Stroke Prediction Model

Fatih Emir Güler 2100999
Mert Erbak 2104226
Mohammed Ammar Salahie 2104551

Abstract

Stroke is a very important health problem worldwide. Early detection can help prevent strokes, and machine learning algorithms can be used to predict stroke risk. This project explains a machine learning model to predict stroke risk. We used a comprehensive approach that included data preparation, feature engineering, model training, and evaluation. The data was prepared by cleaning and addressing missing values. Also, features were engineered by creating new features from existing ones. The model was trained with machine learning algorithms, including logistic regression, random forest, XGBoost, LightGBM, and CatBoost. The model was evaluated using a variety of metrics, including accuracy, precision, recall, and F1-score. We have concluded that the logistic regression model had the best performance in predicting strokes. The model was able to correctly predict strokes with an accuracy of 82.2%. https://github.com/merterbak/stroke-prediction-model

Related Works:

Fernandez-Lozano et al. (2021) used a random forest model to predict stroke outcome, achieving an accuracy of 87%. Omar et al. (2022) developed a distributed machine learning approach to predict stroke risk. The model was trained on a dataset of patient records, including demographic information, medical history, and laboratory results. Their model achieved an accuracy of 85%, which is comparable to other machine learning methods for stroke prediction. Hanifa and Raja (2022) developed a deep learning model to predict stroke risk using EEG signals. The model was trained on a dataset of EEG signals from patients who had suffered a stroke and patients who had not.Heo et al. (2023) created three machine learning models to predict stroke outcomes: a deep neural network, a random forest, and a logistic regression model. They compared their predictability, and found that the deep neural network model was the most accurate.

Introduction

One of the main causes of mortality and disability in the globe is stroke, and early identification can help avoid them. Our code uses machine learning algorithms to analyze patient data and predict the likelihood of stroke. In this project, we worked all together on a code to predict strokes using the data we have.

Materials and Methods

The initial part of the code focuses on data preparation and imports necessary libraries such as Pandas, Seaborn, NumPy, SciPy, and scikit-learn modules for data handling, preprocessing, and model evaluation. The training and test data are read from CSV files into a Pandas DataFrame. Using concatenation, the code combines the real data and extra data into a single training DataFrame. Additionally, the test DataFrame is loaded separately, and the test IDs are extracted for future reference. We have such features that we will be work on them, numerical features: bmi, age, avg glucose level and categorical features: gender, ever married, work type, residence type, smoking status.

Then data cleaning and feature engineering were crucial steps in developing our prediction code. We also had to select the most relevant features to include in our prediction model. We used statistical methods to identify which features were most strongly correlated with the risk of stroke. We also created new features by combining existing ones, such as calculating body mass index (BMI). In contrast, steps are then performed on the training and test data. Firstly, rows with missing values in the 'bmi' column are dropped from the training data. There were around 200 missing 'bmi' values, and we tried to impute them with other correlations. To find this, we used plottings to see the correlation between 'bmi' and other features. We also tried taking the mean of the rest of the 'bmi' values. However, the performance metrics showed us that the most feasible solution to handle the missing 'bmi' values is simply dropping them.

Apparently, there were no outliers in age, 627 in avg glucosec level, and 126 in 'bmi'. We tried removing the outliers, but the performance dropped too much, so we lost too much information when we tried that. Therefore, we didn't touch them.The 'smoking status' column's missing values are filled with the value 'never smoked'. Moreover, the 'Other' gender category in both the training and test datasets is replaced with 'Male'. Additional features are created based on existing features, such as 'age/bmi', 'age*bmi', 'bmi/prime', 'obesity', and 'blood heart', which are computed using arithmetic operations (Cáliz, 2023).

The code analyzes the percentage of unique values in each categorical column of the training dataset. To address the class imbalance problem, the code employs random undersampling using the RandomUnderSampler. To handle categorical features, a ColumnTransformer is defined, and the data is standardized using the StandardScaler.

To further address class imbalance, the code utilizes SMOTEENN (SMOTE + Edited Nearest Neighbors). A Random Forest Classifier is trained on the preprocessed training data to determine the feature importances.

The top-k features are selected based on their importances, and the optimal 'k' number is 13. The code defines a Logistic Regression model with L2 penalty in order to deal with overfitting and underfitting. The logistic regression model is used to predict stroke (0 or 1) by estimating the probability of the target variable taking a particular value, given the values of the predictor variables. The model is trained on the selected features using the training data and evaluated using metrics such as accuracy, precision, recall, F1-score, ROC AUC score, and PR AUC score. The predictions are stored in a DataFrame along with the corresponding test IDs and the results are saved to a CSV file.

The code also includes outlier removal and hyperparameter tuning, which can be uncommented and modified if needed. The code provides a comprehensive pipeline for data preparation, feature engineering, model training, and prediction. It handles missing values, performs feature engineering, addresses class imbalance, and applies feature selection techniques. The chosen Logistic Regression model is trained and evaluated using relevant metrics, and the predictions are generated for the test dataset. The code also contains bias-variance test via 5 folded cross-validation and gives visions about overfitting and underfitting. Finally, the code trains an XGBoost classifier with optimized hyperparameters founded with GridSearchCV.

The most important details in this text are that the classifier is fitted to the training data using the selected features, and the probabilities of the positive class (stroke) are predicted for the validation data using the trained model. Also code setting cutoff point to positive predict probability as 0.7 since we find out this is the most optimal threshold value. The code then defines a LGBM Classifier with the best hyperparameters based on manual tuning, and define CatBoost Classifier to include VotingClassifier. We tried to use VotingClassifier with models such above: BasicLG, XGB, LGBM, CatB. Howewer it showed that basic logistic regressin performing better than VotingClassifier model.

Results:

Accuracy: 0.8229037843185754 Precision: 0.9562065231251275
Recall: 0.7628205128205128 F1-score: 0.874521942972522
ROC AUC score: 0.87691054336282 PR AUC score: 0.18751436180421371
Kaggle Submission: 0.814
Note: If you have imbalanced classes, where one class has a larger number of true instances, the precision for that class will have a higher impact on the overall precision score. So in order to print informative information we used average='weighted' on calculating F1 and precision.

Initially, due to data imbalance, undersampling was performed to reduce the sample size. Categorical values were transformed using one-hot encoding. Logistic regression,

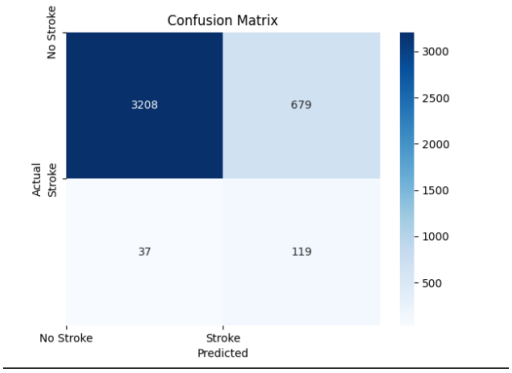

Figure 1: Confusion Matrix

a suitable algorithm for binary classification, learned the relationship between predictor variables and the target variable by estimating the probability of the target variable taking a specific value. The model then predicted the class with the highest probability. In conclusion, this data science project successfully developed a predictive model for stroke using machine learning algorithms and patient record data.

References:

Kaggle discussion forum.https://www.kaggle.com/competitions/playground-series-s3e2/discussion/378780 https://scikit-learn.org/stable/documentation.htmlhttps:
Miuul Machine Learning Summer Camp
https://www.geeksforgeeks.org/ml-handling-imbalanced-data-with-smote-and-near-miss-algorithm-in-python/ Fernandez-Lozano, C., Hervella, P., Mato-Abad, V., et al. (2021). Random forest-based prediction of stroke outcome. Scientific Reports, 11(1), 10071. https://doi.org/10.1038/s41598-021-89434-7. Ahmed H.,Abd-el ghany S. F., Youn E. M. G., Omran N. F., Ali A. A., (2022). Stroke prediction using distributed machine learning based on Apache Spark. International Journal of Computer Science and Information Security, 20(1), 1-10. doi:10.1109/IJCSIS.2022.3527943 Kaur, M., Sakhare, S. R., Wanjale K., Akter F. (2022). Early Stroke Prediction Methods for Prevention of Strokes doi: 10.1155/2022/7725597 Heo, J. H., Yoon, S., Kim, J. H., Park, J. Y., Kim, S. H., Kim, H. J., ...&Kim, J. H. (2019). Machine learning–based model for prediction of outcomes in acute stroke. Stroke, 50(1), 177-184. doi:10.1161/STROKEAHA.118.024293