**IEEE CIS/SMC Challenge**

Wen Dong #110057395 dong23@uwindsor.ca

**Motivation**

I have selected artificial neural network to solve the multi-label and multiclass classification challenge, the selection was due to the requirement of the Challenge and the nature of the dataset.

**Algorithm**

The algorithm employed in the solution is an artificial neural network with multiple hidden layers and a softmax output layer with 10 units. According to my analysis on the dataset, the multi-label problem can be safely converted to a single-label classification task as the two labels are highly correlated:

1. Label 1 is of 10 classes [1,2,3,4,5,6,7,8,9,10]
2. Label 2 is a binary [0,1]
3. Whenever label1 is 7 label2 is 0 and label2 is 1 if label1 is not 7 as below

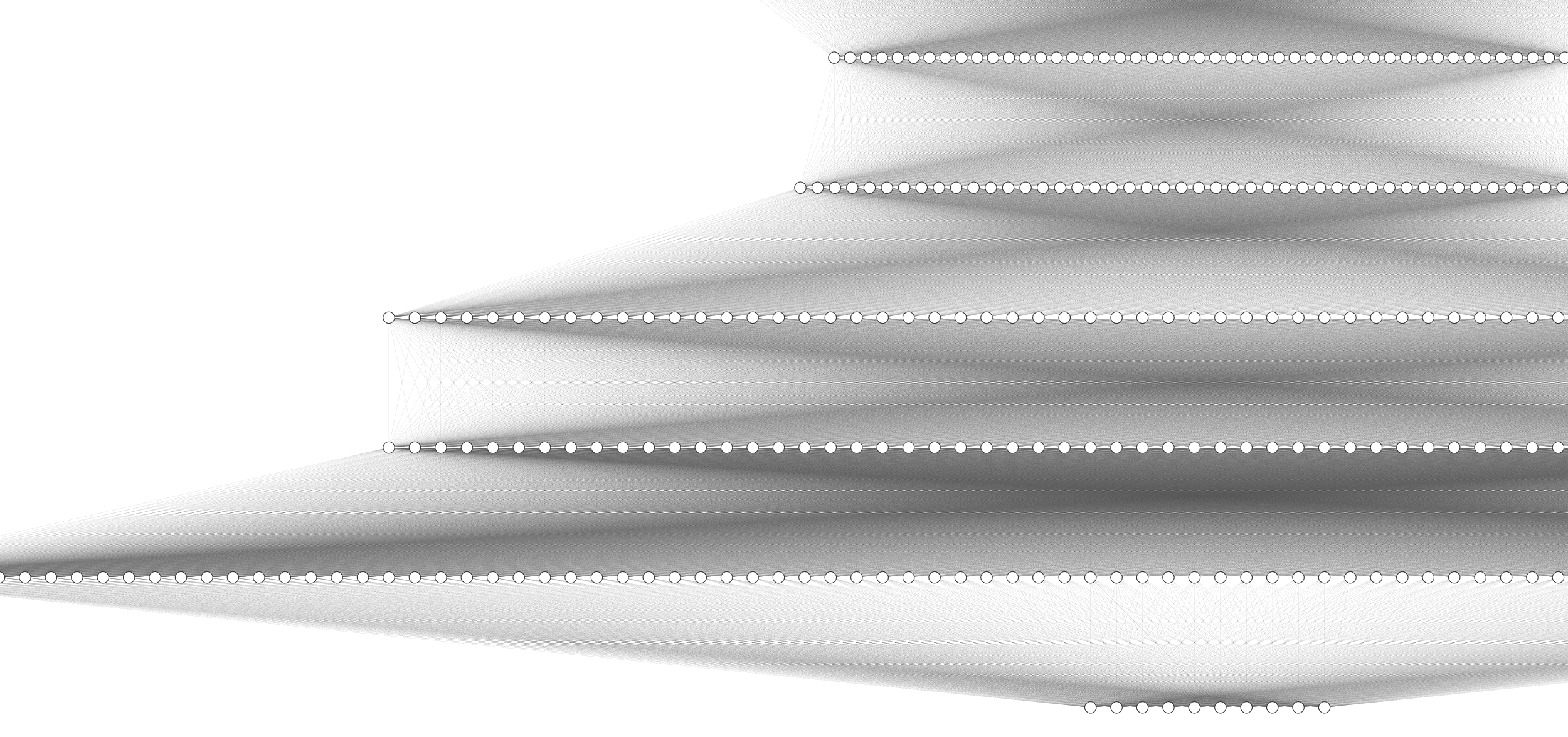
|  |  |
| --- | --- |
| Label 1 | Label 2 |
| 1 | 1 |
| 2 | 1 |
| 3 | 1 |
| 4 | 1 |
| 5 | 1 |
| 6 | 1 |
| 7 | 0 |
| 8 | 1 |
| 9 | 1 |
| 10 | 1 |

**Architecture**

Structure and characteristics of the neural network, concluded by keras tuner automatically searching and identifying the optimal parameters of the neural network

|  |  |  |  |
| --- | --- | --- | --- |
| **Layer** | **Layer type** | **Units** | **Activation** |
| First layer | Dense | 66 | Relu |
| Hidden | Dense | 48 | Tanh |
| Hidden | Dense | 48 | Relu |
| Hidden | Dense | 64 | Relu |
| Hidden | Dense | 64 | Tanh |
| Hidden | Dense | 108 | Relu |
| Output | Dense | 10 | Softmax |

Visualization of the neural network structure



**Experimental Setting**

there is no drop-out in layers.

learning rate is 0.001.

batch size is 128.

number of epochs is 40.

early stopping patience is 5.

Data preprocessing steps:

1. Removed redundant highly correlated features with correlation rate greater than 94%
2. Selected from the remaining columns the most important 21 features by ExtraTreesRegressor to enhance prediction and reduce the chance of over fitting
3. Standardized and normalized the features
4. Convert label1 from [1,2,3,4,5,6,7,8,9,10] to [0,1,2,3,4,5,6,7,8,9] by subtracting 1 to fit it better to the output softmax layer

**Analysis**

Evaluation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Exact Match Loss** | **Exact Match Accuracy** | **Hamming Loss** | **Hamming Accuracy** |
| 1 | 0.542 | 77.47% | 0.168 | 83.19% |
| 2 | 0.563 | 76.19% | 0.178 | 82.18% |
| 3 | 0.567 | 74.33% | 0.198 | 80.18% |
| 4 | 0.550 | 77.08% | 0.172 | 82.82% |
| 5 | 0.532 | 78.07% | 0.160 | 84.00% |
| 6 | 0.556 | 76.39% | 0.177 | 82.26% |
| 7 | 0.534 | 78.03% | 0.162 | 83.84% |
| 8 | 0.530 | 77.49% | 0.168 | 83.15% |
| 9 | 0.561 | 75.06% | 0.191 | 80.90% |
| 10 | 0.538 | 77.36% | 0.169 | 83.07% |
| Mean | 0.547 | 76.75% | 0.174 | 82.56% |
| Standard derivation | 1.314% | 1.186% | 1.159% | 1.159% |

How to run the script

1. download the source code and dataset and layout them like this:
2. unzip source code & dataset package
3. change current working directory to CIS\_SMC as below
4. execute python3 script file main.py

Graphical user interface, application

Description automatically generated

1. it takes approximately an hour to run, in the end, predictions of Test.csv will be saved in ‘predictions.csv’ and following evaluation results will be in the console output

[(0.5420660972595215, 0.7747412919998169, 0.16810596123014143, 0.8318940387698586), (0.5634239912033081, 0.7619273066520691, 0.17819924209298932, 0.8218007579070107), (0.5672074556350708, 0.7433197498321533, 0.198173249769227, 0.801826750230773), (0.5503278374671936, 0.7708181738853455, 0.17175581790798233, 0.8282441820920177), (0.532231867313385, 0.7806563377380371, 0.15996210464946803, 0.840037895350532), (0.5560332536697388, 0.7639435529708862, 0.17740368265073114, 0.8225963173492689), (0.5343391299247742, 0.7803284525871277, 0.1616443181266093, 0.8383556818733907), (0.5301542282104492, 0.7749113440513611, 0.16847033960064128, 0.8315296603993587), (0.5608804821968079, 0.7506073117256165, 0.19102536073458676, 0.8089746392654132), (0.538071870803833, 0.7735509872436523, 0.16930233687994944, 0.8306976631200506)]

std\_variation [0.01314313 0.01185915 0.01159284 0.01159284]

mean [0.54747362 0.76748045 0.17440424 0.82559576]

Thank you for reviewing my anwsers, you comments and advice is apprecaited especailly on how to improve accuracy.