Weka[18] 寻找 K 个邻居

作者：Koala++/屈伟

寻找 K 个邻居，这里先给出一个例子，大部分是王义写的，进行了一点点改动：

```
public static Instances getKNeighbour( Instances sourceIns, Instance
target, int kNN, int numLabels)
    {
        Instances neighbours = null;

        try{
            EuclideanDistance dfunc = new EuclideanDistance();
            int predictors = sourceIns.numAttributes() - numLabels;
            dfunc.setAttributeIndices("first-" + predictors);

            LinearNNSearch lnn = new LinearNNSearch();
            lnn.setDistanceFunction(dfunc);
            lnn.setInstances( sourceIns );
            lnn.addInstanceInfo(target);
            neighbours = lnn.kNearestNeighbours(target, kNN);
        }catch(Exception e){
            System.out.println("Util.getKNeighbour(LinearNNSearch
              m_NNSearch,
              Instance mean, int m_kNN) is wrong!");
        }

        return neighbours;
    }
```

参数分别是 sourceIns 表示你 在哪个数据集中找邻居，target 就是找邻居的样本，kNN 是指多少邻居。最 后一个参数是我自己在多标签中用的。如果你不需要对特征进行选择，只是想除去类别，那么用下面的代码就可以了：

```
    sourceIns.setClassIndex( sourceIns.numAttributes() -1 );
    lnn.setInstances( sourceIns);
```

EuclideanDistance 表示是我用的是欧几里德距离，下面的 predictors 表示最后一个 index，也就是说算距 离我只算 0－predictors 这么 多特征，如果你不明白为什么要这样，因为你不懂多标签。下面是设置你考虑哪些特征，这里的”first-“+predictors 就表示只考虑 0-predictors。LinearNNSearcher 是寻找邻居的类，用 setDistanceFunction 进行距离计算设置，再对数据源和样本信息设置用 setInstances 和 addInstanceInfo，最后设置目标样本和找多少邻居。

这里的 setAttributeIndices 对我很重要，所以我看了一下：

```
    public void setAttributeIndices(String value) {
        m_AttributeIndices.setRanges(value);
       invalidate();
```

```
}
```

m_AttributeIndices 是一个 Range 对象。下面是一个 Range 的例子：

```
public static Instances getInstances( String filename )
{
    try{
        FileReader frData = new FileReader( filename );
        Instances trainData = new Instances(frData);

        return trainData;
    }catch( FileNotFoundException e )
    {
        e.printStackTrace();
    }
    catch( IOException e )
    {
        e.printStackTrace();
    }

    return null;
}

public static void outputRange( Instances ins )
{
    Range range = new Range();
    range.setRanges("first-2, 4, 10, last");
    range.setUpper(ins.numAttributes() - 1);
     boolean[] activeIndices = new boolean[ins.numAttributes()];
     for (int i = 0; i < activeIndices.length; i++)
     {
        activeIndices[i] = range.isInRange(i);
        System.out.print( activeIndices[i] + " " );
     }
}

public static void main(String[] args) throws Exception
{
    Instances ins = getInstances( "D:/DOWNLOAD/data/soybean.arff" );
    outputRange( ins );
}
```

设置 Range 我用的是"first-2,4,10,last"，first 表示第 0 个下标，last 表示最后一个下标,整个的意思就是选择从第 0 个到第 2 个，和第 4 个，第 10 个，最后一个特征。setUpper 表示最大下标是多少，注意 setUpper 函数：

```
public void setUpper(int newUpper) {
     if (newUpper >= 0) {
```

```
            m_Upper = newUpper;
                setFlags();
            }
    }
```

这里 setFlags 函数是得到哪 一些特征被选中，最后在我的例子中，我用 isRange 来判断是否第 i 个特征被选中了。

其它的函数特别简单，也没什么好讲的。