CSC 421

Assignment #1

CIFAR- 10 Classification Report

Hang Ruan

V00923058

Python Version

- Python 3.9.7

How to Run

- To run K-Nearest-Neighbors classification: python ./knn.py
- To run K-Mean classification: python ./k_mean.py
- To run SoftMax classification: python ./softmax.py

Database

I’m using CIFAR-10 datasets. It consists of 60000 32x32x3 images with 10 classes, with 6000 images per classes. 50000 training images and 10000 testing images are loaded using the keras. dataset library.

- “from keras.dataset” import cifar10

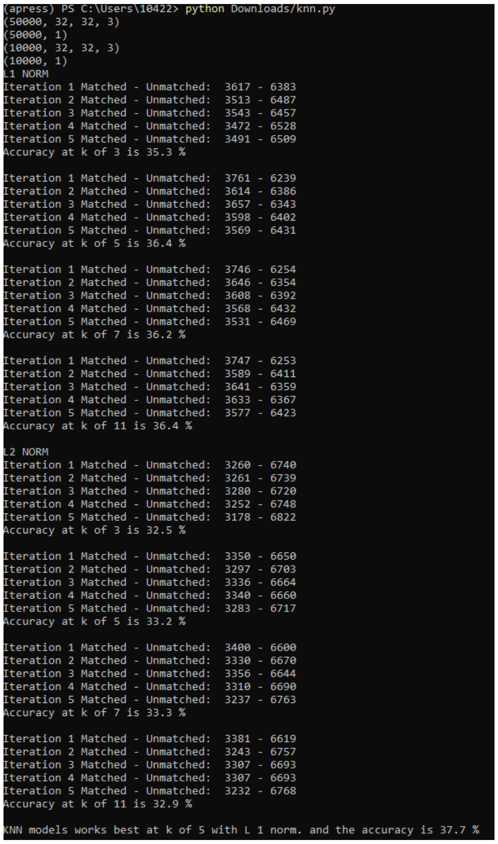
K-Nearest-Neighbors

Description

K-Nearest-Neighbors classification uses the majority label of k nearest neighboring images to classify an input image. It captures the similarity of image pixels.

Method

To better improve the performance of our KNN classification problem, I use 5-fold cross validation to tune the hyperparameter. To calculate the distance between images on a graph, L1 distance and L2 distance calculation are used and compared. Different values of k of 3, 5, 7 ,11 is also used and tuned as hyperparameters. For every combination of distance calculation methods and values of k, 5-fold trainings and validations are completed, and the averages of accuracies are taken and compared to find the best hyperparameter.



```
(apress) PS C:\Users\10422> python Downloads/knn.py
(50000, 32, 32, 3)
(50000, 1)
(10000, 32, 32, 3)
(10000, 1)
L1 NORM
Iteration 1 Matched - Unmatched:  3617 - 6383
Iteration 2 Matched - Unmatched:  3513 - 6487
Iteration 3 Matched - Unmatched:  3543 - 6457
Iteration 4 Matched - Unmatched:  3472 - 6528
Iteration 5 Matched - Unmatched:  3491 - 6509
Accuracy at k of 3 is 35.3 %

Iteration 1 Matched - Unmatched:  3761 - 6239
Iteration 2 Matched - Unmatched:  3614 - 6386
Iteration 3 Matched - Unmatched:  3657 - 6343
Iteration 4 Matched - Unmatched:  3598 - 6402
Iteration 5 Matched - Unmatched:  3569 - 6431
Accuracy at k of 5 is 36.4 %

Iteration 1 Matched - Unmatched:  3746 - 6254
Iteration 2 Matched - Unmatched:  3646 - 6354
Iteration 3 Matched - Unmatched:  3608 - 6392
Iteration 4 Matched - Unmatched:  3568 - 6432
Iteration 5 Matched - Unmatched:  3531 - 6469
Accuracy at k of 7 is 36.2 %

Iteration 1 Matched - Unmatched:  3747 - 6253
Iteration 2 Matched - Unmatched:  3589 - 6411
Iteration 3 Matched - Unmatched:  3641 - 6359
Iteration 4 Matched - Unmatched:  3633 - 6367
Iteration 5 Matched - Unmatched:  3577 - 6423
Accuracy at k of 11 is 36.4 %

L2 NORM
Iteration 1 Matched - Unmatched:  3260 - 6740
Iteration 2 Matched - Unmatched:  3261 - 6739
Iteration 3 Matched - Unmatched:  3280 - 6720
Iteration 4 Matched - Unmatched:  3252 - 6748
Iteration 5 Matched - Unmatched:  3178 - 6822
Accuracy at k of 3 is 32.5 %

Iteration 1 Matched - Unmatched:  3350 - 6650
Iteration 2 Matched - Unmatched:  3297 - 6703
Iteration 3 Matched - Unmatched:  3336 - 6664
Iteration 4 Matched - Unmatched:  3340 - 6660
Iteration 5 Matched - Unmatched:  3283 - 6717
Accuracy at k of 5 is 33.2 %

Iteration 1 Matched - Unmatched:  3400 - 6600
Iteration 2 Matched - Unmatched:  3330 - 6670
Iteration 3 Matched - Unmatched:  3356 - 6644
Iteration 4 Matched - Unmatched:  3310 - 6690
Iteration 5 Matched - Unmatched:  3237 - 6763
Accuracy at k of 7 is 33.3 %

Iteration 1 Matched - Unmatched:  3381 - 6619
Iteration 2 Matched - Unmatched:  3243 - 6757
Iteration 3 Matched - Unmatched:  3307 - 6693
Iteration 4 Matched - Unmatched:  3307 - 6693
Iteration 5 Matched - Unmatched:  3232 - 6768
Accuracy at k of 11 is 32.9 %

KNN models works best at k of 5 with L 1 norm. and the accuracy is 37.7 %
```

Figure 1 Accuracies of KNN outputs

Result

A total number of 8 accuracies were computed for all distance calculation methods and values of k. L1 norm performed overall better than L2 norm and value of 5, 7, 11 offer a high accuracy than 3 as the value of k.

Using L1 norm to calculate distances between images and 5 as value of k showed the most promising results of 36.4% average accuracy on the validation set. Using this hyperparameter I was able to get an accuracy of 37.7% on the testing set.

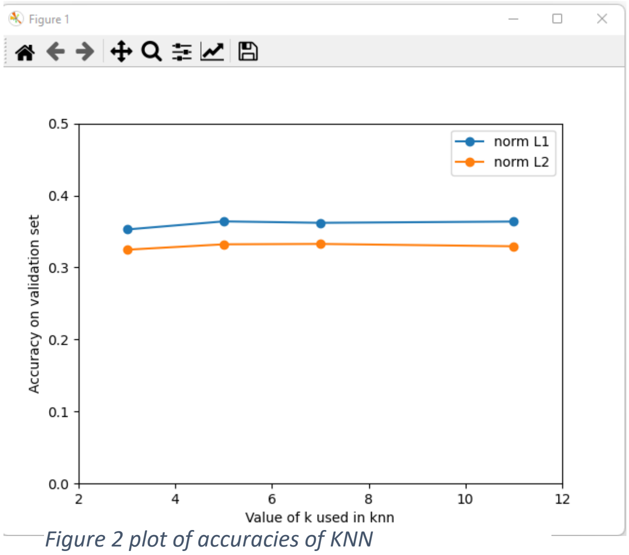


Figure 2 plot of accuracies of KNN

K_mean

Description

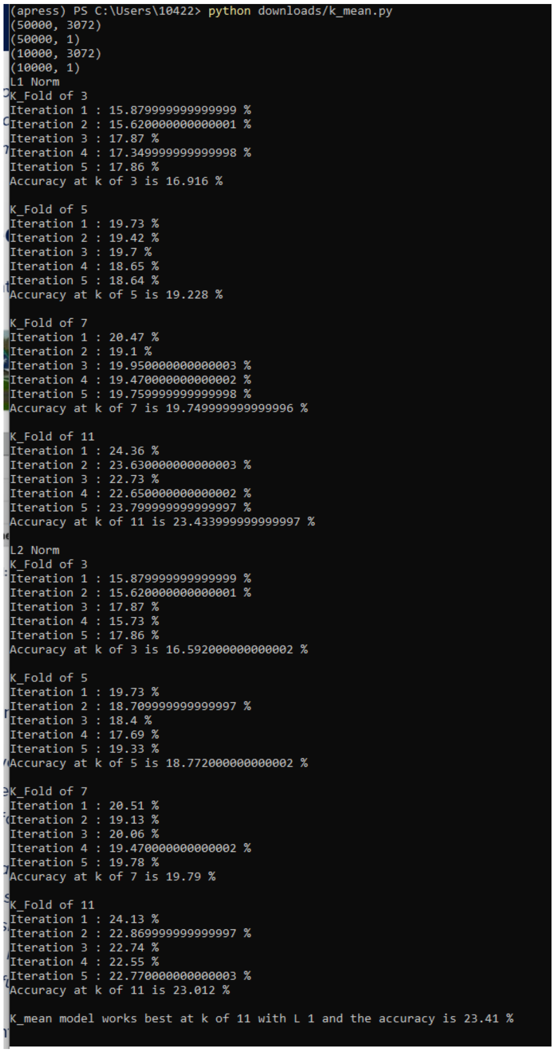
K_mean is a clustering algorithm that group the dataset into k number of non-overlapping clusters. The majority of label of images in a cluster determines the label for the cluster.

Method

The training set is divided into k clusters where the cluster's centroid is the arithmetic mean of all the data points within the cluster. There are two hyperparameters that are tuned using 5-fold cross validation. L1 and L2 norm are both used to calculate the distance between points and different number (3,5,7,11) of clusters are also compared to identify the best hyperparameter.

Result

From all the accuracies that the program outputs, we can conclude that in this classification program, the k_mean model works best when k has a value of 11 while using L1 norm to calculate the distance between points. It predicted the testing dataset at an accuracy of 23.41%



```
(apress) PS C:\Users\10422> python downloads/k_mean.py
(50000, 3072)
(50000, 1)
(10000, 3072)
(10000, 1)
L1 Norm
K_Fold of 3
Iteration 1 : 15.879999999999999 %
Iteration 2 : 15.620000000000001 %
Iteration 3 : 17.87 %
Iteration 4 : 17.349999999999998 %
Iteration 5 : 17.86 %
Accuracy at k of 3 is 16.916 %

K_Fold of 5
Iteration 1 : 19.73 %
Iteration 2 : 19.42 %
Iteration 3 : 19.7 %
Iteration 4 : 18.65 %
Iteration 5 : 18.64 %
Accuracy at k of 5 is 19.228 %

K_Fold of 7
Iteration 1 : 20.47 %
Iteration 2 : 19.1 %
Iteration 3 : 19.950000000000003 %
Iteration 4 : 19.470000000000002 %
Iteration 5 : 19.759999999999998 %
Accuracy at k of 7 is 19.749999999999996 %

K_Fold of 11
Iteration 1 : 24.36 %
Iteration 2 : 23.630000000000003 %
Iteration 3 : 22.73 %
Iteration 4 : 22.650000000000002 %
Iteration 5 : 23.799999999999997 %
Accuracy at k of 11 is 23.433999999999997 %

L2 Norm
K_Fold of 3
Iteration 1 : 15.879999999999999 %
Iteration 2 : 15.620000000000001 %
Iteration 3 : 17.87 %
Iteration 4 : 15.73 %
Iteration 5 : 17.86 %
Accuracy at k of 3 is 16.592000000000002 %

K_Fold of 5
Iteration 1 : 19.73 %
Iteration 2 : 18.709999999999997 %
Iteration 3 : 18.4 %
Iteration 4 : 17.69 %
Iteration 5 : 19.33 %
Accuracy at k of 5 is 18.772000000000002 %

K_Fold of 7
Iteration 1 : 20.51 %
Iteration 2 : 19.13 %
Iteration 3 : 20.06 %
Iteration 4 : 19.470000000000002 %
Iteration 5 : 19.78 %
Accuracy at k of 7 is 19.79 %

K_Fold of 11
Iteration 1 : 24.13 %
Iteration 2 : 22.869999999999997 %
Iteration 3 : 22.74 %
Iteration 4 : 22.55 %
Iteration 5 : 22.770000000000003 %
Accuracy at k of 11 is 23.012 %

K_mean model works best at k of 11 with L 1 and the accuracy is 23.41 %
```

Figure 3 K_mean Accuracy output

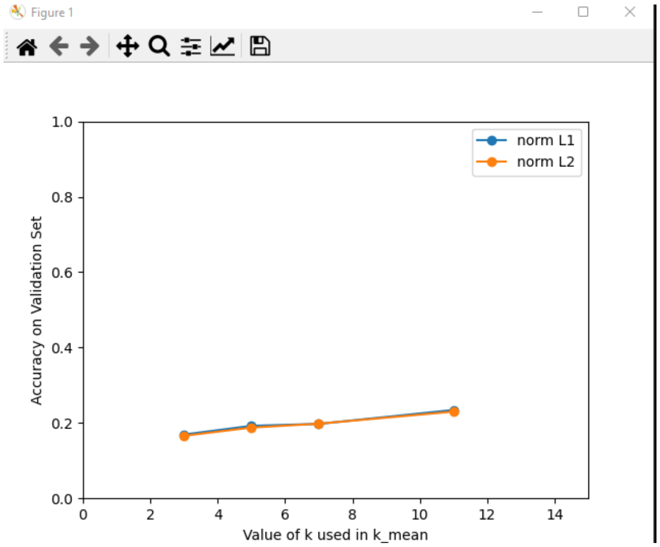


Figure 4 Plot of k-mean accuracy

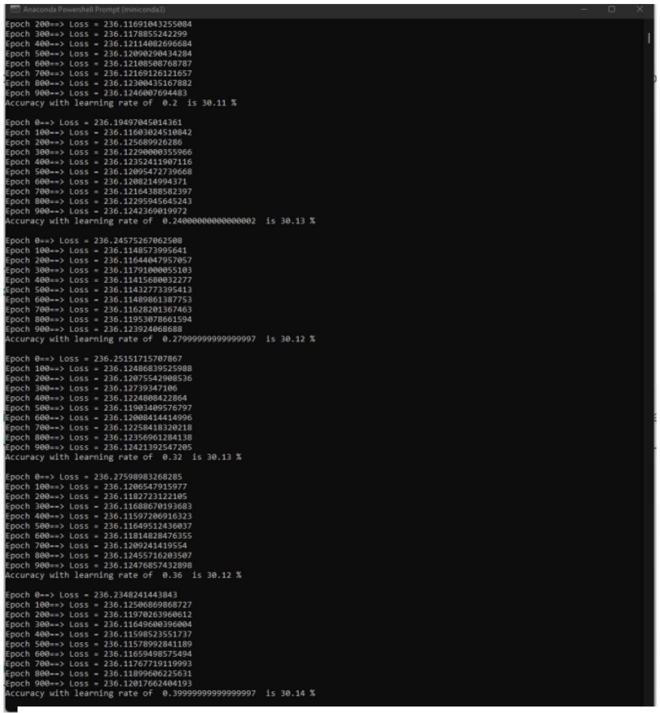
SoftMax

Description

SoftMax classifier uses a mapping function f that takes an input set of data and map them to the output class labels via a linear dot product and a weight matrix. The output score is transformed into probability using the SoftMax activation function.

Method

The weight matrix is initialized to a N x M matrix of decimal numbers and the bias is initialized to a 1 x M matrix. Label probability is then calculated using dot product and SoftMax activation function. Weight matrix and bias matrix is updated using gradient decent. Ten equally spaced learning rate from 0.04 to 0.44 are used in this training. Loss function is implemented to capture how close the predicted label is to the true label.



```
Epoch 200==> Loss = 236.11691043255084
Epoch 300==> Loss = 236.1178855242299
Epoch 400==> Loss = 236.12114082696684
Epoch 500==> Loss = 236.12090290434284
Epoch 600==> Loss = 236.12108508768787
Epoch 700==> Loss = 236.12169126121657
Epoch 800==> Loss = 236.12300435167882
Epoch 900==> Loss = 236.1246007694483
Accuracy with learning rate of  0.2  is 30.11 %

Epoch 0==> Loss = 236.19497045014361
Epoch 100==> Loss = 236.11603024510842
Epoch 200==> Loss = 236.125689926286
Epoch 300==> Loss = 236.12290000355966
Epoch 400==> Loss = 236.12352411907116
Epoch 500==> Loss = 236.12095472739668
Epoch 600==> Loss = 236.1208214994371
Epoch 700==> Loss = 236.12164388582397
Epoch 800==> Loss = 236.12295945645243
Epoch 900==> Loss = 236.1242369019972
Accuracy with learning rate of  0.24000000000000002  is 30.13 %

Epoch 0==> Loss = 236.24575267062508
Epoch 100==> Loss = 236.1148573995641
Epoch 200==> Loss = 236.11644047957057
Epoch 300==> Loss = 236.11791000055103
Epoch 400==> Loss = 236.11415680032277
Epoch 500==> Loss = 236.11432773395413
Epoch 600==> Loss = 236.11489861387753
Epoch 700==> Loss = 236.11628201367463
Epoch 800==> Loss = 236.11953078661594
Epoch 900==> Loss = 236.123924068688
Accuracy with learning rate of  0.27999999999999997  is 30.12 %

Epoch 0==> Loss = 236.25151715707867
Epoch 100==> Loss = 236.12486839525988
Epoch 200==> Loss = 236.12075542908536
Epoch 300==> Loss = 236.12739347106
Epoch 400==> Loss = 236.1224808422864
Epoch 500==> Loss = 236.11903409576797
Epoch 600==> Loss = 236.12008414414996
Epoch 700==> Loss = 236.12258418320218
Epoch 800==> Loss = 236.12356961284138
Epoch 900==> Loss = 236.12421392547205
Accuracy with learning rate of  0.32  is 30.13 %

Epoch 0==> Loss = 236.27598983268285
Epoch 100==> Loss = 236.1206547915977
Epoch 200==> Loss = 236.1182723122105
Epoch 300==> Loss = 236.11688670193683
Epoch 400==> Loss = 236.11597206916323
Epoch 500==> Loss = 236.11649512436037
Epoch 600==> Loss = 236.11814828476355
Epoch 700==> Loss = 236.1209241419554
Epoch 800==> Loss = 236.12455716203507
Epoch 900==> Loss = 236.12476857432898
Accuracy with learning rate of  0.36  is 30.12 %

Epoch 0==> Loss = 236.2348241443843
Epoch 100==> Loss = 236.12506869868727
Epoch 200==> Loss = 236.11970263960612
Epoch 300==> Loss = 236.11649600396004
Epoch 400==> Loss = 236.11598523551737
Epoch 500==> Loss = 236.11578992841189
Epoch 600==> Loss = 236.11659498575494
Epoch 700==> Loss = 236.11767719119993
Epoch 800==> Loss = 236.11899606225631
Epoch 900==> Loss = 236.12017662404193
Accuracy with learning rate of  0.39999999999999997  is 30.14 %
```

Figure 5Softmax Accuracy output

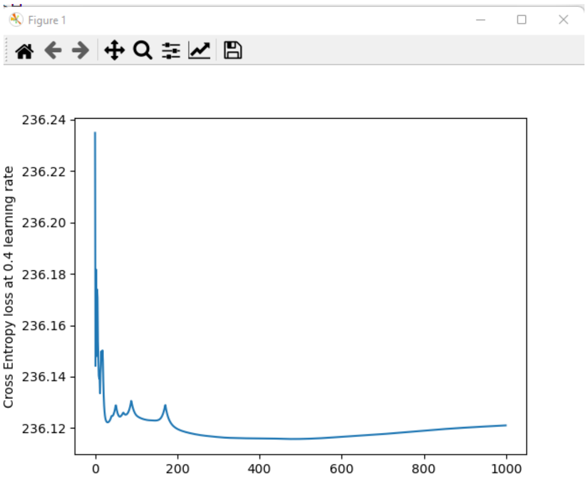


Figure 7 Loss function vs. epoch plot

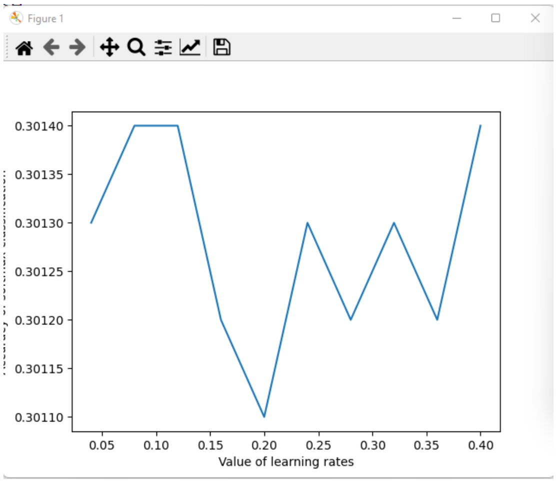


Figure 6 Softmax Accuracy vs. learning rate

Result

SoftMax classifier returns similar results using various learning rate in this test. The accuracy converges to 30% on the testing set. Loss function plot also shows the predicted label converges to the true label as number of iterations increases.

Conclusion

In this experiment, we implemented three different classification models: KNN, K_mean and SoftMax classification. Out of all three models, KNN model obtained the best result of an accuracy of 37.7%.