`

Business Intelligence and Data Analytics – Prof. Anu Thomas

CST8390 - Lab 6
*Classification by **Decision Trees***

Name: Min Li - Id: 040930563

Due Date: Week 8 in corresponding lab section.

Introduction

The goal of this lab is to perform classification on Diabetes dataset using **Decision Trees**.

Steps:

1. Open Diabetes dataset in **text editor** (from datasets that came with **Weka**). Read the information about the file. Fill in the following information (should be typed in).

    a. Number of **instances**: 768.

    b. Number of **attributes**: 8 plus class

    c. List of **attributes** (NOT abbreviation, should be **typed** in):

    1. Number of times pregnant

    2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test

    3. Diastolic blood pressure (mm Hg)

    4. Triceps skin fold thickness (mm)

    5. 2-Hour serum insulin (mu U/ml)

    6. Body mass index (weight in kg/(height in m)^2)

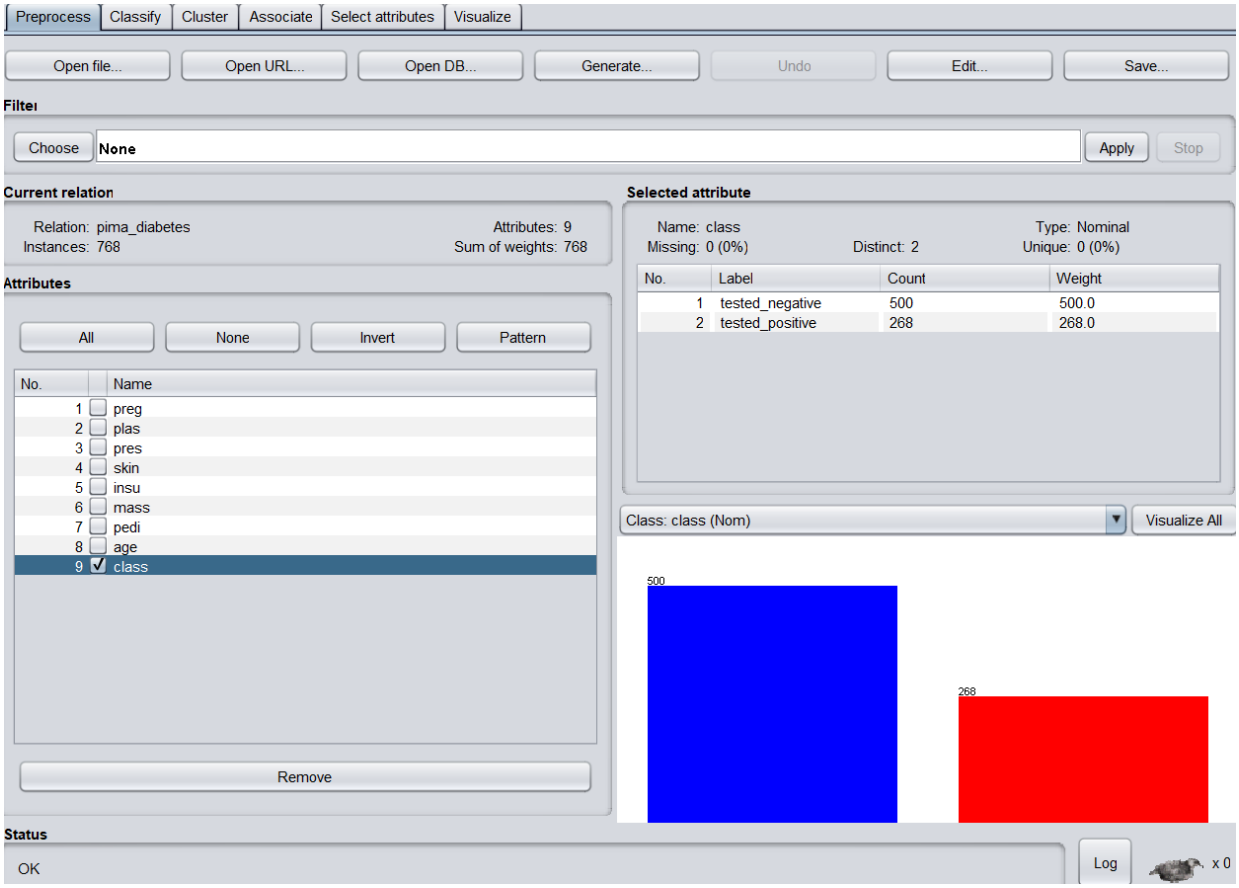
    7. Diabetes pedigree function

8. Age (years)

9. Class variable (0 or 1)

d. Class **labels** and their relabelled values: class value 1 is interpreted as "tested positive for diabetes", Number of **instances** for each class label: 0 500, 1 268.

2. Load the dataset in **Weka**. Take a **screenshot** and paste it below that shows class distribution.



3. Click on the "Choose" button on "Classify" tab and select **J48** from "trees". It is the implementation of the **C4.5 algorithm** which uses entropy to create a decision tree.

4. For testing the classification **accuracy**, make sure that "(Nom) Class" is selected, and cross-validation has 20 folds (Make sure that seed = 1). Click start and you should see a textual version of the **decision tree**.

a. **Copy and paste** the confusion matrix here:

```
=== Confusion Matrix ===

   a   b   <-- classified as
 421  79 |   a = tested_negative
 112 156 |   b = tested_positive
```

.

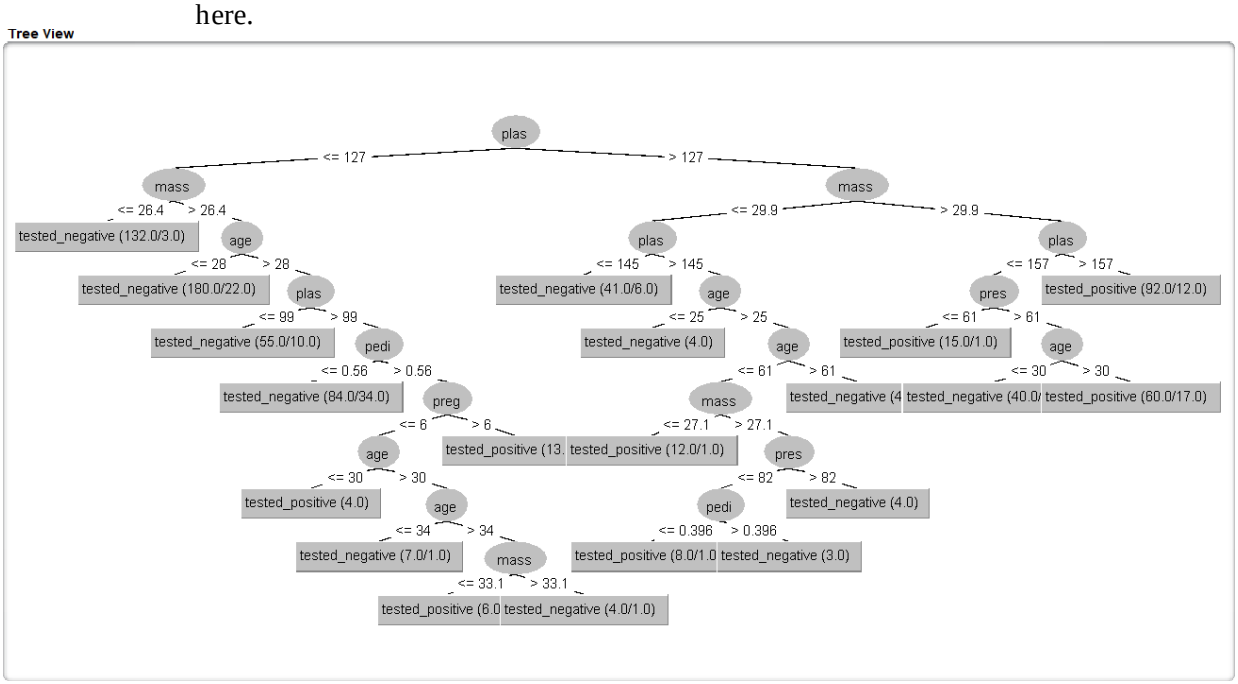
b. Number of **leaves**: 20.

c. **Size** of the tree: 39.

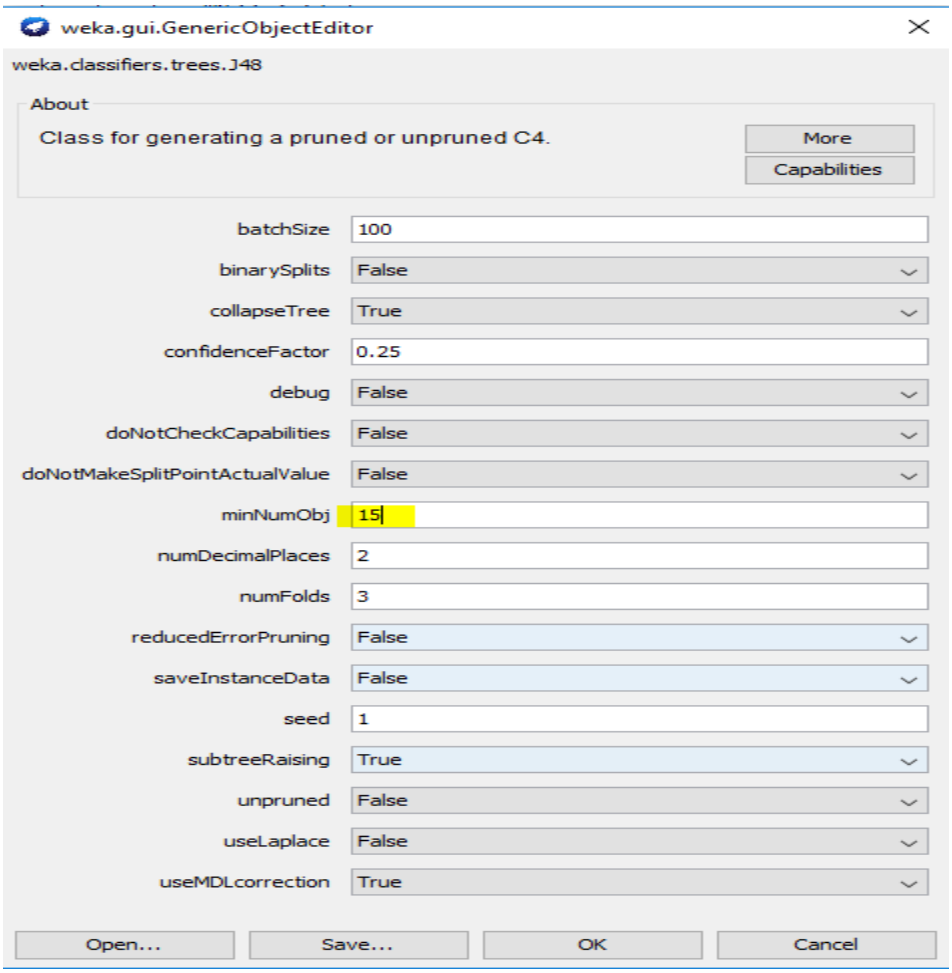
d. **Correctly** classified instances: 577.

5. **Right click** on the result buffer and select "Visualize tree".

- From the **new window**, make it full screen and then **right-click** on the window and select "auto scale". It will draw the tree so that it's wide enough to read the text.
- You might have to **right-click** again on the screen and "Center on Top Node". You can use the mouse to pan around the tree to see all the decision splits.
- **Right-click** again on the screen and select "Fit to Screen". Here you can see the tree all in one place, but the text might be hard to read.
- Have this window open for your lab demonstration. Also, take a **screenshot** and paste it here.



6. Set minNumObj to 15 in the settings window of the classifier, as shown below (This means that don't continue splitting if the nodes get very small. Default value is 2):



Run the classifier with this setting and fill in the following information:

a. **Copy and paste** the confusion matrix here:

```
=== Confusion Matrix ===

   a   b   <-- classified as
 411  89 |   a = tested_negative
 105 163 |   b = tested_positive
```

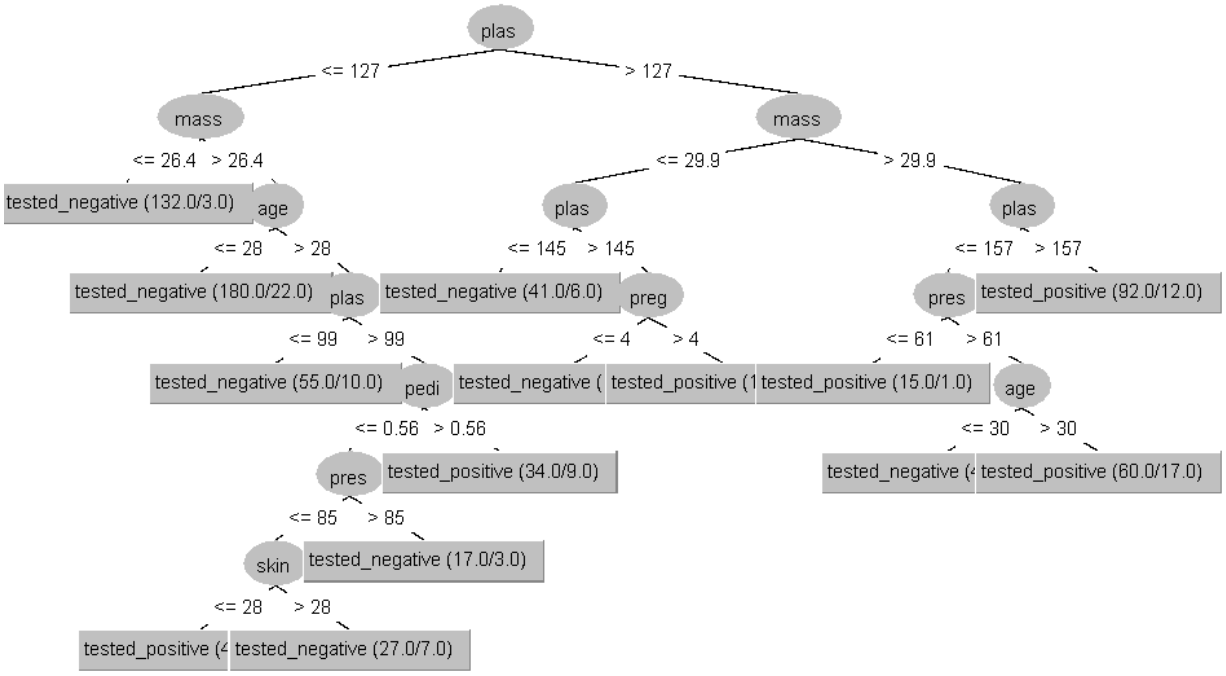
.

b. Number of **leaves**: 14.

c. Size of the **tree**: 27.

d. **Correctly classified** instances: 574.

7. Take a **screenshot** of the tree and paste it here (from "Visualize tree").



8. Now, **turn off pruning** by setting unpruned property to **True** (also, se minNumObj to 5, seed = 1) in the settings window of the classifier, as shown below (this means that we are not reducing the size of the tree even if it is not giving much value for the task):

   Run the classifier with this setting and fill in the following information:

   a. Copy and paste the **confusion matrix** here:

```
=== Confusion Matrix ===

   a   b   <-- classified as
 414  86 |   a = tested_negative
 105 163 |   b = tested_positive
```

   b. Number of **leaves**: 14.

   c. **Size** of the tree: 27.

   d. **Correctly classified** instances: 577.

9. Run the classifier again with unpruned property to **True** and minNumObj to 15, and fill in the answers for the questions below:

a. Copy and paste the **confusion matrix** here:

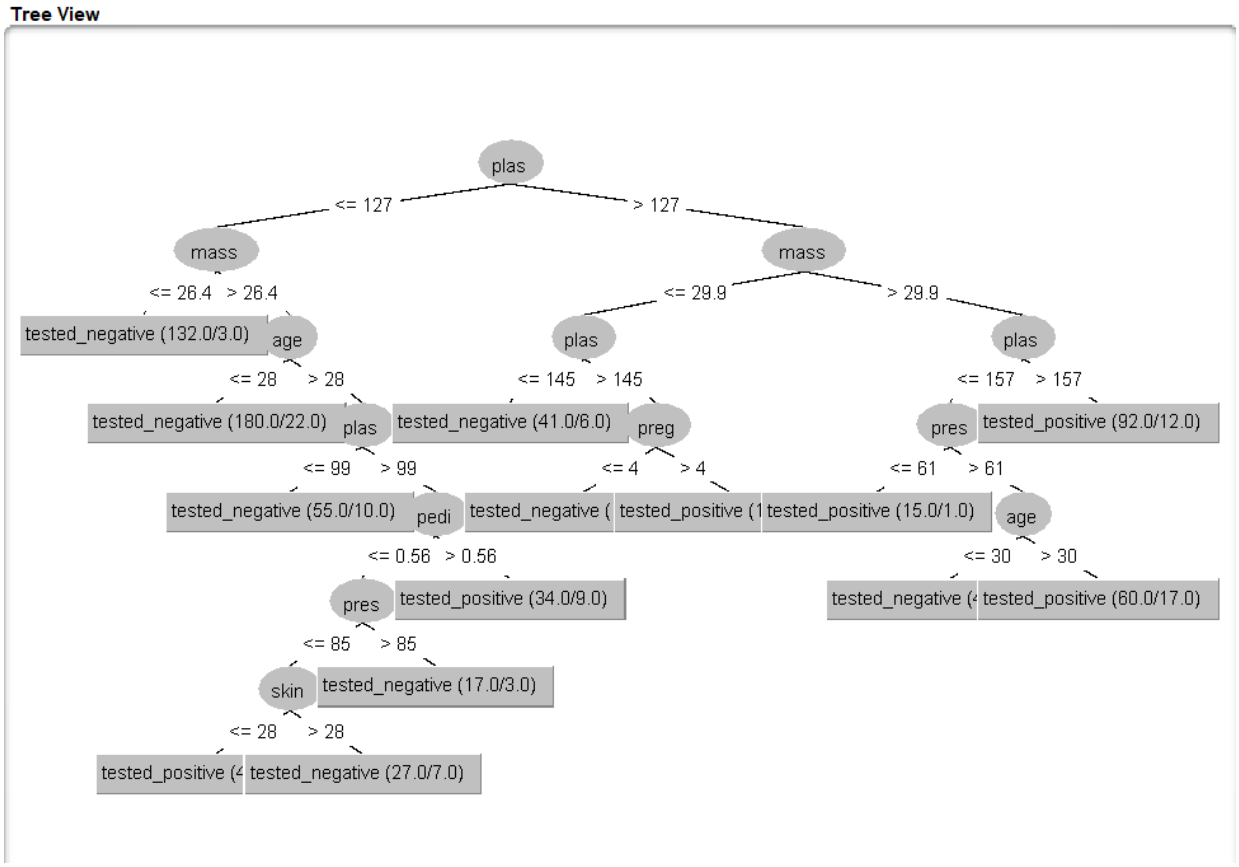
```
=== Confusion Matrix ===

   a   b   <-- classified as
 413  87 |   a = tested_negative
 106 162 |   b = tested_positive
```

b. Number of **leaves**: 14.

c. **Size** of the tree: 27.

d. **Correctly classified** instances: 575.

10. Take a screenshot of the tree and paste it here (from "Visualize tree").



11. **Decision trees** have a problem with **overfitting**.

- One way to correct **overfitting** is with using **random forests**.
- This uses many decision trees, each built with **random subset** of the data.

- When a new item is going to be classified, the trees all vote when classifying each data item, with the majority deciding the **final answer**.
- The **probability** of an outlier being selected to be in several of the trees is highly unlikely so they will have less impact on the final classification.

To run the random forest algorithm, click the "Choose" button and select "Random Forest". Select Run the algorithm and paste the **confusion matrix** here:

a. Details of **random forest**: Bagging with 100 iterations and base learner

b. **Time** taken to build model: 0.27 seconds.

c. **Correctly classified** instances: 578.

Show your **answers** to the lab professor when you are done.

REMEMBER:

You should be ready with all your results in the **result pane**, and should show trees for steps **5**, **7** and **10**.

FOR YOUR ANALYSIS:

**** Option 1**: Explain with your own words what is a **Decision Tree** and where to use it.*

**** Option 2**: Explain how to decide what is the strategy to **decide** how is the better parameter to use in a **root node**.*

Ottawa, Feb 2020.

Option 1:

A decision tree is a tree-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules.

After reading a decision tree, we can get some results or make some predictions. Or the goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data).