**A REPORT ON THE USE OF MACHINE LEARNING TO PREDICT HEART FAILURE SURVIVAL IN PATIENTS**

**BY**

**ABISOLA AKINJOLE**

**C2003133**

**JANUARY 2023**

**TABLE OF CONTENT**

**CHAPTER ONE: INTRODUCTION1**

**CHAPTER TWO: DESIGN IMPLEMENTATION 1**

**CHAPTER THREE: EXPLORATORY DATA ANALYSIS 2**

DATASET DESCRIPTION2

EXPLORATORY DATA ANALYSIS3

Missing Values4

Measure of Location and Spread4

Plots5

**CHAPTER FOUR: MACHINE LEARNING METHODS9**

CLASSIFICATION I10

CLASSIFICATION II11

FEATURE SELECTION15

CLASSIFICATION III17

CONCLUSION20

**CHAPTER FIVE: REFLECTION 21**

**REFERENCE i**

**LIBRARIES ii**

**APPENDIX A: EXPLANATORY DATA ANALYSIS iv**

**APPENDIX B: MACHINE LEARNING METHODS vi**

CLASSIFICATION I (TEST SIZE = 33%)vii

CLASSIFICATION Ix

CLASSIFICATION IIxiv

CLASSIFICATION IIIxxi

**CHAPTER ONE**

**INTRODUCTION**

Heart failure is a serious and growing health concern, with an estimated 26 million people affected worldwide (Yusuf et al., 2001). Prediction of heart failure can play a crucial role in improving patient outcomes and reducing healthcare costs (Ponikowski et al., 2016). Machine learning (ML) has shown promise in this area, with the ability to automatically analyse and uncover patterns in patient data (Vladimir Vapnik, 2013; Ahmadian et al., 2021). This project explores the use of ML to predict heart failure survival in patients, through a combination of programming and analytics techniques. We aim to evaluate the performance of various ML models and select the best one for informed decision-making on patient survival.

**CHAPTER TWO**

**DESIGN IMPLEMENTATION**

The code implementation process involves placing all necessary files, including the csv data file, in the same folder to avoid *IOError*.

Diagram, Teams

Description automatically generated

**Fig 2.1. EDA\_ML Overview**

The *Heart Failure Prediction\_MAIN* imports and uses the *EDA\_ML* module to perform the analysis and solutions on the data.

**CHAPTER THREE**

**EXPLORATORY DATA ANALYSIS**

**DATASET DESCRIPTION**

This project utilised the *heart\_failure\_clinical\_records* dataset, which includes 299 records of patients, 105 of which are women and 194 are men, with ages ranging from 40 to 95, with 203 survived and 96 dead.

Further description of the variables in the dataset are:

Table

Description automatically generated**Table 3.1. Data Overview**

**EXPLORATORY DATA ANALYSIS**

It performs the functions shown below using some Python libraries.

**A screenshot of a phone

Description automatically generated with medium confidence**

**Fig 3.1. EDA Functions**

**Missing Values**

The first function *checking\_missing\_values* checks for missing values in the dataset. The output can be seen in Appendix A Fig 2. This demonstrates that the dataset contains no missing values; therefore, no cleaning technique is required.

**Measure of Location and Spread**

The second function, *measures\_of\_location*, calculates the mean, median, skewness, and kurtosis to determine the centre and shape of the data distribution. The third function, *measures\_of\_spread*, calculates the range, variance, and standard deviation to measure the spread of the data. The results can be found in Appendix A, Figures 3-8.

A screenshot of a computer

Description automatically generated with low confidence

**Table 3.2. Measures of Location and Spread**

**Plots**

The last function *plot* produces histogram, bar chart, boxplot, and correlation heatmap.

**Histogram**

The histogram is used to visualize variable frequency, understand the distribution, and detect skewness.

Chart, waterfall chart

Description automatically generated

**Fig 3.2. Distribution of Binary Variables**

Fig 3.2 shows that the majority of patients in the dataset are non-anaemic, non-diabetic, do not have high blood pressure (hbp), are men, are non-smokers, and have survived.

Chart, histogram

Description automatically generated

**Fig 3.3. Distribution of Numerical Variables**

Fig 3.3 confirms Table 3.2's interpretation.

**Bar Chart**

This project uses bar chart to show the frequencies and distribution of *DEATH\_EVENT* and certain binary features.

Chart, bar chart

Description automatically generated

**Fig 3.4. DEATH\_EVENT and some Binary Variables**

Fig 3.4 shows that patients without diabetes, HBP, and who do not smoke have a higher survival and death rate compared to those with these conditions. Additionally, there is a higher proportion of men among both survivors and non-survivors than women in the sex data, indicating that these variables may not have a correlation with the *DEATH\_EVENT* outcome.

**Correlation Matrix**

The project uses correlation matrix to identify explanatory variables related to the target variable.

Chart

Description automatically generated

**Fig 3.5. Correlation Matrix**

Fig 3.5 indicates that *age* and *serum\_creatinine* have a positive correlation with *DEATH\_EVENT*, while *ejection\_fraction, serum\_sodium* and *time* have a negative correlation with *DEATH\_EVENT*. The result also suggests that *anaemia, creatinine\_phosphokinase, diabetes, high\_blood\_pressure, platelets, sex* and *smoking* have no correlation with *DEATH\_EVENT*, which confirms the previous inference made for *diabetes, high\_blood\_pressure, sex* and *smoking.*

**CHAPTER FOUR**

**MACHINE LEARNING METHODS**

This project uses supervised ML algorithms such as Gaussian Naïve Bayes (GNB), Logistic Regression (LR), K-Nearest Neighbour (KNN), Random Forest Classifier (RFC), Support Vector Machine (SVM) and Multi-Layer Perceptron Neural Networks (MLP). Outliers were handled using RobustScaler (Liland et al., 2016). Hyperparameter tuning with GridSearchCV was used to optimize the models (Ahmad et al., 2022). The evaluation metrics used were accuracy, precision, recall and F1-score, and confusion matrix analysis.

**Optimizing Model Performance: Techniques and Metrics**

This project aims to develop models that generalize well to unseen data. To achieve this, the *cross\_validation* function is used to address the issue of randomness in the training and testing sets (Alpaydin, 2020), and print accuracy score. Additionally, it uses *confusion\_matrix\_plot* function to display the results of the confusion matrix visually. To check for overfitting, the project uses *test\_overfitting* function. These functions are used consistently across all models in the *EDA\_ML* module.

Table

Description automatically generated

**Table 4.1. 33% & 25% Test Size Results**

The 75:25 train-test split was chosen for improved model performance over a 67:33 split, as more data is available for training. This is evident in the results shown in Table 4.1.

**CLASSIFICATION I**

The functions for classification I perform the process as shown in Fig 4.1.

Diagram

Description automatically generated

**Fig 4.1. Prediction Process**

**Confusion Matrix**

Table

Description automatically generated

**Table 4.2. Confusion Matrix**

Table 4.2 shows that the models exhibit a bias towards predicting “survived” cases, resulting in higher rates of True Negative (TN) and False Negative (FN).

**Evaluation Metrics:**

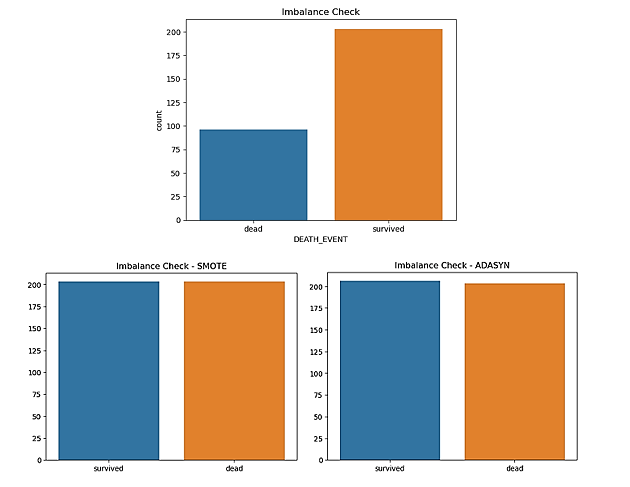
**Table

Description automatically generated**

**Table 4.3. Evaluation Metrics**

Table 4.3 shows a bias in the model towards the "survived" class having higher scores than the "dead" class. The best performing models are LR, RFC and MLP; with RFC having the highest overall accuracy at 87.07%. However, due to bias in recall and F1-score, RFC may not be the best overall model for predicting heart failure survival.

**CLASSIFICATION II**



**Fig 4.2. Class Imbalance Check**

Fig 4.2 shows class imbalance, 96 dead and 203 survived, and illustrates the balancing of the class. SMOTE and ADASYN methods are used in the process.

Diagram

Description automatically generated

**Fig 4.3. Prediction Process**

The functions for classification III perform the process shown in Fig 4.3.

**Confusion Matrix**

**Table

Description automatically generated**

**Table 4.4. Confusion Matrix for Classifications I&II**

Balancing techniques such as SMOTE and ADASYN improve performance and reduce bias towards the majority "survived" class. Table 4.4 shows that RFC and SVM with SMOTE balancing have high TN and TP rates among all models and techniques. RFC with SMOTE is the best, with high TN and TP scores and low FN scores. This indicates that this combination of model and balancing technique is the best at correctly predicting patient survival and death.

**Evaluation Metrics:**

**Table

Description automatically generated**

**Table 4.5. Evaluation Metrics for Classifications I&II**

Table 4.5 demonstrates that using techniques such as SMOTE and ADASYN to balance the data improves the overall performance of prediction models for heart failure survival and reduces bias towards the majority class. The “NONE” column represents results from classification I with imbalanced data. The RFC with SMOTE had the highest accuracy and F1-score, as well as good precision and recall, making it the top performer. Thus, using SMOTE to balance the data results in improved performance for the models in this classification.

**FEATURE SELECTION**

This project employs feature selection techniques to evaluate the significance of features and determine their importance for selection.

**Hypothesis:**

***= 0.05***

Table

Description automatically generated

**Table 4.6. Hypothesis**

Table

Description automatically generated

**Table 4.7. Statistical Test**

Chart, bar chart

Description automatically generated Chart, bar chart

Description automatically generated

**Fig 4.4. Ranked Features**

The Table 4.6 shows the hypothesis and the rule of thumb that was used as a threshold in the *feature\_selection* function for the statistical tests in the *EDA\_ML* module. The results of this feature selection can be found in Table 4.7.

**RFC Importance Plot**

Chart, bar chart

Description automatically generated

**Fig 4.5. RFC** **Plot**

The most significant features for predicting heart failure survival, as determined by the three feature selection techniques, are *age, ejection\_fraction*, *serum\_creatinine,* and *time*. This is evident in Figures 4.4 and 4.5.

**CLASSIFICATION III**

The *classIII* function uses important features (*age, ejection\_fraction, serum\_creatinine, time*) to train and test models for prediction as shown in Fig 4.6.

Diagram

Description automatically generated

**Fig 4.6. Prediction Process**

**Confusion Matrix**

Table

Description automatically generated

**Table 4.8. Confusion Matrix for Classifications I, II and III**

Classification III, as shown in Table 4.8, has a slight bias towards predicting the majority class, resulting in higher TN and FN rates in the confusion matrix, but performs better than classification I for some models. The RFC stands out as the top performer in all classifications, with classification II being the best, with high TN and TP and low FN.

**Evaluation Metrics:**

**Table

Description automatically generated**

**Table 4.9. Evaluation Metrics for Classifications I, II, and III**

Chart, bar chart

Description automatically generated

**Fig 4.7. Model Performance**

Table 4.9 and Fig 4.7 indicates that Classification III generally performs better than Classification I, with higher accuracy, precision, recall, and F1-scores for both classes. This suggests that the selected features can make accurate predictions rather than relying on the entire dataset. However, Classification II still outperforms Classification III.

LR in Classification III shows good results, with an accuracy of 85.24% and the highest recall (96% for survived) and precision (88% for dead). Despite this, the RFC demonstrates the best overall performance across all classifications, with an accuracy of 89.15% in the "balanced with SMOTE" classification, surpassing both "important features" (87.05%) and "unbalanced" (87.07%) classifications.

**CONCLUSION**

Classification II, which employs SMOTE to balance the data, performed the best overall among all the classifications with the parameters shown in Appendix B, Fig 17, as evidenced by the high scores for TN, TP, and F1-score across several models. The RFC model, known for its popularity and accuracy (Lebedev et al., 2014), demonstrated exceptional performance in Classification II, achieving an accuracy of 89.15%, precision of 98%, recall of 80% and F1-score of 81%. The confusion matrix results further confirm that the model has the best balance of correctly identifying both “survived” and “dead” classes. Additionally, this model showed good generalization, avoiding overfitting.

Based on these findings, it can be concluded that the RFC with SMOTE is the recommended model for predicting heart failure survival in patients. The use of SMOTE for balancing the data is also found to be effective in enhancing model performance and reducing bias towards the majority class. This underlines the significance of addressing class imbalance in ML models for accurate predictions and decision-making regarding patient survival.

**CHAPTER FIVE**

**REFLECTION**

This project was a valuable learning experience that allowed me to reinforce my understanding of key concepts in NumPy, Pandas, and ML. My knowledge of these topics was greatly enhanced through my research and hands-on implementation of the project objectives. One area for improvement would be to delve deeper into the Tkinter library and refactor my code to increase its readability, reusability, and minimize repetition. Overall, this project proved to be an enriching learning experience that solidified my understanding of ML concepts and sharpened my coding abilities.

**REFERENCES**

Ahmad, G. N., Fatima, H., Ullah, S., & Saidi, A. S. (2022). Efficient medical diagnosis of human heart diseases using machine learning techniques with and without GridSearchCV. *IEEE Access*, *10*(2169-3536), 80151–80173. https://doi.org/<10.1109/ACCESS.2022.3165792>

Ahmadian, S., Jalali, S. M. J., Raziani, S., & Chalechale, A. (2021). An efficient cardiovascular disease detection model based on multilayer perceptron and moth‐flame optimization. *Expert Systems*, *39(4)*(e12914). https://doi.org/<10.1111/exsy.12914>

Alpaydin, E. (2020). *Introduction To Machine Learning.* Mit Press.

Lebedev, A., Westman, E., Westen, V., Kramberger, M., Lundervold, A., Aarsland, D., Soininen, H., Kłoszewska, I., Mecocci, P., & Tsolaki, M. (2014). Random Forest ensembles for detection and prediction of Alzheimer’s disease with a good between-cohort robustness. *NeuroImage: Clinical*, *6*, 115–125.

Liland, K. H., Naes, T., & Indahl, U. G. (2016). ROSA-a fast extension of partial least squares regression for multiblock data analysis. *Journal of Chemometrics*, *30*(11), 651–662. https://doi.org/<10.1002/cem.2824>

Ponikowski, P., Voors, A. A., Anker, S. D., Bueno, H., Cleland, J. G. F., Coats, A. J. S., Falk, V., González-Juanatey, J. R., Harjola, V.-P., Jankowska, E. A., Jessup, M., Linde, C., Nihoyannopoulos, P., Parissis, J. T., Pieske, B., Riley, J. P., Rosano, G. M. C., Ruilope, L. M., Ruschitzka, F., & Rutten, F. H. (2016). 2016 ESC Guidelines for the diagnosis and treatment of acute and chronic heart failure. *European Heart Journal*, *37*(27), 2129–2200. https://doi.org/<10.1093/eurheartj/ehw128>

Vladimir Vapnik. (2013). *The Nature of Statistical Learning Theory*. Springer Science & Business Media.

Yusuf, S., Reddy, S., Ôunpuu, S., & Anand, S. (2001). Global Burden of Cardiovascular Diseases. *Circulation*, *104*(22), 2746–2753. https://doi.org/<10.1161/hc4601.099487>

**LIBRARIES**

The libraries imported are:

1. matplotlib.pyplot - for creating visualizations and plots.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in science & engineering, 9(3), 90-95.

1. seaborn - a library built on top of matplotlib for creating statistical visualizations.

Waskom, M., Botvinnik, O., Foldes, C., Hoyer, S., Aguilar-Ibanez, M., Halchenko, Y., ... & Rocklin, M. (2019). Seaborn: v0. 9. 0. Zenodo. http://doi.org/10.5281/zenodo.2577116

1. pandas - for data manipulation and analysis.

McKinney, W. (2010). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (pp. 51-56).

1. numpy - for numerical computation and working with arrays.

Oliphant, T. E. (2006). A guide to NumPy. Trelgol Publishing USA.

1. scipy.stats - for performing statistical operations such as skew, kurtosis, and Mann-Whitney U test.

Jones, E., Oliphant, T., Peterson, P., & Others. (2001). SciPy: Open source scientific tools for Python.

1. scikit-learn

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825-2830.

* sklearn.preprocessing - for data preprocessing, specifically the RobustScaler.
* sklearn.model\_selection - for splitting data into train and test sets, grid search and cross validation.
* sklearn.naive\_bayes - for Gaussian Naive Bayes model.
* sklearn.neighbors - for K-Nearest Neighbors model.
* sklearn.linear\_model - for Logistic Regression model.
* sklearn.svm - for Support Vector Machine model.
* sklearn.ensemble - for Random Forest Classifier and Regressor models.
* sklearn.neural\_network - for Multi-layer Perceptron Classifier model.
* sklearn.metrics - for evaluating model performance, specifically classification\_report, confusion\_matrix, ConfusionMatrixDisplay, accuracy\_score, make\_scorer, balanced\_accuracy\_score, mean\_squared\_error.
* sklearn.feature\_selection - for feature selection, specifically the Chi-squared test.
* sklearn.datasets - for generating datasets, specifically make\_regression.

1. warnings - for handling warnings.
2. imblearn.over\_sampling - for oversampling techniques such as SMOTE and ADASYN. Nogueira, A. M., & Coelho, L. A. (2019). imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. Journal of Machine Learning Research, 20(10), 1-5.
3. Tkinter - for creating GUI (Graphical User Interface) applications. It provides a set of interfaces and tools for building graphical user interfaces in a Python environment.

(Tkinter, n.d.) Tkinter. Python Software Foundation. Available from https://docs.python.org/3/library/tkinter.html

**APPENDIX A: EXPLANATORY DATA ANALYSIS**

Text

Description automatically generated with low confidence

**Fig 1: Summary of some Features (Dataset Function)**

Table

Description automatically generated with medium confidence

**Fig 2. Output of *checking\_missing\_values***

**Text

Description automatically generated**

**Fig 3. Measure of Location (Mean)**

**Text

Description automatically generated**

**Fig 4. Measure of Location (Median)**

**Text

Description automatically generated**

**Fig 5. Measure of Spread (Variance)**

**Text

Description automatically generated**

**Fig 6. Measure of Spread (Standard Deviation)**

**Text

Description automatically generated**

**Fig 7. Measure of Shape (Skewness)**

**Text

Description automatically generated**

**Fig 8. Measure of Shape (Kurtosis)**

**APPENDIX B: MACHINE LEARNING METHODS**

**CLASSIFICATION I (TEST SIZE = 33%)**

**Table

Description automatically generated with medium confidence**

**Fig 1: Classification Report and Evaluation Metrics for Gaussian Naive Bayes Model**

**Table

Description automatically generated**

**Fig 2: Classification Report and Evaluation Metrics for Logistic Regression Model**

**Table

Description automatically generated**

**Fig 3: Classification Report and Evaluation Metrics for Nearest Neighbour Model**

**Table

Description automatically generated**

**Fig 4: Classification Report and Evaluation Metrics for Random Forest Classifier**

**Table

Description automatically generated**

**Fig 5: Classification Report and Evaluation Metrics for Support Vector Machine**

**Table

Description automatically generated**

**Fig 6: Classification Report and Evaluation Metrics for Multi-Layer Perceptron Neural Networks**

**CLASSIFICATION I**

**Table

Description automatically generated**

**Fig 7: Classification Report and Evaluation Metrics for Gaussian Naive Bayes Model**

**Table

Description automatically generated**

**Fig 8: Classification Report and Evaluation Metrics for Logistic Regression Model**

**Table

Description automatically generated**

**Fig 9: Classification Report and Evaluation Metrics for Nearest Neighbour Model**

**Table

Description automatically generated with low confidence**

**Fig 10: Classification Report and Evaluation Metrics for Random Forest Classifier**

**Table

Description automatically generated**

**Fig 11: Classification Report and Evaluation Metrics for Support Vector Machine**

**Table

Description automatically generated**

**Fig 12: Classification Report and Evaluation Metrics for Multi-Layer Perceptron Neural Networks**

**Chart

Description automatically generated Chart, scatter chart

Description automatically generatedChart

Description automatically generated**

**Chart, treemap chart

Description automatically generated Chart

Description automatically generated Chart

Description automatically generated**

**Fig 13: Confusion Matrix for the Six Classification Models**

**CLASSIFICATION II**

**Table

Description automatically generated**

**Fig 14: Classification Report and Evaluation Metrics for Gaussian Naive Bayes Model\_SMOTE**

**Table

Description automatically generated**

**Fig 15: Classification Report and Evaluation Metrics for Logistic Regression Model\_SMOTE**

**Table

Description automatically generated**

**Fig 16: Classification Report and Evaluation Metrics for Nearest Neighbour Model\_SMOTE**

**Table

Description automatically generated with medium confidence**

**Fig 17: Classification Report and Evaluation Metrics for Random Forest Classifier\_SMOTE**

**Table

Description automatically generated**

**Fig 18: Classification Report and Evaluation Metrics for Support Vector Machine\_SMOTE**

**Table

Description automatically generated**

**Fig 19: Classification Report and Evaluation Metrics for Multi-Layer Perceptron Neural Networks\_SMOTE**

**Chart

Description automatically generatedChart

Description automatically generated Chart

Description automatically generated**

**Chart

Description automatically generated Chart

Description automatically generated Chart

Description automatically generated**

**Fig 20: Confusion Matrix for the Six Classification Models\_SMOTE**

**Table

Description automatically generated**

**Fig 21: Classification Report and Evaluation Metrics for Gaussian Naive Bayes Model\_ADASYN**

**Table

Description automatically generated**

**Fig 22: Classification Report and Evaluation Metrics for Logistic Regression Model\_ ADASYN**

**Table

Description automatically generated**

**Fig 23: Classification Report and Evaluation Metrics for Nearest Neighbour Model\_ ADASYN**

**Table

Description automatically generated**

**Fig 24: Classification Report and Evaluation Metrics for Random Forest Classifier\_ ADASYN**

**Table

Description automatically generated with medium confidence**

**Fig 25: Classification Report and Evaluation Metrics for Support Vector Machine\_ ADASYN**

**Table

Description automatically generated**

**Fig 26: Classification Report and Evaluation Metrics for Multi-Layer Perceptron Neural Networks\_ ADASYN**

**Chart

Description automatically generated Chart

Description automatically generated Chart

Description automatically generated**

**Chart

Description automatically generatedChart

Description automatically generatedChart

Description automatically generated**

**Fig 27: Confusion Matrix for the Six Classification Models\_ ADASYN**

**CLASSIFICATION III**

**Table

Description automatically generated**

**Fig 28: Classification Report and Evaluation Metrics for Gaussian Naive Bayes Model**

**Table

Description automatically generated**

**Fig 29: Classification Report and Evaluation Metrics for Logistic Regression Model**

**Table

Description automatically generated**

**Fig 30: Classification Report and Evaluation Metrics for Nearest Neighbour Model**

**Table

Description automatically generated**

**Fig 31: Classification Report and Evaluation Metrics for Random Forest Classifier**

**Table

Description automatically generated**

**Fig 32: Classification Report and Evaluation Metrics for Support Vector Machine**

**Table

Description automatically generated**

**Fig 33: Classification Report and Evaluation Metrics for Multi-Layer Perceptron Neural Networks**

**Chart

Description automatically generated Chart, treemap chart

Description automatically generated Chart

Description automatically generated**

**Chart

Description automatically generated Chart, treemap chart

Description automatically generated Chart

Description automatically generated**

**Fig 34: Confusion Matrix for the Six Classification Models**