PROJECT ON HEALTHCARE ANALYTICS

By,

Vishnu Breethy

Contents

Introduction 2
Project Goal 2
Hypothesis Generation 2
4. Data Exploration 2
4.1 Overview of Data 3
4.2 Data Cleaning and Preparation 4
4.3 Feature Engineering 4
5. Modelling Strategy 5
5.1 Model 1 - Naïve Bayes 5
5.2 Model 2 – XGBoost 6
6. Prediction and Results 7
7. Future Insights 8
8. Conclusion 8

1.Introduction

Healthcare organizations are under increasing pressure to improve patient care outcomes and achieve better care. While this situation represents a challenge, it also offers organizations an opportunity to dramatically improve the quality of care by leveraging more value and insights from their data. Health care analytics refers to the analysis of data using quantitative and qualitative techniques to explore trends and patterns in the acquired data. While healthcare management uses various metrics for performance, a patient's length of stay is an important one. Being able to predict the length of stay (LOS) allows hospitals to optimize their treatment plans to reduce LOS, to reduce infection rates among patients, staff, and visitors.

2. Project Goal

The goal of this project is to accurately predict the Length of Stay for each patient so that the hospitals can optimize resources and function better.

3. Hypothesis Generation

Understanding the problem in detail by assuming different factors that impact the outcomes of Length of Stay before any data exploration or analysis. Here the variables can be divided into two levels: Patient-Level and Hospital-Level.

Patient-Level:

- Type of Admission – Patients can be admitted in three levels Urgent, Emergency, and Trauma. Patients admitted to urgent care are likely to stay fewer days. Whereas Trauma patients usually stay longer because they must be monitored until they are qualified to be discharged.
- Severity of Illness – Severity can be classified as Minor, Moderate, and Extreme. A patient recorded as minor will stay fewer days than a patient recorded as extreme.
- Visitors with Patient – Patients with more visitors are like to stay longer in the hospital.
- Age – Infants and older Patients usually take a longer time to recover so they stay longer than younger Patients.
- Admission Deposit – Patients who are likely to deposit a high amount of money at the time of admission might have severe conditions and stay longer.

Hospital-Level:

- Ward Type – Patients allocated in ICU might stay longer than the general ward as their condition is more severe.
- Department – Patients under surgery are likely to stay longer than gynecology as their recovery time is longer.

4. Data Exploration

4.1 Overview of Data

The train data consist of 318438 observations for which patient length of stay can be predicted from 17 variables. The description of all variables is shown in Table 4.1.

Variables	Description
Case_id	Case_ID registered in hospital
Hospital_code	Unique code for the hospital
Hospital_type_code	Unique code for the type of Hospital
City_Code_Hospital	City Code of the Hospital
Hospital_region_code	Region Code of the Hospital
Available Extra Rooms in Hospital	Number of Extra rooms available in the Hospital
Department	Department overlooking the case
Ward_Type	Code for the Ward type
Ward_Facility_Code	Code for the Ward Facility
Bed Grade	Condition of Bed in the Ward
patientid	Unique Patient Id
City_Code_ patient	City Code for the patient
Type of Admission	Admission Type registered by the Hospital
Severity of Illness	Severity of the illness recorded at the time of admission
Visitors with Patient	Patient Number of Visitors with the patient
Age	Age of the patient
Admission_Deposit	Deposit at the time of Admission
Stay	Patient Length of Stay

Table4.1 overview of data

In this data, the target variable “stay” is divided into 11 different classes ranging from 0 days to more than 100 days.

```
array(['0-10', '41-50', '31-40', '11-20', '51-60', '21-30', '71-80',
       'More than 100 Days', '81-90', '61-70', '91-100'], dtype=object)
```

Figure 4.1 Unique values in Stay Column

4.2 Data Cleaning and Preparation

In this data set, variables "City_code_patient" and "Bed Grade" have missing values which needs to be treated as they distort the model performance. So, the missing values are treated using the 'mode' imputation technique.

Since most of the variables in the dataset have ordinal data, we transformed them into levels by using a label encoder to perform further analysis on the data. Table 4.2 ,4.3 shows the number of distinct observations of ordinal data in the dataset.

```
case_id : 318438
Hospital_code : 32
Hospital_type_code : 7
City_Code_Hospital : 11
Hospital_region_code : 3
Available Extra Rooms in Hospital : 18
Department : 5
Ward_Type : 6
Ward_Facility_Code : 6
Bed Grade : 4
patientid : 92017
City_Code_Patient : 37
Type of Admission : 3
Severity of Illness : 3
Visitors with Patient : 28
Age : 10
Admission_Deposit : 7300
Stay : 11
```

Table 4.2 distinct observations of train data

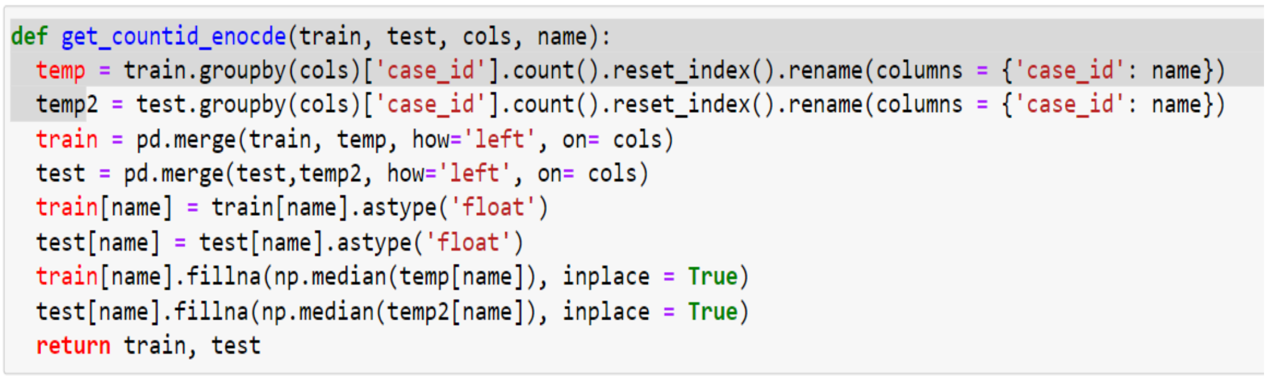
```
case_id : 137057
Hospital_code : 32
Hospital_type_code : 7
City_Code_Hospital : 11
Hospital_region_code : 3
Available Extra Rooms in Hospital : 15
Department : 5
Ward_Type : 6
Ward_Facility_Code : 6
Bed Grade : 4
patientid : 39607
City_Code_Patient : 37
Type of Admission : 3
Severity of Illness : 3
Visitors with Patient : 27
Age : 10
Admission_Deposit : 6609
```

Table 4.3 distinct observations of test data

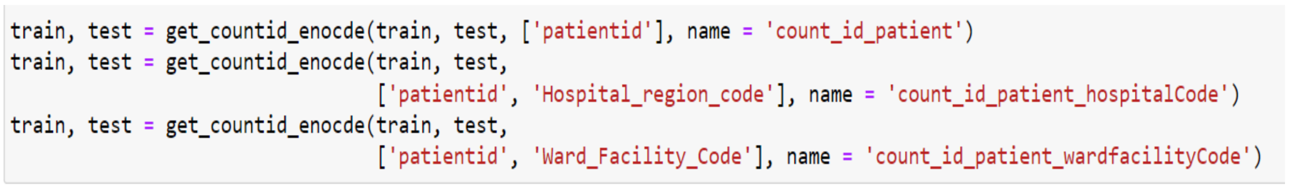
4.3 Feature Engineering

Once the data is cleaned and prepared, we grouped patientid and case_id to extract the new column "count_id_patient". This variable contains the count of multiple admits of a patient under different case_id. Further two more columns "Hospital_region_code" and "ward_facility_code" were grouped to patientid and case_id. These two new variables "count_id_patient_hospitalCode" and "count_id_patient_wardfacilityCode" contain the count of multiple admissions in a hospital region and the count of multiple wards allocated to a patient.

Before getting into analysis, the train data must be split into two parts, the first part with all the feature variables and the second part with a target variable ("Stay"). Then preprocessed into train and validation sets. So, here we are portioning the train set with 80% and validation set with 20% of the data for Naïve Bayes and XGBoost models.



```
def get_countid_enocde(train, test, cols, name):
  temp = train.groupby(cols)['case_id'].count().reset_index().rename(columns = {'case_id': name})
  temp2 = test.groupby(cols)['case_id'].count().reset_index().rename(columns = {'case_id': name})
  train = pd.merge(train, temp, how='left', on= cols)
  test = pd.merge(test,temp2, how='left', on= cols)
  train[name] = train[name].astype('float')
  test[name] = test[name].astype('float')
  train[name].fillna(np.median(temp[name]), inplace = True)
  test[name].fillna(np.median(temp2[name]), inplace = True)
  return train, test
```



```
train, test = get_countid_enocde(train, test, ['patientid'], name = 'count_id_patient')
train, test = get_countid_enocde(train, test,
                                 ['patientid', 'Hospital_region_code'], name = 'count_id_patient_hospitalCode')
train, test = get_countid_enocde(train, test,
                                 ['patientid', 'Ward_Facility_Code'], name = 'count_id_patient_wardfacilityCode')
```

4.3 feature engineering

5. Modelling Strategy

5.1 Model 1- Naïve Bayes

Naïve Bayes is a classification technique that works on the principle of Bayes theorem with an assumption of independence among the variables. Here the goal is to predict Length of Stay i.e., "Stay" column (Target Variable) and it is classified into 11 levels. We must find the probability of each patient's length of stay using feature variables, which contain the patient's condition and hospital-level information. These feature variables are ordinal and naïve Bayes is a perfect multilevel classifier.

In Bayes theorem, given a Hypothesis H and Evidence E, it states that the relation between the probability of Hypothesis P(H) before getting Evidence and probability of hypothesis after getting Evidence P(H|E)

$$P\ (H \mid E) = [\ \boldsymbol{P}\ (\boldsymbol{E} \mid \boldsymbol{H})\ \boldsymbol{P}(\boldsymbol{E})\ ]\ P(H)$$

When we apply Bayes Theorem to our data it represents as follows.

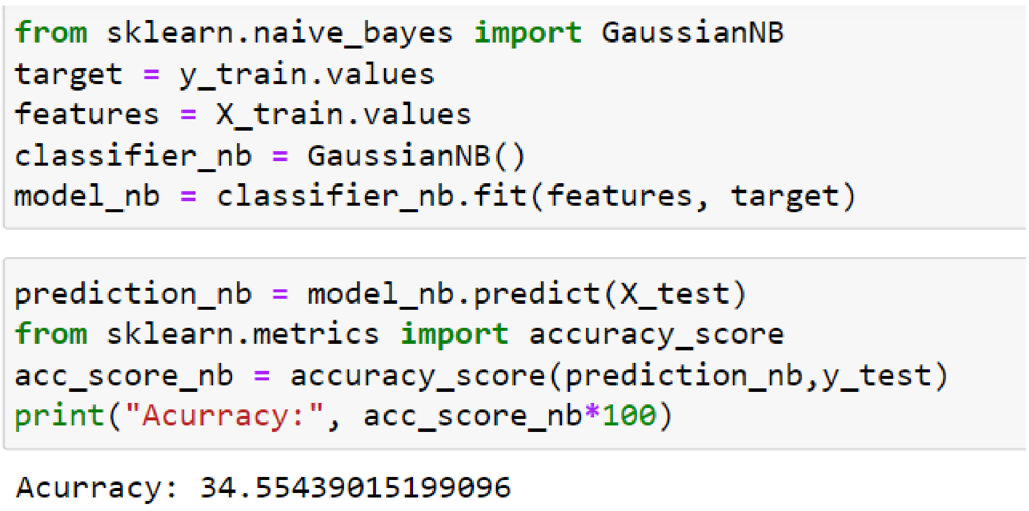
• P(H) is the prior probability of a patient's length of stay (LOS).

• P(E) is the probability of a feature variable.

• P(E|H) is the probability of a patient's LOS given that the features are true.

• P(H|E) is the probability of the features given that patient's LOS is true.

Model is trained using Gaussian Naïve Bayes classifier, partitioned train data is fed to the model in array format then the trained model is validated using validation data. This model gives an accuracy score of 34.55% after validating.



```
from sklearn.naive_bayes import GaussianNB
target = y_train.values
features = X_train.values
classifier_nb = GaussianNB()
model_nb = classifier_nb.fit(features, target)
```

```
prediction_nb = model_nb.predict(X_test)
from sklearn.metrics import accuracy_score
acc_score_nb = accuracy_score(prediction_nb,y_test)
print("Acurracy:", acc_score_nb*100)
```

```
Acurracy: 34.55439015199096
```

5.1 Naïve Bayes model

5.2 Model 2 – XGBoost

Boosting is a sequential technique that works on the principle of an ensemble. At any instant T, the model outcomes are weighed based on the outcomes of the previous instant (T -1). It combines the set of weak learners and improves prediction accuracy. Tree ensemble is a set of classification and regression trees. Trees are grown one after another, and they try to reduce the misclassification rate. The final prediction score of the model is calculated by summing up each and individual score.

Before feeding train data to the XGB Classifier model, booster parameters must be tuned. Tunning the model can prevent overfitting and can yield higher accuracy.

In this XGBoost model, we have used the following parameters for tunning,

• learning_rate = 0.1 - step size shrinkage used to prevent overfitting. After each boosting step, we can directly get the weights of new features, and eta shrinks the feature weights to make the boosting process more conservative. 6

• max_depth = 4 – Maximum depth of the tree. This value describes the complexity of the model. Increasing its value results in overfitting.

• n_estimators = 800 – Number of gradient boosting trees or rounds. Each new tree attempts to model and correct for the errors made by the sequence of previous trees. Increasing the number of trees can yield higher accuracy but the model reaches a point of diminishing returns quickly.

• objective = 'multi:softmax' – this parameter sets XGBoost to do multiclass classification using the softmax objective because the target variable has 11 Levels.

• reg_alpha = 0.5 - L1 regularization term on weights. Increasing this value will make the model more conservative.
• reg_lambda = 1.5 - L2 regularization term on weights and is smoother than L1 regularization. Increasing this value will model more conservative.

• min_child_weight = 2 - Minimum sum of instance weight needed in a child. Once the model was trained and validated, it yields an accuracy score of 43.04%. When compared to the Naïve Bayes model that's an 8.5% improvement.

```
import xgboost
classifier_xgb = xgboost.XGBClassifier(max_depth=4, learning_rate=0.1, n_estimators=800,
                                      objective='multi:softmax', reg_alpha=0.5, reg_lambda=1.5,
                                      booster='gbtree', n_jobs=4, min_child_weight=2, base_score= 0.75)
```

```
model_xgb = classifier_xgb.fit(X_train, y_train)
```

```
prediction_xgb = model_xgb.predict(X_test)
acc_score_xgb = accuracy_score(prediction_xgb,y_test)
print("Accuracy:", acc_score_xgb*100)
```

```
Accuracy: 43.030084160281376
```

5.2 Xg boost model

6. Prediction and Results

In the Naïve Bayes model, patients are more likely to be misclassified. This model is biased towards the duration of 21-30 days, it has classified 72,206 patients for this level (can be observed in Table 6.1).

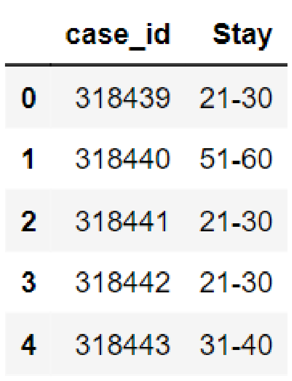
NAÏVE BAYES MODEL

```
Stay
0-10                    2598
11-20                  26827
21-30                  72206
31-40                  15639
41-50                    469
51-60                  13651
61-70                     92
71-80                    955
81-90                    296
91-100                     2
More than 100 Days      4322
Name: case_id, dtype: int64
```

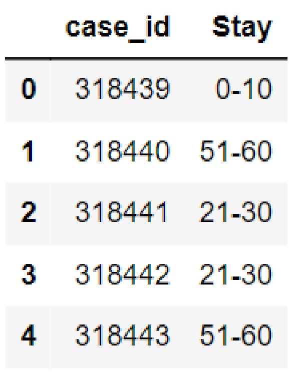
XG BOOST MODEL

```
Stay
0-10                    4462
11-20                  39080
21-30                  58320
31-40                  12554
41-50                     60
51-60                  18900
61-70                     12
71-80                    297
81-90                   1112
91-100                    70
More than 100 Days      2190
Name: case_id, dtype: int64
```

Table 6.1 – Number of observations classified into different levels of Length of Stay from all models



	case_id	Stay
0	318439	21-30
1	318440	51-60
2	318441	21-30
3	318442	21-30
4	318443	31-40



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

Table 6.2 – Predicted Length of Stay for first five cases from different models

Examining these predictions, many of the patients are staying in the hospital for 21-30 days and very few people are staying for 61-70 days

7. Future Insights

• Smart Staffing & Personnel Management: having a large volume of quality data helps health care professionals in allocating resources efficiently. Healthcare professionals can analyze the outcomes of checkups among individuals in various demographic groups and determine what factors prevent individuals from seeking treatment.

• Advanced Risk & Disease Management: Healthcare institutions can offer accurate, preventive care. Effectively decreasing hospital admissions by digging into insights such as drug type, conditions, and the duration of patient visits, among many others.

• Real-time Alerting: Clinical Decision Support (CDS): applications in hospitals analyzes patient evidence on the spot, delivering recommendations to health professionals when they make prescriptive choices. However, to prevent unnecessary in-house procedures, physicians prefer people to stay away from hospitals

• Enhancing Patient Engagement: Every step they take, heart rates, sleeping habits, can be tracked for potential patients (who use smart wearables). All this information can be correlated with other trackable data to identify potential health risks.

8. Conclusion

In this project, different variables were analyzed that correlate with Length of Stay by using patient-level and hospital-level data. By predicting a patient's length of stay at the time of admission helps hospitals to allocate resources more efficiently and manage their patients more effectively. Identifying factors that associate with LOS to predict and manage the number of days patients stay, could help hospitals in managing resources and in the development of new treatment plans. Effective use of hospital resources and reducing the length of stay can reduce overall national medical expenses.