# Data Mining and Decision Systems 600092 Assigned Coursework Report

Student ID: 201603759

Date: 07 October 2019

Due Date: 2 January 2020

**Report must be <u>within</u> 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

This project involves the analysis of legacy data which provides insight on whether a patient is at risk or not regarding certain features/health conditions of the individual. The task here is to preprocess the data, train this cleaned data with various models, and hence produce/fit a model that would provide accurate predictions/classifications that could be used to determine the label (risk) of a patient given certain input features.

This project should follow the CRISP-DM methodology; however, this is not 100% compatible with the university setting as there is no business client/customer and we are unable to deploy this model in the real-world in this situation. This lack of customer means there is a limited business understanding, as you are unable to contact them and ask questions regarding the data and its formatting. Thus, this process follows a slightly modified version of CRISP-DM.

**Initial state of the data**

| **Attribute** | **Initial Value Type** | **Stated Value Type*** | **Number of null/missing values** | **Description**** |
|---|---|---|---|---|
| Random | float | Real | 0 null values<br>Many duplicated values | Real number of help in randomly sorting the data records |
| Id | Integer | Integer | 0 null values | Anonymous patient record identifier: should be unique values unless patient has multiple sessions |
| Indication | Object (Nominal) | Nominal | 3 null values<br>Use of 'ASx' and 'Asx' | What type of cardiovascular event triggered the hospitalization? |
| Diabetes | Object (Nominal) | Nominal | 2 null values | Does the patient suffer from diabetes? |
| IHD | Object (Nominal) | Nominal | 0 null values | Does the patient suffer from coronary artery disease (CAD), also known as ischemic heart disease (IHD)? |
| Hypertension | Object (Nominal) | Nominal | 3 null values | Does the patient suffer from hypertension? |
| Arrhythmia | Object (Nominal) | Nominal | 0 null values | Does the patient suffer from arrhythmia (i.e. erratic heartbeat)? |
| History | Object (Nominal) | Nominal | 2 null values | Has the patient a history of Cardiovascular interventions? |
| IPSI | Float | Integer | 4 null values | Percentage figure for cerebral ischemic lesions defined as ipsilateral |
| Contra | Object | Integer | 0 null values<br>1 empty whitespace string | Percentage figure for contralateral cerebral ischemic lesions |
| Label | Object (Nominal) | Nominal | 3 null values<br>2 values labelled as 'Unknown' | Is the patient at risk (Mortality)? |

*Stated Value Type is the value type of the attribute stated in the original data description table for the legacy data. With many of these attributes, they may reach their stated value type by the removal of a null value or empty whitespace string (in the case of ‘Contra’). The value types may also change from their stated type when preprocessing the data in order to obtain a more accurate model. This could also include the use of dummy variables.

** Taken directly from the data description table.

The use of the ‘*.info()*’ function on the data provides a brief summary for the data set including each attributes type and how many non-null (and hence also null) values for each. In total, there are 17 null values within the data, or 20 if you include the Contra empty whitespace string and the 2 ‘Unknown’ string Label values.

As seen above, there is a significant amount of preprocessing that needs to be performed on the data before it can be trained/modelled. This includes the changing of value types, imputing of missing data, and potentially dropping columns such as Random due to its insignificant impact on the patient risk and duplication of values as discussed later.

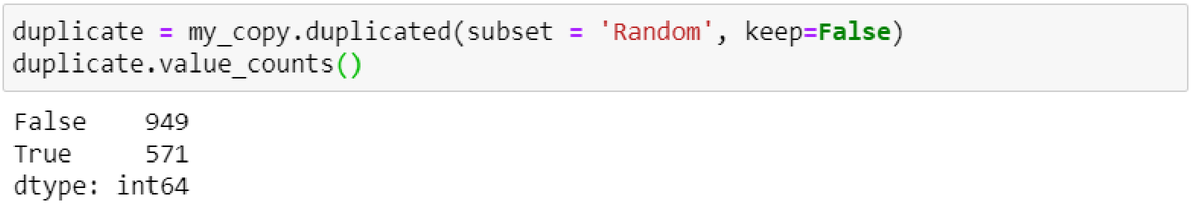
There is also another discrepancy with the initial data description. In the non-clinical description, the inclusion of a ‘Session’ attribute is declared. The description of this is the following:

*‘Anonymous patient session identifier. Should be unique value. Patient can have multiple sessions.’*

However, this ‘Session’ attribute is not included in the data description table, and there is no evidence of it existing anywhere in the actual legacy data file.

The Data Understanding section of the CRISP-DM methodology is clarified/explained in the data description (table). This gives us a better comprehension of what each attribute is and the potential impact it has on the patient label.

Regarding preprocessing the data, the first attribute is the Random column, essentially being a random number for each record. Although this column had no null values, the limitations to its implementation here included it containing many duplicates throughout the data. Code could have been run to loop through all the rows and assign a new non-duplicate random numbers between 0 and 1. However, even without identified duplicates, it would be performing the same role as the ‘Id’ column, and thus the decision was made to remove this attribute from the data set completely. Furthermore, it would have provided no benefit when training the model on the test data. The number of duplicate Random values was found by performing the following code:



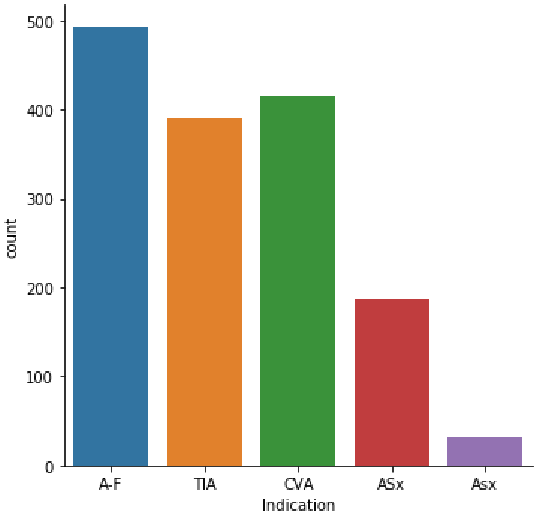
```
duplicate = my_copy.duplicated(subset = 'Random', keep=False)
duplicate.value_counts()
```

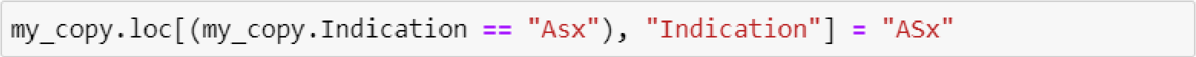
```
False    949
True     571
dtype: int64
```

Moving on to Id, this attribute had no duplicates and no null values. Although similar to the Random column, this attribute would be left in for now in order to help identify rows and impute further data where needed.

Next is the Indication column. Upon inspection via a bar chart, this shows that the data has ‘ASx’ values actually stated in the data as ‘Asx’ (case sensitive), a human error created when inputting the data.

For any rows where the indication was set as ‘Asx’, the value was then changed to ‘ASx’ so that the Indication attribute only had one of these value types. The following single line of code is used for changing this.

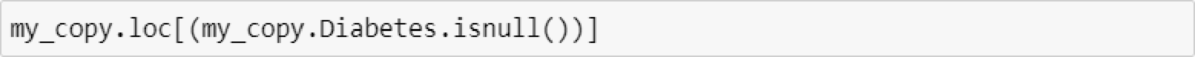




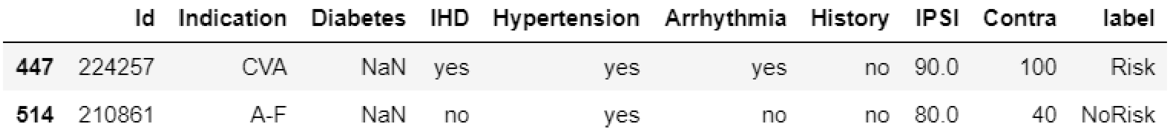
```
my_copy.loc[(my_copy.Indication == "Asx"), "Indication"] = "ASx"
```

Then for the issue of the 3 null indication values found, these rows were listed, and for each of these, the data was searched through for all the rows with matching values for all attributes apart from Id, IPSI and Contra. The mode (most frequent Indication value) from these matching rows was then imputed into the respective row, replacing the null value.

The Diabetes column contained two null values which were identified using the following line of code. These were imputed by locating these rows and displaying them, and then in a similar way to the imputing of indication values, finding the mode value out of yes/no for matching rows and setting that rows Diabetes value to this.



```
my_copy.loc[(my_copy.Diabetes.isnull())]
```



| | Id | Indication | Diabetes | IHD | Hypertension | Arrhythmia | History | IPSI | Contra | label |
|---|---|---|---|---|---|---|---|---|---|---|
| **447** | 224257 | CVA | NaN | yes | yes | yes | no | 90.0 | 100 | Risk |
| **514** | 210861 | A-F | NaN | no | yes | no | no | 80.0 | 40 | NoRisk |

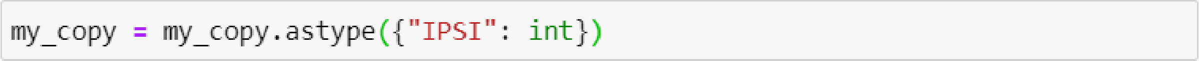
IHD did not need any cleaning/preprocessing as the attribute contained only yes/no values and did not have any null values. Thus, no preprocessing was needed for this column.

Hypertension contained only yes/no values apart from 3 rows which had null values for this attribute. These were imputed in similar ways to indication and diabetes null values. Taking the mode from rows that had all matching values all columns apart from Id, IPSI and Contra.

Arrhythmia, alike IHD, only consisted of yes/no values and did not contain any nulls, and hence did not need any values imputed/cleaned before modelling.

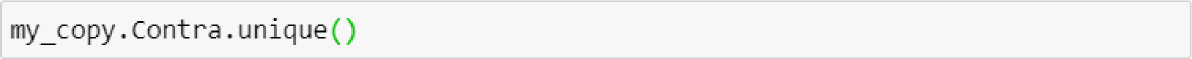
History contained 2 null values that needed cleaning and once more, these values were imputed via similar methods to the Diabetes and Hypertension. The History mode value out of yes/no for the rows with matching values was assigned to the null value for that row.

IPSI contained 4 null values. This repeated the process for each row finding all rows in the main data that matched all attributes. For the null IPSI rows with a Contra value of 100, the Contra value was included in the search criteria for matching rows, however for the rows with Contra values of 50 and 20 respectively, it was not included as these did not have any matching rows for the mean IPSI to be taken from. The IPSI value type was also initially float, so the following line changed its value type to int.



```
my_copy = my_copy.astype({"IPSI": int})
```

For Contra, there were no null values, however its initial value type was Object (String) when it needed to be int. There was one empty string value however (“ ”) which was found by performing the following line of code, which listed all the unique Contra values in the data.



```
my_copy.Contra.unique()
```

When searching through the data for imputing this row with the empty Contra value, there were 5 other rows found which had matching attributes in everything except for id. These rows also all shared a common Contra value of 60, thus it made sense to impute the empty Contra value to 60. As this attribute was also initially an object type, this was then converted to an int using a similar line to the one used for changing IPSI to type int.

For the Label attribute, there were 2 rows which contained ‘Unknown’ as its value, and 3 further rows which has a null value. These were found by performing the .unique() line as seen above for the Contra values. These rows were located via .loc and each of these rows were removed from the main data via the index. The decision to drop these rows was due to the sensitive nature (deciding on the risk of an individual) of the data and importance or training an accurate model instead of manually guessing Risk or NoRisk. These rows could furthermore optionally use an accurately trained model for filling in these values.

At this point the data had been fully preprocessed and the Id column was removed prior to modelling/training and testing.

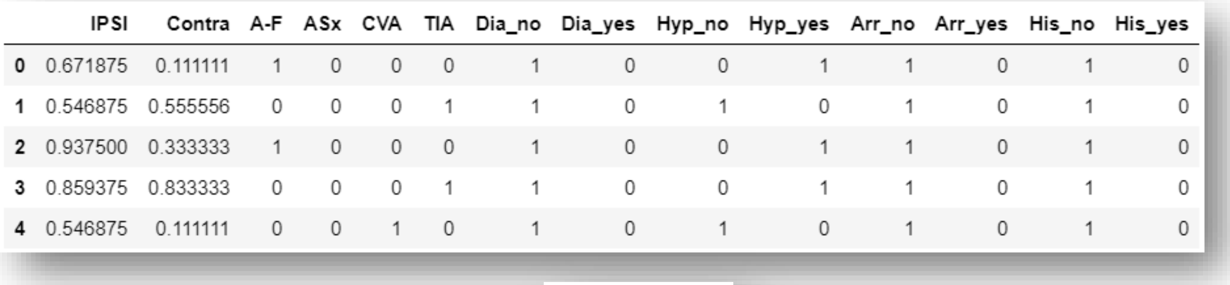
The three models chosen for this included a Multi-Layer Perceptron Classifier, a Decision Tree, and a Logistic Regression model. These were chosen as this was not a linear problem, with MLP and DT being classification models (categorisation of data into predefined classes based on input data – such as Risk/NoRisk in this case). The Logistic model is typically a regression type performed with continuous input; however, it can still use categoric, hence this case.

The IPSI and Contra values were also the subject of Min-Max and Z-Score normalisation. This was because large values next to small values (dummy variables of 0 or 1) may impact the model and skew the importance of them. All 3 models were tested without and with both types of normalisation on the IPSI and Contra values.

Dummy variables were taken for Indication, Diabetes, IHD, Hypertension, Arrhythmia and History and these were concatenated with the respective IPSI and Contra values for each row. This used the pandas .get_dummies and .concat functions respectively.

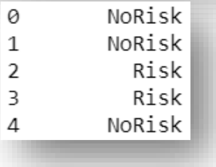
# Results

The X input features were modified with inclusion of dummy variables, so the input looked like this (preview) (this example shows the IPSI and Contra values min-max normalised):



| | IPSI | Contra | A-F | ASx | CVA | TIA | Dia_no | Dia_yes | Hyp_no | Hyp_yes | Arr_no | Arr_yes | His_no | His_yes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.671875 | 0.111111 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| **1** | 0.546875 | 0.555556 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| **2** | 0.937500 | 0.333333 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| **3** | 0.859375 | 0.833333 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| **4** | 0.546875 | 0.111111 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

The Y data is as following (preview):



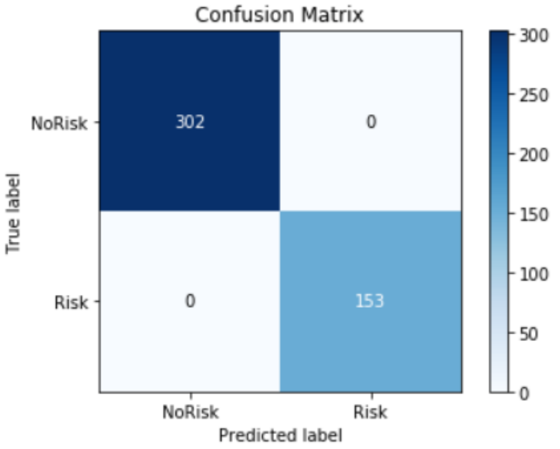
```
0      NoRisk
1      NoRisk
2        Risk
3        Risk
4      NoRisk
```

The following is a table of the performance metrics of all three models trained, with performance recorded initially, then with Z-Score and Min-Max normalisation of the IPSI and Contra values.

| | **MLP** | | | | **Decision Trees** | | | | **Logistic Regression** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Initial | After Z-Score | After Min-Max | | Initial | After Z-Score | After Min-Max | | Initial | After Z-Score | After Min-Max |
| **TP** | 135 | 133 | 153 | | 142 | 136 | 151 | | 123 | 116 | 143 |
| **TN** | 315 | 287 | 302 | | 299 | 286 | 297 | | 309 | 294 | 295 |
| **FP** | 3 | 7 | 0 | | 4 | 8 | 5 | | 9 | 10 | 7 |
| **FN** | 2 | 28 | 0 | | 10 | 25 | 2 | | 14 | 35 | 10 |
| **Sensitivity** | 0.99 | 0.82 | 1.00 | | 0.93 | 0.84 | 0.99 | | 0.90 | 0.77 | 0.93 |
| **Specificity** | 0.99 | 0.98 | 1.00 | | 0.99 | 0.97 | 0.98 | | 0.97 | 0.97 | 0.98 |
| **Precision** | 0.98 | 0.95 | 1.00 | | 0.97 | 0.94 | 0.97 | | 0.93 | 0.92 | 0.95 |
| **Accuracy Score** | 0.98 | 0.92 | 1.00 | | 0.96 | 0.92 | 0.98 | | 0.94 | 0.91 | 0.96 |

**TP** = True Positive (predicted Risk, outcome Risk).
**TN** = True Negative (predicted NoRisk, outcome NoRisk).
**FP** = False Positive (predicted Risk, outcome NoRisk).
**FN** = False Negative (predicted NoRisk, outcome Risk).
**Sensitivity** = True Positive Rate (how many patients who are at risk are correctly classified as Risk) TP / (TP + FN), also called Recall.
**Specificity** = True Negative Rate (how many patients who are not at risk are correctly classified as NoRisk) TN / (TN + FP).
**Precision** = Out of all the patients classified as Risk, how many are actually at Risk TP / (TP + FP).
**Accuracy Score** = As a percentage, the number of correctly classified items out of the whole set.

Below: a confusion matrix produced by a run through of the MLP Classifier with Min-Max normalisation of the IPSI and Contra values.



# Evaluation & Discussion

Regarding how the above results were collected, the three models were all trained and run with the same input test-train split and the relevant metrics were recorded. The models all used the same data test training split so that the results were not bias in favor of any of the models. This also meant that the results provided a just representation of the model performance, especially in comparison to one another. The metrics recorded included the creation of a confusion matrix showing true positive, true negative, false positive and false negative classifications. From this, the sensitivity, specificity and precision were all calculated for each model. A line of code was performed after the implementation of each model's prediction on the test data which provided an accuracy score for the model; effectively a percentage of the classifications in the test data that the model identified correctly. All these metrics where then put into the single table as seen in the results section for easy analysis and comparison between models.

The IPSI and Contra attributes were then both Z-Score normalised and then each of the models were trained and tested once more. All the same recorded metrics including the confusion matrix, sensitivity, specificity, precision and accuracy score were recorded and assigned to the results table section of the Z-Score normalisation for each respective model. Finally, the IPSI and Contra values were Min-Max normalised and once more each of the models were trained and tested. The Min-Max normalisation of these two attributes proved to be far more effective than the Z-Score normalisation of these and the results show evidence of this. All models had greater metrics for sensitivity, specificity, precision and accuracy when run with the Min-Max normalisation than they did with the Z-Score normalisation. One noticeable improvement would be the Logistic Regression sensitivity improvement of 0.77 for Z-score to 0.93 for Min-Max. Although this sensitivity percentage is still not ideal given its significance for this domain which would be correctly classifying a patient who is at risk.

For each model and both normalisation variations of the IPSI and Contra values, the models were trained and tested a few times to ensure that the metrics recorded in the results table were accurate and an appropriate representation of the model's performance. This would further remove bias and anomaly findings, as this could by random chance suggest than an inaccurate model could be better than a model which is usually highly accurate given one model performing worse than usual and the other performing far better than usual.

These metrics chosen were used as they provided a good individual and comparative analysis of model performance. The confusion matrices provided a useful visualisation, displaying color coded performance of the model label classification. This allowed for a better understanding of the model performance and furthermore provided the TP, TN, FP and FN figures which were used to calculate further metrics for the model. The Mean Squared Error (MSE) and Mean Average Error (MAE) metrics were not included when analyzing these models due to these metrics being based on linear regression.

To a certain extent, the sensitivity metric (how many patients who had the label Risk were correctly classified as Risk from the test data set) is very important in this domain along with accuracy score for determining model performance. This is the case here due to the potentially fatal risk of a misdiagnosis; in the outcome of a NoRisk diagnosis for someone who was actually at Risk.

Regarding which is the best model for the prediction of a patients Risk or not, from this data set, all of them provided accuracy scores of above 90%. However, it was the Multi-Layer Perceptron model with the Min-Max normalisation of the IPSI and Contra values that comparatively provided the all-round most accurate metrics. This version on one attempt produced a model that was 100% accurate, correctly classifying all the test data rows into either Risk or NoRisk. This model generally produced close to flawless results, with this instance recording a score of 1.00 for all the metrics.

Regarding what could have been done better / what could have been done differently for this project, there are a few additional techniques/methods that could've been explored/implemented. For the preprocessing of this data prior to modelling, the methods I used for imputing the null values could be considered inefficient on large data sets. Searching the data for all rows that had matching values for most attributes is time consuming and memory taxing, and individually going through every single row to check for matches is far from the most effective method of finding a suitable value to fill the null with.

There is also the consideration which was not implemented that all 20 rows with null, missing or unknown values could have been dropped from the data prior to modelling. The reasoning behind this is because the data set provided had a significant number of entries (1520 rows) and thus removing 20 rows from this would mean a data set of 1500 entries, which would have still been a sufficient amount for training and testing the data appropriately. The removal could have potentially benefitted the model as it would have meant no manually entered data, which was essentially human guesses based on mode and mean values. A model based on entirely accurate rows may have potentially had a higher accuracy score and better metrics when predicting the test data.

One feature which the implementation of could have improved the models would have been the inclusion of K Fold Cross Validation. This would have consisted of splitting up the data into K amount of folds (sections), and then training K number of models, with each model using a different section for testing and then the rest of data for its training. This could have provided a better representation of model performance; however, it would have to be ensured of the use an appropriate K number and section size for each fold to be statistically representative.

Furthermore, the implementation of additional models could have been beneficial. Potential models that could have been researched and implemented for training and testing on this data set include Random Forest (one application of this being the algorithm Netflix uses for recommendations) and Naive Bayes. (Fuchs, 2017)

The deployment phase of the CRISP-DM methodology cannot exist here as this project was undertaken in university setting. However, we can make a reasonable judgement about the hypothetical scenario of a model being deployed into the real world and the implications of deploying this model in a medical domain. Based on the metrics produced by the models created, the MLP classifier model with the Min-Max normalisation of the IPSI and Contra values is by far the most accurate and reliable model. On one attempt producing perfect metrics and a 100% accuracy score. If this model was deployed then it would likely be a very useful tool and asset in identifying patients who were at risk, when supplied with all the patient input features. This model realistically should not be used as a be all and end all when determining a patient label, and realistically it may not always be correct. The potential risks of the model not working well could be fatal, with a suitable sensitivity metric of the model being vital. A misdiagnosis of a patient of is actually at Risk as a noRisk could be potentially life threatening and thus the model could be considered a failure given even the smallest number of patient misdiagnosis. The classification of a patient who isn't at risk as a risk would not be an ideal scenario, and the model should ideally avoid this. However, it would be vital for a model to correctly identify all the patients who are at risk.

In conclusion, for this model to be implemented, it could be more in favor of classifying an individual as a risk, and then via human decision, the patients who were potentially still a noRisk could be manually identified. But overall, it would be vital to maintain a close to perfect sensitivity score for the model to be successfully implemented, as the risks attached to this could be catastrophic.

# References

Fuchs, K. (2017). *Machine Learning: Classification Models*. [online] Medium. Available at: https://medium.com/fuzz/machine-learning-classification-models-3040f71e2529 [Accessed 18 Dec. 2019].