** Tutorial on Mobile Weka &&** **Data Mining **

**EECS397/EECS600, Mobile Computing && Sensor Network, Fall 2015**

**Advisor: Prof. Ming-Chun Huang**

**Introduction:**

In the previous tutorial, you retrieve built-in sensor data and touch screen data when you unlock a random pattern. In this tutorial, you will play with some interesting data-mining algorithms using the data collected through your pattern unlock program.

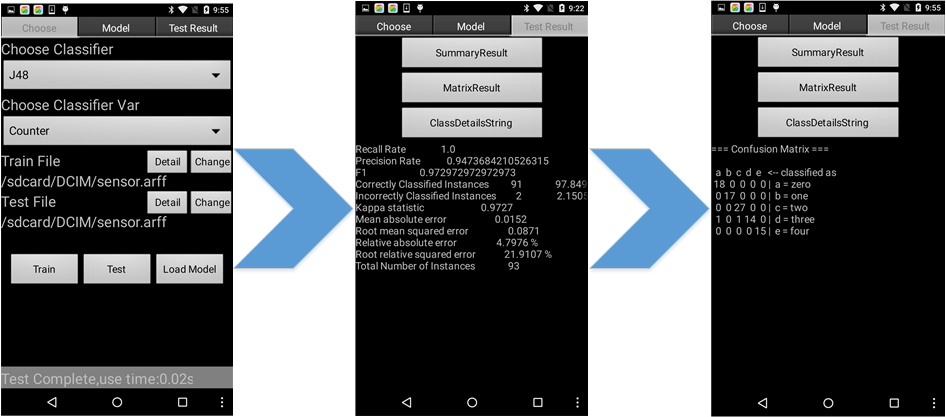


Figure 1 depicts a possible scheme for running J48 algorithm using sensor dataset. Through our tutorial, you should be able to configure the Mobile Weka correctly and try several data mining algorithm successfully. The tutorial is divided into three parts. The first part will introduce how to configure the Mobile Weka and how to add J48 algorithm in Mobile Weka. The second part will explain what is ARFF format and how to convert CSV file to ARFF file and some basic conception about data preprocessing. The third part will introduce our experiment requirement.

Disclaimer: This tutorial is for educational purpose only and it is not a 100% step by step tutorial, which means students still need to finish some parts by themselves. It is intended for students who would like to quickly overview the WEKA and data mining algorithm in Android. Without considering security and reliability, according to this tutorial, students can build a fast-working prototype for their class project.

***Pre-requisite:***

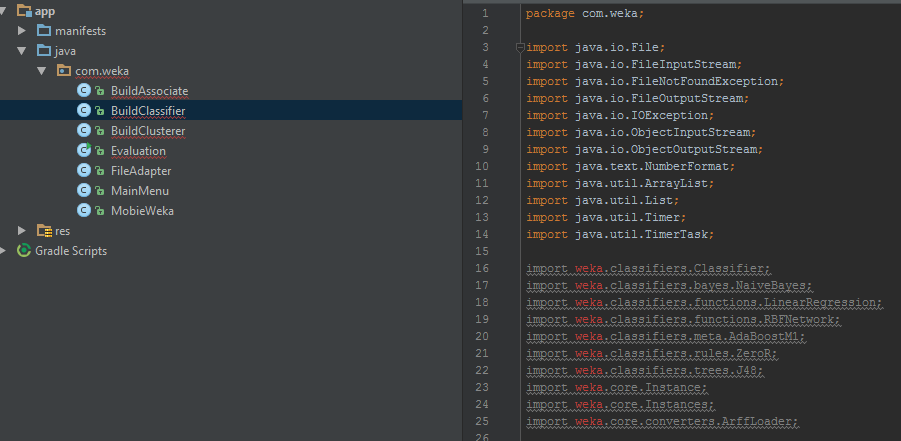
1. EECS 132 and EECS 233

2. Android Studio (recommended).

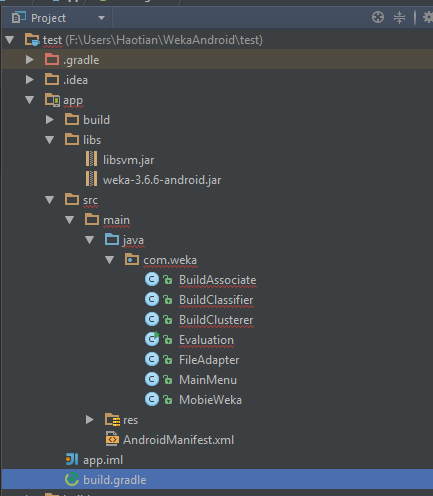
***Part I: Mobile WEKA (Waikato Environment for Knowledge Analysis)***

As we have expressed, Mobile Weka is the mobile version of Weka. The main differences between Mobile Weka and Weka are the running efficiency and their user interface. Mobile Weka support standard data mining tasks, e.g. clustering, classification and association rule mining. It is designed to easily launch data mining on mobile device, and be capable of seamlessly employ all of the algorithms embedded in Weka.

The Mobile Weka project is given directly and you can import it easily. Note that in order to use the machine learning Weka library in Android, you have to integrate Weka-3.6.6-android.jar into your project.



Before you add the library, You should create a *libs* folder under the *app* folder, and copy the *libsvm.jar* and *weka-3.6.6-android.jar* to the *libs* folder.



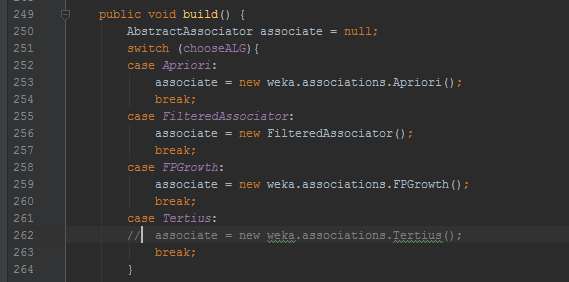
Then, go to the *build.gradle* highlighted in the above figure. Add the dependency as the following figure.



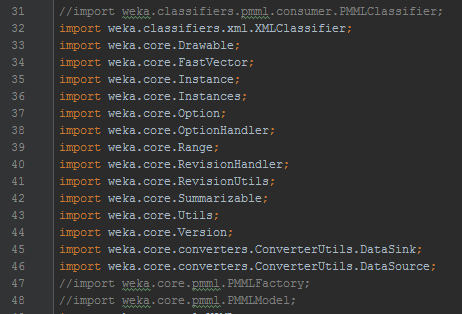
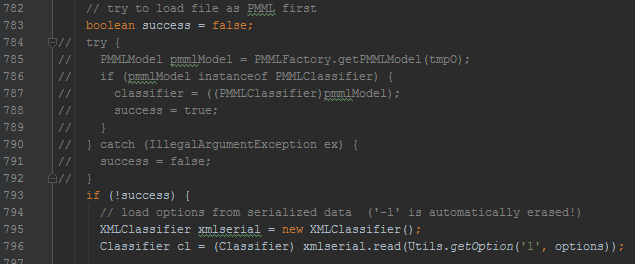
In the end, you should resynchronize the project with Gradle files.

However, for some reasons, there are still some errors in the *BuildAssociate.class* and *Evaluation.class*. weka-3.6.6-android.jar still miss some class files. Fortunately, these class files are not necessary for this tutorial, which means that we can just comment out the error parts.

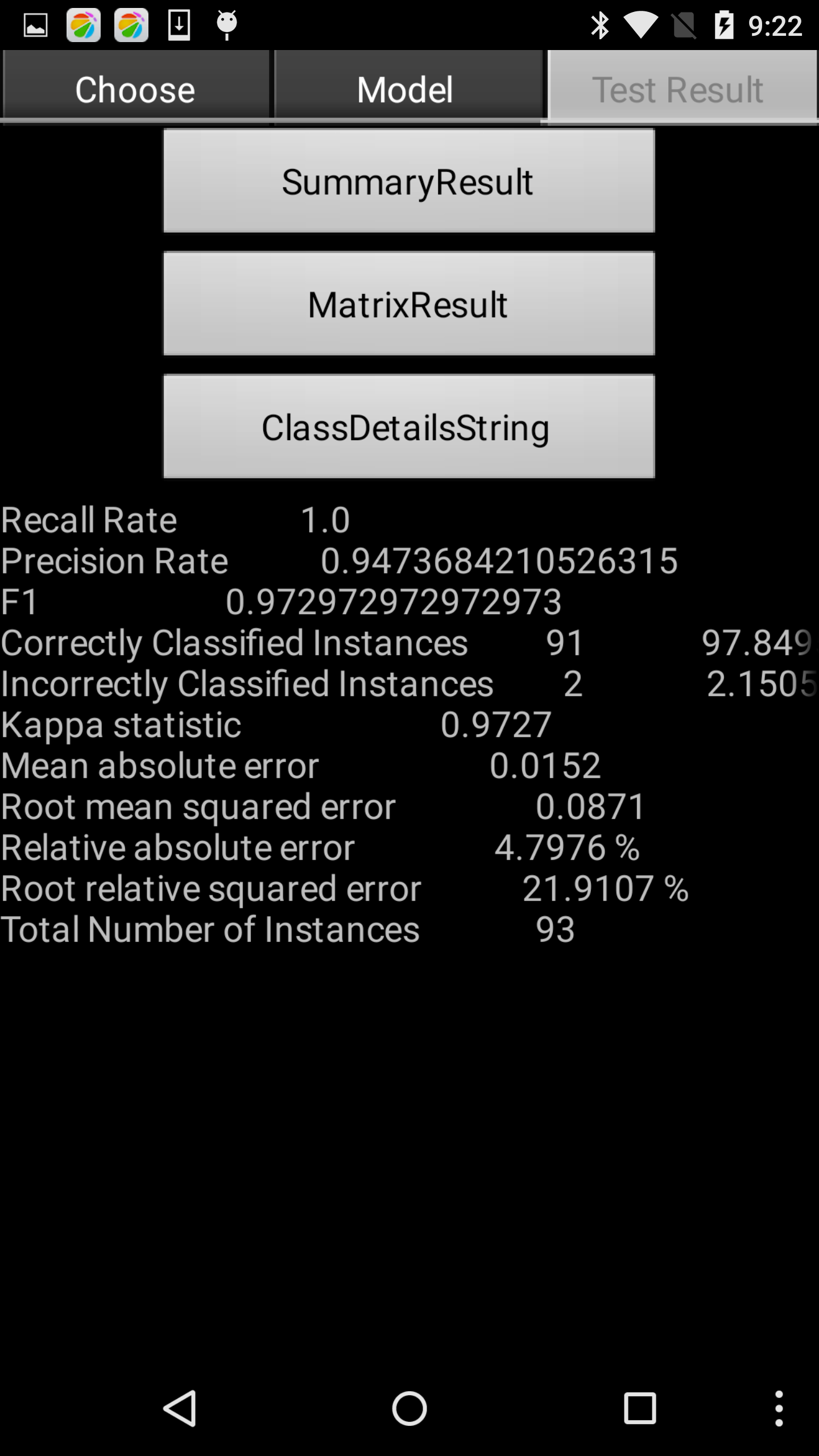
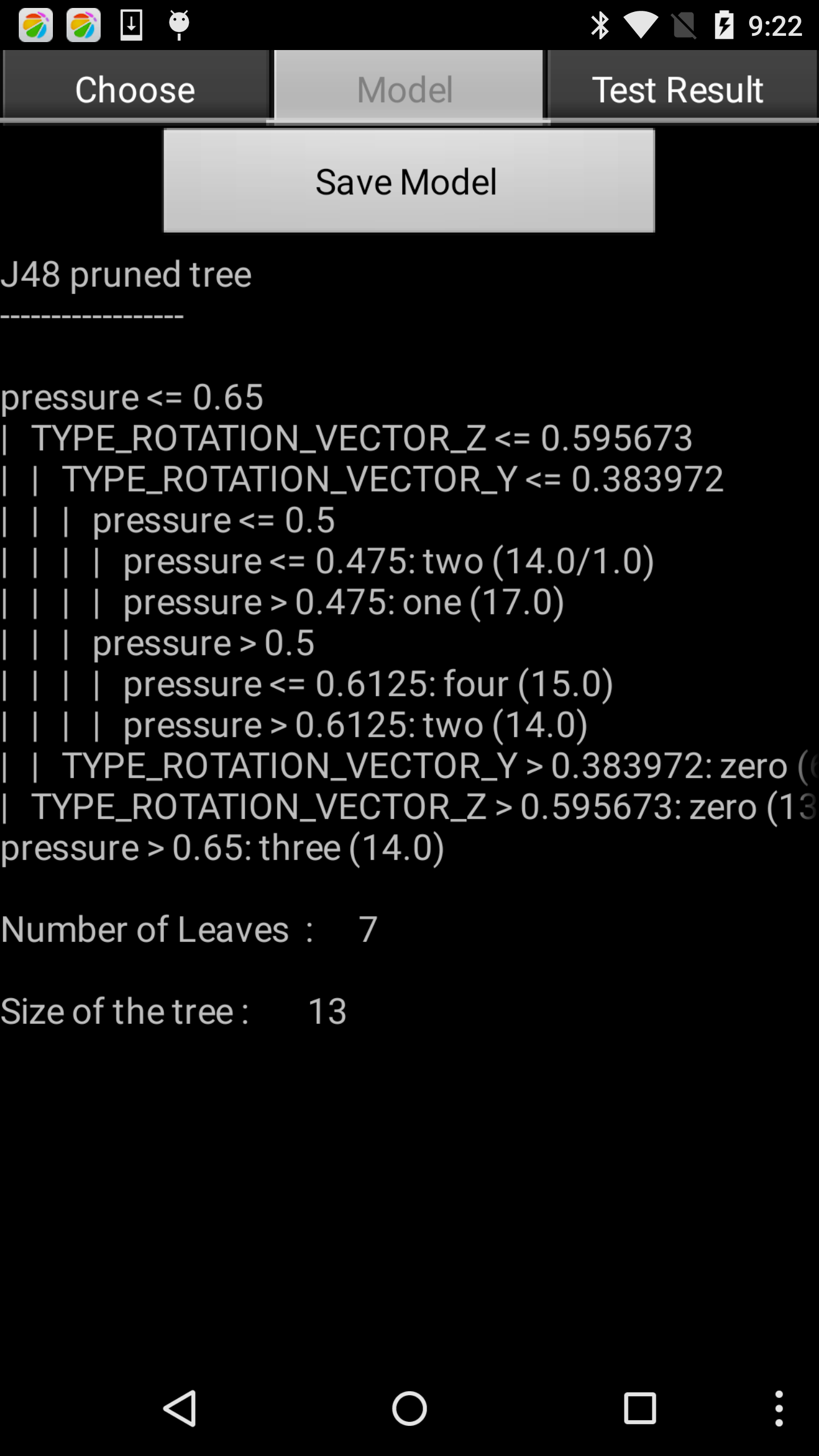
In *BuildAssociate.class*, we comment out case *Tertius* as the following figure.



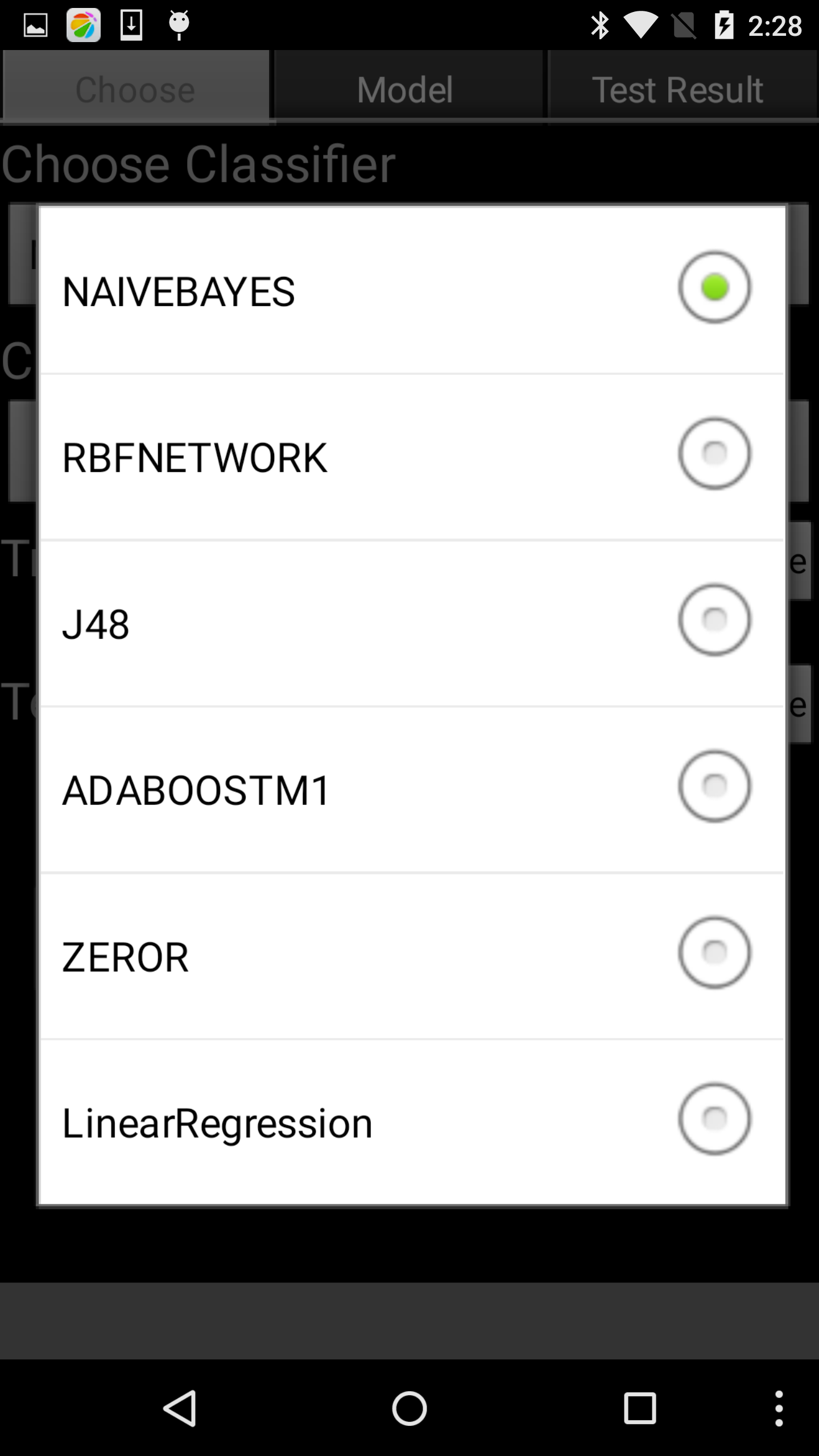
In *Evaluation.class*, we comment out *PMML* related source code as the following figures.



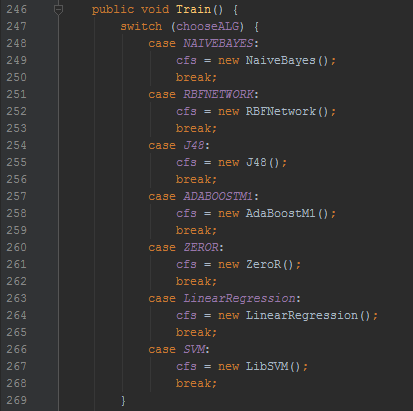
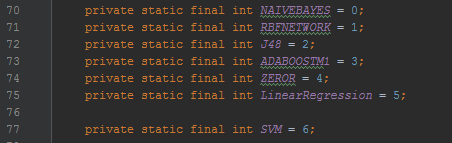
After building and running, you should have installed Mobile Weka on your Nexus 6 successfully. Now, you can use the temporary dataset *sensor.arff* in the zip file to play with J48 (C4.5 Decision Tree) first, so that you can make sure that your configuration is correct. You should get the exactly the same result as the following.



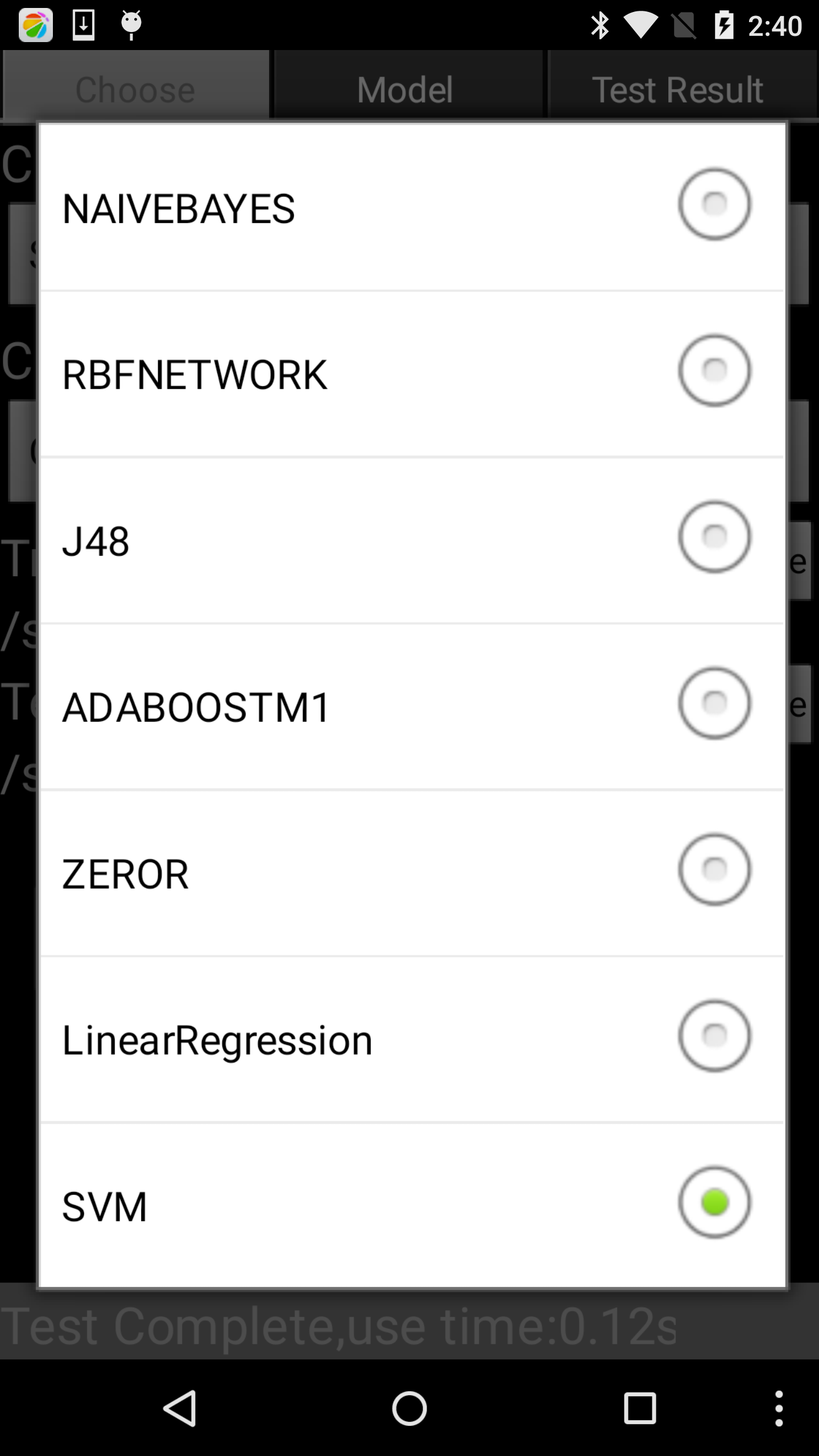
However, when you select other classifier, there is no SVM at all. How to import SVM?



Go to the *BuildClassifier.class*, and add the SVM code as the following figures.



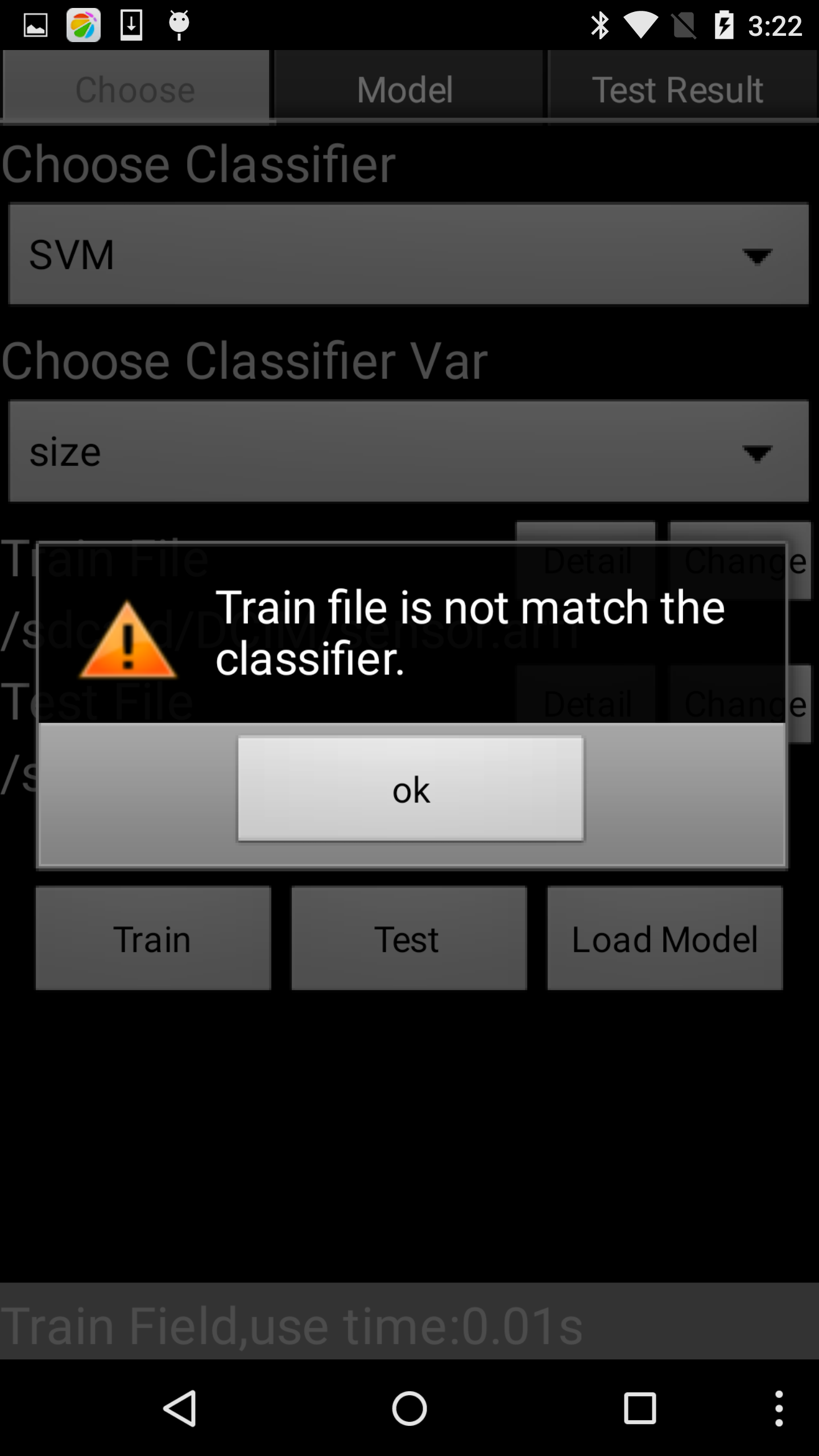
Recompile the whole project. There it is!



***Part II: Attribute-Relation File Format (ARFF) and Data Preprocessing***

An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files were developed by the Machine Learning Project at the Department of Computer Science of The University of Waikato for use with the [Weka machine learning software](http://www.cs.waikato.ac.nz/~ml/).

In the tutorial 1, we get sensor and touch motion dataset and save in CSV (comma separated value) file. However, in order to use the Mobile WEKA, the CSV file needs to be converted into the Weka's ARFF format. CSV files must contain the names of the attributes in the first line. All other lines should contain the data taken from your measurements. You can choose which attributes you want to include in the ARFF file and their type (numeric or nominal). For example, if you want to do the classification task using dataset from tutorial 1, you need to choose label first. For sensor and motion dataset, you can choose a current pattern as a label or you can choose counter as label. If you choose a counter as a label (do not consider the meaning of it, just an example), you need to convert the numeric type to the nominal first, 1 to one, 2 to two, 3 to three. Otherwise, you will get the error as following figure.



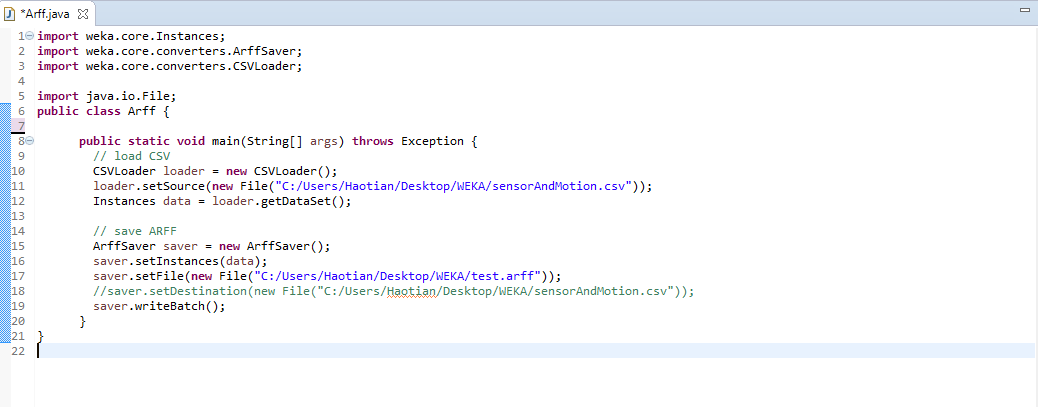
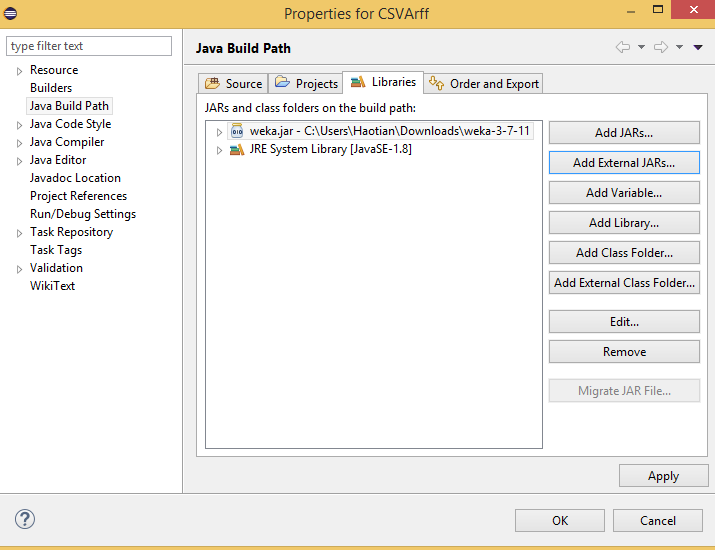
Real-world data is often incomplete, inconsistent, or lacking in certain behaviors and trends, and is  
likely to contain many errors. Data preprocessing is a proven method of resolving such issues and prepares raw data for further processing. The phrase “garbage in, garbage out” is particularly applicable to data mining and machine learning projects. We can do character removal, text replacement, date conversion in the step of data cleaning. Data conversion is also called blocking. It is worth noting that by doing all this blocking, we are potentially losing some information that may be relevant. In some cases this information was unimportant. In other cases, we simply do not have enough data within a group to make a determination about that group, which means that we need to combine some similar group so as to get a big enough group. In all cases, blocking data significantly simplifies the training model.

For our tutorial, it is not that complicated. You should delete the “TimeStamp” attribute because it is useless. The reason is that “TimeStamp” is simply a copy of the system time and it has no relationship with the lock pattern view. Then, you should omit the observation, which is the missing value. For instance, “null”. If there is a null value appears in the line, you should delete the whole line. Then, choose a label as nominal type. Now, you are ready to convert CSV file to ARFF.

There are several ways to convert CSV file to ARFF file.

The easiest way is to import your CSV file in WEKA on PC and save as ARFF.

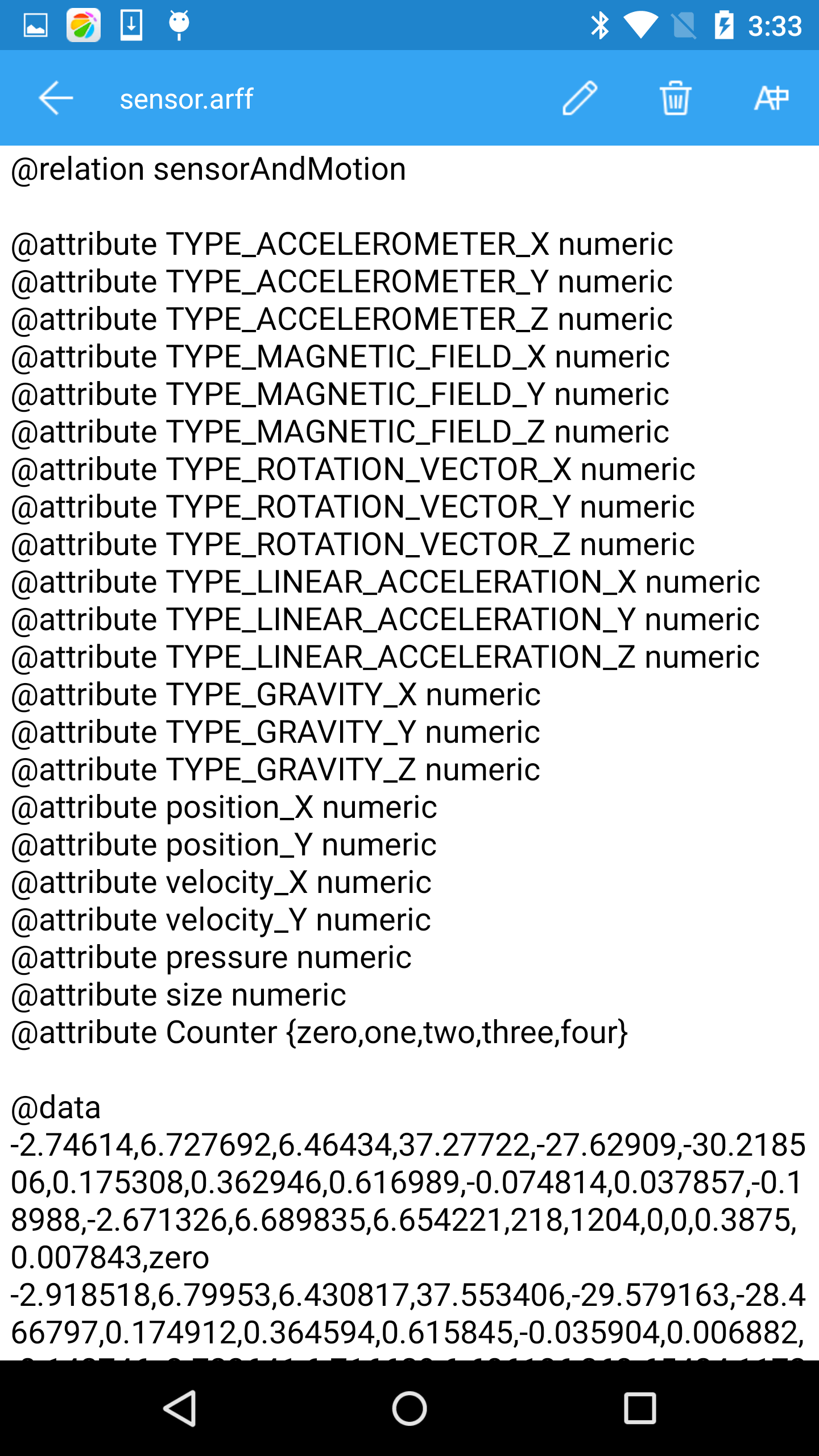
You can also add Weka library as external jar to the Eclipse, and copy the following code to your Eclipse project. You are encouraged to try Eclipse for learning it in depth.



The most convenient but relatively difficult way to convert CSV file to ARFF is in Android. For example, you can define a button and call *convert()* method. Converting CSV in android means once you know the specific storage path of CSV file in smart phone, you can load it using smart phone and do the conversion, then save it as ARFF file in a specific path in your smart phone. The above processes are implemented in Android Activity purely. Later, you should load the converted ARFF file to Mobile Weka and do some data mining task.

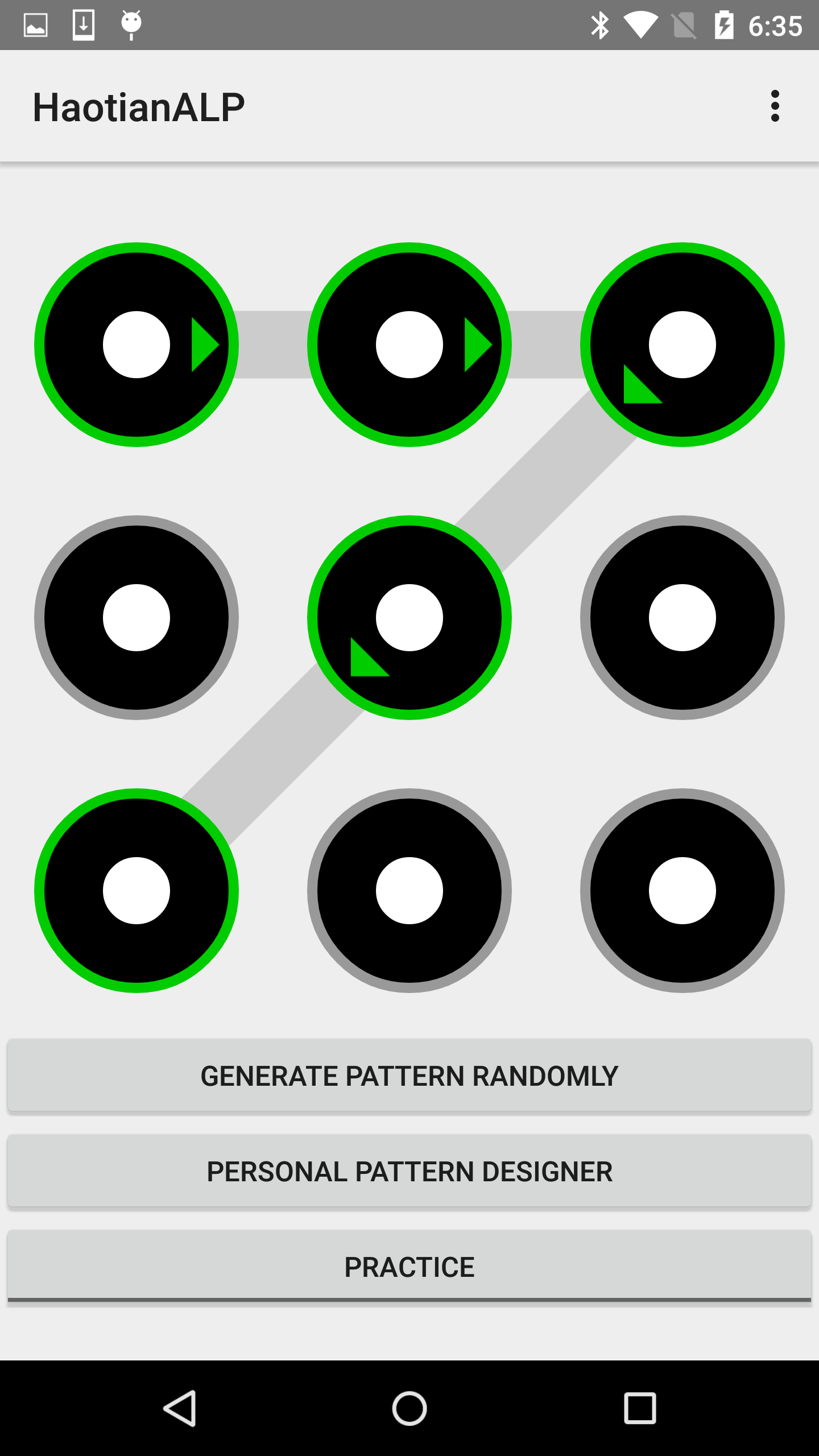
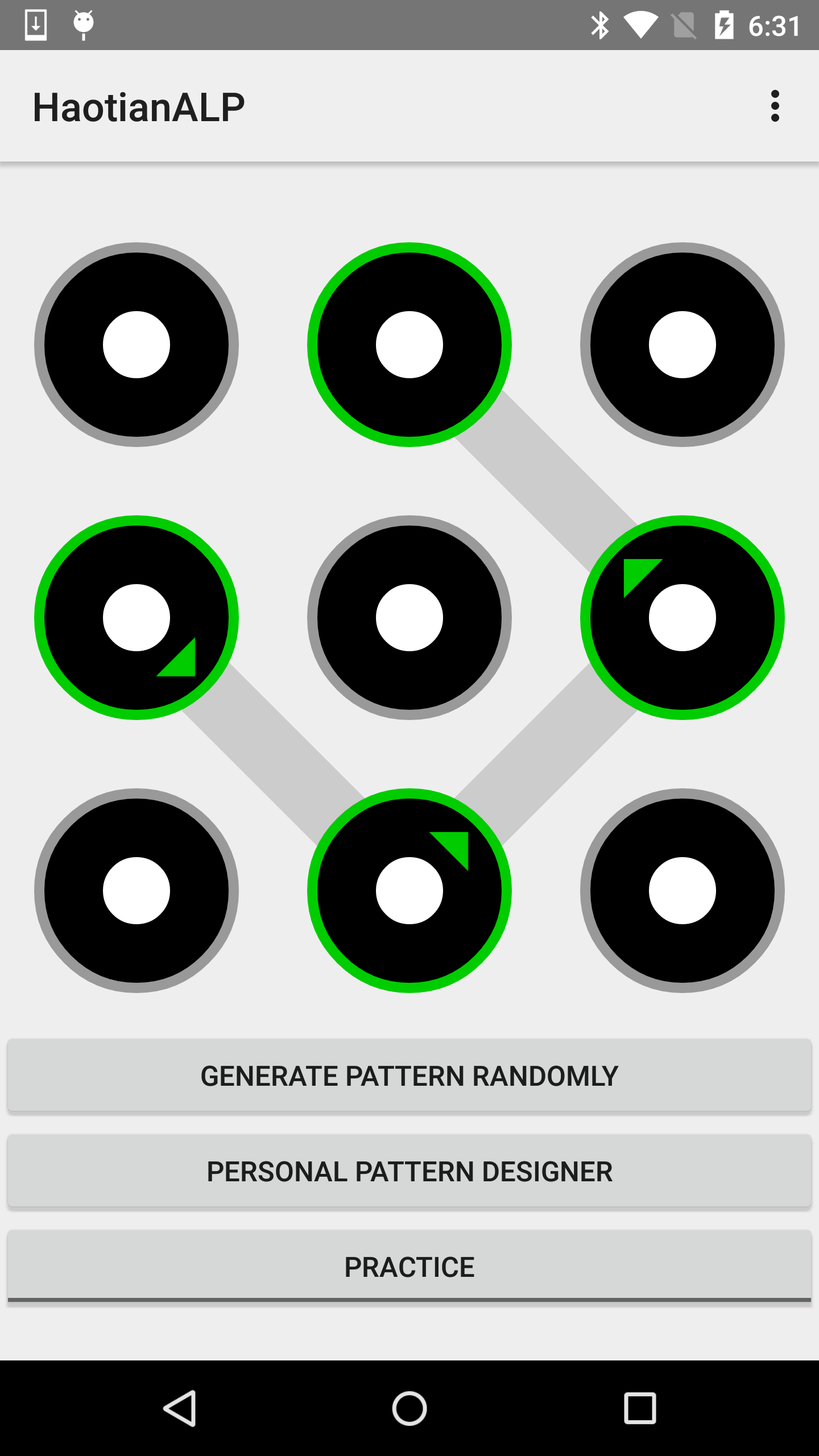
Note that: If you can figure out how to convert CSV to ARFF in Android, you will get great extra bonus! Note that the given snapshot is not a functional code!

You should get a valid ARFF file that looks like this (just an example):

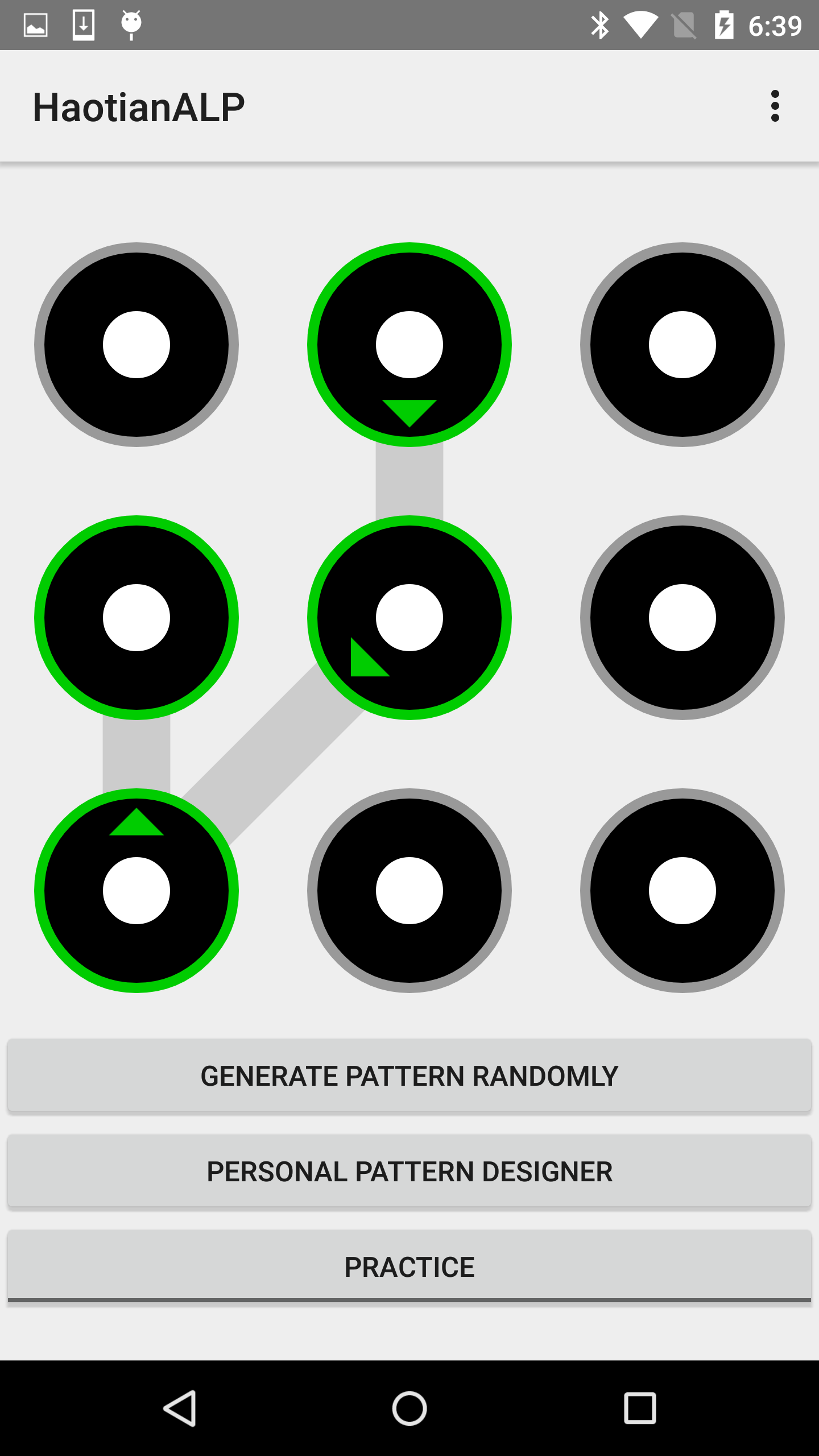
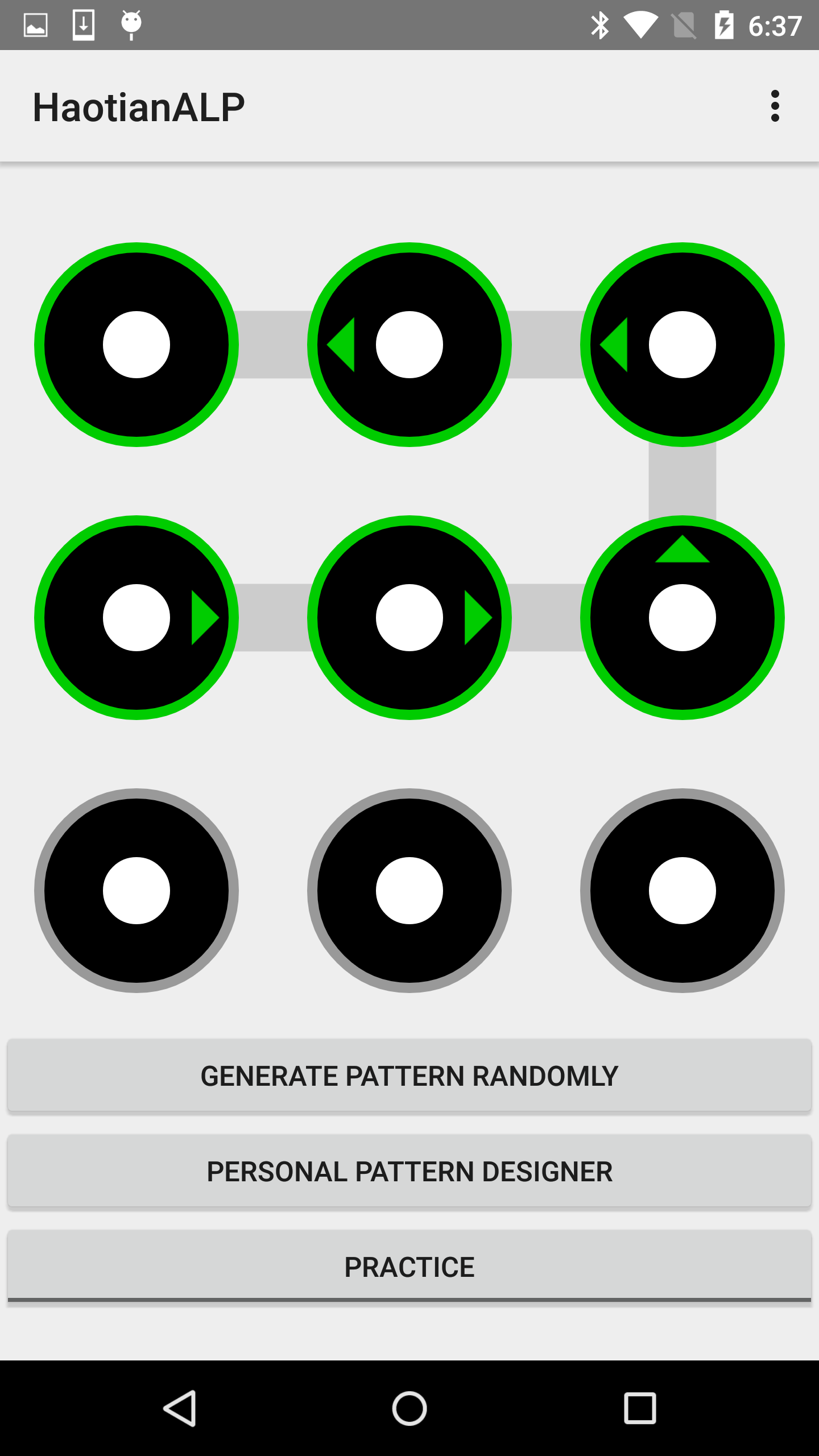


***Part III: Experimental Requirement***

First of all, we need the following four specific patterns.



**1**  **2**



**3**  **4**

**Experiment 1: Fix pattern, can you know the user ID using data mining?**

Each of you need to use pattern 1 as current pattern. Then, in practice mode, unlock the pattern correctly at least 50 times, which means the counter is greater than 49. Now, you and your partner represent label A and label B. Up to now, you have 50x label A and 50x label B (50 represents unlocking 50 times. If each time you get x lines of data, actually you should get 50x label.). You should divide the whole dataset into training set and test set. For training set, there are 35x label A and B. For test set, there are 15x A and B. Given a specific pattern, can you know who draw the pattern and generate the input data?

Note that: label must be nominal type.

For both undergraduate and graduate students, you should use J48 and SVM algorithm to build model and try to do the classification task.

For graduate students only, you should also pick up other two data mining algorithm to do the same task.

Please copy the original CSV dataset generated in the practice mode and ARFF file used in Mobile Weka in the folder, and submit a zip file in the end with your team report.

**Experiment 2: Fix user ID, can you know the pattern using data mining?**

Each of you need to use pattern 1, 2, 3, 4 as current pattern, respectively. Then, in practice mode, unlock each of the pattern correctly at least 50 times, which means the counter is greater than 49. Assign label A, B, C, D to pattern 1, 2, 3, 4 respectively. You should use the same method to divide the new dataset into training set and test set. Then, you can do the classification task. Given a specific person and some input data, can you know the pattern that these data came from?

Note that: label must be nominal type.

For both undergraduate and graduate students, you should use J48 and SVM algorithm to build model and try to do the classification task.

For graduate students only, you should also pick up other two data mining algorithm to do the same task.

Please copy the original CSV dataset generated in the practice mode and ARFF file used in Mobile Weka in the folder, and submit a zip file together with your team report.

**Experiment 3: Do some data preprocessing and repeat Exp 1&2**

Calculate mean and standard deviation for each attribute except for position\_x and position\_y. You should calculate mean and standard deviation for all the lines of data whose counter is equal to 0, 1, 2 ….n, respectively. Then, you can use mean and standard deviation as new attribute to replace the old attribute. In this way, you will have double number of attribute than before (each old attribute will generate two new attributes, mean and standard deviation). Now, you can repeat the experiment 1 and 2. You can compare the results with previous results and analyze the reason in the project report.

Note that: label must be nominal type.

For both undergraduate and graduate students, you should use J48 and SVM algorithm to build model and try to do the classification task.

For graduate students only, you should also pick up other two data mining algorithm to do the same task.

Please copy the original CSV dataset generated in the practice mode and ARFF file used in Mobile Weka in the folder, and submit a zip file together with your team report.

Report format:

1. Introduction of the algorithms

Each team should explain the fundamental of the 2 (undergraduate) or 4 (graduate) algorithms clearly in the project report. You can refer some equations, but you need to explain it to us. If you cite any paper or website, please write the reference list.

2. Data Preprocessing

You should specify how you did the data preprocessing, and what kind of features are used in the experiments.

3. Experimental results

You have to write experimental results. You can use table or figure.

4. Simple Results analysis

You can analyze the reason that why you get these results, and what you learn from it. Feel free to use equations or figures. This is an extra bonus part.

5. What your contribution is and what your partner’s contribution is.

Please specify your contribution. Remember, this is a team project!

Note that: You should write the three experiment in one team report. Then, put the report together with the dataset files in a zip file. If you can figure out how to convert CSV to ARFF in Android, please write your method in the report with a sample code and you will get great extra bonus! In addition, you still need to put all the source code in the zip file, including data preprocessing and ARFF converter, if any.