EndSem Project

Venkata Sreekar (DA24B030)

1 Models Used

I have used multiple methods like

1. K-Nearest Neighbours
2. XGBoost
3. Random Forrest
4. MultiClass Classifier

Of which KNN gave very high accuracy (0.95) without PCA and in very less time.

To run XGBoost for multiclass, I used code for binary class and did one vs rest for every class (0-9) to get XGBoost for multiclass.

2 Architecture

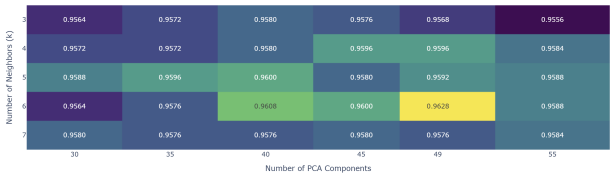
First, I tried all the above listed methods individually.
Then, I ensembled them and made a majority voting between these models (I took multiple models just in KNN like KNN with PCA and with weights etc).

In ensembling, I used 4 models with KNN, 1 model with XGBoost, 1 model with RandomForrest.
And gave the model weights appopriately based on their individual accuracies.

3 Hyper Parameter Tuning

3.1 KNN

The hyper Parameters in KNN are number of nearest neighbours and numbur of principle components used in PCA.
So, I chose k = 6, number of components = 49.



3.2 XGBoost

The hyper Parameters in XGBoost are number of trees, learning rate, max depth. I used

1. number of trees = 50
2. learning rate = 0.3
3. max depth = 6

On increasing any of no of trees, max depth the time taken to run the code is increasing too much,
And on changing learning rate, the accuracy decreases. So I chose these parameters as the final tuning hyper parameters.

3.3 Random Forrest

The hyper parameters in Random Forrest are number of trees.
This has number of features(for each tree) as square root of total number of features.

I used 100 as number of trees for the model because as I increase or decrease, accuracy decreases and time increases if I increase the number of trees.

The hyper paramters I used are

3.4 MultiClass Classifier

The only hyper parameters in Multiclass Classifier are learning rate, number of epochs.

Whatever hyper parameters I use in `SoftmaxRegression`, This is not at all close to `XGBoost` or `KNN`, even if I include this in ensemble, the accuracy is reducing for any hyper parameter tuning.
So I Did not use `SoftmaxRegression` in my `main.ipynb`.

3.5 Ensembling

I ensembled all the above methods at first and took a weighted mode based on their accuracies and the hyper parameters in ensembling is the weights used to their respective methods.

The methods I used in ensembling are

1. KNN (k = 5) without PCA

2. KNN (k = 5) with PCA
3. WeightedKNN (k = 6) without PCA
4. WeightedKNN (k = 6) with PCA
5. XGBoost
6. RandomForrest

Among these, WeightedKNN (k = 6) with PCA gave the highest accuracy (0.9628)

So I gave highest weight to this (3) and 1 to all other models as they are not close to this model.
And this gave me an accuracy of 0.9636.

4 Optimization

4.1 KNN

1. I used numpy vector products wherever possible (to reduce time).
2. I used Weighted KNN this is like giving more weights to the nearest neighbour and gradually decreasing the weights among 1^{st} k nearest neighbours (to increase the accuracy).

4.2 XGBoost

1. I used sparse aware greedy split (given in the previous research paper for assignment-2) and also used random sampling for features.
   This has increased the accuracy and reduced the time taken.

4.3 Ensembling

1. I have used majority voting to decide on the final class based on outputs from multiple models.
2. I have given weights to the models so that more accurate models will have likely to be considered.

5 Summary

We can clearly see that KNN dominated over all the methods (0.9628) without ensembling. This is because same numbers are more or less close to each other w.r.t euclidean distance for pixels.

The discripencies occur when 2 number look alike and their distance is also close ex:- 1, 7.

Doing PCA before KNN also effected it very much (almost by 0.1).

By ensembling, the accuracy increased by a very little amount (0.0008) this is like 2 data points in validation set.

6 Output and accuracy

The below is the confusion matrix for the output of ensembled model.

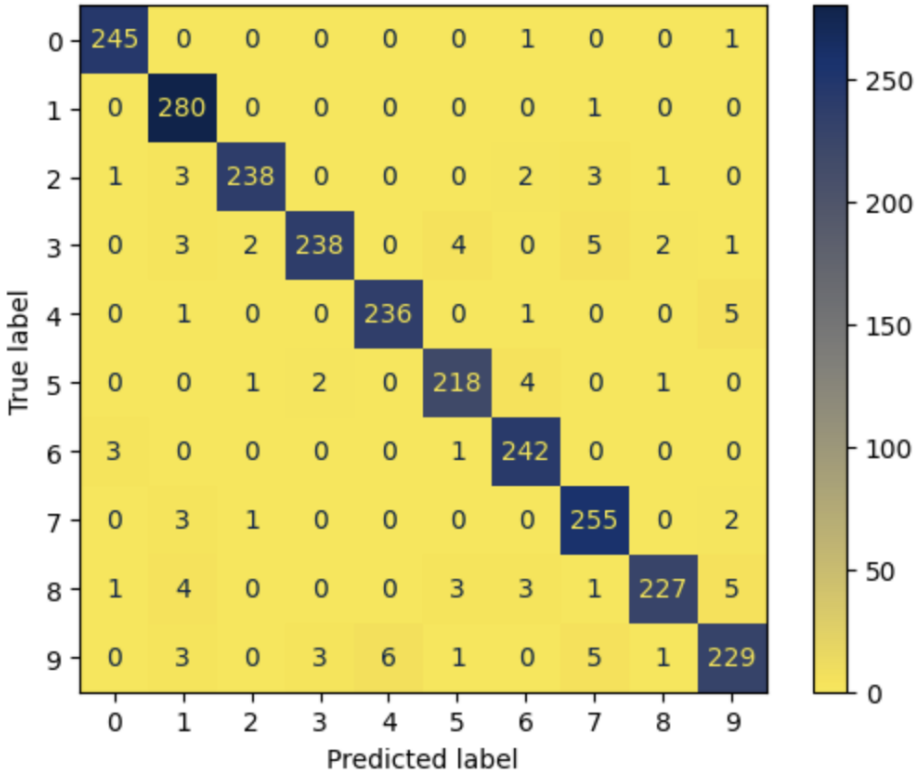


Figure 1: Enter Caption