

MapReduce - Phân loại nhị phân Naive Bayes

Jan 20, 2023

Lập trình MapReduce cho bài toán phân loại hoa Iris sử dụng thuật toán Naive Bayes

Bước 1: Tải file dữ liệu loài hoa Iris

Dữ liệu được lấy tại [Kaggle](#), tạo file **iris.csv** với nội dung như sau:

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
4.3,3.0,1.1,0.1,Iris-setosa
5.8,4.0,1.2,0.2,Iris-setosa
5.7,4.4,1.5,0.4,Iris-setosa
5.4,3.9,1.3,0.4,Iris-setosa
5.1,3.5,1.4,0.3,Iris-setosa
5.7,3.8,1.7,0.3,Iris-setosa
5.1,3.8,1.5,0.3,Iris-setosa
```

5.4,3.4,1.7,0.2,Iris-setosa
5.1,3.7,1.5,0.4,Iris-setosa
4.6,3.6,1.0,0.2,Iris-setosa
5.1,3.3,1.7,0.5,Iris-setosa
4.8,3.4,1.9,0.2,Iris-setosa
5.0,3.0,1.6,0.2,Iris-setosa
5.0,3.4,1.6,0.4,Iris-setosa
5.2,3.5,1.5,0.2,Iris-setosa
5.2,3.4,1.4,0.2,Iris-setosa
4.7,3.2,1.6,0.2,Iris-setosa
4.8,3.1,1.6,0.2,Iris-setosa
5.4,3.4,1.5,0.4,Iris-setosa
5.2,4.1,1.5,0.1,Iris-setosa
5.5,4.2,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.0,3.2,1.2,0.2,Iris-setosa
5.5,3.5,1.3,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
4.4,3.0,1.3,0.2,Iris-setosa
5.1,3.4,1.5,0.2,Iris-setosa
5.0,3.5,1.3,0.3,Iris-setosa
4.5,2.3,1.3,0.3,Iris-setosa
4.4,3.2,1.3,0.2,Iris-setosa
5.0,3.5,1.6,0.6,Iris-setosa
5.1,3.8,1.9,0.4,Iris-setosa
4.8,3.0,1.4,0.3,Iris-setosa
5.1,3.8,1.6,0.2,Iris-setosa
4.6,3.2,1.4,0.2,Iris-setosa
5.3,3.7,1.5,0.2,Iris-setosa
5.0,3.3,1.4,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
5.5,2.3,4.0,1.3,Iris-versicolor
6.5,2.8,4.6,1.5,Iris-versicolor
5.7,2.8,4.5,1.3,Iris-versicolor
6.3,3.3,4.7,1.6,Iris-versicolor
4.9,2.4,3.3,1.0,Iris-versicolor
6.6,2.9,4.6,1.3,Iris-versicolor
5.2,2.7,3.9,1.4,Iris-versicolor
5.0,2.0,3.5,1.0,Iris-versicolor
5.9,3.0,4.2,1.5,Iris-versicolor
6.0,2.2,4.0,1.0,Iris-versicolor
6.1,2.9,4.7,1.4,Iris-versicolor
5.6,2.9,3.6,1.3,Iris-versicolor
6.7,3.1,4.4,1.4,Iris-versicolor

5.6,3.0,4.5,1.5,Iris-versicolor
5.8,2.7,4.1,1.0,Iris-versicolor
6.2,2.2,4.5,1.5,Iris-versicolor
5.6,2.5,3.9,1.1,Iris-versicolor
5.9,3.2,4.8,1.8,Iris-versicolor
6.1,2.8,4.0,1.3,Iris-versicolor
6.3,2.5,4.9,1.5,Iris-versicolor
6.1,2.8,4.7,1.2,Iris-versicolor
6.4,2.9,4.3,1.3,Iris-versicolor
6.6,3.0,4.4,1.4,Iris-versicolor
6.8,2.8,4.8,1.4,Iris-versicolor
6.7,3.0,5.0,1.7,Iris-versicolor
6.0,2.9,4.5,1.5,Iris-versicolor
5.7,2.6,3.5,1.0,Iris-versicolor
5.5,2.4,3.8,1.1,Iris-versicolor
5.5,2.4,3.7,1.0,Iris-versicolor
5.8,2.7,3.9,1.2,Iris-versicolor
6.0,2.7,5.1,1.6,Iris-versicolor
5.4,3.0,4.5,1.5,Iris-versicolor
6.0,3.4,4.5,1.6,Iris-versicolor
6.7,3.1,4.7,1.5,Iris-versicolor
6.3,2.3,4.4,1.3,Iris-versicolor
5.6,3.0,4.1,1.3,Iris-versicolor
5.5,2.5,4.0,1.3,Iris-versicolor
5.5,2.6,4.4,1.2,Iris-versicolor
6.1,3.0,4.6,1.4,Iris-versicolor
5.8,2.6,4.0,1.2,Iris-versicolor
5.0,2.3,3.3,1.0,Iris-versicolor
5.6,2.7,4.2,1.3,Iris-versicolor
5.7,3.0,4.2,1.2,Iris-versicolor
5.7,2.9,4.2,1.3,Iris-versicolor
6.2,2.9,4.3,1.3,Iris-versicolor
5.1,2.5,3.0,1.1,Iris-versicolor
5.7,2.8,4.1,1.3,Iris-versicolor
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
7.1,3.0,5.9,2.1,Iris-virginica
6.3,2.9,5.6,1.8,Iris-virginica
6.5,3.0,5.8,2.2,Iris-virginica
7.6,3.0,6.6,2.1,Iris-virginica
4.9,2.5,4.5,1.7,Iris-virginica
7.3,2.9,6.3,1.8,Iris-virginica
6.7,2.5,5.8,1.8,Iris-virginica
7.2,3.6,6.1,2.5,Iris-virginica
6.5,3.2,5.1,2.0,Iris-virginica
6.4,2.7,5.3,1.9,Iris-virginica

6.8,3.0,5.5,2.1,Iris-virginica
5.7,2.5,5.0,2.0,Iris-virginica
5.8,2.8,5.1,2.4,Iris-virginica
6.4,3.2,5.3,2.3,Iris-virginica
6.5,3.0,5.5,1.8,Iris-virginica
7.7,3.8,6.7,2.2,Iris-virginica
7.7,2.6,6.9,2.3,Iris-virginica
6.0,2.2,5.0,1.5,Iris-virginica
6.9,3.2,5.7,2.3,Iris-virginica
5.6,2.8,4.9,2.0,Iris-virginica
7.7,2.8,6.7,2.0,Iris-virginica
6.3,2.7,4.9,1.8,Iris-virginica
6.7,3.3,5.7,2.1,Iris-virginica
7.2,3.2,6.0,1.8,Iris-virginica
6.2,2.8,4.8,1.8,Iris-virginica
6.1,3.0,4.9,1.8,Iris-virginica
6.4,2.8,5.6,2.1,Iris-virginica
7.2,3.0,5.8,1.6,Iris-virginica
7.4,2.8,6.1,1.9,Iris-virginica
7.9,3.8,6.4,2.0,Iris-virginica
6.4,2.8,5.6,2.2,Iris-virginica
6.3,2.8,5.1,1.5,Iris-virginica
6.1,2.6,5.6,1.4,Iris-virginica
7.7,3.0,6.1,2.3,Iris-virginica
6.3,3.4,5.6,2.4,Iris-virginica
6.4,3.1,5.5,1.8,Iris-virginica
6.0,3.0,4.8,1.8,Iris-virginica
6.9,3.1,5.4,2.1,Iris-virginica
6.7,3.1,5.6,2.4,Iris-virginica
6.9,3.1,5.1,2.3,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
6.8,3.2,5.9,2.3,Iris-virginica
6.7,3.3,5.7,2.5,Iris-virginica
6.7,3.0,5.2,2.3,Iris-virginica
6.3,2.5,5.0,1.9,Iris-virginica
6.5,3.0,5.2,2.0,Iris-virginica
6.2,3.4,5.4,2.3,Iris-virginica
5.9,3.0,5.1,1.8,Iris-virginica

Bước 2: Tạo thư mục đầu vào trong hdfs

```
hdfs dfs -mkdir /iris-input
```

Bước 3: Đẩy file **iris.csv** vào folder **iris-input** vừa tạo:

```
hadoop fs -put C:\naive-bayes\iris.csv /iris-input
```

*NOTE: Thay **C:\naive-bayes\iris.csv** bằng đường dẫn thư mục lưu file*

Bước 4: Tạo project

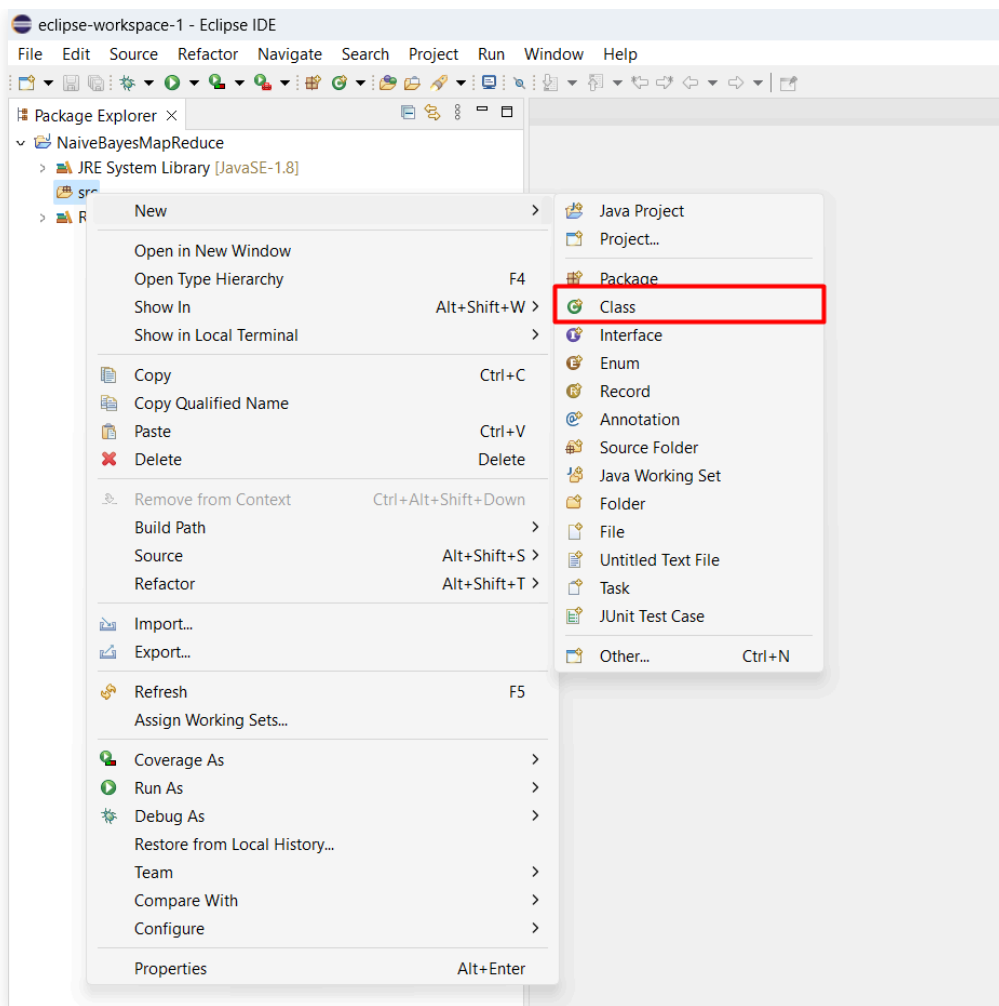
Các bước thực hiện tương tự [MapReduce - Phân cụm Kmeans](#), đặt tên project là **NaiveBayesMapReduce**

Bước 5: Thêm thư viện cần thiết để chạy MapReduce

Các bước thực hiện tương tự [MapReduce - Phân cụm Kmeans](#)

Bước 6: Tạo các class xử lý nhiệm vụ train/test mô hình Naive Bayes

Double click vào project **KmeanMapReduce**, chuột phải vào **src** và chọn **New > Class**



Tạo class để xử lý nhiệm vụ **Map** quá trình train, đặt tên là **NaiveBayesTrainMapper**

New Java Class

Java Class

⚠ The use of the default package is discouraged.

Source folder: NaiveBayesMapReduce/src Browse...

Package: (default) Browse...

☐ Enclosing type: Browse...

Name: NaiveBayesTrainMapper

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add... Remove

Which method stubs would you like to create?
☐ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

Finish Cancel

Nội dung bên trong file **NaiveBayesTrainMapper.java**:

```
import java.io.IOException;
import java.util.HashSet;
import java.util.StringTokenizer;

import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;

public class NaiveBayesTrainMapper extends MapReduceBase implements Mapper<LongWritable,
    String, Text> {

    String delimiter, continuousVariables;
    int targetVariable, numColumns;
    HashSet<Integer> continuousVariablesIndex;

    public HashSet<Integer> splitvariables(String varString){
```

```

        HashSet<Integer> hs = new HashSet<Integer>();
        StringTokenizer tok = new StringTokenizer(varString, ",");
        while(tok.hasMoreElements())
            hs.add(Integer.parseInt(tok.nextToken()));
        return hs;
    }

    @Override
    public void configure(JobConf conf){
        delimiter = conf.get("delimiter");
        numColumns = conf.getInt("numColumns", 0);
        continuousVariables = conf.get("continuousVariables");
        targetVariable = conf.getInt("targetVariable", 0);
        continuousVariablesIndex = new HashSet<Integer>();
        if(continuousVariables!=null)
            continuousVariablesIndex = splitvariables(continuousVariables);
    }

    @Override
    public void map(LongWritable arg0, Text value,
                    OutputCollector<Text, DoubleWritable> output, Reporter arg3)
        throws IOException {
        // TODO Auto-generated method stub
        Integer varIndex = 1;
        String record = value.toString();
        String features[] = record.split(delimiter);
        for(int i = 0 ; i < numColumns ; i++){
            if(varIndex!= targetVariable){
                if(continuousVariablesIndex.contains(varIndex))
                    output.collect(new Text(varIndex+"_"+features[targetVariable-1]));
            }
            varIndex ++;
        }
        output.collect(new Text(targetVariable+"_"+features[targetVariable-1]), new
        output.collect(new Text(targetVariable+""), new DoubleWritable(1.0));
    }
}

```

Tương tự tạo class xử lý nhiệm vụ **Reducer** quá trình train, đặt tên là **NaiveBayesTrainReducer**:

```

import java.io.IOException;
import java.util.HashSet;
import java.util.Iterator;
import java.util.StringTokenizer;

```



```

import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;

public class NaiveBayesTrainReducer extends MapReduceBase implements Reducer<Text, DoubleWritable> {

    String continuousVariables;
    int targetVariable;
    HashSet<Integer> continuousVariablesIndex;

    public HashSet<Integer> splitvariables(String varString){
        HashSet<Integer> hs = new HashSet<Integer>();
        StringTokenizer tok = new StringTokenizer(varString, ",");
        while(tok.hasMoreElements())
            hs.add(Integer.parseInt(tok.nextToken()));
        return hs;
    }

    @Override
    public void configure(JobConf conf){
        continuousVariables = conf.get("continuousVariables");
        targetVariable = conf.getInt("targetVariable", 0);
        continuousVariablesIndex = new HashSet<Integer>();
        if(continuousVariables!=null)
            continuousVariablesIndex = splitvariables(continuousVariables);
    }

    @Override
    public void reduce(Text keyId, Iterator<DoubleWritable> values,
                      OutputCollector<NullWritable, Text> output, Reporter arg3) throws IOException {

        String id = keyId.toString().split("_")[0];
        if(continuousVariablesIndex.contains(Integer.parseInt(id))){
            double sumsqr=0, sum = 0, count=0, tmp;
            double mean, var;
            while (values.hasNext())
            {
                tmp=values.next().get();
                sumsqr+=tmp*tmp;
                sum += tmp;
                count++;
            }
        }
    }
}

```

```

        }

        mean=sum/count;
        var=(sumsq-(sum*sum)/count)/count;
        output.collect(NullWritable.get(), new Text(keyId+" "+mean+", "+var));
    }
    if(targetVariable == Integer.parseInt(id)){
        Double count = 0.0;
        while (values.hasNext())
            count += values.next().get();
        output.collect(NullWritable.get(), new Text(keyId+" "+count.toString()));
    }
}
}

```

Tạo class main xử lý yêu cầu đầu vào quá trình train, đặt tên là **NaiveBayesTrainJob**:

```

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class NaiveBayesTrainJob extends Configured implements Tool{

    @Override
    public int run(String[] arg0) throws Exception {
        Configuration c = getConf();

        // TODO Auto-generated method stub
        JobConf conf = new JobConf(c, NaiveBayesTrainJob.class);
        conf.setJobName("Training");
        conf.setMapOutputKeyClass(Text.class);
        conf.setMapOutputValueClass(DoubleWritable.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(Text.class);
        conf.setNumMapTasks(conf.getInt("numMaps", 4));
        conf.setNumReduceTasks(conf.getInt("numReduce", 1));
    }
}

```

```

        conf.setMapperClass(NaiveBayesTrainMapper.class);
        conf.setReducerClass(NaiveBayesTrainReducer.class);
        FileInputFormat.addInputPath(conf, new Path(conf.get("input")));
        FileOutputFormat.setOutputPath(conf, new Path(conf.get("output")));
        JobClient.runJob(conf);
        return 0;
    }

    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new NaiveBayesTrainJob(),
            System.exit(res));
    }
}

```

Tạo class xử lý nhiệm vụ **Map** quá trình test, đặt tên là **NaiveBayesTestMapper**:

```

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.net.URI;
import java.nio.file.FileSystem;
import java.nio.file.Files;
import java.util.HashMap;
import java.util.HashSet;
import java.util.StringTokenizer;
import java.io.BufferedReader;
import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
import org.apache.hadoop.metrics2.sink.FileSink;
import org.apache.hadoop.fs.Path;

public class NaiveBayesTestMapper extends MapReduceBase implements Mapper<LongWritable,
    String delimiter,continuousVariables,targetClasses;
    int targetVariable,numCols;
    HashSet<Integer> continuousVariablesIndex;
    HashMap<String,String> hm;
    HashSet<String> classesTargetVariables;

```

```

public HashSet<Integer> splitvariables(String varString){
    HashSet<Integer> hs = new HashSet<Integer>();
    StringTokenizer tok = new StringTokenizer(varString, ",");
    while(tok.hasMoreElements())
        hs.add(Integer.parseInt(tok.nextToken()));
    return hs;
}

public HashSet<String> splitstringvariables(String varString){
    HashSet<String> hs = new HashSet<String>();
    StringTokenizer tok = new StringTokenizer(varString, ",");
    while(tok.hasMoreElements())
        hs.add(tok.nextToken());
    return hs;
}

@Override
public void configure(JobConf conf){
    delimiter = conf.get("delimiter");
    numCols = conf.getInt("numColumns", 0);
    continuousVariables = conf.get("continuousVariables");
    targetClasses = conf.get("targetClasses");
    targetVariable = conf.getInt("targetVariable", 0);
    continuousVariablesIndex = new HashSet<Integer>();
    if(continuousVariables!=null)
        continuousVariablesIndex = splitvariables(continuousVariables);
    classesTargetVariables = splitstringvariables(targetClasses);

    hm = new HashMap();
    try {
        URI[] filesIncache = DistributedCache.getCacheFiles(conf);
        for(int i=0;i<filesIncache.length;i++){
            BufferedReader fis = new BufferedReader(new FileReader(-
            String record;
            while ((record = fis.readLine()) != null) {
                String key,value;
                StringTokenizer tokRecord = new StringTokenizer
                key = tokRecord.nextToken();
                value = tokRecord.nextToken();
                hm.put(key, value);
            }
        }
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

```

```

double calculateProbability(int featureID,String value,String label){
    String classCount,valueCount,totalCount;
    classCount = hm.get(targetVariable+"_"+label);
    if(classCount==null)
        return 1.0;
    valueCount = hm.get(featureID+"_"+value+"_"+label);
    if(valueCount==null)
        return 1.0;
    totalCount = hm.get(targetVariable+"");
    double classProbability = (Double.parseDouble(classCount) / Double.parseDouble(totalCount));
    double valueProbability = (Double.parseDouble(valueCount) / Double.parseDouble(totalCount));
    return (classProbability*valueProbability);
}

```

```

double calculateGaussian(int featureID,String value,String label){
    Double mean,variance,val;
    val = Double.parseDouble(value);
    String values = hm.get(featureID+"_"+label);
    if(values!=null){
        StringTokenizer tokMeanVariance = new StringTokenizer(values," ");
        mean = Double.parseDouble(tokMeanVariance.nextToken());
        variance = Double.parseDouble(tokMeanVariance.nextToken());
        if(variance==0.0)
            return 1.0;
        double exponent,denominator;
        denominator = Math.sqrt(2*3.14)*variance;
        exponent = -1*(Math.pow((val-mean),2))/(2*Math.pow(variance, 2));
        return (1/denominator)*Math.exp(exponent);
    }
    return 1.0;
}

```

```

@Override
public void map(LongWritable key, Text value,
                OutputCollector<NullWritable, Text> output, Reporter arg3) throws IOException {
    // TODO Auto-generated method stub
    String record = value.toString();
    int featureID = 1;
    Double probability=0.0;
    Double labelProbability[] = new Double[classesTargetVariables.size()];
    String features[] = record.split(delimiter);
    int labelIndex = 0;
    String labelprobabilityString="";
    for (String labels : classesTargetVariables){
        probability=1.0;
        featureID = 1;
    }
}

```



```

        FileOutputFormat.setOutputPath(conf, new Path(conf.get("output")));
        JobClient.runJob(conf);

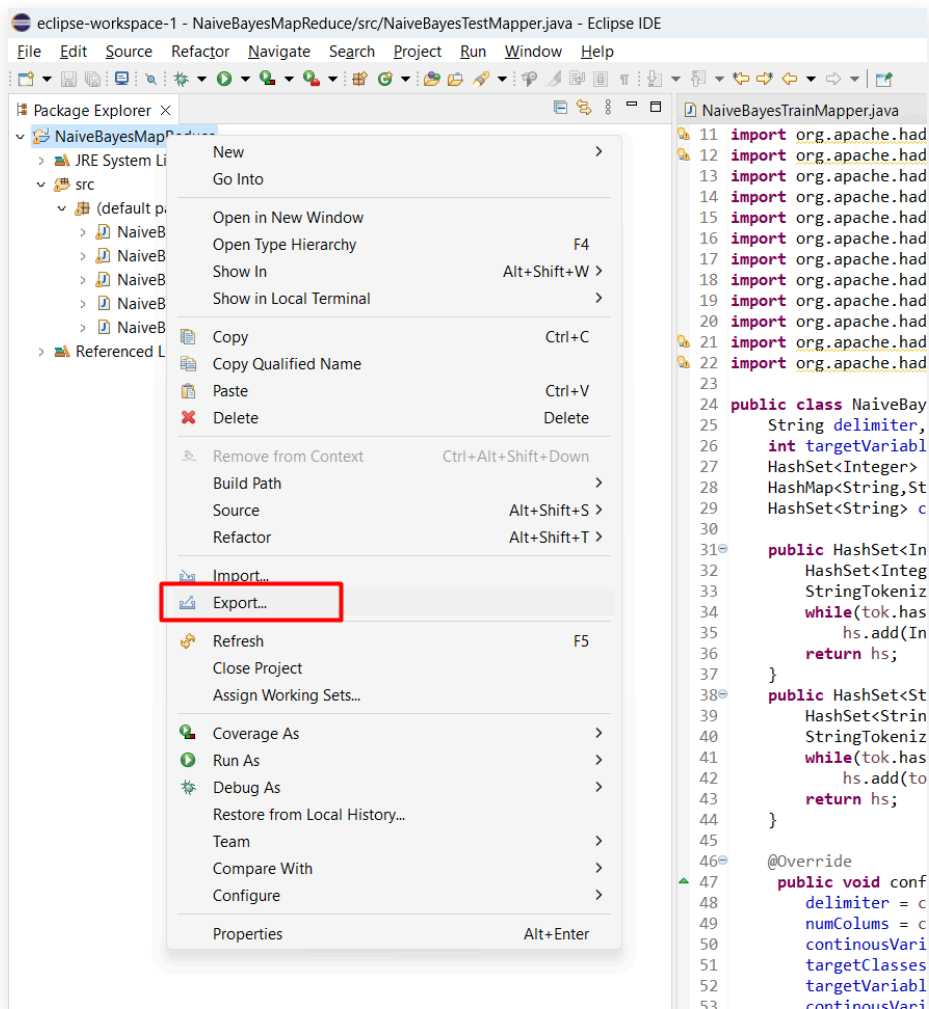
        return 0;
    }

    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new NaiveBayesTestJob(), args);
        System.exit(res);
    }
}

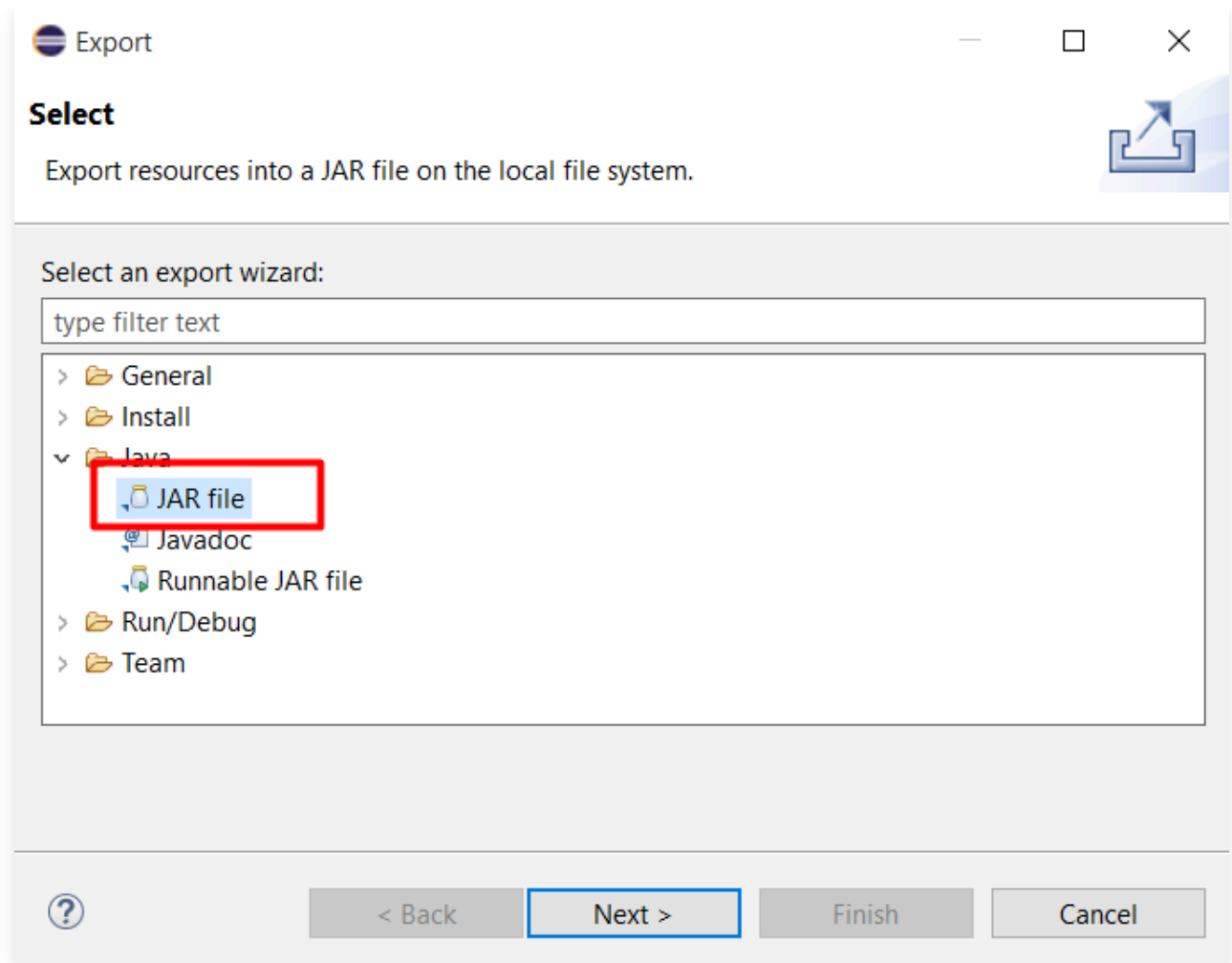
```

Bước 7: Tạo file JAR

Chuột phải vào project **NaiveBayesMapReduce** chọn **Export**



Chọn **Java > JAR File** rồi bấm **Finish**



Vào thư mục chứa lưu file JAR vừa tạo và kiểm tra kết quả

Bước 8: Chạy chương trình Training/Test bộ dữ liệu **iris.csv**

Training bộ dữ liệu **iris.csv** đã tạo ở trên, và kết quả thu được lưu trong thư mục **outputiris**. Chạy lệnh sau:

```
hadoop jar C:\jar\NaiveBayes.jar NaiveBayesTrainJob -D num_mappers="3" -D num_reducers="3"
```

Lưu ý: Thay "C:\jar\NaiveBayes.jar" bằng đường dẫn chứa file JAR ở trên máy

Test dữ liệu chạy lệnh:

```
hadoop jar C:\jar\NaiveBayes.jar NaiveBayesTestJob -D num_mappers="3" -D num_reducers="3"
```

Lưu ý: Để có kết quả trực quan hãy tạo dữ liệu đầu vào khác và thay đổi đường dẫn lưu file dữ liệu test ở tham số input

