Yaofu Li, Solar Shao, Jacky Yin
Math 10
Shuhao Cao
March 23th, 2020

Team 8 Report

The final project informs us to analyze 70,0000 images and classify them into different categories. This problem contains a great amount of data. In order to get the highest accuracy, we applied three models: K-Nearest Neighbor, Convolutional Neural Network, and Random forest classifier. For these three methods, what we expect is to get at least 90% accuracy for the project.

Models	**Training Score**	**Cross-Validation**	**Public Leaderboard**
K-Nearest Neighbor	94.44%	94.24%	86.3%
CNN	99.16%	98.24%	98.1%
Random	90.89%	92.54%	92.48%

DATA Preprocessing

We load the npz files in the Jupyter Notebook first. We want to rescale the data by dividing *X_test* by 255.0 to change the value of features from an integer to a float, and this process also normalized the value of *X_test* into the range between 0 and 1. Also, we import *train_test_split* from Sklearn model to split *X, y* for cross-validation

1. K-Nearest Neighbor

For the KNN model, we tried two methods in order to get the one that better solved our problems. First, we used a simplest model from **sklearn** and import **KNeighborclassifier** (*n_neighbor* = 5, *wrights* = 'distance') which only contains two steps (fit into the model, and then predict the value). The original model cost about 30 minutes to calculate all the distance, and we got the public leaderboard about 95.17%. The original **KNN** model's performance is doing well, but the cost of time is a big problem. To improve this model, we thought of dimensionality reduction. For the second test, we used **PCA** to reduce the dimension of **KNN** sample. We set **PCA** with *n_components* = 200, which would reduce the dimension of the sample from 748 to 200.And by using *np.sum(PCA.explained_variance_ratio)*, we have new date with about 93% accuracy with original data. However, when we plug the data in the **KNN** model, it still takes a long time. Hence, we set the *n_components* = 20, which the accuracy is only 50% as original data. Then, we plugged the new data into the model, and we got the training Score 94.44%, Cross-Validation 94.24%, and Public Leaderboard 86.7%, and it only takes about 20s to calculate all distances. This method saved a lot of time, though dimensionality reduction leaded to a much lower accuracy. Then, we set *n_componen*ts = 0.99, but dimension of *X_test* and *X_train* don't match each other. We keep trying different dimensions of both *X* until component = 0.25, However, after we plug them back into the KNN model, public leaderboard gave us about 85% accuracy, which is lower than before.

2. Random Forest Classifier

At the beginning, we were trying to use the Random Forest Classifier, a tree-based machine learning algorithm to predict our data and in order to use the averaging to improve the predictive accuracy and control the issue of over-fitting.

We first normalized the data through dividing X_test and X_train by 255 to make sure that the value become the range between 0 and 1. Then, we used the KFold to split the X_train and y_train into four chunks respectively in order to get the more efficient model. Then, after several times of testing, we found that in this kind of data set, all four Random Forest models with "100 trees in each forest," the maximum depth of the tree as 20, minimum samples split=2, and minimum samples leaf=1 are the most efficient. Within the model, we started with the selection of random samples from a given dataset and then we will be given prediction from every decision tree. Finally, it would help us select the most voted prediction as the final prediction result.

Then, in each model, we randomly selected three chunks as a training data set and one chunk as a cross validation set. Finally, we used that model to predict our predicted_y_cross validation data and calculate the accuracy between the predicted_y_cross validation data and the actual cross validation data and figure out the best model within these four.

3. Convolutional Neural Network

Searching from google, we found that the convolutional neural network is one model that was strongly suggested for this type of question, since it is commonly used for analyzing visual imagery.

In order to use this model, we added one more axis to our *X_train* and *X_test*, increasing *X_train* and *X_test* from two dimensional { *X_train.shape*=(42000, 28, 28) and *X_test.shape*=(28000, 28, 28) } to three dimensional {*X_train.shape*= (42000, 28, 28, 1) and *X_test*. shape= (28000, 28, 28, 1) }. Instead of using *train_test split*, we used *Validation_split*.

Then, we start our CNN process. We used two Conv2D layers and one Maxpool2D layer. The number of the output filters for the first Conv is 32 and for the second layer is 64. The first Conv layer's input shape should follow the shape of *X_test,* which is (28,28,1). We set the activation function "*relu*". Then, we did a normal neural network process after flattening our data. The number of the first hidden layer is 256 and the second one is 128. To avoid overfitting, we also applied dropout to each hidden layer. We used 40% dropout rates for each hidden layer. There are 10 outputs calculated by *softmax* function, which exactly mapped to our results from 0 to 9.

We chose Adam as our optimizer for its advantages that it has Relatively low memory requirements and usually works well even with little tuning of hyperparameters.

We chose *the sparse_categorical_crossentropy* as our loss function. That is because sparse cross-entropy addresses classification problems with a large number of labels by performing the same cross-entropy calculation of error, without requiring that the target variable be one hot encoded prior to training.

Then, we fit our data after splitting them by applying *validation_split* = 0.10, which is similar to the process of using *train_test_split* and *Kfold*. For the epochs, after trying different values multiple times, we found that 20 is the best number that leads to the highest accuracy.

Conclusion

KNN method doesn't work so well compared to other two test. Since, KNN need to calculate every number each step, high dimensional and large data will take so much time to process the data. However, if we reduce dimension of data, PCA also will decreases accuracy of the data which will lower accuracy of the final result.

For the Random Forest Classifier, the drawback is really obvious that there are 10 classes within this data set, which actually won't improve the accuracy. Therefore. it is also not a good idea to utilize the random forest classifier to make the prediction in this case. Random Forest will deliver a decent prediction when the sample size is relatively small.

The CNN method leaded us the highest accuracy and therefore turned to be the best method for image classify type of question.