Spring 2025

Stroke Risk – Project ISYE 7406

Joshua Mestemacher
Olan Malcolm
Phillip Sley
Syed Asghar

Abstract

Stroke is a leading cause of disability and death worldwide, with early identification being critical to prevention. This project investigates whether machine learning and statistical modeling techniques can accurately predict stroke risk based on clinical and behavioral factors. Using the publicly available Stroke Risk Prediction Dataset, we examined relationships between 16 health indicators and stroke likelihood, exploring both classification (At. Risk) and regression (Stroke.Risk.Percentage) outcomes. Data preprocessing included class rebalancing, correlation analysis, and feature selection to address imbalanced and potentially synthetic data.

A wide range of models were applied, including logistic regression, Firth's regression, LASSO, Ridge, Elastic Net, PCA + Ridge, K-Nearest Neighbors, Naive Bayes, Decision Tree, Random Forest, Gradient Boosting, and Beta Regression. Models were evaluated using metrics such as accuracy, sensitivity, specificity, R^2, MSE, and RMSE. For classification models, Sensitivity was prioritized given the medical context, where missing high-risk individuals could have serious consequences.

Our findings suggest that stroke risk is influenced by a broad combination of symptoms rather than a few dominant predictors. Ensemble models like Gradient Boosting and Random Forest performed well, especially when strong predictors like Age were included, though performance declined without them. More traditional models such as PCA and Beta Regression also provided interpretable and stable performance. Ultimately, models using a combination of predictors offered valuable insights for triage tools or clinical decision support systems. This study highlights the importance of feature selection, dataset quality, and model interpretability in building effective medical risk prediction tools.

Introduction

Stroke is the number one leading cause of disability and one of the leading causes of death worldwide, affecting approximately 795,000 people in the United States each year (Tsao et al.). In the U.S., nearly 1 in 6 cardiovascular-related deaths are due to stroke (National Center for Health Statistics). Early identification of individuals at risk is essential for prevention and timely intervention. However, predicting stroke risk is inherently complex, as it depends on a range of demographic, behavioral, and medical variables, making accurate classification a major challenge.

This report aims to analyze key risk factors that contribute to an increased chance of having a stroke, and to develop predictive models using data mining techniques. Using the publicly available Stroke Risk Prediction Dataset, we explore relationships between variables such as age, chest pain, high blood pressure, and other medical predictors to the likelihood of having a stroke.

The primary data mining challenges we address include handling an imbalanced dataset – there were many more at-risk individuals than not at risk – and ensuring model generalizability without overfitting. Feature selection and working around the synthetic nature of our data also present significant hurdles.

To tackle these issues, we employed several problem-solving strategies. Our preprocessing included handling missing values, encoding categorical variables, and performing class rebalancing to address the dataset's imbalance. We also analyzed the correlation matrix to identify and remove highly correlated variables that could introduce redundancy or multicollinearity. PCA was also used to understand multicollinearity affecting prediction, if at all. Further, we applied feature selection to isolate the most informative predictors for stroke risk, ensuring that our models focused on the most relevant variables. We will explore this point further in the report.

Through this process, we have learned how different modeling approaches handle imbalanced medical datasets and the trade-offs between model complexity, interpretability, and performance. We also observed how ensemble methods may outperform simpler models in predictive accuracy, especially for cases where the prediction (having a stroke) is heavily influenced by a variety of complex variables (stroke predictors) while still offering insights into feature importance.

Problem Statement and Data Sources

Stroke is a medical emergency with potentially fatal outcomes, and early detection is key to reducing long term disability and mortality. However, predicting stroke risk is complicated by the presence of correlated variables and an imbalanced dataset, where there were many more at-risk individuals than not at risk of having a stroke.

There are a few questions we aimed to answer in this project:

1. Can we create a questionnaire for patient intake to appropriately triage patients for stroke?
2. Can we create an accurate model for predicting stroke risk assuming precise medical measurements have been done?
3. What are the most significant risk factors associated with a chance of having a stroke?
4. How does age impact the likelihood of having a stroke?
5. Are there any significant interactions between risk factors predicting stroke?
6. What are the result differences between traditional additional methods (regression) vs machine learning methods (random forest, gradient boosting)?

The data used for this analysis comes from the "stroke_risk_dataset.csv" file, a publicly available dataset on Kaggle compiled to study factors contributing to having a stroke. The dataset contains 70,000 observations and 18 variables. The primary outcome variables are

At.Risk and Stroke.Risk.Percentage. At.Risk is a binary variable where 1 denotes an individual classified as at risk of having a stroke, and 0 being the opposite. Stroke.Risk.Percentage is the actual percentage the individual is at chance of having a stroke. If an individual's percentage was greater than 50%, then they were deemed at risk. All of the predictors would be quickly diagnosable for a patient in triage at a medical facility. Below is a description of the predictors and response variables used in the data set:

Chest Pain	Binary (0/1): Indicates whether the individual experiences chest pain, a common symptom of cardiovascular conditions.
Shortness of Breath	Binary (0/1): Represents whether the person has difficulty breathing, which may indicate heart or lung problems.
Irregular Heartbeat	Binary (0/1): Shows if the person has an irregular heartbeat, a potential stroke risk factor.
Fatigue & Weakness	Binary (0/1): Indicates persistent fatigue and muscle weakness, common signs of cardiovascular issues.
Dizziness	Binary (0/1): Reports whether the individual frequently experiences dizziness, which may be linked to poor circulation.
Swelling (Edema)	Binary (0/1): Indicates swelling in extremities due to fluid retention, a potential cardiovascular issue.
Pain in Neck/Jaw/Shoulder/ Back	Binary (0/1): Describes pain in these areas, which can be a warning sign of stroke or heart attack.
Excessive Sweating	Binary (0/1): Shows whether the individual experiences unusual sweating, which may indicate cardiovascular distress.
Persistent Cough	Binary (0/1): Indicates chronic coughing, which can be associated with heart failure.
Nausea/Vomiting	Binary (0/1): Reports frequent nausea or vomiting, which may be linked to cardiovascular events.
High Blood Pressure	Binary (0/1): Represents whether the person has high blood pressure, a major risk factor for stroke.
Chest Discomfort (Activity)	Binary (0/1): Shows if the individual experiences chest discomfort during physical activity.
Cold Hands/Feet	Binary (0/1): Indicates whether the person often has cold extremities, a possible sign of circulation problems.
Snoring/Sleep Apnea	Binary (0/1): Reports whether the individual has sleep apnea, which can increase stroke risk.
Anxiety/Feeling of Doom	Binary (0/1): Captures whether the person experiences frequent anxiety or a sense of impending doom, which can be related to cardiovascular distress.
Stroke Risk (%)	Continuous (0-100%): The estimated percentage risk of having a stroke, based on symptom severity and medical indicators.

At Risk (Binary)	Binary (0/1): Indicates whether the person is classified as at risk of stroke (1) or not (0).
Age	Integer: The age of the individual, an essential factor in assessing stroke risk.

Variable Descriptions

Some exploratory data analysis (EDA) was done on the data set. For starters, the total number of each categorical variable was looked at (*Supplementary Appendix Figure 1*). Then, a histogram was made for all non-categorical variables. Stroke.Risk.Percentage was almost normally distributed, and Age did not really have any sort of distribution to it (*Supplementary Appendix Figure 2*). Boxplots were also made for each variable to see if there were any outliers in the data set. Stroke.Risk.Percentage was the only variable with outliers detected, with values both under the 25th percentile and above the 75th percentile. We decided to keep all the outliers in the data set as it did not make sense to us as a group to remove them. Looking at the amount of At.Risk = 1 and At.Risk = 0, we found that there were 24,556 total 0's and 45,444 total 1's. To fix this imbalance in our dataset, we took a sample of the total 1's to equal the amount of 0's there were such that there were equal amount of At.Risk and Not At.Risk individuals in our dataset (*Appendix Figure 1*). As a result, when running our models, we ran them with 24,556 total 0's and 24,556 total 1's.

We did a simple correlation matrix between all variables and At.Risk predictor (*Appendix Figure 2*). Strong positive correlation between Age and the outcome variable At.Risk.Binary existed. We saw this as an alarm as a single predictor that has an extremely strong relationship with the binary outcome, is a warning sign that it might be able to perfectly or near-perfectly predict some outcomes. All symptom variables appear to have positive correlations with the outcome. When multiple predictors all point in the same direction, they can collectively create a pattern that perfectly separates the classes. The plot also shows no negative relationships, and the ones that exist are relatively clear and distinct without much noise. In real-world health data, one would see *messier* relationships where certain factors might negatively correlate with the outcome. This "too clean" pattern can indicate that the dataset might have been generated using certain rules or criteria rather than capturing the natural variability found in real patient populations – points to potential synthetic nature of our dataset. We kept Age as a predictor in our dataset to test the models we hoped to run and got nearly perfect results. We confirmed this by further running a few models on a 30% train and 70% test set, to still get perfect results. This confirmed that Age was indeed a perfect predictor of stroke risk. Due to this, we found it best to remove it as a predictor and use the rest of the variables for our models.
To use this dataset for consequent modeling, we split 70% of the observations into training data and the remaining 30% was the testing data.

Proposed Methodology

To predict if an individual is at risk of having a stroke, we explored both traditional statistical and machine learning methods. When using the classification methods, we

dropped the Stroke.Risk.Percentage column and used At.Risk as our response variable. When using regression methods, we dropped the At.Risk column and used Stroke.Risk.Percentage as our response variable. Each method was chosen due to their different approaches in learning patterns from data, and we utilized cross validation on the training data set to avoid overfitting the models. Our overall approach was to perform some baseline testing with our chosen models on the training data, understand the results, and rerun the same models with chosen set of features based on some feature engineering methods. We then used k-folds cross validation to ensure consistency in our model results. K-fold cross-validation divides the training dataset into *k* equal-sized subsets, or "folds" - in this case, 10 folds (*Supplementary Appendix Figure 3*). The model is trained on 9 of the folds and validated on the remaining one. This process is repeated 10 times, with each fold taking a turn as the validation set exactly once. The final performance is typically averaged across all iterations to provide a more reliable evaluation of the model. Below, we discuss the reasoning behind the usage of each model and why they were tuned if applicable.

Logistic Regression

Starting off with a baseline model as logistic regression made sense. The primary objective of our project is to classify individuals as either "at risk" (1) or "not at risk" (0) of experiencing a stroke, based on 16 clinical and behavioral predictors. Logistic regression is a well-established method for binary classification tasks and is suitable when the outcome variable is categorical with two levels. It provides interpretable coefficients, which is useful in a healthcare context where understanding risk factors is essential. Logistic regression was fit on the full training dataset using all 16 predictors including age. The model coefficients provided an initial understanding of the influence of each predictor. However, the presence of high multicollinearity (VIF > 100 for many variables) raised concerns about instability in the estimated coefficients and overfitting. The performance metrics were also unusually high, which indicated potential overfitting or data leakage. Also, when Age was removed, the model returned VIF values of nearly 1, indicating there was not really any correlation between the remaining predictors (*Appendix Figure 3*). Logistic regression was used as a baseline model due to its simplicity, interpretability, and ability to provide quick insights into variable importance. Recursive feature engineering was also performed with 10-fold cross validation to get insight into the most important predictors. The results of cross validated accuracy and number of predictors used are found in *Supplementary Appendix Figure 4*. This, when compared against all other models used gave us a good understanding of the medical predictors we might want to focus on.

Firth's Regression

When we first considered to run Firth's regression, it was because of the overfitting and bias seen in our results for logistic regression including Age. The model was not converging, i.e. the model was not able to compute finite coefficients. Firth's regression is used a bias-reduction technique by using a penalty term in the standard Logistic regressions MLE to find the best fitting coefficients. This penalty is helpful to converge the estimate to finite coefficients – which is extremely helpful for healthcare data that might

have strong correlated predictors for a given problem. It is especially helpful in the context of complete or quasi-complete separation – as Age did in our dataset. We also had class imbalance on the At.Risk response variable, which is where Firth's is also helpful. However, since we balanced our dataset, this was not an issue anymore. While Firth's performed similarly to Logistic regression, we included it anyways to validate our baseline model, and as it inherently reduces overfitting.

LASSO and Ridge Regression

One of our goals, as mentioned earlier, is to be able to have a dependable model that performs well on unseen data i.e. a model that has good generalizability. To build on the basic regression models, we used LASSO regression that applies L1 regularization, which minimizes the sum of absolute coefficient values, effectively selecting important features. LASSO helps prevent overfitting by shrinking large coefficients and enhances model simplicity but may discard some useful predictors – which it did not in our case. It also helped identify the most critical symptoms for predicting stroke risk. Although LASSO reduces overfitting by setting insignificant feature coefficients to zero; it did not do this for our dataset – again citing that the model finds all 15 predictors in our dataset as somewhat influencing the stroke risk prediction strongly. LASSO was implemented using cross-validation to select the optimal λ value.

Ridge regression applies L2 regularization, minimizing the sum of squared coefficients to prevent overfitting while maintaining all features. It stabilizes coefficient estimates, and improves predictive accuracy when many predictors are correlated – which was something we were keen to find out for our stroke risk predictors. While slightly more complex than LASSO, its interpretability is still high and it generalizes well by preventing extreme coefficients but does not perform feature selection. For our model, it retained all symptoms as predictors, stabilizing their importance.

Whether Ridge or Lasso, the optimal lambda chosen via CV was very close to zero. That means almost no regularization is applied and the resulting models approximate to the plain baseline logistic regression model, just with tiny weight shrinkage. We will elaborate this more in the next section. See *Supplementary Appendix Figure 5* for feature importance by both methods.

Elastic Net Regression

Elastic Net regression was also used as a hybrid approach between LASSO and Ridge regression. It combines L1 (Lasso) and L2 (Ridge) regularization to balance feature selection and coefficient shrinkage. It generalizes well in cases with correlated features and helps balance between overfitting and underfitting. We explored with this method hoping that it captures complex relationships between correlated predictors – given the issue of perfect separation and synthetic nature of dataset. See *Supplementary Appendix Figure 6* for feature importance by this method.

Principal Component Analysis (PCA) + Ridge Regression

For further exploring model stability and generalizability, we applied Principal Component Analysis (PCA) for dimensionality reduction, followed by Ridge Regression for classification. PCA transforms the original predictors into a set of uncorrelated principal components that capture the maximum variance in the data. Rather than feeding highly correlated original features into a model, PCA allowed us to use a smaller set of orthogonal components as inputs. Since ridge regression performs well when all variables are retained but struggles when they are collinear, using PCA as preprocessing addressed this weakness. The strategy still leveraged all available information, while minimizing overfitting. All predictors were standardized before applying PCA. After PCA was applied, a subset of top principal components (explaining ~75% of the variance) were retained. We started with a 90% cumulative variance explained via PCA, but brough it back to 75% when we saw minuscule changes in the ridge model accuracy. These components replaced the original features as input into the ridge regression model. While PCA reduces model interpretability, it provided a robust benchmark for model comparison in our case.

Beta Regression

Beta regression is like a logistic model; however, it uses a continuous distribution for the response ranging from 0 to 1. For this purpose, unlike some of our other models, we were required to use Stroke.Risk.Percentage as our response variable as it is continuous. The model was chosen as it can handle asymmetry and heteroskedasticity as it is a non-linear model. Additionally, some variables that may not be significant in a linear model may be significant for Beta regression. For this model, we divided the response by 100.
A downside to beta regression is that a response of "0" or "1," will break the model. As some of our response variables were "1," we subtracted 0.000001 from our response variable to mitigate the model from breaking. No instances of the response variable include a "0" response.

Decision Tree

A decision tree is a supervised learning algorithm, which means it is trained using data where the outcomes are already known. It is commonly used for making predictions, such as estimating a number or choosing between categories. In this case, because we are predicting a percentage, we used Stroke.Risk.Percentage as the target variable instead of a binary label like At.Risk. The tree works by asking a series of questions about the data – such as "Is age greater than 50?" – to split it into smaller, more similar groups. Each split is chosen to reduce the variation in stroke risk within each group. This process continues until the data cannot be split further in a meaningful way, and the final predictions are made at the leaves of the tree. After the tree is built, it is often pruned, or simplified, by removing unnecessary splits to improve performance and avoid overfitting. Pruning was done by finding the simplest complexity parameter (cp) using cross validation, which sets a threshold for how much a split must improve the model to be included.
We decided to use a decision tree for this dataset because they produce easily interpretable results that are also easy to visualize (*Supplementary Appendix Figure 7*). In the case of this dataset, one could show an individual a decision tree and have them follow if they have/do not have the symptoms on the tree to show their chance of having a stroke.

A decision tree was also used because it both handles nonlinear relationships well and handles different types of data (numerical and categorical) without requiring extensive preprocessing.

Random Forest

A random forest is an ensemble learning method (a machine learning technique that combines predictions from multiple models) that combines multiple decision trees to improve prediction accuracy and reduce overfitting. It works by averaging the predictions of multiple decision trees trained on different subsets of data. It is more robust than a single decision tree which reduces the variance of the model compared to the decision tree. It also can maintain the interpretability of each variable. One drawback of a random forest compared to a decision tree is that it is not easy to visualize as it is a combination of many trees. This model was used because it provides higher accuracy and stability compared to a single decision tree. It reduces overfitting by averaging multiple models, leading to better generalization compared to a single tree. Like a single decision tree, a random forest also handles nonlinear relationships effectively. In terms of this dataset, this model can model complex interactions (for example the multiple factors in this dataset) without requiring explicit assumptions.

Random forests have hyperparameters that can be tuned, which in this case two were used – mtry and ntree. Ntree is the number of decision trees used for the aggregation of the random forest. Due to long computation if any larger, we decided to set ntree = 100 which is considered a standard amount. Mtry is the number of features to consider at each split. Instead of setting mtry to a set number, a sequence of numbers was testing, and the model was then reran with the best mtry number it found. Mtry was found by starting at 1 then going in a sequence by 1 to the $\sqrt{Number\ of\ columns}$ rounded down to the nearest whole number. Depending on the number of columns used, mtry came out to be either (1, 2, 3) or (1, 2, 3, 4). K-folds cross validation with a k value = 10 was used to find the best mtry value out of the sequence.

Gradient Boosting

Gradient Boosting is an ensemble machine learning model that works in an iterative fashion where multiple decision trees are built sequentially. A “boosting” process happens where each decision tree is built on correcting the errors of the previous one, which thereby increases the accuracy and performance of the overall model. Each tree predicts the errors of the previous one which represent the gradient of the loss function for it (thus the name “gradient boosting”). Gradient boosting is more robust than a single decision tree, just like Random Forest, having high predictive accuracy while maintaining interpretability, though it is not as easy to visualize as a decision tree. We choose this method as it is known to have relatively high accuracy in general, and can work well with complex, non-linear relationships, which our data may contain. Also, we can obtain feature importances from it, which can help with feature selection. The main disadvantage for it as already stated is that it cannot be visualized very well.

Gradient Boosting has hyperparameters that can be tuned, and the hyperparameters we tune for it here are the number of trees used, the max depth of any tree built, the learning rate, and the minimum observations in a terminal node. The values we tried for number of trees were 100, 200, 400, 600, and 1000, while the values we tried for max depth were 1, 3, 5, and 7 nodes. For learning rate, we tried 0.01, 0.1, and 0.2, while for the minimum number of observations in a terminal node we tried 3, 5, and 7. We used 10-fold cross-validation for tuning these hyperparameters.

K-Nearest Neighbors (KNN)

KNN is a machine learning technique that operates on the principle that similar data points are likely to have similar outcomes. It works by getting the response for the k nearest neighbors to each point, and for regression, averages them to get that point's prediction. A distance metric, commonly Euclidean distance (we use this here), is used to find the k nearest points. The value of k is a hyperparameter that must be set manually by the user. We use 10-fold cross-validation to tune it, and the values we try out for k are going from 3 to 31 in increments of 2 inclusive (3, 5, ..., 31). KNN does not have an explicit model it learns while training, but rather for each point uses the k nearest points in the training set to get a prediction for it. Also, KNN requires standardization of the features beforehand to prevent factors with high magnitudes from unduly influencing the results. We choose KNN as we think that similar patients are likely to have similar stroke risks. Also, the simplicity of the model will make it easier to explain to medical professionals when we are trying to convince them to use our results here. We choose to use the regression version of it as we think that getting the exact stroke risk for a patient will provide us with more meaningful information.

Naïve Bayes

Naïve Bayes is a probabilistic machine learning model that assumes all variables are independent of each other. This method was chosen as it is simpler than other models by using independent probabilities and may uncover unknown relationships between our features and stroke risk. This is important both in feature selection and determining what features may be most useful to know when time is limited for medical interventions by allowing professionals to know what to look for in Stroke Risk. The downside to this model keeping variables independent is that it does not account for relationships between variables, which could lead to overweighting variables that are not truly relevant, impacting specificity and precision.

Feature Selection

With all the methods run above, we ranked the stroke risk features according to our model results. The features in bold below were used to re-run our models as optimal features we wanted to check model performance against stroke risk prediction. We did this by also comparing results between Top 8 and Top 9 risk predictors, where the 9th predictor variable is Irregular heartbeat. Given at triage, patients might not have all symptoms, we wanted to choose the 'best' possible stroke risk predictors to have respectable results. The results and comparison will be discussed next.

	Feature Rank for each method								
Feature	Logistic Regression	Ridge Regression	LASSO	Elastic Net	Naïve Bayes	Decision Tree	Random Forest	xgBoost	Beta Regression
Cold Hands Feet	1	1	1	1	9	2/3 & 4	1	1	1
Persistent Cough	8	8	8	8				7	
Fatigue Weakness	7	7	7	7	3		9	9	9
Chest Pain	9	10	9	9	1	1	4	2	2
Excessive Sweating	5	5	5	5	8		3	3	3
High Blood Pressure	4	4	4	4	5			4	4
Anxiety feeling of Doom	2	2	2	2	4	2/3	10	5	6
Nausea Vomiting	6	6	6	6			6	6	5
Sleep Apnea	3	3	3	3	2		8	7	7
Dizziness					6				
Irregular Heartbeat	10	9	10	10	7	2/3			8
Shortness of Breath							5	10	
Swelling Edema							7	8	

Feature Rankings Table

Analysis and Results

The regression models as seen in *Results Table 1* below yielded relatively similar but informative outcomes. The logistic regression without Age seems to perform best in terms of accuracy, closely followed by the PCA + Ridge model. Interestingly, Ridge, LASSO and the Elastic Net models converged to the logistic regression solution, resulting in nearly identical performance metrics. This was due to the very small optimal Lambda found during tuning (i.e., little to no regularization was applied), the balanced dataset that we ended up using and the removal of Age, which was the primary driver of multicollinearity. In medical context, sensitivity (true positive rate) is extremely important. It reflects the model's ability to identify truly at-risk risk individuals, which is vital in a high stakes situation like stroke risk triage. Missing a high-risk patient can have severe and possibly fatal consequences. As a result, models that prioritize high sensitivity, even at the cost of some false positives, are more clinically appropriate. In this light, logistic regression without Age and PCA + Ridge offered strong sensitivity values, with others performing comparably well. While sensitivity was prioritized due to the medical context, models also maintained reasonable levels of specificity (ranging from 56% to 79%), reducing the risk of excessive false positives and ensuring clinical practicality. The F1 scores and AUC values were also consistently strong across top models, confirming a balanced trade-off between sensitivity and precision.

A particularly insightful observation emerged from the PCA-based approach. The selected principal components, which explained 75% of the cumulative variance, when analyzed using the sum of absolute loadings from the rotation matrix, showed that all original health

predictors contributed meaningfully (*Appendix Figure 4*). This explains why the PCA + Ridge model's performance mirrored that of logistic regression: whether using original variables directly or transforming them through PCA, the models are capturing the same underlying signal. The relatively even importance distribution indicates that stroke risk prediction does not depend on just a few key symptoms, but rather on patterns across multiple indicators. This also reinforces that stroke risk, at least in this synthetic dataset, is not driven by a small subset of features but rather reflects a broad, multifactorial interplay of symptoms.

For Naive Bayes (*Results Table 2*) we had mixed results between the 3 different runs of data using the top 8 predictors, top 9 predictors, and all predictors except Age. While our accuracy was highest with all predictors except age at ~72%, our Sensitivity was ~66%, limiting its value for a questionnaire to diagnose Stroke risk. Nevertheless, this model was able to obtain a 78% specificity, meaning it could relatively accurately rule out Stroke risk, which may have some medical use. With only 8 predictors, we were able to obtain the best sensitivity at ~75%; however, it only achieved ~66% accuracy and ~56% specificity. The top 9 predictors model ended up with results in between both (Neither the best or worst sensitivity, specificity or accuracy). Cross validation was performed and had similar results, confirming the models were not overfitting. It may be useful to use all predictors except for age to rule out Stroke risk due to the strong specificity, while the Top 8 predictors may be used for an initial intake to determine if someone has stroke risk due to its strong sensitivity. For a more comprehensive list of results, please refer to *Results Table 2*.

For Beta regression, we used Stroke Risk Percentage, instead of "At Risk" for our response variable, given beta regression is not a categorical model. Beta regression also does not respond well to exact values of 1 or 0 for a response, therefore we needed to manipulate the data by subtracting 0.00001 from our response variable to ensure any instances of "1" would not break our model. Overall, the model ended up performing well with an MSE and RMSE of ~111 and ~10.5 for our train and test data sets for using all variables except for age. Additionally, the R^2 was ~0.48 for our test and train data sets using all predictors except age, indicating that most of our features were influencing the model. For our model runs of beta regression with top 8 predictors and top 9 predictors, we had decreased performance with a MSE of 155.93 for train and test data of our Top 8, and a MSE of 150.1071 for train and test for top 9 predictors. Cross validation for all 3 models reported similar results, indicating there is no overfitting for these models. For pure performance of these 3 models runs, we would want to select the Beta Regression model using all predictors except age provided the lower MSE and greater R^2. Nevertheless, for a questionnaire or medical purposes, it may not be feasible to ask or calculate all predictors of a patient, so if time is limited, we may want to use our top 8 predictor model as there was a de minimis decrease in performance from the top 8 vs. top 9 models with a reduction in RMSE of <0.25. For full details of results, please refer to *Results Table 2* in the results section.

Further, we had a decreased R^2 coming in at 0.2712 for train and test of top 8 predictors and 0.2984 for train and test of top 8 predictors. The R^2 is not unreasonable, but the MSE of these indicates these models did not perform as well. All performed almost evenly for MSE, with MSE being <1.1 and >1 for all. Our R^2 was highest with all predictors except Age at .4569. The results decreased with less predictors dropping to .3031 and .2755 for our top 9 and top 8 predictor models respectively. For Beta Regression, the R^2 values show that most of our predictors influence the model, and likely should be included. Given there is de minimis difference between the number of predictors of our three model runs and the resulting MSE for the models, we may want to opt for the Beta Regression model with all predictors except for age. This way, we can obtain a higher R^2, and an impressive MSE.

For KNN, we achieved an optimal value of k = 21 for the full model without Age, k = 31 for the model made using the top 8 predictors discussed above, and k=31 for the model made using the top 9 predictors discussed above. The KNN model for the full model without Age performed the best, having a test R^2 of around 0.43, which for medical data is relatively high (our results are shown in *Results Table 3* below). For the top 8 and 9 predictors, the performance for KNN decreases sharply, having a test R^2 of 0.25 for the former and 0.28 for the latter. These are, however, relatively okay for medical data. Limiting ourselves to only the top 8 or 9 predictors appears to worsen our results, suggesting that stroke risk is influenced by most of the predictors in our data set, not just a few, something we found also while doing PCA. Our train and test scores do not differ significantly for our KNN models going off R^2 and MSE, and the same is true for our CV train and test scores, which indicates that KNN does not have a problem with overfitting here. Our optimal values of k here indicate to us that using a relatively large number of similar patients for each subject in the data set is generally preferable. It is likely that trying out more choices of k for the top 8 and 9 predictors models could have produced better results as k = 31 was the max value tried there, though we doubt that we would have gotten anywhere close to what we achieved when using all the features. Overall, we find from KNN that determining the stroke risk of a patient using the stroke risks of other patients is a viable approach.

For Gradient Boosting, for our full model without Age we found that our optimal parameters were learning rate = 0.2, max depth = 1, minimum rows = 7, and number of trees = 300. For our top 8 predictors model, they were learning rate = 0.2, max depth = 1, minimum rows = 7, and number of trees = 153, and for our top 9 predictors model they were learning rate = 0.1, max depth = 1, minimum rows = 5, and number of trees = 364. The full model without Age performed the best, having a test R^2 of 0.48, which for medical data is very high, while the top 8 and top 9 predictors models performed the worst, having test R^2's of 0.27 and 0.30 respectively, which are relatively okay for medical data (the results here are shown in Results Table 4). The train and test R^2's do not differ significantly across the three gradient boosting models fitted, meaning that gradient boosting here does not have a problem with overfitting. We note that the NA's there are due to the gradient boosting function we used in R not allowing us to get the R^2's for the training portion of 10-fold cross-validation. Looking at the feature importances we obtain for our three models, shown in the supplementary files, no specific set of features seems to be dominant for

determining stroke risk. No feature is supreme in its importance for any of the feature importance graphs, and the features shown all have generally close importances, so as with KNN, we find that stroke risk is likely influenced by all the predictors in the data set, not just a small subset. We note that the features shown in the graphs are generally the same across the three models, only differing in where they are ranked. We speculate this may be because of gradient boosting here not having a large amount of variance in its results, though we do not have proof of this. For the optimal hyperparameters we found, a notable finding is that the optimal max depth is 1 for all three of our models. We think this implies that all the regression trees fit for gradient boosting do not end up needing a large number of predictors (in fact only one) to produce a "good" result overall. The simplicity here may be a reason for gradient boosting not having a problem with overfitting for us. Overall, we find that gradient boosting generally produces excellent results and is a very viable approach here.

Unlike the other models built on the full dataset including Age, the decision tree model did not achieve near-perfect performance and instead demonstrated significant limitations in capturing the complexity of the data (*Results Table 3*). The unpruned tree heavily relied on *Age* for its primary splits, first segmenting at Age < 57, and then further breaking into Age < 39 and Age < 72. These splits created distinct risk groups with Stroke Risk Percentages of 41%, 50%, 59%, and 68% across increasing age ranges, essentially modeling stroke risk as a step function of age. Even after pruning – using cross validation to select the optimal complexity parameter – the tree structure remained the same, and model performance metrics were relatively modest, with both training and testing R^2 values around 0.50 and MSE values near 100. This translated into an RMSE of about 10, meaning the model's predictions, on average, deviated from actual stroke risk percentages by about 10 percentage points. Cross-validation yielded no new improvement or tree structure, reinforcing that the model was primarily driven by one highly dominant variable. Once Age was removed from the dataset, the tree was forced to split on weaker predictors (*Supplementary Appendix Figure 8*), which significantly degraded model quality. The R^2 scores dropped below 0.085, and MSEs nearly doubled, highlighting the limited predictive value of the remaining features when modeled with a shallow tree structure. Feature selection confirmed this issue, as models using only the top 8 or 9 predictors performed similarly poorly, suggesting that even the best remaining features lacked the standalone predictive power to support accurate decision tree modeling. Ultimately, this shows that while decision trees are simple and interpretable, they are highly dependent on strong, clear-cut features, and in this case, were unable to effectively model stroke risk without the influence of Age – making them an unsuitable choice for this dataset under these conditions.

The random forest models revealed stark differences (*Results Table 3*) in performance depending on the inclusion of certain features, particularly Age, which emerged as the most influential predictor by a large margin. When Age was included, the model achieved near-perfect results, with training and testing R^2 values of 0.97 and 0.94, respectively, and very low MSEs of 5.82 and 12.47. These metrics suggest that the model not only fit the

training data extremely well but also generalized effectively to unseen data. Cross-validation further confirmed that the best mtry value was 4 – the same as the default setting based on the number of predictors – indicating model stability across training strategies. However, once Age was removed, performance declined sharply: training and testing R^2 values dropped to around 0.58 and 0.44, and MSEs increased dramatically to around 88 and 120, respectively. This demonstrates that *Age* was a dominant signal in the data and that the remaining features, while still somewhat informative, lacked the predictive power to drive strong performance on their own. Further feature selection, which reduced the predictor count to 8 and then 9, led to even more dramatic drops in performance. With 8 features, the R^2 values hovered around 0.26 across all evaluation modes, and MSEs approached 159, while adding a ninth feature produced only marginal improvement. These findings indicate that the model's strong performance initially stemmed from one or two highly predictive variables and removing or limiting those resulted in underfit models with high residual error. The fact that cross-validated performance mirrored regular train/test results also support that the models were not overfitting, but rather constrained by the quality and informativeness of the available features. Overall, these outcomes highlight the power of random forests in capturing complex relationships – when strong predictors are present – but also their sensitivity to feature informativeness, especially in high-dimensional or noisy settings.

Result Table 1

Logistic Regression w/o Age	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
Accuracy	72.13%	72.74%	72.10%	72.74%
Sensitivity	65.77%	66.22%	65.78%	79.17%
Specificity	78.53%	79.17%	78.46%	66.22%
Precision	75.50%	75.82%	75.45%	66.22%
F1	70.30%	70.70%	70.28%	70.70%
AUC	79.93%	80.66%	79.76%	80.66%
Firth's Regression w/o Age	**Same as above**			
Ridge Regression w/o Age				
LASSO Regression w/o Age				
Elastic Net Regression w/o Age				
PCA + Ridge w/o Age	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
Accuracy	71.64%	71.68%		
Sensitivity	69.61%	70.05%		
Specificity	73.68%	73.28%		
Precision	72.68%	72.11%		
Recall	71.11%	71.07%		
F1	79.86%	80.44%		
Logistic Regression w/ Top 8	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
Accuracy	65.58%	66.14%	65.58%	66.14%
Sensitivity	74.70%	75.67%	74.70%	56.74%
Specificity	56.41%	56.74%	56.41%	75.67%
Precision	63.29%	63.30%	63.29%	75.67%
F1	68.52%	68.94%	68.52%	68.94%
AUC	72.05%	72.38%	71.83%	72.38%
Firth's Regression w/ Top 8	**Same as above**			
Ridge Regression w/ Top 8				
LASSO Regression w/ Top 8				
Elastic Net Regression w/ Top 8				
Logistic Regression w/ Top 9	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**

Accuracy	67.33%	68.11%	67.33%	68.11%
Sensitivity	62.48%	63.26%	62.48%	72.90%
Specificity	72.21%	72.90%	72.21%	63.26%
Precision	72.21%	69.72%	69.35%	63.26%
F1	72.21%	66.33%	65.73%	66.33%
AUC	73.37%	73.74%	73.20%	73.74%
Firth's Regression w/ Top 9	**Same as above**			
Ridge Regression w/ Top 9				
LASSO Regression w/ Top 9				
Elastic Net Regression w/ Top 9				

Results Table 2

Naive Bayes w/o Age	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
Accuracy	72.37%	72.15%	72.36%	72.11%
Sensitivity	65.86%	66.18%		
Specificity	78.80%	78.31%		
Precision	75.38%	75.93%		
F1	70.30%	66.18%		
Naive Bayes w/ Top 8	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
Accuracy	65.85%	65.85%	65.70%	65.70%
Sensitivity	75.11%	75.11%		
Specificity	56.28%	55.77%		
Precision	63.00%	63.98%		
F1	68.45%	69.10%		
Naive Bayes w/ Top 9	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
Accuracy	67.67%	67.33%	67.67%	67.33%
Sensitivity	62.75%	62.63%		
Specificity	72.52%	72.20%		
Precision	69.23%	69.96%		
F1	65.83%	66.09%		
Beta Regression w/o Age	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.4778568	0.4810648	0.4778568	0.4769271

MSE	110.3924	111.0222	110.3874	110.4464
RMSE	10.50677876	10.53670727	10.50654082	10.50934822
Beta Regression w/ Top 8	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.2711583	0.2711583	0.2711952	0.2703164
MSE	155.93	155.93	155.9219	156.0869
RMSE	12.48719344	12.48719344	12.4868691	12.4934743
Beta Regression w/ Top 9	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.2983754	0.2983754	0.298421	0.297434
MSE	150.1071	150.1071	150.0973	150.2976
RMSE	12.25182027	12.25182027	12.25142033	12.25959216

Results Table 3

Regression Tree w/ Age	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.5233079	0.5147031	0.5233079	0.5147031
MSE	100.7817	103.8326	100.7817	103.8326
RMSE	10.03901	10.18983	10.03901	10.18983
Regression Tree w/o Age				
R^2	0.07328899	0.0660639	0.08375329	0.07790365
MSE	195.9243	199.8219	193.7119	197.2888
RMSE	13.9973	14.13584	13.91804	14.04595
Regression Tree w/ Top 8				
R^2	0.07328899	0.0660639	0.08375329	0.07790365
MSE	195.9243	199.8219	193.7119	197.2888
RMSE	13.9973	14.13584	13.91804	14.04595
Regression Tree w/ Top 9				
R^2	0.8375329	0.7790365	0.8375329	0.7790365
MSE	193.7119	197.2888	193.7119	197.2888
RMSE	13.91804	14.04595	13.91804	14.04595
Random Forest w/ Age	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.9724497	0.941699	0.9725421	0.9419982
MSE	5.824488	12.47304	5.805112	12.40988

RMSE	2.413398	3.531719	2.40938	3.522766
Random Forest w/o Age				
R^2	0.5831558	0.4405766	0.5834098	0.4413979
MSE	88.1282	119.6843	88.07507	119.5167
RMSE	9.38766	10.94003	9.384832	10.93237
Random Forest w/ Top 8				
R^2	0.2618918	0.2571659	0.2610146	0.2566973
MSE	156.0455	158.9236	156.2355	159.0346
RMSE	12.4823	12.603	12.49682	12.6078
Random Forest w/ Top 9				
R^2	0.290141	0.2805805	0.289695	0.2802372
MSE	150.0733	153.9142	150.1719	152.9981
RMSE	12.25106	12.40725	12.24938	12.406643

Results Table 4

KNN w/o Age	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.48767	0.43748	0.4839	0.43134
MSE	108.3136	120.346	109.1087	120.1769
KNN for Top 8	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.25676	0.24974	0.25885	0.24718
MSE	157.1299	160.5129	156.6883	159.1125
KNN for Top 9	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.29693	0.27927	0.29893	0.2749
MSE	148.6384	154.1944	148.2145	153.2529

Gradient Boosting w/o Age	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.4794689	0.4823856	N/A	0.478818
MSE	110.0469	110.7396	110.1385	110.13016
Gradient Boosting for Top 8	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.2678252	0.27095	N/A	0.267466

MSE	154.7911	155.9746	154.8476	154.81319
Gradient Boosting for Top 9	**Train Score**	**Test Score**	**CV Train Score**	**CV Test Score**
R^2	0.2979091	0.2983686	N/A	0.29749
MSE	148.431	150.1086	148.4939	148.46461

Conclusion

In our analysis of stroke risk prediction, we identified logistic regression as the most effective classification model. All other classification methods – such as LASSO, Ridge, Elastic Net, and even more complex models like Naïve Bayes – ultimately converged to similar results as logistic regression. Given its simplicity, interpretability, and comparable performance, it stands out as the best option, especially for healthcare settings where explainability is key.

For regression for predicting Stroke.Risk.Percentage, both Random Forest and Gradient Boosting emerged as strong contenders for being the best models. Their performance was nearly identical across different versions of the dataset, and both handled nonlinear relationships and complex feature interactions effectively. Since neither clearly outperformed the other, both can be considered equally viable for predicting stroke risk percentages. Beta regression and KNN also performed well here, but due to Random Forest and Gradient Boosting being known to handle non-linear relationships effectively, our preference is for them from a real-world applicability perspective.

Some limitations of our project include the possible synthetic nature of the dataset, which may explain why nearly all features appeared to strongly influence stroke risk. In a real-world setting, we would expect more noise and less perfectly clean patterns. Additionally, the absence of important clinical factors such as family history or cholesterol levels limits the broader applicability of these models. Also, the overwhelming influence of Age on the response made separating by age groups not a viable path (we tried and were getting perfect results), even though we would have wished to explore that avenue for our research here.

Below are responses to our guiding research questions:

1. **Can we create a questionnaire for patient intake to appropriately triage patients for stroke?** Yes, using a small subset of predictors yielded decent sensitivity, making it feasible for quick screening tools.

2. **Can we create an accurate model for predicting stroke risk assuming precise medical measurements have been done?** Somewhat. While accuracy improves with more detailed input, predictive power is still limited without robust data.

3. **What are the most significant risk factors associated with a chance of having a stroke?** Age was by far the strongest predictor, but after its removal, no single factor stood out—likely due to correlated symptom-based inputs in synthetic data.

4. **How does age impact the likelihood of having a stroke?** Massively. Age alone could nearly predict stroke risk perfectly, so it had to be excluded to evaluate other variables meaningfully.

5. **Are there any significant interactions between risk factors predicting stroke?** No. We did not find strong interactions or correlations between individual predictors.

6. **What are the result differences between traditional statistical methods and machine learning models?** Both performed comparably in many cases. Logistic regression was strong for classification, while ensemble methods like gradient boosting slightly outperformed in regression tasks. This supports using both types depending on the context and goal—interpretability vs. accuracy.

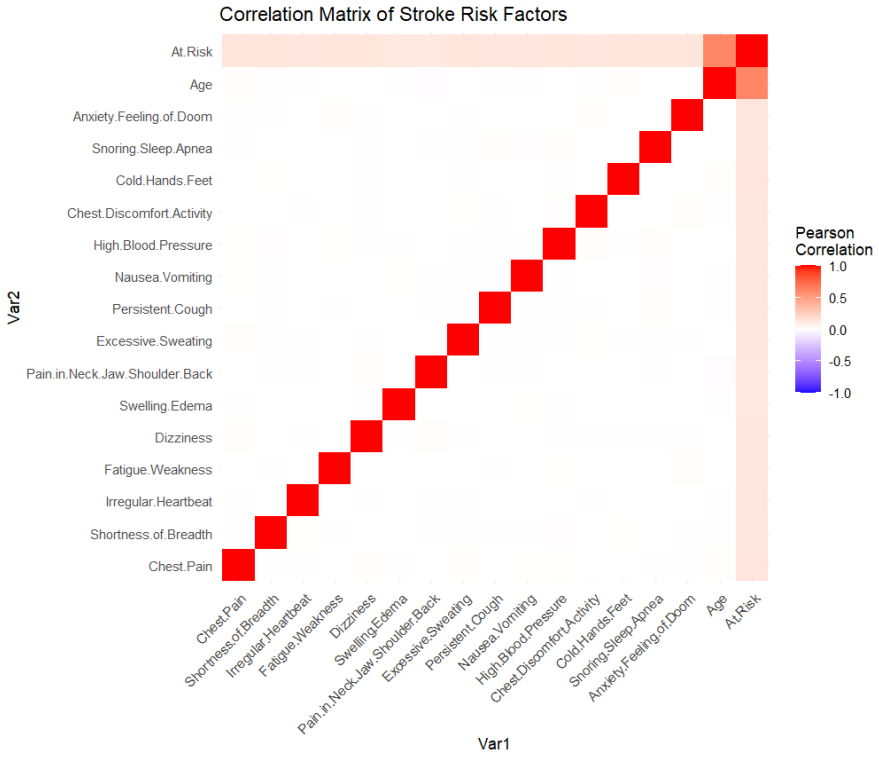
Lessons learned

One major lesson we learned is that ensuring data quality is paramount to analytical applications for healthcare solutions – including avoiding synthetic or overly clean datasets – is critical before modeling. Also, as exploratory models are being created, we could question and look for other information out there that could perhaps enhance our overall objective. As a result, for a future project, we would be more careful when selecting a data set. Additionally, many regression models yield similar results when regularization is minimal, we may try out more regularization or advanced techniques. However, we suspect that if the dataset was not as clean, we would have benefited from regularization anyways. Finally, regression trees performed well in capturing stroke likelihood when strong predictors were present but struggled when they were removed, highlighting the importance of feature selection and model robustness in real-world applications. If we were to do more analysis on predicting the risks of a stroke, we would probably undertake an in-depth literature search to help us improve our process of feature selection.

Appendix

- **Figure 1: Total amount of 1's and 0's for the At.Risk response variable before and after sampling**

- **Figure 2: Correlation matrix between stroke risk predictors and response variable At.Risk**



- **Figure 3: Table showing the Variance Inflation Factors (VIF) for each stroke risk predictor based off logistic regression model**

Predictor	VIF Value
Chest.Pain	1.0237
Shortness.of.Breadth	1.0229
Irregular.Heartbeat	1.0227
Fatigue.Weakness	1.0253
Dizziness	1.0222
Swelling.Edema	1.0250
Pain.in.Neck.Jaw.Shoulder.Back	1.0221
Excessive.Sweating	1.0261
Persistent.Cough	1.0241
Nausea.Vomiting	1.0258
High.Blood.Pressure	1.0249
Chest.Discomfort.Activity	1.0230
Cold.Hands.Feet	1.0256
Snoring.Sleep.Apnea	1.0248
Anxiety.Feeling.of.Doom	1.0264

- **Figure 4: Variable importance from PCA application**

	Variable	Importance
Shortness.of.Breadth	Shortness.of.Breadth	3.421398
Swelling.Edema	Swelling.Edema	3.400951
Chest.Discomfort.Activity	Chest.Discomfort.Activity	3.400312
Nausea.Vomiting	Nausea.Vomiting	3.393041
Persistent.Cough	Persistent.Cough	3.310882
Excessive.Sweating	Excessive.Sweating	3.303767
Dizziness	Dizziness	3.252238
Pain.in.Neck.Jaw.Shoulder.Back	Pain.in.Neck.Jaw.Shoulder.Back	3.197575
Fatigue.Weakness	Fatigue.Weakness	3.157781
Chest.Pain	Chest.Pain	3.139747
High.Blood.Pressure	High.Blood.Pressure	3.119675
Snoring.Sleep.Apnea	Snoring.Sleep.Apnea	3.117417
Anxiety.Feeling.of.Doom	Anxiety.Feeling.of.Doom	2.920929
Irregular.Heartbeat	Irregular.Heartbeat	2.819510
Cold.Hands.Feet	Cold.Hands.Feet	2.789790