LOCKDOWN MANIA

SHUBHAM JAIN

18117099

2nd year, ME

*Link to GitHUB Repository -*

About given dataset –

x\_train = 1287\*1000 y\_train= 1287\*1

x\_test = 1332\*1000 y\_test = 1332\*1

* In this multiclass classification with No. of training examples (M)= 1287 and No. of features (N) = 1000, logistic regression can’t be used as it is for binary classification. Neural Networks, SVM with gaussian kernel and K Nearest Neighbors are the most suitable algorithms.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

I applied KNN Algorithm and found optimum results at k=10. Score was 0.91.

On applying Neural Network the results were following –

[ LOSS , ACCURACY | [No. of hidden layers, No. of Nodes, Epochs, Score]

[6.816485986337289, 0.5555555820465088] | 1, 64, 2000, 0.94

[6.739723786935434, 0.5605605244636536] | 2, [64,64], 2000, 0.92

[6.954657893997055, 0.5465465188026428] | 3, [64,64,64], 4000, 0.92

[6.709018893428035, 0.5625625848770142] | 4, [64,64,64,64], 2000, 0.92

[6.801133476578079, 0.5565565824508667] | 2, [128,64], 2000, 0.92

[6.653578908593805, 0.5660659670829773] | 1 32, 2000, 0.94

[6.721633689181583, 0.5575575828552246] | 1, 16, 2000, 0.92

[6.685988606633367, 0.5635635852813721] | 1 24, 2000, 0.92

[6.709018849037789, 0.5625626444816589] | 1 32 6000, 0.94

Hence single hidden layer model with 32 nodes and 2000 epochs give lowest loss, highest accuracy and highest accuracy score of 0.94.

Neural Network appears to be performing better on this dataset.

“It was a nice learning experience.”

**Thank you** **DSG**

**feature engineering**

The feature engineering process is:[[6]](https://en.wikipedia.org/wiki/Feature_engineering#cite_note-6)

* [Brainstorming](https://en.wikipedia.org/wiki/Brainstorming) or [testing](https://en.wikipedia.org/wiki/Software_testing) features;[[7]](https://en.wikipedia.org/wiki/Feature_engineering#cite_note-7)
* Deciding what features to create;
* Creating features;
* Checking how the features work with your model;
* Improving your features if needed;
* Go back to brainstorming/creating more features until the work is done.

Even if some features are irrelevant, having too many is better than missing those that are important. [Feature selection](https://en.wikipedia.org/wiki/Feature_selection) can be used to prevent overfitting

Feature explosion can be caused by feature combination or feature templates, both leading to a quick growth in the total number of features.

* Feature templates - implementing feature templates instead of coding new features
* Feature combinations - combinations that cannot be represented by the linear system

Feature explosion can be stopped via techniques such as: [regularization](https://en.wikipedia.org/wiki/Regularization_(mathematics)), [kernel method](https://en.wikipedia.org/wiki/Kernel_method), [feature selection](https://en.wikipedia.org/wiki/Feature_selection).