1 Method

1.1 Running the modelling scripts

There is a python script (python3)for modelling each of the datasets: classification_abalone_dataset.py and classification_drugs_dataset.py.

The scripts can either be run from the command line i.e. :

python $< filename >$

or if using an IDE such as PyCharm, then can use the Run/Edit Configurations/parameters field to specify the filename.

1.2 Choosing the type of analysis to be run

There are several sections in these scripts which can be switched on or off by setting a flag. A brief description of each flag is given below. The following flags can be set to True/1:

- EXPLORATORY_ALL_DATA (both datasets)
  - plots a pairplot which shows a variety of ways of visualising the data. This was useful to get an idea of the type of distribution the data might come from.
  - plots a histogram which gives an idea of the separation state of the data. It uses all the data to calculate weights using the Fisher method
  - plots the data points onto a line as an easy way of showing the spread of data
  - plots a ROC curve using data that, from pairplots, looks categorical in nature (uses maximum separation method)
  - plots a ROC curve using the Fisher method
- EXPLORATORY_GAUSSIAN_LOOKING_DATA (drug dataset only) – This section has been included because graphs returned from pairplots showed that some dependent variables looked as of they could come from a Gaussian distribution. From reading about discriminant analysis techniques (linear and Fisher's), it is implied that these techniques assume the inputs are from Gaussian distributions (although Fisher's can be more tolerant of data that isn't). Out of curiosity visualisations were plotted here for Gaussian looking inputs only. This section shows visualisations in the form of:
  - histograms for each class
  - ROC curve
- FISHERS - this section does four main things
  - tests what effect different percentages of train/test fractions have on the outcome by looking at a ROC curve
  - Applies the RBF and quadratic basis functions to the inputs and plots the resulting ROC curve
  - Performs cross validation
  - Modifies the discriminant threshold to see if predictions improve
- LOG_REGRESS (both datasets):
  - uses the gradient descent method for calculating weights for a model
  - uses weights from cross validation stage to make prediction on unseen data
  - models data using RBF basis function
- SVM (from the sklearn library) was included to see if data could be separated with a linear on non-linear kernel

- –

1.3 Python imports

Typical data science modules are required to run the scripts. Mainly numpy, pandas, sklearn and matplotlib. The sys and csv modules are also required as well as the fomlads lbrary used in labs. Additional files required are misc_funcs.py, misc2.py, and model_classification.py is used to house various functions, some which are specific to a particular dataset.

1.4 Modification of the datasets

When the abalone classification script is run, a modified version of its respective dataset is created. Mainly the 'sex' column is dropped and instead two dummy variables are created called 'male' and 'female'. The original 'sex' column contained three nominal variables M,F and I for male, female and infant respectively. The dummy variables have been encoded so that if the gender was originally a male then a 1 is entered in the male column. Likewise if female, then a 1 is place in the female dummy column. Therefore, if the sex was 'infant', both male and female dummy variables are set to zero. Using dummy variables in this way is an example of applying one hot encoding.

According to Brownlee (Brownlee, 2017), some algorithms do not work well with string labelled categorical variables therefore it is worth converting these strings to integers. A couple of techniques used to convert a string label to a number are one hot encoding and integer encoding. If the categorical variable has some kind of order then it is referred to as ordinal, otherwise it is nominal. As gender values do not have an order, one hot encoding is the preferred method. It is preferred to integer encoding i.e. assigning successive numbers to the categories (in one column), because by integer encoding could imply a relationship when there isn't one and this could be misleading when it comes to interpreting the result of the model.

The abalone script also modifies the target column (continuous data) to comply with the condition that the instance value should become -1 if the value is less than 10 and 1, otherwise.

The scripts for both datasets continue by splitting the data into training and test portions where the test portion is left alone until after cross validation has been performed. The whole point of cross validation is to test a model's stability and the ultimate way to be convinced is to hold out data purely for testing i.e. data that has not been used as part of the modelling process. If the cross validation process suggests a stable model I.e. the sample data is able to produce consistent and accurate results, then the test data will use the model weights for ultimate testing of the model. In this project, a stable model has been defined as having a tolerance of 10%. This means that a model will be considered stable if the cross validation results produce a min and max prediction which are within 10% difference. For example, if while doing 5 fold cross validation and get predictions of 80, 81, 75, 84, and 83% then with a min and max prediction of 75 and 84 respectively, this is within a 10% tolerance and the model is accepted. 10% is arbitrary as this value will be different for different applications. For example, in medicine a 10% tolerance for disease prediction might not be good enough.

The train/test split for cross validation chosen was a 60:40 split. There was no particular logic to choosing this percentage split and ideally the authors wanted to generate learning curve graphs which show the accuracy or error produced when using different size training data. This graph could have given an idea as to what a good split would be. However, the scripts do include some analysis of prediction accuracy when try different train/test splits. This analysis can be seen when set the FISHERS flag.

Both radial and quadratic basis functions are applied to both datasets because we wanted to see if the data is better separated with a linear or non-linear hyperplane. With regards to applying basis functions, once the design matrix was calculated, a column of 1's was added so that the projected inputs were of the form

$w^T X$.

The main methods of assessment were calculating the true positive rate and false positive rate so that ROC curves could be plotted. Accuracy is also used by calculating the percentage of the sum of true positives plus true negatives.