**人工智能原理与技术实验报告**

**基于MNIST的手写数字识别**

**学院：**

**班级：**

**学号：**

**姓名：**

**日期：**

目录

[一、 问题描述 3](#_Toc440486762)

[二、 实验环境 3](#_Toc440486763)

[三、 算法综述 4](#_Toc440486764)

[1. KNN算法 4](#_Toc440486765)

[2. VSM算法 4](#_Toc440486766)

[3. 模式匹配 4](#_Toc440486767)

[四、 实验原理 6](#_Toc440486768)

[1. Bp神经网络算法 6](#_Toc440486769)

[2. 支持向量机 8](#_Toc440486770)

[3. 决策树 10](#_Toc440486771)

[五、 实验过程与结果分析 13](#_Toc440486772)

[1. BP神经网络 13](#_Toc440486773)

[2. 支持向量机 15](#_Toc440486774)

[3. 决策树 16](#_Toc440486775)

[六、 总结讨论 17](#_Toc440486776)

[七、 参考文献 17](#_Toc440486777)

## 问题描述

手写数字识别(Handwritten Numeral Recognition)是光学字符识别技术

(Optical Character Recognition,简称OCR)的一个分支，它研究的对象是：如何利用电子计算机自动辨认人手写在纸张上的阿拉伯数字。 在整个OCR领域中，最为困难的就是脱机手写字符的识别。到目前为止，尽管人们在脱机手写英文、汉字识别的研究中已取得很多可喜成就，但距实用还有一定距离。

数字识别的应用需求相当紧迫，需要计算机自动处理的数字量非常大，而目前实际利用自动识别的文字量所占的比例非常小。手写体数字识别技术在文件资料自动录入、金融税务系统数据统计、图像文本的压缩存储、自动阅读器、盲人助读器等领域都有广泛的应用。

本实验数据使用MNIST手写体数据集，该数据集包含的共 70000 图像 ； 60000 训练图像 （用于创建红外模型） 和 10000 测试图像 （用于评估模型的精度）。 每个 MNIST 图像是一个单一的手写的数字字符的数字化的图片。 每个图像是 28 x 28 像素大小。 每个像素值是 0，表示白色，至 255，表示黑。 中间像素值表示的灰度级。

## 实验环境

一台win8系统的台式机，Matlab R2015，libsvm\_3.21工具包，MNIST数据集。

## 算法综述

### KNN算法

K最近邻(k-Nearest Neighbor，KNN)分类算法，是一个理论上比较成熟的方法，也是最简单的机器学习算法之一。该方法的思路是：如果一个样本在特征空间中的k个最相似(即特征空间中最邻近)的样本中的大多数属于某一个类别，则该样本也属于这个类别。KNN算法中，所选择的邻居都是已经正确分类的对象。该方法在定类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。 KNN方法虽然从原理上也依赖于极限定理，但在类别决策时，只与极少量的相邻样本有关。由于KNN方法主要靠周围有限的邻近的样本，而不是靠判别类域的方法来确定所属类别的，因此对于类域的交叉或重叠较多的待分样本集来说，KNN方法较其他方法更为适合。

该算法在分类时有个主要的不足是，当样本不平衡时，如一个类的样本容量很大，而其他类样本容量很小时，有可能导致当输入一个新样本时，该样本的K个邻居中大容量类的样本占多数。计算量大，可理解性差，无法给出像决策树那样的规则。

### VSM算法

VSM法即向量空间模型(Vector Space Model)法，其基本思想是将文档表示为加权的特征向量：D=D(T1，W1；T2，W2；…；Tn，Wn)，然后通过计算文本相似度的方法来确定待分样本的类别。当文本被表示为空间向量模型的时候，文本的相似度就可以借助特征向量之间的内积来表示。

在实际应用中，VSM法一般事先依据语料库中的训练样本和分类体系建立类别向量空间。当需要对一篇待分样本进行分类的时候，只需要计算待分样本和每一个类别向量的相似度即内积，然后选取相似度最大的类别作为该待分样本所对应的类别。

由于VSM法中需要事先计算类别的空间向量，而该空间向量的建立又很大程度的依赖于该类别向量中所包含的特征项。根据研究发现，类别中所包含的非零特征项越多，其包含的每个特征项对于类别的表达能力越弱。因此，VSM法相对其他分类方法而言，更适合于专业文献的分类。

### 模式匹配

模式识别又常称作模式分类，常用的方法有：决策理论方法和句法方法。

1. 决策理论方法

又称统计方法，是发展较早也比较成熟的一种方法。被识别对象首先数字化，变换为适于计算机处理的数字信息。一个模式常常要用很大的信息量来表示。许多模式识别系统在数字化环节之后还进行预处理，用于除去混入的干扰信息并减少某些变形和失真。随后是进行特征抽取，即从数字化后或预处理后的输入模式中抽取一组特征。特征抽取过程将输入模式从对象空间映射到特征空间。这时，模式可用特征空间中的一个点或一个特征矢量表示。这种映射不仅压缩了信息量，而且易于分类。在决策理论方法中，特征抽取占有重要的地位，但尚无通用的理论指导，只能通过分析具体识别对象决定选取何种特征。特征抽取后可进行分类，即从特征空间再映射到决策空间。为此而引入鉴别函数，由特征矢量计算出相应于各类别的鉴别函数值，通过鉴别函数值的比较实行分类。

1. 句法方法

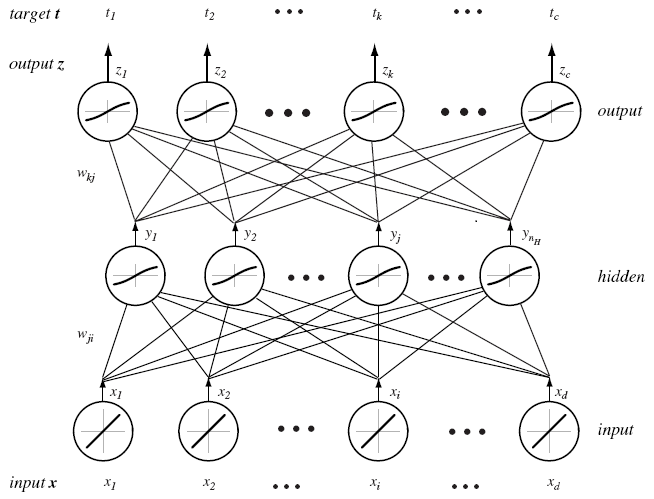
又称结构方法或语言学方法。其基本思想是把一个模式描述为较简单的子模式的组合，子模式又可描述为更简单的子模式的组合，最终得到一个树形的结构描述，在底层的最简单的子模式称为模式基元。在句法方法中选取基元的问题相当于在决策理论方法中选取特征的问题。通常要求所选的基元能对模式提供一个紧凑的反映其结构关系的描述，又要易于用非句法方法加以抽取。显然，基元本身不应该含有重要的结构信息。模式以一组基元和它们的组合关系来描述，称为模式描述语句，这相当于在语言中，句子和短语用词组合，词用字符组合一样。基元组合成模式的规则，由所谓语法来指定。一旦基元被鉴别，识别过程可通过句法分析进行，即分析给定的模式语句是否符合指定的语法，满足某类语法的即被分入该类。

这两种方法不能截然分开，在句法方法中，基元本身就是用决策理论方法抽取的。在应用中，将这两种方法结合起来分别施加于不同的层次，常能收到较好的效果。

## 实验原理

### Bp神经网络算法

BP（Back Propagation）是一种按误差逆传播算法训练的多层前馈网络。BP网络能学习和存贮大量的输入-输出模式映射关系，而无需事前揭示描述这种映射关系的数学方程。它的学习规则是使用最速下降法，通过反向传播来不断调整网络的权值和阈值，使网络的误差平方和最小。BP神经网络模型拓扑结构包括输入层（input）、隐层(hide layer)和输出层(output layer)。



输入层：单元i的输入： xi 单元数量：d

单元i的输出： xi

单元i的激活函数：线性函数

隐 层：单元j的净输入：netj 单元数量： nH

单元j的输出：单元j的激活函数：非线性函数输出层：单元k的净输入： 单元数量：c

单元k的输出：单元k的激活函数：非线性函数

1. 权值的调整

输入一个样本(x,t)，计算网络的输出z，根据z与t的差距调整所有的权系数，使z与t尽可能接近，即使近可能小。

权系数的调整：

1. 激活函数

激活函数f应满足如下基本条件: 非线性、有界、连续、可微，用的激活函数：Sigmoid函数

f(net) = 1/(1+e-net)

(1-e-bnet)/(1+e-bnet)

在训练过程开始前，权系数通常采用随机化方法进行随机赋值，且初始权值不宜过大。典型的方法是：对于每一个权系数，按照均匀分布在区间内随机选择一个数值进行赋值。对于每一个权系数，按照均匀分布在区间内随机选择一个数值进行赋值。

1. 学习速率

学习速率 直接影响权系数调整时的步长。学习速率过小，导致算法收敛速度缓慢。学习速率过大，导致算法不收敛。学习速率的典型取值 。另外学习速率可变。误差函数的局部极小值调整权系数的目标是使误差函数取得最小值。但是，采用梯度下降法(Gradient Descent Procedure)不能保证获得最小值，而只能保证得到一个极小值。如果训练过程无法使误差函数降低到预期的程度，一种常用的方法是：再一次对权系数进行随机初始化，并重新训练网络。

1. 学习曲线

样本集的划分：一般情况下，可把已知的样本集划分为三个子集：练集、确认集、测试集。本次实验只划分为两个子集，即训练集、测试集。

训练集：用来调整权系数，使误差函数尽可能变小。

确认集：用来初步验证神经网络对未来新样本的分类能力，并据此确定最佳的权系数。神经网络的训练过程需要采用训练集及确认集共同完成。

测试集：在训练过程最终结束后，再采用测试集对网络的分类性能进行最后测试，用以评估神经网络在实际应用中的实际性能。

### 支持向量机

支持向量机(SVM)方法是通过一个非线性映射p，把样本空间映射到一个高维乃至无穷维的特征空间中(Hilbert空间)，使得在原来的样本空间中非线性可分的问题转化为在特征空间中的线性可分的问题.简单地说，就是升维和线性化.升维，就是把样本向高维空间做映射，一般情况下这会增加计算的复杂性，甚至会引起"维数灾难"，因而人们很少问津.但是作为分类、回归等问题来说，很可能在低维样本空间无法线性处理的样本集，在高维特征空间中却可以通过一个线性超平面实现线性划分(或回归).一般的升维都会带来计算的复杂化，SVM方法巧妙地解决了这个难题:应用核函数的展开定理，就不需要知道非线性映射的显式表达式;由于是在高维特征空间中建立线性学习机，所以与线性模型相比，不但几乎不增加计算的复杂性，而且在某种程度上避免了"维数灾难".这一切要归功于核函数的展开和计算理论。选择不同的核函数，可以生成不同的SVM，常用的核函数有以下4种：

1) 线性核函数K(x,y)=x\*y；

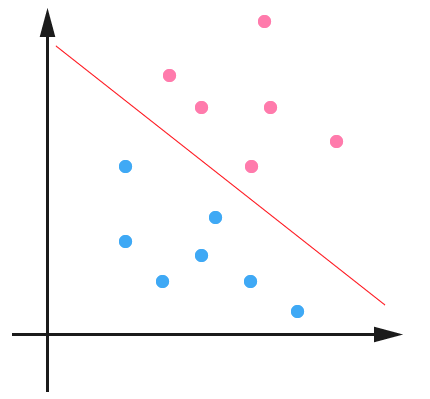
2) 多项式核函数K(x,y)=[(x\*y)+1]^d；

3) 径向基函数K(x,y)=exp(-|x-y|^2/d^2)；

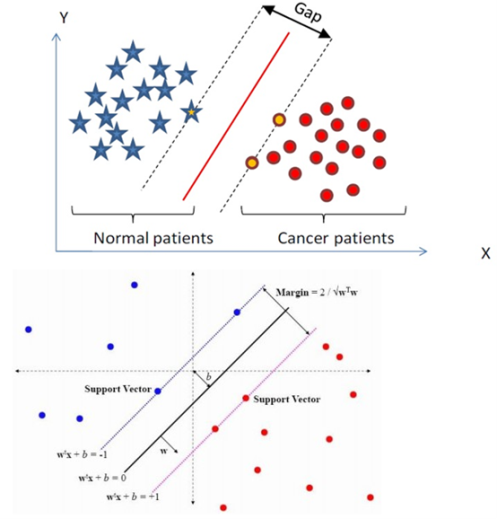
4) 二层神经网络核函数K(x,y)=tanh(a(x\*y)+b)。

支持向量机分类情况如下：

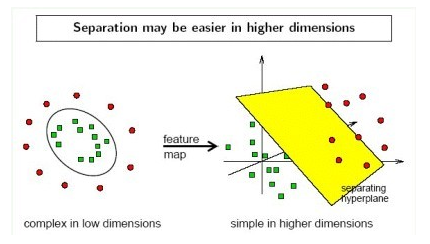
1. 在n维空间中找到一个分类超平面，将空间上的点分类。如下图所示线性分类的例子。



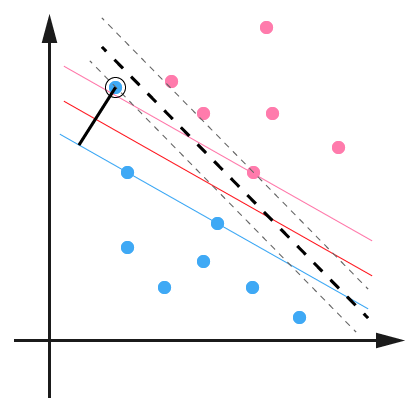
1. 一般而言，一个点距离超平面的远近可以表示为分类预测的确信或准确程度。SVM就是要最大化这个间隔值。而在虚线上的点便叫做支持向量Supprot Verctor。



1. 我们会经常遇到线性不可分的样例，此时，我们的常用做法是把样例特征映射到高维空间中去，如下图所示。线性不可分映射到高维空间，可能会导致维度大小高到可怕的(19维乃至无穷维的例子)，导致计算复杂。核函数的价值在于它虽然也是讲特征进行从低维到高维的转换，但核函数绝就绝在它事先在低维上进行计算，而将实质上的分类效果表现在了高维上，也就如上文所说的避免了直接在高维空间中的复杂计算。



1. 使用松弛变量处理数据噪音。



本次实验使用LIBSVM工具包，LIBSVM是台湾大学林智仁(Lin Chih-Jen)教授等开发设计的一个简单、易于使用和快速有效的SVM模式识别与回归的软件包，他不但提供了编译好的可在Windows系列系统的执行文件，还提供了源代码，方便改进、修改以及在其它操作系统上应用;该软件对SVM所涉及的参数调节相对比较少，提供了很多的默认参数，利用这些默认参数可以解决很多问题;并提供了交互检验(Cross Validation)的功能。该软件可以解决C-SVM、ν-SVM、ε-SVR和ν-SVR等问题，包括基于一对一算法的多类模式识别问题。该工具包可以解决多分类问题。

### 决策树

决策树也称为判定树，是一种有监督的学习方法。决策树代表着决策树的树形结构，可以根据训练集数据构造出决策树。如果该树不能对所有对象给出正确的分类，就选择一些例外加入到训练集数据中。重复该过程，直到形成正确的决策集。决策树方法首先对数据进行处理，利用归纳算法生成可读的规则和决策树，然后使用决策树对新数据进行分析，本质上是通过一系列规则对数据进行分类的过程。决策树的典型算法有ID3，C4.5，CART等，本次采用ID3测试集的方法，将样本集合分为训练集与测试集。根据训练集构建决策树，决策树中的结点逐层展开。每展开一层子结点，并将其设为叶结点，就得到一棵决策树，然后采用测试集对所得决策树的分类性能进行统计。重复上述过程，可以得到决策树在测试集上的学习曲线。根据学习曲线，选择在测试集上性能最佳的决策树为最终的决策树。与决策树相关得到主要算法有：ID3，CLS，C4.5，CART。

1. **CLS算法**

CLS算法是早期的决策树学习算法。它是许多决策树学习算法的基础。

1. CLS基本思想

从一棵空决策树开始，选择某一属性（分类属性）作为测试属性。该测试属性对应决策树中的决策结点。根据该属性的值的不同，可将训练样本分成相应的子集，如果该子集为空，或该子集中的样本属于同一个类，则该子集为叶结点，否则该子集对应于决策树的内部结点，即测试结点，需要选择一个新的分类属性对该子集进行划分，直到所有的子集都为空或者属于同一类。

1. 决策树构建
2. 生成一颗空决策树和一张训练样本属性集;
3. 若训练样本集T 中所有的样本都属于同一类,

则生成结点T , 并终止学习算法;否则

1. 根据某种策略从训练样本属性表中选择属性

A 作为测试属性, 生成测试结点A

1. 若A的取值为v1,v2,…,vm, 则根据A 的取值的

不同,将T 划分成 m个子集T1,T2,…,Tm;

1. 从训练样本属性表中删除属性A;
2. 转步骤2, 对每个子集递归调用CLS;

在步骤3中，根据某种策略从训练样本属性表中选择属性A作为测试属性。没有规定采用何种测试属性。实践表明，测试属性集的组成以及测试属性的先后对决策树的学习具有举足轻重的影响。

1. **ID3算法**

ID3算法主要针对属性选择问题。是决策树学习方法中最具影响和最为典型的算法。该方法使用信息增益度选择测试属性。当获取信息时，将不确定的内容转为确定的内容，因此信息伴着不确定性。从直觉上讲，小概率事件比大概率事件包含的信息量大。如果某件事情是“百年一见”则肯定比“习以为常”的事件包含的信息量大。

1. 信息量大小的度量

Shannon1948年提出的信息论理论。事件ai的信息量I（ ai ）可如下度量：



其中p(ai)表示事件ai发生的概率。

假设有n个互不相容的事件a1,a2,a3,….,an,它们中有且仅有一个发生，则其平均的信息量可如下度量：

 公式1

上式，对数底数可以为任何数，不同的取值对应了熵的不同单位。通常取2，并规定当p(ai)=0时=0

在决策树分类中，假设S是训练样本集合，|S|是训练样本数，样本划分为n个不同的类C1,C2,….Cn，这些类的大小分别标记为|C1|，|C2|，…..,|Cn|。则任意样本S属于类Ci的概率为：



Entropy（S，A）=∑（|Sv|/|S|）\* Entropy（Sv） 公式2

∑是属性A的所有可能的值v,Sv是属性A有v值的S子集|Sv|是Sv 中元素的个数；|S|是S中元素的个数。

Gain（S，A）是属性A在集合S上的信息增益

Gain（S，A）= Entropy（S） -Entropy（S，A） 公式3

Gain（S，A）越大，说明选择测试属性对分类提供的信息越多

1. 决策树建立
2. 决定分类属性；
3. 对目前的数据表，建立一个节点N
4. 如果数据库中的数据都属于同一个类，N就是树叶，在树叶上标出所属的类
5. 如果数据表中没有其他属性可以考虑，则N也是树叶，按照少 数服从多数的原则在树叶上标出所属类别
6. 否则，根据平均信息期望值E或GAIN值选出一个最佳属性作为节点N的测试属性
7. 节点属性选定后，对于该属性中的每个值：

从N生成一个分支，并将数据表中与该分支有关的数据收集形

成分支节点的数据表，在表中删除节点属性那一栏

如果分支数据表非空，则运用以上算法从该节点建立子树。

## 实验过程与结果分析

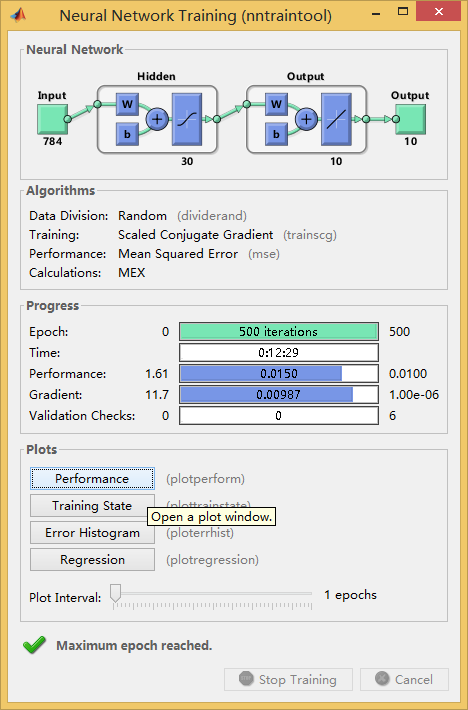
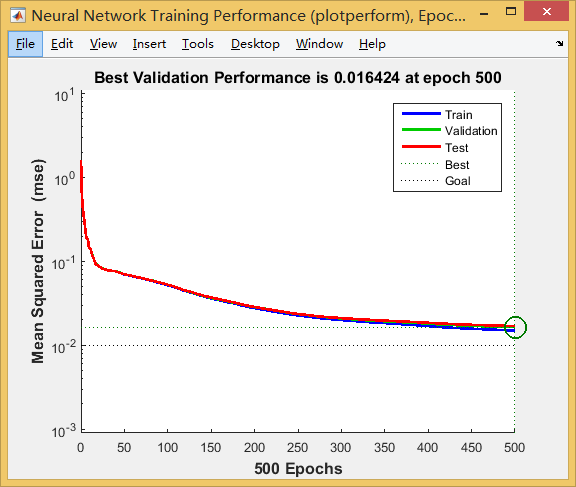
对于MNIST数据集中原本的格式不能直接读取数据，则通过自己定义的功能函数loadMNISTImages() 可以直接将数据集转化成为double型的矩阵，这样在代码里面直接使用loadMNISTImages(‘数据文件名’)函数就可以直接读取数据了；另一种方法是直接通过自定义功能函数readMnist\_train\_images()，readMnist\_train\_labels()，readMnist\_test\_images()，readMnist\_test\_labels()，将MNIST数据读取出来保存成(.mat)文件，通过load()函数调用。

### BP神经网络

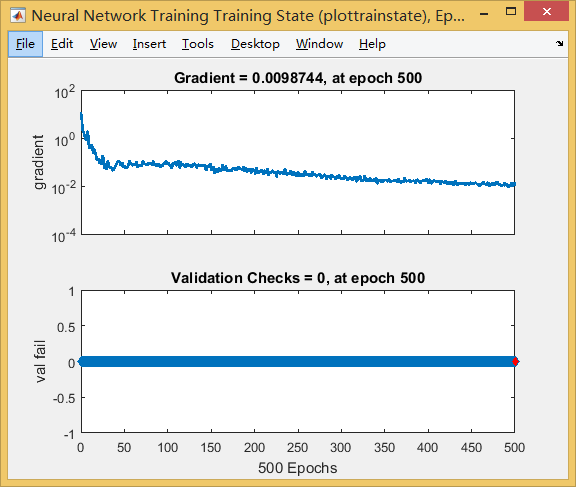
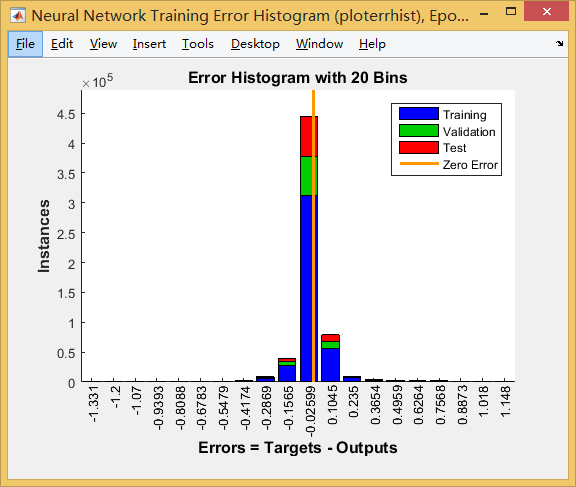
实验结果：

采用一千训练数据与一千的测试数据，设置隐藏层数30，变换函数为trainscg，训练次数500，训练目标最小误差为0.01学习速率0.1，得到准确率为93.3% 。

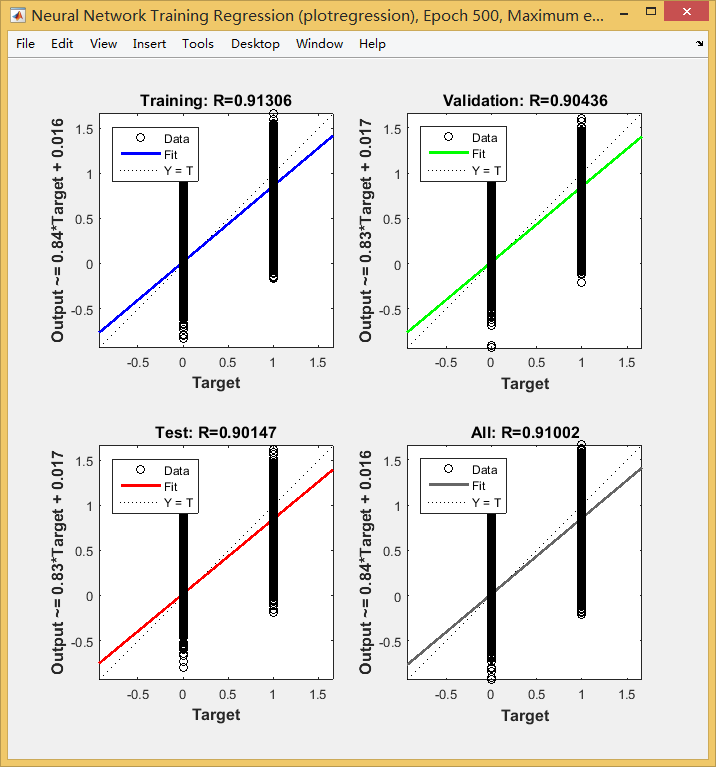
采用全部训练数据与全部的测试数据，设置隐藏层数30，变换函数为trainscg，训练次数500，训练目标最小误差为0.01学习速率0.1，得到准确率为99.25%。

1. network（左）performance（右）

1. Train state（左）error histogram（右）



1. Regression

实验代码：

%% 读入数据

load('mnist\_train\_images.mat');

load('mnist\_train\_labels.mat');

load('mnist\_test\_images.mat');

load('mnist\_test\_labels.mat');

%% 创建网络

net=feedforwardnet(30);

%% 训练参数

net.trainFcn='trainscg';

net.trainParam.goal=0.01; % 训练目标最小误差，这里设置为0.1

net.trainParam.epochs=500; % 训练次数，这里设置为500次

net.trainParam.lr=0.1; % 学习速率，这里设置为0.1net.trainParam.min\_grad=1e-6; % 最小性能梯度

%% 训练网络

net=train(net,trainImages',trainLabels');

%% 测试

test\_out=sim(net,testImages');

test\_out =test\_out' ;

test\_out(test\_out>=0.5)=1;

test\_out(test\_out<0.5)=0;

rate=sum(test\_out==mnist\_test\_labels)/length(mnist\_test\_labels);

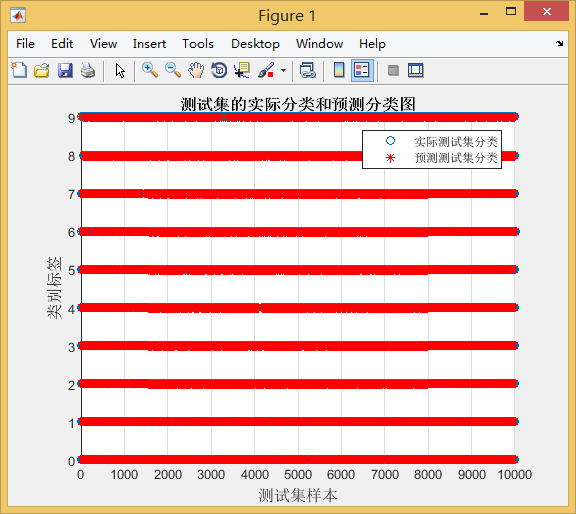
fprintf(' 正确率\n %f %%\n', rate\*100);

### 支持向量机

实验结果

使用二千个训练数据与二千个测试数据所得的准确率为85.5%

使用六万个训练数据与一万测试数据测试出的准确率为96.6%；



1. 六万训练与一万测试

实验代码（主要代码代码）

%% 读入数据 如神经网络

%% SVM网络训练

model = svmtrain(mnist\_train\_labels,mnist\_train\_images);

%% SVM网络预测

[predict\_label,accuracy,decision\_value]=svmpredict(mnist\_test\_labels, mnist\_test\_images, model);

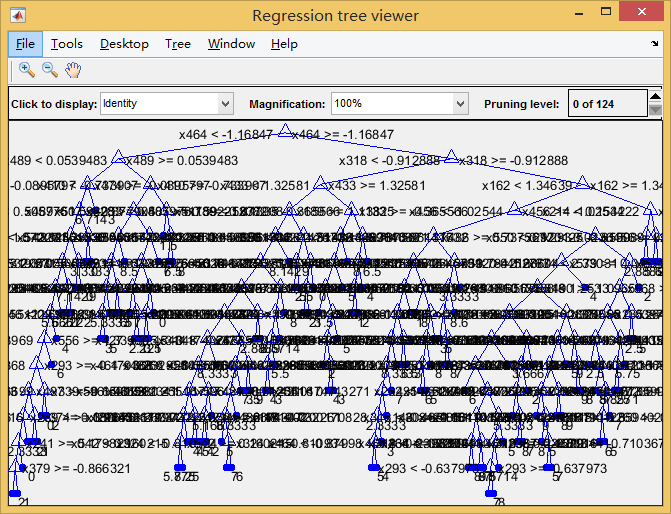
### 决策树

实验结果

采用了一百个样本点进行测试，测试结果显示决策树的COST在6.4和7.8之间，而对应的SECOST在0.72和1.52,分类准确率为91.44%。

采用了一千个样本点进行测试，测试结果显示决策树的COST在6.4和8.1之间，而对应的SECOST在0.22和0.35之间,分类准确率为94.63%；

采用了一万个样本点进行测试，测试结果显示决策树的COST在2.7和8.4之间，其中绝大部分在2.7 和5之间，而对应的SECOST在0.072和0.088之间,分类准确率为97.1%



1. 10000训练数据决策树

实验代码（主要代码）

%% 读入数据 如神经网络

[data,MU,SIGMA] = zscore(data');

t = classregtree(data,label);

view(t);

[testdata,MU,SIGMA] = zscore(testdata');

[COST,SECOST,NTNODES,BESTLEVEL]=

test(t,'test',testdata,testlabel);

## 总结讨论

对于BP神经网络算法，用不同方法时收敛速度差别较大，本实验采用收敛速度较快的梯度下降法和量化共轭梯度法（trainscg），准确率也不同；同时在相同方法时，学习速率较小（从0.1到0.05），收敛也会较慢；相同的训练样本数据时，测试数据增加，准确率也会上升。

SVM分类时使用了libsvm的软件包，简单、易于使用，SVM快速有效的模式识别与回归。使用svmtrainh函数训练支持向量机时，参数设置为：SVM类型为C-SVC，核函数类型为BEF函数等支持向量机默认值。同样，在SVM分类中使用的样本数据越多，分类也越准确。

决策树的创建使用fitctree函数，因此在之前还需要将数据类型从double转变为string类型。同样的，在决策树构建时选择不同的最小叶子节点数产生的分类的准确性也是不同的，但是样本的数据越多时，分类就会越准确。

使用神经网络，支持向量机，决策树对MNIST手写数字数据集进行识别。

整个实验过程中，神经网络的分类准确率略高于支持向量机和决策树，决策树与支持向量机的分类准确差不多。但在实际的应用中，对于不同类型的问题，每种算法各有优点，实际操作时应结合实际问题综合考虑，而不是单一的使用哪种方式。

通过本次的实验，对于神经网络、支持向量机和决策树都有了更加深刻的理解。在使用matlab中逐渐的习惯了通过查看帮助文档来确定使用的函数，以及出现问题时解决程序中出现的问题，在通过前期的查阅资料的过程中，找论文、看论文，也提高了对论文的理解能力。

## 参考文献