An Examination of Ischemic Stroke Data: Exploring Machine Learning Models to Predict Hospital Ratings and Risk Adjusted Rate.

Rachel Kersey   Helen Sparling

Abstract

This paper examines the best machine learning models to use in a dataset about ischemic stroke. Specifically, we used four different machine learning models: decision tree classifier, random forest and k nearest neighbor to predict hospital ratings, and multiple linear regression/a separate random forest model to predict risk adjusted rate. The variety of models gave us a detailed picture of the models that performed best in predicting categorical and numeric variables – specifically, hospital ratings and (categorical) and risk adjusted rate (numeric). The results indicated that the best classification model for hospital ratings was the random forest model, as it produced an ensemble accuracy of 0.93. Further, the decision tree classifier also performed decently, giving an accuracy score of 0.93. For prediction of risk adjusted rate, the ensemble random forest also performed the best, producing a r-squared ensemble of 0.76. The multiple linear regression did not perform well whatsoever, with an r-squared of .005. Overall, this analysis gives a good idea of the best machine learning models to use when evaluating this dataset and making predictions related to hospital ratings and risk adjusted rates. (Rachel)

1. Data

The data set used for this investigation is "Ischemic Stroke Mortality and Readmission Rates at 30-Days and Quality Ratings in CA Hospitals." The dataset is from Data.gov, a reputable website that contains many different kinds of datasets across a range of topics. The hyperlink to the data set is given below. Simply put, this dataset contains information on risk-adjusted mortality and readmission rates at 30 days, as well as quality ratings and deaths/readmissions for Ischemic Strokes treated in hospitals throughout the state of California. Importantly, the risk-adjusted 30 day mortality examines mortality rates for each datapoint as adjusted for the severity of the individuals condition (risk adjusted). This dataset is important as it shows how hospital ratings are related to stroke readmissions and deaths: and machine learning models predicting these variables could demonstrate the need for hospitals to increase their ratings in order to lower readmission or death rates. The URL is available here:

Stroke Dataset Link

The dataset contained a total of 9 variables, 5 of which were numeric, and 4 of which were categorical. Prior to analysis, the data was subsetted to include only the five most populous counties in the state of California in order to make the dataset more manageable. The breakdown of each variable is as follows:

- Year: quantitative discrete, year data was recorded.
- County: categorical, county data was recorded in.
- Hospital: categorical, hospital data was recorded in.
- OSHPDID: categorical, hospital ID, not important for research question.
- Risk Adjusted Rate: quantitative continuous (float), risk rate adjusted for age, health severity, readmissions and deaths.
- Number of Deaths/Readmissions: quantitative continuous (float), number of deaths/readmissions for hospital that year.
- Number of Cases: quantitative continuous (float), number of hospital cases.
- Hospital Ratings: categorical, patient hospital rating.
- Location: quantitative (float), hospital location coordinates. (Rachel Kersey)

Data Dimensions The dataset included 1,026 observations with a total of 9 variables/columns. The cleaned dataset had less variables/columns. Details of that procedure are in the following sections. (Helen Sparling)

Data Quality Notes: The data had missing values that would cause problems moving into analysis and model construction. In order to address this issue, the missing values were replaced with np.Nan. Missing data was not deleted in order to avoid potentially skewing the end results in later analysis, and constructing models. Later in our investigation, however, some models produced errors after running due to the np.Nan. To resolve this, rows with missing data had to be deleted. While this was not ideal, the number of rows that fell into this category were few and we were still able to use our models on a subset that still included a majority of the original data. The impacts of these deletions should be negligible.

Data cleaning included: coercing columns with periods to numeric as a precaution by using pd.to_numeric(), removing unnecessary wording by changing “30-day Readmission” to “Readmission” using the .str.replace(), splitting the year column to change the format from “2011-2012” to “2011”, and changing the year column to numeric using the pd.to_numeric(). Changing the year column required a multiple steps. The year column was split at the “-” character using the .st.split() function. Then, as only the start year was needed, not the end year, the first part of the resulting string was selected and kept in a new column, and the old column was deleted. This method was slightly more cumbersome than likely necessary, but it was successful in the end and resulted in a ”Year” column that was ready for analysis, visualization, and model construction. Together, all of the changes allowed for straightforward data visualization. Additionally, there were no unnecessary dollar signs, commas, or other characters, so no other substantial cleaning was necessary.

As the original dataset was quite large and in order to simplify analysis to further focus the research question, a subset of the dataset including only the 5 most populated counties was selected. The location column contained information about the coordinate points of each data entry. The dataset, however, already contained a different column concerning location in terms of County. County is a far more intuitive and understandable variable to work with compared to coordinates points. Thus, we removed the “location” column as the information was unhelpful and redundant. (Helen Sparling)

There were some outliers in the dataset. Boxplots were made for the three numeric variables: risk adjusted rate, number of cases, and number of deaths/readmissions. All of these variables were right skewed. Future steps will involve transformations to attempt to fix the data. Transformations, described in detail later, included log, square root, and cubed root. (Rachel Kersey)

1.1. Context and Relevance

The goal of this project was to investigate the relationships between year and county and how they relate to risk adjusted rate, deaths, readmission, and case counts in hospitals across the five most populous counties in California. Further, we aimed to find beneficial machine learning models to utilize in predicting hospital ratings and risk adjusted rates. These models could be useful for future hospitals to use when they are predicting their own risk adjusted rate or hospital rating. This investigation is important from a public health standpoint, and provides helpful insight into hospital functioning in the California area. Understanding which hospitals have higher deaths and cases, as well as how hospital ratings affect cases, deaths, and re admissions, would help policy makers identify hospitals in need of improvement, and provide recommendations of how to improve patient care. (Helen and Rachel)

2. Methods

2.1. Pre-Analysis

The following sections outline our pre-analysis plan for our dataset.

2.1.1. Refining the Research Question

First, we used barplots to review associations within our dataset. We also used a correlation matrix, but the result demonstrated no immediately strong correlations among the numeric variables within the dataset.

After reviewing some associations in the data, indicators of a relationship between hospital ratings and risk adjusted rate, number of cases, and number of deaths/readmissions were found. Based on these findings, the research question was refined to: Predict hospital rating (“as expected”, “worse”, “better”) from hospital location, year, risk adjusted rate, deaths, readmission, and case count. We used this question as the frame for applying machine learning models we discussed in class. (Rachel)

2.1.2. Data Splitting Plan

For our data splitting, we utilized an 80/20 train test split through a def(maxmin) function, creating target and feature variables, and using the ”from sklearn.model selection import train test split” in our decision tree classifier and k nearest neighbors. For the random forest models, we used train size = int(.8*N) and then subset df [0:train size] in order to split the data into training and testing rows. We did not split the data for multiple linear regression. (Rachel)

2.1.3. Feature Selection

Prior to running any machine learning, certain categorical variables were one-hot encoded. Variables that were one-hot encoded included County and Measure (mortality or readmission). Measure was a particularly simple one to one-hot encode conceptually because of the binary nature of the variable. Similarly, readmission was a binary categorical variable, and thus one-hot encoding was simple and intuitive.

We attempted to one-hot encode by hospital, but it quickly proved to be an ineffective way of dealing with the data due to the large number of hospitals in the dataset. As hospital was no longer a good predictor of location, county was a far better location-based variable to one-hot encode as there were far fewer counties than hospitals in the dataset. Thus, the original dataset contained three variables related to location (coordinate points, county, and hospital) but only the county variable proved to be useful. Risk adjsuted rate, number of deaths/readmission, and number of cases were the numeric variables we used. (Helen)

2.1.4. Objectives

There were two main objectives for this project:

Objective 1: Create a model that predicts ”Hospital Ratings” based on the predictor variables.

Objective 2: Create a model that predicts the ”Risk Adjusted Rate” based on the predictor variables. (Helen)

2.1.5. Model Selection Plan

We used a classification model for addressing objective 1 because the predicted variable was categorical. We used a multiple linear regression model for objective 2 because the predicted variable was numeric.

Specifically for objective 1, a decision tree classifier model was used because the measure variable was a categorical variable. We will also use a K-nearest neighbor for objective 1 because these models can handle categorical targets, categorical predictors, so long as they are one-hot encoded (which was done previously). Lastly, we used a random forest in order to predict hospital ratings on an ensemble of trees, which gave us a good prediction as lots of trees are used. We used bootstrapping in our random forest here.

Tree based models (random forest) were used for objective 2. This was helpful in predicting risk adjusted rates with an average of many trees. We used bootstrapping in this model. We also did a multiple linear regression model so we would have something to compare the random forest to. (Helen)

2.1.6. Evaluation Strategy

We evaluated our models with accuracy to see how how the various models perform – random forest (hospital ratings), k nearest neighbors, and decision tree classifier. This entailed using ”from sklearn import tree”, making a decision classifier and fitting the classifier, making predictions on the test set, then completing a confusion matrix with an accuracy formula. Further, for our multiple linear regression and random forest (risk adjusted rate), we used r squared to assess how well our model performed. (Rachel)

2.2. Analysis

2.2.1. Model implementation

Decision Tree Classifier: We used a classification model to measure the probability of a hospital being rated “as expected”, “worse”, or “better” based on the independent variables we gave it. To set this up, we used a def(maxmin) function and set our x and y. Our x was our features, which we chose to be “Risk Adjusted Rate”, “# of Deaths/Readmissions”, “# of Cases”, “Year”, “County”, “Hospital”, and “Measure”. Our target variable y was Hospital ratings. Next, we selected the numeric x variables for normalization, and recombined them with the categorical x variables. Then, we split the data on an 80/20 train test split. Next, we one hot encoded all of the categorical variables (County, Measure, and Hospital). Then, we imported the decision tree classifier, created the classifier object, fit the classifier, and made predictions on the test set. We finally computed accuracy to evaluate the model. (Rachel)

Random forest: This model was more complex. Because this model was used to explore the second objective, the target variable was changed to be “Risk Adjusted Rate”. The model resulted in a large quantity of decision trees. The random forest averaged the results. This was more accurate than just one decision tree. To build this model, we first split the data into training rows and testing rows. We then removed the rows with the target to finalize our df_train and df_test. Our next step was bootstrapping where we set T=1000, and we split the data again into an X_train and y_train, X_test and y_test, where the y’s used arcsinh, and the x code dropped the risk adjusted rate. We then re-one hot encoded our categorical columns because we were having issues with our code. We then used a for loop to generate a bootstrapping sample, computed rsq, and made/saved predictions. Our next step was building an ensemble predictor, and we printed our rsq. For hospital ratings, we applied the same methods, with assistance from AI to adjust the code from arcsin to a classifier random forest.

We used random forests for classification on hospital ratings. This included bootstrapping. We built it by defining our target variable y and our features x (all variables except year and OSHPDID), one hot encoding the categorical columns, and making an 80/20 train test split on the data. Then, we imported RandomForestClassifier, fit the classifier, and printed the confusion matrix and accuracy. (Rachel)

Multiple Linear Regression: This was used as a comparison model to the ensemble random forest, and it predicts risk adjusted rate utilizing the other numeric variables in our dataset as predictors. We first dropped NA’s from our numerical variables as it was causing issues in the code. Next, we used def mlr(x,y) to build our multiple linear regression function. We set our intercept equal to one, and set our X to be our intercept + predictors, and our y to be Risk Adjusted Rate. Finally, we printed our MLR coefficients and r-squared. (Rachel)

K Nearest Neighbors: This model was used to determine what class (Hospital rating: “as expected”, “worse”, or “better”) a new case will belong to based on the predictor variables. We first created the target and predictor variables. We used get_dummies to address non-numeric variables. This produced boolean values, so we adjusted the dummy variables to be 0 and 1 rather than “True” and “False”, allowing us to then create the KNN model. Then, we split the data 80/20, and scaled the X_test and X_train so that values would be comparable. Next, we needed to determine the ideal k value using SEE across possible values of 1-50 for k. The ideal K was 1 to minimize classification error. This is a bit unusual, but it is possible that the ideal k can be 1 when the data is clean and lacks excessive noise, which is the case with this dataset. However, this model is not the most ideal as it may be prone to overfitting. (Helen)

2.3. Architecture/structure

Number of trees in random forest: We set T=1000 for the ensemble random forest, which predicted risk adjusted rate. **Optimal k for KNN**: k=1 (Helen and Rachel)

2.4. Pre-processing Steps

Train/Validation/Test split: We split the data into an 80/20 train test split, which allows us to train a substantial portion of the data, while the other portion was used to evaluate performance. We used this method in our decision tree classifier, our random forest for hospital ratings, and our K nearest neighbors model. We followed the code from the random forest classifier class to split our data for random forest. (Rachel)

Scaling: We scaled the data for the KNN model so that their values would be comparable. Once this was completed, we were able to determine the K value that minimizes test classification error, and make the KNN model. We also used maxmin functions for our decision tree classifier. (Helen)

One hot encoding: We one hot encoded measure (readmission/mortality), hospital rating, and county to use in our models. (Rachel)

Feature Selection: The features we used to build our model were ’Risk Adjusted Rate’, ’# of Deaths/Readmissions’, ”# of Cases”, ”Year”, ”County”, ”Hospital”, and ”Measure” if we were predicting hospital ratings. We used these measures because we saw some interesting relationships in our preliminary graphs and wanted to incorporate these variables into the model. For our models predicting predicted risk adjusted rate, we used all of our variables in our dataset in the random forest ensemble (for risk adjusted rate) because we were curious to see the impact of keeping them all would affect ensemble rsq. The features we used in our MLR model included all of our numeric variables, except for our target variable, risk adjusted rate. (Rachel)

3. Results: Evaluation Benchmark

3.1. Metrics Used

Accuracy: The accuracy for our decision tree classifier was 0.93.
(Rachel) The accuracy for our KNN model was 0.93. (Helen) The accuracy for our random forest (hospital ratings) model was 0.926. (Rachel)

Confusion Matrix

Table 1. KNN Confusion Matrix

Actual / Predicted	Class 0	Class 1	Class 2	Class 3
Class 0	189	2	0	0
Class 1	6	1	0	0
Class 2	1	0	0	0
Class 3	5	0	0	2

(Helen)

Table 2. MLR Confusion Matrix

Actual / Predicted	Class 0	Class 1	Class 2
Class 0	182	4	1
Class 1	4	6	0
Class 2	7	0	1

(Rachel)

R-squared: Our ensemble rsq for the ensemble random forest was 0.76, and our rsq for MLR was 0.0056. (Helen and Rachel)

3.2. Model comparison

The KNN model had the best accuracy. As aforementioned, however, the k value with the least test classification error was 1, which makes this model more prone to overfitting. For predicting risk adjusted rate, random forest has a far better r-squared value compared to the multiple linear regression model (0.76 and 0.0056, respectively). (Helen)

3.3. Initial Results and Comparison

Interpretation of which model performed best:For models predicting hospital ratings, k nearest neighbors performed the best in regards to accuracy, but the decision tree classifier accuracy was very close to k nearest neighbors. In addition, the decision tree classifier is less prone to overfitting compared to the k nearest neighbors because our optimal k is 1 for the KNN. Further, from our random forest model, we had an accuracy of 0.926, which was higher than the decision tree classifier, and therefore our best model.

For models predicting risk adjusted rate, MLR performed poorly with a low rsq. The random forest performed better with a significantly higher rsq.

4. Initial Results Analysis

The random forest and decision tree classifier models performed better than the KNN and MLR models. Regarding the KNN model, this model was not as ideal, though it has a high accuracy (0.93) due to the fact that the ideal k for minimizing test classification error was k=1. This model, therefore, is prone to overfitting. The model presenting these characteristics because the cleaned data set was large, clean, and lacked excessive noise. MLR performed poorly because a linear model simply failed to capture the variance in the data. The trends were more complex. The random forest model performed better because this model is non-linear and random forests account for more complex patterns. (Helen)

The random forest was likely the best when it came to predicting hospital ratings, as it takes multiple trees and allows them to “vote” on the outcome, so there are many more trees contributing to the results. (Rachel)

5. Conclusion

5.1. Summary

In this project, five total models were created to address the following questions: how can hospital ratings and risk adjusted rate be predicted based off predictor variables such as county, number of deaths/readmission, number of cases, etc. This question was split into two main objectives, the first predicting hospital ratings, and the second addressing risk adjusted rate.

To address the first question, a K Nearest Neighbor model was used and resulted in an accuracy of 0.93. The ideal K was 1, however, indicating that the model might be prone to overfitting. A Decision Tree Classifier was also used to address this question, and performed well while also being less prone to overfitting. A third model, a random forest, was also used and ended up with the highest accuracy other than the KNN model. Thus, for addressing this question, the random forest model was the best model for predicting Hospital Ratings based on this dataset.

To address the second question, a Random Forest model was used in addition to multiple linear regression. The MLR performs poorly as shown by the low r squared value. The random forest, similar to the first question, proved to be the best model to predict the risk adjusted rate. (Helen and Rachel)

5.2. Defense

5.3. Future & Additional Work

Future work related to hospital stats and rating /risk adjusted rate outcomes could expand the number of variables included. For example, hospital ratings could be further expanded to satisfaction with individual providers or satisfaction with divisions. More variables looking at similar questions would allow future work to conduct principal component analysis and potentially generate a more cohesive and effective prediction.

Future work could also analyze hospital ratings or risk adjusted rate over time in response to policies that effect hospital regulations. The data, as is, simply describes the current situation of hospitalizations within the five most populous counties in California. More data over time would create a more complex picture responding to current events. For example, legislation limiting the kinds of medications available, or medical care coverage would likely have an impact on hospitals satisfaction rates and risk adjusted rates over time. Work such as this would provide legislators with data regarding the impacts of the policies enacted, and provide more direction for moving forward. This would allow future work to be more socially engaged and provide more directive power for change on a broader institutional scale. (Helen and Rachel)

5.4. Issues and Limitations

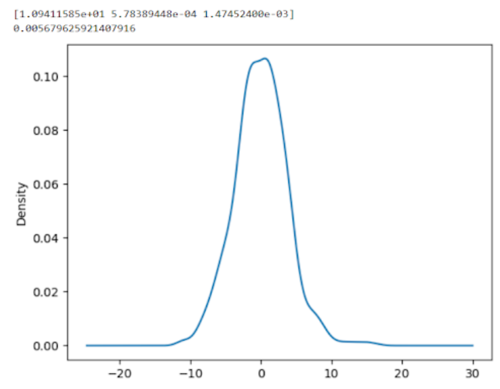
There were initially some complications with the random forest. To be more specific, when predicting hospital ratings with random forests, we needed to change the arc sin h. However, we were able to remedy this as we found a different code syntax in the classwork for random forest classifiers of categorical variables, which still produced a fantastic accuracy and used a standard 80/20 test train split. Repeated errors were corrected by making slight adjustments to the one-hot encoding and the order of the code. This resolved the errors. Further, we planned to transform the risk adjusted rate variable for multiple linear regression because it was skewed to the right. After multiple attempts at transformation, including log, sqrt, and cubed root, the r-squared only reached .02. We concluded that this model is not a good fit for our data. (Helen)

References

Department of Health Care Access and Information. Ischemic stroke 30-day mortality and 30-day readmission rates and quality ratings for ca hospitals. https://catalog.data.gov/dataset/ischemic-stroke-30-day-mortality-and-30-day-readmission-rates-and-quality-ratings-for-ca-h-92036, 2025. Data set, accessed 2025-12-10.

(Department of Health Care Access and Information, 2025)

6. Appendix



```
[1.09411585e+01 5.78389448e-04 1.47452400e-03]
0.005679625921407916
```

Figure 1. MLR model results

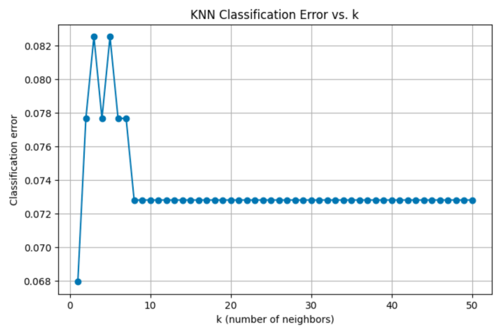


Figure 2. KNN Classification Error vs. k

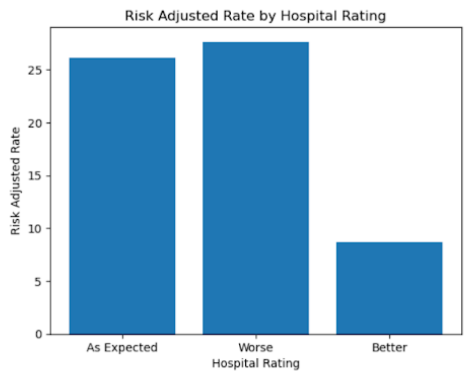


Figure 3. Risk Adjusted Rate by Hospital Rating

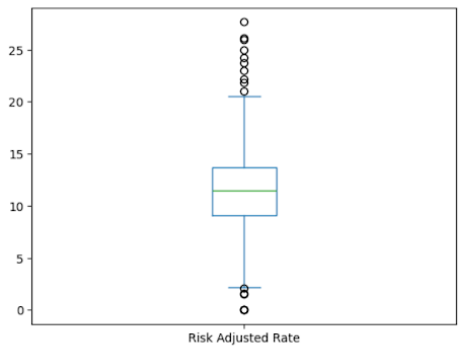


Figure 4. Boxplot for Risk Adjusted Rate

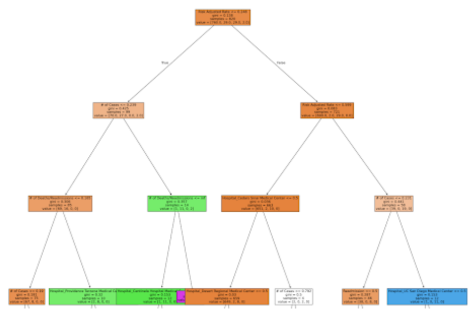


Figure 5. Tree for Decision Tree Model