**LAPORAN TUGAS BESAR**

**IF3170 INTELIGENSI BUATAN**

****

**Disusun oleh :**

**Kelompok Ailuvyou**

**Muhammad Farhan Kemal 13513085**

**Naufal Malik Rabbani 13514052**

**Amal Qurany 13514078**

**Muhammad Azzahid A. S. 13514095**

**I Dewa Putu Deny K. A. 13514096**

**Program Studi Teknik Informatika**

**Sekolah Teknik Elektro dan Informatika**

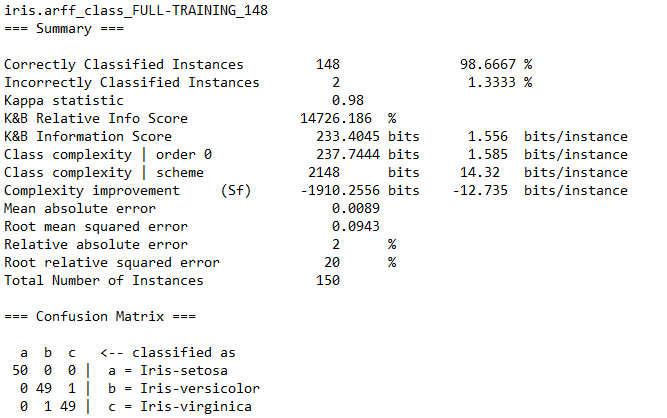
**Institut Teknologi Bandung**

**2016**

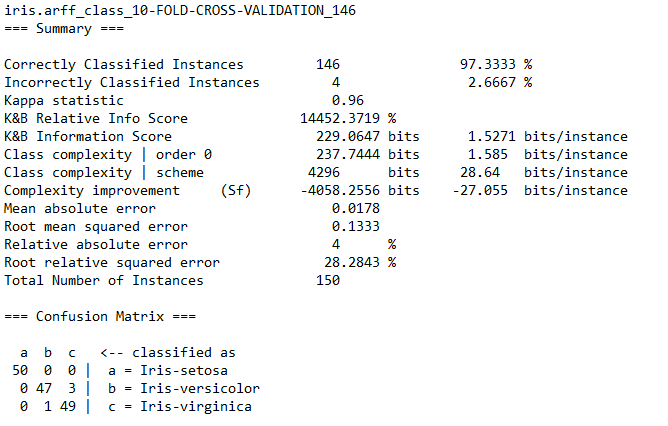
1. **Hasil Pengujian**
   1. *Feed Forward Neural Network*

Nama file : iris.arff

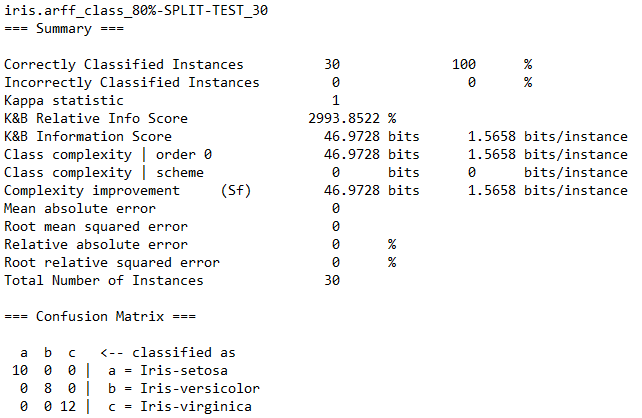
1. *Full Training*



1. *10-fold Cross Validation*

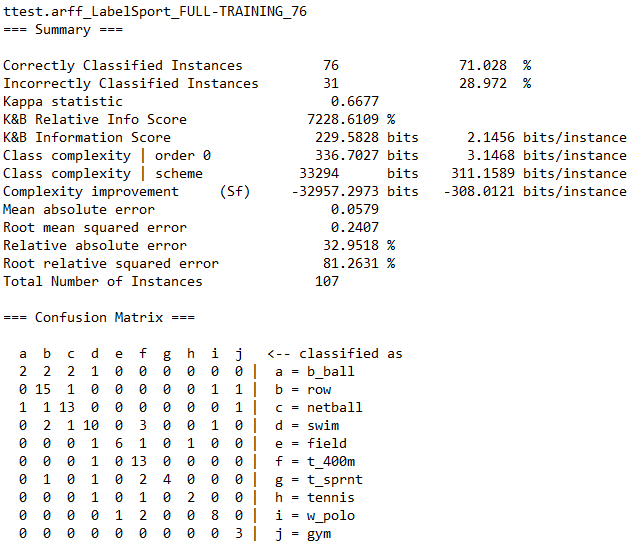


1. *Split Test*

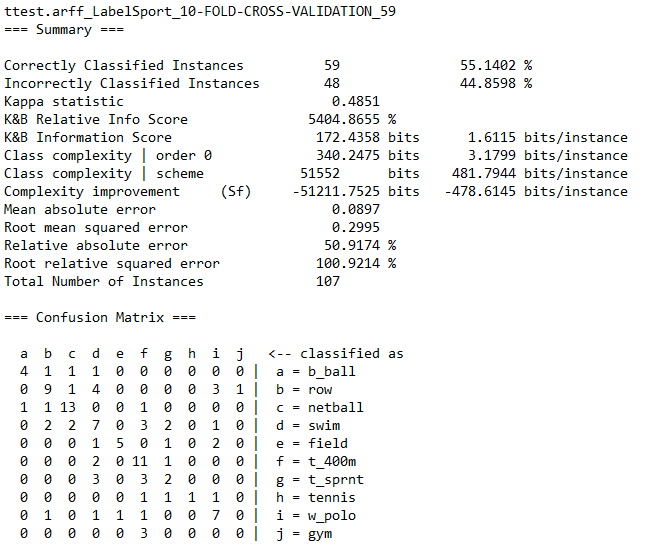


Nama file : Team.arff

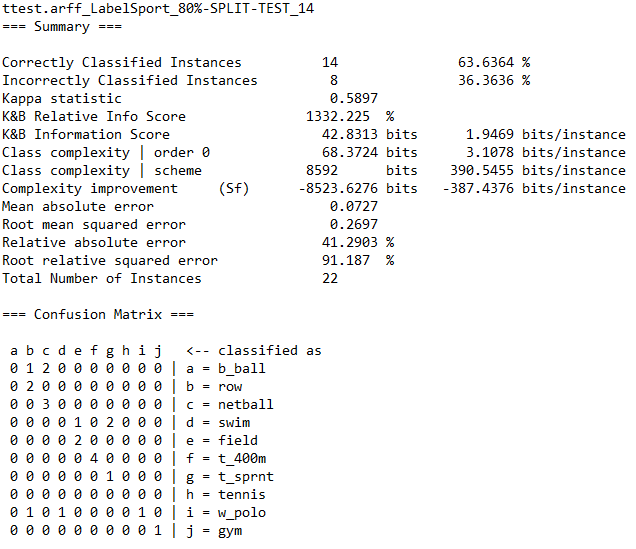
1. *Full Training*



1. *10-fold Cross Validation*

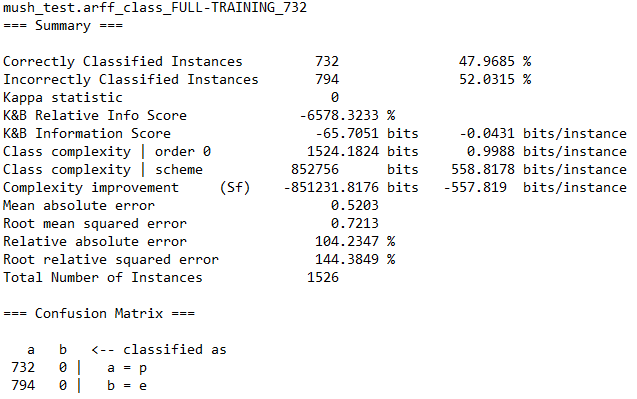


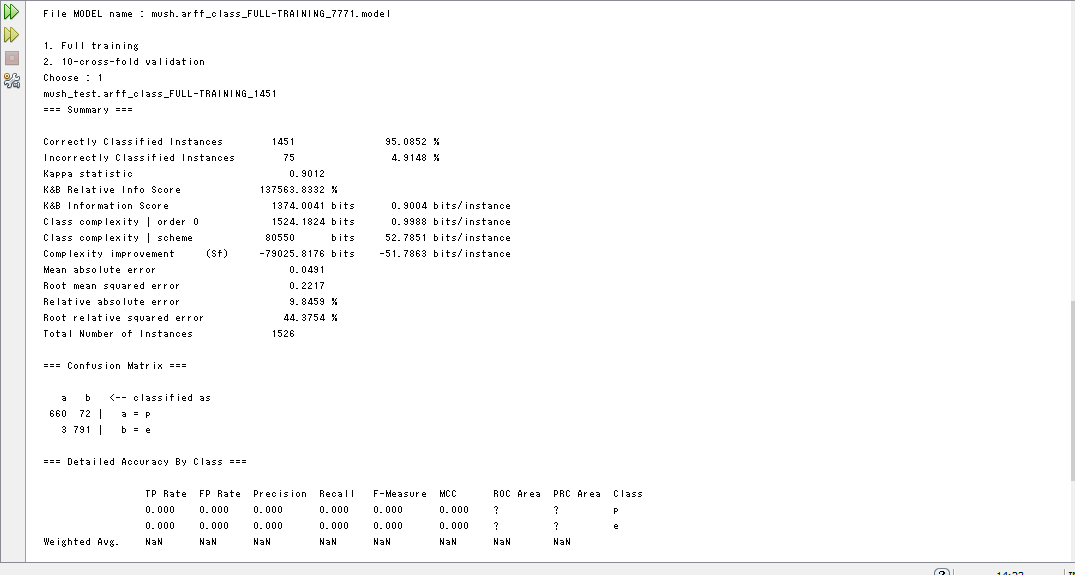
1. *Split Test*

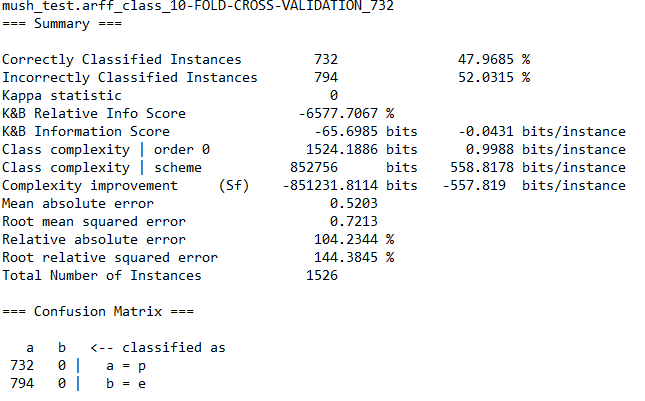


Nama file : Mush\_test.arff

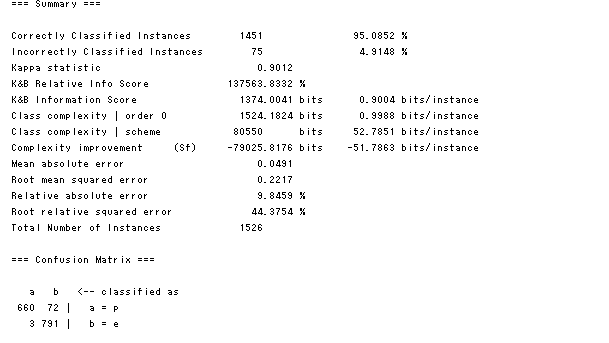
1. *Full Training*



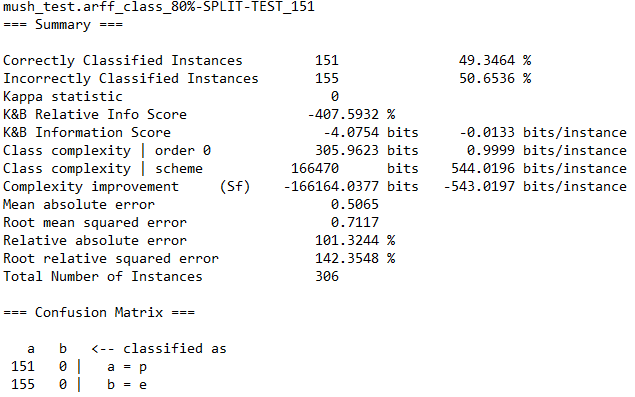
1. *Full Training with Bayes*
2. *10-fold Cross Validation*



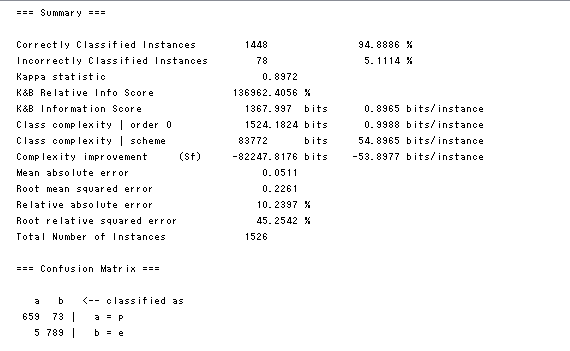
1. *Ten-Fold Cross Validation with Bayes*

**

1. *Split Test*

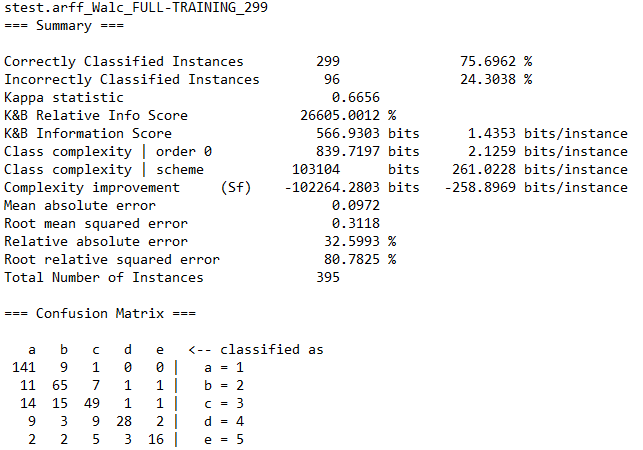


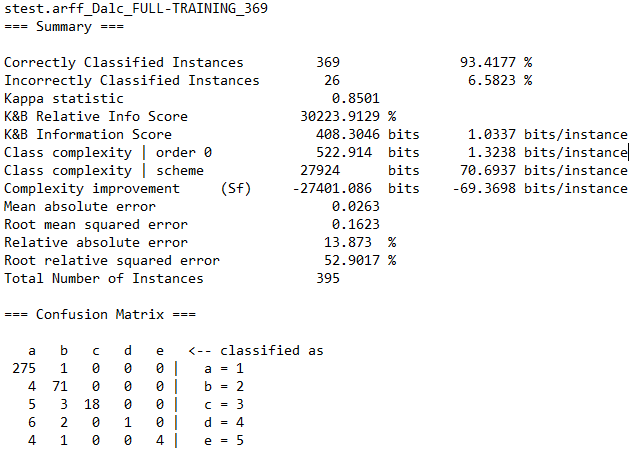
1. *Split Test with Bayes*

**

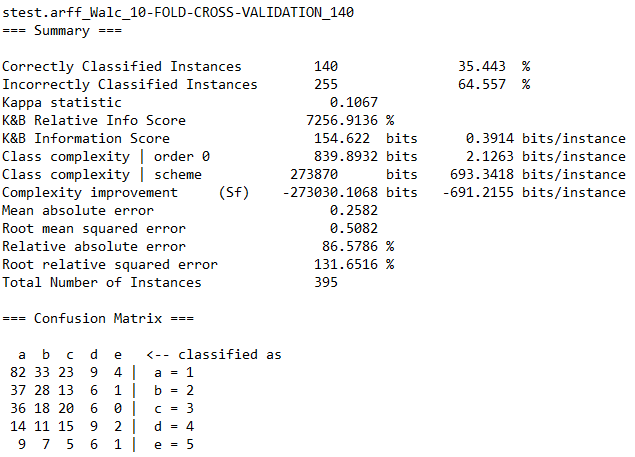
Nama file : student-math-test.arff

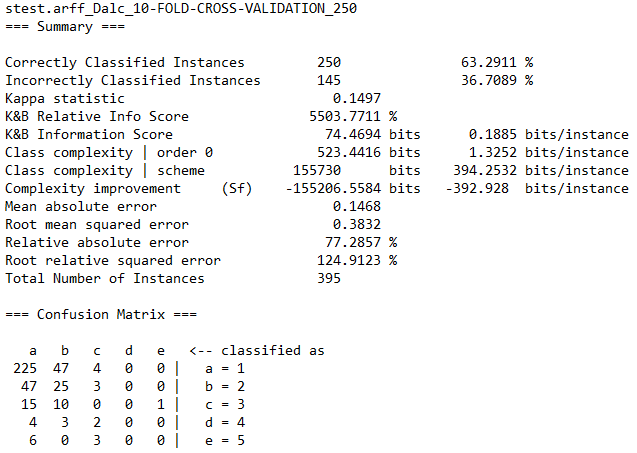
1. *Full Training*



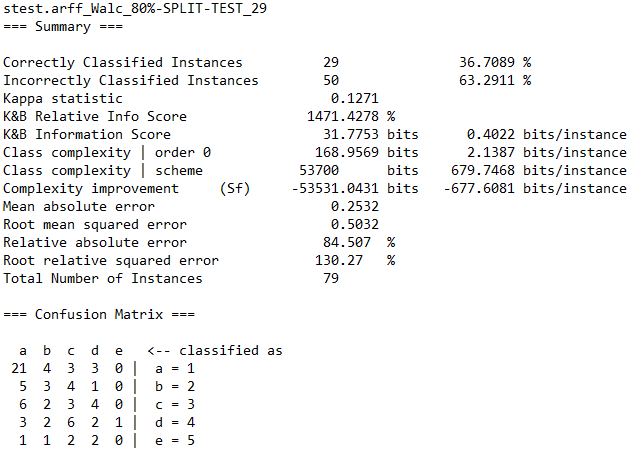


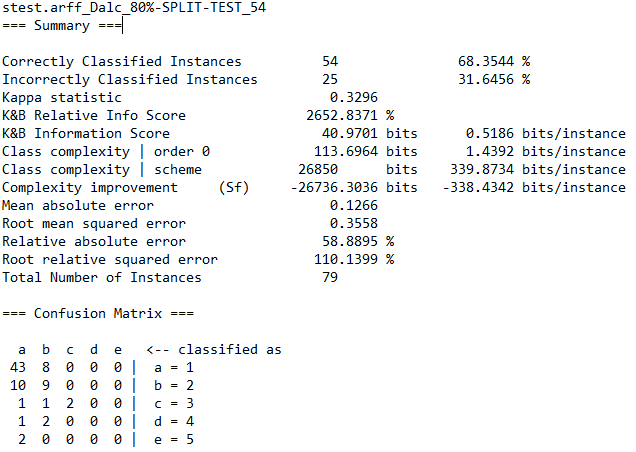
1. *10-fold Cross Validation*





1. *Split Test*



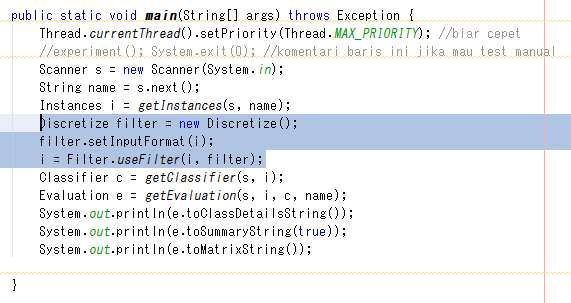


Matriks model hasil pengujian student-math-test.arrf dengan *classifier* Naïve Bayes dan *Feed Forward Neural Network* :

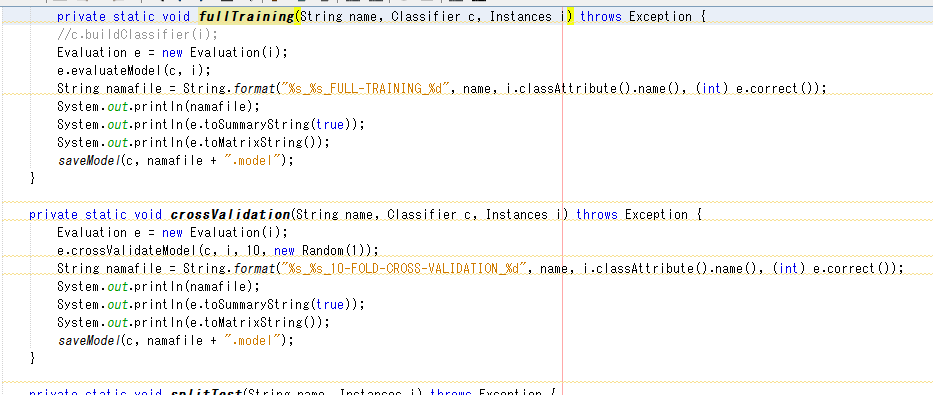
|  |  |
| --- | --- |
| **Walc** | **Dalc** |
| C:\Users\Deny\AppData\Local\Microsoft\Windows\INetCacheContent.Word\bayes1walc.png  Naïve Bayes Model | C:\Users\Deny\AppData\Local\Microsoft\Windows\INetCacheContent.Word\bayes1dalc.png  Naïve Bayes Model |
| Neural Network Model | Neural Network Model |

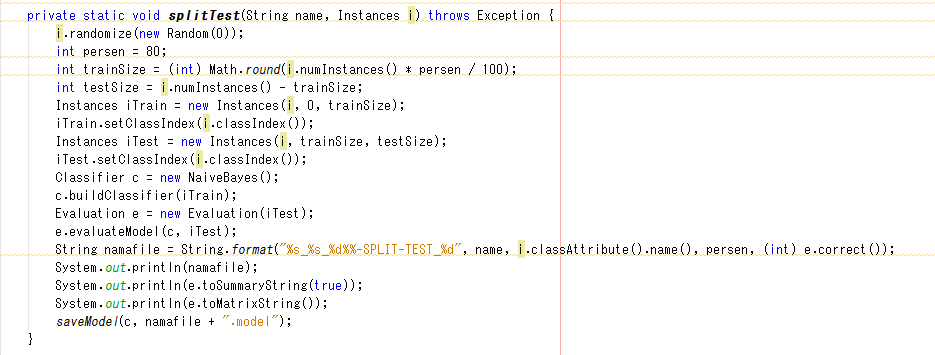
**NB**

Konversi Numeric to Nominal

****

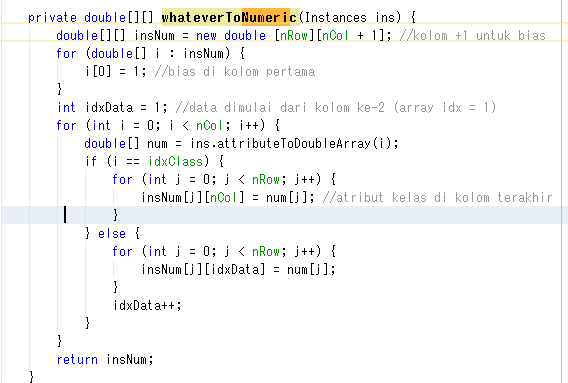
Memilih dari test yang terbaik

****

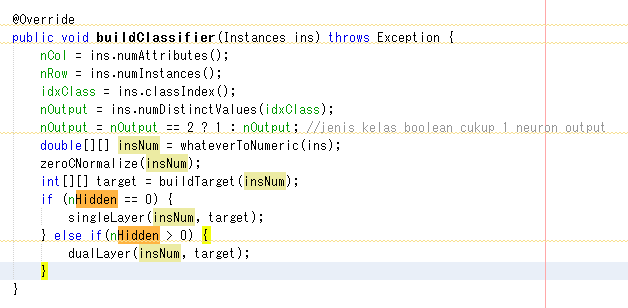
****

**ANN**

Konversi ke numeric

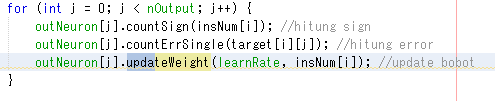


Toggle Hidden Layer



Backpropagation dan sigmoid





|  |
| --- |
| do {  epoch++;  //satu kali epoch = hitung seluruh row  for (int i = 0; i < nRow; i++) {  //menghitung sign untuk setiap neuron hidden  for (int j = 0; j < nHidden; j++) {  hidNeuron[j].countSign(insNum[i]);  signHid[j + 1] = hidNeuron[j].getSign();  }  //menghitung sign dan error untuk setiap neuron output  for (int j = 0; j < nOutput; j++) {  outNeuron[j].countSign(signHid);  outNeuron[j].countErrOut(target[i][j]);  }  //menghitung error dan update bobot untuk setiap neuron hidden  for (int j = 0; j < nHidden; j++) {  double sumErrXW = 0; //jumlah error x weight  for (Neuron n : outNeuron) {  sumErrXW += (n.getError() \* n.getWeight()[j]);  }  hidNeuron[j].countErrHid(sumErrXW);  hidNeuron[j].updateWeight(learnRate, insNum[i]);  }  //update bobot untuk setiap neuron output  for (Neuron n : outNeuron) {  n.updateWeight(learnRate, signHid);  }  }  //setiap beberapa epoch cek akurasinya  if (epoch % 250 == 0 || epoch == maxEpoch) {  Evaluation e = new Evaluation(insTest);  e.evaluateModel(this, insTest);  int correct = (int) e.correct();  //terdapat akurasi lebih tinggi  if (correct > maxCorrect) {  maxCorrect = correct;  //simpen model neuron hidden  maxHidNeuron = new Neuron[nHidden];  for (int i = 0; i < nHidden; i++) {  maxHidNeuron[i] = new Neuron(hidNeuron[i]);  }  //simpen model neuron output  maxOutNeuron = new Neuron[nOutput];  for (int i = 0; i < nOutput; i++) {  maxOutNeuron[i] = new Neuron(outNeuron[i]);  }  }  //akurasi udah 100%  if (correct == insTest.numInstances()) {  break;  }  }  } while(epoch < maxEpoch); |

Menggunakan model terbaik dari eksperimen skema 10 fold cross dan split test

|  |
| --- |
| private static void fullTraining(String name, Classifier c, Instances i) throws Exception {  c.buildClassifier(i);  Evaluation e = new Evaluation(i);  e.evaluateModel(c, i);  String namafile = String.format("%s\_%s\_FULL-TRAINING\_%d", name, i.classAttribute().name(), (int) e.correct());  System.out.println(namafile);  System.out.println(e.toSummaryString(true));  System.out.println(e.toMatrixString());  saveModel(c, namafile + ".model");  }    private static void crossValidation(String name, Classifier c, Instances i) throws Exception {  Evaluation e = new Evaluation(i);  e.crossValidateModel(c, i, 10, new Random(1));  String namafile = String.format("%s\_%s\_10-FOLD-CROSS-VALIDATION\_%d", name, i.classAttribute().name(), (int) e.correct());  System.out.println(namafile);  System.out.println(e.toSummaryString(true));  System.out.println(e.toMatrixString());  saveModel(c, namafile + ".model");  }    private static void splitTest(String name, Instances i,  double LR, int NH, int ME) throws Exception {  i.randomize(new Random(0));  int persen = 80;  int trainSize = (int) Math.round(i.numInstances() \* persen / 100);  int testSize = i.numInstances() - trainSize;  Instances iTrain = new Instances(i, 0, trainSize);  iTrain.setClassIndex(i.classIndex());  Instances iTest = new Instances(i, trainSize, testSize);  iTest.setClassIndex(i.classIndex());  Classifier c = new NaiveBayes();  c.buildClassifier(iTrain);  Evaluation e = new Evaluation(iTest);  e.evaluateModel(c, iTest);  String namafile = String.format("%s\_%s\_%d%%-SPLIT-TEST\_%d", name, i.classAttribute().name(), persen, (int) e.correct());  System.out.println(namafile);  System.out.println(e.toSummaryString(true));  System.out.println(e.toMatrixString());  saveModel(c, namafile + ".model");  } |