Graduate Programme in Health Informatics

Assessed Coursework Submission

Student candidate number:	CHFX2
Module:	CHME0016: Machine Learning in Healthcare and Biomedicine
Date due:	Monday, 13th April 2019, 12:00 midday
Word count: (excluding references, diagrams and appendices)	2485
Disability or other medical condition for which UCL has granted special examination arrangements:	
Formative feedback:	Please address in formative feedback:
	Please ignore in formative feedback:

1 INTRODUCTION

Digital mammography is currently the most effective way to screen for breast cancer and is increasingly replacing traditional methods as the gold standard in the diagnostic and screening process (Schulz-Wendtland, 2009). However, the low positive predictive value of digital mammography caused by mammogram interpretation result in around 70% non-essential biopsies with benign masses (Elter, M., 2007). So computer-aided diagnosis systems were used to support medical professionals for a long time. Breast Imaging-Reporting and Data System (BI-RADS) is one of the quality assurance tools use with mammography and standardize report. It is meaningful to identify the severity of mammography masses to decrease a large number of non-essential biopsies.

Mammographic Mass Data Set (Elter, M., 2007) is a dataset which could use to predict the severity of a mammographic mass lesion. The features including the BI_RADS assessment, the patient's age, and the shape, margin and the density of mammographic mass. The class label is the severity field which including 516 benign (label '0') and 445 malignant (label '1') masses.

From the dataset, we could find five features. The 'BI_RADS' is an ordinal variable that represents the BI-RADS assessment ranging from 1 to 5 which mean from definitely benign to highly suggestive of malignancy. The mean value of it is around 4.3 which is mean that most of the sample is highly suggestive of malignancy. The 'Age' is the patient's age in years which from 18 to 96 and the average age of the patients is around 55. The 'Shape' is a nominal variable which represents the shape of mass, and the range of it is from 1 to 4 (round=1 oval=2 lobular=3 irregular=4). The 'Margin' is a nominal variable which means the margin of the mass (circumscribed=1 microlobulated=2 obscured=3 ill-defined=4 spiculated=5). The 'Density' is an ordinal variable from 1 to 4 which one represents high, and 4 represent fat-containing. The mean value of 'Density' is around 2.9.

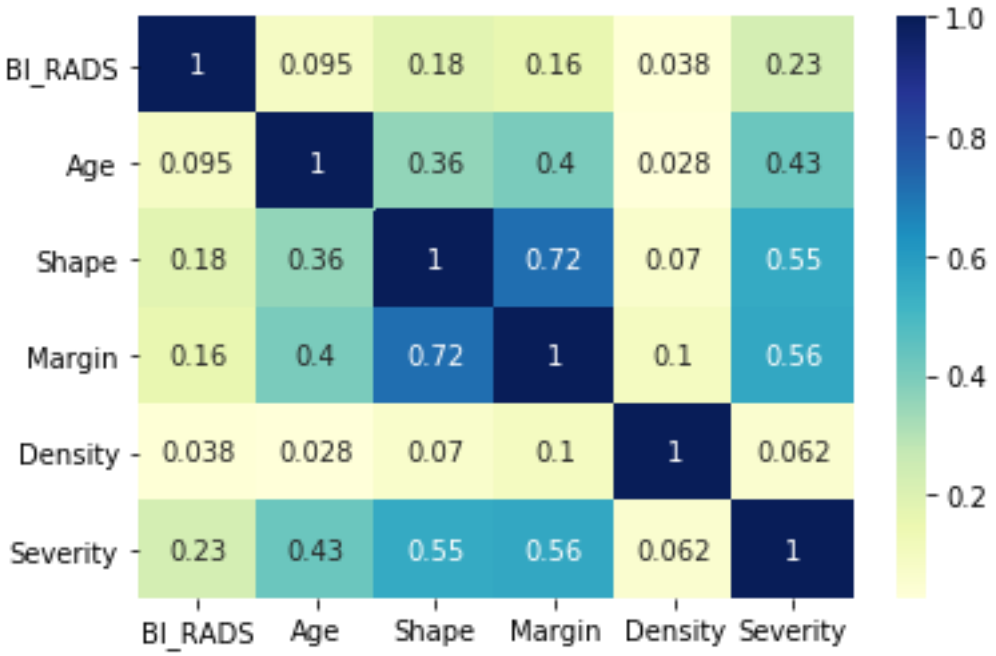

Figure1. Correlation matrix of mammographic mass data set

From Figure1, we could find the label has a strong association with shape and margin of the mass and shape is highly relevant with the margin of the mass. This information could help us in the further process of dimensionality reduction.

By visualise the feature space of the variables, we could find there have some missing value and noise (outlier) in the dataset. So we need to clean the data before any further analysis of the underlying characteristics.

2 METHODOLOGY

By summarizing the data, we could find an outlier in 'BI_RADS'. The 'BI_RADS' is an ordinal variable which the range from 1 to 5, but we could find the value of the 340th sample is 55. As there only have one outlier, we could delete this sample.

Furthermore, we could find missing value exist in all the features, and the number of it in each variable is 2, 5, 31, 48, 76. From Figure2, we could find part of the sample has more than one missing value. However, as the label of all the samples is complete, we could attempt to fill the missing value. In this

project, we attempt to fill in the missingness by the average value of each variable. After cleaning the data, we get 960 complete samples for further research.

Figure2. Missing Value in mammographic masses Data set

By visualise the feature space between each of the variable, we could find none of the variables can be used to identify the severity individually. However, we could find that the distribution of BI_RADS and margin in different severity is significantly different. Most of the BI_RADS of benign masses is 4, and the value for malignant masses is 4 or 5. This is clinical explainable because the higher level of BI_RADS is more tend to unnecessary biopsies. The density looks like useless because the density of both benign and malignant masses is low (value 3) which could be considered to exclude in order to reduce the dimension of the feature.

In the feature pre-processing process, we use the Min-Max Feature scaling to normalize the data. This process could help us to avoid the effect of extreme distribution. Also, it could allow us to compute the ANOVA F-value of each variable as the score of it for the feature selection process. We estimate the degree of linear dependency between the label and each of the feature. The null hypothesis of the test is the variance of the value of the variable in the positive sample (label 1) is the same as the variance in the negative sample (label 0). The larger the F value, the more reason we reject the null hypothesis and select the feature for prediction. By using this method, we could find sort the features by the F-value, and the result is shown in Table1.

Table1. F-Value for feature selection

Feature	F_value
Margin	440.95
Shape	423.85
BI_RADS	326.29
Age	219.62
Density	3.61

From the Table1 we could find the F-value of Margin and Shape is significantly higher than other features and the result of Density in line with our previous speculation that the impact on the density to prediction is small. For the feature BI_RADS and Age, we finally decide to include BI_RADS only based on the results of following validation process.

By visualise the feature space of the choosing feature 'Margin', 'Shape' and 'BI_RADS' in a 3-dimensions space, we could find the benign masses tend to have a lower value of margin and shape and malignant masses are opposite. The effect of BI_RADS is hard to determine, and we need to use the prediction model to identify the type of masses.

The total number of the instance is 960, and we choose 20% of the data for the testing process. The number of instances for training is 768, and the number of instances is 192. We could also find the number of positive instances for training is 356, and the negative sample is 412 which mean that our training data set is imbalance. By using over sampling methods, we could get the same number of positive instances as the negative sample, and the final number of training instances is 824.

The two supervised learning methods using this data is Decision Tree and Multi-layer Neural Network.

A decision tree is a tree structure where the test on an attribute is represented by each internal node, each branch represents an output, and the class label is kept in each leaf node. The classification process of decision tree first gives a tuple with an unknown class label and tests the attribute values of the tuple on the decision tree. The most common three classification trees are ID3, C4.5 and CART tree. The attribute selection measures method for them is information gain, gain ratio, and Gini index. Based on the attribute

selection measures method and allow a path from the root to the leaf node where holds the prediction class for that tuple we can get the prediction class. The pseudo-code of the decision tree is shown in Figure3 (Han, J, 2001).

Algorithm: Generate_decision_tree. Generate a decision tree from the training tuples of data partition, *D*.

Input:

- Data partition, *D*, which is a set of training tuples and their associated class labels;
- *attribute_list*, the set of candidate attributes;
- *Attribute_selection_method*, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting_attribute* and, possibly, either a *split-point* or *splitting subset*.

Output: A decision tree.

Method:

```
(1)  create a node N;
(2)  if tuples in D are all of the same class, C, then
(3)      return N as a leaf node labeled with the class C;
(4)  if attribute_list is empty then
(5)      return N as a leaf node labeled with the majority class in D; // majority voting
(6)  apply Attribute_selection_method(D, attribute_list) to find the "best" splitting_criterion;
(7)  label node N with splitting_criterion;
(8)  if splitting_attribute is discrete-valued and
         multiway splits allowed then // not restricted to binary trees
(9)      attribute_list ← attribute_list − splitting_attribute; // remove splitting_attribute
(10) for each outcome j of splitting_criterion
     // partition the tuples and grow subtrees for each partition
(11)     let Dj be the set of data tuples in D satisfying outcome j; // a partition
(12)     if Dj is empty then
(13)         attach a leaf labeled with the majority class in D to node N;
(14)     else attach the node returned by Generate_decision_tree(Dj, attribute_list) to node N;
     endfor
(15) return N;
```

Figure3. Pseudo-code of the decision tree from training tuples

A multilayer neural network including an input layer, few hidden layers, and an output layer. Each layer is constructed by several neural. The inputs to the NN correspond to each of the features of the training data. These features pass through the input layer and are then weighted and fed to the first hidden layer immediately. The outputs of the hidden layer units can be input to another hidden layer or to the end which is the output layer and each neural in the output layer represent a class label. This is mean that the number of neural in the input layer is the same as the feature number and the number of neural in the output layer

is the same as the number of class. During the training process, the network learns by adjusting the weights to improve the prediction performance, and this process is called backpropagation. Backpropagation learns by iteratively processing the training data and comparing the model's prediction with the final label of the training set. For each training tuple, the training process is an attempt to minimize the mean-squared error between the prediction result and the label of the data. So the optimizer which calculates the error is an essential part of the NN model. The pseudo-code of the multilayer feed-forward neural network is shown in Figure4 (Han, J, 2001).

Algorithm: Backpropagation. Neural network learning for classification or numeric prediction, using the backpropagation algorithm.
Input:

- D, a data set consisting of the training tuples and their associated target values;
- l, the learning rate;
- *network*, a multilayer feed-forward network.

Output: A trained neural network.
Method:

(1) Initialize all weights and biases in *network*;
(2) **while** terminating condition is not satisfied {
(3) **for** each training tuple $\boldsymbol{X}$ in D {
(4) // Propagate the inputs forward:
(5) **for** each input layer unit j {
(6) $O_j = I_j$; // output of an input unit is its actual input value
(7) **for** each hidden or output layer unit j {
(8) $I_j = \sum_i w_{ij} O_i + \theta_j$; //compute the net input of unit j with respect to the previous layer, i
(9) $O_j = \frac{1}{1+e^{-I_j}}$; } // compute the output of each unit j
(10) // Backpropagate the errors:
(11) **for** each unit j in the output layer
(12) $Err_j = O_j(1 - O_j)(T_j - O_j)$; // compute the error
(13) **for** each unit j in the hidden layers, from the last to the first hidden layer
(14) $Err_j = O_j(1 - O_j) \sum_k Err_k w_{jk}$; // compute the error with respect to the next higher layer, k
(15) **for** each weight w_{ij} in *network* {
(16) $\Delta w_{ij} = (l) Err_j O_i$; // weight increment
(17) $w_{ij} = w_{ij} + \Delta w_{ij}$; } // weight update
(18) **for** each bias θ_j in *network* {
(19) $\Delta \theta_j = (l) Err_j$; // bias increment
(20) $\theta_j = \theta_j + \Delta \theta_j$; } // bias update
(21) } }

Figure4. Pseudo-code of the multilayer feed-forward neural network

Based on the Decision Tree and Multilayer Neural Network, we have several hyper-parameters that need to be optimized. For Decision Tree, we need to decide the splitting rules, the depth of decision tree,

minimum samples to split, max number of features to consider and so on. For Multilayer Neural Network, we need to find the most suitable hidden layer, neural number, optimizer, L2 penalty parameter, batch size, learning rate, Learning rate schedule, iterations, whether early stopping, beta1, beta2, and several other hyper-parameters. To optimize the hyper-parameters, we could use the grid search. By providing a list of value to choose from, grid search can attempt all possible combinations of the hyper-parameters and provide the best combinations on the evaluation method you choose.

The model evaluation method we used including AUC, accuracy, precision, recall, and f1. This evaluation process is based on the confusion matrix. The AUC is mean the Area under the Curve of ROC which the X-axis is false positive rate and the Y-axis is true positive rate. The precision is the number of true predictions on the positive sample (TP) divided by the total number of positive predictions. The Recall is the TP divided by the total number of positive samples. The F1 is double the recall multiplied by precision and divided this number by recall plus precision. The accuracy is the total number of true predictions divided by the total number of samples.

3 RESULTS

To optimize the hyper-parameters, we use the grid search, and the criterion is to find the model with the highest accuracy.

In the decision tree, we tried both entropy and Gini index as the splitting rules. We also tried the depth of the decision tree from 1 to 10. The highest accuracy on the validation set was obtained in CART tree which using Gini index, and the depth of the tree is 6. The structure of the decision tree is saved in the 'CHFX2_Decision_Tree.png'.

To evaluate the robustness of our model, we used 10-fold cross-validation The result of the sensitivity analysis of our final model is 0.85 (± 0.04). The sensitivity analysis result with different parameters is shown in the Table2.

Table2. The sensitivity analysis for decision tree with 10-fold cross-validation

splitting rules	max_depth	Mean_recall	Std_ recall	Mean_ROC	Std_ROC
gini	6	0.86	0.05	0.90	0.03
gini	2	0.85	0.09	0.88	0.04
entropy	6	0.86	0.05	0.90	0.03
entropy	2	0.79	0.13	0.88	0.04

Based on the sensitivity analysis, we could find the effect of splitting rules to the final result is significant. In the 10-fold cross-validation, the model with entropy and depth=2 have lower average recall, and the standard deviation of the ten validations result is much higher than another model which is mean that the robustness of the model prediction results is low. However, the effect to the AUC is slight, and we could find a higher max depth tend to have a higher and robustness AUC.

By testing the final CART tree on the testing set, we could find the accuracy is 82.29%, the AUC is 0.823, the precision is 0.793, the recall is 0.830, and the F1 is 0.811.

In the multilayer neural network, the optimizer we attempt including quasi-Newton methods (‘lbfgs’), stochastic gradient descent (‘sgd’), stochastic gradient-based optimizer (‘adam’). The maximum number of iterations we attempt including 1000, 4000 and 16000. The L2 penalty parameter we attempt 0.1 and 0.001. The learning rate we attempt including 1, 0.1 ,0.01, 0.001, 0.0001 and 0.00001. The structure of the hidden layer we attempt including (8,8,8,8), (16,16,16,16) and (2,4,8,4,2) and the number represent the number of neurons in each hidden layer. The highest accuracy on validation set was obtained when using L2 penalty parameter=0.1, hidden layer = (8,8,8,8), the maximum number of iterations = 1000 and the optimizer using the quasi-Newton methods.

Table3. The sensitivity analysis for multilayer neural network with 10-fold cross-validation

optimizer	Max_iter	L2	Learning Rate	Hidden Layer	Mean_recall	Std_recall
lbfgs	1000	0.1	0.001	(8,8,8,8)	0.84	0.04
sgb	1000	0.1	0.001	(8,8,8,8)	0.84	0.05
adam	1000	0.1	0.001	(8,8,8,8)	0.83	0.05
lbfgs	10000	0.1	0.001	(8,8,8,8)	0.84	0.04
lbfgs	1000	0.0001	0.001	(8,8,8,8)	0.84	0.05
lbfgs	1000	0.0001	0.001	(4,4,4,4)	0.60	0.16
lbfgs	1000	0.1	0.001	(2,2,2,2)	0.50	0.00
lbfgs	1000	0.1	0.001	(4,4,4,4,4,4,4)	0.82	0.05

By using 10-fold cross-validation, the result of the sensitivity analysis of this model is 0.84 (± 0.04). The sensitivity analysis result with different parameters is shown in the Table3. From the sensitivity analysis result, we could find the effect of several hidden layers, and the number of neural in each hidden layer has a significant effect on the robustness of our model. This is explainable because an unsuitable setting of hidden layers can result in overfitting or underfitting.

By testing the final multilayer neural network on the testing set, we could find the accuracy is 83.33%, the AUC is 0.833, the precision is 0.811, the recall is 0.830, and the F1 is 0.820. The confusion matrix is shown in Table4.

Table4. Confusion Matrix of the multilayer neural network

	Predicted: Negative	Predicted: Positive
Actual: Negative	TN = 73	FP = 15
Actual: Positive	FN = 17	TP = 87

4 DISCUSSION AND CONCLUSION

In other research, they attempt to use the C4.5 decision tree which using the gain ratio as the splitting rules. It is furthermore using a pruning technique that prevents overfitting by cleaning decision tree. It also

contains extensions that allow samples to have missing attribute values which are very important in our dataset because the missing value in the set may influence the result of our model. The AUC of the C4.5 algorithm is 0.87±0.01 (Elter, M., 2007).

Bayesian classifiers have a good performance in this dataset as it can deal with the missing values and provided the probability distribution of the class with the particular input feature. Markov Blanket estimation algorithm is used to construct the Bayesian classifiers as it ensures that every attribute in the dataset is in the Markov blanket of the node that represents the class attribute (Witten, I.H., 2005). The prediction accuracy of Markov Blanket estimation algorithm is around 0.906 (Elsayad, A.M., 2010).

Bootstrap sampling is also used to evaluate the performance of the model (Mavroforakis et al., 2006), and the bootstrap set is created by randomly selecting n samples from the data set with replacement. It could help us better evaluate the performance and accuracy of our model.

To improve my methodology, I attempt to employ new optimizer in the neural network. By using Tensorflow, I tried Adam algorithm (Kingma, D.P., 2014), Adaptive Gradient algorithm, RMSProp algorithm, Momentum algorithm, and gradient descent algorithm to improve the model. The performance of the model based on Tensorflow is shown in the last part of the ipynb file.

By using the prediction model, CAD systems could better support medical professionals to decrease the proportion of unnecessary biopsies with benign outcomes. It could also reduce the r time and cost of a clinic. However, the machine learning method is a ‘black box' which is hard to explain the reason of decision, and it is hard to be used directly unless the False-Positive result is zero because the consequences of missed diagnosis are irreparable.

REFERENCES

Elsayad, A.M., 2010. Predicting the severity of breast masses using Bayesian networks. 2010 The 7th International Conference on Informatics and Systems (INFOS), pp.1–9.

Elter, M., Schulz-Wendtland, R. & Wittenberg, T., 2007. The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process. Medical Physics, 34(11), pp.4164–4172.

Han, J. & Kamber, M., 2001. Data mining: concepts & techniques. /Jiawei Han., San Francisco; London: Morgan Kaufmann.

Kingma, D.P. & Ba, J., 2014. Adam: A Method for Stochastic Optimization.

Mavroforakis et al., 2006. Mammographic masses characterization based on localized texture and dataset fractal analysis using linear, neural and support vector machine classifiers. Artificial Intelligence In Medicine, 37(2), pp.145–162.

Schulz-Wendtland et al., 2009. Digital mammography: An update. European Journal of Radiology, 72(2), pp.258–265.

Witten, I.H. & Frank, E., 2005. Data mining: practical machine learning tools and techniques /Ian H. Witten, Eibe Frank. 2nd ed., San Francisco, Calif.: Morgan Kaufman.