From talking to Patrick it sounds like EM’s DM\_Neural represents the best single ANN I can hope to quickly build, a paragon if you will, and that RF is specifically an ensemble of “weak” models which implies that as I iterate over NN parameters I should keep each NN built over that design space very small and short. So I will let DM\_Neural run over a weekend on the training data with a “master” holdout defined. That is, randomly picking some percentage which will be known and unchanged the whole time I am working on this project so that any model **designs** can be compared to each other on this holdout, whereas an individual model that got a different training set could not. For instance when building a hive mind I might have a vanilla RBF 2x2 etc etc that, as part of a hive mind, will have bagging, but I could apply all the same design settings to the master version to compare it (expecting much lower accuracy) to the dm\_neural. While I am trying to keep things simple, I may do 2 of these master holdouts to account for artifice (say one that is 15% of the target-known data completely randomly and another that is 10% sampled in some way by target and input).

This means I plan to create code to iterate over all design parameters for NNs, then when I have run these on master holdouts to see how each does and any higher-level comparisons (to strong NNs or other model types altogether) to go back and inject code to do bagging and multiple imputation. I’d like this code to be production quality, which should include the ability to turn things off and just iterate over the design parameters pointing to any train/hold data.

Here’s an initial storyboard:

It seems important to start out this way, there’s no cost to it, only my own impatience is incurred. Too often I have gotten lost in operational details, this should be designed and last me a year at least. If “post-processing” is done correctly I would be able to mix-and-match at least within model type, to see which ensemble does best and whether there is rhyme or reason to that.

I am not a data miner, I am a forecaster. My main goal for this project is to get experience running SVMs. On the way I will work on MI, NNs up to Hive Mind and ensembling but an absolute must is to do SVMs.

Starting on it to avoid the nenana2013 effect, a good middle-way as far as honest partitioning is to do exploration on 90%, plan on later repartitioning a 30% test set likely staged, and a 15% sampled, likely only by survival, at most 2 other binary variables to sample by.