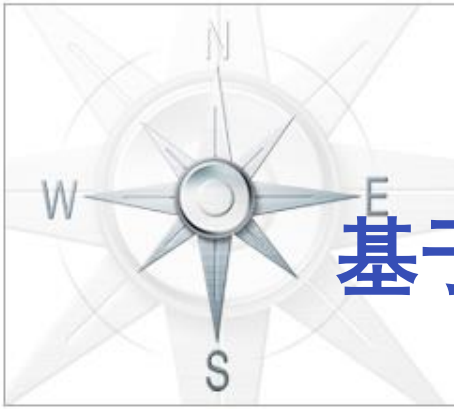


基于象限最近邻填充算法的研究

计算机软件与理论

3130295 张兆玉

导师 陈志泊
导师 王建新

目录

一、研究的内容及目标

- **研究内容**

我国的森林生态站的发展已经取得长足的发展，并积累了大量丰富的观测数据。发展至今，我们面临的问题是该如何利用这些宝贵的数据，来发现其潜在的价值。

由于观测仪器老化、断电、系统故障、网络等原因导致数据缺失是客观存在的。

因此，在对生态站数据进行挖据之前，数据预处理是一项重要的工作。同时，也是保证数据分析结果的准确性必不可少的环节。

一、研究的内容及目标

- **研究目标**

通过对生态站数据特征的分析，采用基于象限最近邻填充算法（**Weighted Quadrant Encapsidated Nearest Neighbor based Imputation 简称：WQENNI**）对生态站的数据进行补全。保证观测指标数据的正确性和准确性，也为聚类算法建立模型提供一个精准的数据基础。

目录

二、研究方法、技术路线、实验方案

- **数据样本的选取**

  - 选取不同年份的相同时间段的指标的离散观测值。比如缺失2014年7月1日12:00的观测值，则选取2004年~2014年7月1日观察的观测值作为相关数据。
  - 依据皮尔逊积矩相关系数相关性分析，选取n个具有相关性的观测指标，获取该n个指标2014年7月1日的所有观测数据，作为QENNI算法的输入数据。

二、研究方法、技术路线、实验方案

- 皮尔逊积矩相关系数

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}.$$

- 加权的基于象限近邻填充算法（WQENNI）

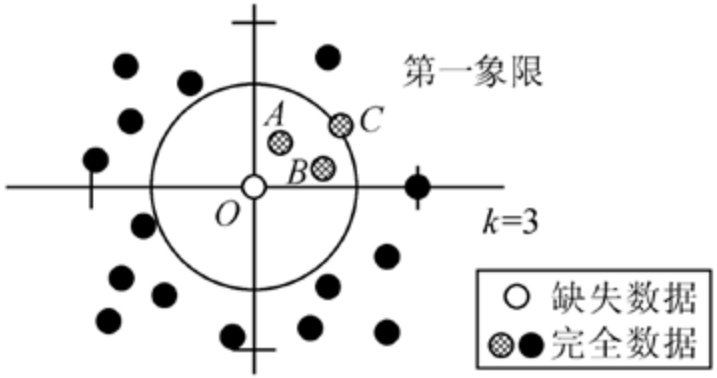


图 1 kNN 算法最近邻点的选择

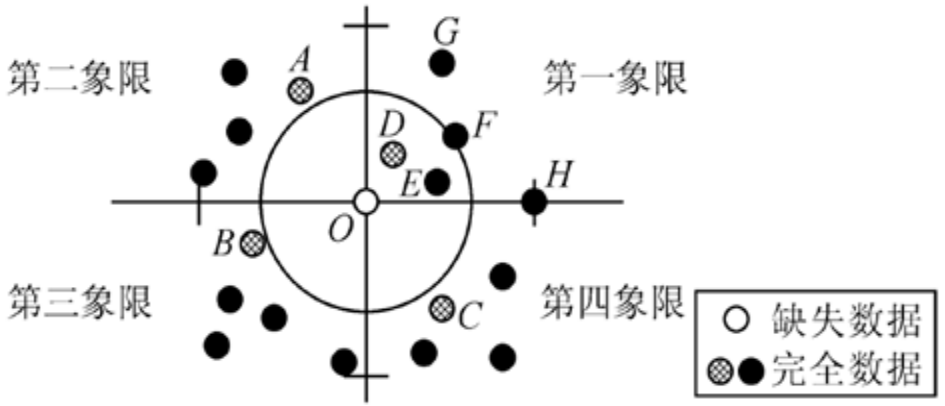


图 2 QENNI 算法最近邻点的选择

二、研究方法、技术路线、实验方案

- **kNN 算法在数据补全中的问题**

➢ kNN算法选取的k个最近邻点可能会有偏好，这使得填充效果相对低效。图1所示，O点代表缺失数据，其他点代表完全数据，我们用kNN来填充缺失数据，找到的k=3的三个临近点就是A、B、C，按照这种方式计算的缺失值偏向于第一象限。

二、研究方法、技术路线、实验方案

- kNN算法的最佳k选取比较困难，对每个数据集进行kNN实验前，必须反复计算得到最优的k。同时，k一旦偏差过大，将大大影响算法的填充性能。因此，如果算法能够消除对k的依赖就会成为最佳选择。
- 分析发现，用来估计缺失值的完全数据必须在近邻的情况下，尽可能地在缺失数据的第一个环形圆上，而且算法要最大限度地消除对参数k的依赖。

二、研究方法、技术路线、实验方案

- **加权的基于象限近邻填充算法（WQENNI）**

➢ 基于象限最近邻填充算法（Quadrant Encapsidated Nearest Neighbor based Imputation 简称：QENNI）。首先，把包含m个条件属性（X_1，X_2，…，X_m）的数看做m维空间上的一个点。并对该m维空间进行构造，以缺失数据为中心建立坐标系，通过坐标轴把m维空间划分成2^m象限。

二、研究方法、技术路线、实验方案

- 当m= 3时，数据的条件属性为（X_1，X_2，X_3），形成的空间坐标系把空间分成2^3个象限。每个数据也都位于唯一确定的象限中。
- 类似的一般情况。对于一般的m，数据的条件属性为（X_1，X_2，…，X_m）以缺失数据为中心形成的坐标系可把m 维空间分成2^m 个象限。同样，把处于象限间的临界数据归到它的某个邻近象限。显然，数据集中任一数据也都位于唯一确定的象限中。

二、研究方法、技术路线、实验方案

- **算法的改进研究**

➢ 选出的偏好点的决策数据与偏好点的距离将决定其对偏好点的影响大小。如果直接利用计算出的决策数据简单平均，将严重影响算法的准确性。明显的改进是根据计算得到的邻近点的贡献加权。将较大的权值赋给较近的近邻。

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\arg\max} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

其中：

$$w_i \equiv \frac{1}{d(x_q, x_i)^2}$$

二、研究方法、技术路线、实验方案

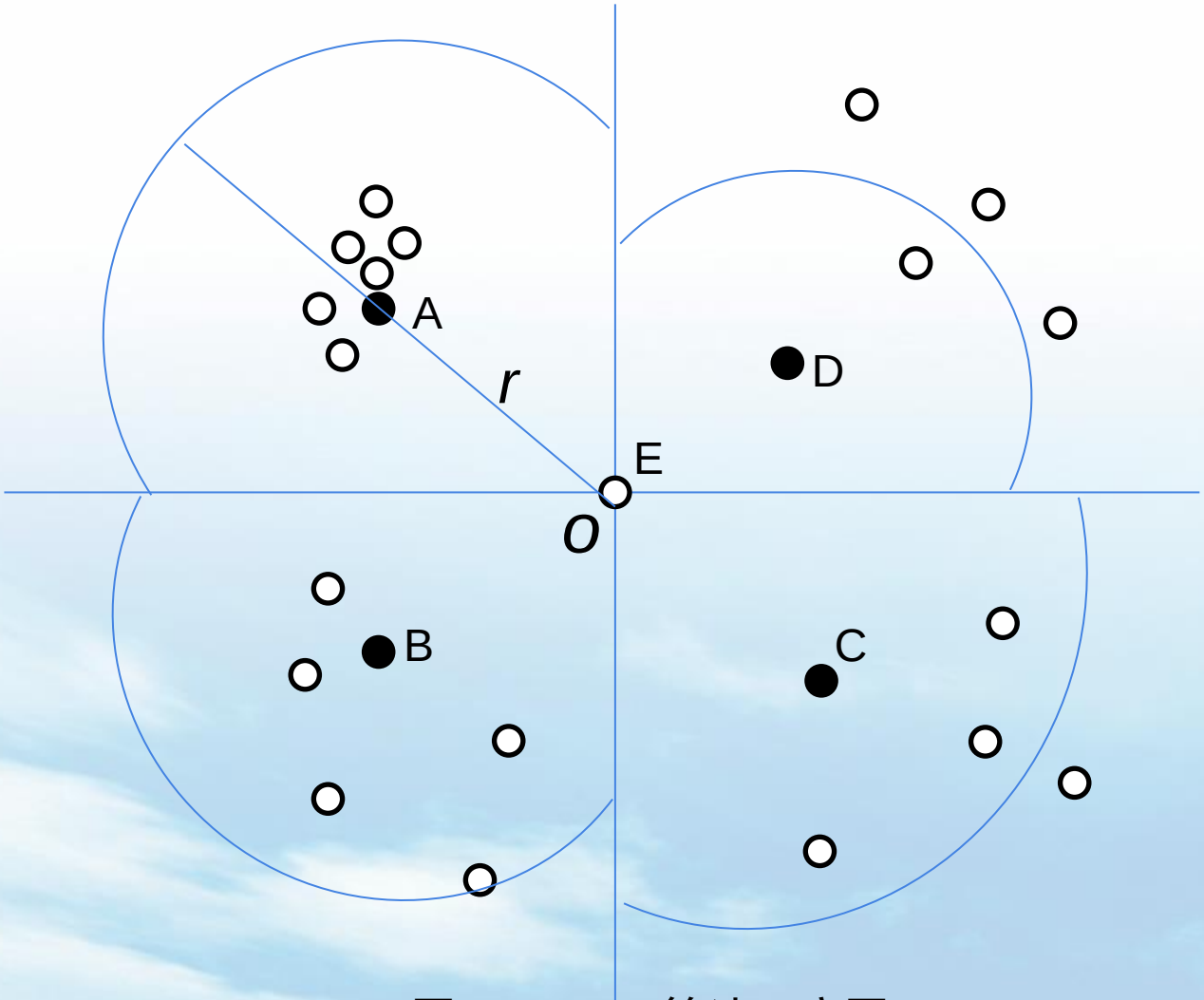


图3 WQENNI 算法示意图

二、研究方法、技术路线、实验方案

- 上述改进能解决偏重影响的一些问题。经过分析，对于数据的分布可能出现如下图所示，A点的数据密度较大，按权重计算A点的影响肯定小于E点，这与实际可能并不相符合。因此，我们除了根据距离加权还需要将数据的密度考虑进去。

二、研究方法、技术路线、实验方案

- 取各个象限选中的点与中心点距离的两倍为r画矩弧线，计算各个象限选中点数的和 $Ni(\mathrm{i}=1...\mathrm{n})$ 及总点数 $S=\sum_{i=1}^{n}Ni$，计算各个象限点数密度所占的权重 Ni/S，最终加上上一步距离加权的结果，即为该象限所占的比重。

$$q=(\sum_{i=1}^{n}((1-\beta)w_i+\beta N_i/S)\mathrm{d}_i)/\sum_{i=1}^{n}((1-\beta)w_i+\beta N_i/S)$$

目录

三、研究已取得的进展

- **研究已取得的进展**
  - 已完成算法模型的设计
  - 学习了Python 编程语言以及numpy 库
  - 完成kNN 算法的Python 程序编写（对比）
  - 完成QENNI 算法的Java 程序编写

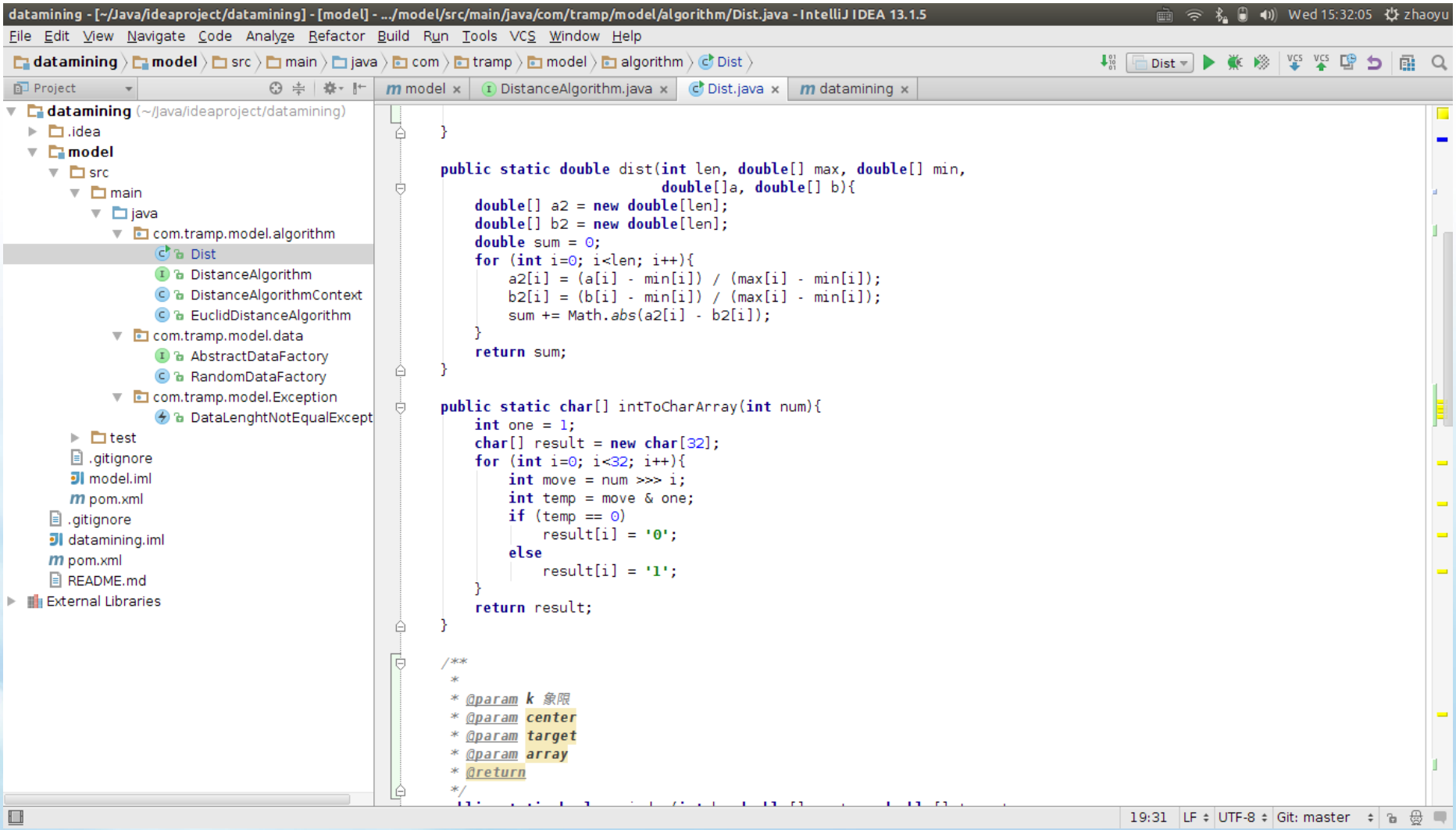
三、研究已取得的进展

zhaoyu@zhangzhaoyu: ~/machine_learning/algorithm — Wed 15:29:35 zhaoyu

```
from os import listdir


def createDataSet() :
    group = array([[1.0, 1.1],
                   [1.0, 1.0],
                   [0, 0],
                   [0, 0.1]
                ])
    labels = ['A', 'A', 'B', 'B']
    return group, labels


group, labels = createDataSet()

print group
print 'size of group : '
print group.shape

print labels


def classify0(inX, dataSet, labels, k) :
   dataSetSize = dataSet.shape[0]
   diffMat = tile(inX, (dataSetSize, 1)) - dataSet
   sqDiffMat = diffMat ** 2
   sqDistances = sqDiffMat.sum(axis=1)
   distance = sqDistances ** 0.5
   sortedDistIndicies = distance.argsort()
   classCount = {}
   print type(classCount)
   for i in range(k) :
       voteIlabel = labels[sortedDistIndicies[i]]
       classCount[voteIlabel] = classCount.get(voteIlabel, 0) + 1
   sortedClassCount = sorted(classCount.iteritems(), key=operator.itemgetter(1),
   reverse=True)
   return sortedClassCount[0][0]


print classify0([0, 0], group, labels, 3)

```

-UU-:----F1 kNN.py   Bot L27   (Python hs AC Abbrev)

三、研究已取得的进展


datamining - [~/Java/ideaproject/datamining] - [model] - .../model/src/main/java/com/tramp/model/algorithm/Dist.java - IntelliJ IDEA 13.1.5
Wed 15:32:05 zhaoyu
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
datamining model src main java com tramp model algorithm Dist
Project
datamining (~/Java/ideaproject/datamining)
.idea
model
src
main
java
com.tramp.model.algorithm
Dist
DistanceAlgorithm
DistanceAlgorithmContext
EuclidDistanceAlgorithm
com.tramp.model.data
AbstractDataFactory
RandomDataFactory
com.tramp.model.Exception
DataLenghtNotEqualExcept
test
.gitignore
model.iml
pom.xml
.gitignore
datamining.iml
pom.xml
README.md
External Libraries
model
DistanceAlgorithm.java
Dist.java
datamining
}
public static double dist(int len, double[] max, double[] min,
double[]a, double[] b){
double[] a2 = new double[len];
double[] b2 = new double[len];
double sum = 0;
for (int i=0; i<len; i++){
a2[i] = (a[i] - min[i]) / (max[i] - min[i]);
b2[i] = (b[i] - min[i]) / (max[i] - min[i]);
sum += Math.abs(a2[i] - b2[i]);
}
return sum;
}
public static char[] intToCharArray(int num){
int one = 1;
char[] result = new char[32];
for (int i=0; i<32; i++){
int move = num >>> i;
int temp = move & one;
if (temp == 0)
result[i] = '0';
else
result[i] = '1';
}
return result;
}
/**
*
* @param k 象限
* @param center
* @param target
* @param array
* @return
*/
19:31 LF UTF-8 Git: master

三、研究已取得的进展

- 下一步工作
  - 将Java 实现的算法改写为Python
  - 实现算法改进部分的程序
  - 跑数据与kNN算法对比
  - 完成小论文的撰写

Thank You !