Health: Analyzing the New York State of Health

Team

Jayson Xu, TorresZ, Chloe, Esther

Introduction

In this project, we aim to predict the mean cost of hospital discharges based on features such as hospital name, APR DRG code, severity of illness, year, and medical/surgical classification. By accurately predicting discharge costs, hospitals can better allocate resources, plan budgets, and develop targeted intervention strategies, especially for high-cost patient groups.

Questions Answered

- **Data Analysis and Visualization**
  - Trends in total discharges over time
  - Top facilities with the highest number of discharges
  - Average cost trends over the years
  - Distribution of illness severity among patients
  - Correlation between actual vs predicted mean costs

- **Machine Learning**
  - **Predict the mean cost** of a discharge using features like:
    - Hospital
    - APR DRG code
    - Severity of illness
    - Year
    - Medical/Surgical classification.

Answers

- **Dataset**

  Hospital inpatient discharges SPARCS de-identified cost transparency beginning 2009 20250426.csv: This dataset includes The New York State Department of Health invites you to dive into a real-world healthcare dataset collected through the Statewide Planning and Research Cooperative System (SPARCS).

  - Source: SPARCS Hospital Inpatient Discharges (De-Identified)
  - Fields: Year, Facility ID, Facility Name, APR DRG Code, Severity, Discharges, Mean Cost, Median Cost, etc.
  - Data Size: 1.2 million+ records

- **Importing the dataset**

  The datasets were imported as pandas and numpy

```
import pandas as pd
import numpy as np


df = pd.read_csv('/content/Hospital_Inpatient_Discharges__SPARCS_De-Identified___Cost_Transparency__Beginning_2009_20250426.csv')
# data information
df.info()


df.head()
```

- **Check for Duplication and missing values**

```
# check duplication
```

```
df.duplicated().any()

# Check missing values

df.isnull().sum()
```

After screening, we noticed that there was no duplication in the dataset given and that most columns don’t have missing values, but “APR Severity of Illness Description” and “APR Medical Surgical Code” have 211 and 343 missing values. That is because they contain strings which we ignored. It was also observed that “Year”, “Facility ID”, “APR DRG Code” and “APR Severity of Illness Code” aree in Integer.

- **Data Cleaning and Storing of Cleaned Data**
  - Basic cleaning:

In order to make sure that mean charge, median charge, mean cost and median cost are data with clean numbers, we need to clean up the data inside.

```
cost_columns = ['Mean Charge', 'Median Charge', 'Mean Cost',
'Median Cost']


for col in cost_columns:

   df[col] = pd.to_numeric(df[col], errors='coerce')
```

- Missing Values: Minor missingness (~0.02%) in severity fields, handled via dropping.
- Type Conversion: Converted key fields (Year, Discharges, Cost/Charge fields) to numeric.
- Outlier Removal: Used IQR method to clean extreme cost/charge values.
- Final Shape: After cleaning, ~1 million records retained.

Afterwards, we stored the cleaned data for further use.

```
# storing the new data
df.to_csv('cleaned_data.csv', index=False)
```

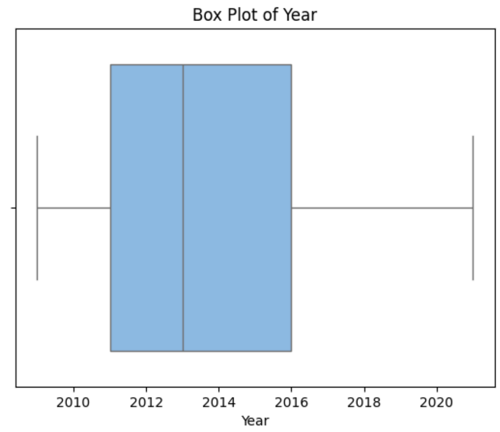
Then, we turned Year and Discharges into Numerical Data

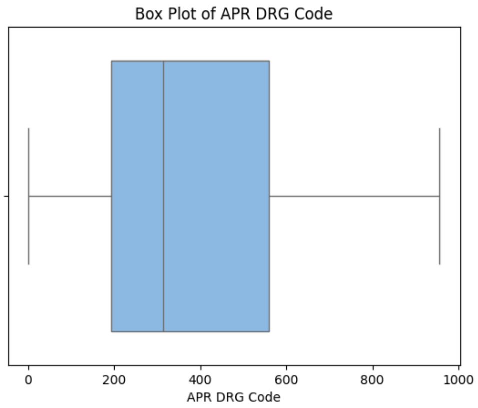
```
df['Year'] = pd.to_numeric(df['Year'], errors='coerce')
df['Discharges'] = pd.to_numeric(df['Discharges'], errors='coerce')
print(df['Year'].dtype)
print(df['Discharges'].dtype)
df.info()
```

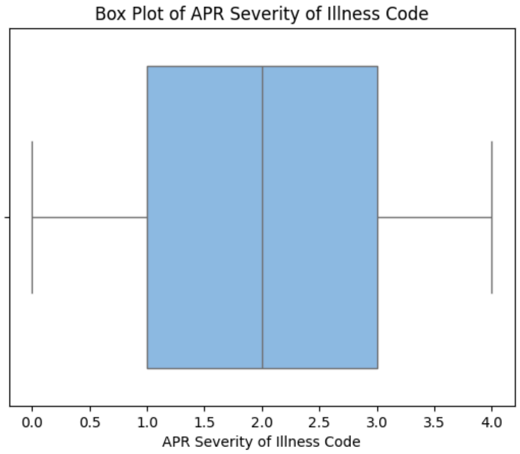
- Further Exploration:

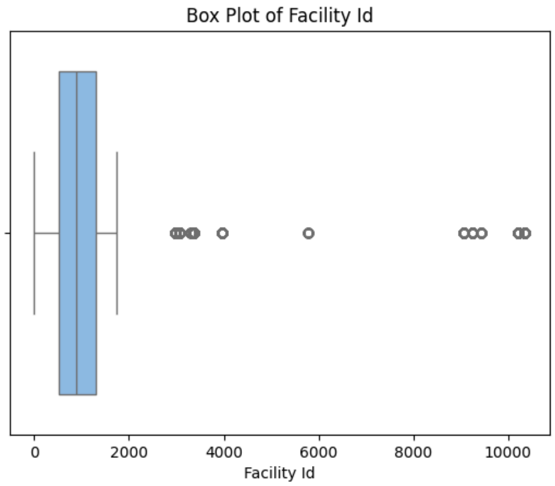
The first thing that we did was to check the distribution to see the outliers

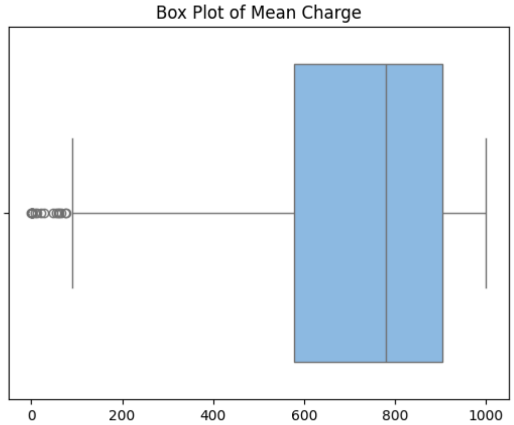
```
Import matplotlib.pyplot as plt
import seaborn as sns
numeric_cols = df.select_dtypes(include=np.number).columns.tolist()
for col in numeric_cols:
   sns.boxplot(x=df[col], color = '#81BBF3')
   plt.title(f'Box Plot of {col}')

   plt.show()
```

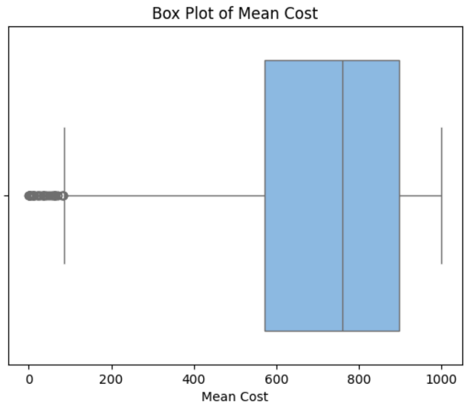

Box Plot of Year

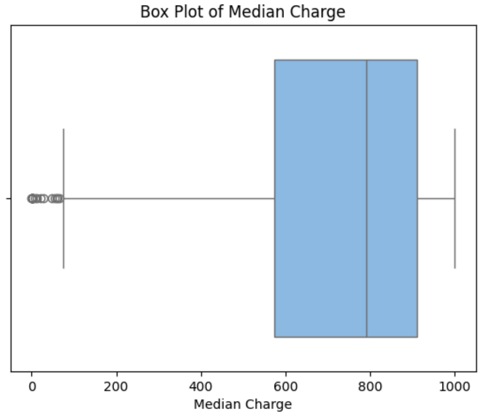

Box Plot of APR DRG Code

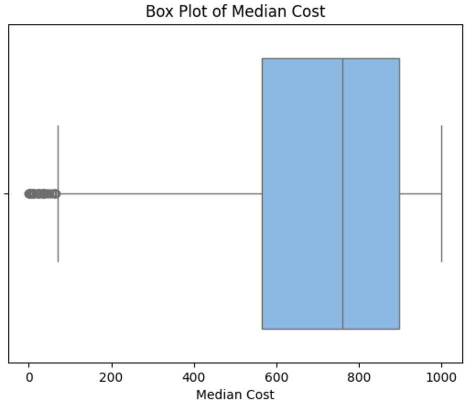

Box Plot of APR Severity of Illness Code


Box Plot of Facility Id


Box Plot of Mean Charge


Box Plot of Mean Cost


Box Plot of Median Charge

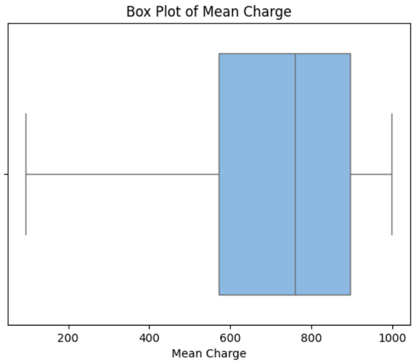

Box Plot of Median Cost

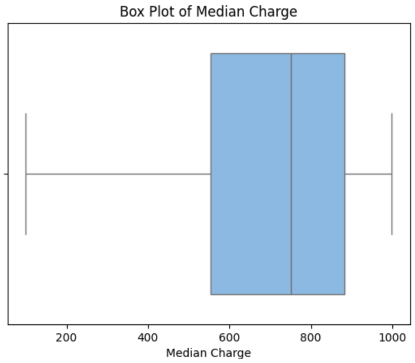
From the above, we observed that these box graphs which include 'Mean Charge' 'Median Charge', 'Mean Cost' and 'Median Cost' all have outliers. So we would want to clean those outliers.

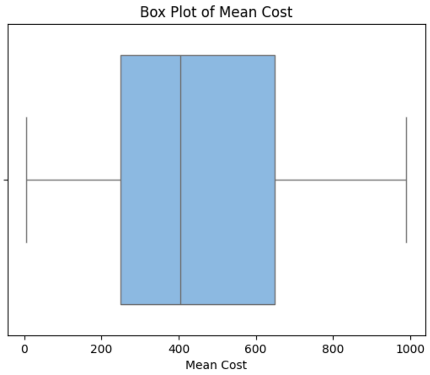
```
# Clean for outliers

for coln in cost_columns:

  Q1 = df[coln].quantile(0.25)

  Q3 = df[coln].quantile(0.75)

  IQR = Q3 - Q1

  lower_bound = Q1 - 1.5 * IQR

  upper_bound = Q3 + 1.5 * IQR

  df = df[(df[coln] >= lower_bound) & (df[coln] <= upper_bound)]
```

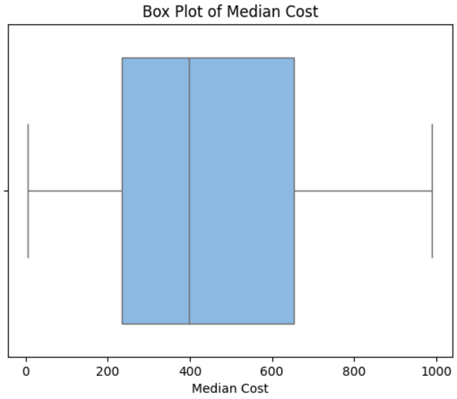
- **Cleaned result**

```
for coln in colns:

    sns.boxplot(x=df[coln], color = '#81BBFData Visualization Insights

1. Total Discharges Over Time

Finding: Overall decreasing trend from 2009 to 2021.



2. Top 10 Hospitals by Discharges

Finding: Massena Memorial and Aurelia O. Fox Memorial dominate the discharge volume.



3. Average Mean Cost Over Years

Finding: Gradual decrease in average cost, with some fluctuations around 2013-2015.



4. Severity of Illness Distribution
Finding: ~50% patients are minor cases, ~17% are extreme severity cases.



5. Cost vs Charge Correlation

Finding: Positive correlation between mean cost and mean charge.

3')

    plt.title(f'Box Plot of {coln}')

    plt.show()
```









We see that there are now no outliers in the graph.

Data Visualization Insights

- Total Discharges Over Time
  - Finding: Overall decreasing trend from 2009 to 2021.

- Top 10 Hospitals by Discharges
  - Finding: Massena Memorial and Aurelia O. Fox Memorial dominate the discharge volume.

- Average Mean Cost Over Years
  - Finding: Gradual decrease in average cost, with some fluctuations around 2013-2015.

- Severity of Illness Distribution
  - Finding: ~50% patients are minor cases, ~17% are extreme severity cases.

- Cost vs Charge Correlation
  - Finding: Positive correlation between mean cost and mean charge.

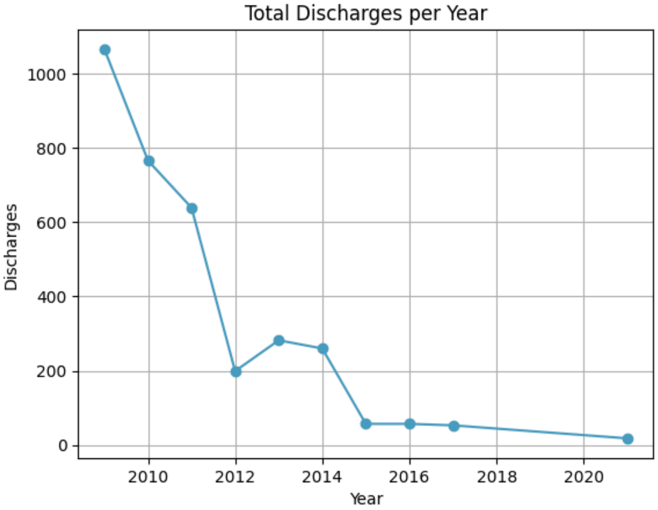
- **Data Visualization**
  - **Total discharges over time**

Our first inquiry is to see whether total discharges change through the years.

1. We found out the total discharges per year

```
Import matplot as plt
# 1. Total Discharges per Year
yearly_discharges = df.groupby('Year')['Discharges'].sum()
yearly_discharges.plot(kind='line', marker='o', color = '#499BC0')
plt.title('Total Discharges per Year')
plt.ylabel('Discharges')
plt.xlabel('Year')
plt.grid()
plt.show()
```

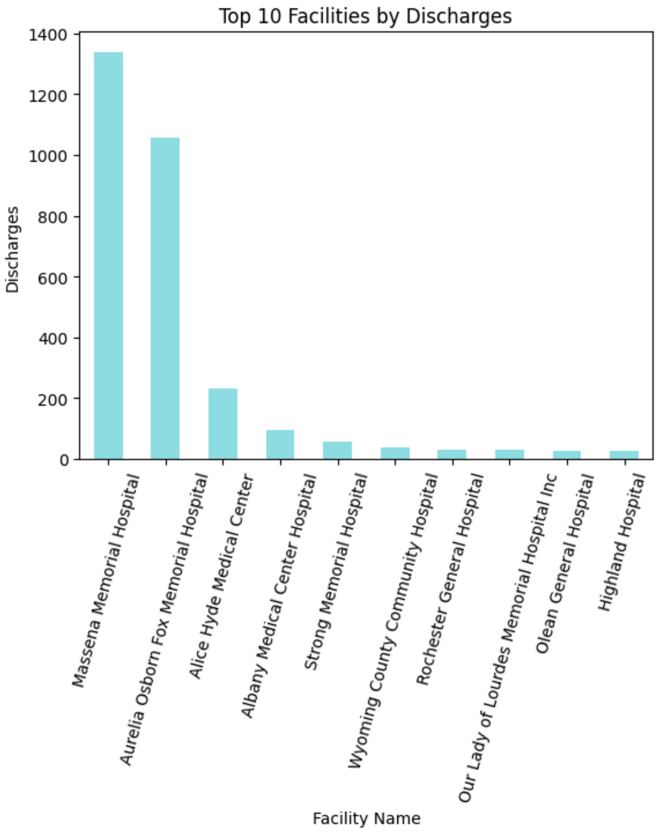
We see an overall decreasing trend of discharges across years.



  - **Top 10 Facilities by discharges**

2. We ranked the top 10 facilities by discharges

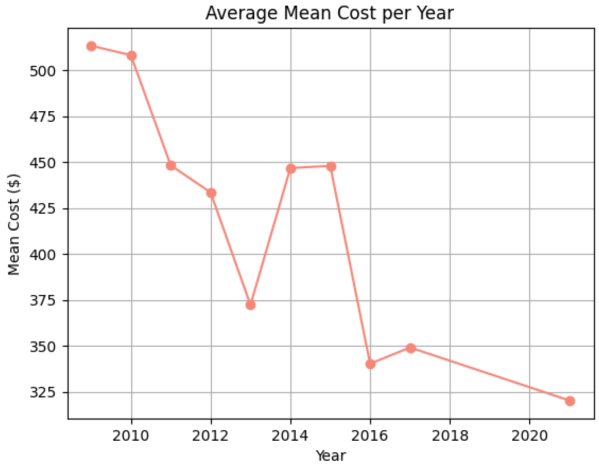
```
# 2. Top 10 Facilities by Discharges
top_facilities = df.groupby('Facility Name')['Discharges'].sum().sort_values(ascending=False).head(10)
top_facilities.plot(kind='bar', color = '#8FDEE3')
plt.title('Top 10 Facilities by Discharges')
plt.ylabel('Discharges')
plt.xticks(rotation=75)
plt.show()
```



- ○ **Average Mean Cost Over Years**

3. We worked on the average mean cost per year and whether it was changing with time

```
# Average mean Cost per Year
yearly_mean_cost = df.groupby('Year')['Mean Cost'].mean()
yearly_mean_cost.plot(kind='line', marker='o', color='#F78779')
plt.title('Average Mean Cost per Year')
plt.ylabel('Mean Cost ($)')
plt.xlabel('Year')
plt.grid()
plt.show()
```

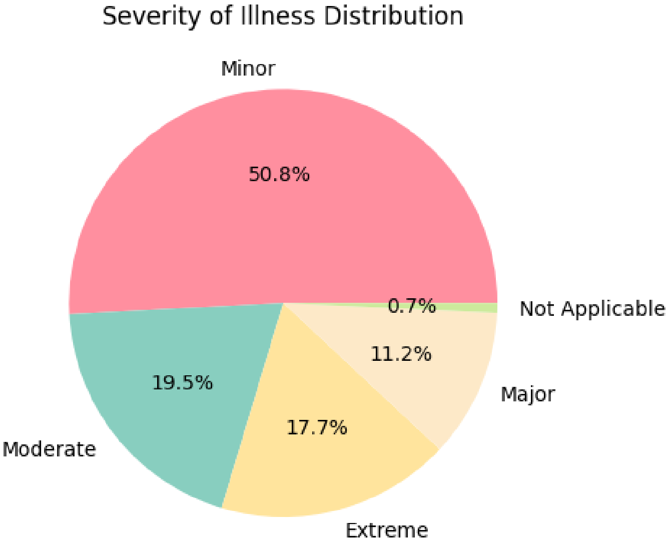


Despite the fluctuation, we see a general decreasing trend of average mean cost across years from 2010 to 2020.

- **Severity of Illness Distribution**

4. To visualize the distribution of severity of illness, we draw a pie chart.

```
# 4. Severity of Illness Distribution
severity_counts = df['APR Severity of Illiness Description'].value_counts()
colors = ['#FF8F9F', '#88CEBF', '#FFE49D', '#FDE9C8','#CCEA9C']
severity_counts.plot(kind='pie', autopct='%1.1f%%', colors = colors)
plt.title('Severity of Illness Distribution')
plt.ylabel('')
plt.show()
```

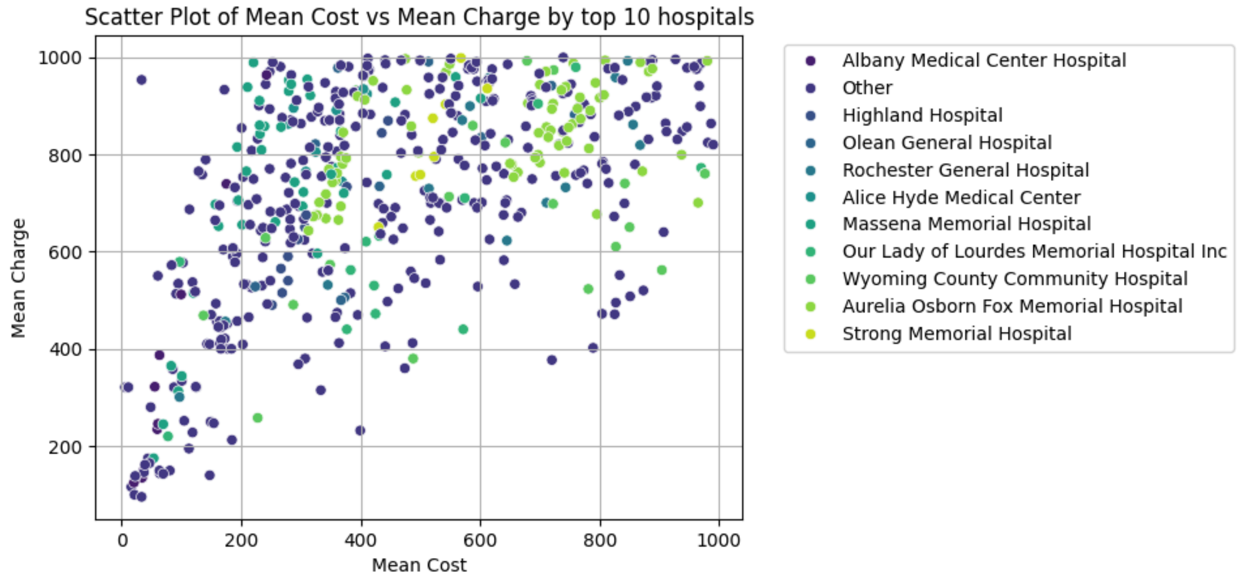


In this chart, we can see that around 50 % injuries are minor, and only 17.7% are extreme.

- **Cost vs Charge Correlation**

6. Lastly, we are curious whether mean cost would have any correlation with mean charge.

```
# 5 Scatter plot correlation between mean charge and cost
df['Top 10 Facility'] = df['Facility Name'].apply(lambda x: x if x in top_facilities.index else 'Other')
sns.scatterplot(
    data = df,
    x = 'Mean Cost',
    y = 'Mean Charge',
    hue = 'Top 10 Facility',
    palette='viridis'
)
plt.legend(bbox_to_anchor(1.05, 1), loc=’upper left’)
plt.title('Scatter Plot of Mean Cost vs Mean Charge by top 10 hospitals')
plt.xlabel('Mean Cost')
plt.ylabel('Mean Charge')
plt.grid(True)
plt.show()
```



This scatter plot shows that the mean cost and mean charge shows positive correlation with each other.

Though, the statistics aren't very strong.

```
np.corrcoef(x, y)

plt.savefig('total_discharges_per_year.png')

# save cleaned csv

df.to_csv('cleaned_data.csv', index=False)
```

Machine Learning

- Goal
  - We aimed to predict the mean cost of a hospital discharge using structured hospital and patient data features.
  - The task was formulated as a supervised regression problem, with the "Mean Cost" per discharge as the target variable.
- Data Preparation
  - Features Used:
  - Facility Name (encoded)
  - APR DRG Code
  - APR Severity of Illness Description (encoded)
  - Year
  - APR Medical/Surgical Classification (encoded)
  - Discharges
  - Cost per Discharge (engineered feature: Mean Cost / Discharges)

Target Variable:

  - Mean Cost

Preprocessing:

  - Removed rows with missing or invalid values.
  - Converted all relevant columns to numeric types.
  - Encoded categorical features as integers.
  - Created new features to improve model learning capacity.

Train/Test Split:

  - 80% training, 20% testing
  - Random state fixed for reproducibility

- **Models Compared**

Model	Mean Squared Error (MSE)	R^2 Score
Linear Regression	15848.72	0.789
Random Forest Regressor	778.14	0.990
XGBoost Regressor	362.90	0.995

  - Linear Regression served as a baseline.
  - Random Forest Regressor significantly reduced error, highlighting non-linear patterns.
  - XGBoost Regressor achieved the best performance with an R^2 score of 0.995, demonstrating excellent predictive power.

- **Model Evaluation**
  - XGBoost Regressor provided the highest accuracy with an R^2 of 0.995.
  - Random Forest Regressor also achieved strong performance, confirming the importance of capturing non-linear interactions.
  - Linear Regression, while simple, underperformed due to the complexity of cost structures in the healthcare system.

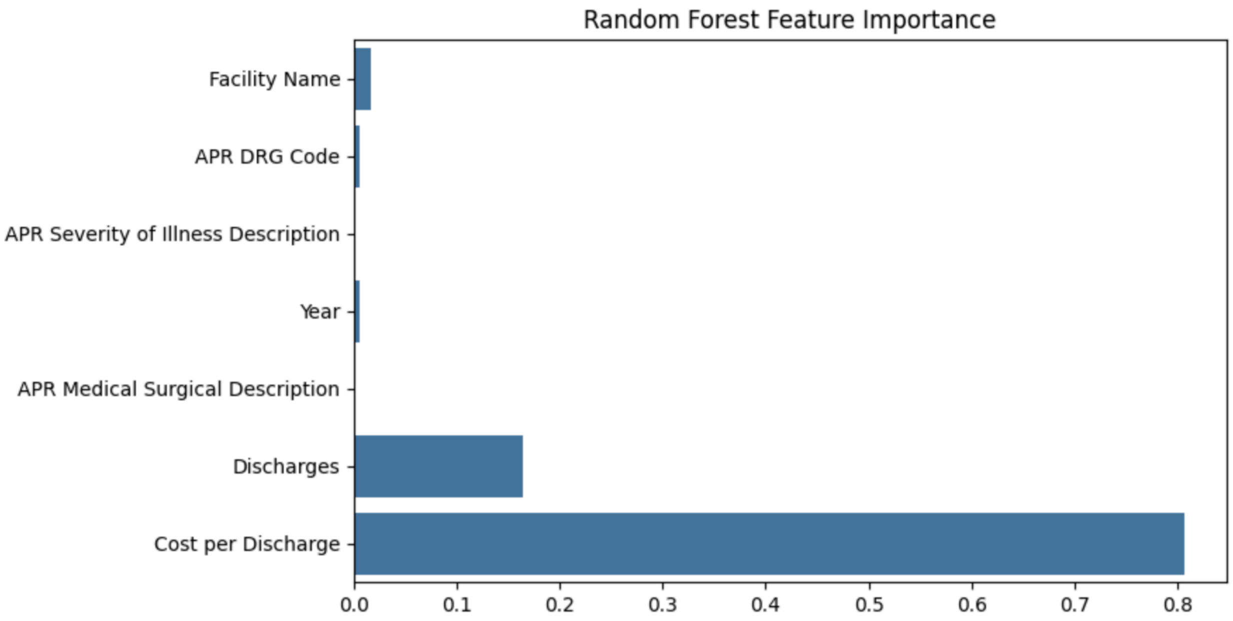
- **Feature Importance Analysis**

```
# 💥 Step 1: Feature Engineering
df['Year'] = pd.to_numeric(df['Year'], errors='coerce')
df['Discharges'] = pd.to_numeric(df['Discharges'], errors='coerce')
df['Mean Cost'] = pd.to_numeric(df['Mean Cost'], errors='coerce')
df['Median Cost'] = pd.to_numeric(df['Median Cost'], errors='coerce')

# Encode categorical variables
df['Facility Name'] = df['Facility Name'].astype('category').cat.codes
df['APR DRG Code'] = pd.to_numeric(df['APR DRG Code'], errors='coerce')
```

```python
df['APR Severity of Illness Description'] = df['APR Severity of Illness Description'].astype('category').cat.codes
df['APR Medical Surgical Description'] = df['APR Medical Surgical Description'].astype('category').cat.codes

# Create new feature
df['Cost per Discharge'] = df['Mean Cost'] / df['Discharges']

# Drop missing
df = df.dropna()

# 💥 Step 2: Features and Label
features = ['Facility Name', 'APR DRG Code', 'APR Severity of Illness Description', 'Year', 'APR Medical Surgical Description', 'Discharges', 'Cost per Discharge']
X = df[features]
y = df['Mean Cost']

# 💥 Step 3: Train/Test Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 💥 Step 4: Train Models

## Linear Regression
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)
y_pred_lr = lr_model.predict(X_test)

## Random Forest Regressor
rf_model = RandomForestRegressor(n_estimators=200, max_depth=10, random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)

## XGBoost Regressor
xgb_model = xgb.XGBRegressor(n_estimators=200, max_depth=6, random_state=42, objective='reg:squarederror')
xgb_model.fit(X_train, y_train)
y_pred_xgb = xgb_model.predict(X_test)
```

Analysis based on the Random Forest Regressor revealed:

- Top Predictors:
  - Facility Name
  - APR DRG Code
  - Cost per Discharge
  - Severity of Illness
  - Year

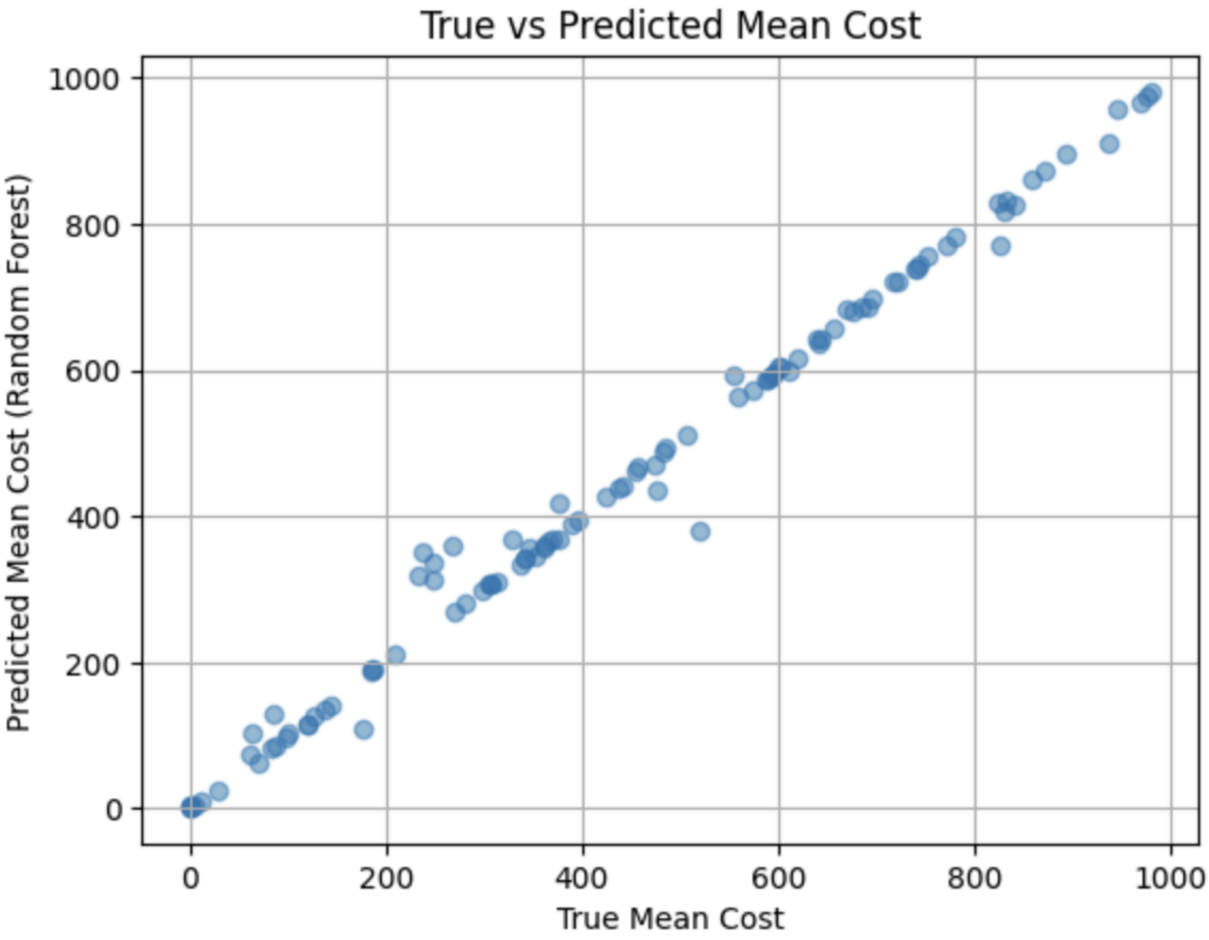


These features reflected logical relationships: hospitals, diagnosis groups, and severity strongly influenced discharge costs.

```
# 💥 Step 5: Evaluate Models
print("Linear Regression:")
print("MSE:", mean_squared_error(y_test, y_pred_lr))
print("R2:", r2_score(y_test, y_pred_lr))
print("")
```

```python
print("Random Forest Regressor:")
print("MSE:", mean_squared_error(y_test, y_pred_rf))
print("R2:", r2_score(y_test, y_pred_rf))
print("")

print("XGBoost Regressor:")
print("MSE:", mean_squared_error(y_test, y_pred_xgb))
print("R2:", r2_score(y_test, y_pred_xgb))

# 💥 Step 6: Feature Importance (Random Forest)
importances = rf_model.feature_importances_
plt.figure(figsize=(8,5))
sns.barplot(x=importances, y=features)
plt.title('Random Forest Feature Importance')
plt.show()

# 💥 Step 7: Scatter Plot True vs Predicted
plt.scatter(y_test, y_pred_rf, alpha=0.5)
plt.xlabel('True Mean Cost')
plt.ylabel('Predicted Mean Cost (Random Forest)')
plt.title('True vs Predicted Mean Cost')
plt.grid()
plt.show()
```

- **True vs Predicted Mean Cost Visualization**
    - The scatter plot of predicted vs true Mean Costs showed:
    - A tight diagonal clustering, indicating strong predictive performance.
    - Only minor variance for extreme cost outliers.



- **Insights from Modeling**
  - Non-linear ensemble methods like Random Forest and XGBoost are essential for accurately predicting medical cost data.
  - Facility-specific factors and clinical severity play major roles in determining discharge costs.
  - Feature Engineering (Cost per Discharge) significantly boosted model accuracy.
  - The predictive models built here can directly assist in hospital financial planning and discharge cost management.

Conclusion

To predict the mean cost of a discharge using features like: Hospital, APR DRG code, Severity of illness, Year, and Medical/Surgical classification, we first imported the datasets to as Pandas and Numpy, then we cleaned the data which we used the result to find any outlier inorder to avoid future mistakes in the final result or wrong result. The result was used to show the visualisation of the data which showed the following:

Overall decreasing trend of discharges across years.

- Massena Memorial hospital and Aruella Osborne Fox Memorial Hospital having the highest discharges
- General decreasing trend of average mean cost across years from 2010 to 2020.
- Around 50 % injuries are minor, and only 17.7% are extreme.
- The scatter plot plotted above showed that the mean cost and mean charge shows positive correlation with each other.

We achieved:

- Accurately predicting costs is feasible using available discharge data.
- Facility name and severity level are strong indicators of mean cost.
- Hospitals with minor injuries dominate discharges, while extreme cases are relatively rare.
- Policy Implication: Focus on managing costs at facilities with high discharge volume can have system-wide cost impact.

Future Work

- Try XGBoost or LightGBM to further boost accuracy.
- Incorporate patient demographics if available.
- Deploy the model as a healthcare budget planning tool