Exercise 3

Due May 11 at 11:59 pm to Gradescope

****Answer any 3 of 4****

Question 1 - Automatic Differentiation (10 points)

a. Using automatic differentiation in PyTorch, JAX, TensorFlow, or another tool of your choice, evaluate

$$\left.\frac{\partial y}{\partial x}\right|_{x=1.2},$$

where

$$y = \sin(x) \cdot x^{2.3} - x^{\log(|x|)}$$

We have some tutorials for doing this in JAX and PyTorch. You can do this in TensorFlow by writing a function $f(x)$ that produces y. Then, create a TensorFlow variable, `x` via

```
tf <- reticulate::import("tensorflow")
x <- tf$Variable(1.2, dtype = tf$float32, trainable = T)
```

Wrap the evaluation of `f(x)` in a "gradient tape":

```
with(tf$GradientTape(persistent = T) %as% tape, {
   y  <- f(x)
})
dy_dx  <- tape$gradient(y, x)
```

If `persistent=F`, you'll only be able to access the gradient information in `tape` once before it's cleared from memory. The documentation for `tf$GradientTape` is useful, just remember you'll need to convert `.`'s to `$`'s when working in with Python packages in `R`.

For review, note that you can also do this in `JAX` by writing a function, `f`, creating its derivative function via `df <- jax$gradient(f)`, and evaluating that at `x=1.2` using `df(1.2)`

b. Using automatic differentiation, evaluate the Hessian matrix of

$$y = \sin(x) \cdot x^{2.3} - x^{\log(|x|)} - z^2 \cdot \log(z)$$

at $x = 1.2, z = 2$. The Hessian matrix is the matrix of second-order derivatives of a function $y = f(\mathbf{I})$:

$$H_{d,d'} = \frac{\partial^2 y}{\partial I_d \, \partial I_{d'}},$$

where I_d refers to the d^{th} input. It is used in various statistical and optimization settings.

In TensorFlow, you can do this by wrapping the output of one gradient tape within another:

```
with(tf$GradientTape(persistent = T) %as% outer_tape, {
 with(tf$GradientTape(persistent = T) %as% inner_tape, {
    y  <- f(x, z)
 })
 dy_dx  <- tape$gradient(y, list(x,z))
})
Hessian_f <- tape$jacobian(dy_dx, list(x,z))

# ! Note: This code is for general illustration only
#  You may need to adapt it for the task at hand!
```

You can also do this in `JAX` by writing a function, `f`, creating its Hessian function via
`Hessian_f <- jax$hessian(f,argnums = 0L:1L)`
and evaluating that at `x=1.2,z=2` using `Hessian_f(1.2, 2)`

c. Explain how, when building a neural network model, the each of the following activation functions could be problematic. Describe how you could potentially address those limitations.

   - $f(x) = \text{sigmoid}(x) = 1/(1 + \exp(-x))$
   - $f(x) = \text{relu}(x) = \max(0, x)$
   - $f(x) = \exp(x)$
   - $f(x) = \mathbb{I}\{x \geq 0\}$

d. Explain where automatic differentiation may be useful in machine-learning related social science research relevant to your interests— that is, where obtaining gradient information may be important but might be hard to get in some other way (i.e. analytically).

Question 1 - Answer

Question 2 - Text Representations & the Trump-Clinton Debates (10 points)

Consider the Trump-Clinton debates data stored in `TrumpClintonDebates_clean.csv`. Each row represents an exchange in the debate. The column `moderator_text` contains the most recent text of the debate moderator; `clinton_text` contains text of the most recent statement by Clinton; `trump_text` contains the text response by Trump. This text has been processed and some words lemmitized. Note that the last moderator statement text is sometimes repeated (i.e. in long Trump-Clinton exchanges where the moderator does not interject in the discussion).

a. Represent the text of the moderator, Clinton, and Trump numerically using a text representation protocol. One approach would be to find the average word embedding for each speaker, for example. In that case, your output would be 3 matrices (of dimensions D, where D is the dimensionality of your vector space). Regardless of your cohice, use PCA to reduce your text representation of the moderator, Trump, and Clinton, and visualize the first 2 PCA dimensions, clearly indicating which points represent the text of each speaker. Do the texts of the three actors seem to cluster in different parts of your text representation space?

b. Using your work from the previous part, analyze the determinants of Trump's text. First, decide on an outcome. One outcome could be, for example, the length of Trump's text. Another could be a particular embedding dimension of his response. Once you've chosen an outcome, use the moderator and Clinton's text representations to predict Trump's response. How predictable is Trump's response? Quantify via some measure of predictive accuracy on an out-of-sample set (classification or regression metrics are okay, depending on your outcome). This task is open ended; explore; explain your choices and your findings, citing strengths and limitations.

Question 2 - Answer

Question 3 - Text Representations & News Article Popularity (10 points)

We're going to use news dataset described here:

`https://archive.ics.uci.edu/ml/datasets/News+Popularity+in+Multiple+Social+Media+Platforms`

We're going to focus on the `News_Final.csv` file, in particular the `Facebook` and `LinkedIn` popularity scores that range from $[-1, \infty]$ and are associated with share metrics on each platform.

a. Construct the following outcome:

$$Y_i = \log(2 + \text{LinkedIn Popularity}_i) - \log(2 + \text{Facebook Popularity}_i).$$

Why do we take logs here? Why do we add 2 here? What would you suspect predicts excess LinkedIn vs. Facebook popularity?

b. We're going to try to use the `Headline` text to predict the difference in popularity of article on LinkedIn and Facebook using the outcome just constructed. Build or apply a machine learning model using the headline text to predict Y_i on a training set with 80% of the data.

   - You could apply a pre-trained LLM; you could use headline-level if-idf weighted word counts (processed in some way) as features fed into a Lasso or Forest model. You could use a Transformer or bidirectional LSTM model and pre-trained word vectors to exploit the time series structure of the language. Your selection should not be based on what you know right now, but on what you would most like to learn so as to incorporate it into your research!

c. List out the headlines associated with the highest and lowest values of Y_i in the held out set as well as the headlines with the largest 5 mis-predictions. Plot the predicted vs. true values of the out-of-sample predictions in a scatterplot.

d. Discuss how you would probe model interpretabilitiy. No calculations are required, but outline what calculations you would do to try to better understand what your model is paying attention to. Answers will vary based on choice of model.

Answer 3 (10 points)

Question 4 - Introduction to Image Models (10 points)

Say that you read the following paper, and now want to replicate a version of the method in your own work using satellite images in the social science context. A benefit of the proposed method is that we can avoid the complicated training of a full image model, but still use rich image features.

> Rolf, Esther, et al. "A generalizable and accessible approach to machine learning with global satellite imagery." *Nature communications* 12.1 (2021): 1-11.

In the `./data/` folder, we have saved `ImageData_Pset3.Rdata`. You can load this data into your `R` session using the `load` function. This data corpus contains satellite imagery for 1000 villages in Nigeria (saved as object `.`) and accompanying outcomes, which you can think of here a measure of poverty, (saved as array `obsY`).

a. What are the dimensions of the array `m`; explain them in turn. How many image bands do we have in these images? What are the spatial dimensions of the image?

   We're now going to implement a similar algorithm to Rolf et al. The core idea is that we are going to first extract *random* features from the images, and then optimize how to combine those random features in the vector representation of the image to predict the outcome using a robust method like ridge regression. Essentially, we are going to fit a non-optimized 1-layer CNN network, whose output is then used in a downstream linear model. We will walk through the steps together.

b. Divide the data into a training (80%) and test set (20%).

c. Generate 100 random 5×5 random filters (random "mental images"). Note that the initial values of the TensorFlow, PyTorch, and JAX filters are themselves random, so you don't have to worry too much about setting that part up.

d. Apply those random filters to the images. Take the global maximum and global average across the convolved image. Concatenate these to form a 200-dimensional (randomized) tabular representation of the image.

e. Your new image representation is in tabular form. Train a ridge regression model using the `cv.glmnet` defaults (with `alpha = 0`) to predict `obsY`. Use only training set data in the training process!

f. Report the Mean Squared Error (MSE) on the test set using the model trained in the prior step. Does your model perform better in terms of out-of-sample MSE than guessing the training set average outcome?

Hints: We might need to break up our computation over the image into batches of 10-20 images so our computer memory usage is still manageable.

Answer 4 (10 points)