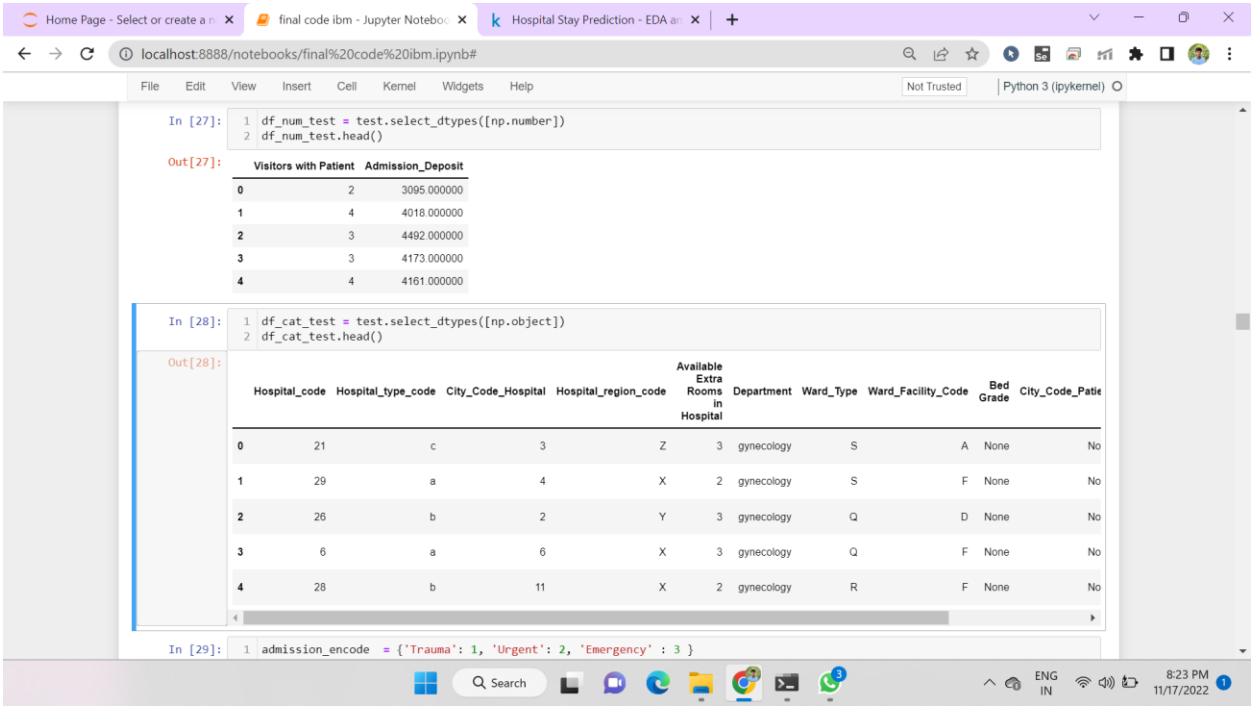
Analytics for Hospitals Health-Care Data

TEAM ID: PNT2022TMID16292

DATA ANALYSIS



```
In [27]:
df_num_test = test.select_dtypes([np.number])
df_num_test.head()
```

Out[27]:

	Visitors with Patient	Admission_Deposit
0	2	3095.000000
1	4	4018.000000
2	3	4492.000000
3	3	4173.000000
4	4	4161.000000

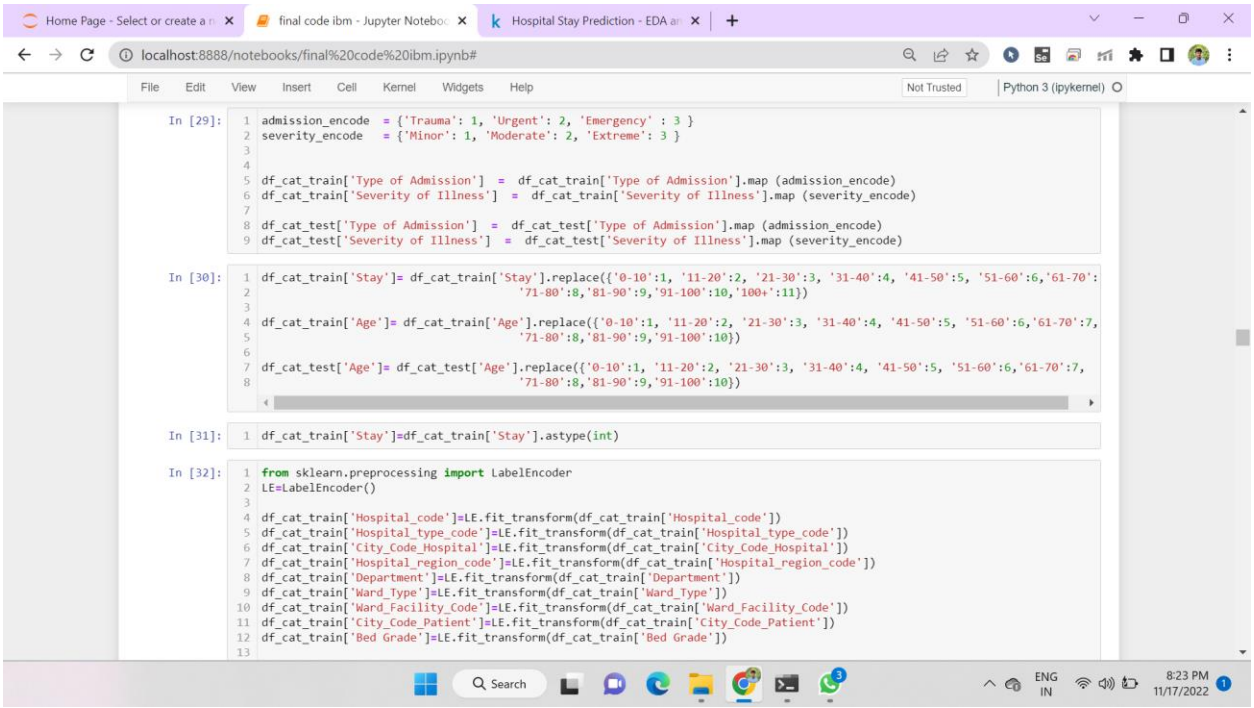
```
In [28]:
df_cat_test = test.select_dtypes([np.object])
df_cat_test.head()
```

Out[28]:

	Hospital_code	Hospital_type_code	City_Code_Hospital	Hospital_region_code	Available Extra Rooms in Hospital	Department	Ward_Type	Ward_Facility_Code	Bed Grade	City_Code_Patie
0	21	c	3	Z	3	gynecology	S	A	None	No
1	29	a	4	X	2	gynecology	S	F	None	No
2	26	b	2	Y	3	gynecology	Q	D	None	No
3	6	a	6	X	3	gynecology	Q	F	None	No
4	28	b	11	X	2	gynecology	R	F	None	No

```
In [29]:
admission_encode  = {'Trauma': 1, 'Urgent': 2, 'Emergency' : 3 }
```

ENCODING



```
In [29]:
admission_encode  = {'Trauma': 1, 'Urgent': 2, 'Emergency' : 3 }
severity_encode   = {'Minor': 1, 'Moderate': 2, 'Extreme': 3 }


df_cat_train['Type of Admission']  =  df_cat_train['Type of Admission'].map (admission_encode)
df_cat_train['Severity of Illness']  =  df_cat_train['Severity of Illness'].map (severity_encode)

df_cat_test['Type of Admission']  =  df_cat_test['Type of Admission'].map (admission_encode)
df_cat_test['Severity of Illness']  =  df_cat_test['Severity of Illness'].map (severity_encode)
```

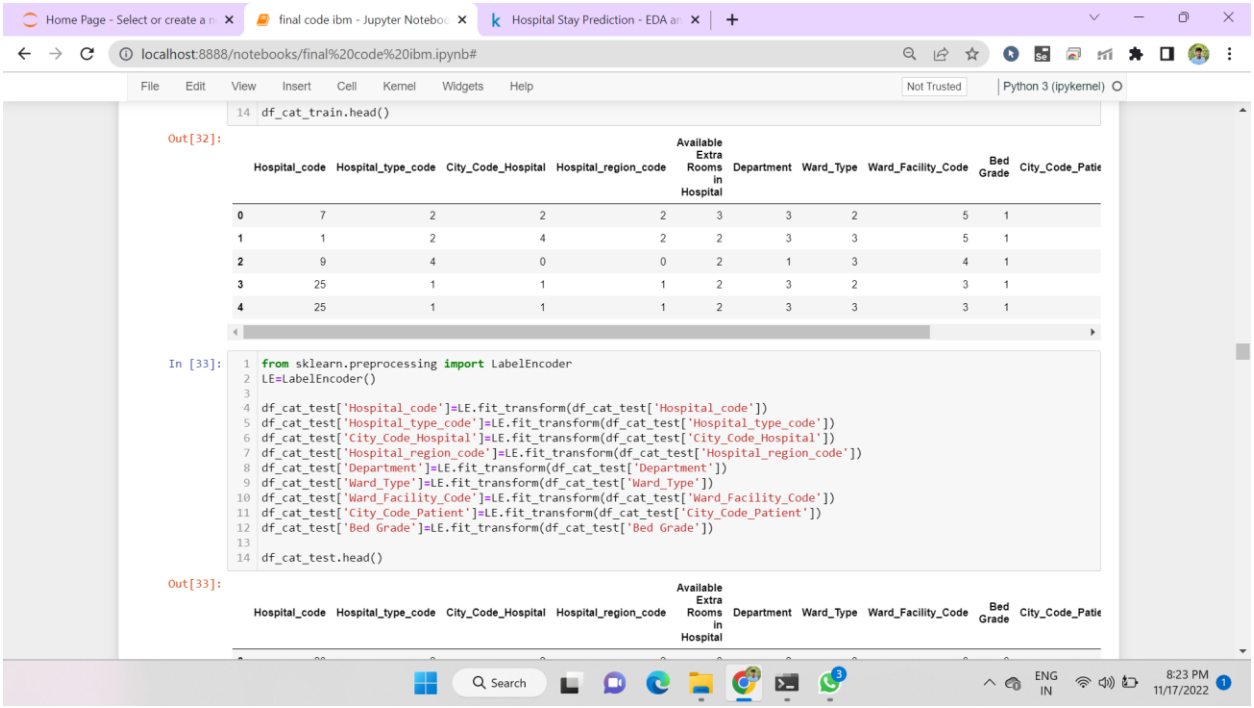
```
In [30]:
df_cat_train['Stay']= df_cat_train['Stay'].replace({'0-10':1, '11-20':2, '21-30':3, '31-40':4, '41-50':5, '51-60':6,'61-70':
                                                    '71-80':8,'81-90':9,'91-100':10,'100+':11})

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,
                                                  '71-80':8,'81-90':9,'91-100':10})

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,
                                                '71-80':8,'81-90':9,'91-100':10})
```

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

```
In [32]:
from sklearn.preprocessing import LabelEncoder
LE=LabelEncoder()

df_cat_train['Hospital_code']=LE.fit_transform(df_cat_train['Hospital_code'])
df_cat_train['Hospital_type_code']=LE.fit_transform(df_cat_train['Hospital_type_code'])
df_cat_train['City_Code_Hospital']=LE.fit_transform(df_cat_train['City_Code_Hospital'])
df_cat_train['Hospital_region_code']=LE.fit_transform(df_cat_train['Hospital_region_code'])
df_cat_train['Department']=LE.fit_transform(df_cat_train['Department'])
df_cat_train['Ward_Type']=LE.fit_transform(df_cat_train['Ward_Type'])
df_cat_train['Ward_Facility_Code']=LE.fit_transform(df_cat_train['Ward_Facility_Code'])
df_cat_train['City_Code_Patient']=LE.fit_transform(df_cat_train['City_Code_Patient'])
df_cat_train['Bed Grade']=LE.fit_transform(df_cat_train['Bed Grade'])
```

Analytics for Hospitals Health-Care Data

TEAM ID: PNT2022TMID16292



```
In [32]:
14 df_cat_train.head()
```

Out[32]:

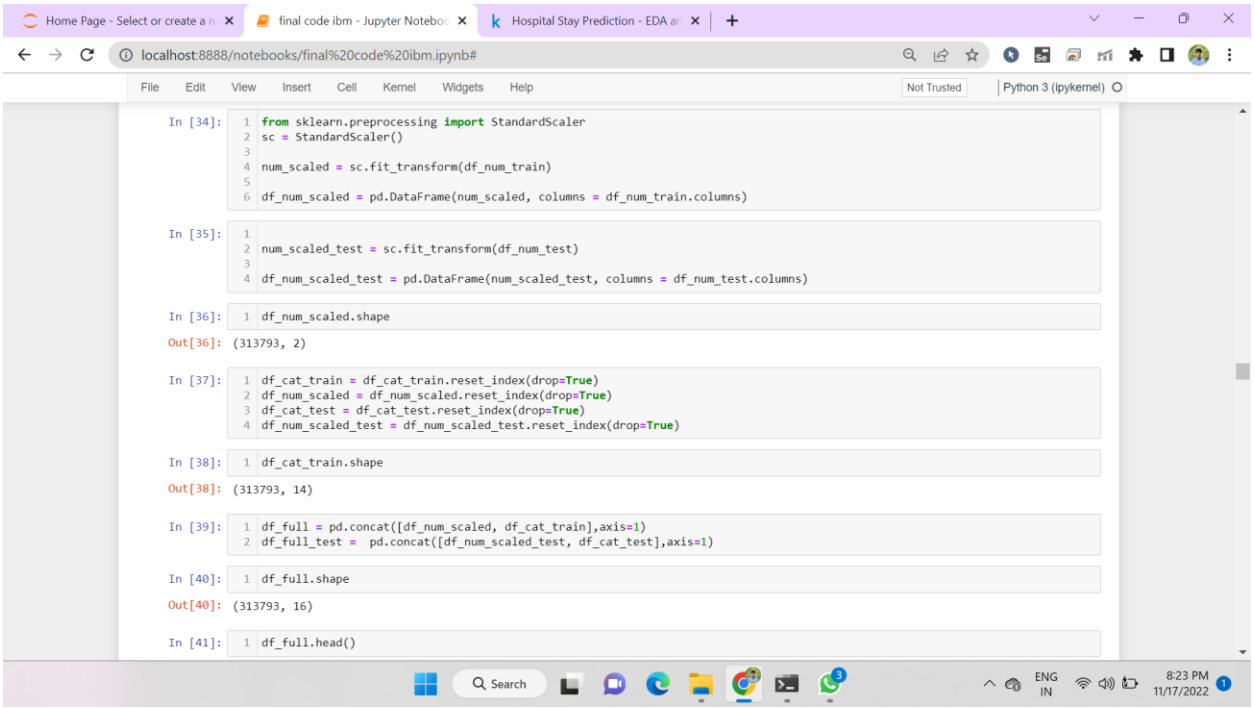
	Hospital_code	Hospital_type_code	City_Code_Hospital	Hospital_region_code	Available Extra Rooms in Hospital	Department	Ward_Type	Ward_Facility_Code	Bed Grade	City_Code_Patie
0	7	2	2	2	3	3	2	5	1	
1	1	2	4	2	2	3	3	5	1	
2	9	4	0	0	2	1	3	4	1	
3	25	1	1	1	2	3	2	3	1	
4	25	1	1	1	2	3	3	3	1	

```
In [33]:
1  from sklearn.preprocessing import LabelEncoder
2  LE=LabelEncoder()
3
4  df_cat_test['Hospital_code']=LE.fit_transform(df_cat_test['Hospital_code'])
5  df_cat_test['Hospital_type_code']=LE.fit_transform(df_cat_test['Hospital_type_code'])
6  df_cat_test['City_Code_Hospital']=LE.fit_transform(df_cat_test['City_Code_Hospital'])
7  df_cat_test['Hospital_region_code']=LE.fit_transform(df_cat_test['Hospital_region_code'])
8  df_cat_test['Department']=LE.fit_transform(df_cat_test['Department'])
9  df_cat_test['Ward_Type']=LE.fit_transform(df_cat_test['Ward_Type'])
10 df_cat_test['Ward_Facility_Code']=LE.fit_transform(df_cat_test['Ward_Facility_Code'])
11 df_cat_test['City_Code_Patient']=LE.fit_transform(df_cat_test['City_Code_Patient'])
12 df_cat_test['Bed Grade']=LE.fit_transform(df_cat_test['Bed Grade'])
13
14 df_cat_test.head()
```

Out[33]:

	Hospital_code	Hospital_type_code	City_Code_Hospital	Hospital_region_code	Available Extra Rooms in Hospital	Department	Ward_Type	Ward_Facility_Code	Bed Grade	City_Code_Patie

SCALE THE DATA



```
In [34]:
1 from sklearn.preprocessing import StandardScaler
2 sc = StandardScaler()
3
4 num_scaled = sc.fit_transform(df_num_train)
5
6 df_num_scaled = pd.DataFrame(num_scaled, columns = df_num_train.columns)
```

```
In [35]:
1
2 num_scaled_test = sc.fit_transform(df_num_test)
3
4 df_num_scaled_test = pd.DataFrame(num_scaled_test, columns = df_num_test.columns)
```

```
In [36]:
1 df_num_scaled.shape
```

Out[36]: (313793, 2)

```
In [37]:
1 df_cat_train = df_cat_train.reset_index(drop=True)
2 df_num_scaled = df_num_scaled.reset_index(drop=True)
3 df_cat_test = df_cat_test.reset_index(drop=True)
4 df_num_scaled_test = df_num_scaled_test.reset_index(drop=True)
```

```
In [38]:
1 df_cat_train.shape
```

Out[38]: (313793, 14)

```
In [39]:
1 df_full = pd.concat([df_num_scaled, df_cat_train],axis=1)
2 df_full_test =  pd.concat([df_num_scaled_test, df_cat_test],axis=1)
```

```
In [40]:
1 df_full.shape
```

Out[40]: (313793, 16)

```
In [41]:
1 df_full.head()
```

Analytics for Hospitals Health-Care Data

TEAM ID: PNT2022TMID16292

```
df_num_scaled = df_num_scaled.reset_index(drop=True)
df_cat_test = df_cat_test.reset_index(drop=True)
df_num_scaled_test = df_num_scaled_test.reset_index(drop=True)
```

In [38]:
```
df_cat_train.shape
```
Out[38]: (313793, 14)

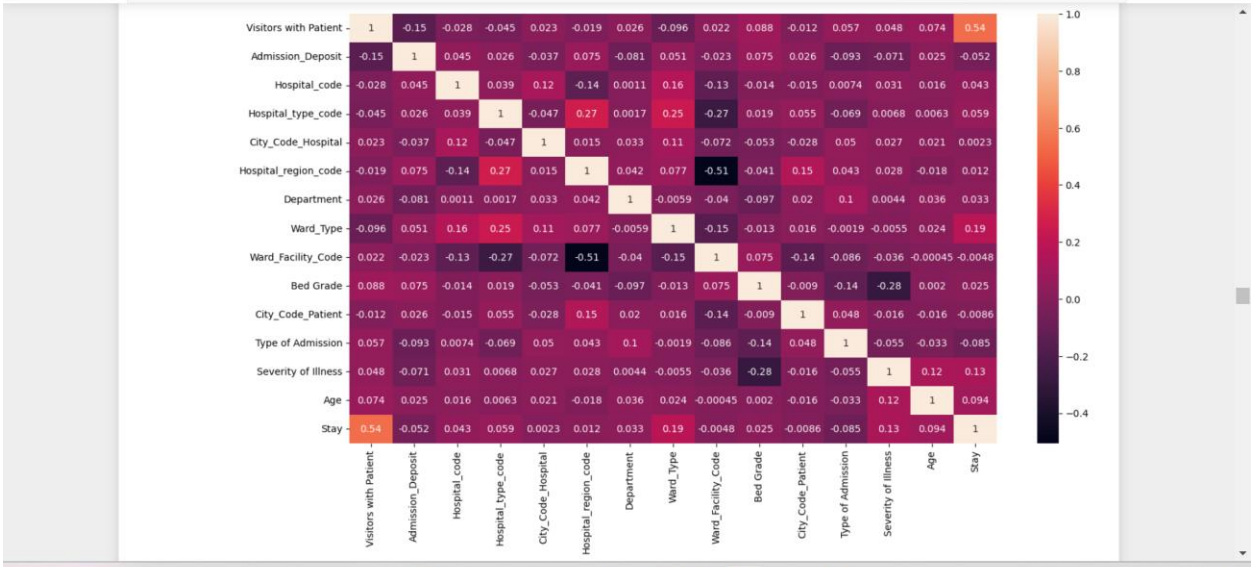
In [39]:
```
df_full = pd.concat([df_num_scaled, df_cat_train],axis=1)
df_full_test =  pd.concat([df_num_scaled_test, df_cat_test],axis=1)
```

In [40]:
```
df_full.shape
```
Out[40]: (313793, 16)

In [41]:
```
df_full.head()
```
Out[41]:

	Visitors with Patient	Admission_Deposit	Hospital_code	Hospital_type_code	City_Code_Hospital	Hospital_region_code	Available Extra Rooms in Hospital	Department	Ward_Type	Ward_Facility
0	-0.727035	0.026796	7	2	2	2	3	3	2	
1	-0.727035	0.986987	1	2	4	2	2	3	3	
2	-0.727035	-0.126025	9	4	0	0	2	1	3	
3	-0.727035	2.200344	25	1	1	1	2	3	2	
4	-0.727035	0.622427	25	1	1	1	2	3	3	

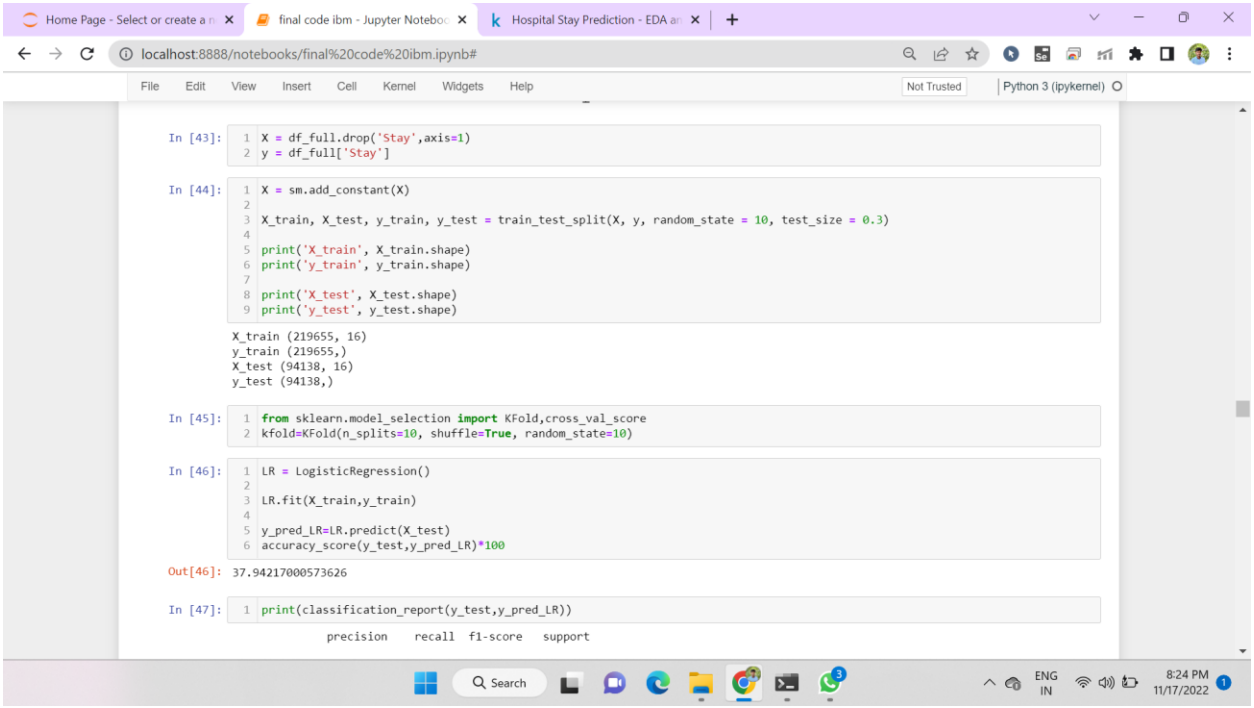
In [42]:
```
sns.heatmap(df_full.corr(), annot = True)
```
Out[42]: <AxesSubplot:>



Analytics for Hospitals Health-Care Data

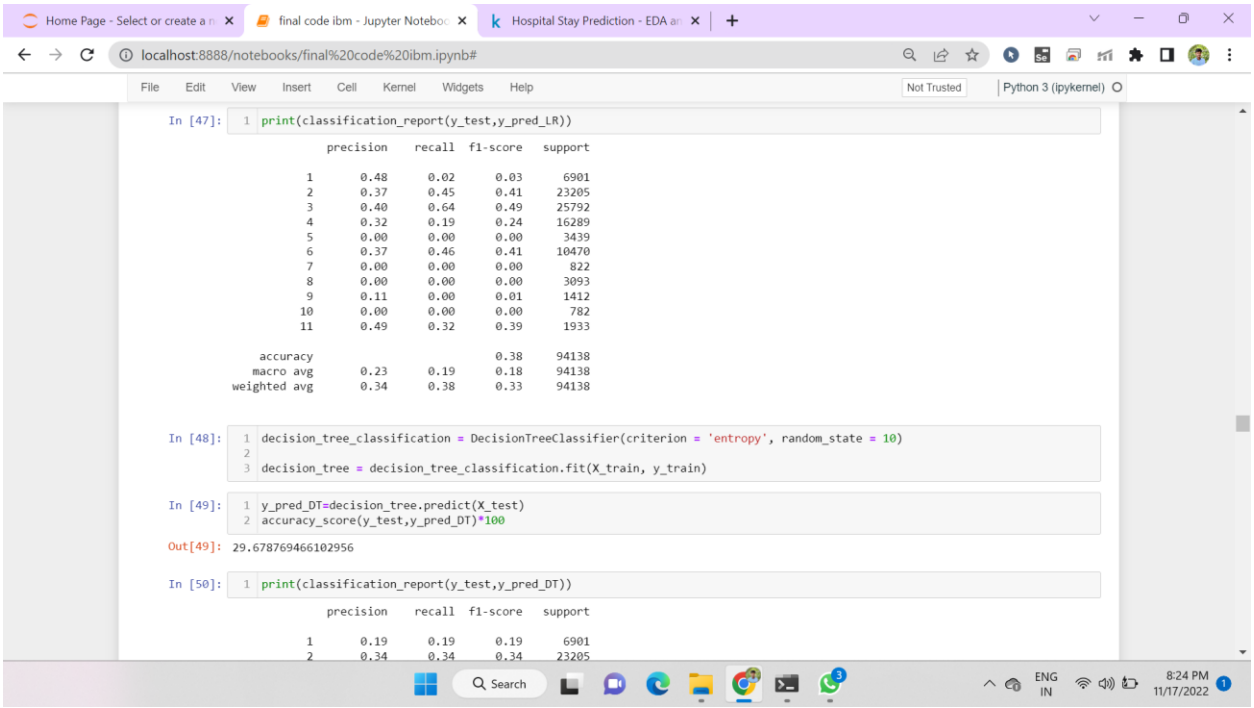
TEAM ID: PNT2022TMID16292

TRAIN TEST SPLIT



```
In [43]:
X = df_full.drop('Stay',axis=1)
y = df_full['Stay']

In [44]:
X = sm.add_constant(X)

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 10, test_size = 0.3)

print('X_train', X_train.shape)
print('y_train', y_train.shape)

print('X_test', X_test.shape)
print('y_test', y_test.shape)

X_train (219655, 16)
y_train (219655,)
X_test (94138, 16)
y_test (94138,)

In [45]:
from sklearn.model_selection import KFold,cross_val_score
kfold=KFold(n_splits=10, shuffle=True, random_state=10)

In [46]:
LR = LogisticRegression()

LR.fit(X_train,y_train)

y_pred_LR=LR.predict(X_test)
accuracy_score(y_test,y_pred_LR)*100

Out[46]: 37.94217000573626

In [47]:
print(classification_report(y_test,y_pred_LR))
              precision    recall  f1-score   support
```

FOLD CROSS VALIDATION AND LOGISTIC REGRESSION

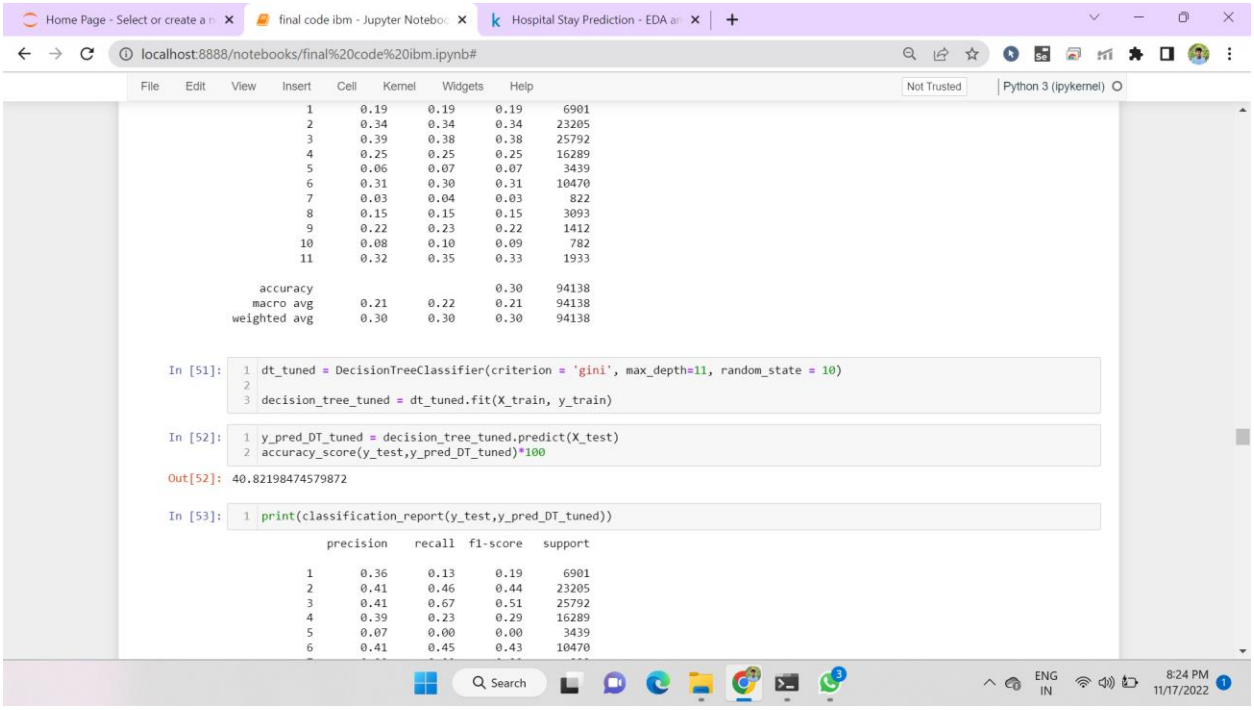


```
In [47]:
print(classification_report(y_test,y_pred_LR))
              precision    recall  f1-score   support

           1       0.48      0.02      0.03      6901
           2       0.37      0.45      0.41     23205
           3       0.40      0.64      0.49     25792
           4       0.32      0.19      0.24     16289
           5       0.00      0.00      0.00      3439
           6       0.37      0.46      0.41     10470
           7       0.00      0.00      0.00       822
           8       0.00      0.00      0.00      3093
           9       0.11      0.00      0.01      1412
          10       0.00      0.00      0.00       782
          11       0.49      0.32      0.39      1933

    accuracy                           0.38     94138
   macro avg       0.23      0.19      0.18     94138
weighted avg       0.34      0.38      0.33     94138


In [48]:
decision_tree_classification = DecisionTreeClassifier(criterion = 'entropy', random_state = 10)

decision_tree = decision_tree_classification.fit(X_train, y_train)

In [49]:
y_pred_DT=decision_tree.predict(X_test)
accuracy_score(y_test,y_pred_DT)*100

Out[49]: 29.678769466102956

In [50]:
print(classification_report(y_test,y_pred_DT))
              precision    recall  f1-score   support

           1       0.19      0.19      0.19      6901
           2       0.34      0.34      0.34     23205
```

Analytics for Hospitals Health-Care Data

TEAM ID: PNT2022TMID16292

DECISION TREE



```
                  1       0.19      0.19      0.19      6901
                  2       0.34      0.34      0.34     23205
                  3       0.39      0.38      0.38     25792
                  4       0.25      0.25      0.25     16289
                  5       0.06      0.07      0.07      3439
                  6       0.31      0.30      0.31     10470
                  7       0.03      0.04      0.03       822
                  8       0.15      0.15      0.15      3093
                  9       0.22      0.23      0.22      1412
                 10       0.08      0.10      0.09       782
                 11       0.32      0.35      0.33      1933

           accuracy                           0.30     94138
          macro avg       0.21      0.22      0.21     94138
       weighted avg       0.30      0.30      0.30     94138
```

```
In [51]:
1 dt_tuned = DecisionTreeClassifier(criterion = 'gini', max_depth=11, random_state = 10)
2
3 decision_tree_tuned = dt_tuned.fit(X_train, y_train)
```

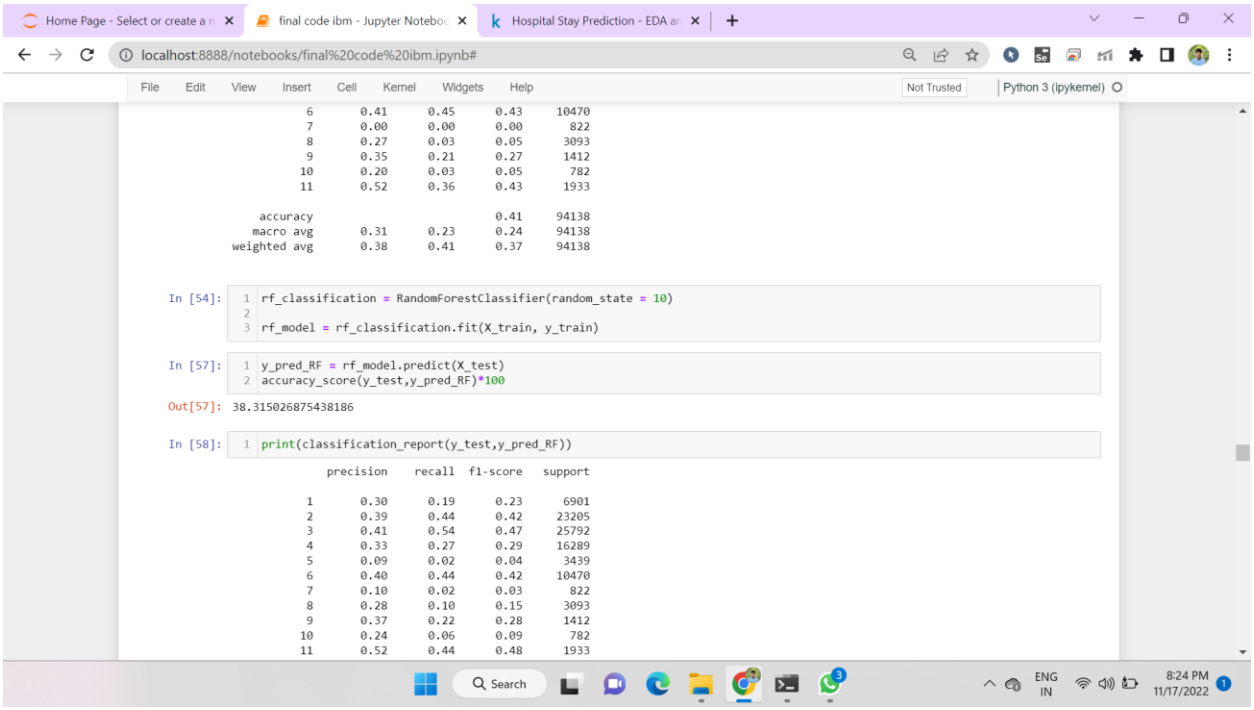
```
In [52]:
1 y_pred_DT_tuned = decision_tree_tuned.predict(X_test)
2 accuracy_score(y_test,y_pred_DT_tuned)*100
```

```
Out[52]: 40.82198474579872
```

```
In [53]:
1 print(classification_report(y_test,y_pred_DT_tuned))
```

```
              precision    recall  f1-score   support

           1       0.36      0.13      0.19      6901
           2       0.41      0.46      0.44     23205
           3       0.41      0.67      0.51     25792
           4       0.39      0.23      0.29     16289
           5       0.07      0.00      0.00      3439
           6       0.41      0.45      0.43     10470
```

TUNED HYPERPARAMETER DECISION TREE



```
                  6       0.41      0.45      0.43     10470
                  7       0.00      0.00      0.00       822
                  8       0.27      0.03      0.05      3093
                  9       0.35      0.21      0.27      1412
                 10       0.20      0.03      0.05       782
                 11       0.52      0.36      0.43      1933

           accuracy                           0.41     94138
          macro avg       0.31      0.23      0.24     94138
       weighted avg       0.38      0.41      0.37     94138
```

```
In [54]:
1 rf_classification = RandomForestClassifier(random_state = 10)
2
3 rf_model = rf_classification.fit(X_train, y_train)
```

```
In [57]:
1 y_pred_RF = rf_model.predict(X_test)
2 accuracy_score(y_test,y_pred_RF)*100
```

```
Out[57]: 38.315026875438186
```

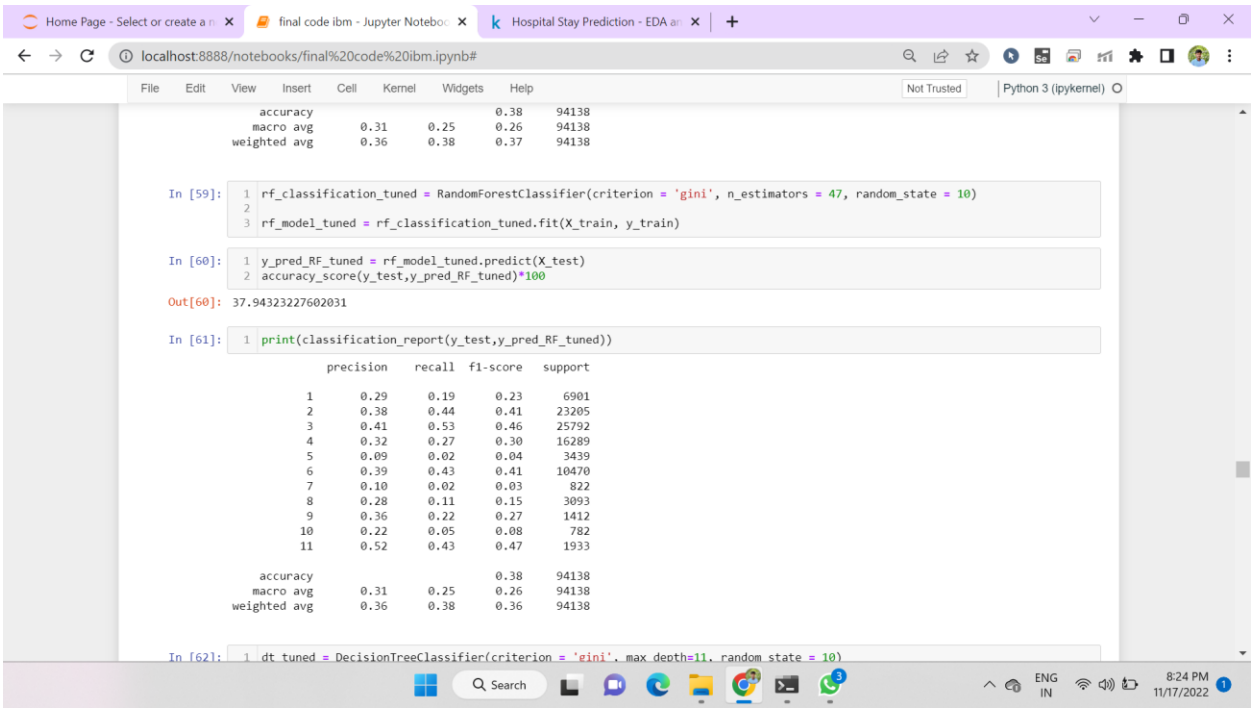
```
In [58]:
1 print(classification_report(y_test,y_pred_RF))
```

```
              precision    recall  f1-score   support

           1       0.30      0.19      0.23      6901
           2       0.39      0.44      0.42     23205
           3       0.41      0.54      0.47     25792
           4       0.33      0.27      0.29     16289
           5       0.09      0.02      0.04      3439
           6       0.40      0.44      0.42     10470
           7       0.10      0.02      0.03       822
           8       0.28      0.10      0.15      3093
           9       0.37      0.22      0.28      1412
          10       0.24      0.06      0.09       782
          11       0.52      0.44      0.48      1933
```

Analytics for Hospitals Health-Care Data

TEAM ID: PNT2022TMID16292

RANDOM FOREST



```
                accuracy                           0.38     94138
               macro avg       0.31      0.25      0.26     94138
            weighted avg       0.36      0.38      0.37     94138
```

In [59]:
```
rf_classification_tuned = RandomForestClassifier(criterion = 'gini', n_estimators = 47, random_state = 10)

rf_model_tuned = rf_classification_tuned.fit(X_train, y_train)
```

In [60]:
```
y_pred_RF_tuned = rf_model_tuned.predict(X_test)
accuracy_score(y_test,y_pred_RF_tuned)*100
```

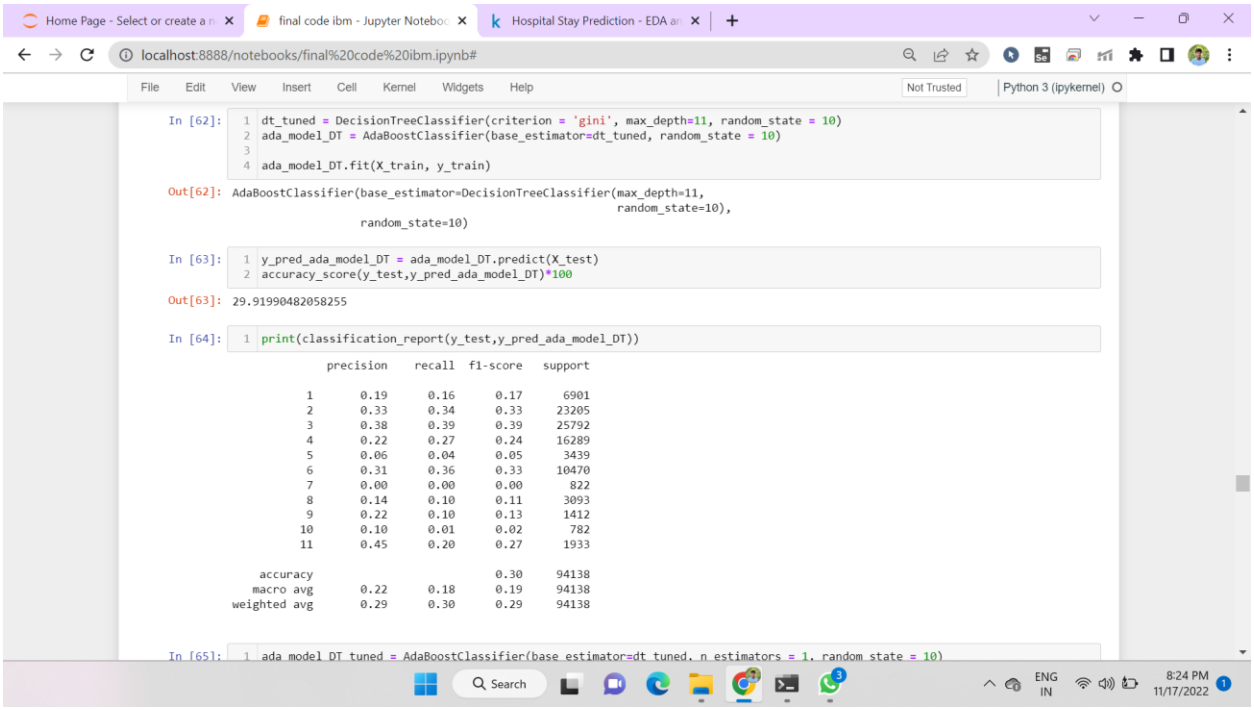
Out[60]: 37.94323227602031

In [61]:
```
print(classification_report(y_test,y_pred_RF_tuned))
```

```
              precision    recall  f1-score   support

           1       0.29      0.19      0.23      6901
           2       0.38      0.44      0.41     23205
           3       0.41      0.53      0.46     25792
           4       0.32      0.27      0.30     16289
           5       0.09      0.02      0.04      3439
           6       0.39      0.43      0.41     10470
           7       0.10      0.02      0.03       822
           8       0.28      0.11      0.15      3093
           9       0.36      0.22      0.27      1412
          10       0.22      0.05      0.08       782
          11       0.52      0.43      0.47      1933

    accuracy                           0.38     94138
   macro avg       0.31      0.25      0.26     94138
weighted avg       0.36      0.38      0.36     94138
```

In [62]:
```
dt_tuned = DecisionTreeClassifier(criterion = 'gini', max_depth=11, random_state = 10)
```

TUNED HYPERPARAMETER RANDOM FOREST AND ADA BOOST-DECISION TREE



In [62]:
```
dt_tuned = DecisionTreeClassifier(criterion = 'gini', max_depth=11, random_state = 10)
ada_model_DT = AdaBoostClassifier(base_estimator=dt_tuned, random_state = 10)

ada_model_DT.fit(X_train, y_train)
```

Out[62]:
```
AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=11,
                                                         random_state=10),
                   random_state=10)
```

In [63]:
```
y_pred_ada_model_DT = ada_model_DT.predict(X_test)
accuracy_score(y_test,y_pred_ada_model_DT)*100
```

Out[63]: 29.91990482058255

In [64]:
```
print(classification_report(y_test,y_pred_ada_model_DT))
```

```
              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                           0.30     94138
   macro avg       0.22      0.18      0.19     94138
weighted avg       0.29      0.30      0.29     94138
```

In [65]:
```
ada_model_DT_tuned = AdaBoostClassifier(base_estimator=dt_tuned, n_estimators = 1, random_state = 10)
```