

A QSAR Model using a Probabilistic Neural Network Ensemble

Ed Ramsden

Lattice Semiconductor Corp

SIAM SDM'11

April 30, 2011, Phoenix Arizona

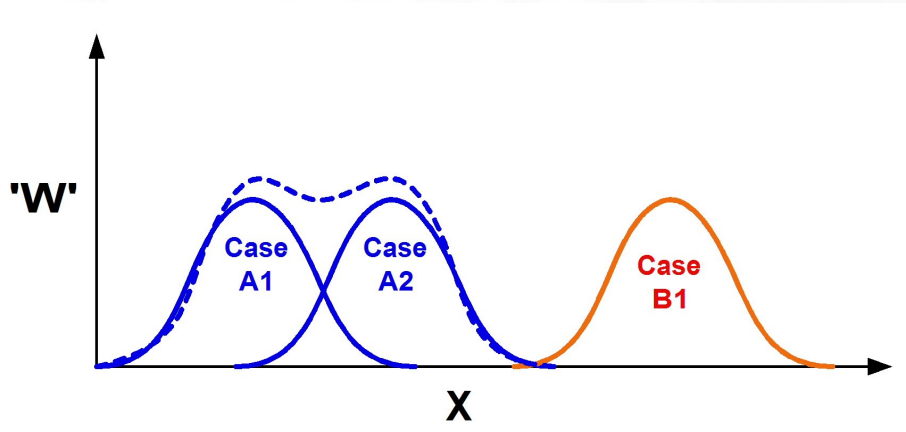
Problem & Data Characteristics

- Goal was to build a model to predict a property Y of a chemical compound based on structural data properties X_i .
- Natures of both predicted property and structural data were not revealed at beginning.
- Training library of 837 compounds with known property Y 's. (674 initially released)
- 242 total X_i 's (structural properties) for each example – high dimensionality, esp. in relation to # of samples!
- Metric of Model Quality is Balanced Youden:

Sensitivity = $TP/(TP+FN)$, Specificity = $TN/(TN+FP)$

Youden_B = $\min(\text{Sensitivity}, \text{Specificity})$

PNN Concept

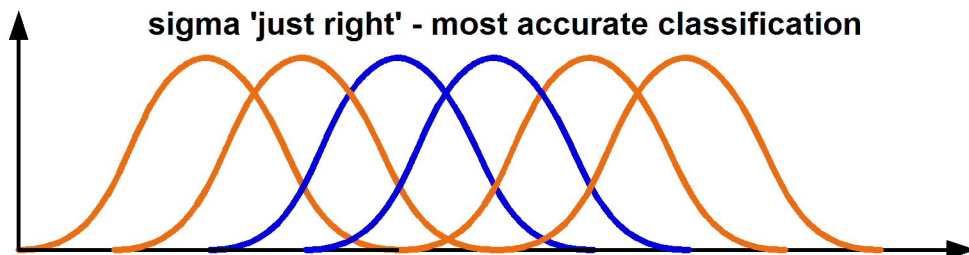
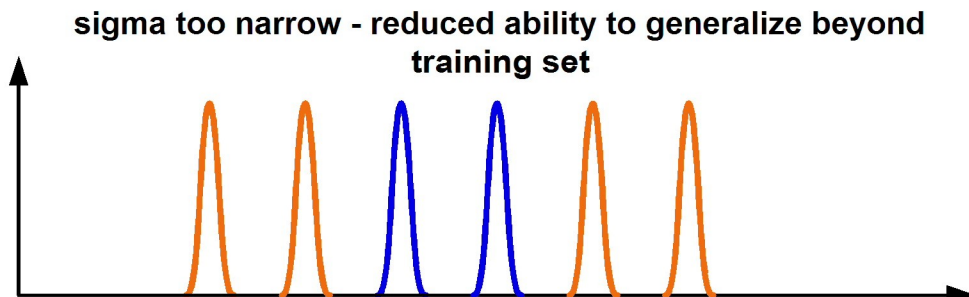
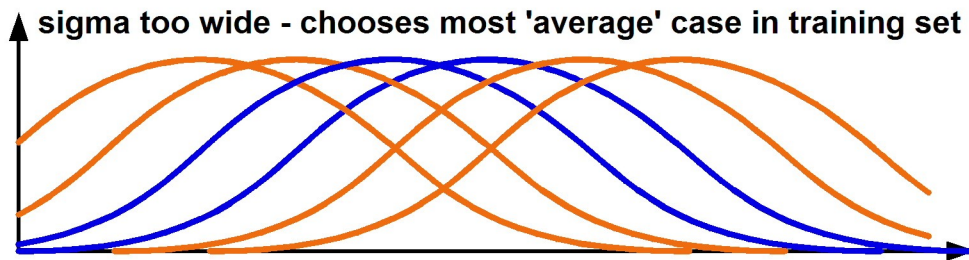


$$W_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \exp\left(-\frac{\|\hat{x} - x_{ci}\|}{\sigma^2}\right)$$

$$\|\hat{x} - x_{ci}\| = \sum_j (\hat{x}_j - x_{cij})^2$$

- Introduced by Donald Specht in 1990
- Instance-based modeling
- Gaussian 'probability distribution' (W) of an unknown case being of same class as known reference.
- Extensible to multi-dimension 'X' vectors

Selection of Kernel Radius σ

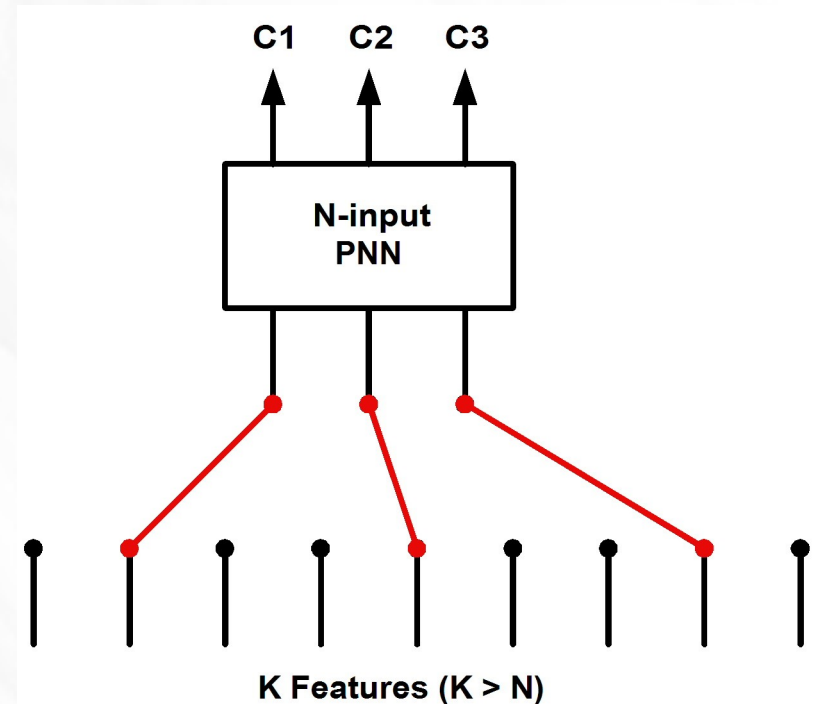


- **Too-wide kernel radius reduces selectivity, in limiting case results in 'mean-of-data' model**
- **Too-narrow kernel radius increases selectivity, reduces ability to generalize**

Curse of Dimensionality

Because 'example-based', as dimensionality grows, # of training examples needed to adequately fill space grows exponentially

- High-dimension data difficult to model
- One solution is to select useful subsets of input features (feature selection)
- 'Random Hill Climb' based on training metric seems to work OK (if slowly) for feature selection



Evaluating Model Quality

Model Evaluation is Key to...

- Finding an appropriate input feature set
- Adjusting theta
- Estimating quality of model (without using the final test data!)

Leave-one-out evaluation procedure

For all data cases D_i ...

- (a) Remove D_i from model
- (b) Evaluate model to predict D_i
- (c) Evaluate overall accuracy

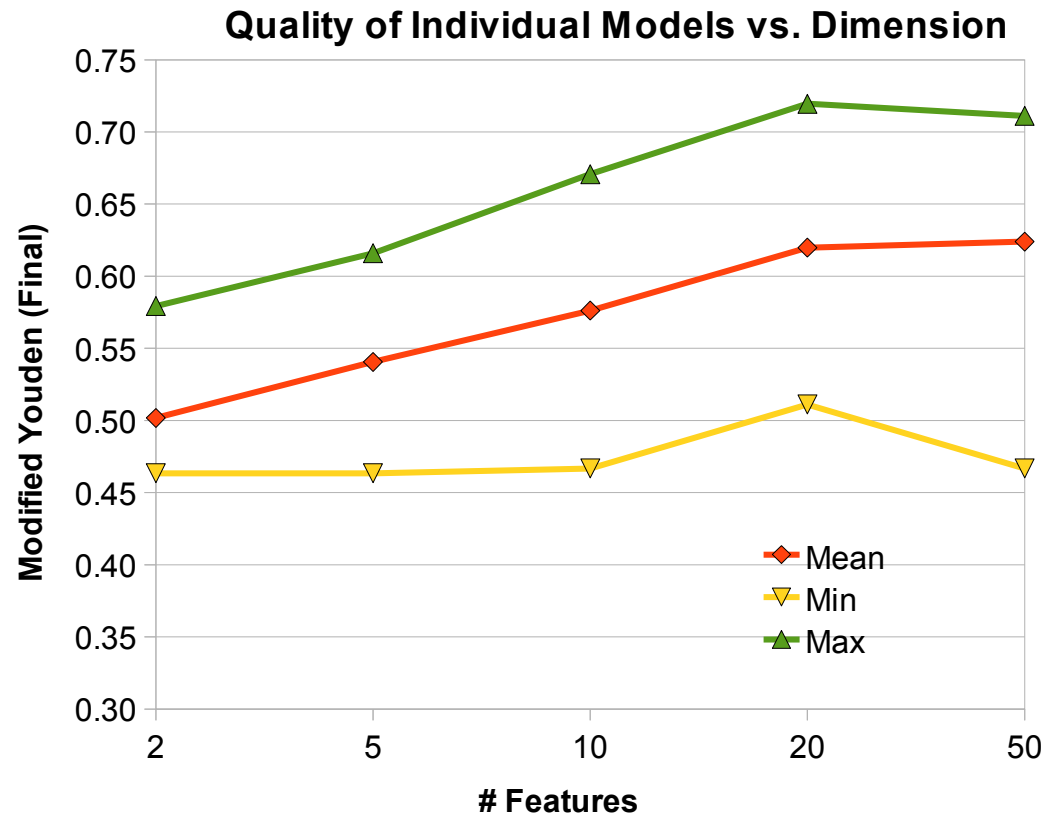
Adjustment of σ and feature set

Iterate N times:

- (1) If $\text{RND} > 0.5$ then $\sigma' := \sigma * M$ else $\sigma' := \sigma / M$
- (2) Evaluate model using σ'
- (3) If result equal or better, $\sigma := \sigma'$
- (4) Randomly select a currently used feature ' F_{current} '
- (5) Replace with a different randomly selected but currently unused feature ' F_{new} '
- (6) Evaluate model
- (7) If result better or equal, keep F_{new} in model, else replace F_{current}

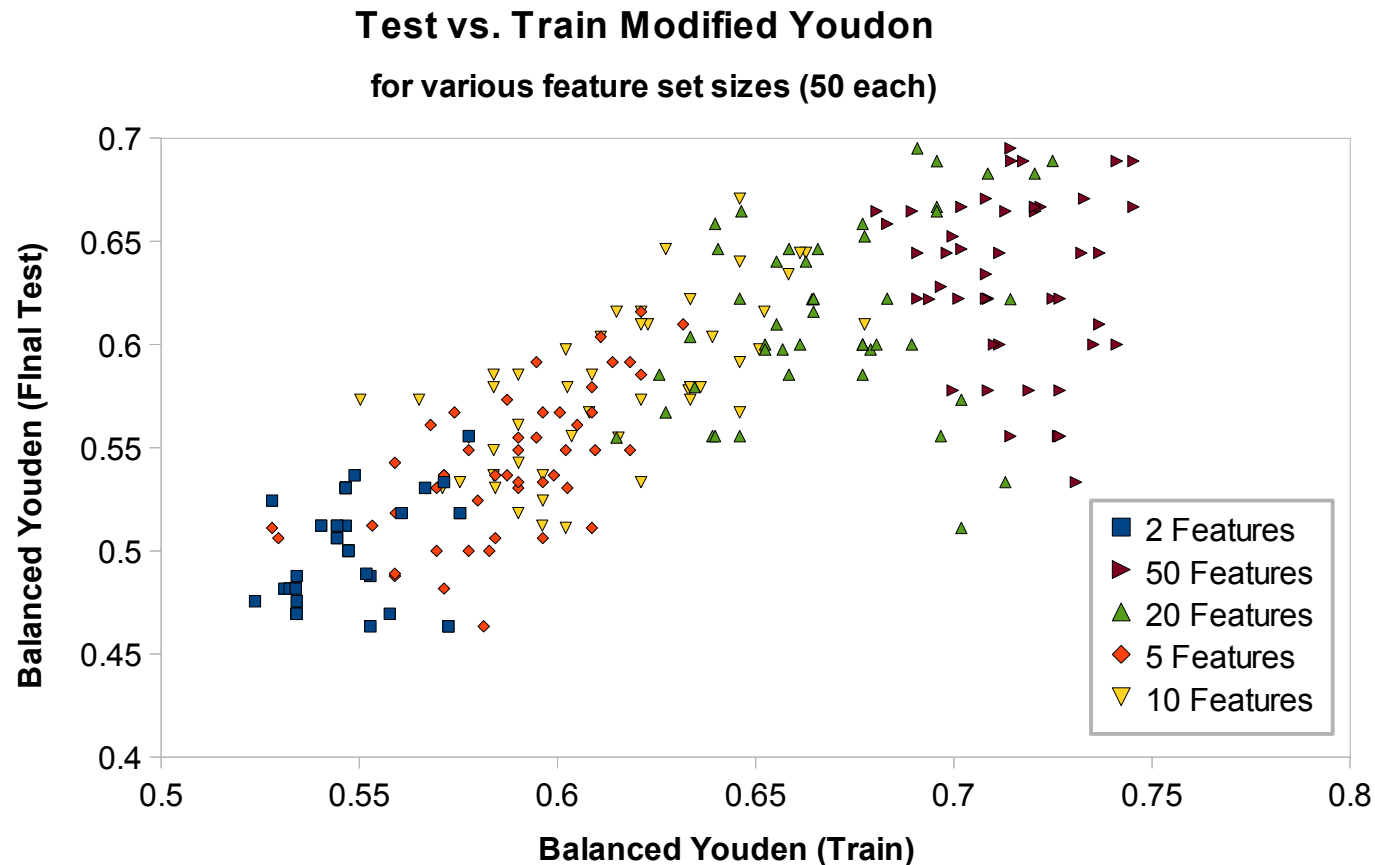
Algorithm does not require calculating gradients, merely evaluating better or worse overall performance metric, and will work with a variety of metrics (e.g. Modified Youden)

Aggregate Model Performance



- 100 Models generated for 2, 5, 10, 20 & 50 features
- Metric is predictive performance on Final Test Data

Individual Model Performances



- **As # of features increases, performance improves up to a point**
- **Too many features increases the variance of the models (overtraining)**

Ensembling Models

**Idea of *Ensembling* models is that a combination of models looking at the data in different ways will provide much better results than any single model.
(Wisdom of Crowds)**

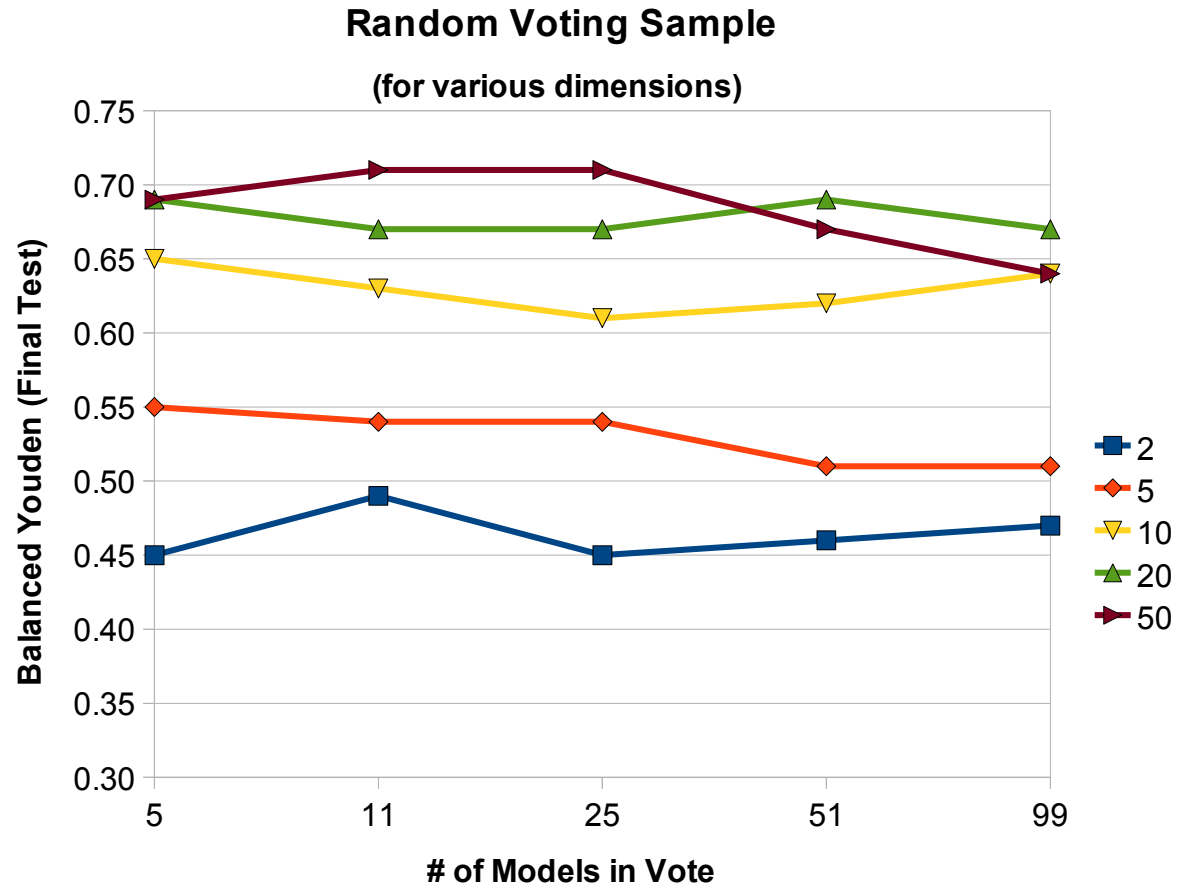
How to Select Models?

- Pick some at random
- Pick the 'best-of-class from larger set (e.g. by Training Metric)

How to Combine Models?

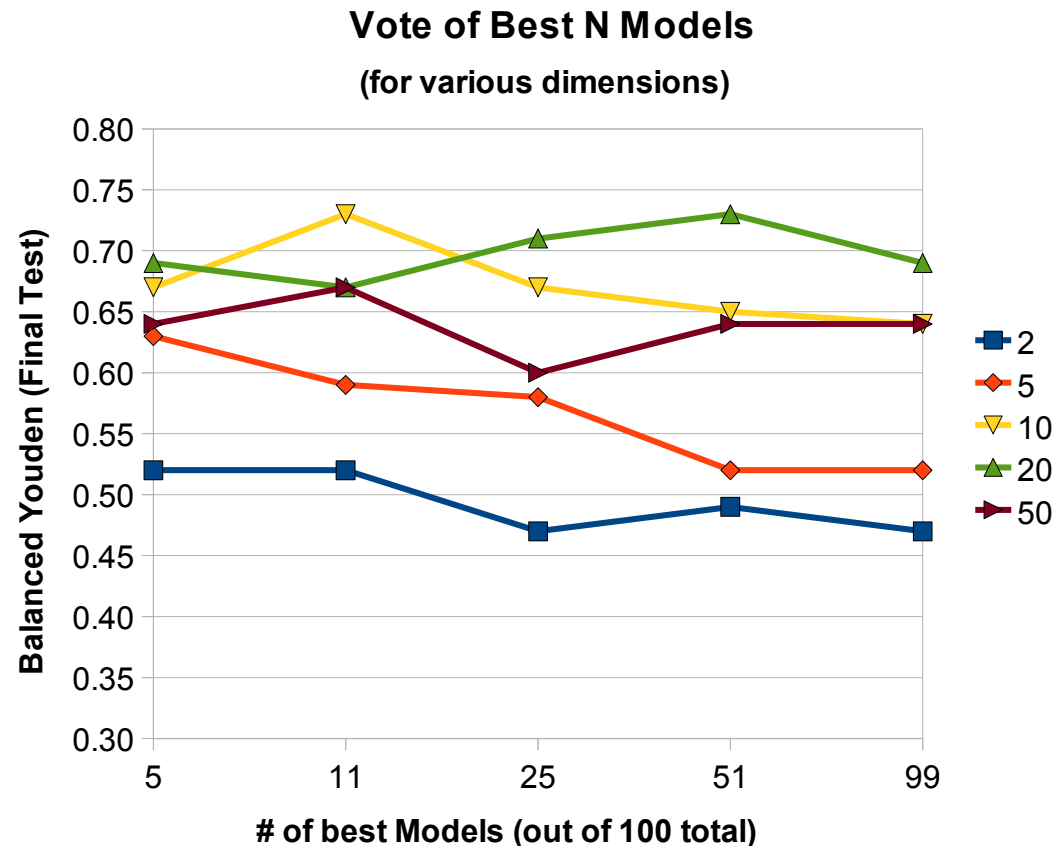
- Simple Majority Vote used for this problem.
(Make sure you have an odd # to break ties!)

Performance of Some Random Ensembles



Some variation by voting sample size, but this likely from 'the luck of the draw'

Performance of 'Best-of-Class' Ensembles



- **Better Performance than 'Random Sample'**
- **Unexpected behavior for 'elite' sample size 20**
 - 20 feature ensemble peaks with top 50% while
 - 5 feature ensemble peaks with top 5%

Competition Model

- **35 Input Features for each sub-PNN**
- **1000 Iterations to Tune Individual Models**
- **Pool of 135 Models Generated**
- **Best 25 Selected for Voting Group (by training Youden)**
- **Performance:**
 - Train Youden: 0.794 (last 163 cases of train set)
 - Final Test Youden: 0.689

Further Work/Room for Improvement...

- **Computational Efficiency!**
Competition model took ~8 hours CPU to create
- **Alternate Approaches to Dimensionality Problem**
- **Handling Mixed Data**
Continuous/Ordinal/Nominal
- **Handling Incomplete Data**
e.g. $X = \{1, 2.5, ???, 3, 5\}$
- **Handling Uncertainty in 'Y' values**
sample weighting schemes like in linear regression
- **Better automation**
Less tweaking to build 'good' models

**Thanks to Simulations Plus and
Robert Fraczekiewicz for organizing this
competition!**

Questions?

References:

D.F. Specht, "Probabilistic Neural Networks", *Neural Networks*, vol.3, pp.109-118, 1990.