All Together Now: A Perspective on the Netflix Prize

Robert M. Bell, Yehuda Koren & Chris Volinsky

All Together Now:
A Perspective on the
NETFLIX PRIZE

Robert M. Bell, Yehuda Koren, and Chris Volinsky

When the Netflix Prize was announced in October of 2006, we initially approached it as a fun diversion from our 'day jobs' at AT&T. Our group had worked for many years on building profiles of customer patterns for fraud detection, and we were comfortable with large data sets, so this seemed right up our alley. Plus, it was about movies, and who doesn't love movies? We thought it would be a fun project for a few weeks.

Boy, were we wrong (not about the fun part, though). Almost three years later, we were part of a multinational team named as the winner of the $1 million prize for having the greatest improvement in root mean squared error (RMSE) over Netflix's internal algorithm, Cinematch.

The predominant discipline of participants in the Netflix Prize appears to have been computer science, more specifically machine learning. While something of a stereotype, machine learning methods tend to center on algorithms (black boxes), where the focus is on the quality of predictions—rather than 'understanding' what drives particular predictions.

In contrast, statisticians tend to think more in terms of models with parameters that carry inherent interest for explaining the world. Leo Breiman's article, "Statistical Modeling: The Two Cultures," which was published in *Statistical Science*, provides various views on this contrast. Our original team consisted of two statisticians and a computer scientist, and the diversity of expertise and perspective across these two disciplines was an important factor in our success.

Fundamental Analysis Challenge

The Netflix Prize challenge concerns recommender systems for movies. Netflix released a training set consisting of data from almost 500,000 customers and their ratings on 18,000 movies. This amounted to more than 100 million ratings. The task was to use these data to build a model to predict ratings for a hold-out set of 3 million ratings. These models, known as collaborative filtering, use the collective information of the whole group to make individualized predictions.

Movies are complex beasts. Besides the most obvious characterization into genres, movies differ on countless dimensions describing setting, plot, characters, cast, and many more subtle features such as tone or style of the dialogue. The Movie Genome Project (*www.jinni.com/movie-genome.html*) reports using "thousands of possible genes." Consequently, any finite model is likely to miss some of the signal, or explanation, associated with people's ratings of movies.

On the other hand, complex models are prone to overfitting, or matching small details rather than the big picture—especially where data are scarce.

For the Netflix data, the numbers of ratings vary by at least three orders of magnitude among both movies and users. Whereas there are some users who have rated more than 10,000 movies, the average number of ratings per user is 208, and more than one quarter rated fewer than 50 movies. Overfitting is particularly a concern for these infrequent raters.

This leads to the fundamental challenge for the Netflix data. How can one estimate as much signal as possible where there are sufficient data without overfitting where data are scarce? Our winning approach combined tried-and-true models for recommender systems with novel extensions to these models, averaged together in an ensemble.

Nearest Neighbors

At the outset of the Netflix competition, the most commonly used collaborative filtering method was nearest neighbors. Gediminas Adomavicius and Alexander Tuzhilin give an overview of the state of the art in "Towards the Next Generation of Recommender Systems: A Survey of the State of the Art and Possible Extensions," published in *IEEE Transactions on Knowledge and Data Engineering*. With nearest neighbors, the predicted rating for an item by a user might be a weighted average rating of similar items by the same user. Similarity is measured via Pearson correlation, cosine similarity, or other metric calculated on the ratings.

For example, we might expect neighbors of the movie "Saving Private Ryan" to include other war movies, other movies directed by Steven Spielberg, and other movies starring Tom Hanks. A typical nearest neighbors model, as described in "Item-Based Collaborative Filtering Recommendation Algorithms" presented at the 10th International World Wide Web Conference by Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, estimates the rating r_{ui} of item i by user u to be

$$\hat{r}_{ui} = \frac{\sum_{j \in N(i;u)} s_{ij} r_{uj}}{\sum_{j \in N(i;u)} s_{ij}} \quad , \qquad (1)$$

where $N(i; u)$ is the set of neighbors of item i that were rated by user u and s_{ij} is the similarity between items i and j. A nice feature of nearest neighbor models

The Contest That Shaped Careers and Inspired Research Papers

Steve Lohr

Back in October of 2006, when Netflix announced its million-dollar prize contest, the competition seemed to be a neat idea, but not necessarily a big one. The movie rental company declared it would pay $1 million to the contestant who could improve its web site's movie recommendation system by 10% or more. The contest presented an intriguing problem—and a lucrative one for the winner. For Netflix, it was a shrewd ploy that promised to pay off in improved service and publicity.

But the Netflix contest, which lasted nearly three years, turned out to have a significance that extended well beyond movie recommendations and money. The competition became a model of Internet-era collaboration and innovation, attracting entries from thousands of teams around the world. The leading teams added members as they sought help to improve their results and climb up the Netflix leaderboard. Team members were often located in different countries, communicating by email and sharing work on the web with people they never met face to face.

This kind of Internet-enabled cooperative work—known as crowdsourcing—has become a hot topic in industry and academia. The Netflix contest is widely cited as proof of its potential.

The Netflix competition also became a leading exhibit by enthusiasts of "prize economics."

Internet technology makes it possible to tap brainpower worldwide, but those smart people need an incentive to contribute their time and synapses. Hence, the $1 million prize. The prize model is increasingly being tried as a new way to get all kinds of work done, from exploring the frontiers of science to piecework projects for companies. The X Prize Foundation, for example, is offering multimillion-dollar prizes for path-breaking advances in genomics, alternative energy cars, and private space exploration. InnoCentive is a marketplace for business projects, where companies post challenges—often in areas such as product development and applied science—and workers or teams compete for cash payments or prizes. A start-up, Genius Rocket, runs a similar online marketplace mainly for marketing, advertising, and design projects.

The emerging prize economy, according to labor experts, does carry the danger of being a further shift in the balance of power toward the buyers—typically corporations—and away from most workers. At first glance, there did seem to be an element of exploitation in the Netflix contest. Thousands of teams from more than 100 nations competed, and it was a good deal for the company. "You look at the cumulative hours and you're getting PhDs for a dollar an hour," said Reed Hastings, the chief executive of Netflix.

Yet, the PhDs I talked to in covering the Netflix contest for *The New York Times* were not complaining. Mostly, they found the challenge appealing, even though hardly any were movie buffs. "It's incredibly alluring to work on such a large, high-quality data set," explained Joe Sill, an analytics consultant who holds a PhD in machine learning from the California Institute of Technology.

Steve Lohr covers technology, economics, and (occasionally) statistics for *The New York Times*. He has written for *The New York Times Magazine*, *The Atlantic*, and *The Washington Monthly* and is the author of *Go To: The Story of the Math Majors, Bridge Players, Engineers, Chess Wizards, Maverick Scientists, and Iconoclasts—the Programmers Who Created the Software Revolution*.

Some professors used the Netflix challenge to introduce graduate students to collaborative research on a big and interesting problem. At Iowa State University, for example, 15 graduate students, advised by five faculty members, tackled the Netflix challenge for a semester, until final exams pulled them away. It proved a rich, real-world test tube for trying out a range of statistical models and techniques. In four months, the Iowa State team improved on the Netflix internal recommendation system by about 4%. "We weren't doing bad[ly], but it was very time-consuming and eventually the rest of the world passed us by," said Heike Hofmann, an associate professor of statistics.

Indeed, the main lure for the contestants—even more than a chance for a big payday—was to be able to experiment in applying tools of statistics, computing, and machine learning to the big Netflix data set, 100 million movie ratings. They spent their own time, or took work time with their companies' blessing, because they knew the lessons learned would be valuable beyond the Netflix case.

In the online world, automated recommendation systems increasingly help—and shape—the choices people make not only about movies, but also books, clothing, restaurants, news, and other goods and services. And large-scale modeling, based on ever-larger data sets, is being applied across science, commerce, and politics. Computing and the web are creating new realms of data to explore—sensor signals, surveillance tapes, social network chatter, public records, and more.

The skills and insights acquired in the Netflix quest promise to be valuable in many fields. So, what was the biggest lesson learned? The power of collaboration and combining many models, the contestants said, to boost the results. In a nail-biter finish, two teams each had the same score above the 10% threshold, though the winner made its entry 20 minutes earlier. Both teams used the mash-up of models approach, with no single insight, algorithm, or concept responsible for the performance.

"Combining predictive models to improve results," said Trevor Hastie, a professor of statistics at Stanford University, "is known as the 'ensemble' approach." (The runner-up team, in fact, was called The Ensemble.) Model averaging is another statistical term used to describe such approaches.

"It may not be very elegant or satisfying—it would be nice if some produced the natural single method—but that is what worked," said Hastie, who was not a contestant. "And success with ensemble methods was made possible by the huge amount of data."

The winning team and the near-miss loser were alliances. The winner, BellKor's Pragmatic Chaos, started as three AT&T researchers. (One later joined Yahoo Research, but remained on the team.) Two other two-person teams, from Austria and Canada, came on board as the contest progressed. The AT&T scientists participated in the contest with their company's approval because it was seen as a worthwhile research project. That would have been a smart decision, the researchers agree, even if their team had not landed the prize.

"The Netflix contest will be looked at for years by people studying how to do predictive modeling," said Chris Volinsky, director of statistics research at AT&T.

The scientists and engineers on the second-place team—and the employers who gave many of them the freedom to compete in the contest—agreed the time and toil was worth it. Arnab Gupta, chief executive of Opera Solutions, took a small group of his leading researchers off other work for two years. "We've already had a $10 million payoff internally from what we've learned," Gupta said.

The Netflix prize contest shaped careers, inspired research papers, and spawned at least one start-up. Shortly after the $1 million contest concluded, Sill said he and other members of The Ensemble team were talking about commercializing their hard-earned knowledge. 'There's nothing concrete yet, but we're exploring several avenues," he said.

is that the resulting recommendations are easy to explain to users by pointing to the neighboring items that the user rated highly.

Although equation (1) is intuitive, it raises many concerns. First, the choice of similarity metric is arbitrary, without formal justification. Second, relative weights do not depend on the composition of a neighborhood, so highly correlated neighbors of a movie tend to get "double counted." Finally, predictions are particularly unreliable for movies with few or no close neighbors.

In "Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights," presented at the 7th IEEE International Conference on Data Mining by Robert Bell and Yehuda Koren, linear regression is used to estimate customized mixing weights for each prediction to account for correlations among the set of available neighbors. To deal with missing data, these regressions use sufficient statistics from the derived estimates of the covariance matrix for item ratings. Empirical Bayes shrinkage is used to improve reliability of estimated covariances for item pairs with few ratings by a common user.

Matrix Factorization

Many of the most successful single models in the Netflix Prize were latent factor models—most notably ones that use matrix factorization. Matrix factorization characterizes both items and users by d-dimensional vectors of latent factors, where d is much smaller than either the number of movies or users, say $d = 50$. For movies, a factor might reflect the amount of violence, drama vs. comedy, more subtle characteristics such as satire or irony, or possibly some noninterpretable dimension. For each user, there is a vector in the same dimensional space, with weights that reflect the user's taste for items that score high on the corresponding factor.

Figure 1 shows output from a typical model, where a selected set of movies is shown with their loadings on the first two latent factors. An inspection of the first factor shows a separation of serious movies from silly comedies. On the right side, we see movies with strong female dramatic leads, contrasted on the left with movies aimed at the fraternity set. The second dimension separates critically acclaimed, independent,
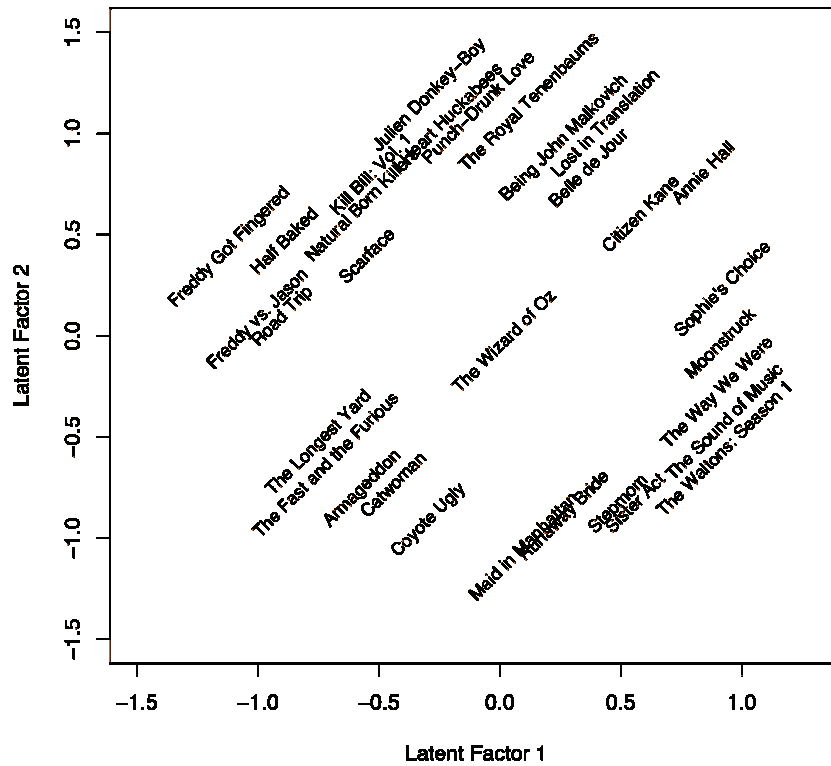
Figure 1. Selected movies and their weights across the first two latent factors for a typical matrix factorization model

quirky movies at the top from large-budget Hollywood star–driven movies on the bottom. Latent factor models are attractive because they can extract these orthogonal 'concepts' out of the data, without requiring external information about genre, actors, or budget.

In the basic matrix factorization model, a user's interest in a particular item is modeled using the inner product of the user (p_u) and item (q_i) factor vectors, plus bias terms (b_u for users and a_i for items) for both. Specifically,

$$\hat{r}_{ui} = \mu + a_i + b_u + q_i^T p_u \ . \quad (2)$$

Ideally, the number of dimensions d could be set large, perhaps in the thousands, to capture as many of the countless subtle movie characteristics that affect users' ratings as possible. For this simple model, however, predictive performance on a hold-out set begins to degrade for $d > 5$. One solution is

to minimize a penalized least squares function, which is one way to address overfitting and sparse data.

Technically, the sum of squared errors (the first term in (3) below) is augmented by additional terms (four in this example):

$$\sum_{(u,i) \in training} (r_{ui} - \hat{r}_{ui})^2 + \lambda_1 \sum_i a_i^2$$
$$+ \lambda_2 \sum_u b_u^2 + \lambda_3 \sum_i || q_i ||_2^2 \quad (3)$$
$$+ \lambda_4 \sum_u || p_u ||_2^2 \ ,$$

where the lambdas, which control the amount of regularization, are chosen to optimize the MSE on a hold-out set. Penalized least squares can be motivated by assuming each of the parameters, other than μ, is drawn from a normal distribution centered at zero. One option for solving equation (3) is to fit an iterative series of ridge regressions—one per user and one per item in each iteration.

Alternatively, as presented in "Bayesian Probabilistic Matrix Factorization Using Markov Chain Monte Carlo," presented at the 25th International Conference on Machine Learning by Ruslan Salakhutdinov and Andriy Mnih, a full Bayesian analysis is feasible for d of at least 300.

In practice, we found that the most effective solution to equation (3) was usually stochastic gradient descent, which loops through the observations multiple times, adjusting the parameters in the direction of the error gradient at each step. This approach conveniently avoids the need for repeated inversion of large matrices.

The power and flexibility of the matrix factorization model combined with efficient computational techniques allowed us to develop several important extensions to the main model. One such extension uses the set of items rated by a user to refine the estimated taste factors derived from the ratings themselves. The idea is that users select which items to rate, so that missing ratings
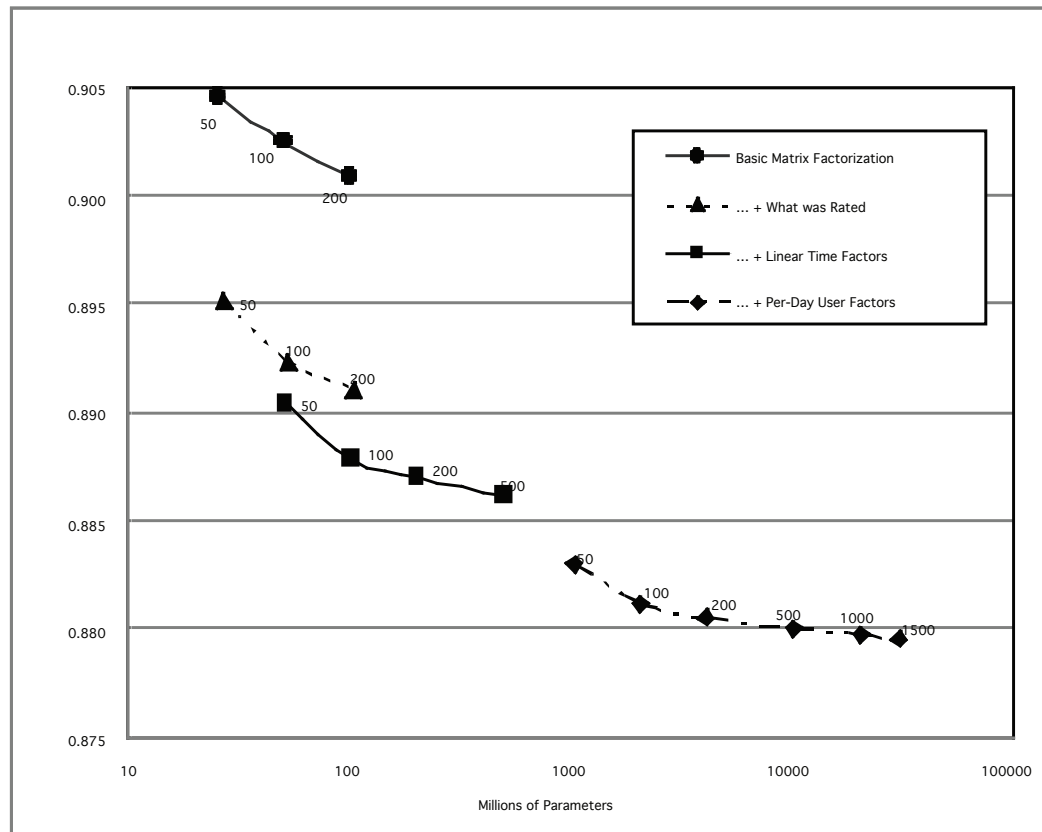
Figure 2. RMSEs for selected matrix factorization models and extensions versus numbers of parameters

are certainly not—in the terminology of Roderick Little and Donald Rubin in *Statistical Analysis with Missing Data*—missing at random. Consequently, the set of movies a user chooses to rate is an additional source of information about the user's likes.

Another extension allows many of the parameters to vary over time. For example, a movie's popularity might drift over time, as might a user's standard for what constitutes four stars. Of more consequence, a user's tastes, as measured by p_u may change over time. Consequently, as described by Koren in "Collaborative Filtering with Temporal Dynamics" (see *http://research.yahoo.com/files/kdd-fp074-koren.pdf*), we allow the parameters $\{a_i\}$, $\{b_u\}$, and $\{p_u\}$ in equation (2) to change gradually or, perhaps, abruptly to allow for the possibility that different members of a family provide ratings on different dates. Additional regularization terms are required to shrink the new parameters.

Figure 2 displays the RMSE on test data for a variety of matrix factorization models with different numbers of parameters. The top curve shows performance for basic matrix factorization with d = 50, 100, and 200, resulting in approximately 25, 50, and 100 million parameters, respectively. The RMSE for Cinematch was 0.9514, so the 100-factor model achieves slightly better than a 5% improvement. For reference, a model that always predicts the movie average yields an RMSE of about 1.05.

The remaining curves illustrate the improvement in RMSE attributable to the extensions outlined above. The second curve (from the top) shows that accounting for what a user rated produces about a 1% improvement for each value of d. The next curve illustrates the additional impact of allowing each $\{a_i\}$, $\{b_u\}$, and $\{p_u\}$ to change linearly over time. The final curve shows results for a model that allows each user bias $\{b_u\}$ and taste vector $\{p_u\}$ to vary arbitrarily around the

linear trend for each unique rating date. We see RMSEs continue to decline even as d grows to 1,500, resulting in 300 parameters per observation. Obviously, regularization was essential.

The More the Merrier

An early lesson of the competition was the value of combining sets of predictions from multiple models or algorithms. If two prediction sets achieved similar RMSEs, it was quicker and more effective to simply average the two sets than to try to develop a new model that incorporated the best of each method. Even if the RMSE for one set was much worse than the other, there was almost certainly a linear combination that improved on the better set. And, if two is better than one, then k + 1 must be better than k. Indeed, a hopelessly inferior method often improved a blend if it was not highly correlated with the other components.

At the end of the first year of the competition, our submission was a linear combination of 107 prediction sets, with weights determined by ridge regression. These prediction sets included results of various forms of both nearest neighbors and latent factor models, as well as other methods such as nearest neighbors fit to residuals from matrix factorization. This blend achieved an 8.43% improvement relative to Cinematch on the quiz data. Notably, our best pure single prediction set at the time (RMSE = 0.8888) achieved only a 6.58% improvement.

The value of ensembles for prediction or classification is not a new insight. Robert Schapire and Yoav Fruend's boosting and Breiman's random forests were derived from the idea of forming ensembles of purposely imperfect classifiers or regressions. Our blend differs fundamentally in that the individual components need not share anything. Others also have found ensembles of heterogeneous methods to be effective in various problems.

Just as we found value in combining many diverse models and algorithms, we benefited from merging with other competitors who brought their own perspectives to the problem. In 2008, we merged with the BigChaos team, Michael Jahrer and Andreas Toscher, two computer science students from Austria. Their approach to blending models used nonlinear blends via neural networks, which we had not tried. In 2009, we added team Pragmatic Theory, consisting of Martin Chabbert and Martin Piotte, two engineers from Montréal. Among many contributions, they brought new models for incorporating rating frequency to model temporal behavior.

After nearly 33 months, our combined team became the first to achieve a 10% improvement over Cinematch, triggering a 30-day period in which all competitors were allowed to produce their best and final submissions.

Of course, what had worked so well for us was open to all. On the second-to-last day, a new team aptly named The Ensemble first appeared on the leaderboard. This mega team, which included members from 23 original teams, leapfrogged slightly ahead of us in the public standings, but the winner was left unclear. The actual winner was to be determined by a held-out test set about which Netflix had not provided any feedback. At the time, nobody knew who actually won the Netflix Prize.

As it turned out, the top two teams' RMSEs on the test set differed by an unbelievably small margin—0.856704 for us, versus 0.856714 for The Ensemble—a difference of 0.00010. The conditions of the contest called for rounding all RMSEs to four decimal places, so the teams were tied. The tiebreaker was submission time of the best entries. We submitted our predictions 20 minutes earlier than The Ensemble. After three years, our margin of victory was 20 minutes. It was like winning a marathon by just a few inches.

Success

The Netflix Prize took three years to win and generated a lot of interest in and attention for the field of collaborative filtering and recommendations in general. It is widely considered a success from different perspectives:

Researchers already working in collaborative filtering were energized by the scale and scope of the Netflix data set, which was orders of magnitude bigger than previous recommendation data sets. The competition also attracted many new researchers into the field, as many of the top teams (including ours) had little or no experience in collaborative filtering when they started.

The Netflix Prize showed the success of bringing together the work of many independent people to develop new technology or business strategy, sometimes called crowdsourcing and popularized in the recent book *The Wisdom of Crowds*, by James Surowiecki. The idea behind crowdsourcing is that an ensemble of opinions, all generated independently and with access to different resources, will perform better than a single expert.

The Netflix Prize drove home the power of ensemble methods. Every leading team created a bucket of heterogeneous models and algorithms. Methods for optimal averaging of an ensemble based on limited information (e.g., we only had RMSE for each model) had a large impact on overall performance.

For the field of statistics, there is a clear lesson. Our team benefited from having teammates from many academic backgrounds, including computer science, machine learning, and engineering. These fields brought different perspectives on problemsolving and different toolboxes to large data sets. Stepping outside of our domain was helpful in making difficult decisions that turned out to be important. As collaboration has always been at the heart of the statistics profession, this lesson should surprise no one. **C**

Further Reading

Adomavicius, G., and A. Tuzhilin. 2005. Towards the next generation of recommender systems: A survey of the state of the art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering* 17:734–749.

Bell, R. M., Y. Koren, and C. Volinsky. 2007. The BellKor solution to the Netflix Prize. *www2.research. att.com/~volinsky/netflix/ProgressPrize-2007BellKorSolution.pdf*.

Bell, R. M., and Y. Koren. 2007. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. Presented at the Seventh IEEE International Conference on Data Mining, Omaha.

Breiman, L. 2001. Statistical modeling: The two cultures (with discussion). *Statistical Science* 16:199–231.

Breiman, L. 2001. Random forests. *Machine Learning* 45(1):5–32.

Koren, Y. 2009. Collaborative filtering with temporal dynamics. *http://research.yahoo.com/files/kdd-fp074-koren.pdf*.

Koren, Y., R. Bell, and C. Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42(8):30–38.

Little, R. J. A., and D. B. Rubin. 2002. *Statistical analysis with missing data* (2nd ed.). Hoboken: John Wiley & Sons.

Salakhutdinov, R., and A. Mnih. 2008. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo." Presented at the 25th International Conference on Machine Learning, Helsinki.

Sarwar, B., G. Karypis, J. Konstan, and J. Riedl. 2001. Item-based collaborative filtering recommendation algorithms. Presented at the 10th International World Wide Web Conference, Hong Kong.

Schapire, R. 1990. Strength of weak learnability. *Journal of Machine Learning* 5:197–227.