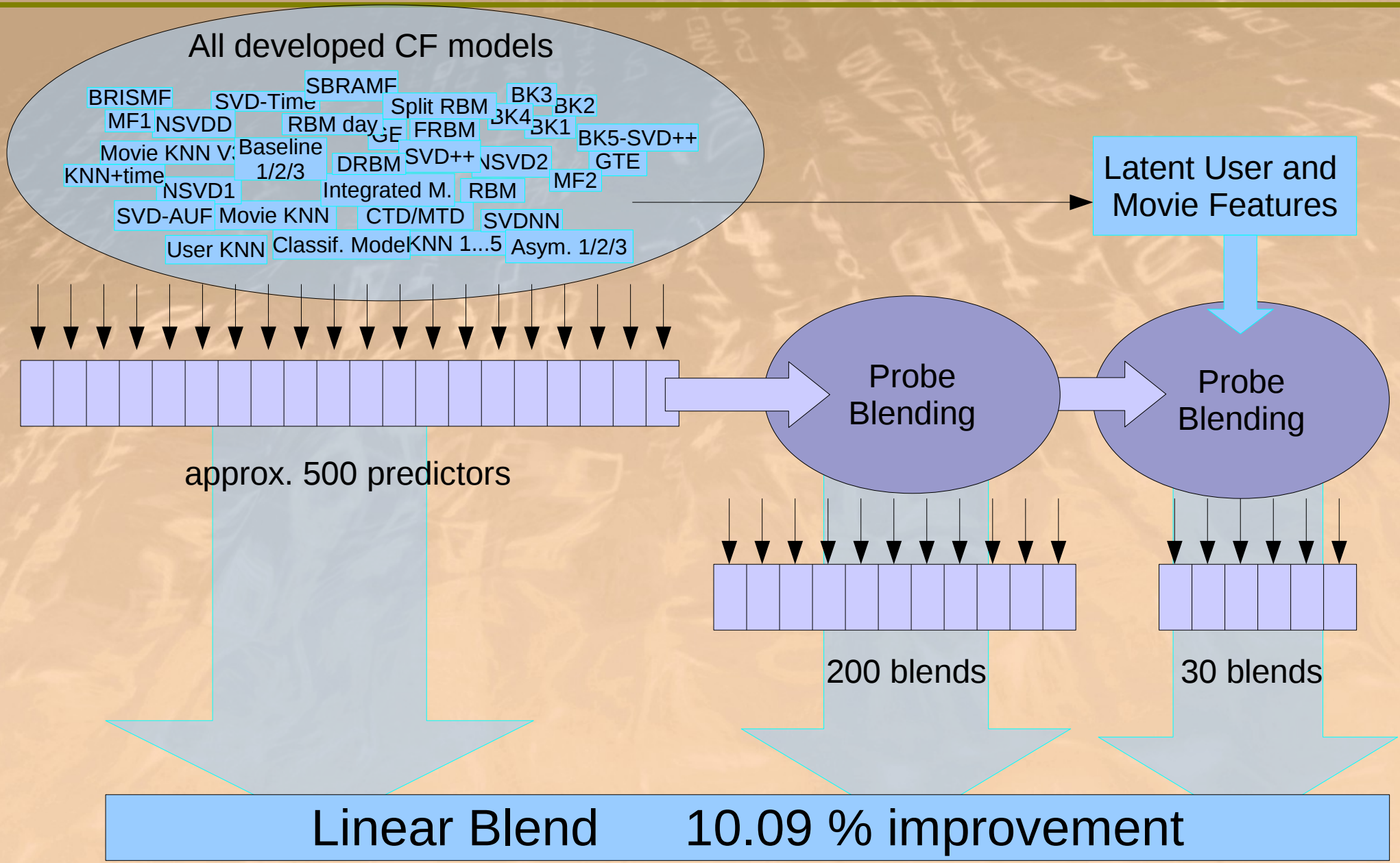
Blending Techniques

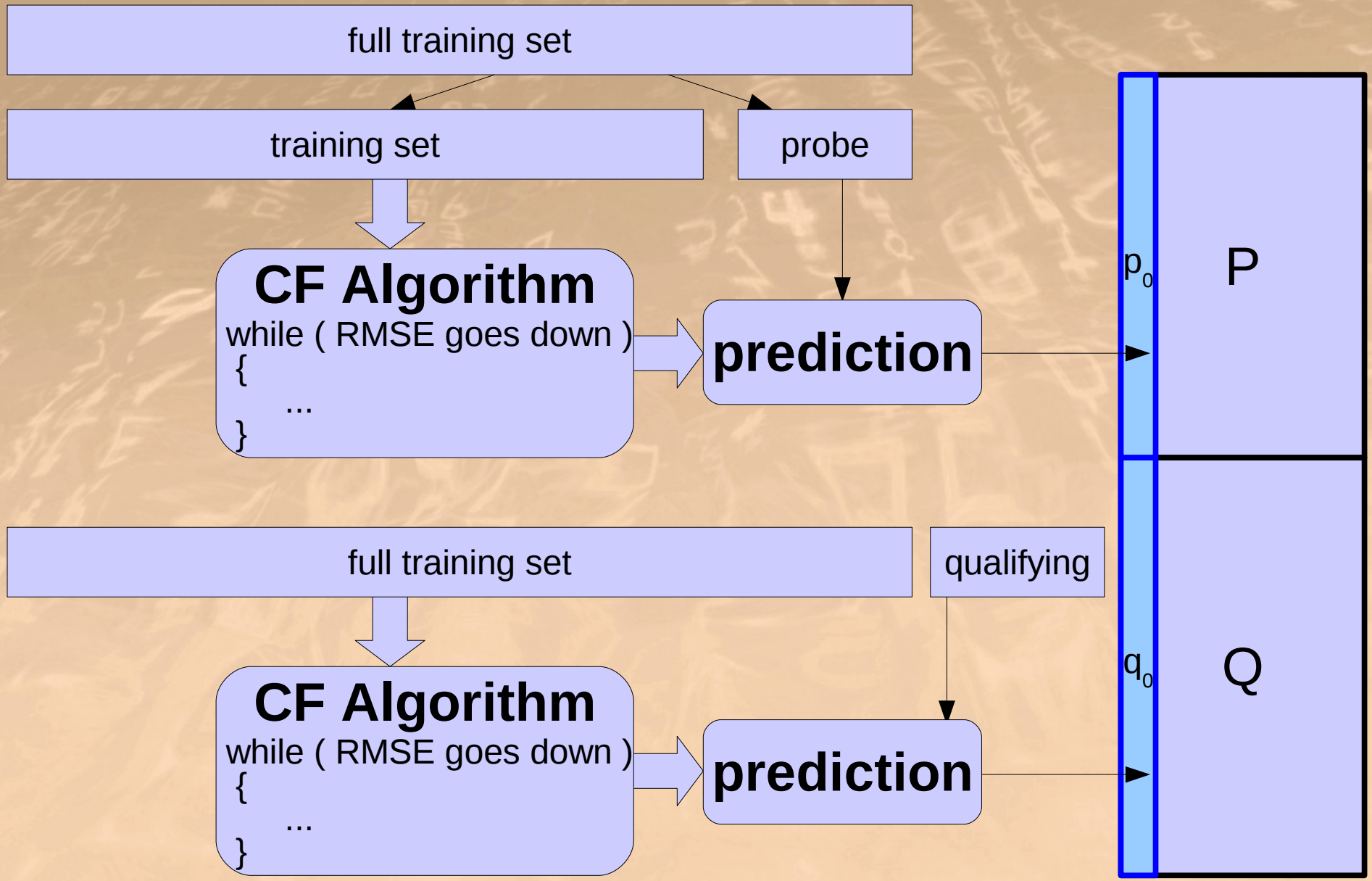
- Methods
  - Linear Combination
  - Neural Networks
  - Tree Blending
  - Quiz Blending
- Compare the Methods

The big picture

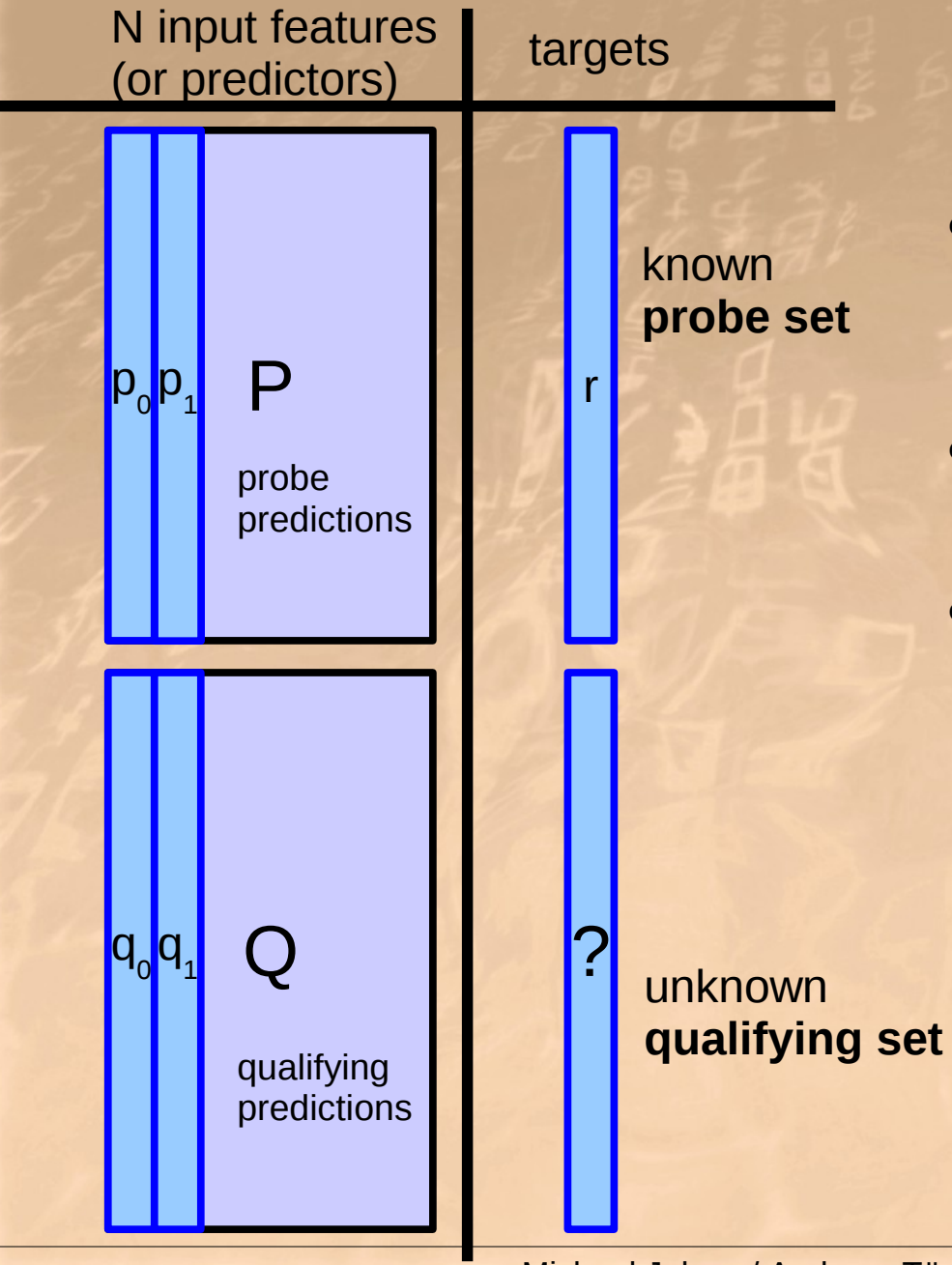
Solution of BellKor's Pragmatic Chaos



The training process – predictor generation

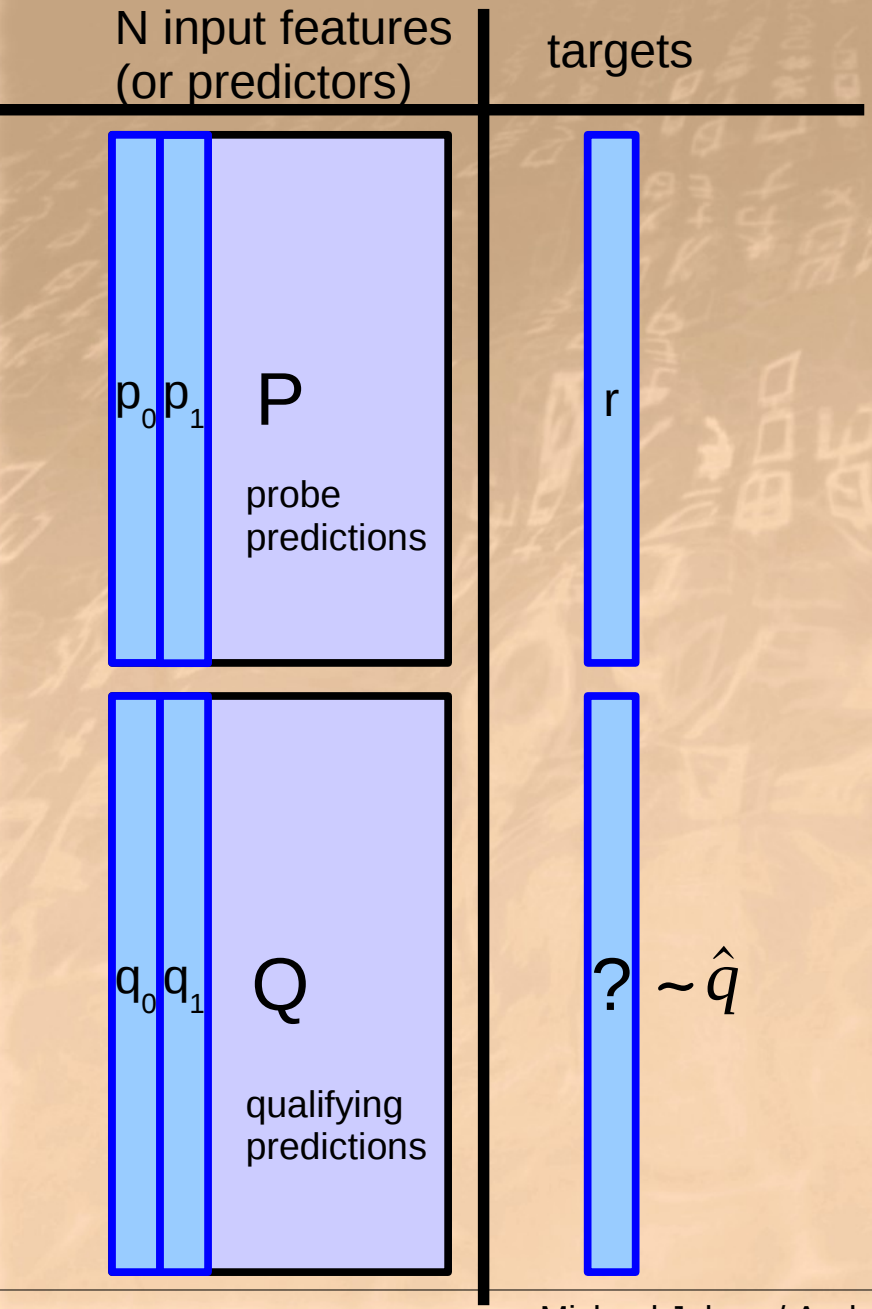


We now have a supervised learning problem



- Large number of training set
  → 1.4M samples
- 100...500 predictors/features
- Slightly different distribution of probe and qualifying set
  → probe trained on **99M**
  → qualifying trained on **100.4M**
  (approx. 0.0070 better in RMSE)

A simple linear blender



1. Estimate blending weights

$$x = (P^T P)^{-1} P^T r$$

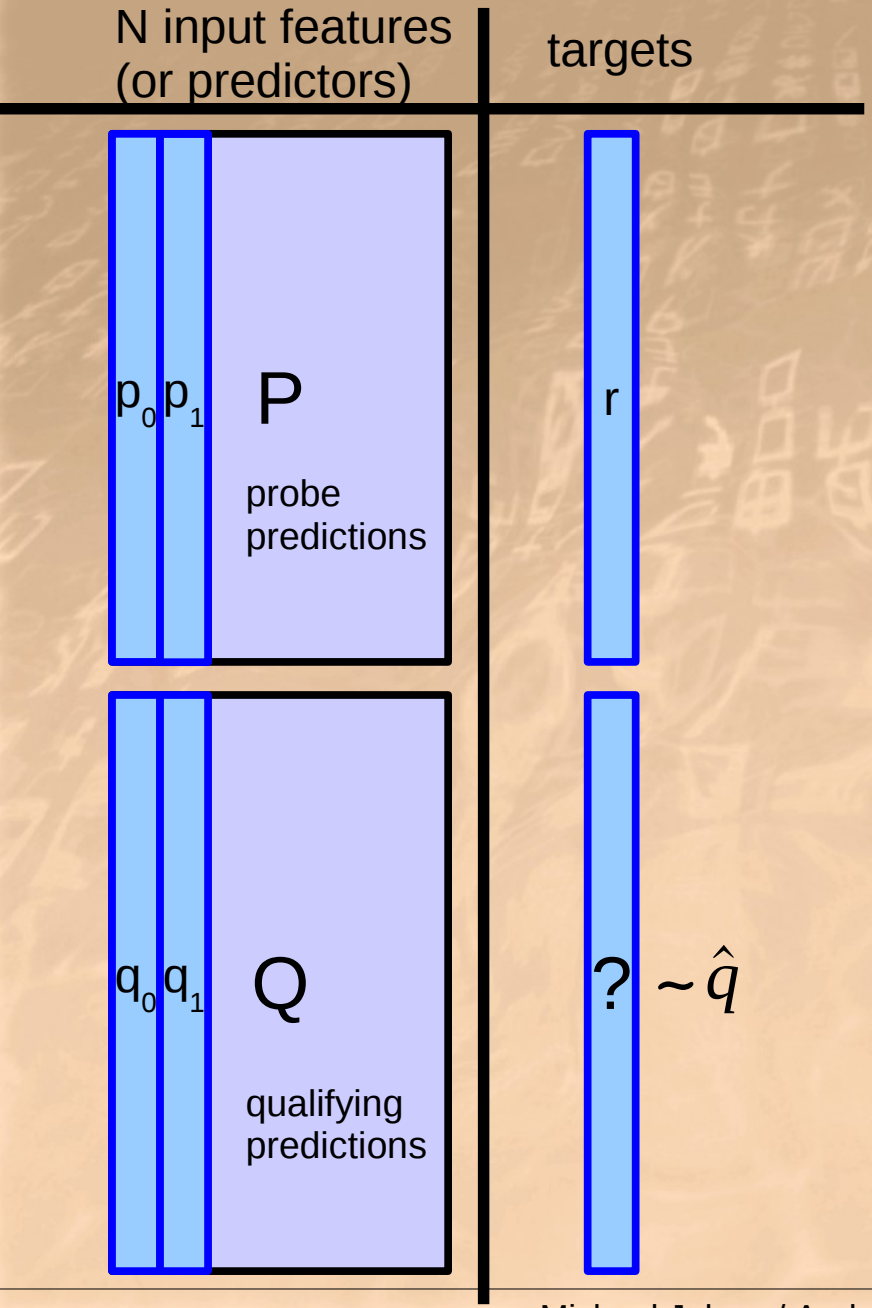
2. Predict the unknowns

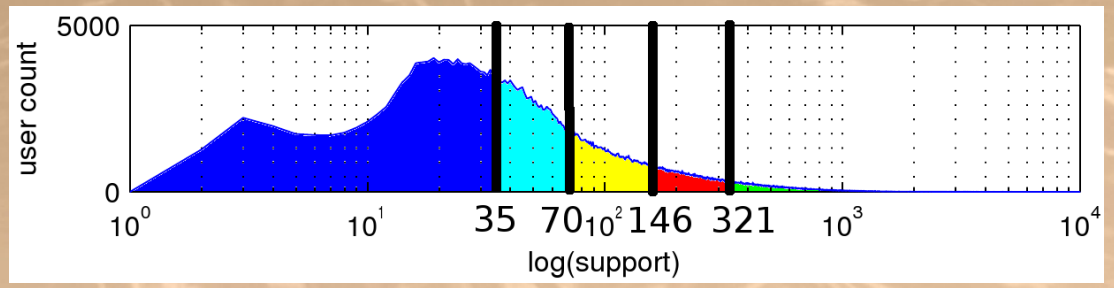
$$\hat{q} = Q\,x$$

(a single prediction)

$$\hat{q}_i = \sum_{n=1}^{N} q_{ni}\, x_n$$

Binned linear blending





1. Divide probe set into bins, e.g.
   - <35 votes: **bin0**
   - 36..70 votes: **bin1**
   - 71...146 votes: **bin2**
   - 147...321 votes: **bin3**
   - >321 votes: **bin4**

2. Estimate blending weights per bin

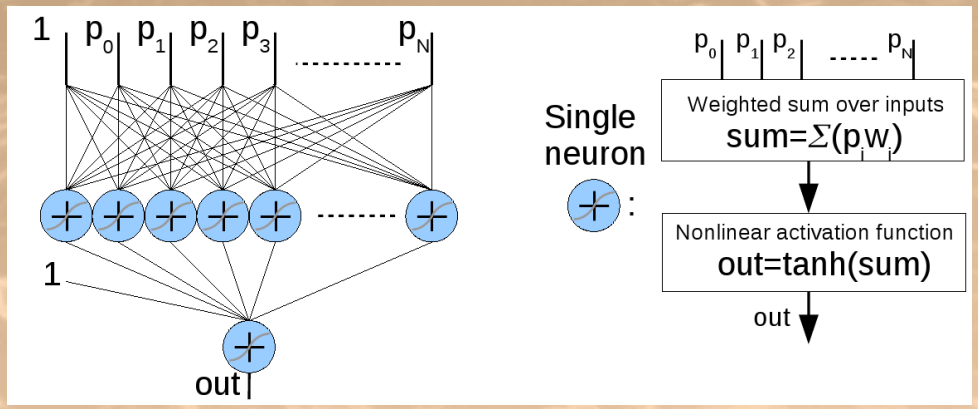
$$x^{(b)} = \left(P_b^T P_b\right)^{-1} P_b^T r_b$$

3. Predict the unknowns

$$\hat{q}_i = \sum_{n=1}^{N} q_{ni} x_n^{(b)}$$

Best result:
PB-100: **RMSE 0.8593**

Neural Network blending - NNBlend

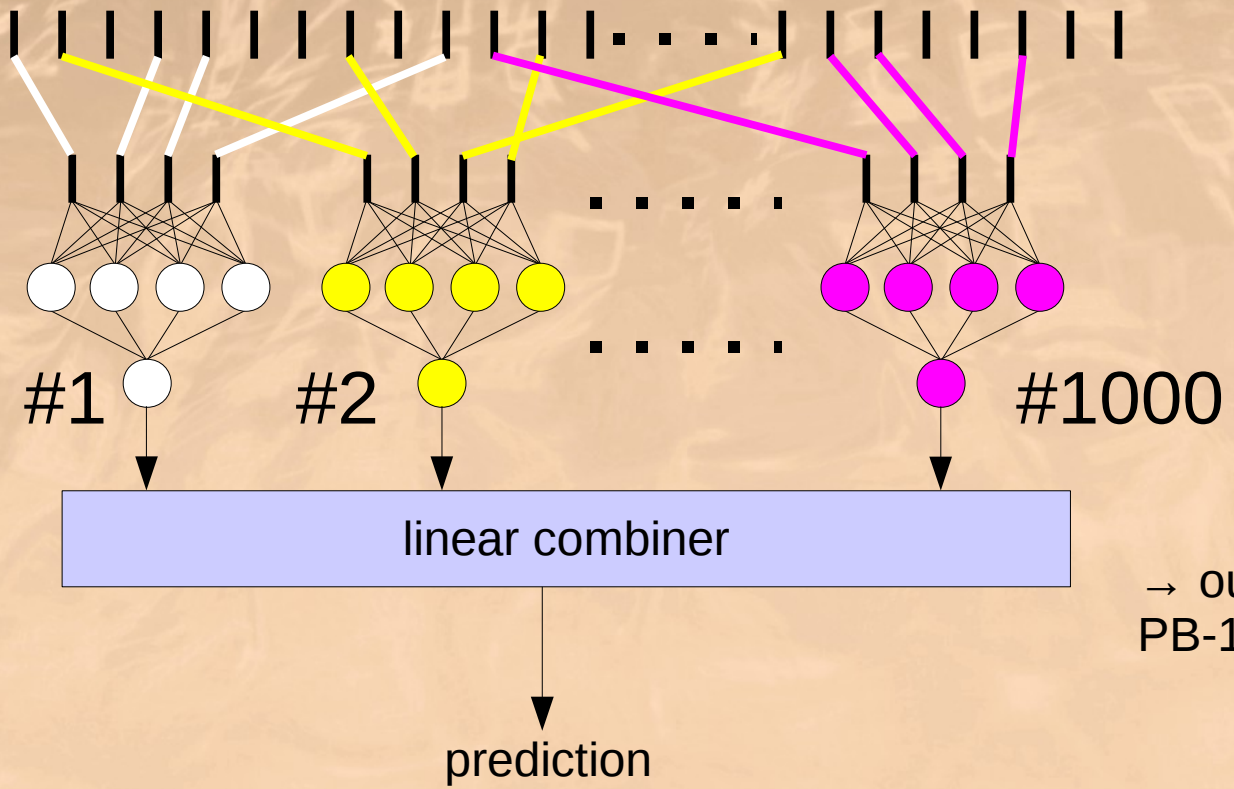


- 1 hidden layer
- 10...20 neurons
- Inputs are normalized
  - centering: $\mu = 0$, standard deviation: $\sigma = 1$
- Pure stochastic gradient descent
  - no batch/mini-batch update
- Decrease initial learning rate
  - Linear from 0.0005 to 0.0001. Per epoch subtract: 3e-7
- Outputs are transformed by: $\hat{out} = out \cdot 3.6 + 3$

NNBlend: 75 predictors
(approx. 1200 weights)

Ensemble NNBlend

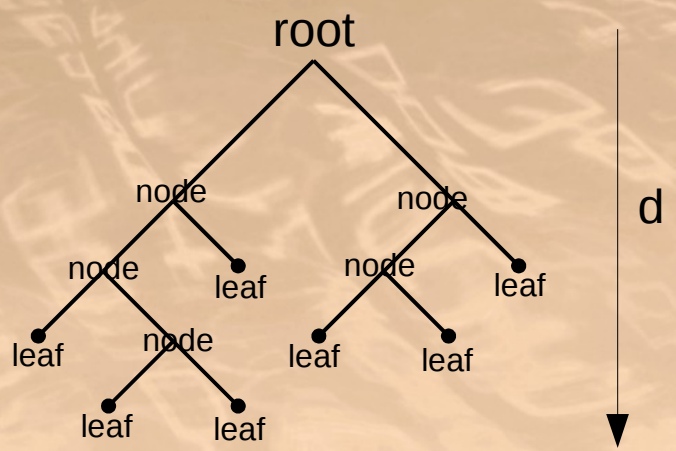
- Train many small NN's (>1000) on a random subset
  - Per net: 20..40 weights
- Combine them linearly



→ our best probe blend:
PB-101: **RMSE=0.8584**

Blending with Decision Trees

- **Basics:**
  - A tree can not model a smooth function
  - Per node: (p[k] > thresh ?)
  - Per leaf: constant value
  - No scaling of the inputs needed

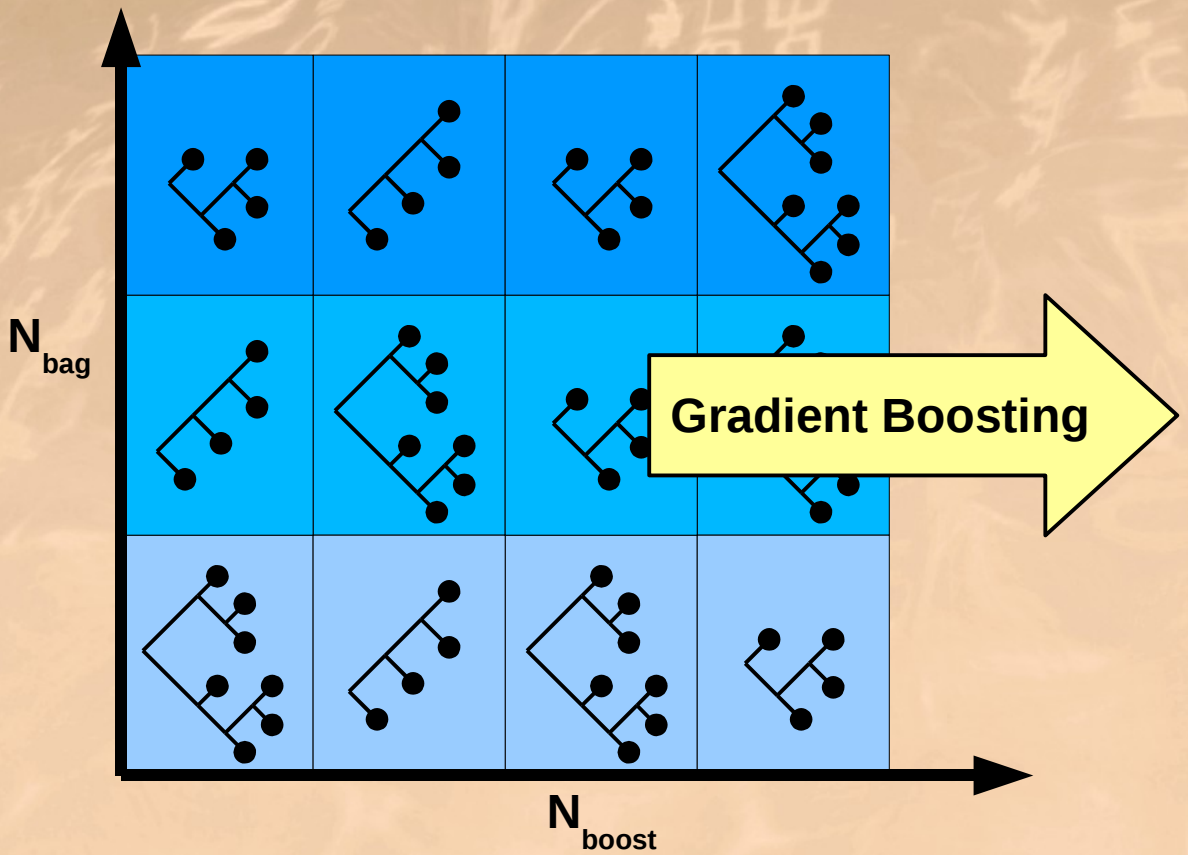


- **Single tree**:
  - Use a random subspace idea
  - We take the split that reduces the RMSE best
  - The depth is limited, max. depth of d=10 works well

Bagged Gradient Boosted Decision Trees - BGBDT

- **Bagging**: model averaging ($\rightarrow$ reduce variance)
- **Gradient Boosting**: Partial reduce of the residual error

$$target_{new} = target_{old} - \eta \cdot prediction$$



Good Parameters:

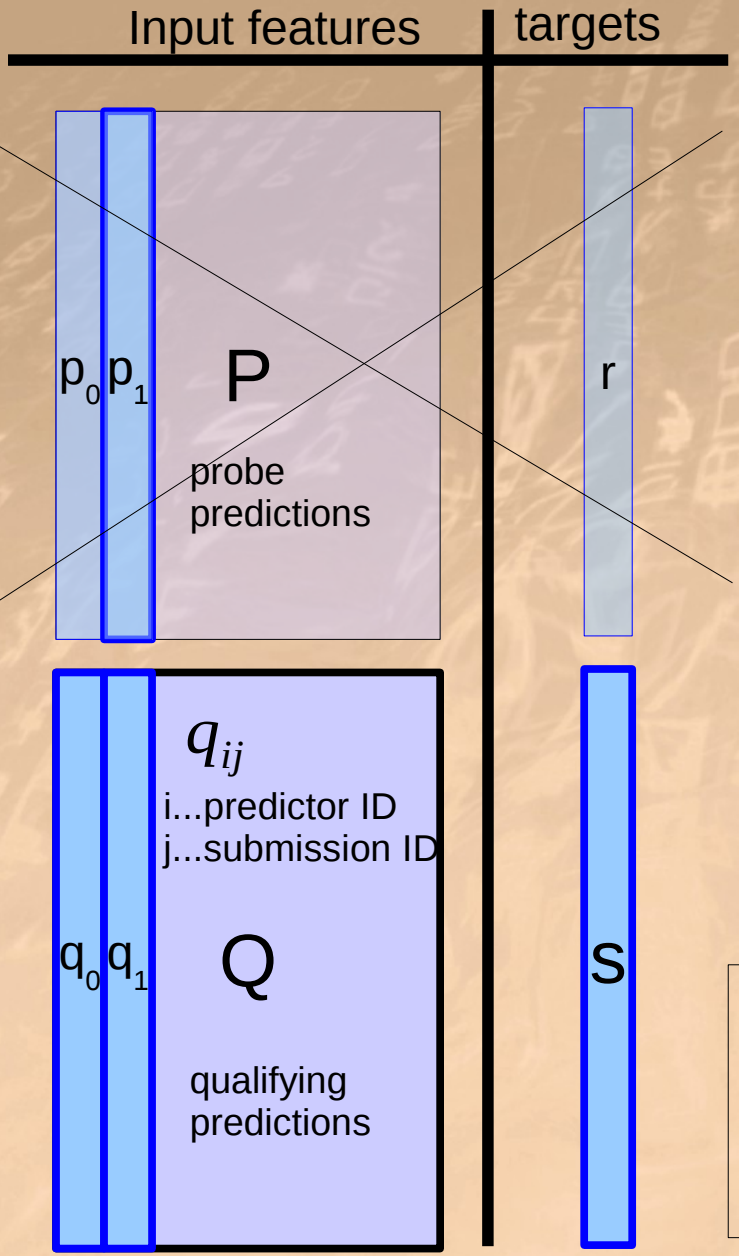
η=0.1

N_{bag}=32

N_{boost}=120

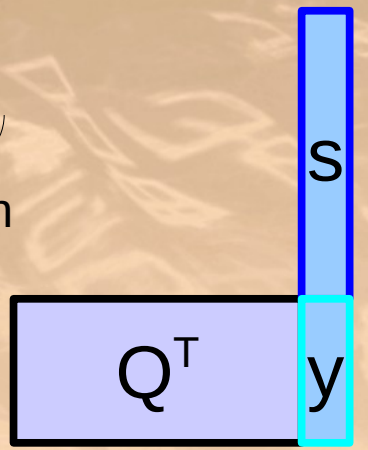
Best result:
PB-164: **RMSE 0.8592**

Quiz Blending



optimal solution

$$x = \underbrace{(Q^T Q)^{-1}}_{\text{known}} \underbrace{Q^T s}_{\text{unknown}}$$



$$y_i = \sum_j q_{ij} s_j$$

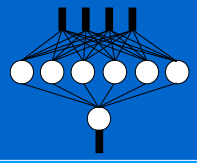
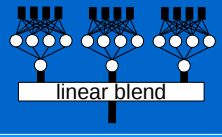
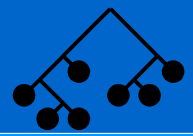
$$y_i = \frac{1}{2}\left(\underbrace{\sum_j q_{ij}^2}_{\text{known}} + \underbrace{\sum_j s_j^2}_{\text{variance quiz set}} - \underbrace{\sum_j (q_{ij} - s_j)^2}_{\text{individual RMSEs}}\right)$$

Quiz Blending is impractical in real environments
→ quiz RMSEs are unknown

binomial formula

$$(a-b)^2 = a^2 - 2ab + b^2$$
$$\frac{1}{2}(a^2 + b^2 - (a-b)^2) = ab$$

Compare the Methods
(probe blending)

	Linear Regression $\hat{q} = Q x$	Binned Linear Regression $\hat{q}_i = \sum_{n=1}^{N} q_{n1} x_{bn}$	Neural Network 	Neural Network Ensemble 	BGBDT Tree Blending 
Accuracy (low RMSE=5 stars)	★★★★★	★★★★★	★★★★★	★★★★★	★★★★★
Speed (fast=5 stars)	★★★★★	★★★★★	★★★★★	★★★★★	★★★★★
Implementation complexity (easy=5 stars)	★★★★★	★★★★★	★★★★★	★★★★★	★★★★★