

COMP47490 Tutorial

Ensembles

Aonghus Lawlor

**School of Computer Science
Autumn 2019**



Tutorial Q1(a)

Q. In Weka, load the *Wine* data set using the ARFF file provided, and evaluate a decision tree classifier (J48) using 10-fold cross-validation. What percentage of instances are correctly classified?

The screenshot shows the Weka Explorer window with the 'Classify' tab selected. The 'Classifier' section shows 'J48' with a confidence of '0.25' and a 'M 2' metric. The 'Test options' section has 'Cross-validation' selected with 'Folds' set to '10'. The 'Classifier output' section displays the results of the stratified cross-validation.

Classifier output Summary:

Stratified cross-validation Summary		
Correctly Classified Instances	167	93.8202 %
Incorrectly Classified Instances	11	6.1798 %
Kappa statistic	0.9058	
Mean absolute error	0.0485	
Root mean squared error	0.2019	
Relative absolute error	11.0723 %	
Root relative squared error	43.0865 %	
Total Number of Instances	178	

Detailed Accuracy By Class:

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC
Weighted Avg.	0.983	0.034	0.935	0.983	0.959	0.938
	0.944	0.056	0.918	0.944	0.931	0.884
	0.875	0.008	0.977	0.875	0.923	0.899
	0.938	0.035	0.940	0.938	0.938	0.906

Result list (right-click for options):

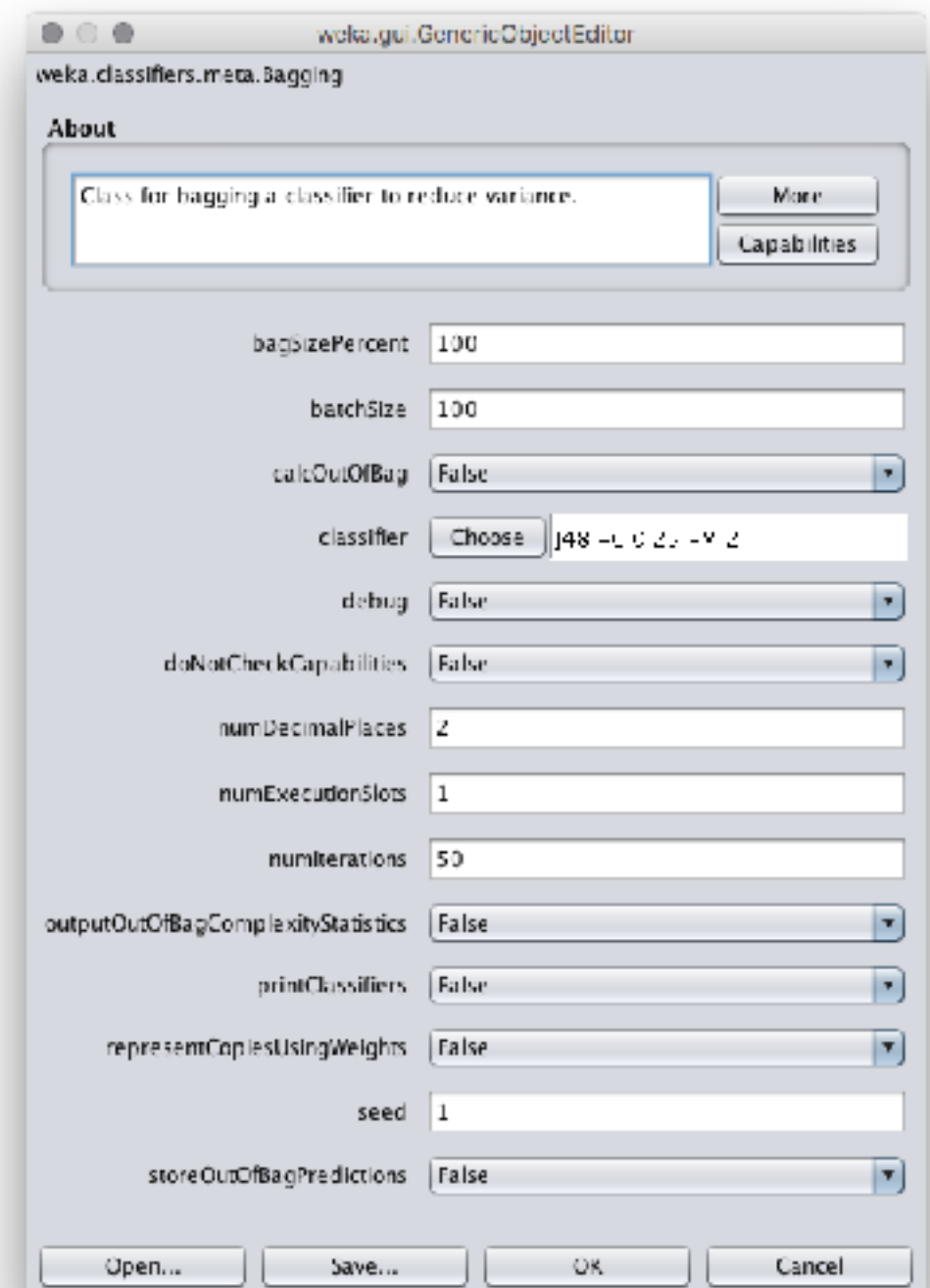
- 12:28:25 - treesJ48

Status: OK

Tutorial Q1(b)

Q. Apply ensemble classification using bagging to achieve diversity and with a decision tree classifier. What percentage of instances are now correctly classified with an ensemble of size 50?

1. Using Weka, click on the *Classify* tab.
2. Click *Choose*, select method *classifiers->meta->Bagging*.
3. Click *Bagging* in the box to the right. The configuration interface of the method appears.
4. Click *Choose*, select *J48*.
5. Set the *numIterations* to 50 Click *OK* button.
6. Click *Start* button to build the ensemble.



Tutorial Q1(b)

Classifier

Choose **Bagging** -P 100 -S 1 -num-slots 1 -I 50 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2

Test options

☐ Use training set
☐ Supplied test set Set...
☒ Cross-validation Folds **10**
☐ Percentage split % **66**
More options...

(Nom) class

Start Stop

Result list (right-click for options)

12:28:25 - trees.J48
13:31:15 - meta.Bagging

Classifier output

```
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances      169      94.9438 %
Incorrectly Classified Instances     9       5.0562 %
Kappa statistic                    0.9232
Mean absolute error                 0.0699
Root mean squared error             0.1553
Relative absolute error             15.9205 %
Root relative squared error         33.1555 %
Total Number of Instances          178

=== Detailed Accuracy By Class ===
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC
	0.966	0.025	0.950	0.965	0.958	0.937
	0.930	0.037	0.943	0.930	0.936	0.894
	0.958	0.015	0.958	0.958	0.958	0.943
Weighted Avg.	0.949	0.027	0.949	0.949	0.949	0.922

```
=== Confusion Matrix ===
 a  b  c  <-- classified as
```

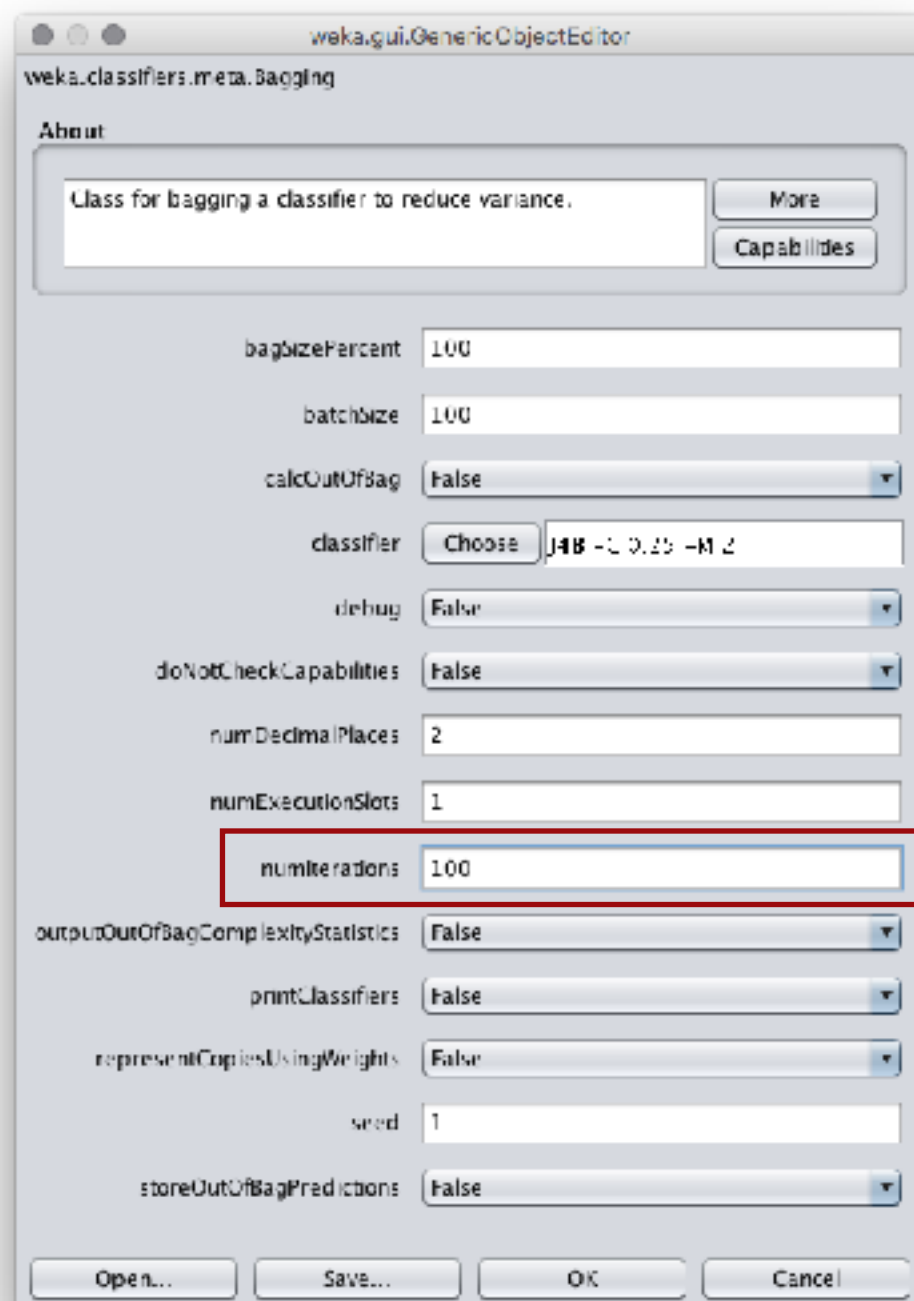
Status

OK

Improvement: 93.82% → 94.94%

Tutorial Q1(c)

Q. Repeat (b), but increase the ensemble size to 100, 250, then 500 classifiers. What level of improvement does this provide, in terms of percentage of instances correctly classified?

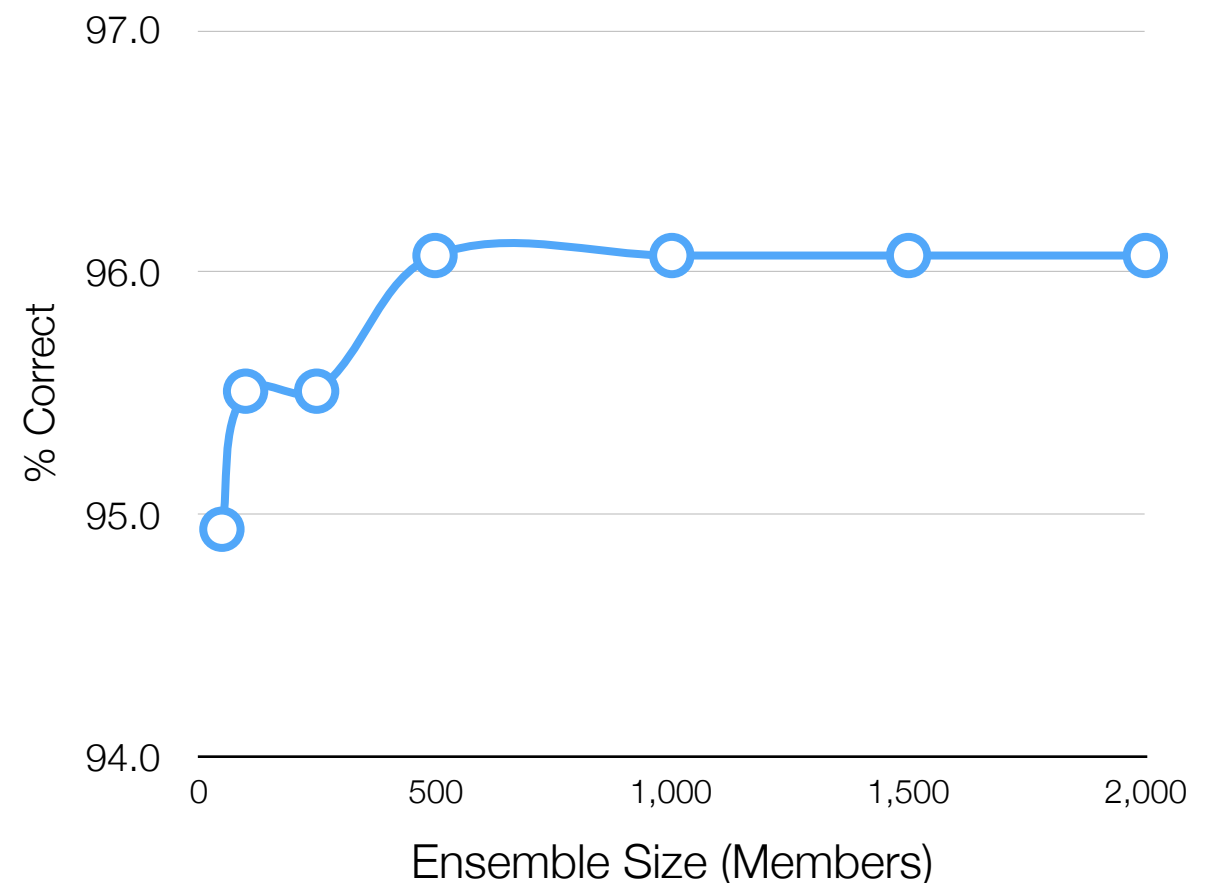


Ensemble Size	% Correct	% Incorrect
50	94.94	5.06
100	95.51	4.49
250	95.51	4.49
500	96.07	3.93

Tutorial Q1(d)

Q. Why does the level of improvement in accuracy often “level off” after an ensemble has been increased to a certain size?

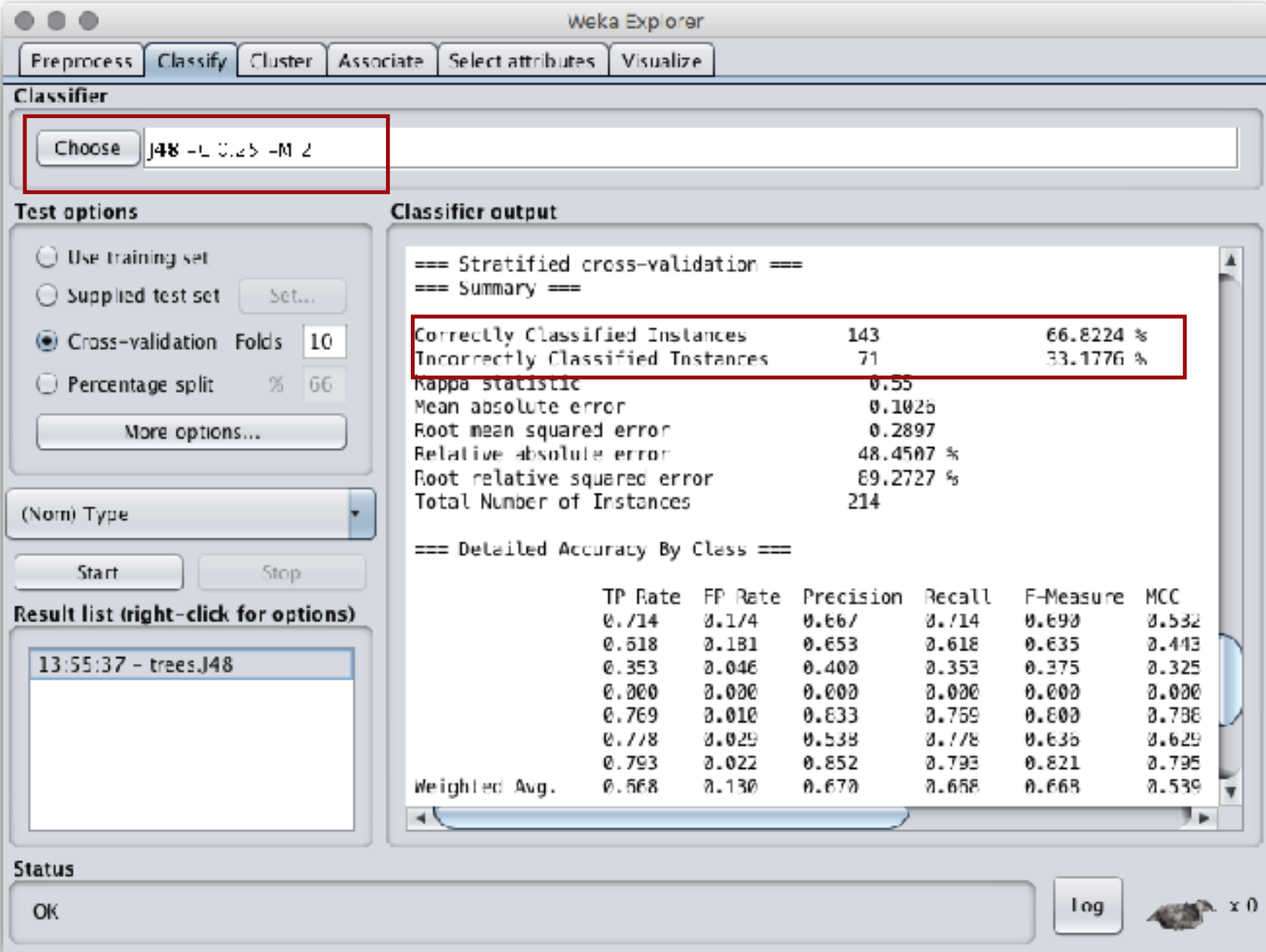
Ensemble Size	% Correct	% Incorrect
50	94.94	5.06
100	95.51	4.49
250	95.51	4.49
500	96.07	3.93
1000	96.07	3.93
1500	96.07	3.93
2000	96.07	3.93



- ➡ Eventually, new ensemble members will have prediction patterns collinear with existing members. No new diversity is added, so ensemble accuracy will plateau.

Tutorial Q2(a)

Q. In Weka, load the *Glass* data set using the ARFF file provided, and evaluate a decision tree classifier (J48) using 10-fold cross-validation. What percentage of instances are correctly classified?



The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The 'Classifier' section shows 'J48 -L 0.25 -M 2' selected. The 'Test options' section shows 'Cross-validation' selected with 'Folds' set to 10. The 'Classifier output' section displays the results of the stratified cross-validation.

Classifier output

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances	143	66.8224 %
Incorrectly Classified Instances	71	33.1776 %

Kappa statistic 0.55
Mean absolute error 0.1026
Root mean squared error 0.2897
Relative absolute error 48.4507 %
Root relative squared error 69.2727 %
Total Number of Instances 214

=== Detailed Accuracy By Class ===

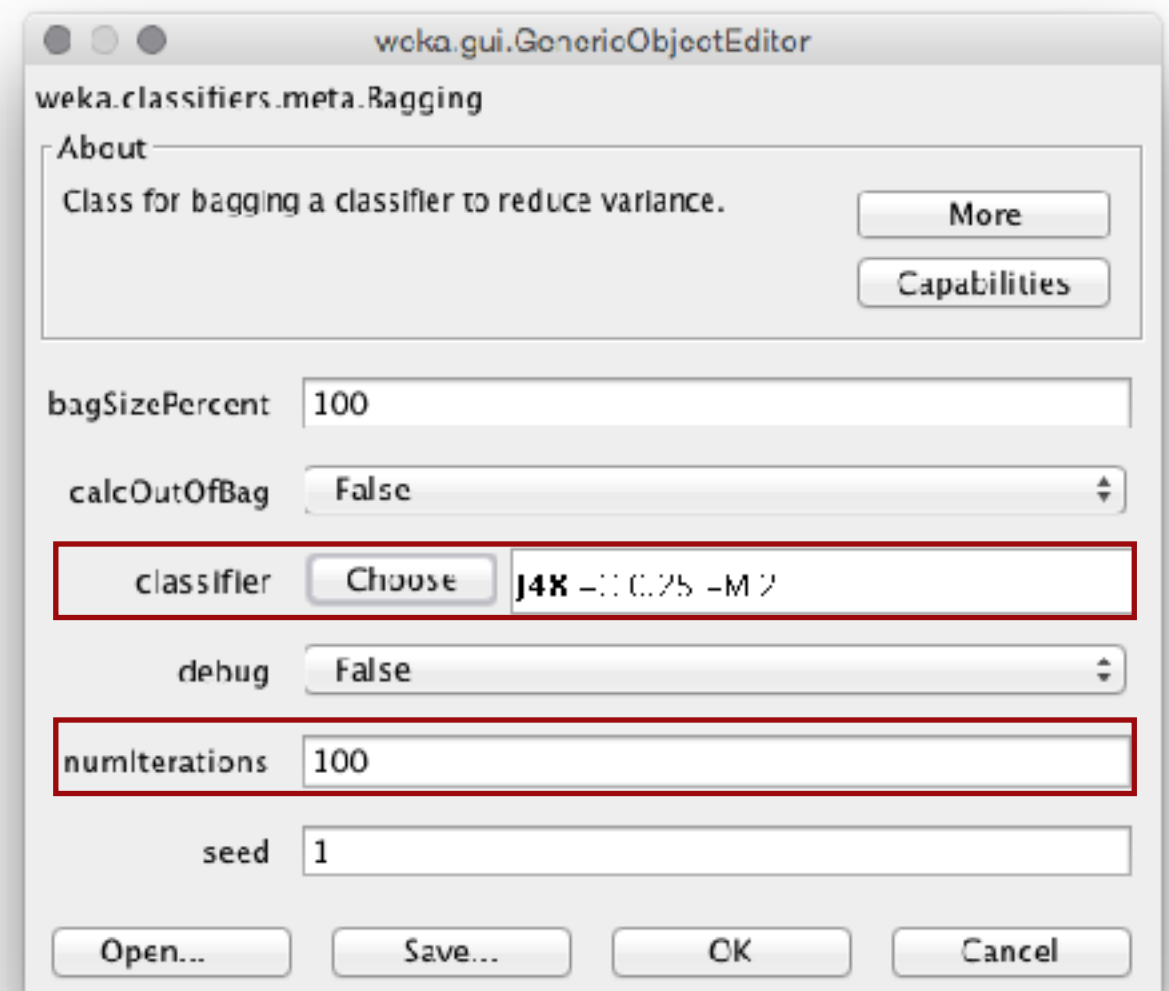
	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC
	0.714	0.174	0.667	0.714	0.690	0.532
	0.618	0.181	0.653	0.618	0.635	0.443
	0.353	0.046	0.400	0.353	0.375	0.325
	0.000	0.000	0.000	0.000	0.000	0.000
	0.769	0.010	0.833	0.769	0.800	0.736
	0.778	0.029	0.538	0.778	0.636	0.629
	0.793	0.022	0.852	0.793	0.821	0.795
Weighted Avg.	0.568	0.130	0.670	0.658	0.668	0.539

Status
OK

Tutorial Q2(b)

Q. Apply bagging with a decision tree classifier for an ensemble size of 100. What is the improvement over a single tree?

1. Using Weka, click on the *Classify* tab.
2. Click *Choose*, select method *classifiers->meta->Bagging*.
3. Click *Bagging* in the box to the right. The configuration interface of the method appears.
4. Click *Choose*, select *J48*.
5. Set the *numIterations* to 100.
6. Click *OK* button.
7. Click *Start* button to build the ensemble.



Tutorial Q2(b)

Weka Explorer

Preprocess **Classify** Cluster Associate Select attributes Visualize

Classifier

Choose **Bagging** -F 100 -E 1 -num-slots 1 -I 100 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2

Test options

☐ Use training set
☐ Supplied test set Set...
☒ Cross-validation Folds **10**
☐ Percentage split % **66**
More options...

(Nom) Type

Start Stop

Result list (right-click for options)

- 13:55:37 - trees.J48
- 13:56:56 - meta.Bagging
- 13:57:06 - meta.Bagging**

Classifier output

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances	159	74.2991 %
Incorrectly Classified Instances	55	25.7009 %
Kappa statistic	0.6496	
Mean absolute error	0.1035	
Root mean squared error	0.2246	
Relative absolute error	48.8939 %	
Root relative squared error	69.2045 %	
Total Number of Instances	214	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC
	0.843	0.125	0.766	0.843	0.803	0.702
	0.711	0.138	0.740	0.711	0.725	0.578
	0.235	0.020	0.500	0.235	0.320	0.306
	0.000	0.000	0.000	0.000	0.000	0.000
	0.692	0.020	0.692	0.692	0.692	0.672
	0.889	0.029	0.571	0.889	0.696	0.698
	0.862	0.022	0.862	0.852	0.862	0.840
Weighted Avg.	0.743	0.097	0.736	0.743	0.734	0.643

Status

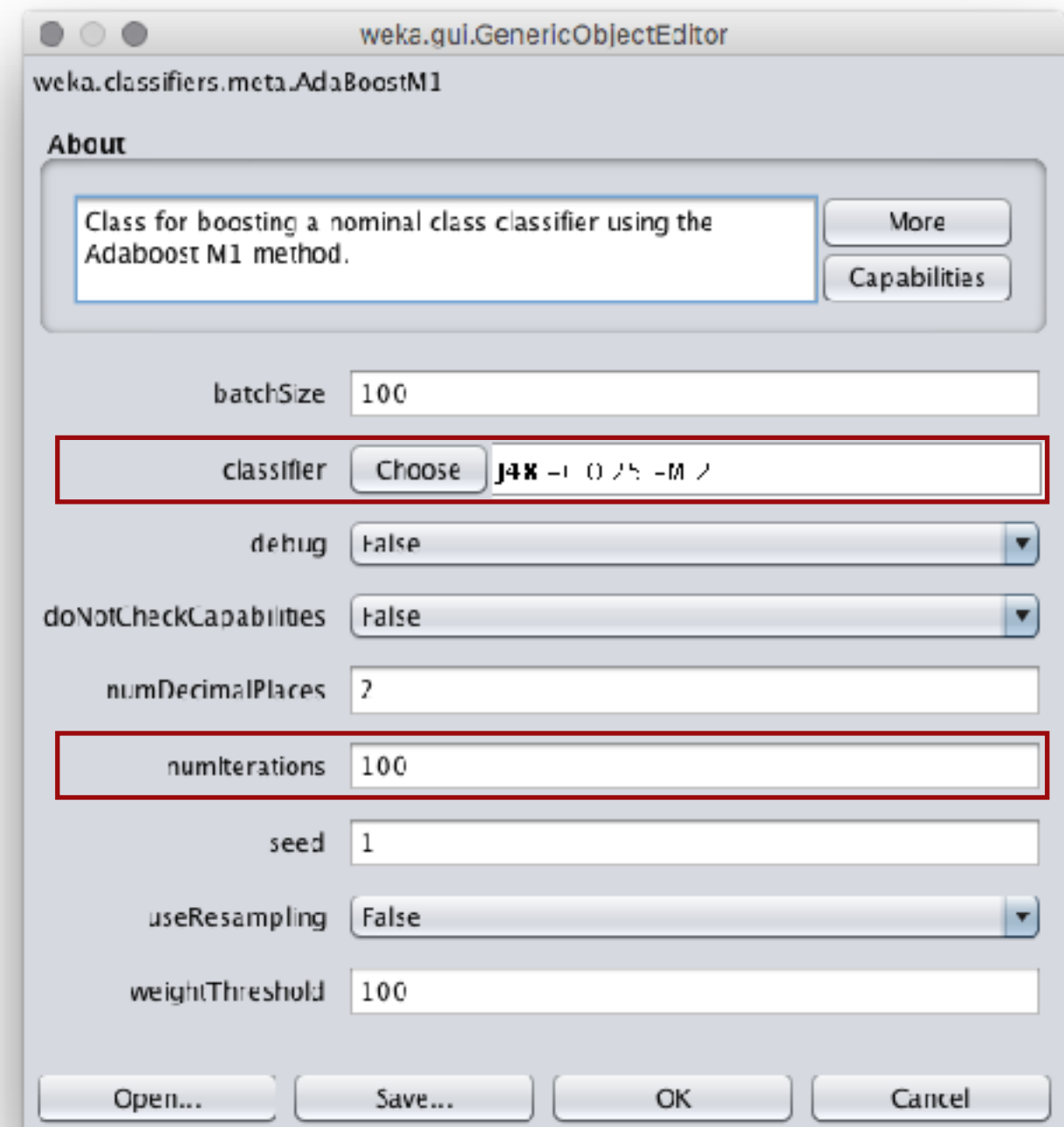
OK

Improvement: 66.82% → 74.30%

Tutorial Q2(c)

Q. Now apply *boosting* with a decision tree classifier for an ensemble size of 100. How does it compare to the results from (b)? How do you explain this difference?

1. Using Weka, click on the *Classify* tab.
2. Click *Choose*, select method *classifiers->meta->AdaBoostM1*.
3. Click *AdaBoostM1* in the box to the right. The configuration interface of the method appears.
4. Click *Choose*, select *J48*.
5. Set the *numIterations* to 100.
6. Click *OK* button.
7. Click *Start* button to build the ensemble.



Tutorial Q2(c)

- ➔ Boosting adds diversity to the ensemble, while also improving accuracy on the difficult examples.

The screenshot shows the Weka Explorer window with the 'Classify' tab selected. The classifier chosen is 'AdaBoostM1' with the command line: `AdaBoostM1 -P 100 -S 1 -I 100 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2`. The 'Test options' section shows 'Cross-validation' selected with 'Folds' set to 10. The 'Result list' on the left shows four entries: '13:55:37 - trees.J48', '13:56:56 - meta.Bagging', '13:57:06 - meta.Bagging', and '14:02:19 - meta.AdaBoostM1'. The 'Classifier output' pane displays the results for the selected model (AdaBoostM1).

Stratified cross-validation Summary

Metric	Value	Percentage
Correctly Classified Instances	170	79.4393 %
Incorrectly Classified Instances	44	20.5607 %
Kappa statistic	0.717	
Mean absolute error	0.2584	
Root mean squared error	0.2407	
Relative absolute error	27.5913 %	
Root relative squared error	74.1669 %	
Total Number of Instances	214	

Detailed Accuracy By Class

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC
Class 0	0.886	0.104	0.805	0.836	0.844	0.764
Class 1	0.803	0.123	0.782	0.803	0.792	0.676
Class 2	0.412	0.020	0.636	0.412	0.500	0.479

Result list (right-click for options)

- 13:55:37 - trees.J48
- 13:56:56 - meta.Bagging
- 13:57:06 - meta.Bagging
- 14:02:19 - meta.AdaBoostM1

Status

OK

Bagging Improvement: 66.82% → 74.30%

Boosting Improvement: 66.82% → 79.44%

Tutorial Q3(a)

Q. In Weka, load the *Glass* data set. Evaluate a k-NN classifier with $k=2$ neighbours using 10-fold cross-validation. What percentage of instances are correctly classified?

The screenshot shows the Weka Explorer window with the 'Classify' tab selected. The classifier chosen is 'IBk' with parameters '-k 2 -W 0 -A \'weka.core.neighborsearch.LinearNNSearch -A \'weka.core.EuclideanDistance -R first-last\''. The test options are set to 'Cross-validation' with 'Folds' set to 10. The classifier output is displayed, showing a summary of results. A red box highlights the 'Correctly Classified Instances' and 'Incorrectly Classified Instances' rows. The 'Result list' on the left shows a single entry '14:23:31 - lazy.IBk'. The status bar at the bottom indicates 'OK'.

Classifier output Summary

Metric	Value	Percentage
Correctly Classified Instances	145	67.757 %
Incorrectly Classified Instances	69	32.243 %
Kappa statistic	0.5476	
Mean absolute error	0.0955	
Root mean squared error	0.2675	
Relative absolute error	45.1011 %	
Root relative squared error	82.4098 %	
Total Number of Instances	214	

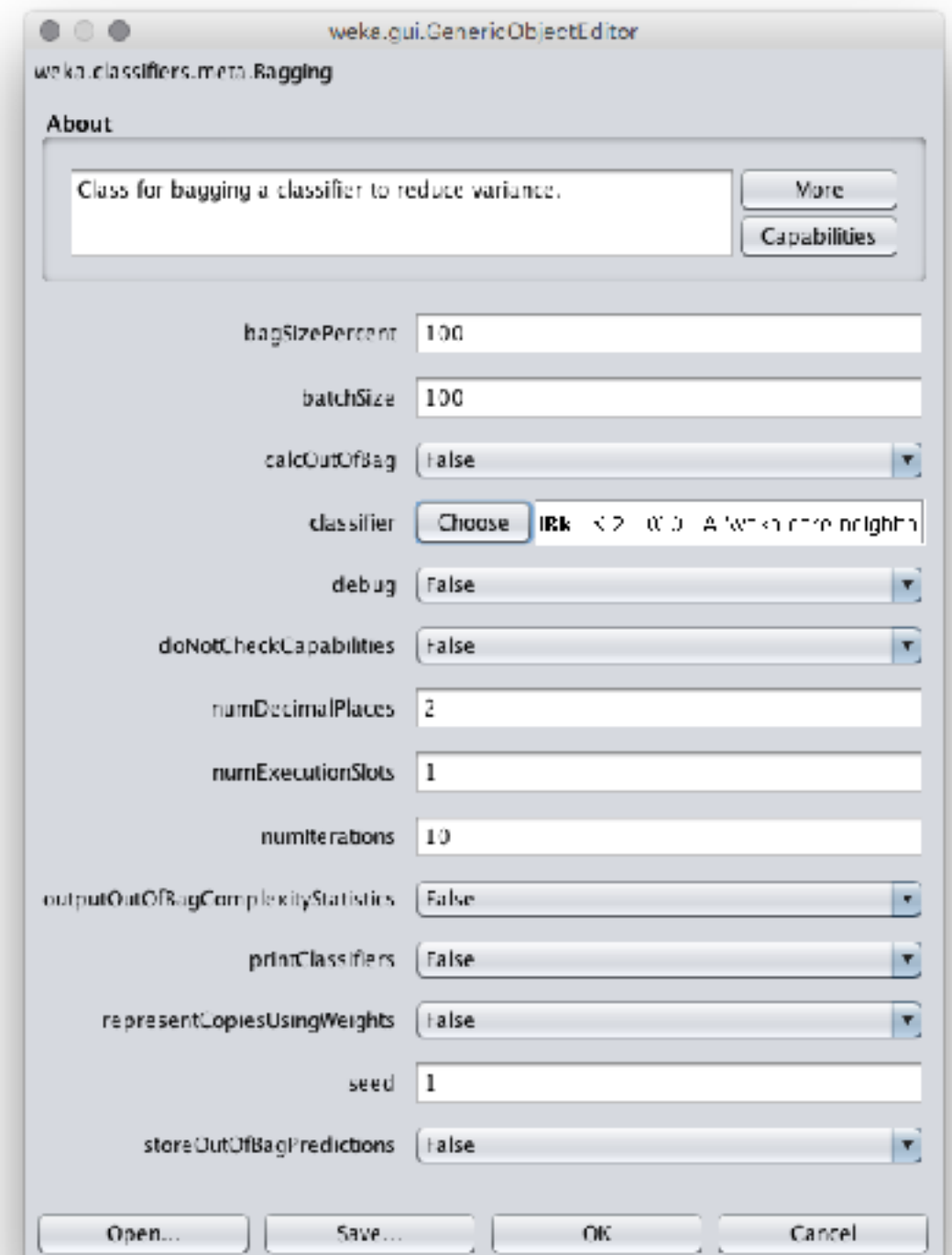
Detailed Accuracy By Class

	TP Rate	FP Rate	Precision	Recall	F-Measure	NCC
0.857	0.264	0.612	0.857	0.714	0.559	
0.618	0.167	0.671	0.618	0.644	0.461	
0.059	0.015	0.250	0.059	0.095	0.087	
0.000	0.000	0.000	0.000	0.000	0.000	
0.692	0.015	0.750	0.692	0.720	0.703	
0.556	0.005	0.833	0.556	0.667	0.670	
0.793	0.005	0.958	0.793	0.868	0.854	
Weighted Avg.	0.578	0.149	0.569	0.673	0.659	0.540

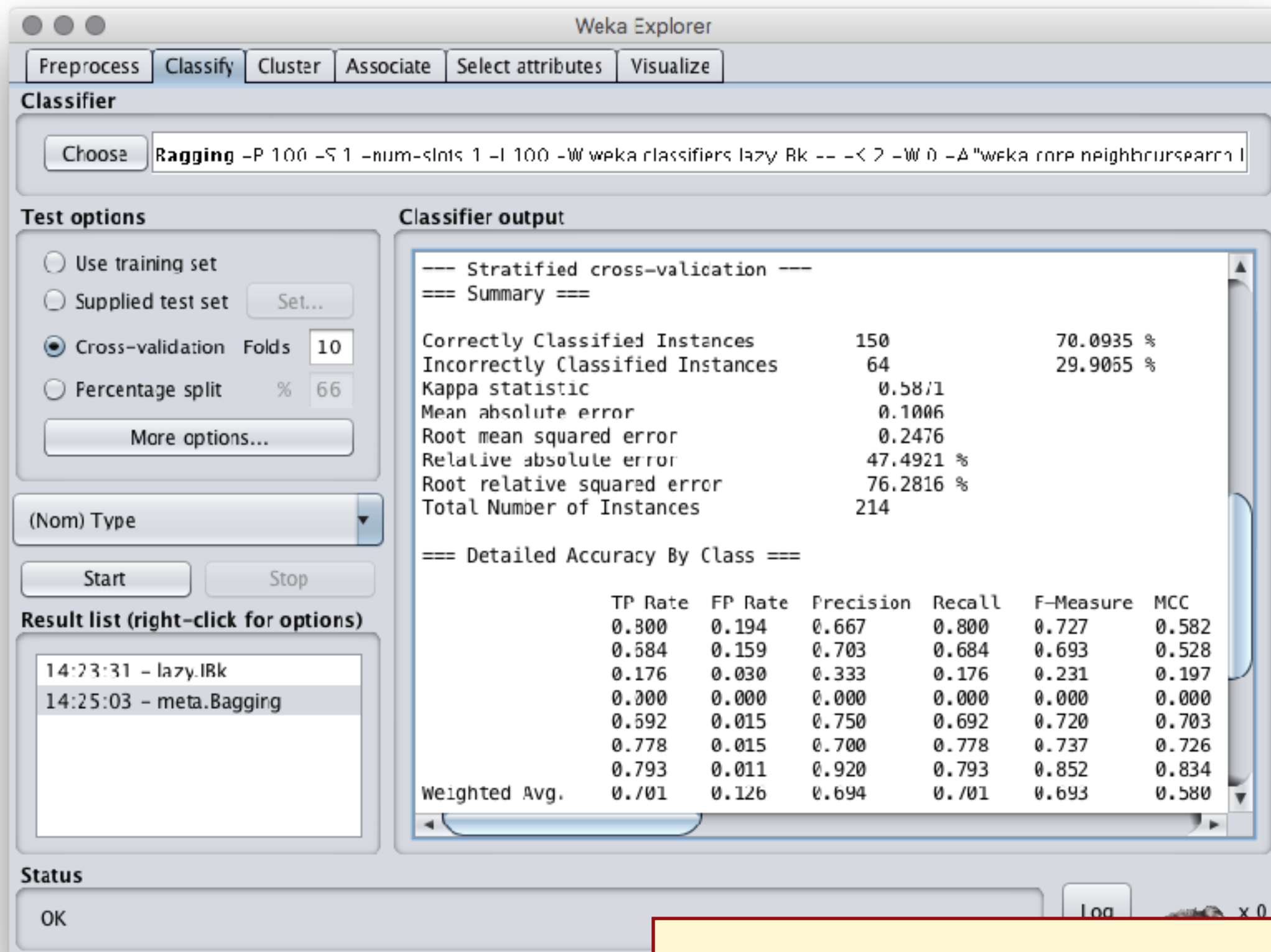
Tutorial Q3(b)

Q. Apply bagging with with a k-NN classifier ($k=2$) for an ensemble size of 100. What is the improvement?

1. Using Weka, click on the *Classify* tab.
2. Click *Choose*, select method *classifiers->meta->Bagging*.
3. Click *Bagging* in the box to the right. The configuration interface of the method appears.
4. Click *Choose*, select *IBk*.
5. Click *IBk*. In the configuration window, set *KNN* to 2. Close window.
6. Set the *numIterations* to 100.
7. Click *OK* button.
8. Click *Start* button to build the ensemble.



Tutorial Q3(b)



Weka Explorer

Preprocess | **Classify** | Cluster | Associate | Select attributes | Visualize

Classifier

Choose **Bagging** -P 100 -S 1 -num-slots 1 -I 100 -W weka.classifiers.lazy.Rk -- -< 2 -W 0 -A "weka.core.neighborsearch.NN1"

Test options

- ☐ Use training set
- ☐ Supplied test set Set...
- ☒ Cross-validation Folds **10**
- ☐ Percentage split % **66**

More options...

(Nom) Type

Start Stop

Result list (right-click for options)

- 14:23:31 - lazy.Rk
- 14:25:03 - meta.Bagging

Classifier output

--- Stratified cross-validation ---
=== Summary ===

Correctly Classified Instances	150	70.0935 %
Incorrectly Classified Instances	64	29.9055 %
Kappa statistic	0.5871	
Mean absolute error	0.1006	
Root mean squared error	0.2476	
Relative absolute error	47.4921 %	
Root relative squared error	76.2816 %	
Total Number of Instances	214	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC
	0.800	0.194	0.667	0.800	0.727	0.582
	0.584	0.159	0.703	0.684	0.693	0.528
	0.176	0.030	0.333	0.176	0.231	0.197
	0.000	0.000	0.000	0.000	0.000	0.000
	0.592	0.015	0.750	0.692	0.720	0.703
	0.778	0.015	0.700	0.778	0.737	0.726
	0.793	0.011	0.920	0.793	0.852	0.834
Weighted Avg.	0.701	0.126	0.694	0.701	0.693	0.580

Status

OK Log x 0

Improvement: 67.76% → 70.09%

Tutorial Q3(c)

Q. Now apply *random subspacing* with a k-NN classifier ($k=3$) for an ensemble size of 100. How does it compare to the results from (b)? How do you explain this difference?

1. Using Weka, click on the *Classify* tab.
2. Click *Choose*, select method *classifiers->meta->RandomSubSpace*.
3. Click *RandomSubSpace* in the box to the right. The configuration interface of the method appears.
4. Click *Choose*, select *IBk*.
5. Click *IBk*. In the configuration window, set *KNN* to 3. Close window.
6. Set the *numIterations* to 100.
7. Click *OK* button.
8. Click *Start* button to build the ensemble.



Tutorial Q3(c)

- Bagging is ineffective using a stable classifier (k-NN).
- Random subspace adds more instability into the classifier (diversity)
⇒ different features used when calculating distances.

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier chosen is 'RandomSubSpace' with parameters: `-F 0.5 -S 1 -num-slots 1 - 100 -W weka.classifiers.lazy.IBk -K 2 -W 0 -A "weka.core.neig+bou"`. The test options are set to 'Cross-validation' with 'Folds' set to 10. The classifier output shows a summary of results for the 'RandomSubSpace' classifier.

Summary

Metric	Value	Percentage
Correctly Classified Instances	159	74.2991 %
Incorrectly Classified Instances	55	25.7009 %
Kappa statistic	0.6443	
Mean absolute error	0.1013	
Root mean squared error	0.2152	
Relative absolute error	47.8218 %	
Root relative squared error	66.2965 %	
Total Number of Instances	214	

Detailed Accuracy By Class

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC
0.829	0.160	0.716	0.829	0.768	0.847	
0.750	0.152	0.731	0.750	0.740	0.594	
0.235	0.015	0.571	0.235	0.333	0.335	
0.000	0.000	0.000	0.000	0.000	0.000	
0.846	0.015	0.786	0.846	0.815	0.803	
0.667	0.015	0.667	0.667	0.667	0.652	

Result list (right-click for options)

- 14:23:31 - lazy.IBk
- 14:25:03 - meta.Bagging
- 14:30:15 - meta.RandomSubSpace

Status

OK

Bagging: Improvement: 67.76% → 68.22%

Subspacing: Improvement: 67.76% → 74.23%