Project Final Report

Cardiovascular Disease Risk Prediction Model

Bootcamp: UNC-VIRT-DATA-PT-09-2023-U-LOLC-MWTH

Project by: Stuti Poudel, Aline Vo, Annie Joseph Rajan, Brian Stumm, and Jules Gikundiro

Project Overview

Purpose:

This project aims to create an effective machine-learning Model capable of accurately predicting heart disease in patients with or without health conditions and with different lifestyle factors.

Data source:

We collected this dataset from Kaggle:
https://www.kaggle.com/datasets/alphiree/cardiovascular-diseases-risk-prediction-dataset

Data cleaning and Preprocessing:

The dataset that we got is already cleaned and consists of 19 variables, which are:

- General Health
- Checkup
- Exercise
- Heart Disease
- Skin Cancer
- Other Cancer
- Depression
- Diabetes
- Arthritis
- Sex
- Age category
- Height (cm)
- Weight (kg)
- BMI
- Smoking history
- Alcohol Consumption
- Fruit Consumption
- Green Vegetable consumption
- Fried Potatoes Consumption.

We imported our CSV file into Jupyter Notebook for preprocessing and did some exploratory analysis. We changed all the categorical columns, such as exercise, heart disease, skin cancer, other cancer, depression, diabetes, arthritis, and general health into numerical columns with mapping using replace (), as this is a crucial step for preparing your dataset for machine learning algorithms, which typically require numerical input. We also converted the age category column into a new age numeric column with mapping. We then dropped the height, weight, and

categoric age columns and finally saved the cleaned dataset with 17 fields, inside the resource folder.

SQL and Tableau Visualization:

We started by creating a unique ID for each row of information. Using SQL we created two dataframes, they are:

- Consumption DB
- Non-Consumption DB

We then created two CSV files, consumptiondata.csv, and healthdata.csv, and imported them into SQL Server. Now, we have created two tables, General Health and Health Consumption, and created an Entity-Relationship Diagram (ERD). ERD is a visual representation of the relationships between entities in a database.
Using Tableau, we then created interactive visualizations and dashboards that gave us insights into the relationship between features like general health, BMI, Diabetes, smoking, exercise, and cancer with heart disease.
Some interesting observations from the dashboards are the following:

- **As the general health decreases, the risk of heart disease increases**. People with poor health have the highest chances (31.79%) of having cardiovascular /heart disease compared to other categories of general health, whereas people with excellent health have significantly less (1.99 %) chances of having heart disease.
- Increasing the BMI will increase the probability of having heart disease. Males are more likely to have heart disease as BMI increases than females.
- **1 out of 4 females who are depressed also have heart disease, while 1 out 6 males with depression also have heart disease.**13.24% and 11.20 % of total people in the dataset who are females and males, respectively, have both depression and diabetes.
- **1 out of 5 individuals with diabetes also has heart disease.** 20.85 % of the total number of people with diabetes will also have heart disease.
- **Male smokers have a higher chance of heart disease than female smokers.** Additionally, Males who are underweight and who smoke (76.92%) are more at risk of having heart disease than females who are underweight and who smoke (61.44%). Interestingly, for both males and females, underweight individuals who smoke have a significantly higher risk of heart disease than other body types.
- **The exercise report is looking at all records in the data set. Because the data is imbalanced, the percentage of individuals with heart disease is lower than anticipated.** With the data we are working with, almost 3 out of 4 individuals who exercise do not have heart disease, which is **72**.34 %, suggesting a possible protective effect of exercise against heart disease.
- The cancer report is looking at all records. It is very rare (<3%) to have cancer(s) and heart disease.

Machine Learning Models:

1) Neural Network Model:

Creating a neural network model is an iterative process that often requires experimenting with different architectures, hyperparameters, and training procedures to get the best results.

The steps taken to create the model are following:

❖ **Data Preprocessing**

- Here, Heart_Disease is the target, and all other variables are the model's features.
- Target [Heart_Disease] column was taken where the values are 0 and 1 by pd. replace () method. Then, we converted all Categorical values to numeric ones using pd. replace (). After defining the correlation matrix, we removed the column 'FriedPotato_Consumption.'

❖ **Compiling, Training, and Evaluating the Model**

- To define the model, we have chosen 2 hidden Layers with 80,30 neurons and the activation Layer is "relu" for both layers.
- In this model, we achieved an accuracy of 91.83% and a loss of 0.2223.

❖ **Optimization:**

Step: 1

- Add more hidden layers.
- Add more neurons to a hidden layer (Add more neurons for both layers).

After the optimization, we achieved an accuracy of 91.94% and a loss of 0.2243. We observed a very slight increase in accuracy.

Step:2

- Used different activation functions for the hidden layers.
- Reduce the number of epochs in the training regimen.

After the optimization, we achieved an accuracy of 92.02% and a loss of 0.2223. There was a slight increase in accuracy, but no significant improvement was found.

2) Support Vector Machine (SVM) Model

The purpose of the analysis here is to create a machine-learning model that is able to accurately predict heart disease in the patients of the provided dataset. The model that we will create for the prediction of our target variable is the Support Vector Machines (SVM) model. Support Vector Machine (SVM) is a powerful, versatile machine learning model capable of performing linear or nonlinear classification, regression, and even outlier detection. It's particularly well-suited for classifying complex but small- or medium-sized datasets.

The steps I followed to create a model using the SVM model are following:

- After importing the dependencies and reading the dataset into Google Collaboratory, some exploratory analyses are done.
- The dataset is split into training and testing data before we create a model.
- Now, the model is created using the 'kernel = rdf' parameter, and then we fit the model with our training data.
- Finally, we make predictions using our testing dataset on the trained model and evaluate the model by checking its accuracy, precision, recall, f1 score, and other parameters with our classification report.

❖ Conclusion with basic SVM model:

Our model here gives the good accuracy of 92 % . The classification report above having a good precision of 0.92 and 1.00 displays that the model is highly effective at identifying class 0, which is column with NO heart disease or patients with negative results. However, it fails to recognize class 1, which is patients with positive herat disease. The precision, recall, and F1-score for class 1 are all 0.00, indicating the model did not correctly identify any instances of class 1. The report suggests that model's high performance at the majority is misleading in this context because of the significant class imbalance (70,955 for 0 > 6,259 for 1) suggesting a need to improve the model , such as resampling by SMOTE.

Optimization 1

❖ Conclusion with SMOTE in SVM:

After applying the SMOTE our overall accuracy went down to 78% . However the classification report above displays that the precision score for class 0 (negative class / No heart Disease) is increased to 96% whereas the precision score for class 1 (positive class/Yes Heart Disease) increased to 20%, which is better compared to our original model, but not so good looking at low recall f1 score for class 1 and 0 both. Likewise, if we look at our confusion matrix we see that the model does predict the 'true negative' result in pretty good numbers, i.e, 56,205, however; the 'true positive' significantly low, i.e, 3,691. Likewise, the number of 'false negative', 2568 which is low , however; the number of false positive', which is 14,750 is also very high. In this way,considering the errors in heart disease diagnosis, specially inaccurately identifying 'true positive' could be fatal for many patients, suggesting us this is not an ideal model to go with.

Optimization 2

❖ Conclusion with Manually specifying the class weights in SVM:

(Giving class 1 a higher weight, class_weights = {0: 1, 1: 10})

Here after the weights respecifying , the accuracy reduced to 74% . The classfication report displays that the model now correctly identifies class '0'/ negative class, 97% of the time, while the precision for class 1/positive class is only increased to 21%. Likewise, the recall values and f1 score overall for both classes are little better but not yet satisfactory.The model still demonstrates an excellent ability to identify class 0 , which is patients with no heart disease instances accurately

but still struggles to do so without generating a large number of false positives for class 1 which is patients with actual heart disease.

Optimization 3:

❖ Conclusion with balanced class weights in SVM:

(model = SVC(kernel='rbf', class_weight='balanced'))

After the evaluation, we see the accuracy even droped to 71% . The classification report above again suggests that the model predicted (class '0'), people with no heart disease correctly (high precision) but tends to misclassify a notable portion of them as (class '1') people with heart disease . Though the recall is higher than before but the significant difference between the model's performance metrics for classes 0 and 1 still suggests issues with model bias, potentially due to class imbalance. In summary , while SMOTE and balanced class weights are powerful tools for combating class imbalance, their effectiveness can be influenced by the quality of features, the choice of model and its complexity, and the underlying data distribution. Further explorations and adjustments in these areas might be required to achieve significant improvements. Another strategy would be to try ensemble methods designed to handle imbalanced data, like Balanced Random Forest or EasyEnsemble.

3) Random Forest Model

A Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the individual trees' classes (classification) or mean prediction (regression). Due to the number of decision trees participating in the final decision, a random forest is considered a highly accurate and robust model. It also deals with many problems, including non-linear ones. This model has a reputation for achieving high accuracy in classification tasks and is a popular choice for the healthcare industry.

The **steps** we took to create the Random Forest Model are as follows:

Here, we created multiple decision trees using different random subsets of the data and features to reach a more accurate prediction or result.

Preprocessing:

- Examined information on data frame
- Checked for NaN values & Unique values for each column
- Examined distribution of Heart Disease to check balance
- Applied oversampling methods to handle the imbalanced data
- Applied encoding to categorical variables
- Applied feature scaling to transform numerical features into a consistent range
- Divided data into training and testing sets using the train_test_split module

Model Training:

- **Model:** We used oversampled data with the `RandomOverSampler` technique.
  - Random oversampling involves randomly duplicating examples from the minority class and adding them to the training dataset.

- Ideal for imbalanced and skewed data.
- The model resulted in a high overall accuracy rate of 98.25% and an exceptional recall for positive hits at 100%.

❖ **Optimization 1:** We used oversampled data with the `SMOTE` technique.
- SMOTE is a statistical technique for increasing the number of cases in your dataset in a balanced way. The component works by generating new instances from existing minority cases that you supply as input data.
- The optimization resulted in an overall accuracy score of 92.05% and a relatively high recall rate of 93%. The overall accuracy decreased from the original model.

❖ **Optimization 2:** We used oversampled data using the 'BorderlineSMOTE' technique.
- BorderlineSMOTE is an improved oversampling algorithm based on SMOTE. It uses only a few class samples on the border to combine new samples, thus improving the sample category distribution. It creates synthetic examples along the decision boundary between the two classes, where misclassified samples are ambiguous and in a region of the edge or border of the decision boundary.
- The optimization resulted in an overall accuracy score of 92.94% and a recall rate of 94%. The overall accuracy decreased from the original model.

❖ **Conclusion:** We used accuracy scores and classification reports to assess the performance of all three models. The RandomOverSampler Model had the best performance at 98.25% accuracy. Precision had 97% correct positive predictions. The Recall was 100%, so the model accurately predicts positive outcomes. Additionally, here, the confusion matrix only showed a 1.41% false positive rate. Ultimately, the Random Forest Classifier model that used the Random Over Sampler had the best results among the three tested models along with their optimizations, exhibiting excellent performance in distinguishing between the two classes, with strengths in both sensitivity and specificity.

Challenges and Limitations:

- **Weak to no correlation between features and target variable**: Our feature Correlation Analysis map displays all the features in our dataset that have a very weak correlation with our target variable, 'Heart Disease.' Hence, deciding which feature to drop became very challenging to ensure the remaining feature would still contribute to maximum accuracy towards the output.
- **Some Key factors missing:** According to the Centers for Disease Control and Prevention (CDC), link: https://www.cdc.gov/heartdisease/risk_factors.htm, besides physical inactivity, unhealthy diet, diabetes, excessive alcohol use, being Overweight, and obesity, there are other key factors as well, such as high Blood Pressure, High Cholesterol levels, Stress, and Family history, that contribute to heart disease. These key factors are completely missing in our dataset.
- **Complex dataset biased towards minority classes:** The complexity and incompleteness of our Dataset are another challenge we faced while completing our projects. Additionally, due to its imbalanced nature in minority classes, we had to recreate, retrain, and rerun several Models to tune them to predict our target variable effectively and accurately.
- **Inadequate domain knowledge:** Since none of the team members of this project are from a medical background, we had to do several tuning in our model in some way to enable it to perform better.
- **More binary variables than numerical variables in the dataset:** Binary variables carry less nuanced information than numerical variables, as they represent only two states. This can be a limitation when reality is more complex and cannot adequately be captured by a binary distinction. As we had more binary variables in our dataset, to effectively capture our target predictive values, we aimed to construct features—whether binary, numerical, or a mix of both—that best represent the information our final model needs to make accurate predictions.

Conclusion:

In our highly imbalanced dataset, Random Forest proved effective, achieving an impressive accuracy of 98.23% and precision scores of 1.0 and 0.97 for classes '0' (No Heart Disease) and '1' (Yes Heart Disease), respectively. This Model did an excellent job in correctly predicting the heart disease (Actual 1) 56,725 times while accurately predicting the negative cases/true negative in high numbers, i.e, 54,823 as well. The false negative and false positive is only 7 times, 1,999 times, respectively, suggesting there might still be room for fine-tuning the model to reduce these occurrences further, depending on the cost or implications of such errors in the specific application domain, which is healthcare in our case. Altogether, the extremely low False Negative rate is

particularly commendable, suggesting that the model is well-suited for applications where failing to detect positive cases has serious consequences.