Please submit a single document for this assignment on Canvas by the beginning of class on **Friday, Oct. 4th**.

In this assignment you will train a nonlinear model to fit the housing data while barring again *over-fitting*.

Linear models have no chance of over-fitting. Non-linear models are in danger of capturing the training data too closely. And that often leads to poor performance on data unseen during training — thereby reducing the predictive power of the trained model.

Said differently, non-linear models can "learn" the training data too well, in which case the trained model might not *generalize* well to unseen data. The model learns the pattern in the training data and perhaps not the relevant pattern in the population of all such data.

To gauge whether our model might be over-fitting, we partition our original dataset into train and test sets. Often the split is 80-20%. We train on the 80% and then *validate* our trained model by testing its performance on the 20% of data not shown to the model during training.

One way to bar against over-fitting is to use so-called *early stopping*. This is a simple technique in which we use the 20% of data not shown to the model during training to *validate* the model *during training*; that is, after each epoch we run the saved-out 20% through model and note down how well the model performs on unseen data.

Note: in the context of the last paragraph, the saved-out 20% is referred to as *validation* data in modern ML parlance.

Since we are observing performance on validation data during training, we can gauge when the model starts over-fitting. A intuitive way to do this is to use a graph.

For the following, we will use DUlib. If you haven't already issue this command at your command line:

```
pip3 install git+https://github.com/sj-simmons/DUlib.git@v0.9.5  --user
```

- In Assignment 7, you were hopefully using the `train` function in DUlib. Assuming the you have `matplotlib` installed, you can easily graph in real-time by setting the keyword argument `graph = 1` when calling `train`.
- Retrain your model from Assignment 7. Be sure to stop training before your model starts over-fitting. Include relevant screenshots in your posted solution file.