This document will serve as the project write-up for the decision tree creation of Sean MacEachern and Zach Arnold at WPI (C Term 2015).

# Feature Selection

Through our own recollection of good heuristics from project 1 and research on the internet of good features for the solved game of connect 4, we have chosen the following features to classify each board state for WEKA.

* The total number of Connect 2’s each person has
  + Justification:
  + We thought this would be a good indicator of the “streak” heuristic which was our most successful. In essence, we were able to show through a thousand games that
* The total number of Connect 3’s each person has
  + Justification:
  + This one is quite easy to justify. Intuitively from playing the game, you need some series of connected two’s or three’s to win with a connect 4. This feature is basically guaranteeing a win as long as there’s not a blocking piece in the way. (We feel as though this will split the J48 tree nicely.)
* Which player has more columns and rows with pieces in them
  + Justification:
  + This is based off of the chess principle called: “zugzwang.” The formal definition of this strange German word is a situation where a player is forced to make a move when he would rather make no move at all. When one player dominates the board, there is more of an opportunity to win. (I have no idea if this will actually work.)
* Total number of potential Connect 4’s each person has
  + Justification:
  + This one is super easy to justify. If they are one token away from winning…they’re probably going to win.
* The total number of pieces on the board
  + Justification:
  + This is simply a measure of the completeness of the board state. How close to done are we? How mature is the game in this state? This feature is a measure of that.
* Row/Column ownership by a player
  + Justification:
  + Our thought here was that similarly to the zugzwang principle, a player that owns a column or row (has only his/her pieces in that column or row) has a greater chance of using that row or column to make a connect 4.

# WEKA/Experiment

We used Weka to classify our data into a tree using the J48 classification algorithm as well as a Random Forest with different features on/off and differing levels of cross-validation.

The first problem we encountered was the NumericToNominal converter. It seemed as though Weka did not like that for one of our features, numbers were symbols. As such, we added a function to our output generating program that turns the result into an “a” if player 1 was victorious, and a “b” if player 2 was victorious. We were now able to see the data in the preprocessing phase properly.

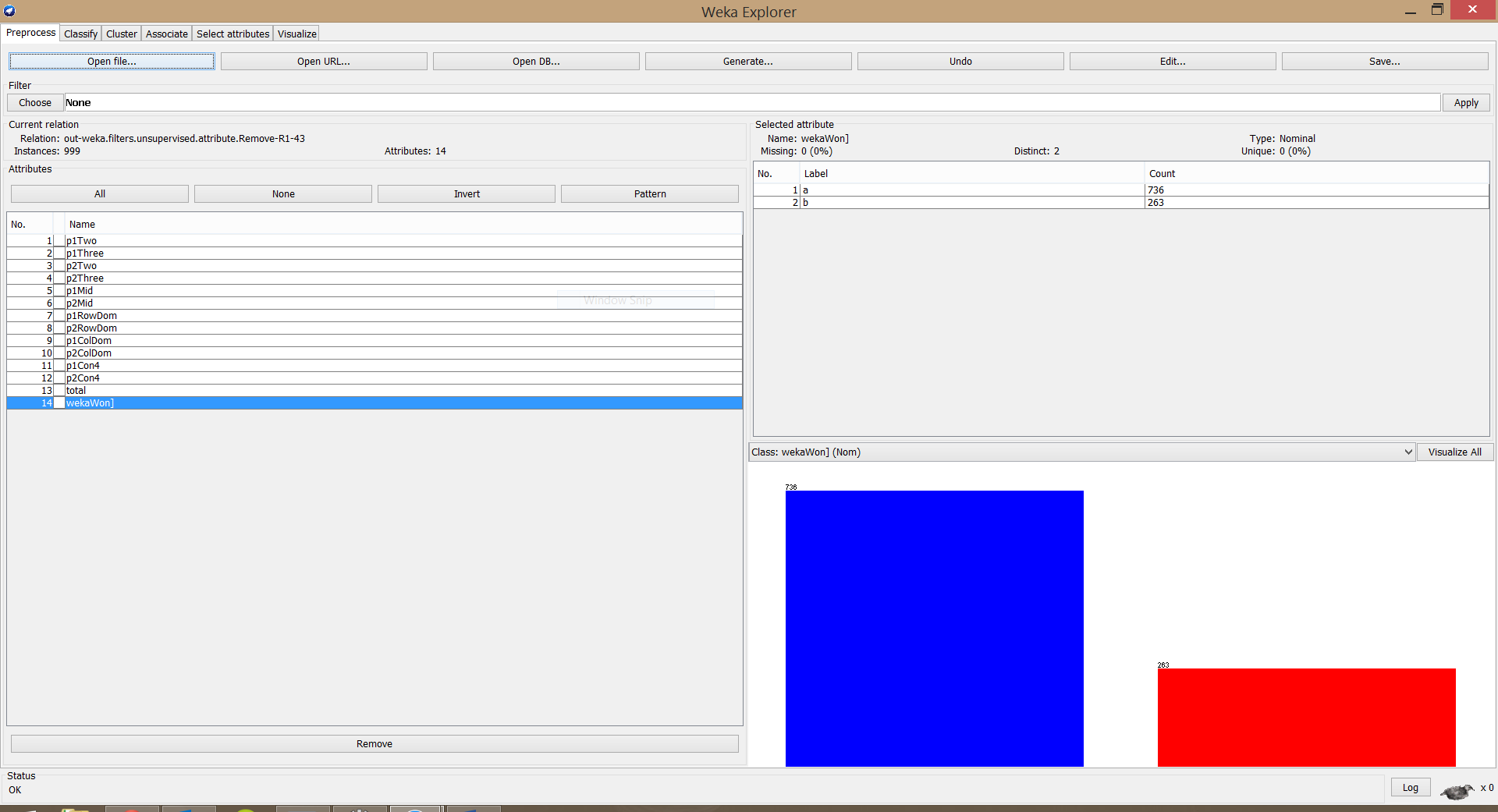


Figure 1 - WEKA Preprocessing Step

Then we were able to start classification:

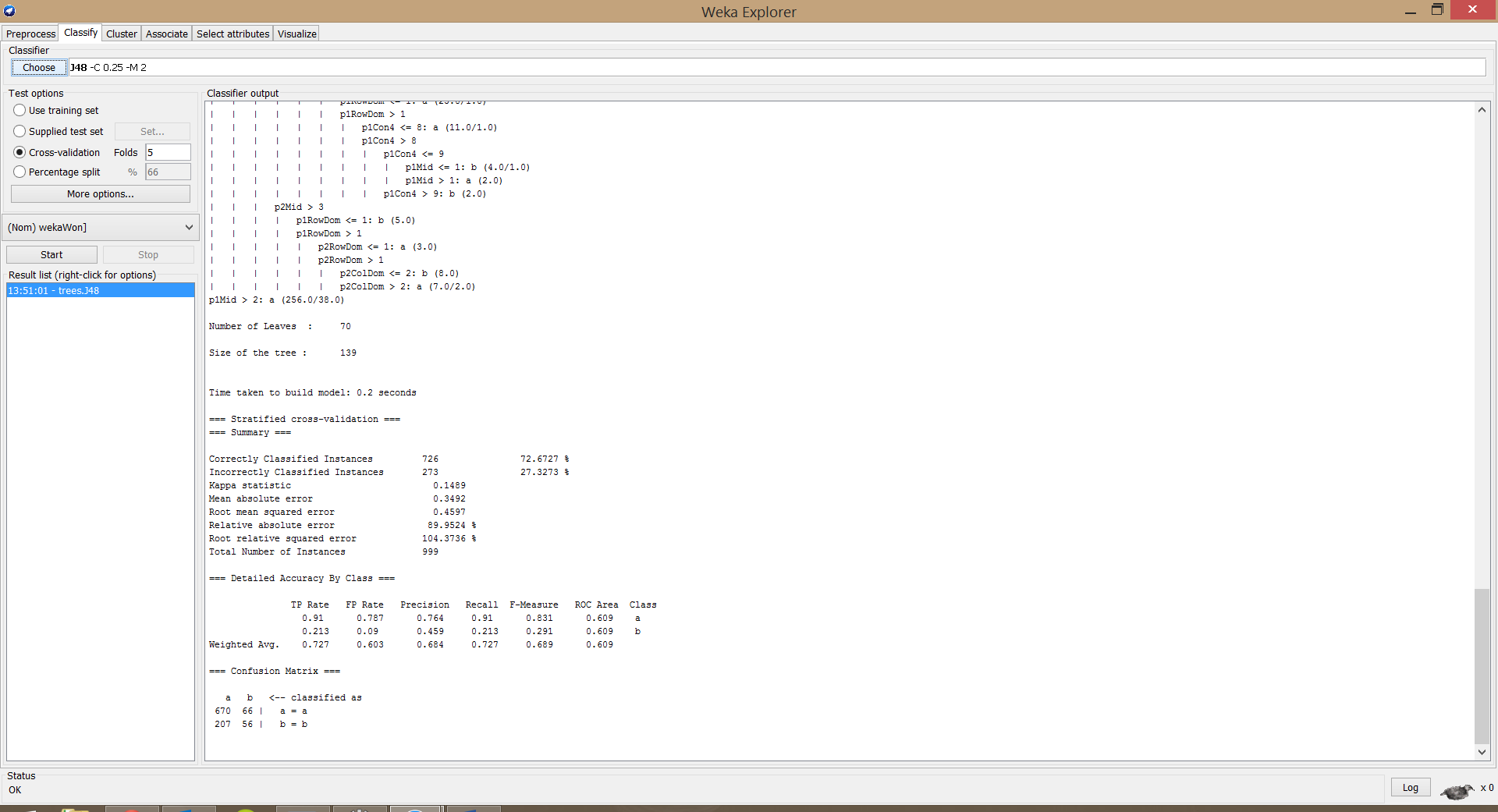


Figure 2 - J48 Classification with All Features On, 5 folds cross validation.

Below is a table of various attempts at classification well explained with what we did, and the results yielded. (You may assume that all of these were attempted using the J48 classification algorithm.)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Features Off** | **K-fold Cross-Validation** | **Correctly Classified** | **Incorrectly Classified** | **Kappa Statistic** | **Mean Absolute Error** | **Relative Absolute Error** |
| None | 5 | 70.3704% | 29.6926% | 0.1603 | 0.3494 | 89.9982% |
| None | 10 | 68.3684% | 31.6316% | 0.0711 | 0.3675 | 94.6772% |
| P1+P2 Row/Col Ownership | 5 | 72.6727% | 27.3273% | 0.1489 | 0.3492 | 89.9524% |
| P1+P2 Row/Col Ownership | 10 | 71.3714% | 28.6286% | 0.1097 | 0.3556 | 91.6125% |
| Connect 2's and 3's | 5 | 71.7718% | 28.2282% | 0.1448 | 0.3578 | 92.1757% |
| Connect 2's and 3's | 10 | 70.6707% | 29.3293% | 0.0866 | 0.3633 | 93.5882% |
| Everything but p1/p2 Potential Connect 4's | 5 | 73.6737% | 26.3263% | 0 | 0.3879 | 99.9279% |
| Everything but p1/p2 Potential Connect 4's | 10 | 73.6737% | 26.3263% | 0 | 0.3879 | 99.9279% |
| Everything but p1/p2 Potential Connect 4's and Connect 2's and 3's. | 5 | 72.9730% | 27.0270% | -0.0067 | 0.3885 | 100.0745% |
| Everything but p1/p2 Potential Connect 4's and Connect 2's and 3's. | 10 | 73.6737% | 26.3263% | 0 | 0.3879 | 99.9279% |

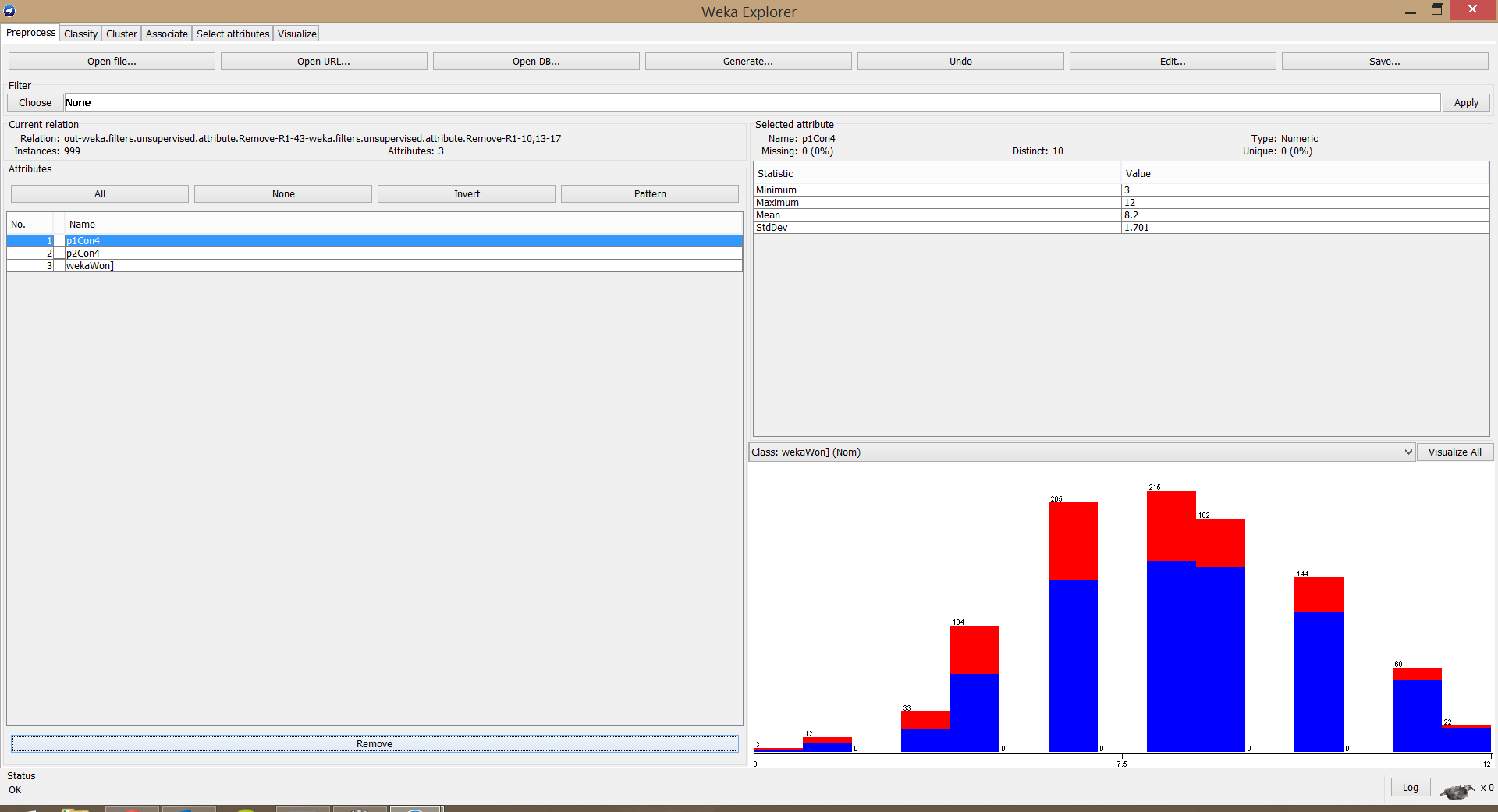


Figure 3 - Experimenting with Removing Features in the Preprocessing Step

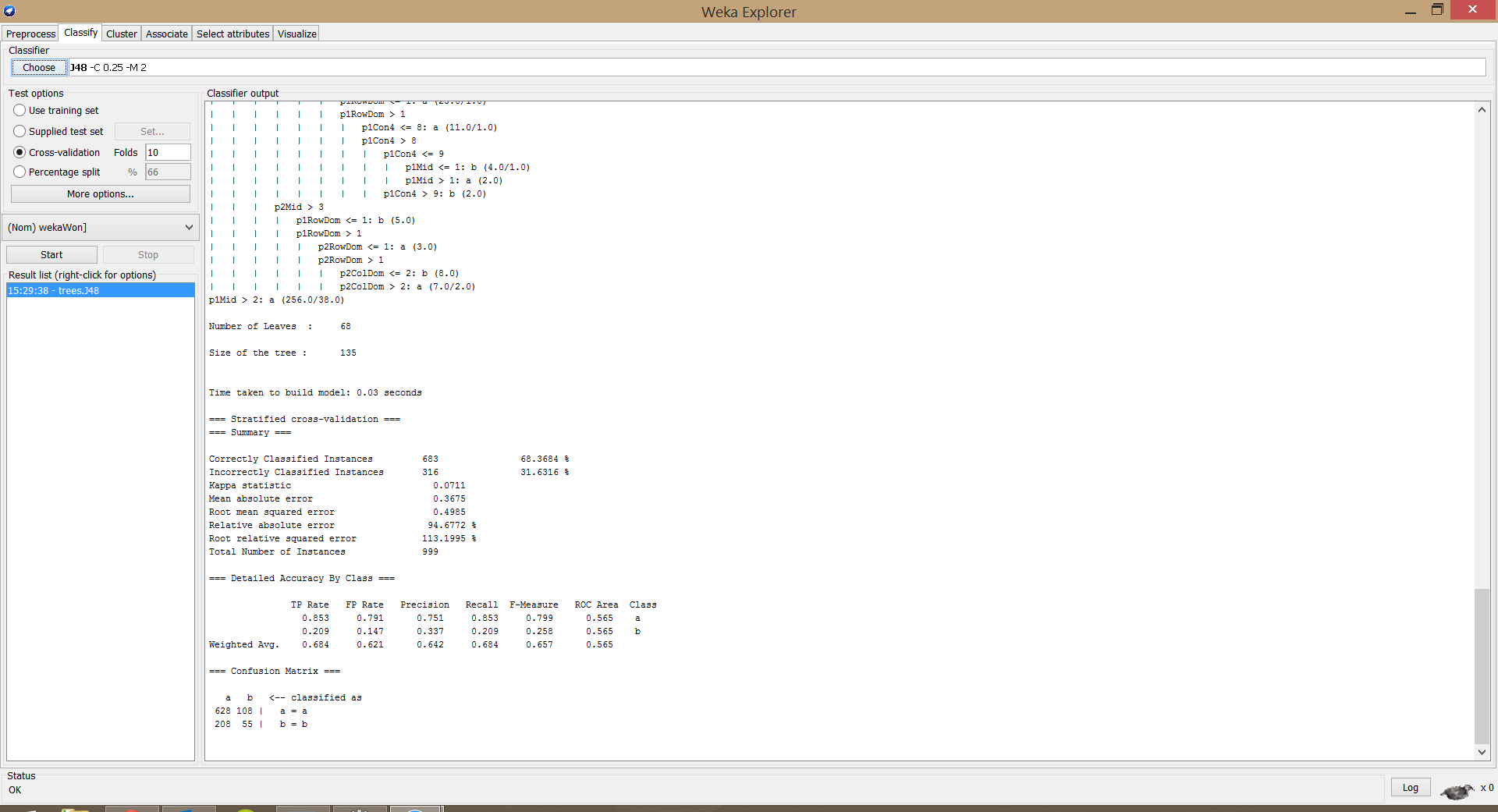


Figure 4 - Experimental Output

# Results

We were able to infer the following from our trial of using varying features and comparing their usability with each other and their overall impact on the classification results. As you can see from the table, the most effective combination of features happened when we removed at least one from the mix. (With the potential connect 2’s, 3’s, and 4’s having the most negative impact.)

## Analysis of Feature Effectiveness

In terms of analysis of the individual features, the more effective features we had were the ones that quantified row and column domination/ownership, as well as middle of the board domination. The least useful feature for the data was the “total” value which was consistently “8” the entire time. The features who had the greatest negative impact on Weka’s J48 classification was the potential connect 4’s, as well as potential connect 2’s and 3’s. With these features there was a negative impact of up to 5% on the percentage of correctly classified instances.

## Analysis of Results

Weka was able to correctly classify 73.6737% of instances in our best case. This is pretty good considering our features. Unfortunately however, the other statistics we have for our best case, are very much not best case. Our Kappa statistic for all runs was less than .2, which is not good. Less than 20% of the time, two people with our model will guess the same thing. In some cases this number is negative, which means not only do they pick separately all the time, but they do so out of spite! Our best relative absolute error is about 90%. So, unfortunately this model is not very well equipped for other data. We did notice that there seems to be an inverse relationship with mean absolute error and correct classification. Somehow, when the average error is higher, the model performs better.

We used different levels of k-fold cross validation to train/test this dataset. We tried every combination on both 5 and 10-fold cross validation, because they are popular numbers from research to train the data on. In general, 5 fold cross validation performed better than 10-fold did. This is most likely due to the fact that more data are used to train the set before it is cross-validated against itself.

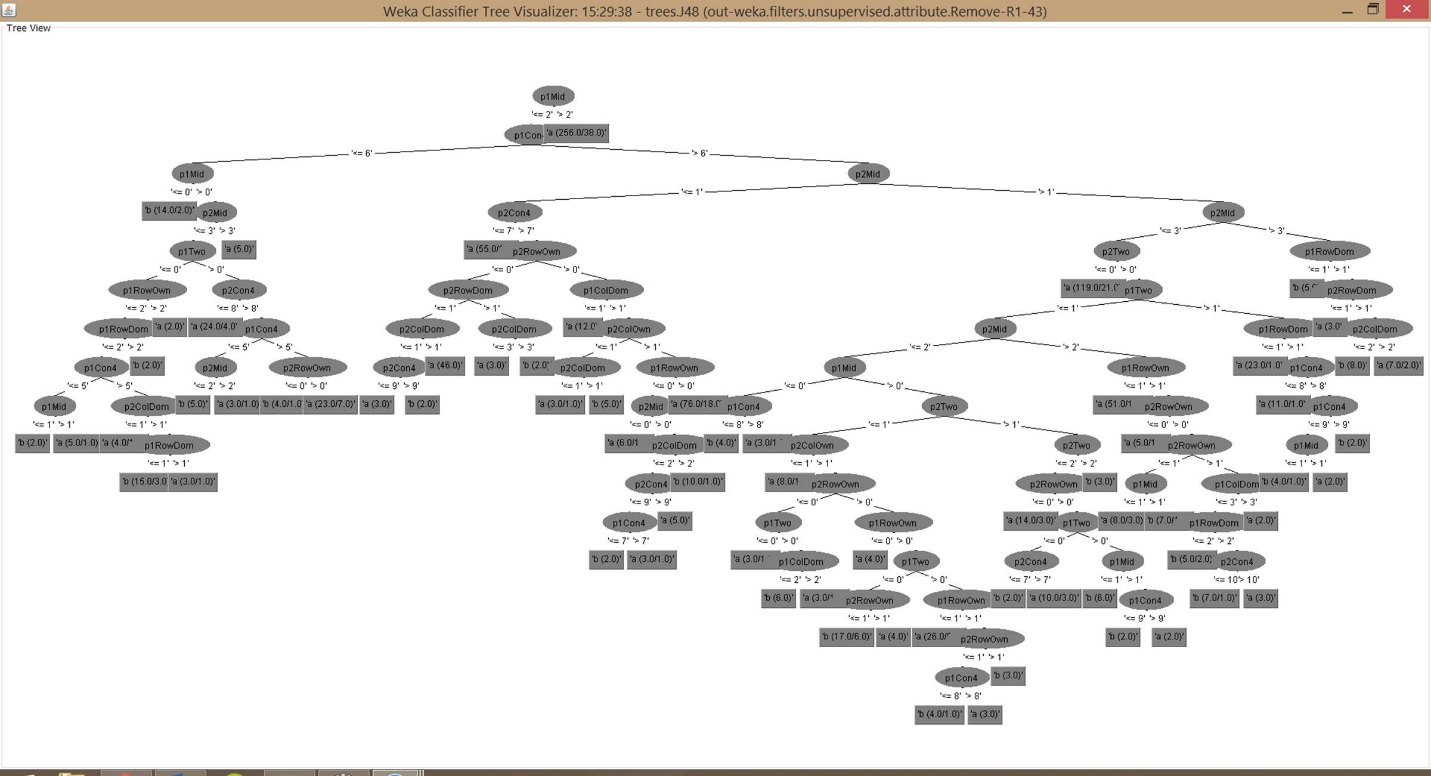


Figure 5 - Example Decision Tree from our Features

# Other Methods

In addition to J48 classification trees, we also used the Random Forest technique to determine if our tree was good based on many other trees. Part of what makes random forest great is that it can tell you which features are important based on many different methods instead of one. The random forest of 100 trees considering 5 random features yielded:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Correctly Classified** | **Incorrectly Classified** | **Kappa Statistic** | **Mean Absolute Error** | **Relative Absolute Error** |
| 72.0721% | 27.9279% | 0.1037 | 0.3661 | 94.2804% |

This result is similar to our best run with a J48 classification tree. Since they are similar, we can suppose that given our feature set, we have an optimal classification model.

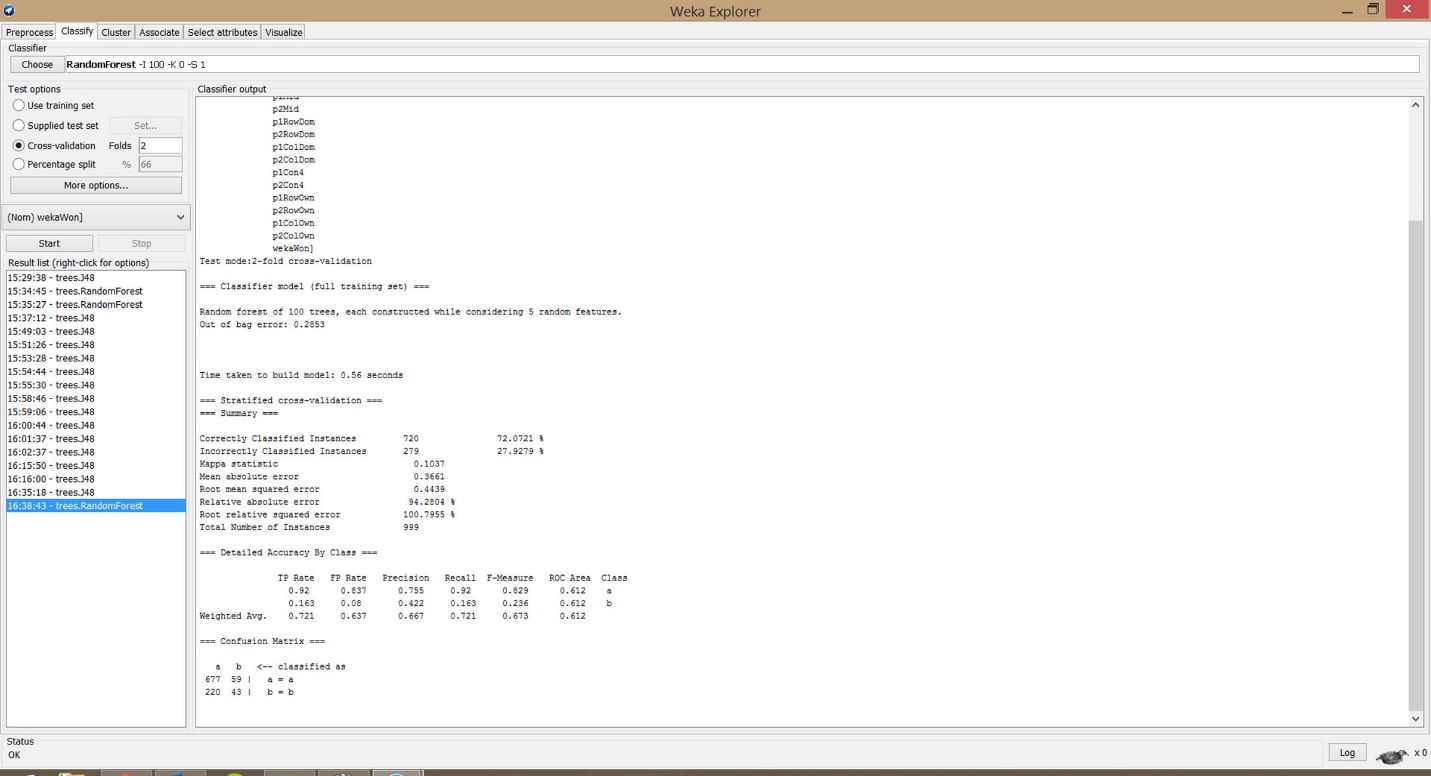


Figure 6 - Random Forrest on Our Data with Our Features