Pattern Recognition and Neural Networks Assignment - 1 Report

Apoorv Pandey, Kaustuv Ray, Samrat Yadav, Pratyush Gauri
IISc Bangalore

In this Assignment, we have implemented different primitive ML Algorithms on Binary Classification problem, Multi-Class Classification Problem, Bounding box regression Problem, Frame classification on audio data, Generative Models respectively as described in our explanation from Q1 to Q5.

I. Binary Classification problem

Here we consider a 2 class classification problem on image data. The dataset is PneumoniaMNIST which contains images from chest X-rays belonging to two classes- Normal and Pneumonia.

A. Hyperparameters Tuning

In K nearest neighbor algorithm, we varied K from 1 to 50.The best K I obtained was K = 10. Since it was taking too much time in a single K nearest neighbor prediction therefore we did not experiment beyond K = 50.
We tuned the regularization rate from 0.001 to 0.1 in Logistic Regression with L1, L2 and Elastic loss. We found 0.001 as the best lambda for all the 3 models. We also tuned L1 ratio from 0.1 to 0.9 in Elastic Net and found 0.9 as the best L1 ratio.
We tuned the learning rates in logistic regression from 1e-3 to 1e-5. The number of iterations required to converge increased as the learning rate decreased without affecting the model accuracy. Therefore we chose 1e-3 as our learning rate.

B. Observations

We fitted the class conditional densities using a
1. Mixture of Gaussians(3 components)
2. Just a single Gaussian
3. Gaussian Parzen window estimate.
The feature vector dimensionality was quite high(784) and doing a Gaussian density estimate led to the same prediction on all inputs. To avoid the curse of dimensionality I tried reducing the dimensionality of the input vector using Principal Component Analysis. Parzen Window, Gaussian MAP, and GMM density estimation all require fitting Gaussians onto the data. Hence for these algorithms, I used the transformed data using PCA to 150 components. Still there was no improvement in accuracy.

C. Training Plot for Logistic Regression

In the training plot we see some spikes which indicate that the learning rate was high enough in that region and so it took a larger step increasing the loss. The Gaussian Naive

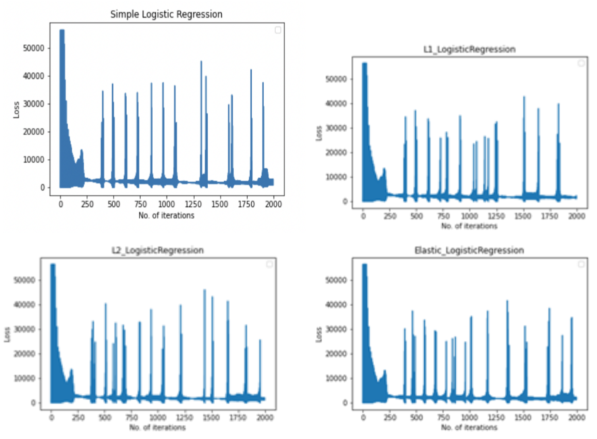

Fig. 1: Training Plot for Logistic Regression

Bayes Model performs quite well on this dataset. Therefore assuming features of each training vector are independent and come from a Gaussian distribution seems to be correct. Since logistic regression performs as well as GNB when data is Gaussian therefore the high accuracy of Logistic Regression also makes sense. Logistic Regression seems to outperform L2 regression which outperforms L1 and Elastic Net regressions. Therefore it seems reasonable that the best lambda for each of these methods is very low(0.001). The KNN algorithm took too much time compared to algorithms since in each iteration it computes distance to all neighbors. Parzen Window, Gaussian MLE and GMM classification predict same label on all test points. Fitting 784-dimensional Gaussian vectors is not correct since in high dimensions Gaussian is a heavy tailed distribution. This clearly demonstrates curse of dimensionality.

II. Multi-Class Classification Problem

In this task, we look at an 8-class classification problem on Blood Cell Microscope images. The dataset BloodMNIST is used here.

A. Hyperparameter tuning

In K nearest neighbor algorithm, we varied K from 1 to 10.The best K we obtained was K = 1. Since it was taking too much time in a single K nearest neighbor prediction therefore we did not experiment beyond K = 10.
We tuned the regularization rate from 0.001 to 0.1 in Multi Class Logistic Regression with L1, L2 and Elastic loss. I found 0.1 as the best lambda for L1,L2 and Elastic Net. We

TABLE I: Binary Classification problem Observations

Algorithm	Logistic Regression	L1 Regression	L2 regression	ElasticNet	Gaussian Naive Bayes	LDA	Parzen Window	Gaussian MAP	GMM	K Nearest Neighbor(K=10)
Accuracy	0.858	0.826	0.852	0.849	0.836	0.836	0.625	0.625	0.625	0.838
F1	0.895	0.876	0.892	0.890	0.872	0.882	0	0	0	0.883
AUC	0.822	0.773	0.810	0.804	0.817	0.789	0.5	0.5	0.5	0.792

TABLE II: Multi-Class Classification Problem Observations

Algorithm	Softmax Regression	L1 Regression	L2 regression	ElasticNet	Gaussian Naive Bayes	LDA	Parzen Window	Gaussian MLE	GMM	K Nearest Neighbor(K=1)
Accuracy	0.678	0.509	0.592	0.532	0.696	0.656	0.194	0.194	0.194	0.697
F1	0.629	0.410	0.544	0.423	0.672	0.643	0	0	0	0.477
AUC	0.651	0.505	0.587	0.477	0.687	0.671	0.125	0.125	0.125	0.720

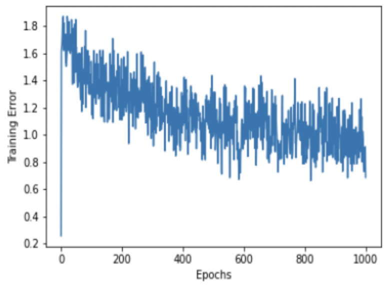

Fig. 2: Training Plot for Softmax Regression

also tuned L1 ratio from 0.1 to 0.9 in Elastic Net and found 0.1 as the best L1 ratio.
We tuned the learning rates in logistic regression from 1e-3 to 1e-5. The number of iterations required to converge increased as the learning rate decreased without affecting the model accuracy. Therefore we chose 1e-3 as our learning rate.

B. Discussion and Comparison

Softmax Regression without regularization is performing better than L1,L2 and Elastic Net. This is may be due to the fact that since training and testing distributions are very similar ,regularization increases the bias in the mode. Gaussian Naive Bayes is performing as good as Logistic regression which means that the assumption that features are conditionally independent and come from a Gaussian distribution is okay. LDA is also performing almost as good as Softmax regression which is expected since the data is continuous and not categorical. None of the generative density modelling(Gaussian MAP,GMM estimation,Parzen window) techniques are working well since they all fit high dimensional gaussians which leads to curse of dimensionality.

C. Challenges and Learning

- Implementing high dimensional multivariate Gaussian was giving NaN values since the probabilities were too low. Therefore I did not calculate directly the probability but instead calculated the log probability. Also calculating the determinant of the covariance matrix using np.linalg.det was giving NaN values so I calculated the log of the determinant using np.linalg.slogdet
- Anywhere where there was an overflow error I added a very small positive quantity to the operand to avoid NaN issues.
- While implementing the EM algorithm using a Gaussian Mixture Model I was getting a singular covariance matrix error. This was happening because for some parameter values the variance became very low and Gaussian collapsed to a single point. To overcome this issue instead of doing an MLE estimation I did a MAP estimation. Also, I added a very small value(1e-6) to the diagonal of the covariance matrix.

III. Bounding box regression Problem

In this problem, we are given images of traffic signs and the task is to find out the bounding boxes that encompass the sign in the image. The task is to take the input image and regress over the co-ordinates of the box. The metrics here are the mean MSE, mean MAE and mean Intersection over Union (mIoU). We have used the road-sign-detection dataset from kaggle for this problem.

A. Data Preprocessing

We have done the data preprocessing part in three steps which are shown below:-

- Transformed the image to Gray from RGB
- Normalized the pixel values
- Resized all the images to (100,100) images and accordingly changed the bounding boxes

B. Hyper Parameter tuning

Learning rate Tuning- We tuned from 1e-3 to 1e-7. I kept the learning rate as 1e-7 as below this rate loss started increasing without bound.
Regularization rate Tuning- We tuned the regularization rate from 0.1 to 0.001 for both L1, L2 and Elastic Net. The best regularization found was 0.1 for all 3 algorithms.
L1 ratio Tuning - We tuned L1 ratio from 0.1 to 0.9(step size 0.1).We found 0.9 as the best L1 ratio. This is depicted in Table IV

TABLE III: Bounding Box Regression Problem Observations

Algorithm	Simple Linear Regression	L1 Regression (lambda = 0.1)	L2 Regression (lambda=0.1)	Elastic Net (lambda=0.1, L1 ratio = 0.9)
MSE	20.378	20.368	20.378	20.369
MIoU	0.0	0.0	0.0	0.0
MAE	4.745	4.744	4.745	4.744

TABLE IV

Algorithm	L1 Regression [MSE,mIoU, MAE]	L2 Regression	Elastic Net(L1 ratio = 0.9)
Reg-Rate = 0.001	[20.378,0.0,4.746]	[20.382,0.0,4.746]	[20.378,0.0,4.746]
Reg-Rate = 0.01	[20.374,0.0,4.745]	[20.380,0.0,4.746]	[20.375,0.0,4.745]
Reg-Rate = 0.1	[20.368,0.0,4.744]	[20.378,0.0,4.745]	[20.369,0.0,4.744]

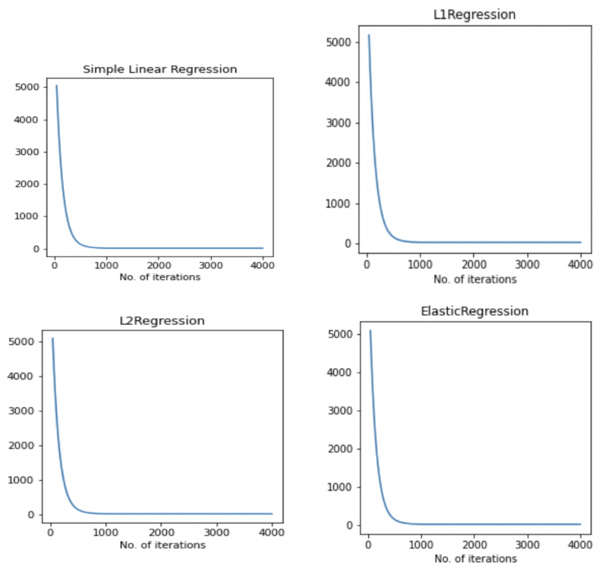

Fig. 3: Training Plots

C. Comparison

All the algorithms perform very poorly on the dataset and mIoU is 0 for all of them. There is not much effect of tuning hyper parameters and the decrease in MSE is very small. Also the performance of all the algorithms is same. This is expected since the data is an image on which typically CNN's are employed and these linear models fail to capture spatially locality of pixels in the image.

D. Challenges and Learning

- The loss is very sensitive to choice of learning rate parameter. It increases if learning rate parameter is not chosen properly.

IV. Frame Classification on Audio Data

In this problem, the task is to classify every sample of a speech/audio signal. We use the TIMITdataset for this purpose. The task here is to classify every sample of the utterance to be belonging to vowel or not vowels. The ground truth information has to be generated from the .phn file that accompanies every .wav file. It lists the phonemes corresponding

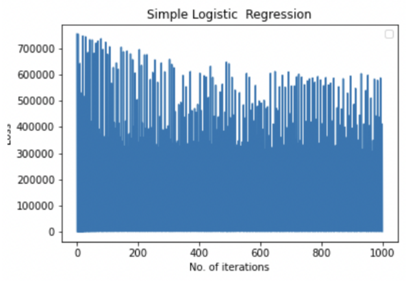

Fig. 4: Training Plot

to time intervals in the utterance. Eg- 0 3050 h 3050 4559 sh 4559 5723 ix 5723 6642 hv 6642 8772 eh 8772 9190 dcl 9190 10337 jh 10337 11517 ih 11517 12500 dcl. Vowels are phonemes that contain /a,e,i,o/ and u in them.The metrics are average true positive, average true negative, average false positive and average false negative.

A. Data Preprocessing

We have transformed the audio and used MFCC's as the feature input x. Also in order to ensure all inputs have same length I used zero padding on MFCC.

B. Hyperparameter tuning

We tuned the regularization rate from 0.001 to 0.1 in Logistic Regression with L1, L2 and Elastic loss. We found 0.001 as the best lambda for all the 3 models. We also tuned L1 ratio from 0.1 to 0.9 in Elastic Net and found 0.1 as the best L1 ratio. We used overall accuracy to determine the best hyperparameter.
We tuned the learning rates in logistic regression from 1e-3 to 1e-5. The number of iterations required to converge increased as the learning rate decreased without affecting the model accuracy. Therefore we chose 1e-3 as my learning rate

C. Comparison

The training plot of logistic regression shows that the loss is not strictly decreasing but on an average decreases. This indicates that the data is not linear.
While tuning lambda for all the 3 linear models we found that on increasing lambda model started to output more positives

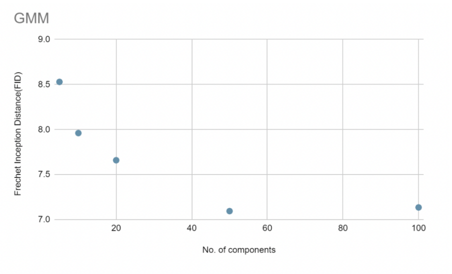

Fig. 5: GMM Curve

thereby increasing true positives and decreasing true negatives. It seems that regularizing the model seems to be biased towards outputting positive predictions.
Gaussian Naive Bayes seems to output only negatives which indicates that we can't treat features conditionally independent of each other. This also makes sense since in an audio sequence there is temporal dependence among MFCC samples.
LDA is also performing quite well and gives similar performance as logistic regression.
None of the unsupervised density modeling techniques are working well since they all fit high dimensional Gaussians which leads to the curse of dimensionality.

V. Generative Models

In this module, we build generative models on Tinyimage net. The dataset contains 200 classes in total and each class contains 500 images. For memory and time constraints, we will include 20 classes and 250 images for each class in our data.

A. Data preprocessing

- Resize the image from 64x64x3 to 28x28x3
- Flattened the image
- Converted from RGB to Gray
- Normalized the pixel values

We fit the GMM to the data for different values of number of components. Then we sampled 1000 images from the GMM. Some image from each GMM are show in Fig.6,7,8,9,10. We used Frechet Inception Distance between the generated images and the real images as a metric of evaluation.

B. Comparison

From the above plot(Fig. 5) we can see that the Frechet Inception Distance(FID) decreases as the number of components increases till 50. Then for 100 components it starts increasing. We get the best FID for 50 components.

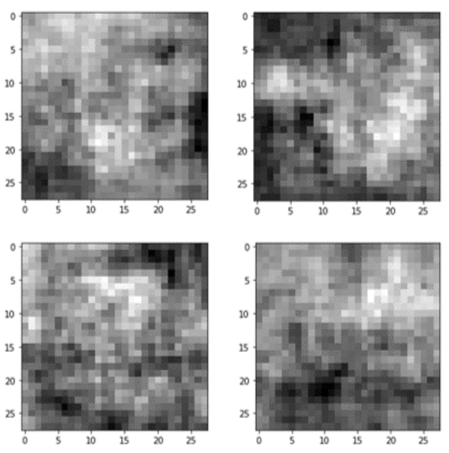

Fig. 6: Generated images from GMM for 5 components

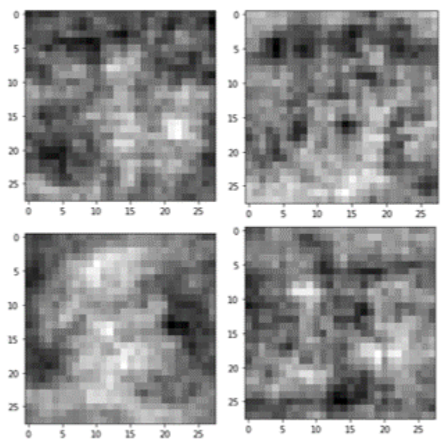

Fig. 7: Generated images from GMM for 10 components

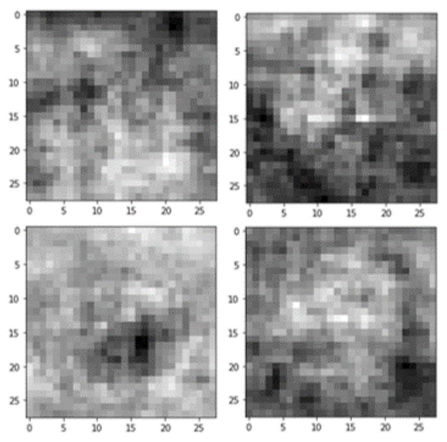

Fig. 8: Generated images from GMM for 20 components

TABLE V: Frame Classification on Audio Data Observations

Algorithm	Logistic Regression	L1 Regression	L2 Regression	ElasticNet	Gaussian Naive Bayes	LDA	Parzen Window	Gaussian MLE	GMM
True Positive	0.554	0.594	0.541	0.620	0	0.548	0	0	0
False Positive	0.117	0.158	0.088	0.189	0	0.082	0	0	0
True Negative	0.235	0.194	0.264	0.163	0.353	0.270	0.353	0.353	0.353
False Negative	0.093	0.053	0.105	00.026	0.647	0.099	0.647	0.647	0.647

TABLE VI: FID Observation for different no. of components

No. of components	Frechet Inception Distance(FID)
5	8.52926
10	7.96066
10	7.96066
50	7.09428
100	7.13524

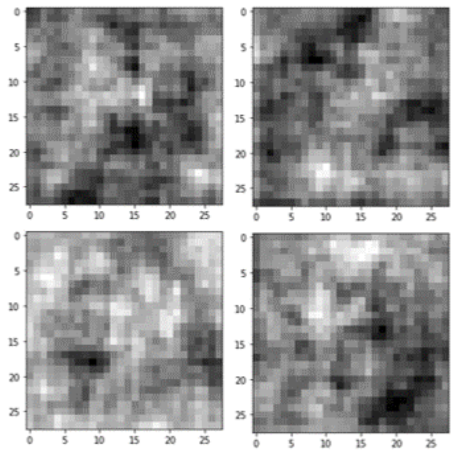
Fig. 9: Generated images from GMM for 50 components

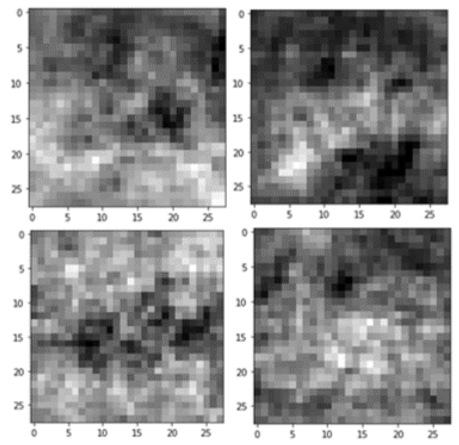
Fig. 10: Generated images from GMM for 100 components