Assessing the Impact of Hospital Ratings and Conditions on Risk-Adjusted Mortality: Evidence from California Hospitals in 2023

Hilda Joseph [1] **Caroline Ngyuen** [1] **Eddie Zhang** [1]

1. Data Description

1.1. Data

The dataset is from data.gov, and refers to a California set about hospital mortality rates for certain conditions and their quality ratings. The data was publicly accessible, and the idea of the paper was established due to the authors' interest in the healthcare field.

Dataset
https://catalog.data.gov/dataset/california-hospital-inpatient-mortality-rates-and-quality-ratings-8e21f

1.2. Key Variables

Hospital, rating, condition, and risk-adjusted mortality rate The hospitals refer to various hospitals in California. The ratings of the hospitals are either 'worse,' 'better,' or 'as expected' than normal. The condition is the disease the person in the case has. And the risk adjusted mortality rate is the hospital's observed death rate to its expected death rate after accounting for patient-specific risk factors like age.

1.3. Research Question

For each Californian hospital in 2023, how are the hospital ratings and medical conditions associated with the risk adjusted mortality rate and the proportion of deaths out of total cases?

1.4. Scope

The data focuses on Californian hospitals in 2023 because it represents the most recent year available in the dataset. Using the latest year ensures that our analysis reflects the most up-to-date information on hospital performance, operations, and outcomes, making the findings more relevant to our research questions. The dataset includes medical conditions such as: AAA Repair Unruptured, AMI Acute Stroke, Acute Stroke Hemorrhagic, Acute Stroke Ischemic, Acute Stroke Subarachnoid, Carotid Endarterectomy, Esophageal Resection, GI Hemorrhage, Heart Failure, Hip Fracture, PCI Pancreatic Cancer, Pancreatic Other, Pancreatic Resection, Pneumonia.

1.5. Challenges in Reading, Cleaning, and Preparing the Data

Since we are working with real-world administrative or performance data, it comes with multiple challenges. Below are key issues and ideas for how to address them:

Small Sample Sizes and Statistical Stability

Some hospitals may have very low case volume for certain conditions or procedures in some years. It is possible that mortality rates derived from small denominators can be highly volatile. Steps need to be taken to normalize the smallest sample sizes. For different case volumes, we will weight each hospital's contribution by the number of cases to reduce the distortion from small-sample hospitals. We will exclude conditions with too few cases that are not statistically meaningful.

Numeric Formatting

Some entries might include non-numeric characters, which must be cleaned or parsed carefully. The risk-adjusted mortality rates come in formats such as X.X per 100 cases, we need to clean or adjust the scale to make it consistent We will convert mortality rates, case counts, and volume numbers to numeric types and watch out for cells with text.

Outliers, Extreme Values, and Anomalies

Before modeling, we need to explore distributions and flag extreme outliers is crucial. We will flag suspiciously high or low mortality rates and investigate whether these are valid small-sample effects or reporting errors.

Missing, Suppressed Values - Lack of Cases

Many hospitals have no values for many conditions, meaning there will need to be a lot of adding of NAs to better sort the data. We will identify which fields are empty and exclude them from rate calculations.

1.6. Steps for Workflow

Read In with Care

We will use a library depending on the language that tolerates missing or malformed entries and reads numeri-

cal/string variables.

Initial Sanity Checks

We plan to count missingness per column, check ranges of numeric variables, look for impossible values (i.e. negatives, greater than 100 percent mortality), narrow down the dataset to the year 2023 only, and inspect sample sizes.

Standardized Column Names and Formats

It is important to unify naming (e.g. "AMI" vs "Acute Myocardial Infarction") and convert rates to consistent numeric scale, coerce types.

Model-Ready Dataset

Ensure to drop or flag residual missing entries, impute if justifiable, and divide into training/test splits if doing an explanatory modeling.

Documentation

We will keep track of cleaning rules for the dataset (e.g. how missing values are treated) to ensure transparency and reproducibility.

2. Pre-Analysis Plan

2.1. Overview Done by Eddie

In the methods section, this paper will discuss the various approaches taken to implement the three models we used. The methods section will begin with a discussion of the models used: linear regression, decision trees, and K-Nearest Neighbors (KNN) regression, and then provide a detailed justification for each of their uses. The methods will continue with the training procedure for each of the models, followed by the model validation plan, and details about each of the models implementations. The methods section aims to be a complete summary on the models used to unpack, for each Californian hospital in 2023, how the hospital ratings and medical conditions are associated with the risk adjusted mortality rate and the proportion of deaths out of total cases.

2.2. Models Used and Justification Done by Caroline

In this analysis, we will employ three distinct predictive modeling approaches (linear regression, decision trees, and KNN regression to explore how hospital ratings and medical conditions relate to risk-adjusted mortality rates and death proportions across Califronian hospitals in 2023. Linear regression will serve as our baseline model due to its interpretability and capacity to estimate the linear association between predictors, such as rating and condition, and continuous outcomes (mortality rate). Decision trees will be used to capture potential nonlinear interactions and hierarchical decision patterns that may emerge between hospital characteristics and outcomes, allowing for an intuitive visualization. KNN regression will provide a non-parametric perspective, estimating outcomes based on local similarity in hospital profiles and patient conditions. Together, these models represent a balanced methodological framework, enabling us to identify both general trends and localized variations in hospital performance.

2.3. Model Training Procedure Done by Eddie

The three models we are training each have unique procedures. Linear regression models optimally weigh explanatory variables in order to predict the outcome variable. To train the linear model, the end point is plotting the x or independent variables or feature against the single linear regression function's production of y_hat, which is b0 +b1 * x. Besides x, b0 is the mean of y or the target outcome minus b1 times the mean of x. Finally b1 is the inner product of x minus the mean of x and y minus the mean of y divided by the inner product of x minus the mean of x and x minus the mean of x. A decision tree is a set of decision nodes, with a set of edges, and a set of terminal nodes. These represent the decision points, choices, and outcomes. The goal is to build a tree from data that predicts outcomes for future cases. After importing from sklearn.tree the DecisionTreeClassifier and plot_tree, we will create a target variable y dataframe and a X with features/predictors that want to be used. Using the classifier and plot tree, we will have a decision tree created. KNN regression computes the distance from x hat to each observation of xi. Then it finds the "nearest neighbors" x*1, x*2, and so on until x*k to x hat in the data, with outcomes y*1, y*2, and so on until y*k. Finally, you compute the average nearest neighbor outcome as the prediction for x hat. To create a KNN regression function, follow the code. First, defining the variable: def knn_reg(x_hat,gdf,K): — Then, computing distances between x_hat and the data: squared_differences = (x_hat - gdf.loc[:,['x1','x2']])**2 — distances = np.sum(squared_differences , axis = 1) — Afterwards, finding the k smallest values in dist: neighbors = np.argsort(distances)[:K].tolist() — Find y values for the nearest neighbors: y_star = gdf['y'].iloc[neighbors].tolist() — Average neighbor values to get prediction: y_hat = np.mean(y_star) — Finally, return a dictionary of computed values of interest: return 'y_hat':y_hat, 'y_star':y_star, 'neighbors':neighbors — and that's it.

2.4. Model Validation Plan Done by Hilda

To ensure our models are accurate and generalize well, we will split the data into training and testing sets. This helps us see how the models perform on data they have not seen before. For the linear regression model, we will check how

well it fits the data using R-squared, adjusted R-squared, and root mean squared error. For the decision tree and KNN regression, we will use cross-validation to prevent overfitting and to find the best parameters. The decision tree will be tuned and pruned to keep it from becoming too complex, while KNN will be tested with different values of k to find the best number of neighbors. We will compare all models using mean absolute error and root mean squared error to see which one predicts the outcomes most accurately.

2.5. Details about Model Implementation Done by Hilda

All models will be built in Python using standard machine learning tools. Before modeling, we will clean the dataset by fixing missing or inconsistent data and convert text categories, like hospital rating and condition (such as heart attack or pneumonia), into numerical form using one-hot encoding. The linear regression model will be used to measure how ratings and conditions relate to the risk-adjusted mortality rate and deaths per case. The decision tree will help us find nonlinear relationships and show which features are most important in predicting outcomes. The KNN regression model will predict outcomes by comparing each hospital to others with similar ratings and conditions. After training the models, we will compare their results using the same validation metrics to decide which one performs best.