Name:- Swapna Vaidya ***Class:-*** MSDS – 422

COLAB NOTEBOOK LINK :- https://colab.research.google.com/drive/1C62cD4rnZ90ILM9rukRFflW0e01YVRK2

Neural Networks Models

(1) ***A summary and problem definition for management***

MNIST ("Modified National Institute of Standards and Technology") is the de facto "hello world" dataset of computer vision. Since its release in 1999, this classic dataset of handwritten images has served as the basis for benchmarking classification algorithms. As new machine learning techniques emerge, MNIST remains a reliable resource for researchers and learners alike. In this competition, the goal is to correctly identify digits from a dataset of tens of thousands of handwritten images.

A financial institution would like to evaluate machine learning technologies for optical character recognition. Their Initial testing is on the MNIST digits. A recommendation needs to be made to the institution about the neural network typology and hyperparameter settings that are most trustworthy?

(2) ***Discussion of the research design, measurement and statistical methods, traditional and machine learning methods employed***

Based on the exploratory data analysis that was done the survey data, below were some of the findings:-

- There are 2 sets of data provided:- Training dataset and Test dataset.
- In the training data set there are 42000 instances with 784 features and 1 label in the data
- In the test data set there are 42000 instances with no label
- There are no missing values in any of the columns in the data set.

(3) ***Overview of programming work***

This assignment was done in Google Colab, using 4 main libraries (Pandas, numpy, Skylearn and Keras).

- ***Ingestion:-*** Using pandas the training and test data was first loaded from Google OnDrive.
- ***Exploration:-*** of the data was done using methods like describe(), info() and corr().
- ***Data Cleaning and Transformation:-*** No special data cleaning and transformation steps were taken on this data set.
- ***Modeling:-*** The train dataset was broken into features and label by removing the column 1 which is the label. Next the train dataset was split into train and test data using the scikit train_test_split() with a test_size of 20% and random seed of 42. The labels were converted to one hot encoding using numpy utilis to_categorical(). Since Neural Networks works best on scalarized data the features were then divided by 255.
  Keras sequential model was used for neural networks with an activation method of "Relu" for the input and hidden layers. While softmax was used for output layer. Different number of hidden layers were explored for instance, 2 and 5. Then within each of the hidden layers, the number of nodes selected differed as follows:-

Number of Hidden Layers	Nodes For Layer 1	Nodes For Layer 2	Nodes For Layer 3	Nodes For Layer 4	Nodes For Layer 5
2	300	100			
5	500	400	300	200	100
3	300	100	50		
2	300	100			
2	300	100			

The model was compiled with the optimizer "adam", loss as "categorical_crossentropy" and metrics as "Accuracy". The model was fit with a batch size of 128 and with various epochs such as 20, 50 and 100. The process time for fitting the model was recorded. Each of the fit model was then used to evaluate the train and test accuracy metrics. These metrics were recorded. Based on evaluating the metrics of the split train and test data, the models seem to be overfitting.

Number of Hidden Layers	Nodes For Layer 1	Nodes For Layer 2	Nodes For Layer 3	Nodes For Layer 4	Nodes For Layer 5	Epochs	Training Set Accuracy	Test Set Accuracy
2	300	100				20	1	0.97833
5	500	400	300	200	100	20	0.997023	0.97571
3	300	100	50			20	0.9974404	0.973571
2	300	100				50	1	0.97904762
2	300	100				100	1	0.97952381

The models were then used to predict the classes of the actual Kaggle Test data. These prediction were then submitted on the Kaggle site for scoring.

Review of results with recommendations for management

The processing time and score for all the iterations for the neural network model is as follows:-

Number of Hidden Layers	Nodes For Layer 1	Nodes For Layer 2	Nodes For Layer 3	Nodes For Layer 4	Nodes For Layer 5	Epochs	Processing Time	Training Set Accuracy	Kaggke Test Accuracy
2	300	100				20	52 seconds	1	0.97828
5	500	400	300	200	100	20	2 mins 2 secs	0.997023	0.97342
2	300	100				50	2 mins 12 seconds	1	0.9790
2	300	100				100	4 mins 40 seconds	1	0.97785

Based on the above scores it can be seen that a model with 2 hidden layers with a cone shape nodes like 300 and 100 and 50 epochs gives the best score of 0.9790 in Kaggle with a processing time of 2mins and 12 seconds. The amount of overfitting also decreases slightly. With 5 hidden layers the score decreases, and the processing time also increases.

My recommendation to the financial institution would be to go with a less complex model which means minimum hidden layers but more nodes and moderate epochs to limit the processing time but get the most accuracy without overfitting. Hence the hyper parameter setting should be 2 hidden layers and increasing the nodes to 300 and 100 for the 2 layers with “Relu” as the activation function with 50 epochs.