**MSc Project Report**

16th September 2019

John H Booth (13133420)

BBK\_BUCI058D7\_1819 - MSc Data Science Project

MSc in Data Science

Department of Computer Science and Information Systems, Birkbeck College, University of London, 2019

Identifying Feature Importance in Pediatric Post-mortem Outcome with Machine Learning Models

Academic Declaration

I have read and understood the sections of plagiarism in the College Policy on assessment offences and confirm that the work is my own, with the work of others clearly acknowledged. I give my permission to submit my report to the plagiarism testing database that the College is using and test it using plagiarism detection software, search engines or meta-searching software.

The report may be freely copied and distributed provided the source is explicitly acknowledged.

Acknowledgements

Contents

[Table of Figures 6](#_Toc18849563)

[1 Abstract 7](#_Toc18849564)

[2 Introduction 8](#_Toc18849565)

[2.1 Background 8](#_Toc18849566)

[2.2 Literary Review 8](#_Toc18849567)

[2.2.1 Paediatric Post-mortems 8](#_Toc18849568)

[2.2.2 Data Extraction 8](#_Toc18849569)

[2.2.3 Data Wrangling 9](#_Toc18849570)

[2.2.4 Analytics 9](#_Toc18849571)

[2.3 Report outline 10](#_Toc18849572)

[2.4 Aim and Objectives 10](#_Toc18849573)

[3 Methods 11](#_Toc18849574)

[3.1 Project Pipeline 11](#_Toc18849575)

[3.2 Data Engineering 11](#_Toc18849576)

[3.2.1 Extract, Transform, Load (ETL) Process 11](#_Toc18849577)

[3.2.2 Creation of Summary and Reporting Attributes 12](#_Toc18849578)

[3.2.3 Identifying Data to be included in this study 12](#_Toc18849579)

[3.2.4 Data Wrangling 13](#_Toc18849580)

[3.3 Analytics 13](#_Toc18849581)

[3.3.1 Visualisation: ggplot2 14](#_Toc18849582)

[3.3.2 Decision Tree: rpart 14](#_Toc18849583)

[3.3.3 Random Forests: randomForest 14](#_Toc18849584)

[3.3.4 Gradient Boosted Decision Tree: xgboost 14](#_Toc18849585)

[3.3.5 Combined Results and Verification of Models 15](#_Toc18849586)

[4 Data Engineering 16](#_Toc18849587)

[4.1 ETL Process 16](#_Toc18849588)

[4.2 Creation of Summary and Reporting Attributes 18](#_Toc18849589)

[4.3 Identifying Data to be included in this study 18](#_Toc18849590)

[4.4 Data Wrangling 19](#_Toc18849591)

[5 Analysis 22](#_Toc18849592)

[5.1 Visualisation 22](#_Toc18849593)

[5.1.1 Complete Data Set 22](#_Toc18849594)

[5.1.2 Study Data Set 23](#_Toc18849595)

[5.1.3 Missing data 24](#_Toc18849596)

[5.1.4 Imbalanced data 25](#_Toc18849597)

[5.2 Decision Tree 26](#_Toc18849598)

[5.2.1 External 27](#_Toc18849599)

[5.2.2 Internal – Stage 1 – Organs 28](#_Toc18849600)

[5.2.3 Internal – Stage 2 – Macro investigation 29](#_Toc18849601)

[5.2.4 Internal – Stage 3 – Histological investigation 30](#_Toc18849602)

[5.2.5 All Stages 31](#_Toc18849603)

[5.3 Ensemble Models 32](#_Toc18849604)

[5.3.1 Random Forest 32](#_Toc18849605)

[5.3.2 Gradient Boosted Decision Tree 33](#_Toc18849606)

[5.4 Compare models with varying random seeds 34](#_Toc18849607)

[6 Results 36](#_Toc18849608)

[6.1 Predictive Accuracy 36](#_Toc18849609)

[6.2 Relative Feature Importance by Stage of Post-mortem 37](#_Toc18849610)

[7 Conclusion 39](#_Toc18849611)

[7.1 Project Summary 39](#_Toc18849612)

[7.1.1 Data Engineering 39](#_Toc18849613)

[7.1.2 Decision Tree 39](#_Toc18849614)

[7.1.3 Ensemble Models 39](#_Toc18849615)

[7.2 Project Evaluation 39](#_Toc18849616)

[7.3 Recommendations for Future Work 39](#_Toc18849617)

[References 40](#_Toc18849618)

[Bibliography 41](#_Toc18849619)

[Glossary of Terms 43](#_Toc18849620)

[Appendix A – Example Project Code 44](#_Toc18849621)

[Python Function with ODBC 44](#_Toc18849622)

[Python Function to Create COD2\_SUMM 45](#_Toc18849623)

[Python Function to Create RDV 49](#_Toc18849624)

[R Model Function 54](#_Toc18849625)

[Appendix B – ETL Process 62](#_Toc18849626)

[Appendix C – Cause of Death Attribute Mapping 63](#_Toc18849627)

[Appendix D –RDV structures 63](#_Toc18849628)

[External 63](#_Toc18849629)

[Internal – Stage 1 – Organs 63](#_Toc18849630)

[Internal – Stage 2 – Macro investigation 63](#_Toc18849631)

[Internal – Stage 3 – Histological investigation 63](#_Toc18849632)

[Appendix E – Detailed output from an analytic process 64](#_Toc18849633)

[Appendix F – Deliverables 64](#_Toc18849634)

# Table of Figures

[Figure 1 - Project Pipeline 10](#_Toc18849635)

[Figure 2 - PM Research Database - Partial Entity Relationship Diagram 16](#_Toc18849636)

[Figure 3 - Example of the use of the Concepts Table within the EAV model 17](#_Toc18849637)

[Figure 4 - HAS Schema Entity Relationship Diagram 17](#_Toc18849638)

[Figure 5 - Snapshot of the original RDV 20](#_Toc18849639)

[Figure 6 - Snapshot adjusted using normalisation and one-hot encoding 20](#_Toc18849640)

[Figure 7 - Linear Regression Plots for Numeric Features 21](#_Toc18849641)

[Figure 8 - Visualisations of Complete Data Set 22](#_Toc18849642)

[Figure 9 - Visualisation of Study Data Set 23](#_Toc18849643)

[Figure 10 - Number of missing measurement values 24](#_Toc18849644)

[Figure 11 – Variability of Categorical Variables 25](#_Toc18849645)

[Figure 12 – Decision Tree – External Examination 27](#_Toc18849646)

[Figure 13 – Decision Tree – Internal Examination – Organs 28](#_Toc18849647)

[Figure 14 - Decision Tree - Internal - Macro Investigation 29](#_Toc18849648)

[Figure 15 - Decision Tree - Internal Histological Investigation 30](#_Toc18849649)

[Figure 16 – Decision Tree – Relative Feature Importance – All Stages 31](#_Toc18849650)

[Figure 17 – Random Forest Confusion Matrices – All stages 32](#_Toc18849651)

[Figure 18 – Random Forest Relative Feature Importance – All Stages 32](#_Toc18849652)

[Figure 19 – XGBoost Confusion Matrices – All stages 33](#_Toc18849653)

[Figure 20 – XGBoost Relative Feature Importance – All Stages 33](#_Toc18849654)

[Figure 21 – Predictive Accuracy of each stage by Model by Run 34](#_Toc18849655)

[Figure 22 – Variability in Accuracy by Model by Stage 34](#_Toc18849656)

[Figure 23 – Compare Feature Importance by Model, Stage: Ext, by Run 35](#_Toc18849657)

[Figure 24 – Compare Feature Importance by Model, Stage: Int3, by Run 35](#_Toc18849658)

[Figure 25 – Relative Feature Importance, Stage: Ext 37](#_Toc18849659)

[Figure 26 – Relative Feature Importance, Stage: Int1 37](#_Toc18849660)

[Figure 27 – Relative Feature Importance, Stage: Int2 38](#_Toc18849661)

[Figure 28 – Relative Feature Importance, Stage: Int3 38](#_Toc18849662)

# Abstract

Post-mortems are complex procedures that utilise a significant amount of hospital resources, yet despite this, cause of death is only determined in 45% of cases. The event itself can be very traumatic for the parents of the child, yet is essential for providing further clinical understanding of the patient’s cause of death. Given this, there is an imperative to extract the greatest possible value from the data. Here, we investigated whether machine learning could be used to derive novel insights from the prediction of post-mortem outcomes.

A post-mortem database containing 7000 records across 300 variables was analysed and categorised into stage of examination (external and internal). The outcome of the examination was summarised as either ‘cause of death determined’ or ‘not determined’. From these summarised data, cases were filtered by children aged <= 2 years, resulting in a dataset of 3,100 post-mortems.

Following this, decision tree, random forest, and gradient boosting machine models were iteratively built for each stage of the post-mortem examination and compared using their predictive accuracy metrics.

The naïve decision tree model using external examination data had a predictive performance of 68%. Model performance notably increased when trained on internal examination data. At each stage of the examination, a core set of data items, of which the final set included age, BMI, and heart weight were highlighted using model feature importance as key variables for determining post-mortem outcome. The use of increasingly complex modelling techniques was able to boost the predictive performance of the model by as much as 10%.

This project clearly shows the value of collecting clinical procedural data which can then be modelled using machine learning techniques to inform clinical practice. With more time, further modelling, including unsupervised clustering could be undertaken to derive further insights.

Supervisor:

Nigel Martin (nigel@dcs.bbk.ac.uk)

Department of Computer Science and Information Systems, Birkbeck College, University of London, London, UK.

# Introduction

## Background

Great Ormond Street Hospital for Children NHS Trust (GOSH) is the country’s leading centre for treating sick children. With the UCL Great Ormond Street Institute of Child Health, GOSH is the largest centre for paediatric research outside the US.

Specialist Paediatric Pathologists perform perinatal, infant and childhood post-mortems including hospital referrals, forensic cases and those on behalf of Her Majesty’s Coroner.

The Pathology Department has established a research database containing details of all post-mortems performed between 1996 and 2017.  The database was originally used specifically for research into Sudden Unexpected Death in Infancy (SUDI). Since then it has been utilised for a number of other projects investigating SUDI, stillbirths and various aspects of paediatric autopsy procedure.

Currently the database holds 7000 records, each record representing an individual post-mortem. Up to 300 items of data can be defined for each post-mortem.

The purpose of this project is to use data science analytic techniques to develop operational strategies that can be applied to paediatric post-mortems to prioritise which data is required to achieve the target of specifying the cause of death.

## Literary Review

A summary of the full literary review presented in the project proposal.

### Paediatric Post-mortems

Paediatric post-mortems have their own specific issues as explained on the Royal College of Pathologists [*RCPath Contributors (2018)*]:

“Paediatric and perinatal pathology is concerned with identification of disease in the fetus, infant and child. It is age-specific rather than organ-specific and includes investigation of that organ unique to the fetus, the placenta. The spectrum of disease in this age range is very different from that seen in adults and the interaction of congenital malformation and growth of the child interact to produce unique pathology.”

The Lullaby Trust, a charity that supports parents who have suffered the sudden loss of a child support research in this field gives a detailed breakdown of the different categories or presentations of post-mortems [*Lullaby Trust (2018)*]:

* TOP: Termination of pregnancy so the patient has not reach full term less than 24 weeks
* Still birth: 24 weeks to full term
* SUDI: Patients less than one year old
* SUDC: Patients over 1 year.

### Data Extraction

The Entity Attribute Value (EAV) model would be a good schema to use to extract the data for analytics. The advantages of using the EAV model for healthcare data are outlined by [*Löper, D., et al, 2013*] and the efficiency of storing data as described in [*Dinu, et al. (2007)] .*A clear example of the flexibility of the EAV model for health care data is given in [*Borodin, et al. (2015)]*

### Data Wrangling

Although the majority of data held on a post-mortem is categorical a significant number of data is numeric data, lengths and weights. The importance of these values in determining cause of death is detailed by [*Horn,et.al., 2004*. ].

Even within the main presentations of post-mortems described above these values can vary considerably. The approach of using growth charts in post-mortem analysis is described in [*Pryce, J.W., et.al. 2014]*

### Analytics

The base analytic technique for this project will be Decision Trees with cross validation. Decision tree methodology is a commonly used data mining method for establishing classification systems based on multiple covariates or for developing prediction algorithms for a target variable [*Song, et al.2015*]. The key advantage of the Decision Tree technique is it simplifies complex relationships between input variables and target variables by dividing original input variables into significant subgroups, thus making the model easier to understand and interpret. [*Song, et al.2015*]

The main disadvantage of the technique is that using a single tree a model will suffer from low variance and high bias [*Analytics Vidhya Contributors (2016)*]. To combat this situation the project will consider ensemble methods which look to combine different techniques to better balance variance versus bias *[Abolfazl R, 2018*.].

The first technique to be considered will be Random Forest where the training data is split into a number of different sets and a tree is calculated for each set and the results combined *[Abolfazl R, 2018.*].

Gradient boosting is another technique that looks to decrease bias. Gradient boosting is a technique that looks to combine parameters that give a low prediction accuracy to produce a higher prediction accuracy [*Prashant G 2017.*]

## Report outline

The back bone of the project is a pipeline from the originating Post-mortem research database to the project results utilising different development strategies at various stages; this pipeline is laid out in detail the methods chapter of the report.

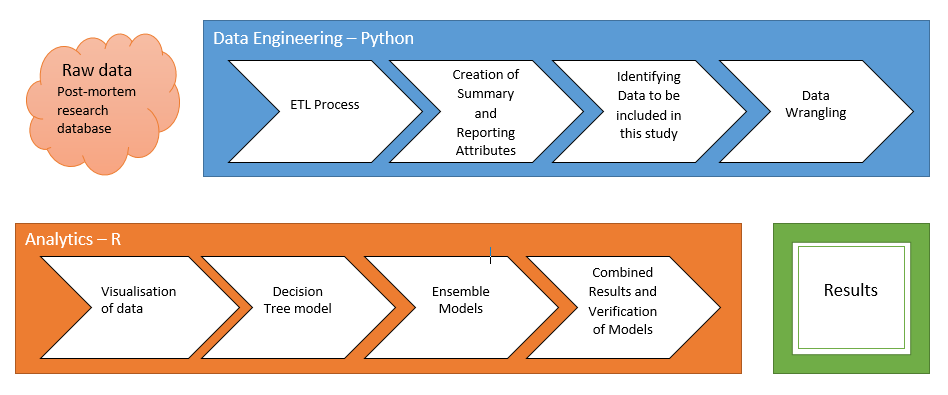


Figure 1 - Project Pipeline

Data Engineering is a major part of this project and the specific challenges faced in undertaking the project has its own chapter. The analytic processes of visualising the data followed by creation of three distinct models tuned to the individual stages of the post-mortem is described primarily using graphics produced during the analytics.

A basic glossary of terms covering both data engineering and healthcare aspects of the project has been included to help the broad range of readers of this report. More detailed descriptions of code layout, ETL process and RDV structures are included in the appendices.

## Aim and Objectives

Take from project proposal; amended based on experience undertaking project?

The aims of this project are:

1. To develop a routine to extract data from the existing Post-mortem Research Database into an entity attribute value schema that will make the data more readily available for data analytics.
2. To apply the Decision Tree Analytical method to the extracted data to develop operational strategies that can be applied to paediatric post-mortems to prioritise which data is required to achieve the target of specifying the cause of death.
3. To investigate ensemble strategies, specifically Random Forests and Gradient Boosting to see how these techniques can improve on the basic Decision Tree method.

# Methods

## Project Pipeline

This overall project has been divided into a number of sections the output of each section provides the input for the following section to form the project pipeline. The process for each section will be coded in an appropriate environment described below but with the overall aim of creating a fully reproducible set of procedures that lead to a set of results.

A GIT hub repository has been created for the project and all project code, documents and images are stored and versioned in this repository. A link to the GIT repository is given in Appendix D - Deliverables.

This chapter have been divided into two major sections; Data Engineering and Analytics.

## Data Engineering

The data engineering aspects of this project will be undertaken using the Python programing language, a general purpose programing language that used extensively in the world of data science. The development of python procedures will be carried out using PyCharm an integrated development environment (IDE).

The data manipulation carried out during this stage will be using structured query language (SQL) and will be instigated using the Python package PyODBC which allows the connection to external databases using ODBC connections and the production and return of SQL queries. An example of a PyODBC procedure is given in Appendix A – Example Project Code.

Where appropriate the data engineering code will be broken down into functions that will be unit tested prior to implementation. All processes will also be developed with integral profiling so that any bottlenecks can be identified and if possible their effect reduced so that the overall processing of the data can be as efficient as possible.

### Extract, Transform, Load (ETL) Process

The fundamental section of this stage is the ETL process on the Post-mortem Research database into the Health Analytics Schema (HAS) model using the Entity Attribute Value (EAV) schema.

The HAS is a specific implementation of the EAV schema used at GOSH to store diverse healthcare related data for research.

The basic structure will be:

* Create HAS Tables
* Create Concepts
* Create Patients and Staff
* Create Events
* Create Event Attributes

A detailed breakdown of the python code developed for the ETL process is given in Appendix B – ETL Process.

The output of this section will be the HAS database created and populated from the originating system.

### Creation of Summary and Reporting Attributes

Having created the base research data from the original data then a number of summary event attributes will be created for reporting and analytic purposes:

* Number of Attributes(ATTRIBUTES)
  + The number of event attributes each event has.
* Cause of Death Summary (COD2\_SUMM)
  + A summary of the COD2 attributes into:
    - Not Determined
    - Determined
    - Unknown
    - Not available
  + More details in the following Data section of the report.
* Macro and Histological Body System Attributes
  + Individual organ internal macro and histological examination results are summarised at the body system level for ease of analytics.
* External and Internal Examinations
  + A simple flag to indicate whether an individual post-mortem event had had an external and or internal examination.

The output of this stage will be the addition of a number of event attributes added to the existing set of events and their originating attributes.

### Identifying Data to be included in this study

This process will be split into 2 stages:

* Include or exclude data
  + COD2\_summ
    - Only include events where the COD2\_summ is either Not Determined or determined.
  + Age category
    - Only include events for the following age categories:
      * Early Neonatal
      * Neonatal
      * Infant
      * Child - under the age of 2 years
  + Measurement Outliers
    - Any numerical values that fall outside what is physically possible.
* Identify for the 4 stages of the post-mortem being considered in this study which features should be included:
  + External
  + Internal Stage 1 (Organ weights)
  + Internal Stage 2 (Macro examination)
  + Internal Stage 3 (Histological examination)

At this stage the issue of missing data for any chosen event will not be addressed.

The output of this stage will be four research data views (RDVs), one for each stage of the post-mortem, in the form of CSV files.

### Data Wrangling

The final section of the data engineering stage will be to produce the data in the format most appropriate for analytics. Two forms of data wrangling will be used:

* One-hot encoding – Categorical features
  + Rather than each categorical feature having a single column of data with the appropriate category; each category has its own column with either a 1 or 0 depending on whether each event has that feature value.
* Numerical normalisation – Numeric features
  + Each numeric value will be normalised based on their predicted value for the age of the patient described by each event. This routine means that each numeric value will be in the range 0 – 1 with only outliers having larger values.

It should be notes that Z-Score standardisation of the numeric data was considered but not pursued as it didn’t take into account the age of the patient in each event.

The output of this section will be four adjusted RDVs, one for each stage of the post-mortem, in the form of CSV files.

The detailed structure of the RDVs in both formats is shown in Appendix C – RDV Structure.

## Analytics

The analytic aspects of this project will be undertaken using the R programming language a language specifically developed for statistical computing. The development of R scripts will be carried out in R Studio an IDE for the R language.

In this section I will describe the key packages that I will be using and the specific parameters that have to be tuned to obtain an optimised model.

A basic tuning procedure will be adopted for all three modelling packages:

* Create model using default parameters for each post-mortem stage
* Define a range for each parameter to be tuned.
* Change each parameter one by one and obtain an optimal value based on predictive accuracy.
* Repeat last step to see whether any changes in the parameters significantly affects each parameter.
* Finalise a set of parameters for each post-mortem stage for each model.

The output for each of the modelling stages is an R function that can be called for that model with a training/test split for each post-mortem stage. The function will save the resulting confusion matrices and relative feature importance in CSV files as well as plots specific to each model as PNG files.

### Visualisation: ggplot2

Ggplot2 is the principal graphics package used within R and is part of the tidyverse, a collection of packages aiming to bring some semblance of order in the slightly anarchic world of R programing.

This section has three main aims:

* Visualisation of the complete post-mortem data set
* Visualisation of the sub set of data to be used for this study
* Develop a basic graphical framework that can be used for all images produced by the various further analytic sections.
  + Colour scheme – viridis a colour blind friendly colour palette
  + Theme.classic – a very basic no frills plotting theme.
  + PNG file naming convention for saving plots

The output of this section are two frames of visualisations saved as PNG files.

### Decision Tree: rpart

The rpart package uses recursive partitioning on trees, both classification and regression to achieve an optimum level of complexity for a given set of data.

The main hyper-parameters that can be tuned are:

* minsplit - the minimum number of observations that must exist in a node in order for a split to be attempted
* minbucket - the minimum number of observations in any terminal node. Use minsplit / 3
* cp – complexity parameter, used to define further pruning after the initial tree is produced

### Random Forests: randomForest

Classification and regression based on a forest of trees using random inputs, based on Breiman (2001)

The main hyper-parameters that can be tuned are:

* Mtry - Number of candidates draw to feed the algorithm. By default, it is the square of the number of columns.
* Maxnodes - Set the maximum amount of terminal nodes in the forest
* ntree - number of trees in the forest

### Gradient Boosted Decision Tree: xgboost

xtreme Gradient Boosting, which is an efficient implementation of the gradient boosting framework from Chen & Guestrin (2016)

The main hyper-parameters that can be tuned are:

* Eta – controls how much information from a new tree is used in boosting.
* max\_depth – controls the maximum depth of a tree.
* gamma - Controls the minimum reduction in the loss function required to grow a new node in a tree.
* min\_child\_weight - Controls the minimum number of observations (instances) in a terminal node.
* Subsample - This parameter determines if we are estimating a Boosting or a Stochastic Boosting.
* colsample\_bytree – Number of features to sample in each new tree.

### Combined Results and Verification of Models

Using the functions developed for each model package a model can be run for each model package for each post-mortem stage for a number of random seeds deriving the training/test data split. The CSV files from each model run can then be combined to produce:

* A comparison of model predictive accuracy for changing random seeds
* A comparison of the change in predictive accuracy of each model at each stage of the post-mortem.
* A comparison of relative feature importance changes for different random seeds for each stage of the post-mortem
* A final predictive accuracy of cause of death determined or not for each model at each stage of the post-mortem. The predictive of accuracy of both not determined and determined cause of death can also be identified by model by stage of post-mortem.
* A final set of relative feature importance by model by stage of post-mortem.

# Data Engineering

This section of the report will look in more detail on the data engineering undertaken to prepare the data for analytics.

## ETL Process

The originating database was developed over a number of years and was optimised for data recording. The tables are divided up into subject groups and the overall structure is defined by primary and foreign keys. A lot of the fields contain no data so it is difficult to comprehend how much meaningful data there really is. The following partial schema only shows the major tables and excludes the 134 look-up tables for clarity.



Figure 2 - PM Research Database - Partial Entity Relationship Diagram

The target database will use Healthcare Analytics Schema (HAS) uses the Event Attribute Value (EAV) model. This model has many fewer tables and the overall structure of the data is all encapsulated in the concepts table.



Figure 3 - Example of the use of the Concepts Table within the EAV model

The above example demonstrates how both the nutrition field is stored within the overall structure of the data as well as all the possible values the field could take.

The ETL process extracts patient details into patient and patient attribute tables. Every post-mortem is represented as a single event and every field of data is represented as an event attribute with the event attribute type linked back to the concepts table. In the case of look-up values then the value is linked back to concepts table also. Every event attribute has a value; there are no NULL values in the EAV model. In the following entity relationship diagram of the HAS schema all tables are displayed and the concepts table is included twice for clarity.



Figure 4 - HAS Schema Entity Relationship Diagram

Development process

* Create HAS\_Tables.py
* Created a separate analytic database to house the HAS schema with linked tables back to the PM Research database. This structure proved to be very inefficient and the bottleneck of the linked tables was identified.
* Changed procedure so that HAS tables still housed in separate database but the PM research database was connected to directly this provided a 10 fold improvement in processing time.
* The details of the EAV process are given in Appendix B.

## Creation of Summary and Reporting Attributes

Having gone through the ETL process to establish the HAS the benefits can start to be realised.

In this section we add additional attributes to the database for use by the analytics process without having to make any structural changes and their relationship to other existing attributes is clearly documented within the concepts table.

Development process

* Modify\_events.py
* Process is creating the COD2\_SUMM attribute value type concept
  + Create COD2\_SUMM Concepts
  + Extract all events with event attribute of type COD2\_COD2ID
  + For each event
    - Apply mapping to COD2\_SUMM. Details in Appendix C – Cause of Death Attribute Mapping.
    - Create new event attribute of type COD2\_SUMM
* Process for creating ATTRIBUTES
  + Create ATTRIBUTES concept value type numeric
  + Select all event attributes group by event count(attributes)
  + For each event
    - Create new event attribute of type ATTRIBUTES with a value of count(attributes)
* Similar processes were developed for:
  + system\_name + macro\_SyFiID and system\_name + histo\_SyHiID
    - Individual organ internal macro and histological examination results are summarised at the body system level.
    - For each system the results for the individual organs were noted and the maximum result, 1-3, for any organ was assigned to the system.
      * 001 - Normal
      * 002 - Abnormal but not COD
      * 003 - Abnormal COD
      * 999 - Other
  + ExternalExam and InternalExam
    - A simple flag to indicate whether an individual post-mortem event had had an external and or internal examination.
      * True/False

## Identifying Data to be included in this study

As with the previous section the EAV model makes it very simple to clearly define which data is to be used for a particular analytic study. Full details of columns in individuals RDV files are given in Appendix D – RDV Structures

Development process

* Modify\_events.py
  + Create INC\_IN\_STUDY concept of value type concept.
    - Create multiple exclusion types to be able to identify why an event was excluded.
  + Add attribute to every event and set to ‘Include’.
    - Check if attribute exists id it does update to ‘Include’. This feature allows the process to be run multiple times
  + Define exclusion attributes
  + For each event
    - Check exclusion attribute
      * Update INC\_IN\_STUDY to appropriate exclusion type if required.
* Create\_rdvs.py
  + Initially developed a generic routine that creates a CSV file based on:
    - List of patient attribute filters
    - List of patient attribute columns
    - List of event attribute filters
    - List of event attribute columns
  + For study RDVs
    - Include filter is event attribute INC\_IN\_STUDY.
    - Defined columns for each stage of the post-mortem stage.
      * Each new stage added additional columns to previous stage.
    - Produced four CSV files one for each stage.

## Data Wrangling

Although not strictly necessary for the basic decision tree model the advanced ensemble based models needed the data to be modified by:

* Normalisation of numeric variables for Age
  + Numeric values represent measurements of different aspects of the human body and vary considerably in magnitude.
  + In principal this difference in magnitude could be removed by standardisation of the variables using Z-Score but this method would lose the intrinsic difference between values of different ages of development.
* Apply one-hot encoding
  + Converts categorical variables to numeric and removes any bias introduced by having different values for each category as this method means that all values can be represented by 1 or 0.



Figure 5 - Snapshot of the original RDV



Figure 6 - Snapshot adjusted using normalisation and one-hot encoding

It was decided for comparison of the models that all models should be developed against the same data structure.

Development process:

* Modify\_csv\_data.py
* Develop a linear regression model for each measurement feature split by age and sex.
  + Used sklearn.linear\_model from python Scikit-Learn package
  + Used matplotlib.pyplot to plot results

|  |  |
| --- | --- |
| Linear Regression plots for ages <= 100 days | Linear Regression plots for ages > 100 days |
|  |  |
| Blue dots and yellow lines = Male, red dots and green lines = Female | |

Figure 7 - Linear Regression Plots for Numeric Features

* Read in original CSV file
* For each row:
  + For each measurement column
    - Add a revised measurement column based on age and sex of patient
  + For each categorical column
    - Add a set of columns for all the possible variants of that category used within the original file.
* Write out adjusted CSV file with the appropriately transformed values.
  + For categorical values insert a 1 where the column matches the original category and 0 in all other columns
  + For measurement values apply the appropriate linear regression parameters based on the sex and age of the patient and store:
    - absolute(actual – predicted)/predicted.
* The final output is an adjusted CSV file for each post-mortem stage.

# Analysis

The visualisation stage is used to make final checks on the data as presented in the RDV’s from the previous stage. Particular attention will be given to missing and imbalanced data.

Once the final modifications are determined the focus changes to the first model; the Decision Tree. Due to the transparency of this model the output is reviewed in more detail for each stage of the post-mortem looking at both tabular data and visualisations in the form of the tree produced, confusion matrix for predicted accuracy and relative feature importance.

For the ensemble models of Random Forest and Boosted Gradient Tree the output is limited to the confusion matrix and relative feature importance.

## Visualisation

### Complete Data Set

The first step in the analytic process is to get to know the data. Initial visualisations were done on the complete data set but focussed on the features that were used to define the data to be included in the study.

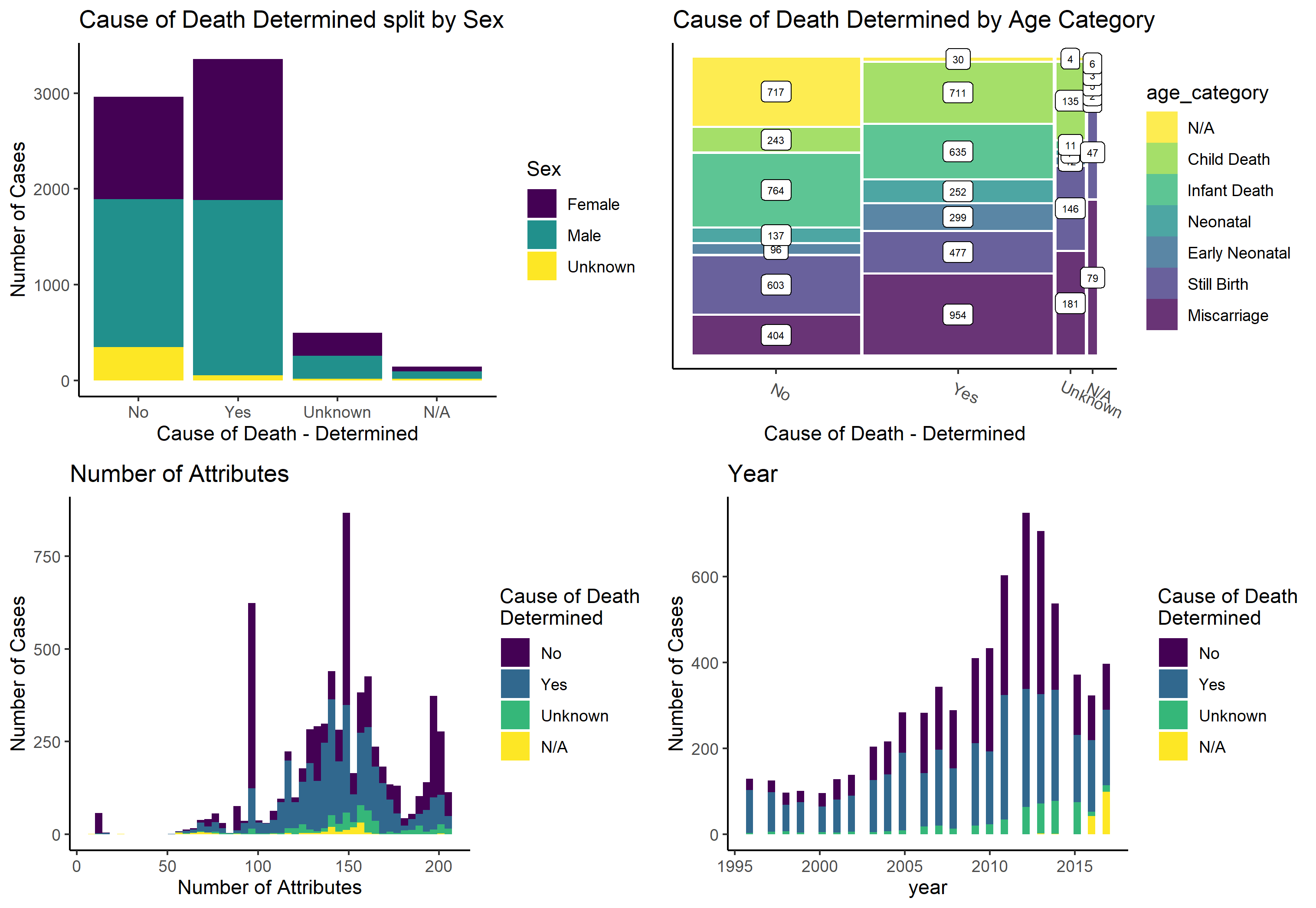


Figure 8 - Visualisations of Complete Data Set

The plots above show that the data is well distributed across cause of death determined and not determined, by age group and by sex. The events have generally a significant number of attributes without any one classification having significantly more than the other and that they are well distributed over time.

### Study Data Set

The next set of visualisations were done to focus on the split of data to either be included or excluded from the study and for the reasons why they were excluded.

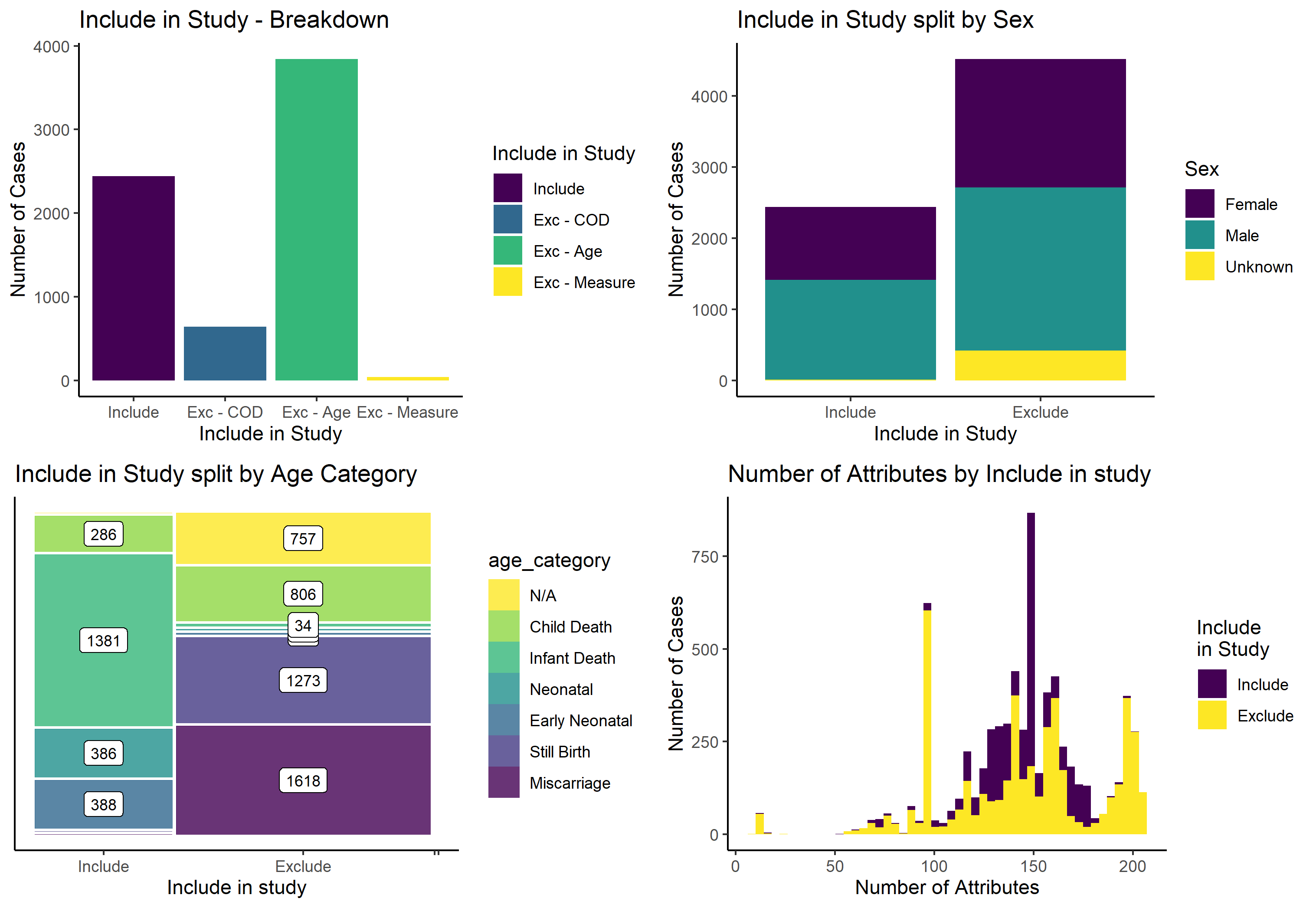


Figure 9 - Visualisation of Study Data Set

The first plot shows that the majority of events were excluded due to the decision to focus on the 1 day to 2 years age group. A significant number of events were excluded due to not having a clear statement about cause of death and then a few events due to incorrect measurements. It is shown that within the events to be included both sexes are well represented and the age groups also good representation. Finally the included events have good numbers of attributes.

### Missing data

Having got a study data set defined it should be checked for missing data:

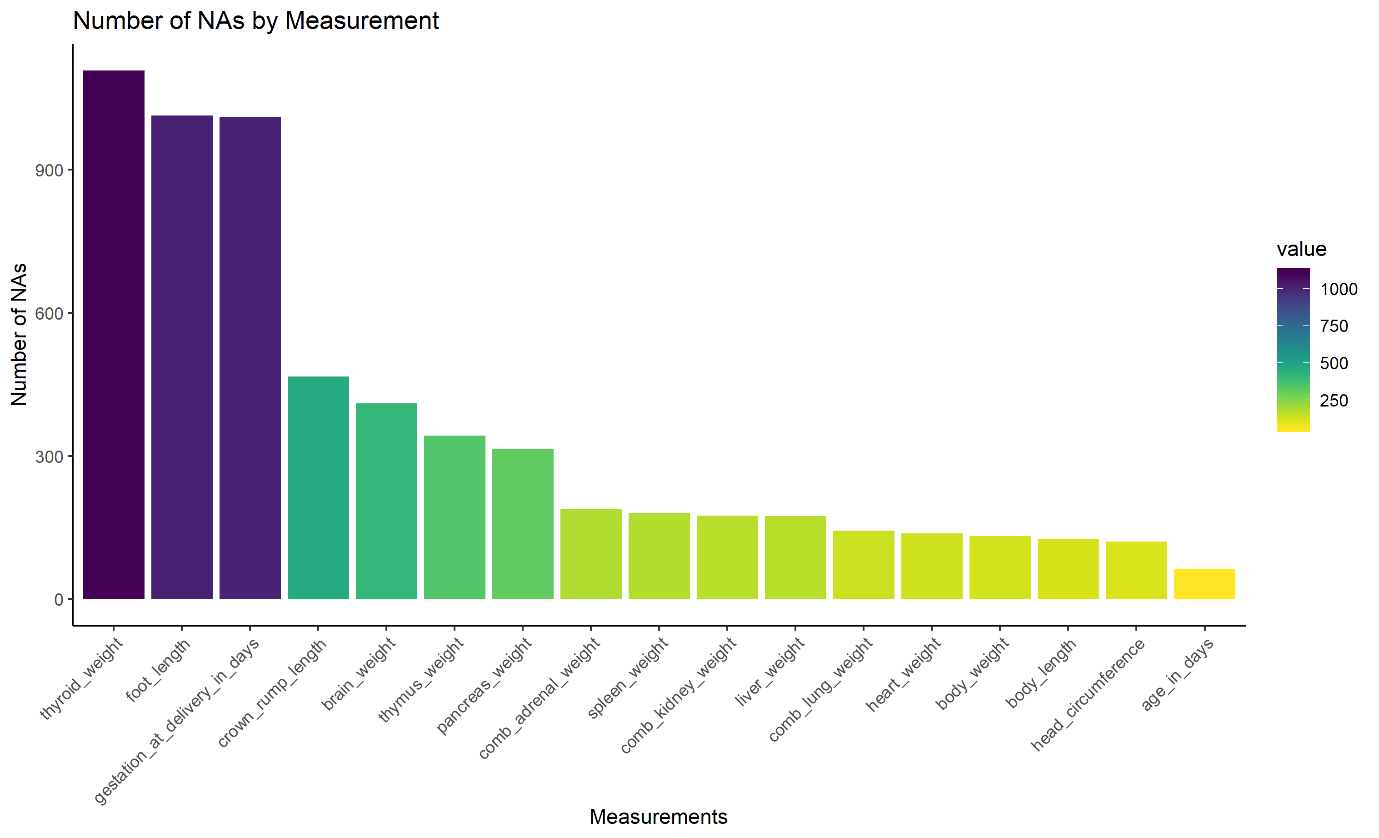


Figure 10 - Number of missing measurement values

Given the significant number of missing values in three features, thyroid\_weight, foot\_length and gestation\_at\_delivery\_in\_days these features will be omitted to maximise the retention of events with other features intact.

It was considered to use imputation of missing values. It was decided that this action could not be applied to gestation\_at\_delivery\_in\_days and to only consider post-mortems where the gestational age was known would have significantly biased the data as gestational age is known for hospital cases but not coroner’s cases.

### Imbalanced data

When using categorical data it is important to look for the presence of imbalanced categories. Below is a plot of all the categorical features used in the study data set.

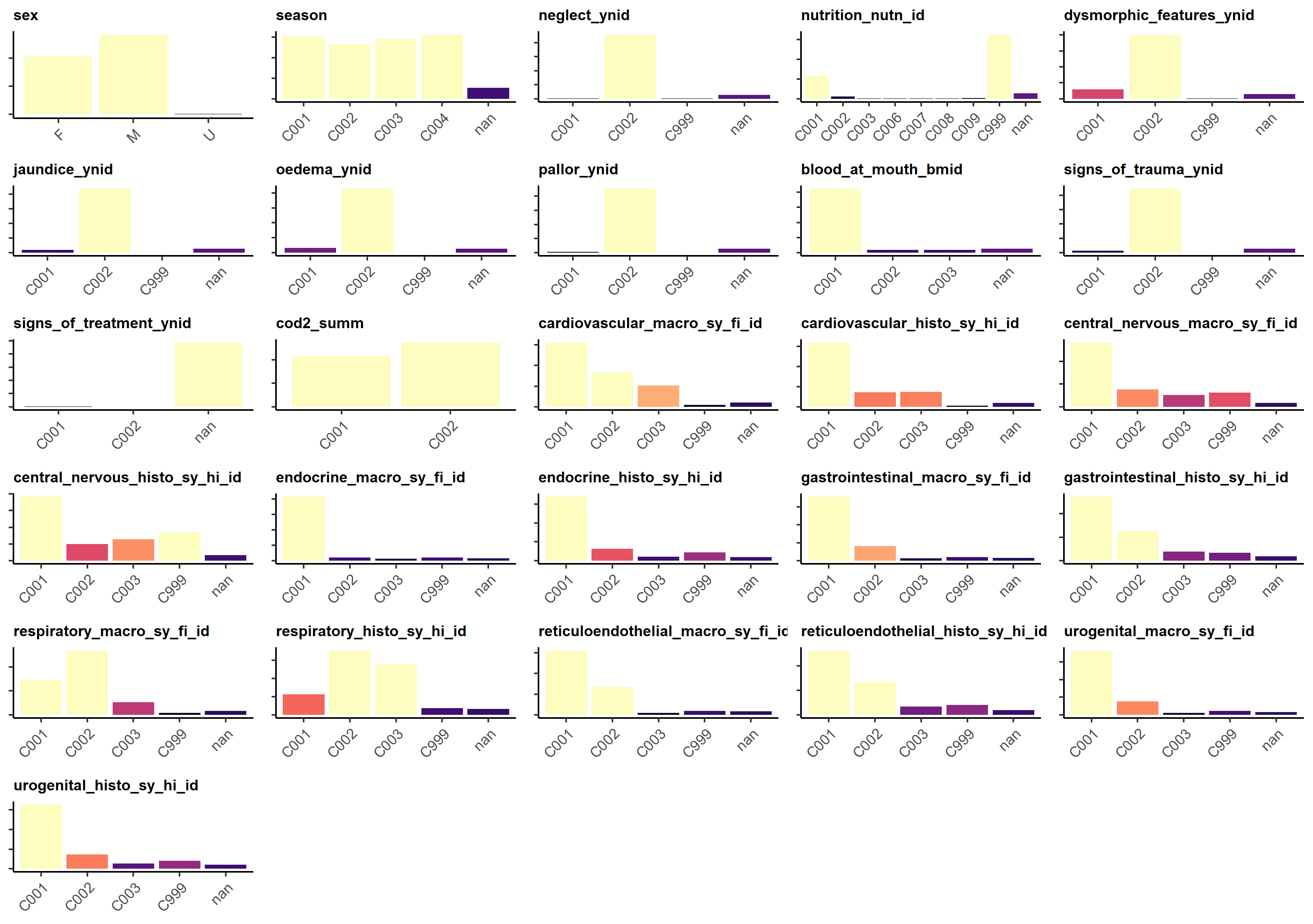


Figure 11 – Variability of Categorical Variables

The light coloured bars indicates that there are more than 500 events with that particular categorical value, the darkest bars indicate very few examples of a particular value exist. By using a categorical value of ‘nan’ then all categories are represented for all events. For the key categorical feature of ‘COD2\_SUMM’ then it is shown to be reasonable balanced.

## Decision Tree

The output captured for the decision tree model at each post-mortem stage is:

* Decision Tree Details
  + Variable Importance
  + Description of the Node and Split (including # going left or right and even surrogate splits.
  + CP Table
  + Split details
    - Split criteria
    - rows in this node
    - Misclassified
    - Predicted Class
    - % of rows in predicted class for this node.
* Decision Tree plot –a visual representation of the split details above
* Confusion Matrix – predictive accuracy based on training and test data split 80% training 20% test. The predictive accuracy of both C001 – Not determined and C002 – determined is also given.
* Relative Feature importance

### External

|  |
| --- |
| Decision Tree Details |
| Classification tree:  rpart(formula = cod2\_summ ~ ., data = data\_train, method = "class",  control = control)  Variables actually used in tree construction:  [1] age\_in\_days body\_weight dysmorphic\_features\_ynid\_c001  [4] oedema\_ynid\_c002  Root node error: 789/1768 = 0.44627  n= 1768  CP nsplit rel error xerror xstd  1 0.110900 0 1.00000 1.00000 0.026492  2 0.060837 2 0.77820 0.80355 0.025558  3 0.040558 3 0.71736 0.75919 0.025223  4 0.016477 4 0.67681 0.71483 0.024839  5 0.010000 5 0.66033 0.71863 0.024874 |
| n= 1768  node), split, n, loss, yval, (yprob)  \* denotes terminal node  1) root 1768 789 1 (0.4462670 0.5537330)  2) age\_in\_days>=15.5 1362 652 0 (0.5212922 0.4787078)  4) age\_in\_days< 275.5 1073 449 0 (0.5815471 0.4184529)  8) oedema\_ynid\_c002>=0.5 971 374 0 (0.6148301 0.3851699)  16) dysmorphic\_features\_ynid\_c001< 0.5 877 311 0 (0.6453820 0.3546180)  32) body\_weight< 0.5913 854 293 0 (0.6569087 0.3430913) \*  33) body\_weight>=0.5913 23 5 1 (0.2173913 0.7826087) \*  17) dysmorphic\_features\_ynid\_c001>=0.5 94 31 1 (0.3297872 0.6702128) \*  9) oedema\_ynid\_c002< 0.5 102 27 1 (0.2647059 0.7352941) \*  5) age\_in\_days>=275.5 289 86 1 (0.2975779 0.7024221) \*  3) age\_in\_days< 15.5 406 79 1 (0.1945813 0.8054187) \* |
| Decision Tree Plot |
|  |
| Confusion Matrix and Relative Feature Importance |
| |  |  | | --- | --- | |  |  | |

Figure 12 – Decision Tree – External Examination

### Internal – Stage 1 – Organs

|  |
| --- |
| Decision Tree Details |
| Classification tree:  rpart(formula = cod2\_summ ~ ., data = data\_train, method = "class",  control = control)  Variables actually used in tree construction:  [1] age\_in\_days body\_weight brain\_weight heart\_weight oedema\_ynid\_c001  Root node error: 624/1340 = 0.46567  n= 1340  CP nsplit rel error xerror xstd  1 0.246795 0 1.00000 1.00000 0.029263  2 0.089744 1 0.75321 0.76122 0.028062  3 0.060897 2 0.66346 0.71154 0.027613  4 0.014423 3 0.60256 0.63462 0.026767  5 0.011218 5 0.57372 0.64744 0.026921  6 0.010000 6 0.56250 0.63942 0.026825 |
| n= 1340  node), split, n, loss, yval, (yprob)  \* denotes terminal node  1) root 1340 624 0 (0.5343284 0.4656716)  2) age\_in\_days>=14.5 1062 408 0 (0.6158192 0.3841808)  4) age\_in\_days< 310.5 862 280 0 (0.6751740 0.3248260)  8) heart\_weight< 0.61165 808 234 0 (0.7103960 0.2896040)  16) brain\_weight< 0.3006 715 185 0 (0.7412587 0.2587413)  32) oedema\_ynid\_c001< 0.5 696 172 0 (0.7528736 0.2471264) \*  33) oedema\_ynid\_c001>=0.5 19 6 1 (0.3157895 0.6842105) \*  17) brain\_weight>=0.3006 93 44 1 (0.4731183 0.5268817)  34) body\_weight< 0.3684 57 22 0 (0.6140351 0.3859649) \*  35) body\_weight>=0.3684 36 9 1 (0.2500000 0.7500000) \*  9) heart\_weight>=0.61165 54 8 1 (0.1481481 0.8518519) \*  5) age\_in\_days>=310.5 200 72 1 (0.3600000 0.6400000) \*  3) age\_in\_days< 14.5 278 62 1 (0.2230216 0.7769784) \* |
| Decision Tree Plot |
|  |
| Confusion Matrix and Relative Feature Importance |
| |  |  | | --- | --- | |  |  | |

Figure 13 – Decision Tree – Internal Examination – Organs

### Internal – Stage 2 – Macro investigation

|  |
| --- |
| Decision Tree Details |
| Classification tree:  rpart(formula = cod2\_summ ~ ., data = data\_train, method = "class",  control = control)  Variables actually used in tree construction:  [1] age\_in\_days cardiovascular\_macro\_sy\_fi\_id\_c003  [3] central\_nervous\_macro\_sy\_fi\_id\_c003 comb\_lung\_weight  [5] respiratory\_macro\_sy\_fi\_id\_c003  Root node error: 624/1340 = 0.46567  n= 1340  CP nsplit rel error xerror xstd  1 0.243590 0 1.00000 1.00000 0.029263  2 0.107372 1 0.75641 0.75641 0.028022  3 0.078526 2 0.64904 0.66987 0.027178  4 0.056090 3 0.57051 0.59936 0.026314  5 0.038462 4 0.51442 0.54487 0.025527  6 0.014423 5 0.47596 0.51603 0.025065  7 0.010000 7 0.44712 0.49359 0.024682 |
| n= 1340  node), split, n, loss, yval, (yprob)  \* denotes terminal node  1) root 1340 624 0 (0.5343284 0.4656716)  2) age\_in\_days>=12.5 1090 423 0 (0.6119266 0.3880734)  4) cardiovascular\_macro\_sy\_fi\_id\_c003< 0.5 1003 346 0 (0.6550349 0.3449651)  8) respiratory\_macro\_sy\_fi\_id\_c003< 0.5 924 282 0 (0.6948052 0.3051948)  16) central\_nervous\_macro\_sy\_fi\_id\_c003< 0.5 839 222 0 (0.7353993 0.2646007)  32) comb\_lung\_weight< 0.50225 781 181 0 (0.7682458 0.2317542)  64) age\_in\_days< 310.5 675 128 0 (0.8103704 0.1896296) \*  65) age\_in\_days>=310.5 106 53 0 (0.5000000 0.5000000)  130) comb\_lung\_weight< 0.20725 66 24 0 (0.6363636 0.3636364) \*  131) comb\_lung\_weight>=0.20725 40 11 1 (0.2750000 0.7250000) \*  33) comb\_lung\_weight>=0.50225 58 17 1 (0.2931034 0.7068966) \*  17) central\_nervous\_macro\_sy\_fi\_id\_c003>=0.5 85 25 1 (0.2941176 0.7058824) \*  9) respiratory\_macro\_sy\_fi\_id\_c003>=0.5 79 15 1 (0.1898734 0.8101266) \*  5) cardiovascular\_macro\_sy\_fi\_id\_c003>=0.5 87 10 1 (0.1149425 0.8850575) \*  3) age\_in\_days< 12.5 250 49 1 (0.1960000 0.8040000) \* |
| Decision Tree Plot |
|  |
| Confusion Matrix and Relative Feature Importance |
| |  |  | | --- | --- | |  |  | |

Figure 14 - Decision Tree - Internal Examination - Macro

### Internal – Stage 3 – Histological investigation

|  |
| --- |
| Decision Tree Details |
| Classification tree:  rpart(formula = cod2\_summ ~ ., data = data\_train, method = "class",  control = control)  Variables actually used in tree construction:  [1] age\_in\_days cardiovascular\_histo\_sy\_hi\_id\_c003  [3] central\_nervous\_macro\_sy\_fi\_id\_c003 respiratory\_histo\_sy\_hi\_id\_c003  Root node error: 626/1340 = 0.46716  n= 1340  CP nsplit rel error xerror xstd  1 0.372204 0 1.00000 1.00000 0.029175  2 0.089457 1 0.62780 0.62780 0.026622  3 0.055911 2 0.53834 0.60224 0.026294  4 0.028754 3 0.48243 0.52396 0.025142  5 0.010000 4 0.45367 0.48882 0.024547 |
| n= 1340  node), split, n, loss, yval, (yprob)  \* denotes terminal node  1) root 1340 626 0 (0.5328358 0.4671642)  2) respiratory\_histo\_sy\_hi\_id\_c003< 0.5 889 284 0 (0.6805399 0.3194601)  4) age\_in\_days>=12.5 751 187 0 (0.7509987 0.2490013)  8) cardiovascular\_histo\_sy\_hi\_id\_c003< 0.5 694 141 0 (0.7968300 0.2031700)  16) central\_nervous\_macro\_sy\_fi\_id\_c003< 0.5 638 104 0 (0.8369906 0.1630094) \*  17) central\_nervous\_macro\_sy\_fi\_id\_c003>=0.5 56 19 1 (0.3392857 0.6607143) \*  9) cardiovascular\_histo\_sy\_hi\_id\_c003>=0.5 57 11 1 (0.1929825 0.8070175) \*  5) age\_in\_days< 12.5 138 41 1 (0.2971014 0.7028986) \*  3) respiratory\_histo\_sy\_hi\_id\_c003>=0.5 451 109 1 (0.2416851 0.7583149) \* |
| Decision Tree Plot |
|  |
| Confusion Matrix and Relative Feature Importance |
| |  |  | | --- | --- | |  |  | |

Figure 15 - Decision Tree - Internal Examination - Histology

### All Stages

For all four stage it can be shown the relative importance of a feature as the different stages of a post-mortem pregresses.

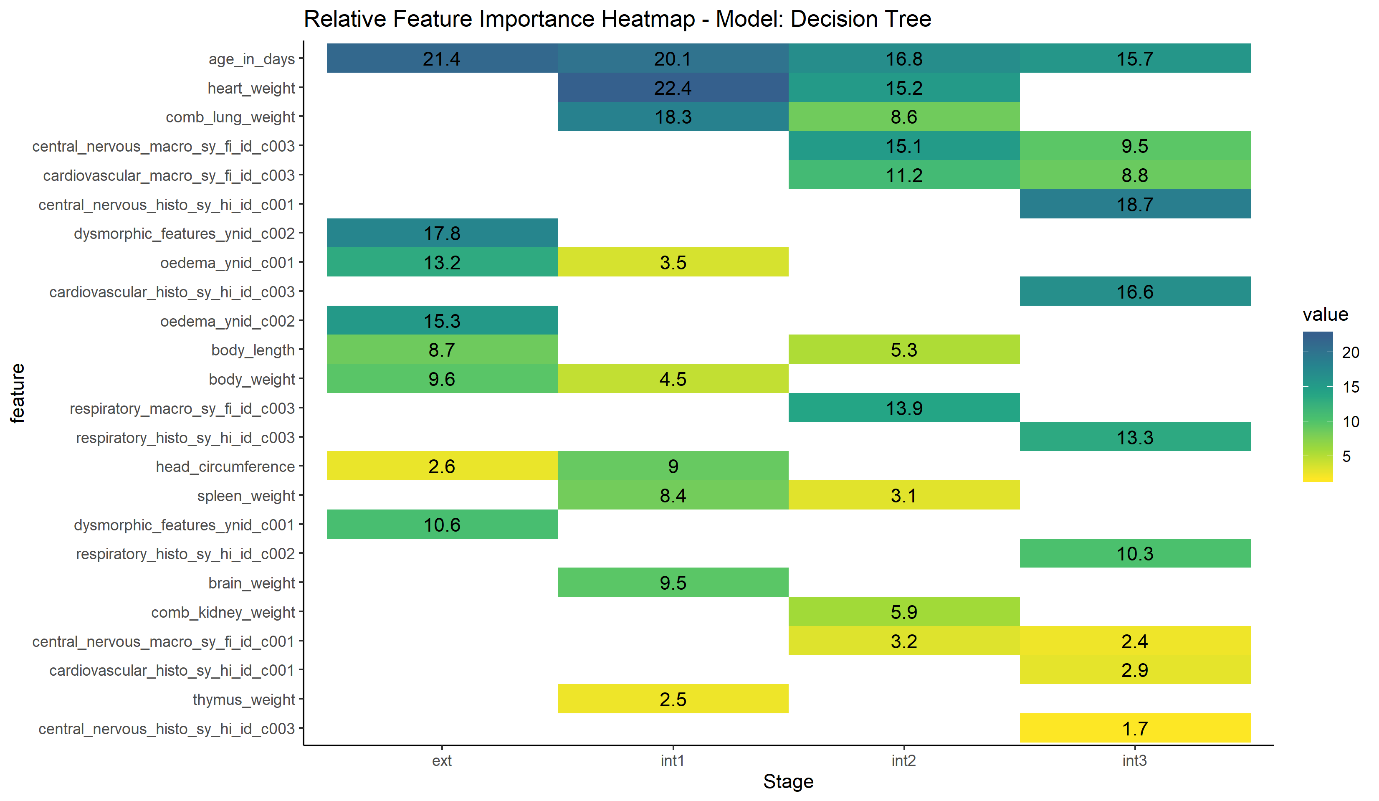


Figure 16 – Decision Tree – Relative Feature Importance – All Stages

## Ensemble Models

For the ensemble models the output is focussed on predictive accuracy shown by the confusion matrices and Relative Feature Importance.

### Random Forest

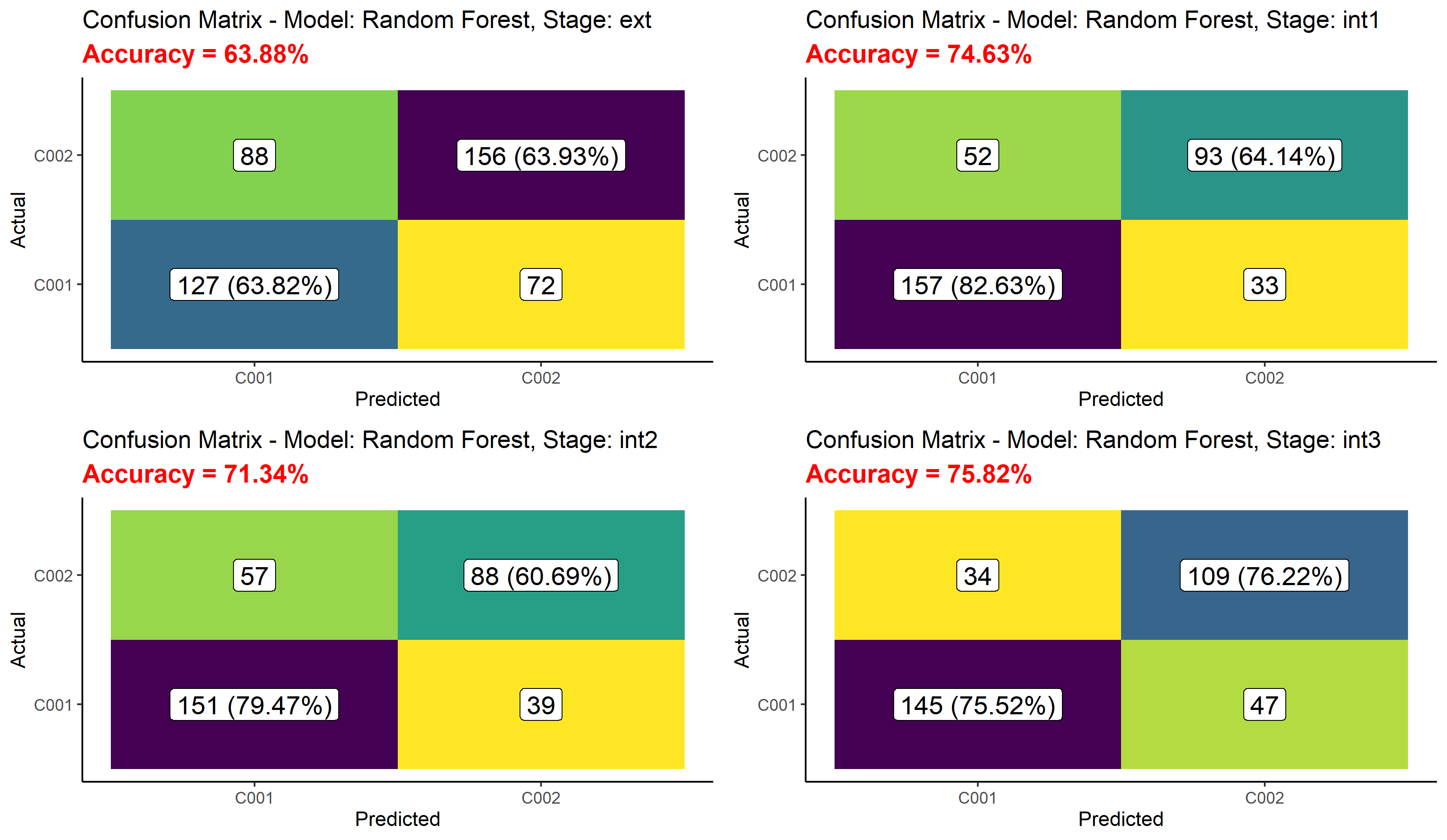


Figure 17 – Random Forest Confusion Matrices – All stages

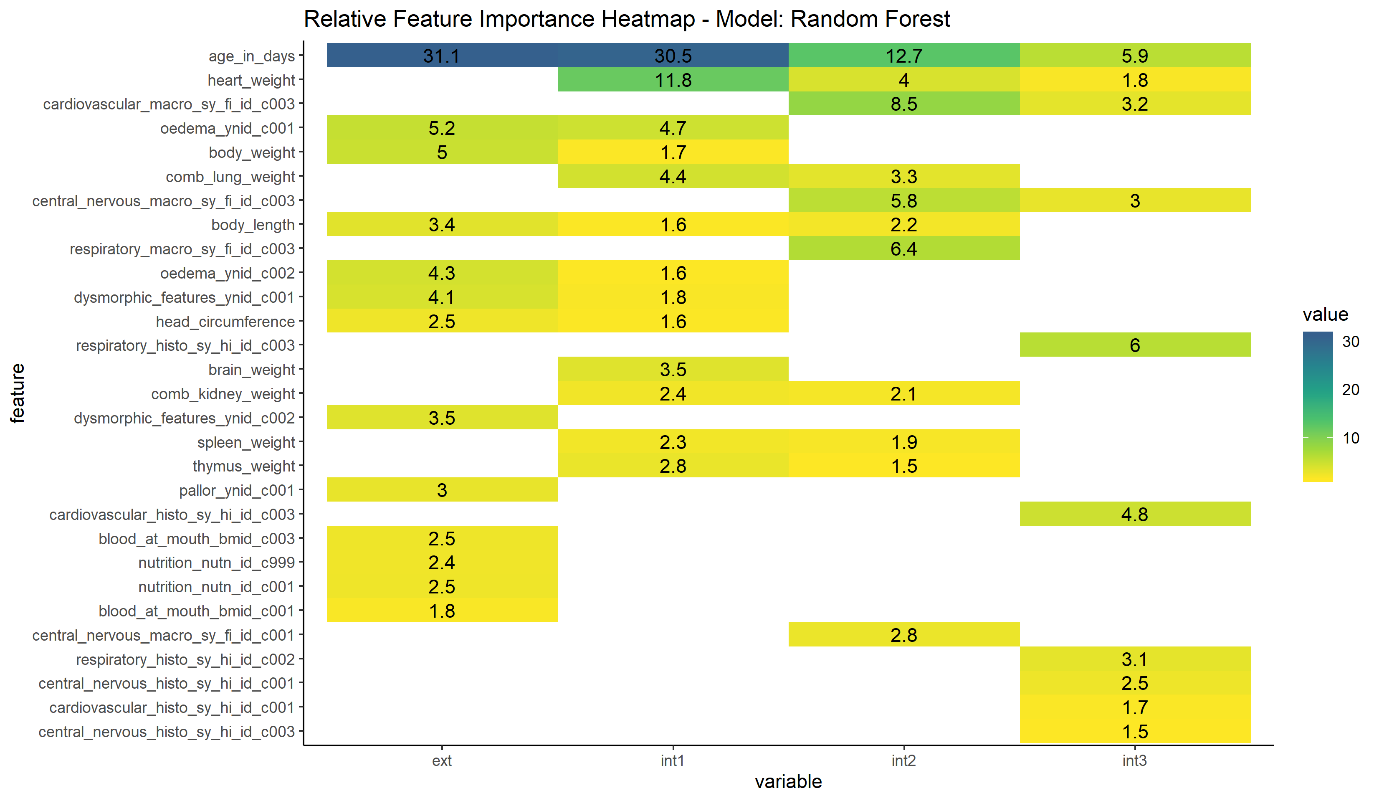


Figure 18 – Random Forest Relative Feature Importance – All Stages

### Gradient Boosted Decision Tree

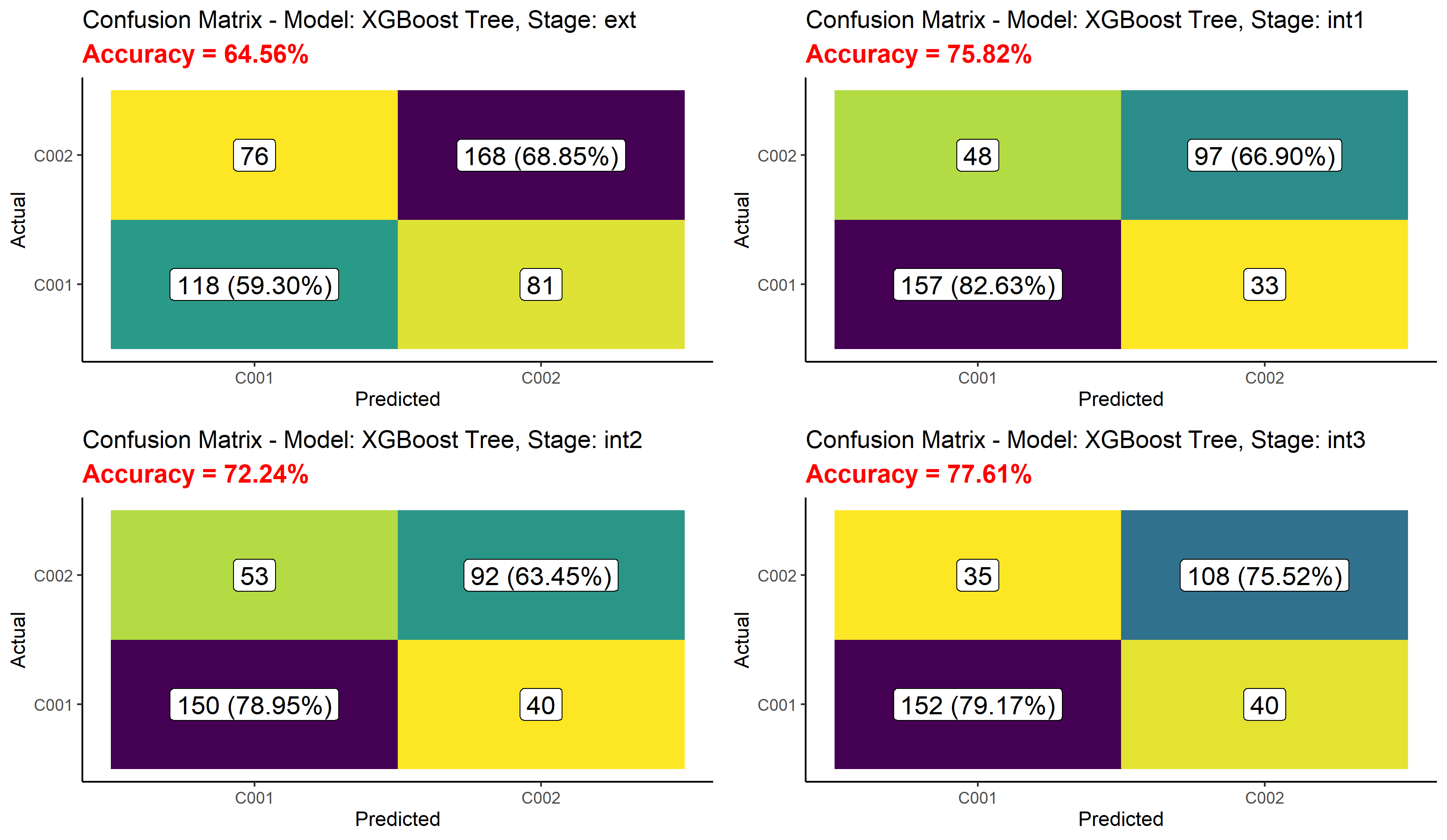


Figure 19 – XGBoost Confusion Matrices – All stages

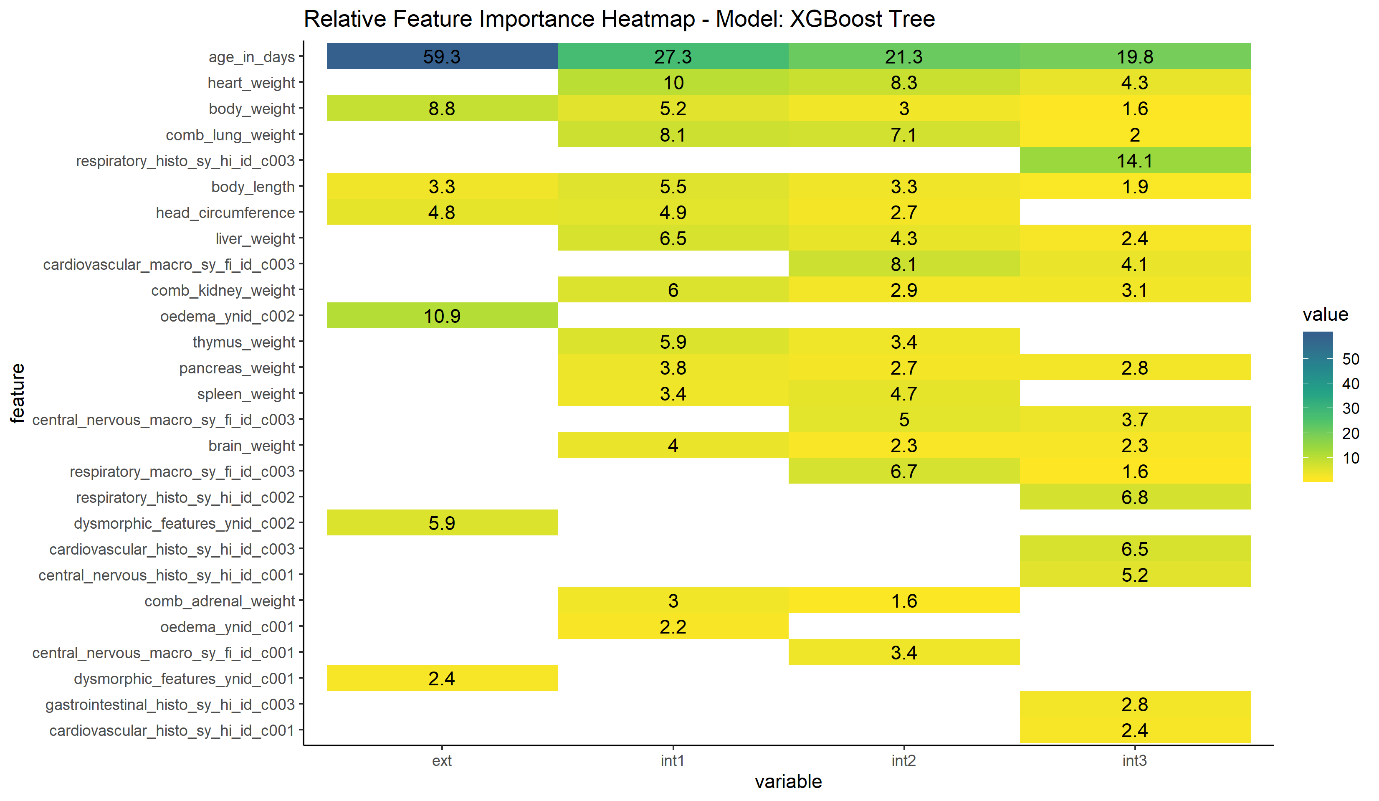


Figure 20 – XGBoost Relative Feature Importance – All Stages

## Compare models with varying random seeds

Each of the models above was run 5 times with different random seeds and the results compared:

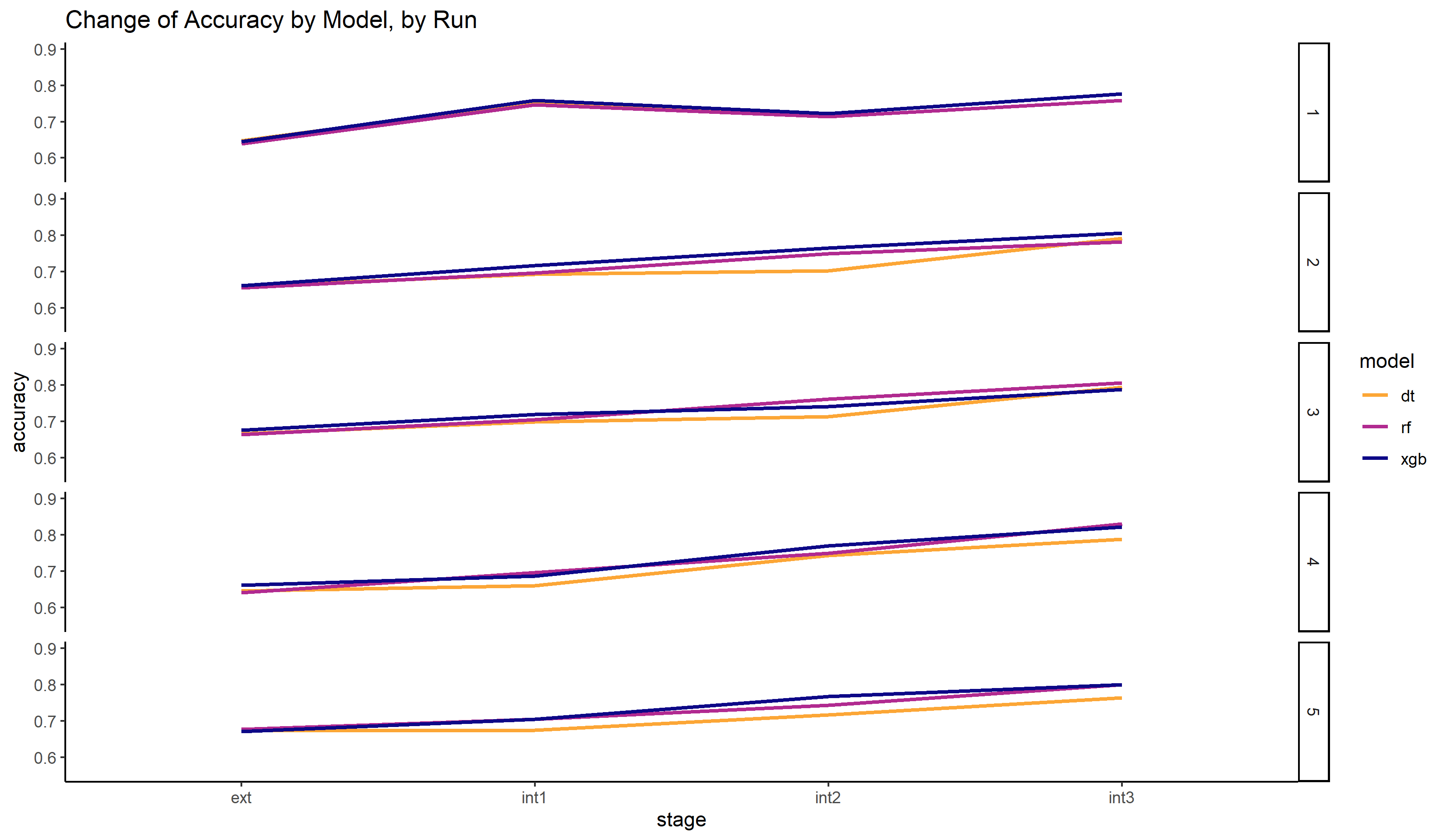


Figure 21 – Predictive Accuracy of each stage by Model by Run

The first plot shows how the different models agree in the increased predictive as the different stages of the post-mortem.

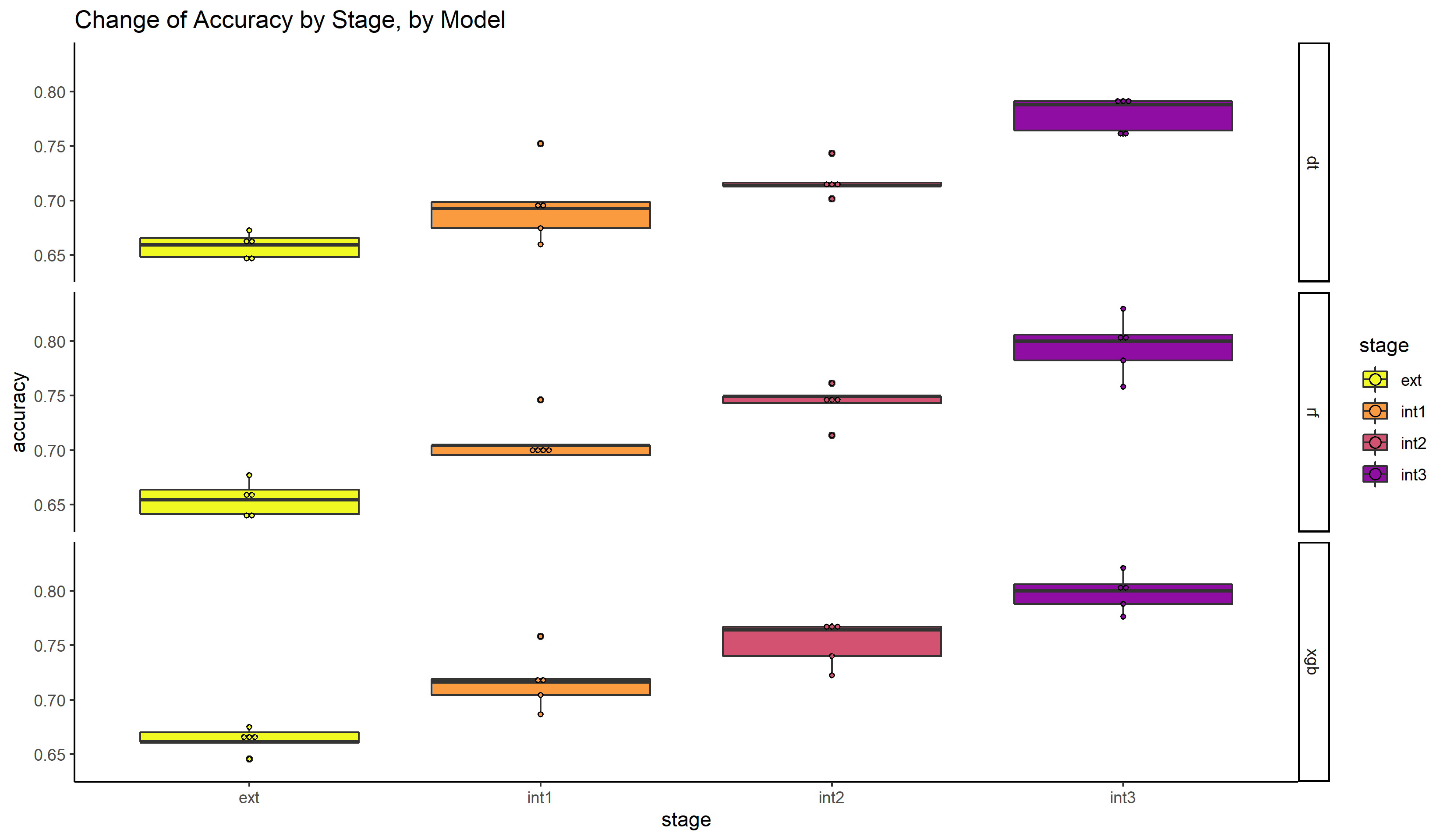


Figure 22 – Variability in Accuracy by Model by Stage

The second plot shows the variability of the predictive accuracy by model by stage for the five different random seeds.

Finally the change in relative feature importance by model for the five different random seeds. We have chosen to just use the first and last stage to demonstrate the relative stability from one random seed to another.

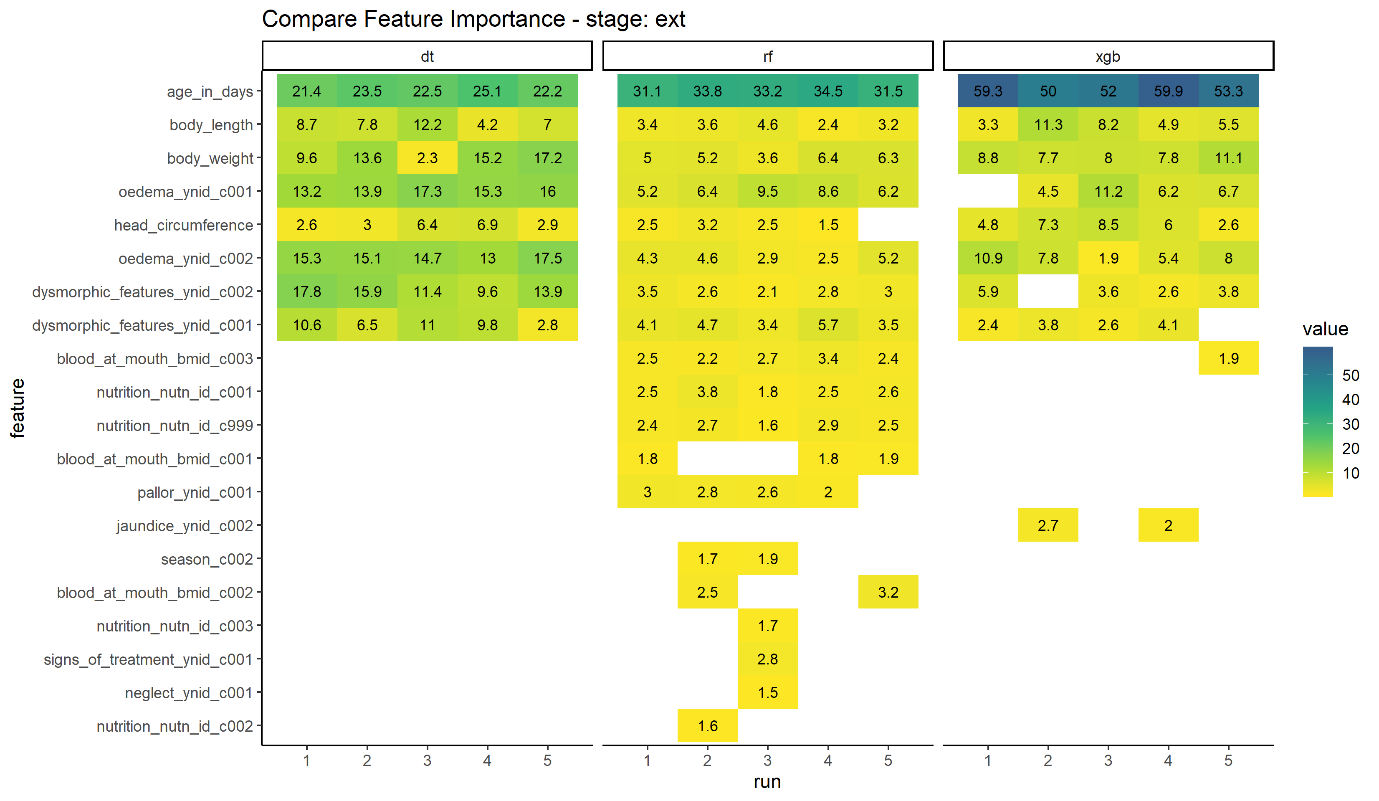


Figure 23 – Compare Feature Importance by Model, Stage: Ext, by Run

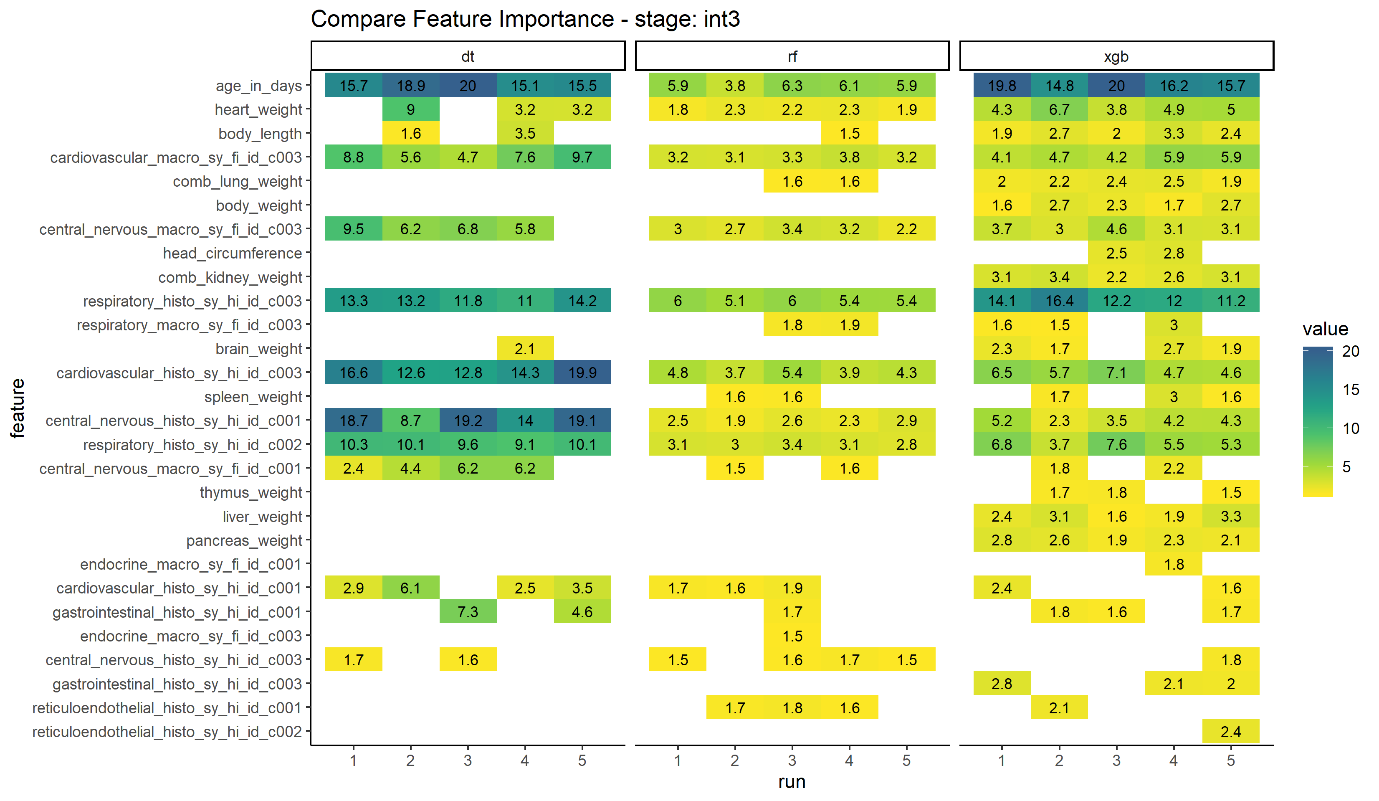


Figure 24 – Compare Feature Importance by Model, Stage: Int3, by Run

# Results

The results of this project are assessed in terms of the predictive accuracy of the different models and then the relative feature importance and their change as the different stages of the post-mortem.

## Predictive Accuracy

From the first table it can be seen that both ensemble methods outperform the basic decision tree model and the XGBoost model outperforms Random Forest but only by a small margin. Also the underlying increase on predictive accuracy as the post-mortem stages progress is reflected across all three models.

When it comes to the predictive accuracy of the individual classifications the picture becomes a little less clear with the Random Forest model outperforming the XGBoost model at certain stages.

Taking these results as a whole the XGBoost model would be classed as the most accurate model. Though the basic analytic interpretation of the decision would still seem to be valid.

## Relative Feature Importance by Stage of Post-mortem

As XGBoost is the most accurate of the models considered then the results for Feature Importance will be used from this model:

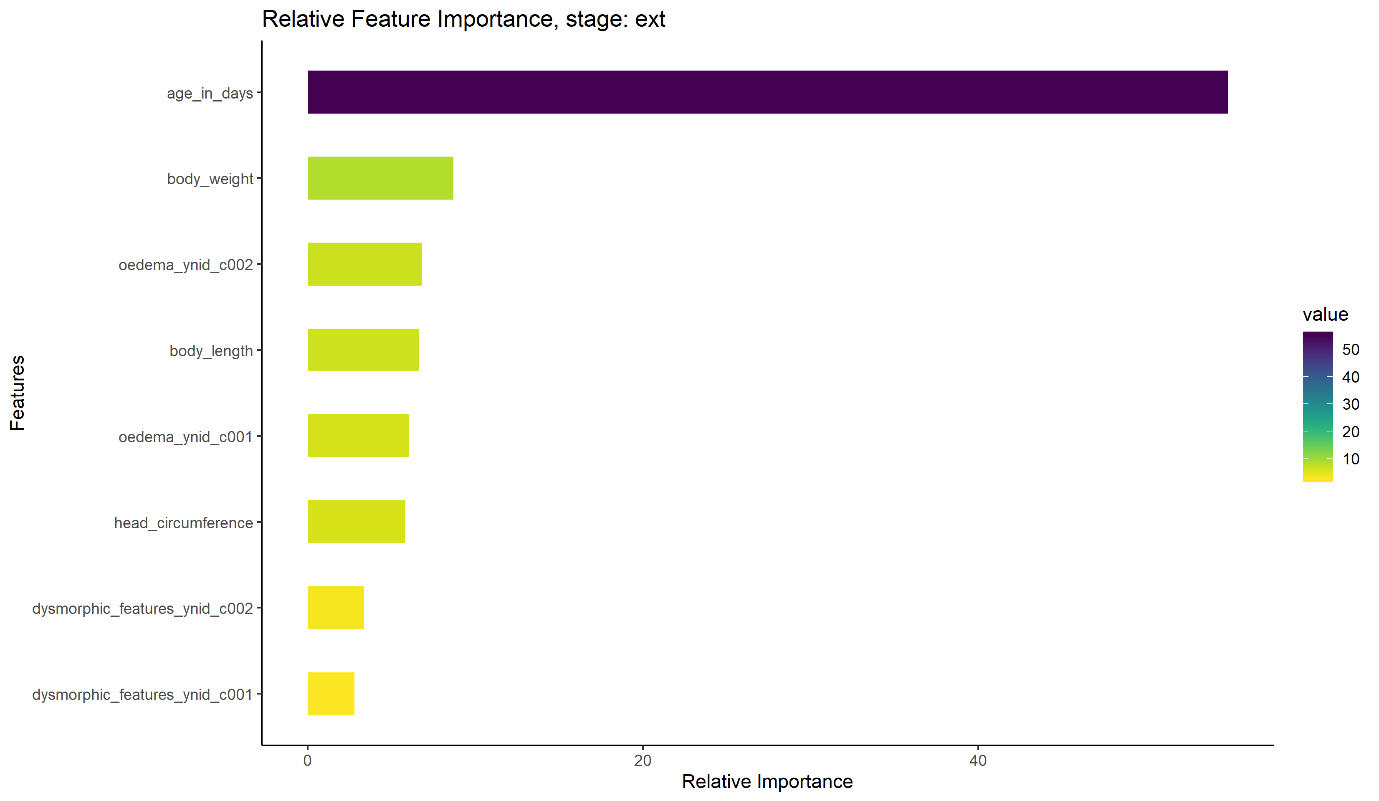


Figure 25 – Relative Feature Importance, Stage: External Examination

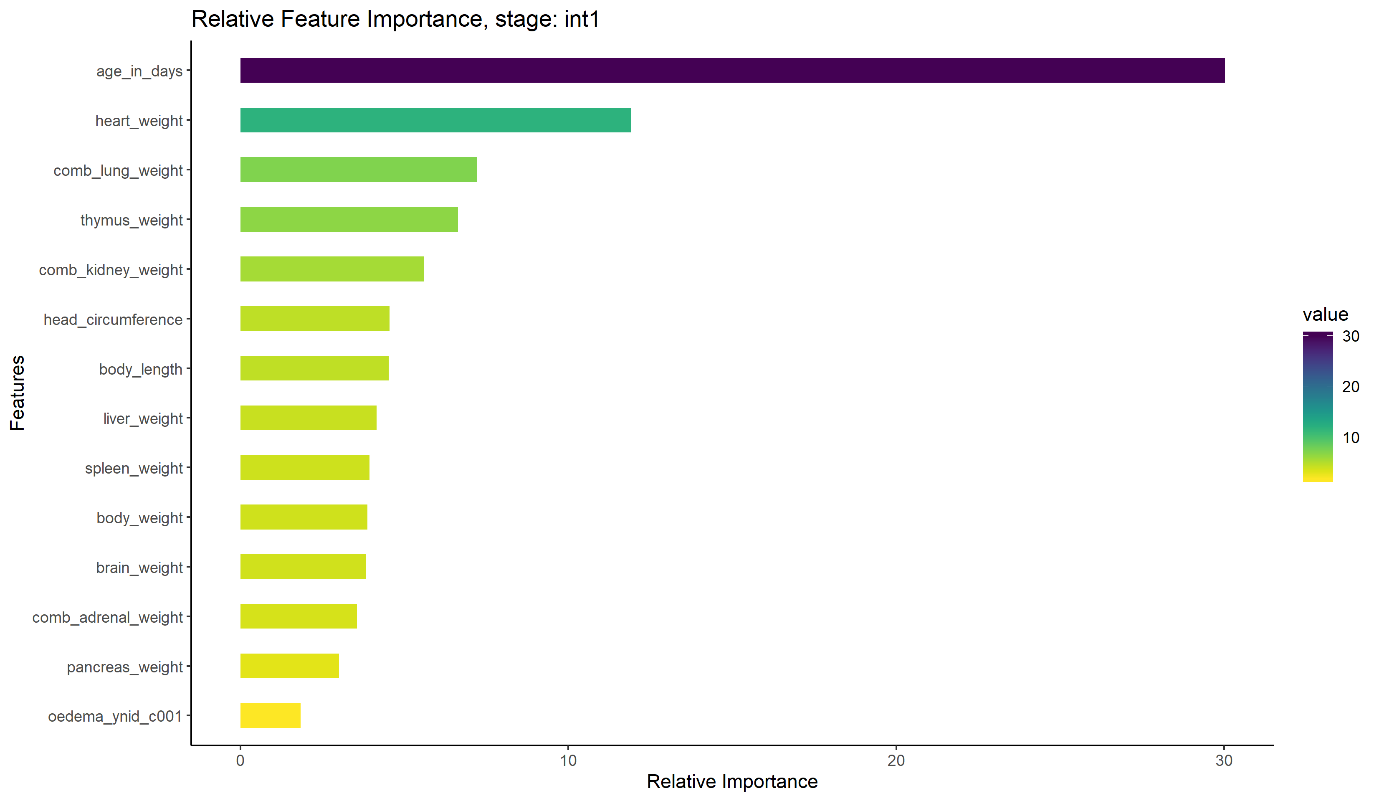


Figure 26 – Relative Feature Importance, Stage: Internal Examination - Organs

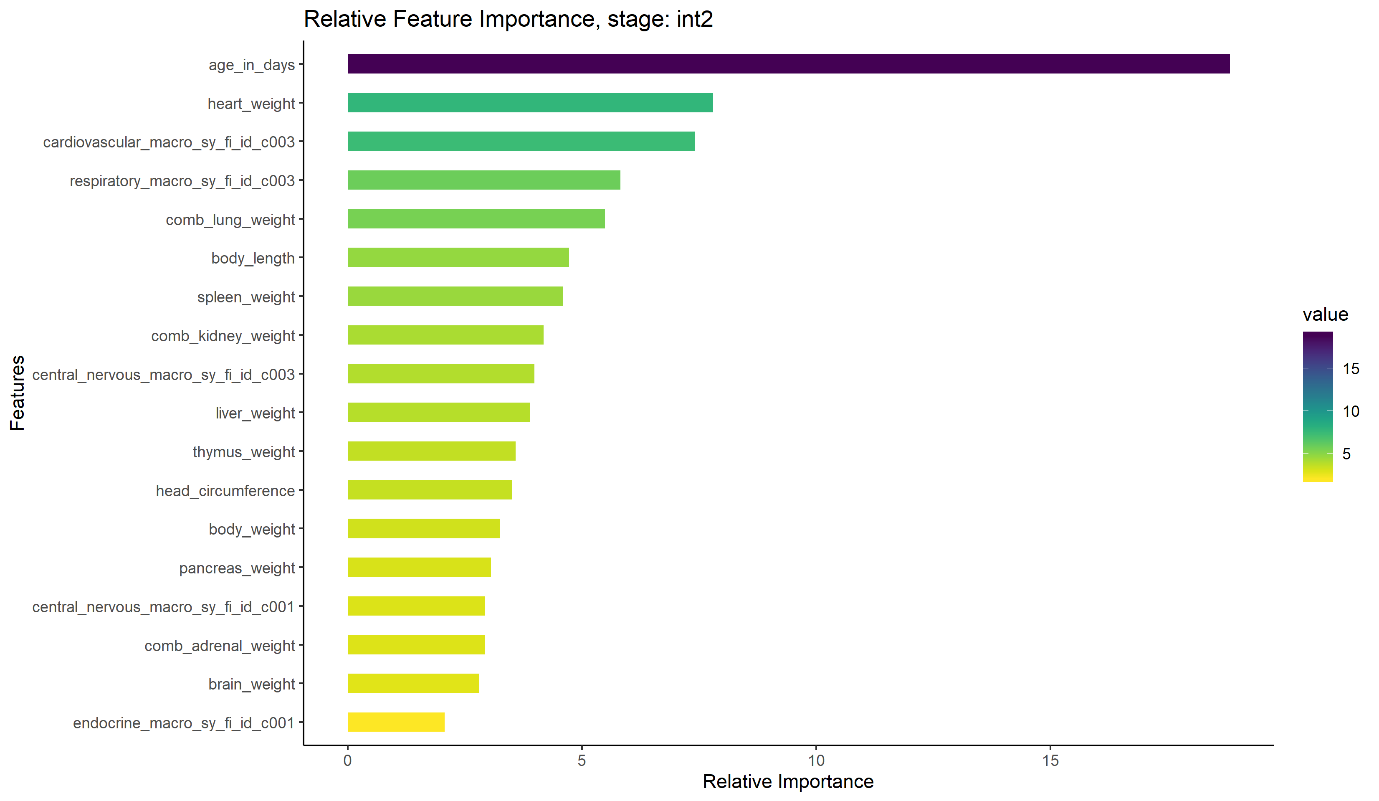


Figure 27 – Relative Feature Importance, Stage: Internal Examination - Macro

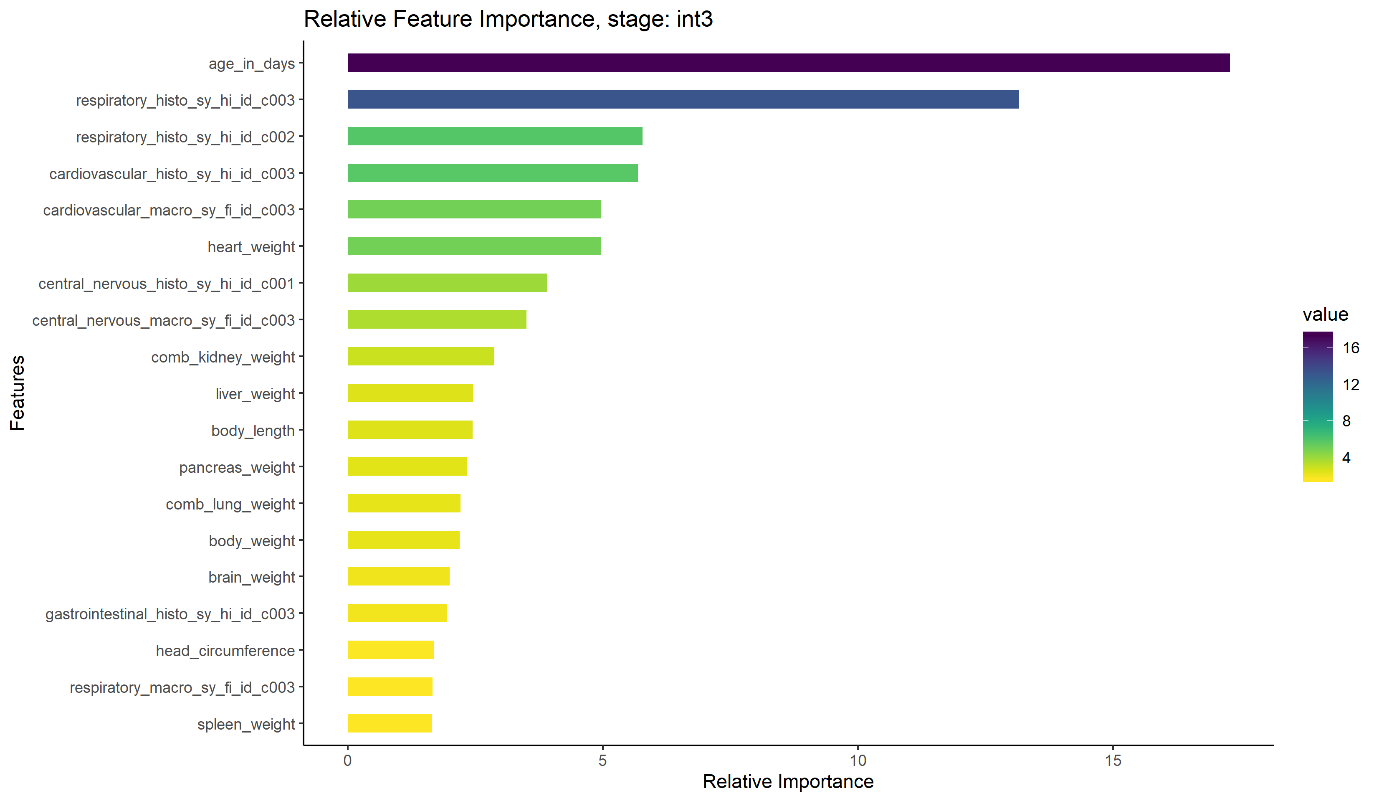


Figure 28 – Relative Feature Importance, Stage: Internal Examination - Histology

# Conclusion

## Project Summary

The project will be evaluated against the aims laid out at the start of the project.

### Data Engineering

The aim was to develop a routine to extract data from the existing Post-mortem Research Database into an entity attribute value schema that will make the data more readily available for data analytics.

* Some use case specific understanding i.e. underlying structure of the source database and how that data could be mapped onto Patients, Events and Event Attributes.
* Generic routines to create and update events and event attributes
* Generic routines to add additional event attributes
* Generic routines to create RDVs for different stages for analysis
* Generic routines to carry out some data wrangling i.e. one-hot encoding.

### Decision Tree

The aim was to apply the Decision Tree Analytical method to the extracted data to develop operational strategies that can be applied to paediatric post-mortems to prioritise which data is required to achieve the target of specifying the cause of death.

### Ensemble Models

The aim was to investigate ensemble strategies, specifically Random Forests and Gradient Boosting to see how these techniques can improve on the basic Decision Tree method.

## Project Evaluation

Identify areas for improvement; discuss what you could have done better (particularly important if you failed some of your targets or your results as not as expected)

* Python versus R
* Agile approach
  + Reviewed progress from last week
  + Set objectives for the coming week
* Underestimated time to tune each model
  + Needed a separate model for each stage.

## Recommendations for Future Work

* Improved measurement normalisation
* Imputation of missing values for certain measurements.
* Incorporate latest post-mortem results.

# References

*Abolfazl Ravanshad, 2018. Ensemble Methods [online] Medium. Available at:* <https://medium.com/@aravanshad/ensemble-methods-95533944783f> *[Accesses 2019-03-24]*

*Abolfazl Ravanshad, 2018. Gradient Boosting vs Random Forest. [online] Medium. Available at:* <https://medium.com/@aravanshad/gradient-boosting-versus-random-forest-cfa3fa8f0d80> *[Accesses 2019-03-24]*

*Analytics Vidhya Contributors (2016). A Complete Tutorial on Tree Based Modeling from Scratch. [online] Analytics Vidhya Avaiable at:* <https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/> *[Accesses 2019-03-24]*

Archer, K.J., 2010. rpartOrdinal: an R package for deriving a classification tree for predicting an ordinal response. *Journal of Statistical Software*, *34*, p.7.

*Ben‐Sasi, K., Chitty, L.S., Franck, L.S., Thayyil, S., Judge‐Kronis, L., Taylor, A.M. and Sebire, N.J., 2013. Acceptability of a minimally invasive perinatal/paediatric autopsy: healthcare professionals' views and implications for practice. Prenatal diagnosis, 33(4), pp.307-312.*

*Borodin, A. and Zavyalova, Y., 2015. On an EAV based approach to designing of medical data model for mobile healthcare service. UBICOMM 2015, p.33.*

*Dinu, Valentin; Nadkarni, Prakash (2007), "Guidelines for the effective use of entity-attribute-value modeling for biomedical databases", International Journal of Medical Informatics,****76****(11–12): 769–79*

*Horn, L.C., Langner, A., Stiehl, P., Wittekind, C. and Faber, R., 2004. Identification of the causes of intrauterine death during 310 consecutive autopsies. European Journal of Obstetrics & Gynecology and Reproductive Biology, 113(2), pp.134-138.*

*Löper, D., Klettke, M., Bruder, I. and Heuer, A., 2013. Enabling flexible integration of healthcare information using the entity-attribute-value storage model. Health information science and systems, 1(1), p.9.*

Luellen, J.K., Shadish, W.R. and Clark, M.H., 2005. Propensity scores: An introduction and experimental test. *Evaluation Review*, *29*(6), pp.530-558.

(Example of using RPART package)

*Lullaby Trust (2018). Facts-and-Figures-for-2015-released-2017. Updated 2018. London: Lullaby Trust.*

*NHS Contributors (2018). Post Mortem. [online] NHS. Available at: https://www.nhs.uk/conditions/post-mortem [Accessed 2019-03-24].*

*Prashant Gupta 2017. Decision Trees in Machine Learning. [online] Medium. Avaiable at:* <https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052> *[Accessed 2019-03-24]*

*Pryce, J.W., Bamber, A.R., Ashworth, M.T., Kiho, L., Malone, M. and Sebire, N.J., 2014. Reference ranges for organ weights of infants at autopsy: results of> 1,000 consecutive cases from a single centre. BMC clinical pathology, 14(1), p.18.*

*RCPath Contributors (2018). Paediatric Pathoogy. [online] Royal College of Pathologists. Available at:* <https://www.rcpath.org/trainees/examinations/examinations-by-specialty/paediatric-pathology.html> *[Accessed 2019-03-24].*

*Song, Yan-Yan and Ying Lu.2015. “Decision tree methods: applications for classification and prediction” Shanghai archives of psychiatry vol. 27,2: p130-5.*

Therneau, T.M. and Atkinson, E.J., 1997. An introduction to recursive partitioning using the RPART routines.

*Weber, M.A., Ashworth, M.T., Risdon, R.A., Hartley, J.C., Malone, M.A.R.I.A.N. and Sebire, N.J., 2008. The role of post-mortem investigations in determining the cause of sudden unexpected death in infancy. Archives of disease in childhood, 93(12), pp.1048-1053.*

*Weber, M.A., Ashworth, M.T., Risdon, R.A., Malone, M., Burch, M. and Sebire, N.J., 2008. Clinicopathological features of paediatric deaths due to myocarditis: an autopsy series. Archives of disease in childhood, 93(7), pp.594-598.*

*Weber, M.A., Ashworth, M.T., Risdon, R.A., Brooke, I., Malone, M. and Sebire, N.J., 2009. Sudden unexpected neonatal death in the first week of life: autopsy findings from a specialist centre. The Journal of Maternal-Fetal & Neonatal Medicine, 22(5), pp.398-404.*

*Weber, M.A., Klein, N.J., Hartley, J.C., Lock, P.E., Malone, M. and Sebire, N.J., 2008. Infection and sudden unexpected death in infancy: a systematic retrospective case review. The Lancet, 371(9627), pp.1848-1853.*

*Wikipedia Contributors (2019). Entity Attribute Model. [online] Wikipedia. Available at:* <https://en.wikipedia.org/wiki/Entity-attribute-value_model> *[Accessed 2019-03-24].*

*Yang, Y., Morillo, I.G. and Hospedales, T.M., 2018. Deep neural decision trees. arXiv preprint arXiv:1806.06988.*

# Bibliography

*Caruana, R. and Niculescu-Mizil, A., 2006, June. An empirical comparison of supervised learning algorithms. In Proceedings of the 23rd international conference on Machine learning (pp. 161-168). ACM.*

*James, G., Witten, D., Hastie, T. and Tibshirani, R., 2013. An introduction to statistical learning (Vol. 112, p. 18). New York: springer.*

*Quinlan, J.R., 1986. Induction of decision trees. Machine learning, 1(1), pp.81-106.*

List of reference to be included:

Python

Pycharm

pyODBC

R

R studio

ggplot2

The viridis color palettes:

<https://cran.r-project.org/web/packages/viridis/vignettes/intro-to-viridis.html>

Rpart

<https://cran.r-project.org/web/packages/rpart/index.html>

Randomforest

<https://cran.r-project.org/web/packages/randomForest/>

<https://cran.r-project.org/web/packages/randomForest/randomForest.pdf>

XGBoost

<https://cran.r-project.org/web/packages/xgboost/>

<https://cran.r-project.org/web/packages/xgboost/xgboost.pdf>

Tuning xgboost in R: Part I

<https://insightr.wordpress.com/2018/05/17/tuning-xgboost-in-r-part-i/>

# Glossary of Terms

From the Royal College of Pathologists:

<https://www.rcpath.org/discover-pathology/what-is-pathology/glossary-of-terms.html>

And Wikipedia:

<https://en.wikipedia.org>

|  |  |
| --- | --- |
| Term | Description |
| Data Wrangling | the process of transforming and mapping data from one "raw" data form into another format with the intent of making it more appropriate and valuable for a variety of downstream purposes such as analytics. |
| Dysmorphic | A dysmorphic feature is a difference of body structure. It can be an isolated finding in an otherwise normal individual, or it can be related to a congenital disorder, genetic syndrome, or birth defect. |
| Entity Attribute Value (EAV) Schema | Entity–attribute–value model (EAV) is a data model to encode, in a space-efficient manner, entities where the number of attributes (properties, parameters) that can be used to describe them is potentially vast, but the number that will actually apply to a given entity is relatively modest. |
| ETL | The process of pulling data out of one database and place it into another database. Within this process the data is cleans and transformed to be in a more appropriate schema for analytics. |
| Histopathology | The branch of pathology that involves looking at tissue under the microscope to diagnose disease. If you have a mole or a breast lump removed, the histopathologist will examine it to work out what it is. |
| Metabolic | A group of overlapping areas of clinical practice with a common dependence on the detailed understanding of basic biochemistry and medicine. These areas fall within the territory of both physicians and chemical pathologists. They include clinical nutrition, lipid abnormalities, diabetes, metabolic bone disease, porphyria and adult inherited metabolic disorders. |
| Microbiology | The diagnosis of infection caused by bacteria, fungi, parasites and viruses; identification of the best treatment options for infection; and the monitoring of antibiotic resistance. It also includes testing for how well a patient is responding to treatment of infection. |
| Oedema | an abnormal accumulation of fluid in the interstitium, located beneath the skin and in the cavities of the body |
| Structured Query Language (SQL) | A domain-specific language used in programming and designed for managing data held in a relational database management system. |
|  |  |

# Appendix A – Example Project Code

## Python Function with ODBC

|  |
| --- |
| **def** return\_null\_string(variable):  **if** pandas.isnull(variable):  **return "NULL"  else**:  **return "'"** + variable.replace(**"'"**,**"''"**) + **"'"**  **def** GetConceptID(cnxn, crsr, category, parent\_concept\_id, code, label = **None**, value\_type\_concept\_id = 1):  *'''  Gets Concept id  If it doesn't exist then adds it.* **:param** *cnxn: ODBC Connection* **:param** *crsr: ODBC Cursor* **:param** *category: String* **:param** *code: string* **:param** *label: string - default blank as not required if concept exist* **:param** *value\_type\_concept\_id: integer; default is concept id* **:return***: integer; 0 if not been able to create and doesn't exist.  '''   # Create SQL string to find concept* SQLstring = **"SELECT "** SQLstring += **" concept\_id "** SQLstring += **"FROM "** SQLstring += **" ha\_concepts "** SQLstring += **"WHERE "** SQLstring += **" category = '"** + category + **"' "** SQLstring += **" AND code = '"** + code + **"'"** SQLstring += **";"** crsr.execute(SQLstring)   row = crsr.fetchone()   *#If concept doesn't exist then insert it* **if not** row:   SQLinsert = **"INSERT INTO ha\_concepts "** SQLinsert += **" (category, parent\_concept\_id, code, label, value\_type\_concept\_id) "** SQLinsert += **"VALUES "** SQLinsert += **" ('"** + category + **"', "** + str(parent\_concept\_id) + **", '"** + code + **"', "** + return\_null\_string(  label) + **", "** + str(value\_type\_concept\_id) + **")"** SQLinsert += **";"** crsr.execute(SQLinsert)  cnxn.commit()   *#Re execute SQL select* crsr.execute(SQLstring)   row = crsr.fetchone()   **if** row:  **return** row.concept\_id  **else**:  **return** 0 |

## Python Function to Create COD2\_SUMM

|  |
| --- |
| **def** create\_cod2\_Summ\_attribute\_from\_cod2\_attribute(cnxn, crsr):  *'''* **:param** *cnxn:* **:param** *crsr:* **:return***:   'build an array of existing key values & labels from ha\_concepts  'Create an array of new attribute key values - add to ha\_concepts  'Assign new keys to old keys.  'process all existing attributes  ' add new event with mapped value   ' code label  ' 1 Unknown - No Abnormal Findings  ' 2 Unknown - Non-Contributory Findings  ' 3 Unknown - Possible/Probable Contributory Findings  ' 4 Infection  ' 5 CNS  ' 6 Respiratory System  ' 7 Cardiovascular System  ' 8 GIT  ' 9 Urogenital System  '10 Lymphoreticular/Haematological  '11 Musculoskeletal  '12 Peripheral Nervous System/Neuromuscular Junction  '13 Metabolic  '14 Anaphylaxis  '15 Immunological/Autoimmune  '16 Neoplasia Benign  '17 Neoplasia Malignant  '18 Chromosomal Abnormality  '19 Congenital Anomalies/Malformation Syndrome  '20 Accidental  '21 NAI/Homicide  '22 Suicide  '23 Wigglesworth 1  '24 Wigglesworth 2  '25 Wigglesworth 3  '26 Wigglesworth 4  '27 Wigglesworth 5  '28 Normal Fetus  '29 Traumatic NOS  '30 Endocrine  '32 No PM, For PM MRI ONLY  '33 No PM, see Memo  '34 Other  '999 N/A   'code label  ' 1 Unknown (1 - 3)  ' 2 Known (4 - 30)  ' 3 Other (31 - 34)  '999 N/A   '''   # Add Attributes and values* parent\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation"**, **None**, **"PostMortem"**)   value\_type\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/"**, **None**, **"Concept"**)   parent\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**,  parent\_concept\_id, **"tblFinalDiagnoses"**, **"tblFinalDiagnoses"**,  value\_type\_concept\_id)   event\_attribute\_type\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr,  **"/EventAttribute/Observation/PostMortem/tblFinalDiagnoses"**,  parent\_concept\_id, **"COD2\_SUMM"**, **"COD2\_SUMM"**,  value\_type\_concept\_id)   parent\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"LookUp"**)  parent\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/LookUp"**,  parent\_concept\_id, **"COD2\_SUMM"**, **"COD2\_SUMM"**,  value\_type\_concept\_id)   value\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr,  **"/EventAttribute/Observation/PostMortem/LookUp/COD2\_SUMM"**,  parent\_concept\_id, **"001"**, **"Unknown"**, value\_type\_concept\_id)  value\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr,  **"/EventAttribute/Observation/PostMortem/LookUp/COD2\_SUMM"**,  parent\_concept\_id, **"002"**, **"known"**, value\_type\_concept\_id)  value\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr,  **"/EventAttribute/Observation/PostMortem/LookUp/COD2\_SUMM"**,  parent\_concept\_id, **"003"**, **"Other"**, value\_type\_concept\_id)  value\_concept\_id = Create\_HAS\_Tables.GetConceptID(cnxn, crsr,  **"/EventAttribute/Observation/PostMortem/LookUp/COD2\_SUMM"**,  parent\_concept\_id, **"994"**, **"N/A"**, value\_type\_concept\_id)   *# Get all Event Attributes of the original type   # Get event\_attributes for current event* SQLstring = **"SELECT "** SQLstring += **" event\_attribute\_id, "** *#* ***TODO we seem to have a duplicate CaseID*** SQLstring += **" event\_id, "** SQLstring += **" event\_attribute\_type\_concept\_id, "** SQLstring += **" value\_text, "** SQLstring += **" value\_numeric, "** SQLstring += **" value\_datetime, "** SQLstring += **" value\_concept\_id, "** SQLstring += **" value\_type\_concept\_id, "** SQLstring += **" code, "** SQLstring += **" label "** SQLstring += **"FROM ha\_event\_attributes "** SQLstring += **" LEFT OUTER JOIN ha\_concepts "** SQLstring += **" ON ha\_concepts.concept\_id = ha\_event\_attributes.value\_concept\_id "** SQLstring += **"WHERE "** SQLstring += **" (ha\_concepts.category = '/EventAttribute/Observation/PostMortem/LookUp/COD2\_COD2ID') "** SQLstring += **";"** crsr.execute(SQLstring)  EventAttributeRows = crsr.fetchall()   row = 0  print(**"Processing Events - COD2\_SUMM Attribute From COD2 Attribute"**)   **for** EventAttributeRow **in** EventAttributeRows:   row += 1  sys.stdout.write(**"\r \r {0}"**.format(str(row)))  sys.stdout.flush()   **if** EventAttributeRow.code **in** (**"001"**, **"002"**, **"003"**, **"999"**):  code = **"001"** text = **"Unknown"  elif** EventAttributeRow.code **in** (**"031"**, **"032"**, **"033"**, **"034"**):  code = **"003"** text = **"Other"  else**:  code = **"002"** text = **"Known"** Create\_HAS\_Tables.AddEventAttribute(cnxn, crsr, EventAttributeRow.event\_id,  **"Observation/PostMortem/tblFinalDiagnoses"**, **"COD2\_SUMM"**, **"COD2\_SUMM"**, **"ID"**,  code, text)  print(**""**)  print(**"Done!"**) |

## Python Function to Create RDV

|  |
| --- |
| **def** populate\_event\_attributes(cnxn, crsr, stage, EventAttributes):   EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblCases"**, **None**,  **"CASEID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblCases"**, **None**,  **"SSN"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblFinalDiagnoses"**, **None**,  **"COD2\_SUMM"**))   **if** stage **in** (**"ext\_x"**,**"ext"**,**"int1"**,**"int2"**,**"int3"**,**"int2\_s"**,**"int3\_s"**):   EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"BodyWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"CrownRumpLength"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"HeadCircumference"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"CrownRumpLength"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"BodyLength"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"FootLength"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"Neglect\_YNID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"Nutrition\_NutnID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"DysmorphicFeatures\_YNID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"Jaundice\_YNID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"Oedema\_YNID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"Pallor\_YNID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"BloodAtMouth\_BMID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"SignsOfTrauma\_YNID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblExternalExams"**, **None**,  **"SignsOfTreatment\_YNID"**))   **if** stage **in** (**"int1\_x"**, **"int1"**, **"int2"**, **"int3"**,**"int2\_s"**,**"int3\_s"**):   EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"HeartWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"CombLungWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"LiverWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"PancreasWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"ThymusWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"SpleenWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"CombAdrenalWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"ThyroidWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"CombKidneyWeight"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/tblInternalExams"**, **None**,  **"BrainWeight"**))   **if** stage **in** (**"int2\_x"**,**"int2"**,**"int3"**):   EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"CardiovascularMacro\_SyFiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"CentralNervousMacro\_SyFiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"EndocrineMacro\_SyFiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"GastrointestinalMacro\_SyFiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"RespiratoryMacro\_SyFiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"ReticuloendothelialMacro\_SyFiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"UrogenitalMacro\_SyFiID"**))   **if** stage **in** (**"int3\_x"**,**"int3"**):   EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"CardiovascularHisto\_SyHiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"CentralNervousHisto\_SyHiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"EndocrineHisto\_SyHiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"GastrointestinalHisto\_SyHiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"RespiratoryHisto\_SyHiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"ReticuloendothelialHisto\_SyHiID"**))  EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"UrogenitalHisto\_SyHiID"**))   **if** stage **in** (**"int2\_s"**,**"int3\_s"**):   EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"CaseMacro\_CsFiID"**))  **if** stage == **"int3\_s"**:   EventAttributes.append(  Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,  **"CaseHisto\_CsHiID"**))  **return** EventAttributes  **def** create\_rdv\_study(cnxn, crsr, stage, include\_null = **True**):   *# Select Patient Attributes* EventPatientAttributes = []  EventPatientAttributes.append(Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/PatientAttribute"**, **None**, **"AC"**)) *# Age Category* EventPatientAttributes.append(Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/PatientAttribute"**, **None**, **"AG"**)) *# Age in Days* EventPatientAttributes.append(Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/PatientAttribute"**, **None**, **"GA"**)) *# Gestational Age at delivery   # Select Patient Attribute Filters* EventPatientAttributeFilters = []  EventPatientAttributeFilterValues = []   *# Select Event Attributes* EventAttributes = populate\_event\_attributes(cnxn, crsr, stage,[])   *# Is this necessary if measurements were made we could use them.* EventAttributeFilters = []  EventAttributeFilters.append(Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem"**, **None**,**"INC\_IN\_STUDY"**))  EventAttributeFilterValues = []  EventAttributeFilterValues.append(Create\_HAS\_Tables.GetConceptID(cnxn, crsr, **"/EventAttribute/Observation/PostMortem/LookUp/INC\_IN\_STUDY"**, **None**, **"001"**))   file\_name = **"rdv\_"** + **"study\_"** + stage   create\_rdv(cnxn, crsr, file\_name, EventPatientAttributes, EventPatientAttributeFilters, EventPatientAttributeFilterValues, EventAttributes, EventAttributeFilters, EventAttributeFilterValues) |

## R Model Function

|  |
| --- |
| # Load libraries  library(dplyr)  library(ggplot2)  library(ggmosaic)  library(grid)  library(gridExtra)  library(gtable)  library(viridis)  library(rpart)  library(rpart.plot)  library(caret)  library(lubridate)  library(reshape2)  # xgboost added:  library(xgboost)  library('DiagrammeR') # NB installed package  library('rsvg') # NB installed package  library('DiagrammeRsvg') # NB installed package  # run.seed - random seed for this run  # rdv.type, - rdv type adjusted or not adjusted (\_adj)  # importance.min, - min importance for feature importance plots (1.5)  # source.dir, - location of RDV files  # results.sub.dir, - location to store results  # file.suffix, - to distinguidh the files for this run  # stage.list, - list of post-mortem stages  # ext.train.index, - training index for external stage  # int1.train.index,  # int2.train.index,  # int3.train.index    RunXGBModel <- function(run.seed,  rdv.type,  importance.min,  source.dir,  results.sub.dir,  file.suffix,  stage.list,  ext.train.index,  int1.train.index,  int2.train.index,  int3.train.index  ) {    set.seed(run.seed)    model.name = "XGBoost Tree"  model.abv = "xgb"  run.str <- substr(file.suffix, nchar(file.suffix) - 1, nchar(file.suffix))    fimp.matrix <- setup.fimp.matrix(rdv.type, source.dir, run.str)    results.matrix <- setup.results.matrix(model.abv,length(stage.list))    for(stage.num in 1:length(stage.list)) {    stage <- stage.list[stage.num]    rm.col <- 1    print(paste0("Run: ", run.str, " Model: ",model.name," Stage: ",stage))    now <- Sys.time()    results.matrix[stage.num,rm.col] = run.str  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = format(now, "%Y-%m-%d %H:%M:%S")  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = rdv.type  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = run.seed  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = stage  rm.col = rm.col + 1    clean\_RDVData <- return\_clean\_rdvdata(source.dir, stage, rdv.type)    results.matrix[stage.num,rm.col] = nrow(clean\_RDVData)  rm.col = rm.col + 1    clean\_RDVData$cod2\_summ <- as.factor(clean\_RDVData$cod2\_summ)      #########################################################  # XGBoost    xgb.data <- clean\_RDVData    num\_class = length(levels(xgb.data$cod2\_summ))  cod2\_summ = clean\_RDVData$cod2\_summ    # Convert from class to numeric  label <- as.integer(xgb.data$cod2\_summ) - 1  xgb.data$cod2\_summ = NULL    if (stage == "ext") {  train.index <- ext.train.index  } else if (stage == "int1") {  train.index <- int1.train.index  } else if (stage == "int2") {  train.index <- int2.train.index  } else if (stage == "int3") {  train.index <- int3.train.index  }    train.data = as.matrix(xgb.data[train.index,])  train.label = label[train.index]  test.data = as.matrix(xgb.data[-train.index,])  test.label = label[-train.index]    # Store proportional split of COD2\_SUMM for this run  results.matrix[stage.num,rm.col] = prop.table(table(train.label))[1]  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = prop.table(table(train.label))[2]  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = prop.table(table(test.label))[1]  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = prop.table(table(test.label))[2]  rm.col = rm.col + 1    # Transform the two data sets into xgb.Matrix  xgb.train = xgb.DMatrix(data=train.data,label=train.label)  xgb.test = xgb.DMatrix(data=test.data,label=test.label)    # Stored tuned model results  if (stage == "ext") {  eta.value <- 0.3  max\_depth.value <- 6  gamma.value <- 5  min\_child\_weight.value <- 4  subsample.value <- 0.75  colsample\_bytree.value <- 0.50  } else if (stage == "int1") {  eta.value <- 0.185  max\_depth.value <- 6  gamma.value <- 2.65  min\_child\_weight.value <- 8  subsample.value <- 0.725  colsample\_bytree.value <- 0.50  } else if (stage == "int2") {  eta.value <- 0.2  max\_depth.value <- 5  gamma.value <- 1  min\_child\_weight.value <- 5  subsample.value <- 0.775  colsample\_bytree.value <- 0.5  } else if (stage == "int3") {  eta.value <- 0.17  max\_depth.value <- 5  gamma.value <- 3.2  min\_child\_weight.value <- 4  subsample.value <- 0.65  colsample\_bytree.value <- 0.525  } else {  # default values  eta.value <- 0.3  max\_depth.value <- 6  gamma.value <- 0  min\_child\_weight.value <- 1  subsample.value <- 1  colsample\_bytree.value <- 1  }    #Store best values  results.matrix[stage.num,rm.col] = eta.value  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = max\_depth.value  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = gamma.value  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = min\_child\_weight.value  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = subsample.value  rm.col = rm.col + 1  results.matrix[stage.num,rm.col] = colsample\_bytree.value  rm.col = rm.col + 1    params=list(  booster="gbtree",  eta=eta.value,  max\_depth=max\_depth.value,  gamma=gamma.value,  min\_child\_weight=min\_child\_weight.value,  subsample=subsample.value,  colsample\_bytree=colsample\_bytree.value,  objective="multi:softprob",  eval\_metric="mlogloss",  num\_class=num\_class  )    # Train the XGBoost classifer  xgb.fit=xgb.train(  params=params,  data=xgb.train,  nrounds=1000,  nthreads=1,  early\_stopping\_rounds=10,  watchlist=list(val1=xgb.train,val2=xgb.test),  verbose=0  )    # Review the final model and results  # xgb.fit    # Predict outcomes with the test data  xgb.pred = predict(xgb.fit,test.data,reshape=T)  xgb.pred = as.data.frame(xgb.pred)  colnames(xgb.pred) = levels(cod2\_summ)    # Use the predicted label with the highest probability  xgb.pred$prediction = apply(xgb.pred,1,function(x) colnames(xgb.pred)[which.max(x)])  xgb.pred$label = levels(cod2\_summ)[test.label + 1]    # Calculate the final accuracy  result = sum(xgb.pred$prediction==xgb.pred$label)/nrow(xgb.pred)    print(result)    results.matrix[stage.num,rm.col] = result  rm.col = rm.col + 1    # Create confusion matrix  table\_mat <- table(xgb.pred$label, xgb.pred$prediction)    # Loop over my\_matrix  for(row in 1:nrow(table\_mat)) {  for(col in 1:ncol(table\_mat)) {  results.matrix[stage.num,rm.col] = table\_mat[row, col]  rm.col = rm.col + 1  }  }    importance <- xgb.importance(model = xgb.fit)  imp <- as.data.frame(importance)    total\_imp = sum(imp$Gain)    for (imp\_row in 1:nrow(imp)){  res\_row = which(fimp.matrix$feature == imp[imp\_row,1])  fimp.matrix[res\_row, stage.num + 2] <- (imp[imp\_row,2] / total\_imp) \* 100  imp[imp\_row,2] <- (imp[imp\_row,2] / total\_imp) \* 100  }    # Remove less import features for clarity  imp <- subset(imp, Gain > importance.min)  plot.title = paste0("Relative Feature Importance - Model: ",model.name,", Stage: ",stage)    p <- ggplot(imp, aes(x=reorder(Feature, Gain), y=Gain))  p <- p + geom\_point()  p <- p + geom\_segment(aes(x=Feature,xend=Feature,y=0,yend=Gain))  p <- p + ggtitle(plot.title)  p <- p + ylab("Relative Importance")  p <- p + xlab("Feature")  p <- p + coord\_flip()  p <- p + theme\_classic()    print(p)    ggsave(paste0(results.sub.dir, "/", model.abv, "\_feature\_importance\_",stage, file.suffix,".png"))    # Single tree plot    p <- xgb.plot.tree(model = xgb.fit, trees = 0, show\_node\_id = TRUE)    print(p)    gr <- xgb.plot.tree(model=xgb.fit, trees=0, show\_node\_id = TRUE, render=FALSE)  export\_graph(gr, paste0(results.sub.dir, "/", model.abv, "\_tree\_",stage, file.suffix,".png"), width=1500, height=1900)  }    #############################  ## graph combined importance  #############################    data <- fimp.matrix  # Order results  data$feature <- with(data, reorder(feature, ext + int1 + int2 + int3))  # Remove 0 values and create structure to plot  data.m.ss <- subset(melt(data), value > importance.min)  # Create plot  plot.title = paste0("Relative Feature Importance Heatmap - Model: ",model.name)  p <- ggplot(data.m.ss, aes(x=variable, y=feature))  p <- p + ggtitle(plot.title)  p <- p + geom\_tile(aes(fill = value))  p <- p + scale\_fill\_viridis\_c(direction = -1, begin = .3, end = 1)  p <- p + geom\_text(aes(label = round(value, 1)))  p <- p + theme\_classic()    print(p)    ggsave(paste0(results.sub.dir, "/", model.abv, "\_feature\_importance\_hm", file.suffix, ".png"))  #############################  ## output results CSV files  #############################    write.csv(results.matrix, file = paste0(results.sub.dir, "/", model.abv, "\_results\_matrix", file.suffix, ".csv"),row.names=FALSE, na="")  write.csv(fimp.matrix, file = paste0(results.sub.dir, "/", model.abv, "\_feature\_importance\_matrix", file.suffix, ".csv"),row.names=FALSE, na="")  file.text <- "\_results\_matrix"  file.suffix <- sprintf("\_%02d", run.num)    p.list <- list()    for(stage.num in 1:length(stage.list)) {    stage.abv <- stage.list[stage.num]    save\_confusion\_matrix\_plot(model.abv, model.name, stage.abv, file.text, results.sub.dir, file.suffix)    p.list[[stage.num]] <- plot\_confusion\_matrix\_plot(model.abv, model.name, stage.abv, file.text, results.sub.dir, file.suffix)  }    g <- do.call(grid.arrange,p.list)    ggsave(paste0(results.sub.dir, "/", model.abv, "\_confusion\_matrix\_grid\_",stage.abv, file.suffix,".png"),g)    #################################  } |

# Appendix B – ETL Process

# Appendix C – Cause of Death Attribute Mapping

# Appendix D –RDV structures

## External

## Internal – Stage 1 – Organs

## Internal – Stage 2 – Macro investigation

## Internal – Stage 3 – Histological investigation

# Appendix E – Detailed output from an analytic process

RDM file output

# Appendix F – Deliverables

Project GIT HUB:

<https://github.com/jbooth04BBK/MScProject>