### 二叉树根节点到指定节点的路径

题目描述：给定一棵二叉树和二叉树中一个节点，输出根节点到指定节点间的路径。

  　　　　10     
  　　　　/ \     
 　　　　5  12     
  　　　 / \      
  　　  4    7

指定节点7，那么输出路径应该是10-5-7。

分析与解法：

这个题目是在我做过蛮多二叉树的题目之后总结的一道题目。发现很多题目都可以抽象出来这个题目。其实二叉树本来就是最典型的递归数据结构，只要完全掌握三种遍历方法，一切题目都是浮云。

这道题目和博客[http://blog.csdn.net/getnextwindow/article/details/23326843](http://blog.csdn.net/getnextwindow/article/details/23326843" \t "http://m.blog.csdn.net/article/_blank)输出所有满足条件的路径颇为相似。只不过这道题目只要找到目的路径就要尽快返回，不要再遍历下去。不多解释，我会在代码中给予注释。

代码如下：

bool specialPath(Node \*pRoot,Node \*pNode,vector<int> &v)

{

if(pRoot==NULL)

{

return false;

}

v.push\_back(pRoot->m\_value);

bool found=false;

if(pRoot==pNode)//还是比较指针稳妥，节点值有可能重复

{

for(int i=0;i<v.size();i++)

cout<<v[i]<<" ";

cout<<endl;

return true;

}

if(!found && pRoot->m\_pLeft)

{

found=specialPath(pRoot->m\_pLeft,pNode,v);

}

//一旦左子树中找到节点，就不需要再遍历右子树

if(!found && pRoot->m\_pRight)

{

found=specialPath(pRoot->m\_pRight,pNode,v);

}

if(!found)

v.pop\_back();

return found;

}

注：if(!found) v.pop\_back()，只有在左右子树都没发现目标节点才弹出。如果我们仅仅在上面的函数中输出路径，那么没有这条判定语句也是对的，但是如果想使用v保存路径并在函数之外使用，必须加上这条语句，因为没有判定语句，函数回溯过程会弹出v中节点。希望关注和[http://blog.csdn.net/getnextwindow/article/details/23326843](http://blog.csdn.net/getnextwindow/article/details/23326843" \t "http://m.blog.csdn.net/article/_blank)的区别，上面链接的这篇要求所有满足条件的路径，所以只要知道满足的就打印，然后回退过程要弹出v节点,和本题刚好相反。

Libsvm中最重要的两个参数为C和gamma。C是惩罚系数，即对误差的宽容度。c越高，说明越不能容忍出现误差。C过大或过小，泛化能力变差。gamma是选择RBF函数作为kernel后，该函数自带的一个参数。隐含地决定了数据映射到新的特征空间后的分布，gamma越大，支持向量越少，gamma值越小，支持向量越多。支持向量的个数影响训练与预测的速度。这个问题[Chih-Jen Lin](http://www.csie.ntu.edu.tw/~cjlin/index.html)在其[主页](http://www.csie.ntu.edu.tw/~cjlin/libsvm/)上有详细的介绍。

而Liblinear的C参数也是非常重要的。

因此在系统中会通过5-flods交叉验证的方法对一定范围内的C,gamma进行grid 搜索