INLS 623 – Assignment: Spam Detection

Date Assigned: April 6, 2017

Completion Date: TBD

Software issues:

If you feel there are mistakes in this assignment, check Piazza for corrections, and report them to us if

they have not been made.

Assignment

In this assignment, you will create a web application that automatically predicts whether an email is spam or not spam.

Click here to view an example of the application.

This application consists of several parts.

Parts

1. Database (Stored Procedures) HTML Page
2. Weka
3. HTML Page
4. PHP

Database

1. Download the the SQL file to create your table.

a. https://ils.unc.edu/courses/2017_spring/inls623_001/downloads/spamData.sql

b. This table has three columns: id, text, value

   i. id is an autoincremented number

   ii. text is the email text

   iii. value is whether the text is spam or not spam (spam, not spam)

2. Create features

a. Determine which features would be useful

   i. The length of the text field

b. Create a stored procedure named ComputeTextLengthFeature()

   i. This stored procedure should compute the feature length of text field.

   ii. Call the stored procedure. Your shoulds should look similar to the ones in the below image.

id	value	textLength
2795	spam	22442
2796	spam	3793
2797	spam	792
2798	spam	661
2799	spam	2885
2800	spam	693
2801	spam	2390
2802	spam	710
2803	spam	29082
2804	spam	2827
2805	spam	2227
2806	spam	802
2807	spam	837
2808	spam	657
2809	spam	5150

c. Export this table to a csv file. Press the Export button as shown in the screenshot below.

100% 1:14 2 errors found

Result Grid | Filter Rows: Search | Export:

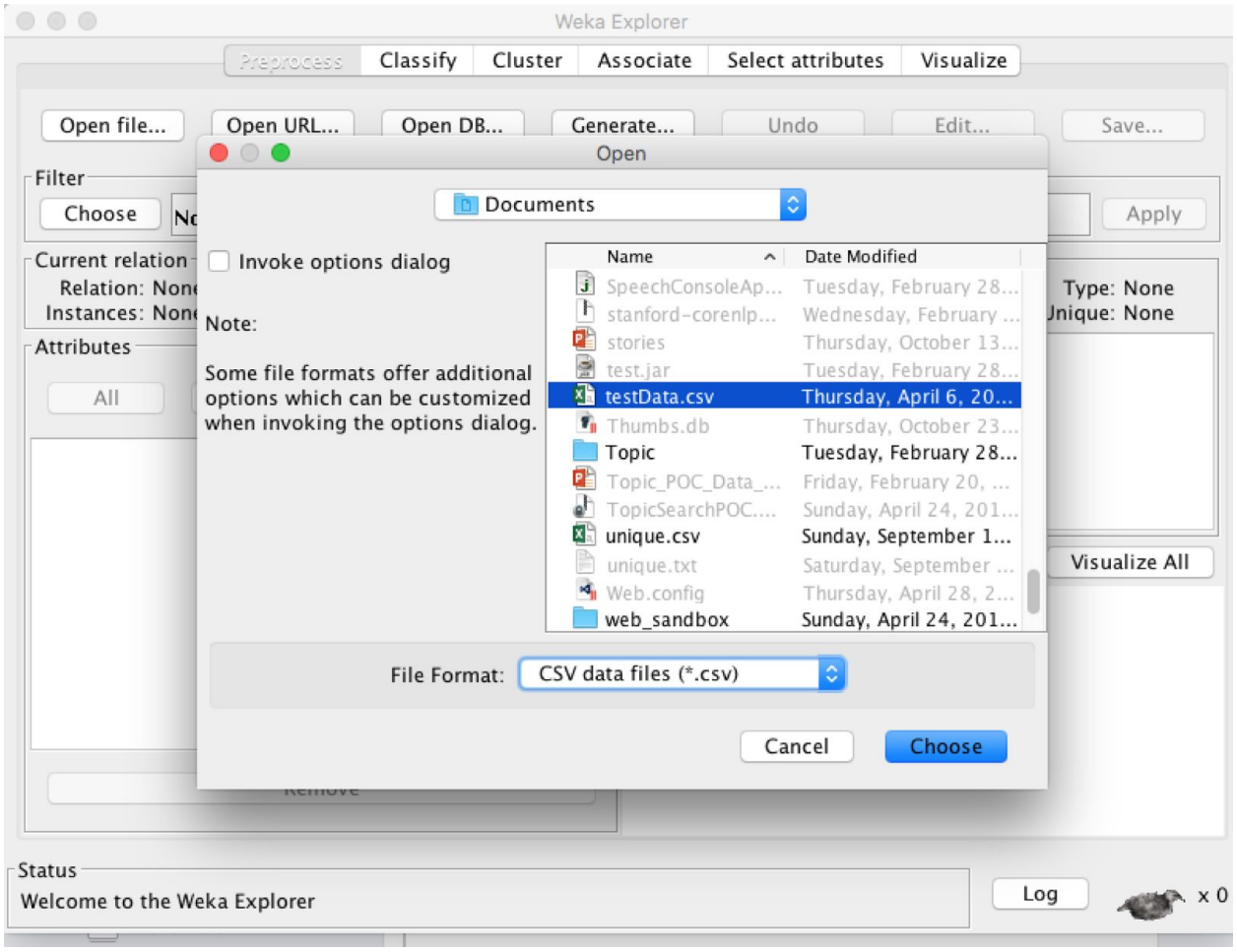
id	value	textLength
2795	spam	22442
2796	spam	3793
2797	spam	792
2798	spam	661
2799	spam	2885
2800	spam	693
2801	spam	2390
2802	spam	710
2803	spam	29082
2804	spam	2827
2805	spam	2227
2806	spam	802
2807	spam	837
2808	spam	657
2809	spam	5150
2810	spam	685
2811	spam	4680
2812	spam	2399
2813	spam	714

Result 58

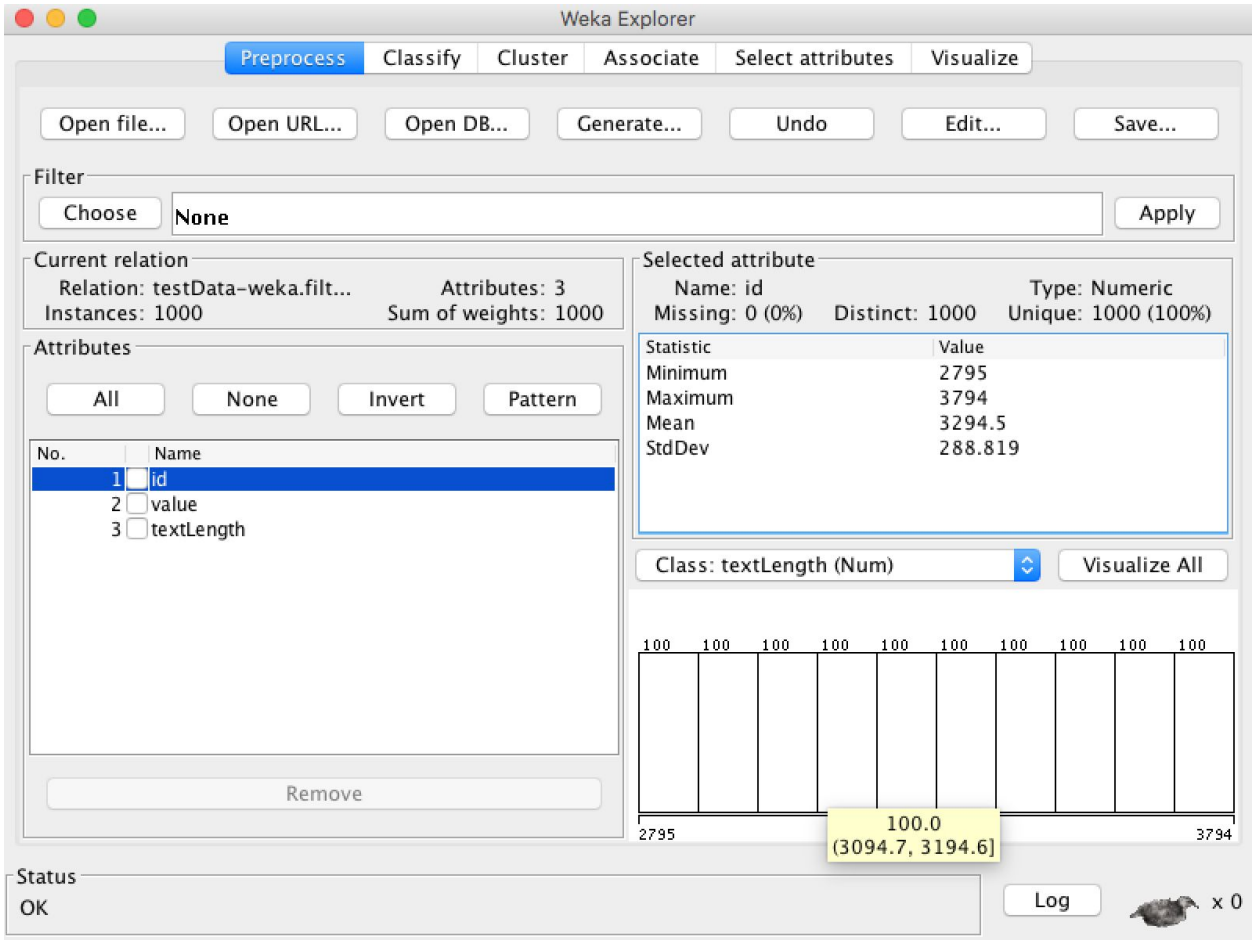
---

d. Open the file in Weka.

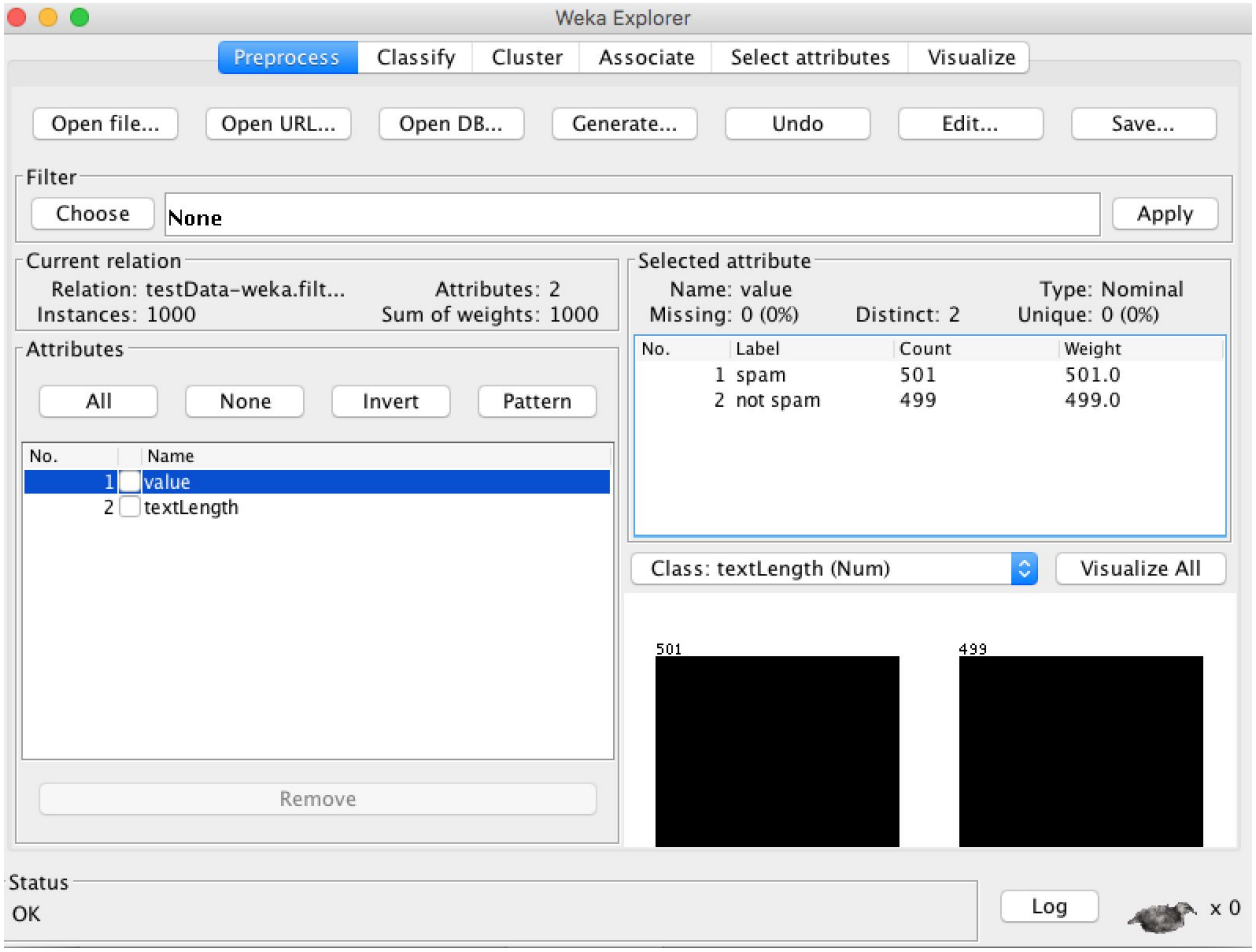
   i. Make sure to choose .csv from the dropdown (shown in the image below).



After you select Choose, your screen should look like this:

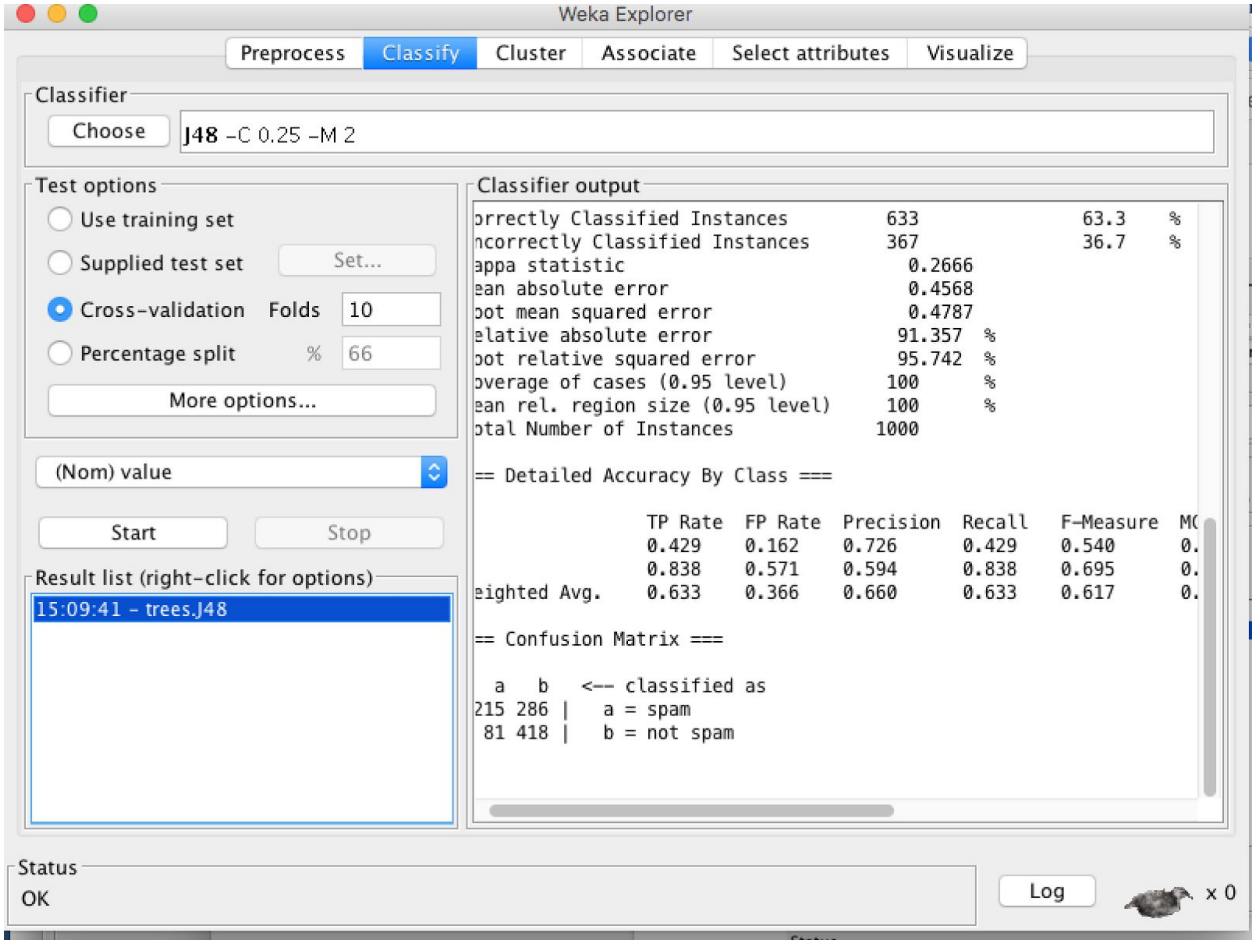


Select id and then click the remove button and your screen should look like the screen below.



Next select, the Classify tab and **make sure the field (Nom) value is selected.** Then press the choose button, select trees and choose J48. J48 is the decision tree classifier. Finally press Start.

Your screen should look like the one below after perform the steps above.



Write down the accuracy for the feature text_length in a table like the one below.

Feature	Accuracy
TextLength	63%

Results

63.3% accuracy decent if we look at our initial data distribution. We have 1000 emails, and of them 499 are not spam, or roughly 49.9% are not spam, so 63.3% accuracy is a decent improvement. (In other words, if we predicted not

spam 100% of the time, 49.9% of the time we would be correct. With our decision tree classifier we improved the accuracy by 13.4%.

We would like to increase the accuracy even more. To do this we need to compute more features.

Compute the following features separately **(in a separate stored procedure)** and write down the accuracy for each feature in a table:

1. Whether the text is html (whether the string “html” is in the text)
2. Whether the text contains a link (whether the string “http” is in the text)
3. Whether the text contains the following spammy words:['join', 'free', 'buy', 'start', 'click', 'discount']

Feature	Accuracy
Text-Length	63%
IsHTML	
HasLinks	
HasSpammyWords	

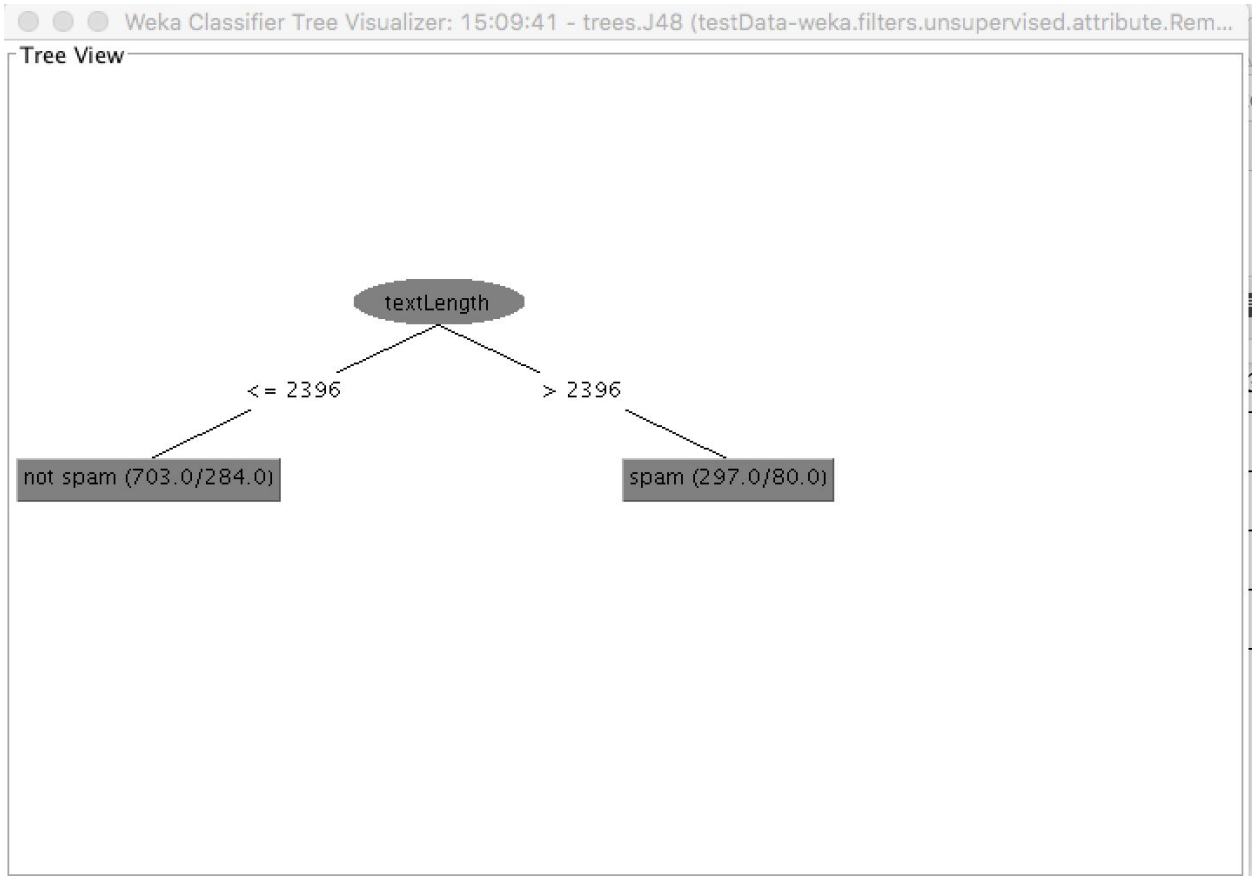
Next, compute all of the features together: TextLength, IsHTML, HasLinks, HasSpammyWords and write down the accuracy

Feature	Accuracy
TextLength	63%
IsHTML	
HasLinks	
HasSpammyWords	

Combination of all features	

Highlight which feature or the combination of all features are the most accurate. Use Weka to visualize the decision tree for the most accurate feature(s).

The example below shows the text length feature. Your most accurate feature may be different.



Create a table named userInputSpam. The columns should be id, text, prediction.

Create a stored procedure named SpamDecisionTree that takes as input parameters the most accurate feature(s) and an output parameter which returns spam or not spam. This stored procedure uses if/else statements to create the rules for your decision tree.

Create a stored procedure named WekaCreateFeatures that has an input parameter userInputId and output parameters for your most accurate feature(s). If your most accurate feature is TextLength then the stored procedure will return TextLength as an out parameter. If your most accurate feature is the combination of all features, then your our parameters will be all of the features.

The stored procedure should compute features for a row with userInputId (input parameter) in the userInputSpam table.

HTML Page

Create an HTML page named "predictSpam.html". This page has a form which contains textarea as an input type and a submit button. It should POST to predictSpam.php.

PHP

Your PHP page should get the value of the text area (which contains text).

1. Insert the value of the text into the userInputSpamTable and get the id of the row you just inserted.
    a. //code to get the last id of the row you just inserted.
    b. `$last_id = $conn->insert_id;`
2. Call the stored procedure, WekaCreateFeatures with the correct input parameters in PHP and return the correct output parameters.

3. Call the stored procedure SpamDecisionTree, which takes the output parameters from WekaCreateFeatures as input parameters.

4. Print out whether the email is spam or not spam