**Question 2: Jamey Blackman and Bradley De Domizio**

**The Dataset:**

The shuttle dataset is contained in the ‘satimage’ folder; it is already split into training and testing datasets, sat.trn and sat.tst respectively. Original images cannot be reconstructed due to how the data has been given; random lines removed and out of order. This denies any image-based confirmation of accuracy of evaluation and predictions.

It was decided that the class 6 of the dataset, mixture class (all types present), was to be removed as a target class due to its absence in the dataset and ambiguity of its validity as a distinct class. Class 7 (very damp grey soil) was re-enumerated as class 6 and all instances of 7 in the target column of the loaded dataset were replaced with a 6.

**Method:**

**Spectral Band Grouping ANN**

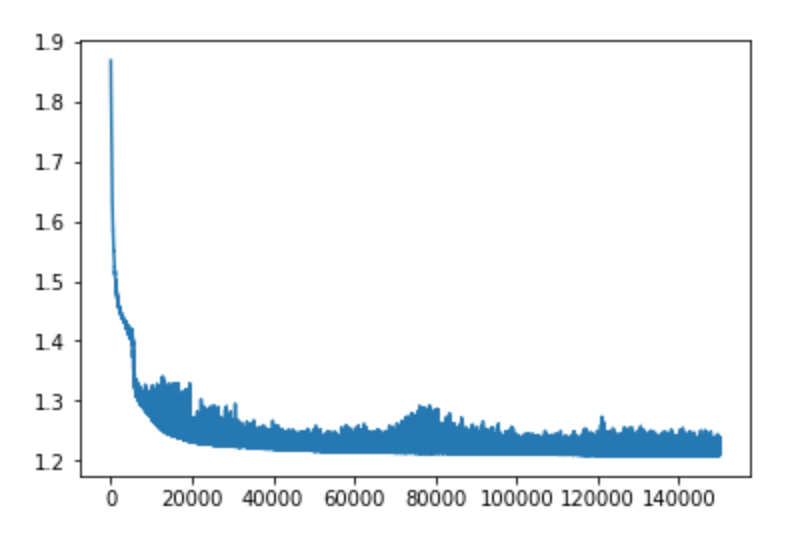
**Pixel Groupings ANN:**

The first attempt at an ANN model can be found in the jupyter notebook file called ‘Q2, ANN-NonKeras.ipynb’. A standard ANN was implemented with tensorflow modules, classes and methods that didn’t involve keras. The number of layers were chosen arbitrarily and the number of nodes for each were chosen to gradually descend. Every one of the 36 attributes were taken as input.

The reproducibility of this model was tedious in regard to initially setting up the layers and training to produce this volume of epochs requires considerable time with no inherent checkpoint saving of a trained model outside of saving the whole model.

(Limits in computing power also affected reproducibility of model)

**Error curve of Pixel Groupings ANN, Non-Keras (below)**

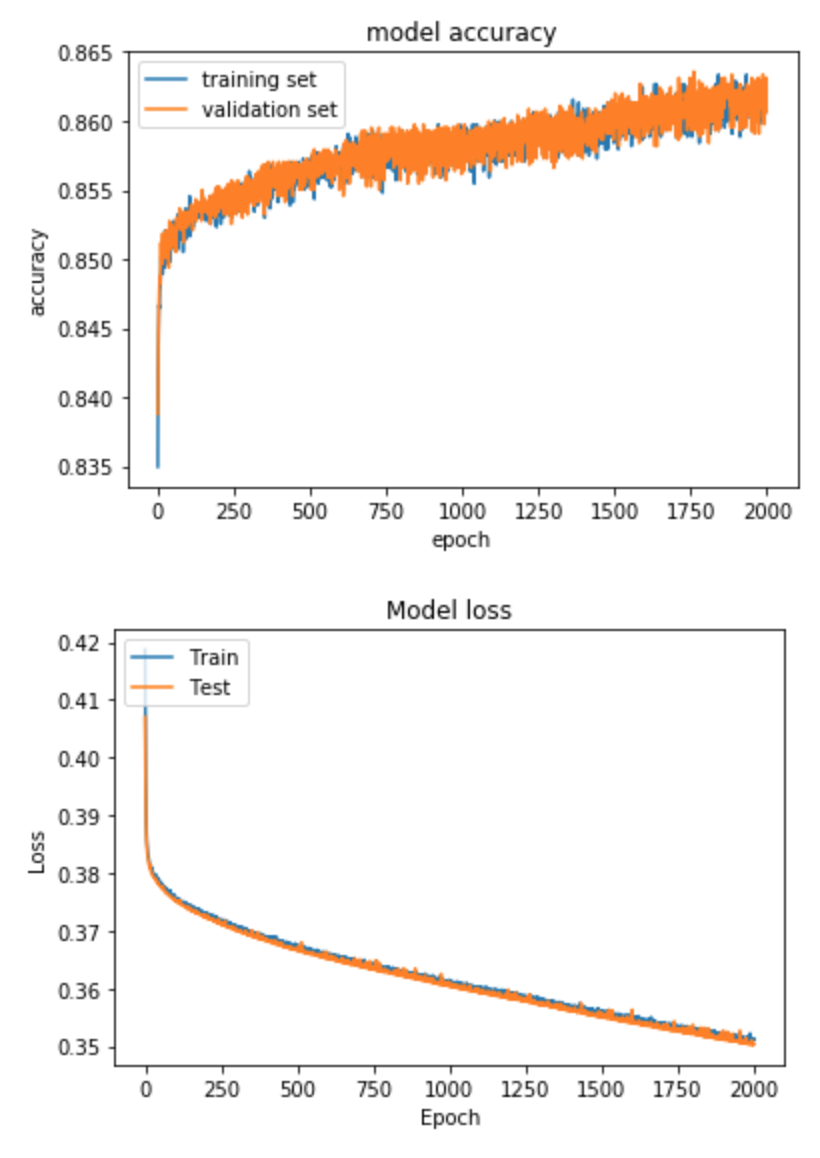


Before attempting to expand experimentation (introducing validation splitting, adding momentum, adding evaluation, adding prediction, etc) this ANN was abandoned and a new ANN model was created which included Keras wrapper modules.

The final ANN model used to produce the results below is found in the jupyter notebook file called ‘Q2, ANN-Keras.ipynb’. It can be loaded by running the module that loads the model named 'fullModel\_v1.h5'. Using keras models allowed for a simpler set up. The dataset was loaded and prepared the same way, with the addition of one hot encoding of the target values. The network architecture contained three hidden layers, and for all layers each layer was regularly densely connected, utilised a rectified linear unit activation function except for the output layer which utilised a softmax activation, and each layer decreased gradually from 25 nodes to 7 at output.

A validation split was used, 70% of train data was used for training, 30% was used for validation. With a SGD optimiser the learning rate was set to 0.001 with no momentum. Loss was set to categorical cross entropy to allow for multi-class targeting. The set amount of training epochs was set to 2000 but early stopping callbacks were put in place that monitored val\_loss was set to a patience of 50 until the accuracy started to pass 0.8 accuracy. Loss and Val\_loss were manually monitored during each fitting run to check if they ever started to increase while accuracy increased which would indicate a divergence or overfitting. Once 0.8+ accuracy was obtained the patience was set to 20 and the learning rate was set to 0.0001. The training and validation accuracy reached a maximum at around 0.83-0.84, with losses fluctuating around 0.4. Momentum was then implemented in the optimiser at 0.9 to attempt to cross this threshold. Alone this did not succeed but by additionally lowering the learning again to 0.00001 the accuracies increased to 0.86 and the losses decreased to 0.35.

**Error curve of Pixel Groupings ANN, Keras, final training run (below)**



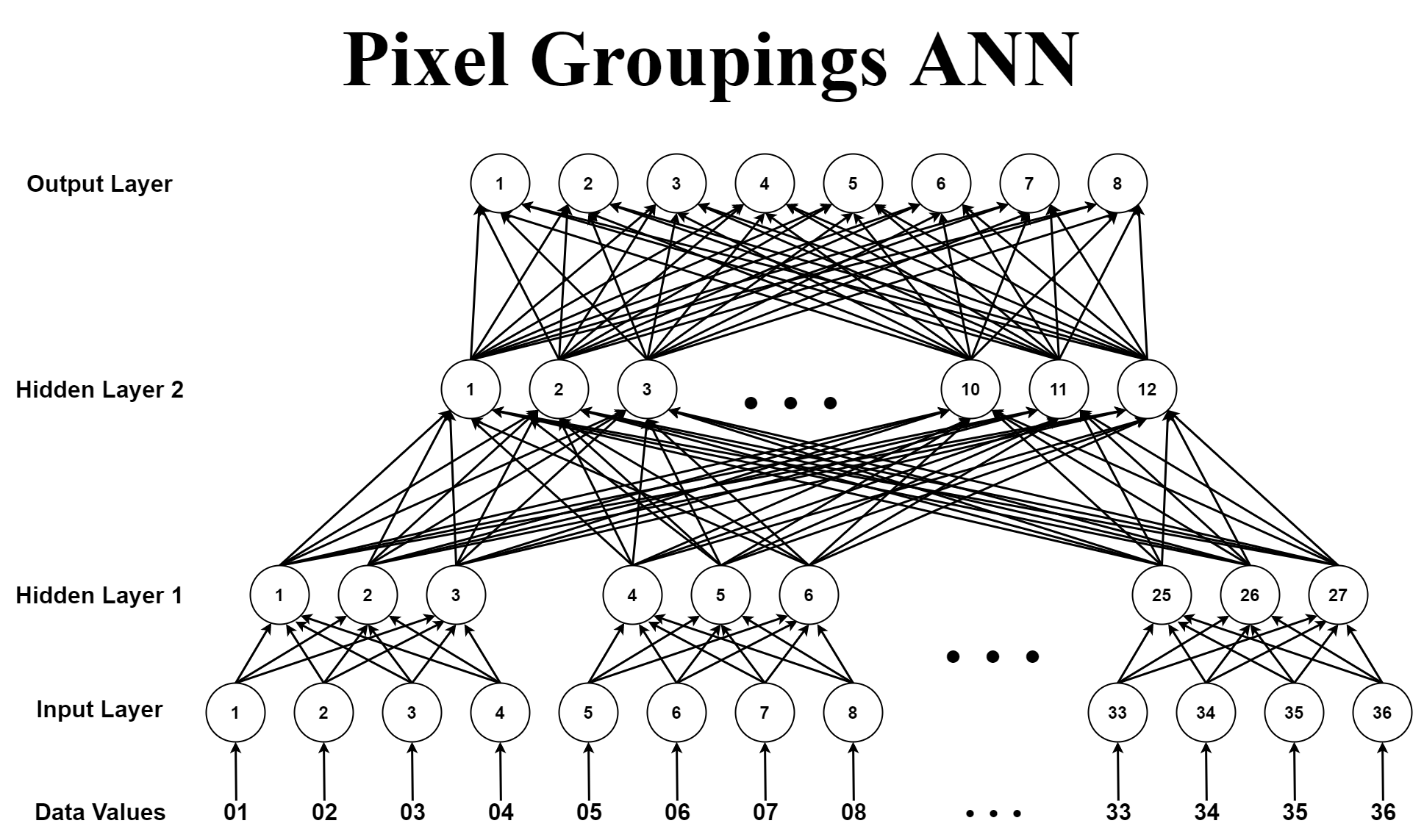
To evaluate, the test dataset file was loaded and prepared the same way as the train dataset file. Test loss returned as 0.463 and test accuracy was 0.818.

The folder contains three .png files displaying confusion matrices, one that evaluates predictions made with the training dataset, and two that evaluate predictions made with the test dataset with one of those being normalised. The results of the confusion matrices show a high percentage of predictions were correct for decision classes 1, 2, 3 and 5. Class 4 suffered a high percentage of incorrect predictions. The trained model determines most instances of Class 4 to be either Class 3 or Class 6. Class 6 has a similar dilemma with incorrect Class 3 predictions yet not nearly to the same degree as Class 4 prediction incorrectness. When considering the prediction phenomena these classes are a part of and what the classes physically represent, we could perhaps induce that these classes’ corresponding soil image are too similar in grey colour and/or the differing dampness of the soil does not produce a light-reflection that considerably differentiates the soil images from each other.

**Alternative ANNs**

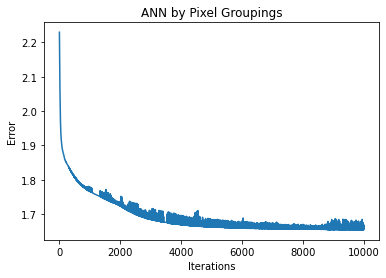
To further our investigation into the use of ANNs to categorize these samples, several atypical structures of ANNs were produced to investigate whether other methods could prove more fruitful.. This was undertaken through the use of separating data into meaningful chunks, and using these chunks to fuel smaller subnetworks, which decisions were then fed into a larger network for determining. Though unusual, these structures were designed to also test whether smaller classification could help determine the larger, overarching output goal. There were two categories explored, both completed using Tensorflow 2.1 in Tensorflow 1.x Compatibility mode:

The First Category’s goal was to determine via having several feed-forward subnetworks determine each pixel first (see Q2\_PixelGroupings.ipynb for Jupyter Notebooks Project), and then feed into a hidden layer before joining the larger network. The structure is shown in the diagram below, with there being 9 subnetworks, each classifying based on spectral values.

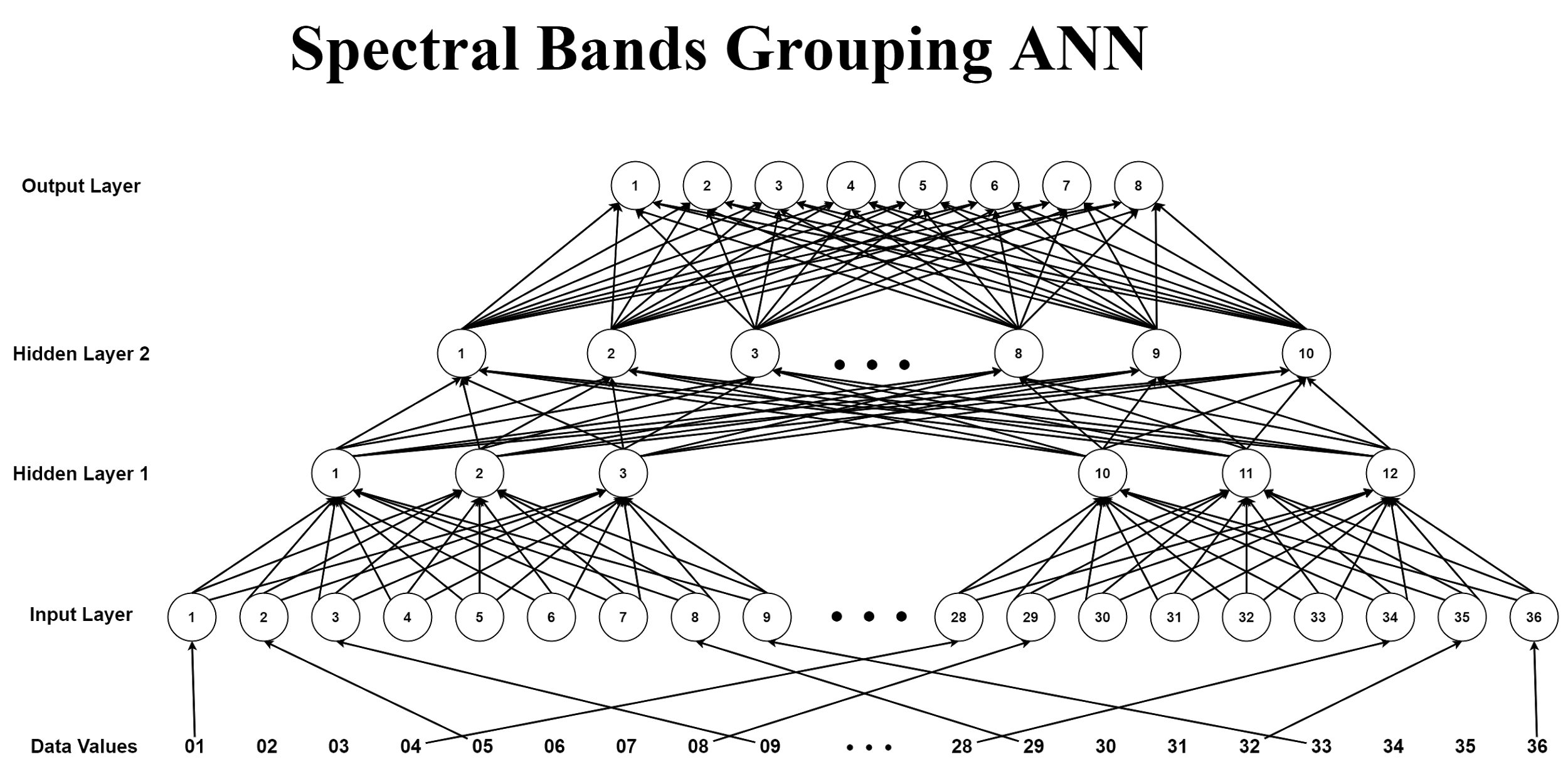


The data values directly correlate to the Input Layer, with Hidden Layer 1 separated based on Pixel Groupings. Hidden Layer 2 deals with combining these pixel layers, and the output layer has 8 values (-1 for data classifier, with 1 being a NULL value as a control).

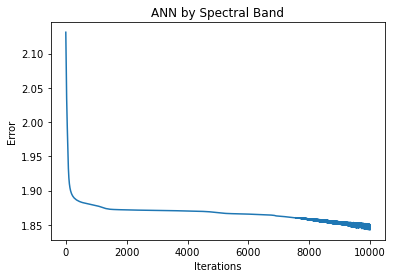
With 10,000 Epochs, and a learning rate of 0.4, this resulted with an accuracy of 33%, with the Error plot shown below



The second Category (Q2\_SpectralBandsGroupings.ipynb) focused instead on whether the spectral bands individually could help determine the decision classes. Each 4th data value corresponds to a spectral value, the offset determining which spectral group it belongs to. This spectral group was given its own Neural Network, that then fed into a larger network, as per the common structure explored in this alternative investigation.



With a learning rate of 0.3 and 10000 iterations, this gave an accuracy of 22.9% on the test data, with the error output shown in the graph below.



As these networks both produced results that were unsatisfactory, being inaccurate along with the error curve plateauing relatively early, these models were not explored further.

**Results:**

In binary classification the measures of true positives, true negatives, false positives, false negatives are codependent and have a “total” force. In classification with more than two classes the measures are considered from the perspective of each individual class. Alternatively the confusion matrix that these measures are calculated from can be collapsed into a 2x2 matrix and regular binary measures can be made again. The prior method was used.

With any given class C, TP is defined as all true instances of C that are predicted as C, FP is defined as all true instances of non-C that are predicted as C, FN is defined as all true instances of C that are predicted as non-C, and TN is defined as all true instances of non-C that are predicted as non-C (these happen to be the correctly predicted true instances of each other class but this information is not relevant to the pertaining class).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Pixel Groupings ANN - Training | | | | | | |
| Target Classes | 1: red soil | 2: cotton crop | 3: grey soil | 4: damp grey soil | 5: soil with vegetation stubble | 6: very damp grey soil |
| Accuracy | 99.30% | 99.95% | 90.58% | 90.36% | 98.71% | 91.53% |
| Overall Accuracy | 89.37% | | Average Accuracy | 95.07% | | |
| Precision | 0.989690721649485 | 0.997912317327766 | 0.72741935483871 | 0.545454545454545 | 0.948717948717949 | 0.813357731015554 |
| Specificity | 0.996031746031746 | 0.999700598802395 | 0.896095911466339 | 0.989498556051457 | 0.992934942596409 | 0.934865900383142 |
| Sensitivity  (Recall) | 0.985074626865672 | 0.997912317327766 | 0.942528735632184 | 0.11566265060241 | 0.94468085106383 | 0.85645472061657 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Pixel Groupings ANN - Test | | | | | | |
| Target Classes | 1: red soil | 2: cotton crop | 3: grey soil | 4: damp grey soil | 5: soil with vegetation stubble | 6: very damp grey soil |
| Accuracy | 99.09% | 99.33% | 88.19% | 94.02% | 96.69% | 88.00% |
| Overall Accuracy | 81.80% | | Average Accuracy | 94.22% | | |
| Precision | 0.980603448275862 | 0.977578475336323 | 0.658928571428571 | 0.64 | 0.905829596412556 | 0.744554455445545 |
| Specificity | 0.992436974789916 | 0.996486296556571 | 0.868998628257888 | 0.994475138121547 | 0.985567010309278 | 0.907127429805616 |
| Sensitivity  (Recall) | 0.986984815618221 | 0.973214285714286 | 0.929471032745592 | 0.075829383886256 | 0.852320675105485 | 0.8 |

Classes 1, 2, and 5 all mostly maintain above 0.90 in almost all their measures, ensuring the models correctness in these class predictions. Class 3 has high specificity and sensitivity, but a lower precision which limits the model to being trustworthy only on negative predictions. Class 4 has terrible sensitivity and low precision which has no trustworthiness or even converse use in the models predictions. Class 6 has slightly lower precision which could potentially limit the model to being trustworthy on negative predictions.