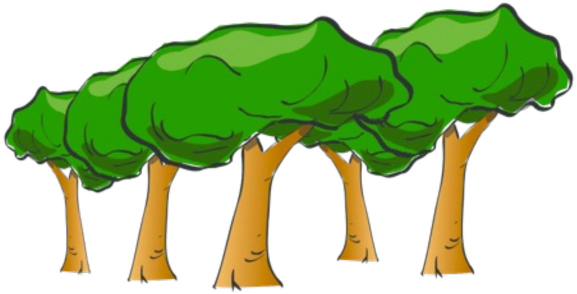
14.5

What’s better than a tree?

Random Forest

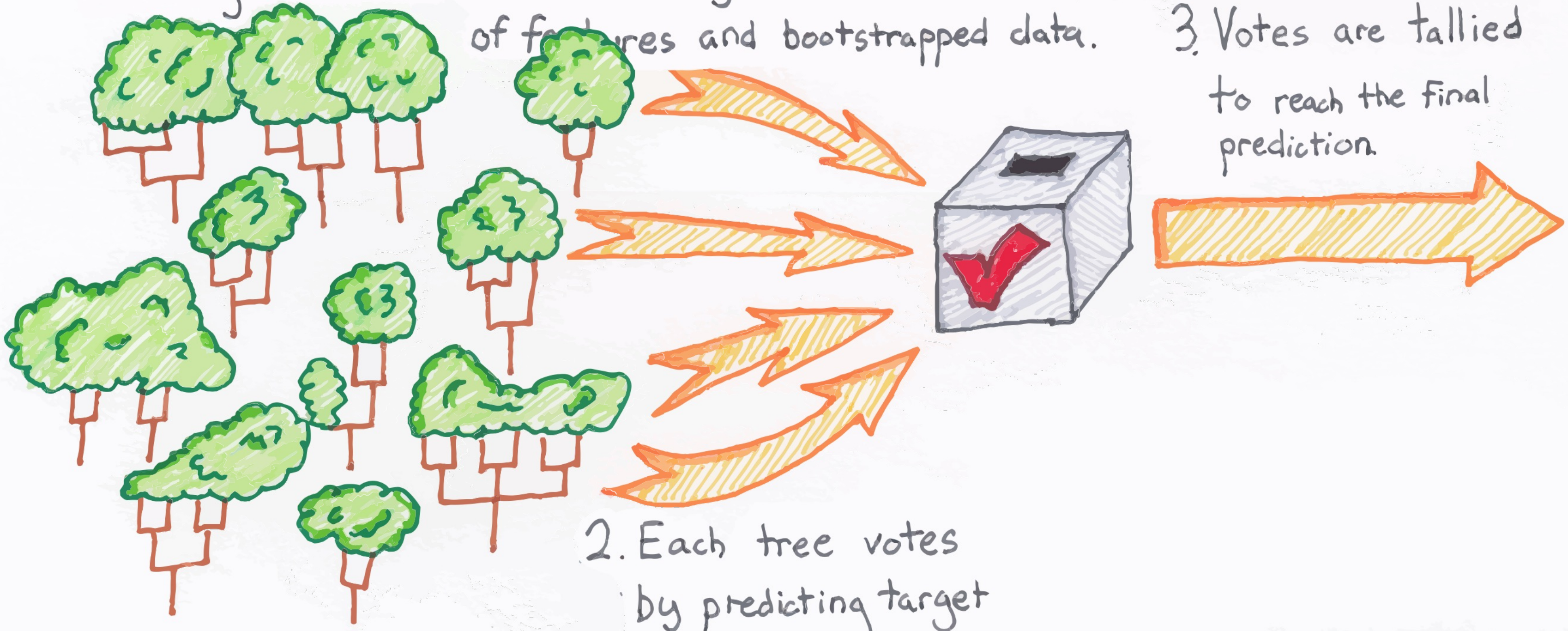


- Can often improve performance of decision tree classifiers using a set of decision trees (a forest)
- Each tree trained on a random subset of training data
- Classify a data instance using all trees
- Combine answers to make classification
  - E.g., vote for most common class

RANDOM FOREST

CLASSIFICATION

1) 1. Many trees are created using random subsets of features and bootstrapped data.



2. Each tree votes by predicting target class.

3. Votes are tallied to reach the Final prediction.

CHRIS ALBON

cf. Wisdom of the Crowd

- Statistician Francis Galton observed a 1906 contest to guess an ox's weight at a country fair. 800 people entered. He noted that their average guess (1,197lb) was very close to the actual weight (1,198lb)
- When getting human annotations training data for machine learning, standard practice is get ≥ 3 annotations and take majority vote

cf. abbreviation (short for Latin: confer/conferatur) refer reader to other material to make a comparison

Random Forests Benefits

- Decision trees not the strongest modeling approach
- Random forests make them much stronger
- => more **robust** than a single decision tree
  - Limits overfitting to given dataset
  - Reduces errors due to training data bias
  - Stable performance if some noise added to training data

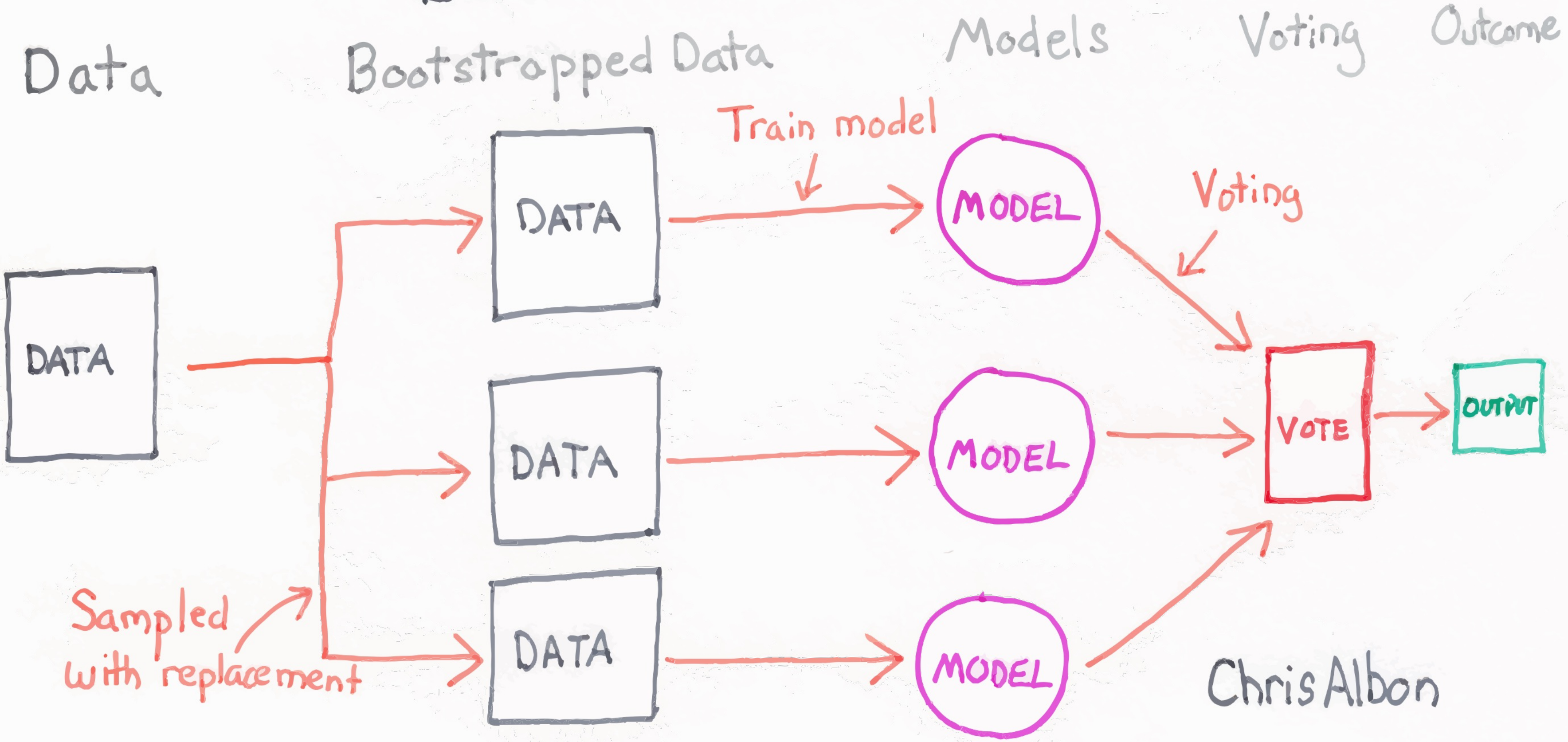
Bagging

- Idea can be used on any classifier!
- Improve classification by combining classify-cations of randomly selected training subsets
- Bagging = *Bootstrap **agg**regat**ing***

  An **ensemble** meta-algorithm that can improve stability & accuracy of algorithms for statistical classification and regression
- Helps avoid overfitting
- AKA ensembling

BAGGING

BOOTSTRAP AGGREGATION



Choosing training data subsets

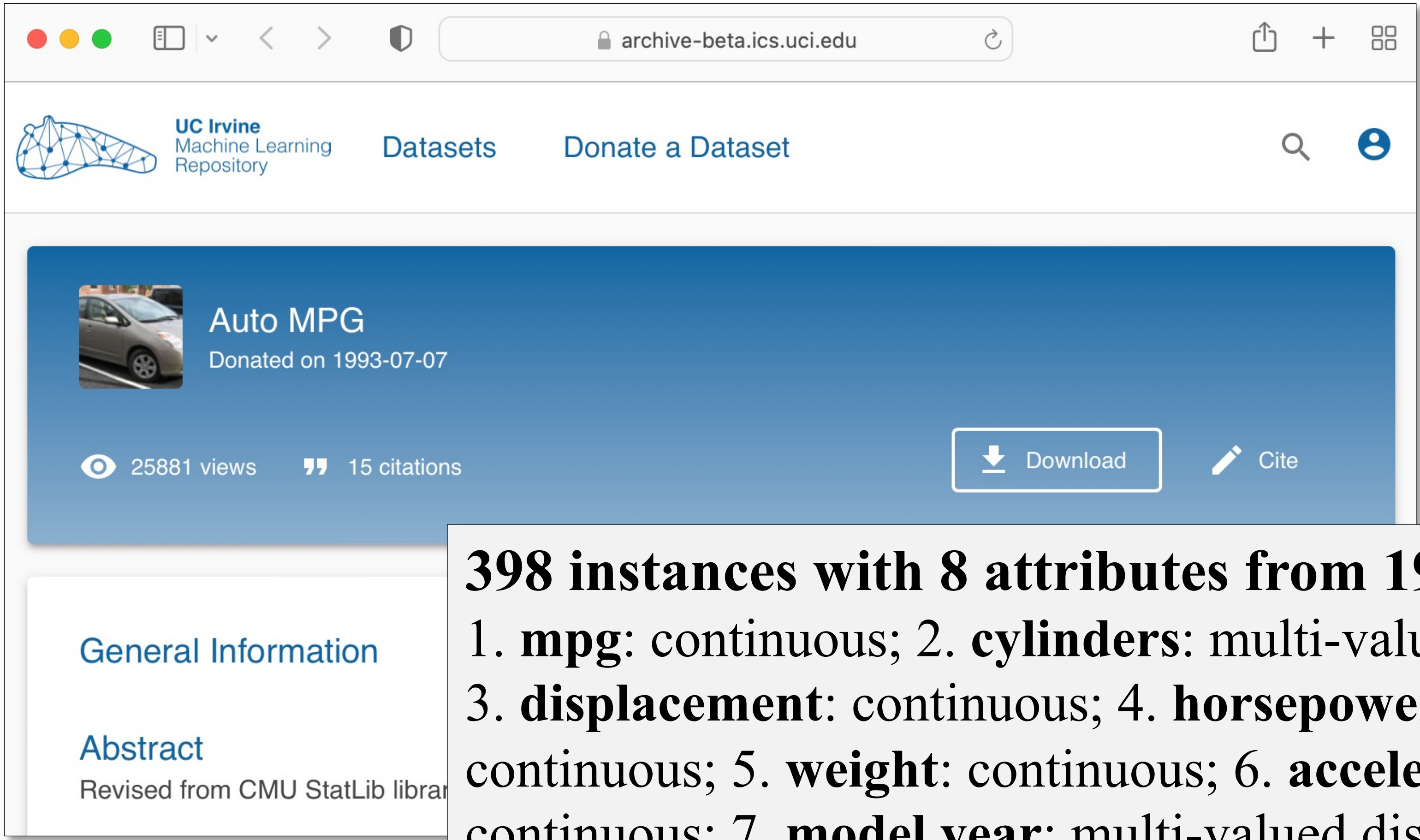
- **Classic bagging**: select random subset of training instances **with replacement**
- **Pasting**: select random subset of training instances (i.e., without replacement)
- **Random Subspaces**: use all training instances, but with a random subset of features
- **Random Patches**: random subset of instances and random subset of features
- **Best?** depends on problem, training data, algorithm

Examples

- Two examples using **Weka**
  - UCI Auto mpg prediction dataset
    - 398 instances,
  - UCI Adult income prediction dataset
    - ~49,000 instances
- **RandomForest** improves over **J48** for the smaller dataset, but not for the larger one
- Takeaway: more data is always best

UCI Auto MGP Dataset

kaggle



Auto MPG

Donated on 1993-07-07

General Information

Abstract

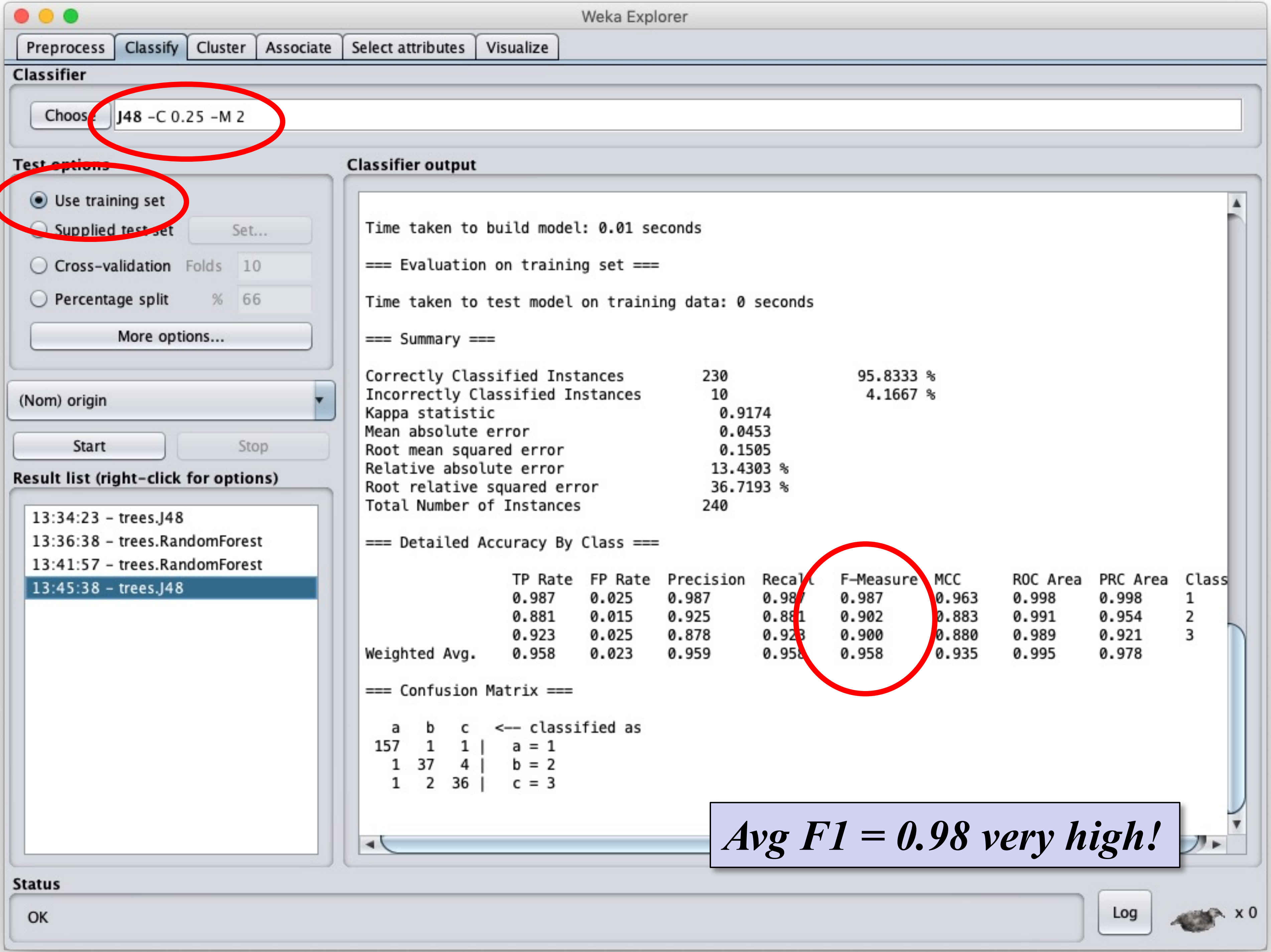
Revised from CMU StatLib librar

398 instances with 8 attributes from 1983:

1. **mpg**: continuous; 2. **cylinders**: multi-valued discrete; 3. **displacement**: continuous; 4. **horsepower**: continuous; 5. **weight**: continuous; 6. **acceleration**: continuous; 7. **model year**: multi-valued discrete; 8. **origin**: multi-valued discrete; 9. **car name**: string (unique for each instance)

Predict MPG from other 7 attributes

Arff training data (240); test data (132)



Classifier: J48 -C 0.25 -M 2

Test options:
- ● Use training set
- ○ Supplied test set
- ○ Cross-validation Folds 10
- ○ Percentage split % 66

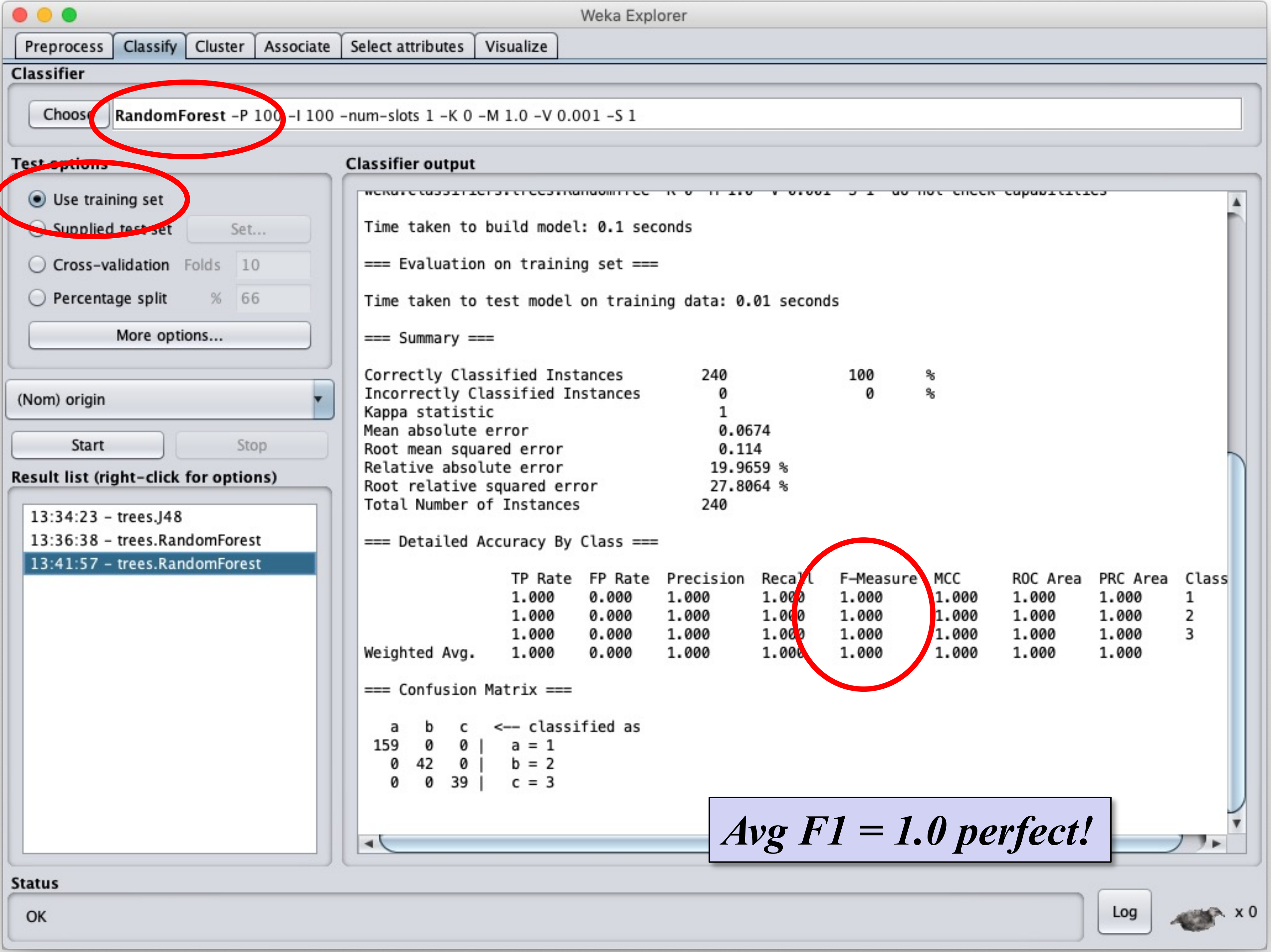
(Nom) origin

Result list (right-click for options)
- 13:34:23 - trees.J48
- 13:36:38 - trees.RandomForest
- 13:41:57 - trees.RandomForest
- 13:45:38 - trees.J48

```
Time taken to build model: 0.01 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances         230               95.8333 %
Incorrectly Classified Instances        10                4.1667 %
Kappa statistic                          0.9174
Mean absolute error                      0.0453
Root mean squared error                  0.1505
Relative absolute error                 13.4303 %
Root relative squared error             36.7193 %
Total Number of Instances              240

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.987    0.025    0.987      0.987    0.987      0.963    0.998     0.998     1
                 0.881    0.015    0.925      0.881    0.902      0.883    0.991     0.954     2
                 0.923    0.025    0.878      0.923    0.900      0.880    0.989     0.921     3
Weighted Avg.    0.958    0.023    0.959      0.958    0.958      0.935    0.995     0.978

=== Confusion Matrix ===

   a   b   c   <-- classified as
 157   1   1 |   a = 1
   1  37   4 |   b = 2
   1   2  36 |   c = 3
```

Status: OK

Avg F1 = 0.98 very high!



RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

● Use training set
○ Supplied test set
○ Cross-validation Folds 10
○ Percentage split % 66

(Nom) origin

13:34:23 - trees.J48
13:36:38 - trees.RandomForest
13:41:57 - trees.RandomForest

```
weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.1 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.01 seconds

=== Summary ===

Correctly Classified Instances         240              100      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0.0674
Root mean squared error                  0.114
Relative absolute error                 19.9659 %
Root relative squared error             27.8064 %
Total Number of Instances              240

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     1
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     2
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     3
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

   a   b   c   <-- classified as
 159   0   0 |   a = 1
   0  42   0 |   b = 2
   0   0  39 |   c = 3
```

OK

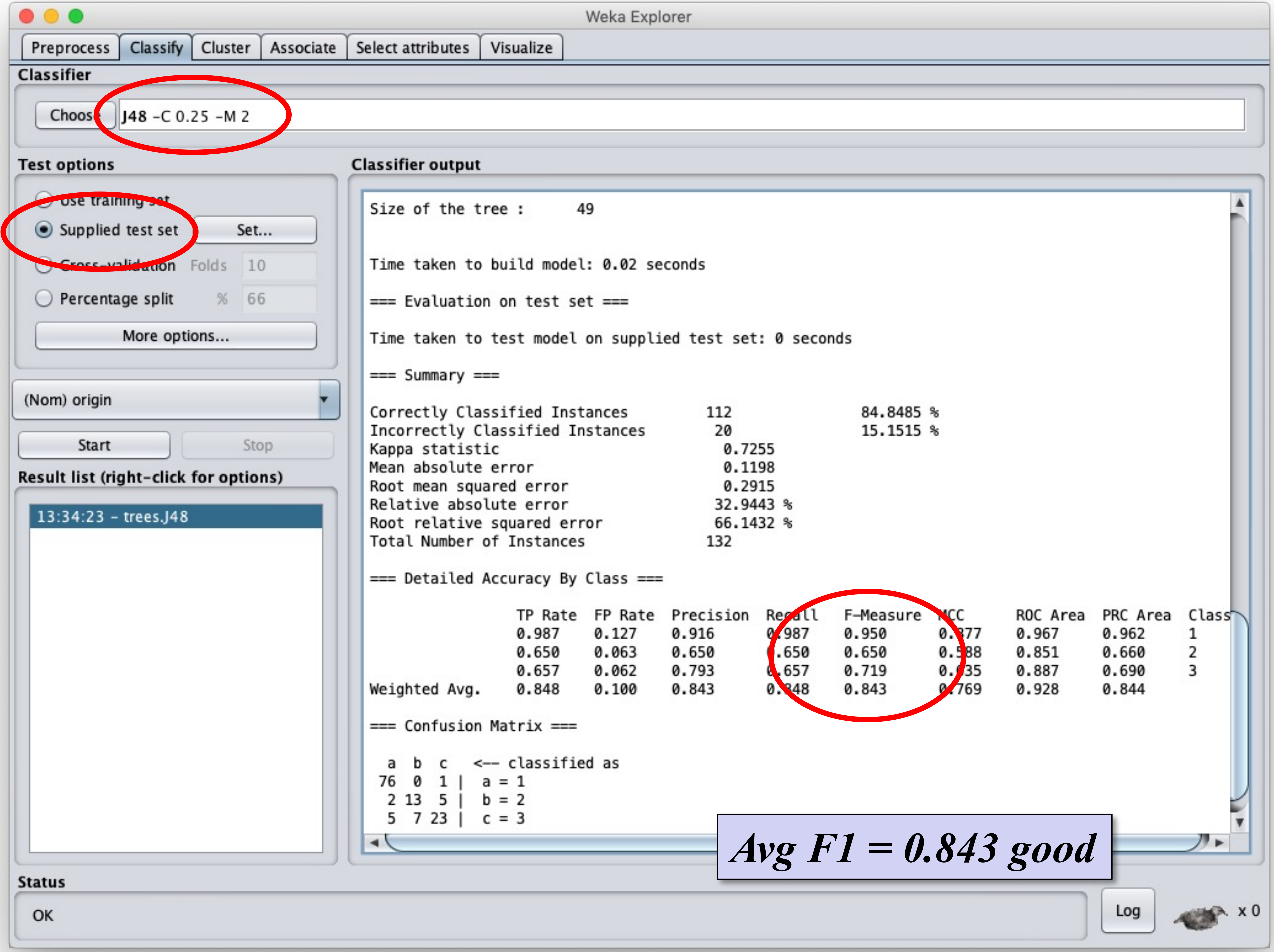
Avg F1 = 1.0 perfect!

100% … Wait, What ?

- Results are **too good to be true!**
  - Something must be wrong
- ML results tend to be asymptotic
  - Asymptotic lines approach a final value but typically never reach it
- Closer you get to F1=1.0, the harder it is to improve
- What did we do wrong?

Results are too good

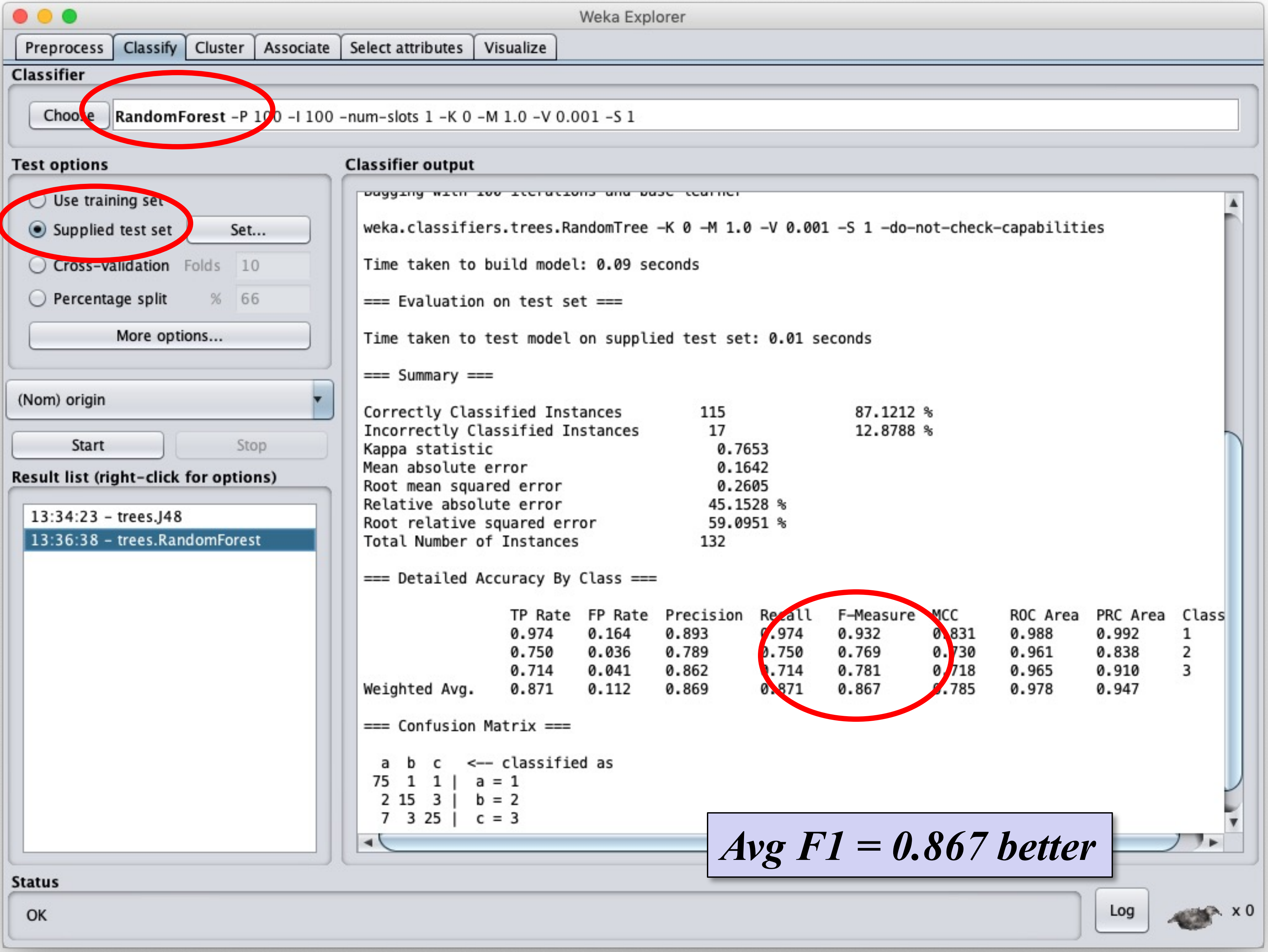
- Relatively small dataset allows construction of a DT model that does very well
- Using Random Forest still got perfect results!
- We trained and tested on the same data!
- Very poor methodology since it overfits to this particular training set
- This training dataset has a separate test data set
    - We can also try 10-fold cross validation



J48 -C 0.25 -M 2

```
Size of the tree :      49


Time taken to build model: 0.02 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         112               84.8485 %
Incorrectly Classified Instances        20               15.1515 %
Kappa statistic                          0.7255
Mean absolute error                      0.1198
Root mean squared error                  0.2915
Relative absolute error                 32.9443 %
Root relative squared error             66.1432 %
Total Number of Instances              132

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.987    0.127    0.916      0.987    0.950      0.877    0.967     0.962     1
                 0.650    0.063    0.650      0.650    0.650      0.588    0.851     0.660     2
                 0.657    0.062    0.793      0.657    0.719      0.635    0.887     0.690     3
Weighted Avg.    0.848    0.100    0.843      0.848    0.843      0.769    0.928     0.844

=== Confusion Matrix ===

  a  b  c   <-- classified as
 76  0  1 |  a = 1
  2 13  5 |  b = 2
  5  7 23 |  c = 3
```

Avg F1 = 0.843 good



Classifier: RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Test options:

○ Use training set

● Supplied test set

○ Cross-validation Folds 10

○ Percentage split % 66

(Nom) origin

Result list:

13:34:23 - trees.J48

13:36:38 - trees.RandomForest

```
Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.09 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances         115               87.1212 %
Incorrectly Classified Instances        17               12.8788 %
Kappa statistic                          0.7653
Mean absolute error                      0.1642
Root mean squared error                  0.2605
Relative absolute error                 45.1528 %
Root relative squared error             59.0951 %
Total Number of Instances              132

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.974    0.164    0.893      0.974    0.932      0.831    0.988     0.992     1
                 0.750    0.036    0.789      0.750    0.769      0.730    0.961     0.838     2
                 0.714    0.041    0.862      0.714    0.781      0.718    0.965     0.910     3
Weighted Avg.    0.871    0.112    0.869      0.871    0.867      0.785    0.978     0.947

=== Confusion Matrix ===

  a  b  c   <-- classified as
 75  1  1 |  a = 1
  2 15  3 |  b = 2
  7  3 25 |  c = 3
```

Status: OK

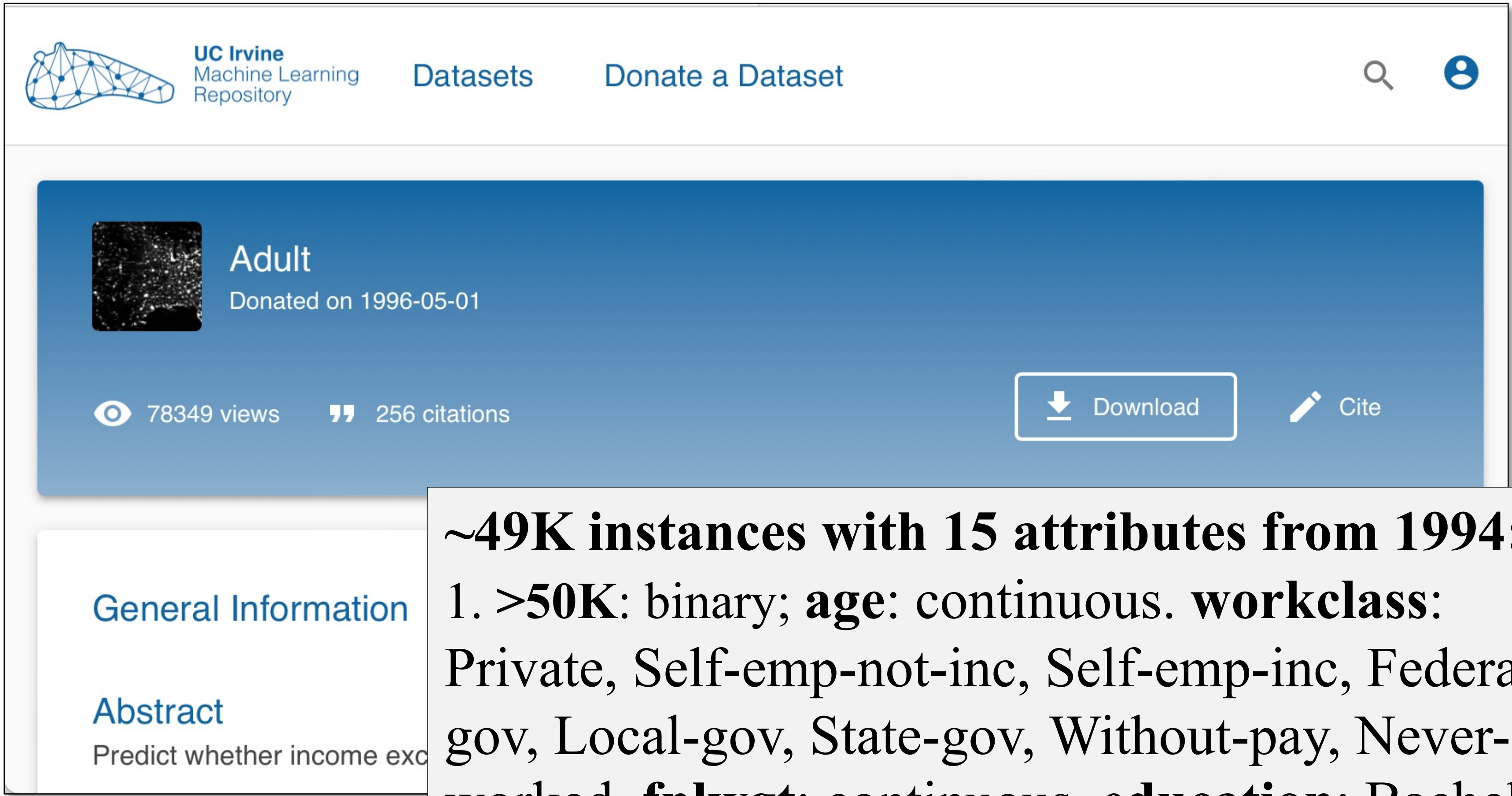
Avg F1 = 0.867 better

New AUTO MPG Results

- Using an independent test set shows more realistic balanced F1 score of **.843**
- Using Random Forest raises this to **.867**
- While the increase is not large, it is probably statistically significant (i.e., not random)
- F1 scores this high are almost always difficult to increase dramatically
  - Human scores for many tasks are often in this range (i.e., 0.8 – 0.9)

UCI Adult Census Income Dataset

kaggle



Adult

Donated on 1996-05-01

General Information

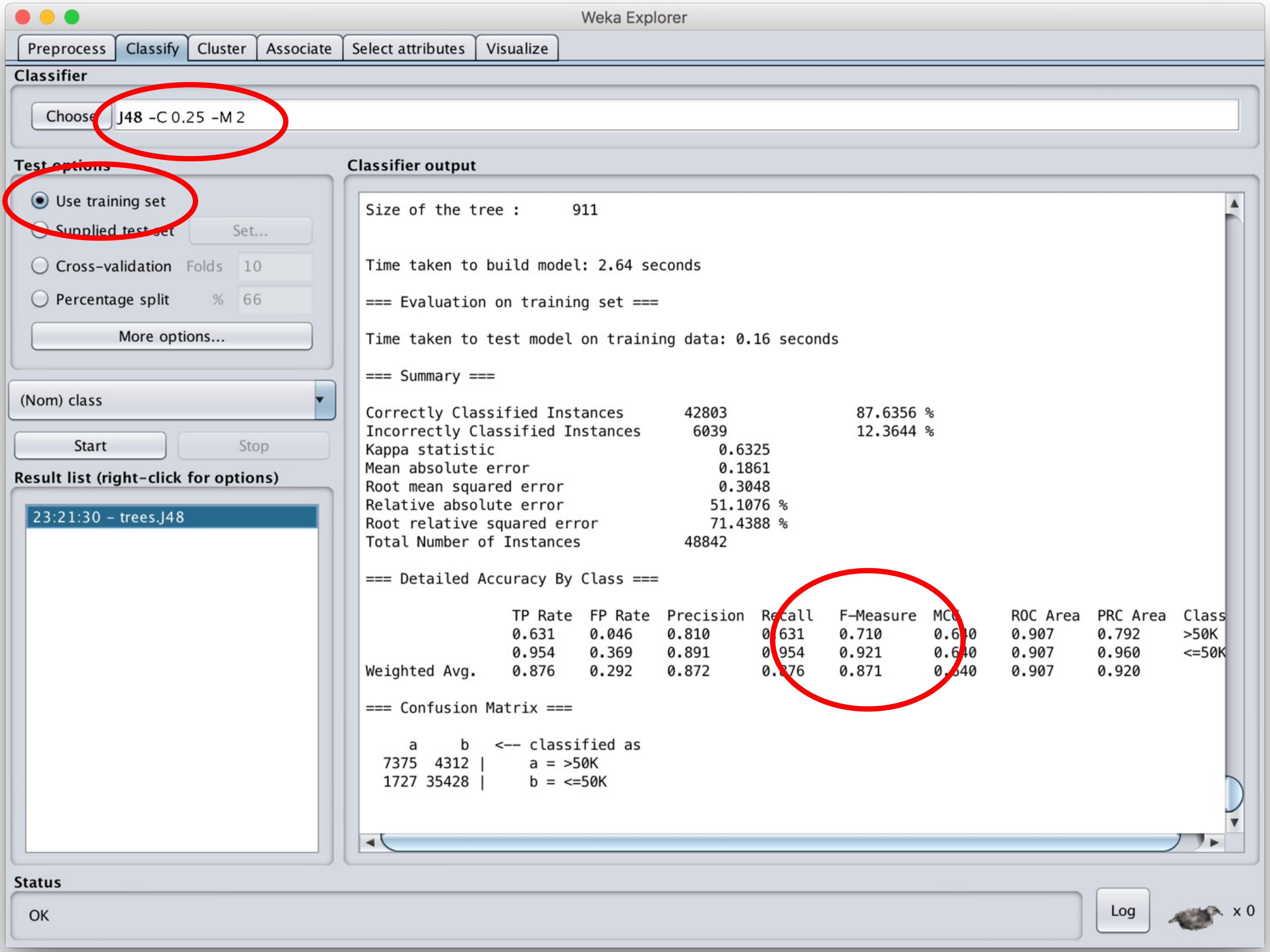
Abstract

Predict whether income exc

~49K instances with 15 attributes from 1994:
1. **>50K**: binary; **age**: continuous. **workclass**: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked. **fnlwgt**: continuous. **education**: Bachelors, Some-college, 11th, HS-grad, Prof-school, …

Predict income >50k from 15 attributes

Arff data



J48 -C 0.25 -M 2

● Use training set

○ Supplied test set

○ Cross-validation Folds 10

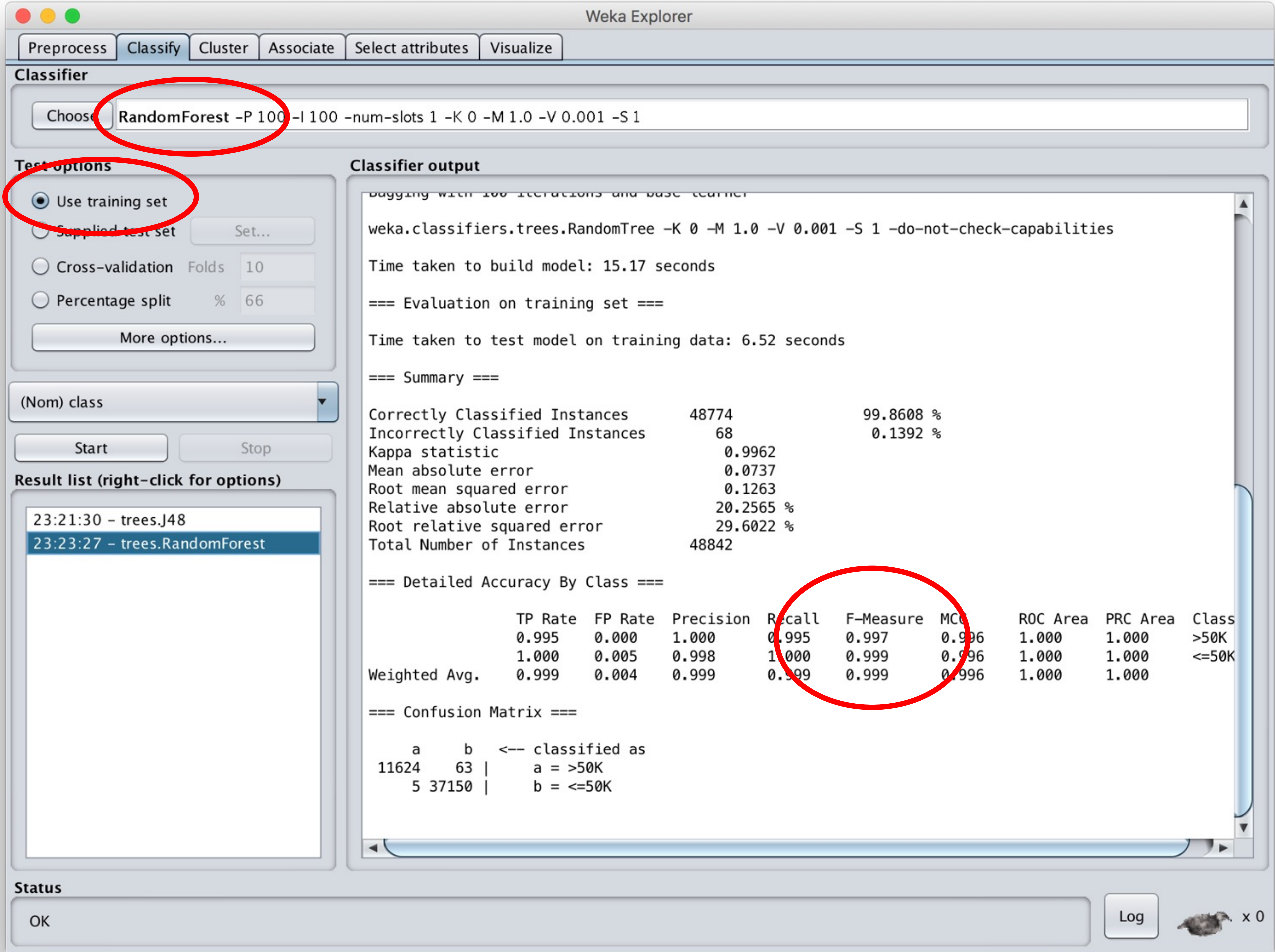
○ Percentage split % 66

(Nom) class

23:21:30 - trees.J48

```
Size of the tree :      911


Time taken to build model: 2.64 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.16 seconds

=== Summary ===

Correctly Classified Instances       42803               87.6356 %
Incorrectly Classified Instances      6039               12.3644 %
Kappa statistic                          0.6325
Mean absolute error                      0.1861
Root mean squared error                  0.3048
Relative absolute error                 51.1076 %
Root relative squared error             71.4388 %
Total Number of Instances            48842

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.631    0.046    0.810      0.631    0.710      0.640    0.907     0.792     >50K
                 0.954    0.369    0.891      0.954    0.921      0.640    0.907     0.960     <=50K
Weighted Avg.    0.876    0.292    0.872      0.876    0.871      0.640    0.907     0.920

=== Confusion Matrix ===

     a     b   <-- classified as
  7375  4312 |     a = >50K
  1727 35428 |     b = <=50K
```

OK



Classifier

RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Test options

● Use training set

○ Supplied test set

○ Cross-validation Folds 10

○ Percentage split % 66

(Nom) class

Result list (right-click for options)

23:21:30 - trees.J48

23:23:27 - trees.RandomForest

Classifier output

```
Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 15.17 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 6.52 seconds

=== Summary ===

Correctly Classified Instances       48774               99.8608 %
Incorrectly Classified Instances        68                0.1392 %
Kappa statistic                          0.9962
Mean absolute error                      0.0737
Root mean squared error                  0.1263
Relative absolute error                 20.2565 %
Root relative squared error             29.6022 %
Total Number of Instances            48842

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.995    0.000    1.000      0.995    0.997      0.996    1.000     1.000     >50K
                 1.000    0.005    0.998      1.000    0.999      0.996    1.000     1.000     <=50K
Weighted Avg.    0.999    0.004    0.999      0.999    0.999      0.996    1.000     1.000

=== Confusion Matrix ===

     a     b   <-- classified as
 11624    63 |     a = >50K
     5 37150 |     b = <=50K
```

Status

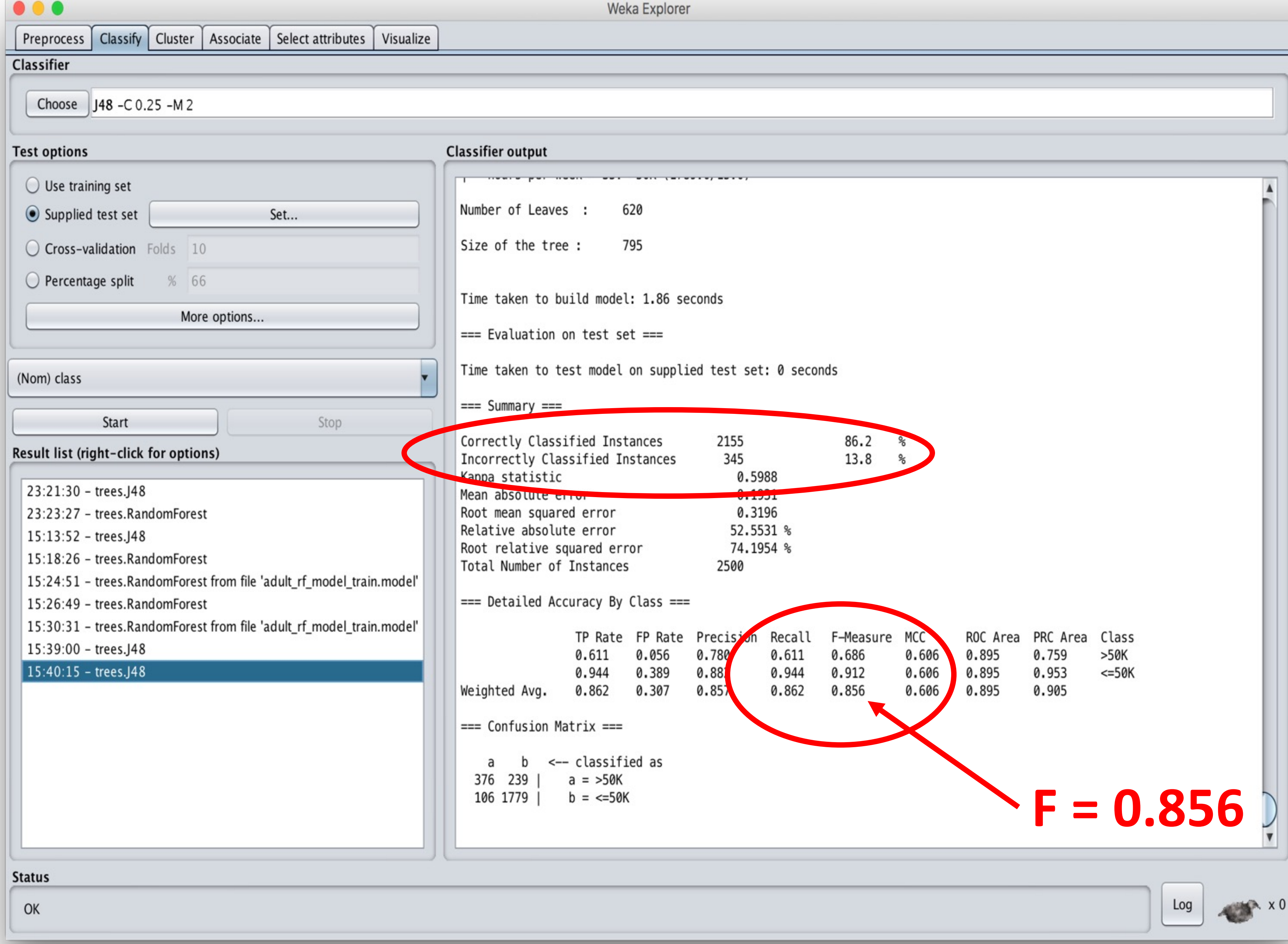
OK

Result

- Significant increase on F1 scores when both trained and evaluated on training set
- This is considered to be poor methodology since it overfits to the particular training set

Create train and test collection

- Train has ~95% of data, test 5%
- Train models for J48 and random forest using train dataset
- Test on test data set
- …



J48 -C 0.25 -M 2

○ Use training set
● Supplied test set
○ Cross-validation Folds 10
○ Percentage split % 66

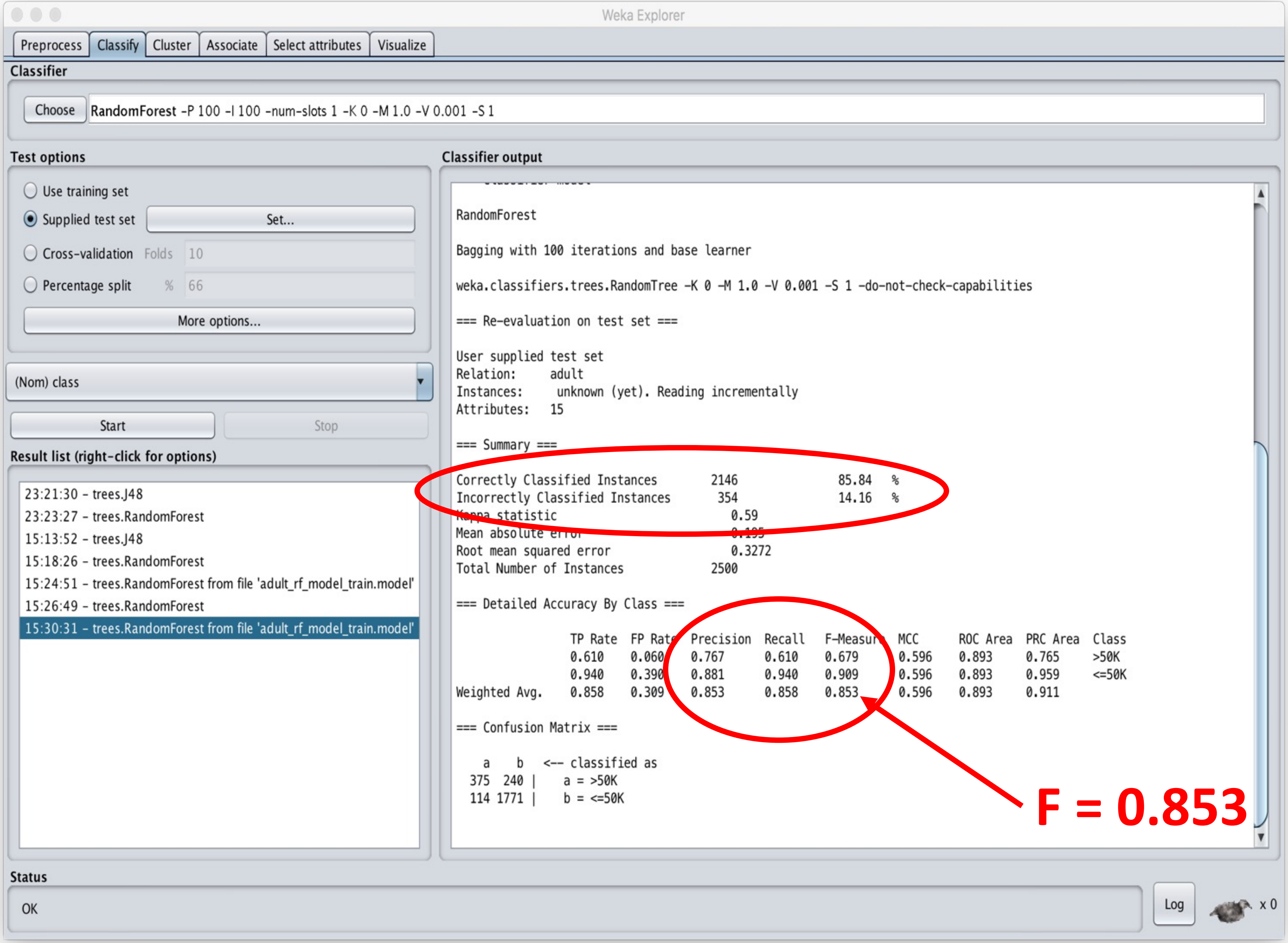
(Nom) class

```
23:21:30 - trees.J48
23:23:27 - trees.RandomForest
15:13:52 - trees.J48
15:18:26 - trees.RandomForest
15:24:51 - trees.RandomForest from file 'adult_rf_model_train.model'
15:26:49 - trees.RandomForest
15:30:31 - trees.RandomForest from file 'adult_rf_model_train.model'
15:39:00 - trees.J48
15:40:15 - trees.J48
```

```
Number of Leaves  :     620

Size of the tree :      795


Time taken to build model: 1.86 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances        2155               86.2    %
Incorrectly Classified Instances       345               13.8    %
Kappa statistic                          0.5988
Mean absolute error                      0.1931
Root mean squared error                  0.3196
Relative absolute error                 52.5531 %
Root relative squared error             74.1954 %
Total Number of Instances             2500

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.611    0.056    0.780      0.611    0.686      0.606    0.895     0.759     >50K
                 0.944    0.389    0.882      0.944    0.912      0.606    0.895     0.953     <=50K
Weighted Avg.    0.862    0.307    0.857      0.862    0.856      0.606    0.895     0.905

=== Confusion Matrix ===

    a    b   <-- classified as
  376  239 |    a = >50K
  106 1779 |    b = <=50K
```

OK

F = 0.856



RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

○ Use training set
● Supplied test set
○ Cross-validation Folds 10
○ Percentage split % 66

(Nom) class

```
23:21:30 - trees.J48
23:23:27 - trees.RandomForest
15:13:52 - trees.J48
15:18:26 - trees.RandomForest
15:24:51 - trees.RandomForest from file 'adult_rf_model_train.model'
15:26:49 - trees.RandomForest
15:30:31 - trees.RandomForest from file 'adult_rf_model_train.model'
```

```
RandomForest

Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

=== Re-evaluation on test set ===

User supplied test set
Relation:     adult
Instances:     unknown (yet). Reading incrementally
Attributes:   15

=== Summary ===

Correctly Classified Instances        2146               85.84   %
Incorrectly Classified Instances       354               14.16   %
Kappa statistic                          0.59
Mean absolute error                      0.195
Root mean squared error                  0.3272
Total Number of Instances             2500

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.610    0.060    0.767      0.610    0.679      0.596    0.893     0.765     >50K
                 0.940    0.390    0.881      0.940    0.909      0.596    0.893     0.959     <=50K
Weighted Avg.    0.858    0.309    0.853      0.858    0.853      0.596    0.893     0.911

=== Confusion Matrix ===

    a    b   <-- classified as
  375  240 |    a = >50K
  114 1771 |    b = <=50K
```

F = 0.853

OK

Create train and test collection

- Train has ~95% of data, test 5%
- Trained models for J48 and random forest using train dataset
- Tested on test data set
- Results were that random forest was (at best) **about the same** as J48
- Large dataset reduced problem of overfitting, so random forest did not help

Conclusions

- **Bagging** helps, especially if training data adequate, but not as large as it should be
  - With lots of data, overfitting less of a problem, so bagging may not help
- While we explore it using decision trees, it can be applied to any classifier
  - Scikit-learn has a **general module** for bagging
- In general, using any of several **ensemble** approaches to classification often helpful
- Training neural networks uses a different approach (dropout) to control overfitting

~~Conclusions~~

- Wait, there's more…
- A classification problem can change over time
  - E.g.: recognizing a spam message from its content and metadata
- We showed that an ensemble approach can detect a change in the nature of spam
  - Which tells us its time to retrain with new data
  - D. Chinavle, P. Kolari, T. Oates, and T. Finin, Ensembles in Adversarial Classification for Spam, ACM CIKM, 2009. link

Recognizing Concept Drift

- Build **ensemble of five models** to classify spam comments left on a blog at time T1
- Note the relative level of agreement
- Detect when one of the models starts to diverge from the others at time T2
  - Time to get new data and retrain
  - Examining disagreements can be enlightening
- We used temporal data spanning several years to verify its effectiveness
  - E.g., spam's focus shift from *viagra* to *weight loss*