

ANALYTICS FOR HOSPITAL AND HEALTHCARE DATA

TEAM ID : PNT2022TMID16345

```
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) O

In [65]: 1 ada_model_DT_tuned = AdaBoostClassifier(base_estimator=dt_tuned, n_estimators = 1, random_state = 10)
          2
          3 ada_model_DT_tuned.fit(X_train, y_train)

Out[65]: AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=11,
                                                                random_state=10),
                           n_estimators=1, random_state=10)

In [66]: 1 y_pred_ada_model_DT_tuned = ada_model_DT_tuned.predict(X_test)
          2 accuracy_score(y_test,y_pred_ada_model_DT_tuned)*100

Out[66]: 29.91990482058255

In [67]: 1 print(classification_report(y_test,y_pred_ada_model_DT_tuned))

              precision    recall  f1-score   support

     1       0.19       0.16       0.17       6901
     2       0.33       0.34       0.33       23205
     3       0.38       0.39       0.39       25792
     4       0.22       0.27       0.24       16289
     5       0.06       0.04       0.05        3439
     6       0.31       0.36       0.33       10470
     7       0.00       0.00       0.00         822
     8       0.14       0.10       0.11       3093
     9       0.22       0.10       0.13       1412
    10       0.10       0.01       0.02         782
    11       0.45       0.20       0.27       1933

 accuracy          0.22          0.30       0.4138
 macro avg         0.22          0.18       0.1938
 weighted avg      0.29          0.29       0.4138

In [68]: 1 rf_classification_tuned = RandomForestClassifier(criterion = 'gini', n_estimators = 47, random_state = 10)
```

```
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) O

In [29]: 1 admission_encode = {'Trauma': 1, 'Urgent': 2, 'Emergency': 3 }
          2 severity_encode = {'Minor': 1, 'Moderate': 2, 'Extreme': 3 }
          3
          4
          5 df_cat_train['Type of Admission'] = df_cat_train['Type of Admission'].map(admission_encode)
          6 df_cat_train['Severity of Illness'] = df_cat_train['Severity of Illness'].map(severity_encode)
          7
          8 df_cat_test['Type of Admission'] = df_cat_test['Type of Admission'].map(admission_encode)
          9 df_cat_test['Severity of Illness'] = df_cat_test['Severity of Illness'].map(severity_encode)

In [30]: 1 df_cat_train['Stay'] = df_cat_train['Stay'].replace({'0-10':1, '11-20':2, '21-30':3, '31-40':4, '41-50':5, '51-60':6, '61-70':
          2 '71-80':8, '81-90':9, '91-100':10, '100+':11})
          3
          4 df_cat_train['Age'] = df_cat_train['Age'].replace({'0-10':1, '11-20':2, '21-30':3, '31-40':4, '41-50':5, '51-60':6, '61-70':7,
          5 '71-80':8, '81-90':9, '91-100':10})
          6
          7 df_cat_test['Age'] = df_cat_test['Age'].replace({'0-10':1, '11-20':2, '21-30':3, '31-40':4, '41-50':5, '51-60':6, '61-70':7,
          8 '71-80':8, '81-90':9, '91-100':10})
          9

In [31]: 1 df_cat_train['Stay'] = df_cat_train['Stay'].astype(int)

In [32]: 1 from sklearn.preprocessing import LabelEncoder
          2 LE = LabelEncoder()
          3
          4 df_cat_train['Hospital_code'] = LE.fit_transform(df_cat_train['Hospital_code'])
          5 df_cat_train['Hospital_type_code'] = LE.fit_transform(df_cat_train['Hospital_type_code'])
          6 df_cat_train['City_Code_Hospital'] = LE.fit_transform(df_cat_train['City_Code_Hospital'])
          7 df_cat_train['Hospital_region_code'] = LE.fit_transform(df_cat_train['Hospital_region_code'])
          8 df_cat_train['Department'] = LE.fit_transform(df_cat_train['Department'])
          9 df_cat_train['Ward_Type'] = LE.fit_transform(df_cat_train['Ward_Type'])
          10 df_cat_train['Ward_Facility_Code'] = LE.fit_transform(df_cat_train['Ward_Facility_Code'])
          11 df_cat_train['City_Code_Patient'] = LE.fit_transform(df_cat_train['City_Code_Patient'])
          12 df_cat_train['Bed_Grade'] = LE.fit_transform(df_cat_train['Bed_Grade'])
          13
```

```
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) O
3 ada_model_rf_tuned.fit(X_train, y_train)

Out[72]: AdaBoostClassifier(base_estimator=RandomForestClassifier(n_estimators=47,
                                                                random_state=10),
                          n_estimators=4, random_state=10)

In [74]: 1 from sklearn.ensemble import GradientBoostingClassifier
2 GB=GradientBoostingClassifier(random_state=10)
3 GB.fit(X_train, y_train)

Out[74]: GradientBoostingClassifier(random_state=10)

In [75]: 1 y_pred_GB = GB.predict(X_test)
2 accuracy_score(y_test,y_pred_GB)*100

Out[75]: 41.5475153498856

In [76]: 1 print(classification_report(y_test,y_pred_GB))

              precision    recall  f1-score   support

     1       0.41       0.12       0.19       6901
     2       0.42       0.51       0.46       23205
     3       0.42       0.66       0.52       25792
     4       0.41       0.17       0.24       16289
     5       0.14       0.00       0.00        3439
     6       0.39       0.53       0.45       10470
     7       0.00       0.00       0.00         822
     8       0.30       0.01       0.02       3093
     9       0.31       0.20       0.24       1412
    10       0.17       0.01       0.01         782
    11       0.52       0.40       0.45       1933

 accuracy          0.42       0.42       0.42       94138
 macro avg         0.32       0.24       0.23       94138
 weighted avg      0.39       0.42       0.37       94138
```

```
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel) O

In [77]: 1 GB_tuned=GradientBoostingClassifier(n_estimators=29, random_state=10)
2 GB_tuned.fit(X_train, y_train)

Out[77]: GradientBoostingClassifier(n_estimators=29, random_state=10)

In [80]: 1 from sklearn.naive_bayes import GaussianNB
2 NB = GaussianNB()
3 NB.fit(X_train,y_train)

Out[80]: GaussianNB()

In [81]: 1 y_pred_NB = NB.predict(X_test)
2 accuracy_score(y_test,y_pred_NB)*100

Out[81]: 36.37850814761308

In [82]: 1 print(classification_report(y_test,y_pred_NB))

              precision    recall  f1-score   support

     1       0.30       0.09       0.14       6901
     2       0.36       0.41       0.39       23205
     3       0.39       0.65       0.49       25792
     4       0.32       0.15       0.21       16289
     5       0.08       0.01       0.01        3439
     6       0.33       0.38       0.36       10470
     7       0.04       0.00       0.00         822
     8       0.10       0.01       0.02       3093
     9       0.12       0.02       0.04       1412
    10       0.50       0.00       0.00         782
    11       0.42       0.37       0.39       1933

 accuracy          0.36       0.36       0.36       94138
 macro avg         0.27       0.19       0.18       94138
 weighted avg      0.33       0.36       0.32       94138
```

```
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (pykernel) O

In [83]: 1 from sklearn.neighbors import KNeighborsClassifier
2
3 knn=KNeighborsClassifier(n_neighbors=565,weights='distance')
4 knn.fit(X_train,y_train)

Out[83]: KNeighborsClassifier(n_neighbors=565, weights='distance')

In [84]: 1 y_pred_KNN = NB.predict(X_test)
2 accuracy_score(y_test,y_pred_KNN)*100

Out[84]: 36.37850814761308

In [85]: 1 print(classification_report(y_test,y_pred_NB))

              precision    recall  f1-score   support

     1       0.30         0.09         0.14         6901
     2       0.36         0.41         0.39         23205
     3       0.39         0.65         0.49         25792
     4       0.32         0.15         0.21         16289
     5       0.08         0.01         0.01          3439
     6       0.33         0.38         0.36         10470
     7       0.04         0.00         0.00           822
     8       0.10         0.01         0.02          3093
     9       0.12         0.02         0.04          1412
    10       0.50         0.00         0.00           782
    11       0.42         0.37         0.39          1933

 accuracy          0.36         0.36         0.36         94138
 macro avg         0.27         0.19         0.18         94138
 weighted avg         0.33         0.36         0.32         94138

In [86]: 1 from catboost import CatBoostClassifier
2
3 cb = CatBoostClassifier(random_state=10,use_best_model=True,iterations=1000)
```

```
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (pykernel) O

In [86]: 1 from catboost import CatBoostClassifier
2
3 cb = CatBoostClassifier(random_state=10,use_best_model=True,iterations=1000)
4 cb.fit(X_train,y_train,use_best_model=True,verbose=100,eval_set=(X_test,y_test))

Learning rate set to 0.120271
0:   learn: 2.1972797      test: 2.1978440 best: 2.1978440 (0)      total: 532ms remaining: 8m 50s
100: learn: 1.5115382     test: 1.5242638 best: 1.5242638 (100)    total: 37.5s remaining: 5m 34s
200: learn: 1.4818453     test: 1.5070444 best: 1.5070444 (200)    total: 1m 17s remaining: 5m 9s
300: learn: 1.4638516     test: 1.5016071 best: 1.5016071 (300)    total: 1m 57s remaining: 4m 32s
400: learn: 1.4484799     test: 1.4983697 best: 1.4983697 (400)    total: 2m 35s remaining: 3m 52s
500: learn: 1.4345747     test: 1.4970449 best: 1.4970449 (500)    total: 3m 14s remaining: 3m 13s
600: learn: 1.4237650     test: 1.4965192 best: 1.4964737 (579)    total: 3m 56s remaining: 2m 37s
700: learn: 1.4123374     test: 1.4963260 best: 1.4961744 (652)    total: 4m 34s remaining: 1m 57s
800: learn: 1.4018164     test: 1.4964416 best: 1.4961744 (652)    total: 5m 13s remaining: 1m 17s
900: learn: 1.3915056     test: 1.4967396 best: 1.4961744 (652)    total: 5m 52s remaining: 38.8s
999: learn: 1.3815565     test: 1.4971529 best: 1.4961744 (652)    total: 6m 37s remaining: 0us

bestTest = 1.496174357
bestIteration = 652

Shrink model to first 653 iterations.

Out[86]: <catboost.core.CatBoostClassifier at 0x1f1be732bb0>

In [87]: 1 cb_pred = cb.predict(X_test)
2 accuracy_score(y_test,cb_pred)*100

Out[87]: 42.54180033567741

In [88]: 1 print(classification_report(y_test,cb_pred))

              precision    recall  f1-score   support

     1       0.41         0.16         0.23         6901
     2       0.43         0.51         0.47         23205
     3       0.43         0.66         0.52         25792
```

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (pykernel)

2	0.43	0.51	0.47	23205
3	0.43	0.66	0.52	25792
4	0.41	0.24	0.30	16289
5	0.24	0.00	0.01	3439
6	0.41	0.48	0.44	10470
7	0.12	0.00	0.00	922
8	0.42	0.03	0.05	3093
9	0.36	0.21	0.27	1412
10	0.29	0.01	0.02	782
11	0.52	0.43	0.47	1933
accuracy			0.43	94138
macro avg	0.37	0.25	0.25	94138
weighted avg	0.41	0.43	0.39	94138

```
In [89]: 1 cb_pred_train= cb.predict(X_train)
2 accuracy_score(y_train,cb_pred_train)*100
```

```
Out[89]: 45.57692745441715
```

```
In [90]: 1 print(classification_report(y_train,cb_pred_train))
```

	precision	recall	f1-score	support
1	0.46	0.19	0.27	16349
2	0.45	0.54	0.49	53890
3	0.45	0.68	0.54	60524
4	0.47	0.26	0.34	38023
5	0.63	0.01	0.02	8102
6	0.45	0.53	0.48	23993
7	0.79	0.01	0.02	1876
8	0.71	0.06	0.10	7003
9	0.55	0.31	0.40	3349
10	0.84	0.07	0.12	1931
11	0.62	0.51	0.56	4615

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (pykernel)

accuracy			0.46	219655
macro avg	0.58	0.29	0.30	219655
weighted avg	0.48	0.46	0.42	219655

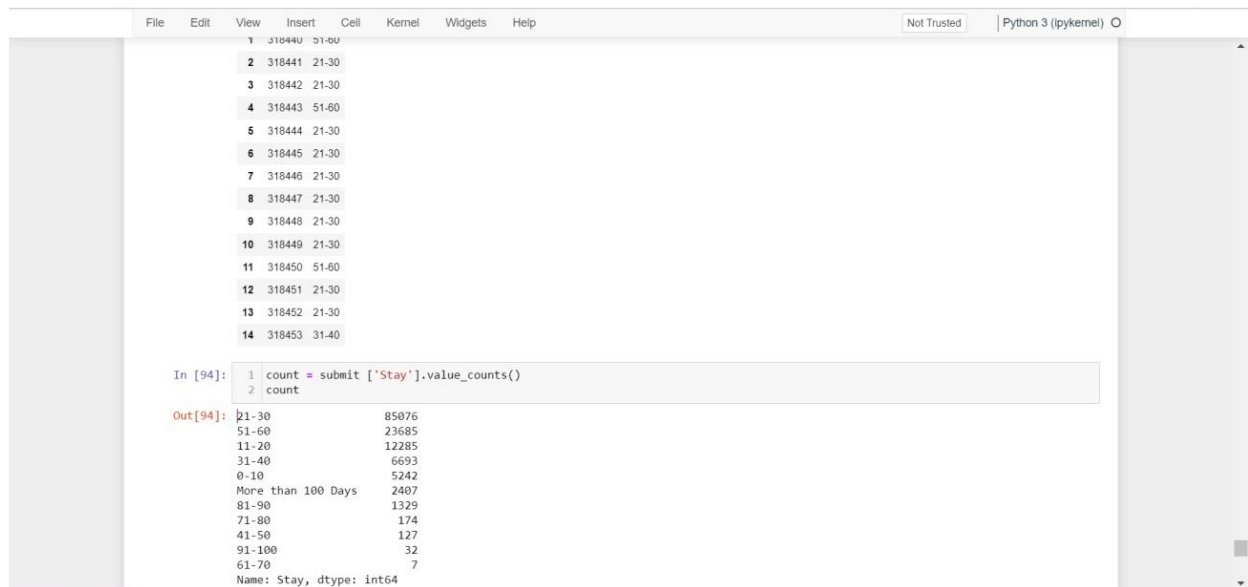
```
In [91]: 1 ls = df_full_test.columns.tolist()
2
3 in_data = df_full_test[ls]
4
5 out_data = cb.predict(in_data)
```

```
In [92]: 1 test = pd.read_csv("D:/HealthCare/test_data.csv")
2 submit = pd.DataFrame()
```

```
In [93]: 1 submit['case_id'] = test['case_id']
2 submit['Stay'] = out_data
3
4 stay_decode = { 1 : '0-10', 2 : '11-20', 3 : '21-30', 4 : '31-40', 5 : '41-50', 6 : '51-60', 7 : '61-70', 8 : '71-8
5               10 : '91-100', 11 : 'More than 100 Days' }
6
7 submit['Stay'] = submit['Stay'].map(stay_decode)
8
9 submit.head(15)
```

```
Out[93]:
```

	case_id	Stay
0	318439	0-10
1	318440	51-60
2	318441	21-30
3	318442	21-30
4	318443	51-60
5	318444	21-30



HOSPITAL PATIENT STAY CHART

