INTRODUCTION TO FOCUS AREAS WS 22/23

Report: Data Science

Abhinav Mishra, Jule Brenningmeyer, Maike Herkenrath, Se Yeon Kim*

Correspondence:
abhinav.mishra@fu-berlin.de
Department of Mathematics and Computer Science, Freie Universität-Institute of Computer Science, Takustraße 9, 14195 Berlin, Germany
Full list of author information is available at the end of the article
*Group 7

Abstract

Project 1: Classification means *finding decision boundary* in both, statistics, and machine learning. Detrano, R et. al. proposed a population based DFA tuned logistic regression algorithm, and compared against *CADENZA* program which used Bayesian method as a classification problem [1] [2].

Our both parametric, and non-parametric classification paradigm composes of *multi-class transformation* to *predictive binary classification* accompanying decision stumps, k-nearest neighbour, generalized linear model with binomial link, and decision trees implementation using a structured dataset. [3].

Boosted logistic outperforms with the diagnostic ability of 85 % accuracy, ($AUC_{ROC} = 0.86$), expected F1 score ($AUC_{PRG} = 0.4$), and Cohen's kappa $\kappa \in (0.5, 0.6)$ [see Figure 13, 12, 14, 16] [see Table 1 & 2].

Out of 14 attributes, ca, thal, cp, and thalach were the top four most important features, that showed positive correlation in exploratory data analysis as well.

A progressed awareness rather than learning for ML pipelines e.g. creating *knitr/markdown* objects efficiently, better visualization using *ggplot2*, importance of tuning parameters, and correlation with performance metrics, understanding the data well enough before analysis, and interpretation with existing literature experimental biology.

From the beginning to end, a lump sum of **25 hours** spent on the entire project.

Project 2: To classify different breast cancer tissues as benign or magnifying a convolutional, shallow and fully connected neural network were trained. To train the neural networks light microscopy images of the data set BreaKHis were used. The resulting CNN perform quit good on the test data set while the other two networks performed just similar to a random process, which was probably caused by an uneven distributed training data set. Our personal key learnings were to use tensorflow for the first time, as well as implementing a neural network for the first time. Learning about the differences of the used neural network types and the different methodes, especially the ROC curve, to evaluate them. Overall 22 hours were spend on this project.

1 Project 1

1.1 Scientific Background

Coronary artery disease (CAD), also known as ischemic heart disease is often a first sign of a heart attack. The cholesterol plaque buildup in the walls of the arteries that supply blood to the heart (called coronary arteries) and other parts of the body is the causal relevance of it. The most common symptom is angina, and some of the diagnosis test include ECG or EKG (electrocardiogram), echo-cardiogram, exercise stress test, X-ray, cardiac catheterization, coronary angiogram, coronary artery calcium scan. Overweight, physical inactivity, unhealthy eating, and smoking tobacco are the known risk factors [4].

An international study aimed to estimate the probabilities of angiographic coronary disease based on the patient data, and pointed out that the coronary disease probabilities derived from discriminant functions are reliable and clinically useful when applied to patients with chest pain syndromes and intermediate disease prevalence [1] [5] [3].

In contrast, this project only uses the cleveland heart disease data set, which consists of 303 instances, and it is structured with both categorical and numerical attributes with only a subset of 14 attributes out of the 76 has always been used in classification problems. Each attributes contain medical information about the patients [1].

As far as finding patterns, and predicting class in a structured data are concerned, classification plays as much role as statistics played in machine learning. Hence, in this project, we impute missing values, deal with unbalanced target attribute (if there's any), visualize & explore attributes and their intra, and inter-relationships, apply four classifiers namely, logistic, random forest, boosted logistic, and k-nearest neighbour. After the model has been generated, and predicted the category for new observations based on the old ones, the model evaluation on performance metrics has been done to see if it found a decision boundary, and how well it performs within that framework, and which feature(s) attribute to that.

1.2 Goal

Performing EDA along with handling of missing values, and training at least three classifiers to diagnose heart disease based on the available data and compare them visually in a grouped bar chart with regard to accuracy, sensitivity and specificity, and then creating a confusion matrix as a heatmap, *ROC* curves for each classifier used in the prediction.

1.3 Data & Preprocessing

A processed multivariate data was used from an open source machine learning database [1]. It contains 76 attributes, including the predicted attribute, but all published experiments refer to using a subset of 14 of them, that are categorical, numeric, and integer based clinical and biomedical attributes for 303 patients, broadly classified into continuous and discrete data points as well for structuring. A detailed description of 75 attributes can be found, but some of them are mentioned below:

1 *goal* refers to the presence of a heart disease in a patient (0,1), and the predictable dependent variable.
2 *thal* refers to the thallium scintigraphy observed on patients with normal, fixed, and reversible defect (3,6,7).
3 *thalach* refers to the maximum heart rate achieved.
4 *cp* refers to the chest pain type (0-4).
5 *ca* refers to number of major vessels (0-3) colored by flourosopy.
6 *exang* refers to exercise induced angina(1 = Yes, 0 = No).

The imputation of missing values (or *NAs*) was carried out using **predictive mean matching** with *MICE* package as there were six missing values, four in *ca*, and two in *thal* [6]. Out of five iterative models generated, second one was used for appending the data, arbitrarily. *fbs*, *slope*, *exang*, *restecg* were passed as *factors* with two or three levels. *sex* was re-coded as a factor with substitution ($0 \leftarrow F, 1 \leftarrow M$). *goal* was also dealt the same ($0 \leftarrow healthy, \{1, 2, 3, 4\} \leftarrow unhealthy$). A column *ID* was added for index.

summary, and *str* functions were used to check if it cleaned the data. Now, the dataset amounted to 164 healthy samples (54 %), and 139 unhealthy ones (46 %) [7].

1.4 Methods

Packages

tidyverse, *skim*, *ggvis*, *caret*, *MLeval*, *mice*, *RColorBrewer*, *ggplot2*, *ggpubr*, *GGally* were used in the *R* script [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [6]. *caret* has a dependency on *randomForest*, *caTools* as a wrapper for functional utility [19] [20]. *pacman* was used to manage and load the packages smoothly [9].

Descriptive statistics

1 *ggplot* used to visualize the distribution of population, grouped by *goal* [21] [10].

2 *ggscatmat* used to create a scatterplot matrix with density plots on the diagonal and correlation printed in the upper triangle, grouped by *goal* [11].

3 *dplyr* & *recode_factor* used to mutate, select, and gather the categorical, and numeric imperative for bargraph, and boxplot, respectively [15].

4 *ggcorr* used to create the correlation matrix, and heatmap using pearson's correlation method [11].

Data partition

createDataPartition from *caret* package was used to split the data as the argument *goal* to this function as a factor, the **random sampling** occurs within each class and preserved the overall class distribution of the data [8]. 20% of data went to training set.

Classifiers

1 **Logistic**: logistic regression model i.e. a generalized linear model using a binomial link using the *caret* function *train()*, and classifying with no tuning parameters, but a model-specific variable importance metric is available [8].

2 **Random forest**: Breiman and Cutler's random forests for classification with *mtry* i.e. number of predictors sampled for splitting at each node as the tuning parameter using package *randomForest* as a method for *train* [8] [19] [22]. For optimum results, the number of variables randomly sampled as candidates at each split can be set as

$$mtry \sim \sqrt{p}$$

where p is number of variables in the data [23].

3 **Boosted Logistic**: An additive logistic regression model by stage-wise optimization of **binomial log-likelihood** using **decision stumps** (one node decision trees) as weak learners with *nIter* as the tuning parameter. *nIter*

is both, number of boosting iterations, and a stopping parameter for which the binomial log-likelihood is maximal. It splits training and testing phases of the classification process into separate functions using package *caTools*'s *LogitBoost* as a method in *train* [20].

4 **kNN**: returns the predicted class label with k as the tuning parameter. For each row of the test set, the k nearest training set vectors are found using **euclidean distance**, and the classification is decided by majority vote among the classes of these k vectors, with ties broken at random or estimating the posterior probabilities by the proportions of the classes among the k vectors. If there are ties for the k_{th} nearest vector, all candidates are included in the vote. *knn* was used from package *class* [23] [24].

Evaluation methods

1 *graphics*, and matrix operations used to create a comparison barchart showing accuracy, sensitivity, and specificity.
2 *caret*, *GGally*, and *ggplot2* used to create confusion matrices as heatmaps [11] [10] [21] [8].
3 *dotplot*, and *resamples* from *caret* used to create a dotplot with Cohen's kappa for parametric models, and resampled/bootstrapped plots of tuning parameters for accuracy [8].
4 *evalm* from *MLeval* used to evaluate four different models generated as *caret's train* object, and generated AUC-PR, AUC-PRG, AUC-ROC, and a binned calibration curve (or reliability diagram) [14].

1.5 Results

The analysis showed 54% healthy & 46% unhealthy samples with imbalanced density plots. Within unhealthy instances, a positive pearson correlation is observed between *cp* with *ca*, and *thal* while in healthy instances, only *thal* & *ca* had a slight non-negative occurrence [Figure 7 & 8]. After the ML training, these features are also the top four features in importance (preserved loss) in the model performance of boosted logistic [Figure 19].

Figure 9 & 10 shows the scaling of categorical (or integer), and numeric attributes, respectively pertaining to descriptive statistics. Figure 18 shows the heatmap for confusion matrices of each classifier.

Table 1 Performance metrics for classifiers

Metric	Log	RF	LogitB	KNN
Accuracy	81.97	78.69	85.25	78.69
Specificity	82.14	71.43	78.57	75.00
Sensitivity	81.82	84.85	90.91	81.82

The trained boosted logistic classifier had an 85 % accuracy with 91 % sensitivity, and 78 % specificity. It classified the unhealthy samples which are actually unhealthy better than the healthy samples [Table 1] [Figure 11]. AUC_{ROC} of 0.87 shows a good threshold for balancing true positives than false negatives [Table 2] [Figure 2]. Furthermore, an expected harmonic *F1 score* of 0.4 (AUC_{PRG}) with boosted logistic classifier is observed [Figure 16].

Table 2 AUC-ROC values

Method	AUC-ROC
Logistic	0.89
Random Forest	0.90
Boosted Logit	0.87
kNN	0.91

In practice, AUC_{PR} can easily favour worse-performing models, that's why the inter-reliability measure *kappa* statistics of more than 0.55 cannot be overlooked [Figure 15]. After re-sampling the values for accuracy's standard deviation, and standard error, random forest has the highest value because of a low error rate, but performs over-fitting that lead to low specificity [Figure 13].

The calibration plot in Figure 17 was generated after five step binning ($n = 30$). The accuracy plots after bootstrapping for all classifiers except logistic shows the downward trend for random forest & boosted logistic due to tuning parameters *mtry*, and *nIter*, respectively. At $k = 13$, the best tuned model is achieved for maximum accuracy in the respective decision boundary [Figure 12].

1.6 Discussion

As the data splitting, and imputation were randomized, we can argue that the four different classifiers performed well to a reasonable degree with good variance vs bias trade-off. Both, parametric, and non-parametric models showed slight expected over-fitting after multiple runs, but it could be overcome by changing the proportion of data partition, or changing the formula for training, altogether. A non-alignment of categorical, and factor imperative with decision-based classifier like *kNN* cause great difficulties to understand, and sometimes, can mislead the goal. Exploring and comparing re-sampling distributions between models could be improved by increasing the number of trees. Moreover, *PCA* could be of practical utility to understand the data better with quality control.

It was a good prototype for a data scientist, but a great starting point for a life sciences data consultant, depending upon if the stakeholder organization is a professional service based firm or a product one. It was tailored in a way that expected the outcome of actionable insights. It would be more typical and realistic to the job role mentioned, if it involved stakeholder or client interpersonal communication, because these things matter, in practice. More or less, the project covered the basic requirements for a data-driven decision making role like data scientist.

Gene expression data is much more compatible with the tools available in *R*, but the analysis of any infamous dataset always kick starts the skill refinement, and depth of understanding science along the implementation.

2 Project 2

2.1 Scientific Background

The Tissue of patients who suffer from cancer differs from healthy tissue, but also between different cancer types. To make a prediction if a patient is healthy or has cancer a distinction can be made by using neuronal networks. In the assignment 2 the dataset BreaKHis which consists of 7909 breast cancer tissue images from the paper "A Dataset for Breast Cancer Histopathological Image Classification" was

used [25], which is an unstructured dataset. The breast tissue images were taken by a light microscope and labeld as benign or malignant. The goal of the authors of the paper was to automatically do a classification of breast tissue which can be used in the future for clinical diagnosis.

For the second assignment three different neural nets were created to classify the images as benign or malignant. A neural network consist of an input layer, one or multiple hidden layers, as well as an output layer. There are several types of neural networks, which have specific features and advantages [26]. In the assignment a convolutional Neural Network (CNN Model), a fully connected multilayer Neural Network and a shallow Neural Network (SNN) have been used. CNN Models can identify patterns and are therefore usefull for Image analysis. They have convolutional layers as hidden layers which have a number of filters that do convolutional operations [26]. Each layers of a fully connected Neural Network is connected to all inputs of the previous layer [27]. A SNN is due to its few layers simple and therefore faster [28].

2.2 Goal

The goal was to develop and train three different predictors that classify malignant tissue based on microscopic images.

2.3 Data & Preprocessing

The images were taken from tissue samples of 82 different patients using a light microscope with four different magnifying factors (40x, 100x, 200x, 400x). Of the 7909 images 5429 images were from malignant and 2480 from benign tissue samples. The exact distribution of the images can be seen in 5. For the training and testing of the neural nets only the images with the magnifying factor of 400x were used. In addition to the data set the authors[25] also provided a python script to create five different folds of training and test images. We used only one of the folds to train and test the different classifiers with.

To makes sure that all images have the same dimensions and fit the expected input of the neural nets we resized them all to 200×200 pixels. The original dataset is divided into four subsets for different cancer types. For our classification however we only differentiated between the labels *malignant* and *benign*.

2.4 Methods

For the classification we used three different neural networks. One was a convolutional neural network consistent of two convolutional layers with max pooling, a layer to flatten the data, two dense layers with dropout, one further dense layer and a dense output layer. The other two neural networks were a SNN and a multilayer network. The SNN had one fully connected hidden layer consisting of 2000 nodes. The multilayer network had five fully connected layers with 2330, 1165, 582, 291 and 146 nodes.

The program was written in python and the libraries numpy, PIL, os, sklearn and tensorflow were used. Numpy is a library for scientific computing, providing support for multidimensional arrays and more [29]. PIL offers tools for image editing and was used to access the images [30]. The library os offers function that help for example

to access, creating or delete a folder. In the program, os was used to identify the paths of all images in the training and test folder [31]. Sklearn is a library which does predictive data analysis. For the program the OneHotEncoder of Sklearn was used [32]. The library tensorflow provides supporting functions for creating machine learning models. For the second assignment the functions Dense, Flatten, Conv2D, MaxPool2D, Dropout and Model from tensorflow.keras were used [33].

The neural nets were trained for five epochs using batch sizes of 32 images.

2.5 Results

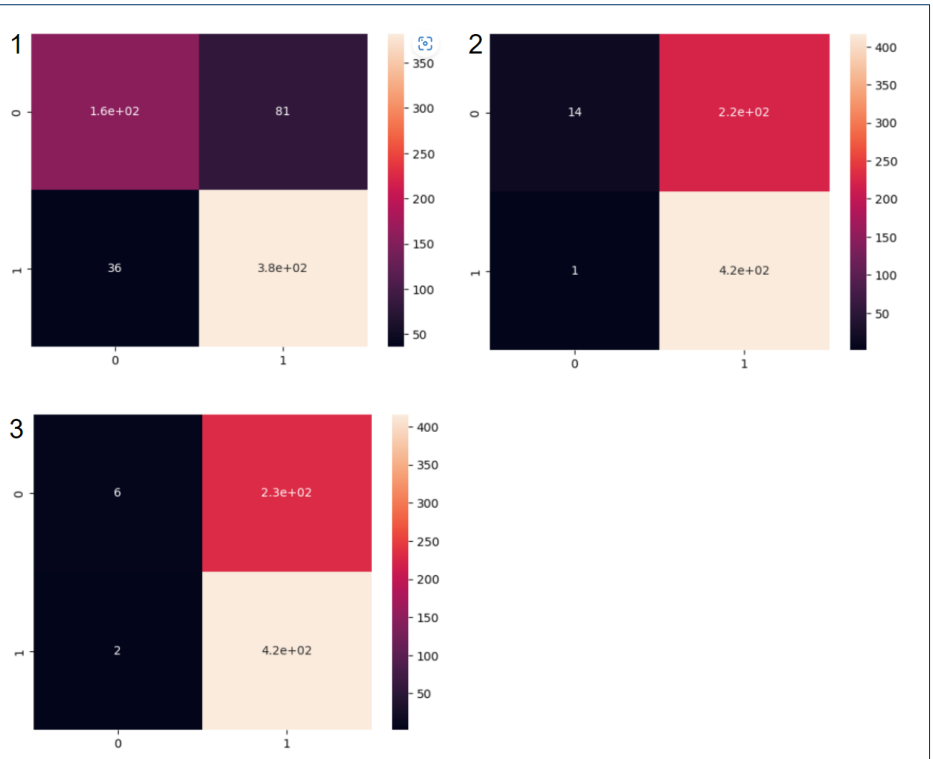

Figure 1 Confusion matrix. 1. CNN, 2. fully connected neural network, 3. SNN. the label "0" stand for benign and the label "1" stands for malignant

The three neural networks were compared by their accuracy, loss and ROC-values. Furthermore, for each neural network a confusion matrix was created for an example test run, which can be seen in Figure 1. The confusion matrix visualizes the performance of the neural networks. The confusion matrix for the CNN shows that most data points were correctly classified, while 81 data points were wrongly classified as benign and 36 data points were wrongly classified as malignant. The confusion matrices for the other two neural networks show that nearly all data points were predicted as malignant.

The accuracy of the three classifiers can be seen in Figure 2. All three networks show an overall increase of the accuracy. For the test set the CNN achieved an accuracy of 82.14%, the fully connected neural an accuracy of 65.80% and the SNN of 64.43%. The loss of CNN was low from the first epoche on, but for the SNN

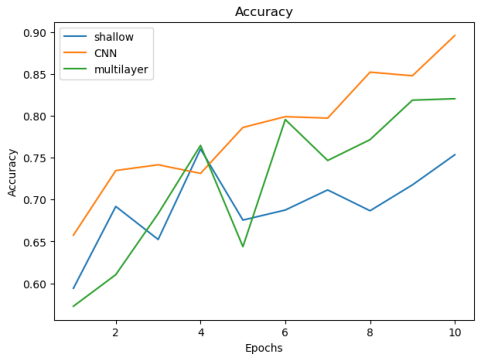

Figure 2 The accuracy of the three classifiers over ten training epochs

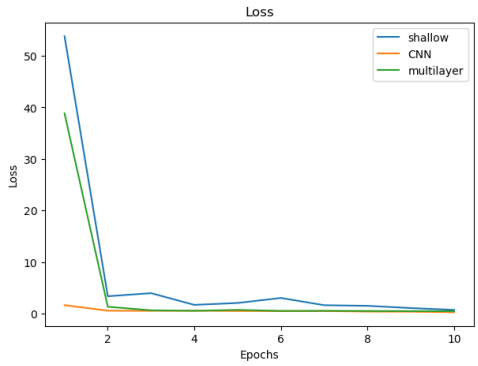

Figure 3 The loss of the three classifiers over ten training epochs

and fully connected neural network the loss was high in the first epoche before it strongly decreased. The loss of the three classifiers can be seen in Figure 3.

When looking at the ROC curves in Figure 4, it can be seen that the values of the fully connected and the SNN are close to the diagonal, while the curve of the CNN lies above the diagonal.

2.6 Discussion

The confusion matrix of the CNN shows that the neural network performed overall good, while the SNN and the fully connected network classified nearly everything as malignant. This is probably because the training data was not balanced. The larger part of images of the training data was malignant. Therefore the networks got an accuracy over 50% by classifying nearly everything as malignant. For a better performance of these networks they could be trained with a more evenly distributed training data set.

When focusing on the accuracy, it can be seen that all three neural networks show an improvement. However, CNN shows a better accuracy for the training data set and also for the test data set. This is probably due to the fact that these classifiers classified nearly everything as malignant. When looking at the ROC-curves it can be seen that because of this the curves of SNN and the fully connected neural network

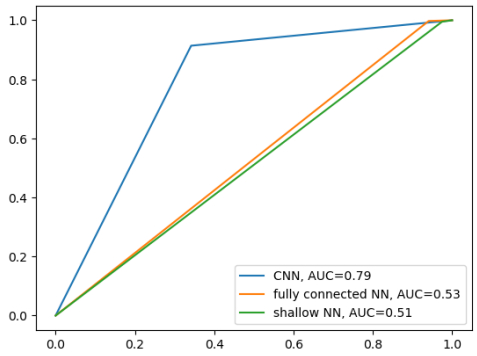

Figure 4 ROC curve

are close to the diagonal, which mean that they are similar good as a random process [34].

Over all it can be said, that the CNN performed good, while the SNN and the fully connected neural network did not preform well due to the unbalanced training data set.

Abbreviations

ML: Machine learning, **DFA**: Discriminant function analysis, **AUC-PR**: Area under the curve - precision recall, **AUC-PRG**: Area under the curve - precision recall gain, **AUC-ROC**: Area under the curve - receiver operating characteristics, **CNN**: Convolutional neural network, **kNN**: k-nearest neighbour, **EDA**: Exploratory data analysis, **SNN**: Shallow neural network, **PCA**: Principal component analysis.

Author's Contributions

1. **Abhinav Mishra** (Bioinfo) wrote the data & preprocessing, methods, results, and discussion section for project 1. He successfully compiled the R script for the analysis, and interpretation. He prepared the presentation for showcasing results for the same with help from Jule, and Maike in restructuring for the final version.
2. **Jule Brenningmeyer** and **Maike Herkenrath** (Bioinfo) wrote the section project 2, together. They assisted Se Yeon Kim with the python code for project 1. They successfully developed and trained three classifier for project 2 alongside helping with preparing the presentation.
3. **Se Yeon Kim** (DS) wrote the scientific background, and goal section for project 1. He successfully compiled a python notebook for the same, but the results have not been included in the report for consistency, and organizational reasons.

The source code and related material is hosted on FU-GitLab, and can be accessed here.
Authors declare no conflict of interest.

References

1. Detrano, R., Janosi, A., Steinbrunn, W., Pfisterer, M., Schmid, J.-J., Sandhu, S., Guppy, K.H., Lee, S., Froelicher, V.: International application of a new probability algorithm for the diagnosis of coronary artery disease. The American journal of cardiology **64**(5), 304–310 (1989)
2. Diamond, G.A., Staniloff, H.M., Forrester, J.S., Pollock, B.H., Swan, H.: Computer-assisted diagnosis in the noninvasive evaluation of patients with suspected coronary artery disease. Journal of the American College of Cardiology **1**(2), 444–455 (1983)
3. Aha, D., Kibler, D.: Instance-based prediction of heart-disease presence with the cleveland database. University of California **3**(1), 3–2 (1988)
4. Coronary artery disease. Centers for Disease Control and Prevention (2021). https://www.cdc.gov/heartdisease/coronary_ad.htm
5. Gennari, J.H., Langley, P., Fisher, D.: Models of incremental concept formation. Artificial intelligence **40**(1-3), 11–61 (1989)
6. van Buuren, S., Oudshoorn, K.G.-: mice: Multivariate imputation by chained equations in r. Journal of Statistical Software **45**(3), 1–67 (2011). doi:10.18637/jss.v045.i03
7. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2022). R Foundation for Statistical Computing. https://www.R-project.org/
8. Kuhn, M.: Caret: Classification and Regression Training. (2022). R package version 6.0-93. https://github.com/topepo/caret/
9. Rinker, T.W., Kurkiewicz, D.: pacman: Package Management For R. Buffalo, New York (2018). version 0.5.0. http://github.com/trinker/pacman

10. Wickham, H.: Ggplot2: Elegant Graphics for Data Analysis. Springer, ??? (2016). https://ggplot2.tidyverse.org
11. Schloerke, B., Cook, D., Larmarange, J., Briatte, F., Marbach, M., Thoen, E., Elberg, A., Crowley, J.: GGally: Extension to Ggplot2. (2021). R package version 2.1.2. https://CRAN.R-project.org/package=GGally
12. Kassambara, A.: Ggpubr: Ggplot2 Based Publication Ready Plots. (2022). R package version 0.5.0. https://rpkgs.datanovia.com/ggpubr/
13. Chang, W., Wickham, H.: Ggvis: Interactive Grammar of Graphics. (2020). R package version 0.4.7. https://ggvis.rstudio.com/
14. John, C.R.: MLeval: Machine Learning Model Evaluation. (2020). R package version 0.3. https://CRAN.R-project.org/package=MLeval
15. Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L.D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T.L., Miller, E., Bache, S.M., Müller, K., Ooms, J., Robinson, D., Seidel, D.P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., Yutani, H.: Welcome to the tidyverse. Journal of Open Source Software **4**(43), 1686 (2019). doi:10.21105/joss.01686
16. Neuwirth, E.: RColorBrewer: ColorBrewer Palettes. (2022). R package version 1.1-3. https://CRAN.R-project.org/package=RColorBrewer
17. Waring, E., Quinn, M., McNamara, A., Arino de la Rubia, E., Zhu, H., Ellis, S.: Skimr: Compact and Flexible Summaries of Data. (2022). R package version 2.1.4. https://CRAN.R-project.org/package=skimr
18. van Buuren, S., Oudshoorn, K.G.-: Mice: Multivariate Imputation by Chained Equations. (2021). R package version 3.14.0. https://CRAN.R-project.org/package=mice
19. Breiman, L., Cutler, A., Liaw, A., Wiener, M.: randomForest: Breiman and Cutler's Random Forests for Classification and Regression. (2022). R package version 4.7-1.1
20. Tuszynski, J.: caTools: Tools: Moving Window Statistics, GIF, Base64, ROC AUC, Etc. (2021). R package version 1.18.2. https://CRAN.R-project.org/package=caTools
21. Wickham, H., Chang, W., Henry, L., Pedersen, T.L., Takahashi, K., Wilke, C., Woo, K., Yutani, H., Dunnington, D.: Ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics. (2022). R package version 3.4.0. https://CRAN.R-project.org/package=ggplot2
22. Liaw, A., Wiener, M.: Classification and regression by randomforest. R News **2**(3), 18–22 (2002)
23. Venables, W.N., Ripley, B.D.: Modern Applied Statistics with S, 4th edn. Springer, New York (2002). ISBN 0-387-95457-0. https://www.stats.ox.ac.uk/pub/MASS4/
24. Ripley, B.: Class: Functions for Classification. (2022). R package version 7.3-20. http://www.stats.ox.ac.uk/pub/MASS4/
25. Spanhol, F.A., Oliveira, L.S., Petitjean, C., Heutte, L.: A dataset for breast cancer histopathological image classification. Ieee transactions on biomedical engineering **63**(7), 1455–1462 (2015)
26. Lei, X., Pan, H., Huang, X.: A dilated cnn model for image classification. IEEE Access **7**, 124087–124095 (2019). doi:10.1109/ACCESS.2019.2927169
27. K. Liu, N.Z. G. Kang, Hou, B.: Breast cancer classification based on fully-connected layer first convolutional neural networks. IEEE Access **6**, 23722–23732 (2018). doi:10.1109/ACCESS.2018.2817593
28. :Erichson NB, Y.Z.B.S.M.M.K.J. Mathelin L: Shallow neural networks for fluid flow reconstruction with limited sensors. The Royal Society (2020). doi:10.1098/rspa.2020.0097
29. NumPy. https://numpy.org/
30. Python Pillow. https://python-pillow.org/
31. Os - Miscellaneaous Operating System Interfaces. https://docs.python.org/3/library/os.html
32. Scikit-learn: Machine Learning in Python. https://scikit-learn.org/stable/
33. Introduction to TensorFlow. https://www.tensorflow.org/learn
34. Gönen, M.: Receiver operating characteristic (roc) curves. SAS Users Group International (SUGI), 210–231 (2006)

Appendix

Magnification	Benign	Malignant	Total
40×	625	1370	1995
100×	644	1437	2081
200×	623	1390	2013
400×	588	1232	1820
Total	2480	5429	7909
# Patients	24	58	82

Figure 5 Image distribution by magnification factor and class

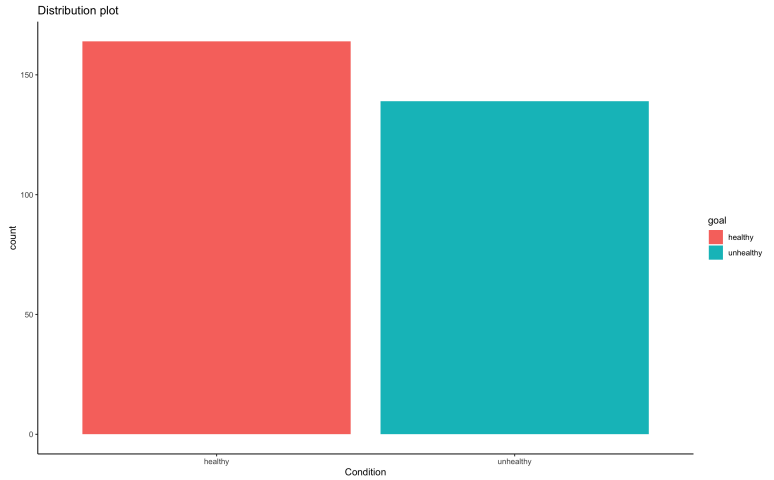

Figure 6 Distribution of samples in *heart* dataset

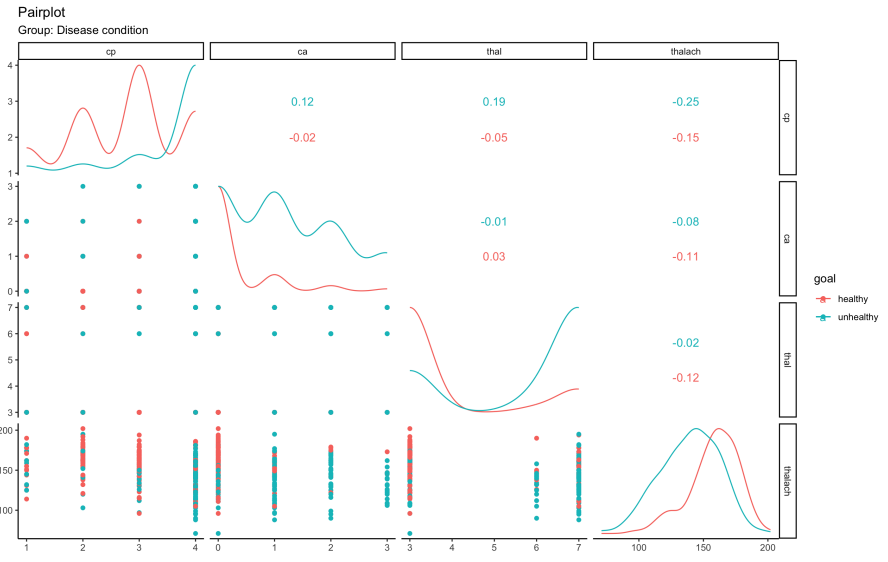

Figure 7 Scatterplot matrix for four attributes with *goal* as a grouping variable

Figure 8 Heatmap for four with *goal* as a grouping variable

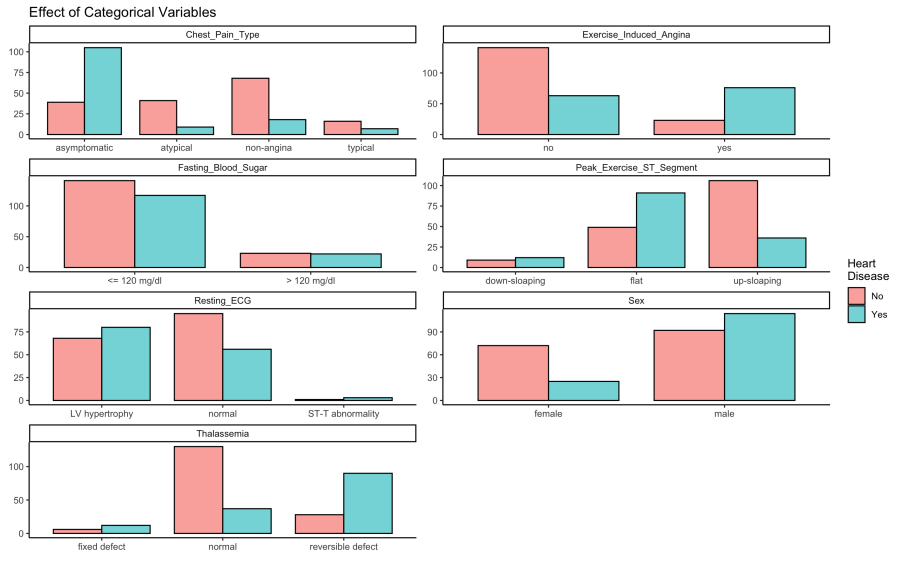

Figure 9 Distribution for categorical attributes grouped by factor levels

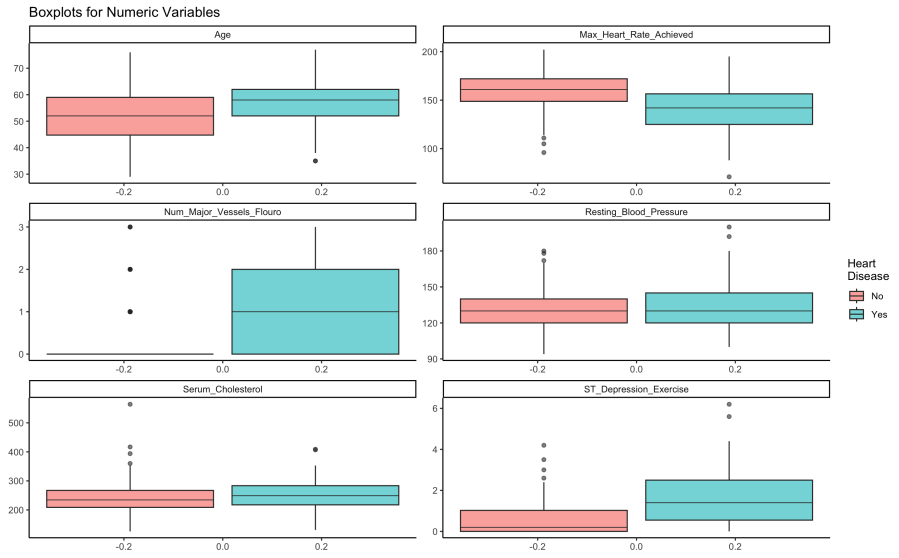

Figure 10 Distribution for numerical attributes grouped by *goal*

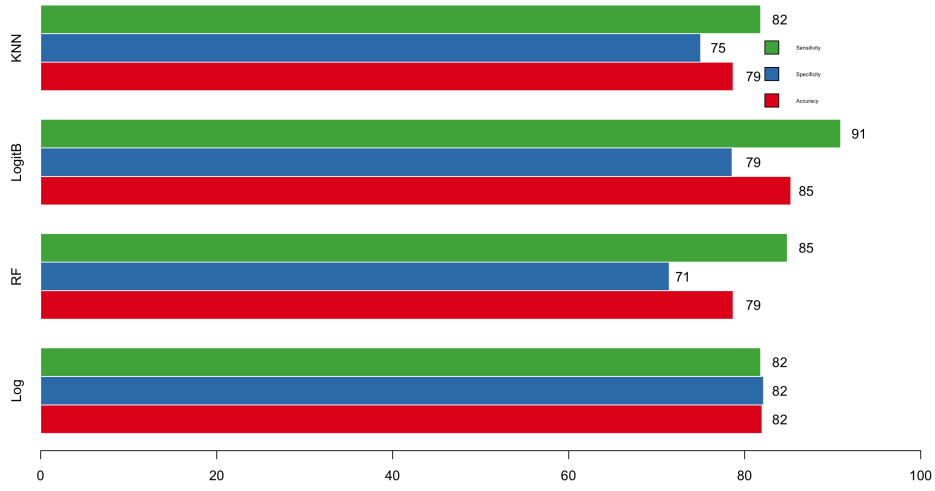

Figure 11 Barplot: Accuracy, Specificity, and Senstivity

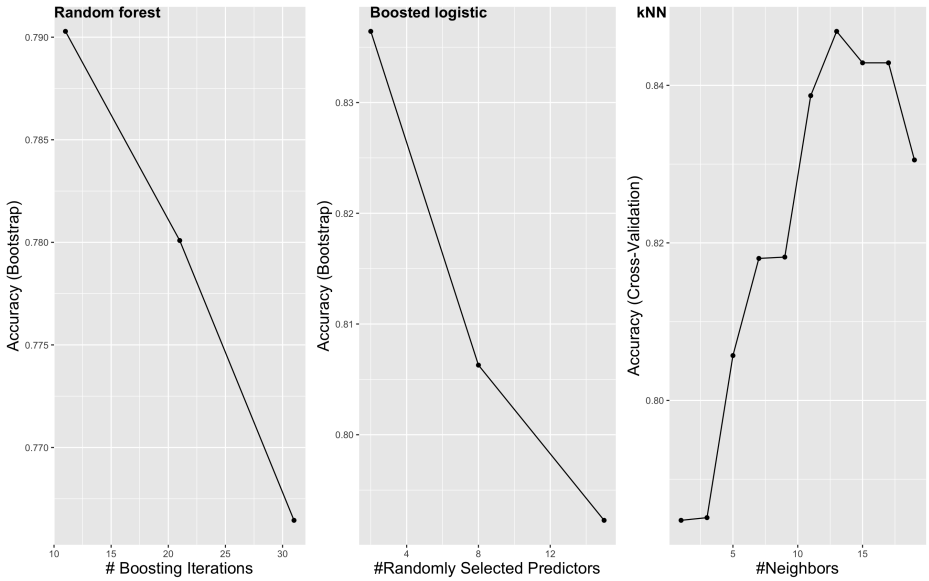

Figure 12 Accuracy plot for classifiers which used resampling and bootstrapping

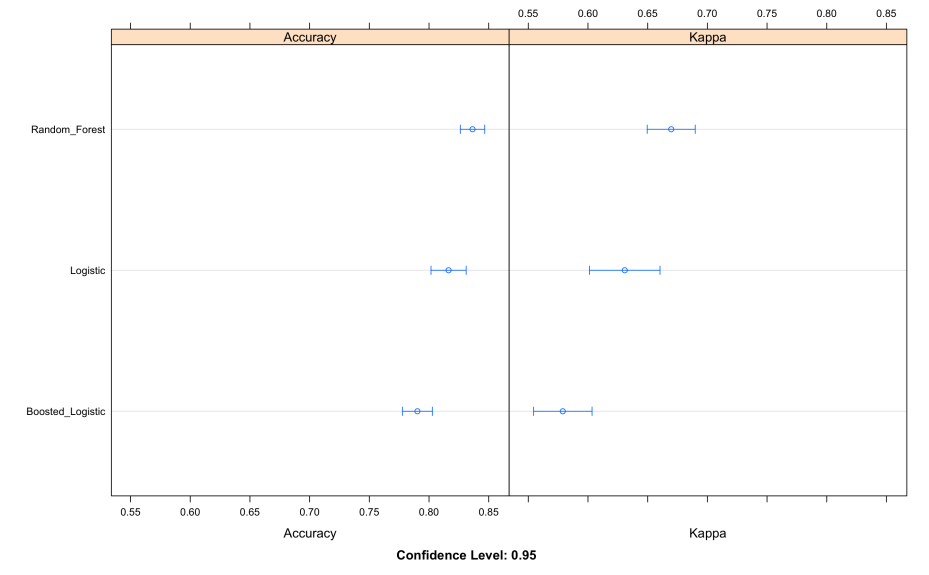

Figure 13 Accuracy dotplot for random forest, logistic, and boosted logistic model with kappa statistics

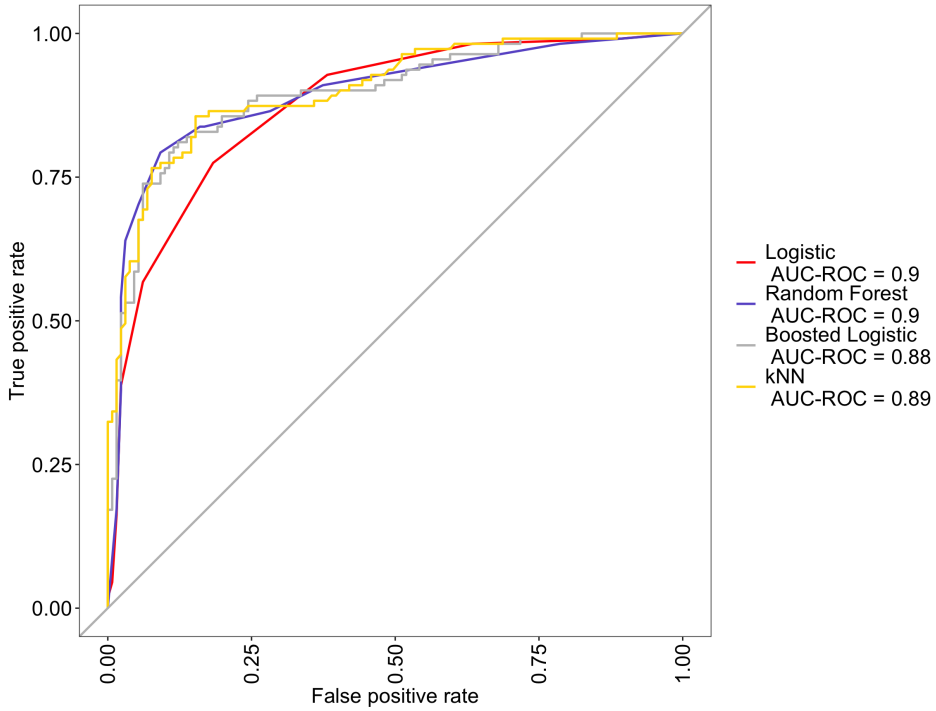

Figure 14 AUC-ROC curve

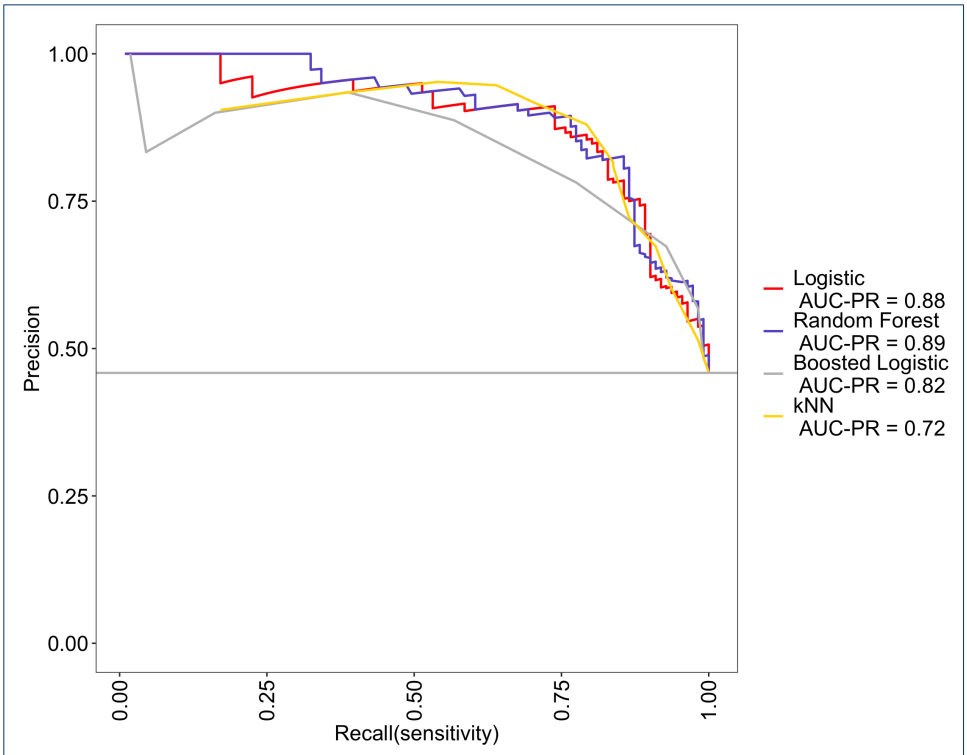

Figure 15 AUC-PR curve

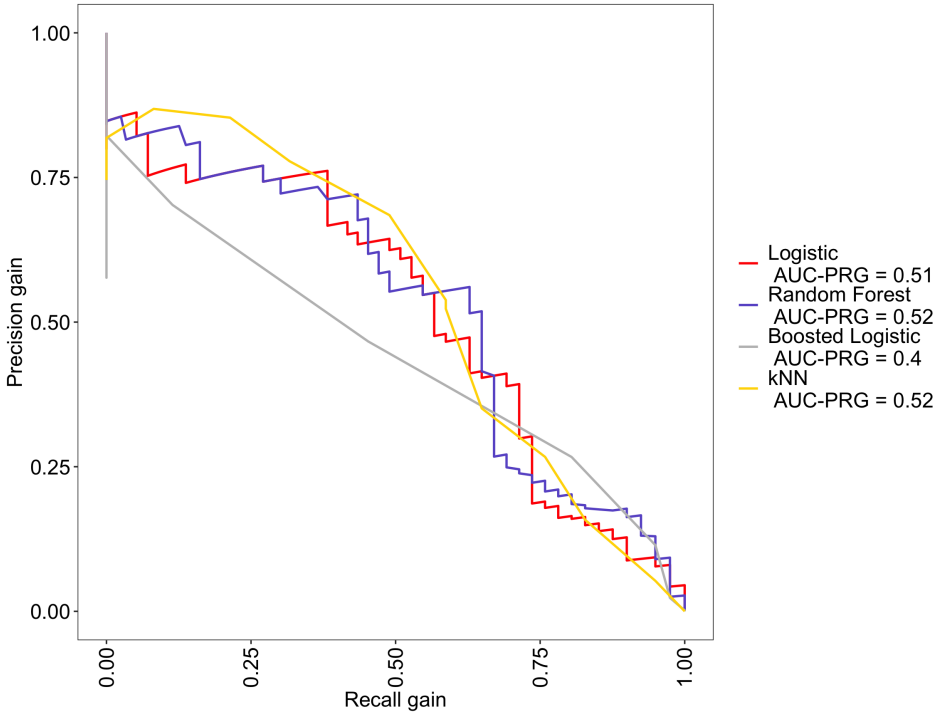

Figure 16 AUC-PRG curve

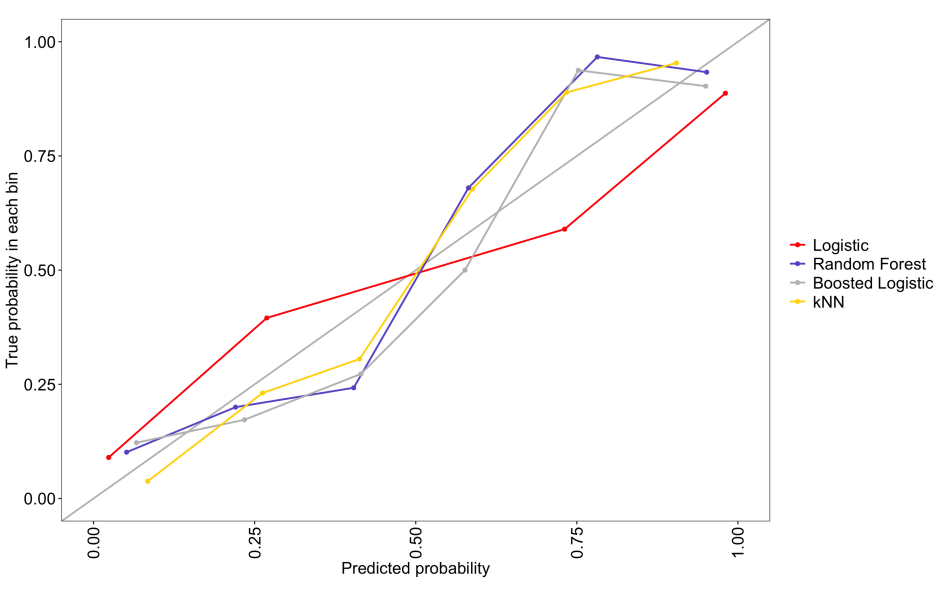

Figure 17 Calibration curve for reliability

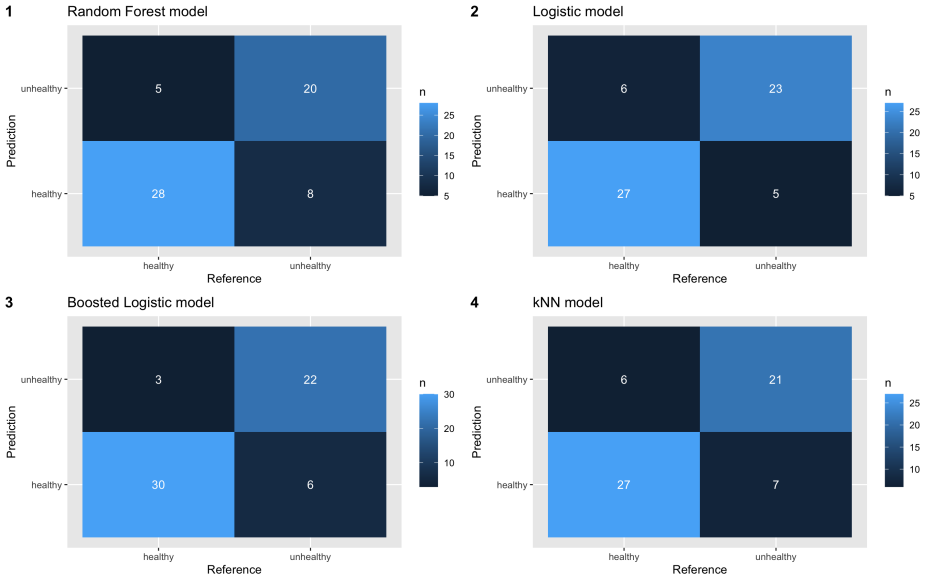

Figure 18 Confusion matrix for each classifier

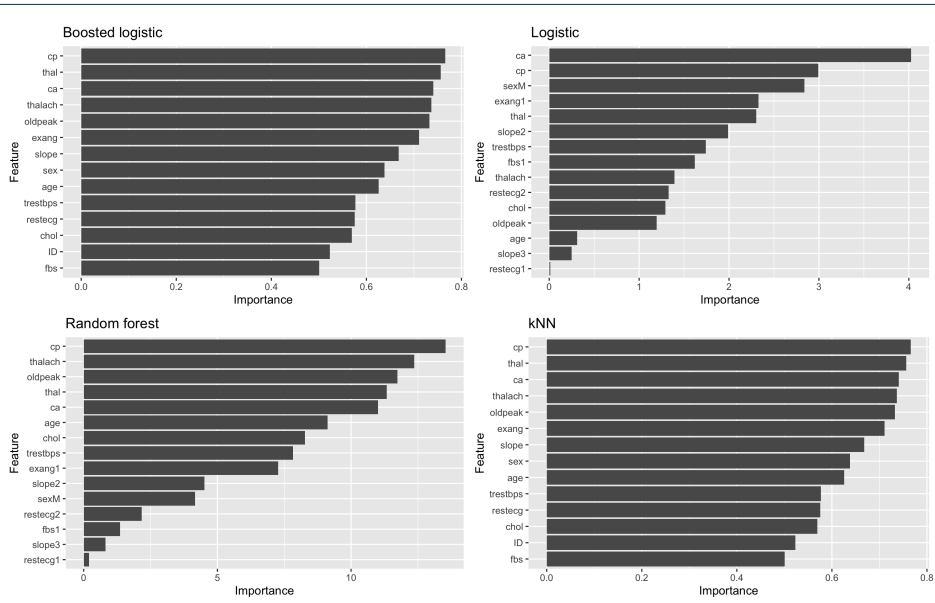

Figure 19 Top features for each classifier