CS4341 Introduction to Artificial Intelligence. A Term 2017

SOLUTIONS Exam 3 - October 10, 2017

Solutions by Prof. Ruiz, Ahmedul Kabir and Michael Sokolovsky
Department of Computer Science
Worcester Polytechnic Institute
Answers provided in green font.

Problem I. Decision Trees [20 Points]

Consider the following *Play Tennis* dataset (adapted from: Quinlan, "Induction of Decision Trees", Machine Learning, 1986) :

ATTRIBUTES:	POSSIBLE VALUES:	
Outlook	{Sunny, Rain, Overcast}	
Temperature	{Hot, Mild, Cool}	
Humidity	{High, Normal}	
Wind	{Strong, Weak}	
PlayTennis	{Yes, No}	<- classification target

ID	**Outlook**	**Temperature**	**Humidity**	**Wind**	**PlayTennis**
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Sunny	Hot	Normal	Weak	Yes
4	Sunny	Mild	Normal	Strong	Yes
5	Sunny	Mild	Normal	Weak	No
6	Rain	Mild	High	Strong	No
7	Rain	Mild	Normal	Weak	Yes
8	Rain	Cool	Normal	Weak	Yes
9	Rain	Mild	Normal	Strong	Yes
10	Rain	Cool	Normal	Strong	No
11	Overcast	Hot	High	Weak	Yes
12	Overcast	Cool	High	Strong	Yes
13	Overcast	Mild	High	Strong	Yes
14	Overcast	Hot	Normal	Weak	Yes

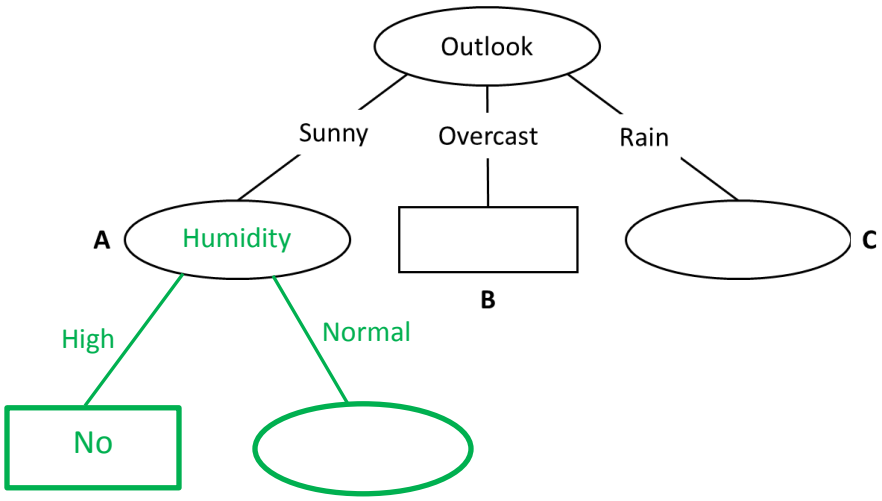
The machine learning task is to predict whether to play tennis or not, based on the data about the weather.

1. Which of the major machine learning categories (supervised, unsupervised, or reinforcement) does this problem fall under? Explain.

   Your answer [1 point]: Supervised

   Justification [2 points]: The data is labelled. That is, values for the target attribute PlayTennis are given in the dataset.

Suppose we are in the middle of inducing the decision tree. The current state of the decision tree is given below:



2. [2 points] What should the tree output In the leaf box labelled B?

   It should output **"Yes"** since all instances where Outlook=overcast have PlayTennis=Yes

3. [2 points] Which data instances should be considered in node A? Write down the relevant instance ID numbers from the table.

   All instances where Outlook = Sunny: data instances **1, 2, 3, 4, 5**

4. [7 points] Calculate the **entropy** at node A for the **Humidity** attribute. Show all the steps of your calculation. For your convenience, the logarithm in base 2 of selected values are provided.

x	1/2	1/3	2/3	1/4	3/4	1
$\log_2(x)$	-1	-1.6	-0.6	-2	-0.4	0

Entropy (Humidity) = Entropy ([0,2], [2,1])
= 2/5 * Entropy ([0.2]) + 3/5 * Entropy ([2,1])
= 2/5 * (- 0*$\log_2 0$ – 1*$\log_2 1$) + 3/5 * (- 2/3 $\log_2(2/3)$ – 1/3 $\log_2(1/3)$)
= 2/5 * 0 + 3/5 * (2/3 * 0.6 + 1/3 * 1.6)
= 0 + 3/5 * 0.9333
= **0.56**

5. [2 points] We have already calculated the entropies of Temperature and Wind at node A, and they are both 0.96. Based on these values, and based on the result you obtained in part 4 above, which attribute should be used to split node A? If needed, break any ties arbitrarily.

   Since the entropy of *Humidity* is smaller than those of *Temperature* and *Wind*, Humidity is used to split node A.

6. [4 points] Write your chosen attribute in the blank space of node A in the tree given in the previous page. Now extend the appropriate number of branches from node A and label them properly. For each branch that leads to a leaf, draw the leaf in the tree on the previous page and mark it with the output that the leaf should produce. Show your work on the tree on the previous page.

   [You can stop working on this decision tree after answering part 6.

   No need to construct the rest of the tree.]

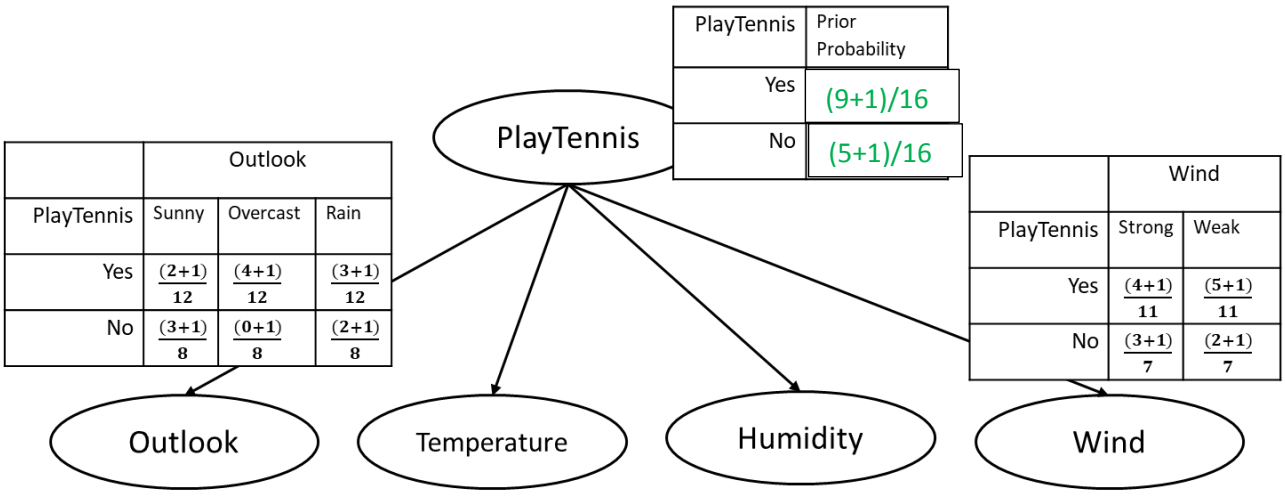
   Work shown on the tree on the previous page

Problem II. Naïve Bayes Models [25 points]

Consider the same *PlayTennis* dataset of Problem I, which is reproduced here for your convenience (though the data instances appear in a different order):

New ID	Outlook	Temperature	Humidity	Wind	PlayTennis
1	Sunny	Hot	Normal	Weak	Yes
2	Sunny	Mild	Normal	Strong	Yes
3	Rain	Mild	Normal	Weak	Yes
4	Rain	Cool	Normal	Weak	Yes
5	Rain	Mild	Normal	Strong	Yes
6	Overcast	Hot	High	Weak	Yes
7	Overcast	Cool	High	Strong	Yes
8	Overcast	Mild	High	Strong	Yes
9	Overcast	Hot	Normal	Weak	Yes
10	Sunny	Hot	High	Weak	No
11	Sunny	Hot	High	Strong	No
12	Sunny	Mild	Normal	Weak	No
13	Rain	Mild	High	Strong	No
14	Rain	Cool	Normal	Strong	No

1. [12 points] From the dataset above, the following Naïve Bayes model is built. Your job is to **fill up the likelihood tables** for the prior probability of **PlayTennis** and the probability of **Humidity** given PlayTennis. The other tables have already been filled for your convenience. Remember that 1 is added to all the counts to avoid the problem of having a probability that is equal to 0.



Temperature

PlayTennis	Hot	Mild	Cool
Yes	(3+1)/12	(4+1)/12	(2+1)/12
No	(2+1)/8	(2+1)/8	(1+1)/8

Humidity

PlayTennis	High	Normal
Yes	(3+1)/11	(6+1)/11
No	(3+1)/7	(2+1)/7

2. [13 points] Use the Naïve Bayes model constructed in the previous page to classify the following new instance. That is, determine which of the two values of PlayTennis (Yes or No) has the highest probability given the outlook, temperature, humidity and wind values of the given instance. **Show all your work and explain your answer.**

Outlook	**Temperature**	**Humidity**	**Wind**	**PlayTennis**
Sunny	Hot	Normal	Strong	?

Predicted value v for PlayTennis:

Predicted v = argmax_{v} P(v) * P(O=Sunny | v) * P(T=Hot | v) * P(H=Normal | v) * P(W=Strong | v)

For v = Yes: P(Yes) * P(O=Sunny | Yes) * P(T=Hot | Yes) * P(H=Normal | Yes) * P(W=Strong | Yes)

= (10/16) * (3/12) * (4/12) * (7/11) * (5/11) = 0.0150

For v = No: P(No) * P(O=Sunny | No) * P(T=Hot | No) * P(H=Normal | No) * P(W=Strong | No)

= (6/16) * (4/8) * (3/8) * (3/7) * (4/7) = 0.0172

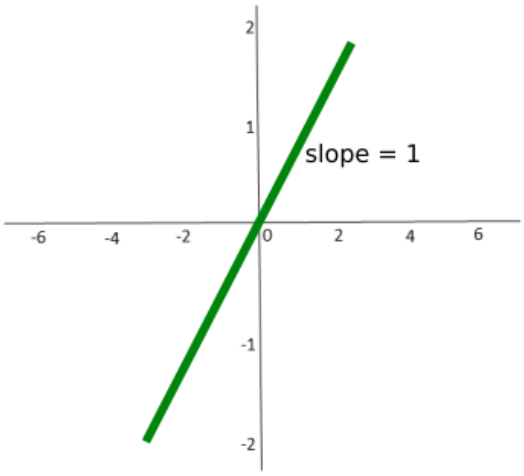
Since 0.0172 > 0.0150, the naïve Bayes model predict PlayTennis = No for this given instance.

At the end of the process, don't forget to say which value of PlayTennis (Yes or Not) the given instance is classified as by this naïve Bayes model.

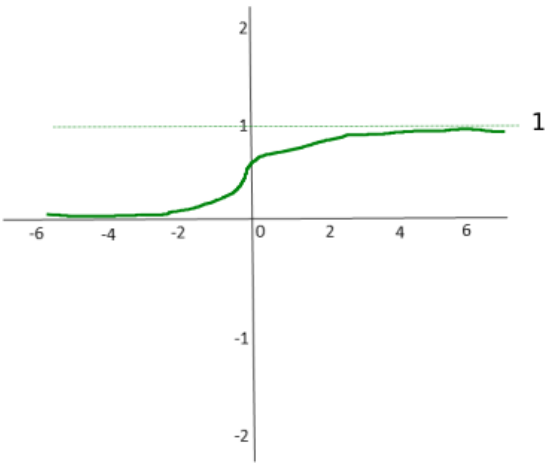
Problem III. Artificial Neural Networks and Deep Learning [25 points]

1. [8 points] For each of the following blank graphs, **draw the activation function** marked in the graph's caption. The horizontal axis marks x, the input to the activation function, and the vertical axis marks f(x), the output. Note that the two axes are scaled differently. The drawings don't have to be perfect, but should capture the essence of the functions.

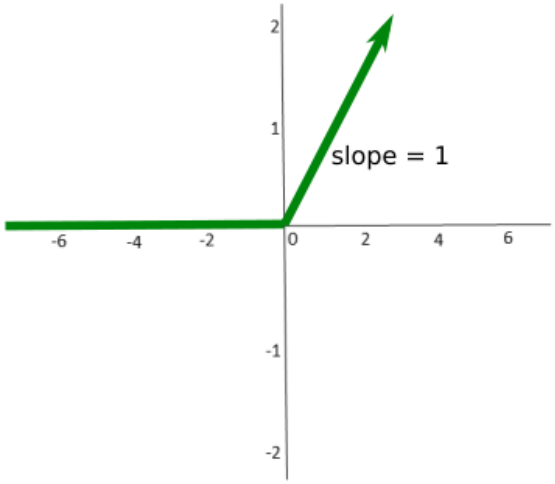
Linear



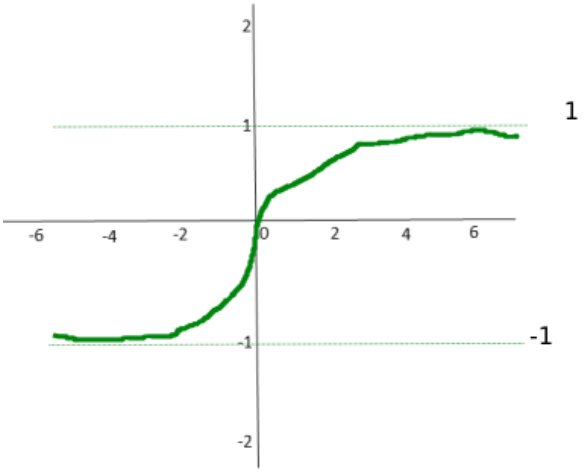
Sigmoid



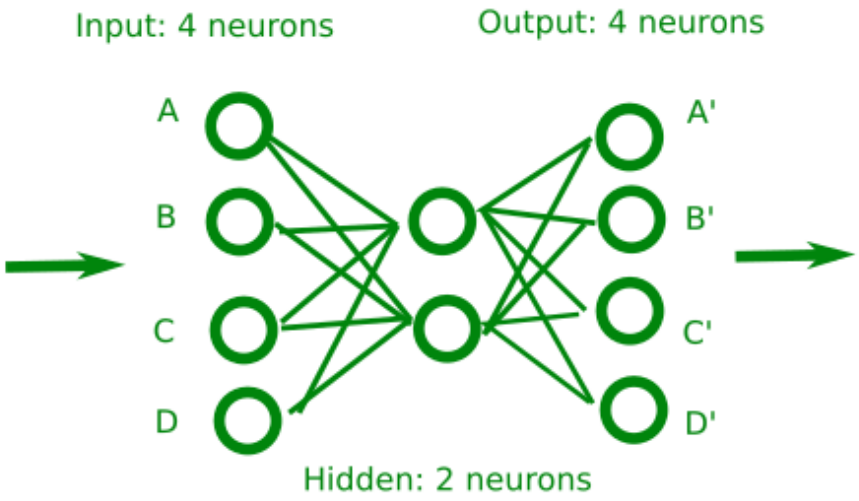
ReLU



tanh



2. Consider an **autoencoder** with 4 nodes in the input layer, 2 nodes in the hidden layer, and 4 nodes in the output layer.
    a) [4 points] Draw the autoencoder by drawing the nodes and showing the connections between them.



    b) Given the 4-dimensional dataset below:

A	B	C	D
1	0	0	0
0	1	1	1
1	0	0	1
1	0	1	0
0	0	1	0

Describe how the autoencoder above is trained by answering the following questions:

i. [3 points] What input values and output values will be provided to the autoencoder? Illustrate your answer with an example.

The input to the autoencoder will be each row in the above table, composed of four binary values A, B, C, and D. The output values provided to the autoencoder will have the same values as the input. This is because autoencoders are trained to try to re-create their inputs. For example, if data row 3: [1 0 0 1] is provided to the autoencoder as input then the autoencoder is expected to produce [1 0 0 1] as its output.

ii. [6 points] Describe in words what the training algorithm will do to find appropriate values for the connection weights of this autoencoder. Provide a concise but complete description of the process, no need for formulas. However, just providing the name of the algorithm/procedure is NOT enough.

Here is a list of steps describing the error back-propagation algorithm to train (i.e., search for appropriate values for the weights of) the neural network. The overall process follows a hill-climbing approach (more precisely a gradient descent approach).

1) The weighs of the network connections are initialized to small, random values.
2) One of the data instances (row) is used as input and fed forward through the neural network. Each node will calculate the weighted sum of its inputs, apply its activation function to this sum, and push forward this value to nodes in the layer next until output values are obtained for the output nodes.
3) The error (or loss) function is calculated on the outputs of the network with respect to the true label of the input data (i.e., the desired outputs).
4) The gradient of the error function is calculated using partial derivatives of this error function with respect to the weights of the edges connected to the output layer. This helps determine a delta for each of these weights (that is, by how much these weights should be changed so that the output layer produces values that are closer to the desired outputs).
5) Using these partial derivates, a "blame" or error is assigned to each output nodes.
6) The "blames" of the output nodes are propagated backwards through the net layer by layer to obtain a blame for each of the hidden nodes.
7) Based on the blames of a hidden node, a delta for the weight of each input edge to that node is calculated, which estimates by how much the weight should be changed.
8) In batch mode (or with a batch size greater than one), steps 2-7 are repeated for each of the data instances in the batch, and the deltas obtained for each weight are aggregated and then the weights are updated as described in 9 below. In stochastic grading descent mode (i.e., batch size = 1), the weights are updated after processing each data instance.
9) The weight of each edge in the network is updated by adding to it its corresponding (aggregated) delta(s).
10) This whole process is repeated using all the data instances for many iterations (or epochs) until one of the termination conditions is met.

iii. [4 points] When does the training terminate? List at least 2 stopping criteria.

There are several possible stopping criteria that a modeler could use:

1) The validation set error goes below an acceptable threshold
2) The validation set error plateaus (not improving) for a predefined number of epochs
3) The number of epochs (i.e., training iterations) reaches a predetermined threshold
4) The changes in weights between epochs is small or negligible
5) The validation set error reaches 0
6) The validation set error starts to increase due to over-fitting
7) Error never decreases and explodes. This could be because the step-size (learning rate) is too large

Problem IV. Machine Vision [10 points]

Answer the following questions related to the problem of recognizing objects from a given image. Concise and decisive answers are better than long, vague answers.

1. [4 points] What is segmentation? What is the input to the segmentation process? What output does it produce?

   Segmentation is the grouping of the pixels in an image into meaningful units, where the pixels within a group share certain characteristics.

   Input to the segmentation process: An image in the form of a matrix of pixels

   Output of the segmentation process: Groups (or regions) of apparently related pixels in the input matrix

2. [3 points] Mention 3 different approaches to determine the shape of an object in an image.

   The shape of an object can be determined from shading, texture, motion, and stereo vision

   Shading:
   - shadows provide an understanding of the curvature of an object
   - sudden shading changes represent corners or boundaries
   - uniform shading in a lit scene corresponds to flat surfaces

   Texture:
   - texture provides a perspective on the object, providing depth
   - a good example is the dimples on a golf ball

   Motion:
   - by monitoring progress as motion occurs, certain points can be traced and a shape model developed

   Stereo vision:
   - Two images of the same object taken from different places may be combined to infer the shape of the object

3. [3 points] After the pose and shape of an object are determined, describe how the final step of object recognition process is performed that identifies what the object is.

   For object recognition, the objects whose shape and pose have been identified in the previous steps, is matched against a database of known objects (usually called "models" or “templates”). For each known objects, several templates of the object may be needed to deal with rotations and/or translations. The model having the closest match is returned as the identification of the object.

Problem V. Natural Language Processing [20 Points]

1. [5 points] Given a sentence in natural language, what is the difference between semantic analysis and pragmatics analysis? Explain your answer.

Semantic analysis

- Translates the sentence into the internal knowledge representation of the computer program, so that the program can reason and use the information/knowledge encoded in the sentence.
- During this stage, a partial translation is made based on the parse tree(s) obtained in the previous syntactic analysis stage.
- Ambiguity is a possible complication for this stage. That is, sentences that have a unique syntactic structure but more than one possible meaning.
- For example, a parse tree containing the sentence "He loves Mary" may be translated in this step to ∃x loves(x,Mary)

Pragmatic analysis

- Translates the original sentence into a more complete internal representation of the computer program by taking into account the context of the sentence to disambiguate as much as possible the meaning of the sentence.
- During this stage, a full translation is made based on the representations obtained in the previous semantic analysis stage.
- Most of the ambiguity is removed at this step. However, possible complications in this stage are difficulties in understanding the intentions of the speaker's message, metaphors, and so on.
- For example, a knowledge representation like ∃x loves(x,Mary) can be translated to loves(John, Mary) by figuring out from context that x refers to John.

2. Suppose the following three documents are given
   Doc1: ***"The cat sat on the couch"***
   Doc2: ***"The dog chases the cat that sits on the couch"***
   Doc3: ***"When the cat is not chased, the cat sits on the couch and relaxes"***

   a) [9 points] Show the state of the bag of words models for these documents after each step of preprocessing. The first row is filled to serve as an example.

	After tokenization	After stop words removal	After stemming
Doc1	The cat sat on the couch	cat sat couch	cat sit couch
Doc2	The dog chases the cat that sits on the couch	dog chases cat sits couch	dog chase cat sit couch
Doc3	When the cat is not chased the cat sits on the couch and relaxes	cat chased cat sits couch relaxes	cat chase cat sit couch relax

3. Using the bag of words obtained after stemming from the previous table, construct a bag-of-words dataset where each word is a feature (= attribute or column) and each document is a data instance (i.e., row). You can do so by following the steps below:
    a) [2 points] Take the collection of words after stemming from the table on the previous page (right-most column) and list them (without repetitions) across the first row of the table below. You may not need all the spaces on the first row.
    b) [4 points] For each of these words and each of the documents, count the frequency (=number of occurrences) of the word in the document, and include that count in the cell at the intersection of the word's column and the document's row.

    One column has been filled up to serve as an example.

	cat	sit	couch	dog	chase	relax		
Doc1	1	1	1	0	0	0		
Doc2	1	1	1	1	1	1		
Doc3	2	1	1	0	1	1		