HOSPITAL LENGTH OF STAY PREDICTION

Data Science & Artificial Intelligence

BYRON O'CONNELL
INSTITUTE OF DATA
Capstone Presentation

Contents

Problem Statement

Patient length of stay is an important indicator of the efficiency of hospital management. Hospitals have limited resources and staff, requiring efficient use of beds and clinician time. With the relatively recent Coronavirus outbreak, combined with lower levels of staffing across Australia, the importance of patient length of stay has been heightened. More than ever, we can see that it is in the best interest of patients, hospitals and public health to limit hospital stays to no longer than necessary and to have a good idea of how long a given patient may need to spend at the hospital.

Stakeholders

Patient length of stay in a hospital is an important indicator of the efficiency of hospital management. Research has shown that the reduction in the number of inpatient days results in:

- Decreased risk of infection for the patient
- Decreased risk of medication side effects for the patient
- Improved quality of treatment for the patient
- More efficient bed management for the hospital – meaning more room for other patients in need of treatment.
- Greater satisfaction for families of the patient who will have a good idea of the length of stay for their family member.

Business question

This project aims to determine which factors are associated with length of hospital stay, based on historical electronic health records, in order to manage hospital stay more efficiently. It also builds a model which aims to predict the length of stay of a patient when presenting to hospital.

Can we accurately predict the length of stay for a patient upon presenting to hospital with a diagnosis?

Dataset

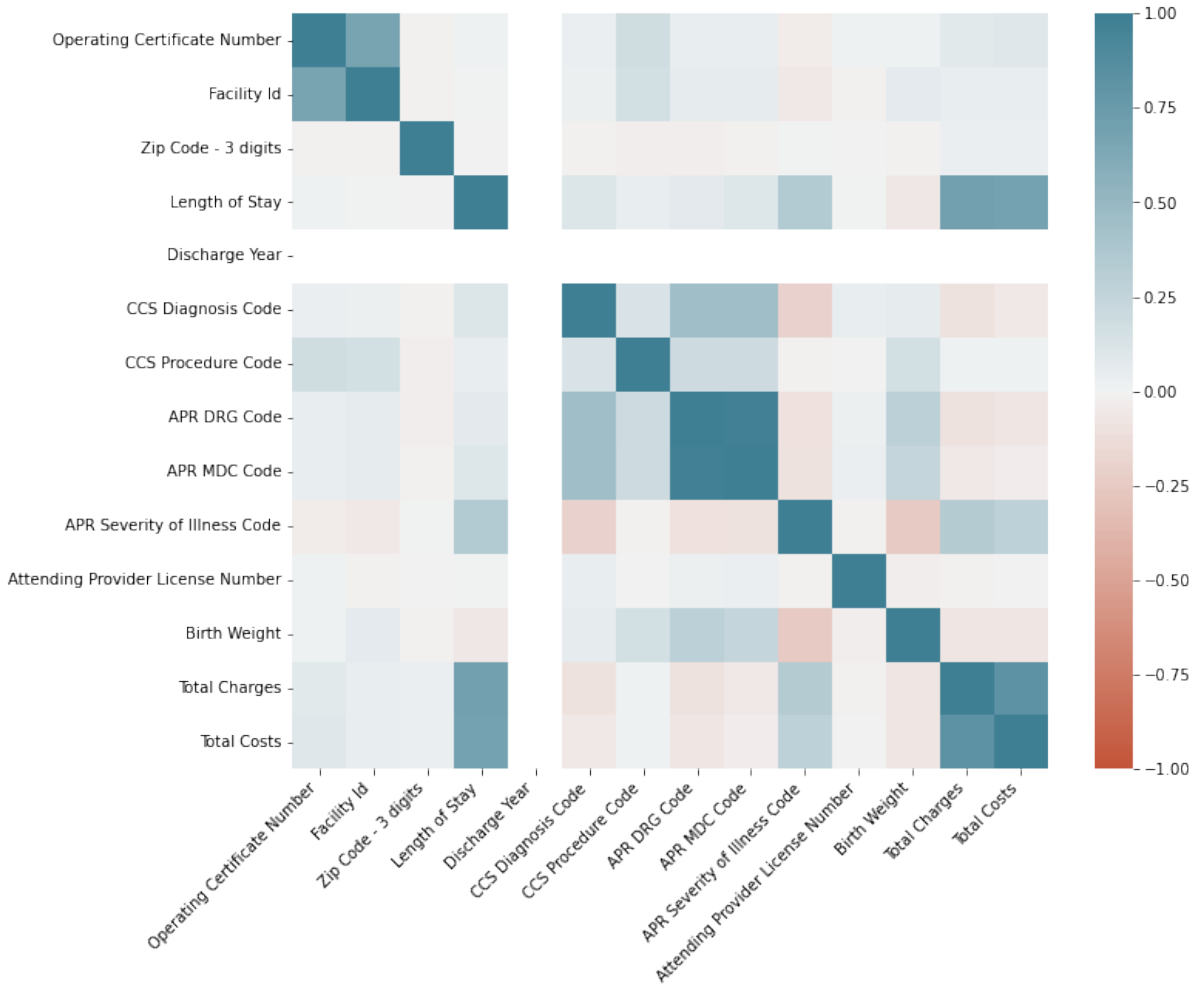
To conduct this analysis, I used the publicly available "2015 de-identified NY inpatient discharge (SPARCS)" (SPARCS stands for Statewide Planning and Research Cooperative System) accessed via Kaggle from the New York State Government health data website. The dataset contains around 2.3 million rows of patient data, including information such as patient demographics, diagnoses, treatments, services and costs. Patient data has been de-identified according to HIPAA (Health Insurance Portability and Accountability Act) regulations.

The bulk of my code uses Pandas and scikit learn.

Below we see all the features and associated datatypes in the dataset.

```
Health Service Area                    object
Hospital County                        object
Operating Certificate Number          float64
Facility Id                           float64
Facility Name                          object
Age Group                              object
Zip Code - 3 digits                    object
Gender                                 object
Race                                   object
Ethnicity                              object
Length of Stay                         object
Type of Admission                      object
Patient Disposition                    object
Discharge Year                          int64
CCS Diagnosis Code                      int64
CCS Diagnosis Description              object
CCS Procedure Code                      int64
CCS Procedure Description              object
APR DRG Code                            int64
APR DRG Description                    object
APR MDC Code                            int64
APR MDC Description                    object
APR Severity of Illness Code            int64
APR Severity of Illness Description    object
APR Risk of Mortality                  object
APR Medical Surgical Description       object
Payment Typology 1                     object
Payment Typology 2                     object
Payment Typology 3                     object
Attending Provider License Number     float64
Operating Provider License Number     float64
Other Provider License Number         float64
Birth Weight                            int64
Abortion Edit Indicator                object
Emergency Department Indicator         object
Total Charges                          object
Total Costs                            object
dtype: object
```

Next, we have a look at how our numerical features are correlated with each other using a correlation heat map.

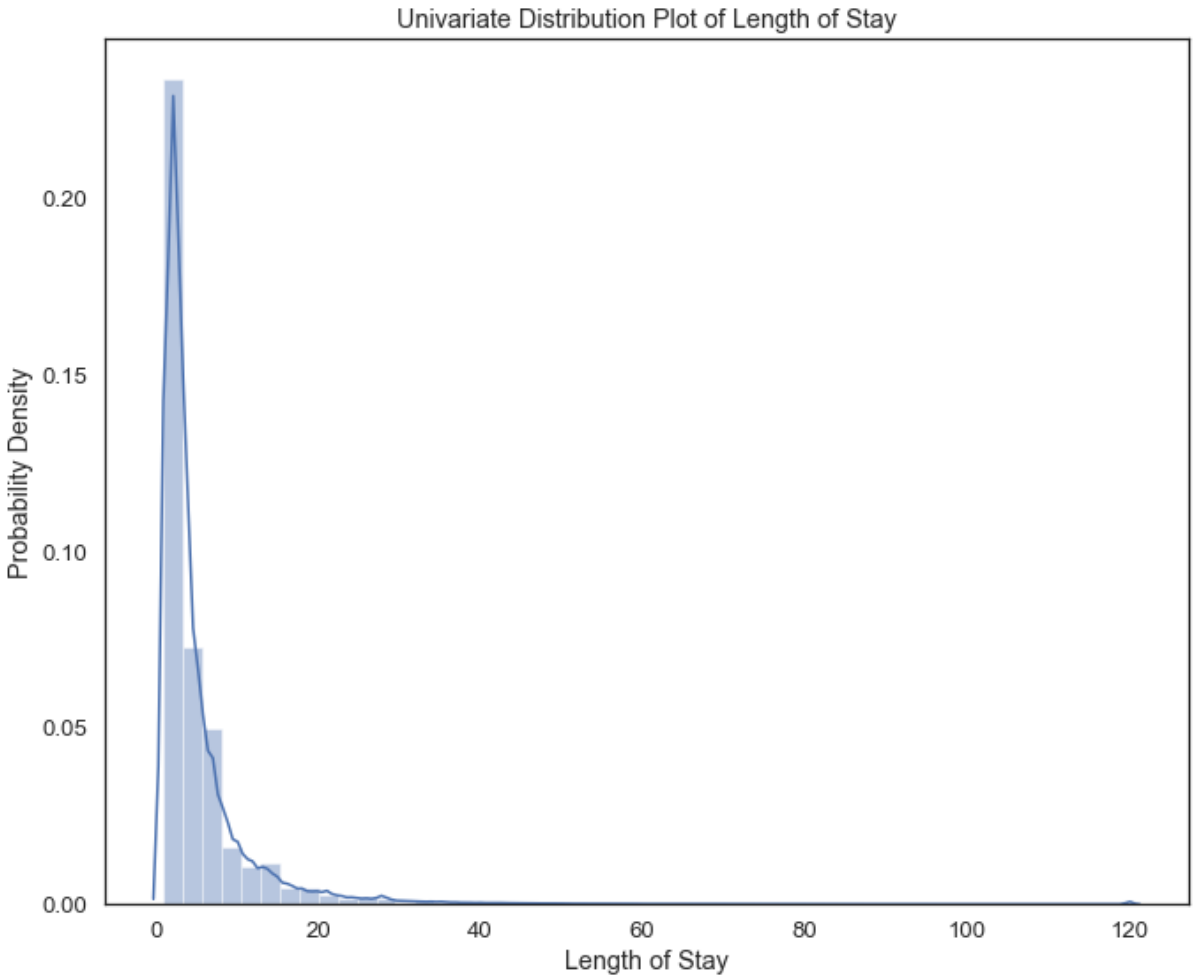


It is evident from this correlation heatmap that some features are strongly correlated with each other and more specifically, with length of stay. These include Total Charges, Total Costs and Severity of Illness code.

Data Science process

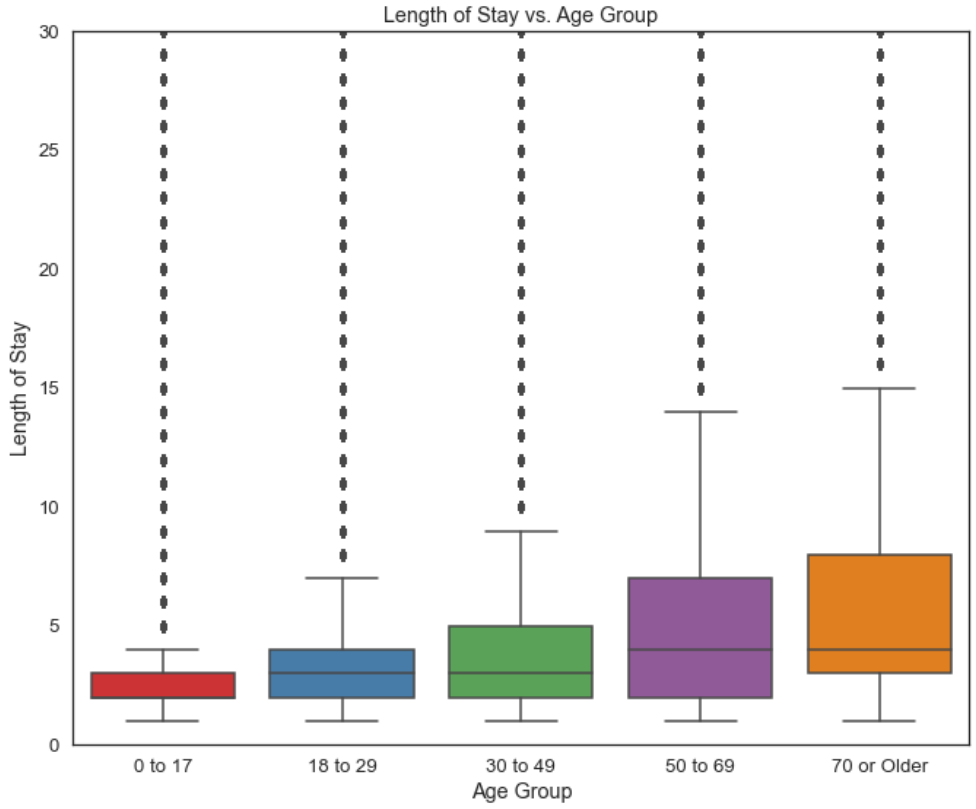
Exploratory Data Analysis

Let's now explore and visualise relationships within the data. **The below plot visualises the univariate (observations on a single attribute) distribution for length of stay**.



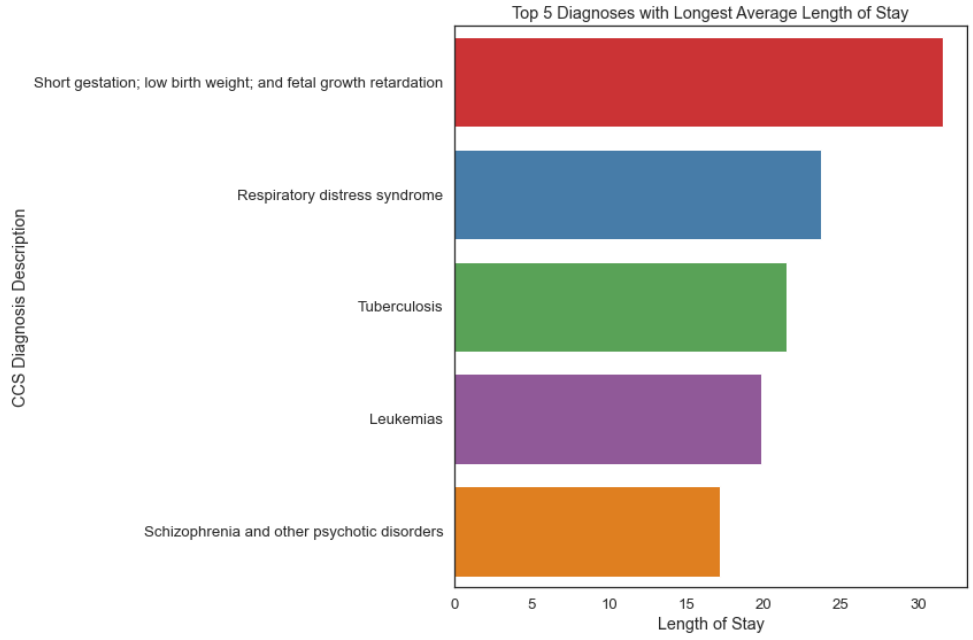
We can see that Length of Stay values range from 1 to 120+ days (the maximum shown is 120 as I have aggregated the 120 number to include lengths of stay longer than 120 days). Additionally, the distribution is very skewed with most of the patients having lengths of stay between 0 to 5 days.

Below we can see a boxplot visualising how different age groups vary in their length of stay distributions.



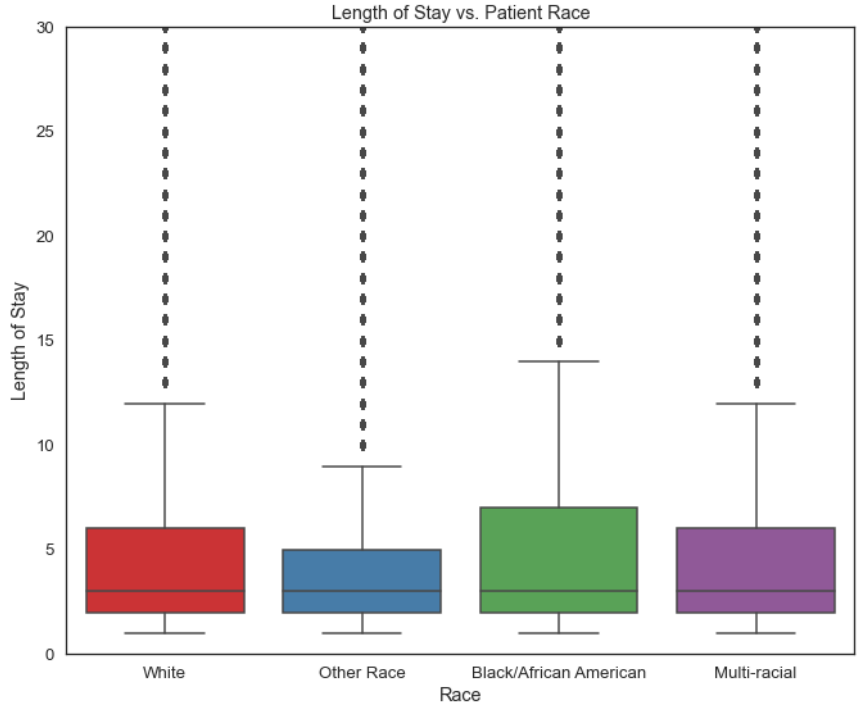
We can clearly see from this plot that as age increases, the length of stay distribution also increases.

Which diagnoses have the longest average length of stay?



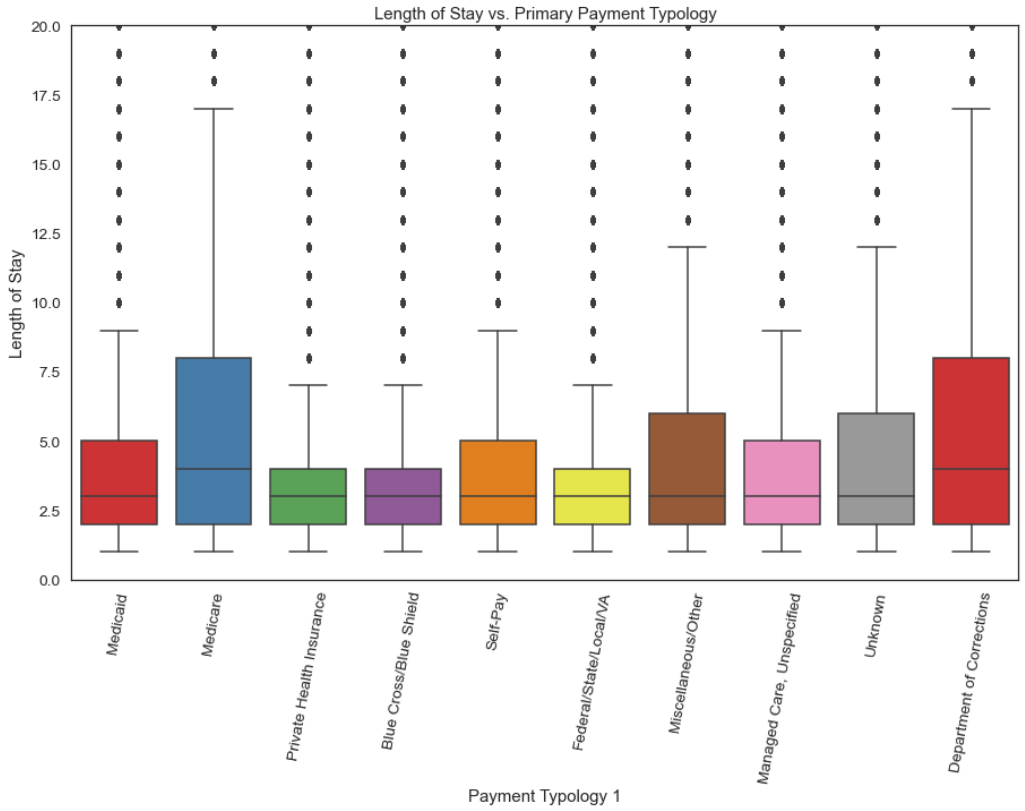
Patients with diagnoses related to birth complications tend to have the longest length of stay on average. This is followed by patients with respiratory diseases.

Let's look at a boxplot visualising length of stay against patient race.



A patient's race does not seem to result in any significant different in distribution of length of stay.

How does length of stay vary with various patient payment types?

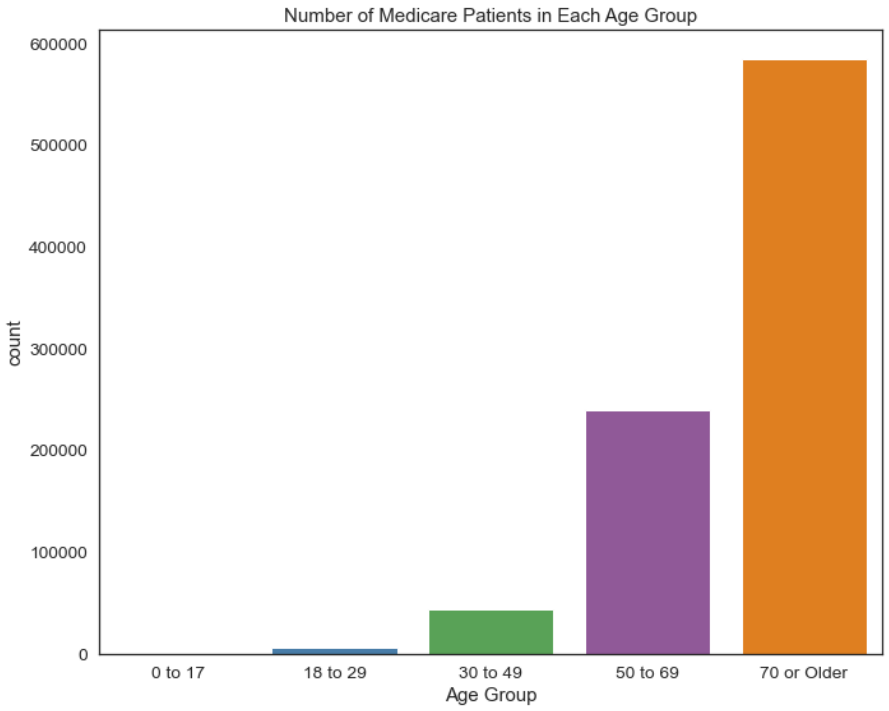


It's evident from this plot that different health insurance forms of payment tend to have different length of stay distributions. These payment typology terms are named specifically for people using

the health system of the United States so aren't directly comparable to the Australian health system. Medicare in the USA, for example, is a government national health insurance program for people aged 65 and over. Some people under 65 can qualify providing they have certain disabilities or terminal illnesses.

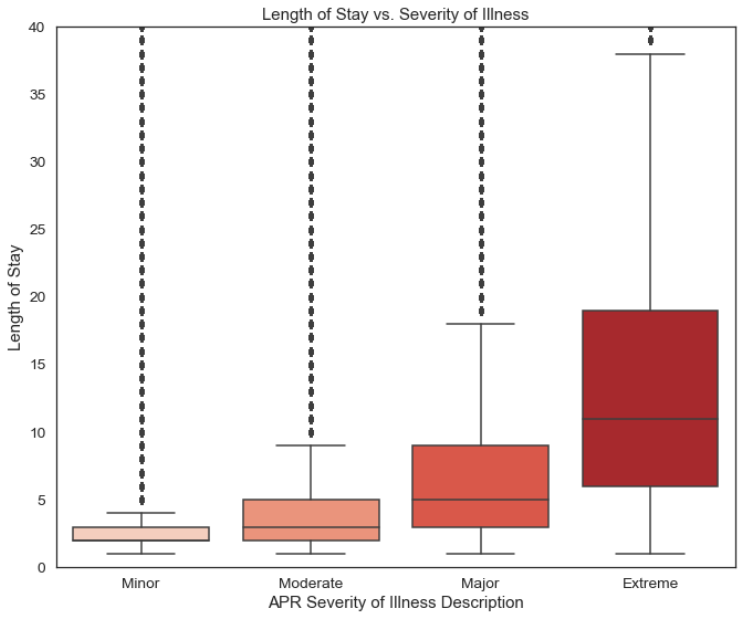
We can see from the plot above that Medicare patients tend to have longer stays – presumably because they're in an older age bracket.

Let's look at the Medicare age distribution.



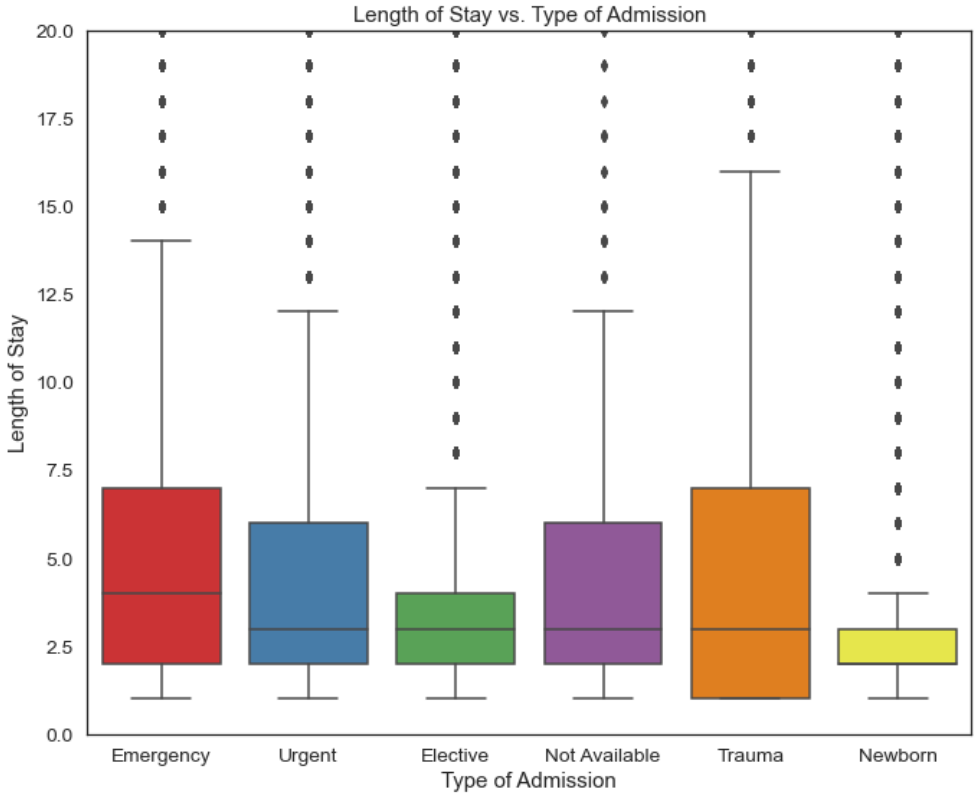
This plot confirms the fact that Medicare patients are predominantly in the older age brackets – correlating with a longer length of stay in hospital.

How much does length of stay vary with the severity of the illness?



We can see there's a significant variance within the APR Severity of Illness feature (Extreme illnesses lead to longer lengths of stay. APR stands for All Patient Refined and is a USA Health system which classifies patients according to severity of illness among other factors.

How does length of stay vary with type of admission?



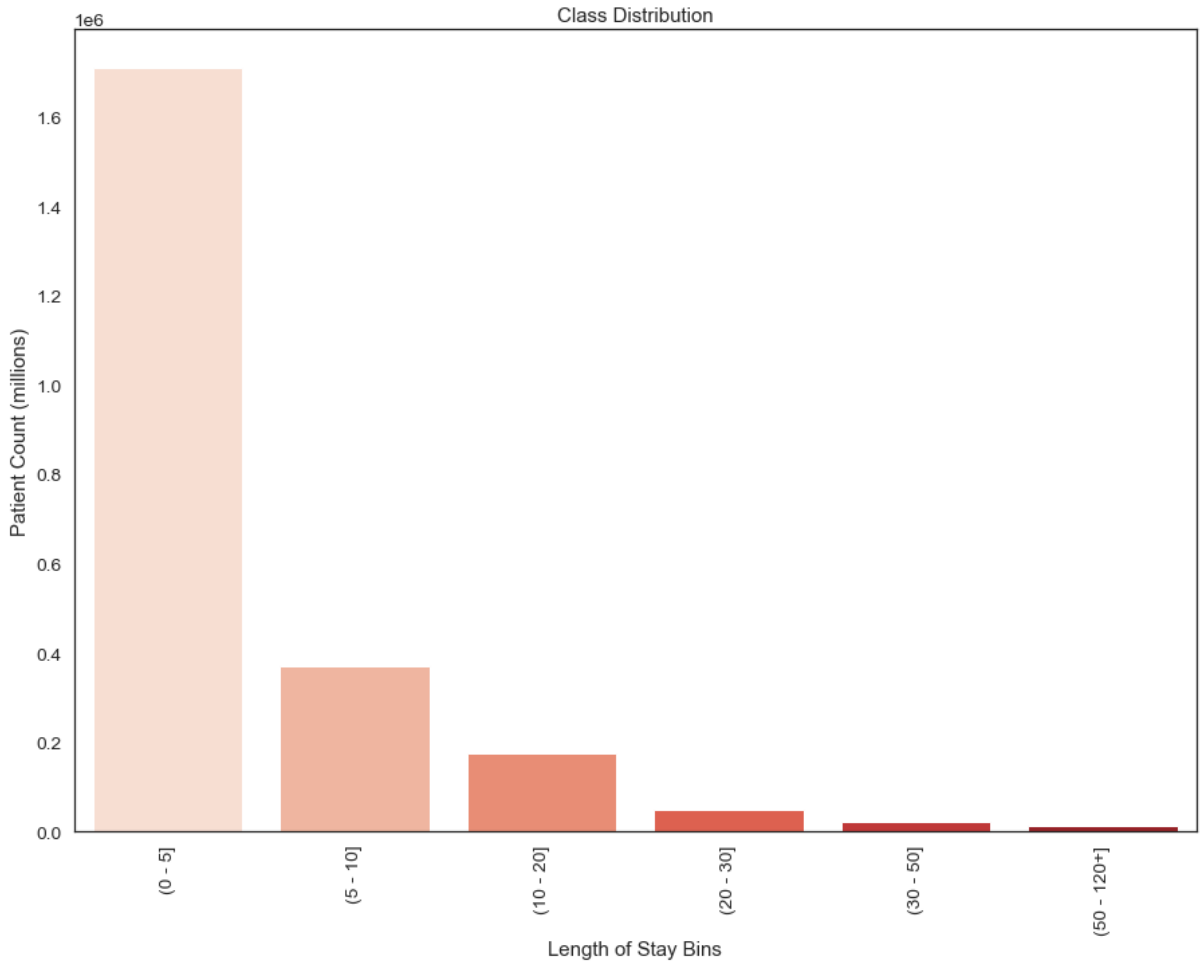
We can see from this plot that the Emergency and Trauma distributions contain the longest length of stay.

Model Preparation

In this section I will be preparing a model to predict length of stay. I will begin by dropping all columns that aren't useful for predictive modelling, they include Zip Code, CCS Diagnosis Description, Operating Certificate Number among others. I'll also drop the Total Costs and Total Charges columns as they would not be present at the time of patient admittance.

I performed feature encoding of all categorical columns containing strings. I also used string indexing for particular subsets of some features. E.g. Age Group column contains '0 to 17' and '70 or older' – I will encode that data to 0 and 5 respectively. One-hot encoding will be performed on the remaining categorical features.

Looking at our predictor feature – length of stay – the integer values range from 1 to 120. I decided to treat the prediction as multi-class classification instead of regression. Instead of treating length of stay as having 120 different classes, I group these values into larger bins which makes more sense for predicting (without a significant loss in accuracy). The following bin format for days was chosen: 1-5, 6-10, 11-20, 21-30, 31-50, 50-120+.



A large class imbalance is apparent looking at the above plot. The imbalanced data must be handled carefully as it can lead to misleading accuracy scores.

Dealing with class imbalance:

I found that under-sampling was less effective than assigning penalties when dealing with the class imbalance – which resulted in higher model accuracies. I used the class weight parameter in scikit

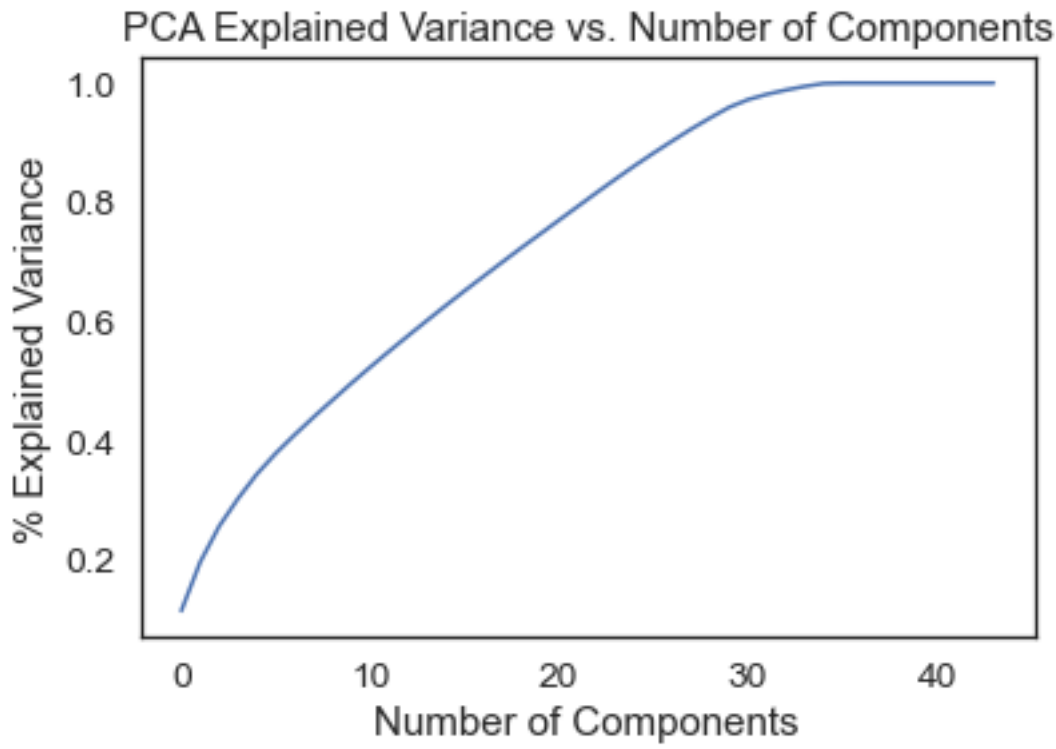
learn and set the parameter to balanced. This effectively assigns a weight to each class inversely proportional to the frequency with which it appears.

Modelling

PCA (Principal Component Analysis)

Prior to training models, I perform PCA on data after using the StandardScaler function to normalize the train and test data sets.

Below I visualise the percentage of explained variance against the number of components from PCA that are used. As shown, 29 components are the minimum number of components required to explain 95% of the variance in the data, so that is the number I keep.



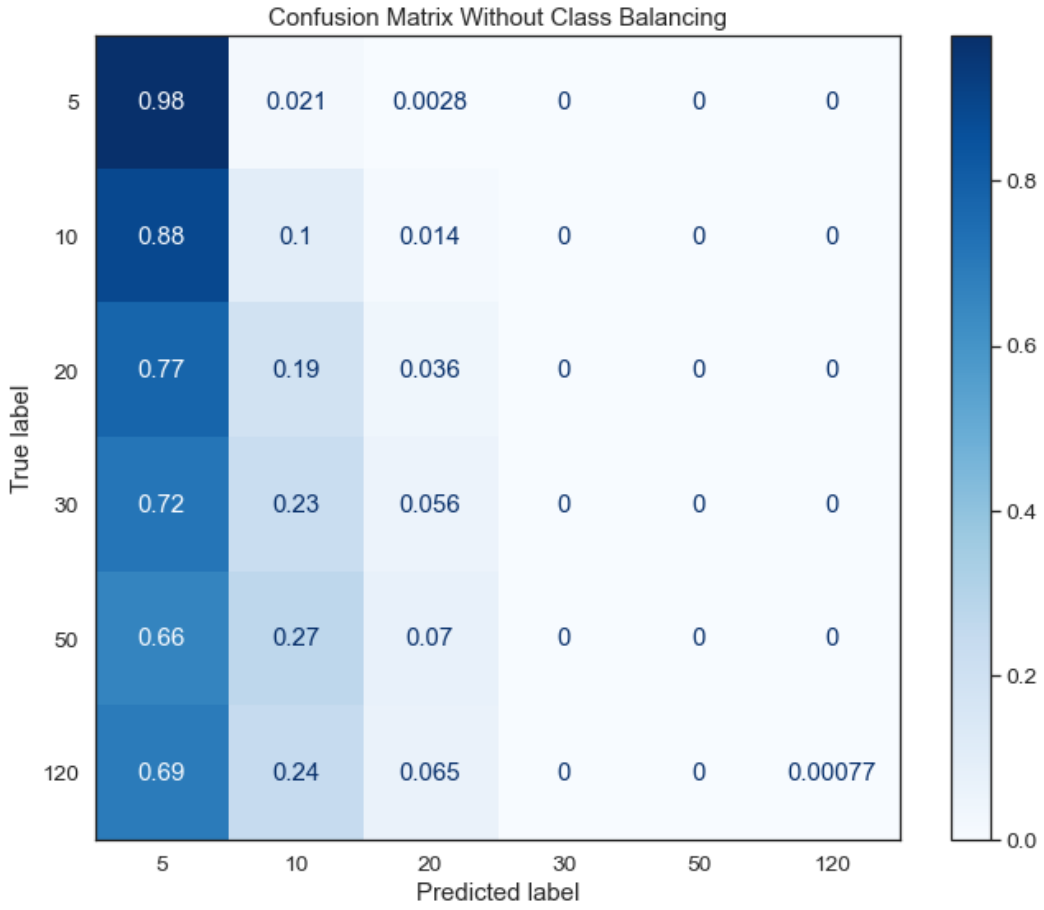
Logistic Regression

Training a model after applying PCA to the standardized data as shown below:

```
pca = PCA(n_components=29)
x_train = pca.fit_transform(x_train)
x_test = pca.transform(x_test)

log_reg = LogisticRegression(multi_class='ovr').fit(x_train, y_train)
y_train_pred = log_reg.predict(x_train)
y_pred = log_reg.predict(x_test)

test_acc = accuracy_score(y_test, y_pred)
train_acc = accuracy_score(y_train, y_train_pred)

print('Test accuracy:', test_acc)
print('Train accuracy:', train_acc)
```

```
Test accuracy: 0.7313748902396157
Train accuracy: 0.7331396221167464
```

Good results are produced here however this model was trained without balancing any class weights. We check out the confusion matrix below:



We can see the model is significantly over predicting on label 5 as it's the label with the highest frequency.

Logistisic Regression with balanced class weight parameter:

```
log_reg = LogisticRegression(class_weight='balanced', multi_class='ovr').fit(x_train, y_train)
y_train_pred = log_reg.predict(x_train)
y_pred = log_reg.predict(x_test)

test_acc = accuracy_score(y_test, y_pred)
train_acc = accuracy_score(y_train, y_train_pred)

print('Test accuracy:', test_acc)
print('Train accuracy:', train_acc)
```

```
Test accuracy: 0.5881790378657316
Train accuracy: 0.5895639911829681
```

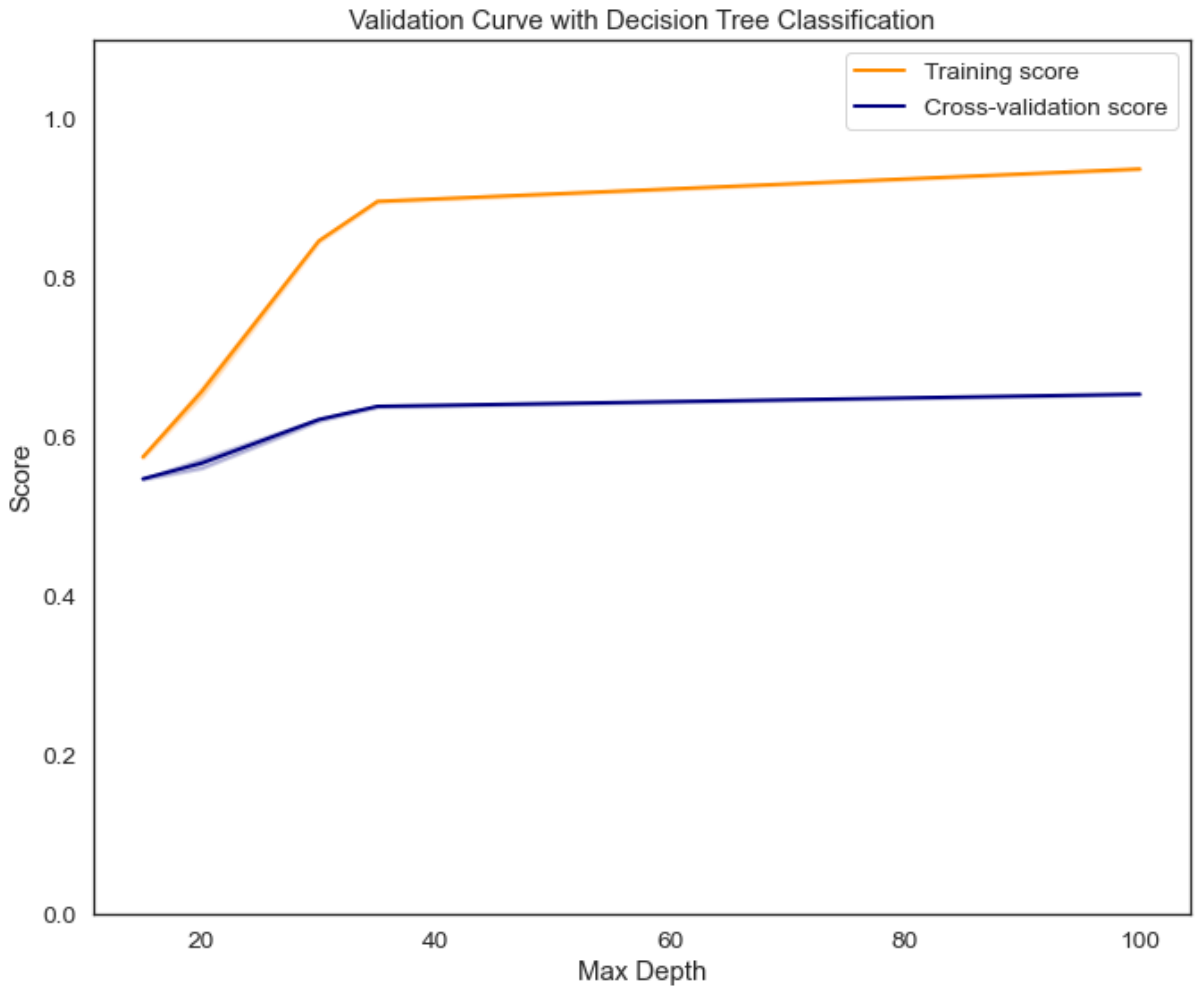
We can see that defining the class weight parameter prevented the classifier from over predicting the highest frequency classes. Even though the overall accuracy dropped, our model is more realistic.

Let's look at some classification metrics, particularly f1-score. We can compare this score with the results of other models.

	precision	recall	f1-score	support
5	0.89	0.70	0.78	514073
10	0.28	0.35	0.31	110799
20	0.18	0.15	0.16	52600
30	0.07	0.21	0.11	14490
50	0.04	0.12	0.06	7057
120	0.03	0.35	0.05	3648
accuracy			0.59	702667
macro avg	0.25	0.32	0.25	702667
weighted avg	0.71	0.59	0.64	702667

Decision Trees:

Here's a validation curve utilising 3-fold cross validation to show the effect of hyperparameter optimization.



Here we see an improvement on model accuracy when implementing these parameters.

```
dtree=DecisionTreeClassifier(max_depth= 50, max_leaf_nodes=1000, class_weight='balanced')
dtree.fit(X_train,y_train)

train_predictions = dtree.predict(X_train)
test_predictions = dtree.predict(X_test)
print("Train Accuracy:",metrics.accuracy_score(y_train, train_predictions))
print("Test Accuracy:",metrics.accuracy_score(y_test, test_predictions))
```

```
Train Accuracy: 0.6218502104840706
Test Accuracy: 0.6190386057691624
```

	precision	recall	f1-score	support
5	0.93	0.71	0.81	514073
10	0.28	0.41	0.33	110799
20	0.23	0.28	0.25	52600
30	0.15	0.26	0.19	14490
50	0.08	0.32	0.12	7057
120	0.09	0.56	0.15	3648
accuracy			0.62	702667
macro avg	0.29	0.42	0.31	702667
weighted avg	0.74	0.62	0.67	702667

We can see accuracy has improved with decision tree classifier as well as F1 scores. Let's try Random Forest now.

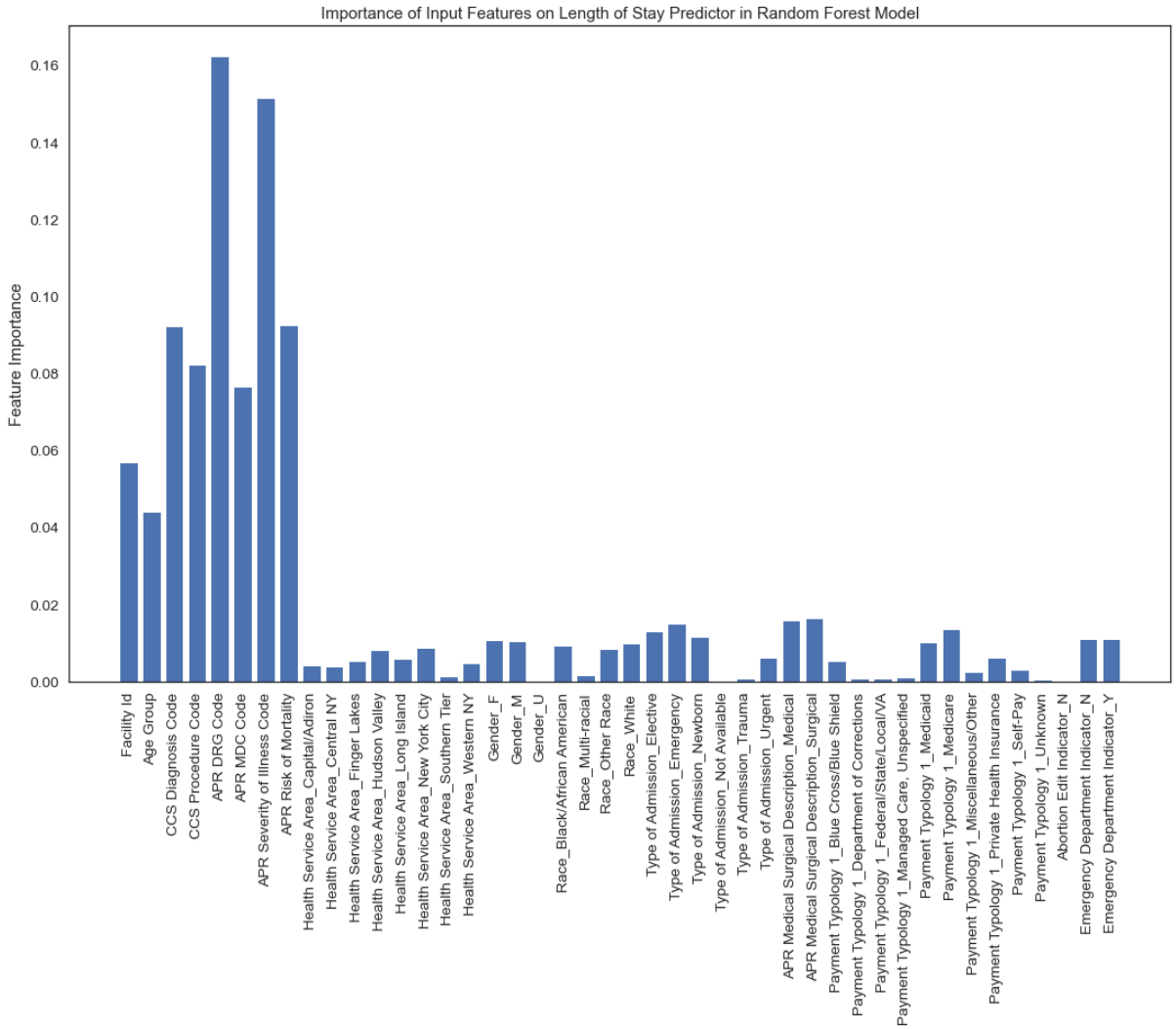
Random Forest:

Model parameters were chosen by trial and error as grid/randomized search was very slow to execute.

```
rf = RandomForestClassifier(n_estimators=150, max_depth=15, class_weight='balanced')
rf.fit(X_train,y_train)

train_predictions = rf.predict(X_train)
test_predictions = rf.predict(X_test)
print("Train Accuracy:",metrics.accuracy_score(y_train, train_predictions))
print("Test Accuracy:",metrics.accuracy_score(y_test, test_predictions))
```

```
Train Accuracy: 0.6734593676085082
Test Accuracy: 0.645675689907168
```

	precision	recall	f1-score	support
5	0.92	0.73	0.82	514073
10	0.29	0.48	0.37	110799
20	0.28	0.29	0.28	52600
30	0.18	0.25	0.21	14490
50	0.10	0.30	0.15	7057
120	0.12	0.53	0.19	3648
accuracy			0.65	702667
macro avg	0.32	0.43	0.34	702667
weighted avg	0.75	0.65	0.68	702667


Importance of Input Features on Length of Stay Predictor in Random Forest Model
Feature Importance
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
Facility Id
Age Group
CCS Diagnosis Code
CCS Procedure Code
APR DRG Code
APR MDC Code
APR Severity of Illness Code
APR Risk of Mortality
Health Service Area_Capital/Adiron
Health Service Area_Central NY
Health Service Area_Finger Lakes
Health Service Area_Hudson Valley
Health Service Area_Long Island
Health Service Area_New York City
Health Service Area_Southern Tier
Health Service Area_Western NY
Gender_F
Gender_M
Gender_U
Race_Black/African American
Race_Multi-racial
Race_Other Race
Race_White
Type of Admission_Elective
Type of Admission_Emergency
Type of Admission_Newborn
Type of Admission_Not Available
Type of Admission_Trauma
Type of Admission_Urgent
APR Medical Surgical Description_Medical
APR Medical Surgical Description_Surgical
Payment Typology 1_Blue Cross/Blue Shield
Payment Typology 1_Department of Corrections
Payment Typology 1_Federal/State/Local/VA
Payment Typology 1_Managed Care, Unspecified
Payment Typology 1_Medicaid
Payment Typology 1_Medicare
Payment Typology 1_Miscellaneous/Other
Payment Typology 1_Private Health Insurance
Payment Typology 1_Self-Pay
Payment Typology 1_Unknown
Abortion Edit Indicator_N
Emergency Department Indicator_N
Emergency Department Indicator_Y

Conclusion:

I was ultimately able to predict a patient's length of stay using data supplied upon the moment a patient is diagnosed with an accuracy of around 67%. This model could improve hospital management however further improvements could be made. Additional machine learning algorithms could be explored. Future models could also look at the cost associated with a certain length of stay.

In the end, as seen in the above graph, APR DRG Codes and APR Severity of Illness Codes (e.g. Minor, Moderate, Major, etc) are the two most important features in predicting a patient's length of stay. The Medicare subgroup of Payment Typology also has a relatively high importance in predicting the length of stay – when comparing to other payment typology groups.

References

https://www.kaggle.com/datasets/jonasalmeida/2015-deidentified-ny-inpatient-discharge-sparcs

https://health.data.ny.gov/Health/Hospital-Inpatient-Discharges-SPARCS-De-Identified/82xm-y6g8

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5898738/

https://chat.openai.com/

https://datascience.stackexchange.com/questions/76253/validation-curve-interpretation

https://www.bhi.nsw.gov.au/nsw_patient_survey_program/adult_admitted_patient_survey

https://www.aihw.gov.au/reports/hospitals/australias-hospitals-at-a-glance/contents/spending-on-hospitals

https://www.aihw.gov.au/reports-data/myhospitals