Ashwini Iyer
CS 4641
ML Project 1

Supervised Learning Report

In this assignment I explore core techniques in supervised learning by designing two interesting classification problems. The two classification problems I chose were identifying images and handwritten digits. I chose image classification tasks to distinguish the capabilities of attribute based classifiers from those that might require more data to generalize. In this paper, I will elaborate on my process and findings.

Datasets & Justification

I chose the CIFAR-10 dataset and the MNIST data set as my classification data sets.

I find these data sets interesting not only because of the practical real life solutions they could lead to, but also because of the results they provided when the supervised learning techniques were applied to them. Image classification and understanding are classic tasks Both datasets are image classification tasks of varying difficulty but are well studied benchmarks that I could use to compare the strengths and weaknesses of the Decision Trees, KNN, boosting, neural nets, and SVMs.

The MNIST dataset contains images of handwritten digits with approximately 50,000 training images and 10,000 testing images, where each sample is a 28 x 28 image with no color channels. This dataset is well benchmarked and could serve as an easier baseline to test the classifiers in case they fail on the hard CIFAR data. As we must compare the methods as they work across dataset size, 60,000 images is a strong number for a 10 classification task. As preprocessing, I flattened the image dimensions into a vector of 784 and performed an argmax over the label dimension.

For a more challenging image classifying problem I chose to use the CIFAR-10 Dataset. The CIFAR-10 Dataset consists of 60,000 32x32 colour images in 10 classes, with 6000 images per class. The classes in the dataset includes 10 classes, including airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. There are 50,000 training images and 10,000 test images, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.

Occasionally in the interest of time, performance, and testing on little data, partial datasets were used from the overall image corpuses. Classification with different amounts of data is an active research field and can be crucial in choosing the right classifier. A classic comparison is the SVMs tend to need less data than neural networks which is why they were more popular in the early 2000s for many tasks that didn't have as much data and computing power. Even now, in fields like deep learning, few shot and one shot learning are hot topics.

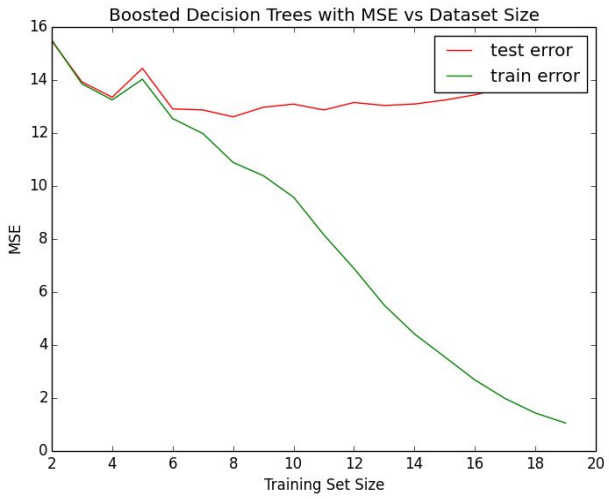
Decision Trees

Decision Trees are a non-parametric supervised learning method commonly applied to classification and regression problems. The goal of Decision Trees is to create a model that can predict the value of target variables by inferring decision rules from data features. Pruning is a technique used by machine learning programmers to reduce overfitting. They do this by reducing the size of decision trees by removing sections that are not directly related to classifying instances. It reduces overfitting by reducing the complexity of the final classifier.

I decided to prune the decision trees by setting max_length. These are the decision tree results over dataset size for the CIFAR-10 dataset.

CIFAR-10 Decision Tree Data

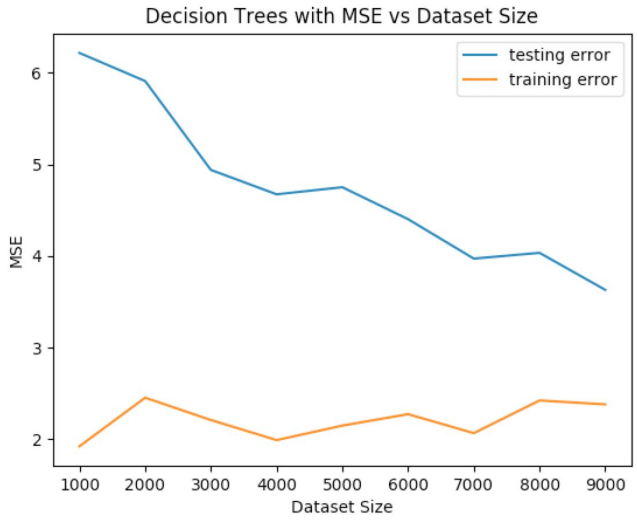
Max Depth	Train Error	Test Error
2	15.50815	15.47215
3	13.8547	13.9267
4	13.258725	13.35255
5	14.0337	14.4424
6	12.544725	12.9135
7	11.98425	12.8732
8	10.893625	12.61435
9	10.399825	12.972
10	9.5861	13.0927
11	8.161425	12.87135
12	6.887375	13.1587
13	5.504	13.04065
14	4.4173	13.093
15	3.55215	13.242
16	2.68145	13.44135
17	1.98655	13.7285
18	1.434375	13.60015
19	1.056625	13.61655



After running the experient, I now realize that max_length is probably not the best pruning approach as it does not prevent it from stopping early. Regardless, judging by the data I think it did a reasonable job in classifying the image classes.

It is clear from the graphs that I get lower training error rates on the deeper trees. This is expected because with more training data the more likely it will have a lower error. It is also visible from the CIFAR graph that after a depth of 5, the test error set goes up slightly as the depth increases while the training error continues to go down. Therefore, decision trees on Cifar-10 are overfitting and poorly generalize to the test set data. This may be due to the nature of decision trees that might pick a few attributes to split on based on information gain for the training data, which don't generalize to the testing data. K-fold cross validation could be used to minimize overfitting. In fact, I expect as the depth increases, the possibilities for overfitting to increase as the trees will split on more attributes in the training set. Other models may not be as likely to overfit.

These are the decision tree results over dataset size for the MNIST dataset.

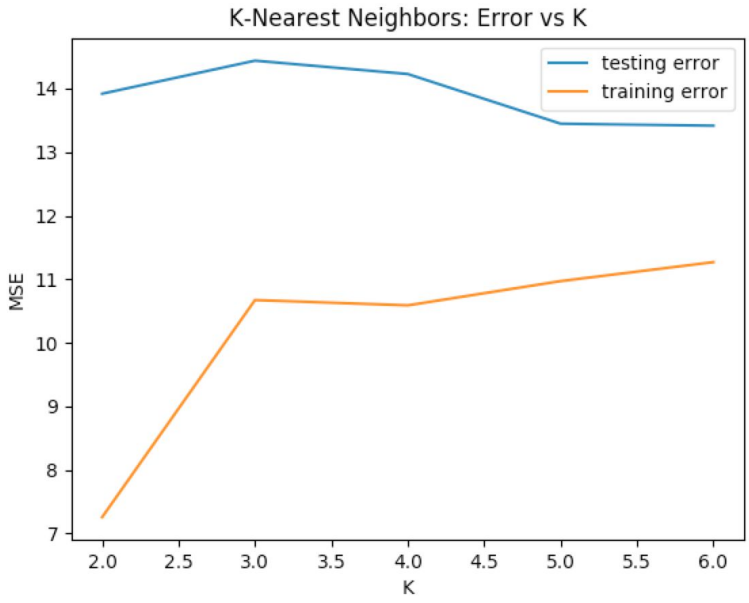


MNIST is another image dataset, but visually it is much less complex than the CIFAR dataset. As we can see, the decision tree classifier does not overfit on this data and the testing error manages to consistently decrease. I fixed the max depth to 8 after some hyperparameter searching. I hypothesize that because a lot of pixels in the 28 by 28 image in the MNIST dataset are unique to the various images, a decision tree could possibly split on those pixels as attributes with lots of information gain minimizing entropy across the images. In comparison to CIFAR, where there is a color channel and visual specificity isn't high in terms of boundaries and unique pixels, it makes sense for the MNIST dataset to be much easier on the decision tree classifier.

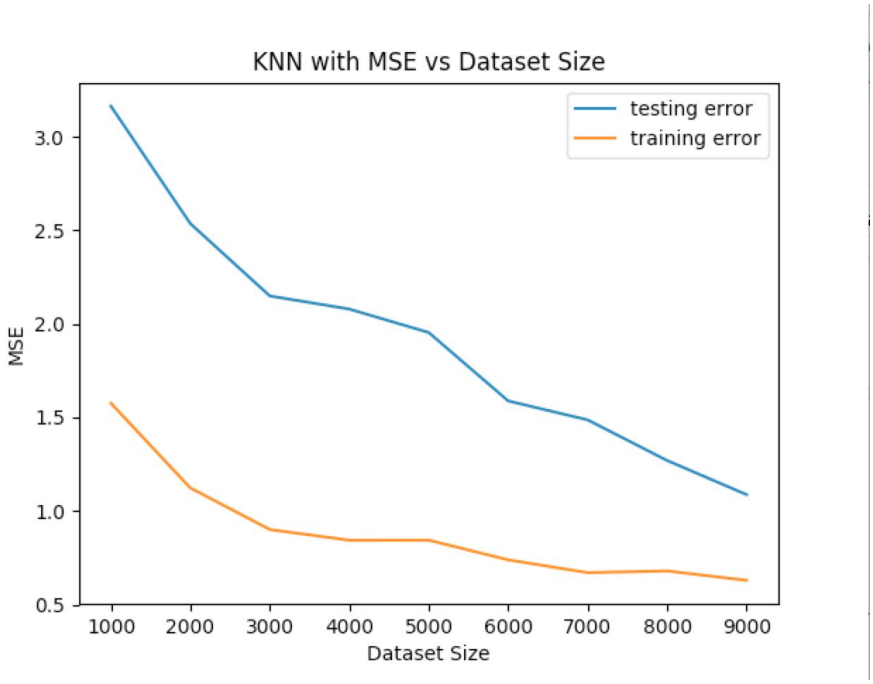
Taking a look at the classifier, the decision trees performed admirably well on the MNIST data and also seemed to overfit on both datasets. To improve performance for the decision trees, I would implement cross validation. I did try random sampling instead to reduce overfitting, but the decision tree would still select attributes that fit too closely to the set of training data. Poor performance on the datasets for decision trees has to do with the dataset selection because if I selected datasets that had more defined attributes that were boolean and a few multi-class attributes, then the decision trees would have performed much better.

K-Nearest Neighbors

The principle of the supervised K-Nearest Neighbors algorithm is to find a certain defined number of of samples in the training set close to a new point and derive a label from that set. In research, K-Nearest Neighbors has been found to be successful in classification and regression problems including handwriting recognition. So, it will be very interesting to see how it performs on the MNIST dataset.



The above graph is for the CIFAR dataset. Because of limited training time, the full dataset for CIFAR could not explored, so I used a much smaller training set size to explore the effect of increasing K on a much smaller dataset size. As I increase the number of neighbors for checking, the testing error decreases slightly but the training error increases. This may be due to the split in training and testing error along the CIFAR dataset. In the CIFAR dataset, the distance in pixels for images is harder to learn than for MNIST. As a result, I chose to display the K-nearest neighbor graph instead of dataset size which doesn't affect MSE too much.



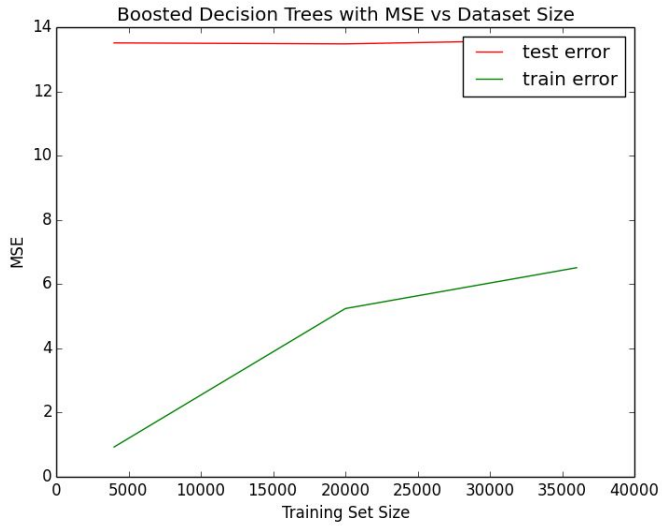
Above is a graph of KNN vs dataset size for MNIST dataset with fixed K at 5 and varying dataset size. Because the CIFAR dataset was much larger than the MNIST dataset, it was hard within the time constraints to accurately vary the dataset size across the CIFAR dataset. For MNIST, however increasing the dataset size provides more examples to draw nearest neighbors from, explaining the decreasing training and testing errors.

KNN is a simple classifier as there is no training time and the performance is solely based on model memorization and how well the distance between points in each of the distinct classes is defined. Binary distance between pixels is hard to define and especially in the CIFAR 10 dataset, so the model performance, again, has to do with the problem chosen. Overfitting wasn't as much of a problem with this classifier, but instead I recommend using PCA to perform dimensionality reduction and cluster the data by classes before using KNN. This is also a common technique when you have high dimensional data and are trying to perform nearest neighbor search.

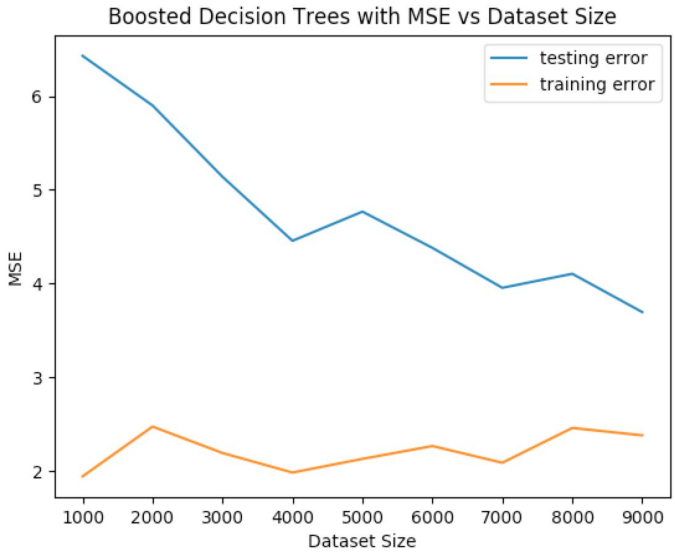
Boosting

Boosting is a general method for increasing the accuracy of supervised learning algorithms, it can be extended to apply to most if not all learning algorithms. I used the AdaBoost algorithm and chose 10 to be the number of estimators run in the experiment, because I observed that any estimators over 10 had little impact on the accuracy of the results.

CIFAR-10 Boosted Decision Tree Graph



MNIST Boosted Decision Tree Graph



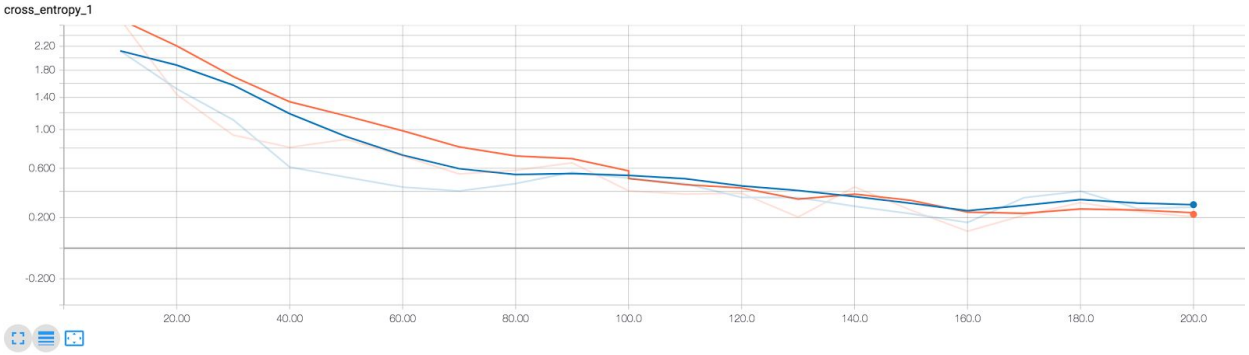
The graph is very similar to decision trees without boosting for both the MNIST and CIFAR-10 data, with the error being a bit lower. I think the reasons for why decision trees work better on MNIST still holds as a lot of pixels in the handwritten digits images uniquely identify the corresponding class, so a decision tree could possibly split on those attributes. Again, the boosted decision tree performs much better on the MNIST dataset than the CIFAR dataset. This is indicated by a decrease in training and testing error as the dataset size increases. I was only able to examine 3 points of dataset size in the CIFAR dataset, but the mean squared error does not decrease interestingly among the training points, indicating that the boosted decision tree cannot classify the CIFAR-10 data well.

Neural Networks

Neural Networks are networks modeled after the neurons in the brain. In Neural Networks the 'artificial neurons' are configured to recognize patterns, and perform tasks like clustering and classification.

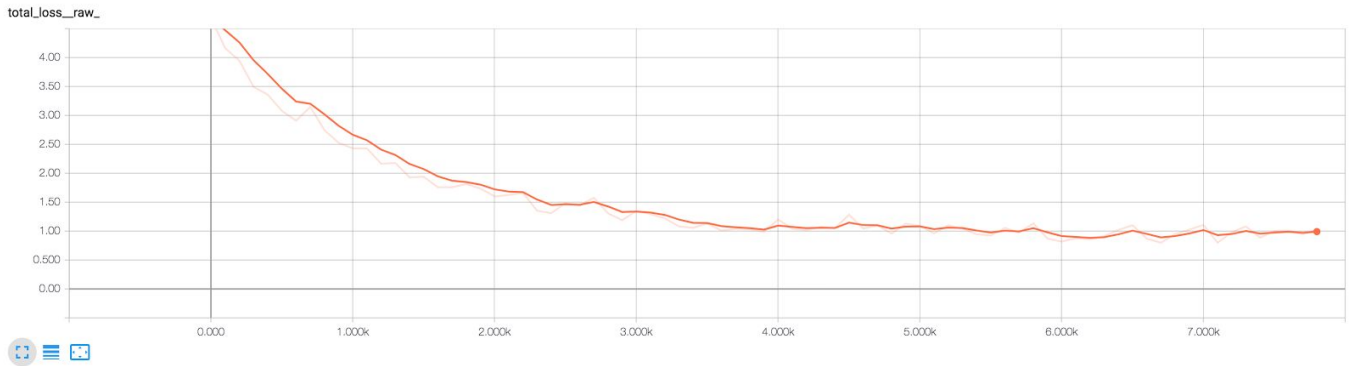
I decided to implement a Convolutional Neural Network because I know that it works well with high dimensional data sets and I am curious to see if this holds true with the MNIST and CIFAR datasets. A Convolutional Neural Network (CNN) consists of learning weights and biases, and at every input a dot product is performed to enforce non-linearity. It uses the SVM/Softmax function as its loss function.

Cross Entropy Loss for MNIST dataset using Convolutional Neural Network



The orange line represents training curve, whereas the blue line is the training curve. The x-axis is the training epochs and y-axis is the cross entropy loss. The corresponding accuracy of classification on the MNIST test set for a convolutional neural network trained in Tensorflow reaches 0.99. Training a convolutional neural network for MNIST also produces amazing results with a 0.99 testing accuracy.

Raw Cross Entropy Loss for CIFAR-10 dataset using Convolutional Neural Networks

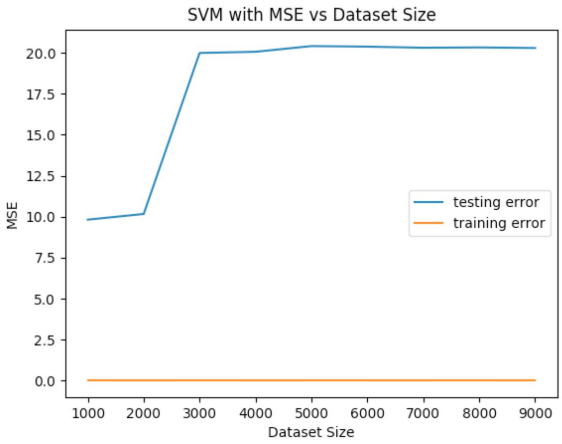


The x-axis is the number of training iterations and the y-axis is the cross-entropy loss. I used cross-entropy loss rather than MSE because it is more stable for training convolutional neural networks. However, these state of the art convolutional neural networks where the architecture is engineered for image processing achieve top accuracy results on the CIFAR-10 dataset around 70% accuracy. Top models including neural networks like inception that incorporate a lot of image techniques into the architecture are pushing the limits of testing accuracy to around 97%. These classifiers have much more model potential but also require a lot more data to effectively tune model parameters in comparison to classifiers like decision trees and SVMs (that perform better with less data).

Support Vector Machines

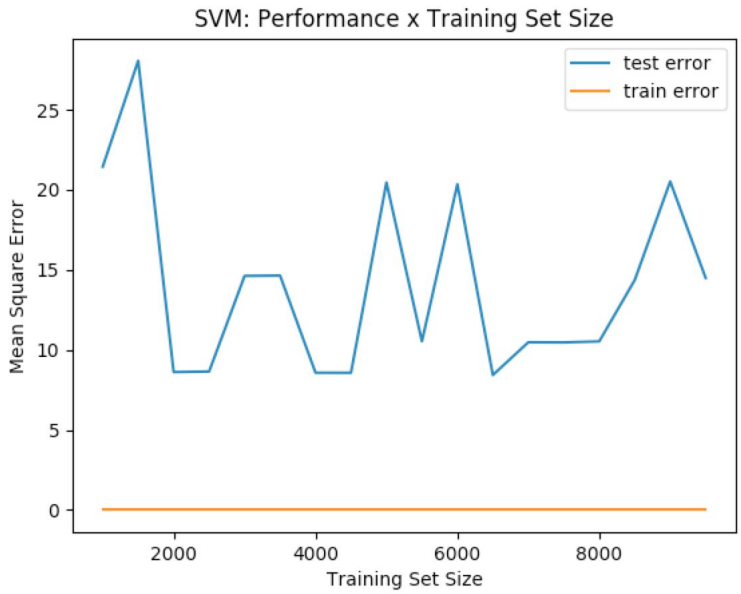
Support Vector Machines, commonly known among the machine learning community as SVMs, are a set of supervised learning techniques designed for classification and outlier detection. SVMs are very effective in high dimensionality sets and is memory efficient.

SVMs vs Training set size for the Iris data set



These results while initially might appear quite unnatural can be explained with the knowledge of support vector machines and this particular dataset. For small dataset sizes, 1-10k the svm classifier tends to overfit a lot with a very low training error and testing error that stays quite high. I attempted to meddle with hyperparameters gamma and the kernel type. Two suggestions for future efforts beyond an ordinary svm are to use dimensionality reduction methods like PCA to reduce dimensions along dimensions with highest variance, simplifying the problem of drawing hyperplanes through the data and easing the burden on the support vector machines. Another suggestion would be to use cross validation to reduce overfitting where folds of validation and training data are cycled through the set of training data.

SVM for Cifar 10 data



Due to the fact that the CIFAR-10 data has such a high dimensionality, we can see here that the SVM did relatively well and had a small and consistent training error. Similarly to the MNIST dataset, the training error is quite low while the test error is nonzero. Interestingly, increasing training set size for a harder dataset to separate linearly does decrease the error.

SVM are great classifiers with smaller dataset sizes and were very popular back in the early 2000s before the advent of strong GPUs and huge datasets. They have high data efficiency and involve drawing separable hyperplanes through the data. However, for datasets that are not very linearly separable SVMs struggle and they also are subject to overfitting. The gamma variable was a hyperparameter that can be tricky to train, but in comparison to decision trees and KNN, they can be more robust for image classification. However, similar to KNN, I recommend using PCA or some dimensionality reduction technique before training or else the MSE loss can be quite high.

Conclusion

In this paper I have compared various learning algorithms including Decision Trees, Boosting, K-Nearest Neighbors, Neural Networks and SVMs on two different data sets: CIFAR-10 and the MNIST dataset. It was rather interesting how most of the algorithms performed much better on the MNIST dataset than the CIFAR-10 dataset. I speculate that this is because of the difference in dimensionality. The CIFAR-10 dataset has a much larger number of features per example than the IRIS data set and as a result finding the required features for classification is much more challenging. Additionally, the MNIST data seems more linearly separable in a hyperplane and could be split better on attributes. As a result, the decision trees and svm methods can still

achieve good performance, while they struggle on the CIFAR dataset. This is why the CIFAR dataset performed so well under the CNN which analyzes combinations of features in a translation-invariant way.

Convolutional neural networks in particular, have outperformed our other models in achieving top accuracy for these image classification tasks. Across all the classifiers, cross validation and random sampling can help reduce overfitting as well as regularization. Learning rates and hyperparameters like gamma and k must be adaptive enough to avoid local optima yet avoid slipping out of hill climbing.

Regardless, another important and core point to make is that the MNIST data set is a simpler and more basic data set than the CIFAR dataset, so in the grand scheme of things it is much more valuable and interesting to see which models performed very well on the CIFAR-10 dataset. The fact that some methods worked better on certain data sets while others didnt is a profound and useful fact as it can help developers design experiments and datasets effectively especially for eager learners.