به نام خدا

عنوان پروژه :

درخت تصمیم گیری

شرح مراحل :

1- Data cleaning
2- Feature Engineering
3- Create Decision Tree Learning Model
4- Train the Model with Restaurant data and Test it
5- Train the Model with Airplane data and Test it

1- Data cleaning :

در این مرحله بایستی دیتاهایی که داریم به فرم عدد در بیایند و داده هاییکه دارای یک بازه گسترده هستند به چند بازه تبدیل بشوند .

- برای دیتای مربوط به رستوران اینکار به صورت دستی انجام شد و به هر لیبل یک عدد اختصاص داده شد ؛ برای مثال برای اتریبیوت Wait_Stimate که دارای 4 بازه زمانی بود به هر بازه به ترتیب اعداد 1 تا 4 اختصاص داده شد .
- Ariplane_Data :

تقسیم بندی ای که من برای دیتایی سوال انتخاب کردم دارای نسبت 60 به 40 برای دیتای train و دیتایtest است .

```
test = pa.read_csv("Airplane_Test.csv")
train = pa.read_csv("Airplane_Train.csv")
test = test.drop("Unnamed: 0",axis=1)
train = train.drop("Unnamed: 0",axis=1)
model_info = [train , test ]
Restuarnt_Data = pa.read_csv("Resturant.csv")
for data in model_info :
    data['satisfaction'] = data['satisfaction'].map({'neutral or dissatisfied' : 0 ,'satisfied' :1} )
    data['Arrival Delay in Minutes'] = data['Arrival Delay in Minutes'].fillna(0)
```

ابتدا مقدارهای ستون satisfaction را به فرم عددی در می اوریم و مقادیر nan را از ستون arrival dely به مقدار 0 تغییر میدهیم .

```
model_info = [train , test ]
for data in model_info :
    # #map goal column to 1 or 0
    # data['satisfaction'] = data['satisfaction'].map({'neutral or dissatisfied' : 0 ,'satisfied' :1} )

    # classify customer type
    data['Customer Type'] = data['Customer Type'].map({'Loyal Customer' : 1 ,'disloyal Customer' :0 } )

    # classify type of travel
    data['Type of Travel'] = data['Type of Travel'].map({'Business travel' : 0 ,'Personal Travel' :1 } )
    # classify Class
    data['Class'] = data['Class'].map({'Business' : 0 ,'Eco' :1 ,'Eco Plus' :2 } )

    #classify Gender
    data['Gender'] = data['Gender'].map({'Male' : 0 ,'Female' :1 } )

    # classify age
    data.loc[data['Age'] <= 16  , 'Age']                          = 0
    data.loc[ (data['Age'] > 16 ) & (data['Age'] <= 32 ) , 'Age'] = 1
    data.loc[ (data['Age'] > 32 ) & (data['Age'] <= 48 ) , 'Age'] = 2
    data.loc[ (data['Age'] > 48 ) & (data['Age'] <= 64 ) , 'Age'] = 3
    data.loc[ (data['Age'] > 64 ) & (data['Age'] <= 80 ) , 'Age'] = 4
    data.loc[ (data['Age'] > 80 ) , 'Age']                        = 5
```

```
data.loc[ (data['Flight Distance'] <= 1000 ) ,  'Flight Distance']                                    = 0
data.loc[ (data['Flight Distance'] > 1000 ) & (data['Flight Distance'] <= 2000 ) ,  'Flight Distance']  = 1
data.loc[ (data['Flight Distance'] > 2000 ) &  (data['Flight Distance'] <= 3000 ) , 'Flight Distance']  = 2
data.loc[ (data['Flight Distance'] > 3000 ) & (data['Flight Distance'] <= 4000 ) , 'Flight Distance']   = 3
data.loc[ (data['Flight Distance'] > 4000 ) , 'Flight Distance']                                       = 4
```

همچنین برای مابقی ستون هاییکه دارای مقادیر غیر عددی بودند نیز اینکار را انجام میدهیم . و ستون age که مقادیرش بازه بندی نشده است را بازه بندی میکنیم .

2- Feature Engineering :

تمام ستون هایی که داریم را از نظر میزان اثری که در نتیجه ستون هدف میگذراند مورد بررسی قرار میدهیم . به این صورت که اندازه میگیریم به ازای تمام گروه بندی هایی که این اتریبیوت برای ما ایجاد میکند به چه درصدی میزان ستون هدف برابر 1 است .

20. Cleanliness

```
print(train[['Cleanliness' , 'satisfaction']].groupby(['Cleanliness'] , as_index=False ).mean() )
```

	Cleanliness	satisfaction
0	0	0.000000
1	1	0.195652
2	2	0.207320
3	3	0.431140
4	4	0.533120
5	5	0.606125

این مقدار برای اتریبیوت cleanliness در عکس بالا قابل مشاهده است که در هر یک از 6 کلاسیکه این اتریبیوت ایجاد میکند این نسبت چگونه برقرار است .

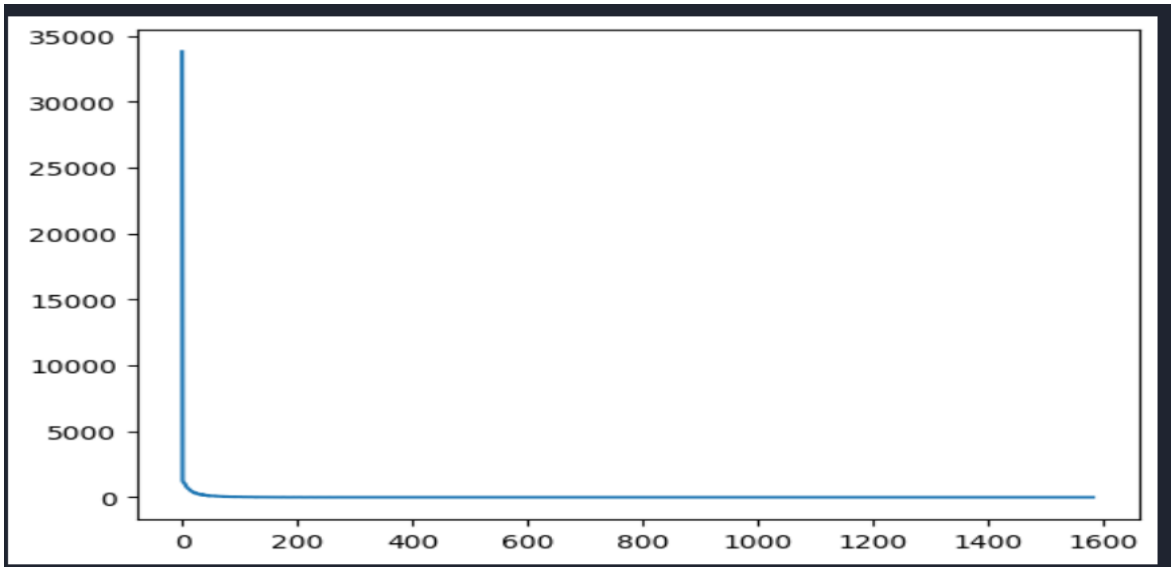
3- Feature Engineering :

حال که دیتا کامل تمیز شده و نسبت اتریبیوت به ستون هدف را داریم میتوانیم یک بعضی از ستون هارا حذف کنیم به علت اضافه کردن به عدم قطعیت در تقسیم بندی .

```
train = train.drop( "id", axis='columns' )
test = test.drop( "id", axis='columns' )
train = train.drop( "Departure/Arrival time convenient", axis='columns' )
test = test.drop( "Departure/Arrival time convenient", axis='columns' )
train = train.drop( "Gender", axis='columns' )
test = test.drop( "Gender", axis='columns' )
```

پس از بررسی نمودارهای دو ستون Departure Delay in Minutes و Arrival Delay in Minutes من به این نتیجه رسیدم که این دو ستون دارای مقادیر با شیب مشابه هستند . پس یک ستون جدید که مقادیرش حاصل میانگین این دو ستون است را ایجاد میکنیم بنام Departure/Arrival Deley in mintutes و جایگزین دو ستون قبلی میکنیم .

```
c0 =  train['Departure Delay in Minutes']
# print("max and min in age")
c1 = train['Arrival Delay in Minutes']
arrival_departure_delay = []
for i in range(len(c0)) :
    arrival_departure_delay.append( (int)((c0[i]+c1[i])/2) )
train.insert(21 , 'Arrival/Departure Delay in Minutes' , arrival_departure_delay)
```



حال دیتای ما اماده است برای ارائه به مدل یادگیریمان .

4- ساخت درخت :

برای ساخت درخت در هر مرحله از دیتایی که داریم با استفاده از متدimportance بهترین اتریبیوت را انتخاب میکنیم .
معیار انتخاب بهترین ویژگی به دو صورت Entroy و Gini index پیاده سازی شده که به عنوان اتریبیوت ورودی به صورت فانکشن پوینتر به درخت داده میشود .

Train the Model with Restaurant data and Test it

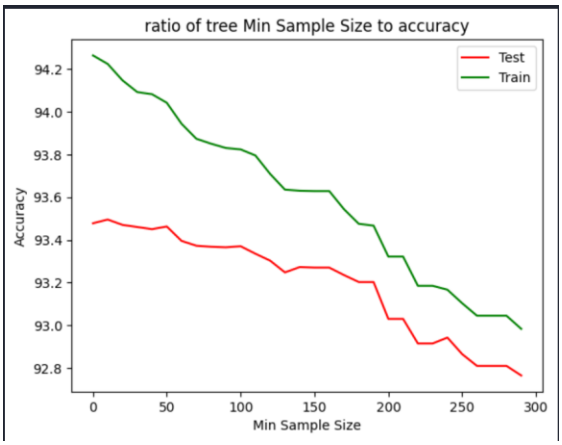
با entropy وgini index دقت 100 درصد را داریم .

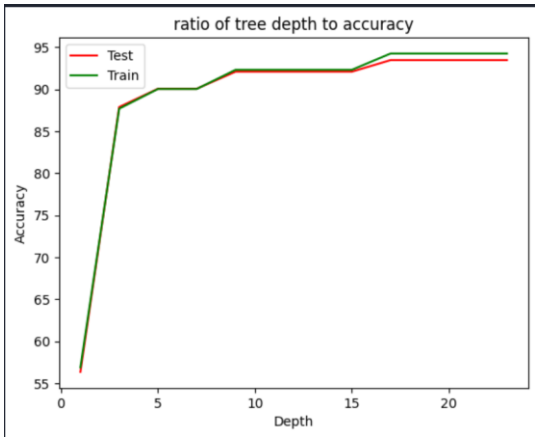
Train the Model with Airplane data and Test it

برای این بخش بایستی دو hyperparameter را مشخص کنیم که از overfit درخت جلوگیری کنیم .

1- Depth of Tree
2- Min Size of Sample

که برای بدست اوردن بهترین اعداد برای این دو پارامتر دو تست انجام دادم و نتایجش را بصورت دو نمودار بدست اوردم .





حالا با توجه به مقادیر بالا میتوان گفت که بهترین عمق 17 و کمترین سایز دیتای سمپل 10 میباشد .

نتیجه نهایی برای مدل :

Accuracy for train data : 94.22

Accuracy for test data : 93.494999