRates	ExitRates	PageValues	SpecialDay	Month	OperatingSystems	Browser	Region	TrafficType	VisitorType	Weekend	Revenue
2	354.000000	0.000000	0.018182	0.0	0.0	May	2	7	1	2	New_Visitor	True	0
8	764.666667	0.025000	0.043750	0.0	0.0	Nov	3	2	4	10	Returning_Visitor	False	0
9	128.500000	0.036364	0.081818	0.0	0.0	Jul	3	2	1	3	Returning_Visitor	True	0
5	198.000000	0.000000	0.014286	0.0	0.0	May	3	3	4	2	New_Visitor	True	0
0	1295.008333	0.000893	0.015595	0.0	0.0	Nov	3	2	4	2	Returning_Visitor	True	1

- Checking the data such as number of columns, rows and type of data(float,integer) with help of data_train.info()

We observe that the above data have integer, object,bool and float.

```
data_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9864 entries, 0 to 9863
Data columns (total 18 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Administrative           9864 non-null   int64
 1   Administrative_Duration  9864 non-null   float64
 2   Informational            9864 non-null   int64
 3   Informational_Duration   9864 non-null   float64
 4   ProductRelated           9864 non-null   int64
 5   ProductRelated_Duration  9864 non-null   float64
 6   BounceRates              9864 non-null   float64
 7   ExitRates                9864 non-null   float64
 8   PageValues               9864 non-null   float64
 9   SpecialDay               9864 non-null   float64
 10  Month                    9864 non-null   object
 11  OperatingSystems         9864 non-null   int64
 12  Browser                  9864 non-null   int64
 13  Region                   9864 non-null   int64
 14  TrafficType              9864 non-null   int64
 15  VisitorType              9864 non-null   object
 16  Weekend                  9864 non-null   bool
 17  Revenue                  9864 non-null   int64
dtypes: bool(1), float64(7), int64(8), object(2)
memory usage: 1.3+ MB
```

```
[7] data_train.shape

(9864, 18)
```

Train data have 9864 Rows and 18 columns

- ❖ Now checking data have Nan value or not.

```
#missing values calumns wise
data_train.isnull().sum(axis=0).sort_values()

Administrative             0
VisitorType                0
TrafficType                0
Region                     0
Browser                    0
OperatingSystems           0
Month                      0
SpecialDay                 0
PageValues                 0
ExitRates                  0
BounceRates                0
ProductRelated_Duration    0
ProductRelated             0
Informational_Duration     0
Informational              0
Administrative_Duration    0
Weekend                    0
Revenue                    0
dtype: int64
```

We observe that the above data have not Nan value.

- ❖ Now,Main focus convert the categorical data into Numerical data with help of one hot encoding method.

```
[9] # we convert the categorical data into numerical data
    my_data=pd.get_dummies(data_train,columns=['Month','VisitorType','Weekend'],drop_first=True)
    my_data.head()
```

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

5 rows × 27 columns

```
10] my_data.info()

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 9864 entries, 0 to 9863
    Data columns (total 27 columns):
     #   Column                         Non-Null Count  Dtype
    ---  ------                         --------------  -----
     0   Administrative                 9864 non-null   int64
     1   Administrative_Duration        9864 non-null   float64
     2   Informational                  9864 non-null   int64
     3   Informational_Duration         9864 non-null   float64
     4   ProductRelated                 9864 non-null   int64
     5   ProductRelated_Duration        9864 non-null   float64
     6   BounceRates                    9864 non-null   float64
     7   ExitRates                      9864 non-null   float64
     8   PageValues                     9864 non-null   float64
     9   SpecialDay                     9864 non-null   float64
     10  OperatingSystems               9864 non-null   int64
     11  Browser                        9864 non-null   int64
     12  Region                         9864 non-null   int64
     13  TrafficType                    9864 non-null   int64
     14  Revenue                        9864 non-null   int64
     15  Month_Dec                      9864 non-null   uint8
     16  Month_Feb                      9864 non-null   uint8
     17  Month_Jul                      9864 non-null   uint8
     18  Month_June                     9864 non-null   uint8
     19  Month_Mar                      9864 non-null   uint8
     20  Month_May                      9864 non-null   uint8
     21  Month_Nov                      9864 non-null   uint8
     22  Month_Oct                      9864 non-null   uint8
     23  Month_Sep                      9864 non-null   uint8
     24  VisitorType_Other              9864 non-null   uint8
     25  VisitorType_Returning_Visitor  9864 non-null   uint8
     26  Weekend_True                   9864 non-null   uint8
    dtypes: float64(7), int64(8), uint8(12)
```

Finally we observe the data are fully cleaned.

- Now we check the data dependency.

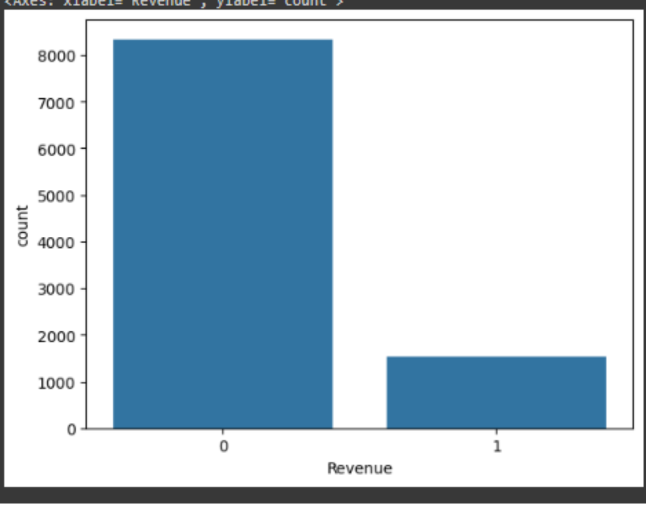
```
1] plt.figure(figsize=(15,15))
   snr.heatmap(data_train.corr(),annot=True)

   <ipython-input-11-0a947d04f4e7>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will defau
     snr.heatmap(data_train.corr(),annot=True)
   <Axes: >
```

0s completed at 11:43 PM

We see that data dependent each other.

❖ Visualizing the Online shopping intention like Special Day, Bounce Rate, Administrative.

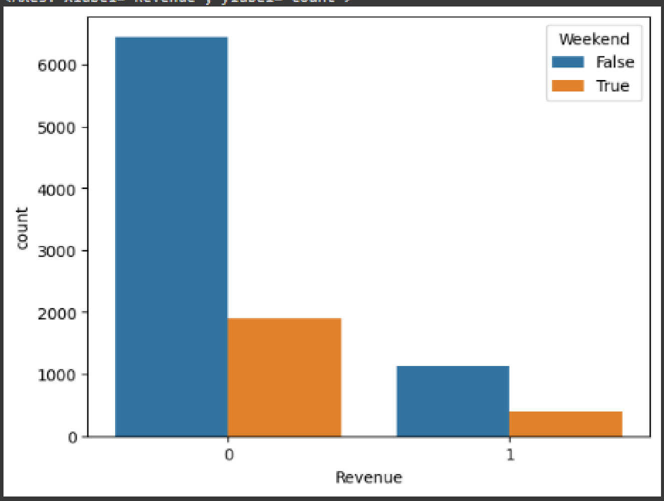
```
[12] snr.countplot(x='Revenue',data=data_train)

<Axes: xlabel='Revenue', ylabel='count'>
```



As per Visualizing the above graph, customer intention is less in Online shopping

```
[13] snr.countplot(x='Revenue',hue='Weekend',data=data_train)
```

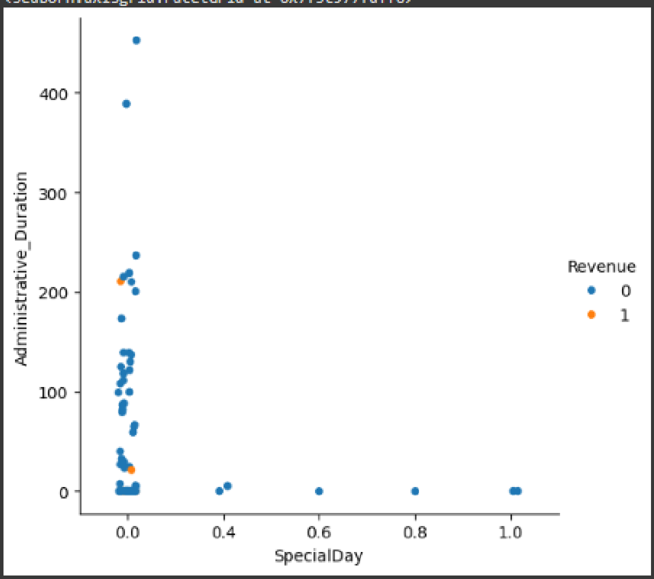
<Axes: xlabel='Revenue', ylabel='count'>



As per Visualizing the above graph, customer intension in weekend ..

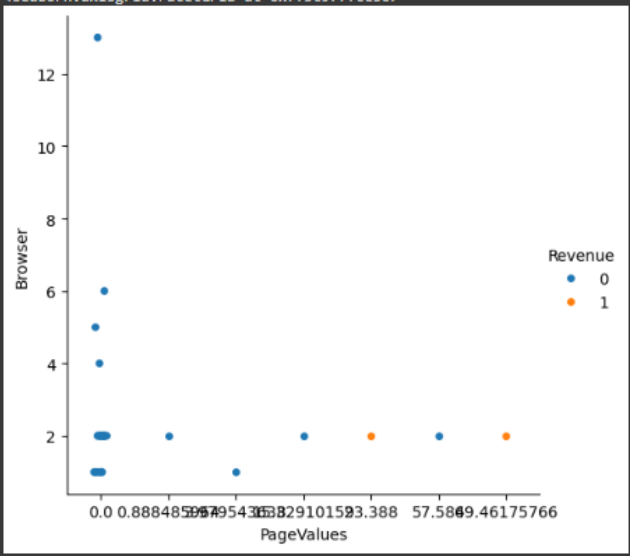
```
[14] snr.catplot(y='Administrative_Duration',hue='Revenue',x='SpecialDay',data=data_train[30:100])
```

<seaborn.axisgrid.FacetGrid at 0x7f3c977fdff0>



As per Visualizing the above graph, people intension in special day..

```
[21] snr.catplot(y='Browser',hue='Revenue',x='PageValues',data=data_train[70:100])
<seaborn.axisgrid.FacetGrid at 0x7f3c977fce50>
```



We observed that customer more search in values.

After visualization of data, we predict Online shopping intention using Machine Learning .

- Splitting the dataset into dependent(y) & independent(x) sets

```
[22] #spilting the data into dependent and independent
     x=my_data.drop(columns=['Revenue'])
     y=my_data['Revenue' ]
```

- Importing train_test_split from sklearn.model library for splitting the data into train and test sets. (we consider train dataset).

spilt the data into train test split

```
[23] from sklearn.model_selection import train_test_split
     x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.85)
```

- Importing logistic regression from sklearn Libaray & then activating the Machine learning Model .Then used regression.fit() to training the model by providing train & test sets as x & y. And then predicted the trained model with help of MLM & the checked score as regression.score(x,y)

```
from sklearn.linear_model import LogisticRegression
regression=LogisticRegression()
```

```
[25] regression.fit(x_train,y_train)

/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
▾ LogisticRegression
LogisticRegression()
```

❖ Checking the accuracy with help of confusion Matrix.

```
[26] y_predict_regression=regression.predict(x_test)

[27] from sklearn.metrics import confusion_matrix,accuracy_score
     ac=accuracy_score(y_test,y_predict_regression)
     cm=confusion_matrix(y_test,y_predict_regression)

[28] print(ac)
     print(cm)

     0.8810810810810811
     [[1220   31]
      [ 145   84]]
```

In the above model we can see that the accuracy obtained is 88%

➢ Now applying new algorithm Knn, then checked score.

```
[29] from sklearn.neighbors import KNeighborsClassifier
     knn=KNeighborsClassifier(n_neighbors=5) #where k=5

[30] knn.fit(x_train,y_train)

     ▾ KNeighborsClassifier
     KNeighborsClassifier()

[31] y_predict_knn=knn.predict(x_test)

[32] ac=accuracy_score(y_test,y_predict_knn)
     cm=confusion_matrix(y_test,y_predict_knn)

[33] print(ac)
     print(cm)

     0.8581081081081081
     [[1210   41]
      [ 169   60]]
```

we can see that the accuracy obtained is 85%

➢ Now applying new algorithm DecisionTree , then checked score.

```
from sklearn.tree import DecisionTreeClassifier
tree=DecisionTreeClassifier()

[36] tree.fit(x_train,y_train)

▾ DecisionTreeClassifier
DecisionTreeClassifier()

[37] y_predict_tree=tree.predict(x_test)

ac=accuracy_score(y_test,y_predict_tree)
cm=confusion_matrix(y_test,y_predict_tree)

[39] print(ac)
     print(cm)

0.8662162162162163
[[1145  106]
 [  92  137]]
```

we can see that the accuracy obtained is 86%

- ➢ Now applying new algorithm RandomForest , then checked score.

```
[40] from sklearn.ensemble import RandomForestClassifier
     random=RandomForestClassifier()

[41] random.fit(x_train,y_train)
     y_predict_random=random.predict(x_test)

[42] ac=accuracy_score(y_test,y_predict_random)
     cm=confusion_matrix(y_test,y_predict_random)
     print(ac)
     print(cm)

0.902027027027027
[[1204   47]
 [  98  131]]
```

we can see that the accuracy obtained with Random forest 90%

We see the accuracy is good but less than Decision Tree and Random forest algorithms.

- ➢ Now we compare all algorithms with accuracy

Algorithms	accuracy
Logistic regression	88%
KNN	85%
Random Forest classifier	90%
Decision Tree classifier	86%

Random Forest algorithms is better than KNN , Decision Tree and Logistic regression.

- ➢ Now recalling the test data set.
- ❖ Loading the csv-dataset in the variable name ‘test_data’ Then viewing the data with test_data.head()

```
test_data=pd.read_csv('/content/testing_data.csv')
test_data.head()
```

	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	PageVal
0	2	19.600000	0	0.0	22	1089.500000	0.000000	0.026087	0.000
1	0	0.000000	0	0.0	13	646.791667	0.005128	0.054359	0.000
2	8	204.107143	0	0.0	15	347.732143	0.000000	0.012281	18.080
3	6	189.250000	0	0.0	25	635.491667	0.006667	0.010222	10.580
4	1	114.000000	1	173.0	36	2542.333333	0.042593	0.059815	33.440

- Splitting into test & train sets as x1_test & x1_train. Then we find the Airline customer satisfaction using Machine Learning(Decision Tree classifier)

```
[49] #spilting the data into training and testing set.
     from sklearn.model_selection import train_test_split
     x1_train, x1_test = train_test_split(my_test_data, test_size=0.1, random_state=0)
```

Applying Decision Tree classifier algorithms for predictions.

```
[52] y_predict_test=random.predict(x1_test)
```

```
[53] y_predict_test

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

Conclusion:- In this test data set we analysed the data we found the less customer intention in Online shopping intention

Thank you