CSCI 362: Machine Learning **Assignment 4**
Fall 2023 **Stochastic Gradient Descent**

Please submit a single document for this assignment on Canvas by the beginning of class on **Monday, Oct. 2rd**, and finish your peer review no later than **Friday, Oct. 6th**.

Part I: Implement mini-batch gradient descent.

- Download or cut-and-paste `ols_nn.py` from our Github repo to your computer.

  The program `ols_nn.py` uses *batch* gradient descent to train a linear *model* to fit the climate data.

  Here *batch* means that the model sees *all* of the data during each pass of the training loop; that is, during each iteration of the training loop, all of the data are used when computing the gradient.

- Modify `ols_nn.py` by defining a variable (before entering the training loop) called `batchsize` or similar. Set `batchsize` to say 4, initially.

  Modify the training loop in `ols_nn.py` so that the model is shown mini-batches of examples, each of size `batchsize` (instead of all of the data), during each forward pass.

  Organize your code so that, if you set `batchsize = 32`, then you recover batch gradient descent; and if you set `batchsize = 1`, you get stochastic gradient descent as discussed in class.

  Make sure your code works as expected when you set `batchsize = 32`. You will likely need to adjust your learning parameters in order to recover high-quality convergence if `batchsize` is significantly smaller than 32.

- Submit a **readable** screenshot of your modified training loop including any surrounding code that you modified or added. Also include a readable screenshot with the output of a run demonstrating good convergence with `batchsize = 4`.

- Note: if one accumulates the loss outside the inner `for` loop, as Simmons did in class, then the appropriate adjustment is: `accum_loss * batchsize / num_examples`.

Part II: Peer review one of your classmate's implementation of SGD.

- On Canvas, you will be assigned to peer review another student's solution for Part I.

- In your peer review, please comment on

  1. the general correctness/style/readability of your classmate's code, as well as
  2. its algorithmic integrity:
     - are all examples in the dataset seen during training?
     - is high-quality stochasticity employed?