

COMP30120 Tutorial

Ensembles

Derek Greene

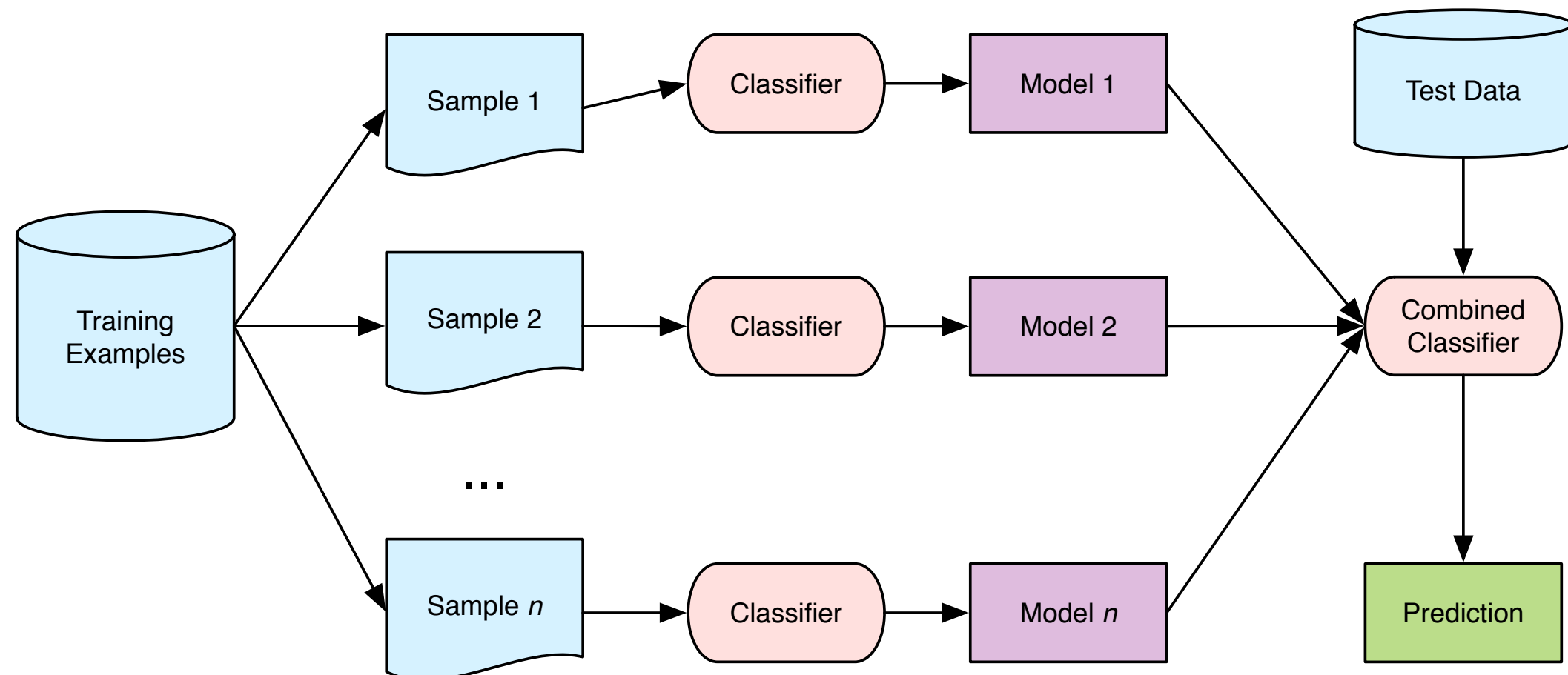
School of Computer Science and Informatics
Autumn 2015



Tutorial Q1(a)

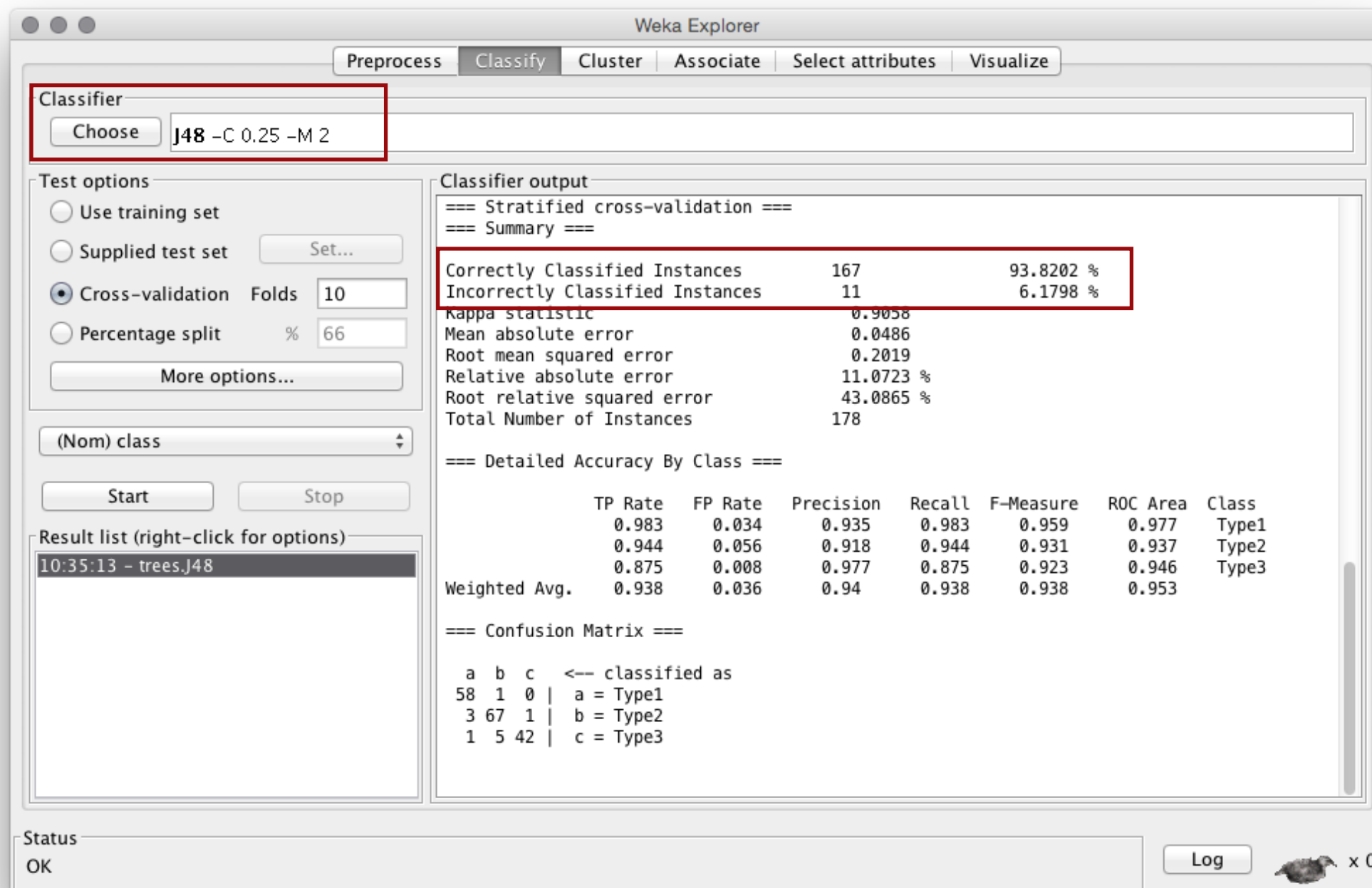
Q. *Bagging* (bootstrap aggregation) has a mechanism for achieving diversity for ensemble classifiers. Explain how it works.

1. Randomly sample from training data with replacement.
2. Apply a classifier to each sample independently.
3. Combine the outputs of the classifiers (e.g. majority voting).



Tutorial Q1(b)

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' dropdown is set to 'J48 -C 0.25 -M 2'. Under 'Test options', 'Cross-validation' is selected with 'Folds' set to 10. The 'Result list' shows a single entry: '10:35:13 - trees.J48'. The 'Classifier output' pane displays the following results:

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

Correctly Classified Instances	167	93.8202 %
Incorrectly Classified Instances	11	6.1798 %

Kappa statistic 0.9058
Mean absolute error 0.0486
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	ROC Area	Class
	0.983	0.034	0.935	0.983	0.959	0.977	Type1
	0.944	0.056	0.918	0.944	0.931	0.937	Type2
	0.875	0.008	0.977	0.875	0.923	0.946	Type3
Weighted Avg.	0.938	0.036	0.94	0.938	0.938	0.953	

=== Confusion Matrix ===

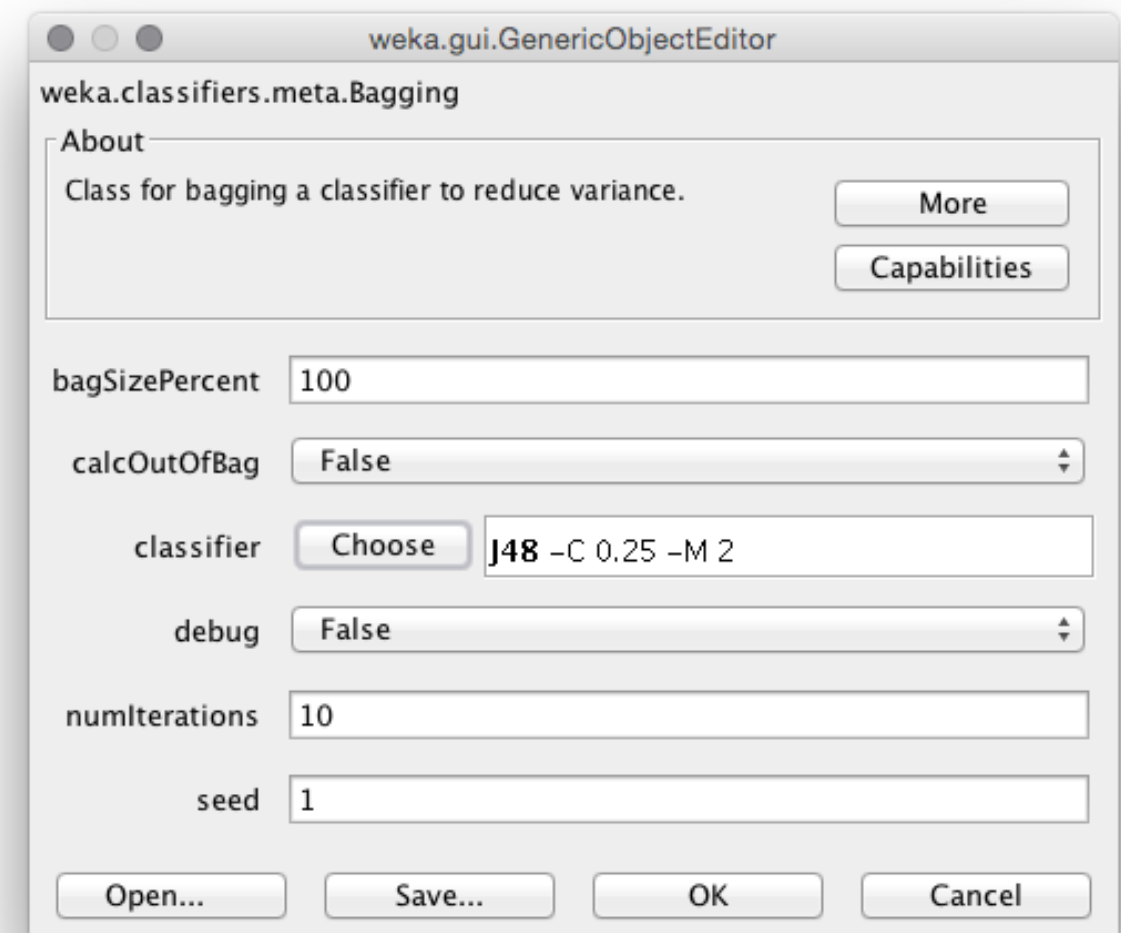
a	b	c	<-- classified as
58	1	0	a = Type1
3	67	1	b = Type2
1	5	42	c = Type3

Status: OK

Tutorial Q1(c)

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 10?

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 10 (default).
6. Click *OK* button.
7. Click *Start* button to build the ensemble.



Tutorial Q1(c)

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The 'Classifier' dropdown is set to 'Bagging' with the command line: `-P 100 -S 1 -I 10 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2`. The 'Test options' section shows 'Cross-validation' selected with 'Folds' set to 10. The 'Classifier output' section displays the following results:

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	168	94.382 %
Incorrectly Classified Instances	10	5.618 %
Kappa statistic	0.915	
Mean absolute error	0.0693	
Root mean squared error	0.1657	
Relative absolute error	15.7898 %	
Root relative squared error	35.3632 %	
Total Number of Instances	178	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.966	0.034	0.934	0.966	0.95	0.996	Type1
	0.901	0.019	0.97	0.901	0.934	0.992	Type2
	0.979	0.031	0.922	0.979	0.949	0.995	Type3
Weighted Avg.	0.944	0.027	0.945	0.944	0.944	0.994	

=== Confusion Matrix ===

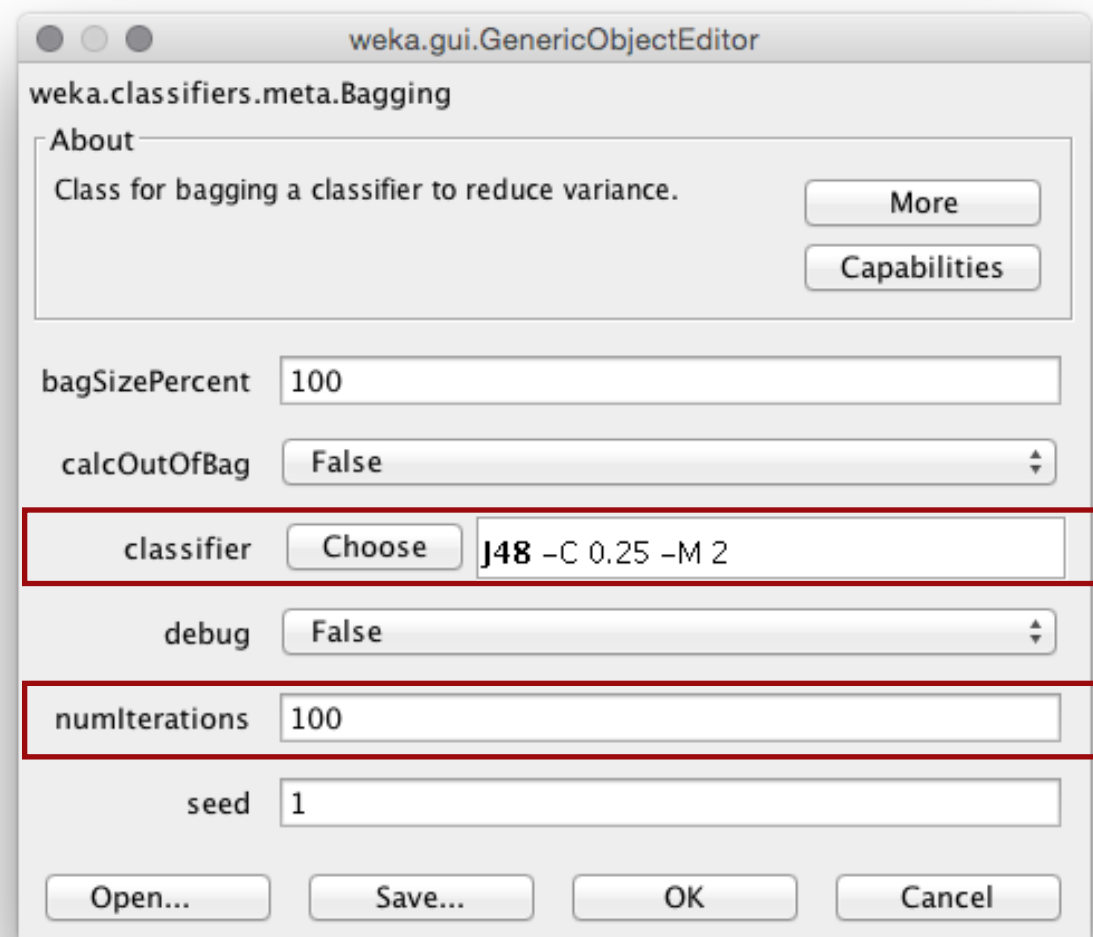
a	b	c	<-- classified as
57	1	1	a = Type1
4	64	3	b = Type2
0	1	47	c = Type3

Status: OK

Improvement: 93.82% → 94.38%

Tutorial Q1(d)

Q. Repeat (c), but increase the ensemble size to 100, 200, then 300 classifiers. What level of improvement does this provide, in terms of percentage of instances correctly classified?

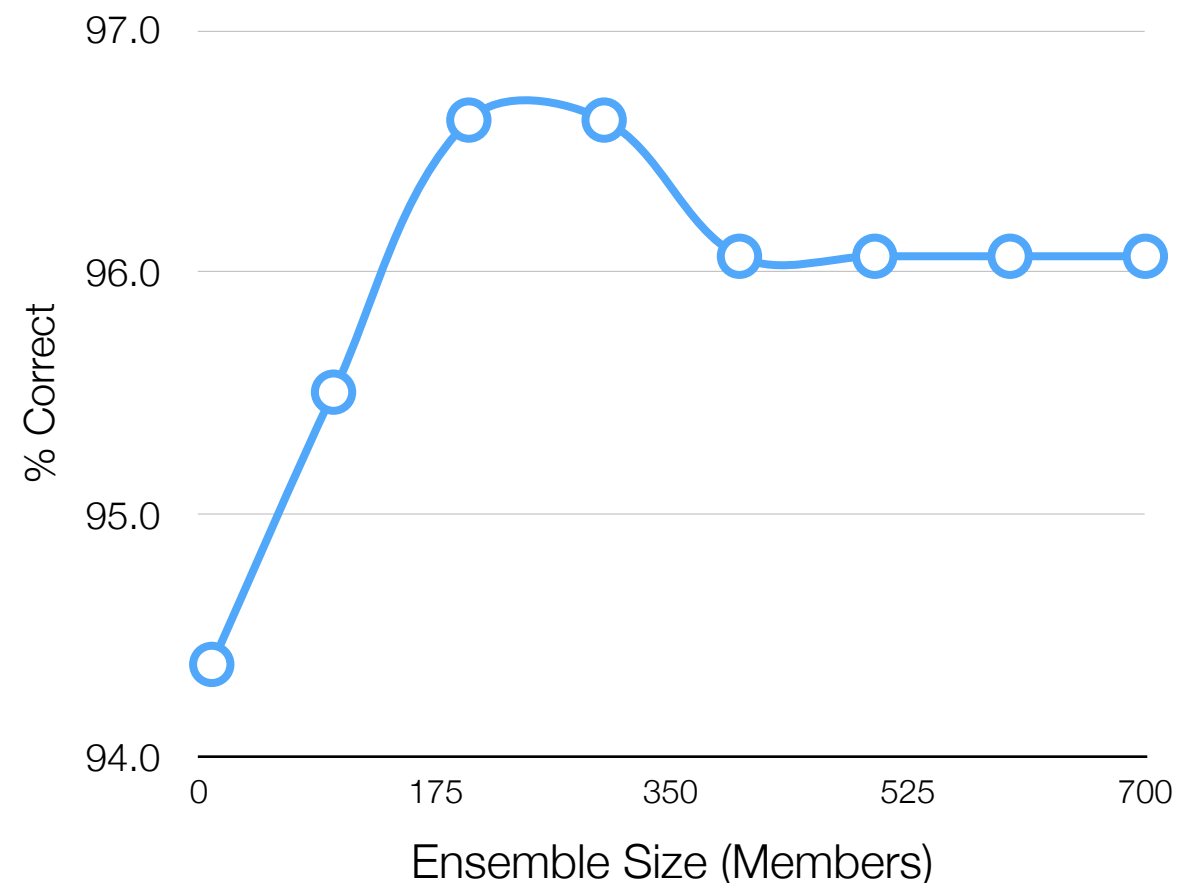


Ensemble Size	% Correct	% Incorrect
10	94.38	5.62
100	95.51	4.49
200	96.63	3.37
300	96.63	3.37

Tutorial Q1(e)

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
10	94.38	5.62
100	95.51	4.49
200	96.63	3.37
300	96.63	3.37
400	96.07	3.93
500	96.07	3.93
600	96.07	3.93
700	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. Explain what differentiates the ensemble members in a *boosting* classifier ensemble.
- ➡ *Boosting*: Train a series of classifiers such that later classifiers are trained to better predict on examples that earlier ones perform poorly on.
 - ➡ Focus on previous errors when building next ensemble member by adjusting weights for all examples.
 - ➡ This leads to both diversity (disagreement) between the base classifiers and increase in overall ensemble accuracy.

Tutorial Q2(b)

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 window with the 'Classify' tab selected. The classifier is set to 'J48 -C 0.25 -M 2'. The test options are set to 'Cross-validation' with 'Folds' set to '10'. The classifier output is displayed, showing the following results:

Metric	Value	Percentage
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	89.2727 %	
Total Number of Instances	214	

Below the classifier output, the 'Detailed Accuracy By Class' is shown:

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.714	0.174	0.667	0.714	0.69	0.806	build wind float
0.618	0.181	0.653	0.618	0.635	0.768	build wind non-float
0.353	0.046	0.4	0.353	0.375	0.766	vehic wind float
0	0	0	0	0	?	vehic wind non-float
0.769	0.01	0.833	0.769	0.8	0.872	containers
0.778	0.029	0.538	0.778	0.636	0.93	tableware
0.793	0.022	0.852	0.793	0.821	0.869	headlamps
Weighted Avg.	0.668	0.13	0.67	0.668	0.668	0.807

The 'Confusion Matrix' is also displayed:

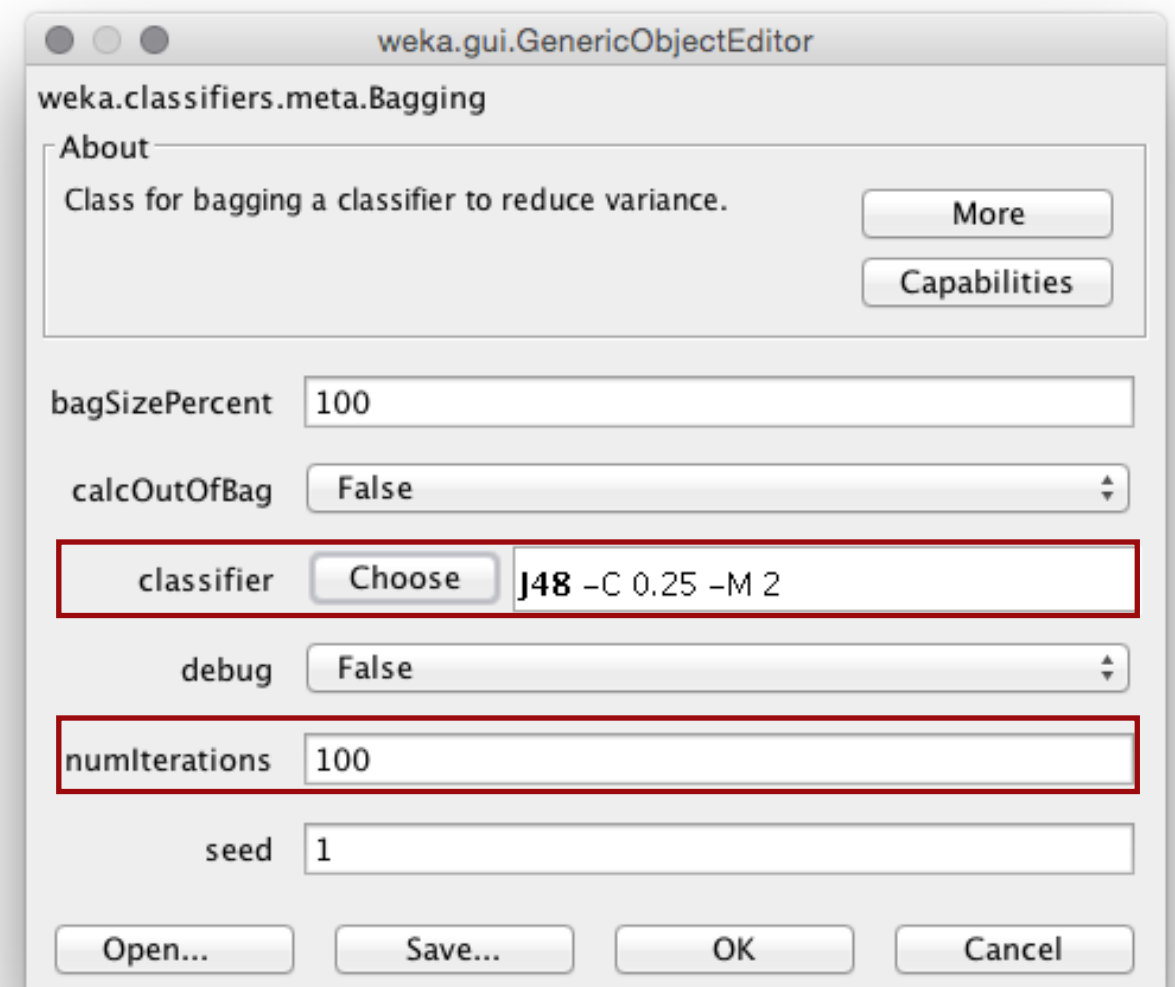
a	b	c	d	e	f	g	<-- classified as
50	15	3	0	0	1	1	a = build wind float
16	47	6	0	2	3	2	b = build wind non-float
5	5	6	0	0	1	0	c = vehic wind float
0	0	0	0	0	0	0	d = vehic wind non-float
0	2	0	0	10	0	1	e = containers
1	1	0	0	0	7	0	f = tableware
3	2	0	0	0	1	23	g = headlamps

The status bar at the bottom shows 'Status OK' and a 'Log' button.

Tutorial Q2(c)

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(c)

Weka Explorer

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

Classifier: Choose **Bagging** -P 100 -S 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)

- 14:27:24 - trees.J48
- 14:30:05 - meta.Bagging**

Classifier output

Correctly Classified Instances 161 75.2336 %
Incorrectly Classified Instances 53 24.7664 %
Kappa statistic 0.6605
Mean absolute error 0.1034
Root mean squared error 0.225
Relative absolute error 48.8089 %
Root relative squared error 69.3144 %
Total Number of Instances 214

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.843	0.146	0.738	0.843	0.787	0.919	build wind float
	0.724	0.13	0.753	0.724	0.738	0.884	build wind non-float
	0.294	0.015	0.625	0.294	0.4	0.879	vehic wind float
	0	0	0	0	0	?	vehic wind non-float
	0.692	0.01	0.818	0.692	0.75	0.944	containers
	0.889	0.024	0.615	0.889	0.727	0.993	tableware
	0.862	0.022	0.862	0.862	0.862	0.944	headlamps
Weighted Avg.	0.752	0.1	0.751	0.752	0.744	0.912	

=== Confusion Matrix ===

	a	b	c	d	e	f	g	
59	8	1	0	0	1	1	1	<-- classified as
13	55	2	0	2	2	2	2	a = build wind float
6	5	5	0	0	1	0	0	b = build wind non-float
0	0	0	0	0	0	0	0	c = vehic wind float
0	3	0	0	9	0	1	1	d = vehic wind non-float
1	0	0	0	0	8	0	0	e = containers
1	2	0	0	0	1	25	25	f = tableware
								g = headlamps

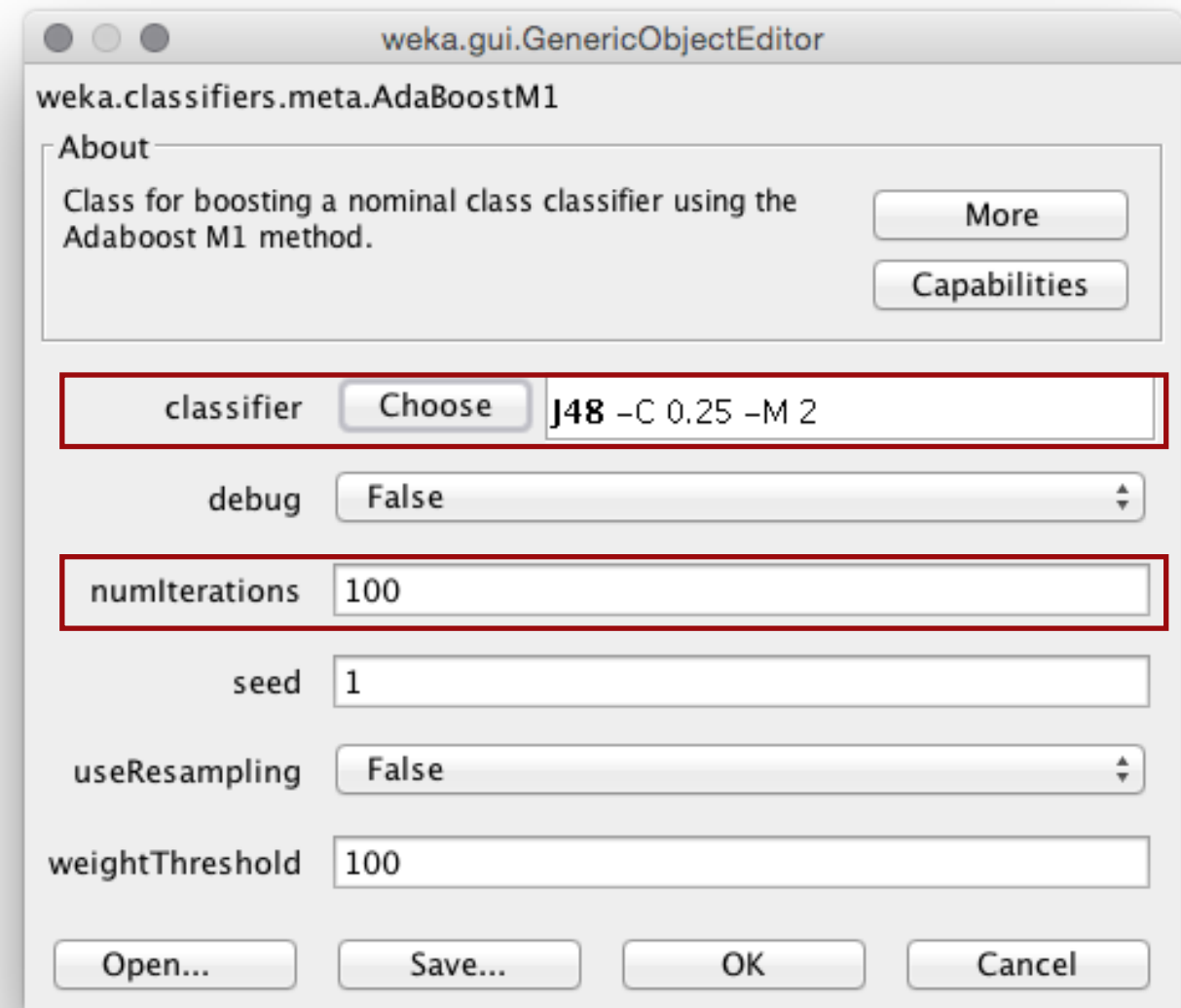
Status: OK

Improvement: 66.82% → 75.23%

Tutorial Q2(d)

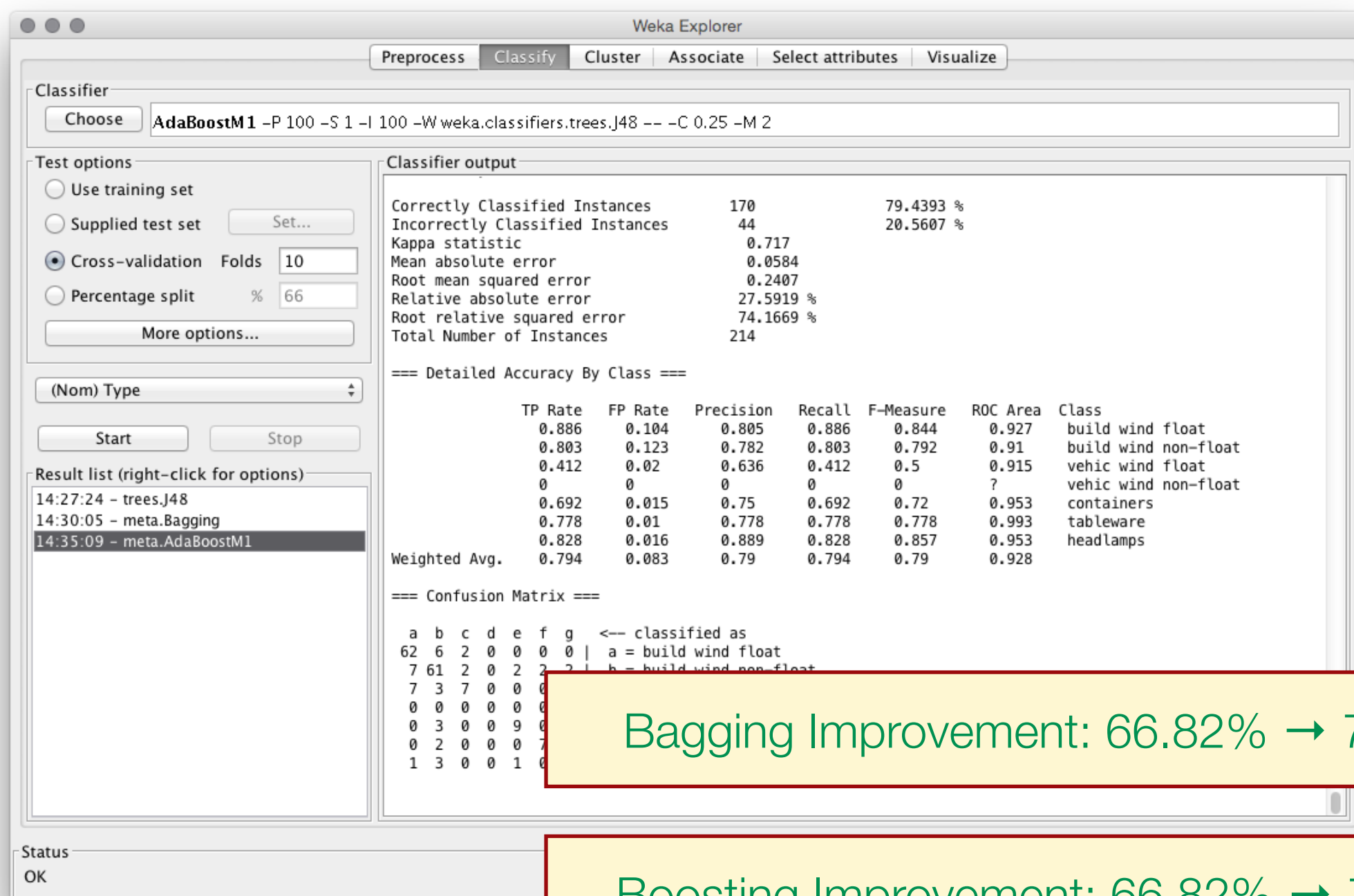
Q. Now apply *boosting* with a decision tree classifier for an ensemble size of 100. How does it compare to the results from (c)? How do you explain this difference?

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(d)

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



The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier chosen is 'AdaBoostM1' with parameters: `-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 10 folds. The 'Classifier output' section displays the following results:

Metric	Value	Percentage
Correctly Classified Instances	170	79.4393 %
Incorrectly Classified Instances	44	20.5607 %
Kappa statistic	0.717	
Mean absolute error	0.0584	
Root mean squared error	0.2407	
Relative absolute error	27.5919 %	
Root relative squared error	74.1669 %	
Total Number of Instances	214	

Below the main output, there is a 'Detailed Accuracy By Class' table:

Class	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area
build wind float	0.886	0.104	0.805	0.886	0.844	0.927
build wind non-float	0.803	0.123	0.782	0.803	0.792	0.91
vehic wind float	0.412	0.02	0.636	0.412	0.5	0.915
vehic wind non-float	0	0	0	0	0	?
containers	0.692	0.015	0.75	0.692	0.72	0.953
tableware	0.778	0.01	0.778	0.778	0.778	0.993
headlamps	0.828	0.016	0.889	0.828	0.857	0.953
Weighted Avg.	0.794	0.083	0.79	0.794	0.79	0.928

At the bottom, a 'Confusion Matrix' is shown, which is partially obscured by a red box. The status bar at the bottom indicates 'OK'.

Bagging Improvement: 66.82% → 75.23%

Boosting Improvement: 66.82% → 79.44%

Tutorial Q3(a)

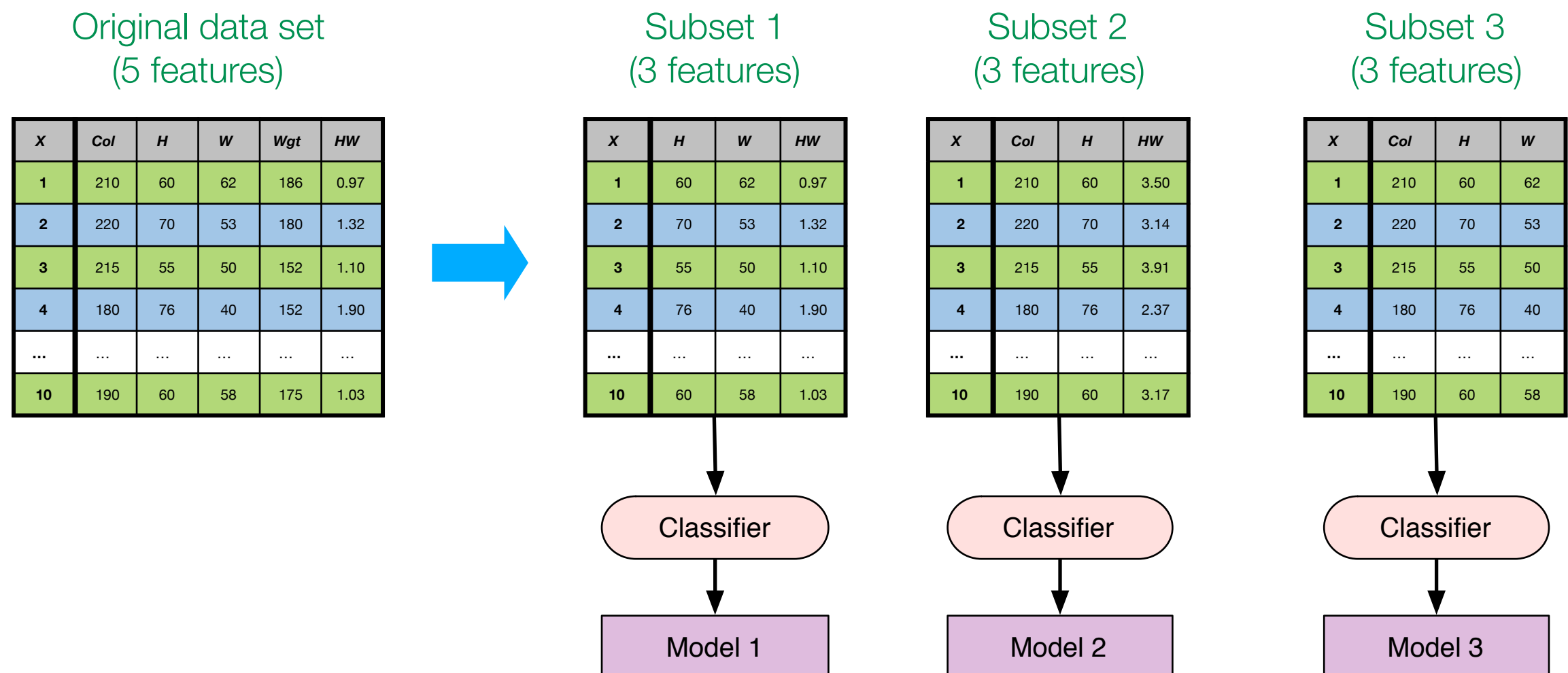
Q. Applying bagging to a “stable” classifier is generally not a good idea. Why is this?

- If a classification algorithm is “unstable”, small changes in the training set lead to larger changes in the model built on the data.
 - ➡ Ensemble is more diverse.
 - e.g. bagging on decision trees, neural networks.
- If a classification algorithm is “stable”, small changes in the training set only lead to small changes in the model.
 - ➡ Little diversity in the ensemble, so little improvement in accuracy.
 - e.g. bagging on k-NN classifiers

Tutorial Q3(b)

Q. Explain how diversity is generated using a *random subspacing* classifier ensemble.

1. A subset of features is randomly selected without replacement.
2. Train a classifier using only selected features to represent training data.
3. Combine outputs of many classifiers trained on different subsets.



Tutorial Q3(c)

Q. In Weka, load the *Glass* data set. Evaluate a k-NN classifier with $k=5$ 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 is 'IBk -K 5 -W 0 -A "weka.core.neighboursearch.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 'Summary' section, which includes the following data:

Summary		
Correctly Classified Instances	145	67.757 %
Incorrectly Classified Instances	69	32.243 %
Kappa statistic	0.5469	
Mean absolute error	0.1085	
Root mean squared error	0.2563	
Relative absolute error	51.243 %	
Root relative squared error	78.9576 %	
Total Number of Instances	214	

Below the summary, the 'Detailed Accuracy By Class' is shown, including TP Rate, FP Rate, Precision, Recall, F-Measure, ROC Area, and Class. The classes are: build wind float, build wind non-float, vehic wind float, vehic wind non-float, containers, tableware, and headlamps. The 'Weighted Avg.' is also provided.

The 'Confusion Matrix' is displayed at the bottom, showing the relationship between the actual and predicted classes. The matrix is a 7x7 grid, with the first row representing the 'build wind float' class and the last row representing the 'headlamps' class. The matrix is as follows:

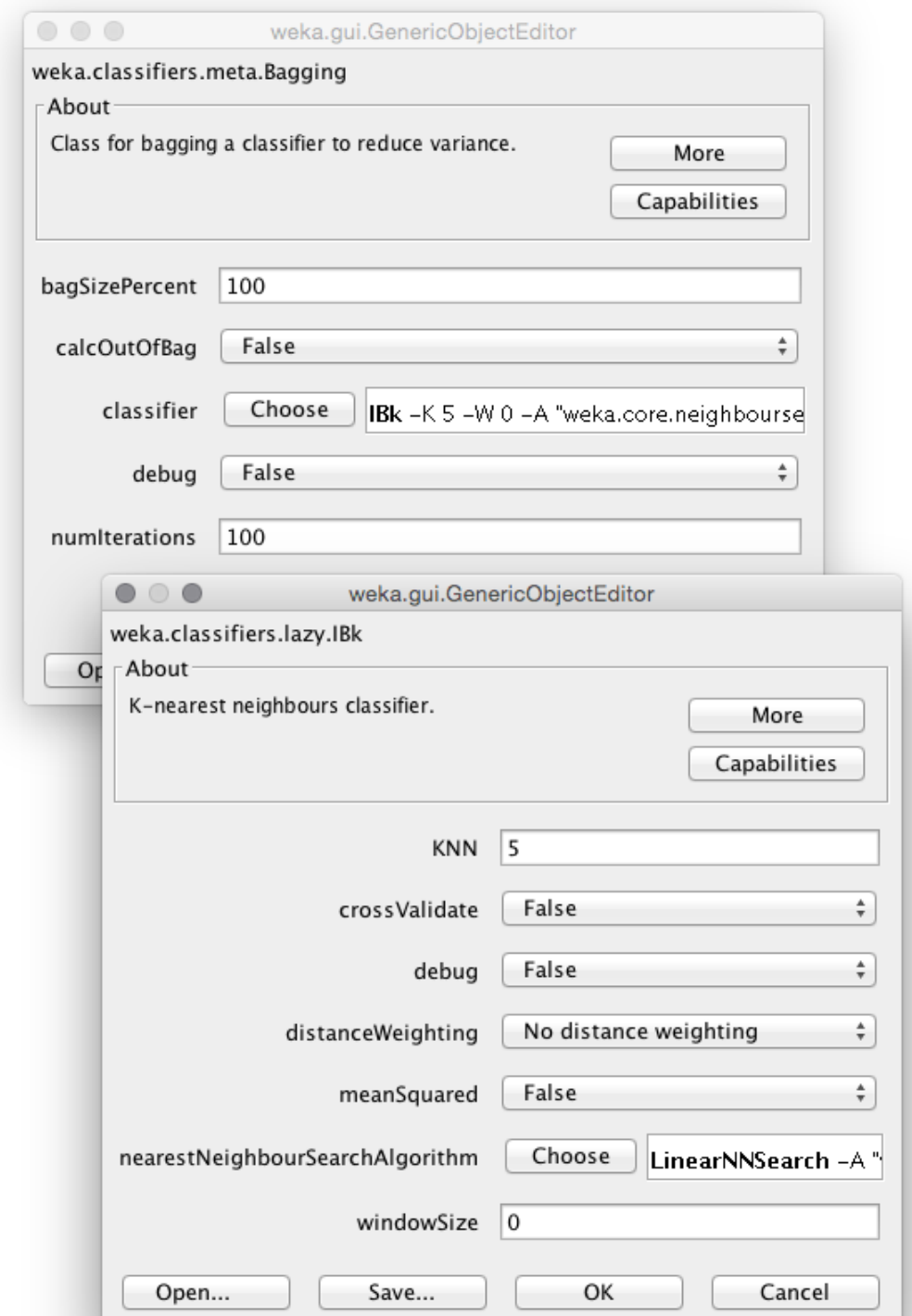
	a	b	c	d	e	f	g	
59	10	1	0	0	0	0	0	a = build wind float
20	52	1	0	3	0	0	0	b = build wind non-float
12	5	0	0	0	0	0	0	c = vehic wind float
0	0	0	0	0	0	0	0	d = vehic wind non-float
0	5	0	0	5	0	3	0	e = containers
0	2	0	0	1	6	0	0	f = tableware
1	2	0	0	1	2	23	0	g = headlamps

The status bar at the bottom shows 'Status OK' and a 'Log' button.

Tutorial Q3(d)

Q. Apply bagging with with a k-NN classifier ($k=5$) 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 5. Close window.
6. Set the *numIterations* to 100.
7. Click *OK* button.
8. Click *Start* button to build the ensemble.



Tutorial Q3(d)

Weka Explorer

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

Classifier: Choose **Bagging** -P 100 -S 1 -I 100 -W weka.classifiers.lazy.IBk -- -K 5 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.

Test options

☐ Use training set

☐ Supplied test set Set...

☒ Cross-validation Folds

☐ Percentage split %

More options...

(Nom) Type

Start Stop

Result list (right-click for options)

15:07:52 - lazy.IBk

15:12:16 - meta.Bagging

Classifier output

Correctly Classified Instances 146 68.2243 %

Incorrectly Classified Instances 68 31.7757 %

Kappa statistic 0.5563

Mean absolute error 0.111

Root mean squared error 0.2478

Relative absolute error 52.3983 %

Root relative squared error 76.3409 %

Total Number of Instances 214

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.814	0.194	0.671	0.814	0.735	0.89	build wind floa
	0.684	0.196	0.658	0.684	0.671	0.874	build wind non-
	0.118	0.005	0.667	0.118	0.2	0.799	vehic wind floa
	0	0	0	0	0	?	vehic wind non-
	0.462	0.035	0.462	0.462	0.462	0.913	containers
	0.667	0.01	0.75	0.667	0.706	0.981	tableware
	0.793	0.016	0.885	0.793	0.836	0.951	headlamps
Weighted Avg.	0.682	0.138	0.686	0.682	0.666	0.891	

=== Confusion Matrix ===

a	b	c	d	e	f	g	<-- classified as
57	12	1	0	0	0	0	a = build wind float
19	52	0	0	5	0	0	b = build wind non-float
8	7	2	0	0	0	0	c = vehic wind float
0	0	0	0	0	0	0	d = vehic wind non-float
0	4	0	0	6	0	3	e = containers
0	2	0	0	1	6	0	f = tableware
1	2	0	0	1	2	23	g = headlamps

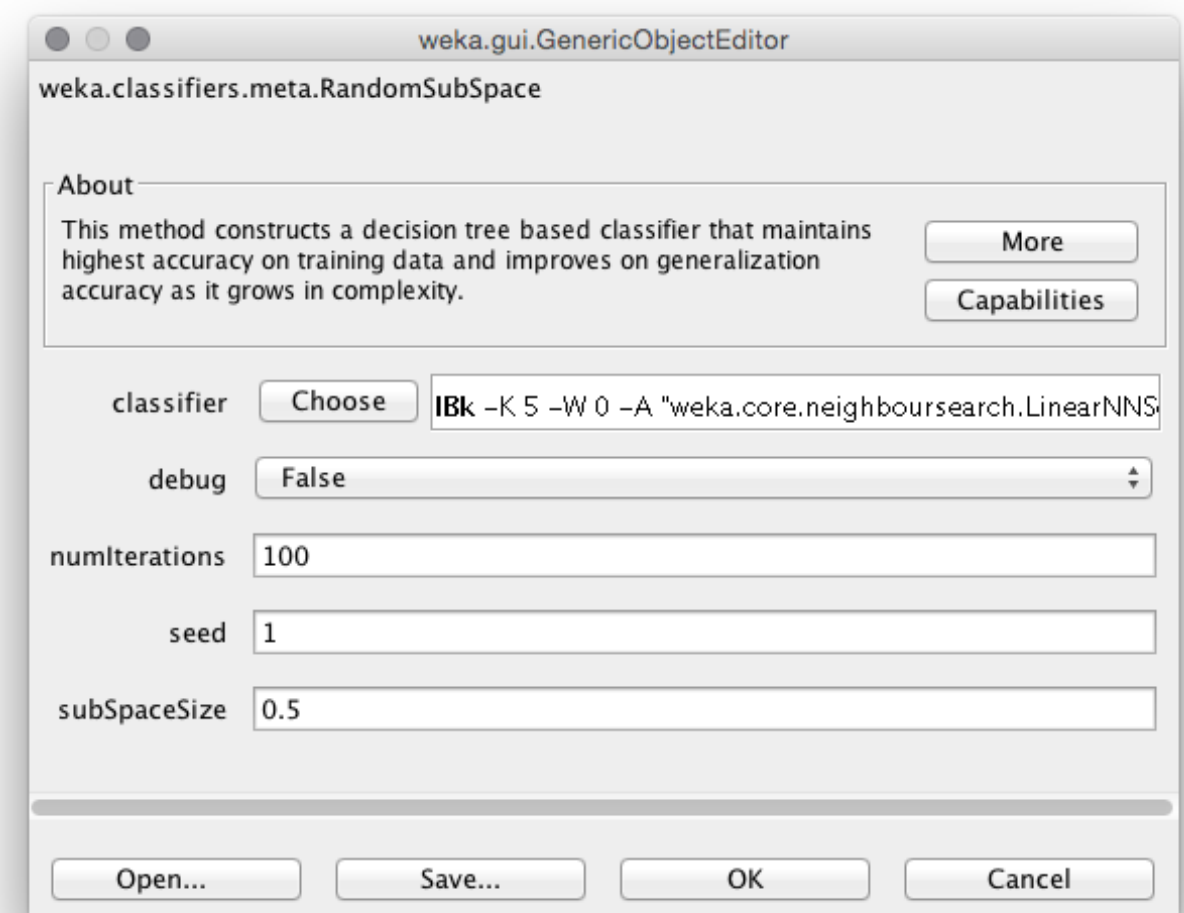
Status OK

Improvement: 67.76% → 68.22%

Tutorial Q3(d)

Q. Now apply *random subspacing* with a k-NN classifier ($k=5$) for an ensemble size of 100. How does it compare to the results from (d)? 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 5. Close window.
6. Set the *numIterations* to 100.
7. Click *OK* button.
8. Click *Start* button to build the ensemble.



Tutorial Q3(d)

- Bagging is ineffective using a stable classifier (k-NN).
- Random subspacing 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: -P 0.5 -S 1 -I 100 -W weka.classifiers.lazy.IBk -- -K 5 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"w\"". The test options are set to 'Cross-validation' with 10 folds and a 66% percentage split. The classifier output shows the following statistics:

Metric	Value	Percentage
Correctly Classified Instances	159	74.2991 %
Incorrectly Classified Instances	55	25.7009 %
Kappa statistic	0.6402	
Mean absolute error	0.1146	
Root mean squared error	0.2276	
Relative absolute error	54.1252 %	
Root relative squared error	70.1188 %	
Total Number of Instances	214	

The 'Detailed Accuracy By Class' table is as follows:

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class	
0.886	0.188	0.697	0.886	0.78	0.926	build wind float	
0.75	0.138	0.75	0.75	0.75	0.911	build wind non-float	
0.176	0.005	0.75	0.176	0.286	0.901	vehic wind float	
0	0	0	0	0	?	vehic wind non-float	
0.538	0.015	0.7	0.538	0.609	0.982	containers	
0.667	0.005	0.857	0.667	0.75	0.99	tableware	
0.828	0.022	0.857	0.828	0.842	0.977	headlamps	
Weighted Avg.	0.743	0.115	0.749	0.743	0.727	0.932	

The 'Confusion Matrix' is shown below:

a	b	c	d	e	f	g	<-- classified as
62	7	1	0	0	0	0	a = build wind float
17	57	0	0	2	0	0	b = build wind non-float
9	5	3	0	0	0	0	c = vehic wind float
0	0	0	0	0	0	0	d = vehic wind non-float
0	3	0	0	7	0	0	e = containers
0	1	0	0	1	6	0	f = tableware
1	3	0	0	0	1	0	g = headlamps

The status bar at the bottom shows 'Status OK'.

Bagging: Improvement: 67.76% → 68.22%

Subspacing: Improvement: 67.76% → 74.23%