

Individual Project 1 Report

Sourav Panda
sbp5911@psu.edu

September 22, 2023

1 Methodologies

1.1 Data Loading and Exploration

The code begins by importing essential data from CSV files using Pandas. It loads the training data from "train.csv," which includes both features and corresponding labels indicating heart disease presence. Simultaneously, it imports the testing data from "test_X.csv," which contains only the features, and a sample submission file from "sample_submission.csv." These datasets serve as the foundation for building, training, and evaluating a machine-learning model to predict heart disease outcomes.

1.2 Data Pre-processing

In the provided code, the `convert_categorical_to_numerical` function plays a crucial role in the data pre-processing phase. Its primary purpose is to transform categorical features into a numerical format, specifically using a technique known as one-hot encoding. This transformation is applied to several categorical variables, including "Sex," "ChestPainType," "RestingECG," "ExerciseAngina," and "ST_Slope." One-hot encoding is employed to represent categorical data as binary vectors, where each category is transformed into a binary column, and the presence or absence of a category is indicated by 1s and 0s, respectively. This process is essential because many machine learning algorithms, including the RandomForestClassifier used in this code, require numerical input data. By converting categorical variables into a numerical format, the code ensures that these features can be effectively utilized in the subsequent modeling steps, ultimately contributing to the model's ability to make accurate predictions based on a diverse range of categorical data.

1.3 Missing Value Imputation

The code initiates its data preprocessing by identifying and addressing missing values in the "Cholesterol" column of the training dataset. It identifies 0 values within this column and replaces them with NaN (representing missing values). This step is essential because missing data can adversely affect the performance of machine learning models. To address these missing values effectively, the code employs K-Nearest Neighbors (KNN) imputation. Specifically, it utilizes the `KNNImputer` from the Scikit-Learn library, a powerful method that replaces missing values by considering the values of their nearest neighbors in the dataset. This approach ensures that the imputed values are derived from the context of similar data points, which often leads to more accurate and reliable imputations. By systematically handling missing data in this manner, the code enhances the quality of the dataset and contributes to the robustness of the subsequent machine learning modeling steps.

1.4 Feature Engineering

After data pre-processing and imputation, the training and testing datasets are prepared for model building by separating features from labels. The "PatientID" column, which doesn't contribute to predictions, is removed. This ensures that the model is trained on relevant input data for accurate predictions.

1.5 Random Forest Classifier

The code utilizes Scikit-Learn's Random Forest Classifier, a highly effective ensemble algorithm, for the heart disease classification task. This ensemble model is renowned for its robustness and accuracy in classifying data.

1.6 K-Fold Cross-Validation

The code employs k-fold cross-validation, specifically with $k=5$, as a means to assess the model's performance robustly. This technique involves dividing the

training dataset into five equally sized "folds." The model is trained and evaluated five times, each time using a different fold as the validation set while the remaining folds serve as the training set. This process helps gauge how well the model generalizes to unseen data and provides a more reliable estimate of its performance. The `cross_val_score` function is used to compute accuracy scores for each fold, allowing for an assessment of consistency and overall accuracy, which helps in making informed decisions about model selection and hyperparameter tuning.

1.7 Model Training

Following cross-validation, the Random Forest model is further refined by training it on the entire training dataset using the `fit` method. This step allows the model to learn from the full dataset, optimizing its ability to make accurate predictions on unseen data.

1.8 Model Evaluation

The code evaluates the model's performance by first calculating the training accuracy on the same training data to assess how well the model performs on the data it was trained on. Subsequently, the trained Random Forest model is used to make predictions on the test data, and the results are saved in a CSV file in the required submission format.

2 Key Ideas and Lessons Learned

- Categorical features are converted into a numerical format through one-hot encoding to make them suitable for machine learning models.
- Missing data in the "Cholesterol" column is handled using KNN imputation to retain valuable information.
- The Random Forest Classifier is chosen as the model for this binary classification task, but other models could also be explored and compared for potentially better performance.
- K-fold cross-validation is used to assess the model's generalization performance and identify any overfitting issues.
- A random seed is set for reproducibility.
- The code provides a structured pipeline for data preprocessing, model building, and evaluation, which is essential in machine learning projects.

3 Other Methodologies Used and Key Takeaways

3.1 Logistic Regression with/without Hyperparameter Tuning

Logistic Regression has several limitations compared to the Random Forest Classifier. Firstly, it struggles to model complex, nonlinear relationships in data due to its linear assumption, whereas Random Forest excels in capturing intricate patterns. Logistic Regression can be sensitive to outliers and is less robust in handling irrelevant features. Imbalanced datasets pose a challenge for Logistic Regression, which may result in biased predictions favoring the majority class. Additionally, it may underfit on complex datasets and has inherent assumptions that might not always hold. While it provides probability estimates, they may require calibration for accuracy. On the other hand, Random Forest offers greater flexibility, robustness, and adaptability to various data structures, making it a preferred choice for tackling complex real-world problems.

3.2 Dropping zero values

In the provided code, when we chose to replace 0 values in the "Cholesterol" column with KNN imputation, we preserved the size and structure of our training dataset by filling in missing values based on nearby data points. This approach retained as much information as possible and allowed the model to learn from the entire dataset. In contrast, when we decided to drop rows containing 0 values, we effectively reduced the amount of training data available to the model. This data loss led to a lower accuracy because the model had access to less information, resulting in a less effective representation of the underlying patterns in the data. The observed difference in accuracy highlights the importance of handling missing values carefully, as removing data can impact the model's ability to make accurate predictions, especially if the missing values contain valuable information for the task at hand.

3.3 Replacement of zero values with Mean

The lower accuracy likely indicates that mean imputation may not be appropriate for handling missing values in this dataset as it contains outliers that affect the mean. Mean imputation assumes that missing values are missing randomly and can be estimated by a constant value, the mean. However, if there is

any underlying pattern or reason for the missing values, this method can introduce inaccuracies and bias into the dataset. The accuracy difference highlights the importance of selecting the right imputation method that aligns with the data characteristics and ensures accurate model predictions.

3.4 Replacement of zero values with Median

The lower accuracy compared to using KNN imputation indicates that median imputation may not have been the optimal choice for handling missing values in your dataset. While median imputation is robust to outliers and suitable for skewed data, its effectiveness depends on the nature of the missing data. In this case, it's possible that the missing values did not conform to the assumption of a similar distribution as the observed values. Median imputation assigns a single value to all missing entries, which can lead to a loss of information and inaccuracies, especially if the missing values are associated with unique patterns or interactions with other features.

3.5 Replacement of zero values with 1st Quartile (25%) Values

This method assumes that the missing values follow a similar distribution as the lower 25% of the observed values, an assumption that may not hold in all cases. Imputing missing values with the first quartile essentially assigns a single constant value to all missing entries, which can lead to information loss and inaccuracies, especially if the missing values have distinct patterns or interactions with other features. It indicates that this particular imputation strategy may not be well-suited for this dataset.

3.6 Replacement of zero values of more than 1 Column

This approach assumes that missing values in different columns can be treated uniformly, which may not hold true if the reasons for missing values vary across columns. It's essential to consider column-specific characteristics and employ imputation methods that best suit each variable's distribution and the nature of the missing data.








#	Team	Members	Score	Entries	Last
1	nobody		0.91707	16	5d
2	siyangni		0.91707	41	4h
3	Asuka		0.91262	10	9h
4	Sree Bhattacharyya		0.91089	42	2h
5	soumya singh		0.91000	23	2d
6	aks7045		0.90909	30	3d
7	Sourav Panda		0.90731	48	7h

Figure 1: Final rank in the Leaderboard

4 Conclusion

In conclusion, this individual project report showcases a systematic and insightful approach to addressing a binary classification problem related to heart disease prediction. The methodologies employed underscore the significance of data preprocessing, transformation, and thoughtful handling of missing data. By converting categorical features into numerical format through one-hot encoding and utilizing K-Nearest Neighbors (KNN) imputation for missing data, the report highlights the critical role of data quality in model performance. The selection of the Random Forest Classifier as the primary model, along with the use of K-fold cross-validation, demonstrates a robust modeling process. Furthermore, the project provides valuable comparisons, emphasizing the advantages of Random Forest over Logistic Regression and offering essential considerations when handling zero values and missing data. Overall, this report serves as a valuable resource for machine learning practitioners, emphasizing the importance of data preparation and thoughtful model selection in achieving accurate predictions for complex real-world problems.