Classifying Breast Cancer Using Supervised Learning Techniques

Jonathan Harper

September 2018

1 Introduction

In this report, I outline my answers to the questions outlined in Section 1.1. Code produced for this task will be included separately to this report.

1.1 Tasks Overview

Using the breast cancer dataset provided by Wisconsin University[1], complete the following tasks:

1. Tell us something useful about the difficulty of classifying this data.
2. Build two classifiers and compare them.
3. What would you do if you come across a dataset that contains missing values.

1.2 Dataset Used

There were two different datasets provided by the University of Wisconsin. One contained missing values (breast-cancer-wisconsin.data), the other did not (wdbc.data).

As the specification of this test specified that our data should not in fact include missing values, the dataset with no missing values (wdbc.data) was selected for use.

1.3 Notes on the Data

Our data set is comprised of 569 observations. There are 32 attributes listed in Appendix A; they include ID, diagnosis, and 30 real-valued input features. We have labelled data, which can belong to one of 2 classes, malignant' or 'benign'. The 'diagnosis' field of the dataset specifies which of these two classes each observation belongs to.

2 Question 1

Classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. Classifier performance depends greatly on the characteristics of the data to be classified.

The main difficulties in classifying this dataset were the presence of irrelevant features, highly correlated values, and the size of the dataset.

2.1 Irrelevant Features

There were some irrelevant features in the dataset, such as 'ID' that needed removing. These features would only add to the dimensionality of the dataset, making it more computationally expensive, and decrease the accuracy if the classifier.

2.2 Highly Correlated Features

In linear models, multi-colinearity can result in outputs that are very varying and possibly numerically unstable. The presence of highly correlated features can make the learning algorithm must slower. Removing these decreases dimensionality and can result in improvement in terms of speed. Highly correlated features can also introduce harmful bias. Furthermore, not being directly related to the target variable they were not useful. Highly correlated features can reduce the interpretability of the model.

[1]https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data

2.3 Size of Dataset

The dataset as a whole is fairly small. If the training dataset itself is small, overfitting is more likely to occur.

A final point to consider is the fact that the values were not normalised. When data used for classification is not normalised, the different ranges of the different columns can introduce bias in the classifier.

3 Question 2

Implemented 2 popular classification models: logistic classification and k-nearest neighbour (kNN).

3.1 Logistic Regression

Essentially, uses the sigmoid function, otherwise known as the logistic function, to make binary predictions based on the linear combinations of independent variables onto a range between 0-1. In this case, the decision bound is located at 0.5, that is when a prediction of 0.6 is made, it corresponds to a final prediction of 1 (the cancer is malignant).

3.1.1 Results

Logistic classification had an accuracy of 97%. The epoch count (the number of iterations taken to train the model) was 1,000,000. The learning rate (limitation to the variance of the weights after each step) was 0.01. Finally, the correlated threshold (which describes the maximum correlation before features are considered highly correlated) was set to 0.95.

3.2 k-Nearest Neighbour

kNN is a non-parametric, lazy learning algorithm, based on feature similarity. Non-parametric meaning that kNN does not make any assumptions on the underlying data distribution. Lazy refers to the fact the kNN does not use the training data points to do any generalisation. Meaning there is no explicit training phase, which makes training very fast, but can be computationally expensive over larger training datasets. When a prediction is required for a unseen data instance, the kNN algorithm will search through the training dataset for the k most similar instances. kNN involves a similarity measure, where the distance between two data instances is calculated and used to find the k most similar neighbours. As out dataset consisted of real-valued data, the Euclidean distance was used.

3.2.1 Results

The kNN classifier was significantly less accurate that logistic classification. It had an accuracy of approximately 63%. The maximum number of neighbouts for each point was set to 4. A ratio of 80/20 for train/test split of the dataset was used. It has previously been found that kNN provides improved results after irrelevant attributes are removed from the data set, but still performs less optimally than logistic classification[2].

[2]https://pdfs.semanticscholar.org/24b0/2fc8b438d9a432ad72111ef2d80f8c148c1c.pdf

4 Question 3

Missing data can introduce bias in a model and make the handling and analysis of the data less efficient. Unfortunately there is no best way to deal with missing values in a dataset, however outline here are some considerations and approaches that I would take when dealing with missing data.

Missing data is representative of the disorder and messy nature of real world data. There are various reasons ranging from human errors during data collection, to instrumental errors (such as incorrect sensor readings). The first step in dealing with missing values should be to consider the reasons why they might be missing in the first place.

There might be a random reason or a natural tendency for values to be missing. There are two forms of randomly missing values. Missing completely at random and missing at random. In the former there is no pattern in the missing data on any variables; the latter case there is a pattern in the missing values but not it is not dependant on any of the primary features.

There is a third case, where there is a pattern in the missing data that may affect the primary dependent variables. For example, lower-income participants are less likely to reveal their salary. This category of missing values is called missing not at random, and is one of the more difficult to deal with.

4.1 Deletion

Deletion is one possible solution for randomly missing data (missing at random and missing completely at random). If the missing values are very few (perhaps less to 10% of the dataset) and there is no underlying reason as to why they might be missing, then it could be safe to simply delete the observations with missing values. If, in contrasts the number of missing values is significant, or there appears to be some underlying reason for the data point being missing, then deletion would not be an option.

In the cases where deletion is an option, there are two possible approaches, list-wise and pair-wise deletion. List-wise deletion only takes into consideration observation with complete data, and discards any observations with missing values. The advantages here are the simplicity of the solution and that it allows for fair comparison of analyses. However, this approach would reduce statistical power as it decreases the number of observations. This could result in biased estimator if there is an underlying reason for the missing values.

Sample size is something important to take into account when considering deletion. If the sample is large enough, then we likely can drop data without substantial loss of statistical power. But the values must be missing at random and we must be sure we are not inadvertently removing a class of observations.

Pair-wise deletion is when the observations that contain some missing data are used in the statistical analysis. Pair-wise deletion uses as much information as possible and keeps as many cases as possible in the analysis. but you can't compare analyses as the sample sizes would be different Pairwise deletion could result in different numbers of observations contributing to different parts a model, which can make interpretation difficult. If an observation has a missing value for a very important feature, it probably cannot be used in the analysis. Some features may play a bigger role than others. When considering how to deal with missing data it is important to consider whether the missing data point is important for good separation of classes.

Most of the time, it can be more beneficial to include incomplete data. Dropping variables should only be considered if the value is insignificant to the analysis being carried out.

4.2 Imputation

An alternative to deletion is imputation. Imputation is the process of replacing missing data with substituted values. For example, filling in missing values with 0. Data imputing is a way to avoid some issues introduced by methods like list-wise deletion.

Imputation works best when many variables are missing in small proportions such that a complete case analysis might render 60-30% completeness, but each variable is perhaps only missing 10% of its values.

Imputation usually doesn't introduce many additional assumptions in most analyses, and can be quite robust. In longitudinal studies, however, we must take care to assess data to determine how informative the missing values are.

There are numerous approaches to data imputation. Mean and mode substitution has some attractive properties for univariate analysis but becomes problematic for multivariate analysis. Mode imputation almost definitely introduces bias into a model, and when means are used those values usually won't display any relationship to other observations.

If data has been collected for human subjects, it could be possible to recover missing values by contacting the participants and asking them to fill out the missing values.

Another approach would be to treat missing values as a separate category. We can create another category for the missing values and use them for analysis.

A possibility would be to create a predictive model to estimate values that will substitute the missing data. In this case, we could also divide our data set into two sets: One set with no missing values for the variable (training) and another one with missing values (test). We can use methods like logistic regression and ANOVA for prediction)

4.2.1 Regression

Regression Substitution uses multiple-regression analysis to estimate missing values. This approach predicts the missing value from the other values. The most likely approach I would take is to take the regression route and take advantage of correlations between variables and observations.

4.3 Time Dependent Data

The data that was used in this project was not time series depended. However, in a time depended dataset, the following are some ways that I would consider to deal with missing data.

Could use the observations are other points in time to imput missing values. Two common approaches are the "Last Observation Carried Forward" and the "Next Observation Carried Backward". This approach could be used in the analysis of longitudinal repeated measures data, where observations may be missing. Longitudinal data tracks the same sample at different points in time. Unfortunately, this approach can introduce bias in analysis and perform poorly when data has a visible trend.

Other approaches include linear interpolation, which is an approach that works well for time series data with some trend but is not suitable for seasonal data. In this case an seasonal adjustment and linear interpolation is an approach that works well for data with both time series trends and seasonality

A basic approach woul be to use the mean, median or mode imputation method. It is a fast solution, however it would not be taking advantage of the time series characteristics or relationship between the variables. but has clear disadvantages. Another disadvantage is that mean imputation reduces variance in a dataset.

A Dataset Features

1. ID
2. Diagnosis (M = malignant, B = benign)
3. Mean radius
4. Mean texture
5. Mean perimeter
6. Mean area
7. Mean smoothness
8. Mean compactness
9. Mean concavity
10. Mean concave points
11. Mean symmetry
12. Mean fractal dimension
13. Radius standard error
14. Texture standard error
15. Perimeter standard error
16. Area standard error
17. Smoothness standard error
18. Compactness standard error
19. Concavity standard error
20. Concave points standard error
21. Symmetry standard error
22. Fractal dimension standard error
23. Worst radius
24. Worst texture
25. Worst perimeter
26. Worst area
27. Worst smoothness
28. Worst compactness
29. Worst concavity
30. Worst concave points
31. Worst symmetry
32. Worst Fractal dimension