Combining networks

• When the amount of training data is limited, we need to avoid overfitting.
  – Averaging the predictions of many different networks is a good way to do this.
  – It works best if the networks are as different as possible.

• If the data is really a mixture of several different “regimes” it is helpful to identify these regimes and use a separate, simple model for each regime.
  – We want to use the desired outputs to help cluster cases into regimes. Just clustering the inputs is not as efficient.
Combining networks reduces variance

- We want to compare two expected squared errors
  - Method 1: Pick one of the predictors at random
  - Method 2: Use the average of the predictors, $\bar{y}$

$$
\bar{y} = < y_i >_i = \frac{1}{N} \sum_{i=1}^{N} y_i
$$

$$
<(d - y_i)^2>_i = <((d - \bar{y}) - (y_i - \bar{y}))^2>_i
= <(d - \bar{y})^2 + (y_i - \bar{y})^2 - 2(d - \bar{y})(y_i - \bar{y})>_i
= <(d - \bar{y})^2>_i + <(y_i - \bar{y})^2>_i \cdots
- 2(d - \bar{y}) <(y_i - \bar{y})>_i
$$

This term vanishes
The predictors that are further than average from $d$ make bigger than average squared errors.

The predictors that are nearer than average to $d$ make smaller then average squared errors.

The first effect dominates because squares work like that.

$$(a + \varepsilon)^2 + (a - \varepsilon)^2 = 2a^2 + 2\varepsilon^2$$

Don’t try averaging if you want to synchronize a bunch of clocks!
How the combined predictor compares with the individual predictors

• On any one test case, some individual predictors will be better than the combined predictor.
  – But different individuals will be better on different cases.
• If the individual predictors disagree a lot, the combined predictor is typically better than all of the individual predictors when we average over test cases.
  – So how do we make the individual predictors disagree? (without making them much worse individually).
Ways to make predictors differ

• Rely on the learning algorithm getting stuck in a different local optimum on each run.
  – A dubious hack unworthy of a true computer scientist (but definitely worth a try).

• Use lots of different kinds of models:
  – Different architectures
  – Different learning algorithms.
Making predictors differ by using different training data for each model

**Bagging**

Resample (with replacement) from the training set: a,b,c,d,e -> a c c d d

**Boosting**

Fit models one at a time. Re-weight each training case by how badly it is predicted by the models already fitted.

- This makes efficient use of computer time because it does not bother to “back-fit” models that were fitted earlier.
Boosting slides

• Boosting was invented and developed by Freund and Shapire.
• They made nice slides.
• Ignore slides 5 & 6. Stop at slide 14.