The BigChaos Solution to the Netflix Grand Prize

Andreas Töscher and Michael Jahrer

commendo research & consulting
Neuer Weg 23, A-8580 Köflach, Austria
{*andreas.toescher,michael.jahrer*}*@commendo.at*

Robert M. Bell*

AT&T Labs - Research
Florham Park, NJ

September 5, 2009

1 Introduction

The team *BellKor's Pragmatic Chaos* is a combined team of *BellKor*, *Pragmatic Theory* and *BigChaos*. *BellKor* consists of Robert Bell, Yehuda Koren and Chris Volinsky. The members of *Pragmatic Theory* are Martin Piotte and Martin Chabbert. Andreas Töscher and Michael Jahrer form the team *BigChaos*. *BellKor* won the Progress Prize 2007 [4]. The Progress Prize 2008 was won by the combined efforts of *BellKor* and *BigChaos* [5][17].

The documentation of the Netflix Grand Prize consists of three parts. In this document we focus on the contribution of *BigChaos* to the combined Grand Prize Solution.

The document is organized as follows: In Section 2 we describe the Netflix dataset and important statistical properties, followed by a detailed explanation of the training procedure of our predictors in Section 3. Section 4 defines the notation, which we use throughout this document. The algorithmic details can be found in Section 5. In order to combine the predictors of *BigChaos* and the whole team to form a final prediction, we used a combination of nonlinear probe blending and linear quiz blending. The nonlinear probe blending techniques are described in Section 6, the linear quiz blend is described in Section 7. In Appendix A a detailed list of all used predictors is attached.

2 The Netflix Prize Dataset

The dataset consists of 5-star ratings on 17770 movies and 480189 anonymous users. It was collected by Netflix in a period of approximately 7 years. In total, the number of ratings is 100480507; the probe set of size 1408395 is a subset of them. The goal of the contest is to predict the qualifying set (size: 2817131 samples) and achieve a RMSE score of at least 0.8563 on the quiz subset, to get qualified for the Grand Prize. The quiz set is an unknown 50% random subset of the qualifying set. The judging criteria for winning the Netflix Grand Prize is the four digits rounded RMSE score on the test set (remaining 50%). In the case of a tie the earliest submission wins. The probe set has equal statistical properties as the qualifying set. Furthermore it is used as a hold-out set during the competition. Full description of the rules can be found under [1].

*The author contributed Section 7

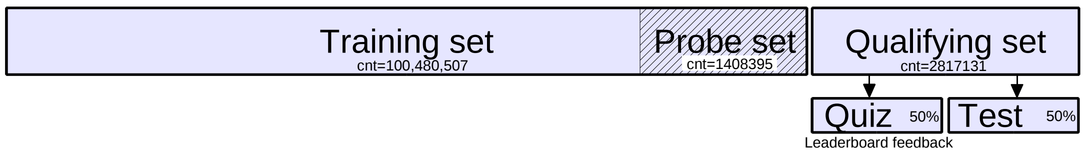

Figure 1: The Netflix Prize dataset in detail. Ratings are available for the training set. Netflix accepts predictions for the qualifying set, the feedback (4 digit precision) is calculated on a 50% random subset of the qualifying set, the quiz set.

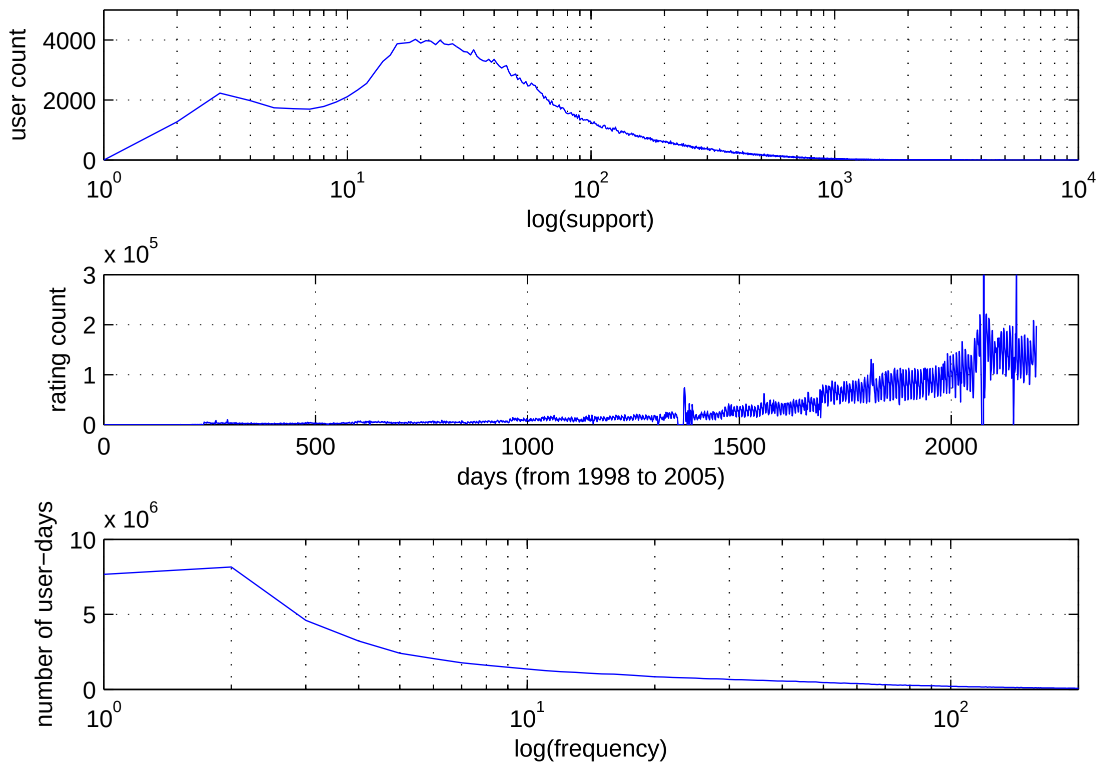

Figure 2: Effects in the rating dataset. First row: User support is the number of votes given from a user. The mode of the user support is on 19 votes, where the average number of votes is 200. Second row: More ratings at the end of the timeline. Third row: Frequency is the number of votes per day per user. Most users gave one or two votes per day. The idea to explore the frequency effect was introduced by our colleagues from Pragmatic Theory.

3 Frameworks

3.1 Optimize the Predictors Individually on the Probe Set

The solutions of the Netflix Progress Prizes of 2007 and 2008 had a focus on the accuracy of the individual collaborative filtering algorithms. Blending techniques were used to combine the independently trained predictors. The predictors were trained to minimize the RMSE on the probe set. First, the probe set is excluded from the training data. The model gets trained to minimize the probe set. For gradient descent methods this means that the training has to stop when the RMSE on the probe set is minimal. Then predictions are stored for the probe set. Afterwards, the probe set gets included into the training data and the training starts again, with exactly the same parameters and initial conditions. After the second training stage, we generate predictions for the qualifying set. These predictions achieve a 0.0030 to 0.0090 better quiz RMSE, compared to their probe RMSE, thanks to expanding the training set.

For every algorithm run, the outcome is a predictor for the probe and qualifying set, which can be used in probe blending, see Section 6. The individual predictor is optimized to achieve the lowest possible probe RMSE. Some algorithms are based on the residuals of others. To calculate the residual error we use the training set predictions of the trained predictor as shown in Figure 3.

3.2 Optimize the Blend

A key observation with ensemble methods is that it is not optimal to minimize the RMSE of the individual predictors. Only the RMSE of the ensemble counts. Thus the predictors which achieve the best blending results are the ones, which have the right balance between being uncorrelated to the rest of the ensemble and achieving a low RMSE individually. An ideal solution would be to train all models in parallel and treat the ensemble as one big model. The big problem is that training 100+ models in parallel and tuning all parameters simultaneously is computationally not feasible.

We approximate this ideal solution by training the models one after another, where each model tries to achieve best results when blended with all preceding models. So the focus shifts from looking at the RMSE of an individual predictor to the RMSE of a blended ensemble. In the following, we refer to the probe RMSE of a linear blend with all preceding predictors as "blend RMSE".

Each predictor tries to achieve best results when it is blended with all preceding ones. Therefore we neither reformulate the error function, nor change any learning rules. We just determine the number of training epochs, which give the best blending improvement. For the first algorithm this means to stop the training when the blend with a constant predictor reaches the minimum. The training of the second algorithm is stopped, when the blend with the first and a constant predictor is minimal. Thus every subsequent predictor stops the training when the blending result with the preceding predictors (including the constant predictor) is minimal. Algorithm 1 provides a pseudo code description of the training's process for one predictor.

All meta parameters, which are tuned automatically, are tuned in order to minimize the blending RMSE instead of the probe RMSE of the individual predictor.

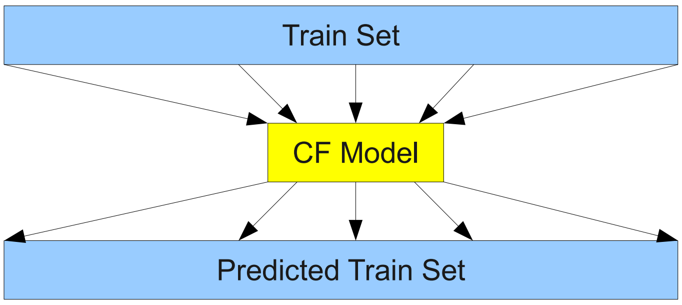

Figure 3: This figure visualizes the standard way for calculating residuals. A CF algorithm is trained on the whole training set and used to predict the training set. In the frameworks 3.1 and 3.2 we always used this method for calculating residuals.

For large ensembles we observed that gradient descent based algorithms tend to train more and more epochs. Therefore, the models are trained over the point of probe minimum. Thus, the RMSE of the individual predictors rises, whereas the RMSE of the ensemble lowers.

An interesting issue is the time and space complexity of this training process. As described above the major difference to the standard training framework is to calculate a blend RMSE instead of the probe RMSE. In the case of a gradient descent algorithm the blend RMSE gets only calculated at the end of each epoch in the training phase. In the retraining phase the calculation of the blend RMSE is not necessary, because the number of needed training epochs were determined in the first training phase. So the time and space complexity does not change for the retraining. In order to calculate the blend RMSE one has to keep the probe predictions of all preceding predictors in memory (about 5.6 MB per file in single precision), so the memory requirement rises linearly with the number of predictors. The time complexity for solving the linear equation system rises with the third power dependent on the number of preceding predictors. For our 101 predictors the calculation of a single blend RMSE can be done in about 1 second, on a modern quad core processor.

The predictor list in Appendix A reports 101 results, which are consecutively optimized to minimize the blend RMSE. Predictions are not clipped. This helps the final blender to improve accuracy. Algorithms with meta parameters, which have strong impact on the outcome, like various MovieKNNs, Global Effects or Global Time Effects, are useful to minimize the blend RMSE. The residuals are always calculated as shown in Figure 3.

```
Input: A matrix P with all previous probe predictions. P always includes a constant predictor (a
       column with ones).
1  Exclude the Probe ratings r from the training set.
2  Initialize the weights.
3  $RMSE_{best} = \infty$
4  $RMSE_{epoch} = 1000$
5  $epochs = 0$
6  while $RMSE_{epoch} \leq RMSE_{best}$ do
7      Train one epoch.
8      if $RMSE_{epoch} \leq RMSE_{best}$ then
9          $RMSE_{best} = RMSE_{epoch}$
10         Save the current weights.
11     end
12     Predict the probe set $\hat{\mathbf{p}}$.
13     Merge current probe prediction $\hat{\mathbf{p}}$ and previous predictions: $\mathbf{X} = [\mathbf{P}\ \hat{\mathbf{p}}]$
14     Calculate blending weights: $\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{r}$
15     Calculate prediction of the current blend: $\mathbf{p} = \mathbf{X} \cdot \mathbf{w}$
16     Calculate the RMSE of the blend: $RMSE_{epoch} = \sqrt{\frac{1}{R}\sum_{i=1}^{R}(p_i - r_i)^2}$; $r_i$ is probe rating $i$, $R$
       is #ratings in the probe set
17     $epochs = epochs + 1$
18 end
19 Load the weights.
20 Generate predictions for the probe set.
21 Insert the probe set into the training set.
22 Initialize the weights (use the same random seed as for the first initialization).
23 while $epochs > 0$ do
24     Train one epoch.
25     $epochs = epochs - 1$
26 end
27 Generate predictions for the qualifying set.
```

Algorithm 1: Pseudo code for training a single predictor with gradient descent and stopping on the optimal blending improvement (Section 3.2 and 3.3). In contrast to the standard procedure described in 3.1 where the training is stopped on the probe minimum, here the training is stopped when the resulting predictor achieves best blending results. This means we do normal epoch wise training and calculate the blending RMSE after each epoch and stop the training when the predictor achieves the best blending results with all preceding predictors. After finding the optimal number of training epochs, we include the probe set and retrain the model using the same initial weights. After the retraining we generate predictions for the qualifying set. As a result we get a predictor with probe and qualifying predictions (like in 3.1), which can be used for any sort of probe blending.

3.3 Optimize the Blend and Correct Residuals

Some of our models deliver very good results on the residuals of others. A well known combination are neighborhood based approaches on the residuals of Restricted Boltzmann Machines (RBM). The first two frameworks are not ideal in this setup. We calculate the residuals for the training set, with a model which was trained on the same ratings (Figure 3). In order to correct this deficiency, we calculate here the residuals with a k-fold cross validation (Figure 4). This brings relatively much computational effort to train each individual model, hence the number of results from this framework results in 10 predictors

(10 complete algorithm runs).

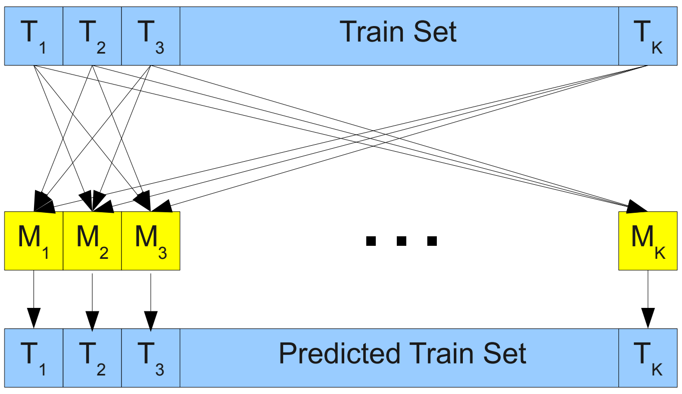

Figure 4: This figure visualizes the idea of "correct residuals". We split the train set into K disjoint sets T_1 to T_K of equal size and train K different CF models M_1 to M_K. The first model M_1 uses the ratings of the sets T_2 to T_K for training and generates predictions for the set T_1. The second model M_2 excludes the set T_2 in the training phase, and calculates predictions for this set. Each rating in the training set is predicted by 1 model. Each rating in the probe and qualifying set is predicted by 34 models. The predictions for the probe and the qualifying set are linear blends of all K models.

As in the previous framework, we train the models to minimize the blend RMSE. We always use $K = 34$, this means 34 models are trained and retrained for generating one predictor. Details can be found in Figure 4. The reason for choosing $K = 34$ is the size of our cluster. We have access to a cluster with 34 compute nodes with each having 4 GB of main memory, and a single head node with 8 GB of main memory. In our experience higher values for K achieve better results, but also the computational efforts increase linearly with K.

4 Notation

The following mathematical notation is used throughout the document. For the description of some algorithms we need additional definitions, which will be made on demand.

- u, v ... users
- i, j ... movies (items)
- r_{ui} ... real rating of user u for item i
- t_{ui} ... the date (days since 01.01.1998) when user u rated item i
- $\widehat{r_{ui}}$, $\widetilde{r_{ui}}$, $\overline{r_{ui}}$ ... prediction of user u for item i
- $N(u)$ ... set of all items, rated by user u including those with unknown ratings on the qualifying set
- $N(i)$ ... set of all users, which rated item i including those with unknown ratings on the qualifying set
- $R(u)$ ... set of items, rated by user u (known ratings)
- $R(i)$ ... set of users, which rated item i (known ratings)
- $R(u, i)$ ... set of k items nearest (largest item-item correlation) to item i which were rated by user u
- α, β, γ, δ, ϵ, ζ, θ, ϑ, κ, ξ ... meta parameters, which can be tuned (small Greek letters)
- For gradient descent based algorithms: learning rate η, regularization λ. Throughout the document, when we use the term regularization, we mean L2-regularization.

- Vectors or matrices have bold face characters, such as $\mathbf{P}$, $\mathbf{Q}$, $\mathbf{p}_i$, $\mathbf{q}_u$, $\mathbf{y}_i$
- In SVD models item features are $\mathbf{p}_i$, user features $\mathbf{q}_u$, asymmetric movie features $\mathbf{y}_i$, biases are μ. If we have more than one item or user feature, we denote it by $\mathbf{p}_i^{(1)}$ and $\mathbf{p}_i^{(2)}$.

5 Collaborative Filtering Algorithms

This section describes all different collaborative filtering (CF) algorithms, which we used in the contest. Some of them are already covered in [17], but still listed here, because most of our Progress Prize 2008 predictors are used in the final blend. For clarity, we state for every algorithm whether it is new, modified or exactly the same as used for the Progress Prize 2008. All the produced predictors are exactly listed in Appendix A, including all used parameters.

5.1 Automatic Parameter Tuning - APT

For some algorithms an automatic tuning of the involved meta parameters is needed. We use APT1 and APT2 methods as described in [17]. Both methods are simple direct search methods, which work without the need of a gradient. APT1 randomly changes single parameters and keeps the changes, when the error improves. APT2 is a structured coordinate search.

APT1 and APT2 are unchanged since the progress prize 2008.

5.2 Movie KNN

Neighborhood based approaches are very popular collaborative filtering methods. The Netflix Prize has clearly shown that these methods can easily be beaten in terms of speed and accuracy by simple factor models. As individual models, they do not achieve outstanding low RMSE values, but in an ensemble of methods they work very well. In the first year of the competition it was shown that they yield great results on the residuals of RBMs. For the Progress Prize 2008 we used a variety of KNN models on the residuals of nearly every algorithm.

Our focus shifted from improving a single predictor or residual-chain of predictors to optimizing the RMSE of the ensemble of predictors. In this context the KNN models do an even better job. The meta parameters are very powerful and help to produce diverse predictors, which blend well. Automatic parameter tuning can do this automatically.

5.2.1 Correlations

We calculate the item-item correlation ρ_{ij} between item i and item j, based on the ratings from users who rated both movies. So all the ratings, where a user rated only one of the movies, are skipped. The vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ are the set of ratings of item i and item j (ratings from common users) with the length L. In our neighborhood models, we use the following correlations:

- Pearson correlation: $\rho_{ij} = \frac{\frac{1}{L-1}\sum_{l=1}^{L}(x_i[l]-\bar{x}_i)(x_j[l]-\bar{x}_j)}{\sqrt{\frac{1}{L-1}\sum_{l=1}^{L}(x_i[l]-\bar{x}_i)^2}\sqrt{\frac{1}{L-1}\sum_{l=1}^{L}(x_j[l]-\bar{x}_j)^2}}$ with $\bar{x} = \frac{1}{L}\sum_{l=1}^{L} x[l]$
- Spearman's rank correlation: $\rho_{ij} = 1 - \frac{6}{L\cdot(L^2-1)} \cdot \sum_{l=1}^{L} d_{ij}[l]^2$ with $d_{ij}[l]$ being the ranking difference
- Set correlation: $\rho_{ij} = \frac{|N(i)\bigcap N(j)|}{\min(|N(i)|,|N(j)|)}$
- MSE correlation: $\rho_{ij} = \frac{1}{\frac{1}{L}\sum_{l=1}^{L}(x_i[l]-x_j[l])^2}$
- Ratio correlation: $\rho_{ij} = \frac{\sum_{l=1}^{L}\omega(x_i[l]-x_j[l])}{L}$ with $\omega(x) = \begin{cases} 1 & |x| \le 1 \\ 0 & \text{else} \end{cases}$

We shrink the correlation ρ_{ij} to zero, based on support $n_{ij} = |N(i)\bigcap N(j)|$:

$$c_{ij} = \frac{\rho_{ij}\cdot n_{ij}}{n_{ij}+\alpha} \tag{1}$$

The ratio correlation works badly, but there are some predictors included, which are based on it. The Pearson and Spearman correlations produced the best results.

5.2.2 KNNMovieV3

This model is exactly the same as in the Progress Prize 2008:

$$c_{ij}^{\text{date}} = \sigma\left(\delta \cdot c_{ij} \cdot \exp\left(\frac{- \mid \triangle t \mid}{\beta}\right) + \gamma\right) \tag{2}$$

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{3}$$

$$\hat{r}_{ui} = \frac{\sum_{j \in R(u,i)} c_{ij}^{\text{date}} r_{uj}}{\sum_{j \in R(u,i)} c_{ij}^{\text{date}}} \tag{4}$$

$\mid \triangle t \mid$ stands for days between the rating r_{ui}, which we are going to predict, and the past rating r_{uj}. We use APT1 to tune this model.

5.2.3 KNNMovieV3-2

The KNNMovieV3 model is extended by 5 meta parameters: ζ, κ, ν, ϑ, ψ. These are used in order to make the model more powerful and flexible. This modification was not part of the Progress Prize 2008:

$$c_{ij}^{\text{new}} = \hat{\sigma}\left(\delta \cdot \text{sign}(c_{ij})|c_{ij}|^{\zeta} \cdot \exp\left(\frac{- \mid \triangle t \mid}{\beta}\right) + \gamma\right) \tag{5}$$

$$\hat{\sigma}(x) = \kappa \cdot \frac{1}{1 + \exp(-x)} + \nu \tag{6}$$

$$L(x) = \begin{cases} x & -\vartheta \leq x \leq \vartheta \\ \vartheta & x > \vartheta \\ -\vartheta & x < -\vartheta \end{cases} \tag{7}$$

$$\hat{r}_{ui} = L\left(\psi \frac{\sum_{j \in R(u,i)} c_{ij}^{\text{new}} r_{uj}}{\sum_{j \in R(u,i)} c_{ij}^{\text{new}}}\right) \tag{8}$$

In the frameworks 3.2 and 3.3 this model yields very good results. In these frameworks we do not optimize the predictors individually, the predictors get optimized to give the best blending results. In contrast to KNNMovieV3 we tune all parameters with APT2, which converges faster.

5.3 Time Dependence Models

Time dependence models account for an over time changing rating mean. We reported all details for the Progress Prize 2008 [17]. The over time changing user mean is called customer time dependence model [CTD] and the flipped version is referred to as movie time dependence model [MTD].

5.4 Restricted Boltzmann Machine - RBM

We use RBMs as described in [15]. Instead of mini batch updates as described in the paper, we use pure stochastic gradient ascent. In our progress prize paper [17], we refer to the conditional RBM with multinomial visible units as RBMV3. The conditional RBM with Gaussian visible units is called RBMV5. The flipped version, where a movie is represented by the users who rated it, is called RBMV6.

The RBMs are unchanged since the Progress Prize 2008. In Section 6.10 we describe how to use the low dimensional representation of users and items on the hidden units as additional features for the blending.

5.5 Global Effects - GE

Global Effects capture statistical corrections applied on user and item side, see [2]. We used them for preprocessing and postprocessing on residuals of other algorithms. The prediction formula of the first global effect (Movie effect) is given by

$$\widehat{r_{ui}} = \theta_i x_{ui} \tag{9}$$

The value of x_{ui} is a static feature, where the value of θ_i is estimated by $\widehat{\theta_i}$ shrunken with α_1

$$\theta_i = \frac{\widehat{\theta_i}\,|R(i)|}{\alpha_1 + |R(i)|} \tag{10}$$

and $\widehat{\theta_i}$ results from a simple estimator from [2]

$$\widehat{\theta_i} = \frac{\sum_{u \in R(i)} r_{ui} x_{ui}}{\sum_{u \in R(i)} x_{ui}^2} \tag{11}$$

All other effects are similar. The explanatory variables (or the fixed features) x_{ui} are different for each effect. For "Movie effect" or "User effect" the x_{ui} are always 1. Thus, global effects can be seen as an SVD approach with either fixed movie of fixed user features. Large values of α shrink the prediction towards 0. In Table 1 we list all used global effects. For some predictors we have not used all 16 effects, so we stated the exact number of used effects for every GE predictor. When not calculating x_{ui} on the residuals of the previous GE, we denote this with "on raw residuals/ratings" in Table 1. In the these cases the x_{ui} are calculated based on the raw ratings or residuals.

For the Progress Prize 2008 we reported 14 GE. In Table 1 we are listing 16, meaning we use 2 new ones. They capture dependencies on the average of the residual directly. One major improvement over last year's implementation is that at the end of the stagewise fitting of optimal α shrinkage, we optimize the parameter α on all effects simultaneously. This is done for 700 epochs with APT2. The runtime per search epoch is about 1 minute. Global Effects are an effective way to optimize the blend RMSE when applied on residuals.

In the framework for minimizing the blend RMSE, we use two minimization steps in Global Effects. The first step optimizes the probe RMSE of the predictor itself, the second step minimizes the blend. This helps to find good start values of all α.

#	effect	shrinkage
1	Movie effect	α_1
2	User effect	α_2
3	User effect: user x sqrt(time(user))	α_3
4	User effect: user x sqrt(time(movie))	α_4
5	Movie effect: movie x sqrt(time(movie))	α_5
6	Movie effect: movie x sqrt(time(user))	α_6
7	User effect: user x average(movie)	α_7
8	User effect: user x votes(movie)	α_8
9	Movie effect: movie x average(user)	α_9
10	Movie effect: movie x votes(user)	α_{10}
11	Movie effect: movie x avgMovieProductionYear(user)	α_{11}
12	User effect: user x productionYear(movie)	α_{12}
13	User effect: user x std(movie)	α_{13}
14	Movie effect: movie x std(user)	α_{14}
15	User effect: user x average(movie) (on raw residuals/ratings)	α_{15}
16	Movie effect: movie x average(user) (on raw residuals/ratings)	α_{16}

Table 1: Global Effects

5.6 Global Time Effects - GTE

Time varying global effects are called global time effects. Their detailed implementation with all formulas to derive a prediction model can be found in [17]. In Table 2 we list all used GTE. GTE benefit from applying two movie and user effects as the first 4 effects. The algorithm has the ability to adapt the kernel width σ per effect. This helps to capture basic movie and user effects on different time scales. The kernel width represents the size of the time-localized window. For the Progress Prize 2008 we reported 19 GTE. These are getting refined and extended to become 24 single effects. The new effects are based on:

- *percentSingleVotes(user):* Percentage of all rating days, where the user gives only one vote.
- *avgStringlenTitle(user):* The average length of the movie title voted by the user.
- *ratingDateDensity(user):* The rating time span (last minus first rating day) divided by the number of rating days (the days on which the user has given at least one rating).
- *percentMovieWithNumberInTitle(user):* The percentage of movies with numbers in the title voted by the user.
- *ratingDateDensity(movie):* The same on movie side.

User side effects need more computational effort (3 minutes per epoch), therefore we limit the automatic parameter tuner APT2 to 120 epochs. On item side we use 400 search epochs. In contrast to Global Effects, optimizing all meta parameters simultaneously is not feasible, due to the huge runtime. However, Global Time Effects are a strong algorithm to minimize the blend RMSE in the ensemble.

#	effect	shrinkage	kernel width
0	Global time mean	α_0	σ_0
1	Movie time effect	α_1	σ_1
2	User time effect	α_2	σ_2
3	Movie time effect	α_3	σ_3
4	User time effect	α_4	σ_4
5	User time effect: user x sqrt(time(user))	α_5	σ_5
6	User time effect: user x sqrt(time(movie))	α_6	σ_6
7	Movie time effect: movie x sqrt(time(movie))	α_7	σ_7
8	Movie time effect: movie x sqrt(time(user))	α_8	σ_8
9	User time effect: user x average(movie)	α_9	σ_9
10	User time effect: user x votes(movie)	α_{10}	σ_{10}
11	Movie time effect: movie x average(user)	α_{11}	σ_{11}
12	Movie time effect: movie x votes(user)	α_{12}	σ_{12}
13	Movie time effect: movie x avgMovieProductionYear(user)	α_{13}	σ_{13}
14	User time effect: user x productionYear(movie)	α_{14}	σ_{14}
15	User time effect: user x std(movie)	α_{15}	σ_{15}
16	Movie time effect: movie x std(user)	α_{16}	σ_{16}
17	User time effect: user x average(movie) (from previous effect)	α_{17}	σ_{17}
18	Movie time effect: movie x average(user) (from previous effect)	α_{18}	σ_{18}
19	Movie time effect: movie x percentSingleVotes(user)	α_{19}	σ_{19}
20	Movie time effect: movie x avgStringlenTitle(user)	α_{20}	σ_{20}
21	Movie time effect: movie x ratingDateDensity(user)	α_{21}	σ_{21}
22	Movie time effect: movie x percentMovieWithNumberInTitle(user)	α_{22}	σ_{22}
23	User time effect: user x stringlengthFromTitle(movie)	α_{23}	σ_{23}
24	User time effect: user x ratingDateDensity(movie)	α_{24}	σ_{24}

Table 2: Overview on 24 Global Time Effects

5.7 Weekday Effect - WE

This algorithm applies mean correction on weekdays. The detailed implementation can be found in [17]. It is unchanged since the Progress Prize 2008.

5.8 Integrated Model - IM

The model with the highest reported accuracy was explained by team BellKor in the 2008 Progress Prize report [5]. This was the inspiration of our Integrated Model. The aim of an integrated model is to explain different effects in the data in one big model. All latent parameters are trained simultaneously by stochastic gradient descent. The training samples are ordered user wise. The asymmetric part receives updates after processing each user. Each latent vector has the same number of features. Time is measured as days since 1/1/1998.

The prediction formula for our Integrated Model is

$$\widehat{r_{uit}} = \mu + \mu_i^{(1)} + \mu_i^{(2)} \cdot d_{it} + \mu_{i,\text{bin}(t)}^{(3)} + \mu_u^{(1)} + \mu_u^{(2)} \cdot d_{ut} + \mu_{u,t}^{(3)} + r_{uit}^{(1)} + r_{uit}^{(2)} + r_{uit}^{(3)} + r_{uit}^{(4)} \tag{12}$$

Item time deviation:

$$d_{it} = \text{dev}(\mu_i^{(t)}, t) \tag{13}$$

User time deviation:

$$d_{ut} = \text{dev}(\mu_u^{(t)}, t) \tag{14}$$

Non linear time deviation is defined by

$$\text{dev}(\mu, t) = \text{sign}\,(t - \mu) \cdot |t - \mu|^{0.5} \tag{15}$$

The mean rating date of a user u:

$$\mu_u^{(t)} = \frac{1}{|R(u)|} \sum_{j \in R(u)} t_{uj} \tag{16}$$

The mean rating date of an item i:

$$\mu_i^{(t)} = \frac{1}{|R(i)|} \sum_{v \in R(i)} t_{vi} \tag{17}$$

The first item-user dot product accounts for complex user-item interactions with time:

$$r_{uit}^{(1)} = \left(\mathbf{p}_i^{(1)} + \mathbf{p}_i^{(2)} d_{it} + \mathbf{p}_{i,\text{bin}(t)}^{(3)}\right)^T \left(\mathbf{q}_u^{(1)} + \mathbf{q}_u^{(2)} d_{ut} + \mathbf{q}_{u,t}^{(3)} + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \left(\mathbf{y}_j^{(1)} + \mathbf{y}_j^{(2)} d_{jt} + \mathbf{y}_{j,\text{bin}(t)}^{(3)}\right)\right) \tag{18}$$

The second part accounts for user-item interaction in frequency. The number of votes by user u on day t is denoted by f_{ut}:

$$r_{uit}^{(2)} = \left(\mathbf{p}_i^{(4)}\right)^T \left(\mathbf{q}_u^{(4)} + \mathbf{q}_{u,f_{ut}}^{(5)}\right) \tag{19}$$

The third part is a neighborhood approach, which models item-item interactions with time. The $\mu_i^{(g)}$ is the item mean and $\mu_u^{(g)}$ the user mean. They are trained at the beginning and kept constant afterwards:

$$r_{uit}^{(3)} = \left(\mathbf{p}_i^{(5)}\right)^T \left(\frac{1}{\sqrt{|R(i)|}} \sum_{j \in R(u)} (r_{uj} - \mu_i^{(g)} - \mu_u^{(g)}) \left(\mathbf{y}_j^{(4)} + \mathbf{y}_j^{(5)} d_{jt}\right)\right) \tag{20}$$

The fourth part is a NSVD [13] with time:

$$r_{uit}^{(4)} = \left(\mathbf{p}_i^{(6)} + \mathbf{p}_i^{(7)} d_{it}\right)^T \left(\frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \left(\mathbf{y}_j^{(6)} + \mathbf{y}_j^{(7)} d_{jt} + \mathbf{y}_{j,\text{bin}(t)}^{(8)}\right)\right) \tag{21}$$

The user frequency f_{ut} is the number of votes from user u on day t. The timeline is divided into a fixed number of bins, bin(t) selects the corresponding bin at time t. The exact number of bins is reported for every predictor in the listing (Appendix A).

All parameters are initialized by uniformly drawing them from the interval $[-1e-3, 1e-3]$; all learning rates η and regularizations λ are optimized by APT2 (1500 search epochs requiring approx. 2 month on 3 GHz Intel Core2 with a latent feature size of 10). The small number of 10 latent features was chosen in order to minimize the runtime for optimization of all meta parameters. We use 10 equal time bins for capturing movie time effects for all integrated model results. The following learn parameters were found by the auto-tuning process. We report 3 results in the predictor list, which have exactly the following values as the learning parameters.

$\eta_{\mu_i} = 0.0017$ $\lambda_{\mu_i} = 8.2807e-06$ $\eta_{\mu_u} = 0.0013$ $\lambda_{\mu_u} = 0.0030$ $\eta_{\mu_{i,\text{bin}(t)}} = 0.0017$ $\lambda_{\mu_{i,\text{bin}(t)}} = 8.2807e-06$ $\eta_{\mu_{u,t}} = 0.0013$ $\lambda_{\mu_{u,t}} = 0.0030$ $\eta_{p_i^{(1)}} = 0.0016$ $\lambda_{p_i^{(1)}} = 0.0328$ $\eta_{p_i^{(2)}} = 0.0071$ $\lambda_{p_i^{(2)}} = 0.1484$ $\eta_{p_{i,\text{bin}(t)}^{(3)}} = 3.1342e-05$ $\lambda_{p_{i,\text{bin}(t)}^{(3)}} = 0.2090$ $\eta_{p_i^{(4)}} = 0.0016$ $\lambda_{p_i^{(4)}} = 0.0324$ $\eta_{p_i^{(5)}} = 1.7027e-04$ $\lambda_{p_i^{(5)}} = 0.0243$ $\eta_{p_i^{(6)}} = 4.7343e-04$ $\lambda_{p_i^{(6)}} = 0.0128$ $\eta_{p_i^{(7)}} = 2.3333e-04$ $\lambda_{p_i^{(7)}} = 1.7976e-04$ $\eta_{q_u^{(1)}} = 0.0057$ $\lambda_{q_u^{(1)}} = 0.0274$ $\eta_{q_u^{(2)}} = 0.0055$ $\lambda_{q_u^{(2)}} = 0.0110$ $\eta_{q_{u,t}^{(3)}} = 0.0018$ $\lambda_{q_{u,t}^{(3)}} = 0.0086$ $\eta_{q_u^{(4)}} = 0.0138$ $\lambda_{q_u^{(4)}} = 0.0063$ $\eta_{q_{u,fut}^{(5)}} = 0.0024$ $\lambda_{q_{u,fut}^{(5)}} = 5.1112e-04$ $\eta_{y_j^{(1)}} = 0.0017$ $\lambda_{y_j^{(1)}} = 7.8809e-04$ $\eta_{y_j^{(2)}} = 4.0333e-04$ $\lambda_{y_j^{(2)}} = 9.3459e-05$ $\eta_{y_{j,\text{bin}(t)}^{(3)}} = 2.7478e-04$ $\lambda_{y_{j,\text{bin}(t)}^{(3)}} = 0.0011$ $\eta_{y_j^{(4)}} = 2.3063e-04$ $\lambda_{y_j^{(4)}} = 0.0218$ $\eta_{y_j^{(5)}} = 0.0021$ $\lambda_{y_j^{(5)}} = 0.0372$ $\eta_{y_j^{(6)}} = 0.0012$ $\lambda_{y_j^{(6)}} = 0.0206$ $\eta_{y_j^{(7)}} = 1.1255e-04$ $\lambda_{y_j^{(7)}} = 0.0021$ $\eta_{y_{j,\text{bin}(t)}^{(8)}} = 2.6278e-04$ $\lambda_{y_{j,\text{bin}(t)}^{(8)}} = 0.0104$

#features	probeRMSE
10	0.8966

Table 3: RMSE on the probe set for the Integrated Model. The automatic parameter tuner APT2 optimizes the RMSE on the probe set until overfitting occurs.

5.9 Maximum Margin Matrix Factorization - MMMF

The maximum margin matrix factorization learns two K-dimensional factor matrices **P**, **Q** and 4 thresholds $\theta_{u1} < \theta_{u2} < \theta_{u3} < \theta_{u4}$ for each user u. The prediction consists of only integer ratings:

$$\widehat{r_{ui}} = \begin{cases} 1, & \mathbf{p}_i^T \mathbf{q}_u < \theta_{u1} \\ 2, & \theta_{u1} \leq \mathbf{p}_i^T \mathbf{q}_u < \theta_{u2} \\ 3, & \theta_{u2} \leq \mathbf{p}_i^T \mathbf{q}_u < \theta_{u3} \\ 4, & \theta_{u3} \leq \mathbf{p}_i^T \mathbf{q}_u < \theta_{u4} \\ 5, & \text{else} \end{cases} \tag{22}$$

The implementation is based on the description in [19]. We were not able to get a RMSE below 1.0 on the probe set.

5.10 NSVD

This is one of the most important approaches to describe user via their rated items. Paterek [13] presents this model in 2007. The model has no explicit user features, so the number of parameters is reduced.

5.10.1 NSVD1

The model learns user and item biases and two sets of item features $\mathbf{p}_i$ and $\mathbf{q}_i$:

$$\widehat{r_{ui}} = \mu_i + \mu_u + \mathbf{p}_i{}^T \left(\frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \mathbf{q}_j \right) \tag{23}$$

A NSVD1 on raw ratings with 500 features achieves a probe RMSE of 0.933.

5.10.2 NSVD2

Simply using the same item features for $\mathbf{p}_i$ and $\mathbf{q}_i$ leads to the NSVD2 model:

$$\widehat{r_{ui}} = \mu_i + \mu_u + \mathbf{p}_i{}^T \left(\frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \mathbf{p}_j \right) \tag{24}$$

A NSVD2 on raw ratings with 500 features achieves a probe RMSE of 0.945.

5.10.3 NSVD1 Discrete - NSVDD

Item features $\mathbf{q}_i$ describe the asymmetric user part in the NSVD1 model. To model the distribution of discrete ratings, we introduce 5 separate item feature sets $\mathbf{q}_{i,d}$ with $d \in \{1,2,3,4,5\}$. A user is explained via a bag of movie rating features, but in contrast to NSVD1 we choose a different item feature vector $\mathbf{q}_{i,d}$, based on the given ratings:

$$\widehat{r_{ui}} = \mu_i + \mu_u + \mathbf{p}_i{}^T \left(\frac{1}{\sqrt{|R(u)|}} \sum_{j \in R(u)} \mathbf{q}_{j,r_{uj}} \right) \tag{25}$$

A NSVD2 on raw ratings with 500 features achieves a probe RMSE of 0.939.

5.11 SBRAMF - Special Biased Regularized Asymmetric Matrix Factorization - and Extensions

Takács et al. [16] suggest a special regularization schema for rating matrix factorization algorithms. The idea is to account for the inverse dependency of regularization λ and learning rate η on the support of user u and item i. By using much more trainable weights than examples (overparameterizing) and auto-tuning of the learning rates and regularizations of the SVD results in a very accurate model. For example the learning rate η_{ui} can be expressed by following function:

$$\eta_{ui} = p_1 + p_2 \frac{1}{\log(|R(u)|+1)} + p_3 \frac{1}{\sqrt{|R(u)|}} + p_4 \frac{1}{|R(u)|} + p_5 \frac{1}{\log(|R(i)|+1)} + p_6 \frac{1}{\sqrt{|R(i)|}} + p_7 \frac{1}{|R(i)|} \tag{26}$$

The basic SVD model with biases is **SBRMF**

$$\widehat{r_{ui}} = \mu_i + \mu_u + \mathbf{p}_i{}^T \mathbf{q}_u \tag{27}$$

In 2008, Koren extended the plain SVD model with implicit information $\frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \mathbf{y}_j$, which is called SVD++ model [11]. In this model a user is represented by the user feature $\mathbf{q}_u$ and a "bag of items" $\mathbf{y}_i$ (the items rated by the user). SBRMF extended by the asymmetric part is equal to SVD++, which we call **SBRAMF**

$$\widehat{r_{ui}} = \mu_i + \mu_u + \mathbf{p}_i{}^T \left(\mathbf{q}_u + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \mathbf{y}_j \right) \tag{28}$$

Add a user and movie time bias $\mu_{u,t}$, $\mu_{i,bin(t)}$: **SBRAMF-UTB**

$$\widehat{r_{uit}} = \mu_i + \mu_u + \mu_{u,t} + \mu_{i,\text{bin}(t)} + \mathbf{p}_i{}^T \left(\mathbf{q}_u + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \mathbf{y}_j \right) \tag{29}$$

Add a user time feature $\mathbf{q}^{(2)}_{u,t}$: **SBRAMF-UTB-UTF**

$$\widehat{r_{uit}} = \mu_i + \mu_u + \mu_{u,t} + \mu_{i,\text{bin}(t)} + \mathbf{p}_i{}^T \left(\mathbf{q}^{(1)}_u + \mathbf{q}^{(2)}_{u,t} + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \mathbf{y}_j \right) \tag{30}$$

Add a movie time feature $\mathbf{p}^{(2)}_{i,\text{bin}(t)}$: **SBRAMF-UTB-UTF-MTF**

$$\widehat{r_{uit}} = \mu_i + \mu_u + \mu_{u,t} + \mu_{i,\text{bin}(t)} + \left(\mathbf{p}^{(1)}_i + \mathbf{p}^{(2)}_{i,\text{bin}(t)} \right)^T \left(\mathbf{q}^{(1)}_u + \mathbf{q}^{(2)}_{u,t} + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \mathbf{y}_j \right) \tag{31}$$

Add an asymmetric time feature $\mathbf{y}^{(2)}_{j,\text{bin}(t)}$: **SBRAMF-UTB-UTF-MTF-ATF**

$$\widehat{r_{uit}} = \mu_i + \mu_u + \mu_{u,t} + \mu_{i,\text{bin}(t)} + \left(\mathbf{p}^{(1)}_i + \mathbf{p}^{(2)}_{i,\text{bin}(t)} \right)^T \left(\mathbf{q}^{(1)}_u + \mathbf{q}^{(2)}_{u,t} + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \left(\mathbf{y}^{(1)}_j + \mathbf{y}^{(2)}_{j,\text{bin}(t)} \right) \right) \tag{32}$$

Add a movie frequency feature $\mathbf{p}^{(3)}_{i,f_{ut}}$: **SBRAMF-UTB-UTF-MTF-ATF-MFF**

$$\widehat{r_{uit}} = \mu_i + \mu_u + \mu_{u,t} + \mu_{i,\text{bin}(t)} + \left(\mathbf{p}^{(1)}_i + \mathbf{p}^{(2)}_{i,\text{bin}(t)} + \mathbf{p}^{(3)}_{i,f_{ut}}\right)^T \left(\mathbf{q}^{(1)}_u + \mathbf{q}^{(2)}_{u,t} + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \left(\mathbf{y}^{(1)}_j + \mathbf{y}^{(2)}_{j,\text{bin}(t)}\right)\right) \tag{33}$$

Add an asymmetric frequency feature $\mathbf{y}^{(3)}_{j,f_{ut}}$: **SBRAMF-UTB-UTF-MTF-ATF-MFF-AFF**

$$\widehat{r_{uit}} = \mu_i + \mu_u + \mu_{u,t} + \mu_{i,\text{bin}(t)} + \left(\mathbf{p}^{(1)}_i + \mathbf{p}^{(2)}_{i,\text{bin}(t)} + \mathbf{p}^{(3)}_{i,f_{ut}}\right)^T \left(\mathbf{q}^{(1)}_u + \mathbf{q}^{(2)}_{u,t} + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \left(\mathbf{y}^{(1)}_j + \mathbf{y}^{(2)}_{j,\text{bin}(t)} + \mathbf{y}^{(3)}_{j,f_{ut}}\right)\right) \tag{34}$$

Model extension (+)	epoch time	#epochs	probeRMSE, $k = 50$ features
SBRMF - SVD with biases	17[s]	69	0.9054
SBRAMF - asymmetric part	50[s]	30	0.8974
+UTB - user time bias	61[s]	50	0.8919
+UTF - user time feature	62[s]	38	0.8911
+MTF - movie time feature	74[s]	37	0.8908
+ATF - asymmetric time feature	74[s]	44	0.8905
+MFF - movie frequency feature	149[s]	46	0.8900
+AFF - asymmetric frequency feature	206[s]	45	0.8886 (0.8846 with $k = 1000$)

Table 4: Our most accurate single model: Matrix factorization with integration of time and frequency effects.

All models are trained user-wise with stochastic gradient descent to minimize the quadratic error function. In Table 5 we list all meta parameters found by APT2.

5.12 Symmetric-View-SVD++

Now we flip the SVD++ model [11] to the movie side by representing a movie as a "bag of users". Efficient training of these models is done user-wise, hence we fix either one side and train the flipped view, while the other side stays constant. The prediction is given by:

$$\widehat{r_{ui}} = \mu_i + \mu_u + \left(\mathbf{p}^{(1)}_i\right)^T \left(\mathbf{q}^{(1)}_u + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} \mathbf{y}_j\right) + \left(\mathbf{q}^{(2)}_u\right)^T \left(\mathbf{p}^{(2)}_i + \frac{1}{\sqrt{|N(i)|}} \sum_{v \in N(i)} \mathbf{z}_v\right) \tag{35}$$

$\mathbf{p}^{(1)}_i$, $\mathbf{y}_j$ and $\mathbf{p}^{(2)}_i$ are item features. The user features are: $\mathbf{q}^{(1)}_u$, $\mathbf{q}^{(2)}_u$ and $\mathbf{z}_v$. This model reach a RMSE of 0.92 on the probe set.

5.13 SVD-Time

This model is the same as reported in Section 9 of the BigChaos Progress Prize 2008 Solution [17], called "Time SVD". A prediction is given by a dot product of the time-shifted item and user feature. The number of time bins per item and user is set by M and N at the beginning of training. Each item and each user has a begin and an end rating date. The item begin and end dates are: $t^{(0)}_i$, $t^{(1)}_i$. For user begin and end dates we denote: $t^{(0)}_u$, $t^{(1)}_u$

The item time shift is:

$$b_{it} = \left\lfloor M \frac{t - t^{(0)}_i}{t^{(1)}_i - t^{(0)}_i} \right\rfloor \tag{36}$$

The user shift bin is:

$$b_{ut} = \left\lfloor N \frac{t - t^{(0)}_u}{t^{(1)}_u - t^{(0)}_u} \right\rfloor \tag{37}$$

parameter	p_1	p_2	p_3	p_4	p_5	p_6	p_7
η_{μ_i}	3.676e-10	5.626e-09	0.00381	0.00382	4.526e-09	0.00027	1.726
η_{μ_u}	0.00069	0.00472	0.00692	0.00011	1.146e-06	1.470e-07	1.770e-07
$\eta_{\mu_{u,t}}$	0.00062	0.0015	0.003045	0.001307	0.000371	2.698e-05	7.619e-05
$\eta_{\mu_{i,\mathrm{bin}(t)}}$	1.989e-05	7.410e-05	1.309e-05	8.186e-06	5.396e-05	0.0001612	8.653e-05
$\eta_{\mathbf{p}_i^{(1)}}$	0.00235	0.000674	0.000179	1.075e-08	0.00211	0.0117	0.552
$\eta_{\mathbf{p}_{i,\mathrm{bin}(t)}^{(2)}}$	9.406e-05	8.828e-05	1.223e-11	3.276e-11	0.000172	0.000141	2.988e-06
$\eta_{\mathbf{p}_{i,f_{ut}}^{(3)}}$	0.000248	0.000931	6.456e-11	1.638e-11	0.000524	6.493e-05	3.917e-06
$\eta_{\mathbf{q}_u^{(1)}}$	8.517e-06	0.00124	0.000617	5.963e-05	0.00469	0.000146	0.000215
$\eta_{\mathbf{q}_{u,t}^{(2)}}$	4.242e-07	4.081e-05	0.00177	5.077e-05	5.331e-09	0.0126	1.757e-05
$\eta_{\mathbf{y}_j^{(1)}}$	3.272e-05	9.223e-05	6.514e-05	0.000220	0.000288	6.498e-05	0.00452
$\eta_{\mathbf{y}_{j,\mathrm{bin}(t)}^{(2)}}$	1.749e-06	1.775e-05	0.000131	1.720e-05	3.744e-08	4e-07	0.000633
$\eta_{\mathbf{y}_{j,f_{ut}}^{(3)}}$	9.232e-06	5.382e-05	6.566e-05	2.454e-05	2.403e-08	4e-07	0.0176
λ_{μ_i}	6.281e-10	4.128e-08	2.481e-07	5.194e-07	2.289e-08	3.423e-07	2.470e-05
λ_{μ_u}	9.348e-05	0.0082	0.0027	3.698	9.872e-06	0.000707	0.00454
$\lambda_{\mu_{u,t}}$	0.00030	0.000187	0.00108	0.000606	0.000558	0.000426	0.00203
$\lambda_{\mu_{i,\mathrm{bin}(t)}}$	0.000244	0.000341	0.000427	0.000484	0.000167	0.000221	0.000829
$\lambda_{\mathbf{p}_i^{(1)}}$	0.00239	0.00710	0.00974	0.189	0.00437	0.127	1.236
$\lambda_{\mathbf{p}_{i,\mathrm{bin}(t)}^{(2)}}$	0.0116	0.0109	0.316	2.755	0.00392	0.00165	0.0299
$\lambda_{\mathbf{p}_{i,f_{ut}}^{(3)}}$	0.000348	0.000485	0.101	2.755	0.000173	7.302e-05	0.00246
$\lambda_{\mathbf{q}_u^{(1)}}$	7.798e-05	0.183	1.603	1.595	6.413e-06	4.249e-05	0.000432
$\lambda_{\mathbf{q}_{u,t}^{(2)}}$	0.000588	0.00733	0.000205	0.000139	0.0184	0.00611	0.00811
$\lambda_{\mathbf{y}_j^{(1)}}$	0.0187	0.0736	0.158	0.0166	0.000152	1.698e-05	0.000168
$\lambda_{\mathbf{y}_{j,\mathrm{bin}(t)}^{(2)}}$	0.0633	0.0633	0.000905	0.00129	0.0633	0.000633	2.8e-05
$\lambda_{\mathbf{y}_{j,f_{ut}}^{(3)}}$	0.191	0.0275	0.00157	0.000646	0.0551	0.000633	2.8e-05

Table 5: For every parameter, 7 constants p_1, ..., p_7 are found by APT2 in approximately 1500 search epochs. Training the SBRAMF-UTB-UTF-MTF-ATF-MFF-AFF with 50 features reaches a RMSE=0.8886 on the probe. One optimization epoch means a complete training, until overfitting on the probe set occurs. The model is trained epoch-wise, the RMSE on the probe set decreases monotonically until it starts to increase. This is the point, where overfitting on the probe set occurs (we denote that point with #epochs in Table 4). The goal was to minimize the RMSE on the probe set.

The prediction of the SVD-Time model is given by:

$$\widehat{r_{uit}} = \sum_{k=1}^{K} p_{i,(k+b_{it})} \cdot q_{u,(k+b_{ut})} \tag{38}$$

In our implementation, the SVD-Time models achieve a RMSE of 0.93 on the probe set.

5.14 SVD with Adaptive User Factors - SVD-AUF

Adaptive User Factors is a post-processing step on user features of a plain SVD model. It was first mentioned by BellKor in [2]. A SVD (or matrix factorization model) approximates the user-item rating matrix by two low-dimensional feature matrices. $\mathbf{P}$ are the item features

$$\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_M] \tag{39}$$

and $\mathbf{Q}$ the user features:

$$\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_U] \tag{40}$$

At this stage we assume that we have access to item features $\mathbf{P}$, which stem from a trained matrix factorization model. Please note that the features can also come from trained NSVD models. For every single prediction $\widehat{r_{ui}}$

$$\widehat{r_{ui}} = \mathbf{p}_i^T \widehat{\mathbf{q}_u} \tag{41}$$

we recompute the user feature in the following way

$$\widehat{\mathbf{q}_u} = \text{SolveLinearEquationSystem}(\mathbf{A}, \mathbf{b}). \tag{42}$$

The matrix $\mathbf{A}$ has $|R(u)|$ rows and K columns, where K is the number of features. The target vector $\mathbf{b}$ consists ratings from user u. Each element in the matrix $\mathbf{A} = \{a_{jk}\}$ is given by

$$a_{jk} = p_{jk} \cdot \text{sim}(i, j) \tag{43}$$

The p_{jk} are values from the K-dimensional item feature $\mathbf{p}_j$. In Equation 42 we use two different linear equation solvers. We call the first one "PINV", which is a standard solver from the LAPACK software package. The second one is the nonnegative solver from [2]. The ridge regression constant in the numeric solvers is denoted by λ. Exact values for λ and the kind of solver are reported for every predictor in Appendix A. In Table 6 we list the similarity measures, which were used in the SVD-AUF

similarity measure	$\text{sim}(i, j)$
DateSim	$\kappa + (\lvert t_{ui} - t_{uj}\rvert + \gamma)^{-\beta}$
SupportSim	$\kappa + \lvert(\lvert R(i)\rvert - \lvert R(j)\rvert)\rvert^{-\beta}$
ProductionYearSim	$\kappa + \lvert \text{year}_i - \text{year}_j \rvert^{-\beta}$
PearsonSim	$\kappa + \sigma(\delta \cdot c_{ij} + \gamma)$

Table 6: SVD-AUF: These similarity measures are used for the SVD with adaptive user factors. The c_{ij} is a shrunken Pearson correlation as in Equation 1. $\sigma(x)$ is the sigmoid function from Equation 3.

The similarity measure and all meta parameters from Table 6 are reported for every results in the predictor list (Appendix A).

5.14.1 SVD-AUF with Kernel Ridge Regression

An extention to the SVD-AUF algorithm is the use Kernel Ridge Regression (see [9] for details). KRR learns a model from the input features $\mathbf{A}$ and targets $\mathbf{b}$, as sketched in Algorithm 2. Standard linear regression needs dot products of input dimensions (columns of feature matrix $\mathbf{A}$), kernel ridge regression relies on kernel dot product of input features (rows of feature matrix $\mathbf{A}$). This forms the Gram matrix $\mathbf{K}$. If a linear kernel is used, KRR is equivalent to linear regression.

As stated in the SVD-AUF, a trained matrix factorization model, especially the item features $\mathbf{P}$ (see Equation 39) must be available. We train a KRR model $\Omega^u(\mathbf{x})$ for each user u. Features and targets are equal to the from Equation 42. The KRR is able to predict a rating for any given item feature $\mathbf{p}_i$

$$\widehat{r_{ui}} = \Omega^u(\mathbf{p}_i) \tag{44}$$

The kernels which were used in SVD-AUF with Kernel Ridge Regression are listed in Table 7.

```
Input: Data matrix $\mathbf{A}$, targets $\mathbf{b}$, new test feature $\mathbf{x}$
Output: The prediction model $\Omega(\mathbf{x})$
Tunable: Ridge regression constant $\lambda$, kernel hyperparameters
1 $\mathbf{K} = k^{\text{dot}}(\mathbf{A}, \mathbf{A}^T)$, where $k(\mathbf{x}_i, \mathbf{x}_j)$ is the kernelized dot product
2 $\mathbf{W} = (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{b}$
3 $\Omega(\mathbf{x}) = k^{\text{dot}}(\mathbf{x}, \mathbf{A}^T) \cdot \mathbf{W}$
```

Algorithm 2: The KRR - "kernel ridge regression" learn algorithm

In Algorithm 2 the steps required to train the KRR algorithm can be seen. The computational and memory intensive step is to invert the Gram matrix $(\mathbf{K} + \lambda\mathbf{I})^{-1}$, which has a time complexity of $O(N^3)$. This limits the number of N training samples, so we are skipping ratings from users with too many votes. In this case, the ratings of user u are sorted according to their movie support. Afterwards we limit the number of ratings per user to 1000.

kernel	$k(\mathbf{x}_i, \mathbf{x}_j)$
polynomial	$(\alpha + {\mathbf{x}_i}^T \mathbf{x}_j)^\beta$
pow	$\kappa^{(\alpha + {\mathbf{x}_i}^T \mathbf{x}_j)\beta}$
extended polynomial	$\lvert\gamma(\alpha + {\mathbf{x}_i}^T \mathbf{x}_j)\rvert^\beta$

Table 7: SVD-AUF: These kernels are used in KRR postprocessing.

5.15 SVD Trained with Alternating Least Squares - SVD-ALS

The SVD-ALS idea was described in [4]. For the SVD model, training of item and user features are done by applying a linear equation solver on item side, while the user features are kept constant. Afterwards the user features are updated with constant item features. The training stops, when the features do not change any more. SVD-ALS has the advantage that no learning rate is required. A regularization for the solver is still required. We use a standard linear equation solver as described in Section 6.1. RMSE values on the probe set are slightly worse compared to a SVD with stochastic gradient descent. Our best models achieve a probe RMSE in the region of 0.91.

5.16 Rating Matrix Factorization - SVD

This is a plain SVD with no user and movie biases. The training is done with stochastic gradient descent over a randomized training list of samples $\mathcal{L} = \{(u_1, i_1), \dots, (u_L, i_L)\}$. Details can be found in [17]. A prediction is given by

$$\widehat{r_{ui}} = {\mathbf{p}_i}^T \mathbf{q}_u \tag{45}$$

Large SVD models achieve a RMSE of 0.905 on the probe set.

5.17 Neighborhood Aware Matrix Factorization - NAMF

This model combines matrix factorization with neighborhood information and is fully described in [17] and [18]. For tuning the involved parameters, we use APT1. This method stayed unchanged since the Progress Prize 2008.

5.18 Regression on Similarity - ROS

Factorization of the item-item or user-user correlation is described in [18]. This method has not changed since the Progress Prize 2008.

6 Probe Blending

Since the progress prize 2008 we made major progress on the probe blending side. Blending a set of predictors is a standard regression problem. The list of methods for regression problems is very long, but due to the 1408395 probe samples, not every method can be used. In the beginning of the competition we used linear regression. For the progress prize 2008 we additionally used neural networks. It turned out that the results can be additionally improved by using ensemble of probe blends.

The Netflix prize showed that a very diverse ensemble of CF algorithms yields great results. We used the same idea for the blending. So we used diverse methods on diverse sets of predictors. In the following we describe all regression methods, additional features and predictor subsets.

6.1 Linear Regression - LinearBlend

Linear Regression is a standard tool and yields good results. The big advantage of pure linear regression is clearly the speed. We use a L2 regularization to control overfitting with the regularization constant λ. A standard least square solver (LAPACK) is used. The time complexity is nm^2, with n being the number of equations and m the number of predictors (features). The space complexity is m^2. Sometimes we restrict the linear equation solver to nonnegative weights. In these cases we use the iterative solver as described in [3].

6.2 Polynomial Regression - PolyRegressionBlend

As an extension to linear regression, we add higher order terms to capture non-linear properties of the predictors. Polynomial regression simply extends the predictor set with higher order terms $\mathbf{X} \cdot \mathbf{w}_0 + \mathbf{X}^2 \cdot \mathbf{w}_1 + \mathbf{X}^3 \cdot \mathbf{w}_2 + ...$ Order 2 means that we extend the set with quadratic duplicates.

6.3 Binned Linear Regression

The idea of binned linear regression is to divide the probe set into disjoint subsets, called bins in the following, and fit a different set of blending weights to every bin. We choose the boundaries in order to have equal sized bins (equal number of ratings) and to prevent the number of bins from getting too large to avoid overfitting each bin.

6.3.1 Support Based Bins - N-SupportBins

We define the support s_{ui} as the minimum of the ratings by user u and item i.

$$s_{ui} = \min(|N(u)|, |N(i)|) \tag{46}$$

Based on this support, we split the probe set into bins. The motivation behind these splits is that some algorithms work better on users with few ratings, while others give great results when there is enough information available. These split criteria work best (compared with date and frequency).

6.3.2 Date Based Bins - N-DateBins

Another way to split the probe set is to base the bins on the rating date. The results are slightly worse, compared to the support based splits.

6.3.3 Frequency Based Bins - N-FrequencyBins

Here the probe set is binned based on the number of ratings a user has given per day. Results are slightly worse compared to the support based splits.

6.3.4 Clustering Based Bins - N-ClusterBins

Every result from the exact residual framework (see Section 3.3) delivers a k-fold cross-validation prediction of the training set. This has the property that strong models cannot overfit the training set, which has a positive effect on the quality of the clusters. Initially, each user and item is assigned to M item and N user clusters $c_i^{(m)}$ and $c_u^{(n)}$. The first step is to calculate linear regression blending weights per cluster. The second step is to move each item i and each user u to the cluster, where the training RMSE lowers best. So assignments to clusters were changed by a greedy schema. This is repeated 100 times, the cluster assignment stabilizes very fast. The final clusters $c_i^{(m)}$ and $c_u^{(n)}$ were used to calculate a binned linear regression on the probe set, this means we calculate a separate linear regression for each cluster.

6.4 Subset Generation

Not all blends got trained on all available predictors. This has two main reasons. The most important is, to introduce diversity in the ensemble of blends. The other reason is that it reduces the computational complexity of the individual blends.

6.4.1 Forward Selection

The goal is to select K predictors, which give good blending results. We start with the best single predictor and iteratively add the predictor which improves the blending RMSE best. Obviously this greedy method does not guarantee to find the optimal subset of predictors.

6.4.2 Backward Selection

The goal is the same as for the forward selection. This time we start with a linear blend, including all available predictors, and iteratively deselect the predictor which contributes the least. The deselection process is stopped, if there are only K predictors left. As for the forward selection, there is no guarantee to find the optimal subset.

6.4.3 Probe-Quiz Difference Selection - PQDiff

The typical predictor has a lower RMSE on the quiz set than on the probe set. We call it probe-quiz difference. This difference comes from the retraining. For the quiz set predictions the algorithms are retrained on the complete available data, including the probe data. So there is more training data available for the quiz predictions. This is why these predictions get more accurate. This probe-quiz difference differs between the algorithms.

The idea is now, to group the predictors by the probe-quiz difference.

6.5 Neural Network Blending - NNBlend

A neural network is a function approximator from a P-dimensional input space to the output space. It is trained by stochastic gradient descent by applying the backprop algorithm. P is the number of predictions used for blending. The output is a scalar, representing the predicted value. We have a detailed description of the setup in BigChaos Progress Prize 2008 report [17]. This is part of our most successful blending schema, in terms of minimizing the RMSE on the quiz. All NNBlends use $\log(|R(u)|+1)$ and $\log(|R(i)|+1)$ as additional inputs. The initial learning rate is $\eta = 0.0005$ and every epoch we subtracted $\eta^{(-)}$. We report $\eta^{(-)}$, the number of epochs where the training stops, number of hidden layers and the neuron configuration. For standard NNBlends we use $\eta^{(-)} = 3e-7$ and train for 1334 epochs. During ongoing improvement of the blending technique we use k-fold cross validation to optimize the net configuration, apply 2 hidden layers and break the training when the RMSE on the validation set is minimal.

6.6 Ensemble Neural Network Blending - ENNBlend

Sampling the P-dimensional predictor space can help to model interactions between single predictors. The blending schema selects k random predictors from P in total and blend them with a small neural net (NNBlend). This results in N predictions, which are again blended by a binned blender. Please note that every NNBlend itself adds two additional support inputs (see 6.5). Good values are $k = 4$, $N = 1200$ and a linear blend on 4-FrequencyBins. The ENNBlend reaches the best quiz RMSE with probe blending.

6.7 Bagged Gradient Boosted Decision Tree - BGBDT

BGBDT combines the idea of Gradient Boosted Decision Trees, as described by Friedman[7],[8], with the bagging and random subspace idea of Random Forests [6]. The splits are simple axis parallel ones. Random Forests use a random subspace idea, which is also used here. At each split point, we select S random features, calculate optimal splitpoints for every selected feature and use the split, which reduces the RMSE best. Every tree is grown to the maximum tree depth d. The splitting is also stopped, if a leaf node has less than $N_{\min}$ datapoints. So in difference to Random Forests, the trees are not fully grown.

The basic idea of gradient boosting is to successively train trees on the residuals of the previous ones. The learning rate λ controls the contribution of a individual tree and N_{boost} stands for the number of trees in a single boosting chain. We additionally use a bagging idea. N_{bag} stands for the bagging size (the number of boosting chains, which are trained simultaneously). Each bagging set of training ratings is drawn with replacement from the originals and has the same size. The structure of a BGBDT can be seen in Figure 5, and the general training's procedure is described in Algorithm 3.

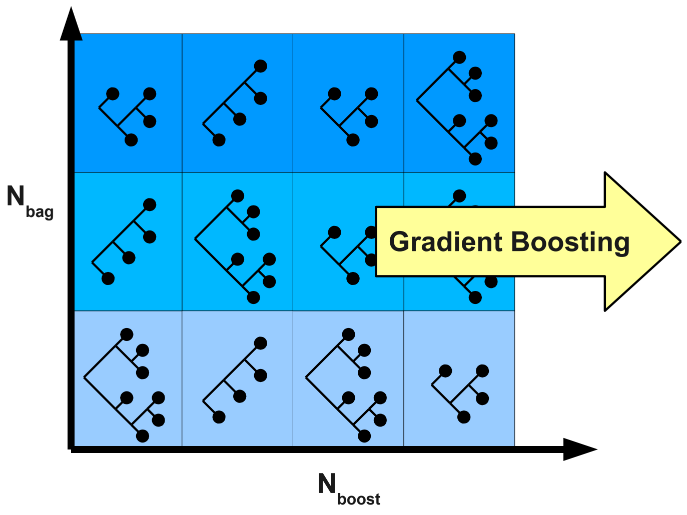

Figure 5: This figure shows the structure of a BGBDT. Each cell represents a simple decision tree. The tree on the left trains on the raw data. The second tree trains on the residual error of the first; the third tree on the residuals of the second and so on. Thus a colored row forms a chain of gradient boosted decision trees. Each colored row represents different training examples, which are drawn with replacement from the original training samples (bagging). So we train multiple chains of gradient boosted decision trees in parallel, whereas each chain uses its own training set.

```
Input: A matrix $\mathbf{P}$ with probe predictions, and a vector with target ratings $\mathbf{r}$.
1  for $i$ = 1 to $N_{bag}$ do
2      Draw $\mathbf{P}_i$ with replacement from $\mathbf{P}$, while $\mathbf{r}_i$ should contain the corresponding target ratings.
3  end
4  $RMSE_{\text{best}} = \infty$
5  $RMSE_{\text{epoch}} = 1000$
6  $j = 0$
7  while $RMSE_{epoch} \leq RMSE_{best}$ and $j < N_{boost}$ do
8      for $i$ = 1 to $N_{bag}$ do
9          $\mathbf{T}_{ij} = TrainSingleTree(\mathbf{P}_i, \mathbf{r}_i, N_{\min}, S)$; trains the tree $\mathbf{T}_{ij}$
10         $\mathbf{r}_i = \mathbf{r}_i - \lambda \cdot GenerateSingleTreePrediction(\mathbf{T}_{ij}, \mathbf{P}_i)$; calculate the residuals
11     end
12     Calculate the $RMSE_{\text{epoch}}$.
13     if $RMSE_{epoch} \leq RMSE_{best}$ then
14         $RMSE_{\text{best}} = RMSE_{\text{epoch}}$
15     end
16     $j = j + 1$
17 end
```
Algorithm 3: The training of a BGBDT.

As additional input features we always use the number of user votes, movie votes and the rating date. A very nice property of this sort of tree is that there is no need for a rescaling or any sort of monotonic transformation (e.g. logarithm) of the features as needed for neural networks.

Experiments show that these parameters are very insensitive. Good parameters are $\lambda = 0.1$, $N_{\text{boost}} = 250$, $N_{\text{bag}} = 32$, $d = 12$, $N_{\min} = 100$, $S = 10$. The exact parameters used for the blendings can be found in the predictors list in Appendix A.5. The RMSEs of the BGBDT blends are not as good as for neural networks, but they do very well in the final quiz blend.

6.8 KRR on a Probe Subset - KRRBlend

We use linear, binned linear regression, non linear blending methods like NNBlend and BGBDT. Kernel regression methods are very hard to apply on the blending problem because of the huge size of the probe set. To make Kernel Ridge Regression work, we use a subset of the probe set as training set for the

KRR. Size of 4000 is a good compromise between training/prediction speed and accuracy. The algorithm we use for blending is exact the same as in Algorithm 2 from Section 5.14.1. For all blending results, a Gauss kernel was used.

6.9 SVD Feature Predictor Extraction

The goal of predictor feature extraction is to generate a probe/qualifying pair from latent features of a collaborative filtering model. A matrix factorization model learns features from the data with stochastic gradient descent. Users and items have their latent factors, which describe their property. These features can be used to extend the blend. We use our most accurate SVD model to produce predictions, which we integrate in various blends. Recall the prediction formula from the SBRAMF-UTB-UTF-MTF-ATF-MFF-AFF model $\widehat{r_{uit}} =$

$$\mu_i+\mu_u+\mu_{u,t}+\mu_{i,\text{bin}(t)}+\left(\mathbf{p}_i^{(1)}+\mathbf{p}_{i,\text{bin}(t)}^{(2)}+\mathbf{p}_{i,f_{ut}}^{(3)}\right)^T\left(\mathbf{q}_u^{(1)}+\mathbf{q}_{u,t}^{(2)}+\overbrace{\frac{1}{\sqrt{|N(u)|}}\sum_{j\in N(u)}\left(\mathbf{y}_j^{(1)}+\mathbf{y}_{j,\text{bin}(t)}^{(2)}+\mathbf{y}_{j,f_{ut}}^{(3)}\right)}^{asymmetric\ part}\right).$$

This model has 4 biases (2 item biases, 2 user biases), 6 item dependent features (3 item, 3 asymmetric) and 2 user dependent features. The asymmetric part of the model is an additional user feature. For every (u, i) in the probe and qualifying set we extract $4 + k \cdot (6 + 2 + 1)$ predictors, where k is the number of features. As an extension, we add all possible dot products between item and user features. This results in 50 additional predictors, which we denote with -cross- in the probe blending Section 6.

6.10 RBM Feature Predictor Extraction

In its essence a RBM calculates a low dimensional representation of the visible units. This property can be used to calculate user and movie features. These features are great additional inputs for the nonlinear probe blends. A user is represented as a bag of movies on the visible layer of the RBM, on the hidden layer you get a low dimensional representation of the user. The same can be done with movies, in order to get a low dimensional movie representation. For the blending features we use less hidden units as we would use for pure CF. In most blends we use a user and movie representation with 20 features (hidden units). In the predictor list, we clearly state, which predictor uses RBM user/movie features.

6.11 KNN Predictor Extraction

The prediction in an item-item KNN is a weighted sum of ratings from k-best neighbors. We extract predictors from the k-best neighbors in the following format:

index	user	item	$k = 1$, item j (1st neighbor)					$k = 2$, item j (2nd neighbor)
1	u	i	$c_{ij} \cdot r_{uj}$	c_{ij}	$\log(s_{ij} + 1)$	r_{uj}	$\exp(-\lvert t_{uj} - t_{ui}\rvert/500)$	...
2	...							

Table 8: KNN predictor extraction

Index is the number of samples in the probe or the qualifying set. c_{ij} is the shrunken Pearson correlation between two items i and j. Shrinkage is set to 200. s_{ij} is the support, the number of common users of item i and j. r_{ij} is the known rating from user u on item j. As listed in Table 8, five values are extracted for every neighbor item j, when $k = 50$ we extract 250 additional predictions.

7 Quiz Blending

The whole ensemble of predictors from *BellKor's Pragmatic Chaos* is blended linearly in the end. The ensemble of predictors includes the individual predictors and the predictors obtained from nonlinear probe blends as described in Section 6.

7.1 Linear Blending on the Quiz Data Set without the Quiz Ratings

For the quiz data set of size N, let $y \in \mathcal{R}^N$ be the *unobserved* vector of true ratings and $\mathbf{x}_1, ..., \mathbf{x}_p \in \mathcal{R}^N$ be p vectors of known predictions. Let $\mathbf{X}$ be the N-by-p matrix with columns $\mathbf{x}_1, ..., \mathbf{x}_p$ For simplicity, and to clarify the impact of selected approximations that follow, assume that the mean [1] has been subtracted from $\mathbf{y}$ and from each column of $\mathbf{X}$.

Our goal is to find the linear combination of $\mathbf{x}_1, ..., \mathbf{x}_p$ that best predicts $\mathbf{y}$. If $\mathbf{y}$ was known, we would use linear regression. That is, we would estimate $\mathbf{y}$ by $\mathbf{X}\hat{\beta}$ where

$$\hat{\beta} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\left(\mathbf{X}^T\mathbf{y}\right) \tag{47}$$

Note that there is no guarantee that the elements of $\hat{\beta}$ will all be non negative or that they will sum to 1.

While it is easy to compute $\left(\mathbf{X}^T\mathbf{X}\right)^{-1}$ (for the full qualifying data set), the missing link is the p-by-1 vector $\left(\mathbf{X}^T\mathbf{y}\right)$. Fortunately, it is possible to estimate each component of this vector with high precision. Consider the j-th element:

$$\sum_u x_{ju} y_u. \tag{48}$$

Simple algebra implies that the above expression can be rewritten as

$$\frac{1}{2}\left[\sum_u y_u^2 + \sum_u x_{ju}^2 - \sum_u \left(y_u - x_{ju}\right)^2\right] \tag{49}$$

Because all predictors are centered, the first term inside the brackets can be closely approximated by N times the quiz set variance, which is known to be 1.274. The second term can be computed exactly (for the full qualifying data). The last term is simply N times the MSE ($RMSE^2$) associated with $\mathbf{x}_j$ for the Quiz data [2]. It is guaranteed to be accurate within about 0.01 percent. Although the constant N is unknown, it cancels out in the formula for $\hat{\beta}$.

However, the final accuracy of $\hat{\beta}$ may be much less if the estimators are highly correlated. Thus, we used ridge regression, as follows:

$$\hat{\beta} = \left(\mathbf{X}^T\mathbf{X} + \lambda N\mathbf{I}\right)^{-1}\left(\mathbf{X}^T\mathbf{y}\right) \tag{50}$$

We used $\lambda = 0.0014$ (in contrast to 0.0010 for the 2008 progress prize).

We are also able to estimate the RMSE for the resulting composite estimator. For $\lambda = 0$, the MSE is simply $1.274 - \text{Var}(\mathbf{X}\hat{\beta})$, where $\text{Var}(\mathbf{X}\hat{\beta})$ is the variance of the linear blend.

For $\lambda > 0$, the calculation is slightly more complicated:

$$\text{MSE}(\mathbf{X}\hat{\beta}) = \sum_i \hat{\beta}_i \left(\text{MSE}(\mathbf{x}_i) - \text{Var}(\mathbf{x}_i)\right) + 1.274\left(1 - \sum_i \hat{\beta}_i\right) + \text{Var}(\mathbf{X}\hat{\beta}) \tag{51}$$

7.2 Estimated Degree of Over Fitting the Quiz Set Relative to the Test Set

As just noted, our final prediction set was a linear combination of prediction sets based on an approximation of ridge regression of ratings on many individual prediction sets for the quiz data. While ordinary least squares regression would minimize the RMSE for the quiz data among all linear combinations, our goal was to minimize RMSE for the test data. It is well known that regression can over fit to training data (the quiz data) leading to poor performance for new data (the test data). Consequently, it was important to understand the extent to which improvement on the quiz RMSE over estimated improvement on the test RMSE.

For ordinary least squares regression, the "bias" of the quiz MSE (MSE_q) as an estimate of the test MSE (MSE_t) is $-2p/N$, where p is the number of prediction sets and N is the size of the quiz data (see

[1]The Quiz mean is known to be 3.674, while its variance is 1.274
see: www.netflixprize.com/community/viewtopic.php?id=503

[2]Because we get the RMSE for the Quiz data, there is some danger of overfitting the Quiz data, leading to poorer predictions for the test data. That danger is greatest if p is large and the $\{\mathbf{x}_j\}$ are highly correlated with each other.

Equation (7.20) of Hastie et al., [10]). Hastie et al. refer to the absolute value of this bias as "optimism". The factor 2 comes in equal parts from the regression over performing on the quiz data relative to the "best" linear model and under performing on the test data. Table 9 shows values of the optimism for the RMSE for various values of p based on RMSE = 0.8558.

p	**Optimism of MSE**	**Optimism of RMSE**	**Optimism** (10^{-4})
200	0.00029	0.00017	1.7
300	0.00043	0.00025	2.5
400	0.00057	0.00033	3.3
500	0.00071	0.00042	4.2
600	0.00086	0.00050	5.0
700	0.00100	0.00058	5.8
800	0.00114	0.00067	6.7
900	0.00129	0.00075	7.5
1000	0.00143	0.00083	8.3

Table 9: Optimism for Estimating the Test Set MSE and RMSE

Of course, while the optimism is constant for a given p, the actual value of $(\text{MSE}_q - \text{MSE}_t)$ varies. First, the magnitude of the over fitting is random, although this source of uncertainty is small. For $p = 800$, the standard deviation around 6.7 points is only 0.3. In contrast, there is substantial variation of the quiz and test MSE's around their expected values due to sampling variability. That is, the ratings in the test set may be inherently harder, or easier, to predict (for known β) than those in the quiz set. Assuming normally-distributed errors for the ratings, we estimate that the standard deviation of the difference between of the quiz and tests RMSE's is about 0.0007. This is consistent with the test RMSE target of 0.8572 being 0.0009 higher than the quiz target.

The idea behind stepwise regression is to reduce over fitting by reducing p. Unfortunately, empirically choosing the "best" 200 predictors out of 800 does not necessarily reduce the optimism value by 75 percent. The problem is that the set of selected predictors is no longer fixed, so that some predictors may be included based on chance correlations on the quiz data that do not generalize to the test data. Consequently, the above formula for the optimism does not hold, making it very difficult to estimate the true value for the optimism.

Ridge regression replaces $\hat{\beta} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$ with $\hat{\beta} = \left(\mathbf{X}^T\mathbf{X} + \lambda N\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}$, shrinking each element of $\hat{\beta}$ towards zero. There are three motivations for ridge regression:

- Regularization, as with everything else in the Netflix Prize. Ridge regression is the Bayesian solution (at least approximately) assuming that each element of β is drawn from a common normal distribution with mean 0 - not an unreasonable assumption.
- To deal with collinearity; this was one of the original motivations for the development of ridge regression and was certainly a big concern for us.
- To help deal with the uncertainty in the elements of $\mathbf{X}^T\mathbf{y}$ due to rounding of the reported RMSE's.

Equation (3.50) of Hastie et al. gives a formula for the effective degrees of freedom of a ridge regression:

$$\text{df}(\lambda) = \text{tr}\left[\mathbf{X}\left(\mathbf{X}^T\mathbf{X} + \lambda N\mathbf{I}\right)^{-1}\mathbf{X}^T\right] \tag{52}$$

For $\lambda = 0.0014$, ridge regression reduced the effective degrees of freedom from 820 to about 521, for an estimated reduction of about 0.0003 in the optimism with little or no observed increase in the quiz RMSE. Larger values of λ appeared to increase the quiz RMSE faster than the corresponding reduction in the optimism.

Nonetheless, even when our quiz RMSE reached an improvement of 10.09%, there was no guarantee that the test RMSE exceeded 10.00%.

8 Discussion

During the nearly 3 years of the Netflix competition, there were two main factors which improved the overall accuracy: The quality of the individual algorithms and the ensemble idea.

On the algorithmic side there was a strong focus on matrix factorization techniques. The big advantage of these methods is, that they can be trained efficiently and predictions can be generated quickly. Additionally, the integration of additional signals and views on the data is easy. So the simple matrix factorization has grown to a big integrated model, which delivers outstanding performance. The competition was not only about matrix factorization, RBMs were successfully shown to yield great results for collaborative filtering. Especially if RBMs are combined with KNN models. The integration of additional signals, such as time is not that easy.

Over the 3 years of the competition, a lot of effects were found in the data. During the first year the biggest discovery was the binary information, accounting for the fact that people do not select movies for rating at random. In the second year there was a focus on temporal effects. Small long term effects and stronger short term effects, especially the one day effect, was very strong [14]. It is hard to say whether this effect is grounded in multiple users sharing the same account, or the changing mood of a person.

The other main driving force in the competition was the ensemble idea. The ensemble idea was part of the competition from the beginning and evolved over time. In the beginning, we used different models with different parametrization and a linear blending. The models were trained individually and the meta parameters got optimized to reduce the RMSE of the individual model. The linear blend was replaced by a nonlinear one, a neural network. This was basically the solution for the progress prize 2008, a ensemble of independently trained and tuned predictors, and a neural network for the blending. In fall 2008, we realized that training and optimizing the predictors individually is not optimal. Best blending results are achieved when the whole ensemble has the right tradeoff between diversity and accuracy. So we started to train the predictors sequentially and stopped the training when the blending improvement was best. Also the meta parameters were tuned, to achieve best blending performance. The next step in the evolution of the ensemble idea was to replace the single neural network blend by an ensemble of blends. In order to maximize diversity within the blending ensemble, the blends used different subsets of predictors and different blending methods. Figure 6 shows the RMSE improvements compared to the number of predictors. Within the first predictors there are a lot of different blends (the first 18 are listed in Appendix C). This clearly shows that the diverse set of nonlinear probe blends is an important part of our solution.

The Netflix prize boosted the collaborative filtering research, because it enabled direct comparison of results and encouraged an open discussion of ideas in the forum and on conferences. It would be great to see similar competitions in future.

References

[1] Netflix Prize homepage. Website, 2006. `http://www.netflixprize.com`.

[2] R. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *IEEE International Conference on Data Mining*. KDD-Cup07, 2007.

[3] R. Bell, Y. Koren, and C. Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 95–104, New York, NY, USA, 2007. ACM.

[4] R. M. Bell, Y. Koren, and C. Volinsky. The BellKor solution to the Netflix Prize, October 2007.

[5] R. M. Bell, Y. Koren, and C. Volinsky. The BellKor 2008 solution to the Netflix Prize, October 2008.

[6] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[7] J. Friedman. Greedy function approximation: A gradient boosting machine. Technical report, Salford Systems, 1999.

[8] J. Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 2002.

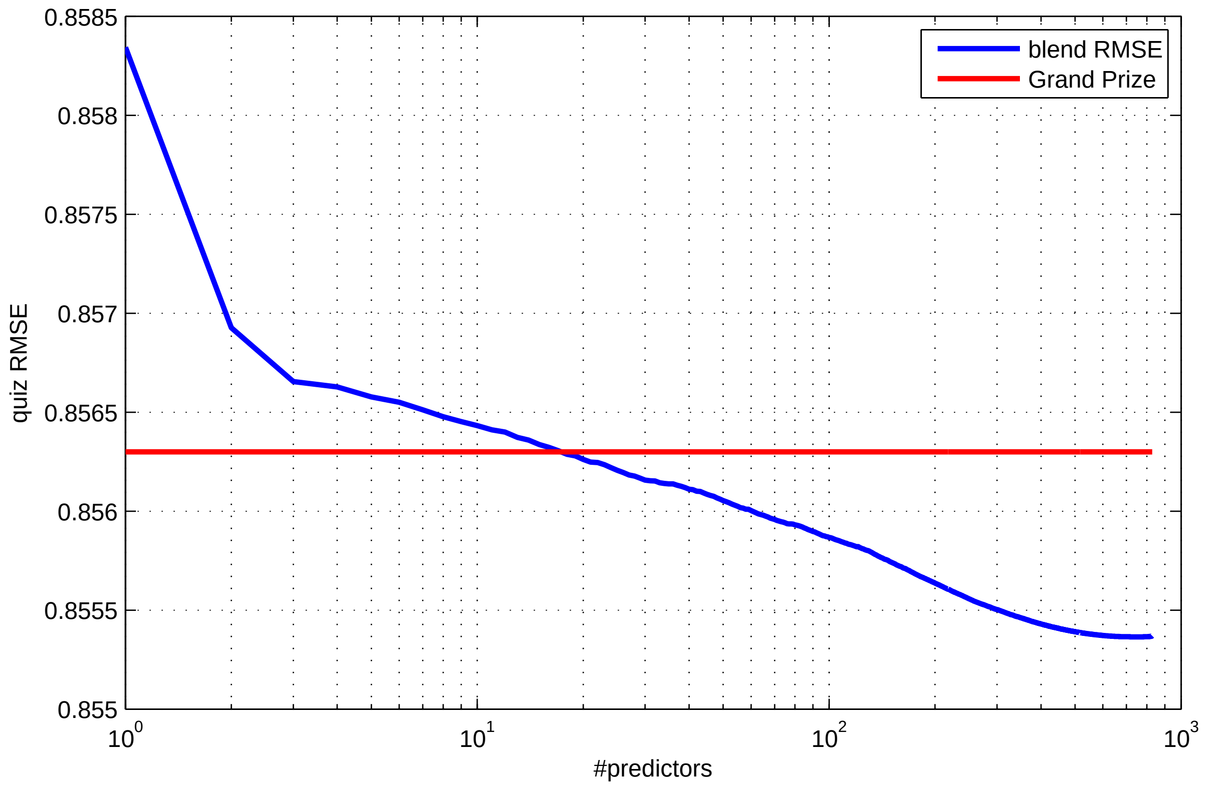

Figure 6: How many results are really needed? With 18 results we breach the 10% (RMSE 0.8563) barrier. Within these results there are 11 nonlinear probe blends and 7 unblended predictors. An ordered list of these predictors can be found in Appendix C.

[9] I. Guyon. *Kernel Ridge Regression tutorial*, accessed Aug 31, 2009. `http://clopinet.com/isabelle/Projects/ETH/KernelRidge.pdf`.

[10] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer, 1st edition, 2001.

[11] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434, New York, NY, USA, 2008. ACM.

[12] Y. Koren. The BellKor solution to the Netflix Grand Prize, 2009.

[13] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. *Proceedings of KDD Cup and Workshop*, 2007.

[14] G. Potter. Putting the collaborator back into collaborative filtering. In *KDD Workshop at SIGKDD 08*, August 2008.

[15] R. Salakhutdinov, A. Mnih, and G. E. Hinton. Restricted boltzmann machines for collaborative filtering. In *ICML*, pages 791–798, 2007.

[16] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Matrix factorization and neighbor based algorithms for the netflix prize problem. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*, pages 267–274, New York, NY, USA, 2008. ACM.

[17] A. Töscher and M. Jahrer. The BigChaos solution to the Netflix Prize 2008. Technical report, commendo research & consulting, October 2008.

[18] A. Töscher, M. Jahrer, and R. Legenstein. Improved neighborhood-based algorithms for large-scale recommender systems. In *KDD Workshop at SIGKDD 08*, August 2008.

[19] M. Wu. Collaborative filtering via ensembles of matrix factorizations. *Proceedings of KDD Cup and Workshop*, 2007.

A Predictor List

This section lists all results produced by team BigChaos and blends of results by Pragmatic Theory and BellKor. All listed results are quiz RMSEs.

A.1 BigChaos Progress Prize 2008 Results: PP-*

60 predictors were taken from the BigChaos Progress Prize 2008 report, the rest was dropped. The reference number and the algorithm in brackets is the one, which is listed in [17].

PP-01 rmse=0.9028
Corresponds to 3. (BasicSVD)

PP-02 rmse=0.9066
Corresponds to 4. (BasicSVD)

PP-03 rmse=0.9045
Corresponds to 6. (BasicSVD)

PP-04 rmse=0.9275
Corresponds to 7. (BasicSVD)

PP-05 rmse=0.917
Corresponds to 8. (BasicSVD)

PP-06 rmse=0.9143
Corresponds to 9. (BasicSVD)

PP-07 rmse=0.9592
Corresponds to 10. (SVD-AUF)

PP-08 rmse=0.9038
Corresponds to 11. (SVD-ALS)

PP-09 rmse=0.9842
Corresponds to 12. (TimeSVD)

PP-10 rmse=0.9378
Corresponds to 14. (TimeSVD)

PP-11 rmse=0.9688
Corresponds to 15. (TimeSVD)

PP-12 rmse=1.0419
Corresponds to 16. (TimeSVD)

PP-13 rmse=0.8981
Corresponds to 18. (NAMF)

PP-14 rmse=0.9070
Corresponds to 26. (RBMV5)

PP-15 rmse=0.9117
Corresponds to 28. (RBMV5)

PP-16 rmse=0.9056
Corresponds to 29. (RBMV5)

PP-17 rmse=0.9039
Corresponds to 30. (RBMV5)

PP-18 rmse=0.9044
Corresponds to 31. (RBMV5)

PP-19 rmse=0.9045
Corresponds to 32. (RBMV5)

PP-20 rmse=0.9041
Corresponds to 34. (RBMV5)

PP-21 rmse=0.9045
Corresponds to 35. (RBMV5)

PP-22 rmse=0.9066
Corresponds to 36. (RBMV5)

PP-23 rmse=0.9008
Corresponds to 37. (RBMV6)

PP-24 rmse=0.9229
Corresponds to 38. (KNN-BASIC)

PP-25 rmse=0.9013
Corresponds to 39. (KNN-BASIC)

PP-26 rmse=0.9151
Corresponds to 41. (KNNMovieV7)

PP-27 rmse=0.9013
Corresponds to 42. (KNNMovieV3)

PP-28 rmse=0.8942
Corresponds to 43. (KNNMovie)

PP-29 rmse=0.9042
Corresponds to 44. (KNNMovieV4)

PP-30 rmse=0.9000
Corresponds to 45. (KNNMovieV7)

PP-31 rmse=0.8832
Corresponds to 46. (KNNMovieV3)

PP-32 rmse=0.8934
Corresponds to 47. (KNNMovieV3)

PP-33 rmse=0.8843
Corresponds to 48. (KNNMovieV6)

PP-34 rmse=0.8905
Corresponds to 49. (KNNMovieV3)

PP-35 rmse=0.8910
Corresponds to 50. (KNNMovieV3)

PP-36 rmse=0.8900

Corresponds to 51. (KNNMovieV3)

PP-37 rmse=0.9102
Corresponds to 52. (KNNMovieV3)

PP-38 rmse=0.8852
Corresponds to 53. (KNNMovieV3)

PP-39 rmse=0.8915
Corresponds to 56. (KNNMovieV3)

PP-40 rmse=0.8929
Corresponds to 57. (KNNMovieV3)

PP-41 rmse=0.9112
Corresponds to 58. (KNNMovieV4)

PP-42 rmse=0.9300
Corresponds to 62. (ROS)

PP-43 rmse=0.9304
Corresponds to 63. (ROS)

PP-44 rmse=0.8970
Corresponds to 65. (ROS)

PP-45 rmse=0.9226
Corresponds to 67. (ROS)

PP-46 rmse=0.9366
Corresponds to 69. (AFM)

PP-47 rmse=0.885
Corresponds to 74. (GTE)

PP-48 rmse=0.8849
Corresponds to 76. (GTE)

PP-49 rmse=0.9482
Corresponds to 77. (GTE)

PP-50 rmse=0.9559
Corresponds to 78. (GTE)

PP-51 rmse=0.955
Corresponds to 79. (GTE)

PP-52 rmse=0.945
Corresponds to 80. (GTE)

PP-53 rmse=0.8835
Corresponds to 81. (GTE)

PP-54 rmse=0.8872
Corresponds to 82. (CTD)

PP-55 rmse=0.8861
Corresponds to 83. (CTD)

PP-56 rmse=0.8865
Corresponds to 84. (CTD)

PP-57 rmse=0.8834
Corresponds to 85. (CTD)

PP-58 rmse=0.8845
Corresponds to 86. (MTD)

PP-59 rmse=0.913
Corresponds to 89. (NN)

PP-60 rmse=0.9178
Corresponds to 91. (NN)

A.2 Optimize the Predictors Individually on the Probe Set: OP-*

These predictors got trained in the same way, as for the progress prize 2008. They are trained and optimized individually in order to minimize the probe set RMSE.

A.2.1 Basic SVD

OP-01 rmse=0.9108
BasicSVD, Residual: 1GE, k=64, $\eta = 0.001$, $\lambda = 0.019$, $\alpha = 2.0$

OP-02 rmse=0.9126
BasicSVD, Residual: 14GE, k=100, $\eta = 0.002$, $\lambda = 0.01$, $\alpha = 2.0$

OP-03 rmse=0.9066
BasicSVD, Residual: OP-18, k=300, $\eta = 0.001$, $\lambda = 0.007$, $\alpha = 2.0$

OP-04 rmse=0.9177
BasicSVD, Residual: no, force non-negative weights, k=100, $\eta = 0.0005$, $\lambda = 0.003$, $\alpha = 2.0$

A.2.2 Neighborhood Aware Matrix Factorization

OP-05 rmse=0.8993
NAMF, Residual: 1GE, k=300, $\eta = 0.004$, $\lambda = 0.02$, $bestCorr = 50$ [Pearson on 10GE]

OP-06 rmse=0.8856
NAMF, Residual: 1GE, k=100, $\eta = 0.002$, $\lambda = 0.02$, $bestCorr = 50$ [Pearson on 10GE], For this predictor we inserted a predicted qualifying set (RMSE of 0.8736) into the training ratings.

A.2.3 RBMV6

OP-07 rmse=0.9270
RBMV6, Residual: 10GE, 50 hidden units, $\eta = 0.002$, $\lambda = 0.00035$

A.2.4 Movie KNN

OP-08 rmse=0.9154
KNNMovieV3, Residual: 10GE, k=48, Pearson correlation, $\alpha = 600$, $\beta = 540$, $\gamma = -2.38$, $\delta = 10.8$

OP-09 rmse=0.8987
KNNMovieV3, Residual: PP-46, k=34, Pearson correlation, $\alpha = 292$, $\beta = 787$, $\gamma = -2.47$, $\delta = 7.8$

OP-10 rmse=0.8973
KNNMovie, Residual: BasicSVD 300 dim, k=80, Pearson correlation

OP-11 rmse=0.8851
KNNMovieV3, Residual: blended RBMV3, k=55, Pearson correlation, $\alpha = 842$, $\beta = 540$, $\gamma = -2.43$, $\delta = 12.7$

A.2.5 Regression on Similarity

OP-12 rmse=0.936
ROS factorized, Residual: 6 GE, on movie-side, k=20, η=3.5e-3, λ=1e-3

OP-13 rmse=0.9241
ROS, unfactorized, Residual: 10 GE, η=1, λ =1e-4

A.2.6 Global Time Effects

OP-14 rmse=0.9545
GTE, Residual: no, 8 effects

OP-15 rmse=0.8884
GTE, Residual: PP-34, 1 effects

A.2.7 Customer Time Dep Model

OP-16 rmse=1.0429
CTD, Residual: OP-18, Linear Weight, $\alpha = 35$, $\beta = 23$, $\gamma = 122$

OP-17 rmse=0.898
CTD, Residual: BasicSVD 300 dim, Near Gaussian Weight, $\alpha = 5693$, $\beta = 7$, $\gamma = 21$

OP-18 rmse=1.0476
CTD, Residual: no, Near Gaussian Weight, $\alpha = 15$, $\beta = 5$, $\gamma = 0.55$

OP-19 rmse=0.9537
CTD, Residual: 10 GE, Near Gaussian Weight, $\alpha = 10400$, $\beta = 8$, $\gamma = 0.49$

A.2.8 Movie Time Dep Model

OP-20 rmse=0.9514
MTD, Residual: OP-19, Near Gaussian Weight, $\alpha = 0.15$, $\beta = 85$, $\gamma = 16$

A.2.9 Neural Network

OP-21 rmse=0.9233
NN, Residual: no, 1-layer nonNegative (10 neurons), η=0.025, λ=8e-4

A.3 Optimize the Predictors on the Blend: OB-*

Following results are trained in the order they get listed here. Every predictor uses all preceding predictors and a constant predictor for determining when the blending improvement is optimal (Section 3.2). So the gradient based methods stop the training at the point with the optimal blending improvement. The automatic parameter tuning algorithms use the blending improvement in order to tune parameters. For example the parameter tuner adjusts the parameters of predictor OB-14 to achieve the best blending result with the preceding predictors OB-01 to OB-14 and a constant predictor. This means the parameters of a predictor depend on the preceding predictors, so we reported all parameters for every predictor in order to make the predictors individually reproducible.

OB-01 rmse=1.0533
GE, Residual: no, 1 effect, $\alpha_1 = 22.195$

OB-02 rmse=0.9597
GE, Residual: no, 10 effects, $\alpha_1 = 22.1671$, $\alpha_2 = 7.08907$, $\alpha_3 = 508.018$, $\alpha_4 = 146.739$, $\alpha_5 = 3894.24$, $\alpha_6 = 362.944$, $\alpha_7 = 69.6393$, $\alpha_8 = 142.646$, $\alpha_9 = 184743$, $\alpha_{10} = 1.21417e-08$

OB-03 rmse=0.9016
SVD, Residual: OB-01, $k = 300$, $\eta_i = 0.0008$, $\eta_u = 0.0008$, $\lambda_i = 0.01$, $\lambda_u = 0.01$, 158 epochs

OB-04 rmse=0.9201
MovieKNNV3, Residual: OP-02, Pearson correlation, $K = 44$, $\alpha = 91.6$, $\beta = 107.7$, $\gamma = -3.05$, $\delta = 9.9$

OB-05 rmse=0.9136
RBMV3, Residual: no, 50 hidden units, $\eta = 0.0003$, $\lambda = 0.0002$

OB-06 rmse=0.8904
MovieKNNV3, Residual: OB-05, Pearson correlation, $K = 28$, $\alpha = 370.9$, $\beta = 244.6$, $\gamma = -2.7$, $\delta = 10.9$

OB-07 rmse=0.971
GE, Residual: no, 16 effects, $\alpha_1 = 3.55142$, $\alpha_2 = 27.6949$, $\alpha_3 = 1.24793e+07$, $\alpha_4 = 390.232$, $\alpha_5 = 1.88449e+06$, $\alpha_6 = 1.52951e+06$, $\alpha_7 = 162.211$, $\alpha_8 = 254.251$, $\alpha_9 = 37552.7$, $\alpha_{10} = 2.64335e-06$, $\alpha_{11} = 3.13113e+06$, $\alpha_{12} = 249946$, $\alpha_{13} = 49129.8$, $\alpha_{14} = 3294.95$, $\alpha_{15} = 3.40405e+06$, $\alpha_{16} = 266915$

OB-08 rmse=0.9023
SVD, Residual: no, $k = 500$, $\eta_i = 0.001$, $\eta_u = 0.001$, $\lambda_i = 0.02$, $\lambda_u = 0.02$, force nonNegWeights, 408 epochs

OB-09 rmse=0.8932
GTE, Residual: OB-08, first 20 effects, $\alpha_0 = 1.03281$, $\sigma_0 = 0.517528$, $\alpha_1 = 1.85286$, $\sigma_1 = 35.7954$, $\alpha_2 = 0.445738$, $\sigma_2 = 0.741892$, $\alpha_3 = 1.32741e-05$, $\sigma_3 = 1198.84$, $\alpha_4 = 2.77125e-06$, $\sigma_4 = 40.995$, $\alpha_5 = 63.8279$, $\sigma_5 = 7.11708e+09$, $\alpha_6 = 13.2191$, $\sigma_6 = 14.2995$, $\alpha_7 = 4.54058$, $\sigma_7 = 29.8179$, $\alpha_8 = 2449.25$, $\sigma_8 = 78.0966$, $\alpha_9 = 5.25749$, $\sigma_9 = 12.7134$, $\alpha_{10} = 0.0306135$, $\sigma_{10} = 15.057$, $\alpha_{11} = 0.104538$, $\sigma_{11} = 5.80681$, $\alpha_{12} = 250657$, $\sigma_{12} = 3.36933e+08$, $\alpha_{13} = 11768.4$, $\sigma_{13} = 167.355$, $\alpha_{14} = 24.3031$, $\sigma_{14} = 57.5974$, $\alpha_{15} = 0.0062943$, $\sigma_{15} = 5.56049$, $\alpha_{16} = 294.705$, $\sigma_{16} = 6.45491$, $\alpha_{17} = 14.5114$, $\sigma_{17} = 10.4752$, $\alpha_{18} = 1.32223$, $\sigma_{18} = 5.65513$, $\alpha_{19} = 0.000446302$, $\sigma_{19} = 19.747$, $\alpha_{20} = 0.232973$, $\sigma_{20} = 542.67$

OB-10 rmse=0.9133
BasicSVD, Residual: OB-01, $k = 300$, $exponent = 1.9$, $\eta_i = 0.0005$, $\eta_u = 0.0005$, $\lambda_i = 0.011$, $\lambda_u = 0.011$, 110 epochs, (BasicSVD is explained in [17]).

OB-11 rmse=0.9608
SVD-ALS, Residual: no, $k = 5$, movieSolver=nonNeg, userSolver=nonNeg, $\lambda_i = 0.15$, $\lambda_u = 0.15$, 479 epochs

OB-12 rmse=0.9933
WE, Residual: OB-01, regU=7.61098 regM=4.27357e+10 regGlob=1.25265e+10 powGlob=0.00033654 powM=0.0310857 powU=3.445

OB-13 rmse=0.9093
SVD, Residual: no, $k = 1000$, $\eta_i = 0.01$, $\eta_u = 0.008$, $\lambda_i = 0.01$, $\lambda_u = 0.04$, force nonNegativeWeights, with user and movie biases, 60 epochs

OB-14 rmse=0.9197
MovieKNNV3, Residual: OB-11, Spearman's rank correlation, $K = 14$, $\alpha = 280$, $\beta = 3572$, $\gamma = -3$, $\delta = 8.34$

OB-15 rmse=0.8878
CTD, Residual: OB-06, weightingFctn=GAUSSIAN-WEIGHT, $\alpha = 77.3653$, $\beta = 11.43$ $\gamma = 0.971957$

OB-16 rmse=1.0047
SVD-ALS, Residual: OB-01, $k = 300$, $\lambda_i = 0.01$, $\lambda_u = 0$, movieSolver=PINV, userSolver=PINV, 9 epochs

OB-17 rmse=0.9135
MovieKNNV3, Residual: OB-16, MSE correlation, $K = 34$, $\alpha = 204.8$, $\beta = 1889$, $\gamma = -3.8$, $\delta = 7$

OB-18 rmse=0.9036
RBMV5, Residual: OB-07, 300 hidden units, $\eta = 0.0002$, $\lambda = 0.0007$

OB-19 rmse=0.8983
MovieKNNV3, Residual: OB-18, Pearson correlation, $K = 86$, $\alpha = 20.3$, $\beta = 38.9$, $\gamma = -4.2$, $\delta = 9.2$

OB-20 rmse=0.8925
MovieKNNV3, Residual: OB-09, MSE correlation, $K = 150$, $\alpha = 2170.7$, $\beta = 165.4$, $\gamma = -2.6$, $\delta = 1$

OB-21 rmse=0.8923
MovieKNNV3, Residual: OB-09, Pearson correlation, $K = 203$, $\alpha = 657.8$, $\beta = 3043$, $\gamma = -4.17$, $\delta = 31.8$

OB-22 rmse=0.9774
MovieKNNV3, Residual: no, Pearson correlation, $K = 94$, $\alpha = 137.2$, $\beta = 1963$, $\gamma = -3$, $\delta = 8$

OB-23 rmse=0.9317
NSVD1, Residual: no, $k = 40$, $\eta_i = 1e-4$, $\eta_u = 1e-4$, $\eta_{\mu_i} = 2e-3$, $\eta_{\mu_u} = 2e-3$, $\lambda_i = 1e-3$, $\lambda_u = 1e-3$, $\lambda_{\mu_i} = 2e-3$, $\lambda_{\mu_u} = 2e-3$, muliply all η with 0.9 from epoch 40, 96 epochs

OB-24 rmse=0.899
SVD-AUF, Residual: OB-23, adaptiveUserFactorMode=KRR, kernelType=polynomial, $\lambda = 29.1733$, $\alpha = 0.39171$, $\beta = 3.64831$

OB-25 rmse=0.9517
SVD-AUF, Residual: OB-11, adaptiveUserFactorMode=dateSim, solver=PINV, $\lambda = 0.00243$, $\gamma 1.31035$, $\beta = 0.356343$, $\kappa = -0.0165$

OB-26 rmse=0.945
RBMV3, Residual: no, 10 hidden units, $\eta = 0.002$, $\lambda = 0.0002$

OB-27 rmse=1.0202
GTE, Residual: no, 24 effects, $\alpha_0 = 0.756448$, $\sigma_0 = 63.3648$, $\alpha_1 = 0.000341077$, $\sigma_1 = 783.599$, $\alpha_2 = 7.66667$, $\sigma_2 = 2.22136$, $\alpha_3 = 441.092$, $\sigma_3 = 2.39512$, $\alpha_4 = 7.9987$, $\sigma_4 = 2.30299$, $\alpha_5 = 2.19463$, $\sigma_5 = 2.50274$, $\alpha_6 = 0.40246$, $\sigma_6 = 3.79979$, $\alpha_7 = 0.00869847$, $\sigma_7 = 0.190355$, $\alpha_8 = 3761.7$, $\sigma_8 = 49.564$, $\alpha_9 = 0.0289626$, $\sigma_9 = 2.16438$, $\alpha_{10} = 0.174869$, $\sigma_{10} = 7.87177$, $\alpha_{11} = 0.723074$, $\sigma_{11} = 0.258699$, $\alpha_{12} = 358857$, $\sigma_{12} = 8.36612$, $\alpha_{13} = 0.415689$, $\sigma_{13} = 0.342104$, $\alpha_{14} = 0.719413$, $\sigma_{14} = 5.46462$, $\alpha_{15} = 0.278113$, $\sigma_{15} = 2.01964$, $\alpha_{16} = 2.9132$, $\sigma_{16} = 0.540534$, $\alpha_{17} = 0.291892$, $\sigma_{17} = 5.01496$, $\alpha_{18} = 1.06935$, $\sigma_{18} = 14.0819$, $\alpha_{19} = 0.000909$, $\sigma_{19} = 8.7371$, $\alpha_{20} = 3.33668$, $\sigma_{20} = 3.6388$, $\alpha_{21} = 124644$, $\sigma_{21} = 2.39229$, $\alpha_{22} = 2.19671$, $\sigma_{22} = 4.83657$, $\alpha_{23} = 3.74613$, $\sigma_{23} = 1.37054$, $\alpha_{24} = 1.59059$, $\sigma_{24} = 1.59185$

OB-28 rmse=0.9226
SVD-AUF, Residual: OB-08, adaptiveUserFactorMode=KRR, kernelType=extended-polynomial, $\lambda = 7.08585$, $\alpha = 0.109493$, $\gamma = 0.495312$, $\beta = 0.00834611$

OB-29 rmse=0.915
SVD-AUF, Residual: OB-13, adaptiveUserFactorMode=KRR, kernelType=polynomial, $\lambda = 12.6179$, $\alpha = 0.790135$, $\beta = 5.13577$

OB-30 rmse=0.9283
NSVD1, Residual: OB-27, $k = 1000$, $\eta_i = 1e-4$, $\eta_u = 1e-4$, $\eta_{\mu_i} = 2e-3$, $\eta_{\mu_u} = 2e-3$, $\lambda_i = 1.5e-3$, $\lambda_u = 1.5e-3$, $\lambda_{\mu_i} = 2e-3$, $\lambda_{\mu_u} = 2e-3$, muliply all η with 0.95 from epoch 35, 150 epochs

OB-31 rmse=0.9022
MovieKNNV3, Residual: OB-30, Pearson correlation, $K = 35$, $\alpha = 291.3$, $\beta = 571$, $\gamma = -2.9$, $\delta = 18.9$

OB-32 rmse=0.9101
RBMV3, Residual: no, 250 hidden units, $\eta = 0.002$, $\lambda = 0.00025$

OB-33 rmse=0.9001
MovieKNNV3, Residual: OB-32, Pearson correlation, $K = 192$, $\alpha = 72.3$, $\beta = 31.3$, $\gamma = -2.6$, $\delta = 11.8$

OB-34 rmse=0.927
SVD-Time, Residual: globalMean, $k = 50$, $N = 5$, $M = 5$, $\eta_i = 0.002$, $\eta_u = 0.002$, $\lambda_i = 0.01$, $\lambda_u = 0.03$, 96 epochs

OB-35 rmse=0.9248
MovieKNNV3, Residual: OB-34, Pearson correlation, $K = 69$, $\alpha = 1036.8$, $\beta = 1678$, $\gamma = -3.6$, $\delta = 19.9$

OB-36 rmse=0.9104
SVD-Time, Residual: OB-26, $k = 20$, $N = 4$, $M = 4$, $\eta_i = 0.002$, $\eta_u = 0.002$, $\lambda_i = 0.02$, $\lambda_u = 0.02$, 158 epochs

OB-37 rmse=0.9082
RBMV3, Residual: no, 250 hidden units, $\eta = 0.002$, $\lambda = 0.0004$

OB-38 rmse=0.8978
MovieKNNV3, Residual: OB-37, Spearman's rank correlation, $K = 225$, $\alpha = 112.6$, $\beta = 1922$, $\gamma = -3.2$, $\delta = 9.3$

OB-39 rmse=0.9213
NSVD1, Residual: OB-02, $k = 800$, $\eta_i = 1e-4$, $\eta_u = 1e-4$, $\eta_{\mu_i} = 2e-3$, $\eta_{\mu_u} = 2e-3$, $\lambda_i = 5e-4$, $\lambda_u = 5e-4$, $\lambda_{\mu_i} = 2e-3$, $\lambda_{\mu_u} = 2e-3$, muliply all η with 0.97 from epoch 35, 200 epochs

OB-40 rmse=0.9091
RBMV3, Residual: no, 300 hidden units, $\eta = 0.002$, $\lambda = 0.0006$

OB-41 rmse=0.913
RBMV5, Residual: globalMean, 1000 hidden units, $\eta = 0.0002$, $\lambda = 0.0009$

OB-42 rmse=0.8998
MovieKNNV3, Residual: OB-39, Pearson correlation, $K = 13$, $\alpha = 658$, $\beta = 2480$, $\gamma = -2.6$, $\delta = 7.5$

OB-43 rmse=0.8971
SVD-AUF, Residual: OB-30, adaptiveUserFactorMode=KRR, kernelType=extended-polynomial, $\lambda = 4.78376$, $\alpha = 0.657533$, $\gamma = 0.720031$, $\beta = 3.27554$

OB-44 rmse=0.9245
GE, Residual: OB-35, 16 effects, $\alpha_1 = 374.977$, $\alpha_2 = 8.90702e-05$, $\alpha_3 = 2535.9$, $\alpha_4 = 900.414$, $\alpha_5 = 1.04115e-05$, $\alpha_6 = 2087.92$, $\alpha_7 = 131.291$, $\alpha_8 = 3173.84$, $\alpha_9 = 1.45471e-06$, $\alpha_{10} = 6.40823e-08$, $\alpha_{11} = 4451.15$, $\alpha_{12} = 274.423$, $\alpha_{13} = 1020.64$, $\alpha_{14} = 0.00758424$, $\alpha_{15} = 3858.57$, $\alpha_{16} = 0.00346888$

OB-45 rmse=0.8998
MovieKNNV3, Residual: OB-40, Spearman's rank correlation, $K = 38$, $\alpha = 667.6$, $\beta = 255.5$, $\gamma = -1.39$, $\delta = 3.3$

OB-46 rmse=0.8958
MovieKNNV3, Residual: OB-41, Pearson correlation, $K = 60$, $\alpha = 804$, $\beta = 231$, $\gamma = -2.6$, $\delta = 17$

OB-47 rmse=0.9777
NSVD2, Residual: no, $k = 50$, $\eta_i = 2e-3$, $\eta_u = 2e-3$, $\eta_{\mu_i} = 2e-3$, $\eta_{\mu_u} = 2e-3$, $\lambda_i = 5e-4$, $\lambda_u = 5e-4$, $\lambda_{\mu_i} = 1e-4$, $\lambda_{\mu_u} = 1e-4$, 3 epochs

OB-48 rmse=0.9344
NSVDD, Residual: no, $k = 200$, $\eta_i = 1e-3$, $\eta_u = 1e-3$, $\eta_{\mu_i} = 1e-3$, $\eta_{\mu_u} = 1e-3$, $\lambda_i = 1e-3$, $\lambda_u = 1e-3$, $\lambda_{\mu_i} = 1e-3$, $\lambda_{\mu_u} = 1e-3$, muliply all η with 0.95 from epoch 30, 120 epochs

OB-49 rmse=0.9205
MovieKNNV3, Residual: OB-47, Pearson correlation, $K = 17$, $\alpha = 274.6$, $\beta = 1712$, $\gamma = -2.9$, $\delta = 20$

OB-50 rmse=0.9049
MovieKNNV3, Residual: OB-48, Pearson correlation, $K = 24$, $\alpha = 385.8$, $\beta = 488$, $\gamma = -2.99$, $\delta = 9.5$

OB-51 rmse=0.9016
GE, Residual: OB-50, 16 effects, $\alpha_1 = 172.971$, $\alpha_2 = 633.616$, $\alpha_3 = 278.061$, $\alpha_4 = 479.608$, $\alpha_5 = 571.286$, $\alpha_6 = 17550.3$, $\alpha_7 = 41.7332$, $\alpha_8 = 107.62$, $\alpha_9 = 8089.62$, $\alpha_{10} = 1510.79$, $\alpha_{11} = 3.28767e-05$, $\alpha_{12} = 38.4236$, $\alpha_{13} = 78.8427$, $\alpha_{14} = 0.00298945$, $\alpha_{15} = 695035$, $\alpha_{16} = 0.0409925$

OB-52 rmse=0.9063
RBMV3, Residual: no, 300 hidden units, $\eta = 0.0012$, $\lambda = 0.0003$

OB-53 rmse=0.9178
SVD-Time, Residual: OB-49, $k = 70$, $N = 6$, $M = 3$, $\eta_i = 0.003$, $\eta_u = 0.003$, $\lambda_i = 0.025$, $\lambda_u = 0.045$, 106 epochs

OB-54 rmse=0.9641
SVD-Time, Residual: OB-01, $k = 20$, $N = 6$, $M = 8$, $\eta_i = 0.002$, $\eta_u = 0.002$, $\lambda_i = 0.01$, $\lambda_u = 0.02$, 22 epochs

OB-55 rmse=0.8971
MovieKNNV3, Residual: OB-52, Pearson correlation, $K = 33$, $\alpha = 612.5$, $\beta = 296$, $\gamma = -0.99$, $\delta = 34.5$

OB-56 rmse=0.9298
MovieKNNV3, Residual: OB-54, Pearson correlation, $K = 168$, $\alpha = 1012$, $\beta = 523$, $\gamma = -3.2$, $\delta = 7$

OB-57 rmse=0.9017
GTE, Residual: OB-52, 24 effects, $\alpha_0 = 1.00571$, $\sigma_0 = 3.61101$, $\alpha_1 = 5.14906$, $\sigma_1 = 15.2058$, $\alpha_2 =$

0.253872, $\sigma_2 = 3.85881$, $\alpha_3 = 6.44612$, $\sigma_3 = 18.5493$, $\alpha_4 = 0.49935$, $\sigma_4 = 3.63728$, $\alpha_5 = 0.430758$, $\sigma_5 = 6.03448$, $\alpha_6 = 30.0492$, $\sigma_6 = 9.40765$, $\alpha_7 = 1.06677$, $\sigma_7 = 26.7225$, $\alpha_8 = 1.04277e-05$, $\sigma_8 = 2.49227$, $\alpha_9 = 49.0741$, $\sigma_9 = 0.262239$, $\alpha_{10} = 5.13061$, $\sigma_{10} = 17.0218$, $\alpha_{11} = 1.7518$, $\sigma_{11} = 3.44103$, $\alpha_{12} = 0.193035$, $\sigma_{12} = 5.10717$, $\alpha_{13} = 2950.42$, $\sigma_{13} = 2.07498$, $\alpha_{14} = 17.3151$, $\sigma_{14} = 14.2747$, $\alpha_{15} = 0.0108529$, $\sigma_{15} = 5.62521$, $\alpha_{16} = 8.30796$, $\sigma_{16} = 5.64531$, $\alpha_{17} = 5.79247$, $\sigma_{17} = 2.20745$, $\alpha_{18} = 0.318347$, $\sigma_{18} = 1.17436$, $\alpha_{19} = 2.11259e-10$, $\sigma_{19} = 11.1742$, $\alpha_{20} = 606.263$, $\sigma_{20} = 1.38733$, $\alpha_{21} = 1666.74$, $\sigma_{21} = 1.21059$, $\alpha_{22} = 11.9562$, $\sigma_{22} = 3.85147$, $\alpha_{23} = 12.9199$, $\sigma_{23} = 10.8872$, $\alpha_{24} = 81.5603$, $\sigma_{24} = 14.5512$

OB-58 rmse=0.8941
MovieKNNV3, Residual: OB-57, Pearson correlation, $K = 34$, $\alpha = 2014$, $\beta = 366$, $\gamma = -1.46$, $\delta = 47.2$

OB-59 rmse=0.8932
MovieKNNV3, Residual: OB-58, Spearman's rank correlation, $K = 74$, $\alpha = 725.6$, $\beta = 94.8$, $\gamma = -3.6$, $\delta = 12.4$

OB-60 rmse=1.0207
MMMF, Residual: no, $k = 200$, $\eta_i = 0.001$, $\eta_u = 0.001$, $\lambda_i = 0.01$, $\lambda_u = 0.01$, 185 epochs

OB-61 rmse=0.9453
NSVD1, Residual: no, $k = 40$, $\eta_i = 1e-4$, $\eta_u = 1e-4$, $\eta_{\mu_i} = 2e-3$, $\eta_{\mu_u} = 2e-3$, $\lambda_i = 1e-3$, $\lambda_u = 1e-3$, $\lambda_{\mu_i} = 2e-3$, $\lambda_{\mu_u} = 2e-3$, muliply all η with 0.9 from epoch 40, 97 epochs

OB-62 rmse=0.9173
MovieKNNV3, Residual: OB-61, Pearson correlation, $K = 42$, $\alpha = 177.6$, $\beta = 23.7$, $\gamma = -2.7$, $\delta = 16.7$

OB-63 rmse=0.9651
GE, Residual: no, 16 effects, $\alpha_1 = 4.76342e-06$, $\alpha_2 = 2.2696$, $\alpha_3 = 135.319$, $\alpha_4 = 103.14$, $\alpha_5 = 0.000150371$, $\alpha_6 = 585.235$, $\alpha_7 = 158997$, $\alpha_8 = 61.2179$, $\alpha_9 = 332.361$, $\alpha_{10} = 1.38317e-07$, $\alpha_{11} = 119256$, $\alpha_{12} = 124.482$, $\alpha_{13} = 1553.02$, $\alpha_{14} = 9.50186e-07$, $\alpha_{15} = 125357$, $\alpha_{16} = 3.36691e+11$

OB-64 rmse=0.9373
MovieKNNV3, Residual: OB-63, Pearson correlation, $K = 21$, $\alpha = 558.7$, $\beta = 1246$, $\gamma = -3.3$, $\delta = 5.4$

OB-65 rmse=0.9331
GE, Residual: OB-64, 16 effects, $\alpha_1 = 1.34006e-07$, $\alpha_2 = 5.3791e-09$, $\alpha_3 = 4.43366e-05$, $\alpha_4 = 271975$, $\alpha_5 = 2588.31$, $\alpha_6 = 43.7395$, $\alpha_7 = 27.2455$, $\alpha_8 = 20.2169$, $\alpha_9 = 0.000139488$, $\alpha_{10} = 6.77118e-06$, $\alpha_{11} = 2.06308e-05$, $\alpha_{12} = 896427$, $\alpha_{13} = 0.000959451$, $\alpha_{14} = 80901.1$, $\alpha_{15} = 9.15078e+07$, $\alpha_{16} = 0.00011957$

OB-66 rmse=0.9338
NSVDD, Residual: OB-63, $k = 1000$, $\eta_i = 1e-3$, $\eta_u = 1e-3$, $\eta_{\mu_i} = 1e-3$, $\eta_{\mu_u} = 1e-3$, $\lambda_i = 5e-3$, $\lambda_u = 5e-3$, $\lambda_{\mu_i} = 5e-3$, $\lambda_{\mu_u} = 5e-3$, 2 epochs

OB-67 rmse=0.9127
NSVDD, Residual: OB-65, $k = 2000$, $\eta_i = 1e-3$, $\eta_u = 1e-3$, $\eta_{\mu_i} = 1e-3$, $\eta_{\mu_u} = 1e-3$, $\lambda_i = 2e-3$, $\lambda_u = 2e-3$, $\lambda_{\mu_i} = 1.5e-3$, $\lambda_{\mu_u} = 1.5e-3$, 23 epochs

OB-68 rmse=0.9099
MovieKNNV3, Residual: OB-66, Pearson correlation, $K = 21$, $\alpha = 476$, $\beta = 780$, $\gamma = -1.7$, $\delta = 1.5$

OB-69 rmse=0.91
GTE, Residual: OB-67, 24 effects, $\alpha_0 = 0.460297$, $\sigma_0 = 0.112732$, $\alpha_1 = 5.07411$, $\sigma_1 = 24.9513$, $\alpha_2 = 0.000821287$, $\sigma_2 = 1.31449$, $\alpha_3 = 26.0328$, $\sigma_3 = 53.549$, $\alpha_4 = 0.0370089$, $\sigma_4 = 3.11902$, $\alpha_5 = 53.8362$, $\sigma_5 = 0.131344$, $\alpha_6 = 5.97711$, $\sigma_6 = 4.25552$, $\alpha_7 = 6.22559$, $\sigma_7 = 19.656$, $\alpha_8 = 588.765$, $\sigma_8 = 26.7259$, $\alpha_9 = 7.44892$, $\sigma_9 = 9.49731$, $\alpha_{10} = 0.00456163$, $\sigma_{10} = 97.7392$, $\alpha_{11} = 0.0455336$, $\sigma_{11} = 38.9575$, $\alpha_{12} = 0.278585$, $\sigma_{12} = 2.80386$, $\alpha_{13} = 31.7377$, $\sigma_{13} = 16.273$, $\alpha_{14} = 864.606$, $\sigma_{14} = 37.458$, $\alpha_{15} = 0.00166922$, $\sigma_{15} = 4.53062$, $\alpha_{16} = 1444.98$, $\sigma_{16} = 1.42299$, $\alpha_{17} = 295.419$, $\sigma_{17} = 9.83184$,

$\alpha_{18} = 1.59155$, $\sigma_{18} = 1.42808e+09$, $\alpha_{19} = 0.216866$, $\sigma_{19} = 16.3573$, $\alpha_{20} = 78.8341$, $\sigma_{20} = 7.42034$, $\alpha_{21} = 26.3644$, $\sigma_{21} = 11.8044$, $\alpha_{22} = 7.49029$, $\sigma_{22} = 28.4897$, $\alpha_{23} = 1$, $\sigma_{23} = 1$, $\alpha_{24} = 53.8362$, $\sigma_{24} = 1.7108$

OB-70 rmse=0.9136
SVD-ALS, Residual: no, $k = 50$, movieSolver=nonNeg, userSolver=nonNeg, $\lambda_i = 0.05$, $\lambda_u = 0.05$, 47 epochs

OB-71 rmse=0.9122
SVD-AUF, Residual: OB-70, adaptiveUserFactorMode=Pearson, solver=PINV, $\lambda = 0.0289428$, $\alpha = 20.1176$, $\delta = 8.55221$ $\gamma = -8.5$, $\kappa = 0.00640817$

OB-72 rmse=0.9099
GE, Residual: OB-71, 16 effects, $\alpha_1 = 0.0014015$, $\alpha_2 = 3.99785$, $\alpha_3 = 207.554$, $\alpha_4 = 213.329$, $\alpha_5 = 59340.4$, $\alpha_6 = 1512.75$, $\alpha_7 = 46.0147$, $\alpha_8 = 103.674$, $\alpha_9 = 0.00256827$, $\alpha_{10} = 1.92437e-05$, $\alpha_{11} = 21.961$, $\alpha_{12} = 5.55441$, $\alpha_{13} = 82.6428$, $\alpha_{14} = 3044.49$, $\alpha_{15} = 53.7824$, $\alpha_{16} = 2244.73$

OB-73 rmse=0.9063
NSVD1, Residual: OB-72, $k = 280$, $\eta_i = 1e-3$, $\eta_u = 1e-3$, $\eta_{\mu_i} = 2e-4$, $\eta_{\mu_u} = 2e-4$, $\lambda_i = 1e-6$, $\lambda_u = 1e-6$, $\lambda_{\mu_i} = 1e-5$, $\lambda_{\mu_u} = 1e-5$, muliply all η with 0.95 from epoch 20, 27 epochs

OB-74 rmse=0.9724
SVD-Time, Residual: OB-63, $k = 500$, $N = 30$, $M = 30$, $\eta_i = 0.001$, $\eta_u = 0.001$, $\lambda_i = 0.01$, $\lambda_u = 0.01$, 333 epochs

OB-75 rmse=1.0203
GTE, Residual: no, 24 effects, $\alpha_0 = 0.756448$, $\sigma_0 = 63.3648$, $\alpha_1 = 0.000341077$, $\sigma_1 = 783.599$, $\alpha_2 = 7.66667$, $\sigma_2 = 2.22136$, $\alpha_3 = 441.092$, $\sigma_3 = 2.39512$, $\alpha_4 = 7.9987$, $\sigma_4 = 2.30299$, $\alpha_5 = 2.19463$, $\sigma_5 = 2.50274$, $\alpha_6 = 0.40246$, $\sigma_6 = 3.79979$, $\alpha_7 = 0.00869847$, $\sigma_7 = 0.190355$, $\alpha_8 = 3761.7$, $\sigma_8 = 49.564$, $\alpha_9 = 0.0289626$, $\sigma_9 = 2.16438$, $\alpha_{10} = 0.174869$, $\sigma_{10} = 7.87177$, $\alpha_{11} = 0.723074$, $\sigma_{11} = 0.258699$, $\alpha_{12} = 358857$, $\sigma_{12} = 8.36612$, $\alpha_{13} = 0.415689$, $\sigma_{13} = 0.342104$, $\alpha_{14} = 0.719413$, $\sigma_{14} = 5.46462$, $\alpha_{15} = 0.278113$, $\sigma_{15} = 2.01964$, $\alpha_{16} = 2.9132$, $\sigma_{16} = 0.540534$, $\alpha_{17} = 0.291892$, $\sigma_{17} = 5.01496$, $\alpha_{18} = 1.06935$, $\sigma_{18} = 14.0819$, $\alpha_{19} = 0.000909059$, $\sigma_{19} = 8.7371$, $\alpha_{20} = 3.33668$, $\sigma_{20} = 3.6388$, $\alpha_{21} = 124644$, $\sigma_{21} = 2.39229$, $\alpha_{22} = 2.19671$, $\sigma_{22} = 4.83657$, $\alpha_{23} = 3.74613$, $\sigma_{23} = 1.37054$, $\alpha_{24} = 1.59059$, $\sigma_{24} = 1.59185$

OB-76 rmse=0.9271
MovieKNNV3, Residual: OB-75, Pearson correlation, $K = 36$, $\alpha = 321$, $\beta = 1807$, $\gamma = -3.27$, $\delta = 5.15$

OB-77 rmse=0.9257
SVD, Residual: OB-75, $k = 2500$, $\eta_i = 0.001$, $\eta_u = 0.001$, $\lambda_i = 0.003$, $\lambda_u = 0.003$, 26 epochs

OB-78 rmse=0.9686
NSVD2, Residual: OB-75, $k = 2000$, $\eta_i = 1e-3$, $\eta_u = 1e-3$, $\eta_{\mu_i} = 1e-3$, $\eta_{\mu_u} = 1e-3$, $\lambda_i = 1e-6$, $\lambda_u = 1e-6$, $\lambda_{\mu_i} = 1e-2$, $\lambda_{\mu_u} = 1e-2$, muliply all η with 0.9 from epoch 30, 45 epochs

OB-79 rmse=0.9471
NSVDD, Residual: no, $k = 2000$, $\eta_i = 1e-3$, $\eta_u = 1e-3$, $\eta_{\mu_i} = 1e-3$, $\eta_{\mu_u} = 1e-3$, $\lambda_i = 4e-2$, $\lambda_u = 4e-2$, $\lambda_{\mu_i} = 5e-2$, $\lambda_{\mu_u} = 5e-2$, muliply all η with 0.9 from epoch 5, 39 epochs

OB-80 rmse=0.9029
MovieKNNV3-2, Residual: OB-79, Pearson correlation, $K = 78$, $\alpha = 1021$, $\beta = 858$, $\gamma = -3.5$, $\delta = 12.2$, $\zeta = 0.9$, $\vartheta = 0.9$, $\kappa = 0.9$, $\nu = 0$, $\psi = 1$

OB-81 rmse=0.9612
GE, Residual: OB-71, 16 effects, $\alpha_1 = 4.16513$, $\alpha_2 = 1.7129e-06$, $\alpha_3 = 456.842$, $\alpha_4 = 178.695$, $\alpha_5 = 553.964$, $\alpha_6 = 7680.3$, $\alpha_7 = 223.568$, $\alpha_8 = 181.913$, $\alpha_9 = 2.43476e-06$, $\alpha_{10} = 4.40681e-06$, $\alpha_{11} = 8098.95$, $\alpha_{12} = 261.822$, $\alpha_{13} = 243.656$, $\alpha_{14} = 550.911$, $\alpha_{15} = 2175.92$, $\alpha_{16} = 6686.61$

OB-82 rmse=0.9334
MovieKNNV3-2, Residual: OB-81, Pearson correlation, $K = 71$, $\alpha = 0.28$, $\beta = 716$, $\gamma = -1.9$, $\delta = 18.2$, $\zeta = 1$, $\vartheta = 0.22$, $\kappa = 1.1$, $\nu = -0.002$, $\psi = 1$

OB-83 rmse=0.9373
NSVD1, Residual: no, $k = 900$, $\eta_i = 1e - 3$, $\eta_u = 1e - 3$, $\eta_{\mu_i} = 1e - 3$, $\eta_{\mu_u} = 1e - 3$, $\lambda_i = 1e - 3$, $\lambda_u = 1e - 2$, $\lambda_{\mu_i} = 1e - 3$, $\lambda_{\mu_u} = 1e - 2$, muliply all η with 0.93 from epoch 0, 43 epochs

OB-84 rmse=1.1148
MovieKNNV3-2, Residual: no, Ratio correlation, $K = 14$, $\alpha = 2.5$, $\beta = 9825$, $\gamma = -9.5$, $\delta = 4.4$, $\zeta = 1$, $\vartheta = 2.2$, $\kappa = 1$, $\nu = 0$, $\psi = 1$

OB-85 rmse=1.01
MovieKNNV3-2, Residual: OB-83, Ratio correlation, $K = 26$, $\alpha = 0.5$, $\beta = 1207$, $\gamma = -7.4$, $\delta = 4.86$, $\zeta = 1$, $\vartheta = 1.7$, $\kappa = 1.05$, $\nu = -0.02$, $\psi = 1$

OB-86 rmse=0.9534
NSVDD, Residual: OB-02, $k = 20$, $\eta_i = 1e - 3$, $\eta_u = 1e - 2$, $\eta_{\mu_i} = 1e - 3$, $\eta_{\mu_u} = 1e - 3$, $\lambda_i = 1e - 2$, $\lambda_u = 1e - 3$, $\lambda_{\mu_i} = 5e - 3$, $\lambda_{\mu_u} = 5e - 3$, 7 epochs

OB-87 rmse=0.9468
MovieKNNV3-2, Residual: OB-86, MSE correlation, $K = 24$, $\alpha = 47.4$, $\beta = 129$, $\gamma = -6$, $\delta = 5$, $\zeta = 1$, $\vartheta = 2.6$, $\kappa = 1.1$, $\nu = 0$, $\psi = 1$

OB-88 rmse=0.928
SVD-AUF, Residual: OB-83, adaptiveUserFactorMode=KRR, kernelType=pow, $\lambda = 9.35378$, $\alpha = 0.1$, $\gamma = 0.2$, $\kappa = 1.04$

OB-89 rmse=0.9228
GTE, Residual: OB-88, 24 effects, $\alpha_0 = 2.3976$, $\sigma_0 = 4.14296$, $\alpha_1 = 5.48903e - 06$, $\sigma_1 = 13.3131$, $\alpha_2 = 1.4251$, $\sigma_2 = 1.2055$, $\alpha_3 = 0.0394946$, $\sigma_3 = 1.53379e + 07$, $\alpha_4 = 5.22812$, $\sigma_4 = 9.03679$, $\alpha_5 = 0.837984$, $\sigma_5 = 2.38426$, $\alpha_6 = 1.30288$, $\sigma_6 = 6.80418$, $\alpha_7 = 3.46388e - 10$, $\sigma_7 = 0.98697$, $\alpha_8 = 902.531$, $\sigma_8 = 0.565299$, $\alpha_9 = 6.0307$, $\sigma_9 = 2.16438$, $\alpha_{10} = 2.80001$, $\sigma_{10} = 0.612924$, $\alpha_{11} = 251.787$, $\sigma_{11} = 2.16069$, $\alpha_{12} = 63023.2$, $\sigma_{12} = 1.89939$, $\alpha_{13} = 1.42503e - 08$, $\sigma_{13} = 18.3012$, $\alpha_{14} = 3.23376$, $\sigma_{14} = 2.64433$, $\alpha_{15} = 3237.21$, $\sigma_{15} = 5.45511$, $\alpha_{16} = 50.9026$, $\sigma_{16} = 1.27025e + 08$, $\alpha_{17} = 829.237$, $\sigma_{17} = 15.3857$, $\alpha_{18} = 5.27389$, $\sigma_{18} = 53.1659$, $\alpha_{19} = 0.385785$, $\sigma_{19} = 1.59994$, $\alpha_{20} = 0.00453852$, $\sigma_{20} = 4.15031$, $\alpha_{21} = 0.00351637$, $\sigma_{21} = 0.444039$, $\alpha_{22} = 0.864983$, $\sigma_{22} = 0.282021$, $\alpha_{23} = 0.236967$, $\sigma_{23} = 4.84254$, $\alpha_{24} = 3.29083$, $\sigma_{24} = 4.51019$

OB-90 rmse=0.8885
SVD++, Residual: no, $k = 1200$, $\eta_i = 5e - 3$, $\eta_u = 5e - 3$, $\eta_y = 1e - 3$ $\eta_{\mu_i} = 1e - 3$, $\eta_{\mu_u} = 1e - 3$, $\lambda_i = 1.5e - 2$, $\lambda_u = 1.5e - 2$, $\lambda_y = 1.0e - 2$, $\lambda_{\mu_i} = 1.5e - 2$, $\lambda_{\mu_u} = 1.5e - 2$, 77 epochs

OB-91 rmse=0.8806
IM, Residual: no, $k = 150$, nMovieTimeBins=10, muliply all η with 0.95 from epoch 13, 59 epochs

OB-92 rmse=0.9032
SVD, Residual: OB-86, $k = 1000$, $\eta_i = 0.0001$, $\eta_u = 0.001$, $\lambda_i = 0.001$, $\lambda_u = 0.001$, 42 epochs

OB-93 rmse=0.8968
MovieKNNV3-2, Residual: OB-92, Pearson correlation, $K = 24$, $\alpha = 2837$, $\beta = 159$, $\gamma = -4.7$, $\delta = 7$, $\zeta = 1$, $\vartheta = 0.8$, $\kappa = 1$, $\nu = 0$, $\psi = 1$

OB-94 rmse=0.8818
IM, Residual: no, $k = 50$, nMovieTimeBins=20, muliply all η with 0.95 from epoch 13, 49 epochs

OB-95 rmse=0.9202

IM, Residual: no, $k = 10$, nMovieTimeBins=20, 98 epochs

OB-96 rmse=0.8896
SBRAMF-UTB-UTF, Residual: no, $k = 150$, 198 epochs

OB-97 rmse=0.9148
SVD++Sym, Residual: no, $k = 10$, 16 epochs

OB-98 rmse=0.8788
SBRAMF-UTB-UTF-MTF-ATF-MFF-AFF, Residual: no, $k = 150$, 62 epochs

OB-99 rmse=0.9466
MovieKNNV3-2, Residual: OB-78, Pearson correlation, $K = 31$, $\alpha = 546$, $\beta = 2030$, $\gamma = -2.67$, $\delta = 7$, $\zeta = 0.8$, $\vartheta = 3.17$, $\kappa = 0.48$, $\nu = 0$, $\psi = 1$

OB-100 rmse=0.8758
MovieKNNV3-2, Residual: OB-98, Pearson correlation, $K = 141$, $\alpha = 1204$, $\beta = 332$, $\gamma = -4.1$, $\delta = 21.7$, $\zeta = 1$, $\vartheta = 0.4$, $\kappa = 1$, $\nu = 0$, $\psi = 1$

OB-101 rmse=0.8752
GE, Residual: OB-100, 16 effects, $\alpha_1 = 0.000129761$, $\alpha_2 = 1.22704e-05$, $\alpha_3 = 13.8956$, $\alpha_4 = 956.858$, $\alpha_5 = 0.000183497$, $\alpha_6 = 93953.6$, $\alpha_7 = 2.36624e-05$, $\alpha_8 = 95.9403$, $\alpha_9 = 282.937$, $\alpha_{10} = 184.181$, $\alpha_{11} = 0.0208714$, $\alpha_{12} = 62.7879$, $\alpha_{13} = 253.117$, $\alpha_{14} = 11940.3$, $\alpha_{15} = 52600.9$, $\alpha_{16} = 93.6713$

A.4 Optimize the Predictors on the Blend and Correct Residuals

For the following predictors we used a $K = 34$ cross fold (3.3). So we have 34 probe/qualifying predictions for a individual predictor. The reported RMSE values are based on a linear probe blend of this 34 predictors.

OBC-01 rmse=0.8977
SVD, Residual: no, $k = 200$, $\eta_i = 0.002$, $\eta_u = 0.002$, $\lambda_i = 0.01$, $\lambda_u = 0.01$, with biases, 61 epochs

OBC-02 rmse=0.8974
KNNMovieV2-2, Residual: OBC-01, Pearson correlation, $K = 50$, $\alpha = 842.5$, $\beta = 144.2$, $\gamma = -5.3$, $\delta = 25.9$, $\zeta = 0.98$, $\vartheta = 1.47$, $\kappa = 0.5$, $\nu = 0.5$, $\psi = 0.015$

OBC-03 rmse=0.8935
KNNMovieV2-2, Residual: OBC-01, Spearman's rank correlation, $K = 30$, $\alpha = 82.5$, $\beta = 9.8$, $\gamma = 0.037$, $\delta = 0.76$, $\zeta = 1.0$, $\vartheta = 1.13$, $\kappa = 1.2$, $\nu = -0.1$, $\psi = 0.46$

OBC-04 rmse=0.9007
RBMV3, Residual: no, 100 hidden units, $\eta = 0.0012$, $\lambda = 0.00025$

OBC-05 rmse=0.8845
KNNMovieV2-2, Residual: OBC-04, Pearson correlation, $K = 39$, $\alpha = 5889$, $\beta = 410.5$, $\gamma = 0.0034$, $\delta = 6.7$, $\zeta = 1.96$, $\vartheta = 2.3$, $\kappa = 1.2$, $\nu = -0.001$, $\psi = 0.69$

OBC-06 rmse=0.8797
GTE, Residual: OBC-05, 24 effects, $\alpha_0 = 0.00526644$, $\sigma_0 = 0.752045$, $\alpha_1 = 0.0262868$, $\sigma_1 = 67.7921$, $\alpha_2 = 1.00398e-09$, $\sigma_2 = 0.145604$, $\alpha_3 = 0.489294$, $\sigma_3 = 958.148$, $\alpha_4 = 0.0791142$, $\sigma_4 = 0.906078$, $\alpha_5 = 0.000544352$, $\sigma_5 = 985.25$, $\alpha_6 = 1.59343$, $\sigma_6 = 4.98191$, $\alpha_7 = 1.52602$, $\sigma_7 = 20.1282$, $\alpha_8 = 856.134$, $\sigma_8 = 80.9295$, $\alpha_9 = 1.17037$, $\sigma_9 = 92.1946$, $\alpha_{10} = 35473.3$, $\sigma_{10} = 19.8605$, $\alpha_{11} = 1.0084$, $\sigma_{11} = 7.89876$, $\alpha_{12} = 2.1725e+07$, $\sigma_{12} = 167.712$, $\alpha_{13} = 3237.21$, $\sigma_{13} = 6.10341e+12$, $\alpha_{14} = 7583.2$, $\sigma_{14} = 91.7444$, $\alpha_{15} = 593.311$, $\sigma_{15} = 0.564514$, $\alpha_{16} = 0.170086$, $\sigma_{16} = 113.133$, $\alpha_{17} = 1.0753$, $\sigma_{17} = 64.5719$, $\alpha_{18} = 28673.5$, $\sigma_{18} = 288.548$, $\alpha_{19} = 0.326539$, $\sigma_{19} = 15.2178$, $\alpha_{20} = 0.00497239$, $\sigma_{20} = 17.0848$, $\alpha_{21} = 1.98397$, $\sigma_{21} = 9.45723$, $\alpha_{22} = 20.8867$, $\sigma_{22} = 9.66172$, $\alpha_{23} = 9.03199$, $\sigma_{23} = 4.75582$, $\alpha_{24} = 0.00154022$, $\sigma_{24} = 3.76928$

OBC-07 rmse=0.8899
SVD++, Residual: no, $k = 250$, $\eta_i = 5e-3$, $\eta_u = 5e-3$, $\eta_y = 1e-3$ $\eta_{\mu_i} = 1e-3$, $\eta_{\mu_u} = 1e-3$, $\lambda_i = 1.5e-2$, $\lambda_u = 1.5e-2$, $\lambda_y = 1.0e-2$, $\lambda_{\mu_i} = 1.5e-2$, $\lambda_{\mu_u} = 1.5e-2$, 39 epochs

OBC-08 rmse=0.9046
SVD-ALS, Residual: no, $k = 100$, movieSolver=nonNeg, userSolver=nonNeg, $\lambda_i = 0.05$, $\lambda_u = 0.05$, 13 epochs

OBC-09 rmse=0.9009
SVD-AUF, Residual: OBC-08, adaptiveUserFactorMode=dateSim, solver=nonNeg, $\lambda = 0.0403887$, $\gamma = 0.195976$, $\beta = 0.191341$, $\kappa = 0.0168$

OBC-10 rmse=0.8826
IM, Residual: no, $k = 50$, nMovieTimeBins=20, 39 epochs

A.5 Probe Blending Results

A.5.1 BigChaos - Probe Predictors Blends

—[Mixes of subsets]—

PB-001 rmse=0.9230
NNBlend, Predictors {OP-12,OP-16,OP-18}, 1HL, 10 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-002 rmse=0.9102
NNBlend, Predictors {OP-14,OP-16,OP-18,OP-19,OP-20}, 1HL, 10 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-003 rmse=0.8805
NNBlend, Predictors {OP-01..OP-06 }, 1HL, 20 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-004 rmse=0.8722
NNBlend, Predictors {OP-01..OP-15 }, 1HL, 20 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-005 rmse=0.8707
NNBlend, Predictors {OP-01..OP-21 }, 1HL, 20 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-006 rmse=0.8694
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-12,}, 1HL, 20 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-007 rmse=0.8685
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-29}, 1HL, 20 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-008 rmse=0.8685
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-31}, 1HL, 20 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-009 rmse=0.8678
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-41}, 1HL, 20 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-010 rmse=0.8677
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-44}, 1HL, 20 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-011 rmse=0.8675
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-46}, 1HL, 15 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-012 rmse=0.8674
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-47}, 1HL, 15 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-013 rmse=0.8674

Group Name	count	Description
BC-Exact-10	10	Predictions from BigChaos, generated by the correct residual framwork {OBC-01..OBC-10}
BC-Exact-340	340	Predictions from BigChaos, {OBC-01..OBC-10}, all results from the 34-cross-fold setup
BC-BlendOpt-101	101	Predictions from BigChaos, generated by the optimize on blend framwork {OB-01..OB-101}
BC-PPrize08-60	60	Subset of predictions from BigChaos 2008 Progress Prize results {PP-01..PP-60}
BC-PPrize08-95	95	Predictions from BigChaos 2008 Progress Prize results {1..95}
BC-ProbeOpt-21	21	Single results, optimizing the probe RMSE, {OP-01..OP-21}
BC-132	132	BigChaos results {BC-Exact-10,BC-BlendOpt-101,BC-ProbeOpt-21}
BC-171	171	BigChaos results {BC-Exact-10,BC-BlendOpt-101,BC-PPrize08-60}
BC-192	192	BigChaos results {BC-Exact-10,BC-BlendOpt-101,BC-PPrize08-60,BC-ProbeOpt-21}
BC-192-Top5	5	Forward selection on BC-192, best 5 results
BC-192-Top10	10	Forward selection on BC-192, best 10 results
BC-192-Top15	15	Forward selection on BC-192, best 15 results
BC-192-Top20	20	Forward selection on BC-192, best 20 results
BC-192-Top15	25	Forward selection on BC-192, best 25 results
BC-192-Top30	30	Forward selection on BC-192, best 30 results
BC-192-Top35	35	Forward selection on BC-192, best 35 results
PT-236	236	Predictions from Pragmatic Theory received on April, 21th 2009
PT-236-Top10	10	Forward selection on PT-236, best 10 results
PT-236-Top20	20	Forward selection on PT-236, best 20 results
PT-236-Top30	30	Forward selection on PT-236, best 30 results
PT-236-Top40	40	Forward selection on PT-236, best 40 results
PT-236-Top50	50	Forward selection on PT-236, best 50 results
PT-236-Top60	60	Forward selection on PT-236, best 60 results
PT-236-Top70	70	Forward selection on PT-236, best 70 results
PT-236-Top80	80	Forward selection on PT-236, best 80 results
PT-236-Top90	90	Forward selection on PT-236, best 90 results
PT-236-Top100	100	Forward selection on PT-236, best 100 results
PT-236-Top110	110	Forward selection on PT-236, best 110 results
PT-236-Top120	120	Forward selection on PT-236, best 120 results
PT-236-Top130	130	Forward selection on PT-236, best 130 results
PT-236-Top140	140	Forward selection on PT-236, best 140 results
PT-242	242	PT-236 and 6 PT results
PT-242-PQDiff00-49	43	PT-242 results with $d = RMSE_{probe} - RMSE_{quiz}$, $d \leq 0.0049$
PT-242-PQDiff50-62	49	PT-242 results with $0.0063 \geq d > 0.0049$
PT-242-PQDiff63-73	48	PT-242 results with $0.0073 \geq d > 0.0063$
PT-242-PQDiff74-82	50	PT-242 results with $0.0082 \geq d > 0.0073$
PT-242-PQDiff83-99	52	PT-242 results with $0.0083 < d$
PT-250	250	PT-242 and 8 PT results
PT-250-Bottom10	10	Forward selection on PT-250, take last 10 (the results which are least significant)
PT-250-Bottom20	20	Forward selection on PT-250, take last 20
PT-250-Bottom30	30	Forward selection on PT-250, take last 30
PT-250-Bottom40	40	Forward selection on PT-250, take last 40
PT-250-Bottom50	50	Forward selection on PT-250, take last 50
PT-250-Bottom60	60	Forward selection on PT-250, take last 60
PT-250-Bottom70	70	Forward selection on PT-250, take last 70
PT-250-Bottom80	80	Forward selection on PT-250, take last 80
PT-250-Bottom90	90	Forward selection on PT-250, take last 90
PT-282	282	PT-236 and 46 PT results
PT-168	168	A set from PT were only probe RMSEs are available
PT-New16	16	16 results from PT received on May, 30th 2009
SVD++10-94	94	Predictors extracted from a 10-dim SBRAMF-UTB-UTF-MTF-ATF-MFF-AFF model
SVD++10-cross-144	144	Predictors extracted from a 10-dim SBRAMF-UTB-UTF-MTF-ATF-MFF-AFF model, with cross interactions
SVD++20-186	186	Predictors extracted from a 20-dim SBRAMF-UTB-UTF-MTF-ATF-MFF-AFF model
SVD++50-454	454	Predictors extracted from a 50-dim SBRAMF-UTB-UTF-MTF-ATF-MFF-AFF model
RBM-20-user	20	Activation values from the hidden unit layer (user as a bag of movies), 20 unit discrete RBM
RBM-20-movie	20	Activation values from the hidden unit layer (movie as a bag of users), 20 unit discrete RBM
RBM-20-movieWeights	120	Visible to hidden weights 5x discrete, 1x conditional, 20 unit Discrete RBM
RBM-50-user	50	Activation values from the hidden unit layer, 50 unit Discrete RBM
KNNk3-101	1515	KNN feature extraction, $k = 3$ neighbors, on all optimize on blend results (OB-01..OB-101)
KNNk50-OB-52-250	250	KNN feature extraction, $k = 50$, on OB-52 (RBMV5)
BK-results-17	17	BellKor results from Progress Prize 2008
BK-results-6	6	BellKor results, 2009
ALL-391	391	Merged results from all 3 teams: {PT-236,BC-132,BK-results-17,BK-results-6}
ALL-391-Top3	3	Forward selection on ALL-391, best 3 results
ALL-391-Top4	4	Forward selection on ALL-391, best 4 results
ALL-391-Top5	5	Forward selection on ALL-391, best 5 results
ALL-391-Top10	10	Forward selection on ALL-391, best 10 results
ALL-391-Top20	20	Forward selection on ALL-391, best 20 results
ALL-391-Top30	30	Forward selection on ALL-391, best 30 results
ALL-391-Top40	40	Forward selection on ALL-391, best 40 results
ALL-391-Top50	50	Forward selection on ALL-391, best 50 results
ALL-391-Top60	60	Forward selection on ALL-391, best 60 results
ALL-391-Top70	70	Forward selection on ALL-391, best 70 results
ALL-391-Top80	80	Forward selection on ALL-391, best 80 results
ALL-391-Top90	90	Forward selection on ALL-391, best 90 results
ALL-391-Top100	100	Forward selection on ALL-391, best 100 results
ALL-391-Top110	110	Forward selection on ALL-391, best 110 results
ALL-391-Top120	120	Forward selection on ALL-391, best 120 results
ALL-391-Top130	130	Forward selection on ALL-391, best 130 results
ALL-391-Top140	140	Forward selection on ALL-391, best 140 results
ALL-391-Bottom40	40	Forward selection on ALL-391, take last 40
ALL-476	476	Merged results from all 3 teams: {BC-171, PT-282, BK-results-17, BK-results-6}
ALL-476-Top75	75	Forward selection on ALL-476, best 75 results

Table 10: Different sets of results used for probe blending.

NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-48}, 1HL, 15 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-014 rmse=0.8674
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-51}, 1HL, 15 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-015 rmse=0.8673
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-53}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-016 rmse=0.8673
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-54}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-017 rmse=0.8672
NNBlend, Predictors {BC-ProbeOpt-21, PP-01..PP-55}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-018 rmse=0.8672
NNBlend, Predictors {BC-ProbeOpt-21, PP-01-PP-58}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-019 rmse=0.8671
NNBlend, Predictors {BC-ProbeOpt-21, BC-PPrize08-60}, 1HL, 11 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-020 rmse=0.8669
NNBlend, Predictors {BC-ProbeOpt-21, BC-PPrize08-60}, 1HL, 14 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-021 rmse=0.8668
NNBlend, Predictors {OP-01..OP-11, BC-PPrize08-95}, 1HL, 14 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-022 rmse=0.8668
NNBlend, Predictors {OP-01..OP-12, BC-PPrize08-95}, 1HL, 16 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-023 rmse=0.8667
NNBlend, Predictors {OP-01..OP-12, BC-PPrize08-95}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-024 rmse=0.8665
NNBlend, Predictors {OP-01..OP-15, BC-PPrize08-95}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-025 rmse=0.8662
NNBlend, Predictors {OP-01..OP-18, BC-PPrize08-95}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-026 rmse=0.8661
NNBlend, Predictors {BC-ProbeOpt-21, BC-PPrize08-95}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-027 rmse=0.8659
NNBlend, Predictors {BC-ProbeOpt-21, BC-PPrize08-60, OB-01..OB-02}, 1HL, 12 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-028 rmse=0.8659
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-02}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-029 rmse=0.8659
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-03}, 1HL, 12 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-030 rmse=0.8658
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-04}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-031 rmse=0.8652

NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-06}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-032 rmse=0.8651
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-10}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-033 rmse=0.8643
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-24}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-034 rmse=0.8642
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-28}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-034 rmse=0.8639
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-37}, 1HL, 12 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-035 rmse=0.8638
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-39}, 1HL, 12 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-036 rmse=0.8635
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-48}, 1HL, 12 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-037 rmse=0.8637
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-68}, 1HL, 12 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-038 rmse=0.8636
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-74}, 1HL, 11 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-039 rmse=0.8634
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-92}, 1HL, 11 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-040 rmse=0.8633
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-93}, 1HL, 10 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-041 rmse=0.8632
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-94}, 1HL, 11 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-042 rmse=0.8631
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-96}, 1HL, 11 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-043 rmse=0.8630
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-97}, 1HL, 10 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-044 rmse=0.8629
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, OB-01..OB-98}, 1HL, 11 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-045 rmse=0.8627
NNBlend, Predictors {BC-PPrize08-60, BC-ProbeOpt-21, BC-BlendOpt-101}, 1HL, 11 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-045 rmse=0.8621
NNBlend, Predictors {BC-PPrize08-60, BC-BlendOpt-101, BC-ProbeOpt-21, BC-Exact-10}, 1HL, 8 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-046 rmse=0.8639
NNBlend, Predictors {BC-PPrize08-60, BC-BlendOpt-101, BC-ProbeOpt-21, BC-Exact-10}, 1HL, 25 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-047 rmse=0.8688
NNBlend, Predictors {BC-Exact-340}, 1HL, 5 neurons

PB-048 rmse=0.8708
NNBlend, Predictors {BC-BlendOpt-101, BC-Exact-340}, 1HL, 40 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs (huge overfitting)

PB-049 rmse=0.8683
NNBlend, Predictors {BC-PPrize08-60, BC-Exact-340}, 1HL, 30 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs (overfitting)

PB-050 rmse=0.8639
NNBlend, Predictors {BC-BlendOpt-101}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-051 rmse=0.8639
NNBlend, Predictors {BC-BlendOpt-101}, 1HL, 11 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

—**[Forward selection of k-best predictions]**—

PB-052 rmse=0.8692
NNBlend, Predictors {BC-192-Top5}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-053 rmse=0.8670
NNBlend, Predictors {BC-192-Top10}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-054 rmse=0.8658
NNBlend, Predictors {BC-192-Top15}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-055 rmse=0.8651
NNBlend, Predictors {BC-192-Top20}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-056 rmse=0.8644
NNBlend, Predictors {BC-192-Top25}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-057 rmse=0.8642
NNBlend, Predictors {BC-192-Top30}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-058 rmse=0.8641
NNBlend, Predictors {BC-192-Top35}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

—**[Blending of SVD and RBM features]**—

PB-059 rmse=0.8912
NNBlend, Predictors {SVD++20-187}, 2HL, 50-10 neurons, $\eta^{(-)} = 1e-6$, 493 epochs

PB-060 rmse=0.8952
NNBlend, Predictors {SVD++20-187}, 2HL, 100-100 neurons, $\eta^{(-)} = 1e-6$, 60 epochs

PB-061 rmse=0.9210
NNBlend, Predictors {SVD++20-187}, 2HL, 10-5 neurons, $\eta^{(-)} = 1e-6$, 493 epochs

PB-062 rmse=0.9448
NNBlend, Predictors {RBM-20-user, RBM-20-movieWeights}, 2HL, 10-5 neurons, $\eta^{(-)} = 1e-6$, 493 epochs

PB-063 rmse=0.8902
NNBlend, Predictors {RBM-50-user, SVD++20-187}, 2HL, 50-10 neurons, $\eta^{(-)} = 1e-6$, 493 epochs

PB-064 rmse=0.9176
NNBlend, Predictors {RBM-20-user, RBM-20-movieWeights, SVD++50-455}, 2HL, 5-3 neurons, $\eta^{(-)} = 1e-6$, 493 epochs

PB-065 rmse=0.9176
NNBlend, Predictors {RBM-20-user, RBM-20-movieWeights, SVD++20-187}, 2HL, 40-8 neurons, $\eta^{(-)} = 1e-6$, 493 epochs

A.5.2 BellKor - Probe Predictors Blends

PB-066 rmse=0.9181
NNBlend, Predictors {BK-results-17 except two predictors, result 100. in [17]}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-067 rmse=0.8682
NNBlend, Predictors {BK-results-17, result 99. in [17]}, 1HL, 13 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

A.5.3 Pragmatic Theory - Probe Predictors Blends

PB-068 rmse=0.8596
NNBlend, Predictors {PT-236}, 1HL, 8 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-069 rmse=0.8651
NNBlend, Predictors {PT-new16}, 1HL, 15 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

—**[Forward selection of k-best predictions of set PT-236]**—

PB-070 rmse=0.8626
NNBlend, Predictors {PT-236-Top10}, 1HL, 180 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-071 rmse=0.8616
NNBlend, Predictors {PT-236-Top20}, 1HL, 90 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-072 rmse=0.8610
NNBlend, Predictors {PT-236-Top30}, 1HL, 60 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-073 rmse=0.8606
NNBlend, Predictors {PT-236-Top40}, 1HL, 45 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-074 rmse=0.8604
NNBlend, Predictors {PT-236-Top50}, 1HL, 36 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-075 rmse=0.8600
NNBlend, Predictors {PT-236-Top60}, 1HL, 30 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-076 rmse=0.8599
NNBlend, Predictors {PT-236-Top70}, 1HL, 25 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-077 rmse=0.8598
NNBlend, Predictors {PT-236-Top80}, 1HL, 22 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-078 rmse=0.8598
NNBlend, Predictors {PT-236-Top90}, 1HL, 20 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-079 rmse=0.8597
NNBlend, Predictors {PT-236-Top100}, 1HL, 18 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-080 rmse=0.8596
NNBlend, Predictors {PT-236-Top110}, 1HL, 16 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-081 rmse=0.8595
NNBlend, Predictors {PT-236-Top120}, 1HL, 15 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-082 rmse=0.8594
NNBlend, Predictors {PT-236-Top130}, 1HL, 13 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-083 rmse=0.8595
NNBlend, Predictors {PT-236-Top140}, 1HL, 12 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-084 rmse=0.8597
ENNBlend, Predictors {PT-236-Top140}, 1HL, 5 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs, select 7 random, 1187 sub-results, 10-FrequencyBins linear blend

PB-085 rmse=0.8596
ENNBlend, Predictors {PT-236-Top140}, 1HL, 5 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs, select 7 random, 1187 sub-results, 20-DateBins linear blend

—[**Probe-Quiz RMSE groups of set PT-242**]—

PB-086 rmse=0.8700
NNBlend, Predictors {PT-242-PQDiff00-49}, 1HL, 13 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-087 rmse=0.8668
NNBlend, Predictors {PT-242-PQDiff50-62}, 1HL, 13 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-088 rmse=0.8632
NNBlend, Predictors {PT-242-PQDiff63-73}, 1HL, 13 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-089 rmse=0.8619
NNBlend, Predictors {PT-242-PQDiff74-82}, 1HL, 13 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-090 rmse=0.8618
NNBlend, Predictors {PT-242-PQDiff83-99}, 1HL, 13 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

—[**Forward selection of k-best predictions of set PT-250, take k-least significant results**]—

PB-091 rmse=0.8670
NNBlend, Predictors {PT-250-Bottom10}, 1HL, 15 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-092 rmse=0.8645

NNBlend, Predictors {PT-250-Bottom20}, 1HL, 15 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-093 rmse=0.8632
NNBlend, Predictors {PT-250-Bottom30}, 1HL, 15 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-094 rmse=0.8622
NNBlend, Predictors {PT-250-Bottom50}, 2HL, 30-30 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-095 rmse=0.8620
NNBlend, Predictors {PT-250-Bottom50}, 1HL, 15 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-096 rmse=0.8618
NNBlend, Predictors {PT-250-Bottom60}, 1HL, 15 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-097 rmse=0.8616
NNBlend, Predictors {PT-250-Bottom70}, 1HL, 15 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-098 rmse=0.8613
NNBlend, Predictors {PT-250-Bottom80}, 1HL, 15 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

PB-099 rmse=0.8604
NNBlend, Predictors {PT-250-Bottom120}, 1HL, 15 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs

A.5.4 BigChaos - BellKor - Pragmatic Theory - Probe Predictors Blends

We put most effort in the production different blends from all team results.

PB-100 rmse=0.8593
LinearBlend, Predictors {ALL-391}, 15-DateBins, $\lambda = 0.003$

PB-101 rmse=0.8584
ENNBlend, Predictors {ALL-476, BC-Exact-340}, 1HL, 4 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs, select 3 random, 1149 sub-results, 2-DateBins linear blend

PB-102 rmse=0.8596
ENNBlend, Predictors {ALL-391}, 1HL, 5 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs, select 3 random, 810 sub-results, 12-DateBins linear blend

PB-103 rmse=0.8592
ENNBlend, Predictors {ALL-391}, 1HL, 5 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs, select 4 random, 909 sub-results, 3-FrequencyBins linear blend

PB-104 rmse=0.8592
PolyRegressionBlend, Predictors {ALL-476-Top75}, Order=2, $\lambda = 0.00012$

PB-105 rmse=0.8607
NNBlend, Predictors {ALL-476, RBM-50-user, SVD++20-187}, 2HL, 30-15 neurons, $\eta^{(-)} = 1e - 6$, 493 epochs

PB-106 rmse=0.8601
ENNBlend, Predictors {ALL-476}, 1HL, 5 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs, select 7 random, 1213 sub-results, 25-ClusterBins(5 user and 5 movie clusters) linear blend

PB-107 rmse=0.8600
ENNBlend, Predictors {PT-168}, 1HL, 5 neurons, $\eta^{(-)} = 3e - 7$, 1334 epochs, select 3 random, 980 sub-results, 4-FrequencyBins linear blend

PB-108 rmse=0.8586
ENNBlend, Predictors {ALL-476}, 1HL, 5 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs, select 3 random, 810 sub-results, 3-FrequencyBins linear blend

PB-109 rmse=0.8586
ENNBlend, Predictors {ALL-476}, 1HL, 10 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs, select 2 random, 993 sub-results, 3-FrequencyBins linear blend

PB-110 rmse=0.8583
ENNBlend, Predictors {ALL-476}, 1HL, 5 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs, select 4 random, 1060 sub-results, 4-FrequencyBins linear blend

PB-111 rmse=0.8589
NNBlend, Predictors {ALL-476}, 1HL, 4 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-112 rmse=0.8621
KRRBlend, Predictors {ALL-476}, 4000 sample probe subset, 8 results blended with LinearBlend, Gauss kernel: $\sigma = 100$, $\lambda = 1e-6$

PB-113 rmse=0.8616
KRRBlend, Predictors {ALL-476}, 4000 sample probe subset, 14 results blended with LinearBlend, Gauss kernel: $\sigma = 100$, $\lambda = 1e-6$

PB-114 rmse=0.8583
ENNBlend, Predictors {ALL-476,SVD++50-455,RBM-50-user}, 1HL, 3 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs, select 10 random, 909 sub-results, 3-FrequencyBins linear blend

PB-115 rmse=0.8587
NNBlend, Predictors {PB-143 ... PB-163,SVD++10-cross-145,RBM-50-user}, 2HL, 17-7 neurons, $\eta^{(-)} = 1e-6$, 493 epochs

PB-116 rmse=0.8589
PolyRegressionBlend, Predictors {PB-143 ... PB-163,SVD++10-cross-145,RBM-50-user}, Order=2, $\lambda = 0.0002$

PB-117 rmse=0.8588
NNBlend, Predictors {PB-143 ... PB-163,SVD++10-cross-145,RBM-50-user}, 2HL, 18-5 neurons, $\eta^{(-)} = 1e-6$, 493 epochs

PB-118 rmse=0.8592
NNBlend, Predictors {PB-143 ... PB-163,SVD++10-95,RBM-50-user}, 2HL, 20-5 neurons, $\eta^{(-)} = 1e-6$, 493 epochs

PB-119 rmse=0.8594
ENNBlend, Predictors {ALL-391}, 1HL, 5 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs, select 6 random, 1021 sub-results, 5-FrequencyBins linear blend

—**[Forward selection of k-best predictions of set ALL-391]**—

PB-120 rmse=0.8645
NNBlend, Predictors {ALL-391-Top3}, 1HL, 500 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-121 rmse=0.8645
NNBlend, Predictors {ALL-391-Top3}, 2HL, 50-50 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-122 rmse=0.8641
NNBlend, Predictors {ALL-391-Top4}, 1HL, 400 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-123 rmse=0.8637
NNBlend, Predictors {ALL-391-Top5}, 1HL, 300 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-124 rmse=0.8637
NNBlend, Predictors {ALL-391-Top5}, 3HL, 48-48 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-125 rmse=0.8623
NNBlend, Predictors {ALL-391-Top10}, 1HL, 150 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-126 rmse=0.8623
NNBlend, Predictors {ALL-391-Top10}, 2HL, 45-45 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-127 rmse=0.8609
NNBlend, Predictors {ALL-391-Top20}, 1HL, 70 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-128 rmse=0.8611
NNBlend, Predictors {ALL-391-Top20}, 2HL, 40-40 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-129 rmse=0.8603
NNBlend, Predictors {ALL-391-Top30}, 1HL, 50 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-130 rmse=0.8604
NNBlend, Predictors {ALL-391-Top30}, 2HL, 35-35 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-131 rmse=0.8601
NNBlend, Predictors {ALL-391-Top40}, 1HL, 44 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-132 rmse=0.8602
NNBlend, Predictors {ALL-391-Top40}, 2HL, 30-30 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-133 rmse=0.8598
NNBlend, Predictors {ALL-391-Top50}, 1HL, 33 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-134 rmse=0.8601
NNBlend, Predictors {ALL-391-Top50}, 2HL, 25-25 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-135 rmse=0.8598
NNBlend, Predictors {ALL-391-Top60}, 1HL, 28 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-136 rmse=0.8595
NNBlend, Predictors {ALL-391-Top70}, 1HL, 23 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-137 rmse=0.8595
NNBlend, Predictors {ALL-391-Top80}, 1HL, 21 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-138 rmse=0.8594
NNBlend, Predictors {ALL-391-Top90}, 1HL, 18 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-139 rmse=0.8592
NNBlend, Predictors {ALL-391-Top100}, 1HL, 16 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-140 rmse=0.8591
NNBlend, Predictors {ALL-391-Top110}, 1HL, 15 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-141 rmse=0.8590
NNBlend, Predictors {ALL-391-Top120}, 1HL, 14 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

PB-142 rmse=0.8624
NNBlend, Predictors {ALL-391-Bottom40}, 1HL, 15 neurons, $\eta^{(-)} = 3e-7$, 1334 epochs

—**[BGBDT - Tree Blends]**—

PB-143 rmse=0.8599
BGBDT, Predictors {ALL-474}, $\lambda = 0.1$, $N_{boost} = 150$, $N_{bag} = 32$, $d = 8$, $N_{min} = 100$, $S = 50$

PB-144 rmse=0.8602
BGBDT, Predictors {ALL-474}, $\lambda = 0.1$, $N_{boost} = 100$, $N_{bag} = 32$, $d = 8$, $N_{min} = 100$, $S = 24$

PB-145 rmse=0.8629
BGBDT, Predictors {ALL-391-Top10}, $\lambda = 0.1$, $N_{boost} = 250$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 4$

PB-146 rmse=0.8618
BGBDT, Predictors {ALL-391-Top20}, $\lambda = 0.1$, $N_{boost} = 250$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 4$

PB-147 rmse=0.8614
BGBDT, Predictors {ALL-391-Top30}, $\lambda = 0.1$, $N_{boost} = 250$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 5$

PB-148 rmse=0.8623
BGBDT, Predictors {ALL-476-Top75, RBM-20-user}, $\lambda = 0.1$, $N_{boost} = 100$, $N_{bag} = 32$, $d = 5$, $N_{min} = 50$, $S = 9$

PB-149 rmse=0.8626
BGBDT, Predictors {ALL-476-Top75, RBM-20-user}, $\lambda = 0.1$, $N_{boost} = 60$, $N_{bag} = 32$, $d = 5$, $N_{min} = 50$, $S = 15$

PB-150 rmse=0.8631
BGBDT, Predictors {ALL-476-Top75, RBM-20-user}, $\lambda = 0.1$, $N_{boost} = 100$, $N_{bag} = 32$, $d = 20$, $N_{min} = 50$, $S = 9$

PB-151 rmse=0.8623
BGBDT, Predictors {ALL-476-Top75, RBM-20-user}, $\lambda = 0.1$, $N_{boost} = 100$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 9$

PB-152 rmse=0.8606
BGBDT, Predictors {ALL-476-Top75, RBM-20-user}, $\lambda = 0.1$, $N_{boost} = 200$, $N_{bag} = 32$, $d = 7$, $N_{min} = 50$, $S = 9$

PB-153 rmse=0.8604
BGBDT, Predictors {ALL-476-Top75, RBM-20-user}, $\lambda = 0.1$, $N_{boost} = 200$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 9$

PB-154 rmse=0.9473
BGBDT, Predictors {RBM-20-user, RBM-20-movie}, $\lambda = 0.1$, $N_{boost} = 200$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 1$

PB-155 rmse=0.9438
BGBDT, Predictors {RBM-20-user, RBM-20-movie}, $\lambda = 0.1$, $N_{boost} = 300$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 2$

PB-156 rmse=0.8598
BGBDT, Predictors {RBM-20-user, RBM-20-movie, ALL-476}, $\lambda = 0.1$, $N_{boost} = 200$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 30$

PB-157 rmse=0.8618
BGBDT, Predictors {PT-282}, $\lambda = 0.1$, $N_{boost} = 80$, $N_{bag} = 32$, $d = 8$, $N_{min} = 100$, $S = 12$

PB-158 rmse=0.8609
BGBDT, Predictors {PT-282}, $\lambda = 0.2$, $N_{boost} = 120$, $N_{bag} = 32$, $d = 8$, $N_{min} = 100$, $S = 12$

PB-159 rmse=0.8596
BGBDT, Predictors {ALL-476, SVD++20}, $\lambda = 0.25$, $N_{boost} = 150$, $N_{bag} = 32$, $d = 10$, $N_{min} = 50$, $S = 15$

PB-160 rmse=0.8599
BGBDT, Predictors {ALL-476, RBM-20-user, RBM-20-movie}, $\lambda = 0.1$, $N_{boost} = 200$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 10$

PB-161 rmse=0.9075
BGBDT, Predictors {SVD++20}, $\lambda = 0.1$, $N_{boost} = 250$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 14$

PB-162 rmse=0.8606
BGBDT, Predictors {SVD++20, KNNk3-101, ALL-476}, $\lambda = 0.25$, $N_{boost} = 250$, $N_{bag} = 32$, $d = 10$, $N_{min} = 50$, $S = 20$

PB-163 rmse=0.8954
BGBDT, Predictors {KNNk50-OB-52-250}, $\lambda = 0.25$, $N_{boost} = 120$, $N_{bag} = 32$, $d = 8$, $N_{min} = 100$, $S = 12$

PB-164 rmse=0.8592
BGBDT, Predictors {PB-143 ... PB-163}, $\lambda = 0.1$, $N_{boost} = 120$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 3$

PB-165 rmse=0.8592
BGBDT, Predictors {PB-143 ... PB-163}, $\lambda = 0.1$, $N_{boost} = 100$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 4$

PB-166 rmse=0.8592
BGBDT, Predictors {PB-143 ... PB-163}, $\lambda = 0.1$, $N_{boost} = 120$, $N_{bag} = 32$, $d = 12$, $N_{min} = 50$, $S = 3$

B Detailed Probe Blend Predictor Groups

group	predictors
PT-236	bk1-a1000x, bk1-a1000x-knn1-3, bk1-a200-knn1-1, bk1-a200-knn1-3, bk1-a200-knn3-1, bk1-a200x-knn1-1, bk1-a50, bk1-a50-2, bk1-a50-2-knn1-1, bk1-a50-2x, bk1-a50-movie5, bk1-b1000, bk1-b200-1, bk1-b200-1-knn1-0, bk1-b200-1-knn1-1, bk1-b200-2, bk1-b200-5, bk1-b200-5x, bk1-b200-5x-knn1-3, bk1-b200-6, bk1-b200-6x, bk1-c200-knn2-1, bk1-c200x, bk2-b200hz-knn1-1, bk3-100g1, bk3-100g2, bk3-100g4, bk3-100ga-knn3-1, bk3-100ga1, bk3-100ga4, bk3-100gax, bk3-100gx, bk3-200g-knn2-1, bk3-200g1, bk3-200g3, bk3-200g4, bk3-200gx, bk3-a50-knn1-3, bk3-b200-knn1-3, bk3-c50, bk3-c50-knn2-1, bk3-c50-knn3-1, bk3-c50x, bk3-d200, bk3-d200z, bk3-d200z-knn4-1, bk4-bias, blend2-ga-knn2-1, blend2-ga1, blend2-ga4, blend2-gax, blend2b, blend3-knn2-1, blend5-knn3-1, blend5-knn4-1, brismf250, brismf250-movie5, brismf40-movie5, brismf760-movie5, brismf760-user2, brismf760n, brismf760n-knn1-1, brismf760n-knn1-3, brismf760n-knn3-1, brismf760n-movie5, brismf760n-movie6, brismf760n-user2, crbm100, crbm100-movie2, crbm100-ssvd-07-00, crbm100-ssvd-07-00-movie2, crbm100-ssvd-07-20, crbm100x-ssvd-03-00, crbm200, crbm200-ssvd-07-00, drbm100-500, drbm100-500-mfw31-m, drbm100-500-user2, drbm160-640, drbm160-640-bk4-knn3-1, drbm160-640-knn3-1, drbm160-640-knn4-1, frbm100-mf27-m, frbm2-100g3, frbm2-100g4, frbm2-100gx-knn2-1, frbm200, frbm200-mf27-flip20, frbm200-mf27-knn2-2, frbm200x, frbm300, frbm300-bk4-knn3-1, frbm300-knn3-1, frbm300-movie8, frbm300x, globalEffect14-movie5, gte14b, integ0-0-0TZ, integ0-0-0TZ-flip20-knn2-1, integ0-0-0TZ-grbm200, integ0-0-0TZ-movie6, integ0-0-0TZmilestone6-200, integ0-0-0TZmilestone6-200-knn2-1, integ0-0-0TZmilestone6-200-movie6, integ0-100-100TZ, integ0-100-100TZ-movie6, integ0-200-200NT, integ0-200-200NT-user2, integ0-200-200TZ, integ0-200-200TZ-knn1-1, integ0-200-200TZ-knn1-3, integ0-200-200TZ-knn2-1, integ0-200-200TZ-knn3-1, integ0-200-200TZ-movie5, integ0-200-200TZ-user2, integ20-100-100NT, integ40-200-0ST, integ40-200-0T, integ60-0-0TS, integ60-0-0TS-movie5, integ60-0-0TS-user2, integ80-80-0TZ-movie5, integ80-80-0TZM, integ80-80-0TZM-movie6, mf01-20-movie3, mf01-40-3-80-movie4, mf27-20, mf27-20-movie4, mf27-20-movie5, mf27-20-u, mf27-20env50-m, mf27-20env50-movie4, mf27-20env50-movie5, mf27-40-3-80-movie5, mfc27-60-10-120, mfc27-60-10-120-m, mfc27-60-10-120-user2, mfw31-00, mfw31-00-movie6, mfw31-00milestone3-100-movie6, mfw31-00milestone4-100-movie5, mfw31-00milestone5-100-movie5, mfw31-00milestone6-100, mfw31-00milestone7-100-movie6, mfw31-00milestone9-100-movie6, mfw31-05-asym3v250, mfw31-05-asym3v250-movie6, mfw31-05-asym3v250-user2, mfw31-05-m, mfw31-10-milestone0-150, mfw31-10-milestone0-150-movie5, mfw31-10-milestone5-150, mfw31-10-milestone5-150-movie5, mfw31-40env50, mfw31-40env50-m, mfw31-40env50-m2, mfw31-60-10-120-m, mfw31-60-10-120-movie6, mfw31-60-x, mfw31-80-x, mfw31-80-x-m, mfw31-80-x-user2, mmean, mp5, mskew, nmf40-60-10, nmf40-60-10-movie3, nmf80-120-20-m, nmf80-120-20-mf27-movie4, nnmf40, nnmf40-ssvd-07-00, nnmf40-ssvd-07-00-movie3, nnmf80-ssvd-39-00-movie4, pmf40-60-10-m, pmf80-120-20-mf27-movie4, rbm100-ssvd-07-00-movie2, ssvd-04-00, ssvd-04-10-m, ssvd-04-40-m, ssvd-07-30-2-movie2, ssvd-31-00-asym1-20-movie6, ssvd-31-00-asym3-100, ssvd-31-00-asym3-20, ssvd-31-00-asym3-200, ssvd-31-00-asym3-200-movie2, ssvd-31-00-asym3v-300-movie2, ssvd-31-00-asym3w-200-movie2, ssvd-31-00-asym4-200, ssvd-31-00-asym4v-200-movie5, ssvd-31-00-movie, ssvd-31-00-movie5, ssvd-31-10-m, ssvd-31-20, ssvd-31-20-cluster-movie2, ssvd-31-20-m, ssvd-31-20-movie2, ssvd-31-20-u, ssvd-31-60, ssvd-39-05, ssvd-39-05-movie4, ssvd-39-10, ssvd-39-10-movie4, ssvd-63-00, ssvd-63-00-movie4, ssvd-63-30, ssvd-63-30-movie4, svd02x, svd05x, trbm100, trbm100-movie2, trbm100-ssvd-31-00, trbm100-ssvd-31-00-movie2, trbm100-ssvd-31-00-user2, trbm150-mf27-m, trbm150-mf27-movie4, trbm150-movie5, trbm150-ssvd-07-00-movie4, trbm150-ssvd-39-00, trbm50-asym3v250, trbm50-asym3v250-mfw27-movie5, trbm50-asym3v250-movie5, trbm50-asym3v250-movie6, trbm50-asym3v250-user2, trbm50-mf27, trbm50-milestone0-150-movie6, trbm50-milestone9-150-movie5, trbm50-ssvd-39-00, ucount, up2, up3, up4, up5, urbm20-1000, urbm20-1000-knn1-3, usermovie, uskew
PT-242	PT-236, blend2-gax-knn1-1B052, blend5-knn1-1B047, bk4-biasZ, integ0-0-0TZ-knn3-1B002, mfw31-10-milestone0-150-knn1-3B022, integ0-0-0TZ-grbm200-knn1-3B025
PT-250	PT-242, integ0-0-0TZ-knn1-1B046, integ0-0-0TZ-knn1-3B015, brismf760n-knn1-1B050, drbm160-640-bk4-knn3-1B011, brismf760n-knn1-3B019, mf27-20-knn1-1B072, frbm300-knn1-1B068, brismf760n-knn4-1B081
PT-168	bk1-a1000x-knn1-4B145, bk1-a1000x-knn5-8B111, bk1-a50-2-knn1-1B059, bk1-a50-2-knn1-3B028, bk1-a50-knn1-1B058, bk1-a50-knn1-3B027, bk1-b200-5-knn1-3B029, bk1-b200-6x-knn1-1B054, bk1-b200-6x-knn1-1B061, bk1-b200-6x-knn1-3B030, bk1-b200-6x-knn3-1B010, bk1-b200-6x-knn4-1B085, bk1-c200-knn1-1B062, bk1-c200-knn1-3B031, bk1-c200x-knn1-1B063, bk1-c200x-knn1-3B032, bk3-100ga-knn1-1B064, bk3-200gx-knn1-1B065, bk3-d200-knn4-1B080, bk3-d200z-nlpp1-knn3-1B109, bk4-b200-knn1-3B099, bk4-c200gx-knn1-1B162, bk4-c200gx-knn1-5B164, bk4-c200gx-knn3-1B167, bk4-c200z-knn1-1B172, bk4-c200z-knn1-3B176, bk4-c200z-knn1-5B182, bk4-c200z-knn3-1B165, bk4-c200z-knn5-1B183, bk4-c500-knn1-3B126, bk4-c500-knn1-4B143, bk4-c500-knn1-5B129, bk4-c500-knn3-1B123, bk4-d500-knn1-4B144, bk4-e200-knn1-1B170, bk4-e200-knn1-3B163, bk4-e200-knn1-5B180, bk4-e200-knn3-1B169, bk4-e200-knn5-1B185, bk4-e50a-knn1-1B188, bk4-e50a-knn1-3B166, bk4-e50a-knn1-5B159, bk4-e50a-knn3-1B175, bk4-e50a-knn5-1B161, bk4-f200gx-knn1-1B223, bk4-f200gx-knn1-2B216, bk4-f200gx-knn1-3B190, bk4-f200gx-knn1-4B241, bk4-f200gx-knn1-5B198, bk4-f200gx-knn3-1B193, bk4-f200gx-knn5-1B228, bk4-f200z4-knn1-2B234, bk4-f200z4-knn1-3B242, bk4-f200z4-knn1-4B224, bk4-f200z4-knn1-5B197, bk4-f200z4-knn3-1B199, bk4-f200z4-knn5-1B218, bk4-f200z4-nlpp1-knn1-1B204, bk4-f200z4-nlpp1-knn1-2B202, bk4-f200z4-nlpp1-knn1-4B221, bk4-f200z4-nlpp1-knn1-5B207, bk4-f200z4-nlpp1-knn3-1B219, bk4-f200z4-nlpp1-knn5-1B192, bk5-b200-knn1-3B094, bk5-b200-knn1-4B147, bk5-b200-knn3-1B096, bk5-b200B089-knn1-1B091, bk5-b200B089-knn3-1B092, bk5-b200B089-knn4-1B093, blend2-gax-knn3-1B008, blend2-gax-knn4-1B083, blend2-gb-knn1-1B213, blend2-gb-knn1-2B238, blend2-gb-knn1-3B212, blend2-gb-knn1-4B210, blend2-gb-knn1-5B214, blend2-gb-knn3-1B217, blend2-gb-knn5-1B191, blend2-gb1-knn1-1B196, blend2-gb1-knn1-2B194, blend2-gb1-knn1-3B233, blend2-gb1-knn1-4B237, blend2-gb1-knn1-5B232, blend2-gb2-knn1-1B203, blend2-gb2-knn1-2B215, blend2-gb2-knn1-3B206, blend2-gb3-knn1-1B205, blend2-gb3-knn1-2B240, blend2-gb3-knn1-3B208, blend2-gb3-knn1-4B195, blend2-gb3-knn1-5B245, blend2-gb4-knn1-1B244, blend2-gb4-knn1-2B230, blend2-gb4-knn1-3B231, blend2-gb4-knn1-5B229, blend5-knn1-3B016, blend5-knn3-1B003, blend5-knn4-1B078, brismf760n-knn1-4B148, brismf760n-knn1-5B134, brismf760n-knn3-1B006, drbm160-640-bk4-knn1-1B055, drbm160-640-bk4-knn1-3B024, drbm160-640-bk4-knn1-4B150, drbm160-640-bk4-knn1-5B136, drbm160-640-bk4-knn4-1B086, drbm160-640-bk4-knn5-8B115, drbm160-640-knn1-1B066, drbm160-640-knn1-3B035, drbm160-640-knn1-4B149, drbm160-640-knn1-5B135, drbm160-640-knn5-8B114, drbm160-640gx-knn1-1B181, drbm160-640gx-knn1-5B174, drbm160-640gx-knn3-1B168, drbm160-640gx-knn5-1B186, frbm200-mf27-flip20-knn1-1B067, frbm200-mf27-flip20-knn1-3B036, frbm300-bk4-knn1-1B045, frbm300-bk4-knn1-3B014, frbm300-bk4-knn1-4B152, frbm300-bk4-knn3-1B001, frbm300-bk4-knn4-1B076, frbm300-bk4-knn5-8B117, frbm300-knn1-3B037, frbm300-knn1-4B151, frbm300-knn1-5B137, frbm300gx-knn1-1B177, frbm300gx-knn1-3B160, frbm300gx-knn1-5B184, frbm300gx-knn3-1B179, frbm300gx-knn5-1B171, gte14b-knn1-1B048, gte14b-knn1-3B017, gte14b-knn3-1B004, gte14b-knn4-1B079, integ0-0-0TZ-grbm200-knn1-1B056, integ0-0-0TZ-grbm200-knn3-1B012, integ0-0-0TZ-grbm200-knn4-1B087, integ0-0-0TZ-knn4-1B077, integ0-0-0TZmilestone6-200-knn1-5B139, integ0-0-0TZmilestone6-200-knn5-8B118, integ0-100-100TZ-knn1-1B069, integ0-100-100TZ-knn1-3B038, integ0-200-200NT-knn1-4B154, integ0-200-200NT-knn1-5B140, integ0-200-200NT-knn5-8B119, integ0-200-200TZ-knn1-1B070, integ40-200-0ST-knn1-3B040, integ80-80-0TZM-knn1-4B155, integ80-80-0TZM-knn1-5B141, integ80-80-0TZM-knn5-8B120, mf27-20-knn1-3B041, mfw31-05-asym3v250-knn1-1B073, mfw31-10-grbm200-knn1-5B142, mfw31-10-grbm200-knn5-8B121, mfw31-10-milestone0-150-knn1-1B053, mfw31-10-milestone0-150-knn3-1B009, mfw31-10-milestone0-150-knn4-1B084, mfw31-10-milestone5-150-knn1-1B074, mfw31-10-milestone5-150-knn1-3B043, mfw31-60-10-120-knn1-1B057, mfw31-60-10-120-knn3-1B013, mfw31-60-10-120-knn4-1B088, ssvd-31-00-asym1-20-knn1-1B075, ssvd-31-00-asym1-20-knn1-3B044, urbm20-1000-knn1-1B051, urbm20-1000-knn4-1B082
PT-282	PT-236, bk1-b200-5-knn1-1B060, bk2-b200h, bk3-200g3-X-bk3-100ga4, bk3-a0z, bk3-d200-knn1-1B049, bk3-d200-knn1-3B018, bk3-d200-knn3-1B005, bk3-d200z-knn1-4B146, bk4-a50, bk4-b200-knn1-1B100, bk4-b200-knn3-1B101, bk4-biasZ, bk4-c200g, bk4-c50, bk4-d50, bk4-d50B128, bk4-e50a, bk4-f200z4-nlpp1-knn3-1, bk4-f200z4, bk5-b200-knn1-1B095, bk5-b200-knn1-5B133, bk5-b200-knn4-1B097, bk5-b200B089-knn1-3B090, bk5-b200B089, blend2-gax-knn1-1B052, blend2-gax-knn1-3B021, blend2-gb, blend5-knn1-1B047, brismf760n-knn1-1B050, brismf760n-knn1-3B019, brismf760n-knn4-1B081, drbm160-640-bk4-knn3-1B011, frbm300-knn1-1B068, integ0-0-0TZ-grbm200-knn1-3B025, integ0-0-0TZ-knn1-1B046, integ0-0-0TZ-knn1-3B015, integ0-0-0TZ-knn3-1B002, integ0-200-200TZ-knn1-3B039, integ40-200-0ST-knn1-1B071, mf27-20-knn1-1B072, mfw31-10-milestone0-150-knn1-3B022, nnmf80-ssvd-39-00-movie4-X-blend2-ga4, ssvd-04-10, ssvd-31-00-user2, ucount-X-drbm100-500, ucount-X-nmf80-120-20-m
PT-New16	bk3-200g3-X-bk3-100ga4, bk3-a0z, bk3-d200-knn1-1B049, bk3-d200-knn1-3B018, bk3-d200-knn3-1B005, bk4-a50, bk4-b200-knn1-1B100, bk4-b200-knn3-1B101, bk5-b200-knn1-1B095, bk5-b200B089, blend2-gax-knn1-3B021, integ0-200-200TZ-knn1-3B039, integ40-200-0ST-knn1-1B071, nnmf80-ssvd-39-00-movie4-X-blend2-ga4, ssvd-04-10, ssvd-31-00-user2
BK-results-6	Probe-Qualifying pairs numbered 1,4,5,2,3,7 at Sec. VII.B of Y. Koren. ”The BellKor Solution to the Netflix Grand Prize 2009”

C The 18 Predictors needed for 10%

In Figure 6 one can see that 18 predictors suffice to reach a 10% improvement on the quiz set. In Table 11 we provide an ordered list of these predictors. Most of them stem from nonlinear probe blends.

We found these predictors with backward selection. We started with all predictors and iteratively removed the predictor with the lowest contribution. Please note that this method is not guaranteed to find the smallest possible subset of predictors which achieve a 10% improvement. So there might be a smaller subset or an 18 predictor subset with a lower RMSE.

#	blend RMSE	Predictor
1	0.85834	PB-101, rmse=0.8584, Ensemble neural network blend, 1149 results from small nets with 4 neurons (29 trainable weights) in the hidden layer, 3-predictor random subsets. All results were combined with a 2-date-bins linear blender. The base set of predictors are {ALL-476, BC-Exact-340}.
2	0.85693	PB-115, rmse=0.8587, Neural network blend on tree blends, extended with SVD and RBM features, where the probe prediction of the trees is the out-of-bag estimate. The net has two hidden layers with 17 and 7 neurons. Training was stopped when the 4-fold CV error has reached the minimum (after 493 epochs). The input is: {PB-143 ... PB-163, SVD++10-cross-145, RBM-50-user}
3	0.85665	PB-107, rmse=0.8600, Ensemble neural network blend on a 168 predictor subset by Pragmatic Theory. 980 results were blended with a 4-frequency-bins linear blender. Each of the results were produced by a 3-predictor random subset blend of the 168. These subsets were blended by a 1HL net with 5 neurons. Input: {PT-168}
4	0.85662	PB-054, rmse=0.8658, Neural network blend on a forward selection by BigChaos' predictors. Here, top-15 are selected with a simple greedy forward selection algorithm. Blending is done with a 1HL net with 13 neurons. Training was stopped after 1334 epochs. Input: {BC-192-Top15}
5	0.85658	rmse=8713, A result by BellKor: last one at Sec. VII of [12]
6	0.85655	PB-112, rmse=0.8621, Kernel Ridge Regression blend on results by all 3 teams. The KRR algorithm is computational very expensive, therefore we draw 4000 random samples from the 1.4M probe set. A Gauss kernel with $\sigma = 100$ is used, the ridge constant is $\lambda = 1e - 6$. 8 results are combined linearly. Input: {ALL-476}
7	0.85651	PB-111, rmse=0.8589, Neural network blend on results by all 3 teams. The net has 1HL with 4 neurons. Input: {ALL-476}
8	0.85648	OB-20, rmse=0.8925, A residual chain of 3 algorithms: First a SVD with non-neg weights on raw ratings (OB-08, rmse=0.9023), second Global Time Effects (OB-09, rmse=0.8932) and third a MovieKNNV3 on residuals of OB-09.
9	0.85645	OB-23, rmse=0.9317, NSVD1 with 40 features on raw ratings.
10	0.85643	rmse=0.9201, A result by Pragmatic Theory: ssvd-31-00-asym4-200 (Asymmetric 4 model on residuals of a Matrix Factorization 2 model)
11	0.85641	rmse=0.9290, A result by BellKor: #14 at [4]
12	0.85638	rmse=0.8943, A result by BellKor: top of page 9 of [5]
13	0.85636	rmse=0.9212, A result by BellKor: #17 of [4]
14	0.85635	PB-063, rmse=0.8902, Neural network blend on SVD and RBM features. We used a 2HL net with 50 and 10 neurons. Input: {RBM-50-user, SVD++20-187}
15	0.85633	PB-106, rmse=0.8601, Ensemble neural network blend on results by all 3 teams. Here we select 8 random predictors 1213 times (net: 1HL with 5 neurons), results were combined by a 25-cluster-bins (5 user and 5 movie clusters) linear blend. Input: {ALL-476}
16	0.85631	rmse=0.8606, A result by BellKor: GBDT blend of top 75 results from Sec. VII of [12]
17	0.85630	PB-117, rmse=0.8588, Neural network blend on tree blends, extended with SVD and RBM features, where the probe prediction of the trees is the out-of-bag estimate. The result is very similar to PB-115, but with a slightly different neuron configuration in the two hidden layers. Input: {PB-143 ... PB-163,SVD++10-cross-145,RBM-50-user}
18	0.85628	rmse=0.8914, A result by BellKor: matrix factorization model with $f = 20$ described at Sec. IV.A of [12]

Table 11: The 18 predictors, which are necessary to reach 10%.