ML-lab-endsem-report

V.Rishikesh - DA24B050

November 2025

1   Introduction

MNIST dataset is a very famous dataset, which is used when we are learning neural networks. But this report will give you an overview of what happens if we use normal models or some ensemble models.

I have tried Softmax Regression, XGBoost with Softmax, XGBoost + RandomForest with Softmax and then finally XGBoost with RandomForest with OvR(One vs Rest). The best model was XGBoost with RandomForest with OvR.

2   Implementing Softmax Regression:

The first model that I tried is the Softmax(Multiclass Logistic) Regression. Since it is a very basic model, I though of taking this as a base model and then start building on it. Surprisingly it attained a very good f1 score.

Without any hyper parameter tuning, by taking learning rate as 0.1, epochs 1000, the achieved f1 score was 0.90 in time of 7.94s.

2.1   Hyperparameter Tuning of Softmax Regression:

Since we follow kind of a greedy method to tune the hyper parameters by changing only one hyper parameter every time, we may miss the actual best combination of the hyper parameters.

So I looped through 2 lists of learning rate and epochs to find the best possible combination of these hyperparameters.

These are the initial values that I tried on:

- lr= [0.01,0.05,0.2,0.3]

- n = [1000,2500,5000]

Since I can't show all the 12 f1 scores of all the possible combinations, these are top 4 f1 scores.

The are the top 4 f1 scores achieved are:

Table 1: Top 4 f1 scores of Softmax Regression

f1 score	lr	epochs	Runtime(in sec)
0.9127	0.3	1000	9.78
0.9123	0.3	2500	24.83
0.9123	0.2	2500	24.89
0.9107	0.05	5000	48.7

Then I increased the learning rate and decreased by epochs to check if there is any improvement in the f1 score.

These are the values that I tried:

- lr = [0.3,0.5]

- n = [200,500,1000]

The best f1 score obtained was 0.913, which is slightly better than the previous best f1 score so, since the f1 score is not improving much I stopped tuning the Hyper Parameters.

$\therefore$ The best f1 score for softmax Regression is 0.913 for the hyper parameters lr = 0.5 and epochs = 1000.

3 Implementing Decision Tree with XGBoost and Softmax:

Since Softmax is giving a best f1 score of 0.913, to improve the f1 score, I tried using decision trees with XGBoost and Softmax. I built an ensemble model with XGBoost and Softmax to make it a multiclass classifier.

So the model builts 10 trees one for each class and computes gradient and hessian for every tree and then it computes softmax probabilities for that data point to belong to every class.

And then the final prediction will be the one with the highest soft max probability.

When building the tree to find the best feature at every split, it uses an approximate method, which divides the feature values into ńbins, and computes

the cumulative gradient and hessian of the bins and evaluates splits only at the boundaries of the bins.

By doing this the time for the model to train such large datasets will be reduced by a huge factor.

The Hyper-parameters for this model are:

- n-estimators: no.of trees

- learning-rate

- max-depth

- n-bins : no.of bins

- lam : regularization parameter

At first, without any hyper parameter tuning, the model achieved an f1 score of 0.923 when n-estimators = 20, lr = 0.3, depth = 3, n-bins = 10, lam = 1 in a time of 145.6s.

3.1   Hyperparameter Tuning for XGBoost with softmax:

Initial values for the hyper parameters n_estimators, lr and max_depth:

- n_estimators = [5,10,20]

- lr = [0.05,0.1,0.2]

- max_depth = [2,3,5]

These are the top 5 f1 scores acheived:

Table 2: Top 5 accuracy scores

f1 score	n_esti	lr	max_depth	Runtime(in sec)	n_bins	λ
0.9399	20	0.2	5	594	10	0.5
0.922	20	0.1	5	578	10	0.5
0.918	10	0.2	2	321	10	0.5
0.905	20	0.05	5	609	10	0.5
0.902	10	0.1	5	339	10	0.5

Since the run time was more than 5 min and the top 5 f1 scores were achieved for d=5 (except for 1) and for n_estimators = 20(except for 2), I fixed these 2 parameters constant and I increased the lr and increased the n_bins but still the time was not reducing and even the f1 score is not increasing much.

So, I also used RandomForest i.e,. I built an ensemble model with XGBoost, RandomForest and Softmax.

4   Implementing XGBoost with RandomForest and Softmax:

In the previous model(XGBoost with Softmax), a lot time is spend in building the trees even though we used the bins approach for that, So I also used RandomForest to minimise the runtime.

The difference between the previous model and this model is that when finding the best feature at every split, the previous model considers all the features, i.e., it checks every feature at various thresholds and finalizes the one with highest gain or highest decrease in loss.

This will also make all the trees similar, i.e., the splits will be same, to diversify the trees and to also reduce the runtime, I will take a random sample of features at every split and check for the best feature among that sample.

This will greatly reduce the runtime and also decreases the variance.

The Hyper parameters are the same as the previous model apart from the addition feature_sample_size, the optimal value for this is square root of no.of features.

4.1   Hyper Parameter tuning for XGBoost + Softmax + RandomForest :

Initial values of Hyper Parameters n_esti, lr and max_depth:

- n_esti = [20,40,50]

- lr = [0.1,0.3,0.5]

- max_depth = [4,5,7]

First I will iterate through this 3 lists ans then I will tune for n_bins and λ.

After tuning the 3 hyper parameters, n_esti, lr and max_depth, these are the top 5 f1 scores achieved.

Now I tried increasing the n_bins and increasing λ hoping that the runtime decreases or there could be some increament in the f1 score, but the f1 score

Table 3: Top 5 accuracy scores

f1 score	n_esti	lr	max_depth	n_bins	λ	Runtime(in sec)
0.948	40	0.3	7	10	0.5	281
0.947	50	0.1	7	10	0.5	393
0.946	50	0.3	5	10	0.5	334
0.9459	50	0.5	5	10	0.5	312
0.9451	40	0.3	5	10	0.5	156

was less than the best f1 score in the table given. The f1 score dropped to 0.942 and the runtime was 225s.

Now, I used softmax in each of these ensemble models , instead of softmax, I will now use one vs rest (OVR). This is computationally efficient than softmax.

5   XGBoost + RandomForest with OVR:

One vs Rest will build k binary models and returns the probabilities of every model and then return the final prediction as the max probability among those k models , in this case k = 10. To calculate the probabilities we use sigmoid function.

The Hyper parameters are the same as the previous model.But still I wanted to check if there would be some increament in the f1 score.

So I changed the lr and max_depth. These are the values I tried:

- lr = [0.1,0.3,0.5]

- d = [5,7]

The top 3 f1 scores achieved are:

Table 4: Top 3 accuracy scores

f1 score	n_esti	lr	max_depth	n_bins	λ	Runtime(in sec)	Bias	Variance
0.953	40	0.3	7	10	0.5	263	0.0000000	0.0496
0.950	40	0.5	7	10	0.5	272	0.0000000	0.0484
0.946	40	0.5	5	10	0.5	155	0.0000000	0.0519

In the table, the bias is 0 for some values, it doesn't mean it is exactly 0, I 've truncated the bias term in my code to 7 decimal places, so that's why it is showing like that. It is some non-zero value close to 0 but not exactly 0.

And you can see that the variance of the best set of hyperparams is higher than that of the 2nd, but the bias is lower for the best set. So this tradeoff between bias and variance should be carefully handled. Since our goal is to minimize the total error which is $bias^2 + variance$, it is actually low for the best set.

Since I have used RandomForest too , the f1 scores will vary between 0.95 to 0.96 every time I run them, but there is no huge change like dropping to 0.93 or even 0.94. It is varying in the range of 0.95 to 0.96 for the best set of hyperparams.

This is the confusion matrix for the best set of parameters of the XGBoost + RandomForest with OvR:
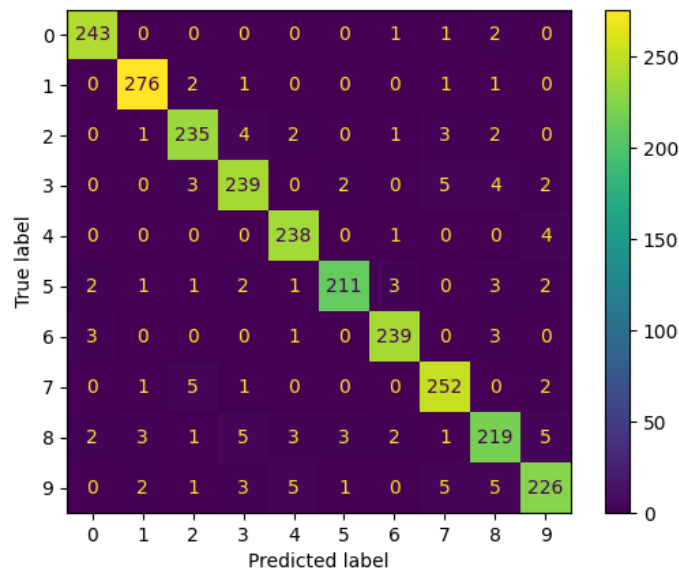

Figure 1: Confusion Matrix

6 My observations from this exercise:

I observed that ensemble models are better than normal and simple models in majority of the cases but not everytime. When I used softmax regression, the model acheived a good f1 score whcich was actually more than the f1 score of some ensemble models mentioned in this report with some pretty good hyper parameters.