

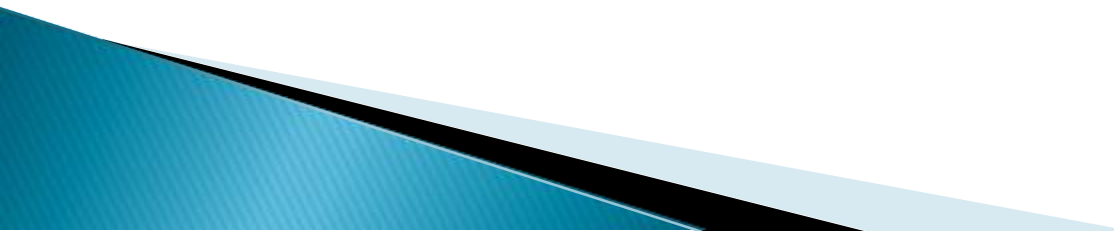
HPredict: A Machine LearningBased Model for Treatment Cost Estimation

Bridging Technology and Healthcare Economics

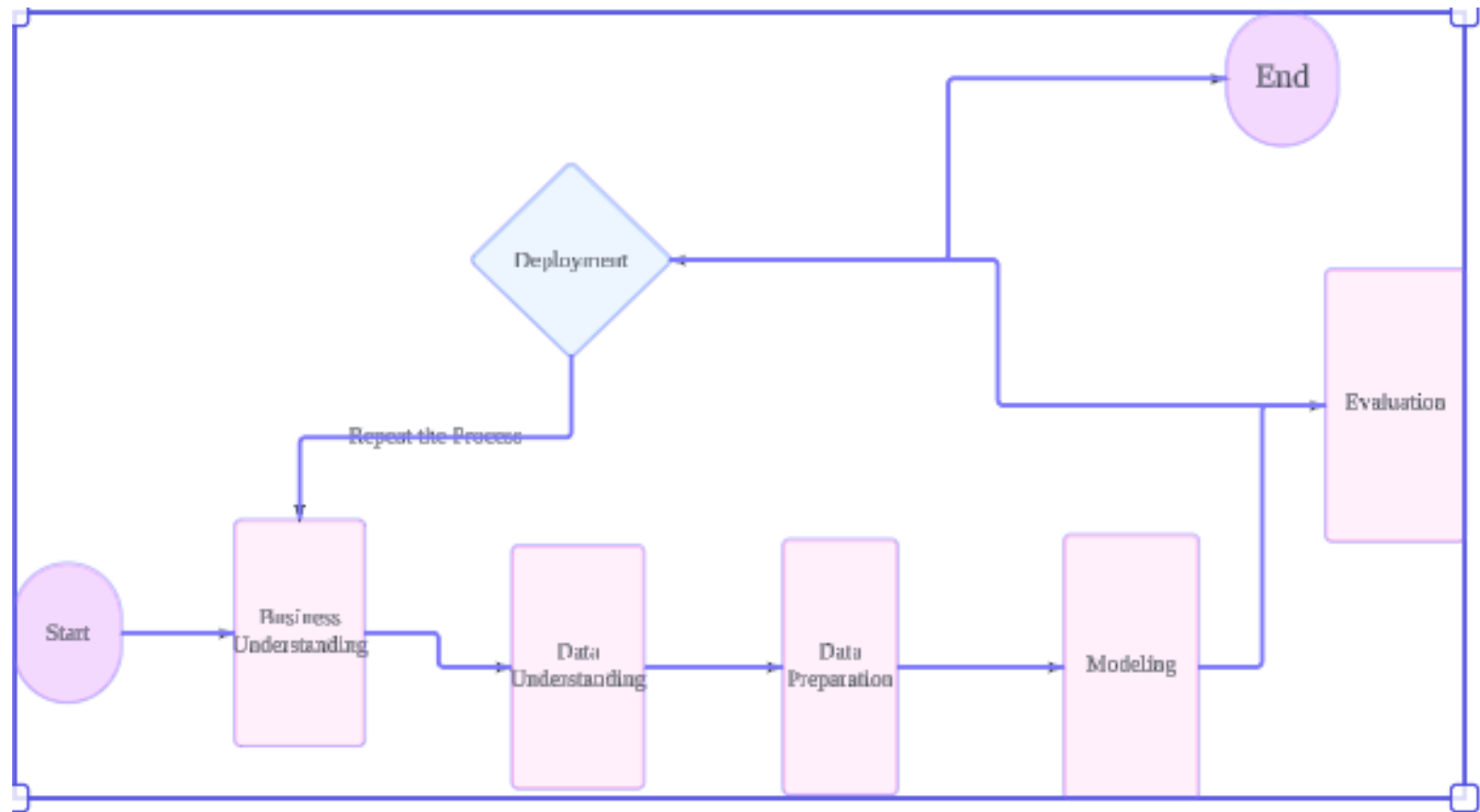
By: Ganapathyusha Puluputhuri Muni, Kavana Anil, Neha Sharma, Nivedita Venkatachalam



Problem Statement

- ▶ **Background:** Healthcare costs in the USA account for around 25% of an average household's annual income, with individuals spending approximately \$13,000 annually on medical expenses.
 - ▶ **Challenge:** Lack of visibility into healthcare costs and the factors influencing them leads to financial strain on patients and inefficiencies in insurance and healthcare operations.
 - ▶ **Objective:** Develop "HPredict" model for transparently estimating healthcare costs based on factors like age, disease type, medical history, and length of stay, aiding financial planning for patients and families.
 - ▶ **Approach:** Use statistical techniques for data preparation and apply advanced machine learning methods such as XGBoost, Random Forest, GB Regressor and regression models.
 - ▶ **Goal:** Improve predictability and transparency of healthcare costs, aiding patients in financial planning and enhancing claim management for insurers and healthcare facilities.
 - ▶ **Evaluation Metrics:** Assess model performance using MSE, MAE, RMSE, and Rsquared.
- 

Project Methodology and Flow



Project Requirements

Functional Requirements

- ▶ Scaling: Standardize and normalize data for uniform model contributions.
- ▶ Encoding: Convert categorical data to numerical values for model processing.
- ▶ Feature Engineering: Identify and utilize significant features for improved model performance.

ML Requirements

- ▶ Machine Learning Models: Implement Linear Regression, Polynomial Regression, Random Forest, XGBoost, and SGD Regressor.
- ▶ Evaluation Metrics: Assess model accuracy using MAE, MSE, RMSE, and Rsquared.

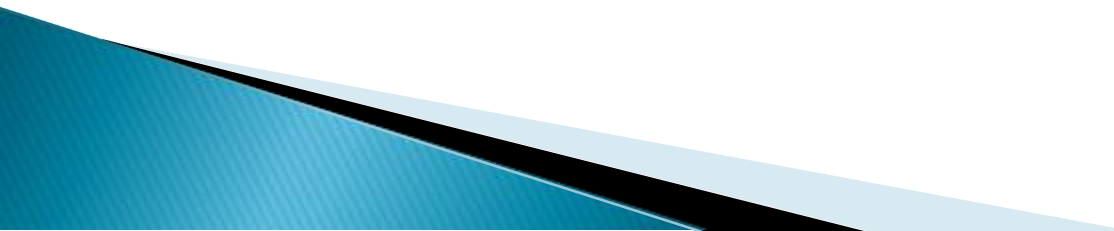
Data Requirements

- ▶ Data Sources: Utilize comprehensive datasets from relevant sources.
- ▶ Data Volume: Manage large datasets, expecting approximately 1 million records postpreprocessing.

Target Outcomes

- ▶ Reliable Cost Estimates: Provide accurate healthcare cost predictions.
- ▶ Decision Support: Aid in insurance claims management and healthcare service provisioning.

Technological Tools and Integration

- ▶ Advanced Analytics: Leverage cuttingedge tools for data processing and model development.
 - ▶ Integration: Ensure seamless integration with existing healthcare IT infrastructures.
- 

Data Process

▶ Data Gathering:

- Collect healthcare data from hospitals and government surveys.
- Capture patient demographics, medical histories, and treatment costs.

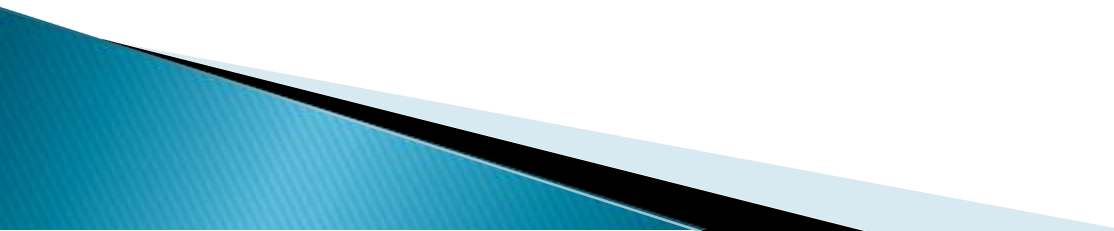
▶ Data Cleaning:

- Remove outliers and duplicates.
- Handle missing values and inconsistencies.
- Ensure dataset reliability through strict procedures.

▶ Data Integration:

- Merge and harmonize data from both sources.
- Standardize data formats.

▶ Feature Engineering:

- Extract insights and enhance predictions.
 - Standardize numeric features and encode categorical variables.
- 

▶ **Model Training:**

- Train ML algorithms to identify patterns.
- Select the most effective model for deployment.

▶ **Model Evaluation:**

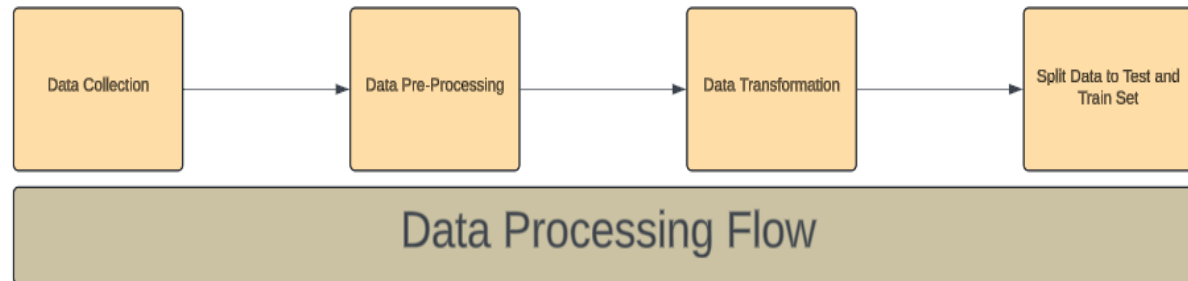
- Assess model performance using metrics like RMSE.
- Iterate and improve model accuracy.

▶ **Model Maintenance:**

- Monitor and update the model regularly.
- Retrain with fresh data to remain relevant.

▶ **Outcome:**

- Guide resource allocation and healthcare planning.
- Improve patient outcomes and operational efficiency.



Data Collection

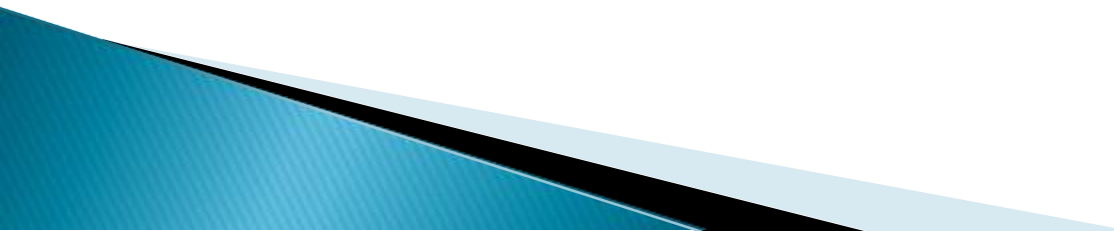
- ▶ Collecting data from 2 sources and merging them to consider all possible contributing features for healthcare cost.
- ▶ Data From Federal Survey

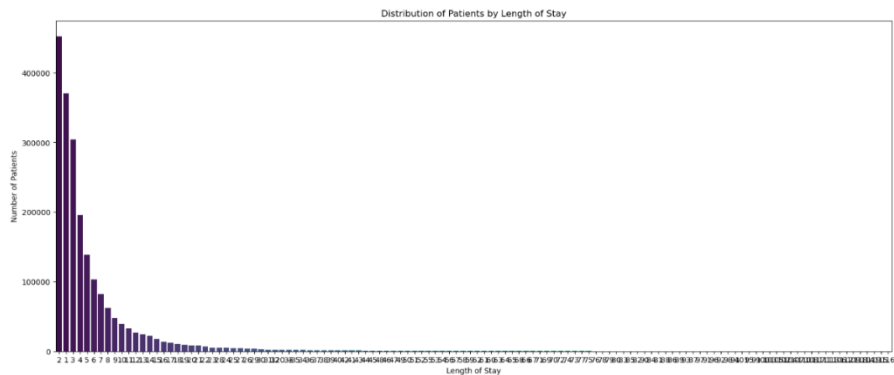
Purpose of data collection?	The objective of gathering information through surveys administered by federal agencies is to acquire a good understanding of various factors affecting the cost of healthcare, encompassing patient demographics, medical background, lifestyle preferences etc.
Utilization of Data	The information gathered will be utilized to create prediction models that will enable the healthcare industry to more easily allocate resources and manage finances by providing an accurate estimate of healthcare expenditures.
Units of measurement	Rows of Data Collected
Variables	Various features such as gender, age, reason for hospitalization etc.
Datatype of variables	Text and numeric
Collection Method	Download from the federal site and merge the data from all years

► Data From Inpatient Discharge

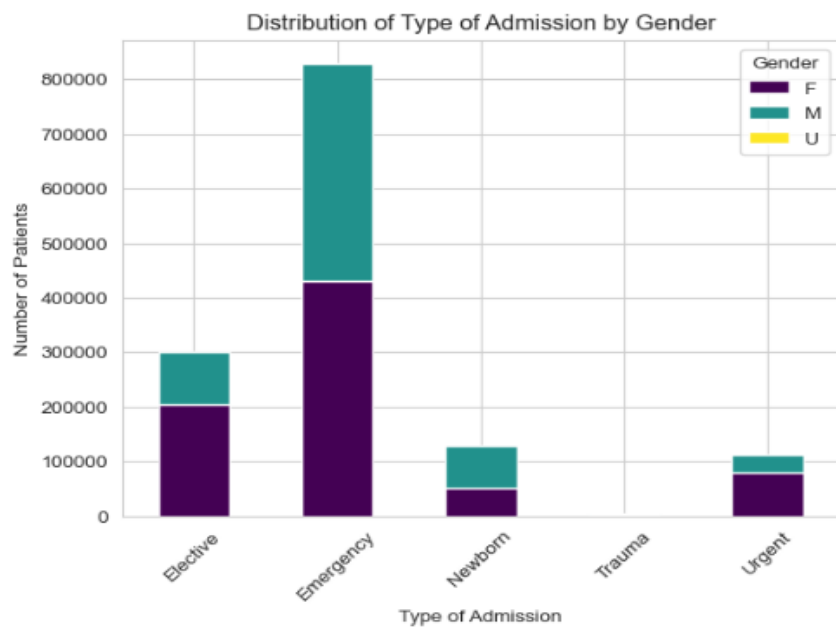
Purpose of data collection?	The objective of gathering information through inpatient admission and their medical expense in each state is to understand how various factors have impacted the cost of healthcare.
Utilization of Data	The information gathered will be utilized to create prediction models that will enable the healthcare industry to more easily allocate resources and manage finances by providing an accurate estimate of healthcare expenditures.
Units of measurement	Rows of Data Collected
Variables	Various features such as gender, age, reason for hospitalization etc.
Datatype of variables	Text and numeric
Collection Method	Download from the federal site and merge the data from all years

Data Preprocessing

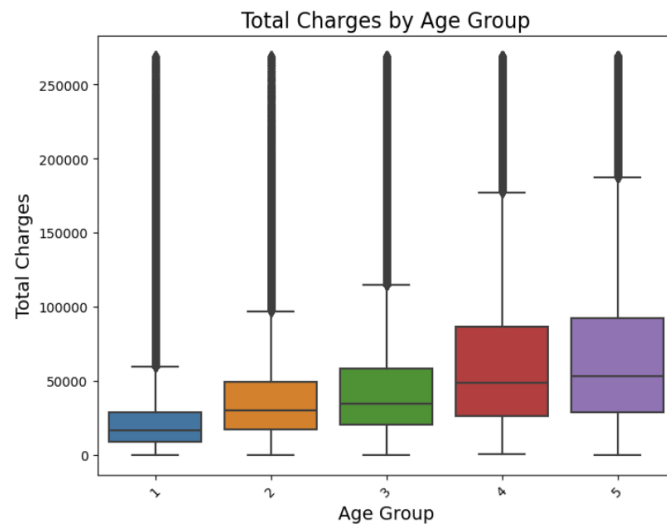
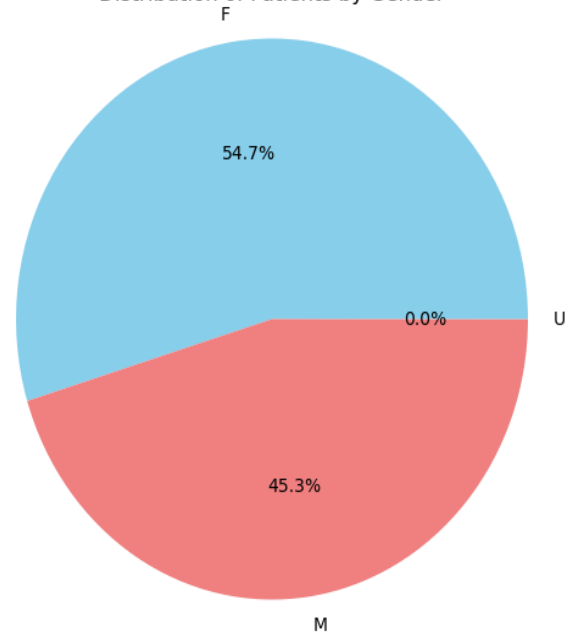
- ▶ Address differences in patterns, column numbers, and features in the final dataset.
 - ▶ Identify and analyze null values
 - ▶ Handle Outliers
 - ▶ Analyzing the datatype and information of each column
 - ▶ Understanding the distribution of each column in the dataset.
- 



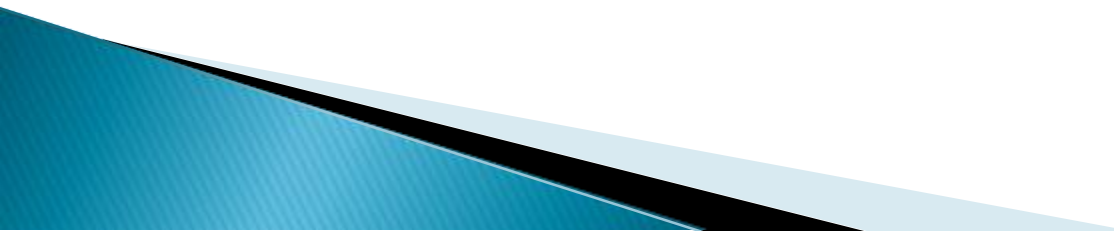
<Figure size 1000x600 with 0 Axes>

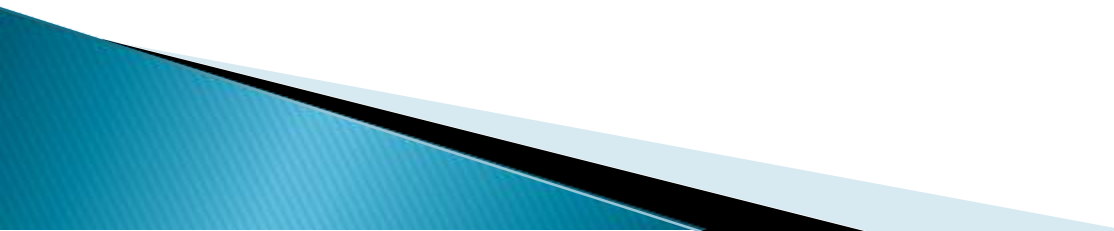


Distribution of Patients by Gender

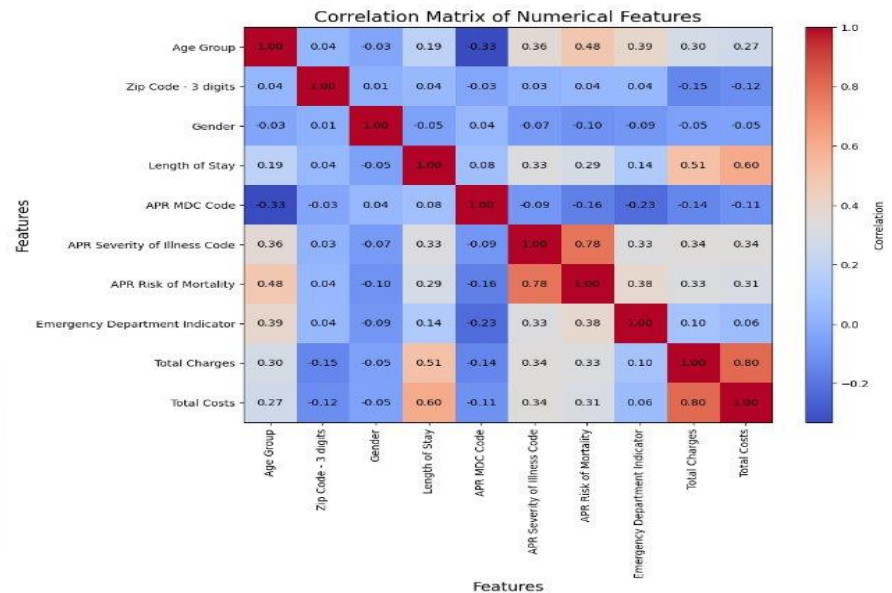
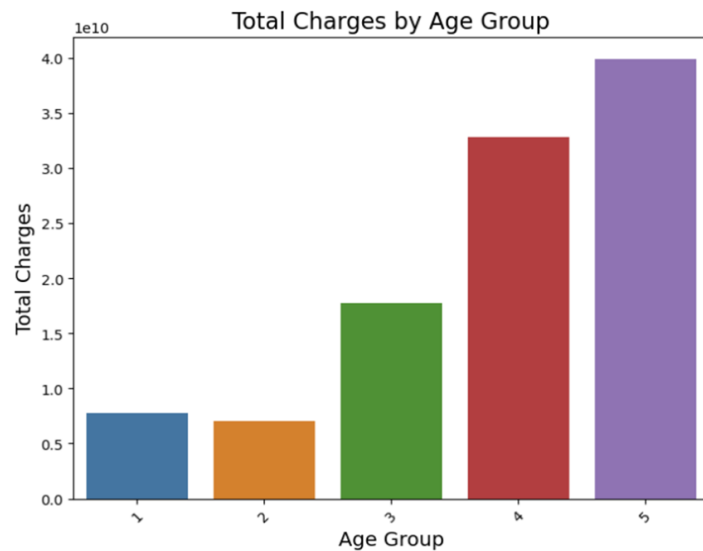
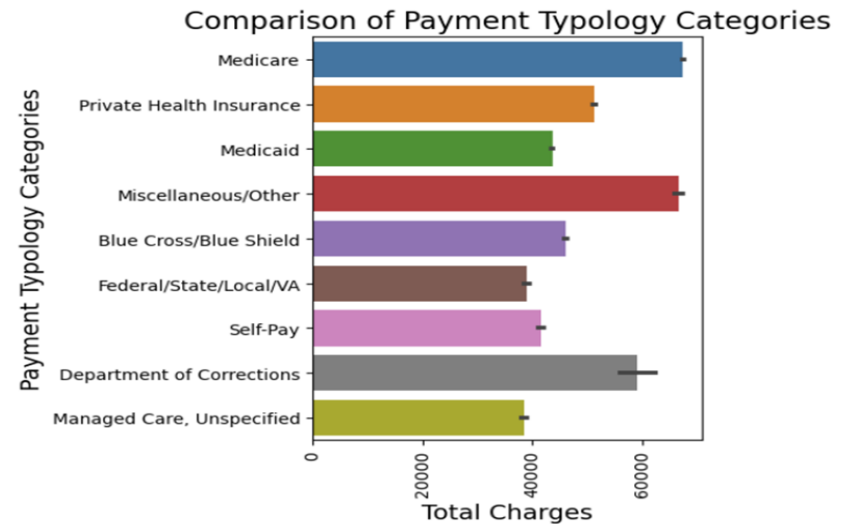
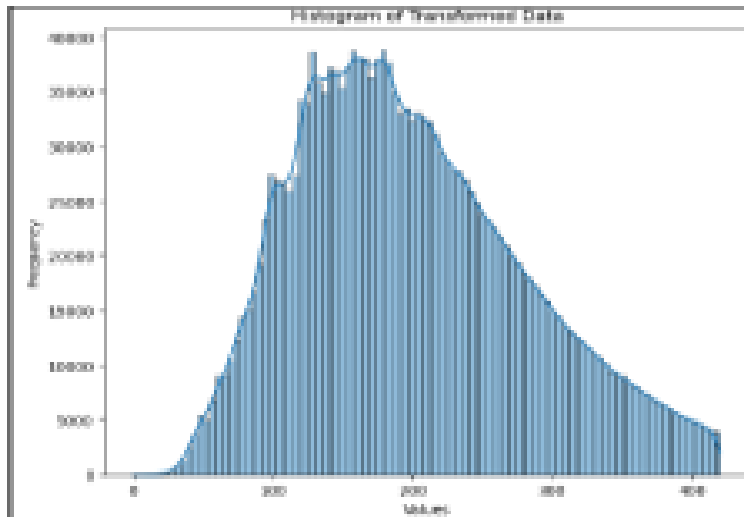


Data Transformation

- ▶ Data transformation is the process of converting data from one format or structure into another.
 - ▶ Tools used : python, pandas, NumPy and Scikitlearn.
 - ▶ In preparing data for machine learning models, we have to ensure that data is properly formatted by using several transformation techniques.
 - ▶ Core techniques in Data Transformation : Normalization, standardization, Encoding, Dimension Reduction.
- 

- ▶ Handling Missing values.
 - ▶ Encoding Age Group, Emergency Department Indicator, APR Risk of Mortality, Gender.
 - ▶ Data type transformations have been performed to ensure that numerical columns are formatted properly for modeling.
- 

EDA after Data Transformation



Data Preparation

- ▶ After preprocessing, we split our data for model training, testing and validation to ensure unbiased, representative subsets.
- ▶ **Purpose of Splitting:**
 - Training (60%): To comprehensively train the regression model predicting hospital costs.
 - Testing (20%): To evaluate model performance on unseen data.
 - Validation (20%): To finetune model hyperparameters and prevent overfitting.
- ▶ **Significance:**
 - Ensures models are trained and tested on diverse and ample data.
 - Helps achieve more accurate and reliable predictions of hospital costs.

Data Statistics

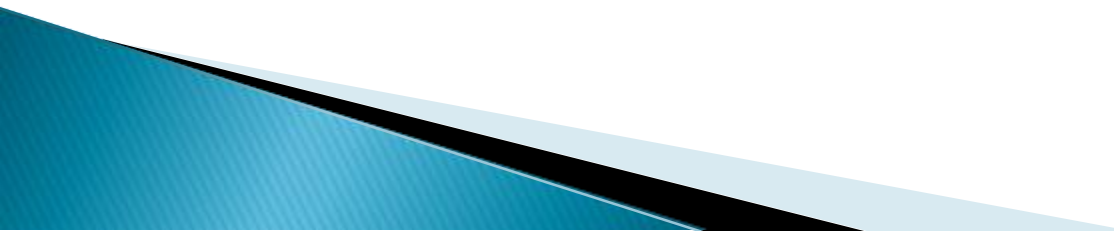
▶ Data Collection:

- Hospital Inpatient Discharges: 2,061,634 rows \times 33 columns.
- CCS Procedure Codes: 329 rows \times 7 columns.
- CCS Diagnostic Codes: 1,295 rows \times 2 columns.

▶ Data Preprocessing Steps:

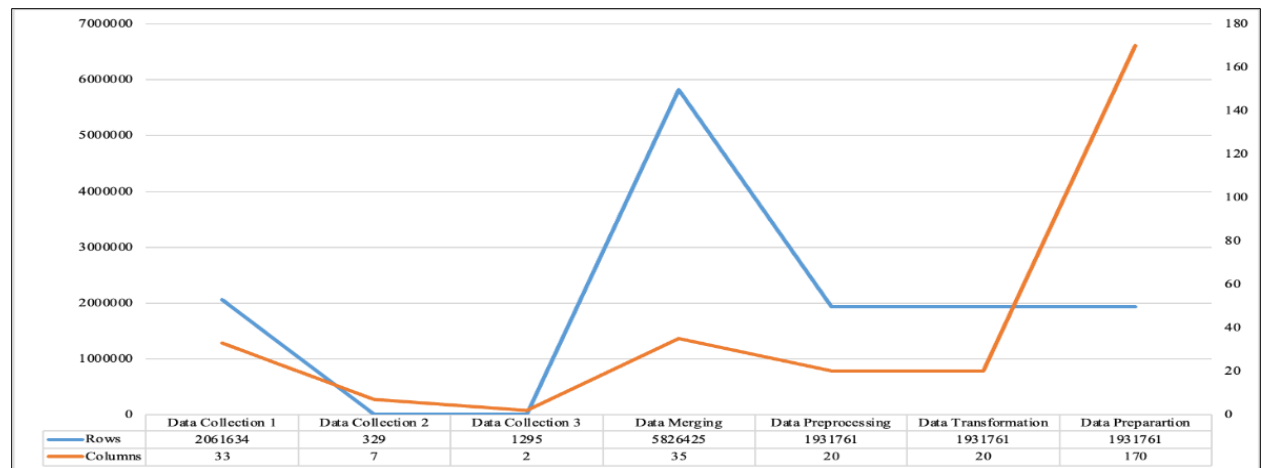
- Merging data sources: Resulted in 5,826,425 rows.
- Removing redundant data, columns with high missing values.
- Replacing missing values in specific columns.
- Outlier handling and data reduction to 1,931,761 rows \times 20 columns.

▶ Data Transformation:

- Encoding categorical variables into numerical categories.
 - Dummy encoding and data type changes.
- 

Data Transformation & Preparation

- ▶ **Data Transformation Continued:**
 - Comprehensive transformation techniques led to a final dataset of 1,931,761 rows \times 170 columns.
- ▶ **Data Preparation:**
 - Data split into training (60%), testing (20%), and validation (20%).
 - Resulting in:
 - Training: 1,159,056 rows \times 170 features.
 - Validation: 386,352 rows \times 170 features.
 - Testing: 386,353 rows \times 170 features.

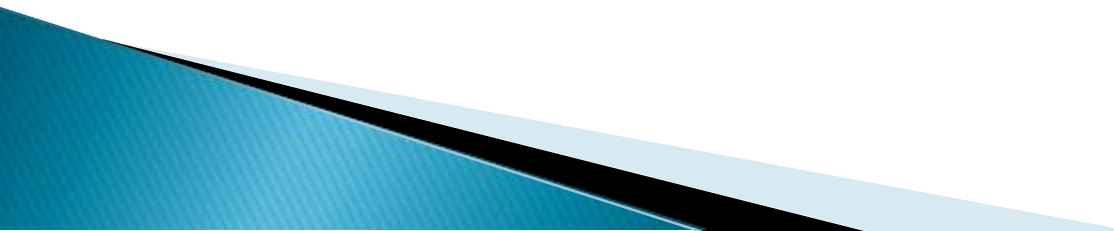


Model Development and Evaluation Metrix

► Models

1. Linear Regression
2. Polynomial Regression
3. Random Forest
4. XG Boost
5. Gradient Boost

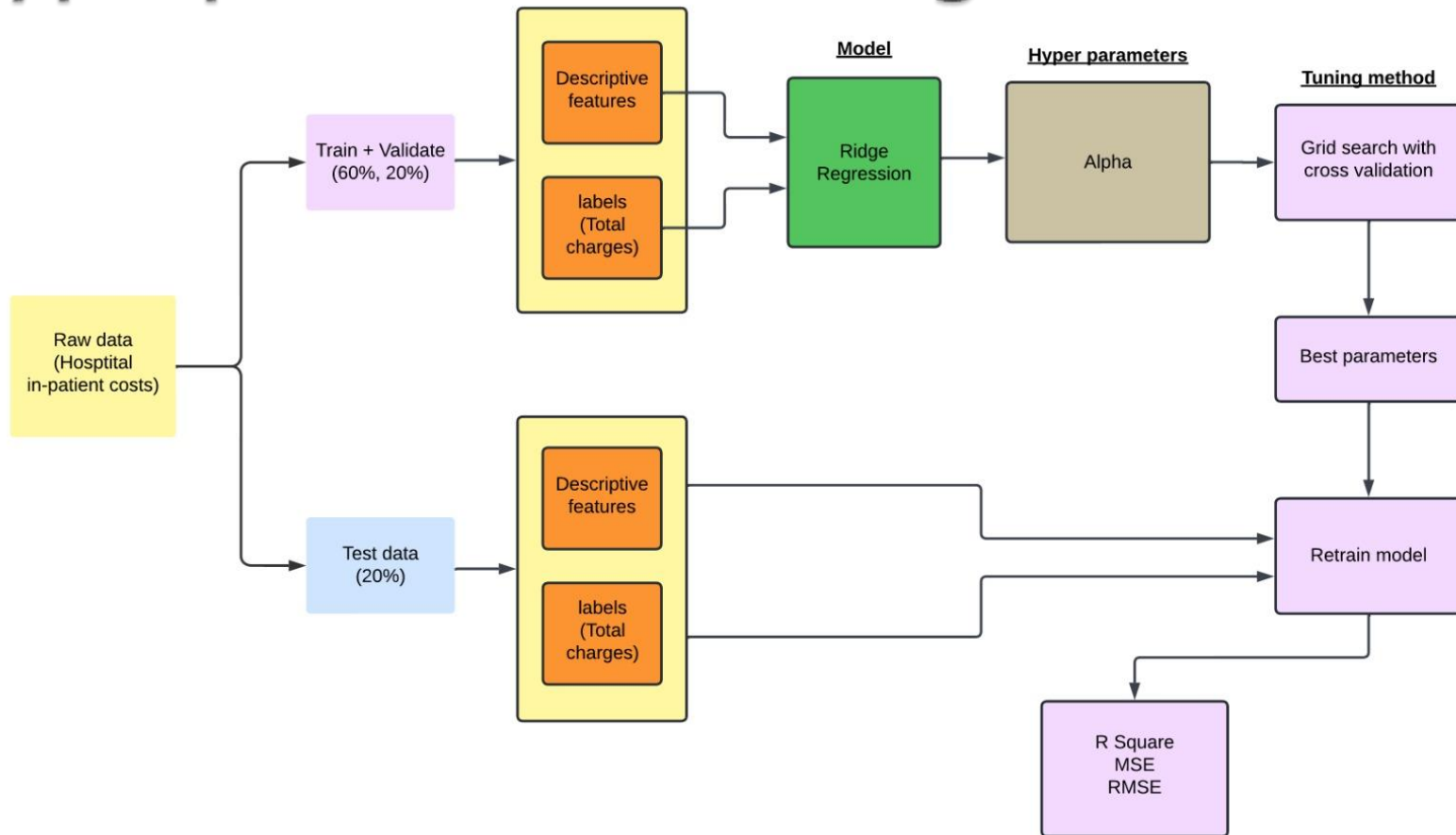
► Evaluation Metrix

1. MSE
 2. RMSE
 3. R^2
 4. MAE
- 

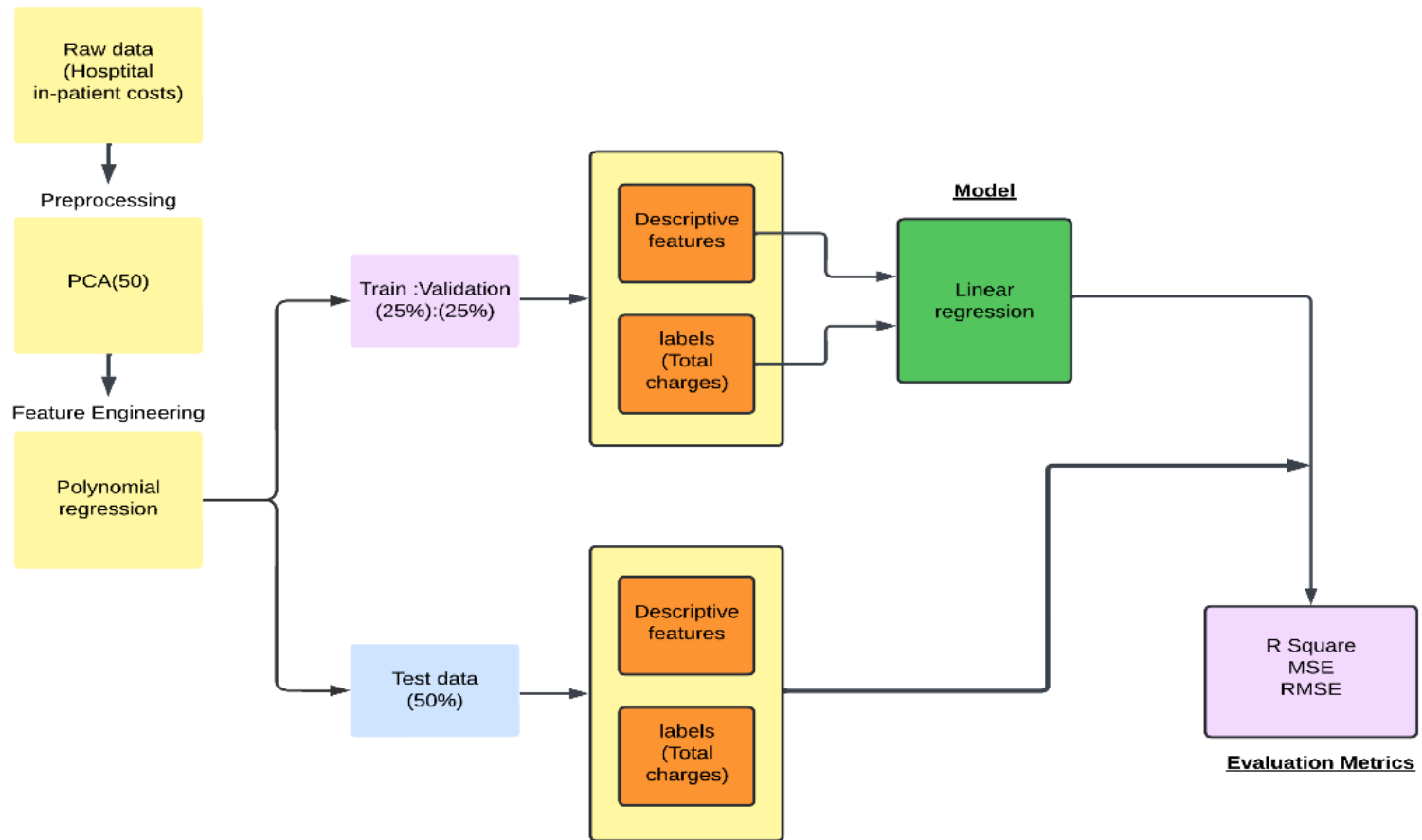
List of libraries used for model development

	Library	Method	Usage
Scikit-learn	sklearn.linear_model	Linear Regression	To import and evaluate the linear regression model.
	sklearn.metrics	Mean_squared_error, r2_score	Imported metrics to evaluate the model
	sklearn.model_selection	train_test_split	To split the data into training, validation, and testing datasets
	sklearn.decomposition	PCA	For dimensionality reduction
	sklearn.preprocessing	StandardScalar	To standardize the data
	sklearn.ensemble	RandomForestRegressor	Evaluate the random forest regression model.
	sklearn.model_selection	GridSearchCV, RandomizedSearchCV	Tuning the parameters
	sklearn.preprocessing	PolynomialFeatures	To evaluate polynomial regression.
xgboost	xgboost	xgb	To evaluate the XG Boost model.
Pandas	DataFrame	info, drop, head, shape, tail	Read data frames, manipulating, them to see the statistics of data before building models.
Numpy	np.log, np.sqrt	Transformation	To transform the right-skewed data.
Matplotlib and seaborn	Pyplot, sns	Plot graphs	Plotting various graphs to understand data and to plot metrics to compare models.

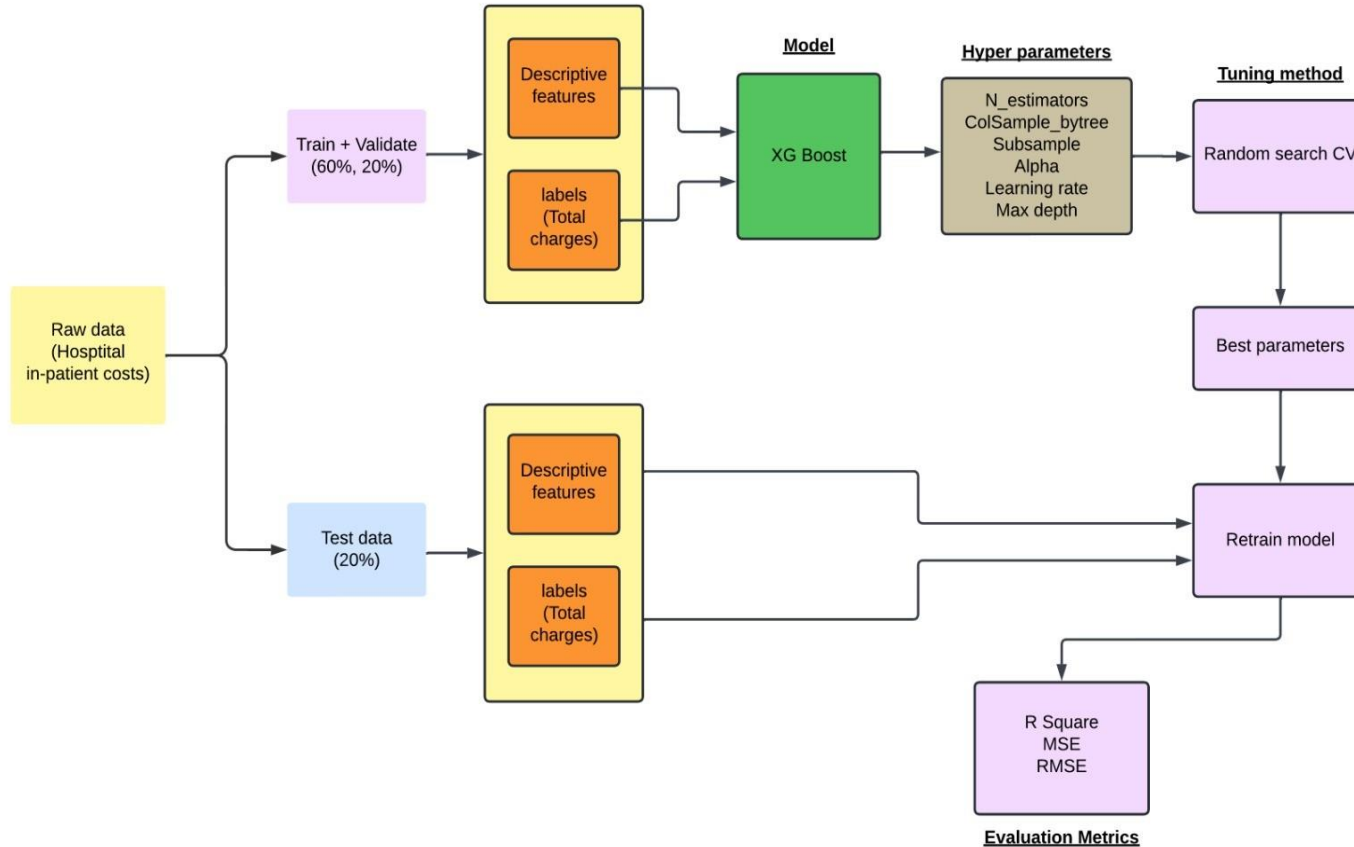
Systematic architecture for Linear regression model with hyperparameter tuning



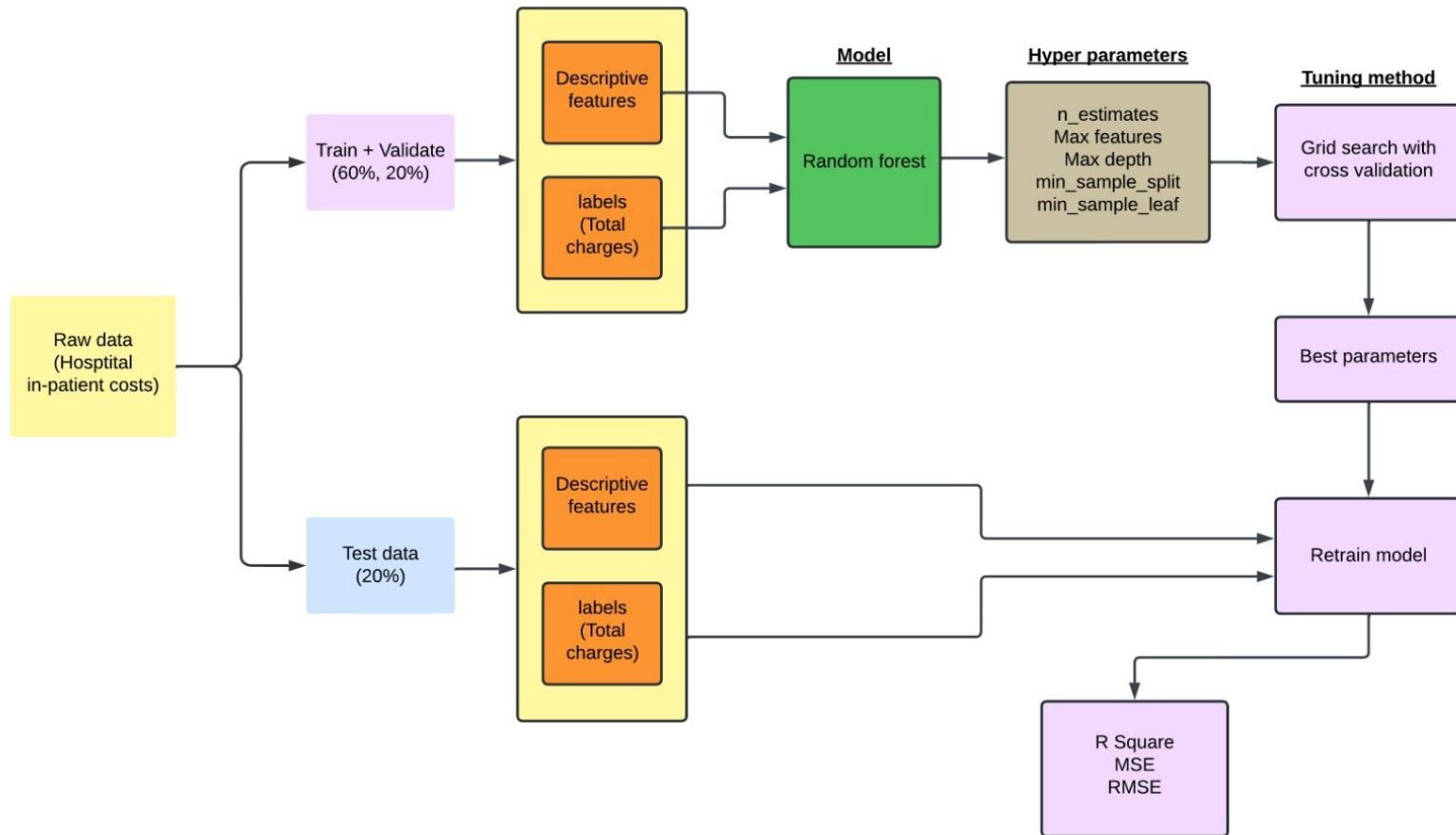
Architecture and dataflow for polynomial regression



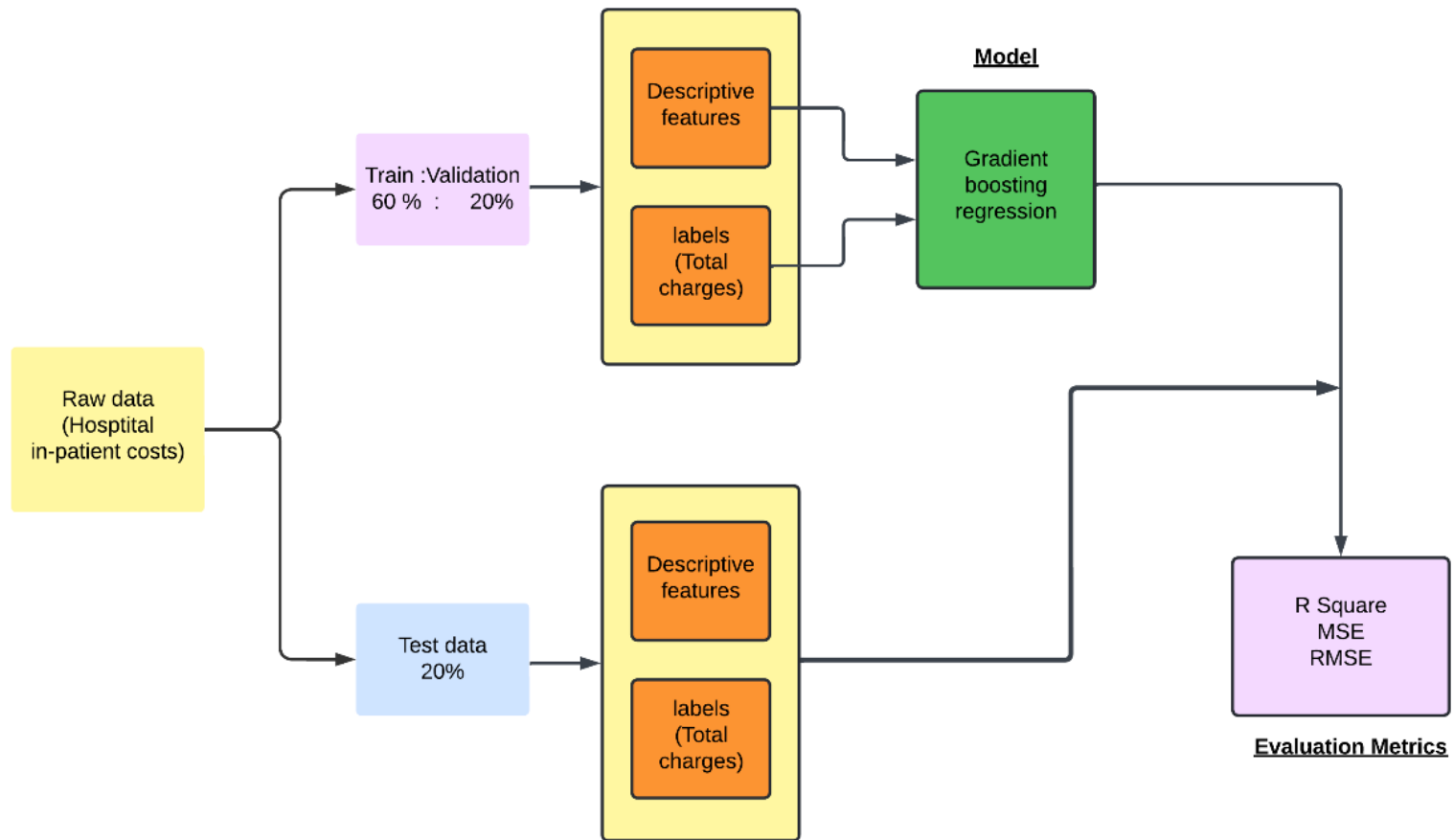
Architecture and dataflow for XG Boost



Architecture and dataflow for Random Forest



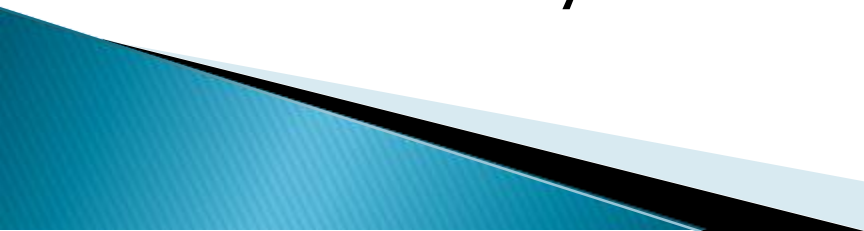
Architecture and dataflow for gradient boosting regressor



Models comparative analysis

	Advantages	Disadvantages
Linear Regression	<ul style="list-style-type: none">• Easy to implement and understand• Requires minimal computational resources• Results are easily interpretable• Provides clear feature importance	<ul style="list-style-type: none">• Limitation if relationship isn't linear• Outliers can heavily impact results• May miss complex patterns in data
Polynomial Regression	<ul style="list-style-type: none">• Fits a wide range of curvatures• Suitable for non-linear relationships	<ul style="list-style-type: none">• Model complexity grows exponentially• Risk of extreme values outside training data range
XGBoost	<ul style="list-style-type: none">• Handles large datasets efficiently• Includes L1 and L2 regularization• Built-in routines improve model robustness	<ul style="list-style-type: none">• Requires careful parameter tuning• Challenging to deploy in real-time applications
Random Forest	<ul style="list-style-type: none">• Works well with many features• Captures interactions without transformation• Provides probabilities for classification and continuous outputs	<ul style="list-style-type: none">• Less interpretable than decision trees• With high features can tend to overfit• Training can be expensive with large datasets
Gradient Boosting Regressor	<ul style="list-style-type: none">• Combines weak models for accurate predictions to give high performance• Suitable for regression and classification tasks• Automatically provides feature importance scores• Manages missing data internally	<ul style="list-style-type: none">• Slow due to sequential tree building• Parameters need careful tuning• Harder to interpret with increased trees• Less efficient on very large datasets

Justifications for models

- ▶ Linear Regression : Ideal for linear relationships, simple and interpretable
 - ▶ Polynomial regression : for non-linear relationships.
 - ▶ Random Forest : For large datasets, Reduce overfitting.
 - ▶ XGBoost : handles large datasets and efficient for its speed and performance.
 - ▶ Gradient Boosting : Efficient in handling large datasets by reducing the loss function.
- 

Evaluations

Linear Regression

► Baseline Model

Evaluation for Linear Regression:

Train Set :

Mean Squared Error (MSE): 1616.0536281829723

Root Mean Squared Error (RMSE): 40.20016950440598

R-squared (R^2) Score: 0.7593232863658076

Validation Set:

Mean Squared Error (MSE): 1617.6616515690644

Root Mean Squared Error (RMSE): 40.220164738213896

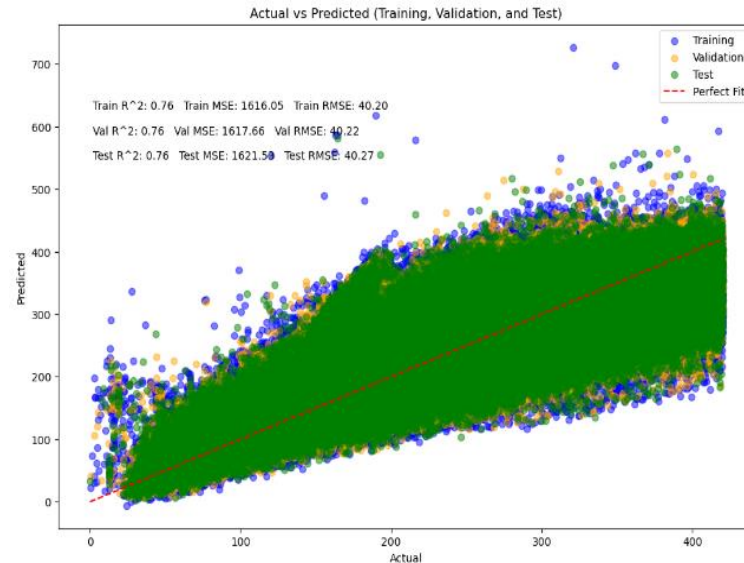
R-squared (R^2) Score: 0.7591227680578773

Test Set:

Mean Squared Error (MSE): 1621.5280787170364

Root Mean Squared Error (RMSE): 40.26820183118482

R-squared (R^2) Score: 0.7586097970692673



► Hyperparameter tuned model

Fitting 5 folds for each of 5 candidates, totalling 25 fits

Evaluation for Ridge Regression or hyperparameter tuned linear regression :

Best Parameters: {'alpha': 1}

Mean Squared Error (MSE) of Best Fit: 1621.5254168382644

Root Mean Squared Error (RMSE): 40.26816877930091

R-squared (R^2) Score: 0.7586101933321907

Polynomial Regression

Train Set:

Mean Squared Error (MSE): 1086.8226879038057

R-squared (R^2) Score: 0.8378609866610095

Validation Set:

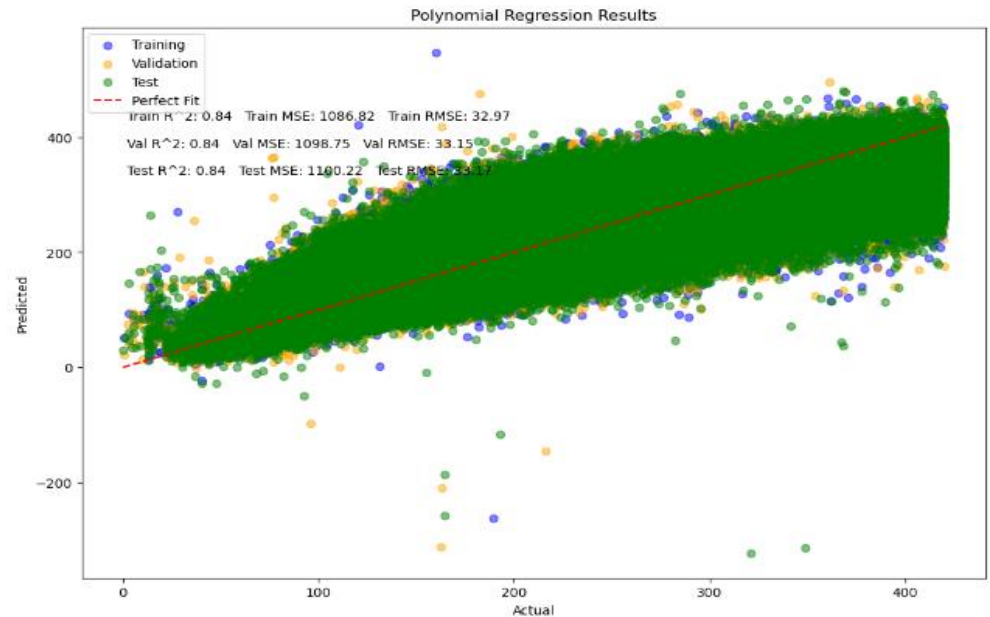
Mean Squared Error (MSE): 1098.745770830839

R-squared (R^2) Score: 0.836229464306121

Test Set:

Mean Squared Error (MSE): 1100.2248194496171

R-squared (R^2) Score: 0.8363934593347323



Gradient Boosting

Evaluation for Gradient Boosting Regressor:

Train Set :

Mean Squared Error (MSE): 1228.2911316779077

Root Mean Squared Error (RMSE): 35.0469846303203

R-squared (R^2) Score: 0.816755838463951

Validation Set:

Mean Squared Error (MSE): 1232.986443504868

Root Mean Squared Error (RMSE): 35.11390669670448

R-squared (R^2) Score: 0.8162205892238459

Test Set:

Mean Squared Error (MSE): 1234.470890496043

Root Mean Squared Error (RMSE): 35.13503793218449

R-squared (R^2) Score: 0.8164306890958309

Random Forest

► Baseline Model

Evaluation for Random forest regression with $n_{\text{estimatoros}} = 10$:

Random Forest – Train Set:

Mean Squared Error (MSE): 118.6110197485897

Root Mean Squared Error (RMSE): 10.890868640681958

R-squared (R^2) Score: 0.9823354188647886

Random Forest – Validation Set:

Mean Squared Error (MSE): 654.1232587331735

Root Mean Squared Error (RMSE): 25.575833490488115

R-squared (R^2) Score: 0.9025980496231842

Random Forest – Test Set:

Mean Squared Error (MSE): 656.0076304478625

Root Mean Squared Error (RMSE): 25.61264590876668

R-squared (R^2) Score: 0.9023428473941634

Evaluation for Random forest regression with $n_{\text{estimatoros}} = 100$:

Random Forest – Train Set:

Mean Squared Error (MSE): 83.94441776289275

Root Mean Squared Error (RMSE): 9.162118628510152

R-squared (R^2) Score: 0.9874982697091403

Random Forest – Validation Set:

Mean Squared Error (MSE): 594.7306317893311

Root Mean Squared Error (RMSE): 24.38709970023765

R-squared (R^2) Score: 0.9114418839083868

Random Forest – Test Set:

Mean Squared Error (MSE): 597.5103670006454

Root Mean Squared Error (RMSE): 24.444025180003504

R-squared (R^2) Score: 0.911051093942437

► Hyperparameter tuned model

► CV=5

Train Set:

Mean Squared Error (MSE): 487.88824291704145

Root Mean Squared Error (RMSE): 22.08819238681702

R-squared (R^2) Score: 0.9273394540389972

Validation Set:

Mean Squared Error (MSE): 797.8072700874644

Root Mean Squared Error (RMSE): 28.245482295182434

R-squared (R^2) Score: 0.8812028419814678

Test Set:

Mean Squared Error (MSE): 799.9298890131954

Root Mean Squared Error (RMSE): 28.283031821450745

R-squared (R^2) Score: 0.8809177338501397

CV=3

Train Set:

Mean Squared Error (MSE): 270.8346524409707

Root Mean Squared Error (RMSE): 16.457054792427794

R-squared (R^2) Score: 0.9596649560689138

Validation Set:

Mean Squared Error (MSE): 751.8578661537214

Root Mean Squared Error (RMSE): 27.420026735102237

R-squared (R^2) Score: 0.888044918764468

Test Set:

Mean Squared Error (MSE): 753.9975206226294

Root Mean Squared Error (RMSE): 27.459015288655735

R-squared (R^2) Score: 0.88775549624995

XGBoost

► Baseline Model:

XGBoost - Train Set:

Mean Squared Error (MSE): 1072.5170413814017

Root Mean Squared Error (RMSE): 32.749305967934674

R-squared (R^2) Score: 0.8402714660363257

XGBoost - Validation Set:

Mean Squared Error (MSE): 1075.464585516861

Root Mean Squared Error (RMSE): 32.79427671891638

R-squared (R^2) Score: 0.8398583955057528

XGBoost - Test Set:

Mean Squared Error (MSE): 1080.254316913756

Root Mean Squared Error (RMSE): 32.86722253117467

R-squared (R^2) Score: 0.8391869914562757

► Hyperparameter Tunned Model:

Loaded XGBoost - Train Set:

Mean Squared Error (MSE): 670.2772397910833

Root Mean Squared Error (RMSE): 25.889713010983403

R-squared (R^2) Score: 0.9001765037475288

Loaded XGBoost - Validation Set:

Mean Squared Error (MSE): 685.8776699411809

Root Mean Squared Error (RMSE): 26.189266311624326

R-squared (R^2) Score: 0.897869672297606

Loaded XGBoost - Test Set:

Mean Squared Error (MSE): 689.4711581248476

Root Mean Squared Error (RMSE): 26.25778281052777

R-squared (R^2) Score: 0.8973612699285936

Results and Comparison of Model performance

Model	Train			Test		
	MSE	RMSE	R Squared	MSE	RMSE	R Squared
Linear Regression	1616.05	40.20	0.759	1621.52	40.27	0.758
Polynomial Regression	1086.82	32.96	0.837	1100.22	33.16	0.836
Random Forest	83.94	9.16	0.987	597.51	24.44	0.911
Gradient Boosting	1228.29	35.04	0.816	1234.47	35.13	0.8164
XGBoost	670.27	25.88	0.900	689.47	26.25	0.897

Thank You

