A Mini-Project Report

ON

Prediction of the ''play'' attribute in the weather dataset

BY

Aniruddha Hore
1PI13IS019

Under the guidance of

Dr. Shylaja S.S.

August 2014 – December 2014

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

PES INSTITUTE OF TECHNOLOGY
100 FEET RING ROAD, BANASHANKARI III STAGE
BANGALORE-560085

PES Institute of Technology,
100 Feet Ring Road, BSK 3rd Stage, Bangalore-560085

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Mini-Project entitled 'Prediction of the ''play'' attribute in the weather dataset ' is presented by **Aniruddha Hore 1PI13IS019** in partial fulfillment for the award of degree of **Bachelor of Engineering in Information Science of the Visvesvaraya Technological University, Belgaum during the year 2014-2015**. It is certified that all corrections / suggestions indicated for Internal Assessment have been incorporated in the report. The Special Topic has been approved as it satisfies the academic requirements in respect of Special Topic prescribed for the Bachelor of Engineering Degree.

Signature of the Guide
Dr. Shylaja S.S

Signature of the HOD
Dr. Shylaja S.S

TABLE OF CONTENTS

1.CERTIFICATE

2.ABSTRACT

3.INTRODUCTION

4.LITERATURE SURVEY

5.SYSTEM DESIGN

6.IMPLEMENTATION

7.RESULTS

8.CONLUSIONS

9.FUTUTRE ENHANCEMENTS

10.REFERENCES

11. ACKNOWLEDGEMENT

ABSTRACT

Our objective was the binary classification of the value of the play attribute in the "weather" dataset by building a decision tree using weka's J48 classifier. We used the concepts of machine learning by importing the Weka library in predicting the value of the play attribute. We collected a dataset which consisted of different attributes namely "min temp","max temp","humidity","rain","play" etc and passed this data(as trianing data) to the J48 classifier(present in the weka library).The J48 classifier built a decision tree based on this data and then the testing dataset was passed. After the testing dataset was passed we got the values predicted for the unlabelled play attribute as either "yes" or "no" (which was the sole aim of the project) according to the decision tree built by the J48 classifer on the basis of training data. We concluded that using the concepts of machine learning one could predict the value of any attribute in a given dataset.This prediction could be further used as useful information for other activites for e.g. conducting a cricket match in some city given we have the past weather data for the city.

INTRODUCTION

Our project predicts the value of the 'play' attribute in the 'weather' dataset. The 'play' attribute holds the value 'yes' when it is favourable to go outside and play , otherwise it holds the value 'no'.

Our project uses the J48 classifier of the WEKA library to build a decision tree and predict the values of the 'play' attribute of the dataset. We use the method of 10 cross split validation for training our model and further calculate the accuracy of the built model.

Finally we label the unknown instances using the model and also display the decision tree that we used for classification.

LITERATURE SURVEY

We referred to several textbooks for getting aquainted with the basic concepts of machine learning as quoted in our references page. Along with the textbooks we also found some code snippets on http://www.programcreek.com/ very helpful. In addition to these we also referred to http://weka.wikispaces.com/ for understanding the Weka library and how to use Weka with java.After referring to these all resources we got familiar with the basic terms and concepts of machine learning along with the Weka library and how to use it with java. The extract of our study for the project is given below:

MACHINE LEARNING: Machine learning is a scientific discipline that deals with the construction and study of algorithms that can learn from data. Such algorithms operate by building a model based on inputs and using that to make predictions or decisions, rather than following only explicitly programmed instructions. There are many real world examples of machine learning like pattern recognition,biometric verification,speech recognition etc. There are different types of machine learning.

1.Classification: Classifying different objects into certain classes they belong to.It is also known as supervised learning because the classes are known ahead of time.

2.Clustering: This is unsupervised learning because the size and shape of clusters is not known to us.

3.Recommendation: Recommend a value for an item on the basis of past behaviour of things related to it.

Inorder to implement machine learning using a programming language the first thing we need is a dataset containg objects and their attributes.Then we need to write a

function or code which will train and learn from the dataset and certain algorithms included in the libraries.

The function separates our data into training data ad test data which is called as split cross validation method.

Once a good number of training data is fed to our function it then takes as input a new set of data called as future data or unseen data and tries to predict or approximate.

The prediction or appoximation is not ideal ofcourse.It includes errors and the calculation of this error is also a fundamental process in machine learning.Error is calculated using error function or loss function.

Dataset:A set of instances with different values of all the attributes for the instances . Generally, no ordering on instances is assumed. Most machine learning algorithms use **ARFF** (attribute relation file format)format.

Training data: The part of the dataset used for training the classifier.

Testing data: The part of the dataset used for testing the classifier.

Attributes/features: the properties of an object from a given dataset.

Label:It is a special attribute which denotes the objects class.

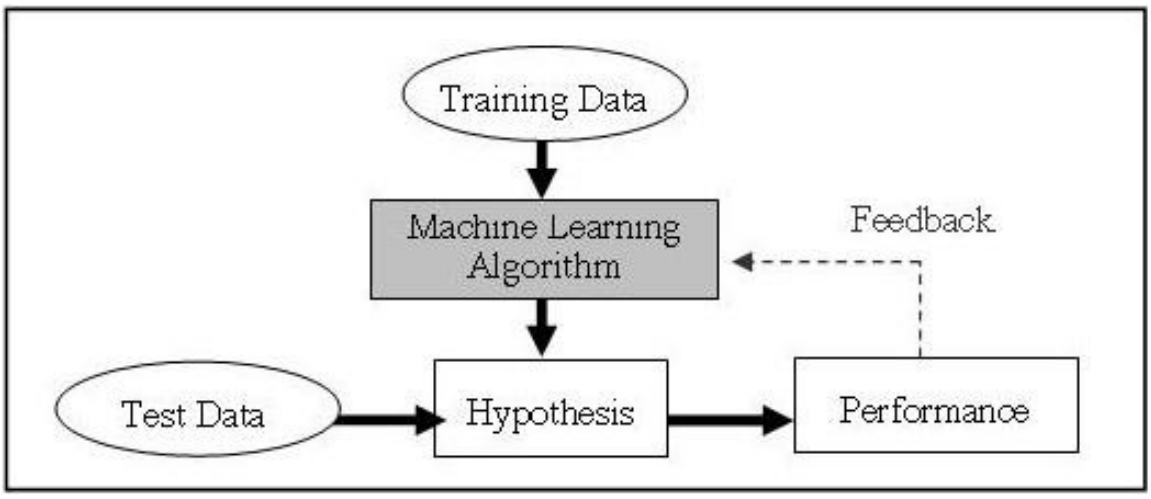
Accuracy:The rate of correct (incorrect) predictions made by the model over a data set (cf. coverage). Accuracy is usually estimated by using an independent test set that was not used at any time during the learning process. More complex accuracy estimation techniques, such as cross-validation and the bootstrap, are commonly used, especially with data sets containing a small number of instances.

Classifier: A classifier is the implementation of a machine learning algorithms which is learned by the machine using the training data fed from a given dataset.

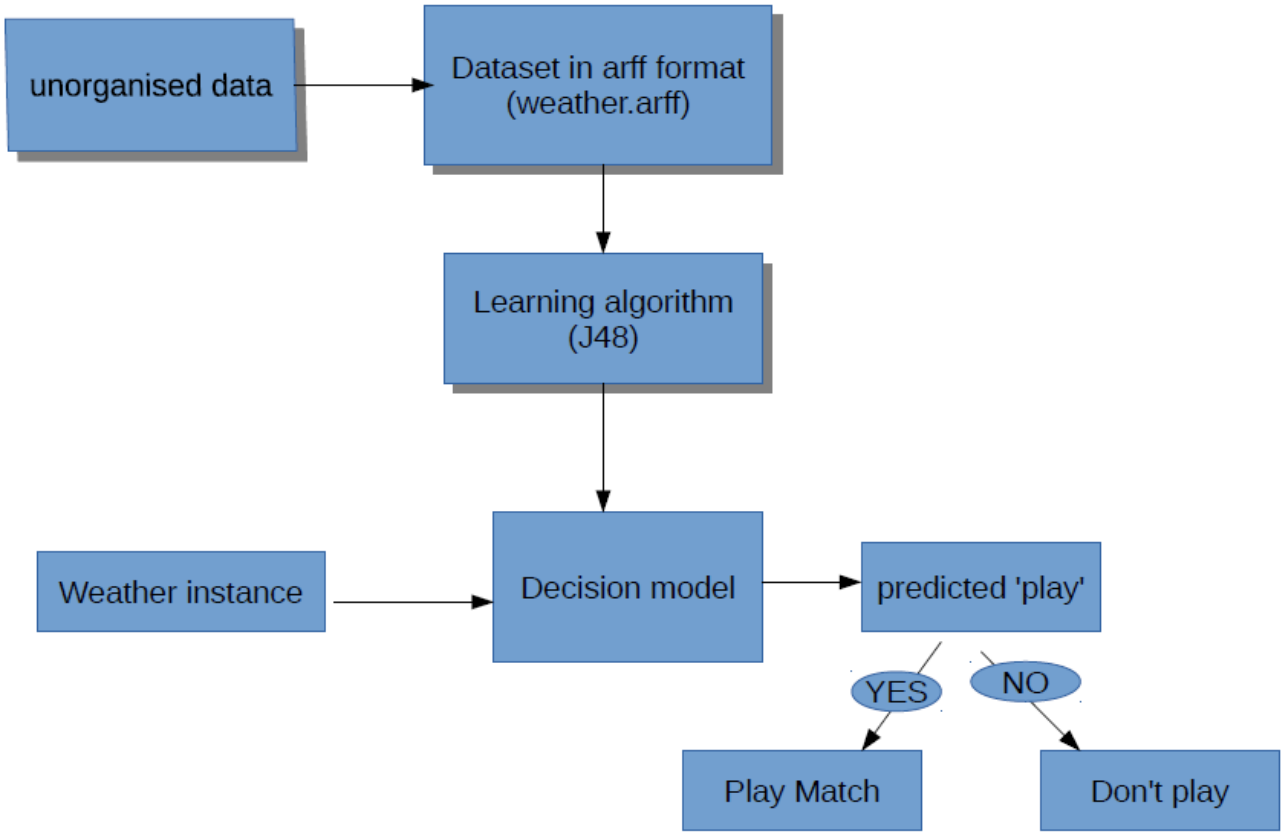
Cross-validation:A method for estimating the accuracy (or error) of an inducer by dividing the data into k mutually exclusive subsets (the ``folds'') of approximately equal size. The inducer is trained and tested k times. Each time it is trained on the data set minus a fold and tested on that fold. The accuracy estimate is the average accuracy for the k folds.

WEKA:Weka stands for Waikato Environment for Knowledge analysis.It is a library for machine learning which contains many algorithms and classes which are very essential and helpful in writing machine learning functions.It has to the ability to process datasets in the proper format required.

The below given block diagram shows all the different components of a machine learning system and how the whole thing works.



SYSTEM DESIGN



The dataset collected is usually in unorganised form and is converted to arff format.This data is our trianing data which is then passed to the J48 classifier which builds a decision tree(model) based on this training data.Now, when the testing data consisting of one or more weather instance for which the value of play weather attribute is unknown is passed to this decison model, the classifier predicts those unknown values as either "yes" or" no."

IMPLEMENTATION

```
// PredictPlay.java

import java.awt.BorderLayout;

import java.io.BufferedReader;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.util.Scanner;


import weka.classifiers.Classifier;

import weka.classifiers.Evaluation;

import weka.classifiers.evaluation.NominalPrediction;

import weka.classifiers.trees.J48;

import weka.core.Attribute;

import weka.core.FastVector;

import weka.core.Instance;

import weka.core.Instances;

import weka.gui.treevisualizer.PlaceNode2;

import weka.gui.treevisualizer.TreeVisualizer;


/**

 * This program reads from arff dataset file or from the user and predicts the

 * value of the unlabeled attribute .It trains the model using 10 cross split valida-
tion

 * and calculates the accuracy of the built model and also displays the built de-
cision tree
```

```
 */
public class PredictPlay {

	private static final int numberOfFolds = 10; // n-cross validation

	private static Scanner sc = new Scanner(System.in); // to input values

	/**
	 * Reads a dataset file
	 *
	 * @param filename
	 *            from which the data has to be read
	 * @return inputReader
	 */
	public static BufferedReader readDataFile(String filename) {
		BufferedReader inputReader = null;

		try {
			inputReader = new BufferedReader(new
FileReader(filename));
		} catch (FileNotFoundException ex) {
			System.err.println("File not found: " + filename);
		}

		return inputReader;
	}
```

```
/**
 * returns Evaluation object for predictions
 *
 * @param model
 * @param trainingSet
 * @param testingSet
 * @return
 * @throws Exception
 */

public static Evaluation evaluate(Classifier model, Instances trainingSet,
                Instances testingSet) throws Exception {
        Evaluation evaluation = new Evaluation(trainingSet);
        model.buildClassifier(trainingSet);
        evaluation.evaluateModel(model, testingSet);
        return evaluation;
}

/**
 *
 * @param predictions
 * @return
 */
public static double calculateAccuracy(FastVector predictions) {
        double correct = 0;
```

```
        for (int i = 0; i < predictions.size(); i++) {

NominalPrediction np = (NominalPrediction) predictions.elementAt(i);
                if (np.predicted() == np.actual()) {
                        correct++;
                }
        }

        return 100 * correct / predictions.size();
}

public static Instances[][] crossValidationSplit(Instances data,
                int numberOfFolds) {
        Instances[][] split = new Instances[2][numberOfFolds];

        for (int i = 0; i < numberOfFolds; i++) {
                split[0][i] = data.trainCV(numberOfFolds, i);
                split[1][i] = data.testCV(numberOfFolds, i);

        }

        return split;
}

public static void main(String[] args) throws Exception {
```

```
BufferedReader datafile = readDataFile("weather.arff");
BufferedReader testfile = readDataFile("predictweather.arff");

Instances data = new Instances(datafile);
Instances testdata = null;

System.out
        .println("Do you want to\n [1] Use your own Instances \n
[2] Label an unlabeled dataset from a file \n [3] Use sample file ");
int choice = sc.nextInt();
switch (choice) {
case 1:
    testdata = addInstances();
    // System.out.println(testdata.toString());
    break;
case 2:
    System.out.println("Enter filename : ");
    String str = sc.next();

    testfile = readDataFile(str);
    testdata = new Instances(testfile);

    break;
case 3:
    testdata = new Instances(testfile);
    break;
```

```
        default:
                System.out.println("Wrong Option ! Exiting !");
                System.exit(0);
                break;
        }

        // setting the class index
        data.setClassIndex(data.numAttributes() - 1);
        testdata.setClassIndex(testdata.numAttributes() - 1);

        // Using a J48() classifier
        Classifier model = new J48();

        // Do 10-split cross validation
        Instances[][] split = crossValidationSplit(data, 10);

        // Separate split into training and testing set
        Instances[] trainingSplits = split[0];
        Instances[] testingSplits = split[1];

// Collect every group of predictions for current model in a FastVector
        FastVector predictions = new FastVector();

        System.out.println();

        // For each training-testing split pair, train and test the classifier
```

```
for (int i = 0; i < trainingSplits.length; i++) {
        Evaluation validation = evaluate(model, trainingSplits[i],
                        testingSplits[i]);
        model.buildClassifier(trainingSplits[i]);
        predictions.appendElements(validation.predictions());
}

// find the correct predicted percentage
double accuracy = calculateAccuracy(predictions);

System.out.println("Accuracy of the model is "
                + String.format("%.2f%%", accuracy) + "\n");

// Classification based on the given training data
for (int i = 0; i < testdata.numInstances(); i++) {
        Instance instance = testdata.instance(i);
        System.out.print("instance : " + (i + 1) + " ");

        System.out.println("");
        if (model.classifyInstance(instance) == 0.0) {
                System.out.println("yes");
        } else if (model.classifyInstance(instance) == 1.0) {
                System.out.println("no");
        } else {
        }
```

```
            }

            // display
            J48 cls = new J48();
            cls = (J48) model;
            System.out.println("Do you want to view the J48 tree ?? {yes/no}");

            String DISPLAY = sc.next();
            if (DISPLAY.equals("yes") || DISPLAY.equals("y") ||
DISPLAY.equals("Y")
                        || DISPLAY.equals("YES"))
                display(cls);
            else
                System.exit(0);

        }

        /**
         * Build a fresh test data
         *
         * @return customData
         */
        private static Instances addInstances() {

            // Numeric
```

```
Attribute Attribute1 = new Attribute("MaxTemp");
// Numeric
Attribute Attribute2 = new Attribute("Mean_Temp");
// Numeric
Attribute Attribute3 = new Attribute("MinTemp");
// Numeric
Attribute Attribute4 = new Attribute("MaxHumidity");
// Numeric
Attribute Attribute5 = new Attribute("MeanHumidity");
// Numeric
Attribute Attribute6 = new Attribute("MinHumidity");
// Numeric
Attribute Attribute7 = new Attribute("WindSpeed");
// Numeric
Attribute Attribute8 = new Attribute("CloudCover");

// Declare play [class attribute] along with its values
FastVector NominalValues = new FastVector(2);
NominalValues.addElement("yes");
NominalValues.addElement("no");
Attribute Attribute9 = new Attribute("play", NominalValues);

// Create the attribute list
FastVector AttributesList = new FastVector(9);
AttributesList.addElement(Attribute1);
AttributesList.addElement(Attribute2);
```

```
        AttributesList.addElement(Attribute3);
        AttributesList.addElement(Attribute4);
        AttributesList.addElement(Attribute5);
        AttributesList.addElement(Attribute6);
        AttributesList.addElement(Attribute7);
        AttributesList.addElement(Attribute8);
        AttributesList.addElement(Attribute9);

        System.out.println("Enter the max number of instances that you
want to add :");
        int NumInstances = sc.nextInt();

        // Create the empty training set
Instances customData = new Instances("My_New_Weather", AttributesList,
                NumInstances);

        // Set class index - "play"
        customData.setClassIndex(customData.numAttributes() - 1);

        int choice = 0;
        do {
           int Htemp, Ltemp, MeanHumd, Meantemp, HHumd,
               LHumd, WindS, CloudCover;

               // Enter the values for the instance
               System.out.println("Enter High Temp (in F)");
```

```
        Htemp = sc.nextInt();

        System.out.println("Enter Low Temp (in F)");
        Ltemp = sc.nextInt();

        Meantemp = (Htemp + Ltemp) / 2;

        System.out.println("Enter the Max Humidity ");
        HHumd = sc.nextInt();

        System.out.println("Enter the Min Humidity ");
        LHumd = sc.nextInt();

        MeanHumd = (HHumd + LHumd) / 2;

        System.out.println("Enter Wind Speed (in km/h)");
        WindS = sc.nextInt();

        System.out.println("Enter the CLoudCover (0-8 Oktas) ");
        CloudCover = sc.nextInt();

        // Create the instance
        Instance iNew = new Instance(9);
        iNew.setValue((Attribute) AttributesList.elementAt(0), Htemp);
        iNew.setValue((Attribute) AttributesList.elementAt(1), Ltemp);
    iNew.setValue((Attribute) AttributesList.elementAt(2), Meantemp);
```

```
            iNew.setValue((Attribute) AttributesList.elementAt(3), HHumd);
            iNew.setValue((Attribute) AttributesList.elementAt(4), LHumd);
        iNew.setValue((Attribute) AttributesList.elementAt(5), MeanHumd);
            iNew.setValue((Attribute) AttributesList.elementAt(6), WindS);
        iNew.setValue((Attribute) AttributesList.elementAt(7), CloudCover);

            // add the instance
            customData.add(iNew);

        System.out.println("Do you want to continue [1 for yes , 0 for no ] :");
            choice = sc.nextInt();
        } while (choice == 1);

        // add the created instance set
        System.out.println("This is your created dataset");
        System.out.print(customData.toString());

        // return customData
        return customData;
}

/**
 * Displays the model in tree format
 * @param a  J48 classifier modal
 * @throws Exception
 */
```

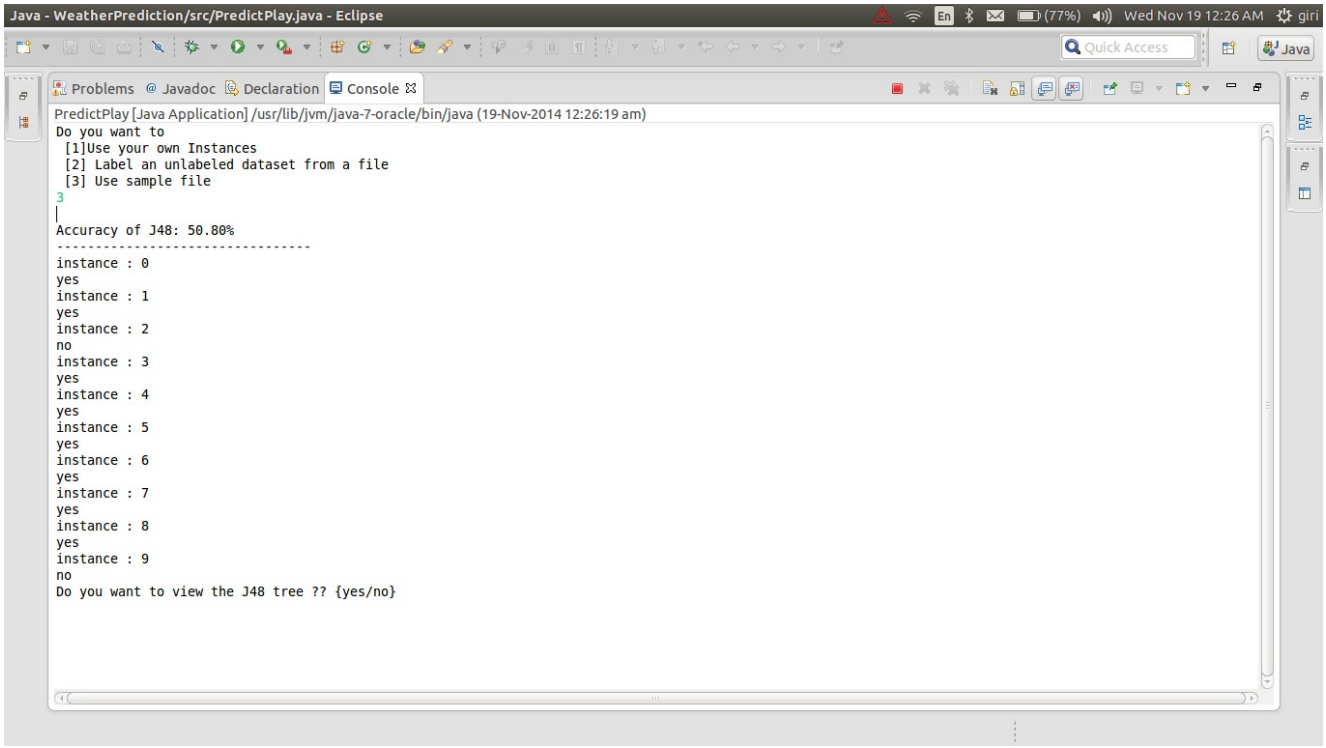
```
    private static void display(J48 cls) throws Exception {
        // TODO Auto-generated method stub
        final javax.swing.JFrame jf = new javax.swing.JFrame
        ("Weka Classifier Tree Visualizer: J48");
        jf.setSize(1200, 1000);
        jf.getContentPane().setLayout(new BorderLayout());
        TreeVisualizer tv = new TreeVisualizer(null, cls.graph(),
                new PlaceNode2());
        jf.getContentPane().add(tv, BorderLayout.CENTER);
        jf.addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosing(java.awt.event.WindowEvent e) {
                jf.dispose();
            }
        });

        jf.setVisible(true);
        tv.fitToScreen();
    }
}
```

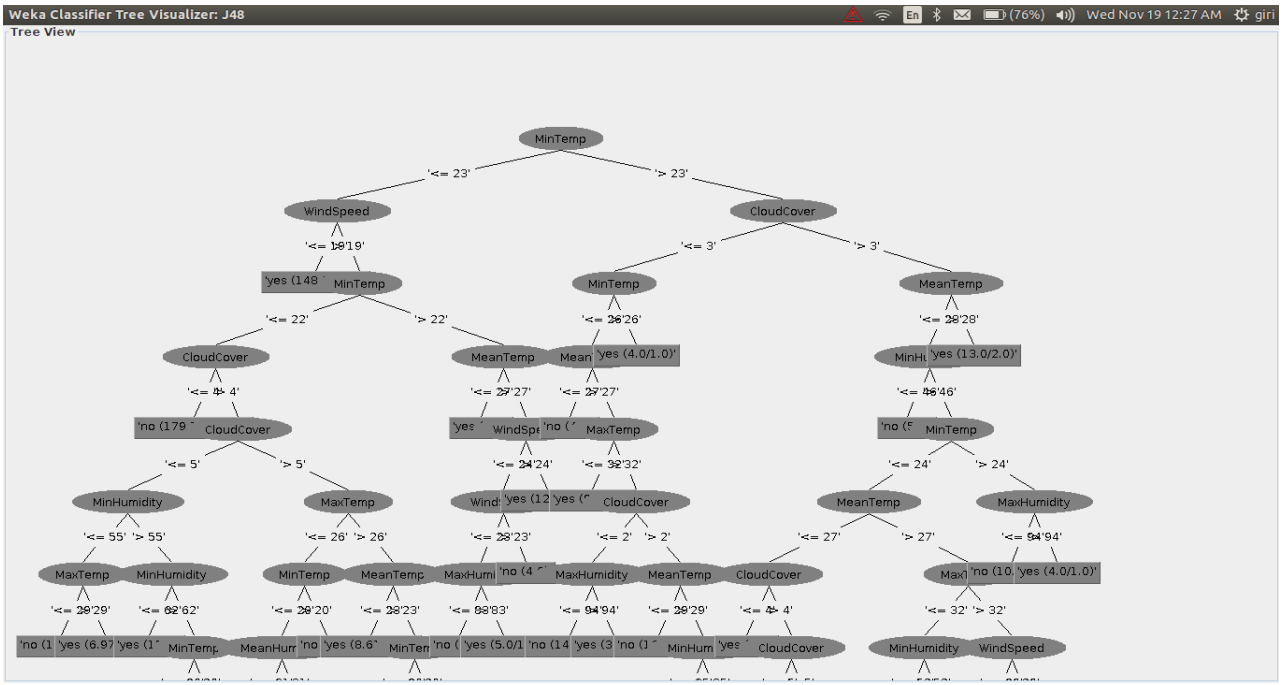
RESULTS

The decision tree was built using the training data set. The values for the play attribute were succesfully predicted for each instance in the testing dataset according to the decision tree built.Thereafter, the accuracy of the prediction was calculated.

A . Predicted values



B. Decision Tree



CONCLUSION

We concluded that using the concepts of machine learning one could easily predict values for a given attribute in a dataset if the values for the other attributes are known.

These prediction values could be further used for other purposes for e.g in our case the value of the play attribute could be used to decide whether a cricket match should be held in a particular city on a particular day or not given we have the weather data for that city.

FUTURE ENHANCEMENTS

We aim to extend our project by collecting other datasets like data of diabetic patients ,heart patients and predict the attributes in those datasets by applying those concepts which we have learnt from this project. And also doing other types of classifications like multiclass classification etc.

We also aim to extend project to some basic concepts of data mining and artificial intelligence, if possible.

REFERENCES

[1] "Introduction to Weka" , https://www.cs.auckland.ac.nz/courses/compsci367s1c/tutorials/IntroductionToWeka.pdf

[2] Zdravko Markov,"An Introduction to the WEKA Data Mining System" , http://www.cs.ccsu.edu/~markov/weka-tutorial.pdf

[3]"Hello World program in WEKA " , http://www.programcreek.com/2013/01/a-simple-machine-learning-example-in-java/

[4] "Using WEKA in Java code" , http://weka.wikispaces.com/Use+WEKA+in+your+Java+code

[5] Ian H. Witten and Eibe Frank on "Introduction to Machine Learning", Data MIning :Practical Machine Learning