XGBoost Classification

September 13, 2025

0.0.1 Now lets go to XGBoost.

We dont have XGBoost in anaconda. so we should install it.

[1]:
```
import pandas as pd
import numpy as np
```

[2]:
```
df = pd.read_csv('hotel_bookings.csv')
df
```

[2]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights
0	Resort Hotel	0	342	2015	July	27	1	0
1	Resort Hotel	0	737	2015	July	27	1	0
2	Resort Hotel	0	7	2015	July	27	1	0
3	Resort Hotel	0	13	2015	July	27	1	0
4	Resort Hotel	0	14	2015	July	27	1	0
...	...	...	...	...	...	...	...	...
119385	City Hotel	0	23	2017	August	35	30	2
119386	City Hotel	0	102	2017	August	35	31	2
119387	City Hotel	0	34	2017	August	35	31	2
119388	City Hotel	0	109	2017	August	35	31	2
119389	City Hotel	0	205	2017	August	35	29	2

```
2                                1                          0
3                                1                          0
4                                1                          0
...                            ...                        ...
119385                          30                          2
119386                          31                          2
119387                          31                          2
119388                          31                          2
119389                          29                          2

        stays_in_week_nights  adults  ...  deposit_type  agent company  \
0                          0       2  ...    No Deposit    NaN     NaN
1                          0       2  ...    No Deposit    NaN     NaN
2                          1       1  ...    No Deposit    NaN     NaN
3                          1       1  ...    No Deposit  304.0     NaN
4                          2       2  ...    No Deposit  240.0     NaN
...                      ...     ...  ...           ...    ...     ...
119385                     5       2  ...    No Deposit  394.0     NaN
119386                     5       3  ...    No Deposit    9.0     NaN
119387                     5       2  ...    No Deposit    9.0     NaN
119388                     5       2  ...    No Deposit   89.0     NaN
119389                     7       2  ...    No Deposit    9.0     NaN

       days_in_waiting_list customer_type     adr  \
0                         0     Transient    0.00
1                         0     Transient    0.00
2                         0     Transient   75.00
3                         0     Transient   75.00
4                         0     Transient   98.00
...                     ...           ...     ...
119385                    0     Transient   96.14
119386                    0     Transient  225.43
119387                    0     Transient  157.71
119388                    0     Transient  104.40
119389                    0     Transient  151.20

        required_car_parking_spaces  total_of_special_requests  \
0                                 0                          0
1                                 0                          0
2                                 0                          0
3                                 0                          0
4                                 0                          1
...                             ...                        ...
119385                            0                          0
119386                            0                          2
119387                            0                          4
119388                            0                          0
```

```
119389                          0                          2

        reservation_status reservation_status_date
0                Check-Out                01-07-15
1                Check-Out                01-07-15
2                Check-Out                02-07-15
3                Check-Out                02-07-15
4                Check-Out                03-07-15
...                    ...                     ...
119385           Check-Out                06-09-17
119386           Check-Out                07-09-17
119387           Check-Out                07-09-17
119388           Check-Out                07-09-17
119389           Check-Out                07-09-17

[119390 rows x 32 columns]
```

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 32 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   hotel                           119390 non-null  object
 1   is_canceled                     119390 non-null  int64
 2   lead_time                       119390 non-null  int64
 3   arrival_date_year               119390 non-null  int64
 4   arrival_date_month              119390 non-null  object
 5   arrival_date_week_number        119390 non-null  int64
 6   arrival_date_day_of_month       119390 non-null  int64
 7   stays_in_weekend_nights         119390 non-null  int64
 8   stays_in_week_nights            119390 non-null  int64
 9   adults                          119390 non-null  int64
 10  children                        119386 non-null  float64
 11  babies                          119390 non-null  int64
 12  meal                            119390 non-null  object
 13  country                         118902 non-null  object
 14  market_segment                  119390 non-null  object
 15  distribution_channel            119390 non-null  object
 16  is_repeated_guest               119390 non-null  int64
 17  previous_cancellations          119390 non-null  int64
 18  previous_bookings_not_canceled  119390 non-null  int64
 19  reserved_room_type              119390 non-null  object
 20  assigned_room_type              119390 non-null  object
 21  booking_changes                 119390 non-null  int64
 22  deposit_type                    119390 non-null  object
 23  agent                           103050 non-null  float64
```

```
 24  company                         6797 non-null    float64
 25  days_in_waiting_list            119390 non-null  int64
 26  customer_type                   119390 non-null  object
 27  adr                             119390 non-null  float64
 28  required_car_parking_spaces     119390 non-null  int64
 29  total_of_special_requests       119390 non-null  int64
 30  reservation_status              119390 non-null  object
 31  reservation_status_date         119390 non-null  object
dtypes: float64(4), int64(16), object(12)
memory usage: 29.1+ MB
```

our traget is df['is_canceled']. We want to predict by another relevant datas that which customers will cancel the booking or not.

[5]:
```
df.describe()
```

[5]:

	is_canceled	lead_time	arrival_date_year
count	119390.000000	119390.000000	119390.000000
mean	0.370416	104.011416	2016.156554
std	0.482918	106.863097	0.707476
min	0.000000	0.000000	2015.000000
25%	0.000000	18.000000	2016.000000
50%	0.000000	69.000000	2016.000000
75%	1.000000	160.000000	2017.000000
max	1.000000	737.000000	2017.000000

	arrival_date_week_number	arrival_date_day_of_month
count	119390.000000	119390.000000
mean	27.165173	15.798241
std	13.605138	8.780829
min	1.000000	1.000000
25%	16.000000	8.000000
50%	28.000000	16.000000
75%	38.000000	23.000000
max	53.000000	31.000000

	stays_in_weekend_nights	stays_in_week_nights	adults
count	119390.000000	119390.000000	119390.000000
mean	0.927599	2.500302	1.856403
std	0.998613	1.908286	0.579261
min	0.000000	0.000000	0.000000
25%	0.000000	1.000000	2.000000
50%	1.000000	2.000000	2.000000
75%	2.000000	3.000000	2.000000
max	19.000000	50.000000	55.000000

	children	babies	is_repeated_guest
count	119386.000000	119390.000000	119390.000000

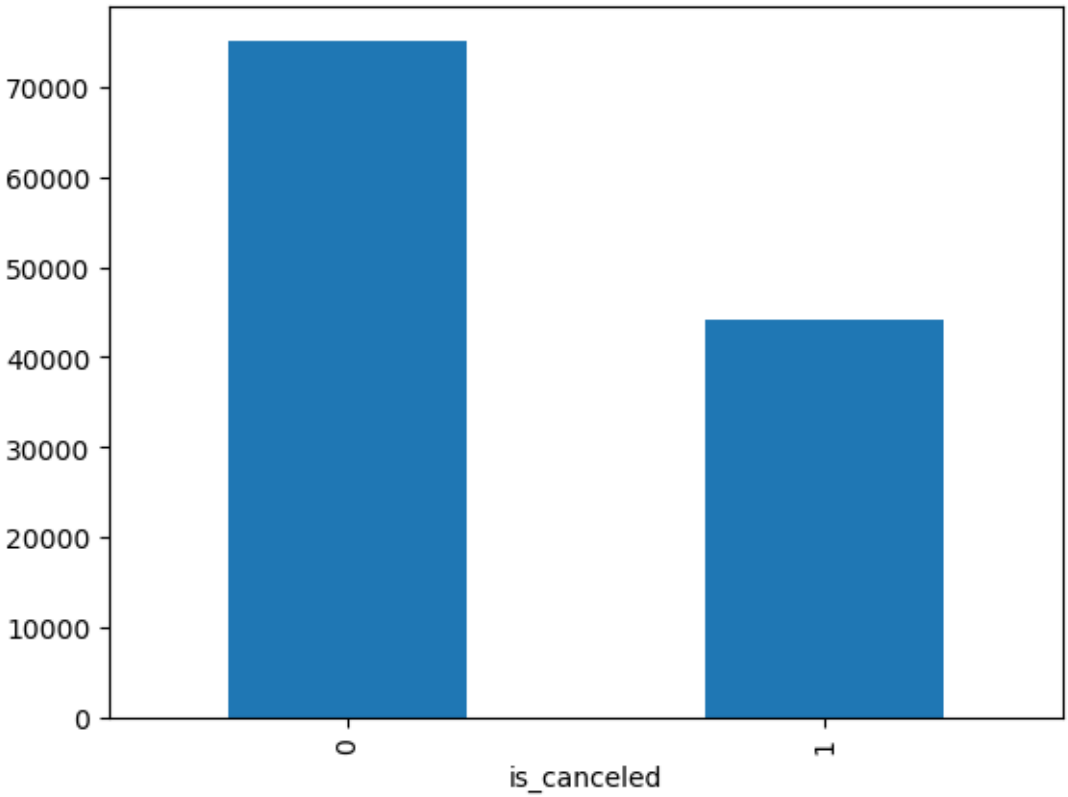
```
mean        0.103890       0.007949           0.031912
std         0.398561       0.097436           0.175767
min         0.000000       0.000000           0.000000
25%         0.000000       0.000000           0.000000
50%         0.000000       0.000000           0.000000
75%         0.000000       0.000000           0.000000
max        10.000000      10.000000           1.000000

       previous_cancellations  previous_bookings_not_canceled  \
count           119390.000000                   119390.000000
mean                 0.087118                        0.137097
std                  0.844336                        1.497437
min                  0.000000                        0.000000
25%                  0.000000                        0.000000
50%                  0.000000                        0.000000
75%                  0.000000                        0.000000
max                 26.000000                       72.000000

       booking_changes          agent      company  days_in_waiting_list  \
count    119390.000000  103050.000000  6797.000000         119390.000000
mean          0.221124      86.693382   189.266735              2.321149
std           0.652306     110.774548   131.655015             17.594721
min           0.000000       1.000000     6.000000              0.000000
25%           0.000000       9.000000    62.000000              0.000000
50%           0.000000      14.000000   179.000000              0.000000
75%           0.000000     229.000000   270.000000              0.000000
max          21.000000     535.000000   543.000000            391.000000

                adr  required_car_parking_spaces  total_of_special_requests
count  119390.000000                119390.000000              119390.000000
mean      101.831122                     0.062518                   0.571363
std        50.535790                     0.245291                   0.792798
min        -6.380000                     0.000000                   0.000000
25%        69.290000                     0.000000                   0.000000
50%        94.575000                     0.000000                   0.000000
75%       126.000000                     0.000000                   1.000000
max      5400.000000                     8.000000                   5.000000
```

[6]:
```
df['is_canceled'].value_counts().plot(kind='bar');
```



[7]:
```
df['is_canceled'].value_counts() / df['is_canceled'].count()
```

[7]:
```
is_canceled
0    0.629584
1    0.370416
Name: count, dtype: float64
```

[8]:
```
df_numeric = df.select_dtypes(include=['number'])  # keep only numeric columns
df_numeric.corr()
```

[8]:

	is_canceled	lead_time	arrival_date_year	\
is_canceled	1.000000	0.293123	0.016660	
lead_time	0.293123	1.000000	0.040142	
arrival_date_year	0.016660	0.040142	1.000000	
arrival_date_week_number	0.008148	0.126871	-0.540561	
arrival_date_day_of_month	-0.006130	0.002268	-0.000221	
stays_in_weekend_nights	-0.001791	0.085671	0.021497	
stays_in_week_nights	0.024765	0.165799	0.030883	
adults	0.060017	0.119519	0.029635	
children	0.005048	-0.037622	0.054624	
babies	-0.032491	-0.020915	-0.013192	

is_repeated_guest	-0.084793	-0.124410	0.010341
previous_cancellations	0.110133	0.086042	-0.119822
previous_bookings_not_canceled	-0.057358	-0.073548	0.029218
booking_changes	-0.144381	0.000149	0.030872
agent	-0.083114	-0.069741	0.063457
company	-0.020642	0.151464	0.259095
days_in_waiting_list	0.054186	0.170084	-0.056497
adr	0.047557	-0.063077	0.197580
required_car_parking_spaces	-0.195498	-0.116451	-0.013684
total_of_special_requests	-0.234658	-0.095712	0.108531

	arrival_date_week_number	\
is_canceled	0.008148	
lead_time	0.126871	
arrival_date_year	-0.540561	
arrival_date_week_number	1.000000	
arrival_date_day_of_month	0.066809	
stays_in_weekend_nights	0.018208	
stays_in_week_nights	0.015558	
adults	0.025909	
children	0.005518	
babies	0.010395	
is_repeated_guest	-0.030131	
previous_cancellations	0.035501	
previous_bookings_not_canceled	-0.020904	
booking_changes	0.005508	
agent	-0.031201	
company	-0.076760	
days_in_waiting_list	0.022933	
adr	0.075791	
required_car_parking_spaces	0.001920	
total_of_special_requests	0.026149	

	arrival_date_day_of_month	\
is_canceled	-0.006130	
lead_time	0.002268	
arrival_date_year	-0.000221	
arrival_date_week_number	0.066809	
arrival_date_day_of_month	1.000000	
stays_in_weekend_nights	-0.016354	
stays_in_week_nights	-0.028174	
adults	-0.001566	
children	0.014544	
babies	-0.000230	
is_repeated_guest	-0.006145	
previous_cancellations	-0.027011	
previous_bookings_not_canceled	-0.000300	

```
booking_changes                                 0.010613
agent                                           0.001487
company                                         0.044858
days_in_waiting_list                            0.022728
adr                                             0.030245
required_car_parking_spaces                     0.008683
total_of_special_requests                       0.003062

                                stays_in_weekend_nights  stays_in_week_nights  \
is_canceled                                   -0.001791              0.024765
lead_time                                      0.085671              0.165799
arrival_date_year                              0.021497              0.030883
arrival_date_week_number                       0.018208              0.015558
arrival_date_day_of_month                     -0.016354             -0.028174
stays_in_weekend_nights                        1.000000              0.498969
stays_in_week_nights                           0.498969              1.000000
adults                                         0.091871              0.092976
children                                       0.045793              0.044203
babies                                         0.018483              0.020191
is_repeated_guest                             -0.087239             -0.097245
previous_cancellations                        -0.012775             -0.013992
previous_bookings_not_canceled                -0.042715             -0.048743
booking_changes                                0.063281              0.096209
agent                                          0.140739              0.182382
company                                        0.066749              0.182211
days_in_waiting_list                          -0.054151             -0.002020
adr                                            0.049342              0.065237
required_car_parking_spaces                   -0.018554             -0.024859
total_of_special_requests                      0.072671              0.068192

                                  adults  children    babies  \
is_canceled                     0.060017  0.005048 -0.032491
lead_time                       0.119519 -0.037622 -0.020915
arrival_date_year               0.029635  0.054624 -0.013192
arrival_date_week_number        0.025909  0.005518  0.010395
arrival_date_day_of_month      -0.001566  0.014544 -0.000230
stays_in_weekend_nights         0.091871  0.045793  0.018483
stays_in_week_nights            0.092976  0.044203  0.020191
adults                          1.000000  0.030447  0.018146
children                        0.030447  1.000000  0.024030
babies                          0.018146  0.024030  1.000000
is_repeated_guest              -0.146426 -0.032859 -0.008943
previous_cancellations         -0.006738 -0.024730 -0.007501
previous_bookings_not_canceled -0.107983 -0.021072 -0.006550
booking_changes                -0.051673  0.048949  0.083440
agent                          -0.035594  0.041066  0.036184
company                         0.207793  0.030931  0.019206
```

```
days_in_waiting_list           -0.008283 -0.033273 -0.010621
adr                             0.230641  0.324854  0.029186
required_car_parking_spaces     0.014785  0.056253  0.037383
total_of_special_requests       0.122884  0.081745  0.097889

                                is_repeated_guest  previous_cancellations  \
is_canceled                         -0.084793               0.110133
lead_time                           -0.124410               0.086042
arrival_date_year                    0.010341              -0.119822
arrival_date_week_number            -0.030131               0.035501
arrival_date_day_of_month           -0.006145              -0.027011
stays_in_weekend_nights             -0.087239              -0.012775
stays_in_week_nights                -0.097245              -0.013992
adults                              -0.146426              -0.006738
children                            -0.032859              -0.024730
babies                              -0.008943              -0.007501
is_repeated_guest                    1.000000               0.082293
previous_cancellations               0.082293               1.000000
previous_bookings_not_canceled       0.418056               0.152728
booking_changes                      0.012092              -0.026993
agent                                0.031527              -0.012488
company                             -0.244586              -0.184574
days_in_waiting_list                -0.022235               0.005929
adr                                 -0.134314              -0.065646
required_car_parking_spaces          0.077090              -0.018492
total_of_special_requests            0.013050              -0.048384

                                previous_bookings_not_canceled  \
is_canceled                                        -0.057358
lead_time                                          -0.073548
arrival_date_year                                   0.029218
arrival_date_week_number                           -0.020904
arrival_date_day_of_month                          -0.000300
stays_in_weekend_nights                            -0.042715
stays_in_week_nights                               -0.048743
adults                                             -0.107983
children                                           -0.021072
babies                                             -0.006550
is_repeated_guest                                   0.418056
previous_cancellations                              0.152728
previous_bookings_not_canceled                      1.000000
booking_changes                                     0.011608
agent                                               0.023252
company                                            -0.208557
days_in_waiting_list                               -0.009397
adr                                                -0.072144
required_car_parking_spaces                         0.047653
```

```
total_of_special_requests                         0.037824

                                booking_changes     agent   company  \
is_canceled                          -0.144381 -0.083114 -0.020642
lead_time                             0.000149 -0.069741  0.151464
arrival_date_year                     0.030872  0.063457  0.259095
arrival_date_week_number              0.005508 -0.031201 -0.076760
arrival_date_day_of_month             0.010613  0.001487  0.044858
stays_in_weekend_nights               0.063281  0.140739  0.066749
stays_in_week_nights                  0.096209  0.182382  0.182211
adults                               -0.051673 -0.035594  0.207793
children                              0.048949  0.041066  0.030931
babies                                0.083440  0.036184  0.019206
is_repeated_guest                     0.012092  0.031527 -0.244586
previous_cancellations               -0.026993 -0.012488 -0.184574
previous_bookings_not_canceled        0.011608  0.023252 -0.208557
booking_changes                       1.000000  0.067010  0.122098
agent                                 0.067010  1.000000  0.350746
company                               0.122098  0.350746  1.000000
days_in_waiting_list                 -0.011634 -0.055151  0.000411
adr                                   0.019618 -0.024695  0.086376
required_car_parking_spaces           0.065620  0.177353 -0.012916
total_of_special_requests             0.052833  0.034162 -0.098558

                                days_in_waiting_list       adr  \
is_canceled                                0.054186  0.047557
lead_time                                  0.170084 -0.063077
arrival_date_year                         -0.056497  0.197580
arrival_date_week_number                   0.022933  0.075791
arrival_date_day_of_month                  0.022728  0.030245
stays_in_weekend_nights                   -0.054151  0.049342
stays_in_week_nights                      -0.002020  0.065237
adults                                    -0.008283  0.230641
children                                  -0.033273  0.324854
babies                                    -0.010621  0.029186
is_repeated_guest                         -0.022235 -0.134314
previous_cancellations                     0.005929 -0.065646
previous_bookings_not_canceled            -0.009397 -0.072144
booking_changes                           -0.011634  0.019618
agent                                     -0.055151 -0.024695
company                                    0.000411  0.086376
days_in_waiting_list                       1.000000 -0.040756
adr                                       -0.040756  1.000000
required_car_parking_spaces               -0.030600  0.056628
total_of_special_requests                 -0.082730  0.172185

                                required_car_parking_spaces  \
```

```
is_canceled                                    -0.195498
lead_time                                      -0.116451
arrival_date_year                              -0.013684
arrival_date_week_number                        0.001920
arrival_date_day_of_month                       0.008683
stays_in_weekend_nights                        -0.018554
stays_in_week_nights                           -0.024859
adults                                          0.014785
children                                        0.056253
babies                                          0.037383
is_repeated_guest                               0.077090
previous_cancellations                         -0.018492
previous_bookings_not_canceled                  0.047653
booking_changes                                 0.065620
agent                                           0.177353
company                                        -0.012916
days_in_waiting_list                           -0.030600
adr                                             0.056628
required_car_parking_spaces                     1.000000
total_of_special_requests                       0.082626

                                total_of_special_requests
is_canceled                                     -0.234658
lead_time                                       -0.095712
arrival_date_year                                0.108531
arrival_date_week_number                         0.026149
arrival_date_day_of_month                        0.003062
stays_in_weekend_nights                          0.072671
stays_in_week_nights                             0.068192
adults                                           0.122884
children                                         0.081745
babies                                           0.097889
is_repeated_guest                                0.013050
previous_cancellations                          -0.048384
previous_bookings_not_canceled                   0.037824
booking_changes                                  0.052833
agent                                            0.034162
company                                         -0.098558
days_in_waiting_list                            -0.082730
adr                                              0.172185
required_car_parking_spaces                      0.082626
total_of_special_requests                        1.000000
```

[9]:
```
df_numeric.corr()['is_canceled'].sort_values(ascending=False)
```

[9]:
```
is_canceled                     1.000000
lead_time                       0.293123
```

```
previous_cancellations            0.110133
adults                            0.060017
days_in_waiting_list              0.054186
adr                               0.047557
stays_in_week_nights              0.024765
arrival_date_year                 0.016660
arrival_date_week_number          0.008148
children                          0.005048
stays_in_weekend_nights          -0.001791
arrival_date_day_of_month        -0.006130
company                          -0.020642
babies                           -0.032491
previous_bookings_not_canceled   -0.057358
agent                            -0.083114
is_repeated_guest                -0.084793
booking_changes                  -0.144381
required_car_parking_spaces      -0.195498
total_of_special_requests        -0.234658
Name: is_canceled, dtype: float64
```

above we do corr() on df['is_canceled'] and we sort the values from high to low. Above data shows that if the lead_time is high, the cancelation is high also. but it is not reliable so much.

acending = False, makes the sort from high to low.

0.0.2 Now we will start to do test & train by XGBoost:

[10]:
```
x , y = df_numeric.iloc[:, 1:].values , df_numeric.iloc[:, 0].values
```

[11]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size= 0.33,␣
 ↪random_state= 123)
```

[13]:
```
import xgboost as xgb
xgb_clf = xgb.XGBClassifier(random_state= 123)
xgb_clf.get_params()
```

[13]:
```
{'objective': 'binary:logistic',
 'base_score': None,
 'booster': None,
 'callbacks': None,
 'colsample_bylevel': None,
 'colsample_bynode': None,
 'colsample_bytree': None,
 'device': None,
 'early_stopping_rounds': None,
```

```
 'enable_categorical': False,
 'eval_metric': None,
 'feature_types': None,
 'feature_weights': None,
 'gamma': None,
 'grow_policy': None,
 'importance_type': None,
 'interaction_constraints': None,
 'learning_rate': None,
 'max_bin': None,
 'max_cat_threshold': None,
 'max_cat_to_onehot': None,
 'max_delta_step': None,
 'max_depth': None,
 'max_leaves': None,
 'min_child_weight': None,
 'missing': nan,
 'monotone_constraints': None,
 'multi_strategy': None,
 'n_estimators': None,
 'n_jobs': None,
 'num_parallel_tree': None,
 'random_state': 123,
 'reg_alpha': None,
 'reg_lambda': None,
 'sampling_method': None,
 'scale_pos_weight': None,
 'subsample': None,
 'tree_method': None,
 'validate_parameters': None,
 'verbosity': None}
```

As above, n_estimator = None. So we will make it 10, then will fit the datas.

[14]:
```
xgb_clf.set_params(n_estimators = 10)
xgb_clf.fit(x_train, y_train)
p = xgb_clf.predict(x_test)
```

[15]:
```
from sklearn import metrics
print(metrics.classification_report(y_test, p))
```

```
              precision    recall  f1-score   support

           0       0.80      0.89      0.84     24838
           1       0.77      0.61      0.68     14561

    accuracy                           0.79     39399
   macro avg       0.78      0.75      0.76     39399
```

```
weighted avg       0.79      0.79      0.78      39399
```

[16]:
```
accuracy = float(np.sum(p == y_test)) / y_test.shape[0]
accuracy
```

[16]:
```
0.7885986953983604
```

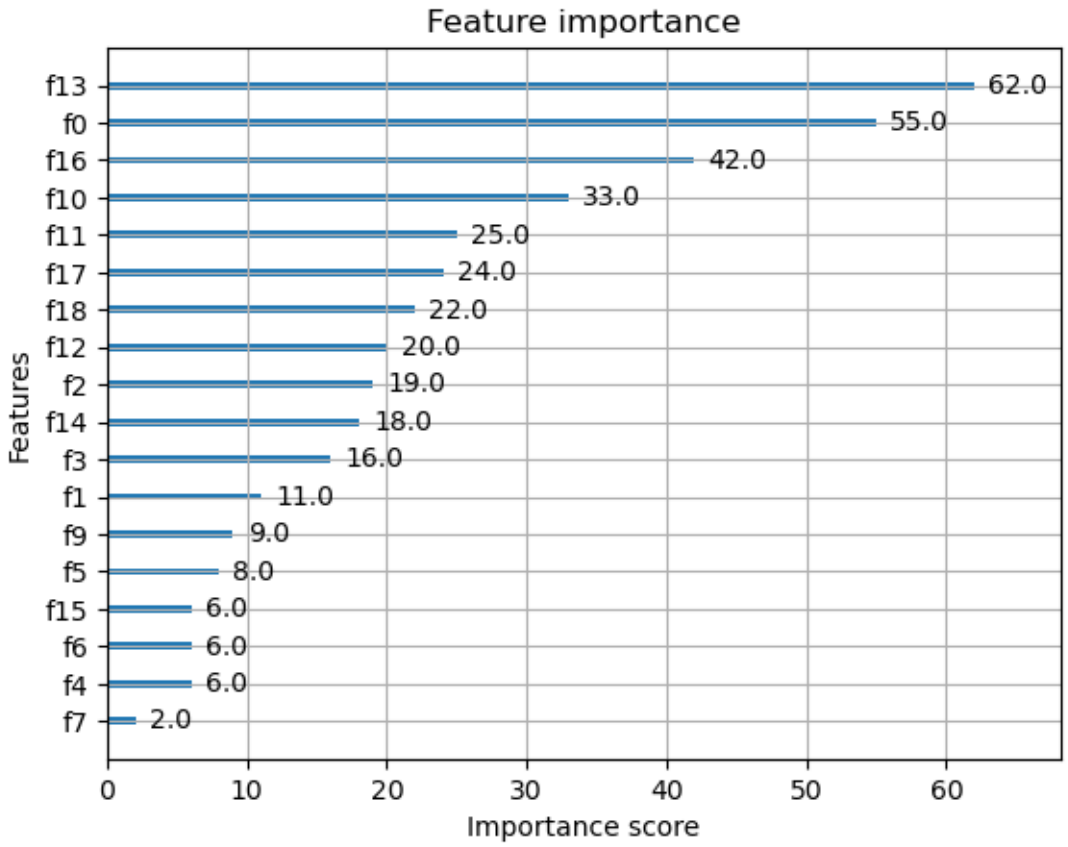
Above we do accuracy by metrics and with formula. Maybe we can increase our accuracy by increasing n_estimators.

[17]:
```
print(metrics.confusion_matrix(y_test, p))
```

```
[[22178  2660]
 [ 5669  8892]]
```

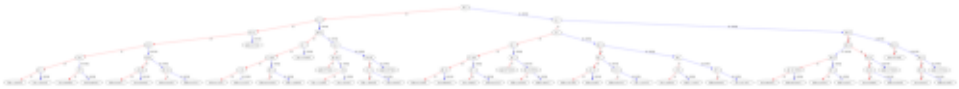
[18]:
```
xgb.plot_importance(xgb_clf)
```

[18]:
```
<Axes: title={'center': 'Feature importance'}, xlabel='Importance score',
ylabel='Features'>
```



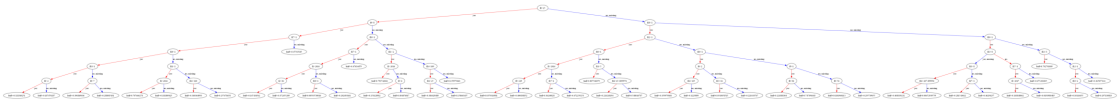
In above, we see the important features like agent, lead_time, adr.

[19]:
```
xgb.plot_tree(xgb_clf, tree_idx = 0);
```



[20]:
```
import matplotlib.pyplot as plt

fig, ax = plt.subplots(figsize=(30, 20))
xgb.plot_tree(xgb_clf, tree_idx=0, ax=ax)
plt.show()
```



Now we want to investigate the datas by DMatrix.

It creates a DMatrix object from your features x and target y. DMatrix is an optimized data structure used internally by XGBoost to train models more efficiently.

Why use DMatrix?

XGBoost's core training API (xgb.train()) requires input in the form of DMatrix, because it:

Optimizes memory usage

Speeds up training

Enables advanced features (like missing value handling and weight management)

[21]:
```
df_dmatrix = xgb.DMatrix(data = x, label = y)
```

We use DMatrix instead of test_train_split.

[22]:
```
params = {'objective':'binary:logistic', 'max_depth':3}
xgb_cv = xgb.cv(dtrain = df_dmatrix, params = params, nfold=3, num_boost_round =␣
↪10, seed = 123, metrics = ['error'])
xgb_cv
```

[22]:

	train-error-mean	train-error-std	test-error-mean	test-error-std
0	0.352207	0.022567	0.351838	0.018844

1	0.322745	0.001637	0.322833	0.001796
2	0.321677	0.001218	0.321744	0.002366
3	0.254519	0.001703	0.255315	0.004664
4	0.247253	0.006795	0.247081	0.009292
5	0.244212	0.000935	0.244384	0.004005
6	0.240594	0.002873	0.241184	0.003715
7	0.237997	0.001129	0.237976	0.002566
8	0.238554	0.001494	0.238504	0.002222
9	0.235861	0.000909	0.236117	0.002708

cross Validation is like below:

Above table shows the errors like (train-error-mean or std) and (test-error-mean or std). the opposite of these means are accuracy.

As the table the error comes down after each trains or tests. train-error-mean starts in 0.352207 and ends in 0.235861. and so on.

Now we want to see the best accuracy of test_mean:

[24]:
```
test_accuracy = 1 - xgb_cv['test-error-mean'].iloc[-1]
test_accuracy
```

[24]: 0.7638830513443524

Lets make the accuracy better:

[25]:
```
xgb_cv = xgb.cv(dtrain = df_dmatrix, params = params, nfold=3, num_boost_round =␣
↪40, early_stopping_rounds = 10, seed = 123, metrics = ['error'])
test_accuracy = 1 - xgb_cv['test-error-mean'].iloc[-1]
test_accuracy
```

[25]: 0.7929474592266847

Now we want to examine xgboost without using cv:

[27]:
```
from sklearn.metrics import accuracy_score
xgb_clf = xgb.XGBClassifier(n_estimators = 25, random_state = 123)
```

This function will measure how accurate your model's predictions are compared to the true labels.

It calculates:

Accuracy = Number of correct predictions / Total predictions

[28]:
```
xgb_clf.set_params(max_depth =10)
```

[28]:
```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              feature_weights=None, gamma=None, grow_policy=None,
              importance_type=None, interaction_constraints=None,
              learning_rate=None, max_bin=None, max_cat_threshold=None,
              max_cat_to_onehot=None, max_delta_step=None, max_depth=10,
              max_leaves=None, min_child_weight=None, missing=nan,
              monotone_constraints=None, multi_strategy=None, n_estimators=25,
              n_jobs=None, num_parallel_tree=None, ...)
```

Updates the classifier to allow each tree to have up to 10 levels deep.

Deeper trees capture more complex patterns but may risk overfitting.

[29]:
```
xgb_clf.fit(x_train, y_train)
p = xgb_clf.predict(x_test)
accuracy_score(y_test , p)
```

[29]:
```
0.8400720830477931
```

lets change one parameter:

[30]:
```
xgb_clf.set_params(colsample_bytree = 0.5)
xgb_clf.fit(x_train, y_train)
p = xgb_clf.predict(x_test)
accuracy_score(y_test , p)
```

[30]:
```
0.8410111931774918
```

colsample_bytree controls the fraction of features (columns) to be randomly sampled for each tree.

colsample_bytree=0.5: use 50% of the features randomly selected per tree.

Helps prevent overfitting, especially when you have many features and It adds diversity among trees (a form of regularization).

lets changes another params and see the results:

[31]:
```
xgb_clf.set_params(subsample = 0.75)
xgb_clf.fit(x_train, y_train)
p = xgb_clf.predict(x_test)
accuracy_score(y_test , p)
```

[31]:
```
0.8396913627249423
```

[32]:
```
xgb_clf.set_params(learning_rate = 0.3)
xgb_clf.fit(x_train, y_train)
p = xgb_clf.predict(x_test)
accuracy_score(y_test , p)
```

[32]:
```
0.8396913627249423
```

[33]:
```
xgb_clf.set_params(reg_alpha = 0.01)
xgb_clf.fit(x_train, y_train)
p = xgb_clf.predict(x_test)
accuracy_score(y_test , p)
```

[33]:
```
0.8393360237569482
```

[34]:
```
xgb_clf.get_params()
```

[34]:
```
{'objective': 'binary:logistic',
 'base_score': None,
 'booster': None,
 'callbacks': None,
 'colsample_bylevel': None,
 'colsample_bynode': None,
 'colsample_bytree': 0.5,
 'device': None,
 'early_stopping_rounds': None,
 'enable_categorical': False,
 'eval_metric': None,
 'feature_types': None,
 'feature_weights': None,
 'gamma': None,
 'grow_policy': None,
 'importance_type': None,
 'interaction_constraints': None,
 'learning_rate': 0.3,
 'max_bin': None,
 'max_cat_threshold': None,
 'max_cat_to_onehot': None,
 'max_delta_step': None,
 'max_depth': 10,
 'max_leaves': None,
 'min_child_weight': None,
 'missing': nan,
 'monotone_constraints': None,
 'multi_strategy': None,
 'n_estimators': 25,
 'n_jobs': None,
 'num_parallel_tree': None,
```

```
 'random_state': 123,
 'reg_alpha': 0.01,
 'reg_lambda': None,
 'sampling_method': None,
 'scale_pos_weight': None,
 'subsample': 0.75,
 'tree_method': None,
 'validate_parameters': None,
 'verbosity': None}
```

0.0.3 Now we want to use Hyperparameter Tuning for xgboost.

For better understanding, we will explain it in next session.

[35]:
```
from sklearn.model_selection import RandomizedSearchCV
```

[36]:
```
params = {
    'max_depth': list(range(3, 12)),
    'alpha' : [0, 0.001, 0.01, 0.1, 1],
    'subsample' : [0.5, 0.75, 1],
    'learning_rate' : np.linspace(0.01, 0.5, 10),
    'n_estimators': [10, 25, 40]
}
xgb_clf = xgb.XGBClassifier(random_state = 123)
xgb_rs = RandomizedSearchCV(xgb_clf, param_distributions = params, cv=3, n_iter␣
 ↪=5, verbose=2, random_state=123)
xgb_rs.fit(x_train, y_train)
```

```
Fitting 3 folds for each of 5 candidates, totalling 15 fits
[CV] END alpha=1, learning_rate=0.22777777777777777, max_depth=5,
n_estimators=10, subsample=0.5; total time=   0.0s
[CV] END alpha=1, learning_rate=0.22777777777777777, max_depth=5,
n_estimators=10, subsample=0.5; total time=   0.0s
[CV] END alpha=1, learning_rate=0.22777777777777777, max_depth=5,
n_estimators=10, subsample=0.5; total time=   0.0s
[CV] END alpha=1, learning_rate=0.11888888888888888, max_depth=6,
n_estimators=40, subsample=1; total time=   0.0s
[CV] END alpha=1, learning_rate=0.11888888888888888, max_depth=6,
n_estimators=40, subsample=1; total time=   0.0s
[CV] END alpha=1, learning_rate=0.11888888888888888, max_depth=6,
n_estimators=40, subsample=1; total time=   0.0s
[CV] END alpha=1, learning_rate=0.11888888888888888, max_depth=8,
n_estimators=40, subsample=0.75; total time=   0.1s
[CV] END alpha=1, learning_rate=0.11888888888888888, max_depth=8,
n_estimators=40, subsample=0.75; total time=   0.1s
[CV] END alpha=1, learning_rate=0.11888888888888888, max_depth=8,
n_estimators=40, subsample=0.75; total time=   0.1s
[CV] END alpha=0.001, learning_rate=0.33666666666666667, max_depth=8,
n_estimators=25, subsample=1; total time=   0.0s
```

```
[CV] END alpha=0.001, learning_rate=0.33666666666666667, max_depth=8,
n_estimators=25, subsample=1; total time=   0.0s
[CV] END alpha=0.001, learning_rate=0.33666666666666667, max_depth=8,
n_estimators=25, subsample=1; total time=   0.0s
[CV] END alpha=0.001, learning_rate=0.17333333333333334, max_depth=10,
n_estimators=40, subsample=0.5; total time=   0.1s
[CV] END alpha=0.001, learning_rate=0.17333333333333334, max_depth=10,
n_estimators=40, subsample=0.5; total time=   0.1s
[CV] END alpha=0.001, learning_rate=0.17333333333333334, max_depth=10,
n_estimators=40, subsample=0.5; total time=   0.1s
```

[36]:
```
RandomizedSearchCV(cv=3,
                   estimator=XGBClassifier(base_score=None, booster=None,
                                           callbacks=None,
                                           colsample_bylevel=None,
                                           colsample_bynode=None,
                                           colsample_bytree=None, device=None,
                                           early_stopping_rounds=None,
                                           enable_categorical=False,
                                           eval_metric=None, feature_types=None,
                                           feature_weights=None, gamma=None,
                                           grow_policy=None,
                                           importance_type=None,
                                           interaction_constrain...
                                           n_estimators=None, n_jobs=None,
                                           num_parallel_tree=None, ...),
                   n_iter=5,
                   param_distributions={'alpha': [0, 0.001, 0.01, 0.1, 1],
                                        'learning_rate': array([0.01      ,
 0.06444444, 0.11888889, 0.17333333, 0.22777778,
        0.28222222, 0.33666667, 0.39111111, 0.44555556, 0.5       ]),
                                        'max_depth': [3, 4, 5, 6, 7, 8, 9, 10,
                                                      11],
                                        'n_estimators': [10, 25, 40],
                                        'subsample': [0.5, 0.75, 1]},
                   random_state=123, verbose=2)
```

[37]:
```
print(xgb_rs.best_params_)
print(xgb_rs.best_score_)
```

```
{'subsample': 0.5, 'n_estimators': 40, 'max_depth': 10, 'learning_rate':
0.17333333333333334, 'alpha': 0.001}
0.834518897544782
```