Model Generalization

Dataset Description

The dataset which we have used in our model building comprises of almost 4.9 lakhs of record, it is based upon a medical survey which has been conducted in USA.

It comprises of total 17 feature variables and 1 target variable as follows:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 490000 entries, 0 to 489999
Data columns (total 18 columns):
 #   Column                              Non-Null Count   Dtype
---  ------                              --------------   -----
 0   Age Group                           490000 non-null  object
 1   Gender                              490000 non-null  object
 2   Length of Stay                      490000 non-null  int64
 3   Type of Admission                   490000 non-null  object
 4   Patient Disposition                 490000 non-null  object
 5   CCS Diagnosis Description           490000 non-null  object
 6   CCS Procedure Description           490000 non-null  object
 7   APR DRG Description                 490000 non-null  object
 8   APR MDC Description                 490000 non-null  object
 9   APR Severity of Illness Description 490000 non-null  object
 10  APR Risk of Mortality               490000 non-null  object
 11  APR Medical Surgical Description    490000 non-null  object
 12  Payment Typology 1                  490000 non-null  object
 13  Payment Typology 2                  315687 non-null  object
 14  Payment Typology 3                  136615 non-null  object
 15  Birth Weight                        490000 non-null  int64
 16  Emergency Department Indicator      490000 non-null  object
 17  Cost INR                            490000 non-null  float64
dtypes: float64(1), int64(2), object(15)
memory usage: 67.3+ MB
```

Generalized Model

After implementing various regression models, we have considered ***'Random Forest Regression'*** as the best fitted model for our dataset.

In the initial step we have split the data in Training and Testing set, then we have performed fitting and prediction over the regression model as follows:

I. Creation of regression model after performing hyper parameter tunning and training the regressor over the training set.

5. Random Forest Regressor

[61]:
```
rf_model = RandomForestRegressor(n_estimators=100,max_depth=6, random_state=42)
rf_model.fit(X_train, y_train)
```

[61]:
```
RandomForestRegressor(max_depth=6, random_state=42)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

RandomForestRegressor?Documentation for RandomForestRegressoriFitted
```
RandomForestRegressor(max_depth=6, random_state=42)
```

II. Performing prediction using the test dataset and evaluation of the test score.

```
[62]: y_pred_rf=rf_model.predict(X_test)
      r2_score(y_test,y_pred_rf)

[62]: 0.6327492122615432
```

III. Checking the over-fitting of the regression model.

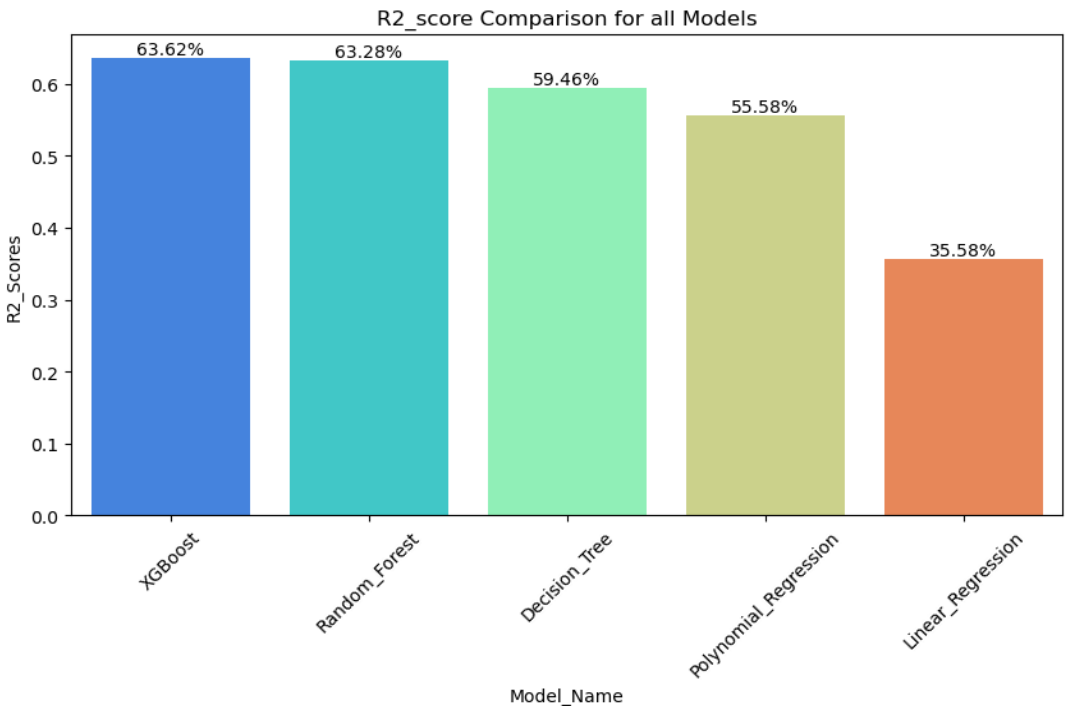
>>> Check for Over-fitting

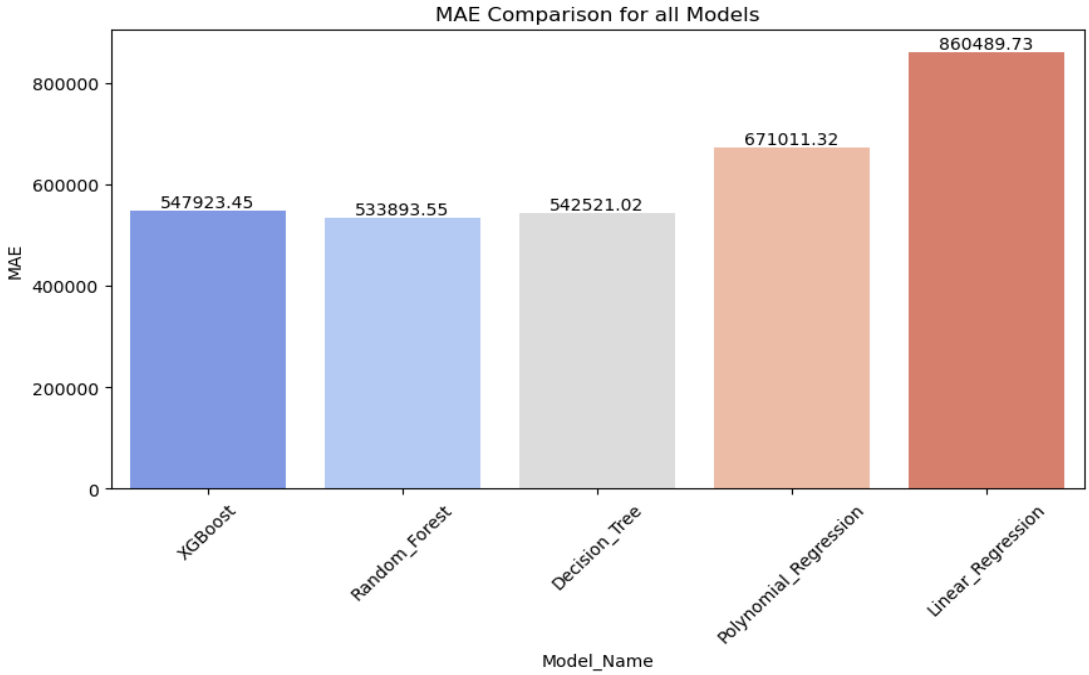
```
[64]: cv_scores_rf = cross_validate(rf_model, X_train, y_train, cv=10, scoring="r2",return_train_score=True)

[65]: print("Test>>>","Min:",cv_scores_rf.get("test_score").min(), "Max:",cv_scores_rf.get("test_score").max())
      print("="*60)
      print("Train>>>","Min:",cv_scores_rf.get("train_score").min(), "Max:",cv_scores_rf.get("train_score").max())

      Test>>> Min: 0.5985728313692291 Max: 0.6843897369638008
      ============================================================
      Train>>> Min: 0.7102635832993396 Max: 0.7148813780915088
```

Why this model?

We have performed predictions using various regression model such as XGBoost, Random Forest, Decision Tree, Polynomial Regression and Linear Regression.

Among them all **Random Forest** comes out to be the best as compare to other models it has the best prediction *accuracy score, least mean absolute error* and the issue of getting *overfit is slightly as less* compared other regression model.





Conclusion

After considering all the regression models and comparing their accuracy score, mean absolute error, Prediction Vs actual plots and residual plots we have come to the conclusion of using **Random Forest as our final Prediction Model.**

After going through multiple iterations of Hyper-parameter tuning, we have reached to this accuracy score for **Random Forest Regressor (around 63%).**