IoT Lab-6 (Week-12 Session-1)

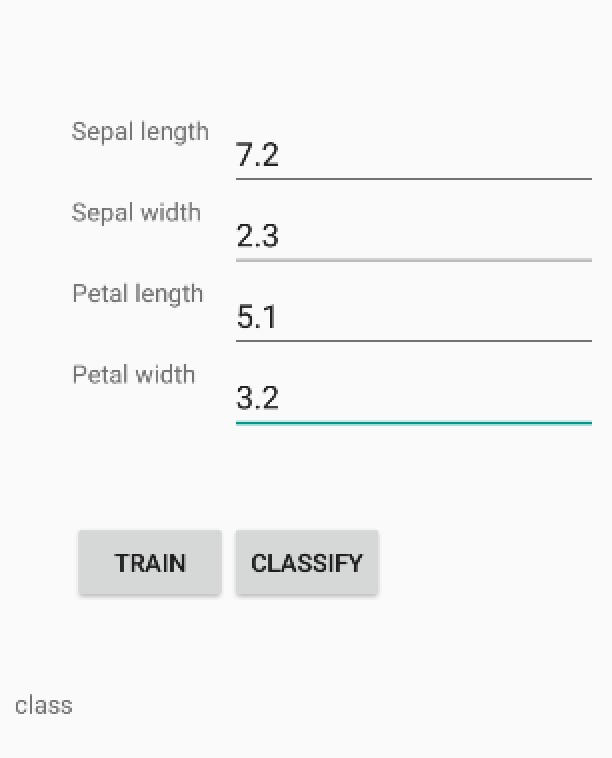
Topic: IoT data analytics

Objective: To design an Android interactive data classifier using SVM

In this lab, you will design an Interactive Android App running on Udoo that allows users to enter the attribute of an Iris flower and predicts the specie using the Weka API and a SVM classifier.

You will improve the Udoo classifier from Lab-5:

1. Use SVM, a more powerful classifier
2. Separate model training from online classification
3. Create an instance from the user input to perform interactive classification



1. Use SVM in Weka

Support vector machine (SVM) is a state-of-the-art data classifier. To use it in Weka, you need to download the following libraries, add them into your Udoo following the same procedure as in Lab-5 for “weka.jar”:

“libsvm.jar”

https://www.csie.ntu.edu.tw/~cjlin/libsvm/

“wlsvm.jar”

https://code.google.com/p/sentimental-crane/source/browse/m2/repository/twitter-sentiment-analysis/wlsvm/1.0/

Of course, you need “weka.jar” as well. To use SVM in your Java code:

WLSVM svmCls = new WLSVM();

Then you can use the buildClassifier() and classifyInstance() methods to train the model and classify new instance, resp., as in the previous lab.

2. Separate model training and online classification

Often, model training requires long processing times. It is not necessary to train the model for every new testing instance. Therefore, it is advantageous to separate the code for model training and instance testing. The system needs to train the model only once, saves the model to a file. Then, the trained model can be loaded whenever it is needed to classify a new instance.

To save your trained model to a file:

weka.core.SerializationHelper.write(svmModel, svmCls);

To load your previously trained model from a file:

WLSVM svmCls = (WLSVM) weka.core.SerializationHelper.read(svmModel);

Here svmModel is a String variable of the model file. Do not forget to add the access permission to your app.

3. Create an instance from the user input

In the previous lab, you load the test instances from a file and perform batch classification. However, interactive classification is often needed in many applications. For example, users may enter the attribute values interactively, or pulse rates are measured from a sensor in real-time. Thus, it is necessary to create an instance on-the-fly in some applications.

To do so, you need to first create a template for the instance. Then you create a new instance based on the template and the received attribute values.

To create the instance template (for the Iris data):

*// Declare four numeric attributes*Attribute Attribute1 = **new** Attribute(**"sepallength"**);  
Attribute Attribute2 = **new** Attribute(**"sepalwidth"**);  
Attribute Attribute3 = **new** Attribute(**"petallength"**);  
Attribute Attribute4 = **new** Attribute(**"petalwidth"**);  
  
*// Declare the class attribute along with its values (nominal)*FastVector fvClassVal = **new** FastVector(3);  
fvClassVal.addElement(**"Iris-setosa"**);  
fvClassVal.addElement(**"Iris-versicolor"**);  
fvClassVal.addElement(**"Iris-virginica"**);  
Attribute ClassAttribute = **new** Attribute(**"class"**, fvClassVal);  
  
*// Declare the feature vector template*FastVector fvWekaAttributes = **new** FastVector(5);  
fvWekaAttributes.addElement(Attribute1);  
fvWekaAttributes.addElement(Attribute2);  
fvWekaAttributes.addElement(Attribute3);  
fvWekaAttributes.addElement(Attribute4);  
fvWekaAttributes.addElement(ClassAttribute);

To create new instances from the received attribute values:

*// Creating testing instances object with name "TestingInstance"*

*// using the feature vector template we declared above*

*// and with initial capacity of 1*

Instances testingSet = **new** Instances(**"TestingInstance"**, fvWekaAttributes, 1);

*// Setting the column containing class labels:*

testingSet.setClassIndex(testingSet.numAttributes() - 1);  
  
*// Create and fill an instance, and add it to the testingSet*Instance iExample = **new** Instance(testingSet.numAttributes());  
  
iExample.setValue((Attribute)fvWekaAttributes.elementAt(0), slValue);  
iExample.setValue((Attribute)fvWekaAttributes.elementAt(1), swValue);  
iExample.setValue((Attribute)fvWekaAttributes.elementAt(2), plValue);  
iExample.setValue((Attribute)fvWekaAttributes.elementAt(3), pwValue);  
iExample.setValue((Attribute)fvWekaAttributes.elementAt(4),

**"Iris-setosa"**); *// dummy*

*// add the instance*testingSet.add(iExample);

Then testingSet can be classified as in the previous lab.

Demonstrate your app to your instructor. Test it with the following attribute values, and report the classification results.

% sepallength, sepalwidth, petallength, petalwidth

4.7, 3.2, 1.6, 0.2

7.0, 3.2, 4.7, 1.4