# Notes for project report

## Method

Explain how the data was gathered and analysed

This section may be divided into sub-sections describing the conceptual design work and the actual implementation separately. Any problems or difficulties and the suggested solutions should be mentioned. Alternative solutions and their evaluation should also be included.

* Reproducible results - Develop Project pipeline
  + PM Research Database => EAV Schema => Summary and Study based Attributes => RDV production => RDV Adjustments => Run models => review results
* Python
  + Data Engineering
  + PyCharm IDE
  + Testing and Profiling
* R code
  + Analytics and visualisation
  + Models evolved through three stages
    - Individual model
    - Model run for all PM stages
    - Model as a function called with multiple random keys.
* Use of GIT Hub

Notes from Machine Learning Coursework Requirements:

1.1 This part should normally describe clearly the method used in your assignment and any relevant parameters (e.g. for neural networks this includes number of hidden nodes, layers, type of activation functions etc.), and the rational for using this method. If you are using a particular library or tool, you still need to describe how the method/algorithm that you are using operates. Citing the library, tool, etc., and mentioning the library functions that you have used is not enough to get a high mark.

1.2 This part should describe any special techniques/algorithms used as part of your methodology. For example, the algorithm used for training a neural network and its parameters – e.g. if you use Rprop backpropagation then initial learning rate values used should be stated. Also, this part should describe any normalisation techniques used, or other pre-processing or balancing methods, and whether you have used some form of cross-validation, or weight decay, providing details of the particular method. Citing the library, tool, etc., and mentioning the library functions that you used is not enough to get a high mark.

## Data

* The nature of the data is that it is not normal
* Use of EAV schema to structure and clean data
* Introduce summary data for analytics and reporting

|  |
| --- |
| HL7® Fast Healthcare Interoperability Resources (FHIR®, pronounced "fire") is a next generation standards framework that leverages the latest web standards and applies a tight focus on implementation and was developed by healthcare standards developing organization, Health Level Seven International® (HL7®) |
| A FHIR resource can contain data about a patient, a device, an observation, and more. For a full list of FHIR resources, see the FHIR Resource Index |

## Results

Notes from Machine Learning Coursework Requirements:

Experiments, findings and discussion: you must present and discuss your results. You are expected to run several experiments and calculate basic statistics to summarise performance. Your report must include at least two figures which graphically illustrate quantitative aspects of your results, such as training/testing error curves, performance for sets of learned parameters, algorithm outputs, descriptive statistics, etc.

In this part, you should provide a detailed account of your experiments and results and discuss your findings. You can use Excel or other packages to provide charts - like the figure below, which uses error bars (Box and Whisker Charts in Excel), to show the performance of your algorithm in terms of generalisation. For example, the figure below shows generalisation with respect to number of hidden nodes used in a neural network based solution. Alternatively, one could use tables to provide the same information by giving for each number of hidden nodes the average value, the minimum value, and the maximum value of generalisation performance (in percentage of successfully recognised patterns) in the tests.

You could also discuss the cost of the computations, e.g. referring to the number of training iterations required or the number of error function evaluations (see figure below for the neural-network based solution discussed above)

In machine learning, overall results are also presented in tables like the one below that shows average performance in terms of recognition success as well as average classification success per class for two methods tested on the same dataset. Confusion matrices can also be used.

Method Class 1 (%) Class 2 (%) Average success (%)

Method 1 83 96 93

Method 2 73 93 88

# Key references

Should try normalised variables with and without one-hot encoding?

## GGPlot with multiple plots – facet

<https://www.r-bloggers.com/beyond-basic-r-plotting-with-ggplot2-and-multiple-plots-in-one-figure/>

## Tuning guides for all models

Good description on why to prune.

<https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/>

<https://www.analyticsvidhya.com/blog/2016/01/xgboost-algorithm-easy-steps/>

Good example of how to prune with RPART

<https://www.edureka.co/blog/implementation-of-decision-tree/>

<https://en.wikibooks.org/wiki/Data_Mining_Algorithms_In_R/Classification/Decision_Trees>

Random Forest

Documentation of RandomForest

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

A start agrees with starting with adjusting mtry then nTree

<https://www.r-bloggers.com/how-to-implement-random-forests-in-r/>

A very good walk through with detailed tuning using grids:

<https://uc-r.github.io/random_forests>

Mentions one-hot encoding

Another example of tuning RandomForest with which parameters to consider:

<https://www.hackerearth.com/blog/developers/practical-tutorial-random-forest-parameter-tuning-r/>

XGBoost

An interesting use of grid to tune XGBoost

<https://datascienceplus.com/chaid-vs-ranger-vs-xgboost-a-comparison/>

Excellent look a basic parameter tuning:

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

<https://www.r-bloggers.com/tuning-xgboost-in-r-part-ii/>

And this one:

<https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/beginners-tutorial-on-xgboost-parameter-tuning-r/tutorial/>

and this one uses grids:

<https://datascienceplus.com/extreme-gradient-boosting-with-r/>

Documentation by XGBoost

<https://github.com/tqchen/xgboost/blob/master/doc/how_to/param_tuning.md>

<https://github.com/tqchen/xgboost/blob/master/doc/parameter.md>

<https://datascience.stackexchange.com/questions/9364/hypertuning-xgboost-parameters>

Paper by the author of XGBoost:

<http://proceedings.mlr.press/v42/chen14.pdf>

I couldn’t access this link at GOSH

<https://rpubs.com/ippromek/336732>

A good page on SVM

<https://en.wikibooks.org/wiki/Data_Mining_Algorithms_In_R/Classification/SVM>

## Style guides

<https://stackoverflow.com/questions/1266279/how-to-organize-large-r-programs>

<https://google.github.io/styleguide/Rguide.xml>

## Project style/layout guides

<https://career-resource-center.udacity.com/portfolio/data-science-reports>

<http://blog.kaggle.com/2016/06/29/communicating-data-science-a-guide-to-presenting-your-work/>

<https://smallbusiness.chron.com/write-data-report-61330.html>

<https://www.dataquest.io/blog/data-science-project-style-guide/>

## GGPlot Multiple lines

<https://stackoverflow.com/questions/17150183/plot-multiple-lines-in-one-graph>

<http://www.sthda.com/english/wiki/ggplot2-line-types-how-to-change-line-types-of-a-graph-in-r-software>

<https://stackoverflow.com/questions/27350243/ggplot-line-graph-with-different-line-styles-and-markers>

## SVM vs XGBoost

<https://community.alteryx.com/t5/Data-Science-Blog/Why-use-SVM/ba-p/138440>

<https://www.researchgate.net/post/When_we_use_Support_Vector_machine_for_Classification>

<https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>

<https://www.quora.com/Why-does-XGBoost-perform-better-than-SVM>

<https://towardsdatascience.com/a-complete-view-of-decision-trees-and-svm-in-machine-learning-f9f3d19a337b>