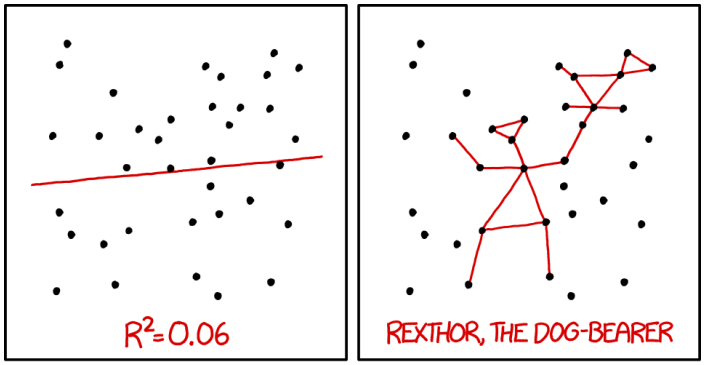
CSE 353: Homework 2 Solutions — Due Friday September 23



(credit: xkcd)

1. **(3 pts, 1 pt each)** *Machine learning fundamentals.* In the following scenarios, discuss whether or not you believe that I did a good job in *model generalization.* Justify your answer.

   (a) I work in real estate. I have observed, over 10 years of experience, that large houses on Long Island tend to sell at around $500,000 or more, while small houses sell at around $400,000 or fewer. I move to the Texas Panhandle, and find a house that is very large, and conclude that it should be worth around $500,000 or more.

   (b) I am a young upshot professional and I would love to live in New York City. I talk to my friend, who lived in NYC in the late 2000's, and she tells me how factors affect rent prices (quality of apt, number of bedrooms, neighborhoods, allow pets, etc.) The year is now 2020. Does her advice generalize to my current situation? (Hint: Think COVID.)

   (c) I am a computer science expert who has spent the past 5 years classifying images of dogs and cats. I'm pretty good at this now, and my 1 million layer neural network achieves 100% accuracy (in testing and training) on this task. Now, the government wants me to apply my neural network to remote sensing satellite images, to detect underground marijuana farms. I use the same neural network, with no tweaking, on this new task, and expect the same level of accuracy.

2. **(2 pts, 0.5 points each)** *Conditional independence vs independence.*

   Tom is a blue-gray cat with a bushy tail, and Jerry is a brown mouse with a rope-like tail. After many years of fighting, they both decided to settle down, and now have thriving families. Tom has 10 kids and Jerry has 40 kids. Tom's kids are all cats like him, with bushy tails. Half of Tom's kids are blue, while the other half is gray. Jerry's kids are all brown mice, with rope-like tails.

   (a) I pick up a baby animal at random. What is the probability that ... (fill in the table)

fur \ tail	furry	rope-like
blue		
gray		
brown		

(b) Are the features "fur color" and "tail texture" independent or dependent, without knowing the type of animal? (Show mathematically.)

(c) Now Tom comes over and says, "I'm very proud of my baby girl, of whom you are holding." What is the probability that (fill in the table)

fur \ tail	furry	rope-like
blue		
gray		
brown		

(d) Are the features "fur color" and "tail texture" independent or dependent, now that I know the animal is Tom's cherished baby daughter? (Show mathematically.)

3. **(2pts)** *Directed graphical models and probability inference* I have 4 tops: a red sweater, a blue T-shirt, a green hoodie, and a white tank top. I need your help to decide what to wear.

(a) I decide what to wear based on four factors:

- if it's raining
- if I want to take a walk outside
- if I feel sick
- the day of the week it is

(0.5pts) Using the Naive Bayes assumption, draw a graphical model that indicates how I will make a decision of what to wear each day.

(b) **(1pt)** To be more specific,

- I only wear the green hoodie when I walk outside, regardless of all other factors.
- If I feel sick, I will walk outside 10% of the time. If I feel well, I will walk outside 60% of the time.
- When it rains, I feel sick 70% of the time; otherwise, I feel sick 15% of the time.

Draw a corresponding graphical model for determining whether I wear a green hoodie. Given that it is raining, infer the probability that I am wearing a green hoodie.

(c) **(0.5pts)** The probability that I wear a tank top, independently of all the other clothes, is 75% if it's raining and 25% if it's not raining.

- Today is Monday and it is raining.
- The probability that it will rain, given that the previous day rained, is 70%. The probability that it will rain, given that the previous day did not rain, is 10%.

Draw a graphical model predicting whether I will wear a tank top on Wednesday, and calculate this probability.

4. **(3 pts) Naive Bayes and Alice in Wonderland.**

- Our goal is now to do word prediction, using the n preceding words as n features for each word. As an example, consider the following text

```
The quick brown fox jumped over the fence
```

Then, if $n = 1$, our training data will be formed as

features	the	quick	brown	fox	jumped	over	the
label	quick	brown	fox	jumped	over	the	fence.

If $n = 2$, our training data will look like

features	the, quick	quick,brown	brown,fox	fox,jumped	jumped,over	over,the
label	brown	fox	jumped	over	the	fence.

and so forth.

That is, if we think of each task as taking a list of features and spitting out a label, then

- Open the python notebook `hw2_alice_naivebayes_release.ipynb` . After running the first couple boxes, you should have loaded the entire text of "Alice in Wonderland" by Lewis Caroll, as an ordered list of words. Our task today will be to do word prediction based on this corpus. Throughout this exercise, this corpus will serve as both our training and testing data.
- **Tokenize** While the exact word means a lot to us, for a (primitive) computer, a word is just some object; in particular, we represent each unique word as a unique number. This is the word's token. Run the 3rd block to tokenize the data, and understand what it is doing.
- **Counting of past words.** In the next box, I give you code that generates a lookup table, which counts how many times a word precedes a previous word. This matrix is then $V \times V$ where $V = 2637$ is the number of distinct words. Each row of that matrix is indexed by y the label, and each column by x the feature.

(a) **(1 pt) Bigram classifier**. Use the table given to create a bigram classifier, e.g. predicting the next word using only the previous word ($n = 1$). That is, construct a calculation for the posterior $\mathbf{Pr}(x|y)$ and the prior $\mathbf{Pr}(y)$ based on the statistics of the corpus. Do not worry about normalization, e.g. the likelihood function can return the first term of

$$\mathbf{Pr}(x|y)\mathbf{Pr}(y) \propto \mathbf{Pr}(y|x).$$

Now construct a Bayes classifier using only this feature. To check your work, the following words should be the highest likely bigrams.

- word: `alice`. Most frequent previous word: `said`, with probability 0.00534.
- word: `the`. Most frequent previous word: `said`, with probability 0.00233.
- word: `cat`. Most frequent previous word: `the`, with probability 0.0396.
- word: `turtle`. Most frequent previous word: `mock`, with probability 0.00221.

Report the classification accuracy over the entire corpus of this classifier.

(b) **(0.5 pts)** Well, that was pretty terrible. Let's try and incorporate not just the past word, but the past k words, where k can be up to 30. (You can use a tensor and a loop to do this.) Using the code given as a guide, construct more tables that records the counts for how often a word is 2 tokens before another word. Use this to fill in the posterior for $\mathbf{Pr}(x|y)$ where now x is the 2nd previous word. Now construct a Naive Bayes classifier that uses both the past and past 2nd word as features. Remember that using the Naive Bayes assumption,

$$\mathbf{Pr}(y|x_1, x_2) \propto \mathbf{Pr}(x_1, x_2|y) = \mathbf{Pr}(x_1|y)\mathbf{Pr}(x_2|y).$$

Make sure that your model respects this independence. Here are some sanity checks:

- seed: `'before', 'she', 'found', 'herself', 'falling', 'down', 'a', 'very', 'deep'`. Next most likely word: `well`, with probability 0.00276
- seed: `'what', 'an', 'ignorant', 'little'`. Next most likely word: `shoulders`, with probability 0.000474
- seed: `'four','thousand'`. Next most likely word: `miles`, with probability 0.000118

Report the classification accuracy over the entire corpus of this classifier, for $n = 3, 5, 10$. (Note that this is not an n-gram classifier, which would be the not-Naive-Bayes version of what we are doing here.)

(c) **(0.5 pts) Text generation** Using the likelihoods computed from the classifier using $n = 3$ (3 past words), and starting with a seed phrase "the mad hatter", generate the next 25 words by always picking the *most likely* next word.

(d) **(1 pt) Text generation** Using the likelihoods computed from the $n = 3$ classifier, and starting with a seed phrase "the mad hatter", generate the next 25 words by always picking the next word *by sampling according to that probability.* (Hint: use `random.choices()`.)