Project on Airline Passenger satisfaction using Machine Learning Models

- AIM: To create a Data Science project, where we'll be predicting the Satisfaction of the Passenger's on the services provided by Airlines using machine learning models with the help of csv-dataset(s) provided which contains different-different aspects of services which is being provided to the passengers.

- 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 Passenger Satisfaction 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 passengers were satisfied 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().

```
data_train=pd.read_csv('/content/1678723001545_train_dataset.csv')
data_test=pd.read_csv('/content/1678722996077_test_dataset.csv')
```

```
data_train.head()
```

	Age	Flight Distance	Inflight wifi service	Departure/Arrival time convenient	Ease of Online booking	Gate location	Food and drink	Online boarding	Seat comfort	Inflight entertainment	...	Departure Delay in Minutes	Arrival Delay in Minutes	Gender_Female	Gende
0	35	971	3	4	5	4	2	3	3	2	...	373	358	0	
1	32	1092	0	0	0	3	1	0	1	1	...	0	0	0	
2	46	2915	0	5	0	5	3	4	5	1	...	0	0	1	
3	56	2556	4	4	4	4	4	4	4	3	...	19	18	0	
4	54	468	1	4	1	4	4	1	4	4	...	0	0	0	

5 rows × 26 columns

-Viewing train_dataset

```
[20] data_test.head()
```

	Age	Flight Distance	Inflight wifi service	Departure/Arrival time convenient	Ease of Online booking	Gate location	Food and drink	Online boarding	Seat comfort	Inflight entertainment	...	Cleanliness	Departure Delay in Minutes	Arrival Delay in Minutes	Gender_
0	46	1622	1	1	1	1	5	5	5	4	...	5	276	270	
1	45	552	3	1	3	4	4	5	5	5	...	5	0	0	
2	52	435	2	2	2	2	3	4	5	4	...	4	0	0	
3	41	655	2	5	2	3	4	2	1	4	...	4	0	0	
4	39	337	2	0	1	3	5	1	5	5	...	5	0	0	

5 rows × 25 columns

-Viewing test_dataset

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

```
[21] #cleaning the train_dataset by changing the categorical values into numerical values
     data_train['satisfaction'] = data_train['satisfaction'].replace('neutral or dissatisfied', 0)
     data_train['satisfaction'] = data_train['satisfaction'].replace('satisfied', 1)
```

-Changing the string values to integer values (train_dataset).

- Step-4: Splitting the dataset into dependent & independent sets (taken only train dataset).

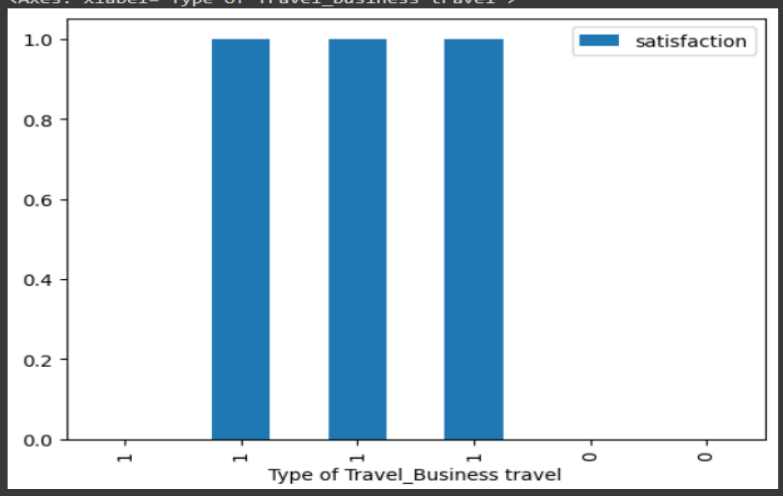
```
#splitting the data into independent & dependent category
x=df1.drop(['satisfaction'],axis=1)
y=df1['satisfaction']
```

Step-5: Visualising the train dataset using Python to obtain some insights of passenger satisfaction.

Q. Considering the first 6 passengers, compute how many of them were satisfied with the airline services after taking a business travel

```
df1.iloc[:6].plot(x='Type of Travel_Business travel',y='satisfaction',kind='bar')
```

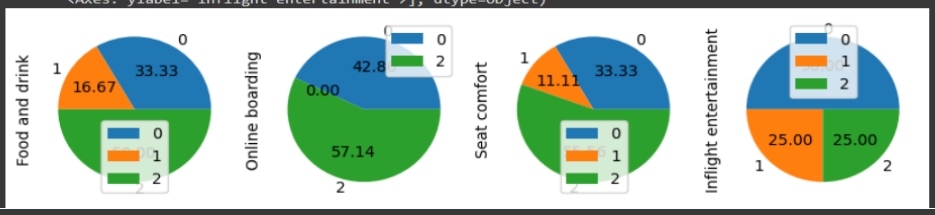
```
<Axes: xlabel='Type of Travel_Business travel'>
```



Q. Creating a pie chart drive the contributions of 'Food and drink', 'Online boarding', 'Seat comfort' and 'Inflight entertainment' features in providing satisfactory services for the first 3 passengers.

```
[23] df1.iloc[:3,6:10].plot.pie(subplots=True,figsize=(10,10),fontsize=10,autopct='%.2f') #here autopact used for showing certain number of decimal points
```

```
array([<Axes: ylabel='Food and drink'>, <Axes: ylabel='Online boarding'>,
       <Axes: ylabel='Seat comfort'>,
       <Axes: ylabel='Inflight entertainment'>], dtype=object)
```



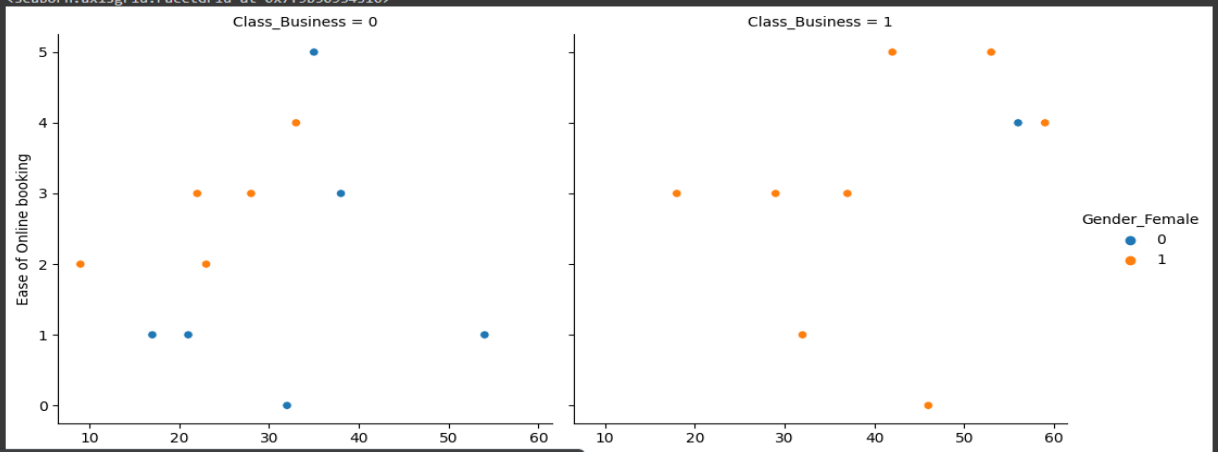
Q. Nowadays e-ticket or online flight tickets have replaced the print ones. Passengers and airline company makes a contract through e-tickets. In this context drive intuition from the dataset for the first 20 passengers as to how age, gender and class of flight(business/economy) are related.

It will help you to understand which age group of passengers boarding which class of flight are more comfortable in booking tickets online

```
[25] df1_new=df1.iloc[0:20]
```

```
[26] snr.relplot(y='Ease of Online booking',x='Age',hue='Gender_Female',col='Class_Business',data=df1_new)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f9b50934310>
```



- Step-6: I have also Visualized the level of satisfaction & other aspects using tableau. You can see the detailed insights on public tableau by clicking here

- Step-7: 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-8: Importing DecisionTreeClassifier from sklearn.model & 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.

```
#importing DecissionTree Classifier
from sklearn.tree import DecisionTreeClassifier
tree=DecisionTreeClassifier()
```

```
[27] tree.fit(x_train,y_train) #using fit() for training
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

- Step-9: Predicting the trained model & the checked accuracy of the model using confusion_matrix & accuracy_score.

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

+ Code + Text

```
[29] #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)
```

```
[30] print(cm) #checking the performance of model using comfusion matrix
```

```
[[5349  332]
 [ 337 4135]]
```

```
[31] print(acc) #checking the accuracy of the model using accuracy score
```

```
0.934108145375751
```

-In the above model we can see that the accuracy is only 93% which is quite good.

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

```
[32] #importing RandomForestClassifier
     from sklearn.ensemble import RandomForestClassifier
     rf=RandomForestClassifier()

[33] rf.fit(x_train,y_train) #using fit() for training

     RandomForestClassifier
     RandomForestClassifier()
```

```
[34] 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)

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

     [[5517  164]
      [ 311 4161]]

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

     0.9532157982862208
```

-From the above model we obtained accuracy of 95% using RandomForestClassifier which is more accurate than DecisionTreeClassifier.

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

[45] model.fit(x_train,y_train)

     SVC
     SVC()

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

[48] #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)

[49] print(con)

     [[4726  955]
      [2431 2041]]

[51] print(acc)

     0.666502511572934
```

-From the above model we have obtained accuracy of 66% using SVM which is not so very accurate as compare to DT & RF models.

- Step-10: Recalling test_dataset as df2 & then splitting into test & train sets as X1_test & X1_train.

```
df2.head() #recalling test_dataset for prediction
```

	Age	Flight Distance	Inflight wifi service	Departure/Arrival time convenient	Ease of Online booking	Gate location	Food and drink	Online boarding	Seat comfort	Inflight entertainment	...	Cleanliness	Departure Delay in Minutes	Arrival Delay in Minutes	Gender_I
0	46	1622	1	1	1	1	5	5	5	4	...	5	276	270	
1	45	552	3	1	3	4	4	5	5	5	...	5	0	0	
2	52	435	2	2	2	2	3	4	5	4	...	4	0	0	
3	41	655	2	5	2	3	4	2	1	4	...	4	0	0	
4	39	337	2	0	1	3	5	1	5	5	...	5	0	0	

5 rows × 25 columns

```
[53] X1_train,X1_test =train_test_split(df2,test_size = 0.2) #traing the test dataset
```

```
[54] X1_train.shape
```

```
(17404, 25)
```

```
[55] X1_test.shape
```

```
(4352, 25)
```

- Step-11: Predicting the satisfaction of passengers using RandomForestModel & SVM model for test_dataset.

Prediction of staisfaction of the passenger using RandomForestClassifier.

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

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

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
[57] predict_test=rf.predict(X1_test) ##predicting the satisfaction level of the passenger using test_dataset
```

```
[58] predict_test ##THE PREDICTIONS OF SATISFACTION
```

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

-Test analysis of RandomForestModel

Prediction satisfaction of passenger using SVM model.

```
from sklearn.svm import SVC
model=SVC()
model.fit(x_train,y_train)
```

```
SVC()
```

```
[60] predict_model=model.predict(X1_test)
```

```
[61] predict_model
```

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

-Test analysis of SVM model

- From the above two test models we can consider RandomForestClassifier over SVM because the accuracy was more in RandomForest as compared to SVM.

➢ <u>Conclusion:</u> In the test_dataset where the satisfaction of passenger's needs to predicted, there we can use the predictions of RandomForestClassifier because it have the better accuracy of 95% whereas the accuracy rate of SVM was only 66%.