ENEE 439M: Machine Learning

PROJECT-2

December 11, 2019

Shreya Gummadi
UID: 116195799
University of Maryland College Park
M.Engg in Robotics

CONTENTS

1 Pre Classification

1.1 Data Extraction

The MNIST dataset has 70,000 data samples. Each image in the dataset has the dimension 28x28, which is unrolled into a vector of dimension 784. Therefore we have a dataset of 70000x784 to build our classifiers with.

1. Support Vector Machine Classifier
   For SVM and Kernel SVM classifier, fetch_openml was used to get the MNIST data. The data is split into train and test using train_test_split module from sklearn. Using this module, gives us the option to play with the percentage splits for test and train.
2. Convolutional Neural Network
   For CNN, keras is used to load the dataset. This loaded dataset is already split into train and test sample. The training consists of 60,000 samples and testing consists of 10,000 samples. This is on par with the MNIST database available at http://yann.lecun.com/exdb/mnist/.

1.2 Dimensionality Reduction

Due to the large number of samples and high feature dimensionality, training the classifier can be time consuming. To speed up the process we can perform dimensionality reduction on the dataset.

1.2.1 Principal Component Analysis (PCA)

To reduce the dimensionality of the feature space we perform Principal Component Analysis (PCA). The goal of dimensionality reduction is to find a linear subspace of the feature space that has a reduced dimensionality.

In this project, PCA is performed using the sklearn module for the same. Using this module, we can define the number of feature components we want to keep.

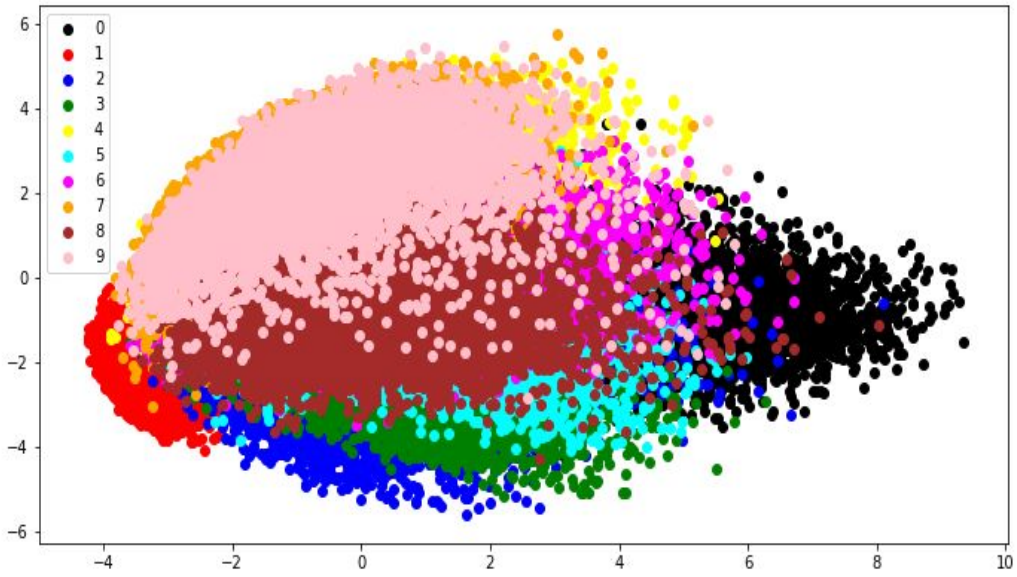

Fig 1. Projecting the data along 2 significant Principal Components.

1.2.2 Fischer Linear Discriminant (LDA/MDA)

LDA can also be used as a tool for data reduction. LDA also aids classification as projecting the dataset increases class separation and reduces the within class variance.

In this project, LDA is performed using the sklearn module for the same. Using this module, we can define the number of feature components we want to keep. It is suggested that the number of components is less than or equal to 9, which is the number of classes. Choosing a higher number of components throws a warning but still proceeds to perform LDA.

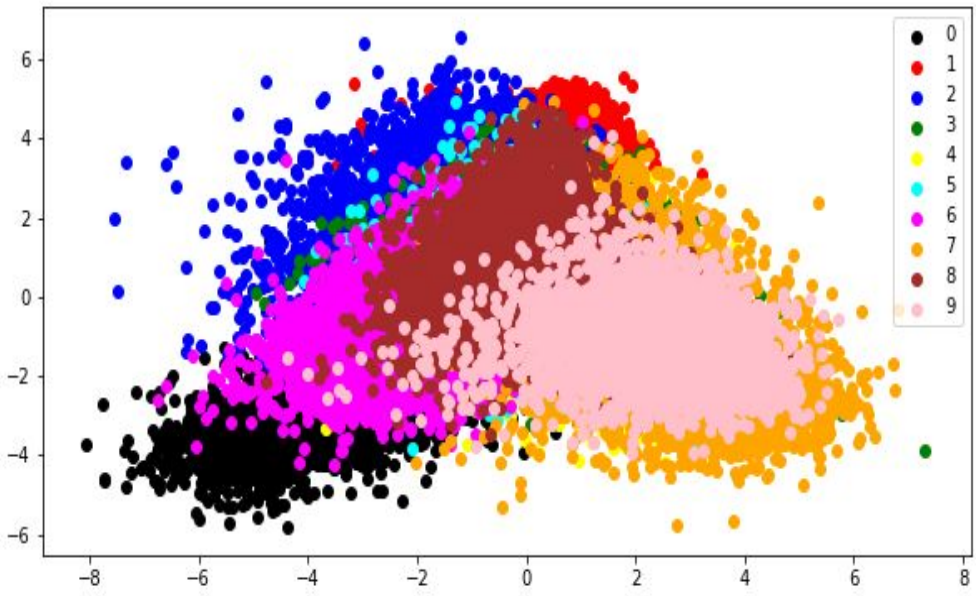

Fig 2. Projecting the data along 2 dimensions using LDA.

2 Classification

2.1 Support Vector Machine

The SVM classifier uses hyperplanes to separate classes from feature vectors. It tries to maximize the separation in order to get good generalized separation. The points closest to the hyperplane form the support vectors and are used to determine the separator.
It is formulated as a convex optimization problem, as below:

$$\min_{b,\mathbf{w}} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{subject to:} \quad \min_{n=1,\ldots,N} y_n(\mathbf{w}^T\mathbf{x}_n + b) = 1$$

SVM is usually used to classify two classes by forming a linear classifier between the two. However, it can be extended to multi class by redefining the problem to separate one class from the rest.

2.2 Kernel Support Vector Machine

Kernel SVM is an extension of SVM. It takes a non-linear transform to get a more general decision boundary. Kernel SVM is used in case of non linearly separable data. It projects the non linearly separable data and projects it to a higher dimension such that points belonging to each class are projected to a different dimension.
In this project we use the polynomial and radial basis function (RBF) kernels. These kernels has hyperparameters that can be tuned to get better results.

1. Polynomial Kernel parameter: Degree of the polynomial.
2. RBF Kernel parameter: C and gamma, where C is a regularization parameter that represents a tradeoff between correct classification and maximization of the decision margin. Gamm determines how far the influence of one sample reaches.

2.3 Convolutional Neural Network

A convolutional neural network takes as input images and learns various aspects of the object and then uses them to differentiate one from the other.
In general CNN has three types of layers:

1. Convolutional layer: The convolutional layer consists of feature maps which map the previous layer to the current layer with certain weights.
2. Pooling layer: After each convolutional layer, the pooling layer performs subsampling by picking a single output from a block. Max pooling takes the largest sample in the block to output.
3. Fully connected layer: After passing through several convolutional and pooling layers, the fully connected layer is placed. It takes hidden units from the previous layer and connects it to each unit in this layer.

2.3.1 CNN Architecture

The architecture used for this project is as follows:

Layer 0 : input
Layer 1: Convolutional layer with 32 filters of size 5x5
Layer 2:MaxPooling layer of 2x2
Layer 3: Convolutional layer with 64 filters of size 5x5
Layer 4: MaxPooling layer of size 2x2
Layer 5: Fully connected layer with 1000 nodes.
Layer 6: Output layer with 10 hidden units.
The architecture uses dropout before layer 5 and 6 to avoid overfitting.

3 Results

3.1 SVM

Train-test split	With PCA (dim =9)	With LDA (dim =9)
60-40	0.829	0.8995
70-30	0.833	0.901
85-15	0.835	0.901

Table 1. Effect of Training Size.

Train-test split for the two experiments below = 85-15

Dimension	PCA
9	0.835
100	0.944
500	0.941

Table 2. Effect of feature dimension.

3.2 Kernel SVM

For these experiments, we do PCA and reduce the dimensionality to 100.

3.2.1 Polynomial Kernel

Polynomial Degree	Accuracy
2	0.9769
4	0.9326
8	0.189

Table 3. Effect of Hyperparameter: Degree

3.2.2 RBF Kernel

C	Gamma	Accuracy
0.1	0.001	0.9147
5	0.05	0.9857
10	0.1	0.9761

Table 4. Effect of Hyperparamter: C and gamma.

3.3 Convolutional Network

A. Batch size = 128, Epochs = 5, Dropout = Yes

Optimizer	Accuracy
Adam Optimizer	0.9924
Stochastic Gradient	0.9658
RMSprop	0.9925

Table 5. Effect of Optimizer.

B. Batch size = 128, Epochs = 5, Optimizer = Adam

Dropout	0.9924
No Dropout	0.9908

Table 6. Effect of dropout.

C. Epochs = 5, Optimizer = Adam, Dropout = Yes

Batch Size	Accuracy
30	0.9929
128	0.9924
500	0.9921
1500	0.9895

Table 7. Effect of batch size.

4 Discussion

4.1 SVM

From the above experiments, we can infer the following:

- For the same dimension (dim =9), LDA outperforms PCA as it separates out classes.
- As we use more data samples for training, the accuracy increases.
- The scikit learn only allows LDA upto (number of classes -1) i.e. 9 in our case. Feature dimension 9 gives the best result with LDA. Increasing dimension (>9) throws a warning, but at the same time doesn't increase the accuracy.
- As we use more feature dimensions while using PCA we get better accuracy until a certain level.

Using PCA with feature dimension 100 and train-test split = 85-15 gives the best accuracy for SVM of 94.4.

4.2 Kernel SVM

From the above experiments, we can infer the following:

- For polynomial kernel, we get high accuracy for degree =2. For a higher polynomial degree, the classifier performs badly.
- For the RBF kernel, we get the best results with an optimal value of C and gamma. With too low or high values we lose on accuracy.

Tuning these hyperparameters is important to get good results. To select RBF kernel parameters we can also use grid search.

4.3 CNN

From the above experiments, we can infer the following:

- Batch size helps speed up the time for each iteration, but overall has no major effect on the accuracy.
- Dropout helps avoid overfitting and thus gives better results.
- Adam and RMSprop optimizers outperform Stochastic Gradient Descent and give better accuracy.

Thus using Adam optimizer with 5 epochs, batch size 128 and dropout gives us an accuracy of 99.24 which is on par with state of the art models.