

电子科技大学计算机科学与工程

学院

标准实验报告

(实验) 课程名称 数据挖掘与大数据分析

电子科技大学 实 验 报 告

学生姓名：朱若愚 学 号：2022150501027 指导教师：邵俊明

实验地点：主楼 A2-413 实验时间：2024.3.10

一、实验室名称：主楼 A2-413

二、实验项目名称：

（一）认识数据与数据预处理

（二）关联规则挖掘

（三）分类算法

三、实验学时：16

四、实验原理：

（一）

1、数据属性最小最大归一化（可尝试其它方法：如 ZSCORE）

$$v' = \frac{v - \min_A}{\max_A - \min_A} (\text{new_max}_A - \text{new_min}_A) + \text{new_min}_A$$

其中 v 是属性 A 的某个观测值， \min_A 和 \max_A 分别是属性 A 的最小值和最大值

上述公式将 A 属性的取值映射到区间 $[\text{new_min}_A, \text{new_max}_A]$

如果令 $\text{new_max}_A = 1, \text{new_min}_A = 0$ ，则将 A 属性映射到区间 $[0,1]$ ，

实现了数据归一化。

2、缺失值处理

对于数据中属性的缺失值，使用该属性的平均值来填补缺失值。

3、特征筛选

信息增益是用来进行特征筛选的常用算法，基本思想是选择那些特征对分类变量 Y 信息增益大，删除那些对分类无用的特征。

信息熵：
$$H(X) = - \sum_{i=1}^n p(x_i) \log p(x_i).$$

条件信息熵：
$$\begin{aligned} H(Y|X) &\equiv \sum_{x \in \mathcal{X}} p(x) H(Y|X = x) \\ &= - \sum_{x \in \mathcal{X}} p(x) \sum_{y \in \mathcal{Y}} p(y|x) \log p(y|x) \end{aligned}$$

信息增益： $IG(Y|X) = H(Y) - H(Y|X)$

(二)

1、频繁项集产生 (Frequent Itemset Generation)

其目标是发现满足最小支持度阈值的所有项集，这些项集称作频繁项集。

2、规则的产生 (Rule Generation)

其目标是从上一步发现的频繁项集中提取所有高置信度的规则，这些规则称作强规则 (strong rule)。

(三)

1、KNN 算法原理

KNN 属于 lazy learning，不会对训练样本数据进行学习，其做法是：

对于一个新数据，计算它与训练集中数据的距离，选择最短的 k 个作为邻居，然后预测它的类别和 k 个邻居中其所属类别最多的一致。算法伪代码如下：

- (1) 读入并保存训练数据
- (2) 预测时，计算已知类别数据集中的点与当前点之间的距离
- (3) 选取与当前距离最小的 k 个点
- (4) 计算前 k 个点所在类别的出现频率
- (5) 返回前 k 个点出现频率最高的类别作为当前点的预测分类

2、决策树算法原理

决策树是一种类似流程图的树结构，决策树中每个内部结点（非叶结点）表示在一个属性上的测试，每个分支代表该测试的一个输出，每个叶子结点代表类标签。给定类标号未知元组 X ，在决策树上测试 X 的属性值，跟踪一条由根到叶结点的路径，该叶结点就存放着该元组的类预测。

决策树的构造

核心思想在于决策树越向下生长，使节点的纯度越大。

自顶向下的分治方式构造决策树

递归的通过选择相应的测试属性，通过选择最有分类能力的属性作为决策树的当前结点，由此划分样本，其中测试属性是根据某种启发信息或者是统计信息来进行选择，如信息增益

决策树伪代码：

输入： 训练集 D

属性集 A

过程： 函数 createTree(D,A)

1: 生成结点 node

2: If D 中样本全属于同一类别 C then

3: 将 node 标记为 C 类叶结点;return

4: end if

5: If A=空集 OR D 中样本在 A 上取值相同 then

6: 将 node 标记为叶结点,其类标记为 D 中样本数最多的类;return

7: end if

8: 从 A 中选择最优划分属性 a

9: for a 的每一个值 a* do

10: 为 node 生成一个分支;令 D*表示 D 中在 a 上取值为 a*的样本子集;

11: if D*为空 then

12: 将分支结点标记为叶结点,其类标记为 D 中样本最多类

13: return;

14: else

15: 以 createTree(D*,A - {a})为分支结点

16: end if

17: end for

输出： 以 node 为根节点的一棵决策树

五、实验目的：

（一）

- 1、了解 Weka 工具包及 Eclipse 编程平台。
- 2、认识 and 了解数据。
- 3、对数据能进行简单的预处理。

（二）

- 1、掌握关联规则挖掘的基本概念、原理和一般方法
- 2、掌握 Apriori 算法
- 3、了解 FP-GROWTH 算法

（三）

- 1、了解分类的基本概念、原理和一般方法
- 2、掌握分类的基本算法
- 3、实现 KNN 或决策树算法

六、实验内容：

（一）

- 1、安装并配置 Java、Eclipse 和 Weka（已配置）。
- 2、使用图形界面的 Weka 工具包，完成数据归一化、缺失值处理、特征筛选的数据预处理操作。
- 3、在 Eclipse 下调用 Weka.jar 包，完成数据归一化、缺失值处理、特征筛选的数据预处理操作。

（二）

- 1、学会调用 WEKA 包实现关联规则的挖掘

2、自己编程实现 Apriori 算法

七、实验器材（设备、元器件）：计算机

八、实验步骤：

（一）

1、配置实验环境

2.1、使用 Weka 图形界面工具完成数据归一化

（1）打开 Weka GUI 工具

（2）选择 Explorer

（3）点击 Open file

（4）选择数据文件 iris.arff

（5）在 Filter 栏目下，选择 Choose

（6）选择 weka->filters->unsupervised->attribute->Normalize

（7）点击 “Normalize - S 1.0 - T 0.0” 所在区域可以调整参数，由于本例选择将数据归一化到[0,1]，使用默认的参数即可实现，点击 Apply 将数据归一化

（8）查看数据各个属性的值，已经归一化到[0,1]

2.2、使用 Weka 图形界面工具完成数据缺失值处理

数据使用 labor.arff，filter 使用 ReplaceMissingValues，参数默认

2.3、使用 Weka 图形界面工具完成特征筛选

数据使用 iris.arff，filter 使用 AttributeSelection，其中参数 evaluator 选择 InfoGainAttributeEval，search 使用 Ranker，需要调节 Ranker 的参数，选择最大特征数目（需小于原始的特征数目）

3.1、在 Eclipse 下调用 Weka.jar 包完成数据归一化

(1) 首先，新建一个 Java 工程

(2) 接下来，导入 weka.jar

(3) 创建一个 package 包

New -> package -> 输入 cn.uestc.preprocessing

(4) 创建 Java 文件

在包下面创建一个新的 class: TestNormalize.java

New -> Class -> 输入 TestNormalize

(5) 编辑 TestNormalize.java 文件，导入包

(6) 在 main 函数中完成数据归一化

(7) 观察数据是否已经归一化，打印实验结果

3.2、在 Eclipse 下调用 Weka.jar 包完成数据缺失值处理

3.3、在 Eclipse 下调用 Weka.jar 包完成特征筛选

(二)

1、寻找合适的数据集并进行适当处理

2、根据实验原理，编写 Apriori 算法。

(1) 读取数据

(2) 首先获取频繁一项集合（注意对于数据结构的选取提高计算效率）

(3) 利用 K 项频繁集生成 K+1 频繁候选集： 自连接+剪枝

(4) 重新扫描数据集，确定 K+1 频繁候选集中真正频繁的项集

重复上面两步

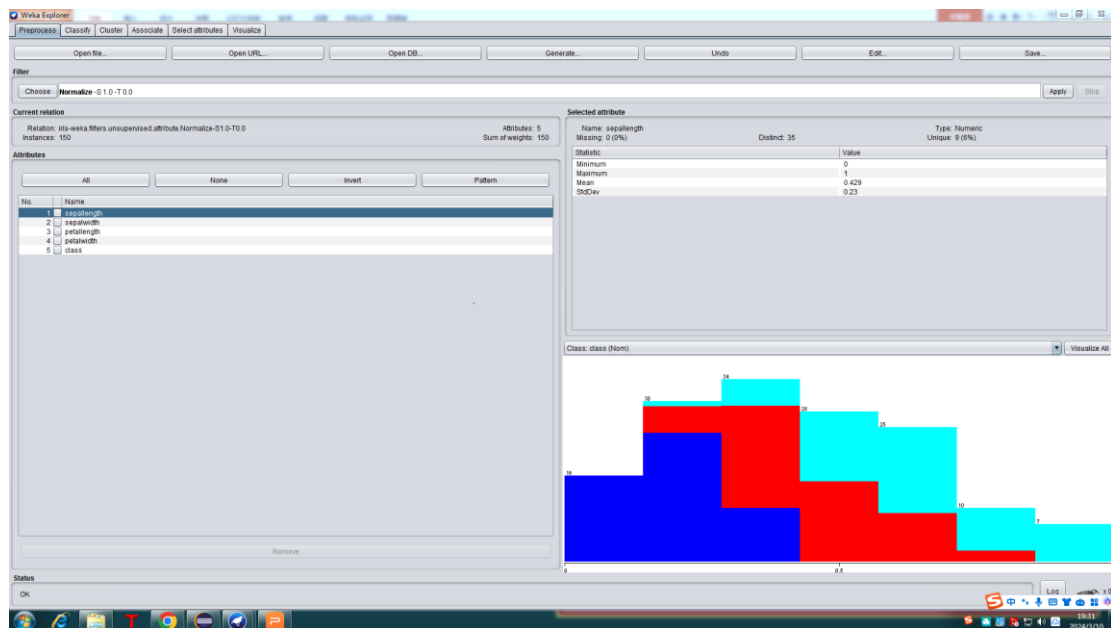
(三)

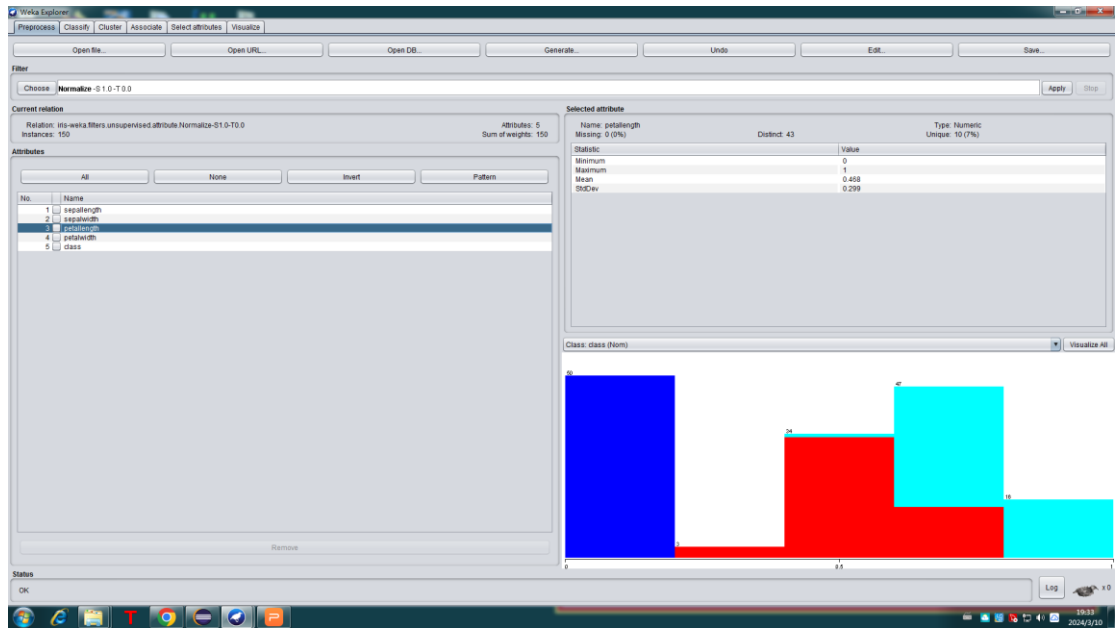
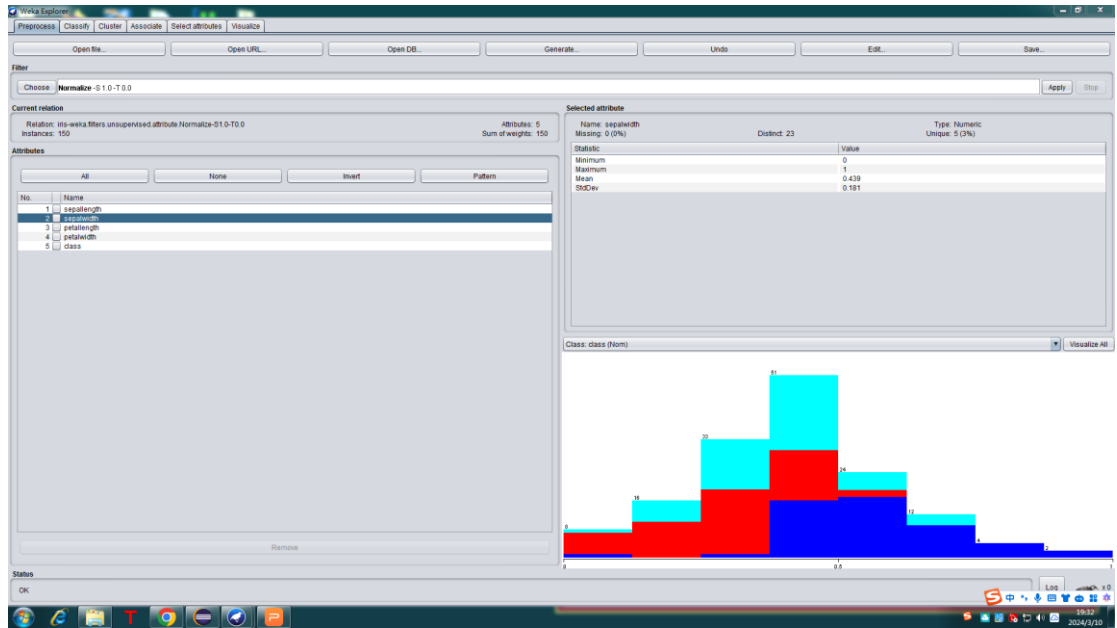
- 1、实现 KNN 算法
- 2、初步学习 MATLAB 实现 ANN 算法
- 3、实现决策树算法

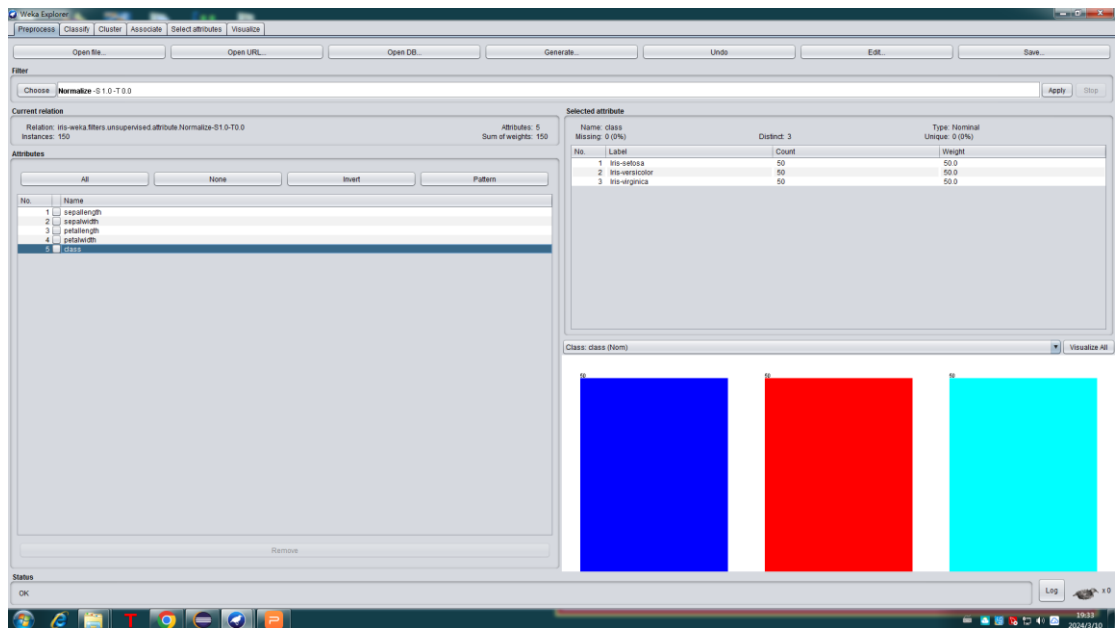
九、实验数据及结果分析：

(一)

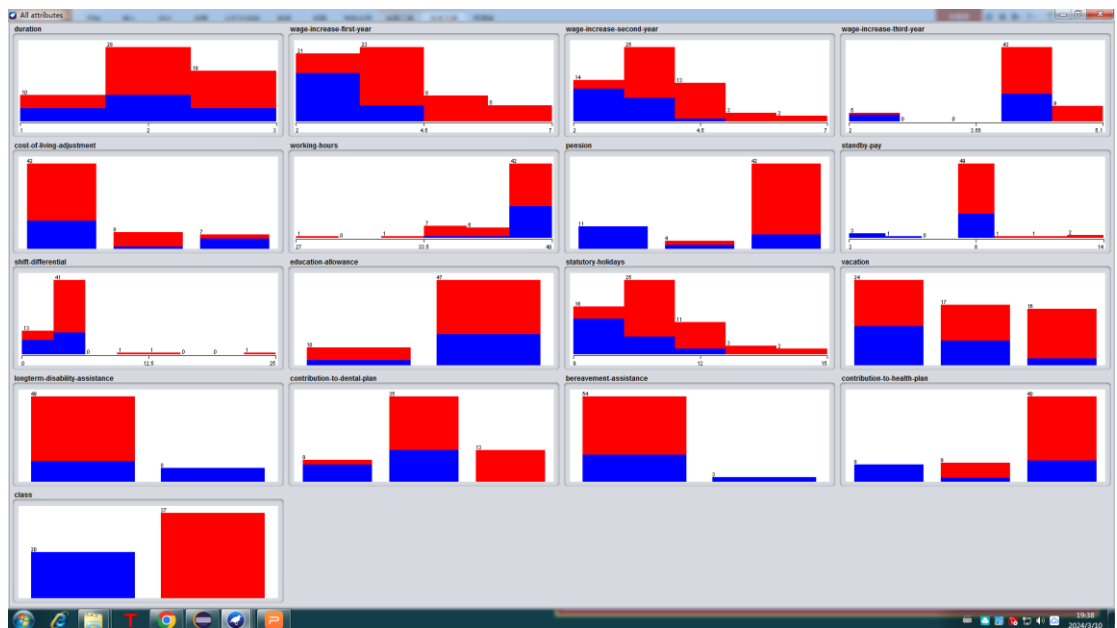
1.1 使用 Weka 图形界面工具完成数据归一化



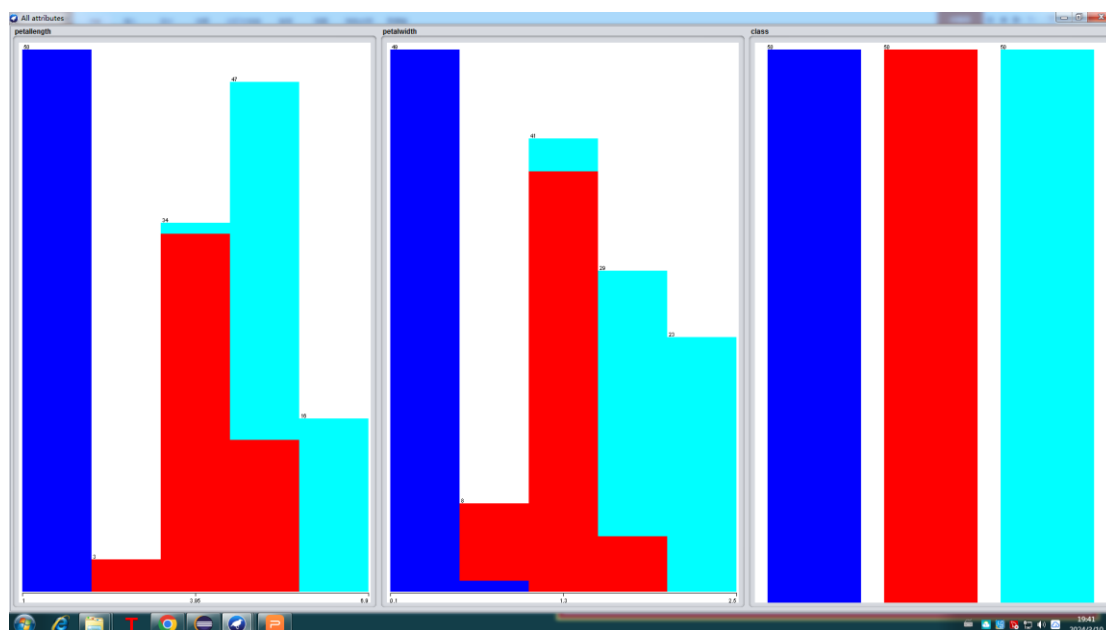




1.2、使用 Weka 图形界面工具完成数据缺失值处理



1.3、使用 Weka 图形界面工具完成特征筛选



2.1、在 Eclipse 下调用 Weka.jar 包完成数据归一化

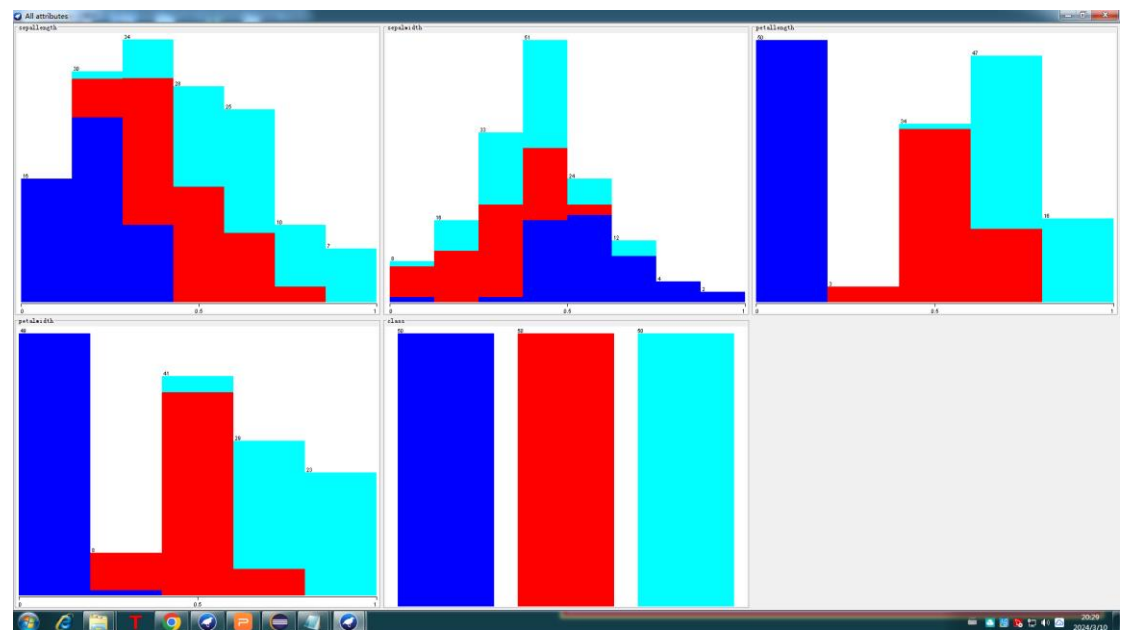
代码如下：

```
130 public static void main(String[] args) throws Exception{
131     System.out.println("*****Example 1:Normalize Data via weka. *****");
132     //weka
133     System.out.println("Step 1. Load...");
134     DataSource source = new DataSource("C:/Program Files/Weka-3-8-5/data/iris.arff");
135     Instances instances = source.getDataSet();
136     //weka
137     System.out.println("Step 2. a=...");
138     Normalize norm = new Normalize();
139     norm.setInputFormat(instances);
140     Instances newInstances = Filter.useFilter(instances, norm);
141     System.out.println("Step 3. a=...");
142     System.out.println("*****");
143     try {
144         //result
145         int numInstances = newInstances.numInstances();
146         for(int i = 0; i < numInstances; i++){
147             Attribute attribute = newInstances.attribute(i);
148             System.out.print(attribute.name() + " ");
149         }
150         System.out.println();
151         //result
152         int numInstances = newInstances.numInstances();
153         for(int i = 0; i < numInstances; i++){
154             Instance instance = newInstances.instance(i);
155             System.out.print(instance.toString() + " ");
156             System.out.println();
157         }
158     } catch (Exception e) {
159         e.printStackTrace();
160     }
161     System.out.println("Step 4. Run-convert...");
162     System.out.println("*****");
163     //weka-convert
164     DetachLinker linker = new DetachLinker("C:/Program Files/Weka-3-8-5/data/iris_norm.arff", newInstances);
165     System.out.println("Congratulations");
166 }
167 }
```

terminated: test [Java Application] C:/Program Files/Java/jre1.8.0_81/bin/javaw.exe (2024年3月10日 下午8:27:20)

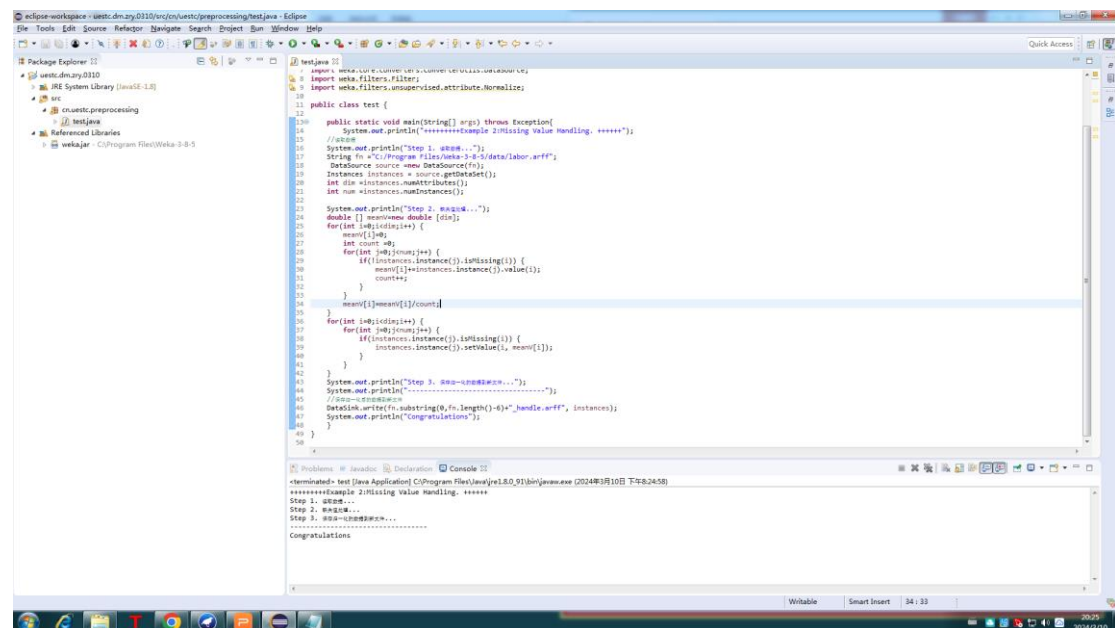
0.69444,0.5,0.83958,0.91667,Iris-virginica
0.66667,0.54167,0.79167,Iris-virginica
0.66667,0.41667,0.71164,0.91667,Iris-virginica
0.55556,0.38333,0.67963,0.79,Iris-virginica
0.61111,0.41667,0.71164,0.79167,Iris-virginica
0.52778,0.58333,0.76767,0.91667,Iris-virginica
0.64444,0.41667,0.69415,0.78833,Iris-virginica

Step 4. Run-convert...
Congratulations

[illegible]

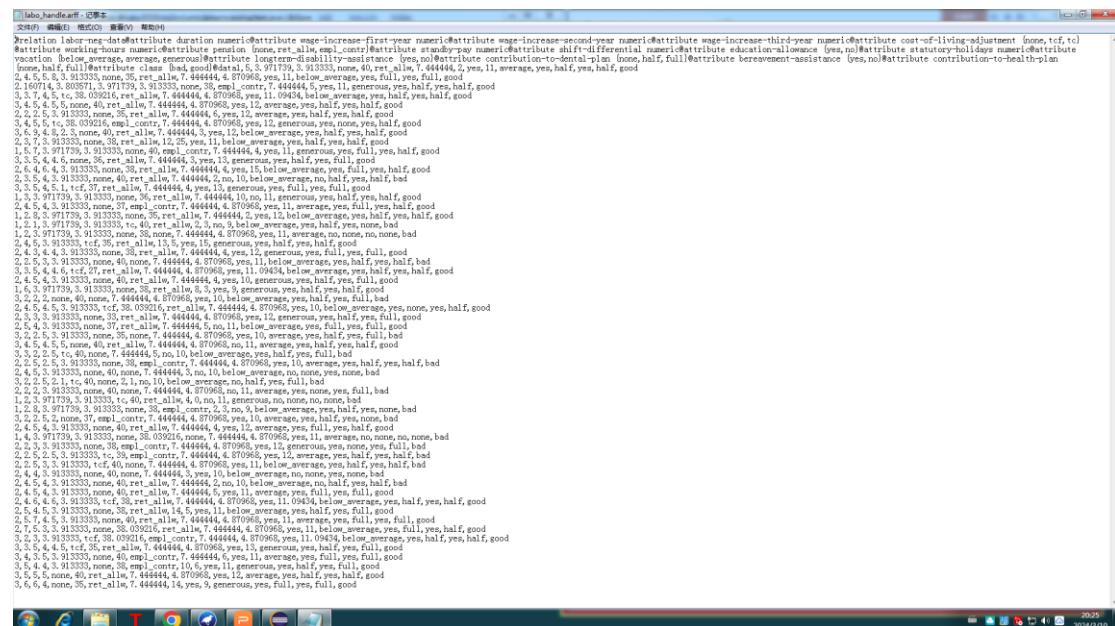
2.2、在 Eclipse 下调用 Weka.jar 包完成数据缺失值处理

代码如下：

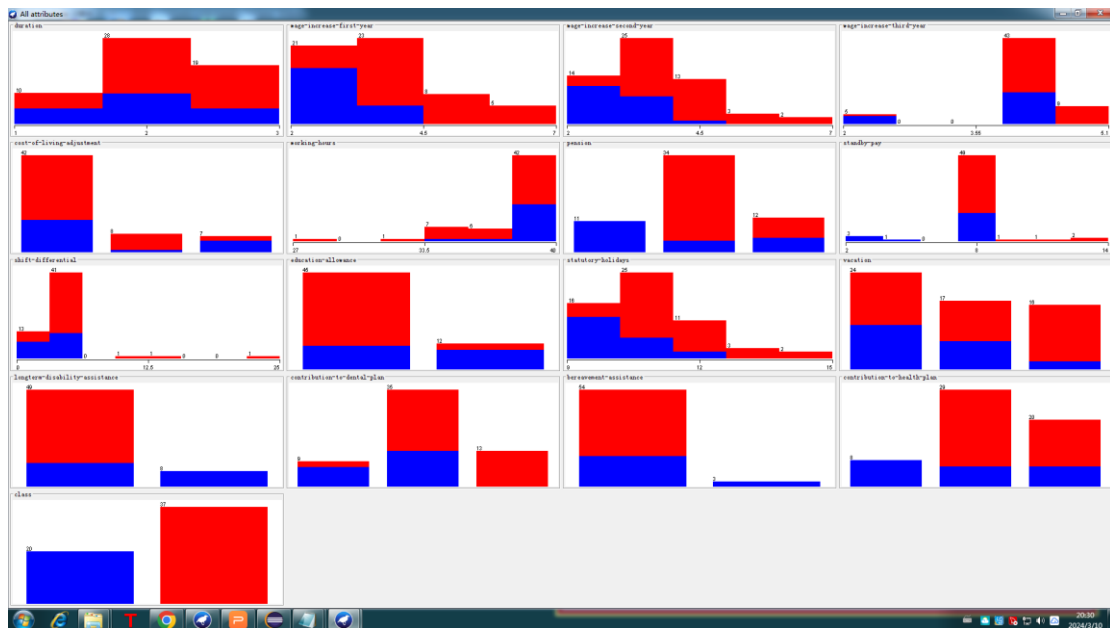


```
1 import weka.core.converters.ArffLoader;
2 import weka.filters.Filter;
3 import weka.filters.unsupervised.attribute.Normalize;
4
5 public class test {
6
7     public static void main(String[] args) throws Exception {
8         System.out.println("=====Example 2:Missing Value Handling. =====");
9
10        // Load
11        String fn = "C:/Program Files/Weka-3.8.5/data/labor.arff";
12        DataSource source = new DataSource(fn);
13        Instances instances = source.getDataSet();
14        int numInstances = instances.numInstances();
15
16        System.out.println("Step 2. Read Data...");
17        double[] mean = new double[instances.numAttributes()];
18        for(int i=0; i<instances.numAttributes(); i++) {
19            mean[i] = 0;
20            for(int j=0; j<instances.numInstances(); j++) {
21                if(!instances.instance(j).isMissing(i)) {
22                    mean[i] += instances.instance(j).value(i);
23                }
24            }
25            mean[i] /= instances.numInstances();
26        }
27
28        for(int i=0; i<instances.numAttributes(); i++) {
29            for(int j=0; j<instances.numInstances(); j++) {
30                if(!instances.instance(j).isMissing(i)) {
31                    instances.instance(j).setValue(i, mean[i]);
32                }
33            }
34        }
35
36        System.out.println("Step 3. Save the modified dataset...");
37        System.out.println("Saving the modified dataset...");
38        DatasetWriter writer = new DatasetWriter(fn, "arff");
39        writer.writeForSave(instances);
40        System.out.println("Congratulations");
41    }
42}
```

数据结果：



```
labor.handle.arff
@relation labor
@attribute duration numeric
@attribute education numeric
@attribute experience numeric
@attribute age-increase-first-year numeric
@attribute age-increase-second-year numeric
@attribute age-increase-third-year numeric
@attribute cost-of-living-adjustment numeric
@attribute vacation numeric
@attribute below-average generic
@attribute long-term-disability assistance yes/no
@attribute contribution-to-dental-plan yes/no
@attribute bereavement-assistance yes/no
@attribute contribution-to-health-plan yes/no
@attribute working-hours numeric
@attribute pension yes/no
@attribute standby-pay numeric
@attribute shift-differential numeric
@attribute education-allowance yes/no
@attribute statutory-holidays numeric
@attribute tenure half/full
@attribute class bad/good
@attribute 3, 91333, none, 35, ret_allow, 7.44444, 4.87096, yes, 11, below_average, yes, full, yes, full, good
2, 16071, 4.80887, 3.97179, 3.91333, none, 35, empl_contr, 7.44444, 4.87096, yes, 11, generous, yes, half, yes, half, good
3, 5, 7, 4, 5, tc, 38.03216, ret_allow, 7.44444, 4.87096, yes, 11, 0.9434, below_average, yes, half, yes, half, good
3, 5, 4, 5, 5, none, 40, ret_allow, 7.44444, 4.87096, yes, 12, average, yes, half, yes, half, good
2, 5, 5, 3, 91333, none, 40, none, 7.44444, 4.87096, yes, 11, below_average, no, half, yes, half, bad
3, 5, 4, 5, tc, 38.03216, empl_contr, 7.44444, 4.87096, yes, 12, generous, yes, none, yes, half, good
3, 5, 4, 5, 5, none, 40, ret_allow, 7.44444, 4.87096, yes, 12, below_average, yes, half, yes, half, good
2, 5, 7, 3, 91333, none, 35, ret_allow, 12.25, yes, 11, below_average, yes, half, yes, half, good
1, 7, 4, 5, 917179, 3.91333, none, 40, empl_contr, 7.44444, 4.87096, yes, 11, generous, yes, full, yes, half, good
3, 5, 4, 4, 6, none, 35, ret_allow, 7.44444, 4.87096, yes, 13, generous, yes, half, yes, full, good
2, 4, 4, 4, 3, 91333, none, 35, ret_allow, 7.44444, 4.87096, yes, 15, below_average, yes, full, yes, half, good
2, 5, 5, 3, 91333, none, 40, ret_allow, 7.44444, 4.87096, yes, 11, below_average, no, half, yes, half, bad
3, 5, 4, 5, 1, tc, 37, ret_allow, 7.44444, 4.87096, yes, 13, generous, yes, full, yes, full, good
1, 3, 917179, 3.91333, none, 35, ret_allow, 7.44444, 4.87096, no, 11, generous, yes, half, yes, half, good
2, 4, 5, 4, 3, 91333, none, 37, empl_contr, 7.44444, 4.87096, yes, 11, average, yes, full, yes, half, good
1, 2, 3, 917179, 3.91333, none, 35, ret_allow, 7.44444, 4.87096, yes, 12, below_average, yes, half, yes, half, good
1, 2, 1, 3, 917179, 3.91333, tc, 40, ret_allow, 2, 3, no, 9, below_average, yes, half, yes, none, bad
1, 2, 3, 917179, 3.91333, none, 35, none, 7.44444, 4.87096, yes, 11, average, no, none, no, none, bad
2, 4, 5, 3, 91333, tc, 35, ret_allow, 15, 5, yes, 10, generous, yes, half, yes, half, good
2, 4, 5, 4, 3, 91333, none, 35, ret_allow, 7.44444, 4.87096, yes, 12, generous, yes, full, yes, full, good
2, 5, 5, 3, 91333, none, 40, none, 7.44444, 4.87096, yes, 11, below_average, yes, half, yes, half, bad
3, 5, 4, 4, 6, tc, 37, ret_allow, 7.44444, 4.87096, yes, 11, 0.9434, below_average, yes, half, yes, half, good
2, 4, 5, 4, 3, 91333, none, 40, ret_allow, 7.44444, 4.87096, yes, 10, generous, yes, half, yes, full, good
1, 6, 3, 917179, 3.91333, none, 35, ret_allow, 8, 3, yes, 9, generous, yes, half, yes, half, good
3, 2, 2, 2, none, 40, none, 7.44444, 4.87096, yes, 10, below_average, yes, half, yes, full, bad
2, 4, 5, 4, 5, 3, 91333, tc, 38.03216, ret_allow, 7.44444, 4.87096, yes, 10, below_average, yes, none, yes, half, good
2, 5, 3, 91333, none, 35, ret_allow, 7.44444, 4.87096, yes, 12, generous, yes, half, yes, full, good
2, 4, 5, 3, 91333, none, 37, ret_allow, 7.44444, 4.87096, yes, 11, below_average, yes, full, yes, full, good
3, 2, 2, 5, 3, 91333, none, 35, none, 7.44444, 4.87096, yes, 10, average, yes, half, yes, full, bad
3, 4, 5, 4, 5, none, 40, ret_allow, 7.44444, 4.87096, no, 11, average, yes, half, yes, half, good
3, 2, 2, 5, tc, 40, none, 7.44444, 4.87096, yes, 10, below_average, yes, half, yes, full, bad
2, 4, 5, 3, 91333, none, 40, none, 7.44444, 4.87096, yes, 10, below_average, no, none, yes, full, bad
3, 2, 2, 5, 3, 91333, none, 40, none, 7.44444, 4.87096, no, 11, average, yes, none, yes, full, bad
1, 2, 3, 917179, 3.91333, tc, 40, ret_allow, 4, 0, no, 11, generous, no, none, no, none, bad
1, 2, 3, 917179, 3.91333, none, 35, empl_contr, 7.44444, 4.87096, yes, 10, average, yes, half, yes, none, bad
3, 2, 2, 5, 2, none, 37, empl_contr, 7.44444, 4.87096, yes, 10, average, yes, half, yes, none, bad
2, 4, 5, 4, 3, 91333, none, 40, ret_allow, 7.44444, 4.87096, yes, 12, average, yes, full, yes, half, good
1, 4, 3, 917179, 3.91333, none, 38.03216, none, 7.44444, 4.87096, yes, 11, average, no, none, no, none, bad
2, 4, 5, 3, 91333, none, 35, empl_contr, 7.44444, 4.87096, yes, 12, generous, yes, none, yes, none, bad
2, 5, 5, 2, 3, 91333, tc, 35, ret_allow, 7.44444, 4.87096, yes, 11, below_average, yes, half, yes, half, bad
2, 5, 5, 3, 91333, tc, 40, none, 7.44444, 4.87096, yes, 11, below_average, yes, half, yes, half, bad
2, 4, 5, 4, 3, 91333, none, 40, none, 7.44444, 4.87096, yes, 10, below_average, no, none, yes, none, bad
2, 4, 5, 4, 3, 91333, none, 40, ret_allow, 7.44444, 4.87096, yes, 11, average, yes, full, yes, full, good
2, 4, 5, 4, 5, 3, 91333, tc, 35, ret_allow, 7.44444, 4.87096, yes, 11, 0.9434, below_average, yes, half, yes, half, good
2, 4, 5, 4, 3, 91333, none, 35, ret_allow, 14, 5, yes, 11, below_average, yes, half, yes, full, good
2, 7, 4, 5, 3, 91333, none, 40, ret_allow, 7.44444, 4.87096, yes, 11, average, yes, full, yes, full, good
2, 7, 4, 5, 3, 91333, tc, 38.03216, empl_contr, 7.44444, 4.87096, yes, 11, 0.9434, below_average, yes, half, yes, half, good
3, 5, 4, 5, tc, 35, ret_allow, 7.44444, 4.87096, yes, 13, generous, yes, half, yes, full, good
3, 5, 4, 3, 91333, none, 40, empl_contr, 7.44444, 4.87096, yes, 11, average, yes, full, yes, full, good
3, 5, 4, 3, 91333, none, 35, empl_contr, 10, 5, yes, 11, generous, yes, half, yes, full, good
3, 5, 5, 5, none, 40, ret_allow, 7.44444, 4.87096, yes, 12, average, yes, half, yes, half, good
3, 5, 6, 4, none, 35, ret_allow, 7.44444, 14, yes, 9, generous, yes, full, yes, full, good
```



2.3、在 Eclipse 下调用 Weka.jar 包完成特征筛选

代码如下：

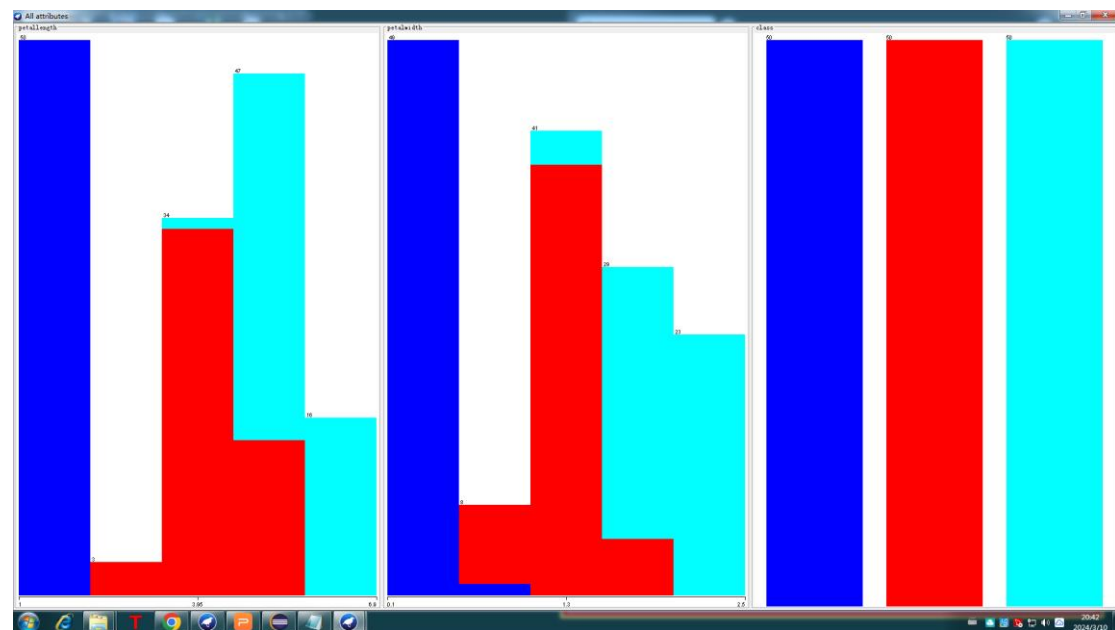
```
1 package ch.unist.preprocessing;
2
3 import weka.attributeSelection.AttributeEval;
4 import weka.attributeSelection.InfoGainAttributeEval;
5 import weka.attributeSelection.Ranker;
6 import weka.core.Instances;
7 import weka.core.converters.ConvertUtils;
8 import weka.core.converters.DataSource;
9 import weka.filters.Filter;
10 import weka.filters.supervised.attribute.AttributeSelection;
11
12
13 public class test {
14
15     public static void main(String[] args) throws Exception{
16         System.out.println("=====Example 3: Feature Selection via weka. =====");
17         //Step 1
18         System.out.println("Step 1. Read...");
19         DataSource source = new DataSource("C:/Program Files/Weka-3-8-5/data/iris.arff");
20         Instances instances = source.getDataSet();
21
22         System.out.println("Step 2. @INFO...");
23         int n = instances.numInstances();
24         AttributeSelection as = new AttributeSelection();
25         Ranker rank = new Ranker();
26         rank.setThreshold(0.0);
27         rank.setNumSelect(5);
28
29         AttributeEval ae = new InfoGainAttributeEval();
30         as.setEvaluator(ae);
31         as.setSearch(rank);
32         as.setNumFolds(10);
33         Instances reducedData = Filter.useFilter(instances, as);
34
35         System.out.println("Step 3. Save...");
36         System.out.println("Saving...");
37         //Save to arff
38         ConvertUtils.save(reducedData, "C:/Program Files/Weka-3-8-5/data/iris_reduced.arff");
39         System.out.println("Congratulations");
40     }
41 }
42
```

Problems | JavaDoc | Declaration | Console

```
<terminated> test [Java Application] C:/Program Files/Java/jre1.8.0_91/bin/javaw.exe (2024年3月10日 下午6:41:26)
=====Example 3: Feature Selection via weka. =====
Step 1. Read...
Step 2. @INFO...
Step 3. Save...
Congratulations
```

数据结果：

```
iris_reduced.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
Pratition iris=weka.filters.supervised.attribute.AttributeSelection-@weka.attributeSelection.InfoGainAttributeEval-@weka.attributeSelection.Ranker -T 0.0 -M 2 @attribute petal.length numeric @attribute petal.width numeric @attribute class
iris=petosa,iris-versicolour,iris-virginica#@a1.4,5,2,iris=petosa
1.4,0.2,iris=petosa
1.5,0.2,iris=petosa
1.5,0.2,iris=petosa
1.5,0.2,iris=petosa
1.7,0.4,iris=petosa
1.4,0.3,iris=petosa
1.5,0.2,iris=petosa
1.4,0.2,iris=petosa
1.5,0.1,iris=petosa
1.5,0.2,iris=petosa
1.6,0.2,iris=petosa
1.4,0.1,iris=petosa
1.1,0.1,iris=petosa
1.2,0.2,iris=petosa
1.5,0.4,iris=petosa
1.3,0.4,iris=petosa
1.4,0.3,iris=petosa
1.7,0.3,iris=petosa
1.5,0.3,iris=petosa
1.7,0.2,iris=petosa
1.5,0.4,iris=petosa
1.0,2,iris=petosa
1.7,0.5,iris=petosa
1.0,0.2,iris=petosa
1.6,0.2,iris=petosa
1.6,0.4,iris=petosa
1.5,0.2,iris=petosa
1.4,0.2,iris=petosa
1.6,0.2,iris=petosa
1.6,0.2,iris=petosa
1.5,0.4,iris=petosa
1.5,0.1,iris=petosa
1.4,0.2,iris=petosa
1.5,0.1,iris=petosa
1.2,0.2,iris=petosa
1.3,0.2,iris=petosa
1.5,0.1,iris=petosa
1.3,0.2,iris=petosa
1.5,0.2,iris=petosa
1.3,0.3,iris=petosa
1.3,0.2,iris=petosa
1.6,0.6,iris=petosa
1.5,0.4,iris=petosa
1.4,0.3,iris=petosa
1.6,0.2,iris=petosa
1.4,0.2,iris=petosa
1.5,0.2,iris=petosa
1.4,0.2,iris=petosa
1.4,0.2,iris=petosa
4.7,1.4,iris=versicolour
4.5,1.5,iris=versicolour
4.5,1.5,iris=versicolour
4.5,1.5,iris=versicolour
4.7,1.4,iris=versicolour
4.7,1.4,iris=versicolour
3.3,1,iris=versicolour
4.6,1.3,iris=versicolour
3.5,1.4,iris=versicolour
3.5,1,iris=versicolour
```



(二)

代码如下：

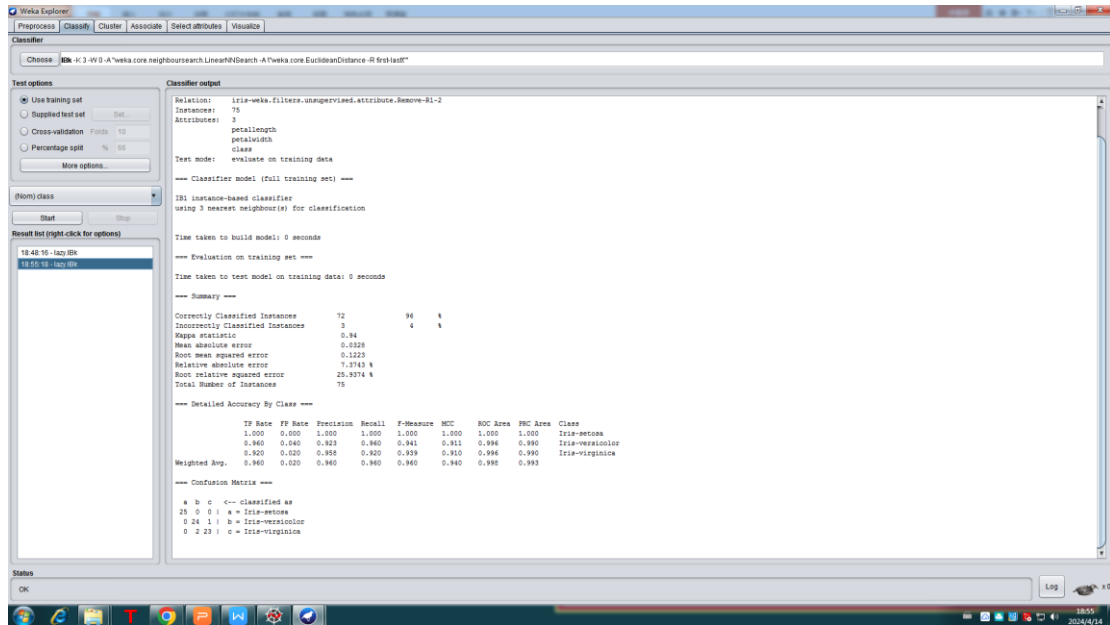
```
dm X
C:\Users\Albert> Documents > Visual Studio 2022 > dm > ...
1 import collections
2 import itertools
3
4 traDatas = ['abe','ae','abc','ade',]
5
6 class Apriori:
7     traDatas = []
8     traLen = 0
9     k = 1
10    traCount = {}
11    freTran = {}
12    sup = 0
13    conf = 0
14    freAllTran = {}
15    def __init__(self, traDatas, sup, conf):
16        self.traDatas = traDatas
17        self.traLen = len(traDatas)
18        self.sup = sup
19        self.conf = conf
20
21    def scanFirDatas(self):
22        tmpStr = ''.join(traDatas)
23        self.traCount = dict(collections.Counter(tmpStr))
24
25    def getFreSet(self):
26        self.freTran = {}
27        for tra in self.traCount.keys():
28            if self.traCount[tra] >= self.sup and len(tra) == self.k:
29                self.freTran[tra] = self.traCount[tra]
30                self.freAllTran[tra] = self.traCount[tra]
31
32    def cmpTwoSet(self, setA, setB):
33        setA = set(setA)
34        setB = set(setB)
35        if len(setA-setB) == 1 and len(setB-setA) == 1:
36            return True
37        else:
38            return False
39
40    def selfConn(self):
41        self.traCount = {}
42        for item in itertools.combinations(self.freTran.keys(), 2):
43            if self.cmpTwoSet(item[0], item[1]) == True:
44                key = ''.join(sorted(''.join(set(item[0]).union(set(item[1])))))
45                if self.cutBranch(key) != False:
46                    self.traCount[key] = 0
47
```

```
dm X
C:\Users\Albert> Documents> Visual Studio 2022> dm> ...
6 class Apriori:
40 def selfConn(self):
41     self.traCount = {}
42     for item in itertools.combinations(self.freTran.keys(), 2):
43         if self.cmpTwoSet(item[0], item[1]) == True:
44             key = ''.join(sorted(''.join(set(item[0]).union(set(item[1])))))
45             if self.cutBranch(key) != False:
46                 self.traCount[key] = 0
47
48 def scanDatas(self):
49     self.k += 1
50     for tra in traDatas:
51         for key in self.traCount.keys():
52             self.traCount[key] += self.findChars(tra, key)
53
54 def cutBranch(self, key):
55     for subkey in list(itertools.combinations(key, self.k)):
56         if ''.join(list(subkey)) not in self.freTran.keys():
57             return False
58
59 def findChars(self, str, chars):
60     for char in list(chars):
61         if char not in str:
62             return False
63     return 1
64
65 def permutation(self, string, pre_str, container):
66     if len(string) == 1:
67         container.append(pre_str + string)
68     for idx, str in enumerate(string):
69         new_str = string[:idx] + string[idx + 1:]
70         new_pre_str = pre_str + str
71         self.permutation(new_str, new_pre_str, container)
72
73 def genAssRule(self):
74     container = []
75     ruleSet = set()
76     for item in self.freTran.keys():
77         self.permutation(item, '', container)
78     for item in container:
79         for i in range(1, len(item)):
80             print(item[:i] + " " + item[i:])
81             ruleSet.add(''.join(sorted(item[:i])), ''.join(sorted(item[i:])))
82     for rule in ruleSet:
83         if self.calcConfi(rule[0], rule[1]) > self.conf:
84             print(rule[0] + " -> " + rule[1])
85
86 def calcConfi(self, first, last):
87     return self.freAllTran[''.join(sorted(first+last))]/self.freAllTran[''.join(sorted(first))]
```

```
dm X
C:\Users\Albert> Documents> Visual Studio 2022> dm> ...
6 class Apriori:
87
88 def algorithm(self):
89     self.scanInDatas()
90     while self.traCount != {}:
91         self.getFreSet()
92         self.selfConn()
93         self.scanDatas()
94         print(self.freAllTran)
95         print(self.freTran)
96         self.genAssRule()
97
98 apriori = Apriori(traDatas, 2, 0.7)
99 apriori.algorithm()
100
```

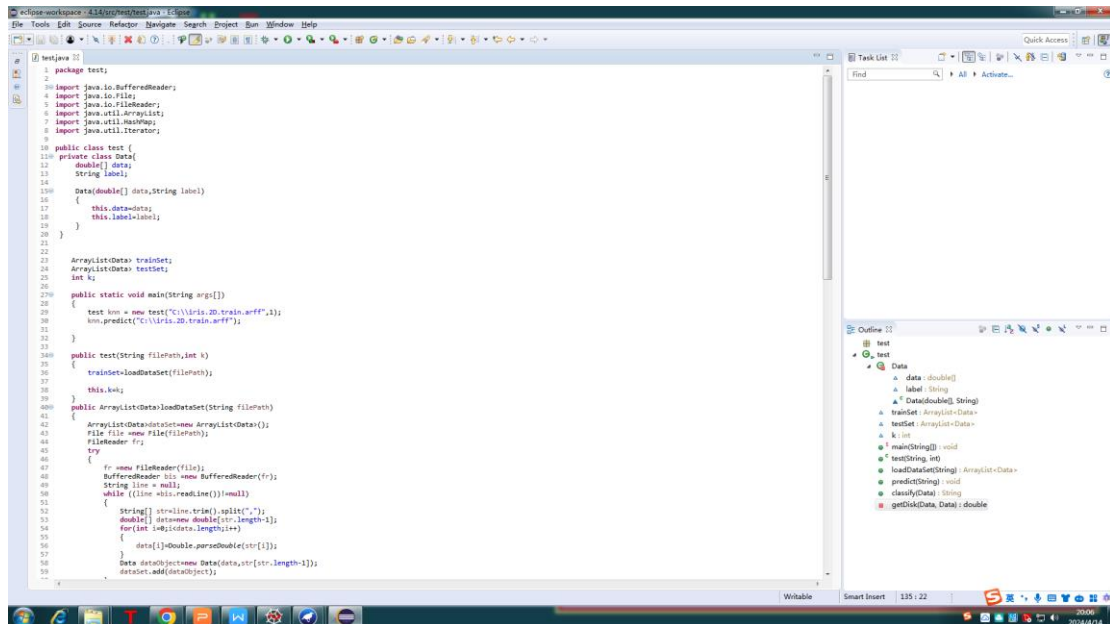
行 1, 列 19 空格 4 UTF-8 CRLF Python 3.11.8 64-bit

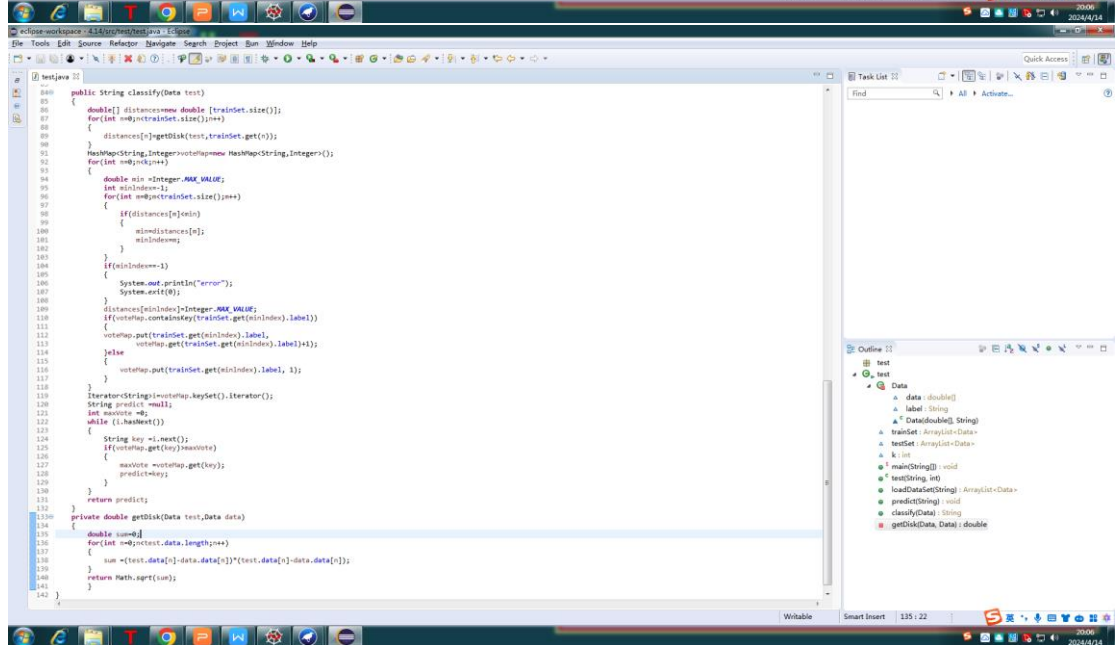
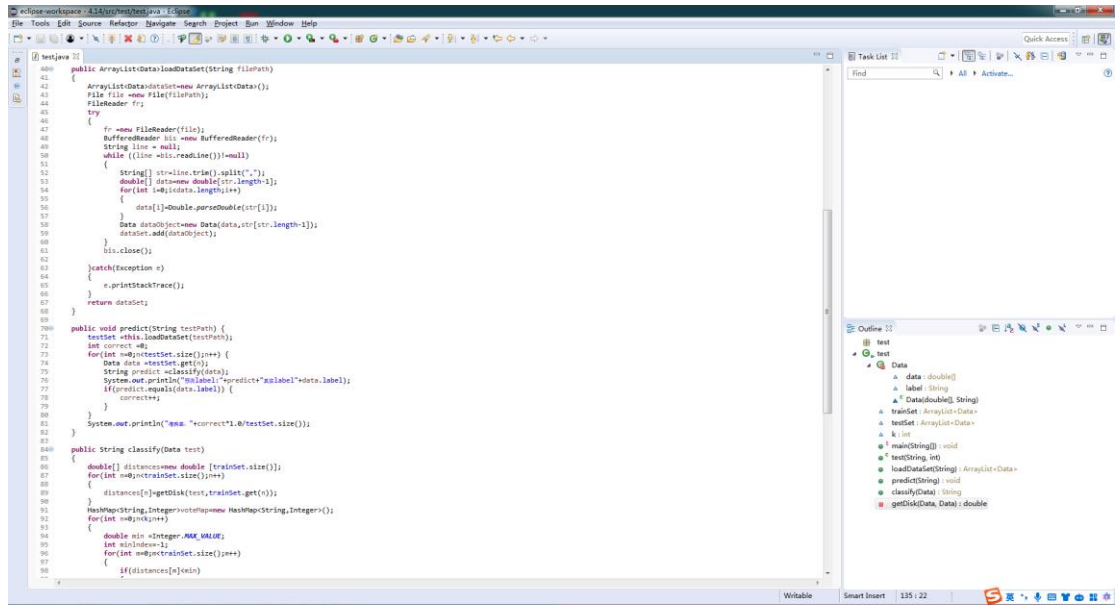
数据结果：



2、使用 JAVA 编写 KNN 算法

代码如下：



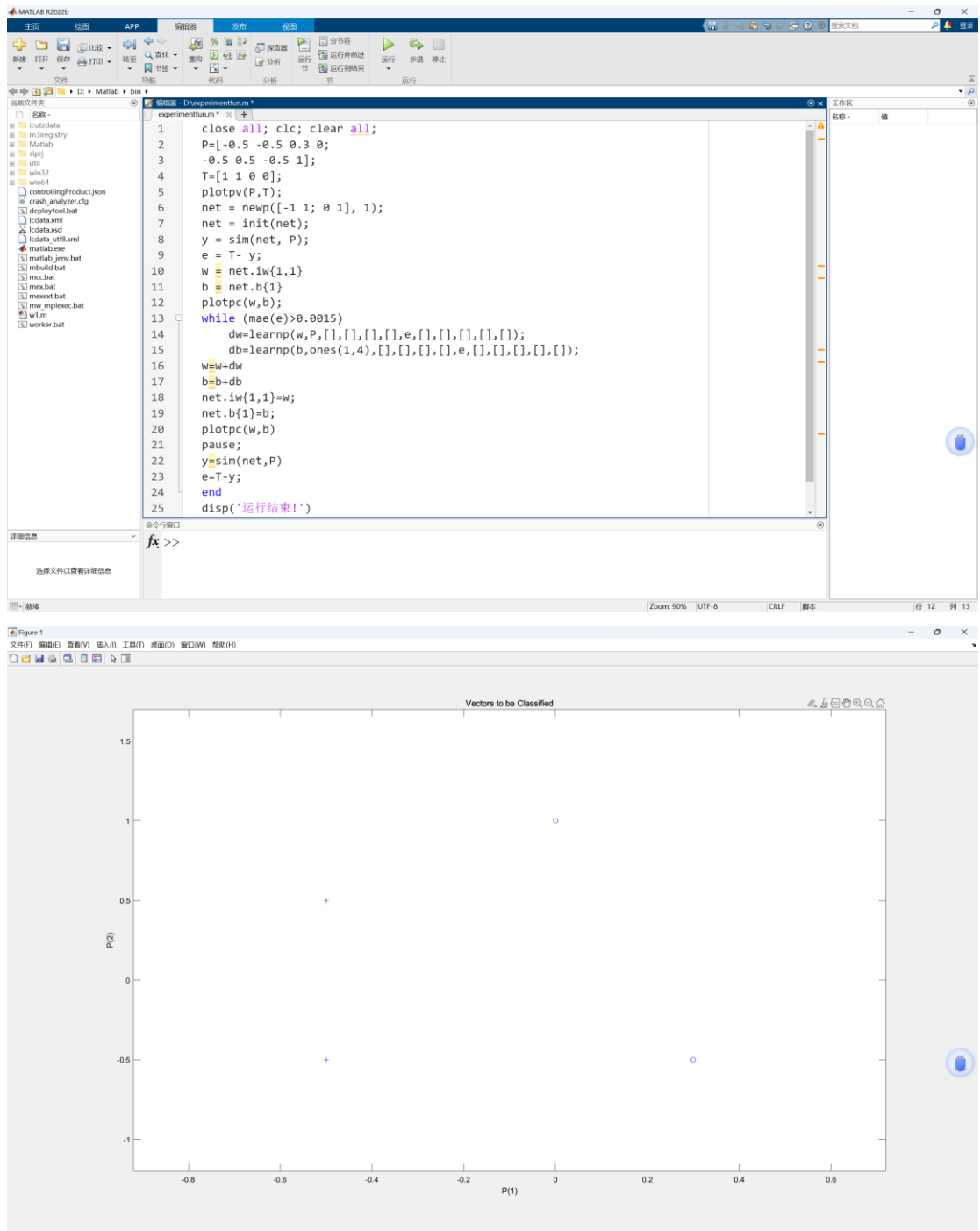


运行结果：

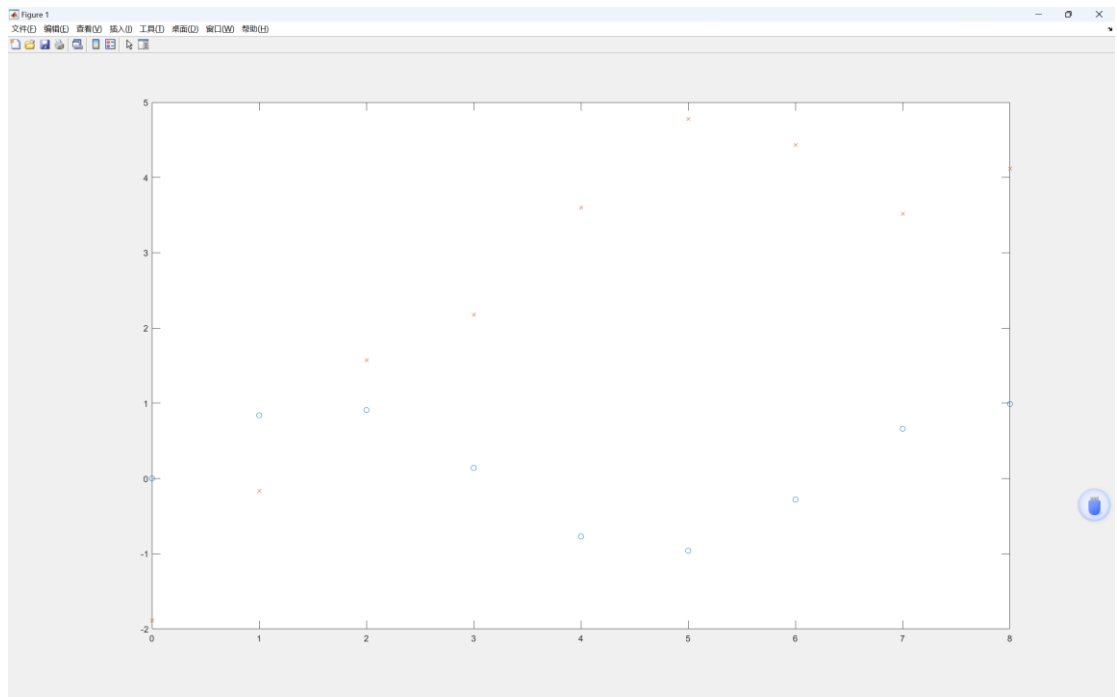
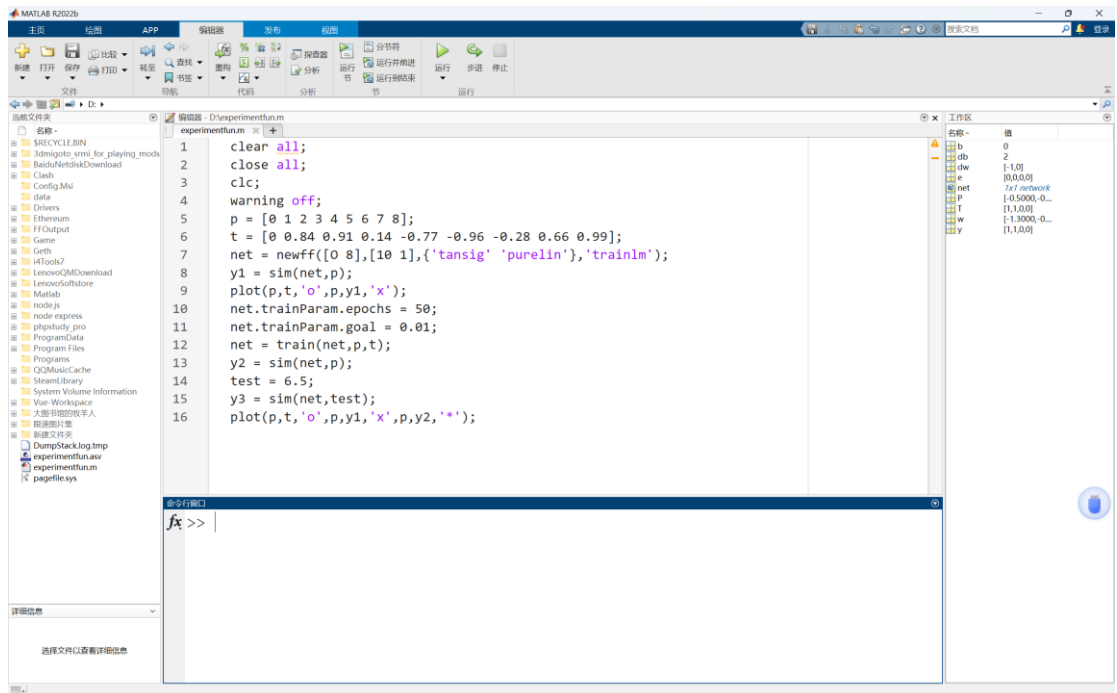
The image displays two screenshots of the Eclipse IDE. The top screenshot shows the source code of a Java class named 'test'. The code includes imports for various Java classes like 'BufferedReader', 'File', 'FileReader', 'ArrayList', 'HashMap', and 'Iterator'. It defines a 'Data' class with attributes 'data' (double[]) and 'label' (String). The 'main' method initializes a 'trainSet' and 'testSet' of 'Data' objects, loads data from files, and performs a classification task using a neural network. The bottom screenshot shows the 'Console' view with the execution output. It displays the results of the classification, including the predicted class for each data point and the overall accuracy of the model.

```
1 package test;
2
3 import java.io.BufferedReader;
4 import java.io.File;
5 import java.io.FileReader;
6 import java.util.ArrayList;
7 import java.util.HashMap;
8 import java.util.Iterator;
9
10 public class test {
11     private class Data{
12         double[] data;
13         String label;
14
15         Data(double[] data,String label)
16         {
17             this.data=data;
18             this.label=label;
19         }
20     }
21
22     ArrayList<Data> trainSet;
23     ArrayList<Data> testSet;
24     int k;
25
26     public static void main(String args[])
27     {
28         test t = new test("C:\\iris.20.train.arff",1);
29         t.trainSet=t.loadDataSet("C:\\iris.20.train.arff");
30     }
31
32     public test(String filePath,int k)
33     {
34         trainSet=loadDataSet(filePath);
35     }
36     this.k=k;
37
38     public ArrayList<Data>loadDataSet(String filePath)
39     {
40         ArrayList<Data>dataSet=new ArrayList<Data>();
41         try {
42             BufferedReader br=new BufferedReader(new FileReader(filePath));
43             String line;
44             while((line=br.readLine())!=null){
45                 String[] items=line.split(",");
46                 double[] data=new double[items.length-1];
47                 for(int i=0;i<data.length;i++){
48                     data[i]=Double.parseDouble(items[i]);
49                 }
50                 String label=items[items.length-1];
51                 Data d=new Data(data,label);
52                 dataSet.add(d);
53             }
54         } catch (Exception e) {
55             e.printStackTrace();
56         }
57         return dataSet;
58     }
59
60     public void train()
61     {
62         // Training logic
63     }
64
65     public void test()
66     {
67         // Testing logic
68     }
69
70     public void predict(String filePath)
71     {
72         // Prediction logic
73     }
74
75     public void printResult()
76     {
77         // Print result logic
78     }
79
80     public void printAccuracy()
81     {
82         // Print accuracy logic
83     }
84
85     public void printConfusionMatrix()
86     {
87         // Print confusion matrix logic
88     }
89
90     public void printPrecisionRecall()
91     {
92         // Print precision recall logic
93     }
94
95     public void printF1Score()
96     {
97         // Print F1 score logic
98     }
99
100     public void printROC()
101     {
102         // Print ROC logic
103     }
104
105     public void printAUC()
106     {
107         // Print AUC logic
108     }
109
110     public void printPrecision()
111     {
112         // Print precision logic
113     }
114
115     public void printRecall()
116     {
117         // Print recall logic
118     }
119
120     public void printF1()
121     {
122         // Print F1 logic
123     }
124
125     public void printROC()
126     {
127         // Print ROC logic
128     }
129
130     public void printAUC()
131     {
132         // Print AUC logic
133     }
134
135     public void printPrecision()
136     {
137         // Print precision logic
138     }
139
140     public void printRecall()
141     {
142         // Print recall logic
143     }
144
145     public void printF1()
146     {
147         // Print F1 logic
148     }
149
150     public void printROC()
151     {
152         // Print ROC logic
153     }
154
155     public void printAUC()
156     {
157         // Print AUC logic
158     }
159
160     public void printPrecision()
161     {
162         // Print precision logic
163     }
164
165     public void printRecall()
166     {
167         // Print recall logic
168     }
169
170     public void printF1()
171     {
172         // Print F1 logic
173     }
174
175     public void printROC()
176     {
177         // Print ROC logic
178     }
179
180     public void printAUC()
181     {
182         // Print AUC logic
183     }
184
185     public void printPrecision()
186     {
187         // Print precision logic
188     }
189
190     public void printRecall()
191     {
192         // Print recall logic
193     }
194
195     public void printF1()
196     {
197         // Print F1 logic
198     }
199
200     public void printROC()
201     {
202         // Print ROC logic
203     }
204
205     public void printAUC()
206     {
207         // Print AUC logic
208     }
209
210     public void printPrecision()
211     {
212         // Print precision logic
213     }
214
215     public void printRecall()
216     {
217         // Print recall logic
218     }
219
220     public void printF1()
221     {
222         // Print F1 logic
223     }
224
225     public void printROC()
226     {
227         // Print ROC logic
228     }
229
230     public void printAUC()
231     {
232         // Print AUC logic
233     }
234
235     public void printPrecision()
236     {
237         // Print precision logic
238     }
239
240     public void printRecall()
241     {
242         // Print recall logic
243     }
244
245     public void printF1()
246     {
247         // Print F1 logic
248     }
249
250     public void printROC()
251     {
252         // Print ROC logic
253     }
254
255     public void printAUC()
256     {
257         // Print AUC logic
258     }
259
260     public void printPrecision()
261     {
262         // Print precision logic
263     }
264
265     public void printRecall()
266     {
267         // Print recall logic
268     }
269
270     public void printF1()
271     {
272         // Print F1 logic
273     }
274
275     public void printROC()
276     {
277         // Print ROC logic
278     }
279
280     public void printAUC()
281     {
282         // Print AUC logic
283     }
284
285     public void printPrecision()
286     {
287         // Print precision logic
288     }
289
290     public void printRecall()
291     {
292         // Print recall logic
293     }
294
295     public void printF1()
296     {
297         // Print F1 logic
298     }
299
300     public void printROC()
301     {
302         // Print ROC logic
303     }
304
305     public void printAUC()
306     {
307         // Print AUC logic
308     }
309
310     public void printPrecision()
311     {
312         // Print precision logic
313     }
314
315     public void printRecall()
316     {
317         // Print recall logic
318     }
319
320     public void printF1()
321     {
322         // Print F1 logic
323     }
324
325     public void printROC()
326     {
327         // Print ROC logic
328     }
329
330     public void printAUC()
331     {
332         // Print AUC logic
333     }
334
335     public void printPrecision()
336     {
337         // Print precision logic
338     }
339
340     public void printRecall()
341     {
342         // Print recall logic
343     }
344
345     public void printF1()
346     {
347         // Print F1 logic
348     }
349
350     public void printROC()
351     {
352         // Print ROC logic
353     }
354
355     public void printAUC()
356     {
357         // Print AUC logic
358     }
359
360     public void printPrecision()
361     {
362         // Print precision logic
363     }
364
365     public void printRecall()
366     {
367         // Print recall logic
368     }
369
370     public void printF1()
371     {
372         // Print F1 logic
373     }
374
375     public void printROC()
376     {
377         // Print ROC logic
378     }
379
380     public void printAUC()
381     {
382         // Print AUC logic
383     }
384
385     public void printPrecision()
386     {
387         // Print precision logic
388     }
389
390     public void printRecall()
391     {
392         // Print recall logic
393     }
394
395     public void printF1()
396     {
397         // Print F1 logic
398     }
399
400     public void printROC()
401     {
402         // Print ROC logic
403     }
404
405     public void printAUC()
406     {
407         // Print AUC logic
408     }
409
410     public void printPrecision()
411     {
412         // Print precision logic
413     }
414
415     public void printRecall()
416     {
417         // Print recall logic
418     }
419
420     public void printF1()
421     {
422         // Print F1 logic
423     }
424
425     public void printROC()
426     {
427         // Print ROC logic
428     }
429
430     public void printAUC()
431     {
432         // Print AUC logic
433     }
434
435     public void printPrecision()
436     {
437         // Print precision logic
438     }
439
440     public void printRecall()
441     {
442         // Print recall logic
443     }
444
445     public void printF1()
446     {
447         // Print F1 logic
448     }
449
450     public void printROC()
451     {
452         // Print ROC logic
453     }
454
455     public void printAUC()
456     {
457         // Print AUC logic
458     }
459
460     public void printPrecision()
461     {
462         // Print precision logic
463     }
464
465     public void printRecall()
466     {
467         // Print recall logic
468     }
469
470     public void printF1()
471     {
472         // Print F1 logic
473     }
474
475     public void printROC()
476     {
477         // Print ROC logic
478     }
479
480     public void printAUC()
481     {
482         // Print AUC logic
483     }
484
485     public void printPrecision()
486     {
487         // Print precision logic
488     }
489
490     public void printRecall()
491     {
492         // Print recall logic
493     }
494
495     public void printF1()
496     {
497         // Print F1 logic
498     }
499
500     public void printROC()
501     {
502         // Print ROC logic
503     }
504
505     public void printAUC()
506     {
507         // Print AUC logic
508     }
509
510     public void printPrecision()
511     {
512         // Print precision logic
513     }
514
515     public void printRecall()
516     {
517         // Print recall logic
518     }
519
520     public void printF1()
521     {
522         // Print F1 logic
523     }
524
525     public void printROC()
526     {
527         // Print ROC logic
528     }
529
530     public void printAUC()
531     {
532         // Print AUC logic
533     }
534
535     public void printPrecision()
536     {
537         // Print precision logic
538     }
539
540     public void printRecall()
541     {
542         // Print recall logic
543     }
544
545     public void printF1()
546     {
547         // Print F1 logic
548     }
549
550     public void printROC()
551     {
552         // Print ROC logic
553     }
554
555     public void printAUC()
556     {
557         // Print AUC logic
558     }
559
560     public void printPrecision()
561     {
562         // Print precision logic
563     }
564
565     public void printRecall()
566     {
567         // Print recall logic
568     }
569
570     public void printF1()
571     {
572         // Print F1 logic
573     }
574
575     public void printROC()
576     {
577         // Print ROC logic
578     }
579
580     public void printAUC()
581     {
582         // Print AUC logic
583     }
584
585     public void printPrecision()
586     {
587         // Print precision logic
588     }
589
590     public void printRecall()
591     {
592         // Print recall logic
593     }
594
595     public void printF1()
596     {
597         // Print F1 logic
598     }
599
600     public void printROC()
601     {
602         // Print ROC logic
603     }
604
605     public void printAUC()
606     {
607         // Print AUC logic
608     }
609
610     public void printPrecision()
611     {
612         // Print precision logic
613     }
614
615     public void printRecall()
616     {
617         // Print recall logic
618     }
619
620     public void printF1()
621     {
622         // Print F1 logic
623     }
624
625     public void printROC()
626     {
627         // Print ROC logic
628     }
629
630     public void printAUC()
631     {
632         // Print AUC logic
633     }
634
635     public void printPrecision()
636     {
637         // Print precision logic
638     }
639
640     public void printRecall()
641     {
642         // Print recall logic
643     }
644
645     public void printF1()
646     {
647         // Print F1 logic
648     }
649
650     public void printROC()
651     {
652         // Print ROC logic
653     }
654
655     public void printAUC()
656     {
657         // Print AUC logic
658     }
659
660     public void printPrecision()
661     {
662         // Print precision logic
663     }
664
665     public void printRecall()
666     {
667         // Print recall logic
668     }
669
670     public void printF1()
671     {
672         // Print F1 logic
673     }
674
675     public void printROC()
676     {
677         // Print ROC logic
678     }
679
680     public void printAUC()
681     {
682         // Print AUC logic
683     }
684
685     public void printPrecision()
686     {
687         // Print precision logic
688     }
689
690     public void printRecall()
691     {
692         // Print recall logic
693     }
694
695     public void printF1()
696     {
697         // Print F1 logic
698     }
699
700     public void printROC()
701     {
702         // Print ROC logic
703     }
704
705     public void printAUC()
706     {
707         // Print AUC logic
708     }
709
710     public void printPrecision()
711     {
712         // Print precision logic
713     }
714
715     public void printRecall()
716     {
717         // Print recall logic
718     }
719
720     public void printF1()
721     {
722         // Print F1 logic
723     }
724
725     public void printROC()
726     {
727         // Print ROC logic
728     }
729
730     public void printAUC()
731     {
732         // Print AUC logic
733     }
734
735     public void printPrecision()
736     {
737         // Print precision logic
738     }
739
740     public void printRecall()
741     {
742         // Print recall logic
743     }
744
745     public void printF1()
746     {
747         // Print F1 logic
748     }
749
750     public void printROC()
751     {
752         // Print ROC logic
753     }
754
755     public void printAUC()
756     {
757         // Print AUC logic
758     }
759
760     public void printPrecision()
761     {
762         // Print precision logic
763     }
764
765     public void printRecall()
766     {
767         // Print recall logic
768     }
769
770     public void printF1()
771     {
772         // Print F1 logic
773     }
774
775     public void printROC()
776     {
777         // Print ROC logic
778     }
779
780     public void printAUC()
781     {
782         // Print AUC logic
783     }
784
785     public void printPrecision()
786     {
787         // Print precision logic
788     }
789
790     public void printRecall()
791     {
792         // Print recall logic
793     }
794
795     public void printF1()
796     {
797         // Print F1 logic
798     }
799
800     public void printROC()
801     {
802         // Print ROC logic
803     }
804
805     public void printAUC()
806     {
807         // Print AUC logic
808     }
809
810     public void printPrecision()
811     {
812         // Print precision logic
813     }
814
815     public void printRecall()
816     {
817         // Print recall logic
818     }
819
820     public void printF1()
821     {
822         // Print F1 logic
823     }
824
825     public void printROC()
826     {
827         // Print ROC logic
828     }
829
830     public void printAUC()
831     {
832         // Print AUC logic
833     }
834
835     public void printPrecision()
836     {
837         // Print precision logic
838     }
839
840     public void printRecall()
841     {
842         // Print recall logic
843     }
844
845     public void printF1()
846     {
847         // Print F1 logic
848     }
849
850     public void printROC()
851     {
852         // Print ROC logic
853     }
854
855     public void printAUC()
856     {
857         // Print AUC logic
858     }
859
860     public void printPrecision()
861     {
862         // Print precision logic
863     }
864
865     public void printRecall()
866     {
867         // Print recall logic
868     }
869
870     public void printF1()
871     {
872         // Print F1 logic
873     }
874
875     public void printROC()
876     {
877         // Print ROC logic
878     }
879
880     public void printAUC()
881     {
882         // Print AUC logic
883     }
884
885     public void printPrecision()
886     {
887         // Print precision logic
888     }
889
890     public void printRecall()
891     {
892         // Print recall logic
893     }
894
895     public void printF1()
896     {
897         // Print F1 logic
898     }
899
900     public void printROC()
901     {
902         // Print ROC logic
903     }
904
905     public void printAUC()
906     {
907         // Print AUC logic
908     }
909
910     public void printPrecision()
911     {
912         // Print precision logic
913     }
914
915     public void printRecall()
916     {
917         // Print recall logic
918     }
919
920     public void printF1()
921     {
922         // Print F1 logic
923     }
924
925     public void printROC()
926     {
927         // Print ROC logic
928     }
929
930     public void printAUC()
931     {
932         // Print AUC logic
933     }
934
935     public void printPrecision()
936     {
937         // Print precision logic
938     }
939
940     public void printRecall()
941     {
942         // Print recall logic
943     }
944
945     public void printF1()
946     {
947         // Print F1 logic
948     }
949
950     public void printROC()
951     {
952         // Print ROC logic
953     }
954
955     public void printAUC()
956     {
957         // Print AUC logic
958     }
959
960     public void printPrecision()
961     {
962         // Print precision logic
963     }
964
965     public void printRecall()
966     {
967         // Print recall logic
968     }
969
970     public void printF1()
971     {
972         // Print F1 logic
973     }
974
975     public void printROC()
976     {
977         // Print ROC logic
978     }
979
980     public void printAUC()
981     {
982         // Print AUC logic
983     }
984
985     public void printPrecision()
986     {
987         // Print precision logic
988     }
989
990     public void printRecall()
991     {
992         // Print recall logic
993     }
994
995     public void printF1()
996     {
997         // Print F1 logic
998     }
999
1000    public void printROC()
1001    {
1002        // Print ROC logic
1003    }
1004
1005    public void printAUC()
1006    {
1007        // Print AUC logic
1008    }
1009
1010    public void printPrecision()
1011    {
1012        // Print precision logic
1013    }
1014
1015    public void printRecall()
1016    {
1017        // Print recall logic
1018    }
1019
1020    public void printF1()
1021    {
1022        // Print F1 logic
1023    }
1024
1025    public void printROC()
1026    {
1027        // Print ROC logic
1028    }
1029
1030    public void printAUC()
1031    {
1032        // Print AUC logic
1033    }
1034
1035    public void printPrecision()
1036    {
1037        // Print precision logic
1038    }
1039
1040    public void printRecall()
1041    {
1042        // Print recall logic
1043    }
1044
1045    public void printF1()
1046    {
1047        // Print F1 logic
1048    }
1049
1050    public void printROC()
1051    {
1052        // Print ROC logic
1053    }
1054
1055    public void printAUC()
1056    {
1057        // Print AUC logic
1058    }
1059
1060    public void printPrecision()
1061    {
1062        // Print precision logic
1063    }
1064
1065    public void printRecall()
1066    {
1067        // Print recall logic
1068    }
1069
1070    public void printF1()
1071    {
1072        // Print F1 logic
1073    }
1074
1075    public void printROC()
1076    {
1077        // Print ROC logic
1078    }
1079
1080    public void printAUC()
1081    {
1082        // Print AUC logic
1083    }
1084
1085    public void printPrecision()
1086    {
1087        // Print precision logic
1088    }
1089
1090    public void printRecall()
1091    {
1092        // Print recall logic
1093    }
1094
1095    public void printF1()
1096    {
1097        // Print F1 logic
1098    }
1099
1100    public void printROC()
1101    {
1102        // Print ROC logic
1103    }
1104
1105    public void printAUC()
1106    {
1107        // Print AUC logic
1108    }
1109
1110    public void printPrecision()
1111    {
1112        // Print precision logic
1113    }
1114
1115    public void printRecall()
1116    {
1117        // Print recall logic
1118    }
1119
1120    public void printF1()
1121    {
1122        // Print F1 logic
1123    }
1124
1125    public void printROC()
1126    {
1127        // Print ROC logic
1128    }
1129
1130    public void printAUC()
1131    {
1132        // Print AUC logic
1133    }
1134
1135    public void printPrecision()
1136    {
1137        // Print precision logic
1138    }
1139
1140    public void printRecall()
1141    {
1142        // Print recall logic
1143    }
1144
1145    public void printF1()
1146    {
1147        // Print F1 logic
1148    }
1149
1150    public void printROC()
1151    {
1152        // Print ROC logic
1153    }
1154
1155    public void printAUC()
1156    {
1157        // Print AUC logic
1158    }
1159
1160    public void printPrecision()
1161    {
1162        // Print precision logic
1163    }
1164
1165    public void printRecall()
1166    {
1167        // Print recall logic
1168    }
1169
1170    public void printF1()
1171    {
1172        // Print F1 logic
1173    }
1174
1175    public void printROC()
1176    {
1177        // Print ROC logic
1178    }
1179
1180    public void printAUC()
1181    {
1182        // Print AUC logic
1183    }
1184
1185    public void printPrecision()
1186    {
1187        // Print precision logic
1188    }
1189
1190    public void printRecall()
1191    {
1192        // Print recall logic
1193    }
1194
1195    public void printF1()
1196    {
1197        // Print F1 logic
1198    }
1199
1200    public void printROC()
1201    {
1202        // Print ROC logic
1203    }
1204
1205    public void printAUC()
1206    {
1207        // Print AUC logic
1208    }
1209
1210    public void printPrecision()
1211    {
1212        // Print precision logic
1213    }
1214
1215    public void printRecall()
1216    {
1217        // Print recall logic
1218    }
1219
1220    public void printF1()
1221    {
1222        // Print F1 logic
1223    }
1224
1225    public void printROC()
1226    {
1227        // Print ROC logic
1228    }
1229
1230    public void printAUC()
1231    {
1232        // Print AUC logic
1233    }
1234
1235    public void printPrecision()
1236    {
1237        // Print precision logic
1238    }
1239
1240    public void printRecall()
1241    {
1242        // Print recall logic
1243    }
1244
1245    public void printF1()
1246    {
1247        // Print F1 logic
1248    }
1249
1250    public void printROC()
1251    {
1252        // Print ROC logic
1253    }
1254
1255    public void printAUC()
1256    {
1257        // Print AUC logic
1258    }
1259
1260    public void printPrecision()
1261    {
1262        // Print precision logic
1263    }
1264
1265    public void printRecall()
1266    {
1267        // Print recall logic
1268    }
1269
1270    public void printF1()
1271    {
1272        // Print F1 logic
1273    }
1274
1275    public void printROC()
1276    {
1277        // Print ROC logic
1278    }
1279
1280    public void printAUC()
1281    {
1282        // Print AUC logic
1283    }
1284
1285    public void printPrecision()
1286    {
1287        // Print precision logic
1288    }
1289
1290    public void printRecall()
1291    {
1292        // Print recall logic
1293    }
1294
1295    public void printF1()
1296    {
1297        // Print F1 logic
1298    }
1299
1300    public void printROC()
1301    {
1302        // Print ROC logic
1303    }
1304
1305    public void printAUC()
1306    {
1307        // Print AUC logic
1308    }
1309
1310    public void printPrecision()
1311    {
1312        // Print precision logic
1313    }
1314
1315    public void printRecall()
1316    {
1317        // Print recall logic
1318    }
1319
1320    public void printF1()
1321    {
1322        // Print F1 logic
1323    }
1324
1325    public void printROC()
1326    {
1327        // Print ROC logic
1328    }
1329
1330    public void printAUC()
1331    {
1332        // Print AUC logic
1333    }
1334
1335    public void printPrecision()
1336    {
1337        // Print precision logic
1338    }
1339
1340    public void printRecall()
1341    {
1342        // Print recall logic
1343    }
1344
1345    public void printF1()
1346    {
1347        // Print F1 logic
1348    }
1349
1350    public void printROC()
1351    {
1352        // Print ROC logic
1353    }
1354
1355    public void printAUC()
1356    {
1357        // Print AUC logic
1358    }
1359
1360    public void printPrecision()
1361    {
1362        // Print precision logic
1363    }
1364
1365    public void printRecall()
1366    {
1367        // Print recall logic
1368    }
1369
1370    public void printF1()
1371    {
1372        // Print F1 logic
1373    }
1374
1375    public void printROC()
1376    {
1377        // Print ROC logic
1378    }
1379
1380    public void printAUC()
1381    {
1382        // Print AUC logic
1383    }
1384
1385    public void printPrecision()
1386    {
1387        // Print precision logic
1388    }
1389
1390    public void printRecall()
1391    {
1392        // Print recall logic
1393    }
1394
1395    public void printF1()
1396    {
1397        // Print F1 logic
1398    }
1399
1400    public void printROC()
1401    {
1402        // Print ROC logic
1403    }
1404
1405    public void printAUC()
1406    {
1407        // Print AUC logic
1408    }
1409
1410    public void printPrecision()
1411    {
1412        // Print precision logic
1413    }
1414
1415    public void printRecall()
1416    {
1417        // Print recall logic
1418    }
1419
1420    public void printF1()
1421    {
1422        // Print F1 logic
1423    }
1424
1425    public void printROC()
1426    {
1427        // Print ROC logic
1428    }
1429
1430    public void printAUC()
1431    {
1432        // Print AUC logic
1433    }
1434
1435    public void printPrecision()
1436    {
1437        // Print precision logic
1438    }
1439
1440    public void printRecall()
1441    {
1442        // Print recall logic
1443    }
1444
1445    public void printF1()
1446    {
1447        // Print F1 logic
1448    }
1449
1450    public void printROC()
1451    {
1452        // Print ROC logic
1453    }
1454
1455    public void printAUC()
1456    {
1457        // Print AUC logic
1458    }
1459
1460    public void printPrecision()
1461    {
1462        // Print precision logic
1463    }
1464
1465    public void printRecall()
1466    {
1467        // Print recall logic
1468    }
1469
1470    public void printF1()
1471    {
1472        // Print F1 logic
1473    }
1474
1475    public void printROC()
1476    {
1477        // Print ROC logic
1478    }
1479
1480    public void printAUC()
1481    {
1482        // Print AUC logic
1483    }
1484
1485    public void printPrecision()
1486    {
1487        // Print precision logic
1488    }
1489
1490    public void printRecall()
1491    {
1492        // Print recall logic
1493    }
1494
1495    public void printF1()
1496    {
1497        // Print F1 logic
1498    }
1499
1500    public void printROC()
1501    {
1502        // Print ROC logic
1503    }
1504
1505    public void printAUC()
1506    {
1507        // Print AUC logic
1508    }
1509
1510    public void printPrecision()
1511    {
1512        // Print precision logic
1513    }
1514
1515    public void printRecall()
1516    {
1517        // Print recall logic
1518    }
1519
1520    public void printF1()
1521    {
1522        // Print F1 logic
1523    }
1524
1525    public void printROC()
1526    {
1527        // Print ROC logic
1528    }
1529
1530    public void printAUC()
1531    {
1532        // Print AUC logic
1533    }
1534
1535    public void printPrecision()
1536    {
1537        // Print precision logic
1538    }
1539
1540    public void printRecall()
1541    {
1542        // Print recall logic
1543    }
1544
1545    public void printF1()
1546    {
1547        // Print F1 logic
1548    }
1549
1550    public void printROC()
1551    {
1552        // Print ROC logic
1553    }
1554
1555    public void printAUC()
1556    {
1557        // Print AUC logic
1558    }
1559
1560    public void printPrecision()
1561    {
1562        // Print precision logic
1563    }
1564
1565    public void printRecall()
1566    {
1567        // Print recall logic
1568    }
1569
1570    public void printF1()
1571    {
1572        // Print F1 logic
1573    }
1574
1575    public void printROC()
1576    {
1577        // Print ROC logic
1578    }
1579
1580    public void printAUC()
1581    {
1582        // Print AUC logic
1583    }
1584
1585    public void printPrecision()
1586    {
1587        // Print precision logic
1588    }
1589
1590    public void printRecall()
1591    {
1592        // Print recall logic
1593    }
1594
1595    public void printF1()
1596    {
1597        // Print F1 logic
1598    }
1599
1600    public void printROC()

```

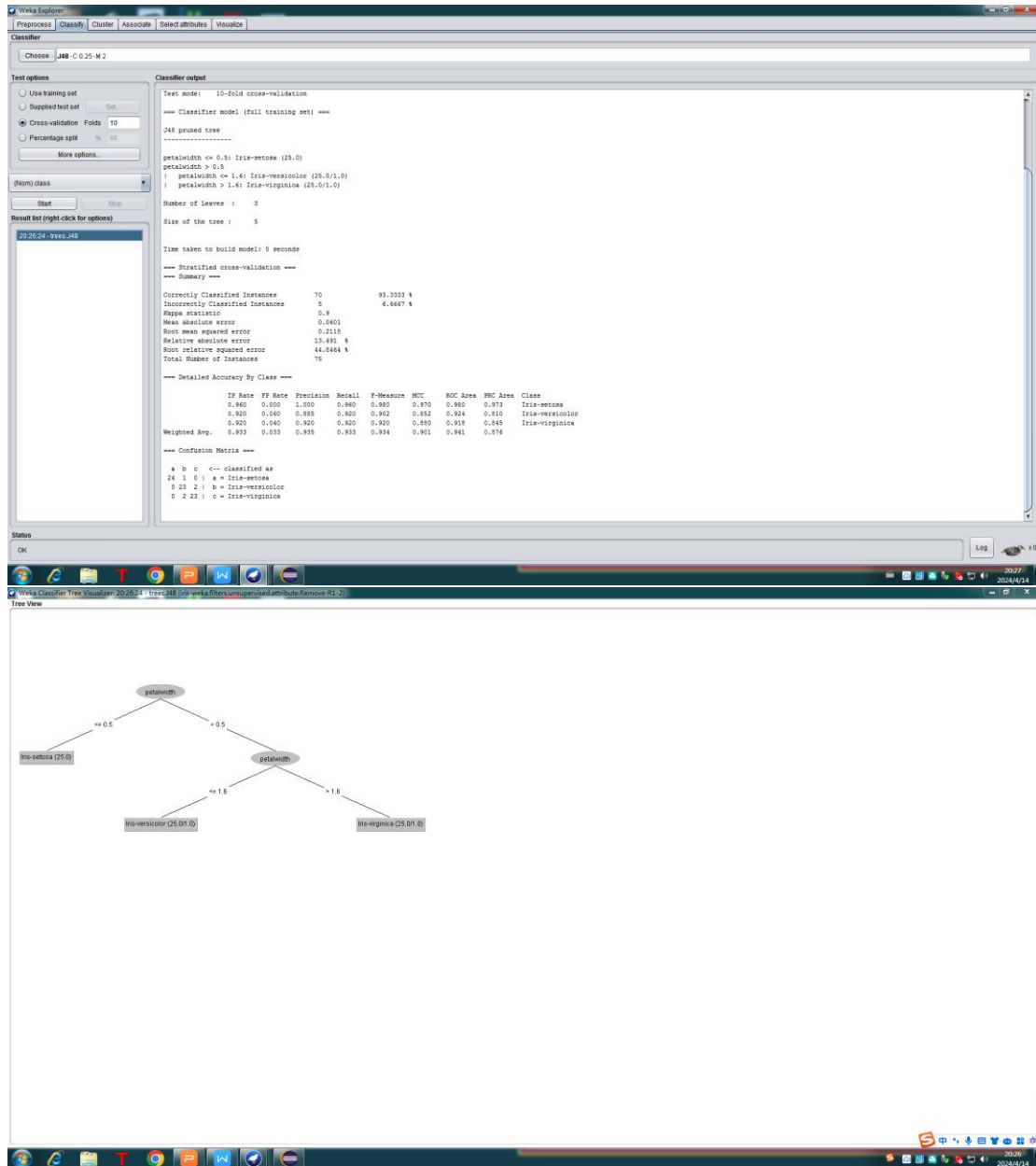


(2) BP 网络





4、使用 WEKA 图形界面实现决策树



十、实验结论：

(一)

使用 weka 和通过 Eclipse 调用 weka.jar 可以得到相同的数据处理结果。

(二)

Apriori 算法可以对离散数据进行频繁项集产生和规则产生

(三)

KNN 算法简单，快速，准确率较高，通过 weka 选择相应算法可以以树的形式直观呈现出决策树。

十一、总结及心得体会：

(一)

通过使用 weka 以及调用 weka.jar 可以方便的对数据进行处理，得出归一化、缺失值处理、特征筛选的数据预处理结果。

(二)

使用 python 语言可以更快捷方便地对数据进行 Apriori 算法的频繁项集产生和规则产生

(三)

由于 KNN 算法属于 lazy learning，不会对训练数据集做深入学习，而是单纯的考虑数据之间的距离，所以 KNN 的预测效果强依赖于训练数据。

十二、对本实验过程及方法、手段的改进建议：

(一)

尝试使用 C++ 语言对数据进行归一化、缺失值处理、特征筛选等操作。

(二)

用 Java 语言，同样可以实现 Apriori 算法，但是可能代码量更多。

(三)

使用更多训练数据后观察 KNN 算法的预测准确度变化。

报告评分：

指导教师签字：