Assignment #10

Due: 23:55pm April 29, 2020

Upload at: https://www.gradescope.com/courses/75501/assignments/466553

Assignments in COS 302 should be done individually. See the course syllabus for the collaboration policy.

Remember to append your Colab PDF as explained in the first homework, with all outputs visible.
When you print to PDF it may be helpful to scale at 95% or so to get everything on the page.

Problem 1 (18pts)

Consider the univariate function

$$f(x) = x^3 + 6x^2 - 3x - 5\,.$$

Find its stationary points and indicate whether they are maxima, minima, or saddle points.

Problem 2 (30pts)

In this problem, you will use the entire 50k-digit MNIST data set. To remind you, the data are 28×28 greyscale images of the digits 0 through 9. Download the `mnist_full.pkl.gz` file and load it with something like this:

```
import pickle as pkl
import numpy as np
import gzip
with gzip.open('mnist_full.pkl.gz', 'rb') as fh:
  mnist = pkl.load(fh)
```

This will give you a dictionary `mnist` with keys and values that should be self-explanatory. It is possible that when you run the code to solve the problems below, that you run into issues in which the covariance matrix is not positive definite. To solve this, add a little bit of diagonal to it, i.e., add $\alpha\mathbb{I}$, for $\alpha \approx 10^{-6}$.

(A) Compute the empirical mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ of the training images.

(B) Reshape and render the mean as an image using `imshow`.

(C) Generate 5 samples from the multivariate Gaussian with parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ from (A). Do this only using `numpy.random.randn` and linear algebra operations. Reshape and render these samples using `imshow`.

(D) Now iterate over each of the possible labels from 0 to 9. Compute the mean and covariance of the training data that have that label. Here's a sketch of some code for getting the correct images:

```
for label in range(10):
  indices = train_labels == label
  images = train_images[indices,:]
```

Render the mean of each as an image, and generate 5 samples from the Gaussian distribution with the label-specific mean and covariance. Render these samples.

Problem 3 (20pts)

Consider the following joint distribution over random variables X and Y:

Y \ X	x_1	x_2	x_3	x_4	x_5
y_1	0.01	0.02	0.03	0.1	0.1
y_2	0.05	0.1	0.05	0.07	0.2
y_3	0.1	0.05	0.03	0.05	0.04

Use Colab to compute the following quantities. Remember to compute these in **bits** which means using base 2 for logarithms. Here's the PMF in Python to make life a bit easier:

```
PXY = np.array([[0.01, 0.02, 0.03, 0.1, 0.1],
                [0.05, 0.1, 0.05, 0.07, 0.2],
                [0.1, 0.05, 0.03, 0.05, 0.04]])
```

(A) What is the entropy $H(X)$?

(B) What is the entropy $H(Y)$?

(C) What is the conditional entropy $H(X \mid Y)$?

(D) What is the conditional entropy $H(Y \mid X)$?

(E) What is the joint entropy $H(X, Y)$?

(F) What is the mutual information $I(X\,;\,Y)$? Compute it once using the PMF and then once using relationships between the quantities you used in (A)-(E) to verify your answer.

Problem 4 (30pts)

In this problem you'll compute some empirical estimates of entropy for text. We'll use the collected works of Shakespeare, with some preprocessing. Then we'll look at statistics of the characters (ASCII characters that is!). To get started, grab the file `shakespeare.txt` and load it using code something like this:

```
import re
with open('drive/My Drive/COS 302/shakespeare.txt', 'r') as fh:
  text = ' '.join([line.lower() for line in fh.readlines()])
unigrams = list(re.sub(' +', ' ', re.sub('[^a-z ]+', '', text)))
bigrams = [s1 + s2 for (s1, s2) in zip(unigrams[:-1], unigrams[1:])]
```

You don't need to understand in detail what I'm doing here, but unpacking this a little bit for pedagogical purposes: this loads all the lines of the file, makes them lowercase, and joins them into one big string. Then it uses two regular expression substitutions; one strips out all characters that aren't letters or spaces, the other collapses consecutive whitespace into one whitespace. Then it calls `list()` on the string to make it an explicit list of characters. Then to get pairs of characters, I'm indexing twice one starting at zero and the other starting at one, so offset. The `zip()` function takes two lists and turns it into a list of aligned tuples. Then I do a list comprehension over that to join the characters into one string.

(A) Imagine that Shakespeare produced random characters selected uniformly from the 27 we're considering here. What would the entropy of the random characters be?

(B) Presumably Shakespeare did not draw characters randomly (infinite monkey theorem notwithstanding). Use `numpy.unique` to estimate the character-wise entropy.

(C) If Shakespeare instead wrote text by producing uniformly random pairs of characters, what would the entropy of those pairs be? (You can ignore the fact that we're removing consecutive whitespaces.) Explain how this number should relate to (A).

(D) Do the same thing as in (B) but compute the empirical estimate of entropy for bigrams. Think of this as the joint entropy of two consecutive characters. (Remember that when computing entropy we consider $0 \cdot \log_2(0) = 0$.)

(E) Based on these computations, what is your estimate of the mutual information between two consecutive characters?

Problem 5 (2pts)
Approximately how many hours did this assignment take you to complete?

My notebook URL: https://colab.research.google.com/XXXXXXXXXXXXXXXXXXXXXX

Changelog

- 26 April 2020 – Fixed typo in shakespeare.txt URL.
- 26 April 2020 – Fixed typo in 2D code.
- 21 April 2020 – Initial version