Washington State University

School of Electrical Engineering and Computer Science Fall 2019

CptS 440/540 Artificial Intelligence

**Homework 10**

Due: November 14, 2019 (11:59pm)

**General Instructions**: Put your answers to the following problems into a PDF document and submit as an attachment under Content  Homework 10 for the course CptS 440 Pullman (all sections of CptS 440 and 540 are merged under the CptS 440 Pullman section) on the Blackboard Learn system by the above deadline. Note that you may submit multiple times, but we will only grade the most recent entry submitted before the above deadline.

For this homework we will be applying two different learning methods to the following 11 examples. Each example describes the Weather  {clear, cloudy, rain}, whether or not your AI homework is done (AIDone {yes, no}), and whether or not you have a costume (Costume 

{yes, no}). The class HaveFun  {yes, no} indicates whether or not you will have fun at a party under these conditions.

|  |  |  |  |
| --- | --- | --- | --- |
| **Weather** | **AIDone** | **Costume** | **HaveFun** |
| clear | yes | yes | yes |
| clear | yes | no | yes |
| clear | no | yes | yes |
| clear | no | no | no |
| cloudy | yes | yes | yes |
| cloudy | no | yes | no |
| cloudy | no | no | no |
| rain | yes | yes | yes |
| rain | yes | no | yes |
| rain | no | yes | no |
| rain | no | no | no |

1. *Naïve Bayes*. Suppose we want to classify the new instance <Weather=cloudy, AIDone=yes, Costume=no> using the naïve Bayes learning method. Compute the following. Show your work. Note: only use the “add 1 / |values|” technique if the original P(attribute | class) = 0.
   1. Compute the prior probabilities P(HaveFun=yes) and P(HaveFun=no).

P(HaveFun=yes) = 6/11 = 0.54

P(HaveFun=no) = 5/11 = 0.45

* 1. Compute P(Weather | HaveFun) for all values of Weather  {clear, cloudy, rain} and HaveFun  {yes, no}.

P(Weather = clear | HaveFun = yes) = 3/6 = 1/2 = 0.5

P(Weather = clear | HaveFun = no) = 1/5 = 0.2

P(Weather = cloudy | HaveFun = yes) = 1/6 = 0.16

P(Weather = cloudy | HaveFun = no) = 2/5 = 0.4

P(Weather = rain | HaveFun = yes) = 2/6 = 1/3 = 0.33

P(Weather = rain | HaveFun = no) = 2/5 = 0.4

* 1. Compute P(AIDone | HaveFun) for all values of AIDone  {yes, no} and HaveFun 

{yes, no}.

P(AIDone = yes | HaveFun = yes) = 5/6 = 0.83

P(AIDone = no | HaveFun = yes) = 1/6 = 0.16

P(AIDone = yes | HaveFun = no) = 0/5 = 0

P(AIDone = no | HaveFun = no) = 5/5 = 1

* 1. Compute P(Costume | HaveFun) for all values of Costume  {yes, no} and HaveFun

 {yes, no}.

P(Costume = yes | HaveFun = yes) = 4/6 = 0.66

P(Costume = no | HaveFun = yes) = 2/6 = 0.33

P(Costume = yes | HaveFun = no) = 2/5 = 0.4

P(Costume = no | HaveFun = no) = 3/5 = 0.6

* 1. Compute P(HaveFun=yes | Weather=cloudy, AIDone=yes, Costume=no) and P(HaveFun=no | Weather=cloudy, AIDone=yes, Costume=no).

P(HaveFun=yes | Weather=cloudy, AIDone=yes, Costume=no) = 𝛼 P(Weather = cloudy | HaveFun= yes) P(AIDone=yes | HaveFun= yes) P(Costume=no | HaveFun= yes)

where 𝛼 = P( Weather=cloudy, AIDone=yes, Costume=no )

= 𝛼 ( 1/6 ) ( 5/6 ) ( 1/3 ) = 𝛼 (0.0462)

Similarly, P(HaveFun=no | Weather=cloudy, AIDone=yes, Costume=no) = 𝛼 P(Weather = cloudy | HaveFun= no) P(AIDone=yes | HaveFun= no) P(Costume=no | HaveFun= no)

= 𝛼 ( 2/5 ) ( 0/5 ) ( 3/5 )

but 0/5 becomes, 0+1/5+2 = 1/7

= 𝛼 ( 2/5 ) ( 1/7 ) ( 3/5 )

= 𝛼 (0.0342)

Therefore, 𝛼 = 1/ (0.0462 + 0.0342) = 12.437

P(HaveFun | Weather=cloudy, AIDone=yes, Costume=no) = < 0.57, 0.43>

* 1. Which class would Naïve Bayes choose for the new instance?

P(HaveFun | Weather=cloudy, AIDone=yes, Costume=no) = < 0.57, 0.43>

On taking argmax we will get the prediction ‘Yes’

1. *Perceptron*. Train a perceptron on the 11 examples in the above table and then use the perceptron to classify the new instance. Specifically,
   1. First, translate the examples (including the HaveFun class value) according to the mapping: clear  1, cloudy  2, rain  3, no 0, yes  1. Show a new table of examples using this mapping.

The new table:

|  |  |  |  |
| --- | --- | --- | --- |
| Weather | AIDone | Costume | HaveFun |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 2 | 1 | 1 | 1 |
| 2 | 0 | 1 | 0 |
| 2 | 0 | 0 | 0 |
| 3 | 1 | 1 | 1 |
| 3 | 1 | 0 | 1 |
| 3 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 |

* 1. Train the perceptron below by updating the weights according to the perceptron learning rule (slide 35 of Learning lecture). Assume the initial weights are all equal to 1.0, and  = 0.5. Consider each example in the order presented in the table above and show the weight updates for each incorrectly-classified example. Continue until the perceptron correctly classifies all the training examples, and show the final perceptron weights. *Hint*: The perceptron should correctly classify all 11 examples on the 3rd pass through the examples.

The initial weights are all equal to 1.0, and 𝜂 = 0.5

1. (1)(1) + (1)(1) + 1(1) + (1)(1) = 3 - yes

2. (1)(1) + (1)(1) + 0 (1) + 1(1) = 3 - yes

3. 1(1) + 0(1) + 1(1) + 1(1) = 3 - yes

4. 1(1) + 0(1) + 0(1) +1(1) = 2 – yes - Wrong

weight update:

w0 = w0 + (0.5)(0-1)(1) = 1 - 0.5 = 0.5

w1 = w1 + (0.5)(0-1)(1) = 1 - 0.5 = 0.5

w2 = w2 + (0.5)(0-1)(0) = 1 – 0 = 1

w3 = w3 + (0.5)(0-1)(0) = 1 - 0 = 1

4. 1(0.5) + 0(1) + 0(1) +1(0.5) = 1 – yes – Wrong

weight update:

w0 = w0 + (0.5)(0-1)(1) = 0.5 - 0.5 = 0

w1 = w1 + (0.5)(0-1)(1) = 0.5 - 0.5 = 0

w2 = w2 + (0.5)(0-1)(0) = 1 – 0 = 1

w3 = w3 + (0.5)(0-1)(0) = 1 - 0 = 1

4. 1(0) + 0(1) + 0(1) +1(0) = 0 -yes - Wrong

weight update:

w0 = w0 + (0.5)(0-1)(1) = 0 - 0.5 = -0.5

w1 = w1 + (0.5)(0-1)(1) = 0 - 0.5 = -0.5

w2 = w2 + (0.5)(0-1)(0) = 1 – 0 = 1

w3 = w3 + (0.5)(0-1)(0) = 1 - 0 = 1

4. 1(-0.5) + 0(1) + 0(1) +1(-0.5) = -1 - no

5. 2(-0.5) + 1(1) + 1(1) + 1 ( -0.5) = 0.5 - yes

6. 2(-0.5) + 0(1) + 1(1) + 1(-0.5) = -2.5 – no

7. 2(-0.5) + 0(1) + 0(1) + 1(-0.5) = -1.5 – no

8. 3(-0.5) + 1(1) + 1(1) + 1(-0.5) = 0 – yes

9. 3(-0.5) + 1(1) + 0(1) + 1(-0.5) = -1 – no – Wrong

weight update:

w0 = w0 + (0.5)(1-0)(1) = -0.5 + 0.5 = 0

w1 = w1 + (0.5)(1-0)(3) = -0.5 + 1.5 = 1

w2 = w2 + (0.5)(1-0)(1) = 1 + 0.5 = 1.5

w3 = w3 + (0.5)(1-0)(0) = 1 + 0 = 1

9. 3(1) + 1(1.5) + 0(1) + 1(0) = 4.5- yes

10. 3(1) + 0(1.5) + 1(1) +1(0) = 4 -yes - Wrong

weight update:

w0 = w0 + (0.5)(0-1)(1) = 0 - 0.5 = 0

w1 = w1 + (0.5)(0-1)(3) = 1 - 1.5 = -0.5

w2 = w2 + (0.5)(0-1)(0) = 1.5 – 0 = 1.5

w3 = w3 + (0.5)(0-1)(1) = 1 – 0.5 = 0.5

10. 3(-0.5) + 0(1.5) + 1(0.5) +1(0) = -1 - no

11. 3(-0.5) + 0(1.5) + 0(0.5) + 1(0) = -1.5 – no

Final perceptron weights are:

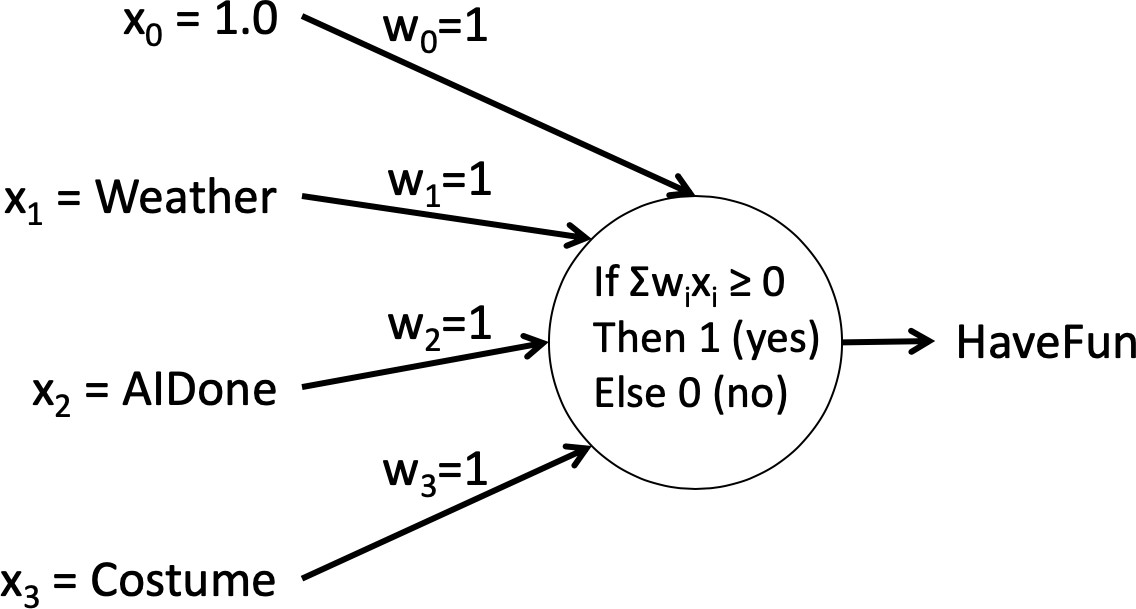
w0 = 0

w1 = -0.5

w2 = 1.5

w3 = 0.5

* 1. How would the learned perceptron classify the new instance <Weather=cloudy, AIDone=yes, Costume=no>? Show your work.



<Weather=cloudy, AIDone=yes, Costume=no>?

weather = cloudy = 2

AIDone = yes = 1

Costume = no = 0

* Value(weather)\*w1 + Value(AIDone)\*w2 + Value(costume)\*w3 + x0\*w0
* 2(-0.5) + 1(1.5) + 0(0.5) + 1(0) = 0.5 = yes

The perceptron would classify it as yes to HaveFun.

1. *CptS 540 Students Only*: Put the 11 training examples from the initial table above (used in Problem 1) into an ARFF file suitable for input to WEKA. Follow the procedure below to run the Naive Bayes classifier.
   1. Download and install WEKA from [www.cs.waikato.ac.nz/ml/weka/downloading.html.](http://www.cs.waikato.ac.nz/ml/weka/downloading.html)
   2. Start WEKA and choose the Explorer mode.
   3. Under the Preprocess tab, choose “Open file...” and load your ARFF file.
   4. Under the Classify tab, choose the “bayesNaiveBayes” classifier.
   5. Under Test options, choose “Use training set”.
   6. Click Start to run the classifier on your data.
   7. Include your ARFF file and WEKA’s output in your submission.

ARFF file:

@relation havefun

@attribute weather {clear, cloudy, rain}

@attribute aidone {yes, no}

@attribute costume {yes, no}

@attribute havefun {yes, no}

@data

clear,yes,yes,yes

clear,yes,no,yes

clear,no,yes,yes

clear,no,no,no

cloudy,yes,yes,yes

cloudy,no,yes,no

cloudy,no,no,no

rain,yes,yes,yes

rain,yes,no,yes

rain,no,yes,no

rain,no,no,no

WEKA output:

=== Run information ===

Scheme: weka.classifiers.bayes.NaiveBayes

Relation: havefun

Instances: 11

Attributes: 4

weather

aidone

costume

havefun

Test mode: evaluate on training data

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

Naive Bayes Classifier

Class

Attribute yes no

(0.54) (0.46)

=============================

weather

clear 4.0 2.0

cloudy 2.0 3.0

rain 3.0 3.0

[total] 9.0 8.0

aidone

yes 6.0 1.0

no 2.0 6.0

[total] 8.0 7.0

costume

yes 5.0 3.0

no 3.0 4.0

[total] 8.0 7.0

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.01 seconds

=== Summary ===

Correctly Classified Instances 10 90.9091 %

Incorrectly Classified Instances 1 9.0909 %

Kappa statistic 0.8197

Mean absolute error 0.2093

Root mean squared error 0.2438

Relative absolute error 42.1626 %

Root relative squared error 48.9498 %

Total Number of Instances 11

=== Detailed Accuracy By Class ===

TP Rate FP Rate Precision Recall F-Measure MCC ROC Area PRC Area Class

0.833 0.000 1.000 0.833 0.909 0.833 1.000 1.000 yes

1.000 0.167 0.833 1.000 0.909 0.833 1.000 1.000 no

Weighted Avg. 0.909 0.076 0.924 0.909 0.909 0.833 1.000 1.000

=== Confusion Matrix ===

a b <-- classified as

5 1 | a = yes

0 5 | b = no