Assignment 2: Clustering and Classification on MNIST Dataset

Zhang Xudong 12011923

January 3, 2024

Contents

1 Introduction

This assignment is based on the dimensionality reduction, classification and clustering. In this assignment, what have been implemented is the dimensionality reduction using PCA (Principal Component Analysis) and LDA (Linear Discriminant Analysis), clustering using K-Means and hierarchical clustering on a subset of MNIST dataset. Also, two-class classification in a 5-fold cross validation setting using SVM with a RBF kernel, SVM with a linear kernel and a neural network classifier with one hidden layer has been implemented on this subset of MNIST.

The MNIST dataset, or Modified National Institute of Standards and Technology database, is a widely used collection of handwritten digits in machine learning and computer vision projects. The purpose of MNIST dataset is providing data on which to compare methods to classify handwritten digit images into 10 digits (0-9). In this assignment, the dataset used is a sub-set of the MNIST dataset, containing images from three handwritten digits (3, 6, 9). Each digit class contains 1500 images of size 28x28 pixels.[1] **Fig.1** shows several images in this sub-set.

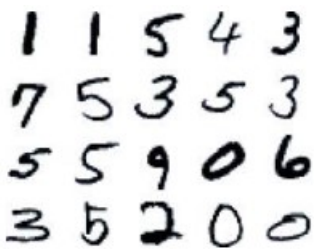


Figure 1: a snap shot of 20 sample images from the MNIST dataset

In Part I, PCA is used to reduce the dimensions of each image descriptor to two using the first two principal components and then K-Means and hierarchical clustering are used to cluster those data points in 2-D space into 3 cluster. Comparing the clustering results of the two different methods, the accuracy is 93.3556% for K-Means using cosine distance and 93.1333% for hierarchical clustering using average linkage.

In Part II, the procedure in Part I is repeated using LDA. The data points obtained using LDA are with larger between-class distance and smaller within-class distance compared to that obtained using PCA. Therefore, the accuracy of K-Means and hierarchical clustering is higher than that of PCA, which is 98.8% and 98.8444% respectively.

In Part III, the image of digit '6' is separated from the rest (the images of digit '3' and '9') and classified the dataset in a 5-fold cross validation setting using 3 classifiers: SVM with a linear kernel, SVM with a RBF kernel and a neural network with one hidden layer. The average accuracy of 5-fold cross validation is 99.40%, 99.82% and 98.58% respectively,and the AUCs are 99.32%, 99.82% and 98.42%. In addition , the RBF kernel parameter γ is tuned to adjust the performance of SVM with RBF kernel. The appropriate range of γ is in the interval $[0.0001, 0.1]$.

In Part IV, another three number ('1', '5', '8') are chosen to classify. Between the two experiences, what is the same is that the classification accuracy and AUC of SVM with RBF kernel are the highest among the three classifiers. What is the different is that the classification accuracy and AUC of neural network with one hidden layer are the lowest among the three classifiers in classifying '3', '6' and '9' while that are the second highest in classifying '1', '5' and '8'.

2 Assignment Requirements

2.1 Part I: Dim-reduction by PCA and Clustering

In this part, what is required to do is to use PCA to reduce the dimensions of each image descriptor to two using the first two principal components, and cluster those data points in the 2-D space into 3 clusters using two of the clustering methods (K-Means, hierarchical clustering, GMM, etc). For each method, a scatter plot is need to plot to show the cluster labels and evaluation about the result of clustering is need to discussed, i.e, whether images from the same digit are clustered into one region.

2.2 Part II: Dim-reduction by LDA and Clustering

In this part, what is required to do is use LDA for dimensionality and repeat the procedure in Part I. After applying LDA, the scatter plot is required to plot and comparison of performance of the two methods are need to discuss.

2.3 Part III: Binary Classification by Three Classifiers

In this part, what is required to do is to separate the images of digit '6' from the rest (the images of '3' and '9'), which is a two-class classification problem, using SVM with a RBF kernel, SVM with a linear kernel and a neural network classifier with one hidden layer in a 5-fold cross validation setting. The ROC curves and corresponding AUC of result from different classifier are need to compare. What's more, one parameter from SVM should be picked to tune to obtain more accurate classification results.

2.4 Part IV: Binary Classification on another three numbers

In this part, another three numbers are chosen to classify to see whether there is any difference from Part III.

3 Methodology

This part is to introduce the principle of PCA (Principal Component Analysis), LDA (Linear Discriminant Analysis), K-Means, hierarchical clustering, SVM and neural network.

3.1 Dimensionality Reduction

Dimensionality reduction is a technique used to reduce the number of features in a dataset while retaining as much of the important information as possible. In other words, it is a process of transforming high-dimensional data into a lower-dimensional space that still preserves the essence of the original data.

In machine learning, high-dimensional data refers to data with a large number of features or variables. The curse of dimensionality is a common problem in machine learning, where the performance of the model deteriorates as the number of features increases. This is because the complexity of the model increases with the number of features, and it becomes more difficult to find a good solution. In addition, high-dimensional data can also lead to overfitting, where the model fits the training data too closely and does not generalize well to new data.

Dimensionality reduction can help to mitigate these problems by reducing the complexity of the model and improving its generalization performance. There are two main approaches to dimensionality reduction: feature selection and feature extraction.

Feature Selection : Feature selection involves selecting a subset of the original features that are most relevant to the problem at hand. The goal is to reduce the dimensionality of the dataset while retaining the most important features. There are several methods for feature selection, including filter methods, wrapper methods, and embedded methods. Filter methods rank the features based on their relevance to the target variable, wrapper methods use the model performance as the criteria for selecting features, and embedded methods combine feature selection with the model training process.

Feature Extraction : Feature extraction involves creating new features by combining or transforming the original features. The goal is to create a set of features that captures the essence of the original data in a lower-dimensional space. There are several methods for feature extraction, including principal component analysis (PCA), linear discriminant analysis (LDA), and t-distributed stochastic neighbor embedding (t-SNE). PCA is a popular technique that projects the original features onto a lower-dimensional space while preserving as much of the variance as possible.

Here, the basic principle of PCA and LDA will be introduced.

3.1.1 PCA (Principal Components Analysis)

Principal component analysis (PCA) is a linear dimensionality reduction technique with applications in data analysis, visualization and data preprocessing. This is accomplished by linearly transforming the data onto a new coordinate system (principal components) such that the directions capturing the largest variation in the data can be easily identified. The goal is to find the directions that best explain the variations of the data, which means to find several directions (a subspace) such that the variance of the projected data in the subspace is maximized.[2]

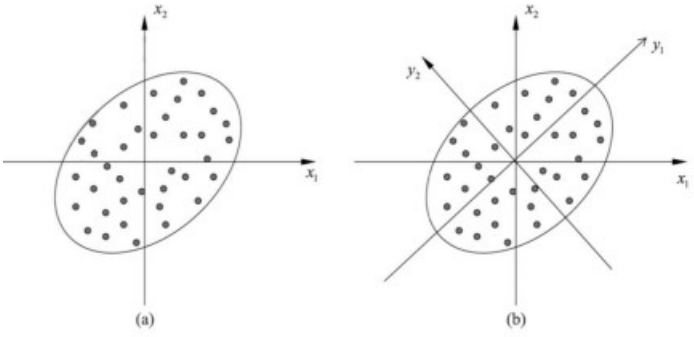


Figure 2: Map the data points above from x1-x2 space to y1-y2 space so that data projections on y1 axis has the largest variance

Given N data points $\mathbf{x_1}$, ... ,$\mathbf{x_N}$ in $\mathcal{R}^d$, choose an unit vector $\mathbf{u}$ in $\mathcal{R}^d$ that captures the most data variance. Therefore, a new set of features can be constructed by projecting the data onto those directions:

$$\mathbf{u}(\mathbf{x_i}) = \mathbf{u}^T(\mathbf{x_i} - \mu) \tag{1}$$

where μ is the mean of data points.

To find the direction that maximizes the variance of the projected data, the **Eq.2** should be maximized.

$$\arg\max \frac{1}{N}\sum_{i=1}^{N}\mathbf{u}^T(\mathbf{x_i}-\mu)(\mathbf{u}^T(\mathbf{x_i}-\mu))^T \tag{2}$$

$$= \mathbf{u}^T[\frac{1}{N}\sum_{i=1}^{N}(\mathbf{x_i}-\mu)((\mathbf{x_i}-\mu))^T]\mathbf{u} \tag{3}$$

$$= \mathbf{u}^T\Sigma\mathbf{u} \tag{4}$$

$$subjectto||\mathbf{u}|| = 1 \tag{5}$$

The direction that maximizes the variance is the eigenvector associated with the largest eigenvalue of Σ. The k eigenvectors with the largest eigenvalues are the first k principal components, and a low-dimensional space can be obtained by computing the projections of data in these components.

Therefore, the implementation of PCA on a dataset $\mathbf{X}$ can be divided into the following steps:

- Normalize the data into $[0, 1]$.
- Subtract the mean $\overline{\mathbf{X}} = \mathbf{X} - E[\mathbf{X}]$: subtract the mean makes variance and covariance calculation easier by simplifying their equations, The variance and covariance values are not affected by the mean value.
- Calculate the covariance matrix $\mathbf{C} = \frac{1}{N}\overline{\mathbf{X}}\overline{\mathbf{X}}^T$
- Calculate the eigenvectors and eigenvalues of the covariance matrix
- Select the k eigenvectors with the largest eigenvalues as the principal components.
- Compute the projected data on these k eigenvectors as the dimensionality reduced data

Limitations: The results of PCA depend on the scaling of the variables. This can be cured by scaling each feature by its standard deviation, so that one ends up with dimensionless features with unital variance[3]. Another limitation is that PCA relies on a linear model.In particular, PCA can capture linear correlations between the features but fails when this assumption is violated[4].

3.1.2 LDA (Linear Discriminant Analysis)

Linear Discriminant Analysis (LDA) is a supervised learning algorithm used for classification tasks in machine learning. It is a technique used to find a linear combination of features that best separates the classes in a dataset. LDA works by projecting the data onto a lower-dimensional space that maximizes the separation between the classes. It does this by finding a set of linear discriminants that maximize the ratio of between-class variance to within-class variance,as shown in **Fig.3**. In other words, it finds the directions in the feature space that best separate the different classes of data.

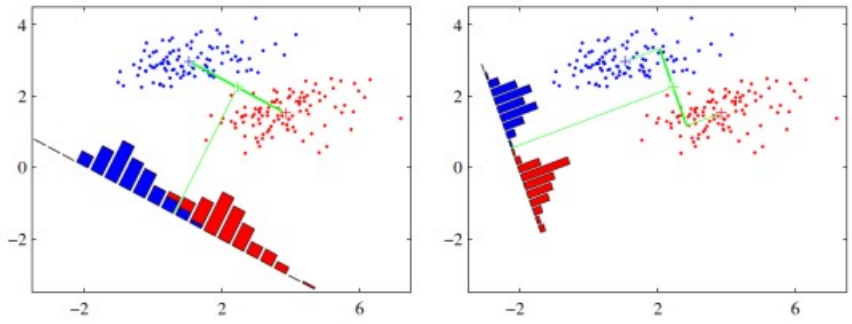


Figure 3: Map the data points to a line with maximum between-class distance and minimum within-class distance

Denote the dataset $\mathbf{D} = \{(x_1, y_1), ..., (x_n, y_n)\}$, where $y_i \in \{C_1, ..., C_n\}$. Define that $\mathbf{X_j}$ is the collected data in class C_j, μ_j is the mean of $\mathbf{X_j}$ and μ is the mean of all class. Also define the between-class scatter $\mathbf{S_b}$ and the within-class scatter $\mathbf{S_w}$.

$$\mathbf{S_b} = \mathbf{M_b}{\mathbf{M_b}}^T \tag{6}$$

$$\mathbf{S_w} = \mathbf{M_w}{\mathbf{M_w}}^T \tag{7}$$

where $\mathbf{M_b} = [\mu_1 - \mu, ..., \mu_n - \mu]$, $\mathbf{M_w} = [\mathbf{X_1} - \mu_1, ..., \mathbf{X_n} - \mu_n]$.

The between-class distance is equal to the trace of between-class scatter and he within-class distance is equal to the trace of within-class scatter. Then discriminant criterion in mathematical formulation is

$$\arg\max \frac{trace(\mathbf{W}^T\mathbf{S_b}\mathbf{W})}{trace(\mathbf{W}^T\mathbf{S_w}\mathbf{W})} \tag{8}$$

The optimal transformation is given by solving a genearalized eigenvalue problem ${\mathbf{S_w}}^{-1}\mathbf{S_b}$. In particular, $\mathbf{S_w}$ is required to be non-singular. When $n << d$, $\mathbf{S_w}$ is singular. In terms of this case, βI is added into $\mathbf{S_w}$.

Therefore, the implementation of LDA can be divided into the following steps:

- Normalize the data into $[0, 1]$.
- Compute the mean for each class and the whole class.
- Compute the between-class scatter $\mathbf{S_b}$ and the within-class scatter $\mathbf{S_w}$
- Find the eigenvalues and eigenvectors of ${\mathbf{S_w}}^{-1}\mathbf{S_b}$.
- Select the d eigenvectors with the largest eigenvalues as the basic vector.
- Compute the projected data on these d eigenvectors as the dimensionality reduced data

Advantages: It is a simple and computationally efficient algorithm. It can work well even when the number of features is much larger than the number of training samples. Also, it can handle multicollinearity (correlation between features) in the data[5].

Limitations: It assumes that the data has a Gaussian distribution, which may not always be the case. Also, it assumes that the covariance matrices of the different classes are equal, which may not be true in some datasets, and the data is linearly separable, which may not be the case for some datasets[5].

3.2 Clustering

Clustering is the task of dividing the unlabeled data or data points into different clusters such that similar data points fall in the same cluster than those which differ from the others. In simple words, the aim of the clustering process is to segregate groups with similar traits and assign them into clusters. Here, the basic principle of K-Means and hierarchocal clustering will be introduced.

3.2.1 K-Means

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity.

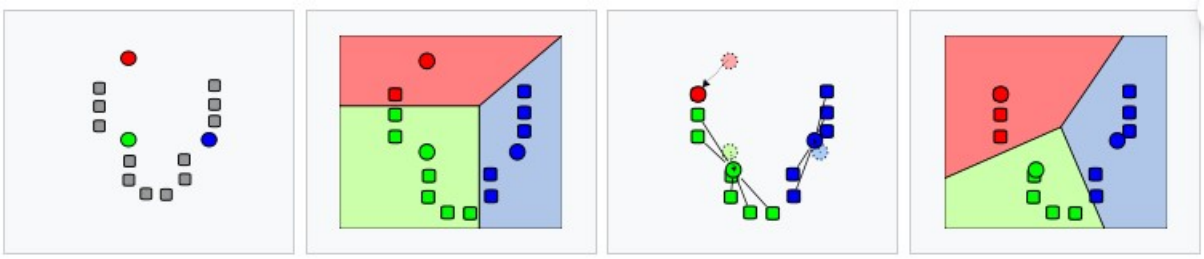


Figure 4: the process of K-Means

For choosing "k", one popular approach is testing different numbers of clusters and measuring the resulting Sum of Squared Errors(**Eq.9**), choosing the "k" value at which an increase will cause a very small decrease in the error sum, while a decrease will sharply increase the error sum. This point that defines the optimal number of clusters is known as the "elbow point".

$$E = \sum_{i=1}^{k} \sum_{j=1}^{n_i} ||\mathbf{x_{ij}} - \mathbf{m_i}||^2 \tag{9}$$

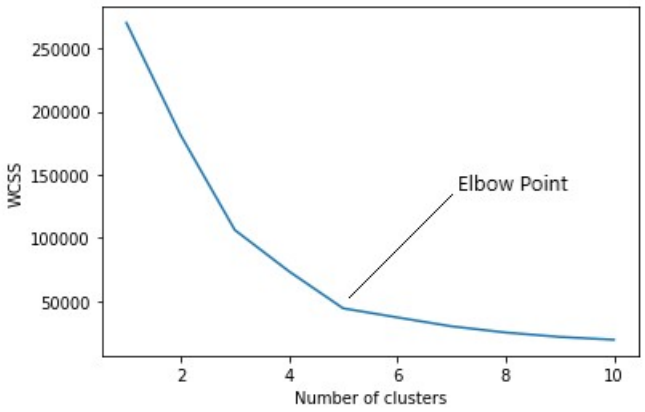


Figure 5: the process of choosing k

The implementation of K-Means can be divided into the following steps:

- **Step 1**: Randomly select k point as k clusters
- **Step 2**: Assign each object to the cluster with the nearest seed point
- **Step 3**: Compute the mean point as the centroids of the clusters of the current partition
- Go back to **Step 2**, stop when the mean point doesn't change for each new cluster.
- **Step 4**: Output the final partition

Advantages and Limitations: Since it scales to large data sets, K-Means is relatively simple to implement, guarantees convergence, can warm-start the positions of centroids, it easily adapts to new examples, and generalizes to clusters of different shapes and sizes, such as elliptical clusters. However, it presents downsides. The most obvious one is that the number of clusters is need to be defined manually, and, although some ways are put up to find the optimal "k", this is a decision that will deeply affect the results. Also, K-means is highly dependent on initial values. For low values of "k", this dependence can be mitigated by running K-means several times with different initial values and picking the best result. As "k" increases, advanced versions of K-means should be used to pick better values of the initial centroids (called K-means seeding). K-means produces clusters with uniform sizes (in terms of density and quantity of observations), even though the underlying data might behave in a very different way. Finally, K-means is very sensitive to outliers, since centroids can be dragged in the presence of noisy data[6].

3.2.2 Hierarchical Clustering

A hierarchical clustering approach is based on the determination of successive clusters based on previously defined clusters. It's a technique aimed more toward grouping data into a tree of clusters called dendrograms, which graphically represents the hierarchical relationship between the underlying clusters. Cluster assignments are determined by building a hierarchy. This is implemented by either a bottom-up or a top-down approach:

- The bottom-up approach is called agglomerative clustering and merges the two points that are the most similar until all points have been merged into a single cluster.
- The top-down approach is divisive clustering and starts with all points as one cluster and splits the least similar clusters at each step until only single data points remain.

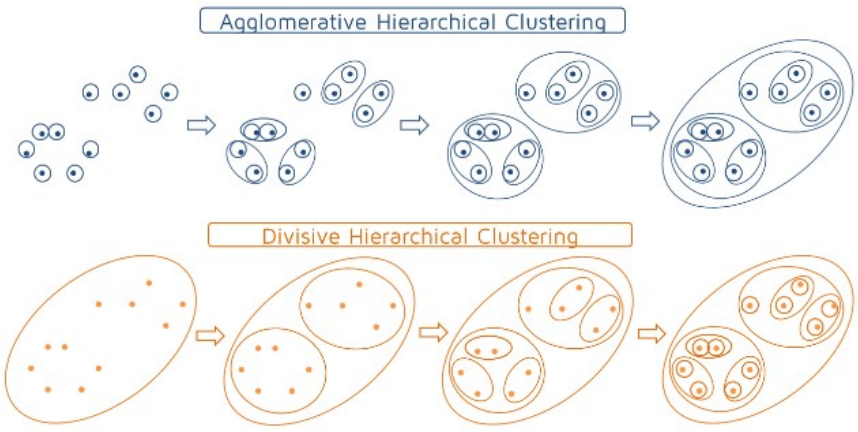


Figure 6: the process of hierarchical clustering

These methods produce a tree-based hierarchy of points called a dendrogram. The number of clusters "k" is often predetermined by the user, and clusters are assigned by cutting the dendrogram at a specified depth that results in "k" groups of smaller dendrograms.

The implementation of K-Means can be divided into the following steps:

- **Step 1:** Compute the distance matrix containing the distance between each pair of data points using a particular distance metric.
- **Step 2:** Merge the two clusters that are the closest in distance.
- **Step 3:** Update the distance matrix with regard to the new clusters.
- **Step 4:** Repeat **steps 1**, **2**, and **3** until all the clusters are merged together to create a single cluster.

Advantages and Limitations: Hierarchical clustering is a deterministic process, which means that assignments won't change when you run an algorithm multiple times on the same input data. Hierarchical clustering methods often reveal the finer details about the relationships between data objects and provide interpretable dendrograms. On the other hand, they're computationally expensive with respect to algorithm complexity and sensitive to noise and outliers[7].

3.3 Classification

Classification is the process of predicting the class of given data points. Classes are sometimes called targets, labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y.) Classification belongs to the category of supervised learning where the targets are also provided with the input data. Classification can be applied to a wide-variety of tasks, including credit approval, medical diagnosis and target marketing, etc. Here, the basic principle of SVM and neural network will be introduced.

3.3.1 SVM with Linear Kernel

Support Vector Machines (SVMs) are a type of supervised learning algorithm that can be used for classification or regression tasks. The main idea behind SVMs is to find a hyperplane that maximally separates the different classes in the training data. This is done by finding the hyperplane that has the largest margin, which is defined as the distance between the hyperplane and the closest data points from each class. Once the hyperplane is determined, new data can be classified by determining on which side of the hyperplane it falls. SVMs are particularly useful when the data has many features, and/or when there is a clear margin of separation in the data.

As shown in **Fig.7**, $\mathbf{w}^T\mathbf{x} + b = 0$ is a separating hyperplane. What need to do is to find the "maximum-margin hyperplane" that divides the group of points $\mathbf{x}_i$ for which $y_i = 1$ from the group of points for which $y_i = -1$, which is defined so that the distance between the hyperplane and the nearest point $\mathbf{x}_i$ from either group is maximized.

If the training data is linearly separable, two parallel hyperplanes can be selected to separate the two classes of data, so that the distance between them is as large as possible. The region bounded by these two hyperplanes is called the "margin", and the maximum-margin hyperplane

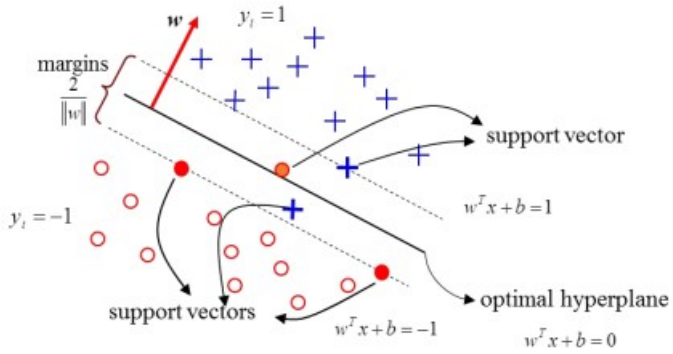


Figure 7: SVM with Linear Kernel

is the hyperplane that lies halfway between them. With a normalized or standardized dataset, these hyperplanes can be described by the equations

$$\mathbf{w}^T\mathbf{x} + b = 1 \tag{10}$$
$$\mathbf{w}^T\mathbf{x} + b = -1 \tag{11}$$

So the function of margin can be derived:

$$margin = \frac{2}{||\mathbf{w}||} \tag{12}$$

To maximize the margin, the optimization problem can be formulated as below:

$$\max \frac{2}{||\mathbf{w}||} \Rightarrow \min \frac{||\mathbf{w}||}{2} \tag{13}$$
$$\Rightarrow \min \frac{1}{2}||\mathbf{w}||^2 \tag{14}$$
$$s.t. y_i(\mathbf{w}^T x_i + b) \geq 1 \tag{15}$$

Applying Lagrange Mulitipler method to solve this problem in above, the solution has the form:

$$\mathbf{w} = \sum \alpha y_i \mathbf{x_i} \tag{16}$$
$$b = y_k - \mathbf{w}^T \mathbf{x_k} \tag{17}$$

Limitations: Linear SVM is suitable for the datasets which are linearly separable. When the dataset isn't linearly separable, the accuracy of the classification can be very low. This problem can be solved by introducing kernel functions to construct nonlinear SVM[8].

3.3.2 SVM with RBF kernel

Radial Basis Function Support Vector Machine (RBF SVM) is a powerful machine learning algorithm that can be used for classification and regression tasks. It is a non-parametric model that works well with non-linear and high-dimensional data. RBF SVM works by mapping the input data into a higher-dimensional feature space, where the classes can be separated by

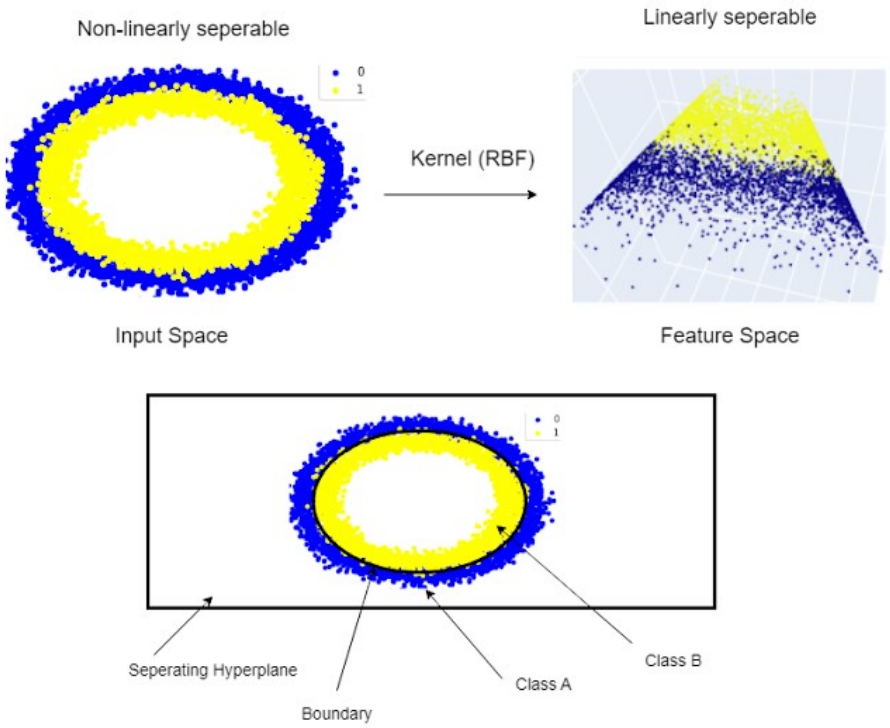


Figure 8: SVM with RBF Kernel

a hyperplane. The algorithm uses a kernel function, such as the Radial Basis Function, to measure the similarity between pairs of data points in the feature space.

The kernel function is

$$\mathbf{K}(\mathbf{X_1}, \mathbf{X_2}) = \exp\left(-\frac{||\mathbf{X_1} - \mathbf{X_1}||^2}{2\sigma^2}\right) \tag{18}$$

When introducing a new parameter $\gamma = \frac{1}{2\sigma^2}$, the equation will be

$$\mathbf{K}(\mathbf{X_1}, \mathbf{X_2}) = \exp\left(-\gamma||\mathbf{X_1} - \mathbf{X_1}||^2\right) \tag{19}$$

If γ is too small, the influence region of each support vector will contain almost the entire training dataset. The model will behave like a high-density linear hyperplane. If γ is too large, the radius of the influence region of the support vector will be so small that it can only affect itself. In this case, the model will be over-fitting, and will get low accuracy when predicting for new data.

3.3.3 Neural Network

Neural networks are artificial systems that were inspired by biological neural networks. These systems learn to perform tasks by being exposed to various datasets and examples without any task-specific rules. The idea is that the system generates identifying characteristics from the data they have been passed without being programmed with a pre-programmed understanding of these datasets. Neural networks are based on computational models for threshold logic. Threshold logic is a combination of algorithms and mathematics. Neural networks are based either on the study of the brain or on the application of neural networks to artificial intelligence.

The work has led to improvements in finite automata theory. Components of a typical neural network involve neurons, connections which are known as synapses, weights, biases, propagation function, and a learning rule. Neurons will receive an input $p_j(t)$ from predecessor neurons that have an activation $a_j(t)$, threshold θ_j, an activation function f, and an output function f_{out}. Connections consist of connections, weights and biases which rules how neuron i transfers output to neuron j. Propagation computes the input and outputs the output and sums the predecessor neurons function with the weight. The learning of neural network basically refers to the adjustment in the free parameters i.e. weights and bias. The structure of a simple neural network with one hidden layer is shown in **Fig.9**.

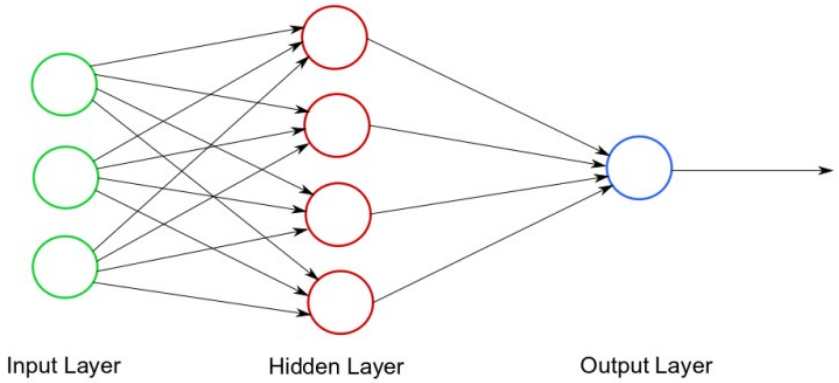


Figure 9: The structure of a simple neural network with one hidden layer

The structure of a computational neuron in hidden layer is shown in **Fig.10**.

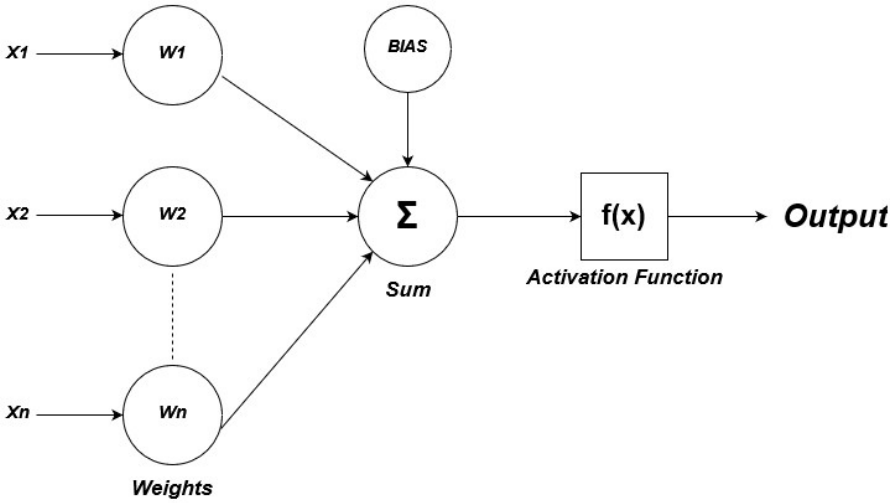
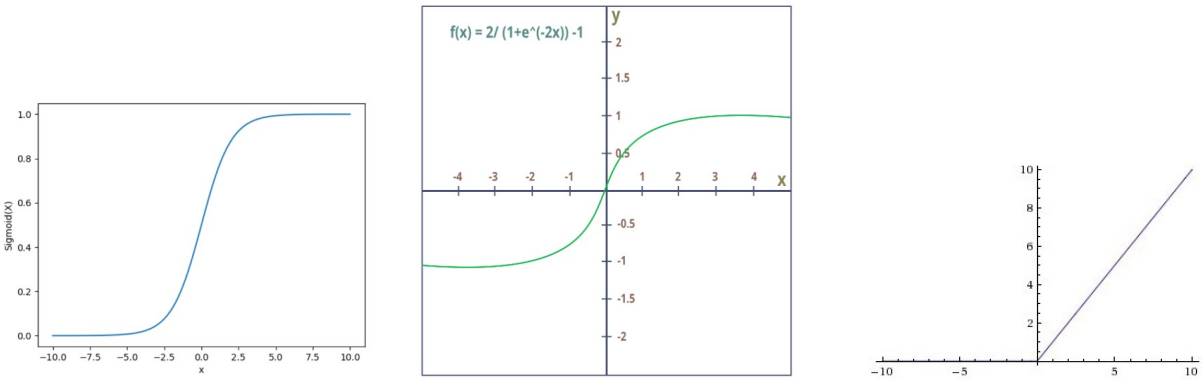


Figure 10: The structure of a computational neuron in hidden layer

It is clearly that the neuron computes the output of weighted inputs by activation function. The common used activation functions are *sigmoid* function, tanh function and *ReLU* function.



(a) Sigmoid Function (b) Tanh Function (c) ReLU Function

Figure 11: Activation Functions

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{20}$$

$$\tanh(x) = \frac{2}{1 + \exp(-2x)} - 1 \tag{21}$$

$$ReLU(x) = \max(0, x) \tag{22}$$

The most common used loss function is cross entropy error:

$$E = -\sum_{i}^{n} [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)] \tag{23}$$

Cross entropy error is the basis for optimizing parameters (mainly the weights of inputs). A commonly used method is gradient descent. Gradient descent is a method for unconstrained mathematical optimization. It is a first-order iterative algorithm for finding a local minimum of a differentiable multivariate function. The idea is to take repeated steps in the opposite direction of the gradient (or approximate gradient) of the function at the current point, because this is the direction of steepest descent. Conversely, stepping in the direction of the gradient will lead to a local maximum of that function; the procedure is then known as gradient ascent. It is particularly useful in machine learning for minimizing the cost or loss function[10].

3.3.4 Cross Validation

Cross validation is a technique for evaluating a machine learning model and testing its performance. Cross validation is commonly used in applied ML tasks. It helps to compare and select an appropriate model for the specific predictive modeling problem. Cross validation is easy to understand, easy to implement, and it tends to have a lower bias than other methods used to count the model's efficiency scores. All this makes cross-validation a powerful tool for selecting the best model for the specific task.

There are a lot of different techniques that may be used to cross-validate a model. Still, all of them have a similar algorithm:

- Divide the dataset into two parts: one for training, other for testing
- Train the model on the training set

- Validate the model on the test set
- Repeat 1-3 steps a couple of times. This number depends on the cross validation method

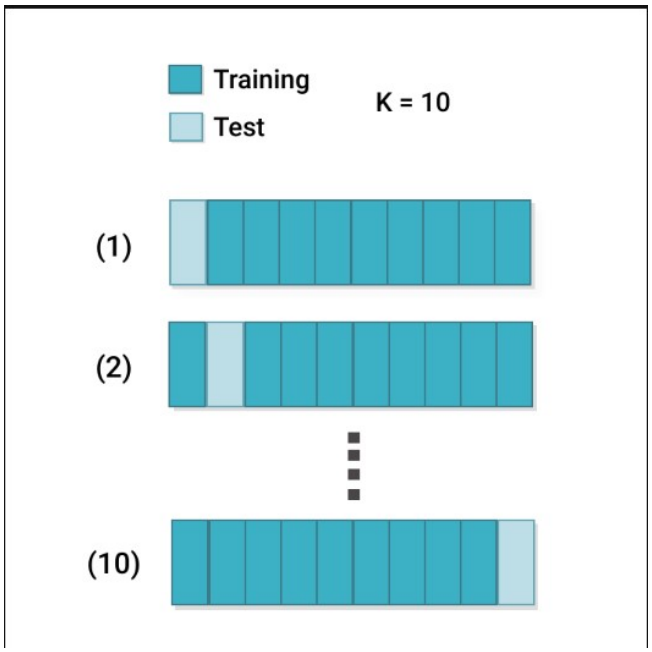
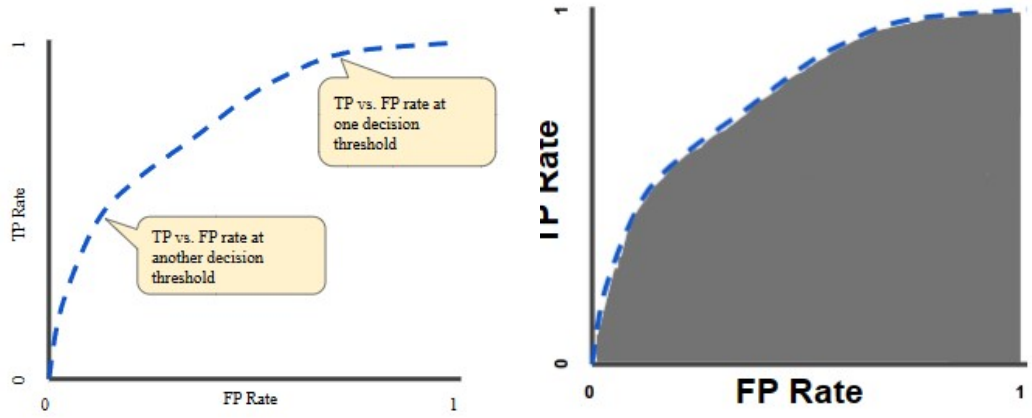


Figure 12: 10-fold cross validation

3.3.5 ROC Curve and AUC



(a) ROC Curve (b) AUC

Figure 13: ROC Curve and AUC

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: true positive rate and false positive rate. An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1). AUC provides an aggregate measure of performance across all possible classification thresholds.

One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example.

4 Results and Discussions

4.1 Part I: Dim-reduction by PCA and Clustering

4.1.1 Dimensionality Reduction by PCA

The first two principal components obtained by using PCA to reduce the dimension of each image descriptor are shown in **Fig.14**.

It is clearly that the data point after PCA have obvious differences in distribution, which is shown as data points of the same category are closer to each other that data points of different categories, and the data points of different categories occupy different areas in 2-D space.

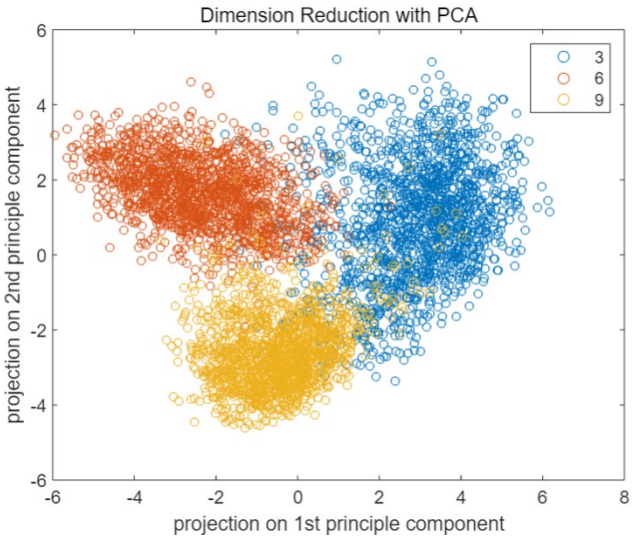
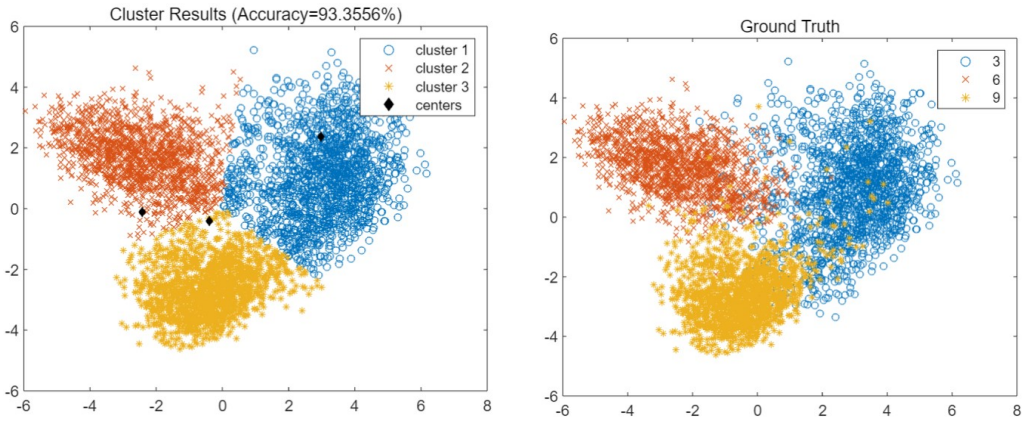


Figure 14: The First Two Principal Components after PCA

4.1.2 Cluster by K-Means

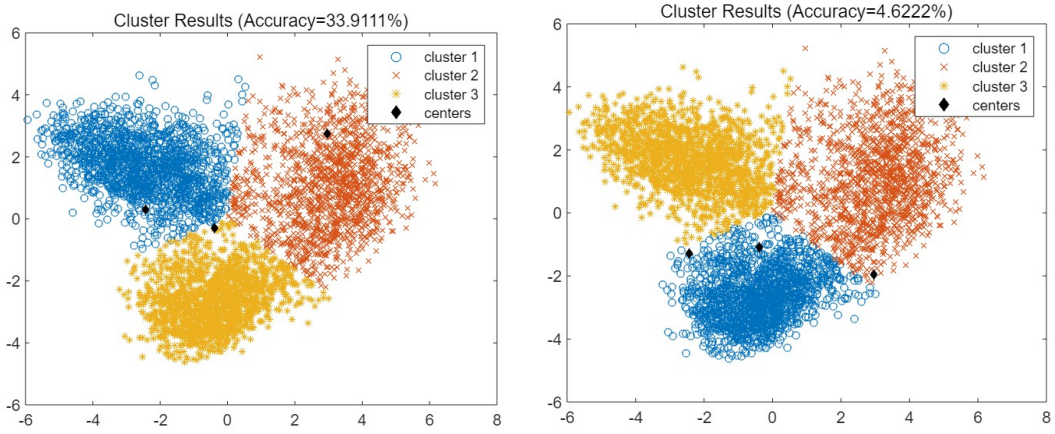


(a) K-Means Result (b) Ground Truth

Figure 15: Comparison of the Clustering Result by K-Means and Ground Truth

K-Means is used to cluster those data points in the 2-D space into 3 clusters, whose result is shown in **Fig.15**. The distance metric used is consine distance, which can reflect the difference in direction between vectors, to measure the distance between two data points. The scatter in **Fig.15** shows the result when the start points of K-Means are appropriate. So the final centers nicely belongs to 3 classes, i.e., '3', '6' and '9'. In this case, there are the most samples of the same class being in the same cluster.

However, after running the program several time, it is discovered that there exists situations that accuracy is very low, which is shown in **Fig.16**.



(a) K-Means Result (b) K-Means Result

Figure 16: performance of K-Means

Discussion: To quantitatively evaluate the results of clustering, the accuracy of clustering is defined as:

$$ACC = \frac{\sum_i^k (cluster(i) + 2 = label(i))}{total\ number\ of\ samples} \tag{24}$$

But this define isn't accurate because K-Means is unsupervised algorithm. This means when the data points of the same category is grouped together, the label of cluster assigned to this category by K-Means do not necessarily satisfy the relationship $cluster(i) + 2 = label(i)$. In other words, for K-Means, non-supervision means clustering, not categorizing. The same as for hierarchical clustering. So, in order to quantitatively evaluate the results of clustering correctly, the accuracy of clustering is redefined as:

$$ACC = \frac{\sum_i^k \max number\ of\ samples\ clustered\ into\ the\ same\ region\ with\ i\ label}{total\ number\ of\ samples} \tag{25}$$

Though the definition of the accuracy is not rigorous and sometimes misevaluate the result (e.g. if the majority of samples with two labels are clustered into one region, the accuracy calculated by **Eq.25** can be high, but it isn't reasonable), it can still reflect the performance of clustering to some extent. After redefine the accuracy, the value of accuracy is always 93.3556%, but the accuracy is not reasonable. As shown in **Fig.17**, the accuracy is high while the final centers don't belong to 3 classes separately, which also implies the poor robust of K-Means

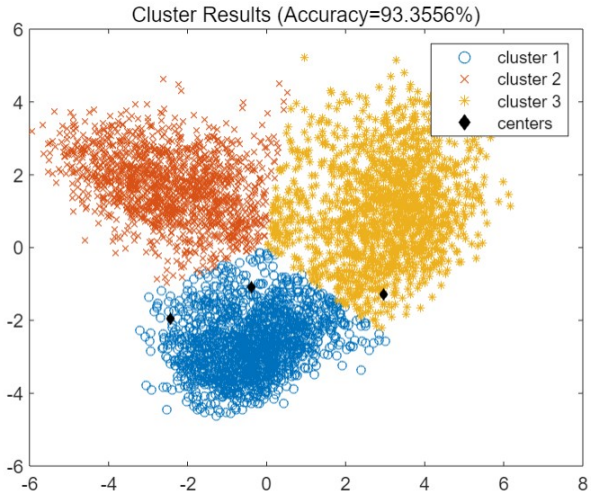
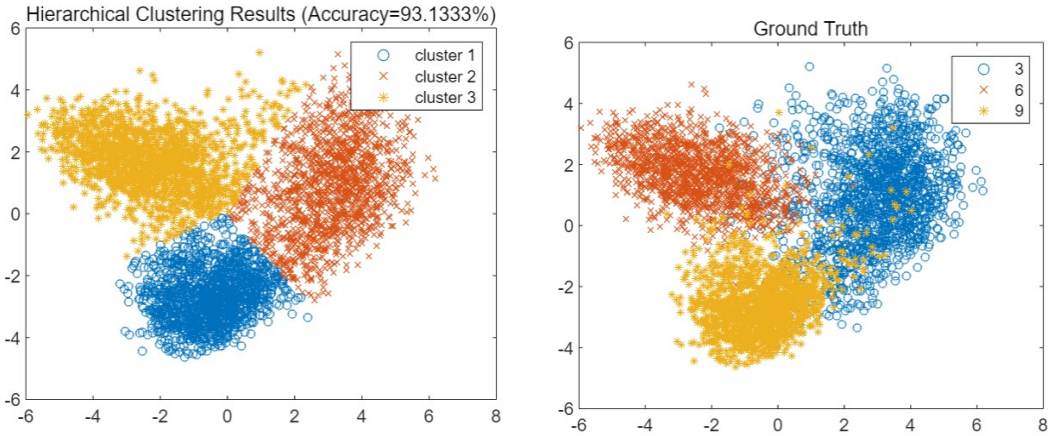


Figure 17: Poor Robust of K-Means

4.1.3 Hierarchical Clustering

Hierarchical clustering is used to cluster those data points in the 2-D space into 3 clusters, whose result is shown in **Fig.18**. The distance metric used is consine distance and average linkage.



(a) Hierarchical clustering Result

(b) Ground Truth

Figure 18: Comparison of the Clustering Result by Hierarchical clustering and Ground Truth

The accuracy of Hierarchical clustering is 93.1333%, which is slightly lower than that of K-Means. What's more, the result of hierarchical clustering doesn't change, which means that hierarchical clustering has better robust than K-Means.

4.2 Part II: Dim-reduction by LDA and Clustering

4.2.1 Dimensionality Reduction by LDA

The first two principal components obtained by using LDA to reduce the dimension of each image descriptor are shown in **Fig.18**.

Compared with the result obtained by PCA, data points of the same category are distributed more tightly, the spacing between different classes is larger, which is a excellent reflection of the basic principle of LDA-maximize bwtween-class distance and minimize within-class distance.

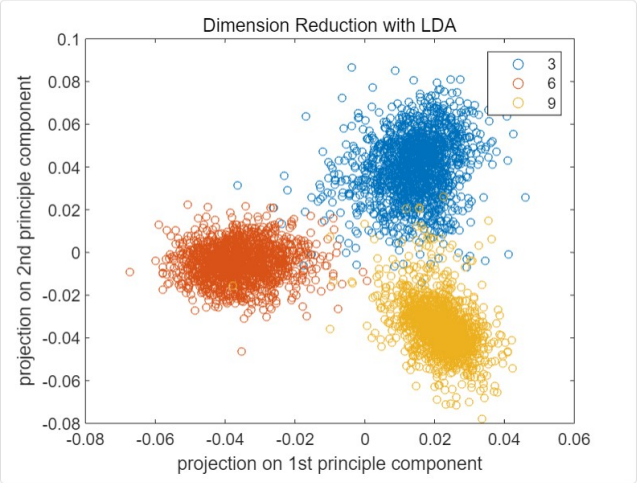
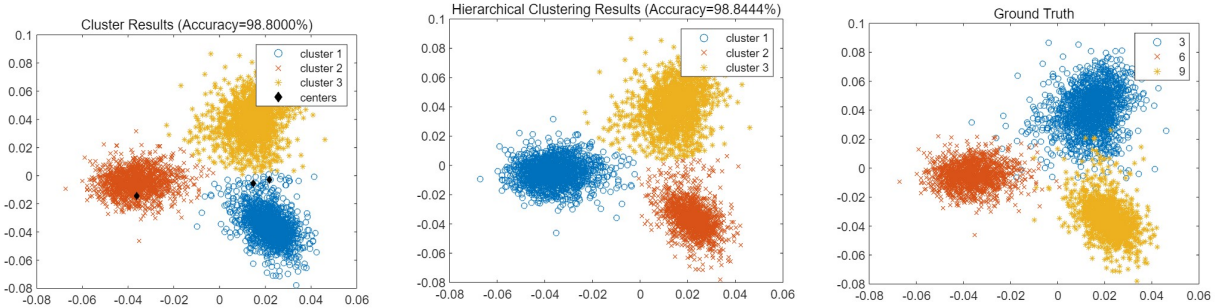


Figure 19: The First Two Principal Components after LDA

4.2.2 Cluster by K-Means and Hierarchical Clustering

K-Means and HIerarchical is used to cluster those data points in the 2-D space into 3 clusters, whose result is shown in **Fig.20**.

The accuracy of K-Means and HIerarchical clustering are quite close, which are 98.8% and 98.8444% respectively.



(a) K-Means Result (b) Hierarchical clustering Result (c) Ground Truth

Figure 20: Comparison of the Clustering Result by K=Means, Hierarchical clustering and Ground Truth

4.2.3 Comparison with Part I

PCA is unsupervised while LDA is supervised. When reducing the dimensions of each image descriptor to two using the first two principal components, LDA makes the data points with different categories more dispersed and the data points with the same category more concentrated compared to PCA. The reason behind this is that PCA is to find the directions such that the variance of the projected data in the subspace is maximized while LDA maximizes the between-class distance and minimizes the within-class distance.
Besides, the result of clustering data points after using LDA is better and the accuracy is higher compared to PCA.

Validation	1	2	3	4	5	Average
accuracy(%)	99.42	99.44	99.28	99.36	99.50	99.40

Table 1: Cross Validation Accuracy of SVM with Linear kernel

Validation	1	2	3	4	5	Average
accuracy(%)	99.83	99.89	99.78	99.81	99.81	99.82

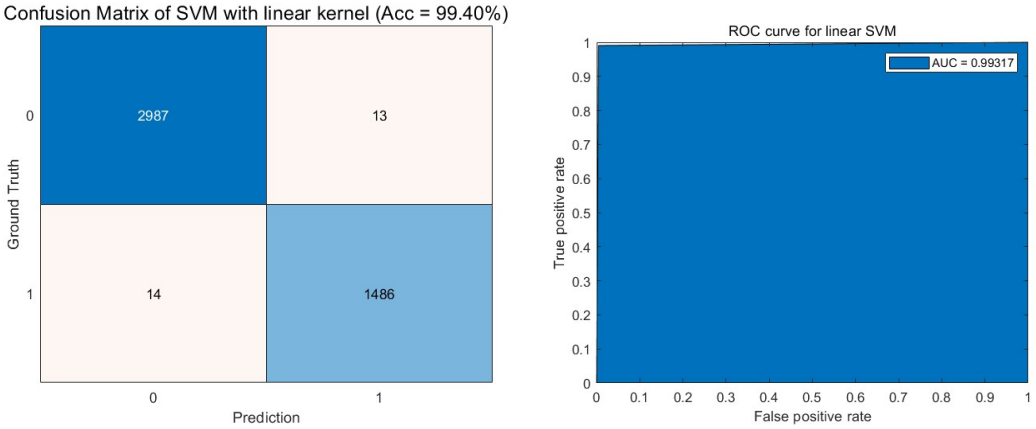
Table 2: Cross Validation Accuracy of SVM with RBF kernel

4.3 Part III: Binary Classification

4.3.1 SVM with Linear Kernel

The accuracy of the 5-fold cross validation is listed in **Tab.1**.

The corresponding confusion matrix and ROC curve are shown in **Fig.21**. The average accuracy using SVM with linear kernel is very high, which is 99.40%. And the AUC is 99.32%.



(a) Confusion Matrix (b) ROC Curve

Figure 21: Confusion Matrix and ROC Curve of SVM with Linear Kernel

4.3.2 SVM with RBF kernel

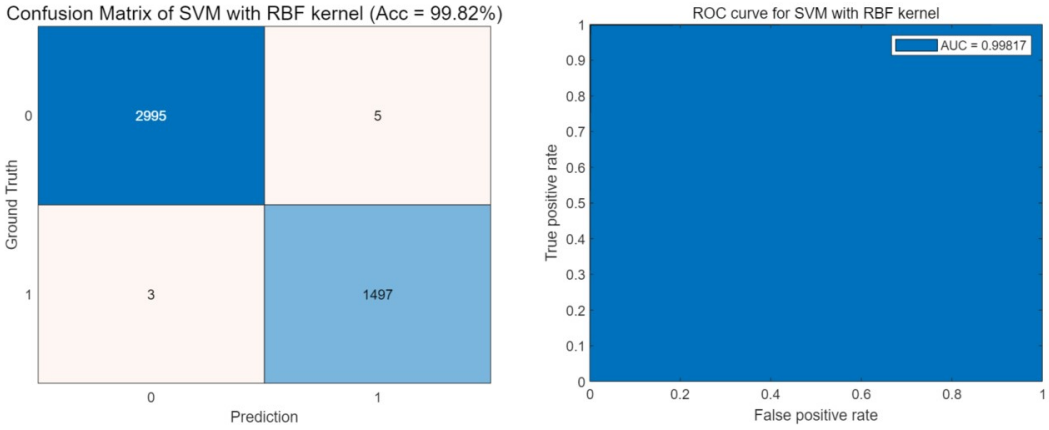
The accuracy of the 5-fold cross validation is listed in **Tab.2**.

The corresponding confusion matrix and ROC curve are shown in **Fig.22**. The average accuracy using SVM with RBF kernel is very high, which is 99.82%. And the AUC is 99.82%.

4.3.3 Neural Network with One Hidden Layer

The accuracy of the 5-fold cross validation is listed in **Tab.3**.

The corresponding confusion matrix and ROC curve are shown in **Fig.23**. The average accuracy using neural network with one hidden layer is relatively high, which is 98.58%. And the AUC is 98.42%.

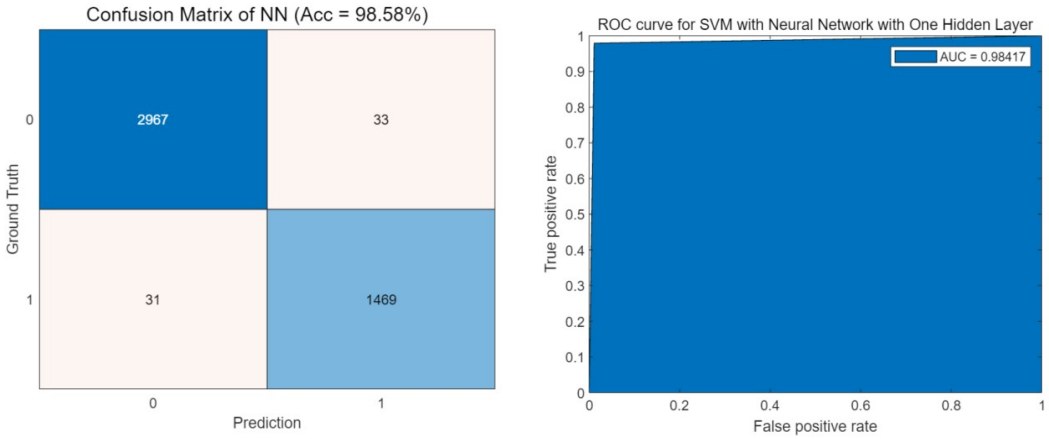


(a) Confusion Matrix (b) ROC Curve

Figure 22: Confusion Matrix and ROC Curve of SVM with RBF Kernel

Validation	1	2	3	4	5	Average
accuracy(%)	98.78	98.44	98.22	98.89	98.56	98.58

Table 3: Cross Validation Accuracy of Neural Network with One Hidden Layer



(a) Confusion Matrix (b) ROC Curve

Figure 23: Confusion Matrix and ROC Curve of Neural Network with One Hidden Layer

γ	0.0001	0.0005	0.001	0.005	0.01	0.03	0.05	0.08
accuracy	97.91%	98.80%	99.07%	99.62%	99.82%	99.91%	99.76%	98.09%
AUC	0.9740	0.9867	0.9902	0.9963	0.9982	0.9988	0.9963	0.9713
γ	0.1	0.5	1	5	10	50	500	1000
accuracy	93.93%	66.67%	66.67%	66.67%	66.67%	66.67%	66.67%	66.67%
AUC	0.9090	0.5	0.5	0.5	0.5	0.5	0.5	0.5

Table 4: accuracy and AUC under different γ

4.3.4 Comparison

Based on AUC ROC curves of the 3 classifier are shown together in **Fig.24**. The AUC of SVM with RBF kernel is 0.99817, which is the highest and close to that of SVM with linear kernel(0.99317).

Based on Accuracy Among the 3 classifiers, the accuracy of SVM with RBF kernel is 99.82%, which is also the highest and close to that of SVM with linear kernel (99.40%).

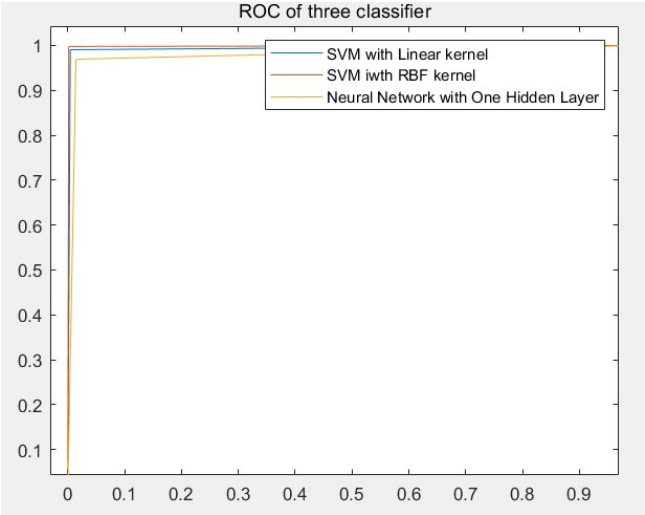


Figure 24: ROC curves of the 3 classifier

4.3.5 Parameter Adjusment

The RBF kernel parameter γ is chosen to tune to adjust the performance of SVM with RBF kernel. The parameter γ is tuned from 0.0001 to 1000, and the corresponding accuracy and AUC under different γ are shown in **Tab.4** and **Fig. 25**.

From **Fig.25**, it is clear that when γ increases from 0.0001 to 1000, the classification and AUC firstly increases and then decrease. When $\gamma \geq 0.5$, the performance of SVM with RBF kernel is just like random classifier.

4.4 Part IV: Exploration with Another Three numbers

In this part, another three number ('1', '5', '8') are chosen to repeat the procedure in Part III. The classification accuracy and ROC curve of the thre classifiers are shown in **Fig.26**, **Fig.27** and **Fig.28** respectively. The classification accuracy of the three classifiers is 95.87%,

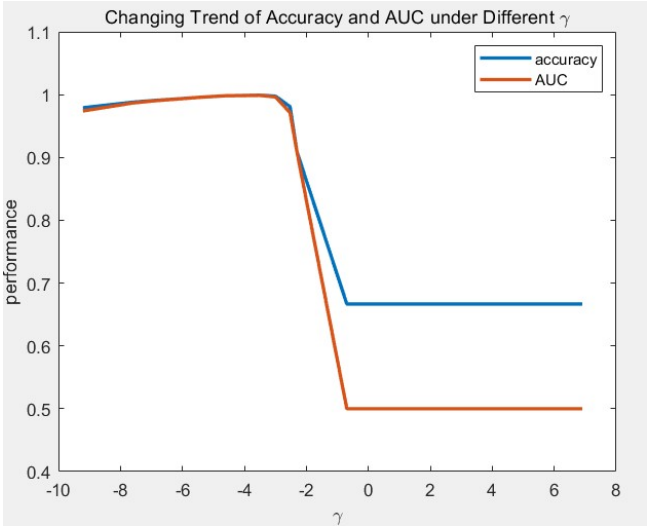
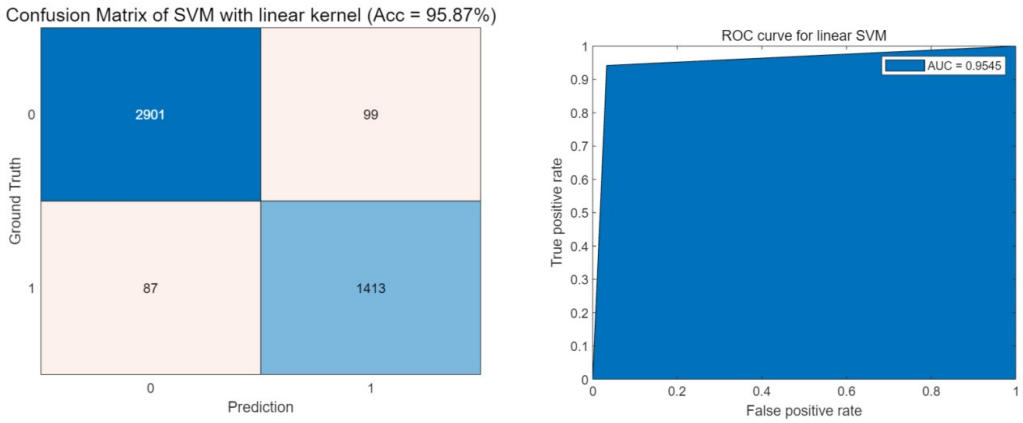


Figure 25: Changing Trend of Accuracy and AUC under Different γ

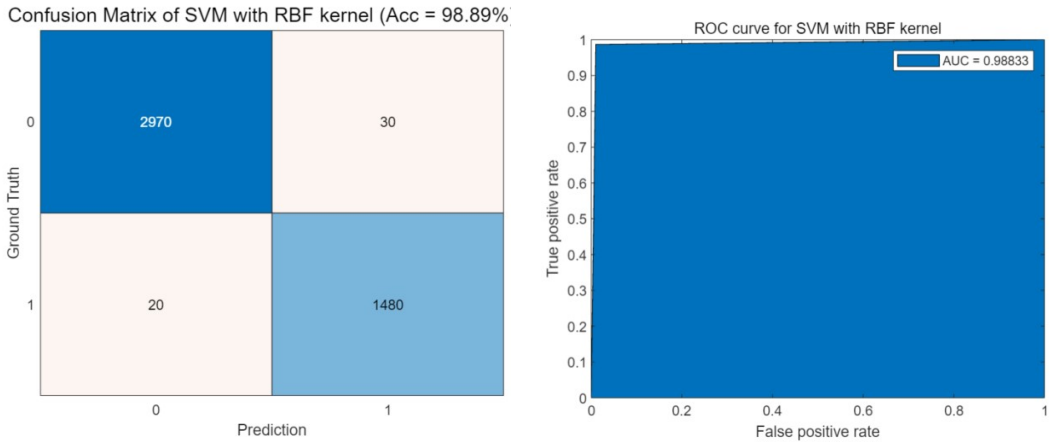
98.89% and 96.04% and the AUCs of the three classifiers are 0.9545, 0.98833 and 0.9555 respectively.



(a) Confusion Matrix (b) ROC Curve

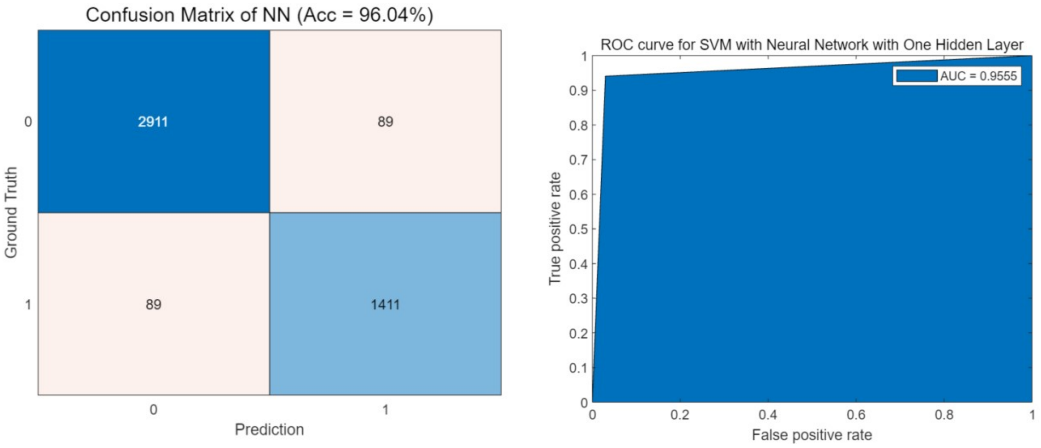
Figure 26: Confusion Matrix and ROC Curve of Classifying 1, 5 and 8 Using SVM with Linear kernel

Between the two experiences, what is the same is that the classification accuracy and AUC of SVM with RBF kernel are the highest among the three classifiers. What is the different is that the classification accuracy and AUC of neural network with one hidden layer are the lowest among the three classifiers in classifying '3', '6' and '9' while that are the second highest in classifying '1', '5' and '8'.



(a) Confusion Matrix (b) ROC Curve

Figure 27: Confusion Matrix and ROC Curve of Classifying 1, 5 and 8 Using SVM with RBF Kernel



(a) Confusion Matrix (b) ROC Curve

Figure 28: Confusion Matrix and ROC Curve of Classifying 1, 5 and 8 Using Neural Network with One Hidden Layer

References

[1] "THE MNIST DATABASE of handwritten digits". Yann LeCun, Courant Institute, NYU Corinna Cortes, Google Labs, New York Christopher J.C. Burges, Microsoft Research, Redmond.

[2] Jolliffe, Ian T.; Cadima, Jorge (2016-04-13). "Principal component analysis: a review and recent developments". Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences. 374 (2065): 20150202. Bibcode:2016RSPTA.37450202J

[3] Leznik, M; Tofallis, C. 2005 Estimating Invariant Principal Components Using Diagonal Regression.

[4] Jonathon Shlens, A Tutorial on Principal Component Analysis.

[5] McLachlan, G. J. (2004). Discriminant Analysis and Statistical Pattern Recognition. Wiley Interscience. ISBN 978-0-471-69115-0. MR 1190469.

[6] Hartigan, J. A.; Wong, M. A. (1979). "Algorithm AS 136: A k-Means Clustering Algorithm". Journal of the Royal Statistical Society, Series C. 28 (1): 100–108. JSTOR 2346830.

[7] Nielsen, Frank (2016). "8. Hierarchical Clustering". Introduction to HPC with MPI for Data Science. Springer. pp. 195–211. ISBN 978-3-319-21903-5.

[8] Support Vector Machines — scikit-learn 0.20.2 documentation". Archived from the original on 2017-11-08. Retrieved 2017-11-08.

[9] Aizerman, Mark A.; Braverman, Emmanuel M. Rozonoer, Lev I. (1964). "Theoretical foundations of the potential function method in pattern recognition learning". Automation and Remote Control. 25: 821–837.

[10] Boyd, Stephen; Vandenberghe, Lieven (2004-03-08). Convex Optimization. Cambridge University Press. ISBN 978-0-521-83378-3.