Contents

1 Introduction **2**

2 Introduction to the training set **2**

3 Naive Bayes **3**

3.1 Naive Bayes classification model 3

3.2 Design steps 5

3.3 Results 8

3.4 Algorithm Features 9

4 KNN algorithm **10**

4.1 Introduction to KNN Algorithm 10

4.2 Design steps 10

4.3 Results 13

4.4 Algorithm Features 13

5 Performance comparison **14**

6 Conclusion **14**

References **16**

appendix **17**

1 Introduction

Handwritten digit recognition is a traditional research field of pattern recognition. It mainly studies how to use computer to automatically recognize handwritten Arabic numerals. It is divided into offline handwritten digit recognition and online handwritten digit recognition. This paper mainly studies the offline handwritten digit recognition. Arabic numerals are widely used and are common international symbols, and often involve strict and sensitive fields such as finance and finance [1] . Therefore, it is of great significance to improve the accuracy of recognition and reduce the rate of misrecognition and rejection. posed a huge challenge.

Machine learning is an important approach to pattern recognition, which can be regarded as different aspects of the same field. The scientific process of each disciplinary field has its own characteristics, but observation, creation of hypotheses, testing based on decisive experiments or observations, and models or theories that can be tested intelligibly are common to all disciplines, and every aspect of its abstraction process , and machine learning have developed accordingly, leading to the proper, partial automation of all aspects of the scientific method, from hypothesis generation and model construction to decisive experiments.

Using effective machine learning methods, the observed data can be converted into models, which can solve many practical problems such as network data, financial data, biological data, meteorological data, seismic observation data, classified data, etc. intelligent.

This paper designs a handwritten digit recognition system by using two algorithms, Naive Bayes classification and KNN classification, and introduces the steps, results and characteristics of these two algorithms.

2 Introduction to the training set

Yann LeCun used NIST (National Institute of Standards and Technology's handwritten digit library to build a subset MNIST that is convenient for machine learning research. MNIST consists of 70,000 handwritten digits (0~9) pictures (grayscale images), which are composed of many different Written by people, of which 60,000 are the training set and the other 10,000 are the test set, each image is 28 x 28 pixels in size, and the number is 20 x 20 in the center of the image.

3 Naive Bayes

3.1 Naive Bayes classification model

NBC is the simplest model in the Bayesian classification model. It is a tree structure composed of a parent node and multiple child nodes. The schematic diagram of the structure is shown in Figure 1 [2]. It assumes that when the class attribute variables are known, each condition attribute variable is independent of each other. However, in real life, the assumption that attributes are independent of each other does not hold in many cases, that is, when there is a strong dependence between attributes, this assumption will have a negative impact on the classification accuracy of NBC, but on the other hand, it will greatly affect the classification accuracy of NBC [3]. It simplifies the complexity of building a Bayesian classification model, so NBC has the characteristics of simplicity and efficiency, so it is widely used.

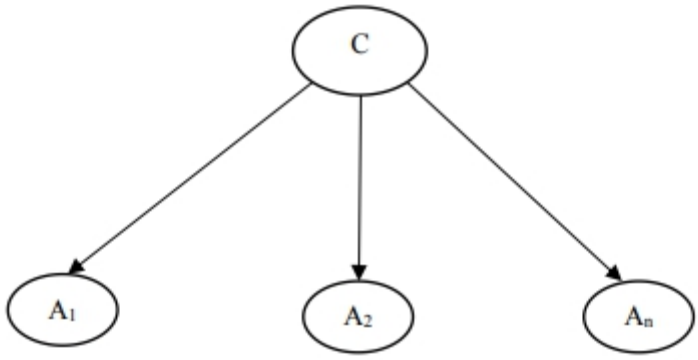

Figure 1 Naive Bayes Model

Assume that input training dataset $D=\{X_1,X_2,\cdots X_p\}$,where $A_1,A_2,\cdots,A_n$ represents attributions, C represents class $C_1,C_2,\cdots,C_m$ .Samples to be classified $X=\langle x_1,x_2,\cdots,x_n\rangle$,where x_i is value of A_i ,The probability that X belongs to C_k is:

$$P(C_k \mid X)=\frac{P(X \mid C_k)P(C_k)}{P(X)} \qquad (1\text{-}1)$$

Due to calculate the value of $P(X \mid C_k)$ is complex,Assuming that attribute variables are independent of each other:

$$P(X \mid C_k) = \prod_{i=1}^{n} P(x_i \mid C_k) \tag{1-2}$$

Since P(X) is a constant ,just maximize $P(X \mid C_k)P(C_k)$, the NBC is obtained according to the maximum posterior hypothesis as:

$$C(X) = \arg\max_{C_k \in C} P(C_k) \prod_{i=1}^{n} P(x_i \mid C_k) \tag{1-3}$$

The probability and specific calculation formula of $P(C_k)$ & $P(x_i \mid C_k)$ are as follows:

$$P(C_k) = \frac{\sum_{j=1}^{n} \delta(C_j, C_k) + 1}{n + |C|} \tag{1-4}$$

$$P(x_i \mid C_k) = \frac{\sum_{j=1}^{n} \delta(x_{ji}, x_i)\delta(C_j, C_k) + 1}{\sum_{j=1}^{n} \delta(C_j, C_k) + |A_i|} \tag{1-5}$$

where n is the number of training instances, C_j is the class label of the j instance, and x_{ji} represents the value of the column i and row j training instance, where:

$$\delta(C_j, C_k) = \begin{cases} 1, j = i \\ 0, j \neq i \end{cases} \tag{1-6}$$

The specific implementation process of NBC can be represented by the following figure 2[4]:

The implementation of the entire NBC is mainly divided into three steps:

Step 1 Data preprocessing. The purpose of this process is to pave the way for the classification of NBC, convert all the acquired data into numerical types, and fill in the missing data. If it is continuous data, it needs to be discretized. Finally, the data It is divided into two parts: training data and test data.

Step 2: Construct the classification model. The task of this process is to learn the classification model, and then count the frequency of occurrence of each category in

the training instance and the conditional probability value of the conditional attribute variable in each category. The input is the training sample, and the output is the classification model.

Step 3 Test the classification model. The task of this process is to use the classification model to classify the test data set, its input is the classification model and the test data set, and the output is the classification result of the test data set.

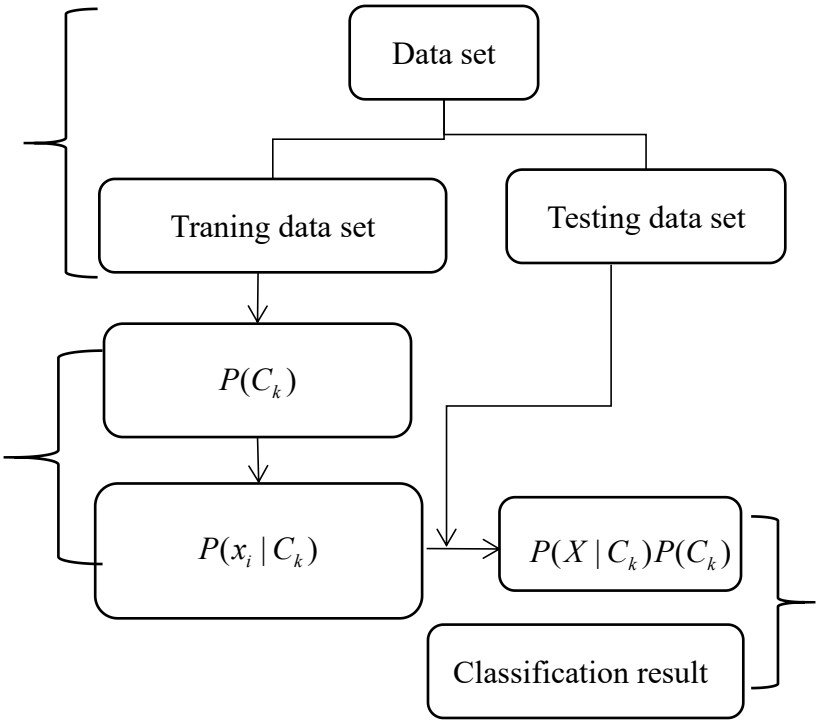

Figure 2 Flowchart of the Naive Bayes classification model

3.2 Design steps

Step 1: Convert the dataset to mat data.

(1). Convert the image to mat data.

```
function imgsmat = gzimgs2mat(fn)
fp = fopen('D:\研究生文件\研究生课程\威尔士\Artificial
Intelligence\t10k-labels.idx1-ubyte', 'rb');

assert(fp ~= -1, ['该文件无法打开', fn, '']);
ma = fread(fp,1,'uint32');
assert(ma == 2051, ['逻辑出错 ', fn, '']);
imgsNum = fread(fp, 1, 'uint32', 0, 'ieee-be');
rows = fread(fp, 1, 'uint32', 0, 'ieee-be');
cols = fread(fp, 1, 'uint32', 0, 'ieee-be');
imgs = fread(fp, inf, 'uint8');
imgs = reshape(imgs, rows, cols, imgsNum);
imgs = permute(imgs,[2 1 3]);

imgs = reshape(imgs, size(imgs, 1) * size(imgs, 2), size(imgs, 3));
imgs = double(imgs) / 255;
fclose(fp);

save train_images.mat imgs
```

(2) Convert the labels to mat data

```
function labelsmat = gz2mat(filename)


fp = fopen('D:\研究生文件\研究生课程\威尔士\Artificial
Intelligence\t10k-labels.idx1-ubyte', 'rb');
assert(fp ~= -1, ['该文件无法打开', filename, '']);

ma = fread(fp,1,'uint8');
assert(ma == 2049, ['逻辑出错 ', filename, '']);

numLabels = fread(fp, 1, 'uint8', 0, 'ieee-be');

labels = fread(fp, inf, 'uint8');

assert(size(labels,1) == numLabels, '出错的 label 数量为');

fclose(fp);

save test_labels.mat labels

end
```

Step 2:

Convert mat data to vector

```
%    mat2vector.m

function [data_]= mat2vector(data,num)
        [row,col,~] = size(data);
        data_ = zeros(num,row*col);
        for page = 1:num
                for rows = 1:row
                        for cols = 1:col
                                data_(page,((rows-1)*col+cols))                                =
im2double(data(rows,cols,page));
                        end
                end
        end
end
```

Step 3:

Process the data once. The Bayesian classifier requires that the variance of a feature in the same category in the input data cannot be 0. We can know from the formula that when calculating the posterior probability, if all the features in a column are the same, then the posterior probability is calculated. When the denominator will become 0, in order to avoid this situation, the data needs to be processed in advance. data_proc.m is a preprocessing function for image data, mainly to solve the problem that all the column vectors in the training sample feature space are zero, so that There are cases where the denominator is zero when calculating the posterior probability.

```
% data_proc.m
function [output,position] = data_proc(data,label)
	position = cell(1,10);

	for i = 0:9
		temp = [];
		pos = [];
		for rows = 1:size(data,1)
			if label(rows)==i
				temp = [temp;data(rows,:)];
			end
		end
		for cols = 1:size(temp,2)
			var_data = var(temp(:,cols));
			if var_data==0
				pos = [pos,cols];
			end
		end
		position{i+1} = pos;
		data(:,pos)=[];
	end
```

Step 4:

The processed data can be put into model training, using the matlab naive Bayes model fitcnb function.

```
nb_model = fitcnb(data_train,train_labels1(1:train_num));
```

3.3 Results

The results are as follows:

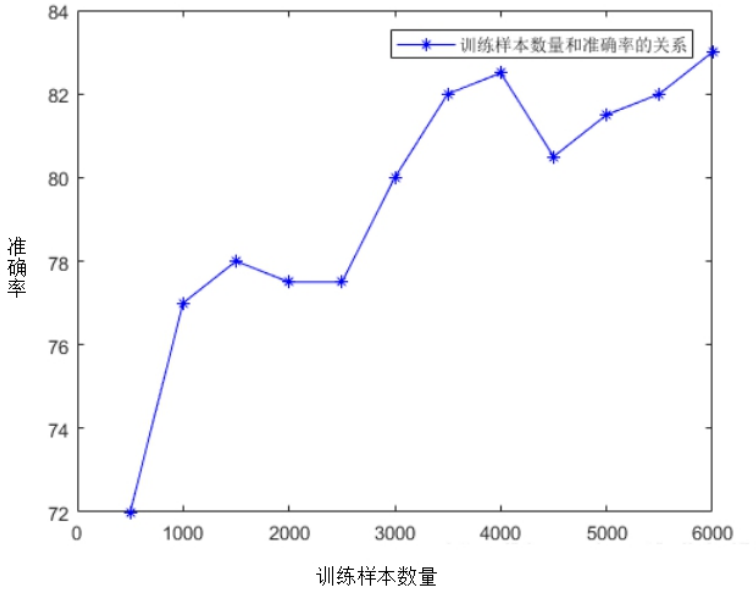

Figure 3 Running results

3.4 Algorithm Features

Advantages of Naive Bayes:

1) The principle of the naive Bayesian model comes from the mathematical ideas in mathematical statistics, and the classification effect of this model is relatively stable.

2) It works well on a small number of data sets, and the naive Bayes model can also be used to handle multi-classification tasks. For example, when the amount of data exceeds the upper memory limit of the machine, the data can be divided into the model. Do incremental training.

Disadvantages of Naive Bayes:

1) From the model principle of Naive Bayes, its correct rate should be significantly higher than other classification methods. But this is not the case in this experiment. The accuracy rate is significantly lower than that of the KNN algorithm. The reason may be that the X, Y... attributes in the assumption of the Naive Bayes model are independent of each other, and it is difficult to have completely independent properties in real life. thing. When the independence between attributes is weak, the classification effect is not good, and when the independence of attributes is strong, the performance of Naive Bayes is the best.

2) The prior probability needs to be known, and the prior probability often depends on the hypothesis. There can be many kinds of hypothetical models, so in some cases, the prediction effect will be poor due to the hypothesized prior model.

3) Since we determine the probability of the posterior through the prior and data to determine the classification, there is a certain error rate in the classification decision.

4) It is very sensitive to the expression form of the input data.

.

.

4 KNN algorithm

4.1 Introduction to KNN Algorithm

KNN is a relatively simple algorithm in classification algorithms, both in principle and understanding, and it is also widely used. The KNN algorithm is a supervised learning algorithm. In short, it needs to know the answer and do the question at the same time, which is essentially different from the unsupervised learning algorithm K-means.

For distance calculation, if you want to measure the distance of the midpoint of the space, there are several measurement methods, such as the common Manhattan distance calculation, Euclidean distance calculation, etc [6] . However, the Euclidean distance is usually used in the KNN algorithm, here is just a brief introduction, take two Taking the 2D plane as an example, the formula for calculating the Euclidean distance between two points in a 2D space is as follows：

$$\rho = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2-1}$$

Extending to multi-dimensional space, the formula is:

$$d(x, y) = \sqrt{(x_1 - y_1)_2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{2-2}$$

4.2 Design steps

Step 1:

(1) Convert the images dataset to mat data.

```
function imgsmat = gzimgs2mat(fn)

fp = fopen('D:\研究生文件\研究生课程\威尔士\Artificial
Intelligence\t10k-labels.idx1-ubyte', 'rb');
assert(fp ~= -1, ['该文件无法打开', fn, '']);

ma = fread(fp,1,'uint32');
assert(ma == 2051, ['逻辑出错 ', fn, '']);

imgsNum = fread(fp, 1, 'uint32', 0, 'ieee-be');
rows = fread(fp, 1, 'uint32', 0, 'ieee-be');
cols = fread(fp, 1, 'uint32', 0, 'ieee-be');
imgs = fread(fp, inf, 'uint8');
imgs = reshape(imgs, rows, cols, imgsNum);
imgs = permute(imgs,[2 1 3]);

imgs = reshape(imgs, size(imgs, 1) * size(imgs, 2), size(imgs, 3));
imgs = double(imgs) / 255;
fclose(fp);

save train_images.mat imgs

end
```

(2) Convert the labels data set to mat data

```
function labelsmat = gz2mat(fn)


fp = fopen('D:\研究生文件\研究生课程\威尔士\Artificial
Intelligence\t10k-labels.idx1-ubyte', 'rb');
assert(fp ~= -1, ['该文件无法打开', fn, '']);

ma = fread(fp,1,'uint8');
assert(ma == 2049, ['逻辑出错 ', fn, '']);

numLabels = fread(fp, 1, 'uint8', 0, 'ieee-be');

labels = fread(fp, inf, 'uint8');

assert(size(labels,1) == numLabels, '出错的 label 数量为');

fclose(fp);

save test_labels.mat labels

end
```

Step 2:

Convert mat data to vector

```
%    mat2vector.m

function [data_]= mat2vector(data,num)
      [row,col,~] = size(data);
      data_ = zeros(num,row*col);
      for page = 1:num
            for rows = 1:row
                  for cols = 1:col
                        data_(page,((rows-1)*col+cols))                    =
im2double(data(rows,cols,page));
                  end
            end
      end
end
```

Step 3:
(1) Select parameter K
(2) Calculate the distance between the unknown instance and all known instances (Euclidean distance)
(3) Select the most recent K known instances
(4) According to the majority-voting rule, the unknown instance is classified into the category with the most number among the K nearest neighbors..

4.3 Results

The results are as follows:

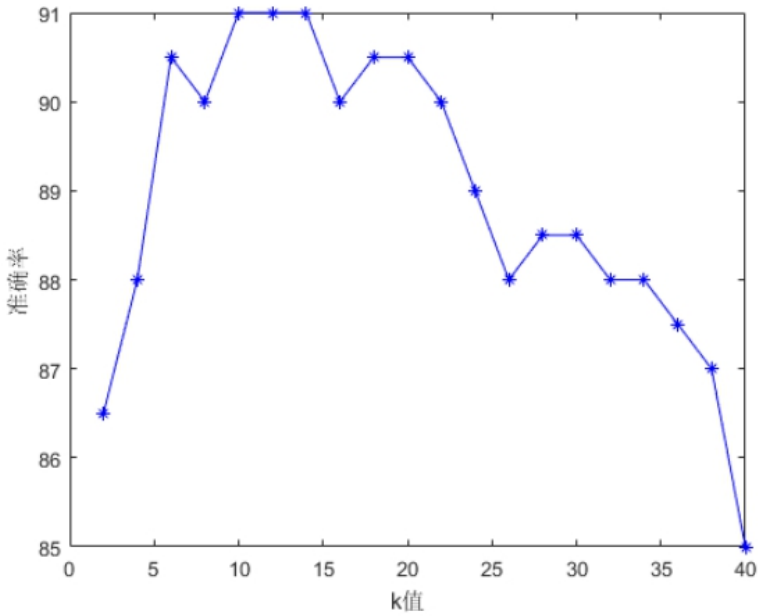

Figure 4 Running results

It can be seen that when the k value is 10, the accuracy rate is the highest, reaching 91%.

4.4 Algorithm Features

The advantages of the KNN algorithm: the principle is simple and clear, it is easier to understand, the model training speed is fast, the computer configuration requirements are not high, and the prediction results are highly accurate as well.
However, the disadvantages are also obvious: higher memory requirements, because the algorithm stores all training data. The prediction stage may be slow. Sensitive to

irrelevant features and data size.

5 Performance comparison

A better way to evaluate the performance of a classifier is a confusion matrix. The general idea is to count the number of times an instance of class A is divided into class B. For example, to know how many times the classifier confused the number 3 and the number 5, just pass the confusion matrix Look at row 5, column 3.

Naive		Predicted	
		T	F
Actual	T	847	161
	F	181	811
Accuracy= 1658/2000=0.829			

KNN		Predicted	
		T	F
Actual	T	937	86
	F	98	879
Accuracy = 1816/2000 =0.908			

6 Conclusion

By comparing the two methods for 2000 data, it is found that the accuracy of the KNN algorithm is higher than that of Naive Bayes.

When using the KNN algorithm, there is no need for additional processing of the data set, which means that it has a high degree of adaptation to the data, which can greatly save the cost of use. There are also many options for distance calculation. This

experiment uses Euclidean distance. However, although the KNN algorithm does not need to preprocess the data, the accuracy of the KNN algorithm can be higher for samples without extreme data.[8].

The naive Bayesian method has higher requirements for data and more preprocessing processes, so it is still applicable to the case of less data.

References

[1] Xiong Ling. Overview of Image Matching in Computer Vision [C]// Proceedings of Hubei Province, 2006 Academic Annual Meeting of Mechanical Engineering Society of 12 Provinces, Regions and Municipalities. 2006.
[2] Qin Feng, Ren Shiliu, Cheng Zekai, et al. Naive Bayesian Classification Algorithm Based on Attribute Weighting [J]. Computer Engineering and Applications, 2008, 44(6):3.
[3] Jiang Liangxiao. Naive Bayes classifier and its improved algorithm research [D]. China University of Geosciences, 2009.
[4] Xie Xiaojun. Improved Naive Bayesian Algorithm and Its Application in Image Classification [D]. Guilin University of Electronic Science and Technology, 2016.
[5] Zhang Junli, Zhang Fan. Application of Improved KNN Algorithm in Spam Filtering [J]. Modern Library and Information Technology, 2007(4):4.
[6] Bu Fanjun. Improvement of KNN algorithm and its application in text classification [D]. Jiangnan University, 2010.
[7] Cui Haixia, Yang Hong, Liu Zuolian. Research on Handwritten Digit Recognition of MNIST Postal Code [J]. Journal of Guangzhou University: Natural Science Edition, 2009(4):5.
[8] Bu Wenrui. Comparative Research on Parameter Optimization Algorithms Based on MNIST Data Set [J]. Electronic Technology and Software Engineering, 2021(11):2.

appendix

```
%Naive Bayesian.m
clear
clc
%test_images=load ('./test_images.mat');
%test_labels1=load ('./test_labels.mat');
%train_images=load ('./train_images.mat');
%train_labels1=load ('./train_labels.mat');
train_num = 2000;
test_num = 200;
%训练数据

%图像转向量
data_test = loadMNISTImages('D:\研究生文件\研究生课程\威尔士\Artificial
Intelligence\mnist\t10k-images.idx3-ubyte');
data_train = loadMNISTImages('D:\研究生文件\研究生课程\威尔士\Artificial
Intelligence\mnist\train-images.idx3-ubyte');
test_labels1 = loadMNISTLabels('D:\研究生文件\研究生课程\威尔士\Artificial
Intelligence\mnist\t10k-labels.idx1-ubyte');
train_labels1 = loadMNISTLabels('D:\研究生文件\研究生课程\威尔士
\Artificial Intelligence\mnist\train-labels.idx1-ubyte');
%data_train = mat2vector(train_images(:,:,1:train_num),train_num);
% 这里对数据进行一次处理。贝叶斯分类器要求输入数据中同类别中一个特征的方差不能
为 0
% 由公式我们可以知道，计算后验概率时，如果某一列特征全部相同，那么后验概率计算
时分母会变为 0，
% 为了避免该种情况，需要对数据进行提前处理
[data_train,position] =
data_proc(data_train,train_labels1(1:train_num)');
% 对训练数据进行处理后，同时也要对测试数据进行同样的处理
for rows = 1:10
data_test(:,position{1,rows})=[];
end
%模型部分
% 超参数全部取了默认值，比较重要的，如类别的先验概率，如果不进行修改，则计算输
入数据中类别的频率
% 查看 nb_model 即可确认所使用的超参数
nb_model = fitcnb(data_train,train_labels1(1:train_num));
```

```
%训练模型
%测试结果
result = predict(nb_model,data_test);
result = result.';
fprintf('预测结果：');
result(1:20)
%取 20 个打印出来对比
fprintf('真实分布：');
test_labels1(1:20)
% 整体正确率
acc = 0.;
for i = 1:test_num
if result(i)==test_labels1(i)
acc = acc+1;
end
end
fprintf('精确度为：%5.2f%%\n',(acc/test_num)*100);
```

```
%knn.m
clc
clear variables

load ('../test_images.mat');
load ('../test_labels.mat');
load ('../train_images.mat');
load ('../train_labels.mat');
% 选取数据容量
train_num = 2000;
test_num = 200;
data_train = mat2vector(train_images(:,:,1:train_num),train_num);% 转换为行向量
data_test = mat2vector(test_images(:,:,1:test_num),test_num);
% 参数说明：knn 要求在一个范围内选取近邻样本，这个容量需要手动给出
% 其他参数如距离计算模式，距离权重均为默认，如欧式距离，等权值等。
% 具体结果可以在运行后查看 knn_model 来详细观察
knn_model = fitcknn(data_train,train_labels1(1:train_num),'NumNeighbors',10);
result = predict(knn_model,data_test);
acc = 0.;
for i = 1:test_num
    if result(i)==test_labels1(i)
        acc = acc+1;
    end
end
```

```
% mat2vector.m
% 输入：图片数据（矩阵），样本个数
% 函数作用：将图片组转化为行向量的组合，每个行向量作为一张图片的特征
% 输出：样本数*图片像素数量大小的矩阵
function [data_]= mat2vector(data,num)
[row,col,~] = size(data);
data_ = zeros(num,row*col);
for page = 1:num
for rows = 1:row
for cols = 1:col
data_(page,((rows-1)*col+cols)) = im2double(data(rows,cols,page));
end
end
end
end

% data_proc.m
% 函数功能：删除同类数据特征中方差为 0 的特征列
% 输入：行向量数据及标签
% 输出：删除列之后的数据以及删除的列标
function [output,position] = data_proc(data,label)
position = cell(1,10);
%创建 cell 存储每类中删除的列标
for i = 0:9
temp = [];
pos = [];
for rows = 1:size(data,1)
if label(rows)==i
temp = [temp;data(rows,:)];
end
end
for cols = 1:size(temp,2)
var_data = var(temp(:,cols));
if var_data==0
pos = [pos,cols];
end
end
position{i+1} = pos;
data(:,pos)=[];
end
output = data;
end
```

```
function labelsmat = gz2mat(fn)


fp = fopen('D:\研究生文件\研究生课程\威尔士\Artificial
Intelligence\t10k-labels.idx1-ubyte', 'rb');
assert(fp ~= -1, ['该文件无法打开', fn, '']);

ma = fread(fp,1,'uint8');
assert(ma == 2049, ['逻辑出错 ', fn, '']);

numLabels = fread(fp, 1, 'uint8', 0, 'ieee-be');

labels = fread(fp, inf, 'uint8');

assert(size(labels,1) == numLabels, '出错的 label 数量为');

fclose(fp);

save test_labels.mat labels

end
```

```
function imgsmat = gzimgs2mat(fn)

fp = fopen('D:\研究生文件\研究生课程\威尔士\Artificial
Intelligence\t10k-labels.idx1-ubyte', 'rb');
assert(fp ~= -1, ['该文件无法打开', fn, '']);

ma = fread(fp,1,'uint32');
assert(ma == 2051, ['逻辑出错 ', fn, '']);

imgsNum = fread(fp, 1, 'uint32', 0, 'ieee-be');
rows = fread(fp, 1, 'uint32', 0, 'ieee-be');
cols = fread(fp, 1, 'uint32', 0, 'ieee-be');
imgs = fread(fp, inf, 'uint8');
imgs = reshape(imgs, rows, cols, imgsNum);
imgs = permute(imgs,[2 1 3]);

imgs = reshape(imgs, size(imgs, 1) * size(imgs, 2), size(imgs, 3));
imgs = double(imgs) / 255;
fclose(fp);

save train_images.mat imgs

end
```