UNIVERSIDADE DO MINHO

Mini-teste 2

Mestrado Integrado em Engenharia Informática

Mineração de Dados
(1.º Semestre / 2020-2021)

A84003 Beatriz Rocha

Braga,
Dezembro 2020

1. Apresente uma análise ao desempenho dos algoritmos *NaiveBayes*, *BayesNet*, J48 e *AdaBoost* sobre *NB*, *Bagging* sobre J48 no *dataset ionosphere*. Considere as seguintes alíneas:

1.1. Usando a decomposição *bias-variance* estude o evoluir da complexidade da árvore J48 (por aplicação de diferentes níveis de *pruning*) e o efeito no erro e suas componentes.

Intervalo de confiança de 0.1

- Tamanho da árvore

```
Number of Leaves:  10
Size of the tree:  19
```

- Erro e suas componentes

```
Bias-Variance Decomposition

Classifier   : weka.classifiers.trees.J48-C 0.1 -M 2
Data File    : ionosphere.arff
Class Index  : last
Training Pool: 100
Iterations   : 50
Seed         : 1
Error        : 0.1555
Sigma^2      : 0
Bias^2       : 0.0754
Variance     : 0.0785
```

Intervalo de confiança de 0.2

- Tamanho da árvore

```
Number of Leaves:  16
Size of the tree:  31
```

- Erro e suas componentes

```
Bias-Variance Decomposition

Classifier   : weka.classifiers.trees.J48-C 0.2 -M 2
Data File    : ionosphere.arff
Class Index  : last
Training Pool: 100
Iterations   : 50
Seed         : 1
Error        : 0.1568
Sigma^2      : 0
Bias^2       : 0.075
Variance     : 0.0802
```

Intervalo de confiança de 0.3

- ▷ Tamanho da árvore

```
Number of Leaves:  18
Size of the tree:  35
```

- ▷ Erro e suas componentes

```
Bias-Variance Decomposition

Classifier   : weka.classifiers.trees.J48-C 0.3 -M 2
Data File    : ionosphere.arff
Class Index  : last
Training Pool: 100
Iterations   : 50
Seed         : 1
Error        : 0.1593
Sigma^2      : 0
Bias^2       : 0.0745
Variance     : 0.0831
```

Intervalo de confiança de 0.4

- ▷ Tamanho da árvore

```
Number of Leaves:  18
Size of the tree:  35
```

- ▷ Erro e suas componentes

```
Bias-Variance Decomposition

Classifier   : weka.classifiers.trees.J48-C 0.4 -M 2
Data File    : ionosphere.arff
Class Index  : last
Training Pool: 100
Iterations   : 50
Seed         : 1
Error        : 0.1593
Sigma^2      : 0
Bias^2       : 0.0745
Variance     : 0.0831
```

Intervalo de confiança de 0.5

- ▷ Tamanho da árvore

```
Number of Leaves:  18
Size of the tree:  35
```

▷ Erro e suas componentes

```
Bias-Variance Decomposition

Classifier   : weka.classifiers.trees.J48-C 0.5 -M 2
Data File    : ionosphere.arff
Class Index  : last
Training Pool: 100
Iterations   : 50
Seed         : 1
Error        : 0.1593
Sigma^2      : 0
Bias^2       : 0.0745
Variance     : 0.0831
```

Confidence Factor	***Size of the tree***	***Error***	***Bias*** ^{2}	***Variance***
0.1	19	0.1555	0.0754	0.0785
0.2	31	0.1568	0.075	0.0802
0.3	35	0.1593	0.0745	0.0831
0.4	35	0.1593	0.0745	0.0831
0.5	35	0.1593	0.0745	0.0831

Tabela 1: Resultados para os vários intervalos de confiança

O erro de um modelo pode ser decomposto em três componentes:

- Viés - mede a capacidade da previsão média de um algoritmo de aprendizagem acertar no objetivo;
- Variância - mede a variabilidade da previsão de um algoritmo para diferentes conjuntos de treino de um dado tamanho;
- Erro irredutível - é o limite mínimo de erro esperado para qualquer algoritmo de aprendizagem.

Na Tabela 1, podemos observar o tamanho da árvore, o erro, o viés^2 e a variância para vários intervalos de confiança. De frisar que, para obter estes três últimos valores, recorri ao terminal, onde executei o comando

```
java -cp "~/Downloads/weka-3-8-4/weka.jar" weka.classifiers.BVDecompose
-t ionosphere.arff  -W weka.classifiers.trees.J48 -- -C X -M 2
```

onde X corresponde ao intervalo de confiança aplicado.

A partir daqui, é possível estudar a evolução da complexidade da árvore J48 e o efeito no erro e suas componentes. À medida que o intervalo de confiança aumenta, o tamanho da árvore também aumenta, uma vez que o *pruning* vai, consecutivamente, diminuindo. Podemos, ainda, observar que o erro e a variância também aumentam, enquanto o viés^2 diminui. Isto poderá significar que o modelo está a entrar em *overfitting*, visto que *overfitting* ocorre se um modelo apresentar viés baixo e variância alta.

1.2. Apresente os resultados do erro por validação cruzada dos cinco modelos referidos. Elabore uma justificação para cada resultado obtido.

NaiveBayes

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

Correctly Classified Instances         290               82.6211 %
Incorrectly Classified Instances        61               17.3789 %
Kappa statistic                          0.6394
Mean absolute error                      0.1736
Root mean squared error                  0.3935
Relative absolute error                 37.7001 %
Root relative squared error             82.0203 %
Total Number of Instances              351
```

BayesNet

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

Correctly Classified Instances         314               89.4587 %
Incorrectly Classified Instances        37               10.5413 %
Kappa statistic                          0.7681
Mean absolute error                      0.1069
Root mean squared error                  0.3179
Relative absolute error                 23.2213 %
Root relative squared error             66.2607 %
Total Number of Instances              351
```

J48

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

Correctly Classified Instances         321               91.453  %
Incorrectly Classified Instances        30                8.547  %
Kappa statistic                          0.8096
Mean absolute error                      0.0938
Root mean squared error                  0.2901
Relative absolute error                 20.36   %
Root relative squared error             60.4599 %
Total Number of Instances              351
```

Adaboost sobre *NB*

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

```
Correctly Classified Instances         323               92.0228 %
Incorrectly Classified Instances        28                7.9772 %
Kappa statistic                          0.8273
Mean absolute error                      0.1043
Root mean squared error                  0.2611
Relative absolute error                 22.6412 %
Root relative squared error             54.4218 %
Total Number of Instances              351
```

Bagging sobre J48

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

Correctly Classified Instances         326               92.8775 %
Incorrectly Classified Instances        25                7.1225 %
Kappa statistic                          0.8422
Mean absolute error                      0.1109
Root mean squared error                  0.2431
Relative absolute error                 24.0792 %
Root relative squared error             50.6629 %
Total Number of Instances              351
```

	Mean absolute error	***Root mean squared error***	***Relative absolute error***	***Root relative squared error***
NaiveBayes	0.1736	0.3935	37.7001%	82.0203%
BayesNet	0.1069	0.3179	23.2213%	66.2607%
J48	0.0938	0.2901	20.36%	60.4599%
Adaboost* sobre *NB	0.1043	0.2611	22.6412%	54.4218%
***Bagging* sobre J48**	0.1109	0.2431	24.0792%	50.6629%

Tabela 2: Resultados do erro por validação cruzada dos cinco modelos

Antes de mais, convém explicar as várias medidas de erro apresentadas pelo *Weka*:

- *Mean absolute error* – quantidade usada para medir quão próximas as previsões estão dos resultados finais;
- *Root mean squared error* - medida das diferenças entre os valores (amostrais e populacionais) previstos por um modelo ou estimador e os valores realmente observados;
- *Relative absolute error* - quanto o resultado se desvia do valor real;
- *Root relative squared error* - medida em percentagem em relação ao valor real.

Adicionalmente, é importante explicar que *Bagging* tem tendência a baixar a variância do erro dos classificadores originais, enquanto *Boosting* reduz viés e variância (e, consequentemente, o erro).

Pela observação da Tabela 2, podemos comprovar esta premissa, visto que, quando se aplica *Bagging* a J48, obtêm-se menores valores em alguns tipos de erro e, quando se aplica *Adaboost* a *NB*, obtêm-se menores valores de erro para todos os parâmetros.

1.3. Para a classe b e considerando os cinco modelos referidos:

- **Qual seria o melhor modelo a recuperar os casos desta classe?**
- **E o modelo com melhor qualidade de previsão nesta classe?**

Justifique as suas respostas.

NaiveBayes

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.865	0.196	0.712	0.865	0.781	0.648	0.935	0.917	b
0.804	0.135	0.914	0.804	0.856	0.648	0.935	0.958	g

BayesNet

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.825	0.067	0.874	0.825	0.849	0.769	0.947	0.918	b
0.933	0.175	0.905	0.933	0.919	0.769	0.949	0.966	g

J48

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.825	0.036	0.929	0.825	0.874	0.813	0.892	0.855	b
0.964	0.175	0.908	0.964	0.935	0.813	0.892	0.894	g

Adaboost sobre *NB*

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.897	0.067	0.883	0.897	0.890	0.827	0.943	0.931	b
0.933	0.103	0.942	0.933	0.938	0.827	0.943	0.945	g

Bagging sobre J48

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.857	0.031	0.939	0.857	0.896	0.844	0.962	0.957	b
0.969	0.143	0.924	0.969	0.946	0.844	0.962	0.960	g

	Precision	***Recall***
NaiveBayes	0.712	0.865
BayesNet	0.874	0.825
J48	0.929	0.825
Adaboost* sobre *NB	0.883	0.897
***Bagging* sobre J48**	0.939	0.857

Tabela 3: Valores de *Precision* e *Recall* dos cinco modelos

Precision é a proporção de identificações positivas que estava realmente correta. Assim sendo, o melhor modelo a recuperar os casos da classe b seria *Bagging* sobre J48, pois é este que apresenta o maior valor para este parâmetro (0.939).

Recall é a proporção de positivos reais identificada corretamente. Posto isto, o modelo com melhor qualidade de previsão seria *Adaboost* sobre *NB*, pois é aquele que apresenta o maior valor para esta métrica (0.897).

2. Que tipo de benefícios esperaria da aplicação de *Bagging* sobre *Naive Bayes* num *dataset* específico, sabendo que o resultado do modelo individual *Naive Bayes* nesse *dataset* é erro = 0.005. Justifique.

Bootstrap aggregating, também conhecida por *Bagging*, é uma técnica de *Machine Learning* desenvolvida para melhorar a estabilidade e precisão de algoritmos de *Machine Learning* usados em classificação estatística e regressão. Este algoritmo também reduz a variância do erro do classificador e ajuda a evitar *overfitting*.

Embora esta técnica seja normalmente aplicada com sucesso a algoritmos instáveis como aqueles que envolvem árvores de decisão (por exemplo, J48), esta também pode ser aplicada a algoritmos mais estáveis (por exemplo, *Naive Bayes*). Contudo, no último caso, esta técnica pode chegar a aumentar a variância, visto que *Naive Bayes* já apresenta uma variância bastante baixa.

Uma vez que a taxa de erro apresentada pelo algoritmo *Naive Bayes* sem recorrer à técnica de *Bagging* é quase nula, pode-se concluir que a aplicação desta técnica, nesta situação, poderá trazer desvantagens e aumentar a taxa de erro nas instâncias corretamente classificadas.

3. Para um determinado conjunto de teste com 10 exemplos, a seguinte tabela representa as previsões obtidas com os modelos M1 e M2 para a classe A. Os modelos são classificadores binários por definição de *threshold*. O valor de *threshold* usado é 0.9. A coluna *Class* indica a classe efetiva de cada caso de teste.

M1			M2		
#	Score	Class	#	Score	Class
1	0.996	A	1	0.999	A
2	0.995	A	2	0.998	A
3	0.977	A	3	0.997	B
4	0.951	A	4	0.979	A
5	0.915	B	5	0.931	A
6	0.895	B	6	0.920	A
7	0.881	B	7	0.915	B
8	0.795	B	8	0.812	B
9	0.786	A	9	0.775	B
10	0.675	B	10	0.771	B

a) Apresente o valor de rácio de erro para os dois modelos.

Sabendo que o valor de *threshold* é 0.9, temos de ter em conta o que se apresenta na figura seguinte:

M1			M2		
#	Score	Class	#	Score	Class
1	0.996	A	1	0.999	A
2	0.995	A	2	0.998	A
3	0.977	A	3	0.997	B
4	0.951	A	4	0.979	A
5	0.915	B	5	0.931	A
6	0.895	B	6	0.920	A
7	0.881	B	7	0.915	B
8	0.795	B	8	0.812	B
9	0.786	A	9	0.775	B
10	0.675	B	10	0.771	B

Threshold = 0.9

Threshold = 0.9

Figura 1: Localização do *threshold* na tabela

Para além disso, convém recordar e explicar o formato de uma matriz de confusão:

		Classe real	
		P	***N***
Classe prevista	***P***	*TP*	*FP*
	N	*FN*	*TN*

Tabela 4: Formato de uma matriz de confusão

onde P = *Positive*, N = *Negative*, TP = *True Positive*, FP = *False Positive*, FN = *False Negative* e TN = *True Negative*.

Posto isto, na Figura 1 podemos ver que, no modelo M1, existem:

- 4 instâncias da classe A corretamente classificadas como A;
- 4 instâncias da classe B corretamente classificadas como B;
- 1 instância da classe A incorretamente classificada como B;
- 1 instância da classe B incorretamente classificada como A.

	A	**B**
A	4	1
B	1	4

Tabela 5: Matriz de confusão do modelo M1

Ainda na Figura 1, podemos ver que, no modelo M2, existem:

- 5 instâncias da classe A corretamente classificadas como A;
- 3 instâncias da classe B corretamente classificadas como B;
- 0 instâncias da classe A incorretamente classificadas como B;
- 2 instâncias da classe B incorretamente classificadas como A.

	A	**B**
A	5	2
B	0	3

Tabela 6: Matriz de confusão do modelo M2

Por fim, só falta calcular o valor de rácio de erro para os dois modelos, que é dado pela seguinte fórmula:

$$\textit{Rácio de erro} = \frac{(FP + FN)}{(TP + TN + FP + FN)} \tag{1}$$

$$Rácio\ de\ erro\ (M1) = \frac{(1+1)}{(4+4+1+1)} = \frac{1}{5} \tag{2}$$

$$Rácio\ de\ erro\ (M2) = \frac{(2+0)}{(5+3+2+0)} = \frac{1}{5} \tag{3}$$

b) Qual devia ser o modelo escolhido para esta classe? Justifique.

O valor de rácio de erro nem sempre é a melhor medida para comparar modelos. Assim sendo, podemos recorrer à AUC (*Area Under the Curve*) que oferece uma melhor forma de discriminação entre modelos e também é insensível à distribuição de classes. Podemos calcular a AUC usando *ranks* dos *scores* derivados por cada classificador, mas, primeiro, temos de ordenar ascendentemente os mesmos, tal como podemos ver na Tabela 7.

M1			**M2**		
#	***Score***	***Class***	#	***Score***	***Class***
1	0.675	B	1	0.771	B
2	0.786	A	2	0.775	B
3	0.795	B	3	0.812	B
4	0.881	B	4	0.915	B
5	0.895	B	5	0.920	A
6	0.915	B	6	0.931	A
7	0.951	A	7	0.979	A
8	0.977	A	8	0.997	B
9	0.995	A	9	0.998	A
10	0.996	A	10	0.999	A

Tabela 7: *Scores* derivados ordenados ascendentemente

A fórmula para calcular a AUC apresenta-se de seguida:

$$AUC = \frac{S_0 - n_0(n_0+1)/2}{n_0 n_1} \tag{4}$$

onde S_0 = posições onde *Class* é A, n_0 = número de intâncias onde *Class* é A e n_1 = número de instâncias onde *Class* é B.

Posto isto, só nos resta calcular AUC(M1) e AUC(M2):

$$AUC(M1) = \frac{(2+7+8+9+10) - 5*6/2}{5*5} = \frac{21}{25} \tag{5}$$

$$AUC(M2) = \frac{(5+6+7+9+10) - 5*6/2}{5*5} = \frac{22}{25} \tag{6}$$

Visto que, quanto maior a AUC, melhor é o modelo, concluímos que o modelo M2 deveria ser o modelo escolhido para esta classe.

c) Sabendo que *precision* = *TP*/(*TP*+*FP*) e *FPR* = *FP*/(*FP*+*TN*) calcule *precision*(M1) e *FPR*(M2) para a classe A e interprete os valores obtidos (nota: *FPR* = *false positive rate*).

Calculando *precision*(M1) e *FPR*(M2), temos:

$$precision(M1) = \frac{4}{4+1} = \frac{4}{5} \tag{7}$$

$$FPR(M2) = \frac{2}{2+3} = \frac{2}{5} \tag{8}$$

A precisão determina a fração de registos que efetivamente são positivos no grupo que o classificador declarou como uma classe positiva. Assim sendo e tendo em conta a equação (7), podemos verificar que, no modelo M1, $\frac{4}{5}$ dos registos que o classificador declarou como uma classe positiva são efetivamente positivos. Visto que este valor é relativamente alto e avaliando apenas por esta métrica, pode concluir-se que M1 é um bom modelo (quanto maior a precisão, melhor é o modelo).

Já a taxa de falsos positivos trata-se da probabilidade de rejeitar incorretamente a hipótese nula para um teste particular. Posto isto e tendo em conta a equação (8), podemos observar que, no modelo M2, existe uma probabilidade de $\frac{2}{5}$ de rejeitar incorretamente a hipótese nula. Uma vez que este valor é relativamente baixo e avaliando apenas por esta métrica, pode concluir-se que M2 é um bom modelo (quanto menor a taxa de falsos positivos, melhor é o modelo).