In [1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import datasist as ds
```

In [2]:
```python
train_df=pd.read_csv(r"C:\Users\ATN-PC\Desktop\kc zeph\it_spend_files\Train.csv")
```

In [3]:
```python
test_df = pd.read_csv(r"C:\Users\ATN-PC\Desktop\kc zeph\it_spend_files\Test.csv")
```

In [4]:
```python
#ds.structdata.describe(train_df)
```

checking columns to drop

In [5]:
```python
col_drop = []
for col in train_df.columns:
    if col not in test_df.columns:
        col_drop.append(col)
        print(col)
print(col_drop)
```

```
total_cost
['total_cost']
```

In [6]:
```python
train_df.columns
```

Out[6]:
```
Index(['ID', 'country', 'age_group', 'travel_with', 'total_female',
       'total_male', 'purpose', 'main_activity', 'info_source',
       'tour_arrangement', 'package_transport_int', 'package_accomodation',
       'package_food', 'package_transport_tz', 'package_sightseeing',
       'package_guided_tour', 'package_insurance', 'night_mainland',
       'night_zanzibar', 'payment_mode', 'first_trip_tz', 'most_impressing',
       'total_cost'],
      dtype='object')
```

In [7]:
```python
train_df.drop(columns="ID",inplace=True)
test_df.drop(columns="ID",inplace=True)
```

In [8]:
```python
train_df.shape,test_df.shape
```

Out[8]:
```
((4809, 22), (1601, 21))
```

In [9]:
```python
train_df.isnull().sum()
```

Out[9]:
```
country                     0
age_group                   0
travel_with              1114
total_female                3
total_male                  5
purpose                     0
main_activity               0
info_source                 0
tour_arrangement            0
package_transport_int       0
package_accomodation        0
package_food                0
package_transport_tz        0
package_sightseeing         0
package_guided_tour         0
package_insurance           0
night_mainland              0
night_zanzibar              0
payment_mode                0
first_trip_tz               0
most_impressing           313
total_cost                  0
dtype: int64
```

filling missing values

In [10]:
```python
#train_df =ds.feature_engineering.fill_missing_num(train_df,method ="mean")
```

In [11]:
```python
train_df["most_impressing"].fillna(method ="bfill",inplace =True)
train_df["total_female"].fillna(train_df["total_female"].mode()[0],inplace =True)
train_df["total_male"].fillna(train_df["total_male"].mode()[0],inplace =True)
train_df["travel_with"].fillna(method ="bfill",inplace =True)

test_df["most_impressing"].fillna(method ="bfill",inplace =True)
test_df["total_female"].fillna(test_df["total_female"].mode()[0],inplace =True)
test_df["total_male"].fillna(test_df["total_male"].mode()[0],inplace =True)
test_df["travel_with"].fillna(method ="bfill",inplace =True)
```

In [12]:
```python
#train_df.info()
```

processing of categorical data

In [13]:
```python
# join train_df and test_df together so as to perform the OHE or getdummies or labe encoder once
all_data,ntrain,ntest = ds.structdata.join_train_and_test(train_df,test_df)
```

In [14]:
```python
ds.structdata.get_cat_feats(train_df)
```

Out[14]:
```
['country',
 'age_group',
 'travel_with',
 'purpose',
 'main_activity',
 'info_source',
 'tour_arrangement',
 'package_transport_int',
 'package_accomodation',
 'package_food',
 'package_transport_tz',
 'package_sightseeing',
 'package_guided_tour',
 'package_insurance',
 'payment_mode',
 'first_trip_tz',
 'most_impressing']
```

those with two options(like yes and No) as obtained from class count

'tour_arrangement','package_transport_int', 'package_accomodation', 'package_food', 'package_transport_tz', 'package_sightseeing', 'package_guided_tour' 'package_insurance',

'first_trip_tz'

In [15]:
```python
#ds.structdata.class_count(train_df)
```

In [16]:
```python
#label encode those categories that are too large and use get dummies for onces that are like two features
from sklearn.preprocessing import LabelEncoder
lb =LabelEncoder()
cols_2_encode = ['country',
 'age_group',
 'travel_with',
 'purpose',
 'main_activity',
 'info_source',

 'payment_mode',

 'most_impressing']

for col in cols_2_encode:
    lb.fit(all_data[col])
    all_data[col] = lb.transform(all_data[col])
```

In [17]:
```python
all_data.head().T
```

Out[17]:

	0	1	2	3	4
country	101	111	111	111	17
age_group	3	2	2	2	0
travel_with	2	0	0	3	0
total_female	1.0	1.0	0.0	1.0	1.0
total_male	1.0	0.0	1.0	1.0	0.0
purpose	1	1	5	1	1
main_activity	7	3	3	7	7
info_source	0	7	0	5	5
tour_arrangement	Independent	Independent	Independent	Package Tour	Independent
package_transport_int	No	No	No	No	No
package_accomodation	No	No	No	Yes	No
package_food	No	No	No	Yes	No
package_transport_tz	No	No	No	Yes	No
package_sightseeing	No	No	No	Yes	No
package_guided_tour	No	No	No	Yes	No
package_insurance	No	No	No	No	No
night_mainland	13.0	14.0	1.0	11.0	7.0
night_zanzibar	0.0	7.0	31.0	0.0	4.0
payment_mode	0	0	0	0	0
first_trip_tz	No	Yes	No	Yes	Yes
most_impressing	2	6	1	2	4
total_cost	674602.5	3214906.5	3315000.0	7790250.0	1657500.0

In [18]:
```python
# using getdummies from OHE to perform other at once
all_data = pd.get_dummies(all_data)
```

In [19]:
```python
all_data.head().T
```

Out[19]:

	0	1	2	3	4
country	101.0	111.0	111.0	111.0	17.0
age_group	3.0	2.0	2.0	2.0	0.0
travel_with	2.0	0.0	0.0	3.0	0.0
total_female	1.0	1.0	0.0	1.0	1.0
total_male	1.0	0.0	1.0	1.0	0.0
purpose	1.0	1.0	5.0	1.0	1.0
main_activity	7.0	3.0	3.0	7.0	7.0
info_source	0.0	7.0	0.0	5.0	5.0
night_mainland	13.0	14.0	1.0	11.0	7.0
night_zanzibar	0.0	7.0	31.0	0.0	4.0
payment_mode	0.0	0.0	0.0	0.0	0.0
most_impressing	2.0	6.0	1.0	2.0	4.0
total_cost	674602.5	3214906.5	3315000.0	7790250.0	1657500.0
tour_arrangement_Independent	1.0	1.0	1.0	0.0	1.0
tour_arrangement_Package Tour	0.0	0.0	0.0	1.0	0.0
package_transport_int_No	1.0	1.0	1.0	1.0	1.0
package_transport_int_Yes	0.0	0.0	0.0	0.0	0.0
package_accomodation_No	1.0	1.0	1.0	0.0	1.0
package_accomodation_Yes	0.0	0.0	0.0	1.0	0.0
package_food_No	1.0	1.0	1.0	0.0	1.0
package_food_Yes	0.0	0.0	0.0	1.0	0.0
package_transport_tz_No	1.0	1.0	1.0	0.0	1.0
package_transport_tz_Yes	0.0	0.0	0.0	1.0	0.0
package_sightseeing_No	1.0	1.0	1.0	0.0	1.0
package_sightseeing_Yes	0.0	0.0	0.0	1.0	0.0
package_guided_tour_No	1.0	1.0	1.0	0.0	1.0
package_guided_tour_Yes	0.0	0.0	0.0	1.0	0.0
package_insurance_No	1.0	1.0	1.0	1.0	1.0
package_insurance_Yes	0.0	0.0	0.0	0.0	0.0
first_trip_tz_No	1.0	0.0	1.0	0.0	0.0
first_trip_tz_Yes	0.0	1.0	0.0	1.0	1.0

In [20]:
```python
# checking all_dataset to know if ready for ML
#all_data.info()
```

In [21]:
```python
train =all_data.iloc[0:ntrain]
test =all_data.iloc[ntrain:]
```

In [22]:
```python
#checking shape accuracy with respect to original shape
train.shape,test.shape
```

Out[22]:
```
((4809, 31), (1601, 31))
```

In [23]:
```python
train.to_csv("trainuniag2.csv",index=False)
test.to_csv("testunilag2.csv",index=False)
```

In [24]:
```python
y= train["total_cost"]
testz= test.drop(columns="total_cost",inplace =True)
train.drop(columns="total_cost",inplace =True)
```

```
C:\Users\ATN-PC\AppData\Local\Temp\ipykernel_8252\2855698001.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  testz= test.drop(columns="total_cost",inplace =True)
C:\Users\ATN-PC\AppData\Local\Temp\ipykernel_8252\2855698001.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  train.drop(columns="total_cost",inplace =True)
```

In [25]:
```python
#test.info()
```

In [26]:
```python
#train.info()
```

In [27]:
```python
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test= train_test_split(train,y,test_size =0.1,random_state =42,shuffle =True)
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
sc.fit(X_train)
X_train_std = sc.transform(X_train)
X_test_std = sc.transform(X_test)
```

In [28]:
```python
from sklearn.linear_model import LinearRegression

model = LinearRegression()



gbm = model.fit(X_train, y_train)

from sklearn.metrics import mean_squared_error
```

In [ ]:

In [ ]:

In [29]:
```python
from sklearn.metrics import mean_absolute_error

# After fitting linear regression model:
y_pred = model.predict(X_test)  # Predicted values on test data
mae = mean_absolute_error(y_test, y_pred)  # Calculate MAE

print("Mean Absolute Error:", mae)
```

```
Mean Absolute Error: 5698801.961778337
```

In [ ]:

In [30]:
```python
from sklearn.tree import DecisionTreeRegressor,ExtraTreeRegressor
from sklearn.ensemble import RandomForestRegressor,ExtraTreesRegressor,GradientBoostingRegressor
#metrics
from sklearn.metrics import mean_absolute_error
def mae(y_test,y_pred):
    mae = mean_absolute_error(y_test, y_pred)
    return mae
```

In [31]:
```python
dt = DecisionTreeRegressor(max_depth =1,random_state =1)
dt.fit(X_train_std,y_train)

pred =dt.predict(X_test)
print(mae(y_test,pred))
```

```
6261231.509607027
C:\Users\ATN-PC\anaconda3\Lib\site-packages\sklearn\base.py:457: UserWarning: X has feature names, but DecisionTreeRegressor was fitted without feature names
  warnings.warn(
```

In [32]:
```python
test.columns
```

Out[32]:
```
Index(['country', 'age_group', 'travel_with', 'total_female', 'total_male',
       'purpose', 'main_activity', 'info_source', 'night_mainland',
       'night_zanzibar', 'payment_mode', 'most_impressing',
       'tour_arrangement_Independent', 'tour_arrangement_Package Tour',
       'package_transport_int_No', 'package_transport_int_Yes',
       'package_accomodation_No', 'package_accomodation_Yes',
       'package_food_No', 'package_food_Yes', 'package_transport_tz_No',
       'package_transport_tz_Yes', 'package_sightseeing_No',
       'package_sightseeing_Yes', 'package_guided_tour_No',
       'package_guided_tour_Yes', 'package_insurance_No',
       'package_insurance_Yes', 'first_trip_tz_No', 'first_trip_tz_Yes'],
      dtype='object')
```

After testing the model performance with x_train and y_train and checking it's MAE with y_test and y_pred(mae(y_test,pred)). Then i now train the catboost_train_data with whole training and y data ,Pool(train,y)

In [33]:
```python
from catboost import CatBoostRegressor,Pool
catboost_train_data = Pool(train,y)
```

In [34]:
```python
cb_model =CatBoostRegressor(
    loss_function="MAE",
    eval_metric="MAE",
    iterations=1000,
    learning_rate=0.07,
    random_seed=42,
    depth=4,
    bagging_temperature=0.7,

    early_stopping_rounds=100,
    verbose=False,  # Disable training verbosity for a cleaner output
    reg_lambda=0.01
)
cb_model.fit(catboost_train_data,plot=True)
```

```
MetricVisualizer(layout=Layout(align_self='stretch', height='500px'))
```

Out[34]:
```
<catboost.core.CatBoostRegressor at 0x114138f1f10>
```

In [35]:
```python
## pred =cb_model.predict(X_test)
#print(mae(y_test,pred))
```

In [36]:
```python
test_id = pd.read_csv(r"C:\Users\ATN-PC\Desktop\kc zeph\it_spend_files\Test.csv")
```

In [37]:
```python
test_id.ID
```

Out[37]:
```
0          tour_1
1        tour_100
2       tour_1001
3       tour_1006
4       tour_1009
          ...
1596     tour_988
1597     tour_990
1598     tour_992
1599     tour_996
1600     tour_998
Name: ID, Length: 1601, dtype: object
```

In [38]:
```python
y_pred = cb_model.predict(X_test)
print(mae(y_test,y_pred))
```

```
4097105.893549782
```

In [39]:
```python
predcf =cb_model.predict(test)
```

This result was uploaded to Zindi competition website on tanzania tourism prediction and my best result was obatained using CatBoostRegressor and i obtained 11th position in the competition

In [40]:
```python
output =pd.DataFrame({"test_id":test_id.ID ,"total_cost": predcf})
output.to_csv("Zincatfff.csv",index =False)
output
```

Out[40]:

	test_id	total_cost
0	tour_1	2.134284e+07
1	tour_100	7.098960e+06
2	tour_1001	9.302364e+06
3	tour_1006	2.678122e+06
4	tour_1009	1.726829e+07
...	...	...
1596	tour_988	3.668368e+05
1597	tour_990	1.922577e+07
1598	tour_992	1.036344e+06
1599	tour_996	3.175156e+05
1600	tour_998	4.454162e+06

1601 rows × 2 columns

In [ ]:

In [41]:
```python
import lightgbm as lgb

lgb_train = lgb.Dataset(train, label=y)
```

In [42]:
```python
modell = lgb.LGBMRegressor(objective="regression",iterations=1000,
    learning_rate=0.05,
    random_seed=42,
                        metric="mae")  # optional early stopping

modell.fit(train,y)
```

```
[LightGBM] [Warning] Unknown parameter: iterations
[LightGBM] [Warning] Found whitespace in feature_names, replace with underlines
[LightGBM] [Warning] Unknown parameter: iterations
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.001788 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 258
[LightGBM] [Info] Number of data points in the train set: 4809, number of used features: 30
[LightGBM] [Info] Start training from score 8114388.775496
```

Out[42]:
```
LGBMRegressor
LGBMRegressor(iterations=1000, learning_rate=0.05, metric='mae',
              objective='regression', random_seed=42)
```

In [43]:
```python
y_pred = modell.predict(X_test)
print(mae(y_test,y_pred))
```

```
[LightGBM] [Warning] Unknown parameter: iterations
4189565.1779530183
```

In [44]:
```python
predl =modell.predict(test)
```

```
[LightGBM] [Warning] Unknown parameter: iterations
```

This result was uploaded to Zindi competition website on tanzania tourism prediction

In [45]:
```python
output =pd.DataFrame({"test_id":test_id.ID ,"total_cost": predl})
output.to_csv("Zinlgbm3.csv",index =False)
output
```

Out[45]:

	test_id	total_cost
0	tour_1	2.318674e+07
1	tour_100	1.011334e+07
2	tour_1001	1.559144e+07
3	tour_1006	4.958598e+06
4	tour_1009	2.566547e+07
...	...	...
1596	tour_988	7.371609e+05
1597	tour_990	2.457233e+07
1598	tour_992	1.355749e+06
1599	tour_996	9.764005e+05
1600	tour_998	6.473842e+06

1601 rows × 2 columns

In [ ]:

In [ ]:

In [ ]: