Heart Disease Prediction

Introduction

Machine Learning can be used to predict outcomes from inputs in various fields. In this project, we use various features to predict if a person has a cardiovascular disease. A machine learning model can help with early detection of CVDs and will allow medical professionals to start early treatment. We will make use of the machine learning and data science libraries, sklearn and pandas to perform data manipulation and compare various classification algorithms.

Problem Formulation

Data:

Source: Heart Failure Prediction Dataset | Kaggle

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54	M	NAP	150	195	0	Normal	122	N	0.0	Up	0

Figure 1 - Example of Data Points

We will treat this problem as a binary classification problem. The data set containing patient data with 11 features will be used to predict if the patient will suffer from a CVD. Using the method, "get_dummies()" from the pandas library, I converted ChestPainType, RestingECG, ExerciseAngina, ST_Slope to one-hot vectors.

Train-Test-Validation:

Using GridSearchCV from sklearn, and train_test_split() from the pandas library, we use 75% of the data for training, and 25% for testing. GridSearchCV using KFold algorithm to perform cross validation on the training set.

Input Features:

1. Age: age of the patient (years)
2. Sex: sex of the patient (M: Male, F: Female)
3. ChestPainType: type of chest pain the patient is experiencing (TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic)
4. RestingBP: resting blood pressure (mm Hg)
5. Cholesterol: serum cholesterol (mm/dl)
6. FastingBS: fasting blood sugar (1: if FastingBS > 120 mg/dl, 0: otherwise)

7. RestingECG: resting electrocardiogram results (Normal: Normal, ST: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV), LVH: showing probable or definite left ventricular hypertrophy by Estes' criteria)
8. MaxHR: maximum heart rate achieved (Numeric value between 60 and 202)
9. ExerciseAngina: exercise-induced angina (Y: Yes, N: No)
10. Oldpeak: oldpeak = ST (Numeric value measured in depression)
11. ST_Slope: the slope of the peak exercise ST segment (Up: upsloping, Flat: flat, Down: downsloping)

Output:

1. HeartDisease: output class (1: heart disease, 0: Normal)

Classifiers:

We will compare the performance between three types of models and a trivial baseline. The trivial baseline model will be a majority guess. The three types of classifier we will compare are Logistic Regression, Neural Network (Multi-Layer Perceptron), and Support Vector Machine algorithms.

Hyperparameter Tuning:

Hyperparameter Tuning in this project is mainly done by using the GridSearchCV method from the machine learning package, sklearn. This package uses K-fold cross validation, and a scoring method that we choose to test different parameters.

1. Logistic Regression: The parameters for logistic regression will be lambda, which is constant to set the significance of the regularization term. The two types, l1 and l2, of regularizations were also tested. I used recall as the scoring method for this model as it produces the best accuracy and recall.
2. Neural Network (Multi-Layer Perceptron): The parameters for MLP include learning rate (constant, adaptive, invscaling), the type of gradient descent(sgd, adam), the activation function(relu, tanh, sigmoid), the number of neurons/ hidden layers and their organization ((100), (50, 50), (50)). Moreover, I used balanced accuracy as the scoring metric for tuning the hyperparameters for this model since using recall causes the MLP to get 1.0 recall with a very poor accuracy.
3. Support Vector Machine: The parameters for SVM are the type of kernel (linear, polynomial, sigmoid), lambda, the regularization parameter (For linear kernel), and degree (For polynomial kernel). I used balanced accuracy as the scoring metric as recall gives very poor accuracy.

Evaluation Metrics:

Since we are using these models to predict if a patient has a high of suffering from a CVD, the cost of false negative predictions is high. For example, if a patient has a high risk of suffering from a CVD but our model predicts that he likely has no CVD. Because of this, we will use a combination of F1 score and accuracy to compare the performances of the models.

Results

Trivial Baseline – Majority Guess:

	precision	recall	f1-score	support
0	0.00	0.00	0.00	112
1	0.51	1.00	0.68	118
accuracy			0.51	230
macro avg	0.26	0.50	0.34	230
weighted avg	0.26	0.51	0.35	230

Figure 2 - Performance Report of Majority Guess Model

As expected, the accuracy of this model is $\frac{Number\ of\ True\ Positives}{Total\ Number\ of\ Data\ points} = \frac{118}{230} = 0.513$.

Logistic Regression:

	precision	recall	f1-score	support
0	0.88	0.79	0.83	112
1	0.82	0.90	0.85	118
accuracy			0.84	230
macro avg	0.85	0.84	0.84	230
weighted avg	0.85	0.84	0.84	230

Figure 3 - Performance Report of Logistic Regression with l1 regularization

After comparing various parameter, GridSearchCV returns the best parameters to be l1 regularization with lambda = 1. As expected, Logistic Regression outperforms the trivial baseline. It has an 90% positive recall, 84% accuracy, and 85% f1-score.

Neural Network (Multi-Layer Perceptron):

	precision	recall	f1-score	support
0	0.92	0.77	0.84	112
1	0.81	0.94	0.87	118
accuracy			0.86	230
macro avg	0.87	0.85	0.85	230
weighted avg	0.87	0.86	0.86	230

Figure 4 - Performance Report of the MLP

The selected hyperparameters for neural network are sigmoid activation function, alpha = 0.01, hidden layers organization is two hidden layers with 50 neurons each, constant learning rate, and an adam solver. The neural network does better than logistic regression in all the metrics. It has a 94% positive recall, 85% accuracy, 87% f1-score.

Support Vector Machine:

	precision	recall	f1-score	support
0	0.89	0.83	0.86	112
1	0.85	0.90	0.87	118
accuracy			0.87	230
macro avg	0.87	0.86	0.86	230
weighted avg	0.87	0.87	0.86	230

Figure 5 - Performance Report of SVM

The selected hyperparameters for the SVM classification model are linear kernel, with the regularization parameter = 0.5. The SVM perform similar to the neural network by f1-score. However, the SVM has lower positive recall of 90% but higher negative recall of 83%. This model performed slightly better than the MLP.

Conclusion

In conclusion, the baseline model did the worst in all regards and is unusable in a real-world setting. If we are comparing the models using just f1 score without any context, it would be wise to choose the neural network or SVM over the logistic regression model. This is as expected since neural network will do as good as or better than the logistic regression model. However, since we are working to predict a disease, it is far more punishing to predict false negatives than false positives. So, comparing positive recalls, we can conclude that the neural network is best performing model in terms of usability in this context. One downside is that MLPs take longer than SVMs to train.

References:

Brownlee, Jason. “How to Configure the Number of Layers and Nodes in a Neural Network.” *Machine Learning Mastery*, 6 Aug. 2019, https://machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network/.

Kjytay, et al. “What Is Balanced Accuracy?” *Statistical Odds & Ends*, 8 Oct. 2020, https://statisticaloddsandends.wordpress.com/2020/01/23/what-is-balanced-accuracy/.

Luca, Gabriele De. “SVM vs Neural Network.” *Baeldung on Computer Science*, 25 Aug. 2021, https://www.baeldung.com/cs/svm-vs-neural-network#:~:text=What%27s%20more%20important%2C%20though%2C%20is,NNs%20tend%20to%20outperform%20SVMs.

Shung, Koo Ping. “Accuracy, Precision, Recall or F1?” *Medium*, Towards Data Science, 10 Apr. 2020, https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9.

“Sklearn.linear_model.Logisticregression.” *Scikit*, https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

“Sklearn.model_selection.GRIDSEARCHCV.” *Scikit*, https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

“Sklearn.neural_network.MLPClassifier.” *Scikit*, https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html.

“Sklearn.svm.SVC.” *Scikit*, https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.