2018

CS5014
–
MACHINE LEARNING
–
P1
–
CLASSIFICATION OF OBJECT COLOUR USING OPTICAL SPECTROSCOPY

UNIVERSITY OF ST ANDREWS

STUDENT ID: 170027939

Table of Contents

Introduction

This report details the procedures used in predicting the class of an object based on the given spectrum of the of the object.

The practical therefore aims to help develop knowledge and experience in using and performing several Machine Learning techniques on large sets of data. The following are the list of objectives:

- Loading and Cleaning the data.
- Analyzing and visualizing the data.
- Preparing the inputs and choosing/creating suitable subsets of features from given dataset.
- Selecting and training a classification model.
- Evaluating the performance of the classification model.
- Critical discussion of the results and your approach.

Tasks Achieved

Data Description

The data used in this practical comprised of two sets (binary and multiclass datasets), of optical reflectance spectroscopy data acquired with different colours placed underneath a spectroscopy device (Flame-S-Vis-NIR, Ocean Optics). The binary dataset and multiclass datasets contained training data and test data to be used in Binary and Multiclass classification respectively.

In an attempt to gain better understanding of the datasets, an overview of the sets of data, was made in MS[1] Excel. This was done to also investigate interesting features (i.e. if there were any) that would be of particular interest. These initial observations were made:

- The sets of data contained Wavelength data (Wavelength.csv), a data key (key.txt), Training data for x's and y's (i.e. X.csv and y.csv) and Test Data to be classified (XToClassify.csv)
- The training and test data sets in the binary folder had the same number of features with different number of samples i.e. 920 features, 180 samples for training as well as 20 samples for test. The binary data was to be classified into two colours, i.e. Green or Red, represented by 0 and 1 respectively from the 'key' file.
- The training and test data sets from the multiclass folder had the same number of features as the binary but with 450 and 50 samples of data for Training and testing sets respectively. The multiclass data was to be classified into 5 distinct classes of colours namely Blue, Green, Pink, Red and Yellow, represented by the number 0, 1, 2, 3 and 4 respectively.

[1] Microsoft

Loading and Cleaning the Data

A decision was made, after the previous overviewing process to use the data as it was and not clean it, because negative and zero values were also to be taken as true values, i.e. in optical reflectance spectroscopy, those values could also be valid.

The data, made available as a CSV[2] file, was loaded into the python (script) by using the Pandas[3] Data Analysis Library and stored as a Pandas data-frame. This decision was made for flexibility and to allow for easier data manipulation.

Data Analysis and Visualisation

A regression plot was initially used to visualise that data to help map out trends and potential anomalies within the data set. The training data sets of x was plotted against their respective y's in each type of classification. This was intended to also aid with figuring out and gauging what to expect from a classification model, if applied on the dataset.

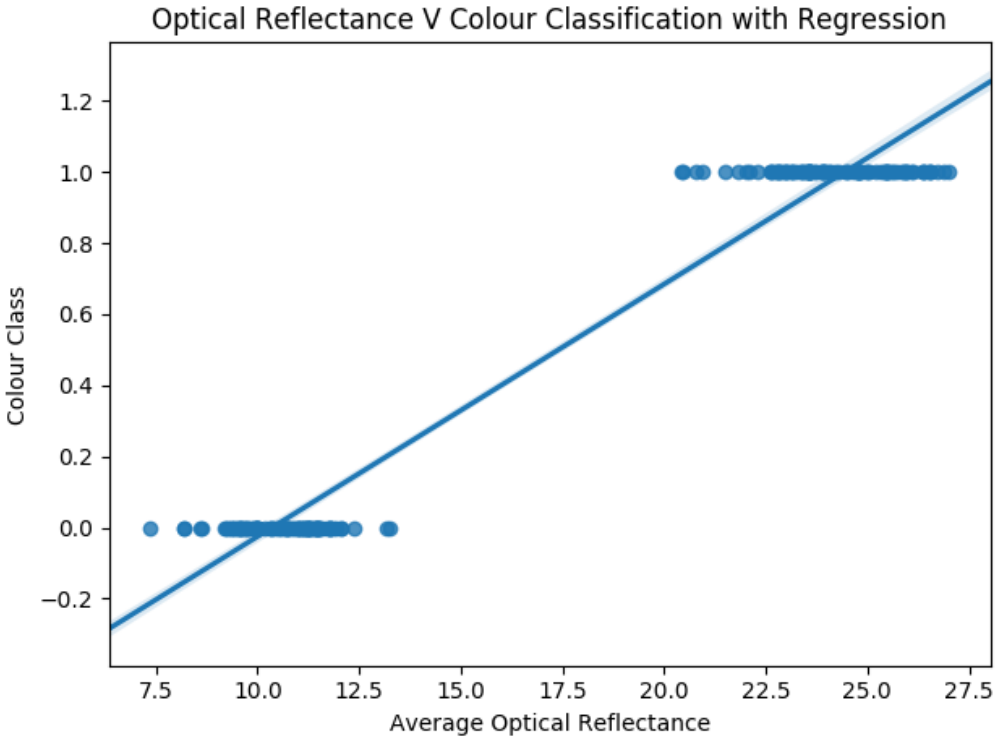


Figure 1 – Binary Classification: Optical Reflectance V Colour Classification

The blue dots in the image of Figure 1 (Binary) represent the average optical reflectance and its colour class while the blue line represents the 'ideal' linear regression model of the data. This also applies for Figure 2 (Multiclass).

[2] Comma-Separated Value
[3] Pandas - https://pandas.pydata.org

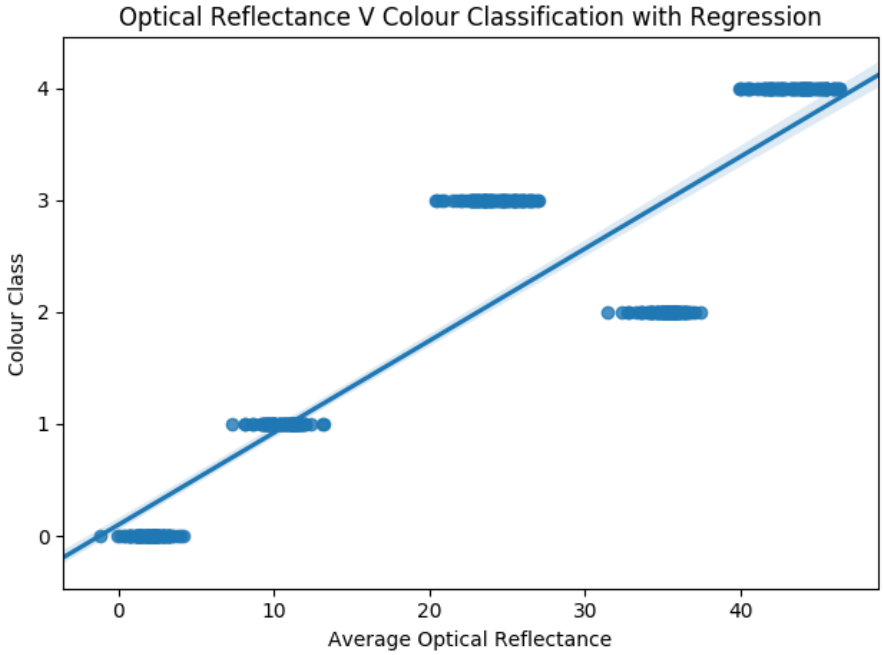


Figure 2 – Multiclass Classification Optical Reflectance V Colour Classification

Observations

- In the image representing the binary regression (Figure 1) the data points were grouped into two parts, with all data points yielding either 0 or 1 on the Colour Class axis. Average optical reflectance for colour class 0 (representing Green) ranged from 0 to 14 where as that of colour class 1 (representing Red) ranged from about 18 to 27.5.

- The image representing the multiclass regression (Figure 2) plotted and mapped the average optical reflectance to 5 distinct numbers i.e. 0, 1, 2, 3 and 4, representing Blue, Green, Pink, Red and Yellow colours respectively. It was noticed that reflectance in the range of 0-5 were mapped to 0, indicating Blue, with a range of 6-15 being Green, 19 – 27 suggesting Pink, and the ranges of 31-39 and 40-49 suggested to be Red and Yellow colours respectively.

Conclusions

- Binary dataset indeed was data to be classified across two distinct labels of 0 and 1 corresponding to colours of Green and Red.
- The two charts suggested a correlation between average optical reflectance and their corresponding classes within certain ranges.
- Multiclass dataset contain data to be classified across 5 distinctive data classes of colours, as explained above.

Preparation of Inputs

This process entailed trial and error process of choosing several attributes and train a potential classification model in the next step. The decision was made to use three sets of data.

- **Data Model 1: Raw/All Reflectance**

The raw data of each type of the data set (binary or multiclass), was used as downloaded and untouched with all 180 samples and 920 Features. This was intended to form a base model from which improvements, if needed, would be made, and would also help serve as a reference point and a bench mark.

This was also chosen over simply averaged the features because it would give the truest representation to what the raw regression with look like. The inputs here for x are all x's; and for the y is all y's

- **Data Model 2: Average Reflectance**

This data model entailed averaging the total reflectance of each sample over the number of feature, i.e. 920, in each sample, essentially reducing it to a 1-dimensional x-data being mapped onto the 1-dimensional y-data. This would, in theory, be quicker to process, but would probably be less accurate in predictions due to lack of detail. Nevertheless, it would show a clear contrast and distinction from the previous data model.

This model was chosen primarily to aid in investigating the effect of having more features, in attempt to gauge a balance between detail and efficiency. The inputs in this data model are average x (over 920); with y's being all y's.

- **Data Model 3: Average and Wavelength**

This model would include and multiple the above result with the wavelength data after averaging, which would serve as an added layer of detail to the averages data.

Selection and Training of Classification Models

Two classification algorithms were chosen to learn and predict the test data sets. The algorithms, both from the 'Sci-Kit Learn'[4] packages in python, were as follows:

Logistic Regression

This algorithm was used to create a Logistic Regression Classifier to learn and class the two data models discussed in the previous part. This was chosen because it, being based on a linear model of regression, used to classify groups of objects, had the potential of giving some simple regression models, which would be easy to understand. This also meant that it would be a good starting point to form a base of comparison.

Neural Network Multi-Layer Perceptron

This, more advanced model than the previous, is based on a neural network model, where the data is learnt and predicted via the use of multi-layer perceptions. This was chosen as

[4] Sci-Kit Learn: http://scikit-learn.org/stable/

the advanced model that was intended to be more efficient, compared to the Logistic model and therefore might be of an interesting contrast.

Results and Charts

The results from test data predictions, in charts have been attached in the appendix as follows:

- **Binary Logistic Regression on Data Model 1 = Appendix 1**
- **Binary Logistic Regression on Data Model 2 = Appendix 2**
- **Binary Neural Network on Data Model 1 = Appendix 3**
- **Binary Neural Network on Data Model 2 = Appendix 4**

- **Multiclass Logistic Regression on Data Model 1 = Appendix 5**

- **Multiclass Logistic Regression on Data Model 2 = Appendix 6**
- **Multiclass Neural Network on Data Model 1 = Appendix 7**
- **Multiclass Neural Network on Data Model 2 = Appendix 8**

Observations

For evaluation purposes, the metrics used were Accuracy Score, Log Loss, Confusion Matrix.

The results from evaluation and observations made, along with conclusions drawn from the evaluation analysis have been detailed below:

Accuracy Score

This metric was used to check if the models had truly learnt their training data. It was used as a means of validation. The accuracy score gives a percentage (of range 0-1), of the percentage of training data learnt.

The results were as follows:

- **Binary Logistic Regression on Data Model 1 = 100%**
- **Binary Logistic Regression on Data Model 2 = 100%**
- **Binary Neural Network on Data Model 1 = 100%**
- **Binary Neural Network on Data Model 2 = 50%**

- **Multiclass Logistic Regression on Data Model 1 = 100%**

- **Multiclass Logistic Regression on Data Model 2 = 100%**
- **Multiclass Neural Network on Data Model 1 = 100%**
- **Multiclass Neural Network on Data Model 2 = 99.8%**

Log Loss

This is the logarithmic loss function of the classifier, which measured and in effect, penalised wrong classifications and hence was useful was a validation metric to gauge the prediction accuracy of each of the models.

- **Binary Logistic Regression on Data Model 1 = 0.048%**
- **Binary Logistic Regression on Data Model 2 = 1.944%**
- **Binary Neural Network on Data Model 1 = 99.9e-14%**
- **Binary Neural Network on Data Model 2 = 62.5%**

- **Multiclass Logistic Regression on Data Model 1 = 18.3 e-4%**

- **Multiclass Logistic Regression on Data Model 2 = 0.89%**
- **Multiclass Neural Network on Data Model 1 = 3.01e-9%**
- **Multiclass Neural Network on Data Model 2 = 41%**

Confusion Matrix

The confusion matrix was used to judge how well, each model distinguished between its respective potential class. Fundamentally, the question, the confusion matrix attempts to answer is, given an actual instance e.g. green, how many greens did the model 'mistake' or confuse for the other colours.

Computing this type of metric was useful because it shed some light on classes which may have been interpreted as similar, or may've had similar features, to the model.

- **Binary**

All 4 confusion matrices for the four binary classification models were "perfect", in the sense that out of 90 instances of either targets, 0 or 1, none was confused or mistaken for the other. The output image below, Fig 3, was the same for all binary models.

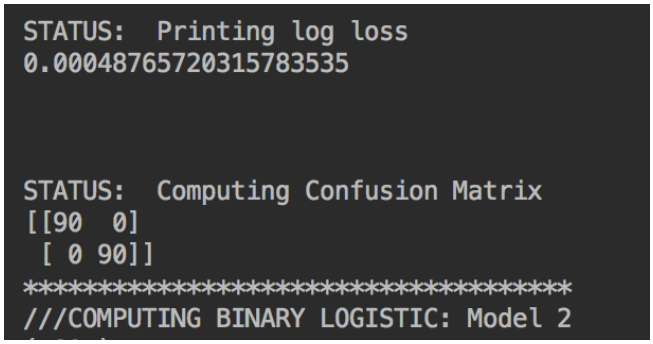


Figure 3 – Confusion Matrix of Binary Logistic Regression on Data Model 1

NOTE: Rows of confusion matrix represent Actual classes, while columns represent Predicted classes.

- **Multiclass**

In the multiclass, the story was a little different. The output confusion matrix results for all but the neural network model on data model 2 was perfect.

For the first 3 multiclass classification models, (i.e. multi-logistic data1, multi-logistic data 2 and multi-neural data 1), there was no confusion reported. Out on 90 instances, all 3 methods predicted the right class.

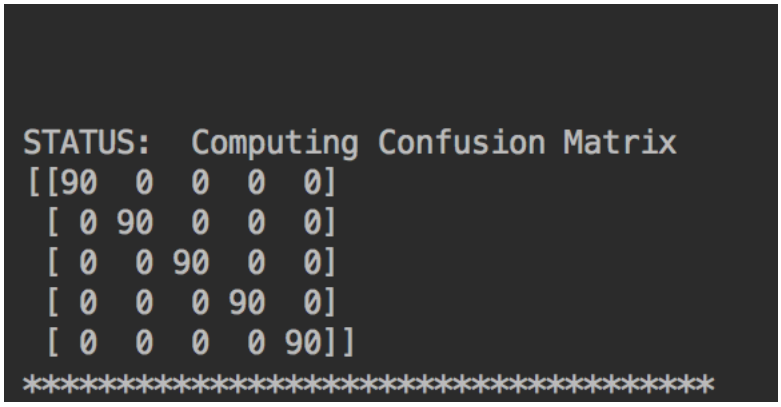


Figure 4 – Confusion Matrix of Multiclass Logistic Regression on Data Model 1

However, the final model, i.e. multi-class neural net with data 2, got confused, as Figure 5 would show, and mistook an instance of 'Pink' represented by 2, for 'Yellow' represented by the number 4. The output images have been attached below as Figures 4 and 5

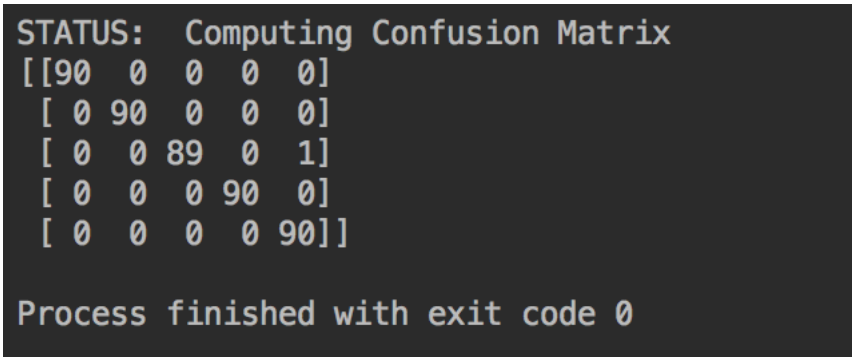


Figure 5 – Confusion Matrix of Multiclass Neural Network on Data Model 2

NOTE: Rows of confusion matrix represent Actual classes, while columns represent Predicted classes.

Discussions

After the several observations made, as discussed in the previous tasks, the results were reflected upon and conclusions were made as a result. The following conclusions were drawn.

- Comparing the outputs form the metrics above, i.e. binary and multiclass algorithms, it appears that the logistic regression algorithm was much more 'accurate' at predicting classes, than the neural network, at least on average. This is due to only 2 of the neural networks reporting an accuracy score of less than 100% compared to all 4 logistic models posting accuracies on 100% on validation.
  This could be due the lack of use of optimisation techniques to enhance the neural models but at the very least, and on somewhat 'equal' grounds, the results seem to indicate that logistic regression models would yield better accuracy on validation tests than neural network classifiers.
  A more interesting observation, however, is that of the accuracy score, when using the 'averages' data, compared with the actual data sets on neural networks; which suggests that this would be a very bad idea because, it yielded as low as 50% accuracy.

- The metric of the log loss had a similar result to the above, where I essentially confirmed the idea that the neural based algorithms performed worse than the linear logistic algorithms. With losses of as high as 41% and 62.5 %, the neural based models posted the worst log losses. This could potentially be due to either, the "manipulation of averages" dataset, not containing as much detail as the 'full' dataset, OR, the neural network models not optimised well enough, or a combination of both.
  The above is a sensible enough assumption because, learning only an approximation of the data (i.e. using averages as the learning data), could actually mark and hide important local minimums, which would've otherwise been visible to the prediction algorithm. Approximating the dataset in search of 'processing speed', seems to prove to be a costly method as it may seem as though the model struggles to predict accurately.

Evaluation

The dataset proved to be very interesting. This practical gave insights into how algorithms work with data, particularly in machine learning.

It was made clear that the data, regardless of how inconclusive and irrelevant it may seem, is always very important when attempting to train a model. Manipulating the data in such a way, may've been very drastic in the sense that 920 features were 'crunched' into 1. This is believed to be the main issue as to why the two algorithms (i.e. logistic and neural) both struggled somewhat with prediction in the results.

Certainly, the intention was to test the effect of data manipulation within machine learning algorithms and, in that context, this result did not disappoint as it portrayed the contrasting results between the two approaches.

A better way of handling this, would be to use an optimised advanced algorithm such as the neural network, with an appropriate number of hidden layers and the right biases and weight terms at each layer/node, learning on the full set of data, and using it to predict the outcomes of the test data.

Other short-comings were experienced as well, one of which was, not attempting to do use more metrics for more in-depth analysis. The issue here was that, I kept running into bugs and errors when trying to use certain advanced metrics because of conversion issues from "Numpy" arrays to "Pandas" data frames. A more simplistic approach was chosen in favour of a compilation of advanced metrics for this reason. More familiarity with the data structures and their manipulations would be helpful as it would yield more interesting results.

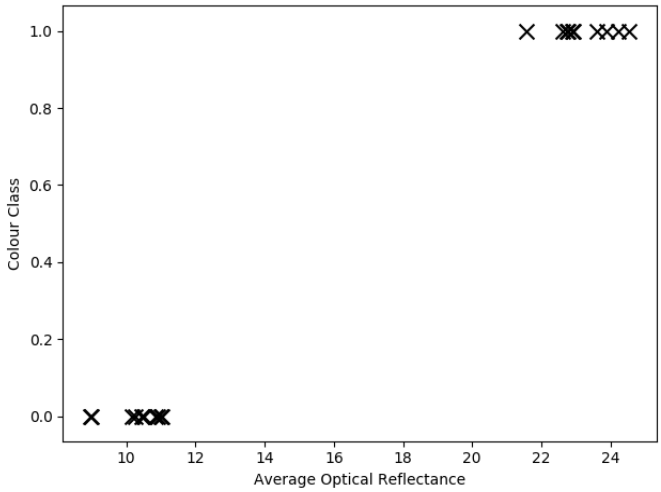
Another choice which was made, due to certain time constrains and the wavelength data being particularly continuous, was that, the third data model (i.e. multiplying the averages from x's with the corresponding wavelength data) was not implemented. This could've improved on the somewhat inaccurate model which were produced by using the second data model.

Conclusion

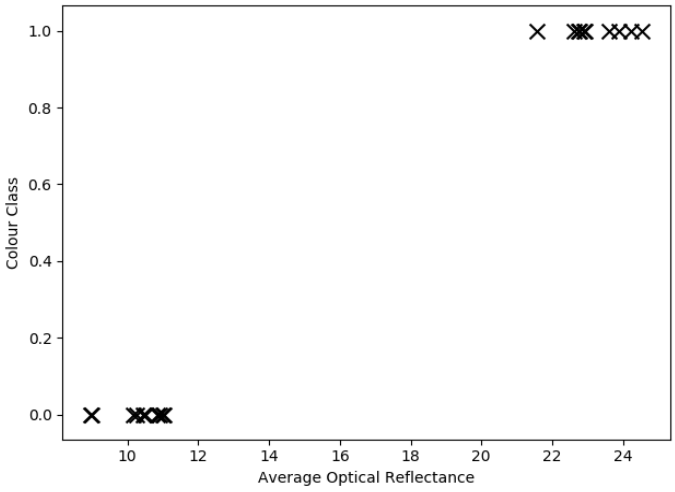
This was an interesting practical, as it had elements of both supervised and unsupervised learning implementations integrated in it. It served as a practical means of education as it enlightened me of certain practices to avoid when using 'unknown' data in supervised learning particularly.

Appendix

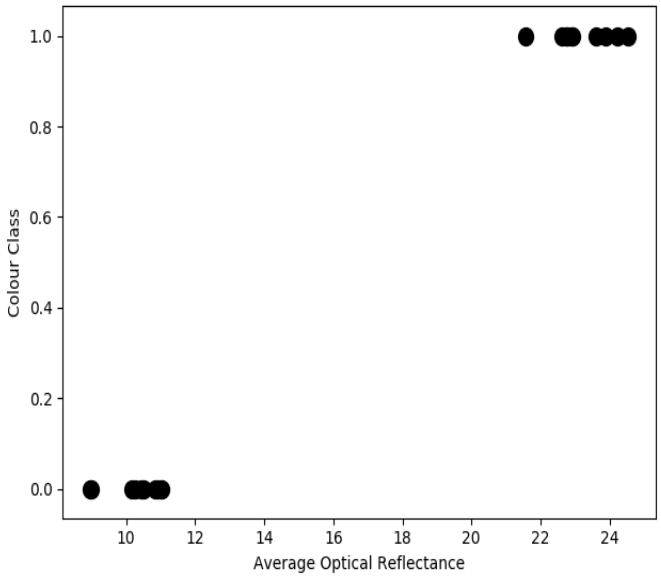
Appendix 1:
Binary Logistic Regression on Data Model 1



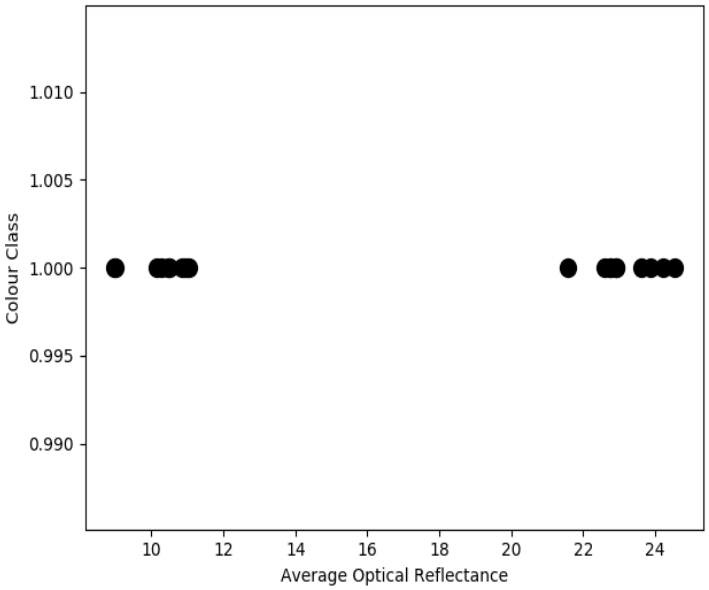
Appendix 2:
Binary Logistic Regression on Data Model 2



Appendix 3:
Binary Neural Network on Data Model 1

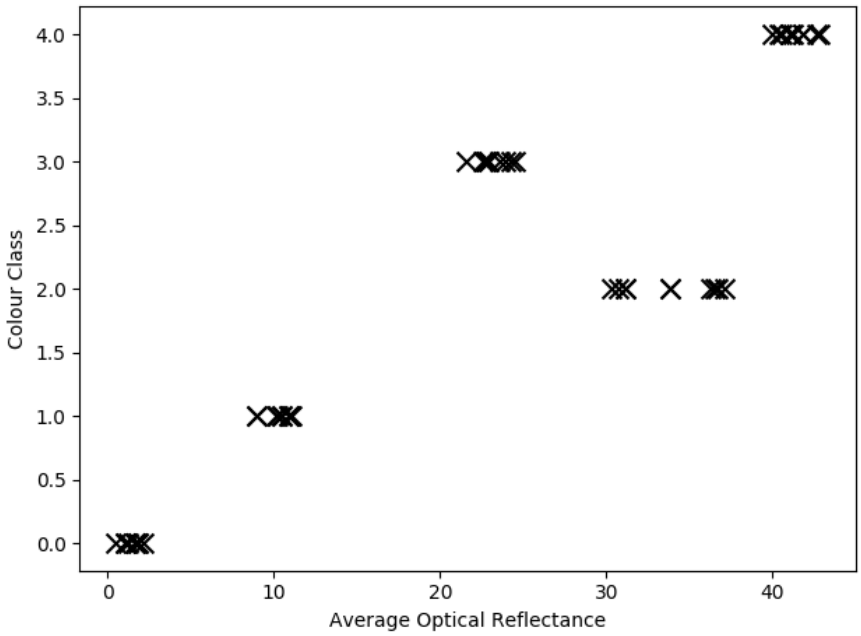


Appendix 4:
Binary Neural Network on Data Model 2



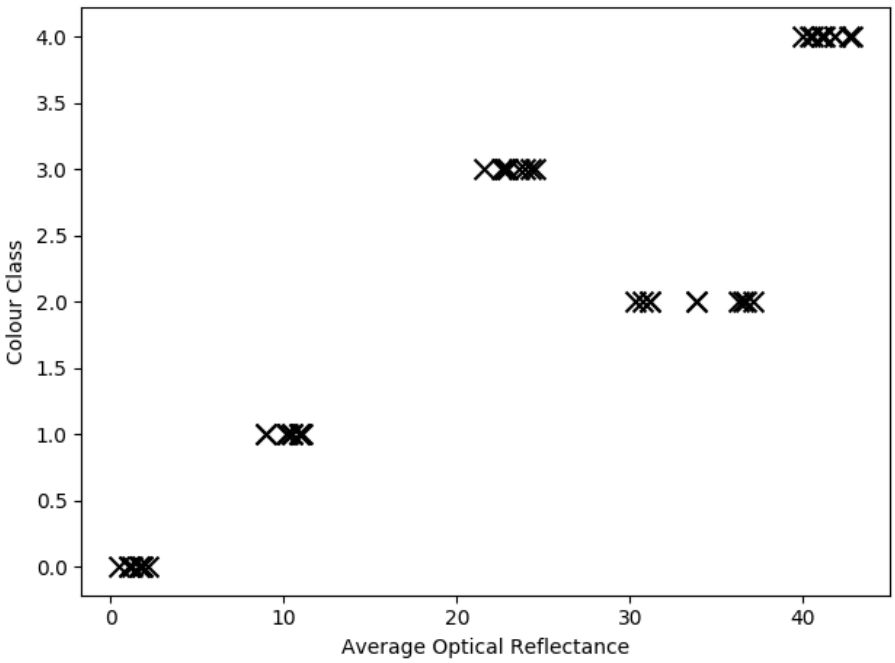
Appendix 5:

Multiclass Logistic Regression on Data Model 1 = Appendix 5

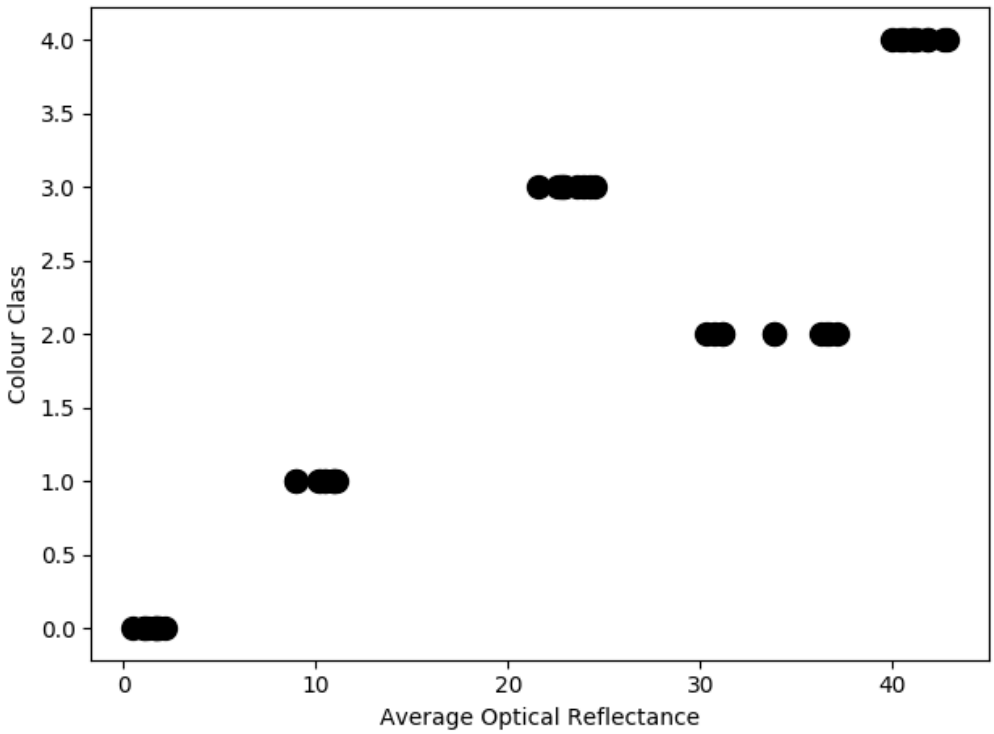


Appendix 6:

Multiclass Logistic Regression on Data Model 2



Appendix 7:
Multiclass Neural Network on Data Model 1



Appendix 8:
Multiclass Neural Network on Data Model 2