**LAPORAN TUGAS KECIL II IF3170 INTELEGENSI BUATAN**

# Eksplorasi Waikato Environment for Knowledge Analysis (WEKA)


**Disusun oleh:**

**13514039 – Ari Pratama Zhorifiandi**

**13514086 – Azka Hanif Imtiyaz**

**TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

# DAFTAR ISI

# BAB I. DESKRIPSI PERSOALAN DAN TOOLS

## A. DESKRIPSI PERSOALAN

Program yang dibuat pada tugas ini adalah sebuah program yang mengimplementasikan API dari WEKA. Program tersebut:

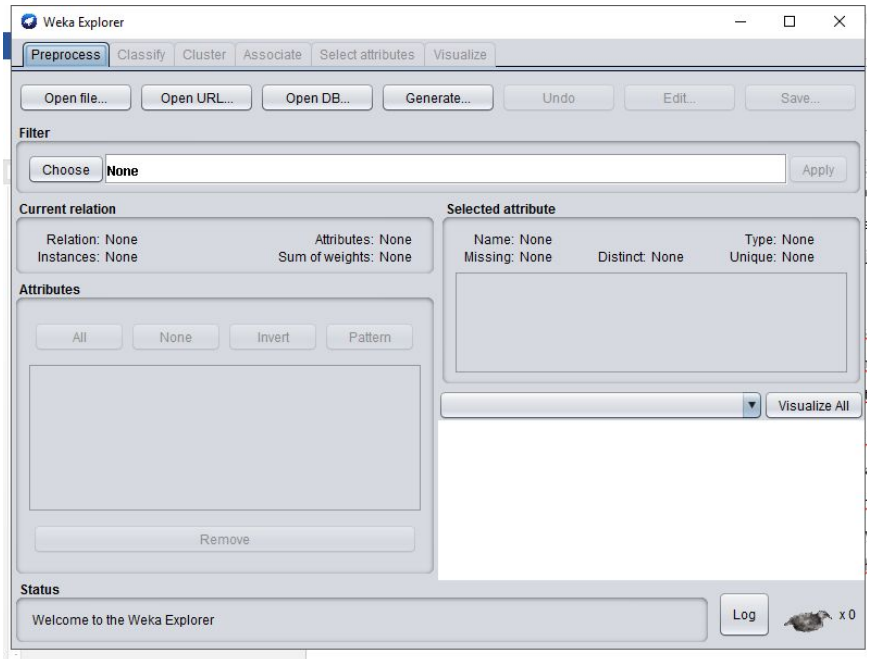
1. Diimplementasikan dengan Bahasa Java
2. Dapat membaca dataset yang diberikan
3. Dapat mengaplikasikan filter yang mengubah tipe atribut, misalnya Discretize atau NumericToNominal.
4. Dapat melakukan pembelajaran dataset dengan skema 10-fold cross validation
5. Dapat melakukan pembelajaran dataset dengan skema full-training
6. Dapat menyimpan (save) model/hipotesis hasil pembelajaran ke sebuah file eksternal
7. Dapat membaca (read) model/hipotesis dari file eksternal
8. Dapat membuat instance baru sesuai masukan dari pengguna untuk setiap nilai atribut
9. Dapat melakukan klasifikasi dengan memanfaatkan model/hipotesis dan instance sesuai masukan pengguna pada poin 8.
10. Mengimplementasikan kelas dengan menggunakan iris.arff

## B. TOOLS YANG DIGUNAKAN

Weka adalah aplikasi yang menyimpan koleksi algoritma machine learning untuk pengerjaan data mining. Algoritmanya dapat di terapkan langsung pada dataset atau digunakan dari program Java buatan sendiri. Weka memiliki tools untuk data pre-processing, classification, regression, clustering, association rules, dan visualization. Weka juga sangat cocok untuk membangun skema machine learning yang baru.

Semua teknik Weka didasarkan pada asumsi bahwa data tersedia sebagai satu file atau relasi yang flat, di mana setiap titik data digambarkan oleh sejumlah atribut yang tetap (biasanya, atribut numerik atau nominal, tetapi beberapa jenis atribut lainnya juga didukung) .

Weka juga menyediakan akses ke database SQL menggunakan Java Database Connectivity dan dapat memproses hasil yang dikembalikan oleh query database. Weka tidak mampu melakukan multi-relasional data mining, tetapi ada perangkat lunak terpisah untuk mengkonversi koleksi tabel database dihubungkan ke satu tabel yang cocok untuk memproses menggunakan Weka. Bidang lain yang saat ini tidak tercakup oleh algoritma yang terdapat dalam Weka adalah sequence modeling.




*Gambar 1 Aplikasi Desktop Weka pada Windows*

# BAB II. IMPLEMENTASI WEKA API PADA JAVA

## A. SOURCE CODE

```
/*
 * Made by Ari Pratama and Azka Hanif
 */
package weka;

import java.io.BufferedReader;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.ObjectOutputStream;
import java.util.Random;
import java.util.Scanner;
import weka.classifiers.Classifier;

import weka.core.Instances;
import weka.classifiers.Evaluation;
import weka.classifiers.bayes.NaiveBayes;
import weka.core.DenseInstance;
import weka.filters.Filter;
import weka.filters.supervised.attribute.*;

public class WEKA {

   void saveModel(Classifier C, String namaFile) throws Exception {
      //SAVE
       // serialize model
           ObjectOutputStream oos = new ObjectOutputStream(
                              new FileOutputStream(namaFile));
           oos.writeObject(C);
           oos.flush();
           oos.close();
   }

   public static void main(String[] args) throws Exception {
      // IMPORT file *.arff
      WEKA w = new WEKA();

      //Pilihan SKEMA
      boolean validasi = false;
      do {
         //Read iris.arff
         BufferedReader breader = null;
         breader = new BufferedReader(new FileReader("src\\weka\\iris.arff"));
         Instances inputTrain = new Instances (breader);
         inputTrain.setClassIndex(inputTrain.numAttributes() -1);
```

```
        breader.close();

        //FILTER
        Discretize filter = new Discretize();
        filter.setInputFormat(inputTrain);
        Instances outputTrain = Filter.useFilter(inputTrain,filter);
        Evaluation eval = new Evaluation(outputTrain);

        //ALGORITMA YANG DIGUNAKAN
        NaiveBayes nB = new NaiveBayes();

        //Menu
        Scanner scan = new Scanner(System.in);
        System.out.println("\n\n=================\n==== OPTION ====");
        System.out.println("1. Full Training Scheme");
        System.out.println("2. 10 Fold Validation Scheme");
        System.out.println("3. Load");
        System.out.println("4. Create new instance");
        System.out.println("5. Exit");
        System.out.print("Enter your option (1/2/3/4) : ");
        int pilihan = scan.nextInt();

        switch (pilihan) {
           case 1:
              {
                 nB.buildClassifier(outputTrain);
                 eval.evaluateModel(nB,outputTrain);
                 //OUTPUT
                 System.out.println(eval.toSummaryString("=== Stratified cross-validation ===\n" +"=== Summary
===",true));

                 System.out.println(eval.toClassDetailsString("=== Detailed Accuracy By Class ==="));

                 System.out.println(eval.toMatrixString("===Confusion matrix==="));

                 System.out.println(eval.fMeasure(1)+" "+eval.recall(1));

                 System.out.println("\nDo you want to save this model(1/0)? ");

                 int c = scan.nextInt();
                 if (c == 1 ){
                    System.out.print("Please enter your file name (*.model) : ");

                    String infile = scan.next();
                    w.saveModel(nB,infile);
                 }
                 else {
                    System.out.print("Model not saved.");
                 }     break;
```

```
            }
          case 2:
            {
               nB.buildClassifier(outputTrain);
               eval.crossValidateModel(nB, outputTrain, 10, new Random(1));

               //OUTPUT
               System.out.println(eval.toSummaryString("=== Stratified cross-validation ===\n" +"=== Summary
===",true));

               System.out.println(eval.toClassDetailsString("=== Detailed Accuracy By Class ==="));

               System.out.println(eval.toMatrixString("===Confusion matrix==="));

               System.out.println(eval.fMeasure(1)+" "+eval.recall(1));

               System.out.println("\nDo you want to save this model(1/0)? ");

               int c = scan.nextInt();
               if (c == 1 ){
                  System.out.print("Please enter your file name (*.model) : ");

                  String infile = scan.next();
                  w.saveModel(nB,infile);
               }
               else {
                  System.out.print("Model not saved.");
               }     break;
            }
          case 3:
            //LOAD
            // deserialize model
            System.out.print("Please enter the file name : ");
            String namaFile = scan.next();
            Classifier cls = (Classifier) weka.core.SerializationHelper.read(namaFile);

            eval.crossValidateModel(cls, outputTrain, 10, new Random(1));

            System.out.println(eval.toSummaryString("=== Stratified cross-validation ===\n" +"=== Summary
===",true));

            System.out.println(eval.toClassDetailsString("=== Detailed Accuracy By Class ==="));

            System.out.println(eval.toMatrixString("===Confusion matrix==="));

            System.out.println(eval.fMeasure(1)+" "+eval.recall(1));
            break;
          case 4:
            System.out.println();
```

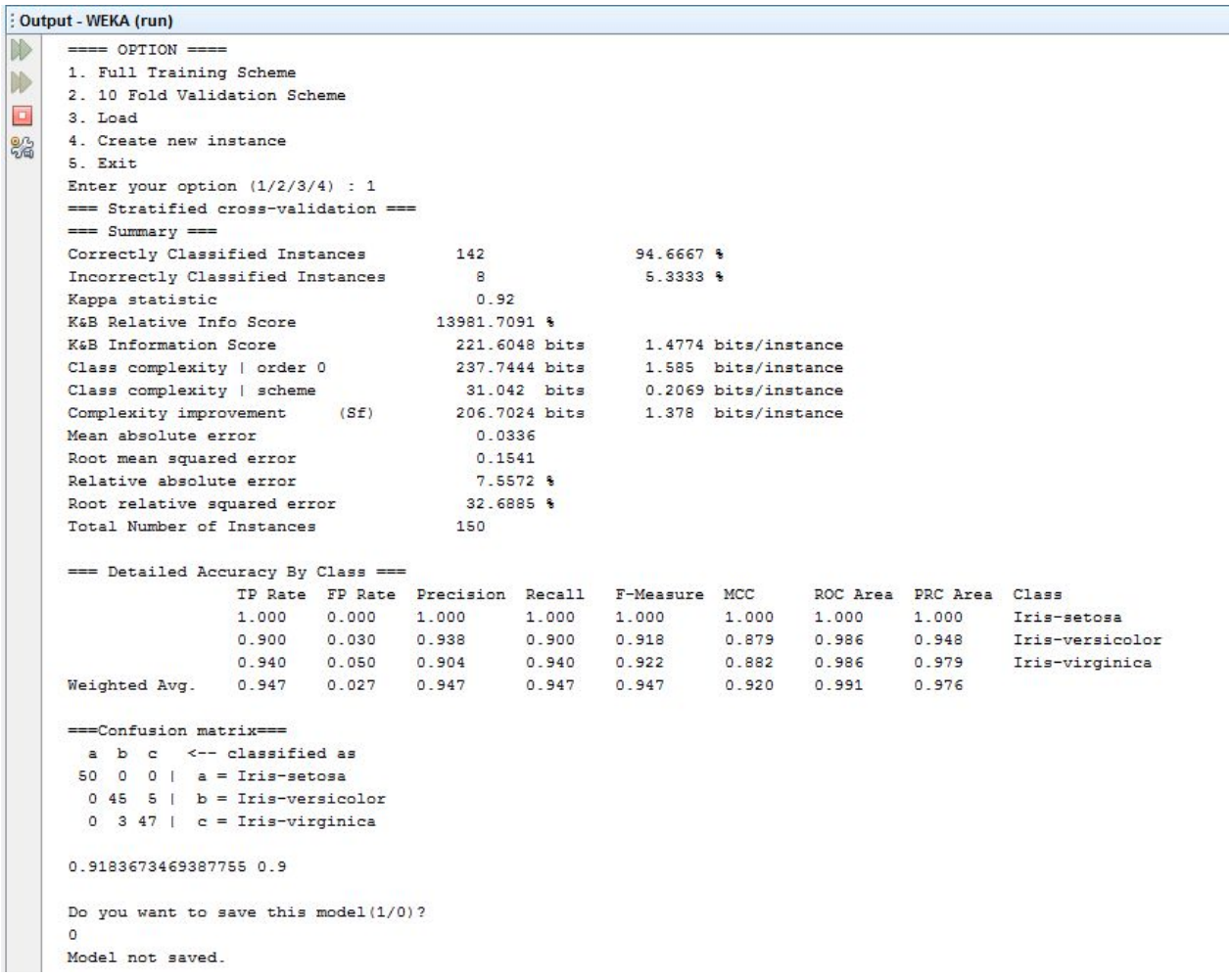
```
            //ADD New Instance
            nB.buildClassifier(inputTrain);
            //Copy attributes from instances
            DenseInstance buffer = new DenseInstance(inputTrain.firstInstance());

            //Initialization
            buffer.setDataset(inputTrain);
            buffer.setMissing(inputTrain.classIndex());
            //Input
            for (int i = 0; i < inputTrain.classIndex(); i++){
              System.out.print("Enter the value for " + buffer.attribute(i).name() + ": ");

              double val = scan.nextDouble();
              buffer.setValue(i, val);
            }
            //Classify
            double res = nB.classifyInstance(buffer);
            buffer.setValue(inputTrain.classIndex(), res);
            inputTrain.add(buffer);
            System.out.println("Class: " + buffer.stringValue(inputTrain.classIndex()));

            break;
          case 5:
            validasi = true;
            break;
          default:
            System.out.println("Wrong input!");
            break;
        }
      }
      while (!validasi);
    }
}
```

***full source code: https://github.com/zhorifiandi/WEKAstarter***

# B. HASIL IMPLEMENTASI

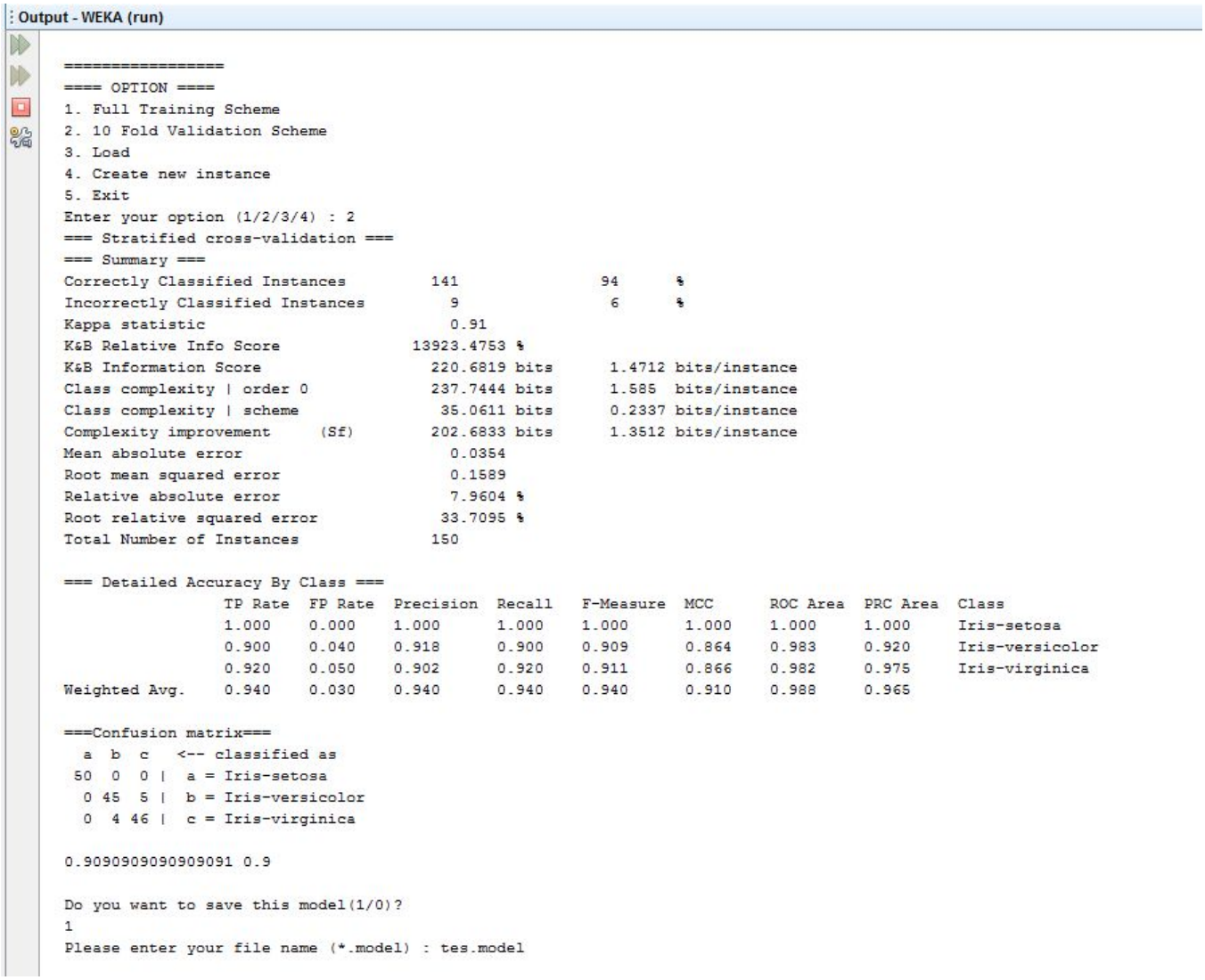
Untuk pengujian hasil implementasi, dataset yang digunakan adalah iris.arff

## Pembelajaran dengan Skema Full Training



```
Output - WEKA (run)
==== OPTION ====
1. Full Training Scheme
2. 10 Fold Validation Scheme
3. Load
4. Create new instance
5. Exit
Enter your option (1/2/3/4) : 1
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances         142               94.6667 %
Incorrectly Classified Instances         8                5.3333 %
Kappa statistic                          0.92
K&B Relative Info Score              13981.7091 %
K&B Information Score                  221.6048 bits      1.4774 bits/instance
Class complexity | order 0             237.7444 bits      1.585  bits/instance
Class complexity | scheme               31.042  bits      0.2069 bits/instance
Complexity improvement     (Sf)        206.7024 bits      1.378  bits/instance
Mean absolute error                      0.0336
Root mean squared error                  0.1541
Relative absolute error                  7.5572 %
Root relative squared error             32.6885 %
Total Number of Instances              150

=== Detailed Accuracy By Class ===
                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     Iris-setosa
                 0.900    0.030    0.938      0.900    0.918      0.879    0.986     0.948     Iris-versicolor
                 0.940    0.050    0.904      0.940    0.922      0.882    0.986     0.979     Iris-virginica
Weighted Avg.    0.947    0.027    0.947      0.947    0.947      0.920    0.991     0.976

===Confusion matrix===
  a  b  c   <-- classified as
 50  0  0 |  a = Iris-setosa
  0 45  5 |  b = Iris-versicolor
  0  3 47 |  c = Iris-virginica

0.9183673469387755 0.9

Do you want to save this model(1/0)?
0
Model not saved.
```

## Pembelajaran dengan Skema 10 Fold Validation

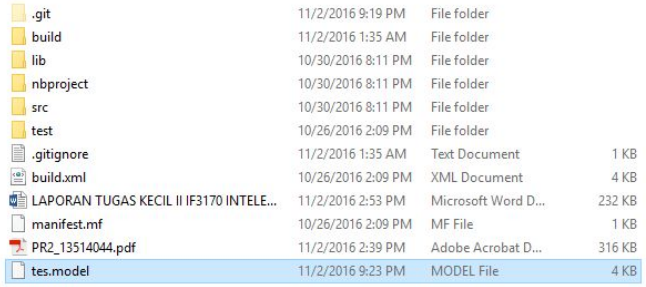


Output - WEKA (run)

```
=================
==== OPTION ====
1. Full Training Scheme
2. 10 Fold Validation Scheme
3. Load
4. Create new instance
5. Exit
Enter your option (1/2/3/4) : 2
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances         141               94      %
Incorrectly Classified Instances         9                6      %
Kappa statistic                          0.91
K&B Relative Info Score              13923.4753 %
K&B Information Score                  220.6819 bits      1.4712 bits/instance
Class complexity | order 0             237.7444 bits      1.585  bits/instance
Class complexity | scheme               35.0611 bits      0.2337 bits/instance
Complexity improvement     (Sf)        202.6833 bits      1.3512 bits/instance
Mean absolute error                      0.0354
Root mean squared error                  0.1589
Relative absolute error                  7.9604 %
Root relative squared error             33.7095 %
Total Number of Instances              150

=== Detailed Accuracy By Class ===
                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     Iris-setosa
                 0.900    0.040    0.918      0.900    0.909      0.864    0.983     0.920     Iris-versicolor
                 0.920    0.050    0.902      0.920    0.911      0.866    0.982     0.975     Iris-virginica
Weighted Avg.    0.940    0.030    0.940      0.940    0.940      0.910    0.988     0.965

===Confusion matrix===
  a  b  c   <-- classified as
 50  0  0 |  a = Iris-setosa
  0 45  5 |  b = Iris-versicolor
  0  4 46 |  c = Iris-virginica

0.9090909090909091 0.9

Do you want to save this model(1/0)?
1
Please enter your file name (*.model) : tes.model
```

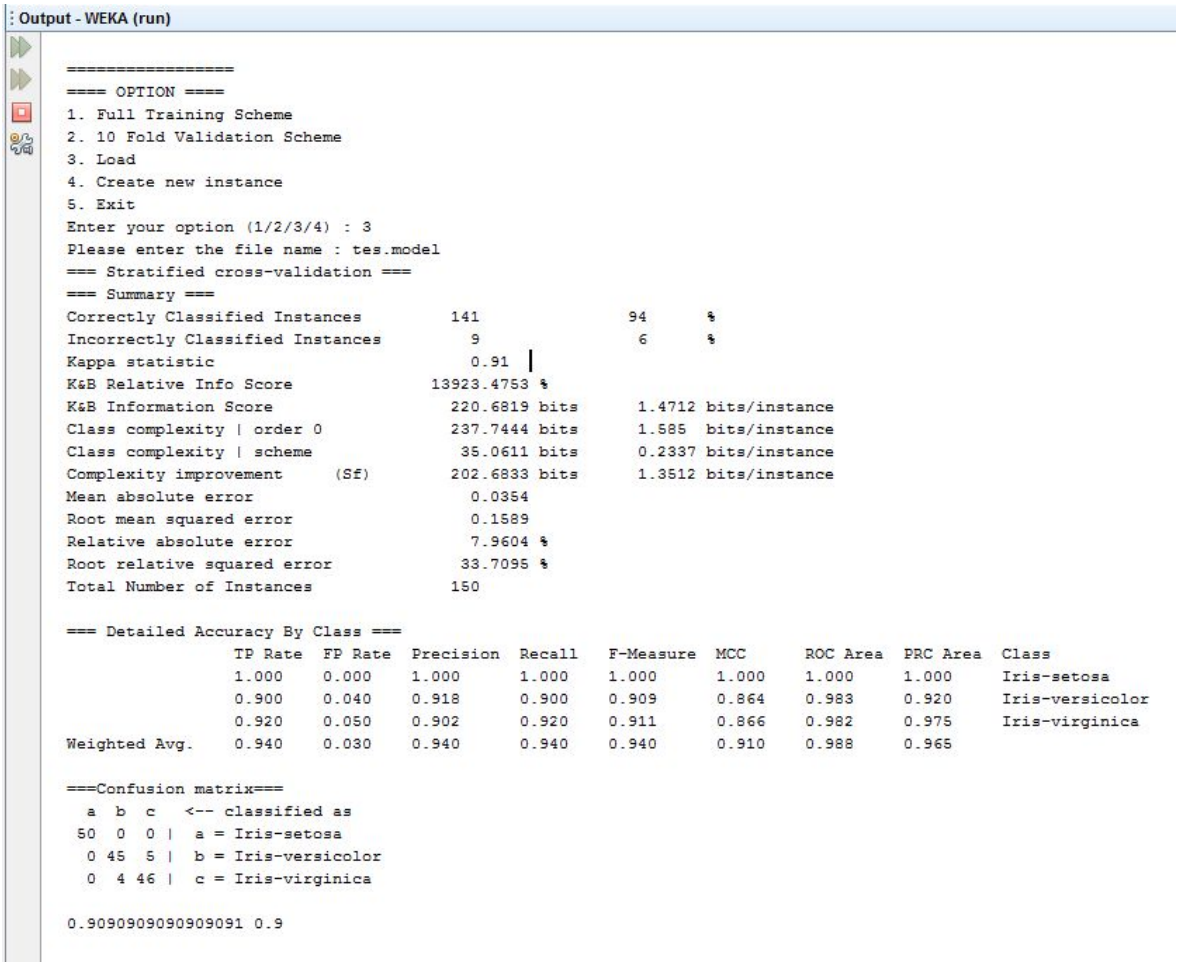
## Menyimpan Model pada File Eksternal

```
Do you want to save this model(1/0)?
1
Please enter your file name (*.model) : tes.model
```



| | | | |
|---|---|---|---|
| .git | 11/2/2016 9:19 PM | File folder | |
| build | 11/2/2016 1:35 AM | File folder | |
| lib | 10/30/2016 8:11 PM | File folder | |
| nbproject | 10/30/2016 8:11 PM | File folder | |
| src | 10/30/2016 8:11 PM | File folder | |
| test | 10/26/2016 2:09 PM | File folder | |
| .gitignore | 11/2/2016 1:35 AM | Text Document | 1 KB |
| build.xml | 10/26/2016 2:09 PM | XML Document | 4 KB |
| LAPORAN TUGAS KECIL II IF3170 INTELE... | 11/2/2016 2:53 PM | Microsoft Word D... | 232 KB |
| manifest.mf | 10/26/2016 2:09 PM | MF File | 1 KB |
| PR2_13514044.pdf | 11/2/2016 2:39 PM | Adobe Acrobat D... | 316 KB |
| tes.model | 11/2/2016 9:23 PM | MODEL File | 4 KB |

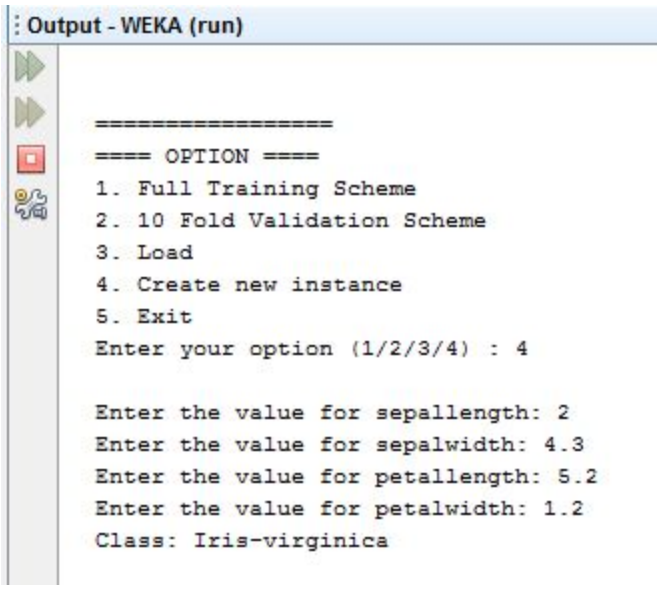
## Membaca Model dari File Eksternal



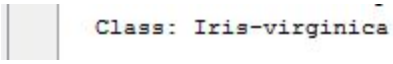
Output - WEKA (run)

```
==================
==== OPTION ====
1. Full Training Scheme
2. 10 Fold Validation Scheme
3. Load
4. Create new instance
5. Exit
Enter your option (1/2/3/4) : 3
Please enter the file name : tes.model
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances         141               94      %
Incorrectly Classified Instances         9                6      %
Kappa statistic                          0.91
K&B Relative Info Score              13923.4753 %
K&B Information Score                  220.6819 bits      1.4712 bits/instance
Class complexity | order 0             237.7444 bits      1.585  bits/instance
Class complexity | scheme               35.0611 bits      0.2337 bits/instance
Complexity improvement     (Sf)        202.6833 bits      1.3512 bits/instance
Mean absolute error                      0.0354
Root mean squared error                  0.1589
Relative absolute error                  7.9604 %
Root relative squared error             33.7095 %
Total Number of Instances              150

=== Detailed Accuracy By Class ===
                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     Iris-setosa
                 0.900    0.040    0.918      0.900    0.909      0.864    0.983     0.920     Iris-versicolor
                 0.920    0.050    0.902      0.920    0.911      0.866    0.982     0.975     Iris-virginica
Weighted Avg.    0.940    0.030    0.940      0.940    0.940      0.910    0.988     0.965

===Confusion matrix===
  a  b  c   <-- classified as
 50  0  0 |  a = Iris-setosa
  0 45  5 |  b = Iris-versicolor
  0  4 46 |  c = Iris-virginica

0.9090909090909091 0.9
```

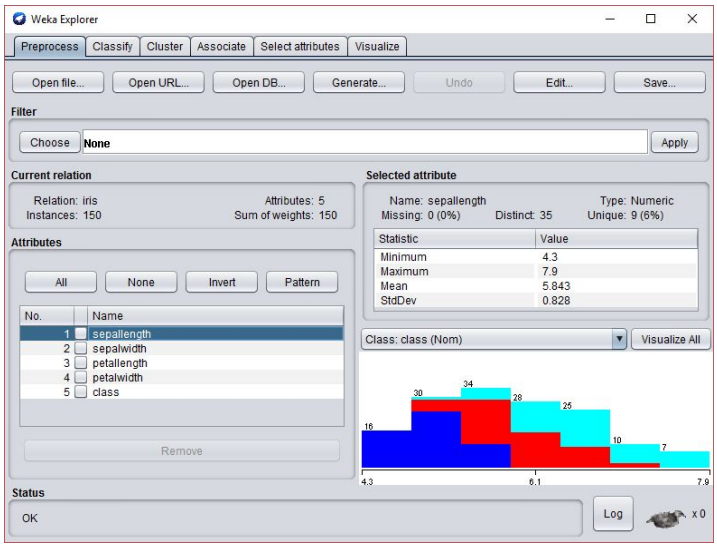
## Membuat Instance Baru sesuai Masukan dari Pengguna




## Klasifikasi Instance Baru

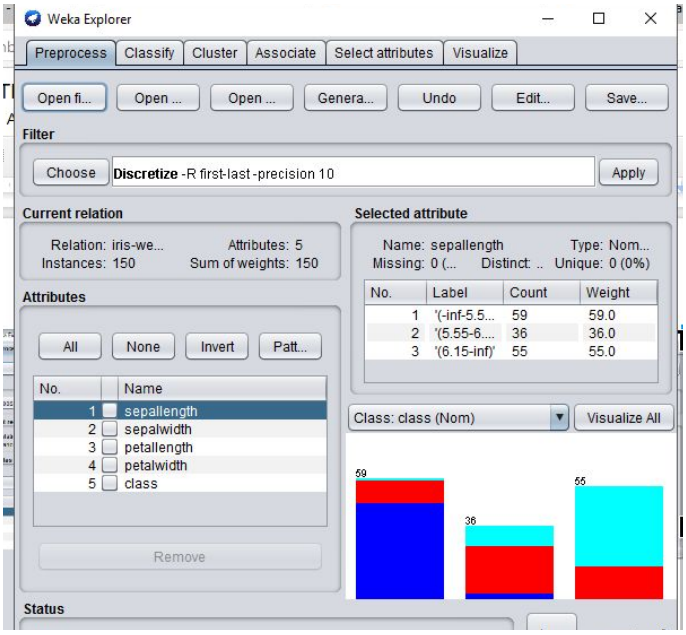


# BAB III. ANALISIS



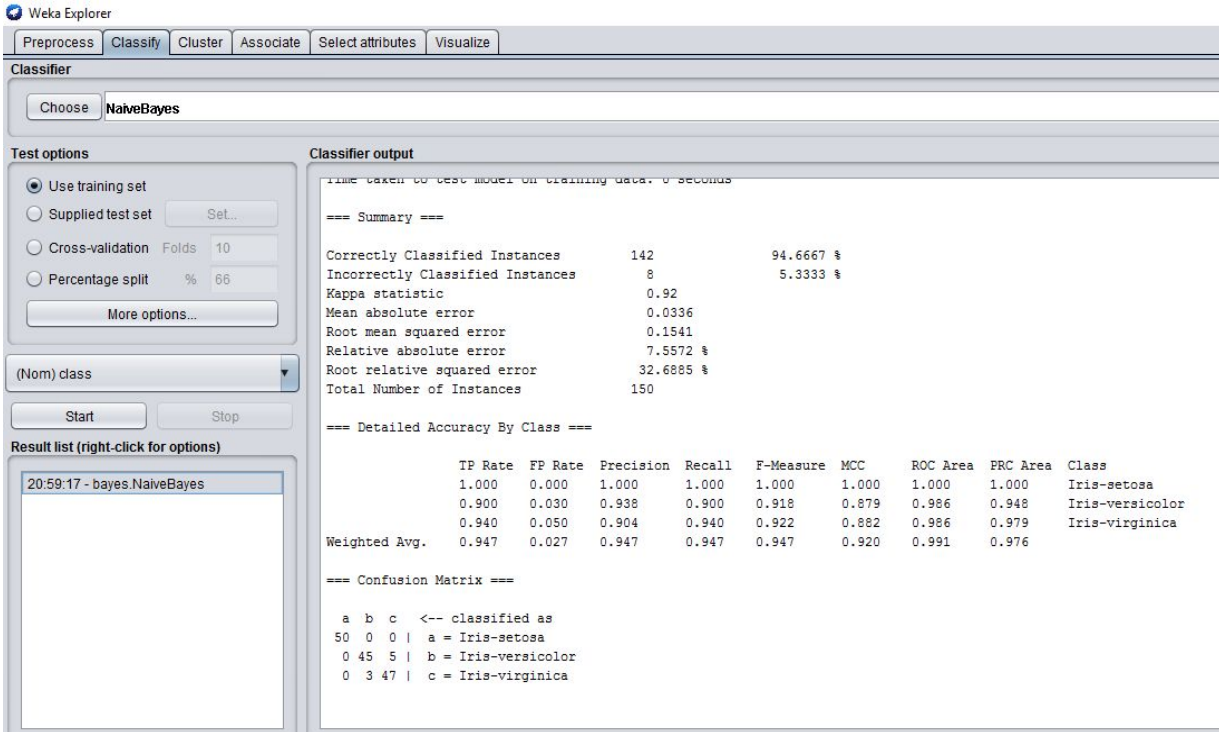

*Gambar 2. iris.arff yang baru dibuka pada GUI Weka*

Untuk perbandingan, kita akan memberikan filter discretize pada dataset iris. Algoritma classifier yang akan digunakan adalah Naive Bayes, sama seperti implementasi program yang telah kami buat.

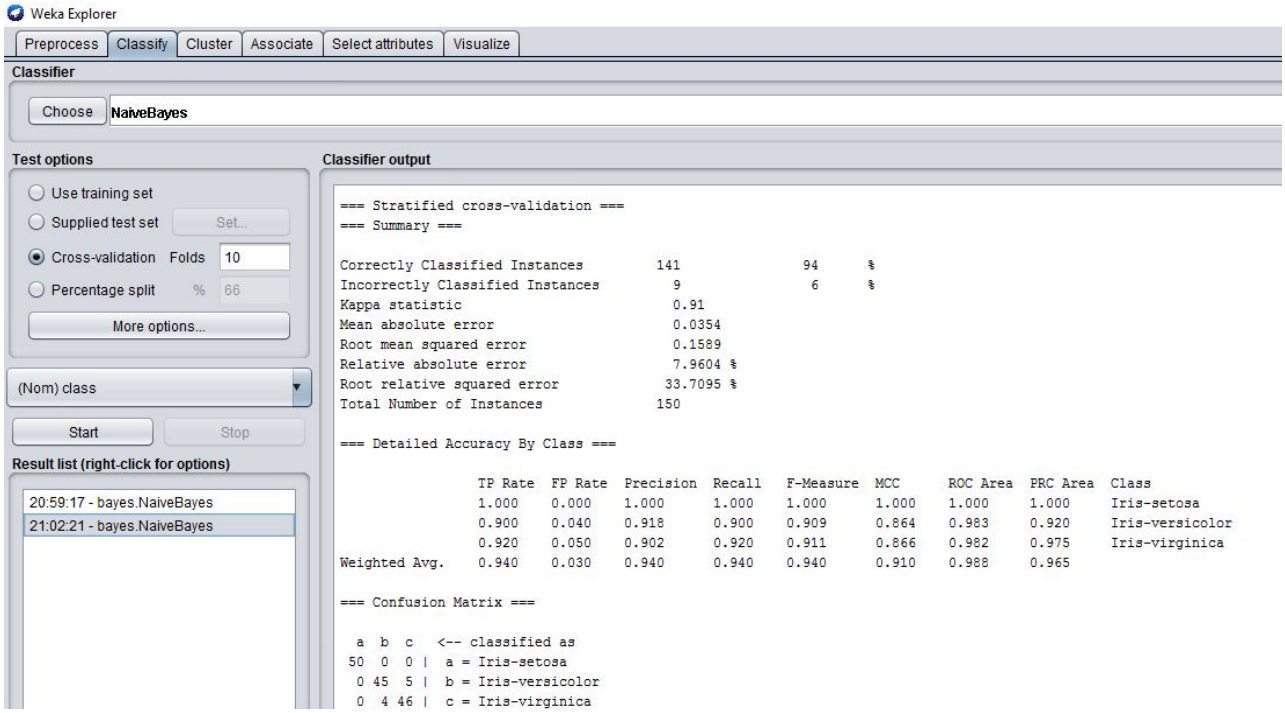



*Gambar 3. Dataset iris yang sudah di filter*

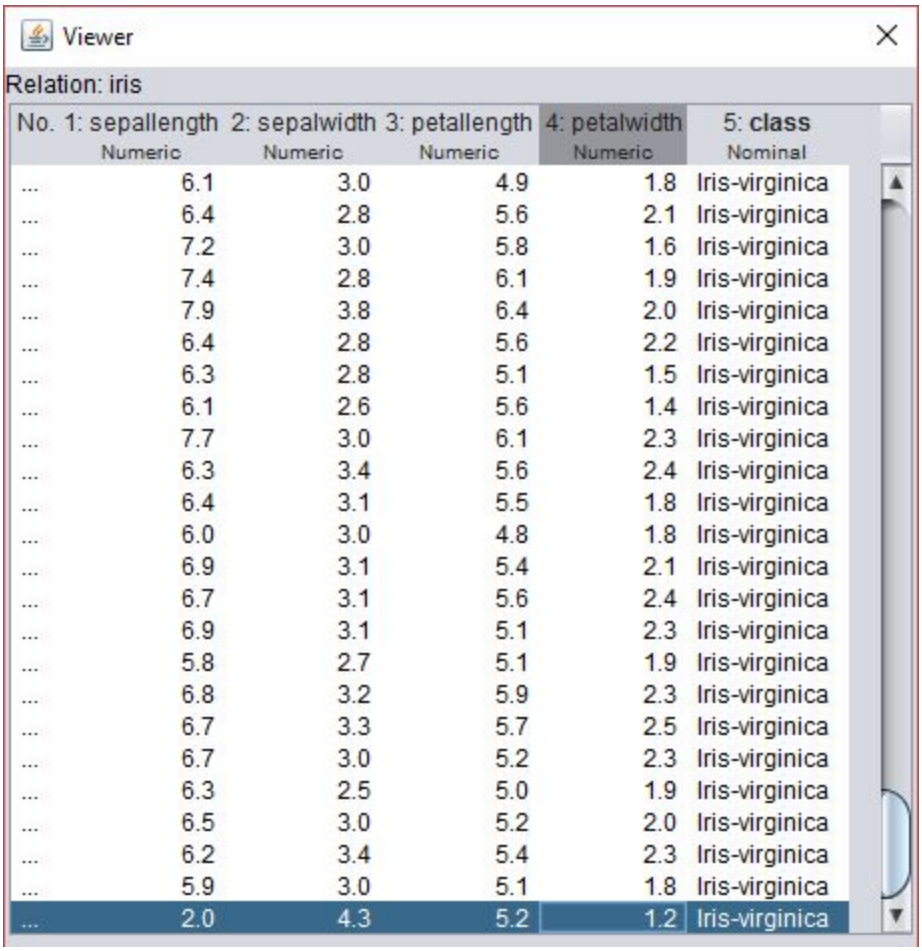
## Pembelajaran dengan Skema Full-Training




## Pembelajaran dengan Skema 10 Fold Validation



## Penambahan Instance Baru tanpa Kelas



# BAB IV. KESIMPULAN

- Seperti yang dapat dilihat pada hasil implementasi dan analisis pada GUI, hasil implementasi Program Java kami dengan Program Weka GUI adalah sama.
- Implementasi API ini dapat digunakan untuk pembelajaran dataset pada aplikasi yang dibuat secara langsung tanpa perlu melakukan pembelajaran manual menggunakan GUI, sehingga mempermudah programmer nantinya.