P2 - CLASSIFICATION OF SEAL IMAGES

5th May 2020

190020774
University of St. Andrews
CS5014 Machine Learning

Contents

1 Introduction

This practical is concerned with using Machine Learning (ML) techniques in order to classify images of seal pups in the North Sea. As the nature of the practical suggests, this requires the use of logistic regression in order to place the images into their respective categories.

2 Running Instructions

The file names should be self-explanatory. They represent two different models for both a binary and multiclass case, for a total of 4 files.

In order to ensure that the data gets read in correctly, the training and testing data must be in a separate directory above the directory where the code is located. Consider the following as an example:

```
X_in = pd.read_csv('../data/binary/X_train.csv', dtype='float64')
```

Obviously if you want to move these directories somewhere else you can, but the lines that read in data must be changed accordingly to reflect this.

3 End-to-end Machine Learning Project

3.1 Look at the Big Picture

The first task that must be performed when embarking on a machine learning project is to look at the overall problem and establish the goals to achieve from the project before moving forward.

3.1.1 Frame the Problem

The objective of this model is to classify images taken from the North Sea which may or may not contain seal pups. For the binary case, we are only concerned with whether or not a seal is contained within the image. For the multiclass classification, the seals can be in one of 4 categories: [whitetail, juvenile, dead pip, moulted pup].
There is technically no current solution to this problem, except by manually classifying the data by hand - and as there are thousands of unclassified images this is a very inefficient solution.

3.1.2 Select a Performance Measure

The next step is to select a suitable performance measure for the classification. As this is a classification task the most suitable measure is the *accuracy*.

- FP = Number of false positives
- FN = Number of false negatives
- P = Total instances of class 1
- N = Total instances of class 2

$$\text{Accuracy} = \text{Classification Rate} = \frac{TP+TN}{P+N}$$

We can also define a number of various others performance measures in classification:

- Precision = $\frac{TP}{TP+FP}$
- Recall = $\frac{TP}{TP+FN}$
- F1-score = $2 * \frac{Precision*Recall}{Precision+Recall}$

Now that we have framed the problem, we are ready to begin designing our system.

3.2 Get the Data

The data for this problem was already provided to us via StudRes, so thankfully no effort was required in obtaining this data.
For each of the two problems (binary and multiclass), we are provided with three data sets:

- `X_train.csv`
- `Y_train.csv`
- `X_test.csv`

Where `X_train` contains features from each image and `Y_train` contains the corresponding image classes.
As the data has already been split into training and testing data, thankfully there is no need to create a separate test set.

3.2.1 Check the size and type of data

The training data is an extremely large dataset, with 62209 samples and 964 features. In `X_train`, the data is split into three segments. The first 900 columns correspond to a histogram of orientated gradients, the next 16 are drawn from a normal distribution and the final 48 columns correspond to colour histograms. The data does not include any sensitive information, so we can include it all for now.

3.3 Explore the Data to Gain Insights

Once the data has been split into training and testing sets, it is important to visualise the data in order to see if we can gain any insights into the type of regression model used.

3.3.1 Visualise the Data

With nearly 1000 input features, it would be exhaustive and unnecessary to plot each of these against the outputs individually. Therefore, a suitable starting point for visualisation are the mean of the gradients and colours respectively.
The figure below shows the average gradient value for each image plotted against their respective classes.

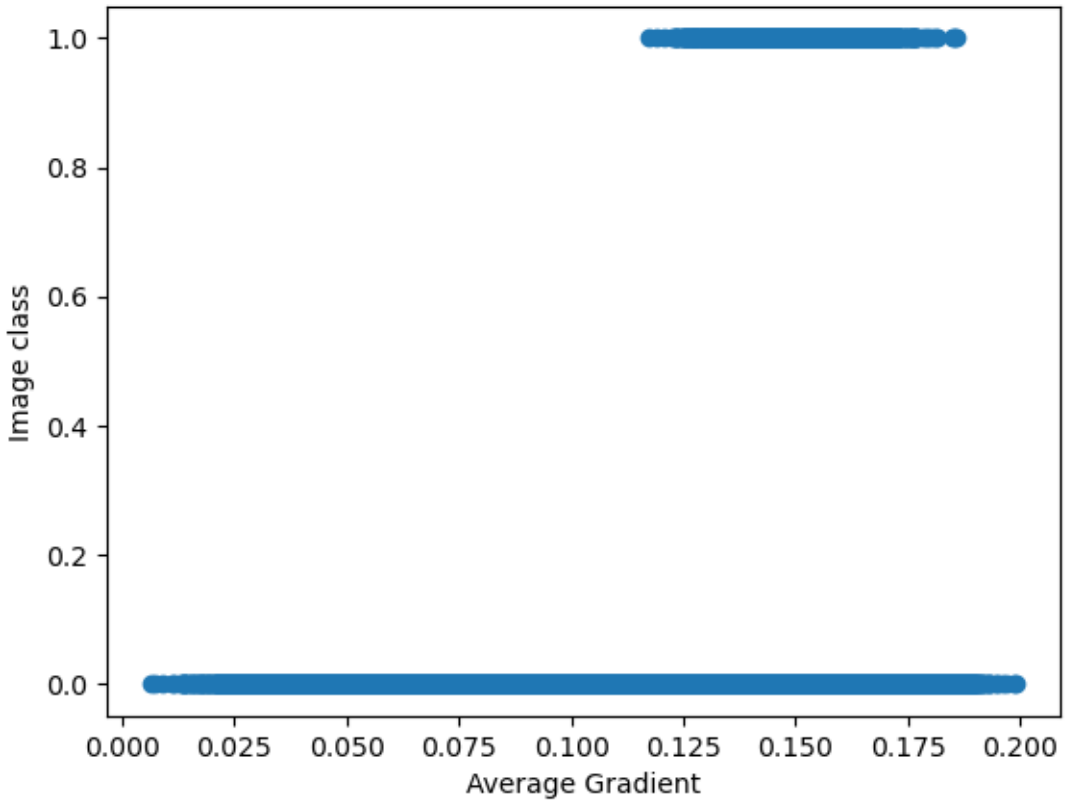

Figure 1: Image class vs Average Gradient

From this figure, it is difficult to see any clear decision boundary between the two classes based on the average gradient alone. Furthermore, when plotting the average colour, we find that the average of all colours for each image is the same value - 225, therefore we cannot gain any real insights from this plot.

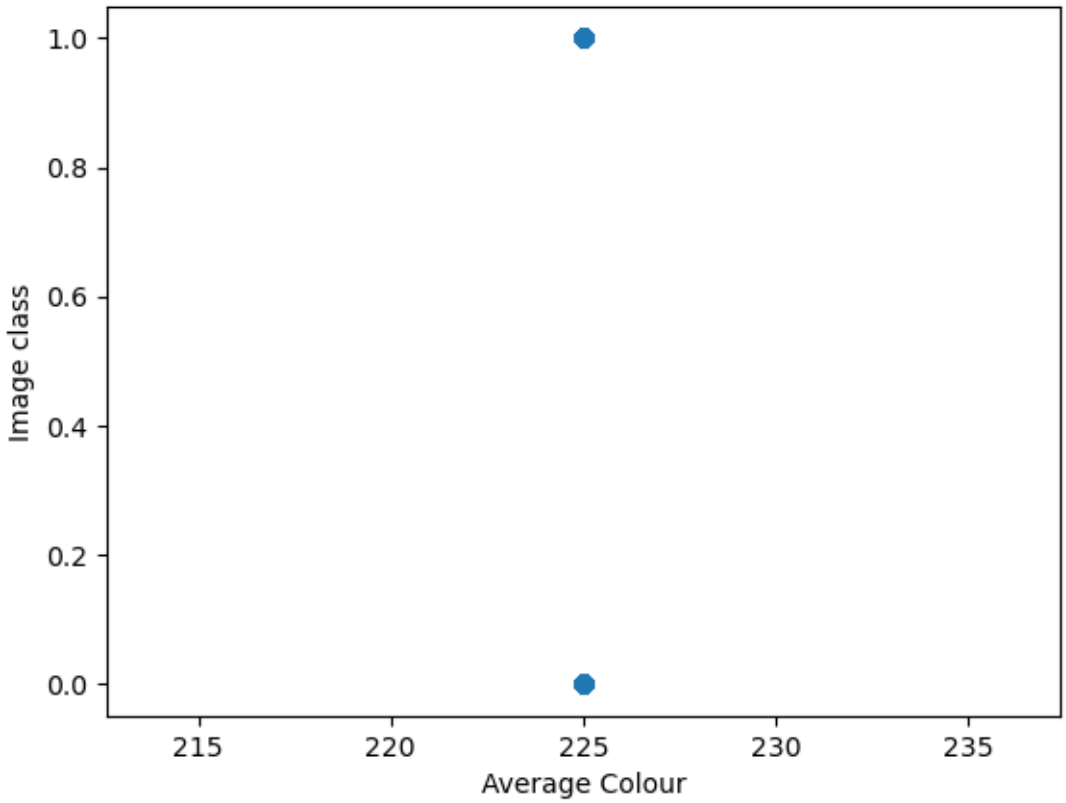

Figure 2: Image class vs Average Colour (Binary)

This is the same case for the multi class classification as seen below.

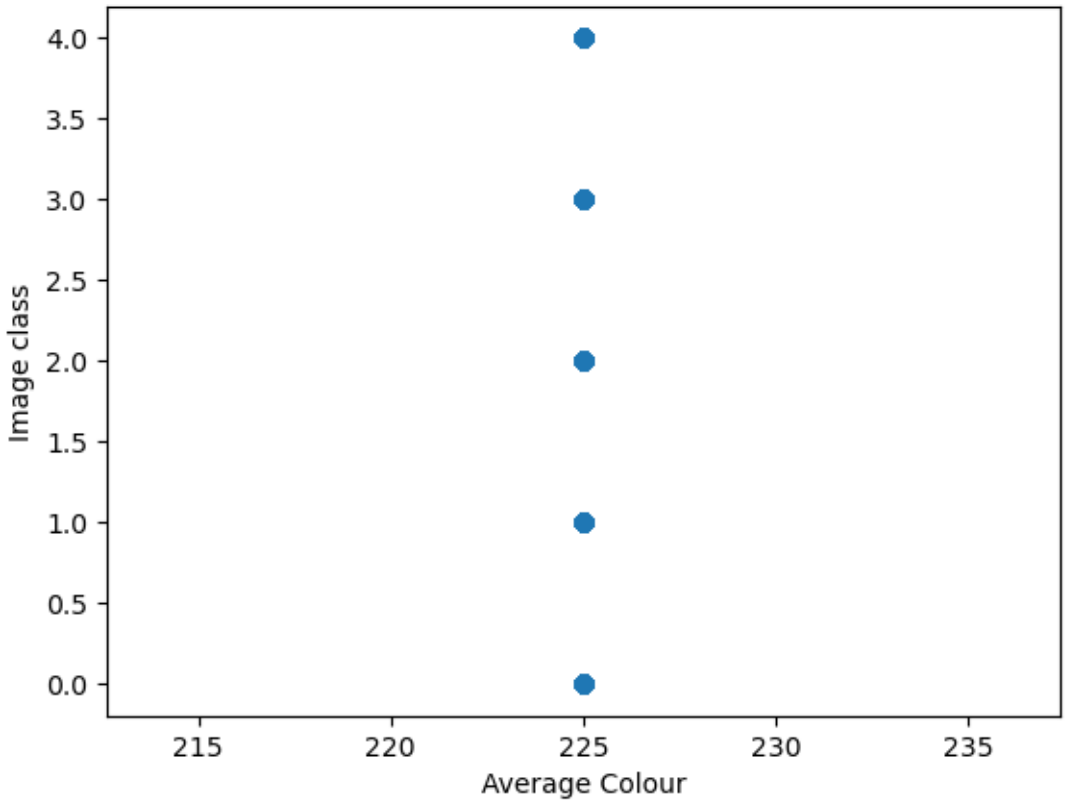

Figure 3: Image class vs Average Colour (Multi)

3.4 Prepare the Data

Once we have visualised the data, it is important to prepare the data for use in a machine learning application. This step is important as it removes any outliers and formats the data in the best way to feed into an algorithm.

3.4.1 Data Cleaning

Data cleaning is the process of ensuring that all the data is in the right format. Firstly, we can remove columns 900 - 916 of the input features. This is because all these values are drawn from a normal distribution and no further explanation is given as to why this data is included. Therefore, we can drop this data from the training set.

3.4.2 Feature Subset Selection

Since there are such a large amount of features, it is necessary to select a subset of all the features for two reasons. Firstly, some of the features may not have any correlation with the output. And secondly, reducing the number of features reduces the computation time for our machine learning model. Also, using a large number of features may lead to overfitting, meaning our model will not adapt well to unseen data. Feature selection is performed using sklearn's `SelectKBest`, for convenience, I have chosen the best 100 features.

3.4.3 Scaling the Data

Many machine learning algorithms do not perform well on data with large ranges, therefore before running our model the data needs to be scaled in a suitable manner. There are two ways to approach this scaling - normalisation and standardisation.

Normalisation shifts the data so that it is in the range [0, 1]. This is performed by the following equation for an input feature X:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Standardisation scales the data such that it is centered around 0, and as such is much less affected by outliers. Standardisation is computed in the following manner:

$$X' = \frac{X - \mu}{\sigma}$$

Where μ, σ are the mean and standard deviation of the original data set, respectively. Standardisation is much less effected by outliers in the data, so we will use this method of scaling

3.5 Explore Different Models

Since this is a classification task requiring two different classification models, I have chosen support vector machines and artificial neural networks since these perform well and are very common in classification tasks.

3.5.1 Support Vector Machines

Support vector machines are a maximal margin classifier which attempts to find the maximum decision boundary between classes. This model was created using Sklearn's `svm` class.

3.5.2 Artificial Neural Networks

The second model used was an artificial neural network. This was performed using Sklearn's `MLPClassifier` with a logistic activation function using stochastic gradient descent.

3.5.3 Comparison of Models

I attempted to experiment with hyperparameters for both support vector machines and artificial neural networks. However since training the models takes a long time this was not possible to explore in detail.

Once the models were trained, the testing data was fed into the model and an output was produced for the unknown images. This was then compared to the actual outputs on the leaderboard. The accuracies are given below:

Model	Type	Accuracy
SVM	Binary	0.92742
SVM	Multi	0.74856
ANN	Binary	0.93179
ANN	Multi	0.90081

3.6 Evaluation

We can also evaluate the two models based on their regular accuracy, balanced accuracy and recall score. I unfortunately did not have time to collect the scores for SVM, and ANN binary task, however the scores for the multiclass neural network model are shown below:

Regular Accuracy	Balanced Accuracy	Recall
0.89421	0.26887	[0.99061214 0.33353414 0, 0, 0.02024648]

4 Conclusion

After running both support vector machines and artificial neural network models on binary and multiclass classification problems, I have chosen ANNs to be the more suitable model since it has the higher accuracy between the two models, and has a much lower runtime.