CSCI 362: Machine Learning **Assignment 6**
Fall 2023 **training a nonlinear model (part 1)**

Please submit a single document for this assignment on Canvas by the beginning of class on **Wednesday, Oct. 25th**.

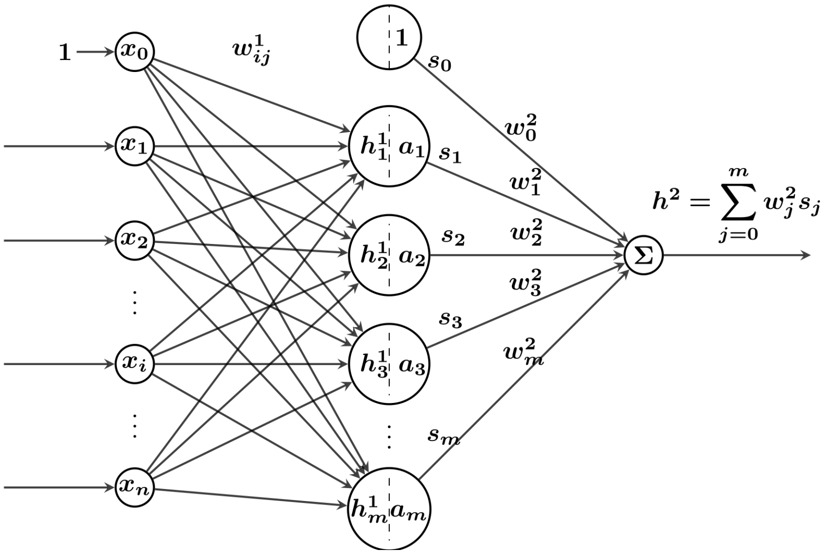
In this assignment we will train a nonlinear model to fit the housing data.

- Copy your final program for assignment 5 to a new file called say `nonlinear.py`. We are going to modify the model.

  Your model class is likely currently called `LinearModel` or similar. Change the name because below we are not going to restrict our models to linear ones.

- Per our discussion in class, add a single, intermediate, layer with say 10 nodes.

  The schematic below represents your model if $n = 13$ and $m = 10$.



  In the `forward` method of your model class, make sure to add nonlinearities (`relu` or otherwise) on the output side of the intermediate nodes — otherwise your model is equivalent to a linear model.

- Toward the end of your version of `nonlinear.py`, you should find the following lines:

```
print("total number of examples:", num_examples, end='')
print("; batch size:", batch_size)
print("learning rate:", learning_rate)
```

  Add another line that prints the momentum.

- Since we are still using a relatively small dataset, you could train your model using (full) batch gradient descent.

  Don't use full batch. Instead, **add some stochasticity**; that is, find learning parameters that lead to high quality convergence using mini-batch gradient descent with a mini-batch size of say 20 or 30.

  Note: when employing stochasticity, you likely want some momentum.

- With nonlinearity, you should fairly easily achieve *explained variance* higher than 0.86 on training data. Once you find such learning parameters, include a **readable screenshot** of a run of your program. Make sure that the **learning rate, momentum, and batchsize** are displayed as well as the **proportion of the variance** captured by your now non-linear hyper-surface.

  Please also **upload your entire program**.