COMP 421 – Final: SMS Spam Detection

Date Assigned: April 19, 2021

Completion Date: May 4, 2021 11:55pm

Description

PART I

In this project, you will create a program that automatically predicts whether an SMS message is spam or ham (not ham).

This project will be split into your midterm and final.

For Part I, you will create a PHP program that reads SMS messages from a database table, converts the SMS messages into features, and put those features in a CSV file.

The SMS messages along with their class label (spam, ham) are in a file named **spam.sql**. The first column is the class label spam or ham. Ham means the text message is not spam. The second column is the text message.

Specification

1. Create a MySQL database named predict_spam.
2. Use the SQL statements in the spam.sql file to create a table named spam. The statements in the SQL file will also insert data into the spam.sql table.
3. Create a PHP file named **compute_features.php**.
4. The PHP file should read each row in the database, compute features, and print out a csv file with each feature.

You should compute the following features:

- **doesHaveLinks**
    - This feature is True if a SMS message has links and False if a SMS message doesn't have links.
- **doesHaveSpammyWords**
    - This feature is True if the SMS contains spammy words and False if a SMS message does not have spammy words
    - To determine what words are spammy, look through the dataset and pay close attention to curse words and any words that are in the spam category but not in the ham category.
    - You should choose 10 spammy words.
- **lengthOfText**

    - The number of characters including spaces in the text message.

These features should be printed to a csv file named **features.csv.**

The file should have the following columns: **doesHaveLinks, doesHaveSpammyWords, lengthOfText**, **class label.**

The class label is SPAM or HAM.

At a minimum, your program should have:

- A main function
- A function for each feature
    - doesHaveLinks(email)
    - doesHaveSpammyWords(email)
    - lengthOfText(email)

- A function for writing the features to the CSV

You can create more functions if you want to.

The name of your PHP file should be **compute_features.php**.

PART II

For Part II, you will:

1. Use Weka to read features from the csv file created in Part I to create a decision tree diagram.
2. Convert the decision tree diagram into PHP using conditionals (if/elif/else).

Specification

At a minimum, your program should have:

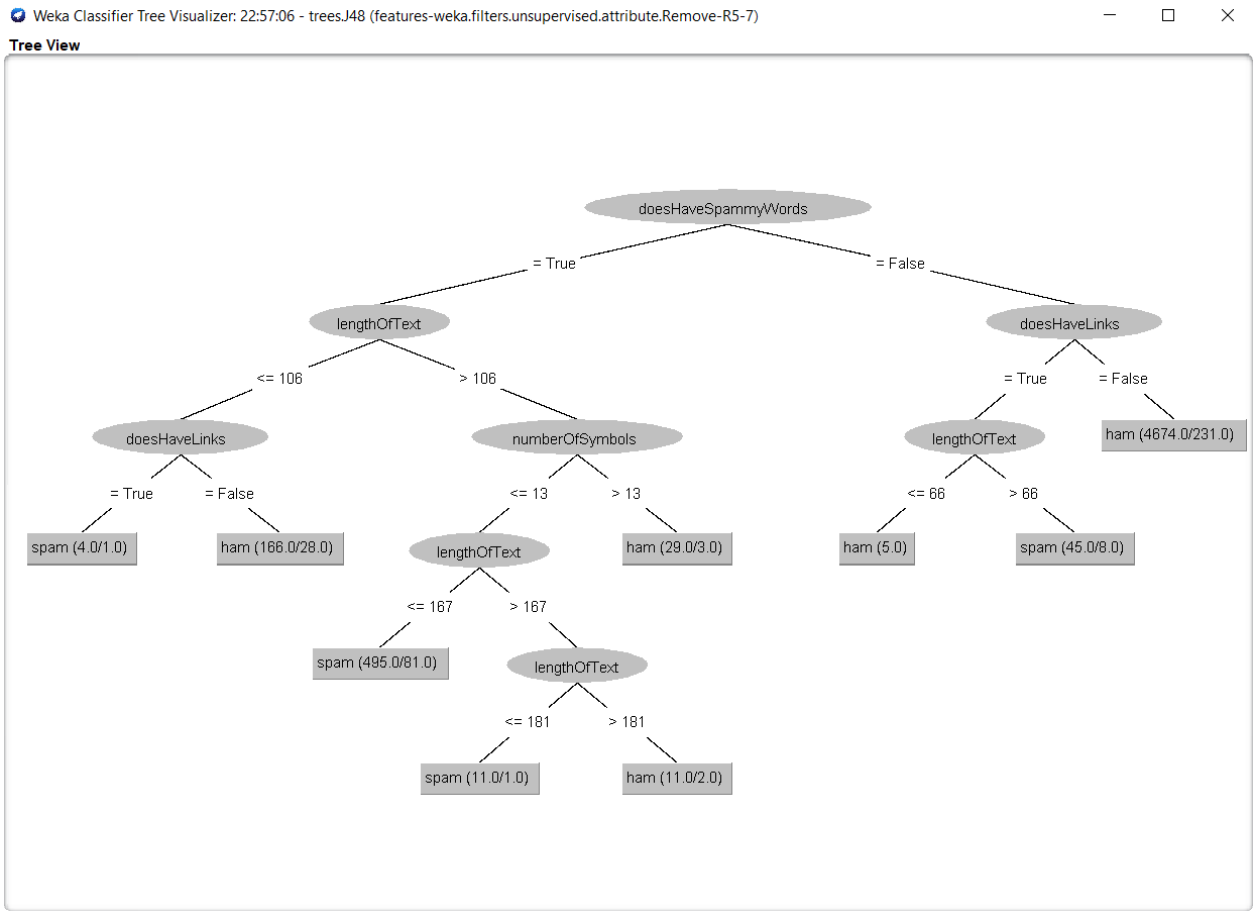
- A main function
- A function named make_prediction
    - This function takes as input the features and returns the prediction (this is where the decision tree code goes)
- The functions from compute_features.php that compute features

Create a html form that contains a textarea. This textarea is where you can copy and paste email messages. The html form should also contain a button. Place these input controls inside of an html form.

The name of the html file should be **predict_spam.html**. The html form should **POST** to a file named **predict_spam.php**.

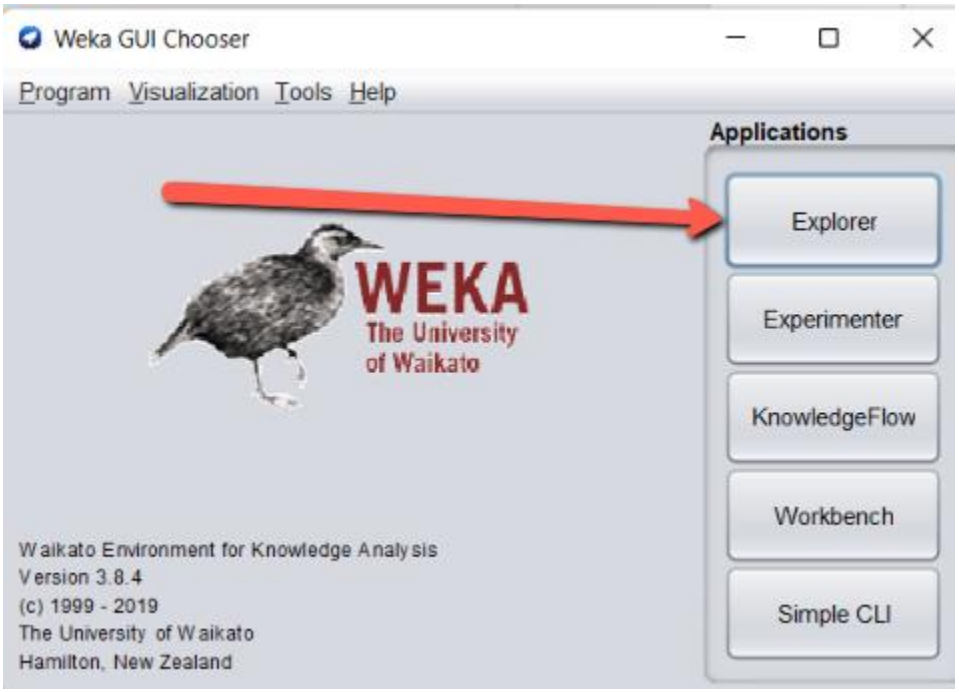
predict_spam.php will get the email message, convert the email message into features (using the functions you created in **compute_features.php**), use the if/else statements from the decision tree to predict whether the email is spam or ham. The program should print spam or ham.

You can copy and paste the functions to create features from **compute_features.php** into **predict_spam.php**.

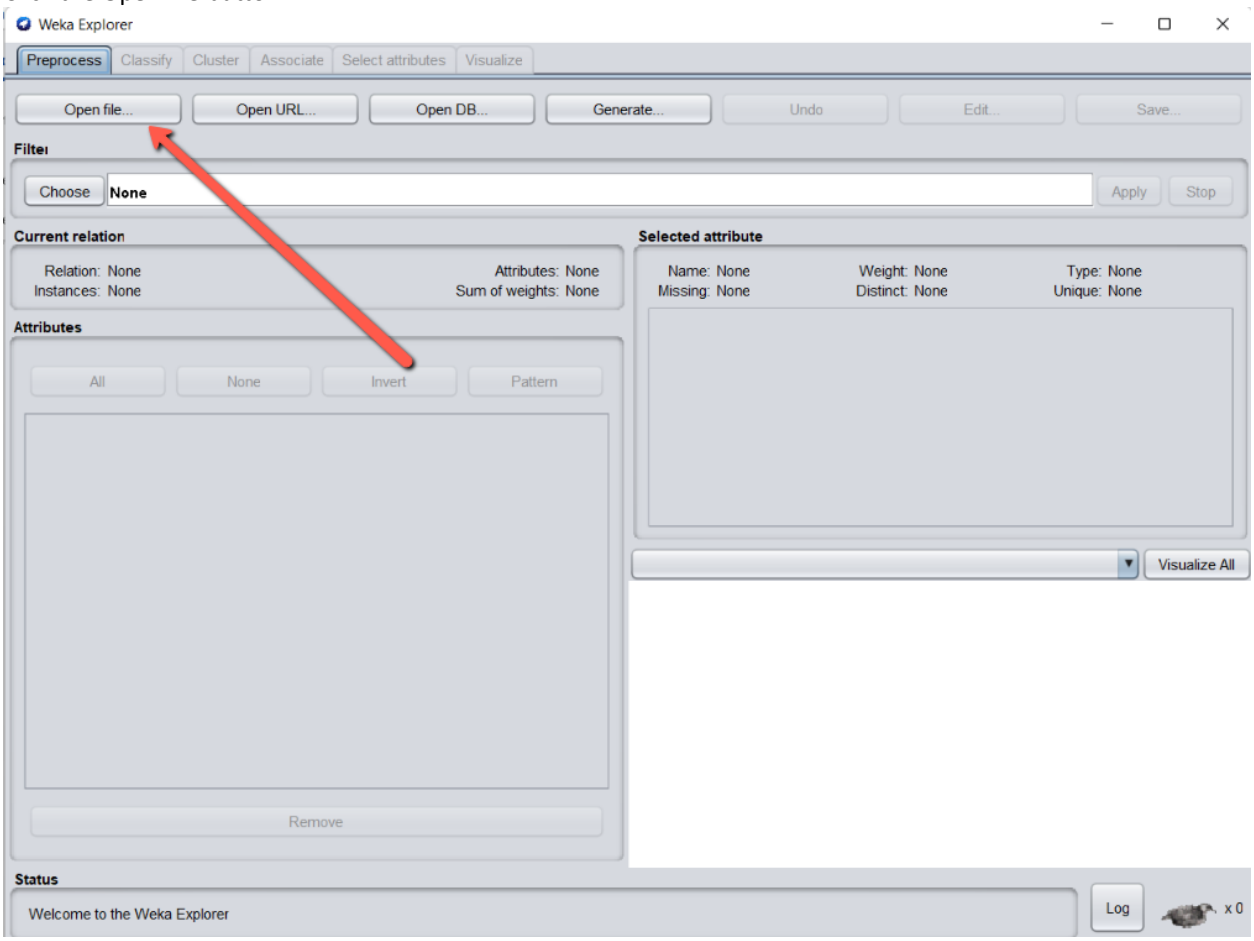


To generate the decision tree:

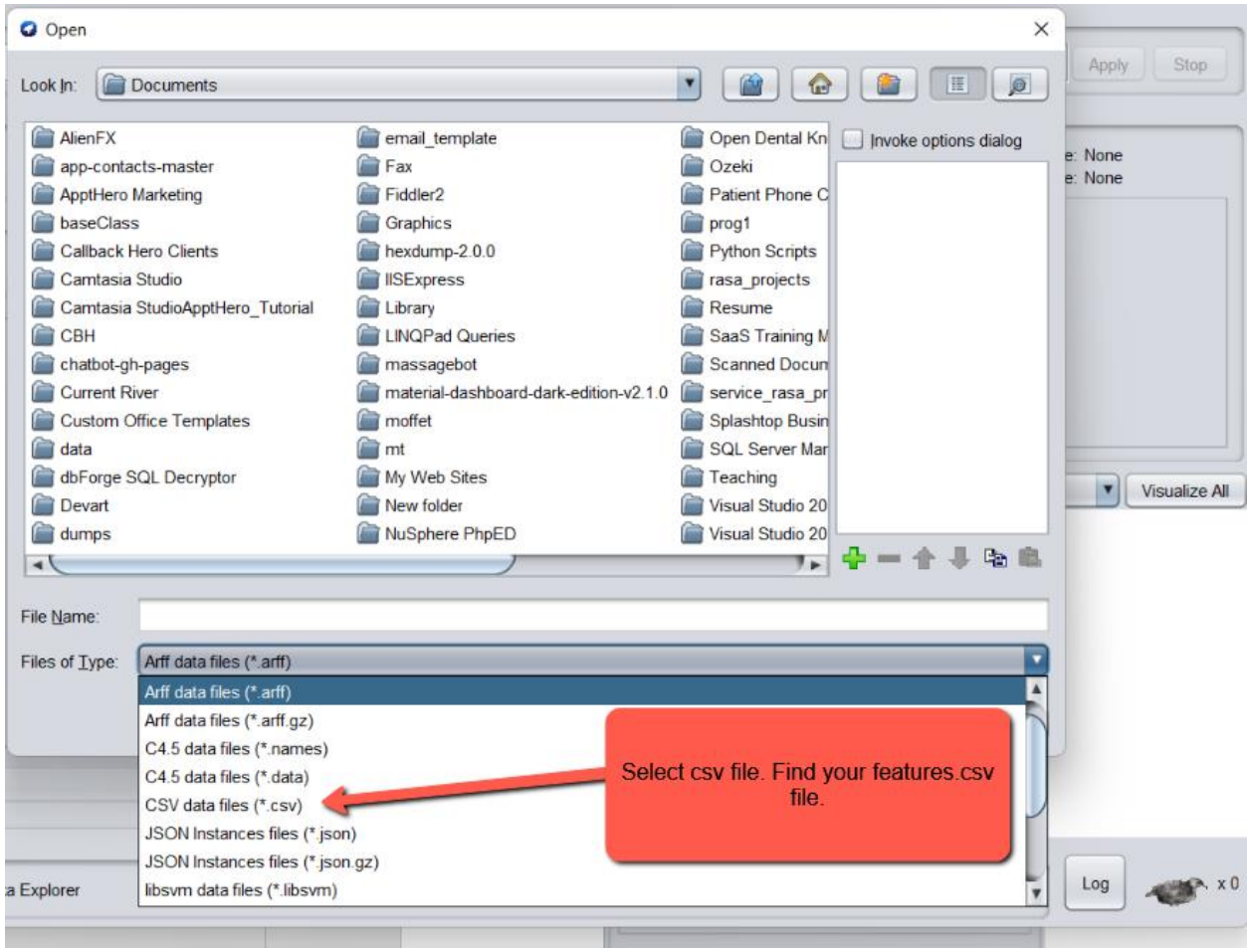
1. Open Weka
2. Click the Explorer button



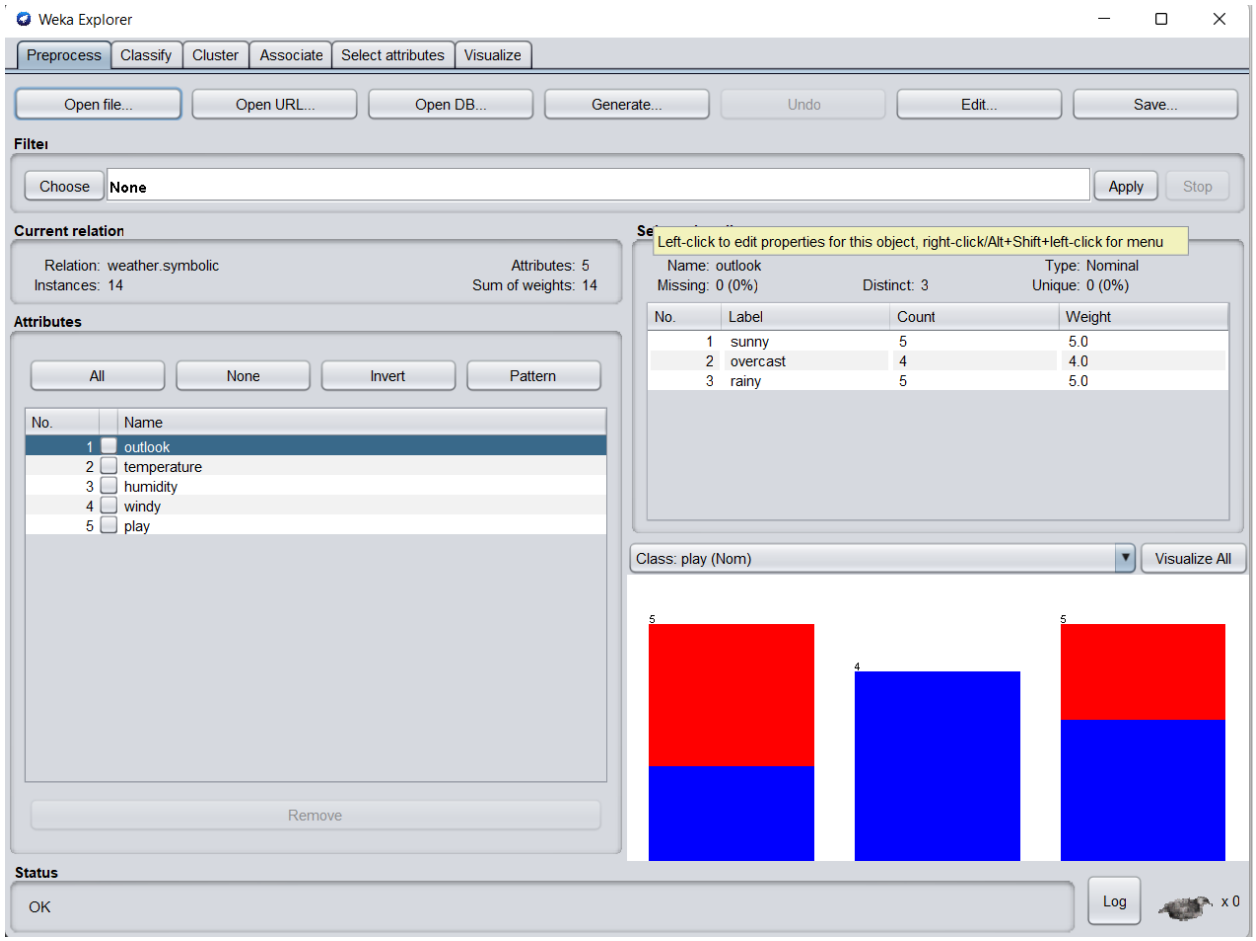
3. Click the Open file button



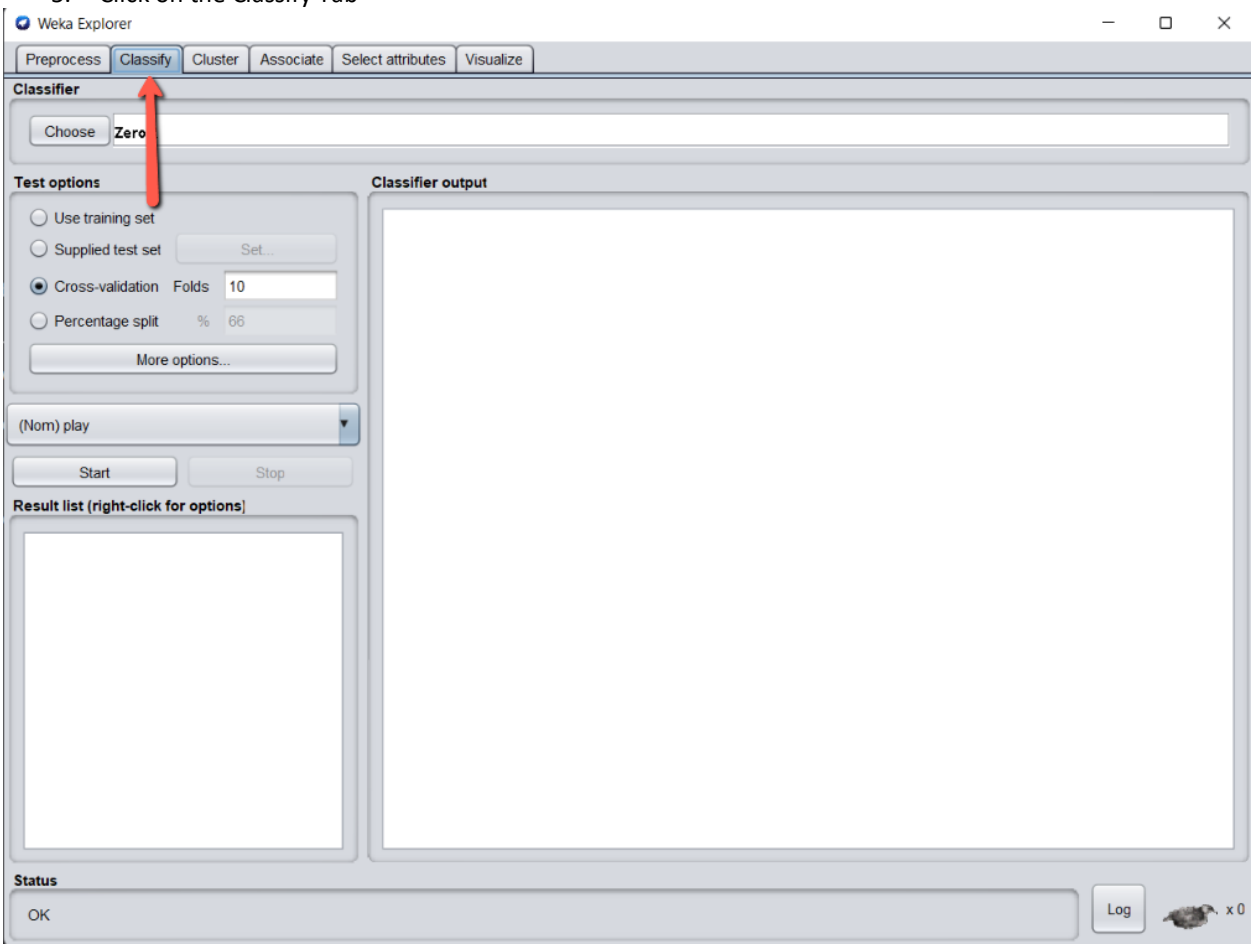
4. Find the file features.csv and select it



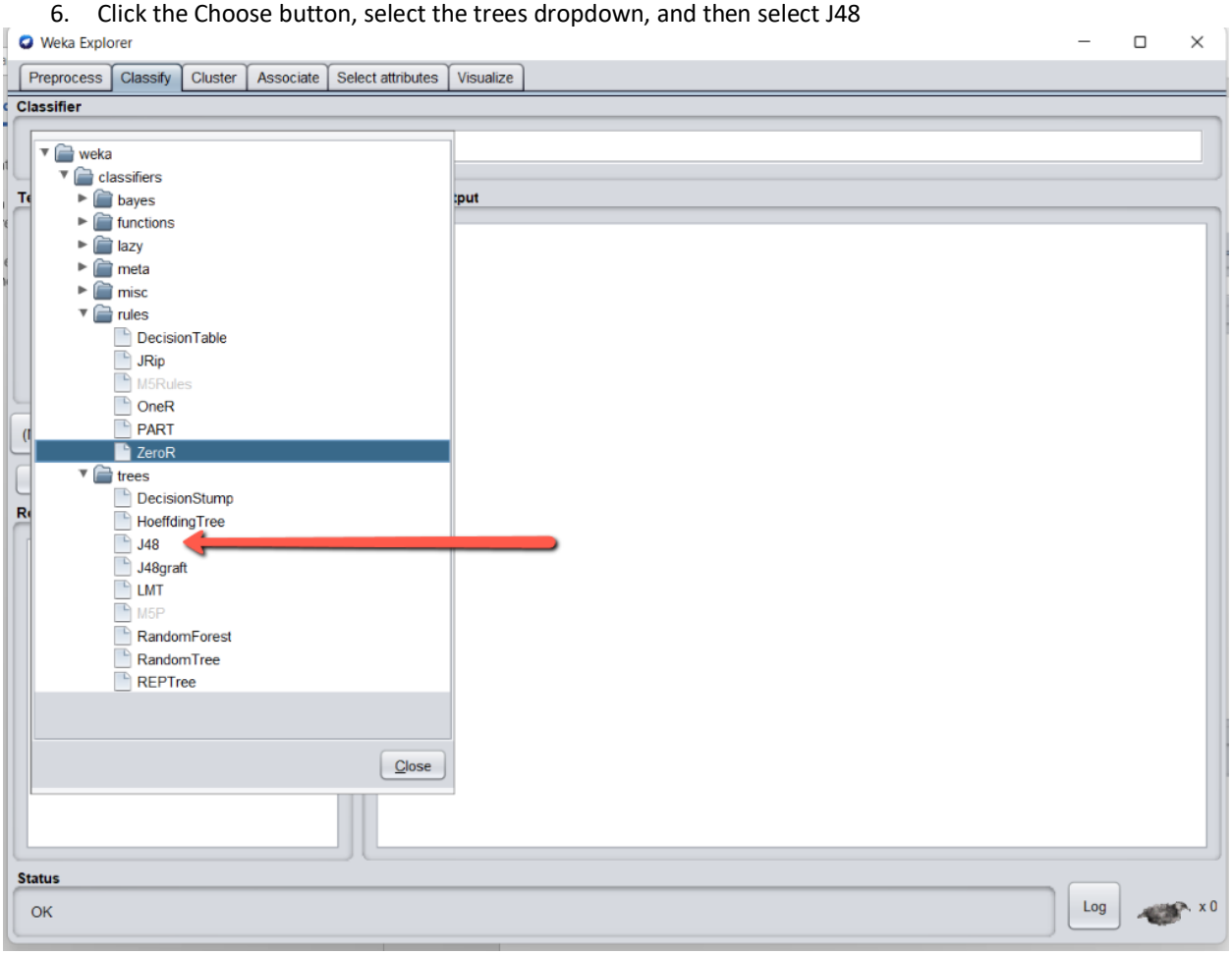
Your Weka program should look similar to this, but with your features.



5. Click on the Classify Tab



6. Click the Choose button, select the trees dropdown, and then select J48



7. Make sure class_label is selected (right above the Start button)
8. Press the Start button

9. In the results pane, right click a result, and select Visualize tree