FTE 4560 Group Project

1.Qingwen Deng 2.Jingyu Xu 3.Yifan Shen
4.Yufeng Yang 5.Zhitong Lin
(*Sort according to alphabetical order)

Contents

1. **Abstract (Page2)**
2. **Introduction (Page 2)**
3. **KNN & LDA methods for classification (Page 2-5)**
4. **Softmax Regression methods for classification (Page 6-9)**
5. **Neural Network methods for classification (Page 9-11)**
6. **Final Conclusion (Page 11)**

Part 1: Abstract

Contributor: Yufeng YANG 117010351

Nowadays, with the increasing demand for classification problem in the areas of Computer Vision, Financial Engineering, many methodologies have been proposed to handle these data. In this project report, state-of-art methods like KNN& LDA, Softmax Regression. The task for each method is applied to two datasets. One is Yale Faces' dataset, another is bankruptcy dataset. In conclusion, we found as the model complexity increases, the accuracy is in general better. For parameter model like soft-max regression and neural network, with the assistance of appropriate hyper-parameter for regularization, the accuracy on these two datasets also increases. However, we found with the increase of data input may not increases the accuracy. The reason we deduce is due to the dissimilarity between training samples decreases the accuracy.

Part 2: Introduction

Contributor: Yufeng YANG 117010351

In this report, several state-of-art methods are used on two datasets: 1. Yale Faces Dataset, 2. Bankruptcy Data. All data is scaled both in size and image and vectorized in matrix form. The reported for experiment is divided by 3 parts: 1. KNN and LDA+ KNN; 2. Softmax Regression; 3. Neural Networks. Each experiment gives accuracy for test data after fitting the models. Especially, for KNN method, the author (XU) considers the effect of normalizing, which concludes that the test accuracy improves a lot after scaling. Thus, all of the following methods adopts data scaling during training. Also, for Softmax Regression experiment, the author (YANG) gives mean and variance of each label after 10 times' repetition. For neural network models, the authors (DENG, LIN, SHEN) construct neural network from using numpy packages and compare the effect of numbers of hidden layer nodes for accuracy. The following is the detailed report for different classification methods on the 2 datasets.

Part 3: KNN & LDA

Contributor: Jingyu XU 118020066

I. Methodology of KNN & LDA

KNN is a non-parametric method used for classification and regression. KNN output a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). For example, if k = 1, then the object is simply assigned to the class of that single nearest neighbor.

Consider the input of KNN and it's algorithm characteristics, normalizing the training data can improve the accuracy dramatically. Since this algorithm relies on

distance for classification, if the features represent different physical units or come in vastly different scales, then normalizing is of vital importance, or the result will be greatly disordered.

For high-dimensional data (e.g., with number of dimensions more than 10) dimension reduction is usually performed prior to applying the k-NN algorithm in order to avoid the effects of the curse of dimensionality. The curse of dimensionality in the k-NN context basically means that Euclidean distance is unhelpful in high dimensions because all vectors are almost equidistant to the search query vector (imagine multiple points lying more or less on a circle with the query point at the center; the distance from the query to all data points in the search space is almost the same).

Feature extraction and dimension reduction can be combined in one step using principal component analysis (PCA), linear discriminant analysis (LDA), or canonical correlation analysis (CCA) techniques as a pre-processing step, followed by clustering by KNN on feature vectors in reduced-dimension space. This process is also called low-dimensional embedding.

LDA produces linear decision boundary using the estimated mean and variance from a normal distribution. LDA assumes that the observations are drawn from the normal distribution with common variance in each class, while logistic regression does not have this assumption.

LDA would do better than Logistic Regression if the assumption of normality hold, otherwise logistic regression can outperform LDA

KNN is completely non-parametric: No assumptions are made about the shape of the decision boundary.

Advantage of KNN: We can expect KNN to dominate both LDA and Logistic Regression when the decision boundary is highly non-linear

Disadvantage of KNN: KNN does not tell us which predictors are important (no table of coefficients)

II. Experiment on Bankruptcy Data

The bankruptcy data is about bankruptcy prediction of Polish companies. The dataset has 64 features (numerical information of the companies) and 2 classes (0 or 1). The data have already been split into two subsets, of which the larger one (949 samples) is for training and the smaller one (151 samples) is for testing. So you only need to report a single result. Do not need to perform multiple trials and report mean and standard deviation. See the CSV files. the last column in each of the files denotes the class label, 0 for “not bankrupted” and 1 for “bankrupted”.

For the missing values in both training set and testing set (denoted "?"), the values are replaced by the mean of the variable.

Following is the result of KNN and KNN+LDA without normalizing the training and testing data.

Model	K=1	K=3	K=5	K=7	K=9
KNN	0.7350993377	0.7152317880	0.7284768212	0.7284768212	0.7218543046
LDA+KNN	0.5231788079	0.5231788079	0.4304635762	0.3708609272	0.3178807947

K=1 provides the best result for both KNN and KNN+LDA.

Following is the result of KNN and KNN+LDA after normalizing the training and testing data.

The normalization is done by Z-score. The method is based on the mean and variation of the raw data, applicable to various kind of original data (thus being assigned as the default method by many tools). The data after processing is 0 mean 1 variation normal distribution. However, the possible drawback is that it may changes the original data structure.

Model	K=1	K=3	K=5	K=7	K=9
KNN	0.5827814569	0.7019867549	0.7019867549	0.7152317881	0.7284768212
LDA+KNN	0.5165562913	0.5827814569	0.5894039735	0.5894039735	0.5894039735

K=5 provides the best result for KNN+LDA, while the prediction accuracy of KNN goes up with the rise of k.

The general prediction accuracy for the two models after normalizing the data has a noticeable improve especially for KNN+LDA.

The possible reason is that, the structure of the raw data is not suitable to use LDA as the dimension reduction method. LDA performs well when the original data is clustered by mean. The result of KNN outperforms LDA may also suggest that the classification boundary of the data is highly non-linear. The improve of performance after normalization also verifies the high standards LDA imposed on the base data.

III. Experiment on Yale faces Database

Yale Face Database contains 165 grayscale images in GIF format of 15 individuals. There are 11 images per subject, one per different facial expression or configuration: center-light, w/glasses, happy, left-light, w/no glasses, normal,

right-light, sad, sleepy, surprised, and wink. Here are two different versions. The size of all images is 320x243.

Pre-processing:

- Rename the images. The images are rename in the format of '01'. The produced data form of the images are in dictionary type, with image names as keys.
- Rescale the images. Since the raw images are quite big, the images are resized to a smaller size, 0.2 of the original width and height or 64x48.
- Rescale the image pixel value. Scaled into [0, 1] for convenience.
- Organize the images into matrices. For each subject, vectorize every image into a column vector and form a matrix of size d×165.

The following 2 tables are the result of classification by KNN and LDA+KNN respectively.

	KNN				
# per subject	K=1	K=3	K=5	K=7	K=9
4	0.847619	0.838095	0.752381	0.752381	0.704762
8	0.955556	0.977778	0.955556	0.955556	0.955556

	LDA +KNN				
# per subject	K=1	K=3	K=5	K=7	K=9
4	0.704762	0.628571	0.714286	0.676190	0.666667
8	0.6	0.4	0.555556	0.355556	0.511111

Looking inside each of the tables. For KNN model, intuitively speaking, the more subjects put in training, the more accurate the result will be. The experiment result verified the intuition, as we can see the significant accuracy rise after putting more samples for each subject into training. However, as for the LDA+KNN model, the rise of training samples make the accuracy of the result suffered a descent. The possible reason may be in the process of dimension reduction, the exceed samples affected the training process just like the effect of overfitting.

Parallel comparing, KNN model still outperform LDA+KNN, and it also has a tendency that the models are likely to perform better when k is small. This maybe the case that the raw data has a sufficient density, and when k gets larger, the exceed neighbours will disturb the classification outcome since in this model we didn't assign weights to the neighbours. And the LDA result was also greatly influenced by the original characteristics of the raw data, thus provides unsatisfactory result.

Part 4:Softmax Regression Report

Contributor: Yufeng YANG 117010351

1. Methodology of Softmax Regression

For multi-class classification problem, one classical method is called softmax regression, which is the general form of logistic regression. In this part, the construction of softmax regression and optimization method used for solving softmax regression will be introduced.

The kth classifier is defined as:

$$C_i: z^j = \theta_j^T x$$

and calculate the probability of some classifier belonging to class i:

$$P(y = i|x, \theta) = \frac{\exp(z^j)}{\sum_{j=1}^{k} \exp(z^j)}$$

We aim to maximize the probability that classifier i belongs to class i. Thus, with the assistance of maximum likelihood function the objective function is defined as:

$$\max L(\theta) = \prod_{l=1}^{m} P(y = i|x, \theta)$$

which is equivalent to:

$$\min J(\theta) = -\log(L(\theta))$$

The objective function can be simplified as:

$$\min\ J(\theta) = \min\ \frac{1}{m}\sum_{1}^{m}(\log(\sum_{1}^{k}\exp(z^j)) - z^{label})$$

Taking gradient of objective function, we can get:

$$\frac{\partial J(\theta)}{\partial \theta^j} = \begin{cases} p(y = j|x;\theta)\cdot x, & j \neq label \\ (p(y = j|x;\theta) - 1)\cdot x, & j = label \end{cases}$$

Then, the gradient descent algorithm can be described as follows:

$$\theta_j = \theta_j - \alpha\frac{\partial J(\theta)}{\partial \theta_j}$$

In this experiment, the step size is always restricted to 0.01.

In reality, in order to improve the estimation performance, regularization term is introduced in the objective function. Thus, the modified objective function is defined as:

$$J(\theta) = \frac{1}{m}\sum_{1}^{m}(\log(\sum_{1}^{k}\exp(z^j)) - z^{label}) + \lambda||\theta||^2$$

Thus, the modified optimization algorithm will be:

$$\theta_j = \theta_i - \alpha(\frac{\partial J}{\partial \theta_j} + \lambda\theta_i)$$

2. Essential Data Processing before Experiment

First, reading the dataset Yale Faces, and then read the file name and get the label, which is from class 1 to 15. In my report, in order to be convenience, I scale the image to 100×100. For pixel value, I scale it into the interval [0,1] and then use packages to read the images and labels into matrix form and vector respectively.

Then, I spilt the dataset into train and test dataset. For Yale Faces dataset, I randomly choose 4 or 8 images from 11 images in each subject to do repeated

experiment. For Bankruptcy data, I first fill up the missing value in dataset. Because the data has already partitioned into train and test data. Here, I will not spilt them for convenience.

After splitting Yale Faces data and Bankruptcy data, we also need to scale the matrix data and transforming vector y into an indicator matrix. The widely used data scaling method is:

$$\hat{X} = \frac{X - \bar{X}}{\sigma(X)}$$

where $\bar{X}$ is the mean vector and $\sigma(X)$ is standard deviation. For Yale Faces dataset, the input matrix form is 165x10000, each row represents one image's information, thus scaling is operated on each row. However, I found it will always report error in logarithm operation, which is because of the antilogarithm number is close to 0. Thus, for Yale Faces dataset, I directly delete the numerator part of standard deviation. For bankruptcy data, because each column represents one variable, thus the scaling operation is operated on each column. During scaling, the standard scaling method is used.

As for labels, the input form is the index of each images. In order to convenience the computing, we need to transform the label vector into a matrix. The following is the procedure for doing this:

i.e.: y= $[0,1,2,3,4,5]^T$

After transforming, the matrix y will be a 5x5 matrix like this:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Under this way, each row and column can be a probability indicator for each label. Then the calculation of gradient can be formulated in matrix form:

Y=$[1(y = 1), 1(y = 2), \dots 1(y = k)]^T_{k \times m}$

P=$[P(y = 1|x_i, \theta) \dots P(y = k|x_i, \theta)]^T_{kxm}$

Then the matrix form is gradient is:

G=$\frac{1}{m}(P - Y)X^T$, where X is m×n matrix.

3. Experiment on Yale Faces and Bankruptcy Data

The target for this experiment is to test accuracy of softmax regression to classification tasks. Before the experiment, I will give the definition of accuracy I used during this experiment as follows:

$$\text{Accuracy} = \frac{\#of\ corrected\ estimated\ labels}{\#\ of\ total\ labels}$$

Also, the mean is the estimated mean of labels. The true value is equal to the index. Variance is the variance for repeated (10 times) estimation of each label. During the 10 repeated experiments, 5 different hyperparameters for regularization are repeated 2

times each. Thus, the Table 3-2 is the overall estimation of mean and variance of each label.

The following table is the experiment data for Yale Faces Database:

Table 3-1, Softmax Classification Accuracy for Yale Faces Database

	Softmax Classification Accuracy for Yale Faces Database				
# of training image	$\lambda = 0$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1$
4(Accuracy)	58.1%	67.6%	76.2%	72.4%	73.3%
8(Accuracy)	48.9%	80%	66.7%	64.4%	68.9%

Table 3-2, Estimated Mean and Variance for Each Subject
(Each Subject with 4 train images)

# Sub=4	Sub1	Sub2	Sub3	Sub4	Sub5	Sub6	Sub7	Sub8	Sub9
Mean	2	4.53	4.11	4.77	6.71	6.9	6.93	8.14	8.95
Variance	7.26	21.84	11.01	7.58	11.33	15.11	0.35	1.70	9.36
	Sub10	Sub11	Sub12	Sub13	Sub14	Sub15			
Mean	9.33	10.37	10.34	12.70	11.00	12.17			
Variance	7.51	7.00	10.12	9.01	23.84	21.94			

Table 3-3, Estimated Mean and Variance for Each Subject
(Each Subject with 8 train images)

# Sub=8	Sub1	Sub2	Sub3	Sub4	Sub5	Sub6	Sub7	Sub8	Sub9
Mean	2.57	6.67	6.3	6.3	4.93	8.83	7.5	8.67	9.5
Variance	14.97	26.22	25.96	19.74	1.59	19.47	3.05	4.02	8.98
	Sub10	Sub11	Sub12	Sub13	Sub14	Sub15			
Mean	10.03	11	11.56	9.96	8.9	11.57			
Variance	5.76	0	13.499	10.46	28.82	21.05			

The conclusion from the table above is that regularization term generally improves the accuracy. However, without an appropriate choice for hyperparameter, it will not improve a lot. In experiment, we can estimate the hyperparameter near 0.001 and 0.01 will give us optimal result.

As for mean and variance, I conclude that softmax regression gives some labels with very perfect estimation. However, for subject 1, 2, 3, 13, 14, 15, the estimation performance is very bad. Increasing the size from 4 images each class to 8 images doesn't improve the estimation performance.

The table 3-3 and 3-4 are the experiment about bankruptcy data. The accuracy, mean and variance are defined in the same as experiment for Yale Faces. Similarly, these 5 hyperparameters for regularization are repeated 2 times each. Thus, the table 3-4 is the overall estimation of mean and variance.

Table 3-3, Softmax Regression Accuracy for Bankruptcy Data

	Softmax Regression Accuracy for bankcruptcy data				
	$\lambda = 0$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1$
Accuracy	84.1%	84.1%	84.2%	84.1%	82.8%

Table 3-4, Mean and Variance Estimation of Labels 0 and 1

	Class: 0	Class: 1
Mean (of integer label)	0.38	0.73
Variance (of integer label)	0.24	0.19

The conclusion from the table above is that the test accuracy for binary labels is much higher than multi-class task. Similarly, compared with experiment 1 for Yale Faces, regularization generally improves experiment accuracy. However, if the hyperparameter is too large, the accuracy cannot even surpass the accuracy without regularization in objective function. Thus, a well-chosen hyperparameter is important for improving experiment accuracy.

From the estimated mean and variance for 0 and 1, I conclude that for binary label estimation, the result is closer to the true label. Also, the variance is small, which indicated the wrong value is much less than the true value.

IV. Conclusion of Softmax Regression for Classification

From the experiment result, I conclude regarding on softmax regression problem, adding regularization term is essential for improving accuracy. However, a well-chosen hyperparameter is needed. From the experiment, the hyperparameter λ should be chosen among 0.001 and 0.01.

As we can see from above results, although softmax regression is designed for multi-class classification, it seems if number of classes for softmax regression task is too large, the classification may not perform well. The performance varies class by class. Hence, a more complicated model is needed to achieve a better accuracy.

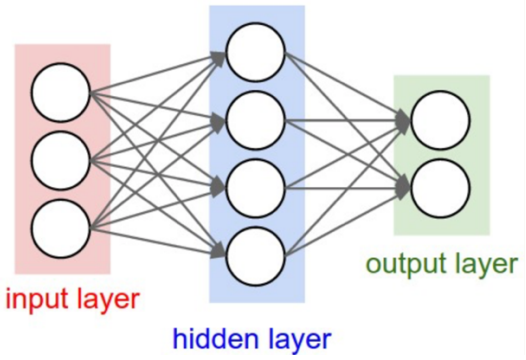
Part 5 Neural Network

Contributor: Qingwen Deng 117010049; Yifan SHEN 117020234; Zhitong LIN 118010178

Methodology of Neural Network

Neural network is a very popular method of data classification and image recognition. In this section, we used a simple neural network with one hidden layer to do the classification on Yale Face Database and Polish companies bankruptcy data. The theoretical knowledge and construction of neural network will be introduced first.

The fully connected neural network we constructed contains one input layer, one hidden layer and one output layer, the model is shown below:



The mathematical relation between layers can be defined as:

$$hidden\_inputs = inputs \times W_{ih} + B_{ih}$$
$$final\_inputs = hidden\_outputs \times W_{ho} + B_{ho} \quad (1)$$

where W is the weights between layers, B is the biases in the transformation.

The weights are initialized at first by randomization in normal distribution. The inputs in hidden layer and output layer are activated by the activation function. he activation function we chose is the sigmod function:

$$f(x) := sigmod(x) = \frac{1}{1 + e^{-x}} \quad (2)$$

With this function, we can derive the output function:

$$hidden\_outputs = sigmoid(hiddent\_inputs)$$
$$outputs = sigmoid\left(final_{inputs}\right) \quad (3)$$

The cost function is defined as

$$E = expected - output \quad (4)$$

The weights are updated by gradient decent under Back Propagation:

$$W_{ho} = W_{ho} - \eta \times \frac{\partial E}{\partial W_{ho}}$$
$$W_{ih} = W_{ih} - \eta \times \frac{\partial E}{\partial W_{ih}} \quad (5)$$

After calculating the gradient of the errors at the weights, the updated weights are:

$$W_{ho} = (1 - \frac{\eta\lambda}{n}) \times W_{ho} + output\_errors \times final\_outputs \times (1 - final\_outputs) \times hidden\_outputs$$
$$W_{ih} = \left(1 - \frac{\eta\lambda}{n}\right) \times W_{ih} + hidden_{errors} \times hidden_{outputs} \times \left(1 - hidden_{outputs}\right) \times inputs \quad (6)$$

where η is the learning rate in the updating process, λ is the parameter for the regularization or weight decay, n is the number of samples.

Classification on Yale Face Database

1. Construct the Neural Networks

Two 3-layer neural networks are constructed with the main difference in the number of neurons in the hidden layer. There are 64 * 48 = 3072 nodes, which is consistent to the image size, in the input layer and 15 nodes, which equals to the

individual number, in the output layer of each neural network. However, 100 nodes are defined in the hidden layer of neural network A (NNA) while 200 nodes are defined in the one of network B (NNB). The experiment is designed to test the accuracy and efficiency of the two neural networks.

2. Experiment Results

Neural Network A # of units in the hidden layer: 100

# of training image per subject	$\lambda = 0$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1$
4	76.19%	79.05%	82.86%	78.1%	74.29%
8	82.22%	86.67%	88.89%	77.78%	53.33%

Neural Network B # of units in the hidden layer: 200

# of training image per subject	$\lambda = 0$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1$
4	81.90%	80.95%	79.05%	80.00%	59.05%
8	88.89%	75.56%	86.67%	84.44%	48.89%

3. Conclusion

The Neural Network B shows a better performance in the accuracy of classification than the Neural Network A when the λ is not so large. The comparison indicates that the accuracy is improved by adding more nodes to the hidden layer to solve underfitting problem while the parameter for the weight decay is not very large. Besides, training more image in one subject may also increase the accuracy rate by keeping a low λ at the same time. Therefore, training many images of one subject in a neural network with large number of neurons in the hidden layer will get a result of high accuracy for solving the underfitting problem.

Classification on Polish companies bankruptcy data

1. Construct the Neural Networks

Two 3-layer neural networks are constructed with the main difference in the number of neurons in the hidden layer. There are 64 nodes, which is consistent to the number of attributes, in the input layer and 1 node, which indicates the bankruptcy, in the output layer of each neural network. However, 16 nodes are defined in the hidden layer of neural network A (NNA) while 24 nodes are defined in the one of network B (NNB). The experiment is designed to test the accuracy and efficiency of the two neural networks.

2. Experiment Results

Neural Network A # of units in the hidden layer: 16

bankruptcy	$\lambda = 0$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1$
accuracy	69.54%	72.19%	74.17%	76.82%	72.85%

Neural Network B # of units in the hidden layer: 24

bankruptcy	$\lambda = 0$	$\lambda = 0.001$	$\lambda = 0.01$	$\lambda = 0.1$	$\lambda = 1$
accuracy	74.17%	70.86%	72.85%	67.55%	71.52%

3. Conclusion

The Neural Network B shows a better performance in the accuracy of classification than the Neural Network A only when the λ equals 0. The comparison indicates that the accuracy is not improved by adding more nodes to the hidden layer while the parameter for the weight decay is larger than 0. The reason may be the overfitting problem due to too many neurons in the hidden layer. Therefore, training bankruptcy data in a neural network with large number of neurons in the hidden layer may cause overfitting problem, thus getting a result of low accuracy.

Part 6: Conclusion

Contributor: All team members

Through first experiment, we conclude that LDA method doesn't improves the classification result. However, with the improvement on training subject, the accuracy is improving. This phenomenon maybe the case that the raw data has a sufficient density, and when k gets larger, the exceed neighbors will disturb the classification outcome since in this model we didn't assign weights to the neighbors. And the LDA result was also greatly influenced by the original characteristics of the raw data, thus provides unsatisfactory result.

For Softmax Regression model, we conclude that only with the assistance of appropriate hyper-parameter for regularization can the model improve its performance. Increasing number of input data from each subject cannot improve the accuracy in general. We deduce it may be because of the complex data structure, which cannot be learned by such a simple model.

For Neural Network model, we conclude that increasing hidden units cannot improve the accuracy greatly. But similar as the result indicating in Softmax Regression, only with the appropriate hyper-parameter for regularization can improve the accuracy. Increasing numbers of input data from each subject also decrease the accuracy in general. Thus, we propose that image processing (like rotation, denoising) should be tried for further discussion. Also, some complex neural network like Generative Adversarial Networks and Contrastive Learning may also be a better solution.