BIRZEIT UNIVERSITY

Faculty of Engineering and Technology

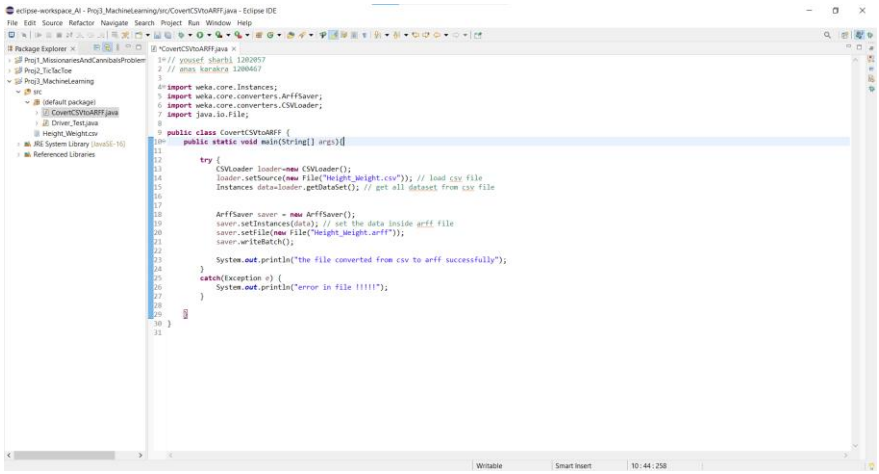
Computer Science Department

COMP338 – ARTIFICIAL INTELLIGENCE

	Students Name	Students Number
1	Yousef Sharbi	1202057
2	Anas karakra	1200467

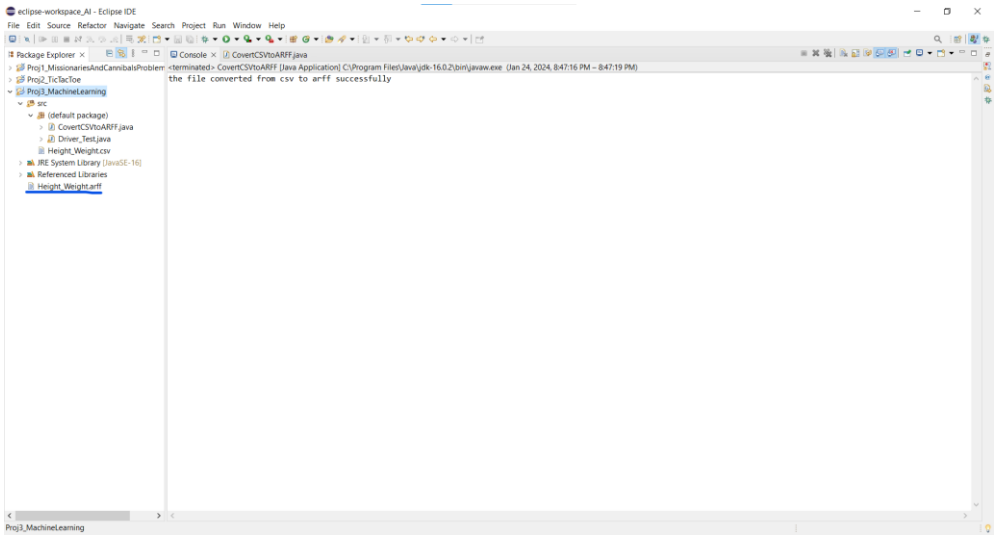
Project 3: Machine Learning (Regression)

This class convert .csv file to .arff file:



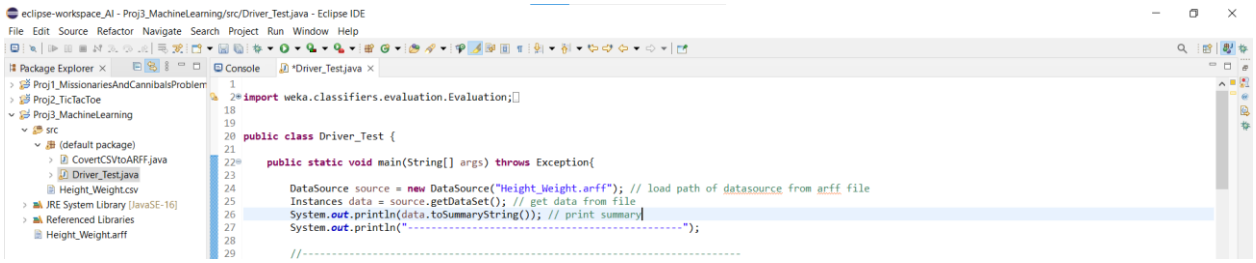
```
// yousef sharbi 1202057
// anas karakra 1200467

import weka.core.Instances;
import weka.core.converters.ArffSaver;
import weka.core.converters.CSVLoader;
import java.io.File;

public class CovertCSVtoARFF {
    public static void main(String[] args){

        try {
            CSVLoader loader=new CSVLoader();
            loader.setSource(new File("Height_Weight.csv")); // load csv file
            Instances data=loader.getDataSet(); // get all dataset from csv file


            ArffSaver saver = new ArffSaver();
            saver.setInstances(data); // set the data inside arff file
            saver.setFile(new File("Height_Weight.arff"));
            saver.writeBatch();

            System.out.println("the file converted from csv to arff successfully");
        }
        catch(Exception e) {
            System.out.println("error in file !!!!!");
        }

    }
}
```

Output:

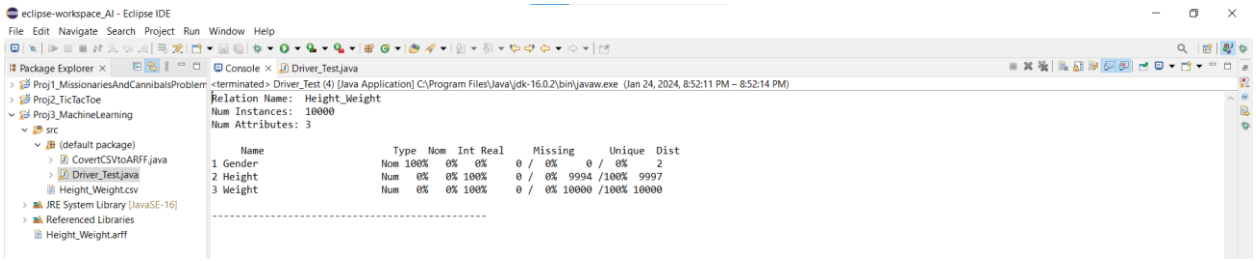


```
the file converted from csv to arff successfully
```

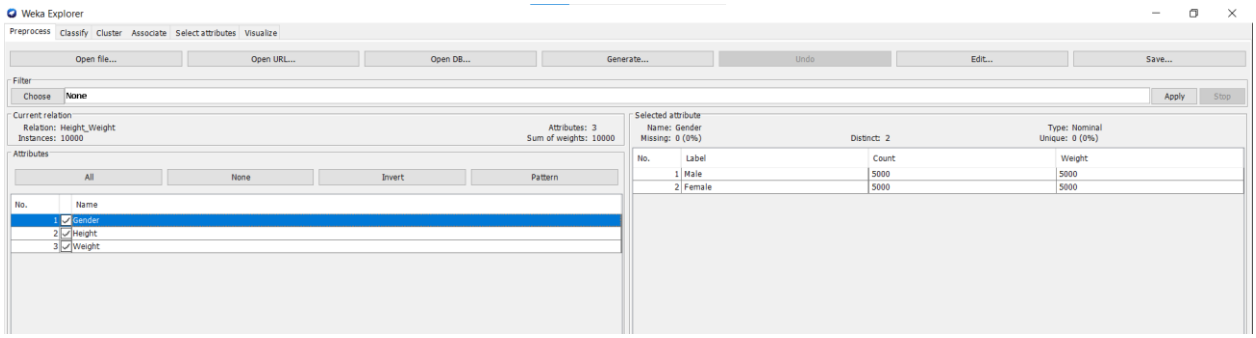
print data summary from dataset:



Output:



Weka application:

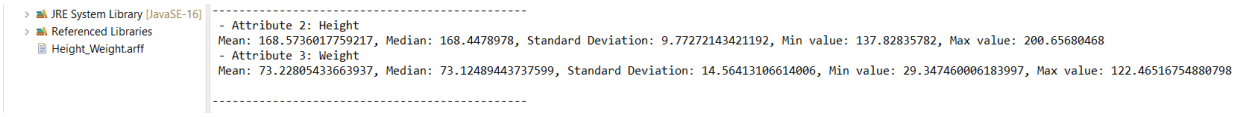


1) convert the height from inches to cms and the weight from pounds to kilograms:

2) Print the main statistics of the features (i.e., mean, median, standard deviation, min, and max values) in a proper table:


```
237
238
239
240    public static double calculateMedian(double[] values) {
241        int middle = values.length / 2;
242        if (values.length % 2 == 0) {
243            return (values[middle - 1] + values[middle]) / 2.0;
244        } else {
245            return values[middle];
246        }
247    }
248
249
250
251
```

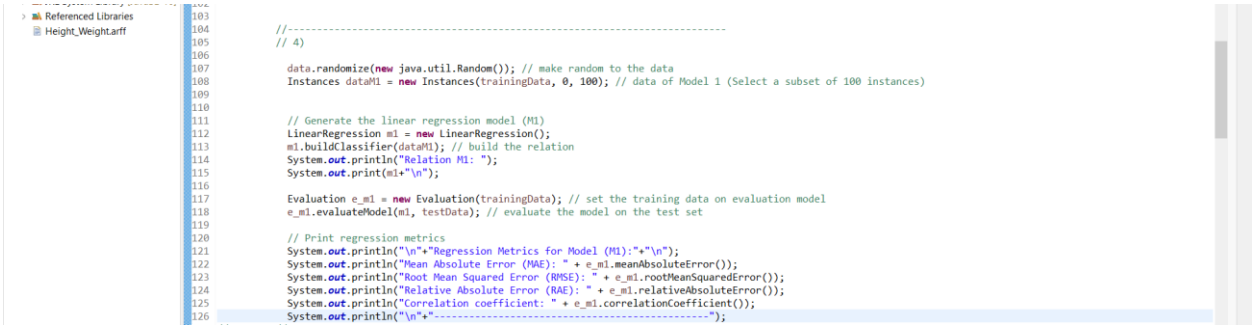
Output:



3) For each of the following models, you have to split the data into 70% training and 30% test:

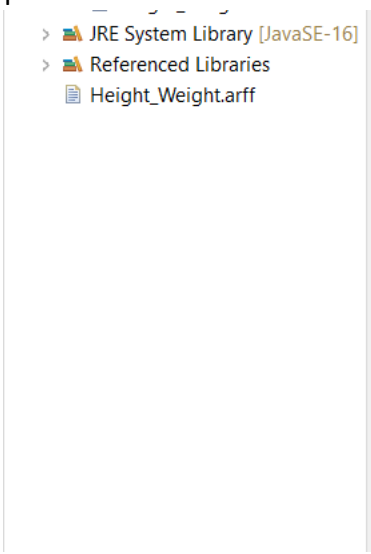
```
93          // 3)
94
95          data.setClassIndex(data.numAttributes()-1); // or you can use this code: data.setClass(data.attribute("Weight"));
96
97          int trainSize = (int)(data.numInstances() * 0.7); // size of training data
98          int testSize = data.numInstances() - trainSize; // size of test data
99
100         Instances trainingData = new Instances(data, 0, trainSize); //trainingData: 0 start ... till train size 70
101         Instances testData = new Instances(data, trainSize, testSize); //testData: 70 start ... till 100
102
103
```

4) Select a subset of 100 instances from randomly selected from the dataset and generate the first model (called M1) and test this models performance using appropriate regression metrics:




```
103
104         //------------------------------------------------------------------------
105         // 4)
106
107           data.randomize(new java.util.Random()); // make random to the data
108           Instances dataM1 = new Instances(trainingData, 0, 100); // data of Model 1 (Select a subset of 100 instances)
109
110
111           // Generate the linear regression model (M1)
112           LinearRegression m1 = new LinearRegression();
113           m1.buildClassifier(dataM1); // build the relation
114           System.out.println("Relation M1: ");
115           System.out.print(m1+"\n");
116
117           Evaluation e_m1 = new Evaluation(trainingData); // set the training data on evaluation model
118           e_m1.evaluateModel(m1, testData); // evaluate the model on the test set
119
120           // Print regression metrics
121           System.out.println("\n"+"Regression Metrics for Model (M1):"+"\n");
122           System.out.println("Mean Absolute Error (MAE): " + e_m1.meanAbsoluteError());
123           System.out.println("Root Mean Squared Error (RMSE): " + e_m1.rootMeanSquaredError());
124           System.out.println("Relative Absolute Error (RAE): " + e_m1.relativeAbsoluteError());
125           System.out.println("Correlation coefficient: " + e_m1.correlationCoefficient());
126           System.out.println("\n"+"-------------------------------------------------");
```

Output:




```
-------------------------------------------------
Relation M1:

Linear Regression Model

Weight =

      1.045  * Height +
    -98.5208

Regression Metrics for Model (M1):

Mean Absolute Error (MAE): 8.975152480831019
Root Mean Squared Error (RMSE): 9.98000623355699
Relative Absolute Error (RAE): 54.37054813872104
Correlation coefficient: 0.8502130121504533

-------------------------------------------------
```

5) Select a subset of 1000 instances from randomly selected from the dataset and generate the first model (called M2) and test this models performance using appropriate regression metrics:

```
//------------------------------------------------------------------
// 5)

  data.randomize(new java.util.Random()); // make random to the data
  Instances dataM2 = new Instances(trainingData, 0, 1000); // data of Model 2 (Select a subset of 1000 instances)

  // Generate the linear regression model (M2)
  LinearRegression m2 = new LinearRegression();
  m2.buildClassifier(dataM2); // build the relation
  System.out.println("Relation M2: ");
  System.out.print(m2+"\n");

  Evaluation e_m2 = new Evaluation(trainingData); // set the training data on evaluation model
  e_m2.evaluateModel(m2, testData); // evaluate the model on the test set

  // Print regression metrics
  System.out.println("\n"+"Regression Metrics for Model (M2):"+"\n");
  System.out.println("Mean Absolute Error (MAE): " + e_m2.meanAbsoluteError());
  System.out.println("Root Mean Squared Error (RMSE): " + e_m2.rootMeanSquaredError());
  System.out.println("Relative Absolute Error (RAE): " + e_m2.relativeAbsoluteError());
  System.out.println("Correlation coefficient: " + e_m2.correlationCoefficient());
  System.out.println("\n"+"-----------------------------------------------");
```

Output:

```
JRE System Library [JavaSE-16]
Referenced Libraries
Height_Weight.arff

-----------------------------------------------
Relation M2:

Linear Regression Model

Weight =

      1.0437 * Height +
    -97.9889

Regression Metrics for Model (M2):

Mean Absolute Error (MAE): 9.28496854321847
Root Mean Squared Error (RMSE): 10.269561955403626
Relative Absolute Error (RAE): 56.24738189393165
Correlation coefficient: 0.8502130121502433

-----------------------------------------------
```

6) Select a subset of 5000 instances from randomly selected from the dataset and generate the first model (called M3) and test this models performance using appropriate regression metrics:

```
//       //------------------------------------------------------------------------------
         // 6)

               data.randomize(new java.util.Random()); // make random to the data
               Instances dataM3 = new Instances(trainingData, 0, 5000); // data of Model 3 (Select a subset of 5000 instances)

               // Generate the linear regression model (M3)
               LinearRegression m3 = new LinearRegression();
               m3.buildClassifier(dataM3); // build the relation
               System.out.println("Relation M3: ");
               System.out.print(m3+"\n");

               Evaluation e_m3 = new Evaluation(trainingData); // set the training data on evaluation model
               e_m3.evaluateModel(m3, testData); // evaluate the model on the test set

               // Print regression metrics
               System.out.println("\n"+"Regression Metrics for Model (M3):"+"\n");
               System.out.println("Mean Absolute Error (MAE): " + e_m3.meanAbsoluteError());
               System.out.println("Root Mean Squared Error (RMSE): " + e_m3.rootMeanSquaredError());
               System.out.println("Relative Absolute Error (RAE): " + e_m3.relativeAbsoluteError());
               System.out.println("Correlation coefficient: " + e_m3.correlationCoefficient());
               System.out.println("\n"+"-----------------------------------------------");

//       //------------------------------------------------------------------------------
```

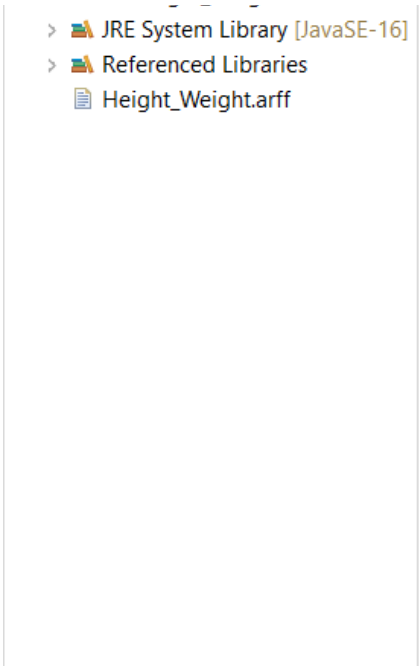
Output:

```
Height_Weight.csv
JRE System Library [JavaSE-16]
Referenced Libraries
Height_Weight.arff

-----------------------------------------------
Relation M3:

Linear Regression Model

Weight =

      1.0647 * Height +
   -101.8309

Regression Metrics for Model (M3):

Mean Absolute Error (MAE): 8.860976969821895
Root Mean Squared Error (RMSE): 9.871383250933473
Relative Absolute Error (RAE): 53.67888466772786
Correlation coefficient: 0.8502130121504335

-----------------------------------------------
```

7) Use the entire dataset and generate the first model (called M4) and test this models performance using appropriate regression metrics:

```
173         //7)
174            LinearRegression m4 = new LinearRegression();
175            m4.buildClassifier(trainingData);  // build the relation
176            System.out.println("Relation M4: ");
177            System.out.print(m4+"\n");
178
179            // Evaluate the model on the test set
180            Evaluation e_m4 = new Evaluation(trainingData); // set the training data on evaluation model
181            e_m4.evaluateModel(m4, testData); // evaluate the model on the test set
182
183            // Print regression metrics for Model M4
184            System.out.println("\nRegression Metrics for Model (M4):\n");
185            System.out.println("Mean Absolute Error (MAE): " + e_m4.meanAbsoluteError());
186            System.out.println("Root Mean Squared Error (RMSE): " + e_m4.rootMeanSquaredError());
187            System.out.println("Relative Absolute Error (RAE): " + e_m4.relativeAbsoluteError());
188            System.out.println("Correlation coefficient: " + e_m4.correlationCoefficient());
189            System.out.println("\n-----------------------------------------------");
190
191 //        //------------------------------------------------------------------------
```

Output:

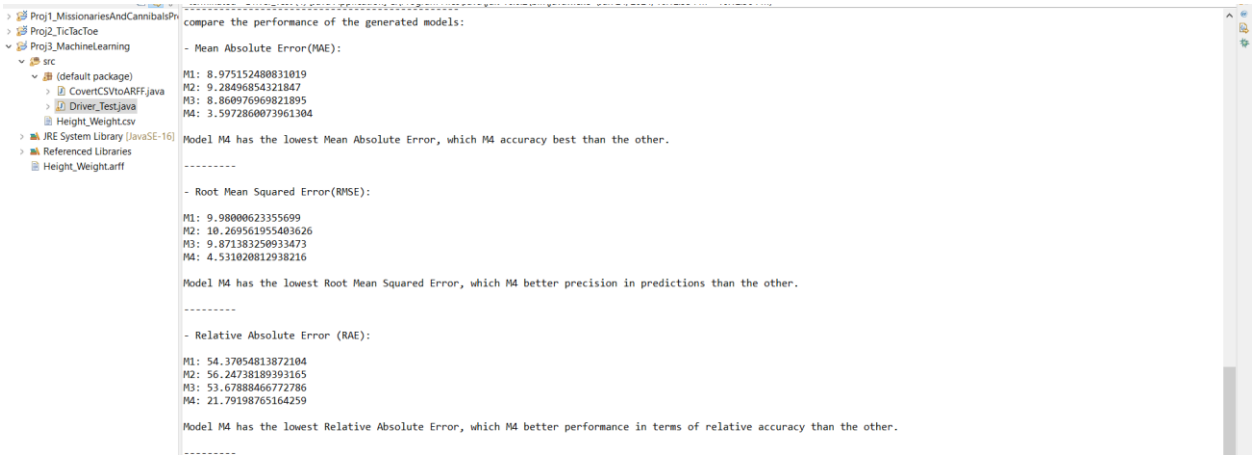



```
-----------------------------------------------
Relation M4:

Linear Regression Model

Weight =

      8.9048 * Gender=Male +
      1.0649 * Height +
   -110.7855

Regression Metrics for Model (M4):

Mean Absolute Error (MAE): 3.5972860073961304
Root Mean Squared Error (RMSE): 4.531020812938216
Relative Absolute Error (RAE): 21.79198765164259
Correlation coefficient: 0.8502130121505559

-----------------------------------------------
```

8) Print the appropriate performance metrics for Regression and compare the performance of the generated models in your own words:

```
        //8)

        System.out.println("compare the performance of the generated models:"+"\n");
        System.out.println("- Mean Absolute Error(MAE):"+"\n");
        System.out.println("M1: "+e_m1.meanAbsoluteError());
        System.out.println("M2: "+e_m2.meanAbsoluteError());
        System.out.println("M3: "+e_m3.meanAbsoluteError());
        System.out.println("M4: "+e_m4.meanAbsoluteError());

        System.out.println("\n"+"Model M4 has the lowest Mean Absolute Error, which M4 accuracy best than the other."+"\n");

        System.out.println("---------"+"\n");
        System.out.println("- Root Mean Squared Error(RMSE):"+"\n");
        System.out.println("M1: "+e_m1.rootMeanSquaredError());
        System.out.println("M2: "+e_m2.rootMeanSquaredError());
        System.out.println("M3: "+e_m3.rootMeanSquaredError());
        System.out.println("M4: "+e_m4.rootMeanSquaredError());

        System.out.println("\n"+"Model M4 has the lowest Root Mean Squared Error, which M4 better precision in predictions than the other."+"\n");

        System.out.println("---------"+"\n");
        System.out.println("- Relative Absolute Error (RAE):"+"\n");
        System.out.println("M1: "+e_m1.relativeAbsoluteError());
        System.out.println("M2: "+e_m2.relativeAbsoluteError());
        System.out.println("M3: "+e_m3.relativeAbsoluteError());
        System.out.println("M4: "+e_m4.relativeAbsoluteError());

        System.out.println("\n"+"Model M4 has the lowest Relative Absolute Error, which M4 better performance in terms of relative accuracy than the other."+"\n");

        System.out.println("---------"+"\n");
        System.out.println("- Correlation coefficient:"+"\n");
        System.out.println("M1: "+e_m1.correlationCoefficient());
        System.out.println("M2: "+e_m2.correlationCoefficient());
        System.out.println("M3: "+e_m3.correlationCoefficient());
        System.out.println("M4: "+e_m4.correlationCoefficient());

        System.out.println("\n"+"All models have the same correlation coefficient, which there is a strong linear relationship between the predicted and actual values.");
    }
```

Output:




```
---------

- Correlation coefficient:

M1: 0.8502130121504533
M2: 0.8502130121502433
M3: 0.8502130121504335
M4: 0.8502130121505559

All models have the same correlation coefficient, which there is a strong linear relationship between the predicted and actual values.
```