Assignment 5: Random Undersampling

Shaun Pritchard

Florida Atlantic University

CAP 6778

October 29, 2021

M. Khoshgoftaar

Assignment 5: Random Undersampling

For this analysis, I will be using Weka to implement random under-sampling techniques using Define data set ratios of sub-sampled data. The evaluation will explore random undersampling on the Naïve Bayes and K-nearest Neighbors(5) learners.

Part I: Preliminary classification

Use the full data set to build models with each of these two (NaiveBayes and KNN(5)) classifiers and compare their results in terms of the false positive rate, false-negative rate, and AUC (area under the ROC curve). Use 10-fold cross-validation and no filter for random under-sampling for a baseline.

Part II: Random Undersampling

Here, I will apply random undersampling before building your classification models. Use the "SpreadSubsample" filter in Weka to create random undersampled datasets. The current dataset has a class imbalance of 77:23. I will undersample the original data to produce datasets with 50:50 and 65:35 class ratios.

Then use the random undersample datasets with the classifiers in Part 1 to create a total of four new models. Again, examine the effects of random undersampling in terms of FPR, FNR, and AUC, paying special attention to how these models compare with those built-in Part 1. Additionally, include the filter settings used for each round of undersampling (include screenshots in the report). Use 10-fold cross-validation for each sampled dataset.

Preliminary Overview

Using the Weka random under UnderSampling filter, I adjusted the distribution spread variable to define 50:50 and 65:35 as the ratio goals of this assignment. Whereas, The maximum class distribution spread. (0 = no maximum spread, 1 = uniform distribution, 10 = allow at most a 10:1 ratio between the classes for the full (77:23) ratio of ACL and nonACL classes. The initial variable of the distribution spread at 50:50:= 1 which created the subsets with 46 instances out of the 95 original instances. The distribution spread was calculated as 1.85 for 65:35. Taking 65 / 35 as a ratio, 64 instances were created. It is also recommended to verify the instance based on 65% of 95 original cases, which is approximately 61.75. A value of 1.7 seemed more appropriate which provided 62 instances, corresponding to a ratio of 65:35. For this reason, both (1.8 and 1.7) distribution values are applied to each learner for more accurate results. The second step I took was to divide these instances to check for the 50:50 ratio, whereas 50% of 95 was evaluated at approximately 47.5 instances. Using a uniform distribution value of 1, 46 instances were produced, which is about 47.5.

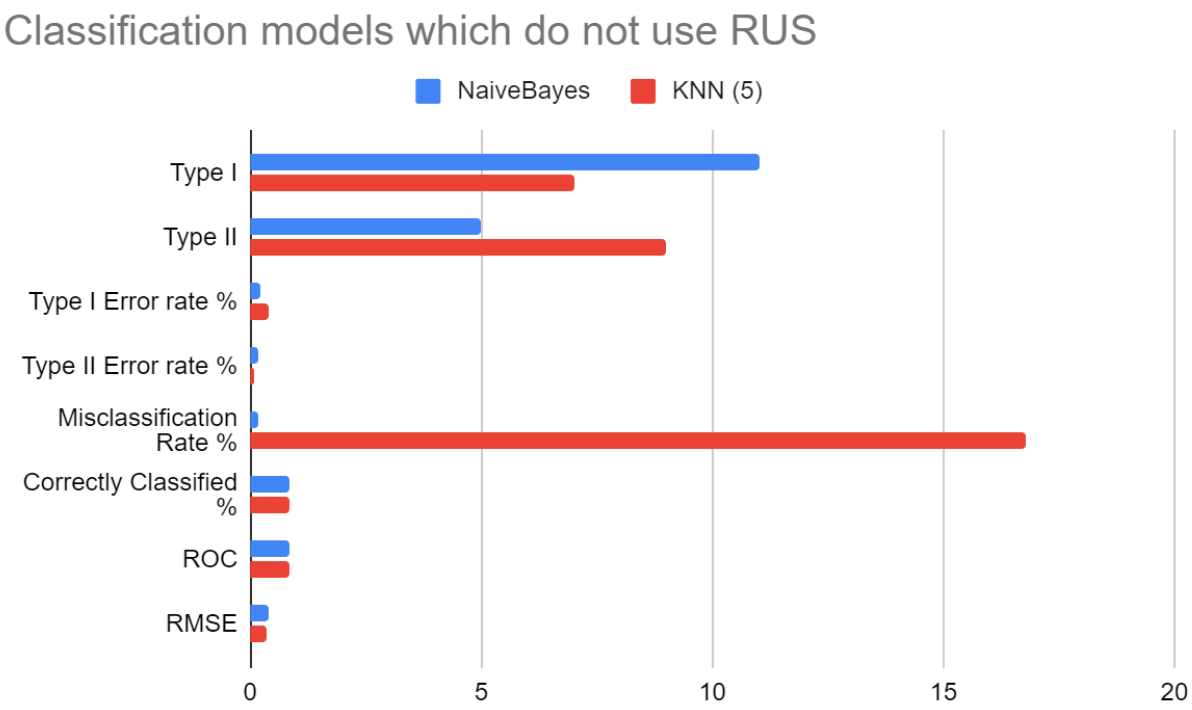
Part I

For this analysis, no Random Undersampling(RUS) was applied to the dataset using the Naive Bayes and (5)K-nearest neighbor classifiers. Table 1-1 and chart 1-1 show that Naive Bayes outperforms the KNN learner in terms of Type II errors, but the KNN learner slightly outperforms the Naive Bayes under the ROC curve and RMSE.

Table 1-1

Evaluation for Assignment 5 - Part 1 -Random Undersampling								
Classifier / Learners	**Type I**	**Type II**	**Type I Error rate %**	**Type II Error rate %**	**Misclassification Rate %**	**Correctly Classified %**	**ROC**	**RMSE**
NaiveBayes	11	5	21.70%	15.30%	16.84%	83%	0.844	0.4115
KNN (5)	7	9	39.10%	9.70%	16.8	83%	0.863	0.3573

Chart 1-1



Part II

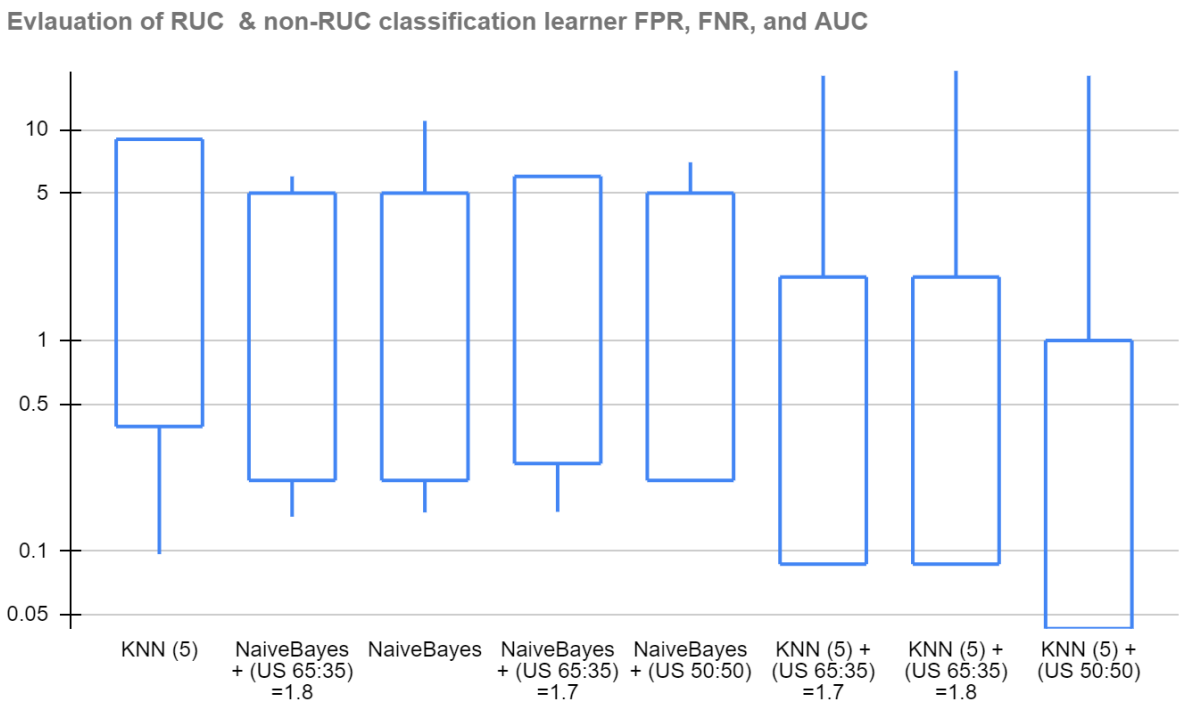
This section evaluates the use of Random Undersampling(RUS) using the spreadSubSmaple filter with distribution weight ratios applied to 50:50 and 65:35 of the 77:23

based 95 ratio dataset. To obtain more precision in values to match the 65:35 ratio two initial values were used for the distribution weight value (1.7 and 1.8). The following table2-1 shows the order of cleaner with and without RUC with variation in distribution ratio according to best performing Area Under the ROC curve.KNN without RUC shows to have the best performance in comparison to all instances and also proves to have the lowest Type II error rate.

Evaluation for Assignment 5 - Random Undersampling								
Classifier / Learners	**Type I**	**Type II**	**Type I Error rate %**	**Type II Error rate %**	**Misclassification Rate %**	**Correctly Classified %**	**ROC**	**RMSE**
KNN (5)	7	9	39.10%	9.70%	16.80%	83%	0.863	0.3573
NaiveBayes	11	5	21.70%	15.30%	16.84%	83%	0.844	0.4115
NaiveBayes + (US 65:35) =1.8	6	5	21.70%	14.60%	17.00%	83.00%	0.831	0.4146
NaiveBayes + (US 65:35) =1.7	6	6	26.10%	15.40%	19.30%	80.64%	0.824	0.4399
KNN (5) + (US 65:35) =1.8	19	2	8.70%	46.30%	33.00%	67.10%	0.807	0.4455
NaiveBayes + (US 50:50)	7	5	21.70%	30.40%	26.09%	73.90%	0.803	0.5108
KNN (5) + (US 65:35) =1.7	18	2	8.70%	46.20%	32.20.%	67.70%	0.775	0.4567
KNN (5) + (US 50:50)	18	1	4.30%	78.30%	41.36%	58.60%	0.749	0.5122

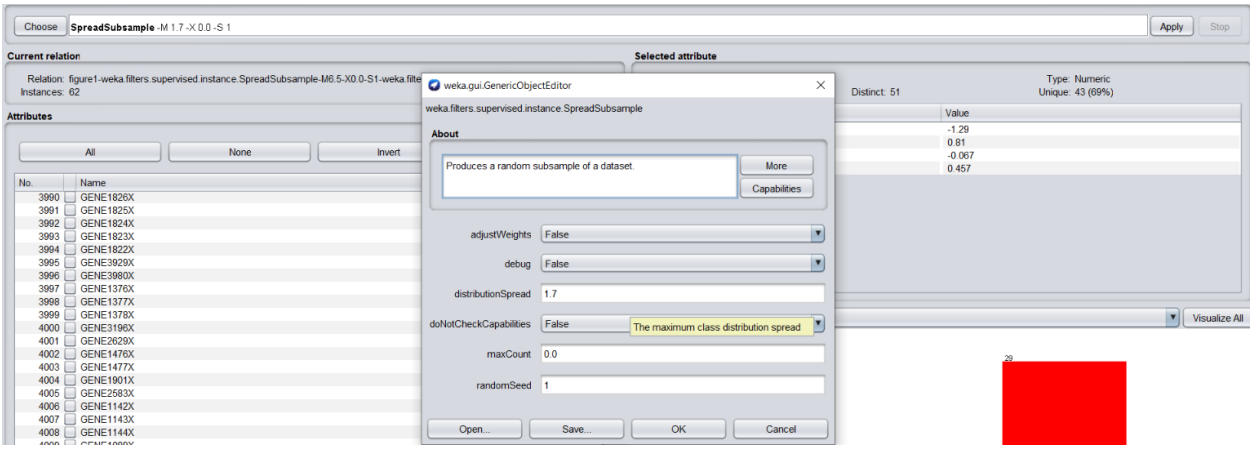
The 50:50 ratio and 65:35 ratio utilizing RUC show that Naive Bayes is the superior solution, with distribution weights set to 1.8, 2nd best ROC value at 0.831, and 14.60% Type II error rate. We see that the 65:35 ratio outperforms the 50:50 ratio distribution for both Naive

Bayes and KNN learners. Also, we see Naive Bayes with 65:35 distribution ratio outperforms KNN with 65:35 distribution ratio as displayed in chart 2-1.

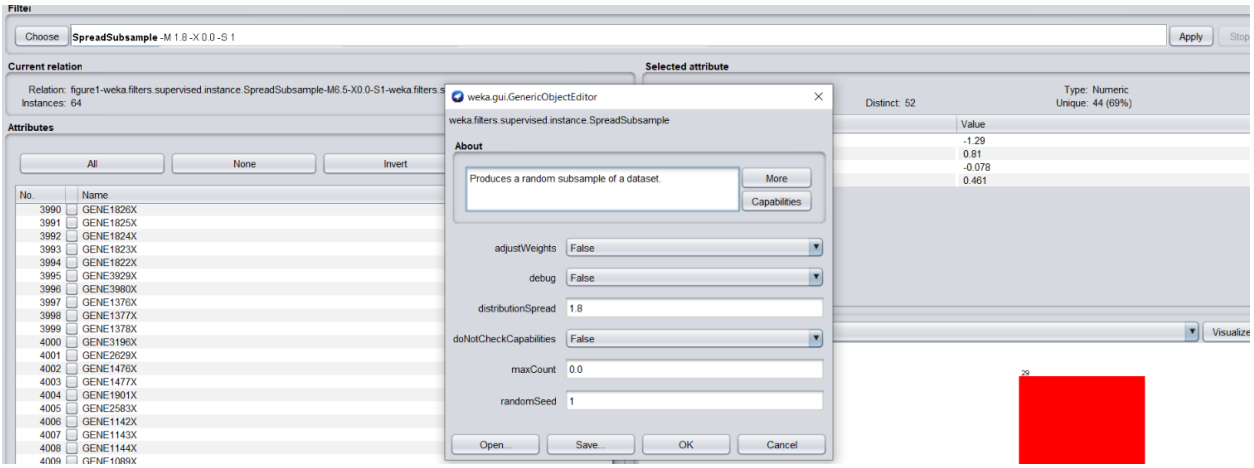


Filter Settings:

Screenshots of Random undersampling(RUC) at first approximate distribution weight to match 65:35 ratio with value of 1.7 and 62 instances.



Screenshots of Random undersampling(RUC) at first approximate distribution weight to match 65:35 ratio with value of 1.8 and 64 instances.



Appendices :

This appendix contains the results of the data evaluation and inference from Weka using the assignment parameters.

1. Naive Bayes with no RUS
2. KNN (5) with no RUS
3. Naive Bayes with RUS 50:50 ratio
4. KNN (5) with RUS 50:50 ratio
5. Naive Bayes with RUS 35:35 ratio value set to 1.8
6. KNN (5)with RUS 35:35 ratio value set to 1.8
7. Naive Bayes with RUS 35:35 ratio value set to 1.7
8. KNN (5)with RUS 35:35 ratio value set to 1.7

I. Naive Bayes with no RUS

Time taken to build model: 0.1 seconds

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

Correctly Classified Instances	79	83.1579 %
Incorrectly Classified Instances	16	16.8421 %
Kappa statistic	0.5785	
Mean absolute error	0.1715	
Root mean squared error	0.4115	
Relative absolute error	46.322 %	
Root relative squared error	95.9589 %	
Total Number of Instances	95	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.783	0.153	0.621	0.783	0.692	0.586	0.851	0.569	ACL
	0.847	0.217	0.924	0.847	0.884	0.586	0.842	0.916	nonACL
Weighted Avg.	0.832	0.202	0.851	0.832	0.838	0.586	0.844	0.832	

=== Confusion Matrix ===

a	b	<-- classified as
18	5	a = ACL
11	61	b = nonACL

II. KNN (5) with no RUS

Time taken to build model: 0 seconds

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

Correctly Classified Instances	79	83.1579 %
Incorrectly Classified Instances	16	16.8421 %
Kappa statistic	0.5271	
Mean absolute error	0.2685	
Root mean squared error	0.3573	
Relative absolute error	72.4907 %	
Root relative squared error	83.3082 %	
Total Number of Instances	95	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.609	0.097	0.667	0.609	0.636	0.528	0.863	0.583	ACL
	0.903	0.391	0.878	0.903	0.890	0.528	0.863	0.947	nonACL
Weighted Avg.	0.832	0.320	0.827	0.832	0.829	0.528	0.863	0.858	

=== Confusion Matrix ===

a	b	<-- classified as
14	9	a = ACL
7	65	b = nonACL

III. Naive Bayes with RUS 50:50 ratio

Time taken to build model: 0.02 seconds

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

Correctly Classified Instances	34	73.913 %
Incorrectly Classified Instances	12	26.087 %
Kappa statistic	0.4783	
Mean absolute error	0.2609	
Root mean squared error	0.5108	
Relative absolute error	52.0161 %	
Root relative squared error	101.8244 %	
Total Number of Instances	46	

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.783	0.304	0.720	0.783	0.750	0.480	0.808	0.735	ACL
0.696	0.217	0.762	0.696	0.727	0.480	0.799	0.752	nonACL
Weighted Avg. 0.739	0.261	0.741	0.739	0.739	0.480	0.803	0.744	

=== Confusion Matrix ===

a	b	<-- classified as
18	5	a = ACL
7	16	b = nonACL

IV. KNN (5) with RUS 50:50 ratio

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 5 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

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

Correctly Classified Instances	27	58.6957 %
Incorrectly Classified Instances	19	41.3043 %
Kappa statistic	0.1739	
Mean absolute error	0.4139	
Root mean squared error	0.5122	
Relative absolute error	82.5248 %	
Root relative squared error	102.1219 %	
Total Number of Instances	46	

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.957	0.783	0.550	0.957	0.698	0.258	0.749	0.690	ACL
0.217	0.043	0.833	0.217	0.345	0.258	0.749	0.745	nonACL
Weighted Avg. 0.587	0.413	0.692	0.587	0.522	0.258	0.749	0.718	

=== Confusion Matrix ===

```
 a  b   <-- classified as
22  1 |  a = ACL
18  5 |  b = nonACL
```

V. Naive Bayes with RUS 35:35 ratio value set to 1.8
Time taken to build model: 0.02 seconds

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

Correctly Classified Instances	53	82.8125 %
Incorrectly Classified Instances	11	17.1875 %
Kappa statistic	0.6303	
Mean absolute error	0.1719	
Root mean squared error	0.4146	
Relative absolute error	37.1792 %	
Root relative squared error	86.2832 %	
Total Number of Instances	64	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.783	0.146	0.750	0.783	0.766	0.631	0.845	0.696	ACL
	0.854	0.217	0.875	0.854	0.864	0.631	0.824	0.851	nonACL
Weighted Avg.	0.828	0.192	0.830	0.828	0.829	0.631	0.831	0.795	

=== Confusion Matrix ===

```
 a  b   <-- classified as
18  5 |  a = ACL
 6 35 |  b = nonACL
```

VI. KNN (5)with RUS 35:35 ratio value set to 1.8
Time taken to build model: 0 seconds

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

Correctly Classified Instances	43	67.1875 %

Incorrectly Classified Instances 21 32.8125 %
Kappa statistic 0.3869
Mean absolute error 0.351
Root mean squared error 0.4455
Relative absolute error 75.9342 %
Root relative squared error 92.7086 %
Total Number of Instances 64

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.913	0.463	0.525	0.913	0.667	0.446	0.807	0.636	ACL
	0.537	0.087	0.917	0.537	0.677	0.446	0.807	0.877	nonACL
Weighted Avg.	0.672	0.222	0.776	0.672	0.673	0.446	0.807	0.790	

=== Confusion Matrix ===

a	b	<-- classified as
21	2	a = ACL
19	22	b = nonACL

VII. Naive Bayes with RUS 35:35 ratio value set to 1.7
Time taken to build model: 0.02 seconds

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

Correctly Classified Instances 50 80.6452 %
Incorrectly Classified Instances 12 19.3548 %
Kappa statistic 0.5853
Mean absolute error 0.1935
Root mean squared error 0.4399
Relative absolute error 41.3121 %
Root relative squared error 90.9232 %
Total Number of Instances 62

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.739	0.154	0.739	0.739	0.739	0.585	0.852	0.701	ACL
	0.846	0.261	0.846	0.846	0.846	0.585	0.808	0.827	nonACL
Weighted Avg.	0.806	0.221	0.806	0.806	0.806	0.585	0.824	0.780	

=== Confusion Matrix ===

```
 a  b   <-- classified as
17  6 |  a = ACL
 6 33 |  b = nonACL
```

VIII. KNN (5)with RUS 35:35 ratio value set to 1.7

Time taken to build model: 0 seconds

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

Correctly Classified Instances	42	67.7419 %
Incorrectly Classified Instances	20	32.2581 %
Kappa statistic	0.3951	
Mean absolute error	0.3559	
Root mean squared error	0.4567	
Relative absolute error	75.9599 %	
Root relative squared error	94.39 %	
Total Number of Instances	62	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.913	0.462	0.538	0.913	0.677	0.452	0.775	0.620	ACL
	0.538	0.087	0.913	0.538	0.677	0.452	0.775	0.864	nonACL
Weighted Avg.	0.677	0.226	0.774	0.677	0.677	0.452	0.775	0.773	

=== Confusion Matrix ===

```
 a  b   <-- classified as
21  2 |  a = ACL
18 21 |  b = nonACL
```