COMP5318

Machine Learning and Data Mining

ASSIGNMENT 2

Group: 20

Group Member:

Shengyu Liu (480505404)

Sai Ma (480490719)

Jiaxin Luo (480244248)

Abstract

Image classification plays an important role in a variety of applications in computer vision. However, due to the complexity in image systems where different scale and deformation may affect the classifier to wrongly predict the outcome, classifying images can be difficult in the field of machine learning. In this study, we propose different algorithms combined with preprocessing techniques for multiclass image classification. Four classifiers that have been applied on the benchmark dataset of Fashion MINIST include KNN, logistic regression, random forest and Linear SVC. Comparisons show that Random Forest outperforms others with the accuracy of 86.21% We conclude that RF brings better performance on Fashion MINIST dataset.

CONTENT

1 INTRODUCTION

Image classification is one of the most foundational problems in computer vision, which has a variety of practical applications such as image indexing and video classification [1]. During the last decade, a large amount of unstructured image data has been increasing with the help of digitalization and network services. This leads to the situation where the conventional methods, such as manual process, become hard to handle large-scaled and unstructured data. In order to improve the efficiency and accuracy for image classification, machine learning can play an important role due to its strength and practicality. By teaching machines to learn the essential features from images with the minimum human intervention, they are able to expose the underlying model from the data and make reliable predictions.

While identifying different images is a very trivial problem for human, it is challenging for machines to do the same with human level accuracy considering the complexity and non-linearity of image systems. There are many machine learning techniques that have been used in image data analysis, such as Random Forest (RF), Logistic Regression (LR), etc. However, the computational cost and classification accuracy are different in these methods. Therefore, there is a need to find a better solution in terms of resource consumption and error rate for image classification problems.

This study aims to find a better solution for multi-class image classification problems by comparing the performance of different classifiers to classify **Fashion-MNIST** images which is a dataset of Zalando article images - consisting of 60,000 sample training sets and 10,000 test sets. Each set is a 28x28 grayscale image associated with 10 categories of tags. The labels both training and test sets are assigned by: (0) T-shirt/top, (1) Trouser, (2) Pullover, (3) Dress, (4) Coat, (5) Sandal, (6) Shirt, (7) Sneaker, (8) Bag, and (9) Ankle boot. Machine learning algorithm performance can be tested directly through it, and its trained classifier can be applied to different kinds of data and representative.

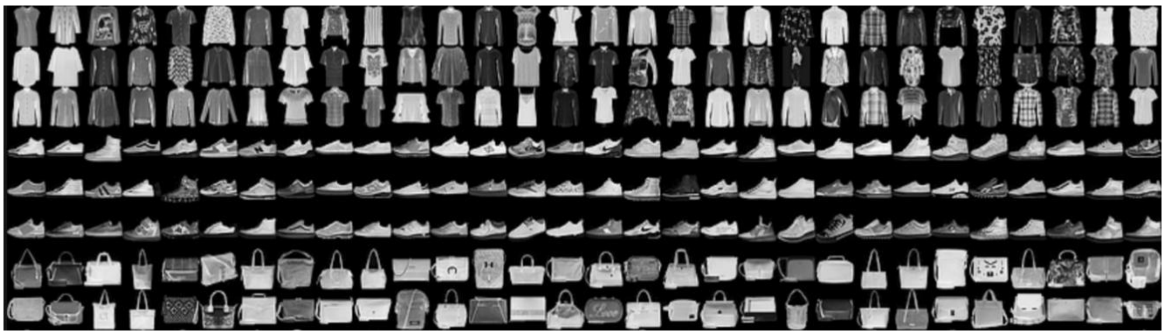


Figure 1：Fashion-MNIST images

The classifiers used in this study to process Fashion-MNIST set:

- K-Nearest Neighbors (K-NN)
- Logistic Regression (LR)
- Random Forest (RF)
- Linear SVC

Based on the computational cost and classification accuracy, the strengths and limitations of each classifier are pointed out as a key research analysis. In addition, different dimensionality reduction method may return the different accuracy of classifiers. Moreover, we would like to discuss our considerations and proposals for future work.

2 PREVIOUS WORK

The problem of classifying images by their type can be regarded as a problem of classifying patterns from images. Specifically, when it comes to fashion-MNIST images, the main challenge is to define and extract the relevant features to describe the images with regard to the high variability of clothing items. [2] In order to propose solutions in this scenario, related work focused on computer-vision based methods using machine learning algorithms. Nowadays, in order to achieve more accurate and fast recognition of images, a simple algorithm cannot meet people's expectations, and a combination of multiple algorithms is a trend in the current research of image recognition.

2.1 Multi-classifier combination

Due to the limitations of single classifier recognition and the complexity of partial image of corn, a method of corn partial image recognition based on adaptive weighting and multi-classifier combination was proposed. Firstly, three modes of color, color co-occurrence matrix and full color localization were extracted from the partial image, and three single classifiers based on SVM were constructed. Then, KNN and cluster analysis were used. The method calculated the dynamic weights of each single classifier; finally, the multi-classifier combination was performed by linear weighting to obtain the final classification result. The average recognition rate of this method could reach 94.71% [2]. Although this method is applied to partial image recognition of plants, its method of extracting feature values and calculating each weight can also be applied to the data set for identifying fashion clothes.

2.2 CNN- classifier (based on Fashion MNIST set)

One popular method used with image data is convolutional neural networks (CNN), In general, CNN is an unsupervised learning algorithm using multilayer perceptron’s to analyze data. Compared to other classification algorithms, an advantage in CNN is that it reduces the influence from prior knowledge. The use of CNN significantly helps to generate high accuracy on apparel classification problems. In one of the research papers presented by using a two-layer CNN consisting of convolutional layers with a 3×3 sized filter and max-pooling layers performing on every 2×2 pixels, along with the use of batch normalization and skip connections, it can achieve an accuracy of 92.54% on Fashion MNIST dataset [3]. There is also empirical data showing that introducing the SVM in CNN architecture could achieve a test accuracy of 91.86% on the same dataset. It trains the CNN as generic feature extractor and then uses these features to train the SVM [4]. The Hierarchical Convolutional Neural Networks (H-CNN) is proposed for apparel classification by first applying hierarchical classification of fashion images using VGGNet with CNN. The result indicates that H-CNN could bring better performance in Fashion MNIST dataset [5].

2.3 KNN- classifier (based on Fashion MNIST set)

Multiple classification models are implemented by Khoi Hoang, such as SVM, Logistic Regression, KNN, etc. PCA dimensionality reduction technology can speed up training time. For Fashion-MNIST data set, the best model is KNN with an accuracy rate of 86 percent through testing. Because KNN is the simplest classifier, it only has one parameter (K) to adjust. Although the accuracy rate to 91% was later increased through CNN, his training method and evolution were obviously ambiguous. For example, different dimensionality reduction methods might affect the accuracy, which required further research and explanation [6].

2.4 Other techniques

Long Short-Term Memory Networks (LSTM) is another typical deep learning technique for image classification tasks, which is a specific network of Recurrent Neural Networks (RNN). It is different from CNN as it learns to recognize image features across time. Although it is not efficient as CNN, it can lead to relatively high accuracy combined with other methods such as Network Pruning. Moreover, applying the Histogram of Oriented Gradient (HOG) feature descriptor into the multiclass SVM also turns out to be useful in the image classification on Fashion MNIST introduced by Greeshma K V and Sreekumar K. [6] In their work, they propose to use HOG for feature extraction combining with multiclass SVM classifier which provides relatively good fashion image classification efficiency.

3 METHODS

3.1 Raw dataset

The raw dataset of Fashion-MNIST is imported from the TensorFlow (keras) database. The imported data has been divided into training and test sets. Each image(data) have 784 features (28x28). PCA, SVD and Auto-encoder are used to reduce the dimension of the image. Using dimensionality reduction techniques not only significantly improves the training time of the model, but also improves the accuracy of the model by eliminating noise and redundancy. Four classifiers are used to train data and test. The four classifiers separately train data from different dimensionality reduction methods, evaluate the better parameters or coefficients through validation and choose the most suitable dimensionality reduction methods, and finally predict the accuracy by the best parameters and dimensionality reduction method.

3.2 Classier Algorithm Theory

3.2.1 K-nearest neighbors

K-Nearest Neighbors (KNN) is one of the most used algorithms in supervised learning for classification tasks. Fundamentally, it is a non-parametric classification approach which makes prediction based on feature similarity (i.e. distance matric) using the majority vote among the k nearest neighbors from training data.

The main process includes:

1) Measuring and sorting distance

In KNN algorithm, the first step is to compute the distance between the sample data point and training data point. A typical distance function used is Euclidean distance (i.e. L2 norm):

$$d(x,y) = \sqrt{\sum_{i}^{k}(x_i - y_i)^2}$$

Another commonly-used distance function is L1 norm:

$$d(x,y) = \sum_{i}^{k}|x_i - y_i|$$

Then, the distance is sorted in ascending order.

2) Choosing k nearest neighbors

The value of k determines the number of neighbors that will have influence on the predicted result. It is often chosen by cross-validation, and the k which produces the smallest error is used.

3) Assigning the class

   The sample data is then assigned to the class which is the most common among its k nearest neighbors using majority vote.

The reason we choose KNN algorithm is mainly because it is quite simple to implement and understand. There are several advantages regarding it:

1) Non-parametric

   Since the outcome of KNN mainly depends on the chosen value of k, it has no explicit assumptions about the data. In other words, the training process of KNN does not require learning from any other model, which is a strength as it avoids the possibility of mismodeling the underlying distribution of the initial data.

2) Versatile to use

   In addition, KNN can be used for both classification and regression problems. Moreover, it is also suitable for both linear and non-linear distributed data, which makes it become a building block in machine learning applications.

3.2.2 Logistic Regression

Logistic regression is one of the basic binary classifiers. Compared with linear regression, it assigns the predicted observations to a discrete set of classes instead of continuous number values. Fundamentally, Logistic Regression is a discriminative method since it directly assumes a parametric model to estimate the probability using sigmoid function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

where

$$z = \varpi^T x + b$$

Consequently, the value of the output falls into the range of $(0,1)$.

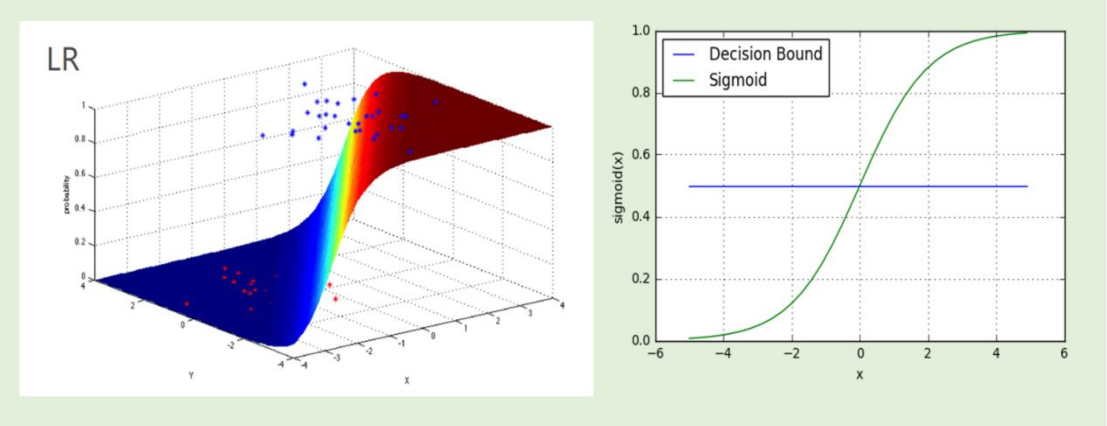


Figure 2. Sigmoid function for logistic regression

Based on the prediction function, the sample data is classified by comparing with a threshold value. For instance, assume the threshold is 5, if the outcome value of prediction function is above 5, the observation is then classified as positive (i.e. label=1); otherwise, negative.

Given that the Logistic Regression is based on a non-linear prediction function, using Mean Squared Error as the cost function cannot result in a convex function, thus, making it difficult to find a minimum value. Instead, it uses a function called Cross-Entropy (i.e. Log Loss):

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost\left(h_\theta\left(x^{(i)}\right), y^{(i)}\right)$$

$$Cost(h_\theta(x), y) = -\log(h_\theta(x)), \text{ if } \ y = 1$$

$$Cost(h_\theta(x), y) = -\log(1 - h_\theta(x)) \ \text{ , if } \ y = 0$$

In such a way, the cost function is converted into a convex function, that is, the function only has one single global minimum.

The above cost function can be written as:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)} log\left(h_\theta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right)\log\left(1 - h_\theta\left(x^{(i)}\right)\right)]$$

which is also called Maximum Likelihood.

The goal is to find the optimal weights that maximize the conditional data likelihood or minimize the cost function. In order to find out the minimum cost, an iterative algorithm called standard gradient decent can be used to continuously update the parameter ϖ:

$$\varpi = \varpi - \eta \times \frac{\delta}{\delta_\varpi} Cost(h_\varpi(x), y)$$

Given its simplicity and the fact that it requires less computational resources, Logistic Regression is a fundamental technique which turns out to perform well with sufficient training data and a small number of classes. In this study, we use it as a method for multiclass classification problem since the size of given dataset is large and there are only 10 classes.

3.2.3 Random Forest (New)

Random Forest is a supervised learning algorithm. Given its simplicity and the fact that it can be used for both classification and regression problems leading to high accuracies, it is considered as one of the most powerful algorithms in machine learning.

The intuition of Random Forest is to build multiple decision trees and merge them together to get more accurate prediction. Individually, predictions made by decision trees may not be accurate as they suffer from high variance. However, by combining them together, the predicted results can be much closer to the true values.

As shown in Figure 4, the main steps include:

1) Constructing decision trees

   The first step is to select a set of random samples from the training data and build decision trees for each sample data.

2) Generating results of decision trees

   The next stage is to generate outcome from each decision tree.

3) Assigning the class

   After gaining the outcome from all the individual decision trees, the class with the most votes can be assigned using the method of majority vote.

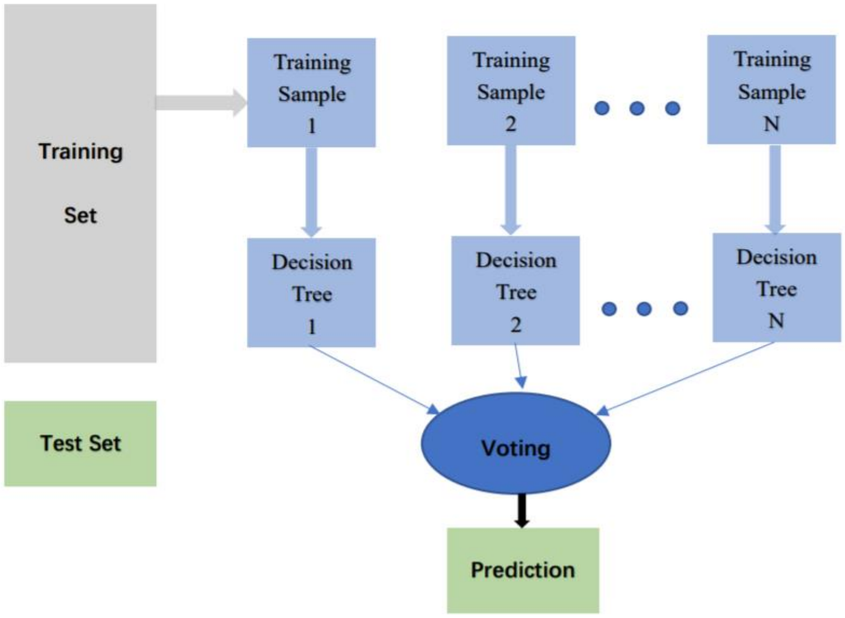


Figure 3. Process of Random Forest

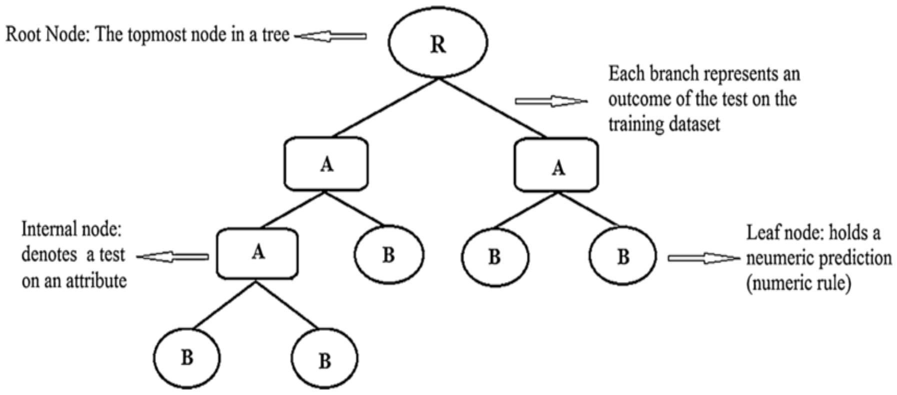
Enabling technique

1) Bagging

Bagging, also known as bootstrap aggregating, is a popular ensemble technique used in machine learning algorithms for classification problems. It is used as a method of reducing the variance in features by replacing the existing data with the new samples generated from the existing ones. In such a way, it can significantly help to improve the stability and accuracy of the classifier.

2) Decision tree

A decision tree is the building block for Random Forest, which helps to visually present the decision making. It is easy to understand as it uses a tree-like model for decision analysis as shown in Figure 4, with the possible solutions to a given problem emerging as the leaves of a tree, and each node representing a point of deliberation and decision. [8]


Root Node: The topmost node in a tree
R
Each branch represents an outcome of the test on the training dataset
A
A
Internal node: denotes a test on an attribute
A
B
B
B
Leaf node: holds a neumeric prediction (numeric rule)
B
B

-

Figure 4. A decision tree model

There are many strengths making Random Forest become one of the most popular machine learning algorithms. We apply it as one of the classifiers mainly due to its advantages:

1) Avoid overfitting

The main advantage of Random Forest is that by applying the technique of bagging, it takes into consideration of enough decision trees and averages these low-bias and high-variance predictors to reduce the variance while not increasing the bias. As a result, this kind of ensemble learning can achieve high accuracy with less possibility of overfitting the training data.

2) Adding additional randomness

Moreover, Random Forest also adds additional randomness by discovering the best feature among a random subset of features, leading to the diversity in the forest.

3) Enabling feature extraction

Another advantage to mention is that Random Forest reveals the relative feature importance. By computing the relevance score of each feature in training data, it allows the selection of the most contributing features while removing the least important ones.

3.2.4 Linear SVC based on SVM

The Support Vector Classifier is the best one of the possible linear classifiers based on the distribution of the training samples. Linear SVC is based on SVM and is a branch of SVM. So here is a focus on SVM. Support vector machine (SVM) is a supervised learning algorithm which has been widely used in both classification and regression tasks. In general, the idea of SVM is to find an optimal hyperplane which is able to correctly classify the data points into distinct classes.

SVM is originally designed for binary classification problems. In order to separate data points into two classes, the objective is to find a hyperplane with the maximum margin, which can be used as decision boundaries to attribute data points based on the location of each data point. There are many different hyperplanes that could be built among two data groups, and the best one is chosen by maximizing the margin between the hyperplane and the data points which are closer to the hyperplanes (i.e. support vectors). Given training data $x_1, x_2, \dots, x_n$, new data points can be classified based on the function:

$$y(x) = sign(\varpi^{\mathrm{T}} \emptyset(x) + b)$$

where ϖ is the weight factor, $\emptyset(x)$ is the basis function, and b acts as a bias. The label of each data point can be defined based on the output value which is either +1 or -1.

The objective of SVM is to find the optimal hyperplane, which can be done by maximizing the margin of the decision surface, where the margin of the hyperplane is the separation between the hyperplane and the closest data point with a given weight vector and bias. The distance between the data points and the hyperplane can be calculated as:

$$r = \frac{\varpi^T \emptyset(x) + b}{||\varpi||}$$

and the distance between support vectors is:

$$\rho = 2r = \frac{2}{||\varpi||}$$

Hence, in order to maximize the margin, the goal is to minimize the value of $||\varpi||$ or to maximize the value of ρ, in other words, to find ϖ and b such that:

$\emptyset(x) = ||\varpi||^2 = \varpi^{\mathrm{T}}\varpi$ is minimized,

for all (x_i, y_i) where $i = 1 \dots n$ $y_i\big(\varpi^{\mathrm{T}}\emptyset(x) + b\big) \geq 1$

where $||\varpi||^2$ is the Euclidean norm (i.e. L2 norm)? The problem can thus be converted to optimizing a quadratic function with linear constraints, that is, to gain the weight factor by solving an optimization problem with a hinge loss function:

$$\min \frac{1}{n} \varpi^{\mathrm{T}} \varpi + C \sum_{i=1}^{n} \max\left(0, 1 - y_i\left(\varpi^{\mathrm{T}} \emptyset(x) + b\right)\right)^2$$

where C is the penalty parameter.

It can be rewritten as:

$$\min \frac{1}{n} \|\varpi\|^2 + C \sum_{i=1}^{n} \max\left(0, 1 - y_i\left(\varpi^{\mathrm{T}} \emptyset(x) + b\right)\right)^2$$

3.3 Data scaling

Since the given training data has 10 classes, some of the features with a larger range may dominate over others, leading to the bias in the classification outcome. Therefore, it is crucial to adjust the values of all the numeric features to make them fall into a common scale without distorting the differences in the ranges of values. In order to reduce the influence of unnormalized data, there are two ways.

3.3.1 Data standardization (Z-score normalization)

A common way to rescale the features is done by removing the mean and scaling it to unit variance, so that the features will have the properties of a Gaussian distribution.

$$data_{standardized} = \frac{data_{original} - mean}{standart\ deviation}$$

3.1.2 Data normalization (Min-max scaling)

Another solution is to shrink the dataset to the range of $[0,1]$. Given each input data is an image which can be presented as a collection of pixels, and the range of each pixel falls into 0 to 255, thus, the data can be normalized in such a way:

$$data_{normalized} = \frac{data_{original}}{255}$$

As a result, the range of all the input values can be standardized into same range between 0 to 1.

Scaling methods dramatically improve the performance on classification problems. While it is proven that distance-based classifiers, such as SVM, KNN and Logistic regression, are more variant to feature scaling, others such as random forest can also benefit from other rescaling methods. Moreover, rescaling data also plays an important role in supporting preprocessing methods such as PCA. Hence, we apply both of the methods in this study to further improve the classification accuracy.

3.4 Pre-processing

Dimensionality reduction is a method of pre-processing data for high-dimensional feature data. Dimensionality reduction is to preserve some of the most important features from high-dimensional data, remove noise and unimportant features, thereby achieving the goal of improving the speed of data processing. In actual production, dimension reduction can save a lot of time and cost in case of certain information loss. Dimensionality reduction has become a very widely used data preprocessing method.

3.4.1 Principal Component Analysis

Principal component analysis (PCA) is an unsupervised mathematical algorithm for dimensionality reduction. It can reduce the dimensionality of the data while retaining the main information. By extracting the principal components with the maximal variation, it can reconstruct a large-scale dataset into a smaller one with lower dimensionality that still holds as much information as possible. Consequently, it allows the visualization of high-dimensionality data samples, making it possible to compare the difference among different classes in the dataset.

Mathematically, the purpose of PCA is to find linearly uncorrelated orthogonal columns (i.e. principal components) from the original high-dimensional data. In other words, PCA tries to get the approximation of the original data. Thus, the objective function of PCA can be written as:

$$J = \sum_{n=1}^{N} |x(n) - \hat{x}(n)|^2$$

PCA algorithm flow:

1) Remove the average (To subtract average value for each feature);
2) Calculate the covariance matrix;
3) Calculate the eigenvalues and eigenvectors of the covariance matrix;
4) Sorting feature values from large to small;
5) Retain the largest eigenvector;
6) Convert the data into a new space constructed by a feature vector.

Figure1 is the result of dimensionality reduction into different dimensions by PCA. From the figure, it can be found that the main component of the image is always retained, but the components that have little influence on the data analysis are removed. Figure 5 shows the cumulative explained variance from the different degree of dimensionality reduction. For example, when the image dimension is reduced to 84 dimensions, the similarity between the new image and the original image is 90 percent.

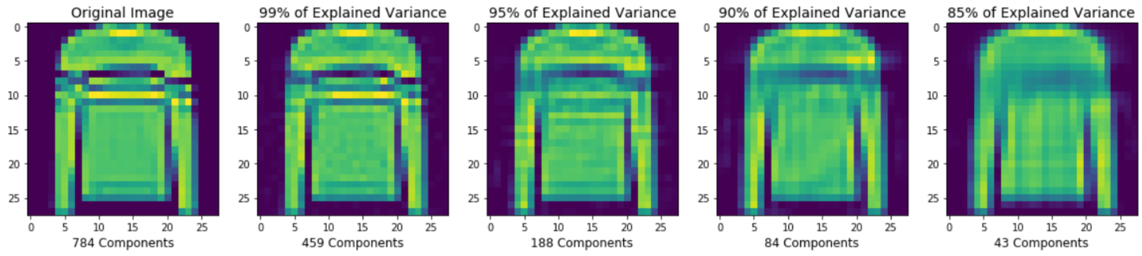

Original Image
99% of Explained Variance
95% of Explained Variance
90% of Explained Variance
85% of Explained Variance
0
5
10
15
20
25
784 Components
459 Components
188 Components
84 Components
43 Components

Figure 5: Dimensionality reduction of PCA

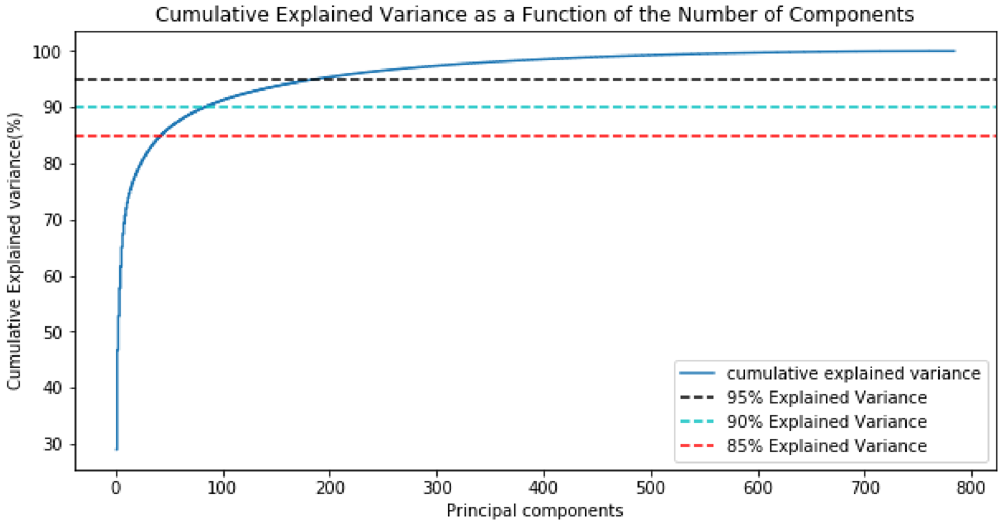


Figure 6: Explained Variance

PCA is a commonly used method of data analysis. PCA transforms the original data into a set of linearly independent sets of dimensions through linear transformation, which can be used to identify and extract the main feature components of the data. PCA dimension reduction makes the data simpler and more efficient, thus the purpose of improving data processing speed is achieved. Dimensionality reduction has become a method which applies in very widely data preprocessing. PCA algorithm has been used in the exploration and visualization of high-dimensional data sets, and can also be used in data compression, data preprocessing, image, voice, communication analysis and processing.

3.4.2 Singular value decomposition

Singular value decomposition (SVD) is another commonly-used dimensionality reduction method. Since it can be used for both real and complex matrices, any matrices can be decomposed using SVD.

The purpose of the SVD is to extract important values from a matrix to help classify the data into families. In other words, it can manipulate matrices by pulling them apart into values that are easier to work with, then stitching those values back together at the end of the computation to obtain some type of result. [7]

Given a matrix X, where $X \epsilon R^{m \times n}$ SVD decomposes it into three elements:

$$X_{m \times n} = U_{m \times r} \Sigma_{r \times r} {V_{r \times n}}^{T}$$

where U and V^T are both column-orthonormal matrix, Σ is a diagonal matrix containing singular values, and r is the rank of $X_{m \times n}$. U is a matrix that contains the important information about the rows in a matrix, where the first column holds the most important information. Similarly, V^T represents information based on the columns of a matrix with the first row contributing to the most important information. Each value in Σ indicates the strength of the corresponding element in both U and V^T.

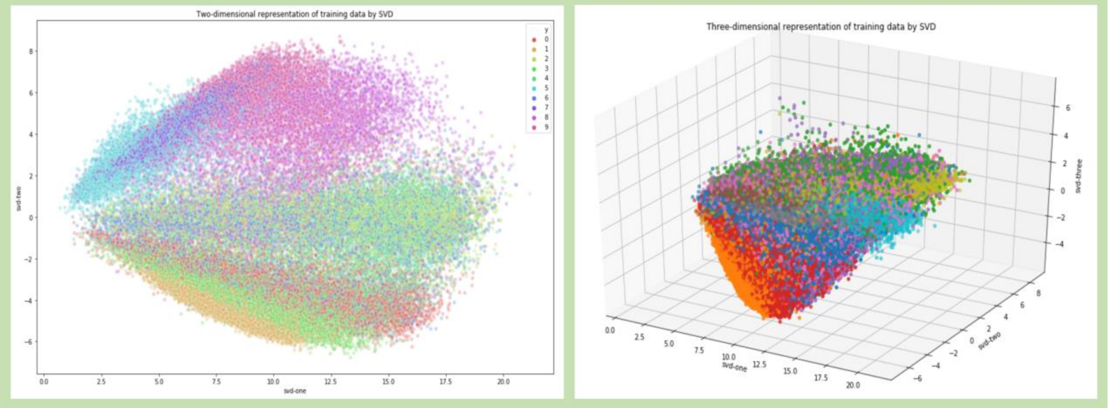


Figure 7: data in 2D and 3D through SVD.

Figure 7 is the result of reducing the dimension of original training data to 2D and 3D through SVD. In this figure, it can be found that the points of the same color can basically come together. The points of same color represent the same type of clothing in Fashion-MNIST dataset. Of course, a small number of points are still lost. This situation can understand the distortion of the image after dimension reduction, or the useless part can be excluded.

3.4.3 Compare PCA with SVD

There are two kinds of implementations of PCA, one is realized by eigenvalue decomposition, and the other is realized by singular value decomposition. When using SVD to complete PCA dimensionality reduction, the expectation E(X)=0 of all samples of data. PCA can be said to be a package for SVD. If SVD is implemented in dimension reduction, it is also realized. The PCA is gone.

3.4.4 Autoencoder

Autoencoder is also a popular feature extraction technique used in deep learning. It is a neural network architecture capable of finding the underlying structure within data to develop a compressed representation of the original data. The basic idea is to take an image as input and reconstruct it using fewer number of bits from the latent layer. The Autoencoder is similar to the PCA as they both convert a high-dimensional data to a lower dimension. However, while the PCA uses the linear transformation, the Autoencoder is often regarded as a non-linear generalization of PCA, which is capable of using non-linear manifolds to describe the original data.

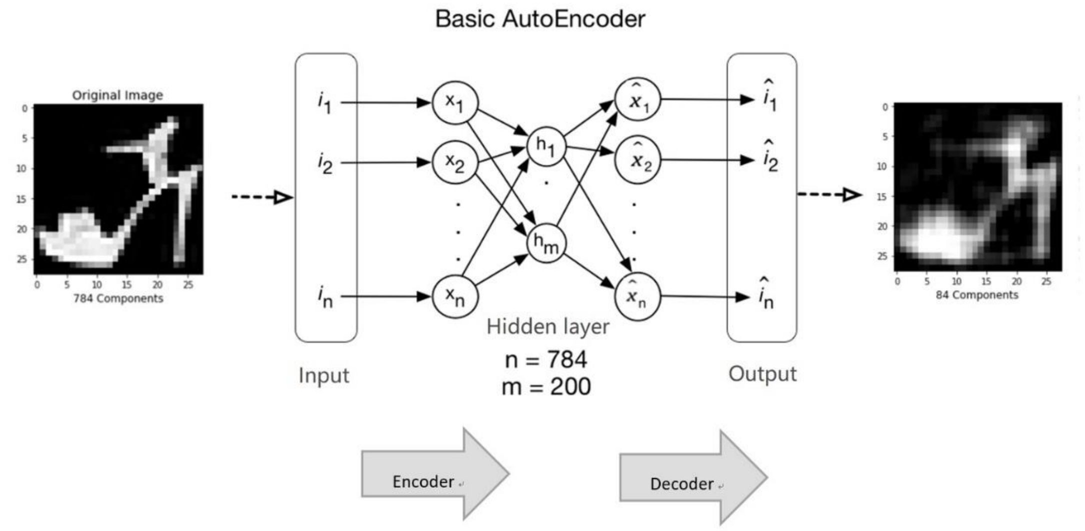


Figure 8: The process of Autoencoder

There are three key elements in the architecture of Autoencoder (Figure 8). The encoder takes input data from the input layer and compresses it into a fewer number of bits. Then the compressed representation of the input is presented in the hidden layer. These hidden layers, also known as bottlenecks, are the key components in the network, since they constrain the amount of information that can pass through the whole network. The next step is to reconstruct the input using the compressed data from the hidden layer. In other words, the Autoencoder tries to learn an approximation of the original data using nonlinear activation functions at each layer, so that the output $\hat{x}$ is similar to x.

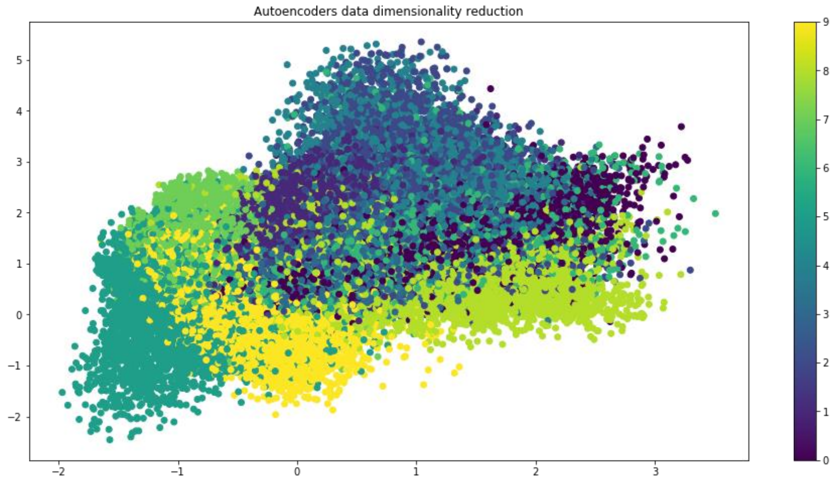
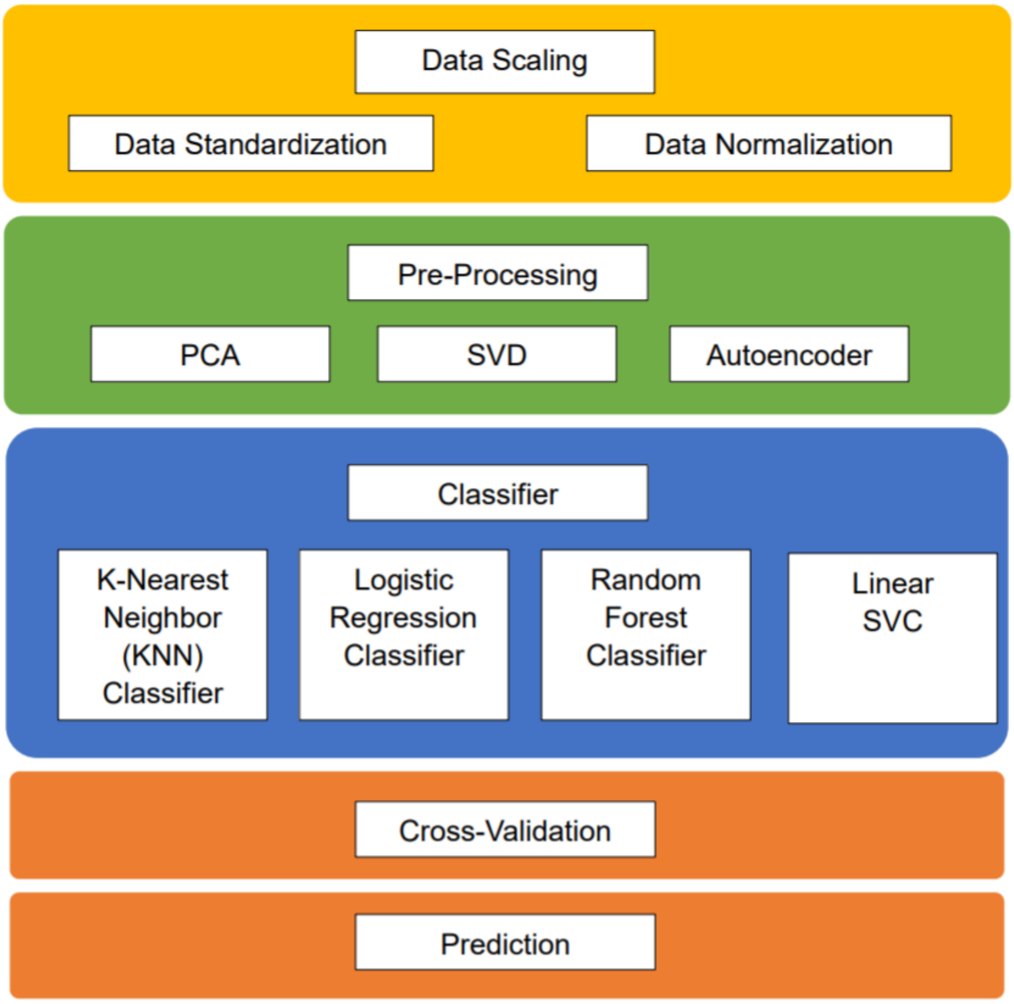


Figure 9: Data in 2D through Autoencoder

There are 10 numbers (10 types of data) in the Figure9, each number is displayed in a 28*28 (784) dimension image. All the original data to 84D data and displayed it in 2D data coordinates. From Figure 5, the same category of data can be classified into the same class. The points of the same color represent data that is assigned to the same class (Lebel is the same). In this experiment, the autoencoder is similar to PCA, which plays a role in dimensionality reduction.

3.5 Design Choices

The classifier methods and other techniques used for the Classification of the Fashion-MNIST data set show in the chart below.



3.5.1 K-Nearest Neighbors

1) The choice of k

In KNN algorithm, the value of k (n_neighbors) plays an important role as it determines the number of nearest neighbors to predict the sample data. A small value of k may increase the impact of noise data on the prediction. Theoretically, if the value of k chosen is large enough, it can suppress the effects of the data from other classes and lead to high accuracy. However, in addition to the extra computational cost it may take, it can also lead to the ambiguity on the classification boundaries. As a result, the model with a large value of k underfits the data. Figure 7 shows that as the value of k increases, the edge between the different classes becomes smoother. Generally, K is better between 5-25. Figure 10 shows that as the value of k increases, the edge between the different classes becomes smoother.

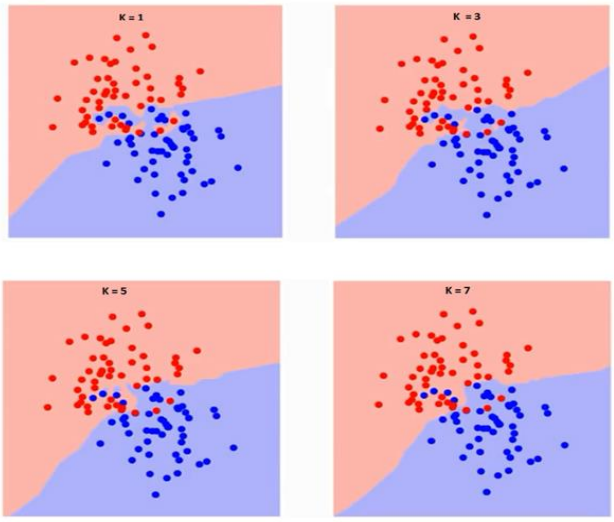


Figure 7. Different value of k

Given that the value of k is non-parametric, setting k as an odd number can avoid choosing the class from two groups with the same number. Moreover, a typical strategy is to set k as:

$$k = \frac{\sqrt{n}}{2}$$

where n is the number of data points in the training data.

Hyperparameter	Values
n_neighbors	1, 3, 5, 7, 9, 11, 13, 15, 17, 19

Table 1. Hyperparameters for KNN

2) The choice of distance function
   Both L1 norm and L2 norm can be used as the similarity metric. The main difference between two distance functions lies in that L1 norm calculates the sum of absolute values of vectors while L2 norm computes the square root of the sum of the squared feature weights. Compared with L1 norm, L2 norm is more unforgiving, that is, imposing less penalization, to large values. In this study, we applied L2 norm since it is a closed form solution which requires less computational cost.

3.5.2 Logistic Regression

For logistic regression, the classifier is optimized by selecting the hyperparameter Cs to improve the accuracy, and Cs is used to describe the reciprocal of the regularization intensity. Smaller values specify stronger regularization. So, the value of the hyperparameter Cs will not be too large. The range of values for the hyperparameters selected in this assignment is as follows:

Hyperparameter	Values
Cs	1, 2, 3, 4, 5

Table 2. Hyperparameters for LR

3.5.3 Random Forest Classifier

1) The choice of n_estimates

   n_ estimates represents the number of decision trees used for Random Forest. Typically, with more trees included in the forest, the model can perform better, since it can make reliable predictions based on a large amount of knowledge from these trees. However, there is a tradeoff that should be considered. While the accuracy is enhanced by adding the number of trees, it also increases the computational cost.

2) The choice of searching strategies

   There are two most used ways for choosing the candidate hyperparameter values from the dataset, which are grid search and random search. In general, grid search is the simplest search strategy in which every possible combination of parameters is evaluated. Random search is another way for selecting a parameter grid in which the values of hyperparameters are randomly chosen from a specified hyperparameter space.

   In our classifier, random search method is applied since it turns out to be more efficient to find the optimal hyperparameters compared with K-Fold.

Hyperparameter	Values
n_estimates	10,50,100
Search strategy	Random search

Table 3. Hyperparameters for random forest

How to choosing the number of n_estimates is based on Chen S, He LL and Zheng JH's research, they found and select some numbers for parameter. When the data is trained by Random Forest Classifier with the parameters they used, the result of accuracy is good. Therefore, the 10,50,100 are selected.

3.5.4 Linear SVC

Linear SVC is a branch of SVM. It can only compute linear kernels. Therefore, Linear SVC will calculate much faster than SVC and kernel input linear parameters. Many people who study SVC and SVM like to use this method. The parameters of this function are penalty, loss, dual, tol, C, multi_class, fit_intercept and class_weight.

parameter	Values
C	1,2
penalty	l2

Table 4. Hyperparameters for random forest

4 EXPERIMENT AND DISCUSSION

4.1 Overall analysis

4.1.1 Parallel Computing

Parallel Computing refers to the process of solving problems by using multiple computing resources at the same time. It is an effective means to improve the computing speed and processing power of computer systems. Its basic idea is to use multiple processors to solve the same problem collaboratively, and to break the problem into several parts, each part is calculated in parallel by an independent processor. Some jobs were selected in the classifier to run and calculate multiple processes, this can increase the running speed, but too many jobs cannot be used, because too many jobs may cause computer crash.

4.1.2 Cross-Validation

Cross-validation is a common method used in machine learning to build models and validate model parameters. Cross-validation is repeated usage data. The Fashion-MNIST data set is divided into 10 parts, of which 9 are training sets and one is validation Set. The training set is used to train the model, and the validation set is used to evaluate the prediction of the model. As shown in Figure 8.

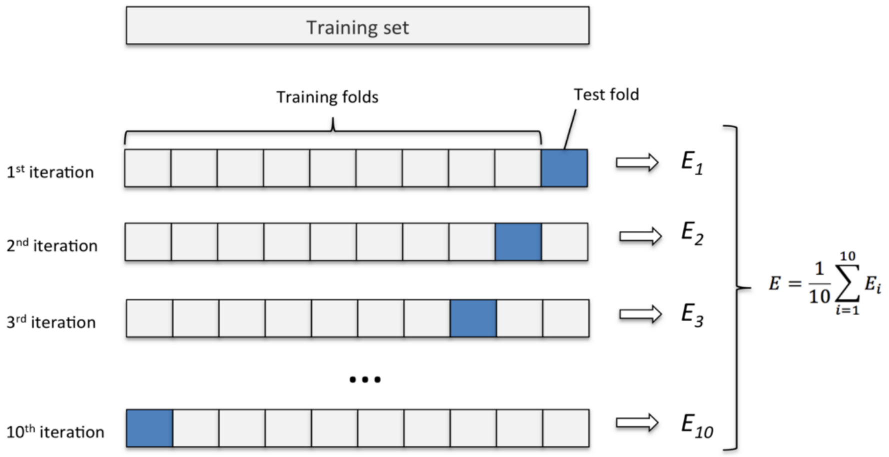


Figure 8: Cross-validation

Cross-validation is used when the data is not sufficient. For example, in my daily project, if the data sample size is less than 10,000 for normal moderate problems, we will use cross-validation to train the optimization selection model.

Based on different methods of segmentation. Cross-validation is divided into the following three types: simple cross-validation, S-Folder Cross Validation, Leave-one-out Cross Validation. Currently, the one of the most popular method is the K-Fold for segmentation.

4.1.3 Overall Comparison

Classifiers	k-Nearest Neighbors	Logistic Regression	Random Forest	Linear SCV
Run Time	23 s	71s	70s	107s
Accuracy	86.19%	83.53%	86.21%	86%

Table 5. Accuracy of different classifier

4.2 K-Nearest Neighbors Classification Results

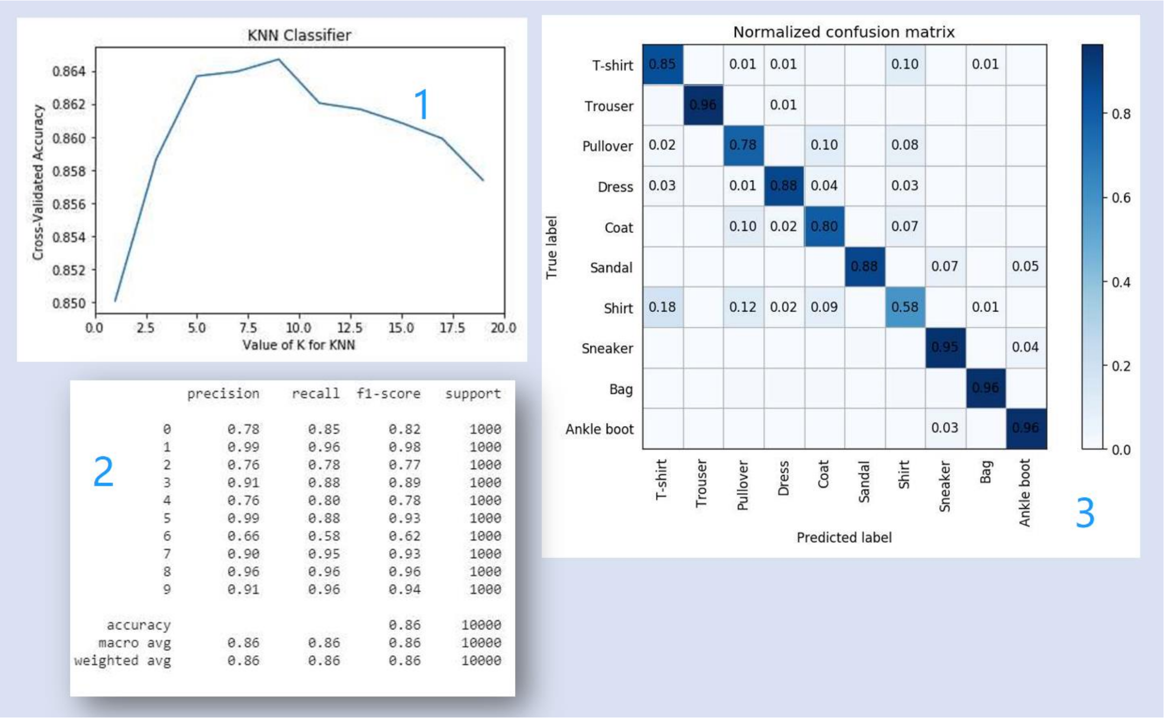
First of all, three methods of dimensionality reduction (PCA, SVD and Autoencoder) already be used, but different methods can make the different result in one same classifier, because their algorithms of dimensionality reduction are different.

For the KNN classifier, the Accuracy is the highest after which dimension reduction method is selected first. This process must have all the parameters of the classifier are consistent.

For the KNN classifier, the one method of dimensionality reduction should be selected when the predicted accuracy is the highest. This process must have the same value for all parameters of the classifier. Then the data can be used from this dimension reduction method.

Dimension	Classifier	Accuracy
84D	PCA	0.8615
84D	SVD	0.8611
84D	Autoencoder	0.8353

Table 5. Accuracy of KNN used the different dimensionality reduction method to process raw data



```
              precision    recall  f1-score   support

           0       0.78      0.85      0.82      1000
           1       0.99      0.96      0.98      1000
           2       0.76      0.78      0.77      1000
           3       0.91      0.88      0.89      1000
           4       0.76      0.80      0.78      1000
           5       0.99      0.88      0.93      1000
           6       0.66      0.58      0.62      1000
           7       0.90      0.95      0.93      1000
           8       0.96      0.96      0.96      1000
           9       0.91      0.96      0.94      1000

    accuracy                           0.86     10000
   macro avg       0.86      0.86      0.86     10000
weighted avg       0.86      0.86      0.86     10000
```

	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot
T-shirt	0.85		0.01	0.01			0.10		0.01	
Trouser		0.96		0.01						
Pullover	0.02		0.78		0.10		0.08			
Dress	0.03		0.01	0.88	0.04		0.03			
Coat			0.10	0.02	0.80		0.07			
Sandal						0.88		0.07		0.05
Shirt	0.18		0.12	0.02	0.09		0.58		0.01	
Sneaker								0.95		0.04
Bag									0.96	
Ankle boot								0.03		0.96

Figure 9: Classification information for K-NN

(1,2,3 in figure 9 is Cross-validation, evaluation metrics and confusion matrix)

According to the data of the table 5, the better method of dimensionality reduction for KNN is PCA (the accuracy is 0.8615). So, I will use the PCA as the dimensionality reduction method.

According to the Cross-validation, the result shows a small range of K value change (1 - 19) did not affect much on accuracy (0.85-0.86). Additionally, the model is able to achieve relatively high accuracy because run time is only 20s.

According to the evaluation matrix, some labels can git higher F1-Scores and some get the lower scores. The matrix shows that it could correctly predict class label 1 (0.99) and 5 (0.99), however, it cannot better predict on label 6 (0.62). Overall, the performance of this classification method is relatively high and run time is fast. The performance of accuracy is good. It could be proved that K-NN Classifier model and PCA Preprocessing together could have a great performance.

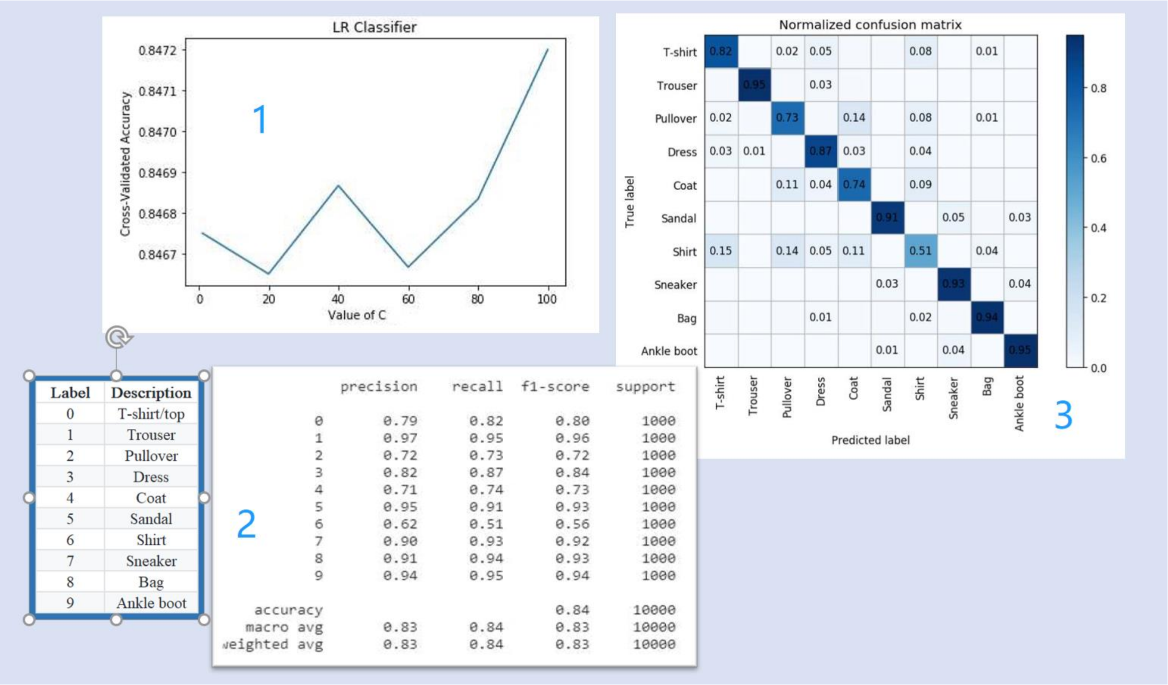
According to the confusion matrix (Normalized), I can find the predicted accuracy of some labels is very good. For example, this shows that most of the 1000 samples of class trouser are classified correctly. However, to many samples of class shirt are classified incorrectly.

4.3 Logistic Regression

The way used to find the appropriate dimensionality reduction method is similar to that of KNN. (To find suitable dimensionality reduction methods by the accuracy.)

Dimension	Classifier	Accuracy
84D	PCA	0.8342
84D	SVD	0.8353
84D	Autoencoder	0.7416

Table 6. Accuracy of LR used the different dimensionality reduction method to process raw data



Label	Description
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

```
              precision    recall  f1-score   support

           0       0.79      0.82      0.80      1000
           1       0.97      0.95      0.96      1000
           2       0.72      0.73      0.72      1000
           3       0.82      0.87      0.84      1000
           4       0.71      0.74      0.73      1000
           5       0.95      0.91      0.93      1000
           6       0.62      0.51      0.56      1000
           7       0.90      0.93      0.92      1000
           8       0.91      0.94      0.93      1000
           9       0.94      0.95      0.94      1000

    accuracy                           0.84     10000
   macro avg       0.83      0.84      0.83     10000
weighted avg       0.83      0.84      0.83     10000
```

Normalized confusion matrix

True label \ Predicted label	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot
T-shirt	0.82		0.02	0.05			0.08		0.01	
Trouser		0.95		0.03						
Pullover	0.02		0.73		0.14		0.08		0.01	
Dress	0.03	0.01		0.87	0.03		0.04			
Coat			0.11	0.04	0.74		0.09			
Sandal						0.91		0.05		0.03
Shirt	0.15		0.14	0.05	0.11		0.51		0.04	
Sneaker						0.03		0.93		0.04
Bag				0.01			0.02		0.94	
Ankle boot						0.01		0.04		0.95

Figure 10: Classification information for LR

(1,2,3 in figure 10 is Cross-validation, evaluation metrics and confusion matrix)

According to the data of the table 6, the better method of dimensionality reduction for LR is SVD (the accuracy is 0.8353). So, I will use the SVD as the dimensionality reduction method.

According to the Cross-validation, the result shows a small range of C value change (0 - 100) did not affect much on accuracy (0.846-0.847). However, the accuracy of LR is not better than KNN.

According to the evaluation matrix, some labels can git higher F1-Scores and some get the lower scores. The matrix shows that it could correctly predict class label 1 (0.96) and 5 (0.93), however, it cannot better predict on label 6 (0.56). Overall, the performance of this classification method is not particularly high and run time is not too fast. But it could be proved that LR Classifier model and SVD Preprocessing together could have a great performance. So, logistic regression algorithms require further research and optimization.

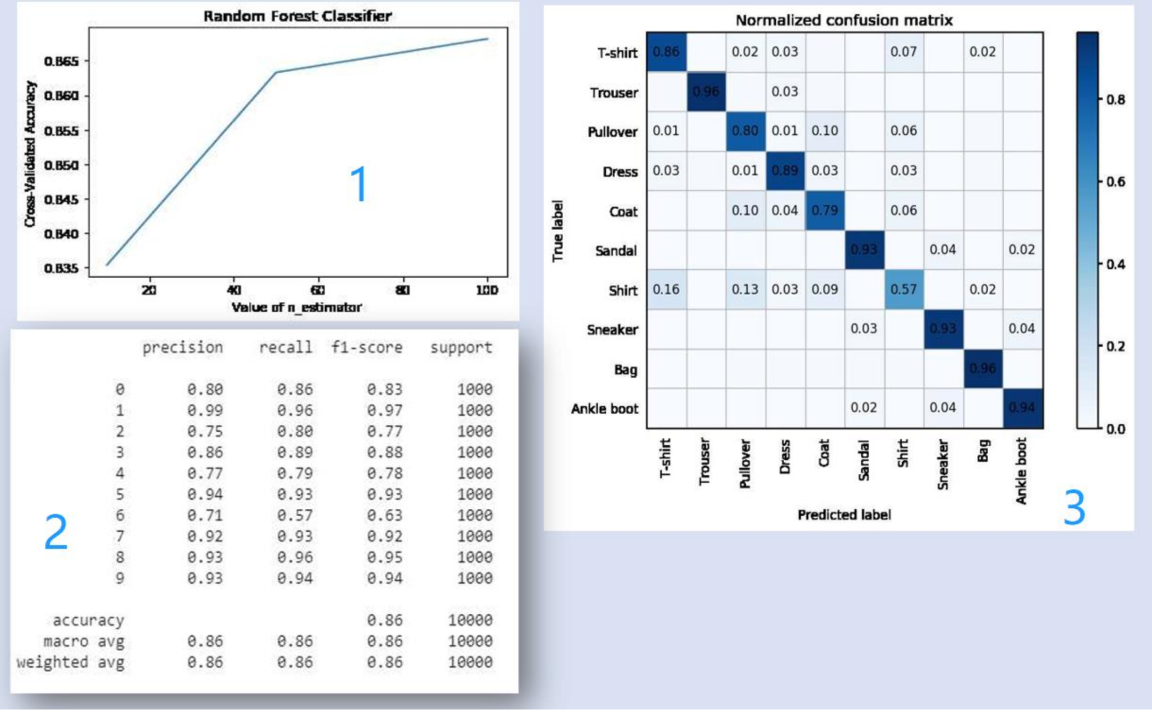
According to the confusion matrix (Normalized), I can find the predicted accuracy of some labels is very good. For example, this shows that most of the 950 samples of class trouser are classified correctly. However, the half of samples of class shirt are classified incorrectly.

4.4 Random Forest

The table below shows a summary of the different dimensionality reduction methods and the results of accuracy.

Dimension	Classifier	Accuracy
84D	PCA	0.8615
84D	SVD	0.8621
84D	Autoencoder	0.8353

Table 7. Accuracy of Random Forest used the different dimensionality reduction method to process raw data



```
              precision    recall  f1-score   support

           0       0.80      0.86      0.83      1000
           1       0.99      0.96      0.97      1000
           2       0.75      0.80      0.77      1000
           3       0.86      0.89      0.88      1000
           4       0.77      0.79      0.78      1000
           5       0.94      0.93      0.93      1000
           6       0.71      0.57      0.63      1000
           7       0.92      0.93      0.92      1000
           8       0.93      0.96      0.95      1000
           9       0.93      0.94      0.94      1000

    accuracy                           0.86     10000
   macro avg       0.86      0.86      0.86     10000
weighted avg       0.86      0.86      0.86     10000
```

Normalized confusion matrix

	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot
T-shirt	0.86		0.02	0.03			0.07		0.02	
Trouser		0.96		0.03						
Pullover	0.01		0.80	0.01	0.10		0.06			
Dress	0.03		0.01	0.89	0.03		0.03			
Coat			0.10	0.04	0.79		0.06			
Sandal						0.93		0.04		0.02
Shirt	0.16		0.13	0.03	0.09		0.57		0.02	
Sneaker						0.03		0.93		0.04
Bag									0.96	
Ankle boot						0.02		0.04		0.94

Figure 11: Classification information for RF

(1,2,3 in figure 10 is Cross-validation, evaluation metrics and confusion matrix)

According to the data of the table 7, the better method of dimensionality reduction for RF is SVD (the accuracy is 0.8621). So, I will use the SVD as the dimensionality reduction method.

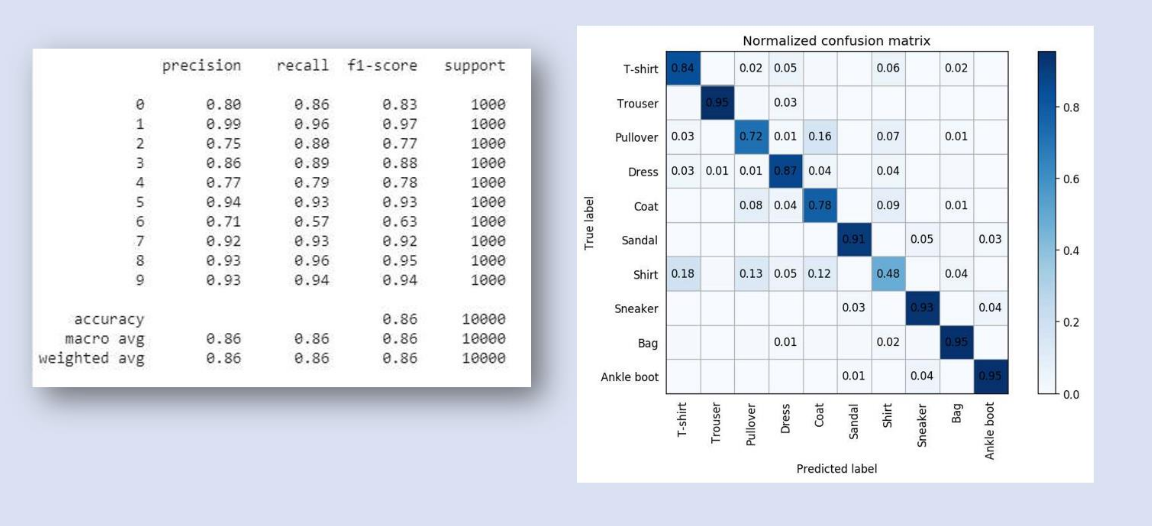
According to the Cross-validation, the result shows a big range of n_estimates value change (10,50, 100) did not affect much on accuracy (0.835-0.870). However, the accuracy of RF is better than KNN, and the running time is longer than the running time of the KNN classifier. Its running time is similar to LR.

According to the evaluation matrix, some labels can git higher F1-Scores and some get the lower scores. The matrix shows that it could correctly predict class label 1 (0.97) and 5 (0.93), however, it cannot better predict on label 6 (0.63). Overall, the performance of this classification method is not particularly high and run time is fast. But it could be proved that RF Classifier model and SVD Preprocessing together could have a great performance.

According to the confusion matrix (Normalized), I can find the predicted accuracy of some labels is very good. For example, this shows that most of the 960 samples of class trouser are classified correctly. However, the half of samples of class shirt are classified incorrectly.

4.5 Linear SVC

The table below shows a summary of the different dimensionality reduction methods and the results of accuracy.



```
              precision    recall  f1-score   support

           0       0.80      0.86      0.83      1000
           1       0.99      0.96      0.97      1000
           2       0.75      0.80      0.77      1000
           3       0.86      0.89      0.88      1000
           4       0.77      0.79      0.78      1000
           5       0.94      0.93      0.93      1000
           6       0.71      0.57      0.63      1000
           7       0.92      0.93      0.92      1000
           8       0.93      0.96      0.95      1000
           9       0.93      0.94      0.94      1000

    accuracy                           0.86     10000
   macro avg       0.86      0.86      0.86     10000
weighted avg       0.86      0.86      0.86     10000
```

Normalized confusion matrix

	T-shirt	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot
T-shirt	0.84		0.02	0.05			0.06		0.02	
Trouser		0.95		0.03						
Pullover	0.03		0.72	0.01	0.16		0.07		0.01	
Dress	0.03	0.01	0.01	0.87	0.04		0.04			
Coat			0.08	0.04	0.78		0.09		0.01	
Sandal						0.91		0.05		0.03
Shirt	0.18		0.13	0.05	0.12		0.48		0.04	
Sneaker						0.03		0.93		0.04
Bag				0.01			0.02		0.95	
Ankle boot						0.01		0.04		0.95

Figure 12: Classification information for linear SVC

(evaluation metrics and confusion matrix)

According to the evaluation matrix, some labels can git higher F1-Scores and some get the lower scores. The matrix shows that it could correctly predict class label 1 (0.97) and 5 (0.93), however, it cannot better predict on label 6 (0.63). Overall, the performance of this classification method is not particularly high and run time is not fast.

According to the confusion matrix (Normalized), I can find the predicted accuracy of some labels is very good. For example, this shows that most of the samples of class trouser are classified correctly. However, the half of samples of class shirt are classified incorrectly.

After the analysis and results of the above classifier, the probability of the shirt being recognized wrong is extremely high. Regardless of the method, its recognition rate has been very low, which may indicate that there are some problems with the image features, or some potential problems. not found. This requires us to continue research and analysis on this issue.

5 DISCUSSION

The experiments show that 4 models have different influence on the classification accuracy. It turns out that RF performs best with the accuracy of 86.21%, while LR presented the lowest accuracy (83.53%). The runtime between two classifiers shows little difference. However, it shows that multiple tasks running simultaneously could affect the runtime and accuracy. On the other hand, although KNN requires much computing time, its simplicity to interpret and implement makes it a practical method for classification. Moreover, using Linear SVC does not have remarkable improvement on the accuracy while it takes the longest runtime among all the classifiers. Therefore, for future work, other techniques could be explored to further improve the performance in Linear SVC.

The results also present that while all the preprocessing methods significantly improve the accuracy on the result, they could lead to different level of accuracy due to different algorithms they apply.

Therefore, by comparing them on each classifier, we determine the optimal preprocessing technique for each classifier which shows the highest accuracy. Meanwhile, there are many advantages regarding the Autoencoder, which is an unsupervised neural network that learns from multiple convolutional layers, making it more efficient compared with PCA. However, it is proven that Autoencoders show better performance on the dataset with high dimensionality. Hence, we only apply PCA and SVD in the experiments since the given dataset of Fashion MINST has low dimension.

Overall, by comparing the performance of 4 classifiers, it shows that while all of them can be used for multiclass classification problems, each of them has different impact on the outcome. Therefore, it is crucial to choose a specific classifier on a given dataset. Furthermore, considering there are some limitations on the Fashion-MINIST dataset, we will try to train these classifiers using a more reliable dataset in the future. In addition, when it comes to the choice of preprocessing techniques, more factors could be considered since we only keep the default value of parameters in this study.

6. CONCLUSION

The experiments show that our proposed methods present good performance for multiclass image classification. In particular, we are able to achieve a higher accuracy of 86.21% on Fashion MINIST dataset by using Random Forest as compared to other models. It also proves that using preprocessing techniques, such as PCA and SVD, can significantly help improve the overall classification accuracy and reduce the computational cost.

Moreover, it should be pointed out that the performance of different classifiers is closely related to the characteristics of the given data. Therefore, there has been no classification method that works best on any given problem. [10] In future work, we will try to train different classifiers to get the best performance by using other techniques through trial and error. Furthermore, considering there are some limitations regarding Fashion MINIST dataset, we will also use other kinds of image categories to explore the difference among various classification methods.

REFERENCE

[1] S. Bhatnagar, D. Ghosal and M. Kolekar, "Classification of fashion article images using convolutional neural networks", 2017 Fourth International Conference on Image Information Processing (ICIIP), 2017. Available: 10.1109/iciip.2017.8313740 [Accessed 30 May 2019].

[2] A. Ferreira and G. Giraldi, "Convolutional Neural Network approaches to granite tiles classification", Expert Systems with Applications, vol. 84, pp. 1-11, 2017. Available: 10.1016/j.eswa.2017.04.053 [Accessed 30 May 2019].

[3] Z. Qi, Z. Jiang, C. Yang, L. Liu and Y. Rao, "Identification of maize leaf diseases based on image technology", Journal of Anhui Agricultural University, vol. 43, no. 2, pp. 325-330, 2016. [Accessed 31 May 2019].

[4] A. Agarap, "An Architecture Combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) for Image Classification", Ph.D, Cornell University, 2017.

[5] Y. Seo and K. Shin, "Hierarchical convolutional neural networks for fashion image classification", Expert Systems with Applications, vol. 116, pp. 328-339, 2019. Available: 10.1016/j.eswa.2018.09.022 [Accessed 30 May 2019].

[6] K. Hoang, "Image Classification with Fashion-MNIST and CIFAR-10", 2019.

[7] G. K V and S. K, "Fashion-MNIST Classification Based on HOG Feature Descriptor Using SVM", International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 8, no. 5, pp. 960-962, 2019. [Accessed 31 May 2019].

[8] Niklaus Wirth, in M. Broy, Ernst Denert Software Pioneers: Contributions to Software Engineering: Programme Development by Stepwise Refinement, Springer, 27 June 2002, p.151

[9] B. Mathews, "Image Compression using Singular Value Decomposition (SVD)", 2014.

[10] M. Jha, Applied mathematics in electrical & computer engineering. [Greece]: WSEAS, 2012, pp. 133-138.

APPENDIX

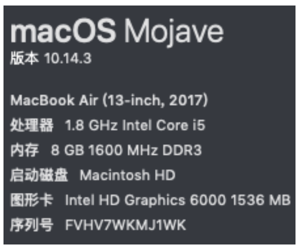
Dataset

MNIST-fashion

```
from tensorflow.keras.datasets import fashion_mnist
```

```
fashion_mnist.load_data()
```

Hardware and Software specifications



macOS Mojave
版本 10.14.3

MacBook Air (13-inch, 2017)
处理器 1.8 GHz Intel Core i5
内存 8 GB 1600 MHz DDR3
启动磁盘 Macintosh HD
图形卡 Intel HD Graphics 6000 1536 MB
序列号 FVHV7WKMJ1WK

How to run the code:

1. There are two files saved in ipynb format so that you can run them on google colab or jyputer or the envirement with Python3.6 or 3.7 version.
2. You don't need to load any datasets from outside.Just run the function fashion_mnist.load_data() in the first part of code in Assignment2_main_algorithm.It can be directly to load.
3. All kfold or cross validation process and result had been ran and done in the Assignment2_cross_validation file.You can check or see them the without run again because most of function will take long time run.
4. We implemented four methods (Knn, Logistic Regression, Random Forest and LinearSVC) in the Assignment2_main_algorithm file.
5. Run the code in the main_algorithm file from top to buttom step by step.
6. If you run in the jupyter or python environment may be need install some package before run the code. using !pip install 'package name' if some error happend.