Assignment 1

CMPUT 466/566 Machine Learning

Due February 10, 2021

Contents

Question 1: Naïve Bayes for Classifying Movie Reviews

For this question we will be using the sentiment polarity dataset available from `http://www.cs.cornell.edu/people/pabo/movie-review-data/`. There are many versions available online. To avoid confusion, please use the version attached to the assignment. There are 5349 positive and 5346 negative movie reviews. The provided code uses the first 5000 reviews from each class as training set, and the remaining reviews as test set. The provide code skeleton uses scikit-learn's `CountVectorizer` to turn each review into a vector. You will need to install scikit-learn[1] on your own machine. You may not use scikit-learn's implementation of Naive Bayes, you need to implement it and the required smoothing yourself.

Part a. (466 students: 40 marks, 566 students: 40 marks)

Complete the provided code skeleton to implement Naïve Bayes with additive smoothing. Use the smoothing parameter $\alpha = 1$ as default.
Assume that each feature is binary (can only take on values 0 or 1). This is called Bernoulli Naïve Bayes, because all features are Bernoulli. Essentially, we don't care if a word appears multiple times in a sentence, we are just recording if it occurs at least once or not at all.

[1] `https://scikit-learn.org/stable/install.html`

Use the skeleton code provided, as we will be automatically testing your code. Recall that for Naïve Bayes, the classification decision is:

$$\hat{y} = \underset{y}{\operatorname{argmax}}\Big(P(y) \prod_{i=1}^{p} P(x_i|y)\Big), \tag{1}$$

where y is the class or label, p is the total number of attributes (features) in $\mathbf{x}$, and x_i is the i-th feature of the vector $\mathbf{x}$. Consult the lecture notes for information on how to calculate these values.

Hint: Naïve Bayes can be made much faster by using numpy and vectorized operations. Without this, running the experiments may require much more time. Be sure to leave enough time to run your code, in any case.

Hint: Computers don't like it when you multiply many small numbers together, and you are likely to get underflow errors if you don't *take the logarithm* of Equation 1. Recall that log is a monotone function, so we will still get the same $\hat{y}$ from Equation 1. Recall also that the log of a product is the sum of logs.

Part b. (466 students: 15 marks, 566 students: 10 marks) Analyzing the Effect of Smoothing Parameters

Implement additive smoothing in a way that you can easily vary the value of α in your code. Hand in a plot of the accuracy on the test set as a function of α, varying from 0.1 to 3 (use steps of 0.1). Based on your experiment, what is the optimal value for α?

Part c. (466 students: 15 marks, 566 students: 10 marks) Stop Words

In this part, we are going to examine the effect of removing stop words from the dataset. Change the arguments of `CountVectorizer` to remove English stop words: stop_words='english'. Include a new plot like that in part b, but include a line for the accuracy when you remove stop words. Does stop word removal help for this classification task? Reason about your answer.

Part d. (466 students 2 ***bonus*** marks, 566 students 10 marks) Another smoothing method

Jelineck-Mercer smoothing is another method for smoothing to account for zero count word,class counts. It trades off an estimate of the word's probability given the class with the word's probability globally. The formula is:

$$p\big(x_d = 1 \,|label = c\big) = (1-\lambda)\frac{count\big(\text{"}x_d = 1\text{"} \;\; and \;\; \text{"}label = c\text{"}\big)}{\sum_j count\big(\text{"}x_j = 1\text{"} \;\; and \;\; \text{"}label = c\text{"}\big)} + \lambda\frac{count(\text{"}x_d = 1\text{"} \; in \; dataset)}{size \; of \; dataset} \tag{2}$$

Implement this new smoothing method, an compare the results to the results you obtained in part c by plotting with and without stop word removal. What do your results tell you?

Question 2: Decision Trees (Same marks for 466 and 566)

a. Exploration (7 points)

To start, visit `https://mlweb.loria.fr/book/en/decisiontree.html`. There, under "In pictures", you will find an interactive decision-tree.

You can use
http://web.archive.org/web/20190620153054/https://mlweb.loria.fr/book/en/decisiontree.html if there are issues accessing the page.

Refresh the page until there are exactly three leaf nodes (this should happen immediately most of the time). By "exactly three leaf nodes", we mean there should be three nodes on the tree titled "label x". See question 2b for an example of what things should look like.

a.i (3/7 points)

To start, hover over the top node of the tree, and examine the data points on the graph. What part of the graph is highlighted?

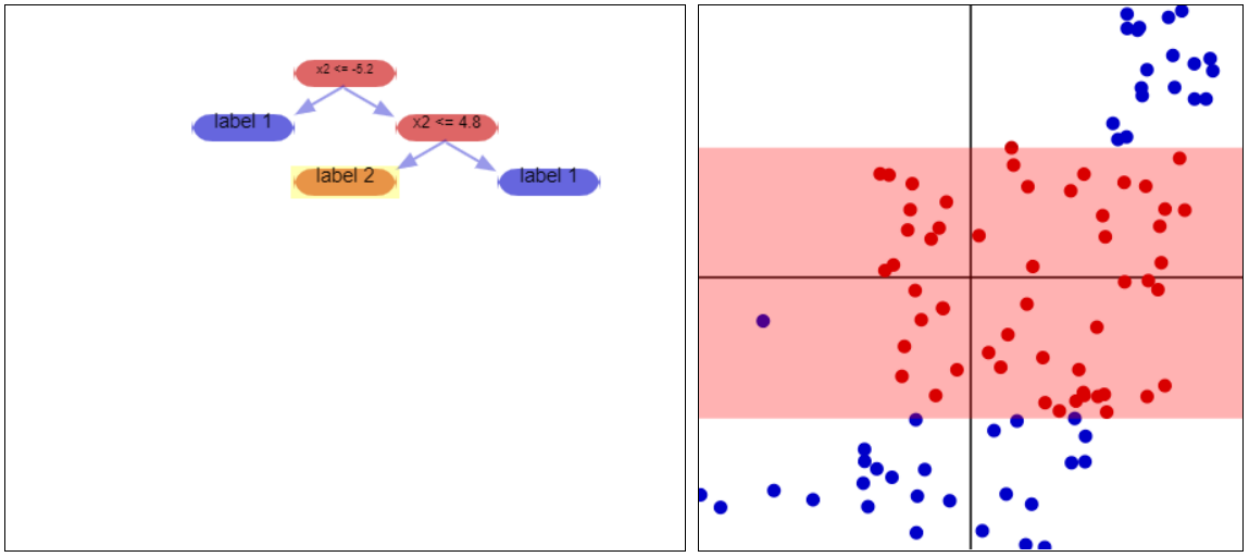
Move your cursor to one of the two nodes below the root node. Why has the highlighted region changed? What does the highlighted region indicate?

a.ii (4/7 points)

Hover over the leaf nodes of the tree (the nodes titled "label x"). If you reach here with a classifier, what does that mean in terms of your classifier result? Include a screen-snip of the graph when hovering over the leaf node covering the largest area/region (this should be a large, horizontal, center red region, like in the image in Question 2b).

1. Do all of the vectors in the highlighted region have the same color (their true label)?
2. What will the classifier output be for this region?
3. Will the classifier correctly label all of the vectors in this region? Why or why not?

b. Evaluation (7 points)



b.i (3/7 points)

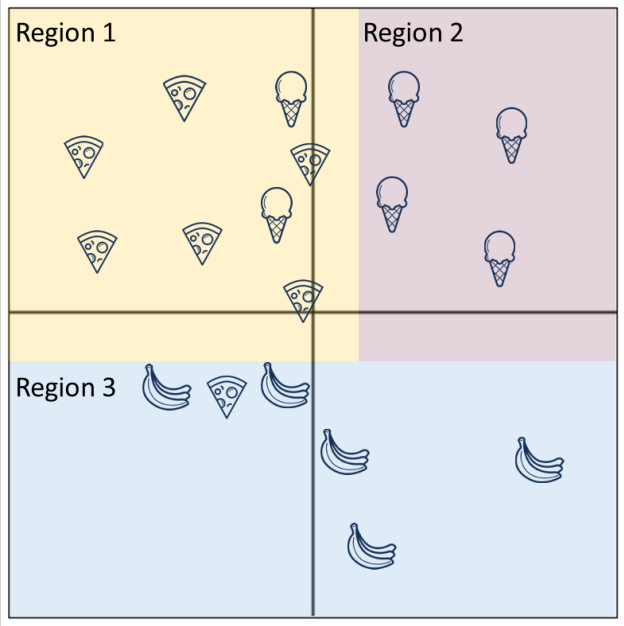
In the above image, there is a lone blue dot in the red region sitting off to the left. What kind of measurement is being used to decide not to split that off into its own region/add new nodes to the graph to accommodate for it? (There are several possible correct answers here - refer to the website or the lecture notes for some specific wordings if in doubt.)

b.ii (4/7 points)

What is a potential pro of splitting the blue dot off into its own node/region? What is a potential con? When answering this, you will want to consider that this the set of vectors we are looking at (and that we have trained on) just represents a sampling from some larger real-world data.

Think about the trade-off between these two, and whether/why you think the current depth of the tree could be useful, and whether you think going further could also potentially be useful. **You do not have to write anything down here**, but mull this over.

c. Calculation (11 points)



The above image has vectors in a plane. The vectors are represented by images corresponding to their "true" labels - so there is a vector near the origin that has a "pizza" label. There are three types of labels - pizza pieces (🍕), ice cream cones (🍦), and bananas (🍌),

c.i (6/11 points)

What is the entropy of the set of vectors in Region 1? Show your work.

What is the entropy of the set of vectors in Region 2? Show your work.

What is the entropy of the set of vectors in Region 3? Show your work.

c.ii (2/11 points)

What is the average entropy across all regions, weighted by cardinality/number of elements? Show your work.

c.iii (3/11 points)

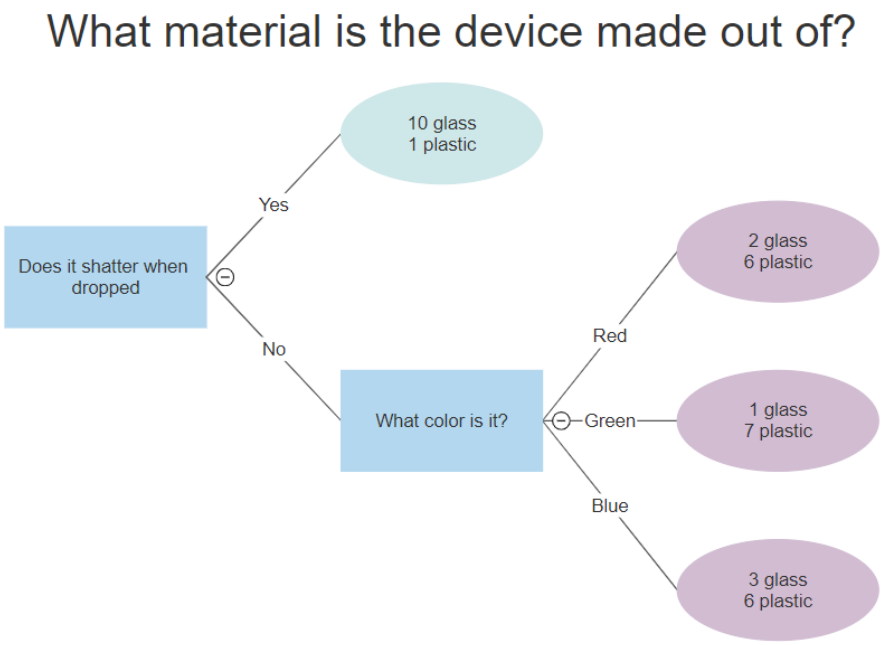
The average entropy (or some other measure of pureness) is what a top-down decision tree induction (like the ID3 algorithm in the lecture notes) uses to construct a tree. In this way, it is what's known as a *greedy algorithm* At each step, it tries to minimize the average entropy of its resulting

regions/nodes (in practice, there's some amount of entropy that the algorithm considers "okay" and stops splitting nodes once it has reached that point).

By greedily minimizing the entropy at each step, do you end up with the simplest tree that classifies everything? Looking up answers to this is fine, but be sure to understand the answer and put it into your own words.

d. Pruning (5 points)

Suppose you had the following decision tree



Recall that we can prune a tree after creating it with the χ^2-test.[2]

In broad strokes, we look at decisions that result in terminal (leaf) nodes and ask, "do the decisions impact the labels of the outputs?" If there's no connection, we eliminate those terminal nodes/the decision that led to them. We then repeat the process until we reach the final tree.

In more specific strokes: first, we examine some leaf nodes with a common parent. We look at the decision used to produce those nodes from that parent and create a null hypothesis: "there is no dependency between the outcomes of this decision and the labels of the vectors matching each

[2]You may wish to consult the "Example chi-squared test for categorical data" on the Wikipedia page: `https://en.wikipedia.org/wiki/Chi-squared_test`. In that example, you could think of the neighbourhoods being the "decision," and the occupation type as the label to be predicted.

outcome." We assume (by default) that the null hypothesis is true, and perform the statistical test to see if that hypothesis is supported by our data.

- If it is ($p \geq 0.05$) - that is, if it seems the outcomes and labels are independent, as we assumed - we *do* prune the nodes.
- Otherwise, we reject the null hypothesis - this indicates the outcomes and labels actually *are* related, contrary to our assumption - and we *don't* prune the nodes.

We repeat this process until all leaf nodes of the *final* tree have been examined in this way (after initial pruning you might have new leaf nodes to consider).

You can use the chi-squared calculator at `https://www.socscistatistics.com/tests/chisquare2/default2.aspx`.

d.i (5/5 points)

Perform pruning on this tree using χ^2-tests (chi-squared tests) for $p < 0.05$. Show the contingency table, the χ^2 value, and the p-value for each of your tests.