Data Mining and Decision Systems  
600092  
Assigned Coursework Report

Student ID: 201601628  
Date: 07 October 2019

## Due Date: 12 December 2019

**Report must be within 8 page maximum. Strict page limits will be enforced. Any extra pages will be ignored and no marks awarded for any work on these. Exclusions to this limit are the front page, the references section, and any appendices. Please keep to the given section headings and format; subsections are permitted.**

# Methodology

**Introduction**

This project looked to solve a classification problem revolving around patient medical data – classifying records (patients) as “Risk” or “NoRisk”.

A context-adapted CRISP-DM methodology was followed, limiting the focus on the, “Business Understanding” and “Deployment” phases, but still utilising its iterative nature. This document outlines the processes used whilst adhering to this framework, the results achieved, and a discussion of findings.

1. **Business Understanding**

This phase sought to understand and summarise the problem at hand, giving due consideration to the domain based on the project specification and personal domain knowledge.

Particular attention was given to medical terminology, leading to assumptions and considerations such as the following:

* The ‘Indication’ feature had a category called ‘a-f’, which stands for ‘arterial fibrillation’. This is a type of arrhythmia, so it was assumed that records with ‘a-f’ as the indication ought to also be marked as positive in the ‘Arrhythmia’ feature.

It was later found that there are many arrhythmias with varying severities, so the assumption changed to view the ‘Arrythmia’ feature as, “one more severe than a-f”.

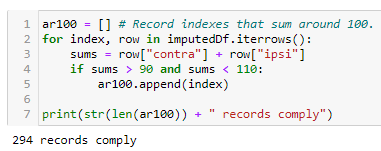
* ****The ‘IPSI’ and ‘Contra’ features refer to the percentage of lesions on the same and opposite sides (of *something*) respectively. The context of the problem appears to relate to strokes, which can happen on a particular side of the body. Therefore, the possibility that IPSI and Contra should total 100% (as a ratio) was considered, but this didn’t hold in the data, with only 294 records coming close.

Figure 1: This looked for records where the sum of IPSI and Contra was around 100%.

1. **Data Understanding**

This phase focused on gaining a high-level understanding of data correctness and patterns in the data. Most time was spent in this phase since a thorough understanding of the data allows for more effective choice of tools and quicker reasoning in the face of poor results.

Conformity to the Data Dictionary

The data dictionary provides an *idealistic* overview of the given data but shouldn’t be (and wasn’t) taken at face value. A python object was created to describe each of features in the data dictionary, creating a collection of expected columns and values, to enable automated checking.

The key findings were:

* The session column which was mentioned, separate to the dictionary, was not present.
* Contra values were string representations of numbers. These were converted to numbers.
* Indication had differently formatted categories (e.g. “Asx” and “ASx") for the same class. Further domain research suggested there was no difference, so they were merged.
* The Random feature was supposed to be unique, but there were 298 duplicate values.
* The Label feature had two records classified as “Unknown”. These were imputed in one data frame and dropped in another, for comparison,

An interest was taken in the possibilities of the id, random, and session features; session data could be encoded *within* random or id, or it could be that the features are mislabelled.

To explore this idea (that random may be patient id), all the records with duplicated ‘Random’ values were inspected for multiple changes to the history and diabetes features. If these values changed multiple times, it would indicate that the Random feature couldn’t possibly be the patient id. None of these contradictions appeared, however, supporting the possibility and leading to the maintained assumption that ‘Random’ could be the patient id and ‘Id’ might be the session id.

Duplicates

Checks for duplicates included code that looked for records with **all but one** attribute the same, and **all but two** attributes the same.

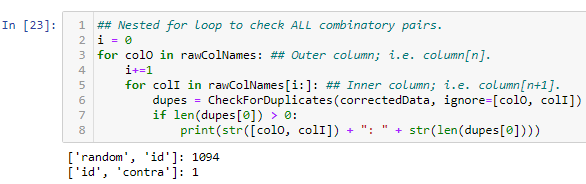


Figure 2: Duplicate finding code and outputs.

The most meaningful finding from these checks was the apparent homogeneity of the data set; when the only two (supposedly) unique columns were excluded from the checks, two thirds of the records were considered duplicated.

Aside from the this, a single record was found when Id and Contra features were ignored, and they appeared only 2 records apart (by index). Assuming human input, it’s possible that values in this column were mistakenly entered. However, carrying the consideration surrounding ‘Random’ and ‘Id’, it appeared that this could be 2 visits from the same patient:

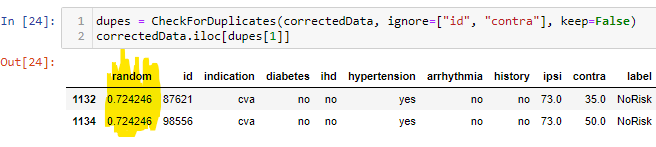


Figure 3: The possible outliers might be two visits from the same patient.

As a single datapoint, a note was made of its existence but the effect of removing it was expected to be negligible.

Missing Data

Checks for missing data were made especially thorough after falling afoul to hidden, missing values that were string literals - which after visualising the indication column. To remedy this, the normal checks (isna()/isnull()) were used alongside regular expressions that looked for values that were blank strings or some case-insensitive version of, “NaN”, and the analyses were re-run.

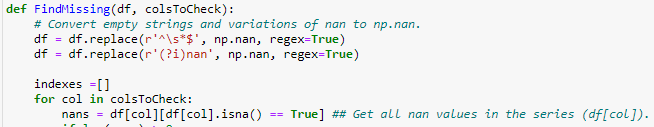


Figure 4: Regular expressions in null-finding code.

This revealed a total of 18 records with missing data. Recalling the homogeneity that the dataset demonstrated in the search for duplicates, a method for imputing values based on the nearest neighbours was created.

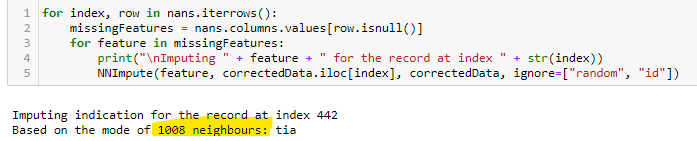


Figure 5: The mode was used to impute because, for example, this record has over 1000 neighbours.

Distribution and Outliers

These checks involved looking at statistics and plots of features to find class imbalances and distribution of the target class.

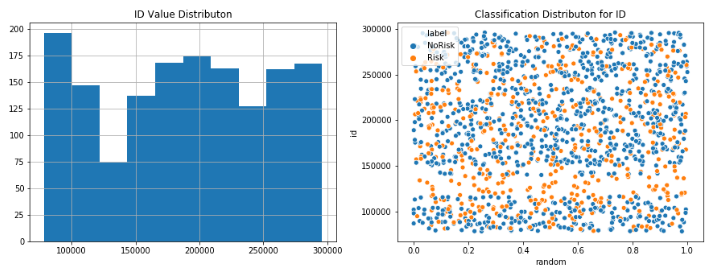


Figure 6: Risk cluster (right) when plotting multivariate graphs with Id.

They revealed the existence of a risk-only cluster that appeared when distributing any feature against Id. A number of causes are plausible. Since the datapoints are so sparse in that region, it appears as though the “NoRisk” cases could actually be missing. Alternatively, ‘Id’ may indeed be the patient identifier, and that cluster could have been part of a study group for a clinical trial.

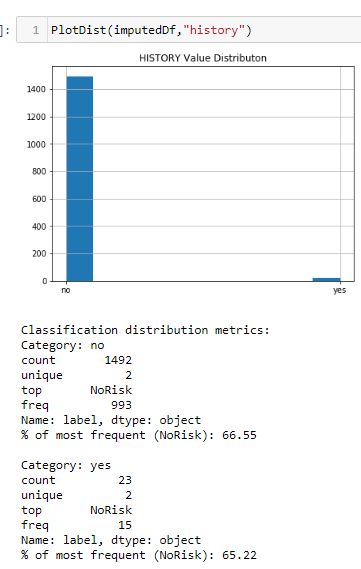
The effect of removing this cluster was evaluated, but there were no significant changes to feature correlations.

Figure 7: History feature imbalances.

There was also several instance of massively imbalanced. One such example is the history feature, which had an imbalance of 1492:23 records.

It was decided that extreme cases like these were best removed from the training data, as they don’t give a good enough representation of the expected distributions.

1. **Data Preparation**

This section served as a convergence point for the various renditions of data that had been imagined in the data understanding phase, as well as transforming and cleaning the models for processing.

Transformation

Most of the data was converted to a binary representation because of the binary categories (e.g. “yes”, “no”). Furthermore, IPSI and Contra were normalised as percentages, by a simple division of 100.

Based on the exploration of the indication feature, two methods of encoding were used. One saw the feature binarized, based on evidence that patients with “cva” or “tia” indications appeared to have a slightly reduced risk, relative to the other 2 classes (asx and a-f). Potential reasoning for this relates to the understanding that asx and a-f are actually precursors of cva (cardiac arrest) and tia (mini stroke).

The other method used the more convention, “get\_dummies” to one-hot encode the 4 values.

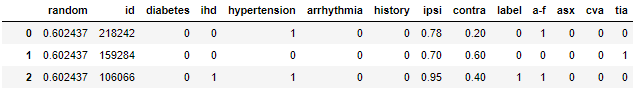


Figure 8: Transformed data, with normalisation and one-hot encoding.

Feature selection

In addition to the features proposed based on the data understanding activities, a random forest classifier was used to weight the features and provide a more objective viewpoint.

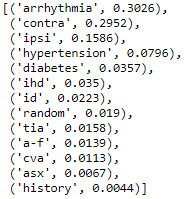


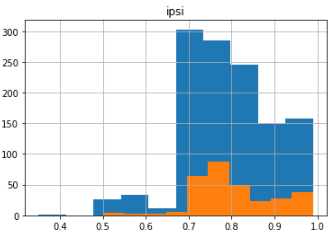
Figure 9: Ordered feature importance’s from the random forest classifier.

The features that were output were ordered as they were manually, with the exception of the now one-hot encoded, “indication “classes. However, their reduced weight is perfectly reasonable when considering 1 feature has essentially been divided by 4. Diabetes was not selected because of the class imbalance it exhibited, as explained in the previous section.

Stratification

To split the data into train and test sets, random stratification was used with a 70:30 train: test split and the random seed was modified until the visualisations presented a similar distribution in both the train and test sets.

Additionally, to generate a confidence score of each model’s ability to handle the full extent of the dataset, stratified k-fold was used to create 5 balanced folds in a similar way. 5 was chosen for the number of folds so that the training data was still substantial (approximately 300 records).



1. **Modelling**

A stepwise approach was taken to the modelling phase, first using k-fold, cross validation to “score” the performance of baseline models (a simple, Logistic Regression Classifier) with different variations of the data.

This was done with the intention of measuring the quality and suitability of each dataset, so that only the most optimal datasets needed to be processed by the real models. By doing so, the focus shifts from perfecting the data to perfecting the model via hyperparameter tuning.

# Results

**Data Variations**

|  |  |
| --- | --- |
| **ID** | **Description/Sample** |
| **1** | All features, all missing values imputed. |
| **2** | Selected features dropped ("id","random", "diabetes", "history"), all missing values imputed (EXCLUDING “label”). |
| **3** | Selected features dropped ("id","random", "diabetes", "history"), all missing values imputed (INCLUDING “label”). |
| **4** | Same as 2, indication changed to binary category. |
| **5** | Selected features dropped ("id","random", "diabetes", "history"), all missing values dropped. |
| **6** | **Same** as 5, “ihd” feature also dropped. |

**Baseline Model (Logistic Regression)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Data Variation ID | Hyperparameters | Cross Validation Score | True Negatives | False Positives | False Negatives | True Positives |
| 1 | default | 0.6654 | 1.00 | 0.00 | 1.00 | 0.00 |
| 2 | default | 0.9349 | 0.95 | 0.11 | 0.05 | 0.89 |
| 3 | default | 0.9432 | 0.96 | 0.12 | 0.04 | 0.88 |
| 4 | default | 0.9453 | 0.97 | 0.13 | 0.03 | 0.87 |
| 5 | default | 0.9467 | 0.97 | 0.12 | 0.03 | 0.88 |
| 6 | default | 0.9480 | 0.97 | 0.11 | 0.03 | 0.89 |

**SVM (using data 6)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model ID | Hyperparameters | Cross Validation Score | True Negatives | False Positives | False Negatives | True Positives |
| 1 | default | 0.9540 | 0.98 | 0.09 | 0.02 | 0.91 |

**Decision Tree (using data 6)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model ID | Hyperparameters | Cross Validation Score | True Negatives | False Positives | False Negatives | True Positives |
| 2 | max\_depth = 5 | 0.9587 | 0.97 | 0.08 | 0.03 | 0.93 |
| 3 | max\_depth = 7 | 0.9740 | 0.98 | 0.06 | 0.02 | 0.94 |

**Random Forest (using data 6)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model ID | Hyperparameters | Cross Validation Score | True Negatives | False Positives | False Negatives | True Positives |
| 4 | max\_depth = 4 | 0.9527 | 0.99 | 0.09 | 0.01 | 0.91 |
| 5 | max\_depth = 7 | 0.9707 | 0.99 | 0.04 | 0.01 | 0.96 |

**MLP (using data 6)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model ID | Hyperparameters | Cross Validation Score | True Negatives | False Positives | False Negatives | True Positives |
| 6 | Layers = [10],  Epochs = 2000 | 0.9633 | 0.97 | 0.60 | 0.03 | 0.94 |
| 7 | Layers = [10,10,10],  Epochs = 2000 | 0.9700 | 1.00 | 0.06 | 0.00 | 0.94 |
| 8 | Layers = [10,10,10, 10],  Epochs = 2000 | 0.9706 | 0.98 | 0.06 | 0.02 | 0.94 |
| 9 | Layers = [30,30,30, 30,30],  Epochs = 2000 | 0.9700 | 0.99 | 0.05 | 0.01 | 0.95 |

# Evaluation & Discussion

**5. Evaluation**

Except for the dataset that hadn’t had any features removed, all baseline models performed remarkably well - to the extent that improving on them was difficult (90%+ cross-validation score, with comparable sensitivity and recall).

This can probably be attributed to the simplicity of the cleaned dataset. As already discussed, the data is quite homogenous with only id and random removed, so processing even smaller feature subsets further reduces the possible patterns/search-space – to the point that complex algorithms are redundant because a person could digest the same amount of data.

This had its benefits however, because it meant decision tree classifiers were especially effective, even with a small, “max\_depth”. This means the deployable solution doesn’t have to be a “black-box” algorithm, which is valuable in the healthcare domain because of the high-stakes (dealing with people’s lives). This transparency is also helpful in having the algorithm approved (e.g. FDA) as a medical device, since the decision process is clearly explainable and likely reasons similar to an expert.

1. **Deployment**

The recommended model is model 5, random forest. Random forest helps reduce the overfitting that is inherent to decision trees. This model also has extremely good true positive values and true negative values which are important in healthcare for both saving patients and saving funds; treatments can be expensive, especially in clinical trials – so it’s important to be able to correctly identify patients to receive treatment.

**Project Evaluation**

In considering the methodology, particularly how it unfolded for this problem, there are several possible improvements that could be made.

Firstly, the medium used (i.e. Jupyter) wasn’t necessarily the best option. It could at least have been managed better. A suggestion would be to export some of the utility functions into external scripts which are imported into the notebook. This would lead to cleaner a more coherent and process. Similarly, the top-down flow of a Jupyter notebook makes it difficult to follow the iterative paths of CRISP DM without repeating headings or cluttering areas of the notebook.

Focusing on actual stages of the methodology:

* Business Understanding:

The business understanding could have been improved greatly if there was some degree of access to a domain-expert or the client/business.

* Data Understanding:

Especially after discovering the risk-cluster related to Id, it would have been useful to explore clustering methods. There is also a lack of multivariate exploration beyond looking at the distribution of ‘label’ within the feature.

* Data Preparation:

Having “clean” data in the data-understanding section to create meaningful visualisations made the preparation phase a little redundant. It may have been simpler and more effective to treat the understanding and preparation phases as one in the context of this project.

One additional activity that would have been interesting, would have been to experiment with aPriori for feature selection, since the datapoint a relatively homogenous.

* Modelling:

The models portion of the notebook quickly became unkempt, with confusion matrices littered everywhere. In hindsight, the individual model generation (and hyperparameter tuning) would be much more manageable if it was done external to main notebook.

Furthermore, additional visualisations that display performance metrics of multiple models in one plot, would have been a big help in understanding and conveying differences.