

Abstract

We analyze the application of ensemble learning to recommender systems on the Netflix Prize dataset. For our analysis we use a set of diverse state-of-the-art collaborative filtering (CF) algorithms, which include: SVD, Neighborhood Based Approaches, Restricted Boltzmann Machine, Asymmetric Factor Model and Global Effects. We show that linearly combining (blending) a set of CF algorithms increases the accuracy and outperforms any single CF algorithm. Furthermore, we show how to use ensemble methods for blending predictors in order to outperform a single blending algorithm. The dataset and the source code for the ensemble blending are available online <http://elf-project.sourceforge.net>.

Collaborative Filtering Algorithms

algorithm	RMSE (Netflix)	training time	prediction time	memory
KNNitem	0.92	$O(U \cdot M^2)$	$O(M \lg(M))$	$O(M^2)$
KNNuser	0.93	$O(M \cdot U^2)$	$O(U \lg(U))$	$O(U^2)$
SVD	0.90	$O(\mathcal{L})$	$O(1)$	$O(M + U)$
AFM	0.92	$O(\mathcal{L})$	$O(1)$	$O(M)$
SVD _e	0.88	$O(\mathcal{L})$	$O(1)$	$O(M + U)$
RBM	0.90	$O(\mathcal{L})$	$O(1)$	$O(M)$
GE	0.95	$O(\mathcal{L})$	$O(1)$	$O(M + U)$
Blend	< 0.87			

FIGURE 1: The prediction time is defined as the asymptotic time needed to generate a single prediction \hat{r}_{ui} . M is the total number of items, U is the total number of users and $|\mathcal{L}|$ the total number of ratings in the training set.

SVD (matrix factorization)

$$\hat{r}_{ui} = \mathbf{p}_i^T \mathbf{q}_u$$

item feature $\begin{bmatrix} q_0 \\ q_1 \\ q_2 \end{bmatrix}$ * user feature $\begin{bmatrix} p_0 \\ p_1 \\ p_3 \end{bmatrix}$

Training is done with stochastic gradient descent.

KNN item-item

$$\hat{r}_{ui} = \frac{\sum_{j \in R(u,i)} c_{ij} r_{uj}}{\sum_{j \in R(u,i)} |c_{ij}|}$$

For one prediction, the KNN selects the k best correlated items $R(u, i)$ to item i . This can take up to $O(M)$ operations. The sorting of this list takes $O(M \cdot \log(M))$ time.

AFM (asymmetric factor model)

$$\hat{r}_{ui} = \mathbf{p}_i^T \left(\frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \mathbf{q}_j \right)$$

$\begin{bmatrix} p_0 \\ p_1 \\ p_3 \end{bmatrix} * \frac{1}{\sqrt{3}} \left(\begin{bmatrix} q_0 \\ q_1 \\ q_2 \end{bmatrix} + \begin{bmatrix} q_0 \\ q_1 \\ q_1 \end{bmatrix} + \begin{bmatrix} q_0 \\ q_1 \\ q_2 \end{bmatrix} \right)$

RBM (restricted boltzmann machine)

$$p(v_i^k | \mathbf{h}) = \frac{\exp(b_i^k + \sum_{j=1}^F h_j W_{ij}^k)}{\sum_{l=1}^K \exp(b_i^l + \sum_{j=1}^F h_j W_{ij}^l)}$$

The RBM is a neural network with one input layer and one hidden layer. For collaborative filtering, the visible units correspond to items.

GE (global effects)

$$\hat{r}_{ui} = \sum_{e=1}^5 a_i^e q_u^e + \sum_{e=1}^5 b_u^e p_i^e$$

Hand-crafted item features a_i^e and user features b_u^e .

Blending Algorithms

We want to minimize the quadratic prediction error: $RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (\Omega(\mathbf{x}_i) - y_i)^2}$
 \mathbf{X} =train set, \mathbf{y} =targets, $\Omega(\mathbf{x})$ =prediction model, N =#samples

Linear Regression

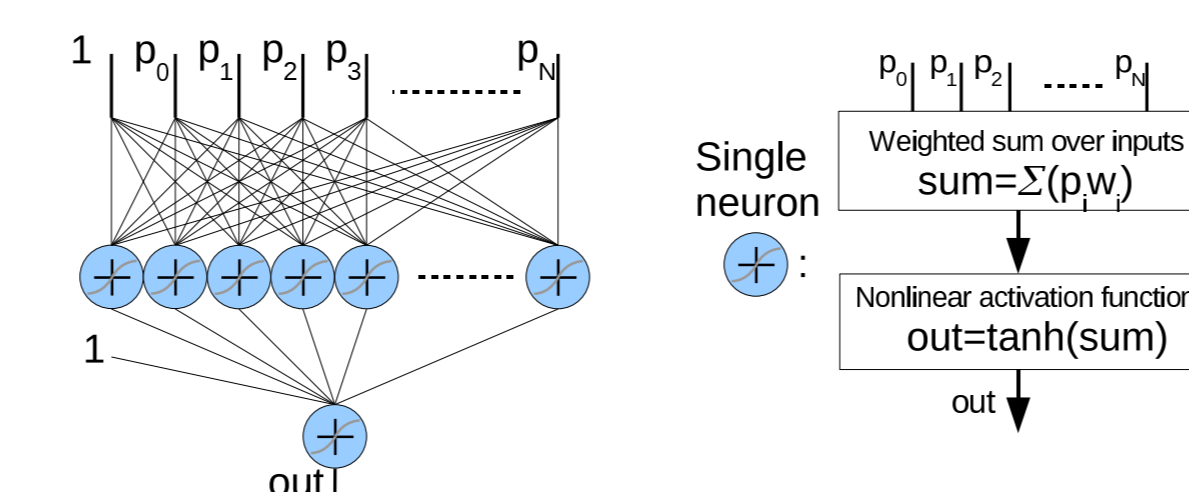
$$\Omega(\mathbf{x}) = \mathbf{x}^T \mathbf{w}$$

Linear model. \mathbf{w} are the regression weights, they are estimated by $\mathbf{w} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$.

Binned Linear Regression

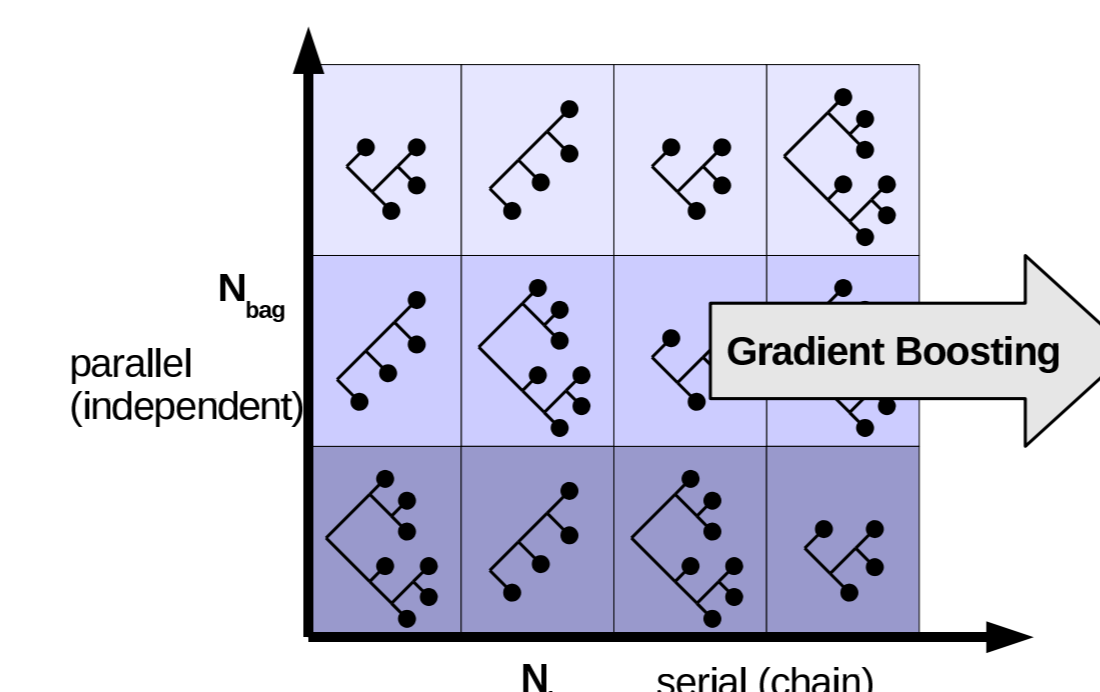
- $$\Omega(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_b$$
- (1) Support : the number of votes by user u
 - (2) Time : the day on which the rating r_{ui} was performed
 - (3) Frequency : the number of ratings from a user at day t

Neural Network



- Stochastic gradient descent
- Linear learnrate decreasing
- Bagging many nets
- Simple output mapping: $out \cdot 2 + 3$

Bagged Gradient Boosted Decision Trees



- Greedy splits (best RMSE reduction)
- max. number of total leaves K
- random feature subspace S
- Gradient Boosting: Chain of trees
- Bagging: Averaging many chains

Kernel Ridge Regression

$$k(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2 / \sigma^2)$$

Gauss kernel. Gram Matrix: $\mathbf{K} = k^{dot}(\mathbf{X}, \mathbf{X}^T)$
 $\mathbf{W} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{b}$, model: $\Omega(\mathbf{x}) = k^{dot}(\mathbf{x}, \mathbf{X}^T) \cdot \mathbf{W}$

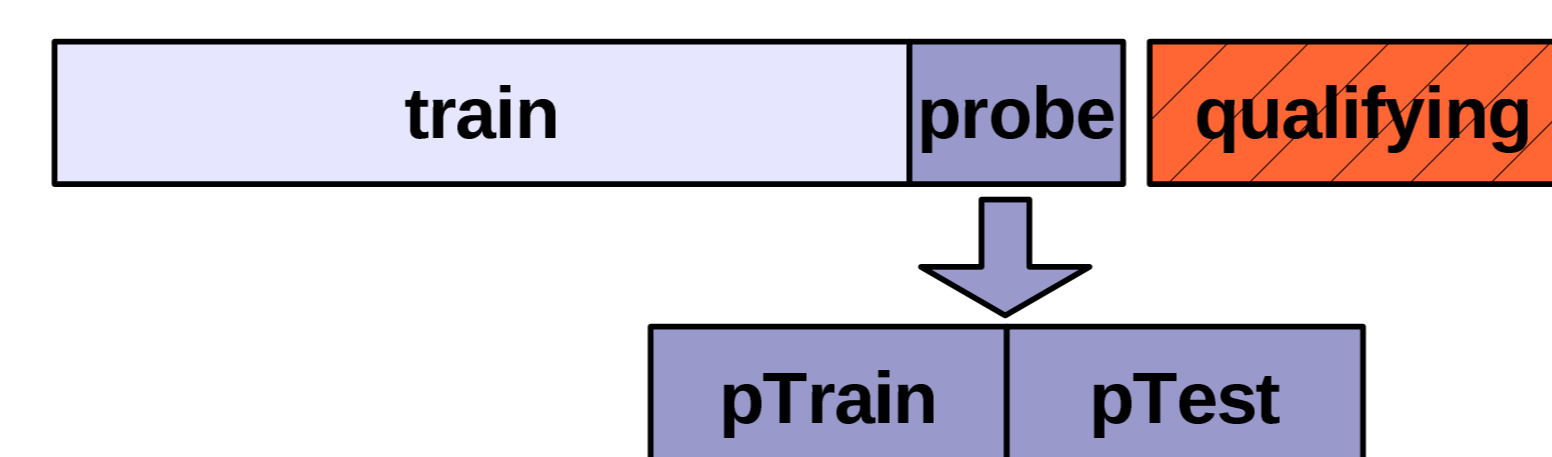
K-Nearest Neighbors

$$\Omega(\mathbf{x}) = \frac{\sum_{k \in D(\mathbf{x})} y_k \cdot d(\mathbf{x}, \mathbf{x}_k)}{\sum_{k \in D(\mathbf{x})} |d(\mathbf{x}, \mathbf{x}_k)|}$$

The set $D(\mathbf{x})$ consists of indices of the k -nearest neighbors to \mathbf{x} . The distance $d(\cdot, \cdot)$ is the inverse of the Euclidean distance.

Dataset

Blending predictions is a supervised machine learning problem. \mathbf{x}_i are F -dimensional predictions. Training set is \mathbf{X} , a $N \times F$ matrix. \mathbf{y} are the targets. $F=19$.



$N_{train} = 100M$
 $N_{probe} = 1.4M$
 $N_{pTrain} = 700k$
 $N_{pTest} = 700k$
 $N_{qual} = 2.8M$

Results

The combination of different kinds of collaborative filtering algorithms leads to significant performance improvements over individual algorithms.

Linear Regression (Baseline RMSE_{pTest}=0.87525)

weight	model (RMSE probe)
-0.083	AFM-1 (0.9362)
-0.084	AFM-2 (0.9231)
-0.077	AFM-3 (0.9340)
+0.088	AFM-4 (0.9391)
+0.098	GE-1 (0.9070)
-0.003	GE-2 (0.9710)
-0.081	GE-3 (0.9443)
+0.176	GE-4 (0.9209)
+0.029	KNN-1 (0.9110)
+0.272	KNN-2 (0.8904)
-0.094	KNN-3 (0.8970)
+0.010	KNN-4 (0.9463)
+0.025	RBM-1 (0.9493)
+0.066	RBM-2 (0.9123)
-0.008	SVD-1 (0.9074)
+0.094	SVD-2 (0.9172)
+0.080	SVD-3 (0.9033)
+0.227	SVD-4 (0.8871)
-0.008	log(support)
+3.673	const. 1

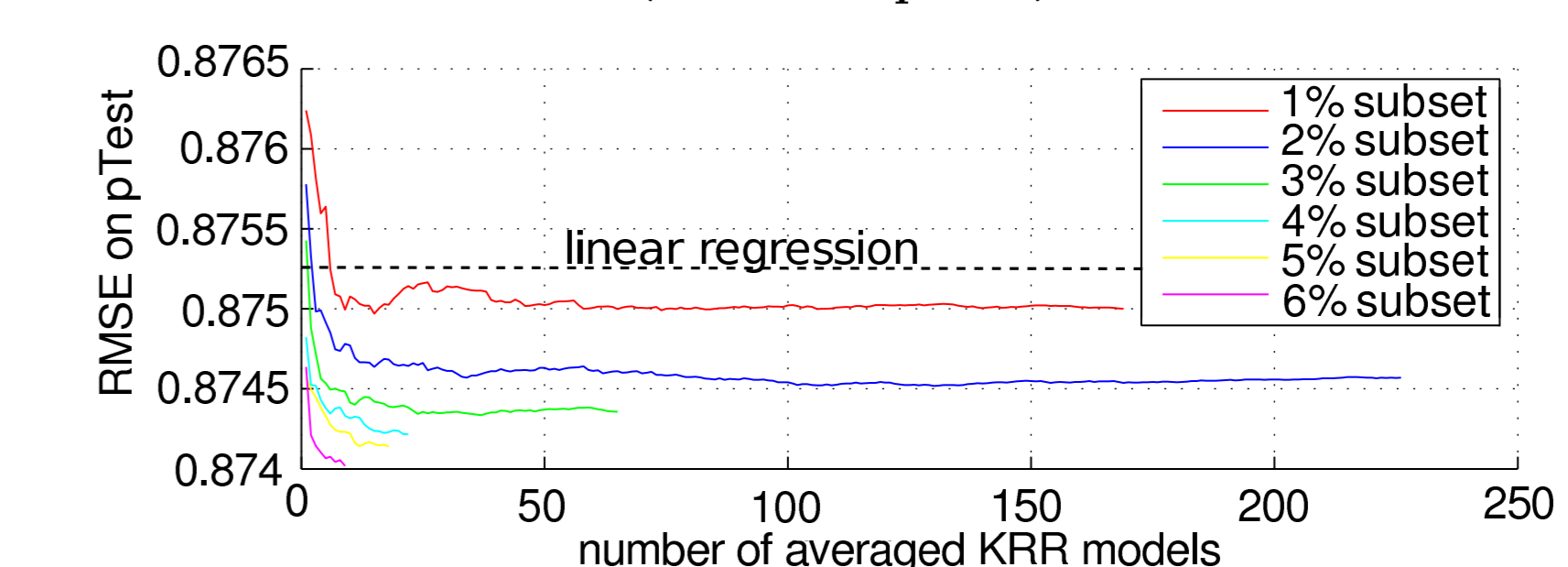
Binned Linear Regression (RMSE_{pTest})

type	RMSE, 2 bins	RMSE, 5 bins	RMSE, 10 bins	RMSE, 20 bins
support	0.8749	0.8747	0.8747	0.8749
date	0.8752	0.8752	0.8753	0.8754
frequency	0.8752	0.8751	0.8751	0.8752

Neural Network

network setup	validation type	RMSE validation	train time	RMSE pTest
19-30-1	bagging size=128	0.873436	17.3[h]	0.873185
19-70-1	bagging size=128	0.87342	33.6[h]	0.873163
19-50-30-1	bagging size=128	0.873455	48.6[h]	0.87318

Kernel Ridge Regression (RMSE_{pTest})



Linear Combination of Many Models (Bagging)

model	RMSE (blend)	weight	parameters
const. 1	-	-0.014	-
NN	0.87345 (0.873445)	0.170	19-100-1, $\alpha = 3.6$, $\beta = 3.0$, $\eta = 5e-4$, $\eta^{(-)} = 5e-7$, 870 epochs, 44.4[h]
GBDT	0.874111 (0.873387)	0.054	$S = 20$, $K = 50$, $\eta = 0.1$, 226 epochs, randomSplitPoint, 6.6[h]
GBDT	0.874603 (0.873384)	0.098	$S = 2$, $K = 300$, $\eta = 0.02$, 267 epochs, optSplitPoint, 8.1[h]
PR	0.874358 (0.87336)	0.141	order=2, $\lambda = 2.4e-6$, with cross interactions, 1.9[h]
PR	0.895951 (0.873351)	-0.033	order=3, $\lambda = 0.054$, no cross interactions, 0.3[h]
NN	0.87345 (0.873296)	0.202	19-100-1, $\alpha = 2$, $\beta = 3.0$, $\eta = 5e-4$, $\eta^{(-)} = 5e-7$, 998 epochs, 47.1[h]
NN	0.873449 (0.873227)	0.371	19-50-30-1, $\alpha = 2$, $\beta = 3.0$, $\eta = 5e-4$, $\eta^{(-)} = 5e-7$, 952 epochs, 49.8[h]
blend	0.873227		total train time: 158.2[h]
	pTest:0.87297		total prediction time 4.5[h]

